Machine learning methods for discrete multi-scale fows : application to finance

Nicolas Mahler

To cite this version:

Nicolas Mahler. Machine learning methods for discrete multi-scale fows : application to finance. General Mathematics [math.GM]. École normale supérieure de Cachan - ENS Cachan, 2012. English. NNT : 2012DENS0022 . tel-00749717

HAL Id: tel-00749717

https://theses.hal.science/tel-00749717

Submitted on 8 Nov 2012

ENS CACHAN

EDSP École Doctorale Sciences Pratiques

THÈSE DE DOCTORAT

DE L'ÉCOLE NORMALE SUPÉRIEURE DE CACHAN

Présentée par

Nicolas MAHLER

pour obtenir le grade de

DOCTEUR DE L'ÉCOLE NORMALE SUPÉRIEURE DE CACHAN

Domaine :

MATHEMATIQUES APPLIQUEES

Machine learning methods for discrete multi-scale fows : application to finance

Thèse présentée et soutenue à Cachan le 05/06/2012 devant le jury composé de :

- M. Rosenbaum Mathieu (rapporteur), professeur - CREST, LPMA
- M. Ralaivola Liva (rapporteur), maître de conférences - LIF de Marseille
- M. Hoffmann Marc, professeur - CREST, LAMA
- M. Lehalle Charles-Albert, docteur - équipe de recherche quantitative de Crédit Agricole Cheuvreux
- M. Vayatis Nicolas (directeur de thèse), professeur
- M. Clemençon Stéphan, professeur - Télécom-ParisTech

Centre de Mathématiques et de Leurs Applications
61, avenue du Président Wilson, 94235 Cachan Cedex

Machine Learning Methods for Discrete Multi-scale Flows: Application to Finance

by

Nicolas Mahler

B.S. (Ecole des Mines de Paris) 2004
M.S. (Université de Paris VI) 2005

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Applied Mathematics

in the

GRADUATE DIVISION

of the

ECOLE NORMALE SUPERIEURE DE CACHAN

February 2012

STRATEGIC RISK MANAGEMENT
TURNING RISK INTO PROFIT

CACHAN

ABSTRACT

Machine Learning Methods for Discrete Multi-scale Flows:
Application to Finance

by

Nicolas Mahler

Doctor of Philosophy in Applied Mathematics

Ecole Normale Supérieure de Cachan

This research work studies the problem of identifying and predicting the trends of a single financial target variable in a multivariate setting. The machine learning point of view on this problem is presented in chapter I. The efficient market hypothesis, which stands in contradiction with the objective of trend prediction, is first recalled. The different schools of thought in market analysis, which disagree to some extent with the efficient market hypothesis, are reviewed as well. The tenets of the fundamental analysis, the technical analysis and the quantitative analysis are made explicit. We particularly focus on the use of machine learning techniques for computing predictions on time-series. The challenges of dealing with dependent and/or non-stationary features while avoiding the usual traps of overfitting and data snooping are emphasized. Extensions of the classical statistical learning framework, particularly transfer learning, are presented. The main contribution of this chapter is the introduction of a research methodology for developing trend predictive numerical models. It is based on an experimentation protocol, which is made of four interdependent modules.

The first module, entitled *Data Observation and Modeling Choices*, is a preliminary module devoted to the statement of very general modeling choices, hypotheses and objectives. The target variable and the corresponding explanatory variables are chosen by the user. Hypotheses regarding their relative distribution and structure of dependence are formulated. The prediction setup - either the regression setup or the binary classification setup - is specified. The time ranges of the training set and of the test set are finally defined.

The second module, *Database Construction*, turns the target and explanatory variables into features and labels in order to train trend predictive numerical models. This module corresponds to Chapter II, which aims at finding a time- series representation, capable of computing homogeneous, interpretable, robust and parsimonious features. In that perspective, chapter II first introduces significant economic and financial variables, which play the role of target series and of explanatory series in chapter IV. Their intra and inter heterogeneities are highlighted. Criteria specific to time-

series representation methods are then drawn up to ensure the transformation of heterogeneous sequences into databases relevant for learning. Representation methods, which meet these criteria satisfactorily, are reported. State-space models - and the Kalman filter - are notably made explicit. The main contribution of this chapter is the presentation of the UniCart method, which is a new multiscale and piecewise approximation method. Inspired by the CART algorithm, it computes a top- down binary tree structured segmentation for any given finite real series. The UniCart procedure has led to the design of two types of databases, whose features are close to the representations used in technical analysis. Both types of databases are extensively illustrated and their stationarities are particularly studied.

The purpose of the third module, entitled *Model Construction*, is the construction of trend predictive numerical models. Chapter III presents thus three different model construction methods, which have been consecutively developed and backtested. The first method, called Kalman LagLasso (KLL), performs linear model selection in the regression setup. It is based on the formulation of strong hypotheses about data, which enable the statement of a linear relation between the (Kalman) filtered target variable and the (Kalman) filtered and lagged explanatory variables. Model selection is achieved by a new procedure called LagLasso, actually a Lasso-type procedure which includes selection of lags for individual factors. The second method aims at learning the distribution of UniCart features and labels in the binary classification set up. We particularly focus on tree aggregation based procedures, such as Random Forests (RF). The third method, whose presentation is the main contribution of this chapter, first states a statistical model that we call model for clustering and classification. In this mixture model, the different component distributions - or regimes - are unknown and non- parametric and have to be identified and learnt. This is achieved by a new transfer learning procedure, called Relabeled Nearest Neighbors (RNN), which learns a pseudometric based on the relative importance of features in the prediction. The RNN procedure is entirely parameter-free.

The fourth and last module, entitled *Backtesting and Numerical Results*, evaluates the accuracy of the trend predictive numerical models over a "significant" test set via two generic backtesting plans. The first plan computes recognition rates of upward and downward trends. The second plan designs trading rules using predictions made over the test set. Each trading rule yields a profit and loss account (P&L), which is the cumulated earned money over time. These backtesting plans are additionally completed by interpretation functionalities, which help to analyze the decision mechanism of the numerical models. These functionalities can be measures of feature prediction ability and measures of model and prediction reliability. They decisively contribute to formulating better data hypotheses and enhancing the time- series representation, database and model construction procedures. This is made explicit in chapter IV. Numerical models, aiming at predicting the trends of the target variables introduced in chapter II, are indeed computed for the model construction methods described in chapter III and thoroughly backtested. The switch from one model construction approach to another is particularly motivated. The dramatic influence of the choice of parameters - at each step of the experimentation protocol - on the formulation of conclusion statements is also highlighted. The RNN procedure, which does not require any parameter tuning, has thus been used to reliably study the efficient market hypothesis. New research directions for designing trend predictive models are finally discussed.

RESUME

Méthodes d'Apprentissage pour des Flots Discrets Multi-Echelles:
Application à la Finance
par
Nicolas Mahler
Docteur en Mathématiques Appliqués
Ecole Normale Supérieure de Cachan

Ce travail de recherche traite du problème d'identification et de prédiction des tendances d'une série financière considérée dans un cadre multivarié. Le cadre d'étude de ce problème, inspiré de l'apprentissage automatique, est défini dans le chapitre I. L'hypothèse des marchés efficients, qui entre en contradiction avec l'objectif de prédiction des tendances, y est d'abord rappelée, tandis que les différentes écoles de pensée de l'analyse de marché, qui s'opposent dans une certaine mesure à l'hypothèse des marchés efficients, y sont également exposées. Nous explicitons les techniques de l'analyse fondamentale, de l'analyse technique et de l'analyse quantitative, et nous nous intéressons particulièrement aux techniques de l'apprentissage statistique permettant le calcul de prédictions sur séries temporelles. Les difficultés liées au traitement de facteurs temporellement dépendants et/ou non-stationnaires sont soulignées, ainsi que les pièges habituels du surapprentrissage et de la manipulation imprudente des données. Les extensions du cadre classique de l'apprentissage statistique, particulièrement l'apprentissage par transfert, sont présentées. La contribution principale de ce chapitre est l'introduction d'une méthodologie de recherche permettant le développement de modèles numériques de prédiction de tendances. Cette méthodologie est fondée sur un protocole d'expérimentation, constitué de quatre modules.

Le premier module, intitulé *Observation des Données et Choix de Modélisation*, est un module préliminaire dévoué à l'expression de choix de modélisation, d'hypothèses et d'objectifs très généraux. La variable cible, ainsi que les variables explicatives associées, sont choisies par l'utilisateur. Des hypothèses sont formulées quant à leurs lois de distribution et structure de dépendance relatives. Le cadre de la prédiction - régression ou classification binaire - est précisé. Les historiques de l'ensemble d'apprentissage et de l'ensemble de test sont finalement définis.

Le second module, *Construction de Bases de Données*, transforme la variable cible et les variables explicatives en facteurs et en labels afin d'entraîner les modèles numériques de prédiction de tendances. Ce module correspond au chapitre II, dont le but est de trouver un mode de représentation des séries temporelles capable de produire des facteurs homogènes, interprétables, robustes

et parcimonieux. Dans cette perspective, ce chapitre introduit d'abord des variables économiques et financières de premier plan, amenées à jouer le rôle de variables cibles et explicatives dans le chapitre IV. Leurs intra- et inter-hétérogénéités sont soulignées. Des critères propres aux méthodes de représentation sont alors établis pour garantir la transformation de séquences hétérogènes en bases de données pertinentes pour l'apprentissage. Les méthodes de représentation remplissant ces critères de façon satisfaisante sont mentionnées. Les modèles à espace d'états sont par exemple explicités. La contribution principale de ce chapitre est la présentation d'une nouvelle méthode multi-échelle d'approximation par morceaux, appelée UniCart. Inspirée par l'algorithme CART, celle-ci calcule pour toute série réelle finie une segmentation récursive indexée par un arbre binaire. La procédure UniCart a permis la construction de deux types de bases de données, dont les facteurs sont proches des représentations utilisées en analyse technique. Ces deux types de bases sont illustrés et leurs stationnarités relatives particulièrement étudiées.

Le troisième module, intitulé *Construction de Modèles*, a pour but la construction de modèles numériques de prédiction de tendances. Le chapitre III, associé au module, présente ainsi trois méthodes différentes de construction de modèles, consécutivement développées et testées. La première d'entre elles, appelée Kalman LagLasso (KLL), est une méthode de sélection de modèles linéaires dans le cadre de la régression. Elle est fondée sur la formulation d'hypothèses fortes sur les données, permettant de proposer une relation linéaire entre la variable cible (Kalman) filtrée et les variables explicatives, également (Kalman) filtrées, et décalées dans le temps suivant une plage finie de retards. La sélection de modèles est effectuée par une nouvelle procédure appelée LagLasso, laquelle est une procédure de type Lasso qui sélectionne à la fois variables explicatives et retards appropriés. La seconde méthode vise l'apprentissage de la loi de distribution des facteurs et labels UniCart dans le cadre de la classification binaire. Nous nous intéressons pour ce faire particulièrement aux procédures d'agrégation d'arbres de classification, telles que les Forêts Aléatoires (FA). La troisième méthode, dont la présentation est la contribution principale de ce chapitre, s'appuie sur un modèle statistique que nous appelons modèle de clustering et de classification. Dans ce modèle de mélanges, les lois composantes - ou régimes - sont inconnues, non-paramétriques et doivent être identifiées et apprises. Ces tâches sont remplies par une nouvelle procédure, appelée Plus Proches Voisins Relabelisés (PPVR), qui apprend une pseudo-métrique fondée sur l'importance relative des facteurs dans la prédiction. La procédure PPVR ne nécessite l'entrée d'aucun paramètre.

Le quatrième et dernier module, intitulé *Backtesting et Résultats Numériques*, évalue la précision des modèles de prédiction de tendances sur un ensemble de test significatif, à l'aide de deux procédures génériques de backtesting. Le première procédure renvoie les taux de reconnaissance des tendances de hausse et de baisse. La seconde construit des règles de trading au moyen des predictions calculées sur l'ensemble de test. Le résultat (P&L) de chacune des règles de trading correspond aux gains et aux pertes accumulés au cours de la période de test. De plus, ces procédures de backtesting sont complétées par des fonctions d'interprétation, qui facilite l'analyse du mécanisme décisionnel des modèles numériques. Ces fonctions peuvent être des mesures de la capacité de prédiction des facteurs, ou bien des mesures de fiabilité des modèles comme des prédictions délivrées. Elles contribuent de façon décisive à la formulation d'hypothèses mieux adaptées aux données, ainsi qu'à l'amélioration des méthodes de représentation et de construction de bases de données et de modèles. Ceci est explicité dans le chapitre IV. Les modèles numériques, propres à chacune des méthodes de construction de modèles décrites au chapitre IV, et visant à prédire les tendances des variables cibles introduites au chapitre II, sont en effet calculés et backtestés. Les raisons du passage d'une méthode de construction de modèles à une autre sont particulièrement étayées. L'influence du choix des paramètres - et ceci à chacune des étapes du protocole d'expérimentation - sur la formulation de conclusions est elle aussi mise en lumière. La procédure PPVR, qui ne requiert aucun calcul annexe de paramètre, a ainsi été utilisée pour étudier de façon fiable l'hypothèse des marchés efficients. De nouvelles directions de recherche pour la construction de modèles prédictifs sont finalement proposées.

FOREWORD

This research work has been conducted within the frame of the French Industrial Agreements for Training Through Research (CIFRE) scheme. It has been a project shared by the Strategic Risk Management company, the CMLA research unit of the Ecole Normale Supérieure de Cachan and the TSI research unit of Télécom ParisTech.

Strategic Risk Management is a leading consulting group, based in Paris, which helps clients identify their financial objectives, concerns and constraints. Its team of economists and statisticians develops strategies for:

- understanding long-term economic and financial trends as well as short-term market inefficiencies.
- monitoring the changing risks and returns related to these shifts.
- reviewing client portfolios regularly to determine whether adjustments are needed.

The Center for Mathematical Studies and their Applications (CMLA) of the Ecole Normale Supérieure de Cachan is a public laboratory. It particularly hosts the Machine Learning Group @CMLA, which involves researchers and engineers working on theoretical and industrial projects in the field of statistical modeling and massive data analysis. Ongoing projects are:

- recommender systems.
- machine learning and finance.
- machine learning for large graphs.
- image classification.
- uncertainty analysis and control of experiments for physical and industrial systems.

The Signal and Image Processing (TSI) department of Télécom ParisTech is devoted to the missions of teaching and research in the domains of signal processing, image processing and machine learning. The main research topics are:

- the development of algorithms and statistical processing techniques, in particular for model learning.
- multimedia indexing, sensor networks, and biometrics.
- coding and transmission for multimedia communications.

Contents

Introduction

A jug fills drop by drop

Buddha

This research work focuses on the application of machine learning methods to the problem of computing predictions on multivariate time-series as input variables. More specifically, considering a single target financial variable, we aim at identifying its current trend and predicting the next one at a fixed horizon using a pool of explanatory economic and financial variables.

Context

This problem is so challenging that merely supposing a solution actually exists is a subject of intense debate among academics and financial professionals. According to the efficient market hypothesis, in an efficient market - that is a market where there are rational and well-informed investors competing for a maximal profit - assets are correctly priced. In other words, because markets incorporate information well, prices are at any given time close to the true values of assets, thus making price changes difficult, if not impossible, to anticipate [Bac00, Sam65, Fam70].

On the other hand, the schools of thought in market analysis - mainly fundamental analysis, technical analysis and quantitative analysis each propose their own methodology for implementing successful investment strategies. Fundamental analysis is based on the careful review of companies' balance sheets and on the study of their economic and financial environments. Fundamental analysts have developed a theory of valuation, based on accounting principles. According to them, the value of any investment is determined by the (expected) cashflows it will generate. They assess the value of any company equity and compare it with its stock price [Lew03, BMA10]. The idea is to favor undervalued stocks before they gain popularity in the market. Fundamentalists and economists have also introduced specific notions of returns and of risks to determine optimal investments in a portfolio of assets in a trade-off between risk and return. This risk and return analysis is called the modern portfolio theory. The intuitive underlying key idea is the risk reduction of the portfolio through diversification. Quantitatively, the portfolio theory investigates the problem of maximizing the portfolio return for a given risk level. The constraints of this problem are defined by the nature of the portfolio, whether it does include at least one risk-free asset - of return variance of zero - or not. Solving this problem leads to the construction of efficient portfolios yielding the best return for a given risk level [Mar52, Mar59, Mer92]. It has also initiated the development of the Capital Asset Pricing Model (CAPM) [Sha70], which states a linear relation between the expected returns of assets and their covariances, these latter being interpreted as new measures of risk. This model also makes specific assumptions regarding the rational behavior of investors, supposed to select efficient portfolios and to agree on the expected returns and covariances of the assets, the available market information and the distribution of returns. The Arbitrage Pricing Theory (APT) proposes another point of view and a more general class of models [Ros76a, Ros76b]. It assumes first that asset returns are linearly driven by a set of macro-economic or financial factors. Then it shows that, in the absence of arbitrage (and under complementary assumptions), the expected asset returns are linearly related to their covariances with the factors, which are again seen as measures of risk. For fundamental analysts, these models are a support for taking relevant investment decisions. They have to be completed by more bottom-up analyses, for example by the study of the relation between the financing decisions taken by a given company, its capital structure and the stock market reaction. Only then can fundamental analysts propose global risk management strategies.

Technical analysis aims at identifying or predicting the trend of financial instruments relying on historical data only to the exclusion of underlying economic theory. Rather, it focuses on the behavior of investors. Actually, technicians assume that the psychology of investors, their rationality and their irrationality have a direct impact in terms of supply and demand conditions. According to technical analysts, the behavior of investors shapes patterns, i.e. geometric shapes produced by prices and volumes, that are specific to buying and selling scenarios, and that can be used for trend prediction [Gay90, Bol67, PF01, Mur99, Ach00, Pri02, EMB07, BB02]. Note that such patterns are usually computed using solely the historical data of the variable to predict. Besides, the discovery of patterns requires considerable expertise and a broad experience of market price formation. Usual criticisms against technical analysis focus precisely on the alleged subjectivity of pattern discovery, as highlighted in [LMW00, Aro06, Sew08].

Now, quantitative analysis in finance first focuses on the discovery of stylized facts characterizing financial series [Sew08]. Stylized facts or empirical properties are actually empirical facts supported by consistent observations made through measurements of a statistical quantity. For example, the absence of autocorrelation of financial returns, or their non-stationarities such as volatility clustering effects, are stylized facts. Observing them is the common first step of the various quantitative analysis approaches - as quantitative analysis hardly provides an unified theory.

Indeed, the traditional approach of quantitative analysis, referred to as mathematical finance or financial mathematics or computational finance, often distinguishes between the works of econometricians devoted to risk and portfolio management and of mathematicians interested in pricing and hedging derivative securities [Meu09, Meu11]. On the one hand, econometricians use discrete-time processes for modeling financial series. Autocorrelated and stationary time series are studied via AutoRegressive Moving Average (ARMA) processes. Long memory phenomenons - characterized by a decay of the autocorrelation at a polynomial rate - are tackled by fractionally differenced processes [Hos81]. Besides, non-stationarities are studied by different classes of processes. AutoRegressive Integrated Moving Average (ARIMA) processes, for example the random walk, deal with unit-root non-stationarity [Phi85, CW88]. Finally, AutoRegressive Conditional Heteroskedasticity (ARCH) processes [Eng82] and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) processes [Bol86] reproduce clustered volatility effects. On the other hand, derivatives pricing is based on continuous-time processes. The counterpart of ARMA processes are Ornstein-Uhlenbeck and Continuous AutoRegressive Moving Average (CARMA) processes [Meu09, Meu10, Meu11]. Famous interest rate models, such as the Vasicek model [Vas77] and the model by Cox, Ingersoll, and Ross (CIR) model [CIR85], derive from Ornstein-Uhlenbeck processes. Long memory is addressed by fractional Brownian motion processes. Besides, Levy processes are actually the continuous-time version of random walks. They include the brownian motion, whose geometric form is used to model stock prices in Black-Scholes model [BS73, Mer73], and Poisson jump processes. As highlighted in [CT08], Levy processes play an important role in modelling non-stationarities. Finally, stochastic volatility models, where volatility follows a stochastic process [CR76, Hes93, Che96, HKLW02], propose, along with subordination models, processes capable of modelling volatility clustering effects fairly well.

Recent trends in quantitative analysis, particularly at instigation of econophysics - or statistical finance [MS99, Bou02, BP04], recommend to place special emphasis on empirical data to the detriment of classical economics axioms. For example, it is highlighted in [Bou08] that the Black-Scholes model is still very popular among financial practitioners, although the probability of extreme events it implies concerning price changes - supposedly Gaussian - is by nature underestimated. According to [Bou02, Bou08], the massive amounts of data recorded in the financial markets should be used to properly question models and to validate, modify or discard them. In that perspective, applying statistical learning methods [HTF01, LGWcs, BBL04, BBL05, Tsy10] to the problems of designing investment strategies is relevant, more and more popular and actively studied, for example in

[LMW00, YSC08, CB08, Cha09, FHST10]. It is motivated indeed first by the ability of these modern methods to deal with massive and high-dimensional data sets [BvdG11]. Second, such methods do not require any *a priori* assumptions about the distribution of the data. Instead, they are designed to solve model selection problems via the optimization of an empirical performance criterion, called the empirical risk. For example, in supervised learning setups, such as the classification or the regression setup, data consist of labelled objects, each object being made of a set of features. The empirical risk stands then naturally for an empirical measure of the prediction error. The minimization of the empirical risk, performed by learning algorithms, returns as output a decision function - a classifier or a regressor - capable of using the features to make "good" predictions of the labels. Theoretical guarantees regarding the generalization of the learnt decision function - the fact that it will perform well when confronted to unseen observations - are clearly established. Relying upon the link between the minimization of the empirical risk and of the theoretical risk, they ensure the success of statistical learning techniques. This thesis aims precisely at applying these techniques to the problem of predicting the trends of a single financial series of interest in a multivariate context.

Challenges

Yet, however appealing the use of learning techniques may be, this problem of trend prediction is an ill-posed problem. It simultaneously and implicitly addresses interconnected - and ambitious - challenges. The first one is to define clearly and precisely the concept of trend, actually a quite loose concept. Generally speaking, the trend of a time series refers to a - deterministic or stochastic - smooth function approximating locally, at a given time and scale, the time series. This consensual definition can lead to very different modelling attempts:

- a first approach, inspired by econometrics [Tsa05], consists in directly specifying a rigid - deterministic or stochastic - structure for the trend, removing it and modelling the residuals.
- in a second approach, common in data mining or signal processing, time series can be turned into new features through a specific representation method, for example based on piecewise constant or linear approximation [YF00, KCHP01], symbolic strings [LKLC03], harmonic analysis (such as Fourier and wavelet-based methods) [Mal99], singular value decomposition and singular spectrum analysis [GNZ01], etc. These methods have all scored a large number of impressive achievements in filtering applications. They could therefore potentially lead first to a natural, intuitive, interpretable and economically meaningful definition of trend, and second to the design of a trend and features extraction method.

Thus, both approaches are able to produce databases made of features and labels. Still, another key aspect concerns the statistical characteristics of the provided features: are they sufficiently stationary and concise to be successfully processed by statistical learning methods?

Indeed, the "classical" framework of statistical learning theory relies on the assumption that data samples are independent and identically distributed (iid). Recent theoretical and applied contributions extend this framework to the case of non-iid sequences. They focus on stationary time-dependent series, usually assumed to have ergodic [AN10] or strong mixing properties [Dou94, Yu94, Mei00, Vid03, MR08, RSS10] (such properties indicating roughly a dependence weakening over time) or to be autoregressive time series [MSS11b, MSS11c]. It is explicitly shown that, under strong mixing assumptions, the use of well-known learning algorithms is theoretically supported [SA09, SHS09, SC09, KV09, MR10]. Besides, learning with non-stationary sequences is also a topic under intensive investigation. Actually, the problem of handling non-stationarities in time series, that are changes in the joint probability distribution of the (streaming) data samples, can be addressed in many ways. The specific field of prediction of individual sequences [CBL06, Sto05] - also referred to as online learning [RST10b, RST10a] or as universal prediction - proposes to view the data samples or sequence as the outcomes of an unknown mechanism instead of a stochastic process. A typical model combines the predictions made by a class of reference forecasters or experts and produces a prediction for each point of the sequence. The performance of each expert is measured by a cumulated loss computed - via a loss function - on the past predictions. The goal of the model is to iteratively minimize the regret, which is the difference between the predictions of the model and the predictions made by the best expert, retrospectively. Strategies based on the exponentially-weighted average of the experts' predictions [Vov90] are known to deal particularly well with non-stationary time series [SJC11, HS09] (and also with stationary ergodic times series [OW10]). Note finally that the ideas of prediction of individual sequences have been widely applied to the problem of designing investment strategies [Cov91, HSSW98, BEYG00, SL05, Gyö06b, AHKS06]. Dynamic Bayesian Networks (DBNs) [Gha97, FMR98, Mur02] have also recently proposed a radically different approach for handling non-stationary time series. These directed graphical models enable the modelling of conditional dependences between a given set of variables within and across time slices. The statement of the dependences between variables is very flexible and not restricted to a specific structure. Dynamic Bayesian Networks encompass thus all State-Space Models (SSMs), including for example Hidden Markov Models (HMMs) [Rab89, CMR05] and Kalman Filter Models (KFMs) [RG99]. In their original form, they are supposed to be stationary. Their parameters (namely the parameters of the conditional distributions) and their structure, once stated or learnt (e.g. through local [SYS$^+$06, Scu10] or global optimization strategies [Mur02, WD09]), remain unchanged through time. Now, non-stationary effects can be modelled by incorporating in DBNs the ability for parameter change [BNI98, PRM00, GH00, PHW01, MPR05, IHS06, ORBD08, FSJW08, GKE10] and/or structure change [PADF02, TL04, Fea06, XM07, GHFX07, WZSS08, RH08, GH09, SKX09, KSX09, KSAX09, FSX09, WKY$^+$11]. DBNs, mostly under the form of KFMs and HMMs, have been widely used in finance [BH, ME07] and econometrics. Finally, the study of non-stationarities can be tackled from the point of view of transfer learning. Indeed, in this very recent and dynamic sub-field of machine learning, the training data and the test data are not supposed to be identically distributed nor even to belong to the same feature space [PY10]. Transfer learning studies then the conditions, restrictions and transfer from the training set to the test set. It is closely related to multi-task learning which aims at learning the knowledge common to multiple tasks [Car97, BDS03] and semi-supervised learning [Zhu05]. It also encompasses the field of domain adaptation [IM06, BMP06, BDBCP07], where the feature spaces are identical and where the training data is made of two particular subsets. The distribution of the first - usually large - subset is different but related to the test set distribution, while the second - usually much smaller - subset is distributed as the test set. When there is no second subset, we speak of learning under sample selection bias [Hec79, Zad04] or covariate shift [Shi00]. These closely related tasks compose the transductive transfer learning setting [PY10]. The knowledge to transfer can take several forms. The statistician usually determines the form under which it can be best transferred. For example, it can be assumed that only a limited part of the training set can be relevant to learn the test set distribution. This problem is then addressed by developing ad hoc strategies for increasing the weights of the useful data in the learning process [DYXY07, JZ07, HSG$^+$07, BBS07, QCSSL09]. Another problem can be to learn the most appropriate parametric model based representation [LP04, EP04, BCW08] or feature based representation [Jeb04, BMP06, BDP07, AMPY07, AEP08] of data in order to enhance the similarity between the two data sets and to improve the learning performances.

Transfer learning also provides a good framework to mention pitfalls common to all model construction approaches, particularly in an evolving environment. A first such trap is overfitting, which occurs when a numerical model, too sophisticated, is unable to generalize. Another related problem is data snooping, which refers to the systematic (re)use of the same training data for inference or model selection [LM90, STW99, Whi00, CG06]. Both problems can be seen as too ambitious attempts of knowledge transfer. We believe that preventing these problems requires a specific methodology enabling model automatic construction, backtesting and analysis. We particularly emphasize the necessity for limiting or abolishing human intervention during the database and model construction. It is crucial to propose an efficient model construction procedure but also to draw valid conclusions about the statistical nature of the data. In that perspective, rigorous backtesting procedures have to implemented to test efficiency the relevance of the model. Besides, developing in parallel interpretablity functionalities is very helpful to analyze the backtesting results and to compare them to the mainstream opinions and theories in economics and finance. This interpretation effort can lead to a deeper understanding of the data but it can also question the reliability of the model and lead to dramatic/potential improvements.

To sum up, the problem of trend prediction raises related and ambitious challenges. The first one is to choose a relevant time series representation method. Indeed, it has to help to define the concept of trend properly and in an economically meaningful way. It also has to turn input time series into more stationary and interpretable features. Secondly, the next challenge is to design a model construction method capable of handling the probable features time dependence and non-stationarity. Additionally, the validity of the representation and of the model have to be checked by specific backtesting plans. Their role is also to analyze (and to help avoiding) the typical effects of overfitting and data snooping. They finally have to enable a better understanding of data via interpretation functionalities and to enhance database and model construction methods.

Methodology and Practical Contributions

We propose an experimentation protocol for studying and improving the construction of trend predictive models. It consists of the following string of four modules (cf. figure 1), which aims, by being executed repeatedly, at solving the aforementioned challenges.

- The first module, entitled *Data Observation and Modeling Choices*, is a preliminary module devoted to the statement of very general modeling choices and objectives. These objectives are expressed first of all by the choice of the setup, which indicates the nature of the prediction task. For example, we have to decide whether we aim at predicting of real labels - this is the regression setup - or binary or categorical labels - this is the classification setup. We also can choose to order the objects according to their propensity of being of positive labels - this is the ranking setup. This choice of setup is crucial as it is of unequal difficulty and as it conditions the possibility of benefiting from the learning procedure of a certain number of interpretation functionalities. Besides, we can specify hypotheses concerning the data. For example, we can suppose that the distribution of the variables is Gaussian (or not). We also can assume a structure of dependence for the variables, e.g. that they are they follow an autoregressive process. Finally, we assume that these hypotheses are valid for time ranges of both the training and test sets, which have to be selected.

- The main task of the second module, *Database Construction*, is to use a well-chosen time series representation method to transform explanatory and target series into features and labels. The set of features and labels are then split up to form a training set and a test set, according to the previously selected time ranges. Both sets have to be sufficiently homogeneous to enable the use of learning techniques. The distribution of their relative features in are compared in that perspective.

- In the third module, *Model Construction*, we extend the classic statistical model for classification by defining a statistical model for clustering and classification. It is actually a mixture model where the different component distributions - or regimes, unknown and non-parametric, have to be identified and learnt. This is achieved via the construction of a single regime attribution function and of regime specific decision functions. These latter deliver predictions for labels.

- The fourth and last module, *Backtesting and Numerical Results*, provides the results of backtesting plans, which aim at analyzing the relevance of the predictions produced by the previous functions over a "significant" test set. Therefore, the first backtesting plan evaluates the precision of the predictions via appropriate error measures. The second one uses the same output predictions to design trading rules and to compute corresponding profit and loss accounts. Backtesting plans are additionally completed by interpretation functionalities helping the statistician to understand the reasons for any regime attribution or prediction.

These functionalities can be related to measures of influence of features in the predictions and therefore to variable ranking and selection. They also can give indications about the reliability of the model and about its predictions. They decisively contribute to formulating better data hypotheses and enhancing the time-series representation, database and model construction procedures.

The experimentation protocol 1 highlights the constant reassessment and control of each of these steps. This empirical approach demands a meticulous and rigorous linking of the steps. In that perspective, the tuning of parameters of each step of the protocol plays a key role and has to be achieved according to clearly identified procedures.

The experimentation protocol has led to the design of new procedures for each module. The database construction is thus achieved via UniCart, a representation method for univariate series based on the CART algorithm. It consists indeed of a multiscale and piecewise approximation method, implemented by a top-down binary tree structured segmentation procedure. Its linear version enables the approximation of a finite signal by a scale specific succession of linear segments. UniCart is actually a simple method which fulfils many interesting - from an economic and financial point of view as well as from a learning point of view - objectives.

- It is first inspired by technical analysis and defines the concept of trend in an intuitive and familiar war for financial practitioners. Besides, both features and labels are deduced from the UniCart representation, which is very convenient. Finally, working with interpretable features helps to question the economic meaning and relevance of both representation and model construction procedures, to understand their performance and their limits intuitively, and to improve them more easily.

- In a supervised learning perspective, one of the most important achievements of UniCart is to provide a robust database, ensuring that similar financial series have close representing features. Its compression ability, the fact that it can provide a concise description of any financial series, was not an absolute priority as the new learning procedures resist the curse of dimensionality well. However, regarding the representation and model interpretability, it is definitely very useful. Besides, UniCart has allowed us to study the importance of building homogenized features and labels, matching the learning hypotheses better (than the raw samples of variables). Finally, the UniCart representation being made of successive trends existing at various trends led us to question lag and scale effects and prediction abilities.

Secondly, a new transfer (supervised) learning procedure, called Relabeled Nearest Neighbors (RNN), has been developed for the construction of efficient trend predictive models. This meta-algorithm reconsiders simultaneously the problems of learning different regimes (i.e. different distributions of trends and labels), of model selection and of overfitting in an original way. It requires a clustering method as well as a generator of hierarchical partitioning rules, typically a base procedure for growing tree classifiers or regressors. Rather than being focused on the optimization of a performance criterion within different regimes, this procedure is centered on learning a pseudometric based on the relative importance of features in the prediction. It leads to the recursive dyadic partitioning of the training set. This recursive partition enables the computation of a score, which indicates for each training data object its propensity of being of positive label. The goal of the RNN classifier is to transfer this hierarchical partitioning structure as well as the corresponding score from the training set to the test set. This is done via an 1-nearest neighbor procedure.

The last contribution of this thesis concerns the design of RNN specific backtesting plans. The first backtesting plan computes thus the recognition rate and the area under the ROC curve of the RNN classifier. The second one designs trading rules based on RNN predictions and scores and analyzes their profit and loss accounts. Backtesting plans are finally completed by a simple and straightforwardly computed measure of features relevance. In practice, outstanding backtesting results are obtained for a set of major financial time series over very significant historical ranges. They particularly prove that the RNN procedure prevents overfitting (contrarily to the Random

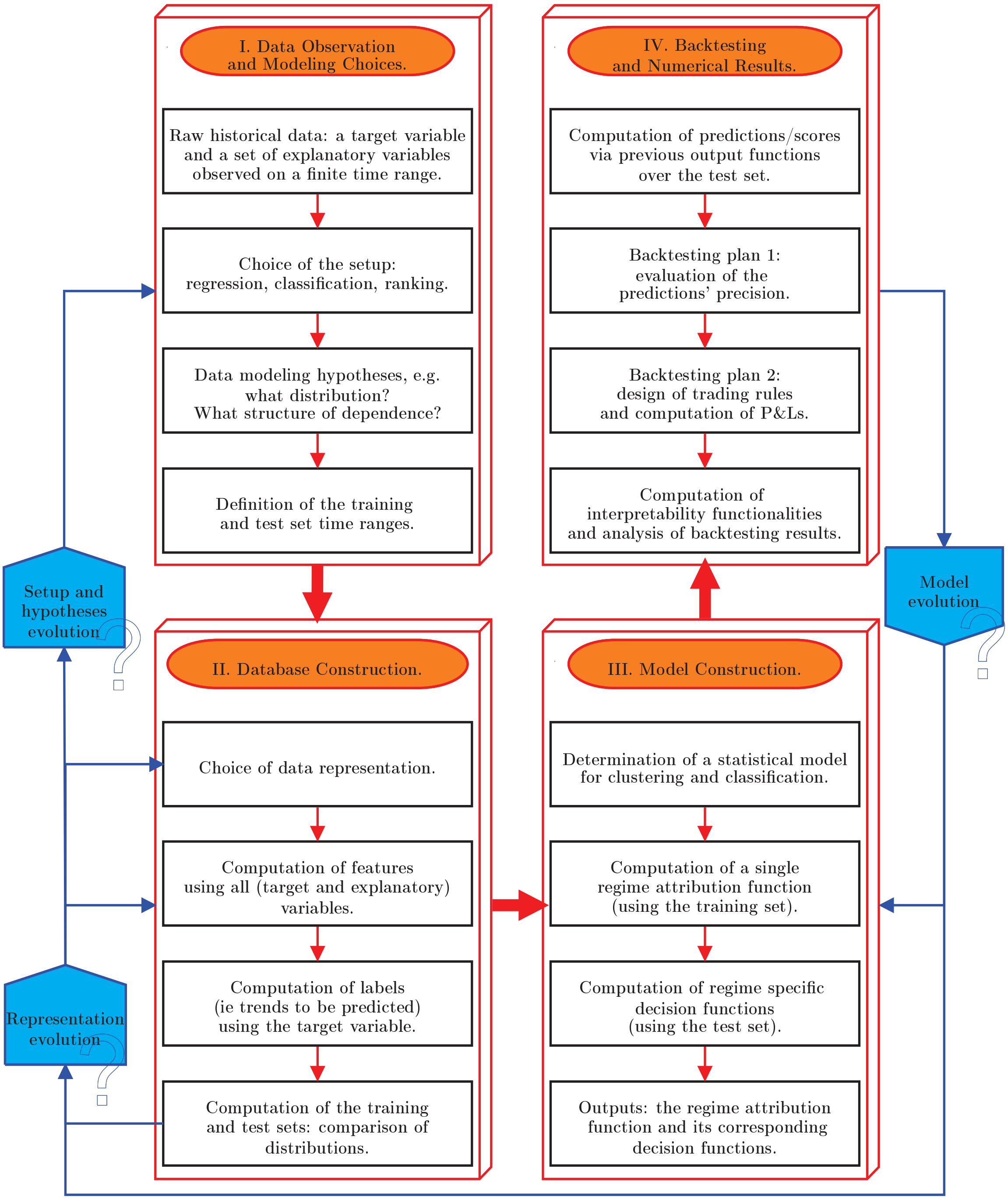

I. Data Observation and Modeling Choices.
Raw historical data: a target variable and a set of explanatory variables observed on a finite time range.
Choice of the setup: regression, classification, ranking.
Data modeling hypotheses, e.g. what distribution? What structure of dependence?
Definition of the training and test set time ranges.
IV. Backtesting and Numerical Results.
Computation of predictions/scores via previous output functions over the test set.
Backtesting plan 1: evaluation of the predictions' precision.
Backtesting plan 2: design of trading rules and computation of P&Ls.
Computation of interpretability functionalities and analysis of backtesting results.
Setup and hypotheses evolution
Model evolution
II. Database Construction.
Choice of data representation.
Computation of features using all (target and explanatory) variables.
Computation of labels (ie trends to be predicted) using the target variable.
Computation of the training and test sets: comparison of distributions.
III. Model Construction.
Determination of a statistical model for clustering and classification.
Computation of a single regime attribution function (using the training set).
Computation of regime specific decision functions (using the test set).
Outputs: the regime attribution function and its corresponding decision functions.
Representation evolution

Figure 1: **Experimentation Protocol**

Forests procedure) and is able to handle heterogeneous data well, without human intervention and user tuning efforts. This allows us to draw reliable conclusions regarding the validity of efficient market hypothesis, the tenets of technical analysis, the existence of regimes, the predictive power of each feature according to its lag and scale, etc.

Organization of the Manuscript

This research work proposes a machine learning point of view on market analysis. It first introduces an experimentation protocol, which has led to the development of new procedures for constructing databases and learning trend predictive models. UniCart, a multiscale piecewise representation procedure, has thus been designed to compute databases, made of homogeneous features inspired by technical analysis. Meanwhile, a new transfer learning procedure, called Relabeled Nearest Neighbors (RNN), has been created to thwart the possible differences between the training and test set feature distributions. It aims at learning a pseudometric in a supervised setup and does not require any tuning subroutine. It has thus allowed us to study reliably the efficient market hypothesis. This document details this research work. It consists of four chapters, summarized hereafter.

A Machine Learning Point of View on Market Analysis

This chapter introduces the efficient market hypothesis. It explains why it stands in contradiction with the main schools of market analysis, which are fundamental analysis, technical analysis and quantitative analysis. These three points of view are thoroughly presented. We focus then on statistical learning theory and on its applicability to the problems of prediction in finance. In that perspective, the classic learning theory is recalled. Challenges preventing the straightforward use of machine learning techniques for computing predictions on financial series are identified. The treatment of non-iid data in different sub-fields of machine learning is particularly discussed. The transfer learning framework, that we believe to be relevant to avoid overfitting and data snooping, is made explicit. Finally, the experimentation protocol, which sums up our research methodology, is presented. It is made of four modules, whose sequential (re-)execution has led to the formulation of a new model for clustering and classification and to the design of the new procedures UniCart and Relabeled Nearest Neighbors. The three following chapters correspond to the three last modules of the experimentation protocol.

Construction of Economic and Financial Databases

This chapter, which stands for the second module of the experimentation protocol, focuses on the construction of databases for learning trend predictive models. It thus first presents our search for an ideal time-series representation method. It introduces in that perspective the economic and financial variables, which play the role of explanatory and target variables in the fourth chapter. Their inter and intra heterogeneity, which is highlighted, help to recapitulate the stakes of time series representation. Properties of approximation, interpretability and homogeneization are expected from the representation method. Two particular methods are thus emphasized. The first one consists of applying the Kalman filter to linear state-space models. It is a well-known and popular method among quantitative analysts for detrending financial series. The second method is UniCart. It is a multiscale piecewise representation method, which shares close ties with the CART algorithm. It implements a top-down binary tree structured segmentation, what is is presented and illustrated. We then explain how the linear interpolation version of UniCart is used to build databases. These databases are of two types. The first type of database is made of linear trends computed at different scales and lags and of corresponding statistical quantities. The features of the second type of database are exclusively returns, considered at different scales and for the most recent lag only. The stationarity of these two types of databases is investigated using the example of the S&P500 index.

Construction of Trend Predictive Models

This chapter represents the third module of the experimentation protocol. It reports three model construction approaches, called respectively Kalman LagLasso (KLL), UniCart Random Forests (RF) and UniCart Relabeled Nearest Neighbors (RNN). These approaches have been consecutively developed to meet the improvements suggested by the backesting results - presented in the next chapter. The KLL approach states a linear model in the regression setup, where the (Kalman) filtered target variable depends on (Kalman) filtered and lagged explanatory variables. Variable and lag selection is performed by a new Lasso-type method called LagLasso, which is described in detail. Yet, the bactesting results show that the regression setup may be too ambitious. They also underline difficulties in assessing the relative importance of each explanatory variable. Thus, to lower the targeted level of precision, the prediction problem that we consider in the following is expressed in the binary classification setup. UniCart databases are used to train numerical models. The label is the sign of the linear trend (or the sign of a specific return) of the target variable at user-chosen scale and horizon. The learning methods are restricted to tree aggregation procedures, as they produce very interpretable numerical models. Tree classifiers are therefore thoroughly introduced, while usual tree aggregation procedures, such as Boosting, Bagging and Random Forests are carefully reviewed. We particularly focus on Random Forests classifiers and show in chapter IV that too large RF classifiers tend to overfit. We believe that this overfitting incident is due to the non-stationarity of the features, which can be studied in the transfer learning framework. We aim particularly at learning partitioning structures transferable from the training set to the test set. We therefore state a new mixture model for the UniCart features and labels. This model consists of an unknown number of unknown non-parametric component distributions, that we call regimes. The Relabeled Nearest Neighbors (RNN) procedure is introduced for identifying the number of regimes and for computing a regime attribution function and accurate regime specific prediction functions. The RNN procedure is a meta-algorithm, which requires a method for generating hierarchical partitioning rules and a clustering method. It consists of three steps which are made explicit. The first step computes a pseudometric using an ensemble of tree classifiers built on the training set. The second step produces new labels for the training set via a dyadic partitioning performed with the clustering method. Finally, in the third step, new labels are attributed to the objects of the test set using the pseudometric for 1-NN classification.

Backtesting and Numerical Results

In this fourth and last chapter, which corresponds to the fourth module of the experimentation protocol, backtesting plans are introduced. They aim at assessing the accuracy of the constructed trend predictive numerical models. The first plan computes trend recognition rates. The second plan designs trading rules using the predictions of the trends and computes their profit and loss accounts (P&Ls). These plans are completed by interpretation functionalities, which evaluate the feature importance and the prediction reliability. Backtesting results, which have motivated the consecutive development of the model construction approaches, are also presented in this chapter. Predictions of the monthly variations of the S&P500 index, computed by KLL numerical models over the last twenty years, are first thoroughly analyzed. It is shown that the KLL numerical models outperform standard competitors. However, the gap existing between the ambitious goal of predicting real labels and the actual precision of the predictions is highlighted. In addition, while the variable and lag selection step does not allow definitive conclusions to be drawn regarding the relevance of the variables, the decisive role played by the filtering step is emphasized. These observations have justified the switch from the regression setup to the binary classification setup. To enhance interpretation functionalities, the UniCart procedure has been designed. The Random Forests procedure

as well as the Relabeled Nearest Neighbors procedure have been used in conjunction with UniCart databases to build classifiers, called respectively UniCart RF and UniCart RNN numerical models. Such models have been employed for predicting the trends of all the target variables introduced in chapter II (including the S&P500 index) over the last five years. The backtesting results show that UniCart RF classifiers tend to overfit, which is not the case of UniCart RNN classifiers. Besides, these numerical models shed new light on the efficient market hypothesis. It is particularly shown that the most reactive UniCart RNN trading rules perform best. New perspectives of development for the UniCart RNN numerical models are finally suggested.

Chapter I

A Machine Learning Point of View on Market Analysis

> La forme, c'est le fond qui remonte à la surface
>
> Victor Hugo

This research work focuses on the application of machine learning methods to the problem of computing predictions on multivariate time-series as input variables. More specifically, considering a single target financial variable, we aim at identifying its current trend and predicting the next one at a fixed horizon using a pool of explanatory economic and financial variables. This very ambitious task stands in contradiction with the Efficient Market Hypothesis (EMH). Yet, different schools of thought in market analysis, whatever they agree to some extent or disagree with the EMH, propose investment strategies to beat the market. This chapter reviews them carefully. It especially explains why technical analysis, and particularly patterns of traditional technical analysis, can be considered as a first step to applying learning techniques to the prediction of financial series. It also highlights the challenges to overcome. These latter motivate the introduction of an experimentation protocol for the rigorous development of trend predictive models. The approach developed in this thesis is sustained by the experimentation protocol. This chapter is therefore organized as follows. The first section is devoted to the presentation of the main viewpoints in market analysis. Basic notions concerning securities and financial markets are first introduced. The efficient market hypothesis and its different forms, expressing various degrees of skepticism towards the existence of successful investment strategies, are then clearly formulated. Conversely, schools of market analysis are shown to propose their own philosophy and methods for the implementation of such strategies. In that perspective, the grounding principles of fundamental and economic analysis, technical analysis and quantitative analysis are made explicit. Special emphasis is placed on technical analysis patterns as they help to grasp the concept of trend and of data representation from the point of view of practitioners. Specific such patterns are exhibited and their capacity to recognize current and future trends is discussed. At last, technical analysis does not propose automated pattern recognition procedures (nor statistical models for financial variables), which is a severe limit to its systematic use and evaluation. Technical analysis can thus be profitably addressed from a machine learning point of view. The goal of the second section is precisely to introduce the classic statistical learning theory, its conditions of use and the extended learning framework - or alternate methods - enabling to handle time-series. Different learning problems, such as classification and regression, are first formally presented. Their classic reduction to a global problem of risk minimization is recalled. It is shown that this problem can be solved in a great many ways very efficiently, even in the case of a very large number of explanatory variables. Besides, the limits of the statistical learning theory, as for instance the requirement of the restrictive iid assumption, are highlighted. Dealing with non- stationary, time dependent and noisy data is thus extremely challenging. In that perspective, extensions of the usual statistical learning framework to time-series are reported, as well as other approaches in machine learning capable of handling non-stationarity effects. Online, semi-supervised, and transfer learning are particularly focused on. The third section points out the steps - or modules - we considered ideal and necessary to develop a method for learning trend predictive models. These modules are summed up and linked in the experimentation protocol, which is described in details. The constant reformulation, analysis and backtesting of module specific procedures has led to the approach - based on UniCart and RNN algorithms - presented in this thesis. The organization of programs and subprograms implementing this new approach is displayed. It will be further clarified in the following chapters. Finally, the statistical model for clustering and classification is provided and its corresponding notations also introduced.

I.1 State of the Art: Viewpoints in Market Analysis

This section is devoted to the presentation of the mainstream opinions in market analysis about the design of investment strategies beating the market. The efficient market hypothesis and its different forms, expressing various degrees of skepticism towards the possibility of anticipating price changes, are first clearly formulated. Conversely, schools of market analysis, which propose each their own philosophy and methods for devising successful investment strategies, are reviewed. Fundamental analysis is first shown to be based on economical and financial theories exclusively. Secondly, the major role played by the study of investors' psychology and behavior in technical analysis is underlined. According to technical analysts, specific mechanisms of greed and regret of market actors are supposed to create price patters. Some of them are exhibited and their capacity to recognize current and future trends is discussed. Unfortunately, technical analysis does not provide systematic pattern extraction and recognition methods (nor clearly identified statistical models for financial variables). This lack, which is a limit to a reliable use of technical analysis relative methods, naturally leads to the study of quantitative analysis. The main goal of the numerical models produced by quantitative methods is to reproduce stylized facts of financial series. The main quantitative methods are presented. The roles of econometrics in risk management and of stochastic calculus in the pricing of derivatives are thus compared. The reasons of the emergence of econophysics, which proposes to reject classic economic axioms and to place special emphasis on data, are made explicit. In that perspective, the relevance of machine learning techniques in market analysis is motivated. Their use in conjunction with data representation specific of technical analysis are also investigated.

I.1.1 The Efficient Market Hypothesis

Financial Markets and Spot Trading in a Nutshell

Financial markets offer the frame and conditions for exchanging financial products, for instance financial securities, such as stocks and bonds, commodities, foreign exchange rates based instruments, etc. Such products can be exchanged, sold or bought from one moral person to another via spot trading - a classical exchange where products are immediately delivered - or via derivatives [FMJF10]. These latter refer to contracts, which specify a delivery date, and whose financial value derives from the fluctuations of an underlying asset (such as stocks, bonds, etc). Besides, the values of financial products are determined by supply and demand and are freely available informations in the market. However, transactions are anonymous (ideally, it is not possible to know who trades what) and do have a cost. Finally, this thesis is focused on spot trading solely. Securities, commodities and foreign exchange rates are all used later in our trend prediction applications.

The term security is actually a broad notion which includes specific types of financial products. The official definition of a security given in the Securities Exchange Act of 1934 is: "Any note, stock, treasury stock, bond, debenture, certificate of interest or participation in any profit-sharing agreement or in any oil, gas, or other mineral royalty or lease, any collateral trust certificate, preorganization certificate or subscription, transferable share, investment contract, voting-trust certificate, certificate of deposit, for a security, any put, call, straddle, option, or privilege on any security, certificate of deposit, or group or index of securities (including any interest therein or based on the value thereof), or any put, call, straddle, option, or privilege entered into on a national securities exchange relating to foreign currency, or in general, any instrument commonly known as a "security"; or any certificate of interest or participation in, temporary or interim certificate for, receipt for, or warrant or right to subscribe to or purchase, any of the foregoing; but shall not include currency or any note, draft, bill of exchange, or banker's acceptance which has a maturity at the time of issuance of not exceeding nine months, exclusive of days of grace, or any renewal thereof the maturity of which is likewise limited.". A more synthetic definition can be found in [Har11], where a "security is a paper certificates (definitive securities) or electronic records (book-entry securities) evidencing ownership of equity (stocks) or debt obligations (bonds)". Financial securities can be therefore reduced to two classes of financial products:

- debt securities (bonds). A bond is a formal contract to repay borrowed money with interest at fixed intervals [OSon]. The borrowed money is called the principal. It has to be paid at a settlement date, called the maturity. The interest is called the coupon. In the following, we will only focus on government bonds also called treasury bonds. Such bonds are issued by the government and are not exposed to default risk.

- equity securities (stocks). The capital stock of a company is the original capital invested by the founders of the company. The stock of a business is divided into shares, also referred to as stocks, in the plural form. Therefore, stocks represent a fraction of the ownership in a business. From an accounting point of view, a business is a separate entity from its owners. Therefore, the capital stock is a liability on the business. More generally, the accounting equation states that all the economic resources of the company, called assets, equals liabilities plus shareholder's equity [MMBWon]. However, in the stock market, the price per share does not correspond to the equity per share calculated in the accounting statements. Stock markets indeed answer the need of investors to trade stocks. Thus, the price of a stocks fluctuate according to supply and demand conditions.

A commodity however is a physical substance, such as metals and agricultural goods, which can be exchanged in a commodity market under the form of fungible trading units. Examples of commodities are grains, metals, livestock, oil, cotton, coffee, sugar, and cocoa [MM00]. The commodities we focused on are oil and gold. Finally, a foreign exchange rate indicates naturally the rate at which two currencies are exchanged one for another. The foreign exchange rates we investigated are the euro against the US dollar, denoted EUR/USD, and the UK pound sterling against the US dollar, denoted UK/USD.

Now, whatever the type of exchange and the type of products they may propose, financial markets are considered by many academics and professionals as informationally efficient - at least to some extent. This very influential assumption is presented in the next subsection.

Statement of the EMH: why Beating the Market is Impossible

An investor strategy which yields consistent profits over a long period of time is said to beat the market. The question whether it is possible to implement investment strategies beating the market or to predict the fluctuations of speculative asset prices is the subject of intense debate among academics and financial professionals. The efficient market hypothesis (EMH) brings a definite answer. It stems from the seminal work of Bachelier [Bac00] and is properly formulated in [Sam65, Fam70]. According to the efficient market hypothesis, in an efficient market, that is a market where there are rational and well-informed investors competing for a maximal profit, assets are correctly priced. In other words, because markets incorporate information well, prices are at any given time close to the true values of assets [Fam65b]. Another definition of the efficient market hypothesis can be found in

[FMJF10]: "Publicly-available, relevant information about the issuers will lead to correct pricing of freely-traded securities in properly-functioning markets.". Thus, if one supposes that financial markets are informationally efficient and that the only way to beat the market is to exploit new information before competitors do, then it is impossible to beat the market [Shi08].

Now, the efficient market hypothesis can be further completed by the random walk theory, introduced in [Bac00]. It is very precisely developed in [Fam65a]: "the theory of random walks says that the future path of the price level of a security is no more predictable than the path of a series of cumulated random numbers. In statistical terms the theory says that successive price changes are independent, identically distributed random variables. Most simply this implies that the series of price changes has no memory, that is, the past cannot be used to predict the future in any meaningful way". The random walk theory states that under the efficient market theory hypothesis, asset prices are random walks, because price changes respond only to new information, which is by nature unpredictable [Shi08]. Therefore, the random walk theory is a refinement of the efficient market hypothesis. Its consequence is that prices are impossible to predict, or rather that the best prediction for an asset price at any horizon is the actual price of the asset.

Actually, three different versions of the efficient market hypothesis have been defined in [Rob67, Fam70] to test the nature of financial markets. Each of them gives a specific definition for information.

- The weak form says that information concerning past prices exclusively is immediately incorporated into prices. It is therefore impossible to beat the market by knowing past prices.
- The semi-strong form claims that all publicly available information is immediately incorporated into prices. It is therefore impossible to beat the market by knowing any kind of information made public.
- The strong form says all information, public and private, is immediately incorporated into prices. There is no way to beat the market.

The weak form of the EMH stands in contradiction with technical analysis, while the semi-strong rejects both technical and fundamental analysis. In the following, the main schools of thought in market analysis, which are the fundamental, technical and quantitative analyses, are reviewed. It is explained first why they oppose the efficient market analysis, and above all the specific approach they propose to design successful investment strategies.

I.1.2 Fundamental Analysis: Strong Economic and Financial Theoretical Grounds

Fundamental analysis is based on the careful review of companies' balance sheets and on the study of their economic and financial environment. The goal is to favor undervalued stocks before they gain popularity in the market. Fundamental analysts have therefore developed a theory of valuation, called discounted cash flow (DCF) analysis, to compute the present value of any investment and to value any company [Lew03, BMA10]. It aims at forecasting the cash flows $(C_t)_{t \geq 1}$, where $C_t > 0, \forall t \geq 1$, generated by the investment $C_0 < 0$ and at discounting them using a rate $r^\star$, which expresses the investment risk and updates the cost of money. The net present value NPV of an investment is the sum of these forecast and discounted cash flows.

$$NPV_t = C_0 + \sum_{t=1}^{+\infty} \frac{C_t}{(1+r^\star)^t}.$$

Particularly, the net present value of the equity (as well as the assets) of any given company can thus be computed by considering its future expected dividends as cash flows. If the efficient market hypothesis were true, the stock price, reflecting all publicly available information about that company, would be a better estimate of the true value of the company than the net present value of the equity. It would actually be the best estimate of its true value. Fundamental analysts assume on the contrary that the market can fail in valuing companies and that it is possible by conjecturing companies' future cash flows to better guess their true value [Lew03, BMA10].

Now, the discount rate $r^\star$ of a given investment usually stands for a reference rate of return, reckoned by market actors for investments of similar risk [Lew03]. Note for example that there are so-called risk-free investments, such as investments in government bonds. Such investments pay indeed a fixed rate of interest, while the risk is very limited as governments (nearly) always honor their debts. The definition of the discount rate implicitly raises two questions. First, it is necessary to define an appropriate measure of risk. Second, it is equally required to be able to compute the expected rate of return of any investment corresponding to its given risk. The modern portfolio theory (MPT) introduces precisely the concept of risk as the variance of the return of a considered portfolio, which refers to an investment made in a fixed number of assets [Mar52, Mar59]. The driving idea in portfolio theory is diversification, which mechanically reduces risk as soon as the portfolio includes non-correlated assets. The purpose of the portfolio theory is to maximize the expected return of the portfolio for a given risk level, or equivalently to minimize the risk for a given level of expected return.

In classic portfolio theory, portfolio returns are considered over a single period of time. Investors allocate their capital among the assets at the beginning of the time period and do not change this allocation until the end of the period. Time subscripts are therefore not used in the following problem statement. The returns of a number N of assets are thus represented by the random vector $X = (X^i)_{1\leq i\leq N} \in \mathbb{R}^N$, whose mean $\mu \in \mathbb{R}^N$ and covariance $\Sigma \in \mathbb{R}^N \times \mathbb{R}^N$ are supposed to be known (or estimated) by the investor. We call portfolio any allocation of the investor capital in the N assets, which is represented by a vector $w = (w^i)_{1\leq i\leq N} \in \mathbb{R}^N$. Each weight w^i stands for a fraction of the invested capital, which is summed up by the relation

$$w'\mathbf{1} = \sum_{i=1}^{N} w^i = 1,$$

where $\mathbf{1} = (1, \dots, 1) \in \mathbb{R}^N$.
The portfolio return is naturally

$$X_P = w'X = \sum_{i=1}^{N} w^i X^i.$$

We call efficient portfolio any portfolio w that maximizes the expected return

$$\mu_P = \mathbb{E}(X_P) = w'\mu$$

for a given level of risk expressed as the variance

$$\sigma_P^2 = \mathbb{V}ar(X_P) = w'\Sigma w.$$

Equivalently, efficient portfolios can be defined as portfolios minimizing the risk (or the portfolio variance) subject to a given expected return. In this mean-variance framework, finding the efficient portfolios reduces to the following problem of constrained optimization:

$$\begin{aligned} w^\star &= \arg\min_w \frac{1}{2} w'\Sigma w, \\ \text{subject to} \quad w'\mu &= r, \\ w'\mathbf{1} &= 1, \end{aligned}$$

where $r > 0$ is the required expected return. We call finally efficient frontier the set of efficient portfolios solving this problem for all (acceptable) values of r. In the next paragraphs, such solutions are derived under specific assumptions and the form of the corresponding efficient frontier is made explicit.

Solving the problem I.1 is achieved via the Lagrange multipliers method. Set the Lagrangian function:

$$f(w,\lambda_1,\lambda_2)=\frac{1}{2}w'\Sigma w+\lambda_1(r-w'\mu)+\lambda_2(1-w'\mathbf{1}).$$

The first-order conditions for optimality are obtained by differentiating this function with respect to each variable and setting the result to zero:

$$\begin{pmatrix}\Sigma & \mu & \mathbf{1}\\ \mu & 0 & 0\\ \mathbf{1}' & 0 & 0\end{pmatrix}\times\begin{pmatrix}w\\ -\lambda_1\\ -\lambda_2\end{pmatrix}=\begin{pmatrix}\mathbf{0}\\ r\\ 1\end{pmatrix},$$

where $\mathbf{0}=(0,\dots,0)\in\mathbb{R}^N$.
If we suppose that Σ is positive definite, which implies that all the N assets are risky (i.e. of strictly positive variance), and that the vectors $\{\mu,\mathbf{1}\}$ are linearly independent, then the above matrix is invertible. This yields the unique solution:

$$w^\star=\Sigma^{-1}\begin{pmatrix}\mu & \mathbf{1}\end{pmatrix}A^{-1}\begin{pmatrix}r\\ 1\end{pmatrix},$$

where $A=\begin{pmatrix}\mu & \mathbf{1}\end{pmatrix}'\Sigma^{-1}\begin{pmatrix}\mu & \mathbf{1}\end{pmatrix}$.
Consequently, if two efficient portfolios u and v are of respective returns r_u and r_v, then the efficient portfolio of return $\alpha_u r_u+\alpha_v r_v$ is necessarily the linear composition $\alpha_u u+\alpha_v v$. This remark plays an important role in the derivation of the Capital Asset Pricing Model (CAPM). Besides, the following relationship between the portfolio variance σ_P^2 and the portfolio expected return μ_P can be straightforwardly deduced:

$$\sigma_P^2=\begin{pmatrix}\mu_P & \mathbf{1}\end{pmatrix}A^{-1}\begin{pmatrix}\mu_P\\ 1\end{pmatrix}.$$

The efficient frontier in the space generated by $\{\mu_P,\sigma_P^2\}$ is therefore an hyperbola. It is referred to as the efficient frontier for risky assets and represented in black in the figure I.1. Besides, the portfolio $w^\star_{MV}=\frac{\Sigma^{-1}\mathbf{1}}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}$ of minimum variance $\sigma^2_{MV}=\frac{1}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}$ is represented in red in the same figure.

We suppose now that there is, in addition of the N risky assets, a single risk-free asset of return μ^0 and of associated weight w^0. Using the same notations as previously, the portfolio return is rewritten

$$X_P=w^0X^0+w'X=w^0X^0+\sum_{i=1}^N w^iX^i.$$

The expected return is

$$\mu_P=\mathbb{E}(X_P)=w^0\mu^0+w'\mu,$$

while the variance remains

$$\sigma_P^2=\mathbb{V}ar(X_P)=w'\Sigma w.$$

The constrained optimization problem becomes:

$$\begin{aligned}w^\star &= \arg\min_w \frac{1}{2}w'\Sigma w,\\ \text{subject to}\quad w^0\mu^0+w'\mu &= r,\\ w^0+w'\mathbf{1} &= 1.\end{aligned}$$

The corresponding Lagrangian function is then:

$$f(w^0,w,\lambda_1,\lambda_2)=\frac{1}{2}w'\Sigma w+\lambda_1(r-w^0\mu^0-w'\mu)+\lambda_2(1-w^0-w'\mathbf{1}).$$

The derivative equations form the following system:

$$\begin{pmatrix} 0 & \mathbf{0}' & \mu^0 & 1 \\ \mathbf{0} & \Sigma & \mu & \mathbf{1} \\ \mu^0 & \mu & 0 & 0 \\ 1 & \mathbf{1}' & 0 & 0 \end{pmatrix} \times \begin{pmatrix} w^0 \\ w \\ -\lambda_1 \\ -\lambda_2 \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{0} \\ r \\ 1 \end{pmatrix}.$$

We suppose again that Σ is positive definite, and that the vectors $\{(\mu^0, \mu), (1, \mathbf{1})\}$ are linearly independent. The previous system, therefore invertible, yields the unique solution:

$$w^{0,\star} = \frac{\begin{pmatrix} \mu & \mathbf{1} \end{pmatrix} A^{-1} \begin{pmatrix} r \\ 1 \end{pmatrix}}{\begin{pmatrix} \mu & \mathbf{1} \end{pmatrix} A^{-1} \begin{pmatrix} \mu^0 \\ 1 \end{pmatrix}}$$

and

$$w^\star = (1 - w^0) w^{risky,\star}, \qquad w^{risky,\star} = \frac{\Sigma^{-1}(\mu - \mu^0\mathbf{1})}{\mathbf{1}'\Sigma^{-1}(\mu - \mu^0\mathbf{1})}.$$

The efficient portfolio consists therefore of an investment of weight w^0 in the risk-free asset and of an investment of weight $(1 - w^0)$ in a specific portfolio made of only risky assets. Notice indeed that $w^{risky,\star}$ does not depend on the required return r.
The efficient frontier in the space generated by $\{\mu_P, \sigma_P^2\}$ is therefore a straight line, called the capital market line, which goes through the points $\{\mu^0, 0\}$ and $\{\mu_{risky,\star}, \sigma^2_{risky,\star}\}$. This latter point, representing the portfolio $w^{risky,\star}$, can be shown to be the tangent point to the hyperbola (being the efficient frontier for only risky assets) of the Capital Market Line. The portfolio $w^\star$ is usually referred to as the tangency portfolio. Both the Capital Market Line and the tangency portfolio are represented in blue in the figure I.1. Another geometric and intuitive way of defining the tangency portfolio is to see it as the portfolio maximizing the Sharpe ratio [Sha66]:

$$SR(w, \mu_0) = \frac{w'\mu - \mu_0}{w'\Sigma w}.$$

Finally, any rational investor applying the mean-variance methodology would split its capital between the risk-free asset and the tangency portfolio. This is the key concept of separation [Tob56, Mer92].

The Capital Asset Pricing Model (CAPM) [Sha63, Sha64, Lin65] can be derived from the important aforementioned results of modern portfolio theory. It is based first on the existence of a risk-free asset. Secondly, it requires that all the investors forecast the same values of the expected mean $\{\mu_0, \mu\}$ and of the covariance Σ of the $N + 1$ assets. It finally assumes that they all aim at building efficient portfolios, possibly with various required returns. We consider the linear composition of all the investors' efficient portfolios. It consists of an investment in the risk-free asset and of the linear composition of investors' efficient portfolios for risky assets only. Using previous results, we know that the latter is necessarily an efficient portfolio too, denoted M, and usually referred to as the market portfolio. All the investors' efficient portfolios define thus a Capital Market Line.

Consider now the portfolio $P(\alpha)$ consisting of a fraction α of capital invested in an arbitrary asset i and of a fraction $(1 - \alpha)$ invested in the market portfolio M. The first two moments of this portfolio are:

$$\begin{aligned} \mu_{P(\alpha)} &= \alpha\mu_i + (1-\alpha)\mu_M, \\ \sigma^2_{P(\alpha)} &= \alpha^2\sigma_i^2 + 2\alpha(1-\alpha)\sigma_{i,M} + (1-\alpha)^2\sigma_M^2, \end{aligned}$$

The market portfolio $M = P(0)$ belongs to two efficient frontiers. The first one is obtained via a mean-variance analysis applied to the N risky assets. The second one stems from the same analysis operated on the asset i and the portfolio M. These efficient frontiers are hyperbola, which intersect

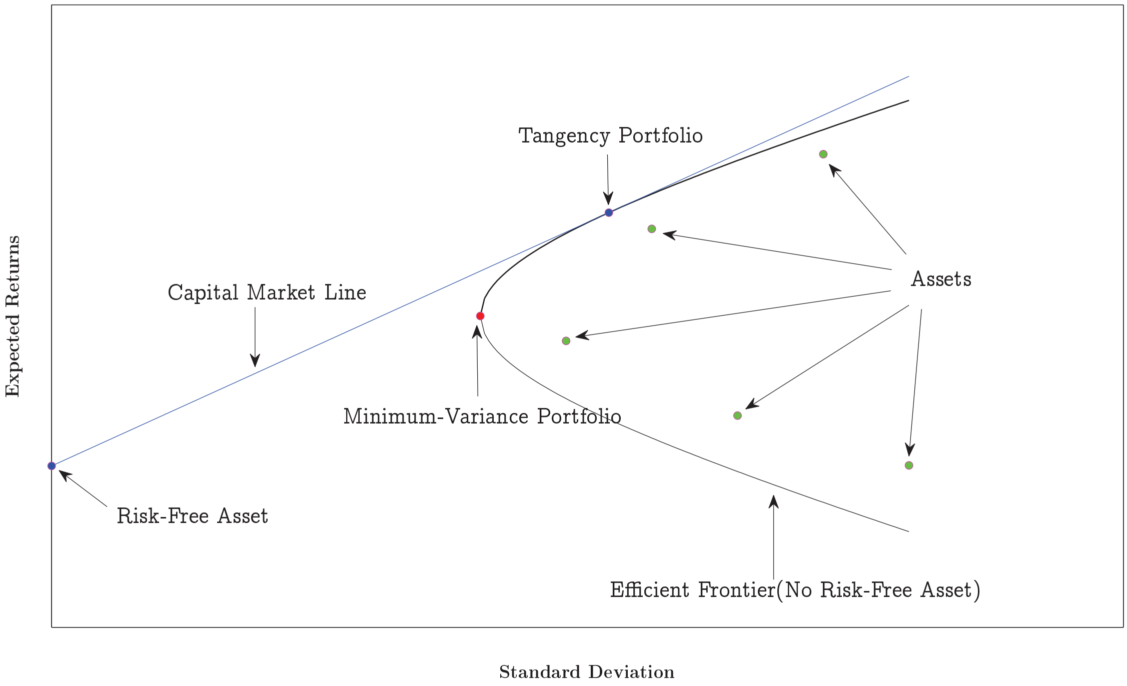


Figure I.1: **MPT**

at $M = P(0)$. Besides, they do not cross the Capital Market Line. Indeed, it would mean otherwise that there exist portfolios which are superior to the efficient frontier of the risky assets. Therefore, the two efficient curves are tangent at $M = P(0)$. Equalling the slopes of their tangents at this point yields the following relationship:

$$\frac{d\mu_{P(\alpha)}}{d\sigma_{P(\alpha)}}_{\{\alpha=0\}} = \frac{\frac{d\mu_{P(\alpha)}}{d\alpha}}{\frac{d\sigma_{P(\alpha)}}{d\alpha}}_{\{\alpha=0\}} = \frac{\mu_M - \mu_0}{\sigma_M}.$$

Basic manipulations finally lead to the Capital Asset Pricing Model:

$$\mu_i = \mu_0 + (\mu_M - \mu_0)\frac{\sigma_{i,M}}{\sigma_M^2}.$$

It is also often expressed as follows:

$$\mu_i - \mu_0 = \beta_i(\mu_M - \mu_0),$$

where $\beta_i = \frac{\sigma_{i,M}}{\sigma_M^2}$ is called the beta-coefficient of the asset i, $(\mu_M - \mu_0)$ the market risk premium and $\mu_i - \mu_0$ the risk premium on the asset i. This formula enables the comparison between the risk on the asset i and the market risk. For example, if $\beta_i > 1$, then the asset i is riskier than the market. More generally, a given asset is riskier than another one if it has a higher β.
The Capital Asset Pricing Model can also be derived from the simple one-factor model:

$$X^i = \mu_0 + \beta_i(X^M - \mu_0) + \varepsilon_i,$$

where ε_i is a normal random variable independent of X^M. In this formula, β_i stands for the systematic risk, i.e. the risk that is common to the asset i and to the market and that can therefore not be diversified away, while ε_i refers to the unsystematic risk. This is summed up by the following relationship:

$$\mathbb{V}ar(X^i) = \beta_i^2 \mathbb{V}ar(X^M) + \mathbb{V}ar(\varepsilon_i),$$

where $\beta_i^2 \mathbb{V}ar(X^M)$ is a fraction of variance of the asset i explained by the market. More specifically, β_i measures the sensitivity of the asset i to the overall market movements.
Finally, we call Security Market Line the graphical representation of the Capital Asset Pricing Model formula in the space generated by the risk β and by the expected return r:

$$r = \mu_0 + \beta(\mu_M - \mu_0).$$

It allows the investor to distinguish between overvalued and undervalued assets.More specifically, it can be employed for computing the discount rate $r^\star$ of a given investment made in a specific asset. It enables therefore the computation of the investment net present value, in other words the valuation of the investment.

The CAPM can be extended to sophisticated linear factor models, such as in the Arbitrage Pricing Theory (APT) [Ros76a, Ros76b]. In this framework, asset returns are directly assumed to follow a linear factor model structure, the factors being possibly financial returns or macro-economic variables, that can be the GDP, inflation indicators, the unemployment rate, the housing market, the household savings rate, corporate profits, etc. It can then be shown that, in the absence of arbitrage (and under complementary technical assumptions), the expected asset returns are linearly related to their covariances with the factors, which are again seen as measures of risk. Furthermore, the use of such factor models is motivated by the need to identify the first two moments μ and Σ of a given set of assets before performing any risk and return analysis. It is interesting to note that fundamental analysis has thus a direct link with quantitative analysis, which will be presented in a next subsection. As a conclusion, a typical investor can use the CAPM or any other relevant factor method to compare systematic risks of assets and to value companies and assets. Most often however, investors complete this multivariate analysis by a more bottom-up approach, mainly by analyzing the financing decisions taken by each company, their influence on the capital structure of the company and on its popularity in the market (cf. [BMA10] for more information on financing decisions, payout policies and risk management aspects). They may also be interested in enriching their investment strategy using technical indicators. Technical analysis proposes indeed a totally different point of view (than fundamental analysis) on market analysis. It is briefly reviewed in the next part. Geometric technical patterns are particularly emphasized and commented.

I.1.3 Technical Analysis: Representations, Patterns and Predictions

Technical analysis aims at identifying or predicting the trend of financial instruments. Its grounding hypothesis states that market prices contain all relevant information for this identification or prediction task [Sew08]. Unlike fundamental analysis, technical analysis is not motivated by any underlying economic theory [BB02]. It focuses instead on the influence of investor mimicry in the price formation and on specific buying and selling mechanisms, which are revealed by geometric shapes produced by prices and volumes. More specifically, these patterns usually consist in specific consecutive trends, implicitly defined as price directions which may exist at different scales, possibly completed by characteristic values of particular indicators (such as volume ratios). According to [Gay90, Bol67, PF01, Mur99, Ach00, Pri02, EMB07, BB02], they could be used successfully for trend prediction and for determining adapted trading dates. Technical analysis stands therefore in contradiction with all the forms of the efficient market analysis. Various quantitative studies evaluating the performance of technical strategies also point out the usefulness of technical analysis [BLL92, LeB96, DNW96, NW99, FRGMSR00, LMW00]. These positive views are however tempered in [PI04], which highlights that the majority of studies supporting technical analysis propose biased backtesting procedures (subject to data snooping effects, ex post parameter selection, etc).

Besides, technical analysis is explicitly centered on the concept of trend. It defines the trend of a price as a series of consecutive ascending or descending bottoms and tops which define a trend channel [BB02, Mur99]. The trendline connecting the tops is called the resistance level and stands for the selling pressure - i.e. the supply. Meanwhile, the trendline connecting the bottoms is the support level and represents the buying pressure - i.e. the demand. New support and resistance levels as well as trend changes are created via breaks through current support or resistance levels. A trend is commonly represented by a linear segment, which locally - "at a given scale" - approximates the price. We focus in the following on patterns specific of traditional technical analysis (also called chart analysis), which are supposed to be trend reversal or trend continuation patterns [BB02, Mur99]. Trend channels (cf. figures I.2 and I.3) are thus patterns of trend continuation.

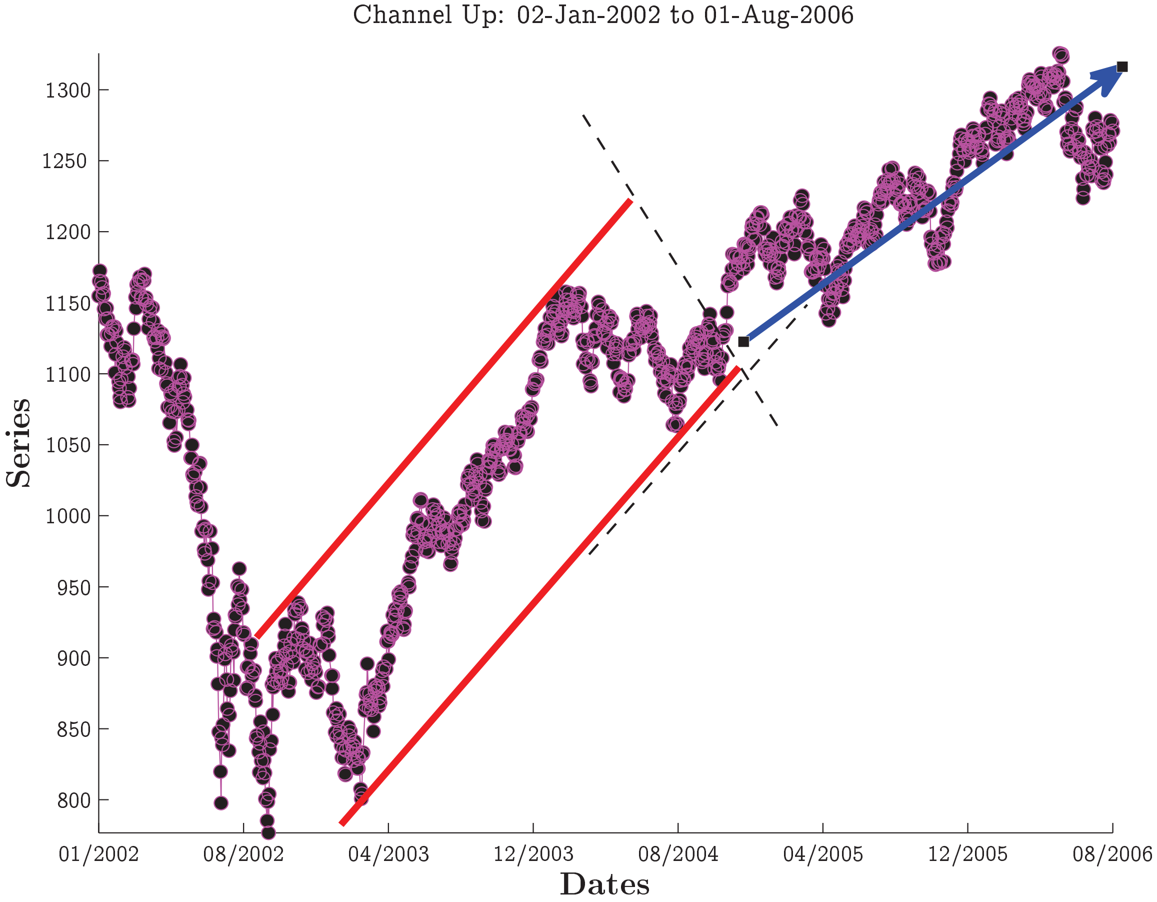


Figure I.2: **SP500 Channel Up**

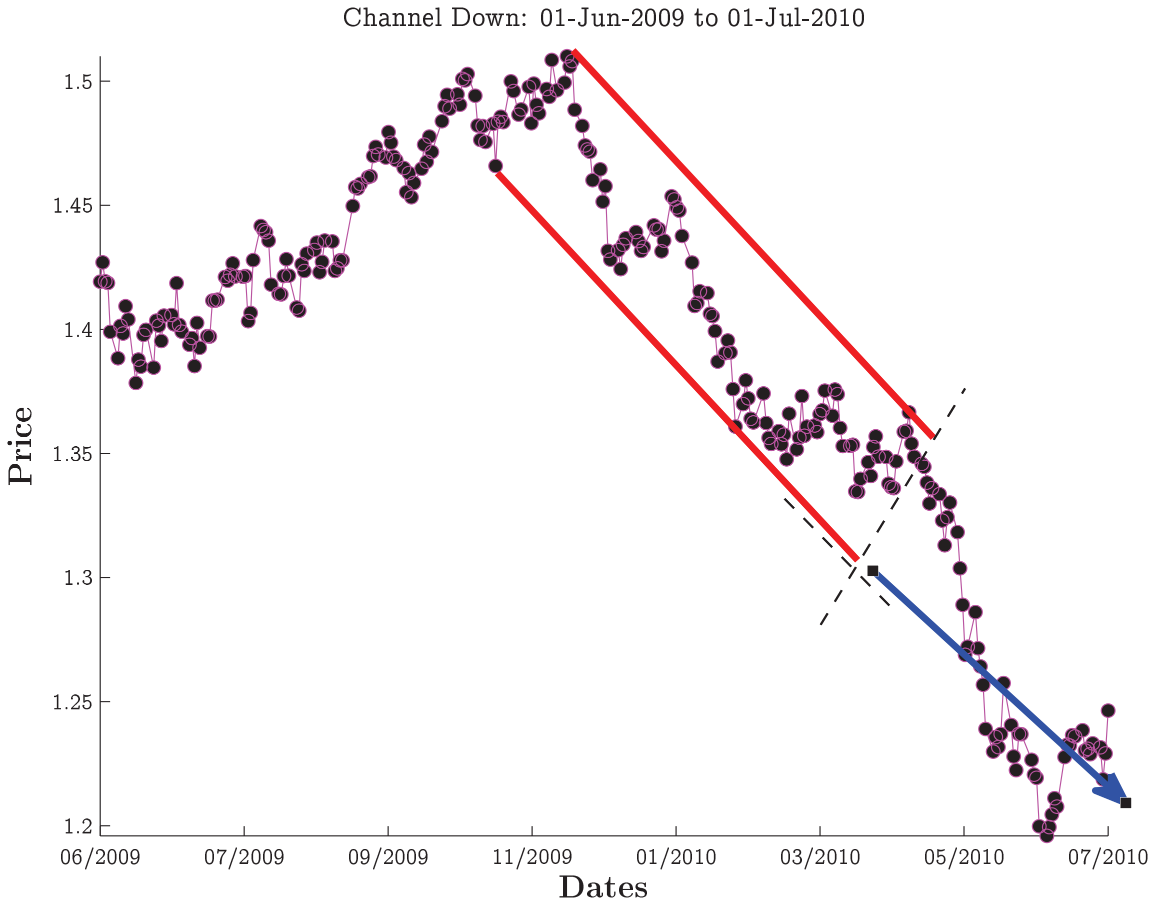


Figure I.3: **EURUSD Channel Down**

Technicians also mention flags, actually small rectangles contained in a broader upward or downward movement and oriented against this movement. Likewise, pennants are also short-term trend continuation patterns, and usually consist in small symmetrical triangles (cf. figures I.4 and I.5). According to technicians, these patterns are formed when some of the market actors stop trading. Newcomers, having observed the current trend, can then choose to bet on a continuation of the trend.

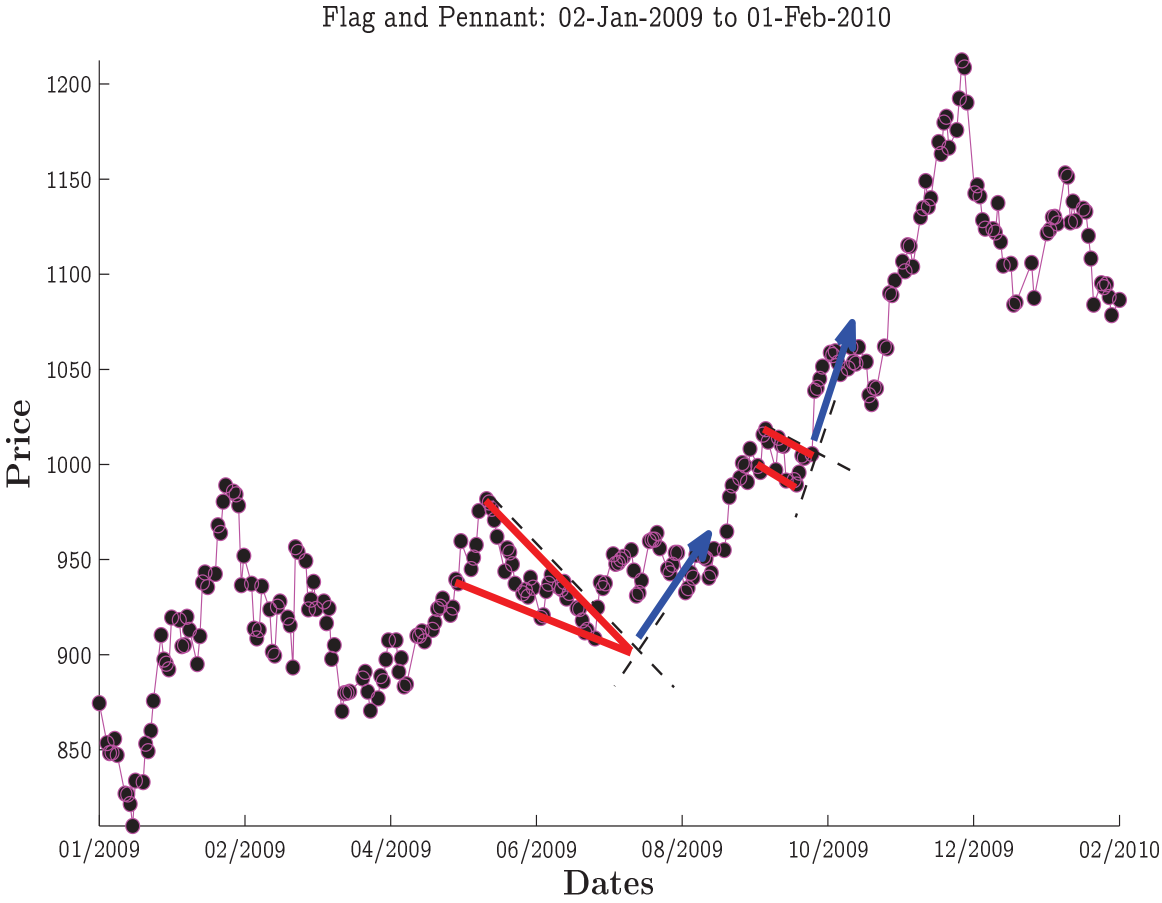


Figure I.4: **GOLD Flag and Pennant**

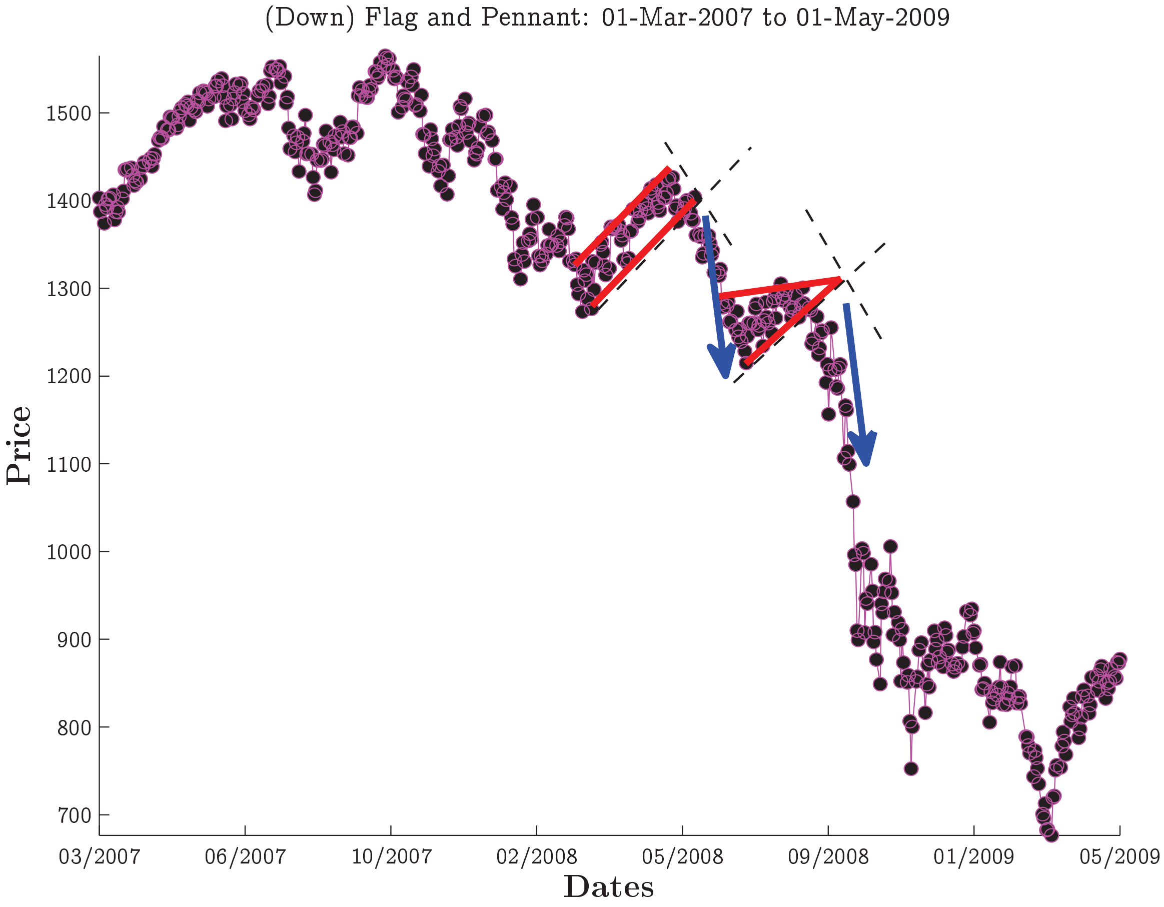


Figure I.5: **SP500 (Down) Flag and Pennant**

Now, triangles are very popular figures among technicians. Although not necessarily contained in a clear major trend, they are considered as trend continuation patterns. They are distinguished between ascending (cf. figures I.7 and I.9)), symmetrical (cf. figure I.6) and descending triangles (cf. figure I.8). An ascending triangle is made of a horizontal resistance level and of an upward support level, which indicates that the buying pressure increases while the market supply remains stable (for example as long as a large sell order has not been executed). Conversely, a descending triangle is made of a horizontal support level and of a downward resistance level. Both ascending and descending triangles are supposed to be reliable trend continuation patterns. On the contrary, a symmetrical triangle is made of a downward resistance level and of an upward support level. Technicians have to wait for one of these trendlines to be broken to know the orientation of the next movement. Therefore, a symmetrical triangle is difficult to anticipate [BB02].

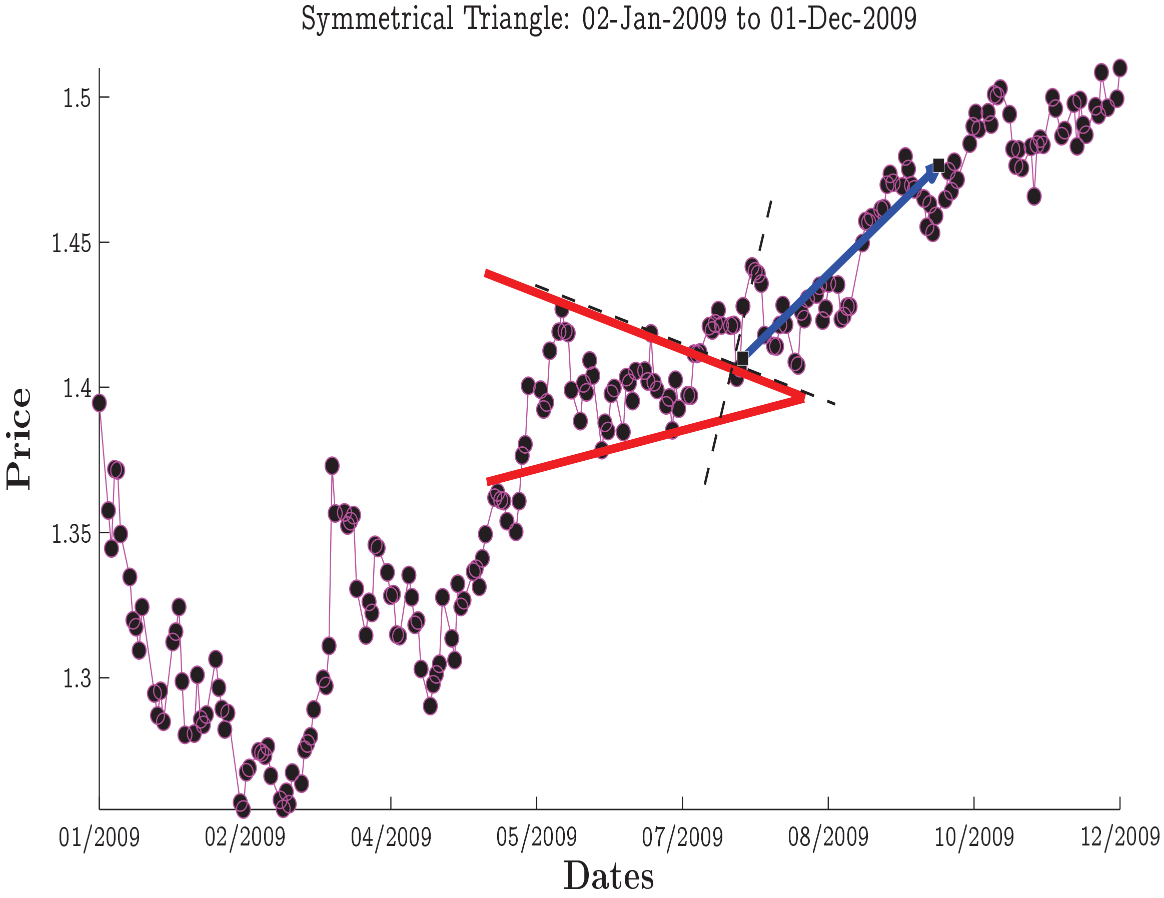


Figure I.6: **EURUSD Symmetrical Triangle**

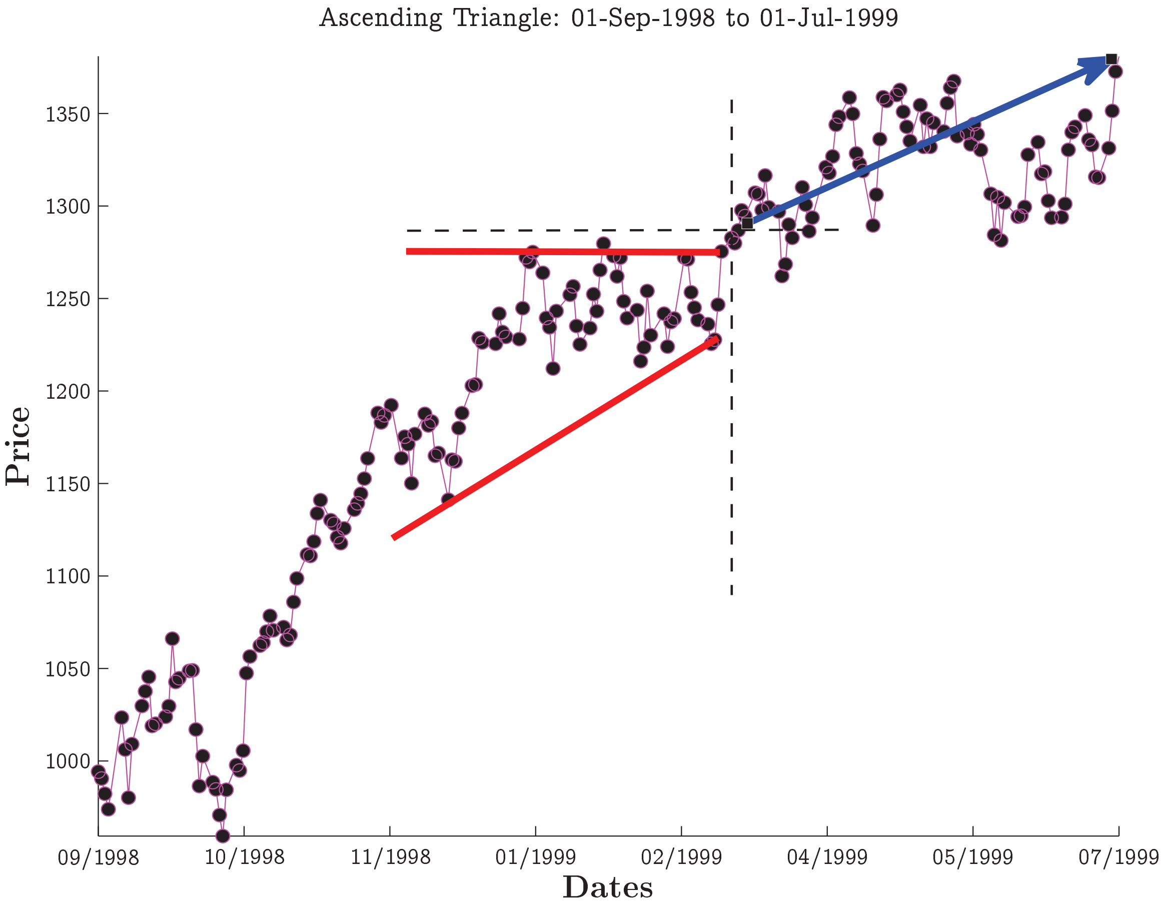


Figure I.7: **SP500 Ascending Triangle**

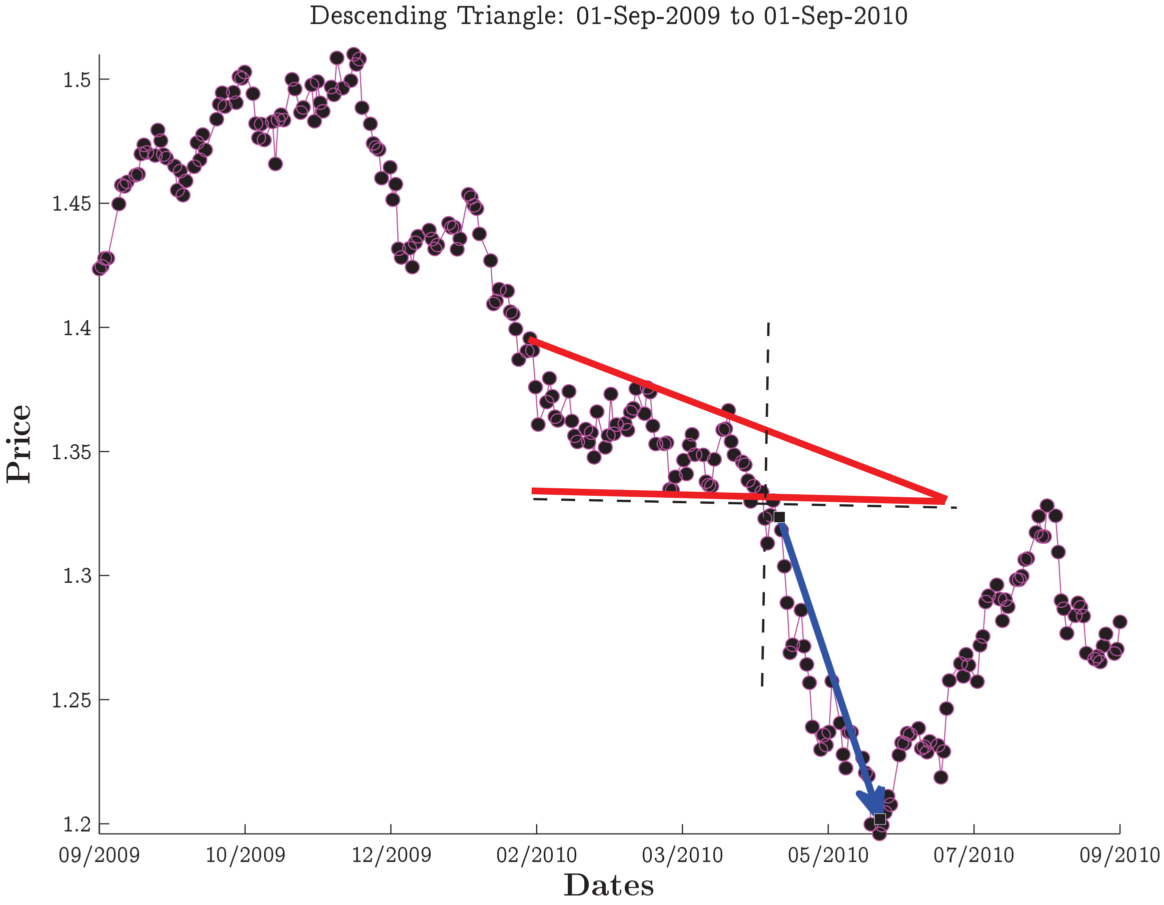


Figure I.8: **EURUSD Descending Triangle**

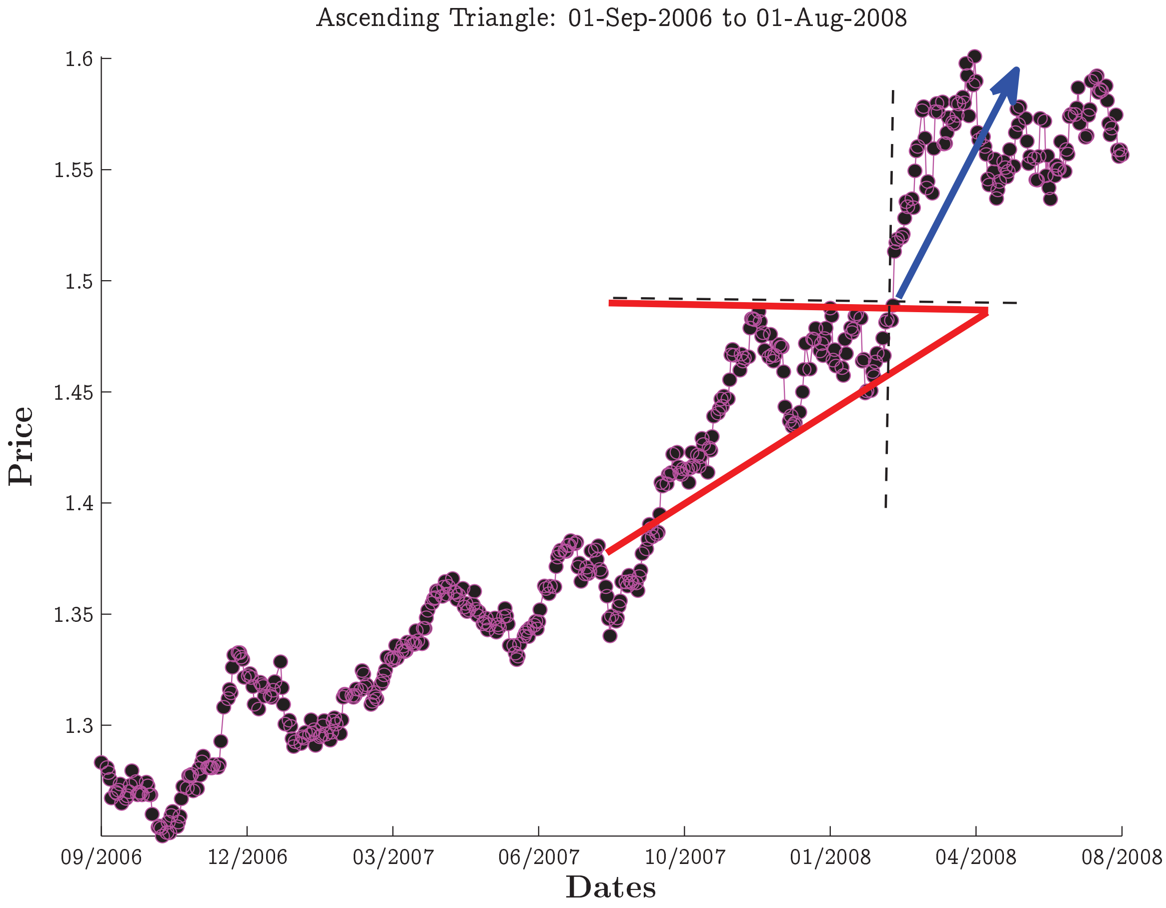


Figure I.9: **EURUSD Ascending Triangle**

Trend reversal patterns are somehow more complex. They are based on investors' tendencies to mimicry. Technicians assume indeed that investors, by alternately regretting having missed a profit opportunity or by greedily wanting to reproduce one, shape consecutive cycles, which are referred to as head and shoulders (cf. figures I.10 and I.11), triple and double bottoms (cf. figure I.12), as well as triple and double tops (cf. figure I.13). Those patterns are also usually confirmed by specific trading volume variations (cf. [BB02]).

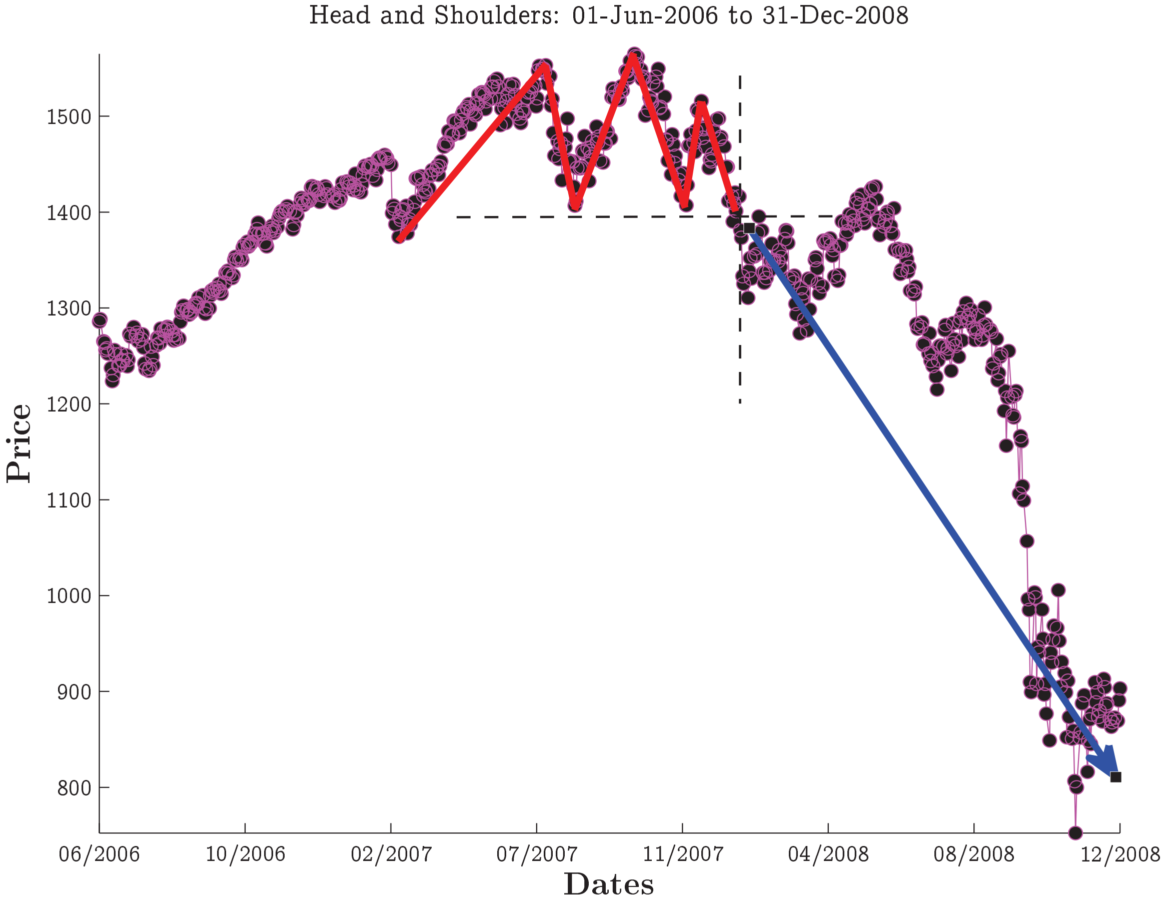


Figure I.10: **SP500 Head And Shoulders**

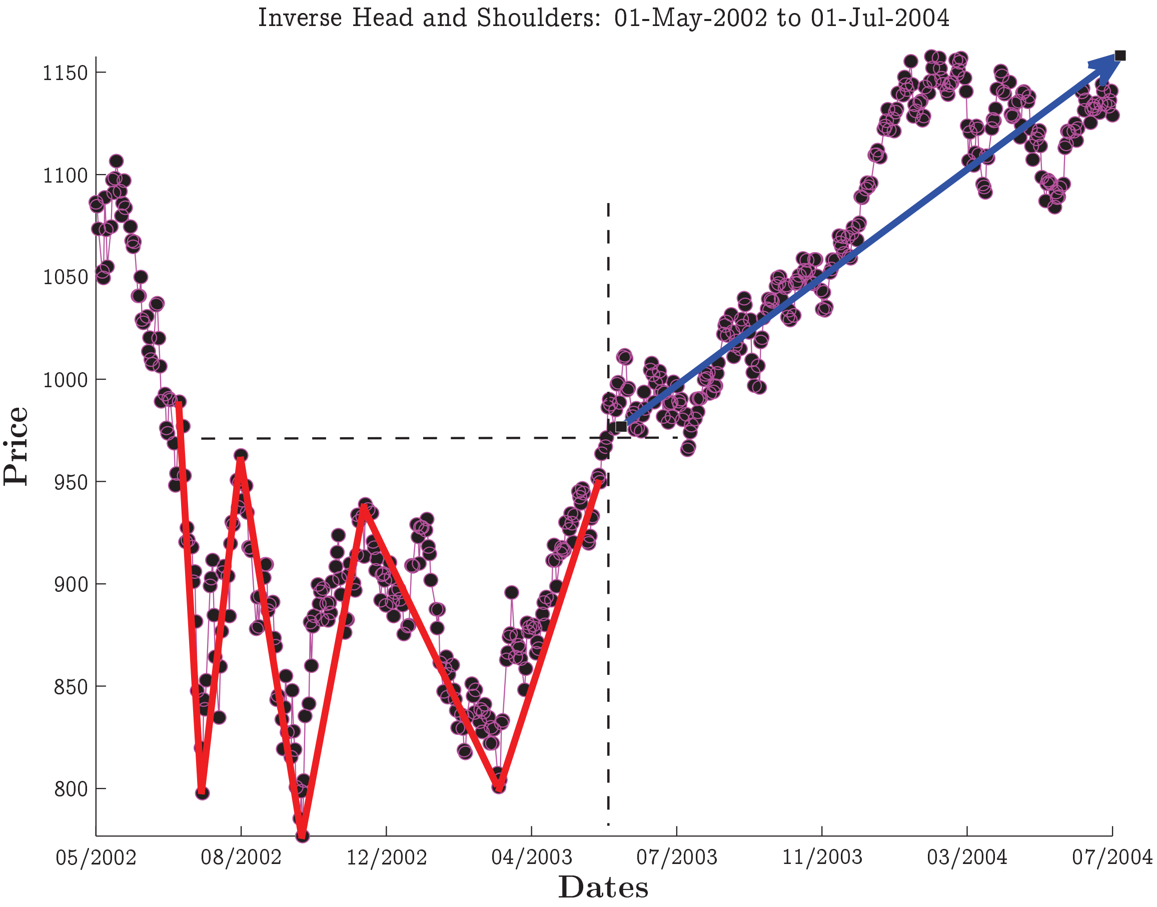


Figure I.11: **SP500 Inverse Head And Shoulders**

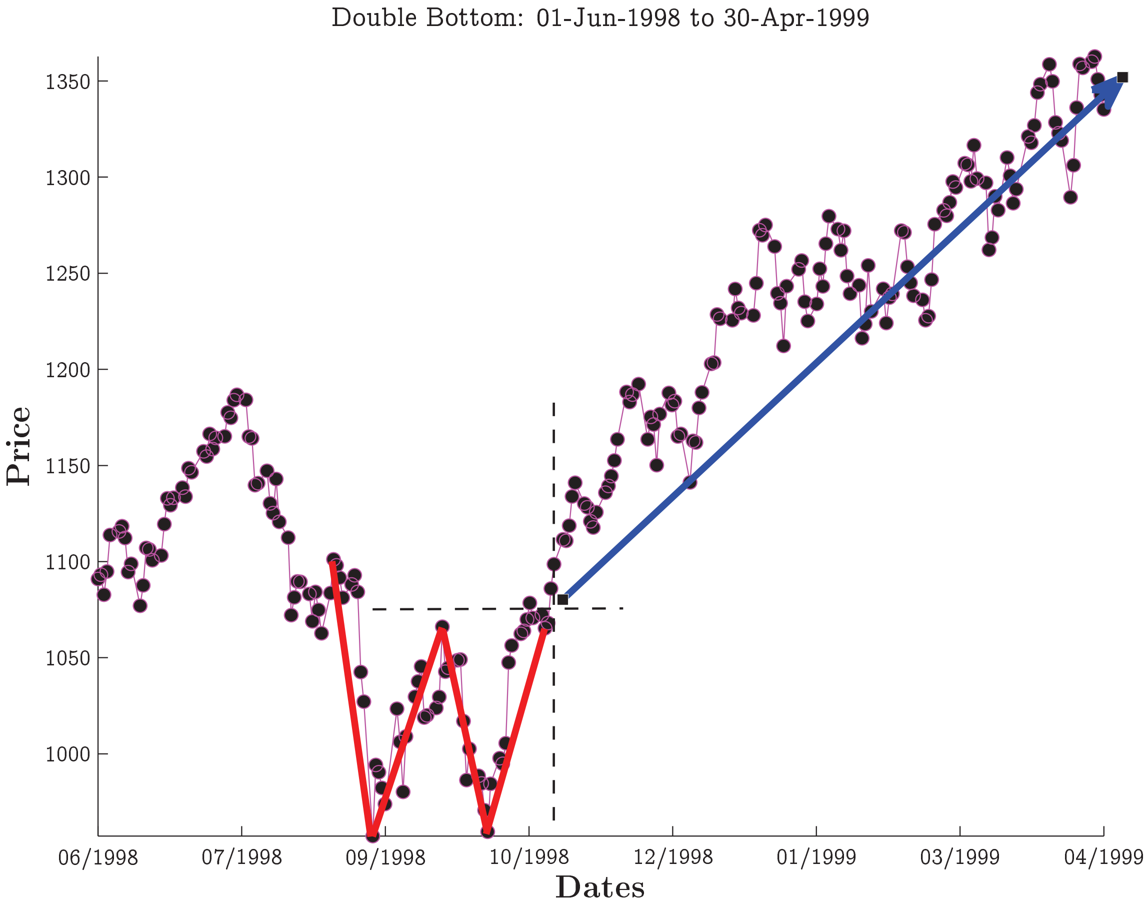


Figure I.12: **SP500 Double Bottom**

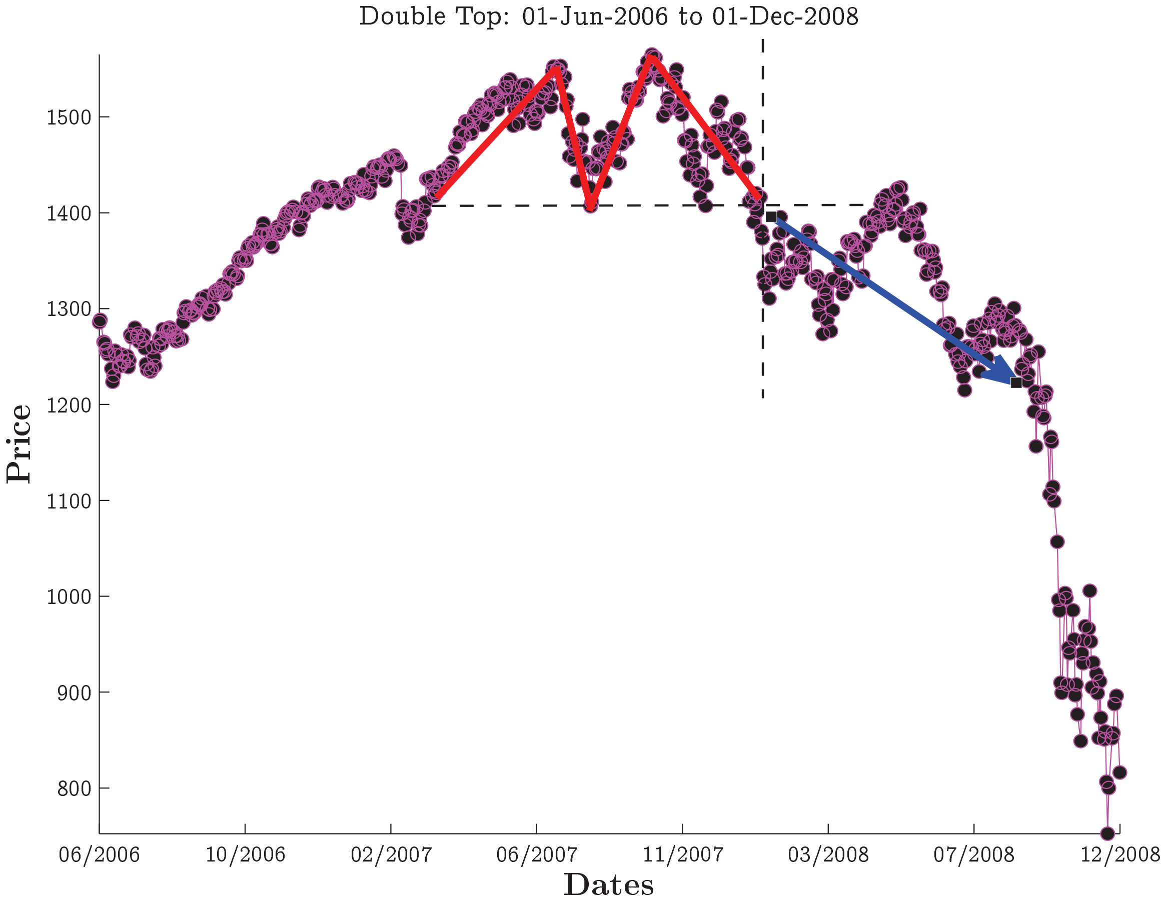


Figure I.13: **SP500 Double Top**

This list of patterns is far from being exhaustive. Yet, it illustrates first the trend based representation system for financial series, which is extensively used in technical analysis. Secondly, it also allows us to understand in an intuitive way the critics formulated against technical analysis. For [Sew08, Aro06], technical analysis - particularly chart analysis - is based on the representativeness heuristic. This psychological phenomenon, introduced in [TK74], refers to the process of intuitive classification, that is accomplished by people in everyday life [Aro06]. This heuristic assumes that each object, member of a given class, displays features which are characteristic of that class. Classification is therefore achieved by evaluating the similarity between the features of a given object and the distinctive features of each class [Aro06]. The difficulty of a recognition by representativeness resides in the choice of the appropriate set of features describing the objects, and in the selection of the characteristic features of each class. Particularly, in the case of technical analysis, features that are the most visually striking (for example, a set of specific consecutive trends) are not necessarily the most useful to perform classification (i.e. to predict the next trend). The most encountered criticism against technical analysis focuses thus precisely on the subjectivity of the patterns, in other words on the alleged lack of convincing proof that these patterns are characteristic features announcing a trend continuation or a trend reversal. That is why we have developed a time series representation method so that any financial series can be automatically and rigorously turned into representative features. This method will be presented in the next chapter. This is also why we have designed a procedure learning the relationship between representative features (of a set of explanatory variables) and the future trend of a target variable. This procedure will be introduced in the fourth chapter. These attempts of representing and modelling financial series belong actually to the field of quantitative analysis, which is explored in the next subsection.

I.1.4 Quantitative Analysis: Focus on Stylized Facts

Quantitative analysis in finance aims at constructing models for both risk management and derivatives pricing purposes [Meu09, Meu11]. These models are usually designed first to reproduce stylized facts, which are general statistical properties, often qualitative, supported by consistent observations made on a wide range of instruments, markets and time periods [Con01, Con07, Sew11]. Next, on the one hand, in the field of derivatives pricing, modelling approaches are conducted using continuous-time processes (and stochastic calculus). On the other hand, in the field of risk management, they are based on discrete-time processes (mainly econometric models) [Meu09, Meu10, Meu11]. We obviously focus on this latter case in this thesis and that is why we now review key concepts in econometrics and classic econometric models. We also insist on the stylized facts they enable to reproduce.

Econometricians usually prefer to work with the returns or logarithmic returns (X_t) of the (univariate) financial series (Z_t), which are closer to stationary signals. The concept of stationarity plays indeed a central role in the development of econometric linear models. The strong form of stationary states that the joint distribution of any vector of returns $(X_{t_1}, \ldots, X_{t_k})$, where $(t_1, \ldots, t_k) \in {\mathbb{R}_+}^k$, remains constant across time. The weak form only assumes that the mean $\mu_t = \mathbb{E}(X_t)$ and the covariance between lagged returns $\Sigma_{t-k,t} = \mathbb{C}\mathrm{ov}(X_{t-k}, X_t)$, where $k > 0$, are time-invariant. It is a more realistic assumption than the strong form. Linear econometric models have thus been developed for weakly stationary returns.
Another key concept is lag autocorrelation, which refers to the existence of a correlation relationship between weakly stationary lagged returns. Specific tests such as the Portmanteau test [BP70] and the Ljung-Box test [LB78] can be used to check whether the returns are serially correlated. If the returns are found to be autocorrelated, model selection procedures, such as the Akaike information criterion [Aka73], the Bayesian information criterion [Sch78], or the careful examination of the partial autocorrelation function computed at different lags [Tsa05] help to determine the order p of the following AutoRegressive Model $AR(p)$:

$$X_t = \phi_0 + \sum_{i=1}^{p} \phi_i X_{t-i} + a_t,$$

where (a_t) are iid random variables (i.e. white noises) often called shocks, innovations, or innovative residuals, and where the real coefficients (ϕ_i) of the model are usually estimated by the least square method. Furthermore, this model implies the linear recurrent equation between the lag-i autocorrelations $\rho(i) = \Sigma_{t-i,t}$:

$$\rho_k = \sum_{i=1}^{p} \phi_i \rho_{k-i}.$$

The associated polynomial equation is called the characteristic equation of the model:

$$1 - \sum_{i=1}^{p} \phi_i x^i = 0.$$

This equation yields solutions, whose inverses $(x_{i,\star}^{-1})_{1 \leq i \leq p}$ are known as characteristic roots in econometric literature. It can be shown that a sufficient and necessary condition ensuring the weak stationarity of the AutoRegressive process (X_t) is that its the characteristic roots are smaller than 1 in modulus, i.e. $|x_{i,\star}| > 1$ [Tsa05].
It is interesting to note that Moving-Average models can be seen as a class of AutoRegressive models of infinite order:

$$X_t = \phi_0 + \sum_{i=1}^{+\infty} \phi_i X_{t-i} + a_t,$$

which, under specific parameter constraints, reduce to the expression:

$$X_t = \theta_0 + \sum_{i=1}^{q} \theta_i a_{t-i},$$

where q is said to be the order of the Moving-Average model $MA(q)$.
AutoRegressive and Moving-Average models can be refined by introducing a combination of both, called AutoRegressive Moving- Average (ARMA) models. An ARMA process of order (p,q) is given by

$$X_t = \phi_0 + \sum_{i=1}^{p} \phi_i X_{t-i} + a_t + \sum_{i=1}^{q} \theta_i a_{t-i}.$$

Usually, this mixed form requires fewer parameters (than pure AR or MA models) to describe dynamic (weak stationary) data [Tsa05]. Besides, the condition for weak stationarity, computed with the AutoRegressive part of the ARMA model, is the same as it would be for this pure AR model. Therefore, $(X_t$ is weakly stationary if the characteristic roots computed with the autoregressive part are smaller than 1 in modulus.
As a consequence, the lag-i autocorrelation $\rho(i)$ of an ARMA process converges exponentially to 0 as the lag i increases. Yet, specific time series can display a decay of the autocorrelation at a polynomial rate. These long-memory effects can be reproduced by fractionally differenced processes [Hos81].
Moreover, ARMA models can be extended by allowing a fixed number of characteristic roots to equal 1. Simple examples are the random walk, defined by

$$X_t = X_{t-1} + a_t,$$

and the random walk with drift

$$X_t = \mu + X_{t-1} + a_t,$$

where μ is a time trend of the series (X_t) and is called the drift of the model. More generally, AutoRegressive Integrated Moving-Average (ARIMA) models of order (p,d,q) are obtained by differencing d times the series X_t and by modelling the residuals with an ARMA model of order (p,q). This type of non-stationarity, called unit-root non- stationarity, can be evidenced by Dickey-Fuller tests [Phi85, CW88].

Now, the autocorrelation of returns of financial series has been extensively studied. According to many empirical works, returns are usually serially uncorrelated (or weakly correlated) but still dependent, which can be illustrated by the positive, significant and slowly decaying as lag increases autocorrelation of absolute or squared returns [Con07]. This latter phenomenon is commonly referred to as volatility clustering. These observations on the dependence and non- stationarity of returns have been made so often that they are considered as stylized facts [Tay, Con07, Sew11]. Volatility models are therefore constructed as follows [Tsa05]. A first model, for instance an ARMA model, is first stated to filter returns. As the autocorrelation is usually weak, this step often reduces to remove the mean μ_t of the returns:

$$X_t = \mu_t + a_t.$$

The existence of a significant serial correlation of the square residuals (a_t^2), referred to as AutoRegressive Conditional Heteroscedasticity (ARCH) effect, is then tested via, for example, a Ljung-Box test [LB78] or via a Lagrange multiplier test [Eng82]. AutoRegressive Conditional Heteroscedasticity models can be employed in an attempt to reproduce ARCH effects. They assume thus that each residual a_t can be described by a volatility factor σ_t multiplied by a white noise ε_t, while the volatility σ_t^2 is given by a linear relation of the lagged squared residuals $a_{t-1}^2, \ldots, a_{t-m}^2$ [Eng82]:

$$a_t = \sigma_t \varepsilon_t, \qquad \sigma_t^2 = \alpha_0 + \sum_{i=1}^{m} \alpha_i a_{t-i}^2.$$

Again, the number of lagged terms m is the order of the model. It can be selected using the partial autocorrelation function, just as is done with AutoRegressive models.
Less specific, Generalized AutoRegressive Conditional Heteroscedasticity (GARCH) models suppose that the volatility σ_t^2 can be expressed as a linear relation of the lagged square innovative residuals

$a_{t-1}^2, \dots, a_{t-m}^2$ and of the lagged volatility factors $\sigma_{t-1}^2, \dots, \sigma_{t-s}^2$ [Bol86]:

$$a_t = \sigma_t \varepsilon_t, \qquad \sigma_t^2 = \alpha_0 + \sum_{i=1}^{m} \alpha_i a_{t-i}^2 + \sum_{j=1}^{s} \beta_j \sigma_{t-j}^2.$$

The order (m, s) has to be specified but in practice, only low order models are used. Note that a great variety of generalized autoregressive conditional heteroscedasticity models has been developed (refer to [Tsa05]) for a complete review). Besides, to prevent the volatility from being governed by a too rigid structure, stochastic volatility models have also been designed, where the description of volatility includes a random additive component [MT90, Tay94, HRS94, JPR94]. Finally, it is worth mentioning linear State-Space Models (SSM), whose specific form is sufficiently general to handle AutoRegressive Integrated Moving-Average Models (ARIMA) or stochastic volatility models [DK01]. Besides, the Kalman filter enables various inferences tasks to be performed on this broad class of models [Ham94]. Linear state-space models and the Kalman filter are discussed in the next chapter.

ARCH and GARCH models are linear in mean and non-linear in variance. Indeed, the mean μ_t is expressed via an ARMA model (linear according to lagged returns and lagged residuals), while the variance σ_t is written as a quadratic function of lagged residuals $a_t = X_t - \mu_t$. Yet, there are more general non-linearity effects in financial time series, as proven in [SL89, Hsi91, ACW97, AP03?]. The study of business cycles, which refer to periods of growth and contraction of the overall economic activity [BM46, SW89, SW88, SW93], provides interesting examples of such non-linearities. Particularly, Markov-switching models [Ham89, KN99] as well as dynamic factor with regime switching models [DR96, KN98, KY95, Cha98, CP10, BG04] have been proven particularly relevant to capture abrupt changes in the distribution of macro-economic and financial variables and to help identifying expansion and recession periods.

As mentioned earlier, if risk management is based on econometrics and discrete-time processes, derivatives pricing is based on stochastic calculus and continuous-time processes [Meu09, Meu10, Meu11]. Yet, there is a deep link between the concepts and the models specific to these two areas of quantitative analysis. The counterpart of ARMA processes are thus Ornstein-Uhlenbeck and Continuous AutoRegressive Moving Average (CARMA) processes [Meu09, Meu10, Meu11]. Famous interest rate models, such as the Vasicek model [Vas77] and the model by Cox, Ingersoll, and Ross (CIR) model [CIR85], derive from Ornstein-Uhlenbeck processes. Long memory is addressed by fractional Brownian motion processes. Besides, Levy processes are actually the continuous-time version of random walks. They include the brownian motion, whose geometric form is used to model stock prices in Black-Scholes model [BS73, Mer73], and Poisson jump processes. As highlighted in [CT08], Levy processes play an important role in modelling non- stationarities. Finally, stochastic volatility models, where volatility follows a stochastic process [CR76, Hes93, Che96, HKLW02], propose along with subordination models processes capable of modelling volatility clustering effects fairly well.

Recent trends in quantitative analysis, particularly at instigation of econophysics - or statistical finance [MS99, Bou02, BP04], recommend to place special emphasis on empirical data to the detriment of classical economics axioms. For example, it is highlighted in [Bou08] that the Black-Scholes model is still very popular among financial practitioners, although it assumes that price changes are Gaussian. Yet, the distribution of returns, especially at high frequencies, is known to have heavy tails - this stylized fact having been checked by many empirical studies [MS99, Con01, Sew11]. According to [Bou02, Bou08], the massive amounts of data recorded in the financial markets should be used to properly question models and to validate, modify or discard them. In that perspective, applying statistical learning methods [HTF01, LGWcs, BBL04, BBL05, Tsy10] to the problems of designing investment strategies is relevant, more and more popular and actively studied, for example in [LMW00, YSC08, CB08, Cha09, FHST10]. It is motivated indeed first by the ability of these modern methods to deal with massive and high-dimensional data sets [BvdG11]. Second, such methods do not require any *a priori* assumptions about the distribution of the data. Learning methods and their applicability to financial series are thoroughly investigated in the next section.

I.2 Machine Learning and Time-Series: Principles and Challenges

This section focuses on the statistical learning theory, which provides a rigorous theoretical framework for the study of prediction problems. These problems are formally introduced as particular instances of a global problem of risk minimization, determined by the choice of the setup (therefore of the model) and of the loss function. Classic assumptions of the learning theory, particularly the hypothesis of independence and identical distribution made on data, are then recalled. Specific model selection algorithms, which aim at solving the minimization problem, are presented as well. The role of the assumptions in the derivation of performance guarantees for such algorithms is highlighted in the context of classification. Yet, these assumptions are also restrictive and stand in contradiction with the non-stationarity of financial data. Challenges preventing the straightforward application of machine learning techniques to financial series are thus carefully identified. Possible extensions of the classic learning theory, which would allow us to explore and to compute predictions on financial data, are finally reviewed.

I.2.1 Introduction to Statistical Learning Theory.

According to [BBL04], "the goal of Machine Learning is to automate the process of data observation and of model construction and use. The goal of Learning Theory is to formalize it". Statistical Learning Theory studies thus three main learning problems, which are classification (or pattern recognition), regression estimation and density estimation. They can be seen as particular instances of the global problem of risk minimization [Vap95], which is formalized hereafter. We consider the measurable space $\mathcal{Z}$. We assume that $Z \in \mathcal{Z}$ is a random vector, which represents the data, and which is distributed according to the unknown distribution P. We observe the sample of independent identically distributed (iid) random variables $Z_{1:n} = (Z_1, \dots, Z_n)$ with distribution P. It is called the set of observations or the training set. Using these observations, we aim at constructing a numerical model $\hat{f}_n = \hat{f}_n(Z_1, \dots, Z_n)$, that we may also call a prediction or decision function, which yields accurate prediction or estimation results when applied to unseen observations (the test set). In that perspective, we introduce the following loss function Q and the risk R:

$$R(f) = \mathbb{E}(Q(Z, f)) = \int Q(z, f) dP(z).$$

We aim at finding a prediction function $\hat{f}_n$ whose risk $R(\hat{f}_n)$ is close to the optimal risk $R^\star = \inf_f R(f)$. Yet, as P is unknown, the risk $R(f)$ can not be directly computed. Learning algorithms consider instead an available and practical criterion, usually a surrogate of the following empirical risk [BBL04]:

$$R_n(f) = \frac{1}{n} \sum_{i=1}^{n} Q(Z_i, f).$$

The returned function $\hat{f}_n$ has thus to be of low empirical risk $R_n(\hat{f}_n)$ and must also generalize well, i.e. display a small risk difference $|R(\hat{f}_n) - R_n(\hat{f}_n)|$. These two conditions are not necessarily compatible. For instance, overfitting occurs when the empirical risk $R_n(\hat{f}_n)$ is much smaller than the true risk $R(\hat{f}_n)$.

We now consider the problem of classification. The random vector Z is then denoted $Z = (X, Y) \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $X = (X^1, \dots, X^p) \in \mathcal{X}$ is a random vector and where $Y \in \mathcal{Y}$ is a label. Labels are usually - though not always - binary, i.e. $\mathcal{Y} = \{-1, +1\}$. The loss function is the classification error:

$$Q((X, Y), f) = L(Y, f(X)) = \mathbb{1}_{\{f(X) \neq Y\}},$$

while the true risk and the empirical risk are expressed as follows.

$$R(f) = \mathbb{E}(\mathbb{1}_{\{f(X) \neq Y\}}) = \int \mathbb{1}_{\{f(x) \neq y\}} dP(x, y), \qquad R_n(f) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{f(X_i) \neq Y_i\}}.$$

Besides, it is common knowledge that the Bayes classifier defined as

$$f^\star(x) = \begin{cases} 1 & \text{if } \eta(x) > \frac{1}{2} \\ -1 & \text{otherwise,} \end{cases} \qquad \text{where } \eta(x) = P(Y = 1|X = x)$$

minimizes the true risk [DGL96]. The goal of classification is then to identify this optimal function. Several theoretical results, variants of the No Free Lunch theorems ([Wol96, WM97, Wol01]), show that overfitting can be prevented by restricting the set of functions in which the minimization is performed, and/or by replacing the risk to be minimized by a regularized criterion penalizing particularly too complicated candidate functions [BBL04]. We introduce therefore the set $\mathcal{F}$ of candidate functions and suppose for simplicity that there exists a unique function $f^\star_{\mathcal{F}} \in \mathcal{F}$ so that

$$f^\star_{\mathcal{F}} = \arg\min_{f \in \mathcal{F}} R(f).$$

To have a better idea of the role played by $\mathcal{F}$, we consider the classic error decomposition:

$$R(\hat{f}_n) - R^\star = (R(\hat{f}_n) - R(f^\star_{\mathcal{F}})) + (R(f^\star_{\mathcal{F}}) - R^\star),$$

where $\hat{f}_n \in \mathcal{F}$. The first term on the right hand side is called the estimation error, also referred to as the variance in statistics. The second term is called the approximation error, or the bias in statistical terms [vS11]. This decomposition turns the search for a good model $\hat{f}_n$ into a trade-off to find between the two errors. However, it does not give indications about a strategy to pursue.

Statistical learning theory focuses primarily on the estimation error [vS11]. A simple and natural strategy to minimize it is provided by the empirical risk minimization (ERM) approach. It consists in searching for a function minimizing the empirical risk $R_n(\hat{f}_n)$ in lieu of the true risk (as this latter quantity is not observable):

$$\hat{f}^{ERM}_n = \arg\min_{f \in \mathcal{F}} R_n(f).$$

The first performance guarantee of the ERM approach is its consistency, i.e. the fact that the minimum of the empirical risk $R_n(\hat{f}^{ERM}_n)$ converges to the minimum of the true risk $R(f^\star_{\mathcal{F}})$:

$$\mathbb{P}(|R(\hat{f}^{ERM}_n) - R(f^\star_{\mathcal{F}})| \geq \varepsilon) \underset{n\to\infty}{\longrightarrow} 0.$$

The study of the ERM consistency is deeply linked with the study of uniform deviations of empirical processes, as:

$$|R(\hat{f}_n) - R(f^\star_{\mathcal{F}})| \leq 2 \sup_{f \in \mathcal{F}} |R(f) - R_n(f)|,$$

which implies:

$$\mathbb{P}(|R(\hat{f}_n) - R(f^\star_{\mathcal{F}})| > \varepsilon) \leq \mathbb{P}(\sup_{f \in \mathcal{F}} |R(f) - R_n(f)| > \frac{\varepsilon}{2}).$$

We conclude by the uniform law of large numbers:

$$\mathbb{P}(\sup_{f \in \mathcal{F}} |R(f) - R_n(f)| > \frac{\varepsilon}{2}) \underset{n\to\infty}{\longrightarrow} 0.$$

The uniform convergence over the class of functions $\mathcal{F}$ is therefore a sufficient condition for consistency of the ERM approach over $\mathcal{F}$. Note that it is actually also a necessary condition [VC71, DGL96, vS11].

Now, consistency is an asymptotic property, which does not provide finite sample performance guarantees for the ERM approach. It is possible to produce generalization bounds, i.e. upper bounds for probabilities of uniform deviations, whatever the size of the class of candidate functions $\mathcal{F}$. If it is finite, i.e. $\mathcal{F} = \{f_1, \ldots, f_m\}$, we need the following deviation inequality.

Theorem 1. *(Hoeffding [Hoe63]): let $X_1, \dots, X_n$ be independent bounded random variables such that $X_i \in [a_i, b_i]$ a.s., where $a_i, b_i \in \mathbb{R}, \forall i$. Then $\forall \varepsilon > 0$, we have:*

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}(X)\right|\right) > \varepsilon) \leq 2\exp\left(-\frac{2\varepsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$

The derivation of the bound is then straightforward.

$$\begin{aligned}
\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) &\leq \sum_{i=1}^{m} \mathbb{P}(|R(f_i) - R_n(f_i)| > \varepsilon) \\
\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) &\leq 2m\exp(-2n\varepsilon^2),
\end{aligned}$$

where Hoeffding's inequality has been applied to the collection of variables $(\frac{1}{n}\mathbb{1}_{\{f_i(X_1)\neq Y_1\}}, \dots, \frac{1}{n}\mathbb{1}_{\{f_i(X_n)\neq Y_n\}})$.

In the infinite (and non-countable) case, the extension of such results requires the introduction of tools specific to the Vapnik-Chervonenkis theory, particularly of capacity measures [DGL96, BBL04, vS11]. Denote by $\mathcal{F}_{Z_{1:n}}$ the class of functions $\mathcal{F}$ projected on the finite sample $Z_{1:n} = (Z_1, \dots, Z_n)$, where $Z_i = (X_i, Y_i)$.

$$\mathcal{F}_{Z_{1:n}} = \{(f(X_1), \dots, f(X_n)), f \in \mathcal{F}\}.$$

The cardinal $|\mathcal{F}_{Z_{1:n}}|$ indicates the number of functions of $\mathcal{F}$ which can be distinguished from each other by their values on the finite sample $X_{1:n} = (X_1, \dots, X_n)$. We call the shattering coefficient $S_{\mathcal{F}}$ of the class of functions $\mathcal{F}$ the function defined by

$$S_{\mathcal{F}}(n) = \max_{X_{1:n}} |\mathcal{F}_{X_{1:n}}|.$$

It indicates the maximal number of ways to classify the sample $X_{1:n}$ with the class of functions $\mathcal{F}$. The shattering coefficient is a capacity measure of classes of functions [DGL96, BBL04, vS11]. It measures the richness of a class of functions via the different values they may take on the finite sample. It is the counterpart of the size of a class of functions in the finite case. Besides, if $S_{\mathcal{F}}(n) = 2^n$, we say that $\mathcal{F}$ shatters $X_{1:n}$.
The following important result uses the shattering coefficient to compute an upper bound of the probability of uniform deviation.

Theorem 2. *(Vapnik-Chervonenkis [VC71]): $\forall \varepsilon > 0$, we have:*

$$\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) \leq 4S_{\mathcal{F}}(2n)\exp\left(-\frac{n\varepsilon^2}{8}\right).$$

The proof of this inequality is based on the replacement of the true risk $R(f)$ by an estimate $R_n^{'}(f)$ computed on an independent sample, denoted $Z_{1:n}^{'} = (Z_1^{'}, \dots, Z_n^{'})$, where $Z_i^{'} = (X_i^{'}, Y_i^{'})$, and called the ghost sample [BBL04]. This intermediate result is referred to as the symmetrization lemma [BBL04].

Lemma 3. *("Symmetrization" [BBL04]): $\forall \varepsilon > 0$, such that $n\varepsilon^2 \geq 2$, we have:*

$$\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) \leq 2\mathbb{P}(\sup_{f\in\mathcal{F}} |R_n(f) - R_n^{'}(f)| > \frac{\varepsilon}{2}).$$

The following inequalities lead then to the Vapnik-Chervonenkis theorem:

$$\begin{aligned}
\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) &\leq 2\mathbb{P}(\sup_{f\in\mathcal{F}} |R_n(f) - R_n^{'}(f)| > \frac{\varepsilon}{2}) \\
&= 2\mathbb{P}(\sup_{f\in\mathcal{F}_{Z_{1:n},Z_{1:n}^{'}}} |R_n(f) - R_n^{'}(f)| > \frac{\varepsilon}{2}) \\
&\leq 2S_{\mathcal{F}}(2n)\mathbb{P}(|R_n(f) - R_n^{'}(f)| > \frac{\varepsilon}{2}) \\
&\leq 4S_{\mathcal{F}}(2n)\exp\left(-\frac{n\varepsilon^2}{8}\right),
\end{aligned}$$

the last inequality being obtained via a simple application of the Hoeffding's theorem.

A drawback of such a generalization bound is the difficulty of computing the shattering coefficient $S_{\mathcal{F}}(n)$ [vS11]. It can be overcome by introducing the Vapnik-Chervonenkis (VC) dimension:

$$VC(\mathcal{F}) = \max\{n \in \mathbb{N}, |\mathcal{F}_{Z_{1:n}}| = 2^n\}.$$

The next combinatorial result relates the VC dimension to the shattering coefficient:

Lemma 4. *(Vapnik-Chervonenkis [VC71], Sauer [Sau72], Shelah [She72]): let $\mathcal{F}$ be a class of functions with finite VC dimension d. Then, $\forall n \in \mathbb{N}$,*

$$S_{\mathcal{F}}(n) \leq \sum_{i=1}^{d} C_n^i.$$

In particular, $\forall n \geq d$,

$$S_{\mathcal{F}}(n) \leq \Big(\frac{en}{d}\Big)^d.$$

The shattering coefficients grow then polynomially with n, providing that the VC dimension of the class of functions $\mathcal{F}$ is finite. The finiteness of the VC dimension is thus a sufficient (and also necessary [VC71, VC81, DGL96]) condition for the consistency of the ERM approach.

Until now, we have obtained distribution-free generalization bounds. Yet, taking into account the underlying probability distribution may lead to substantially sharper bounds [BBL04, vS11]. In that perspective, we introduce the annealed VC entropy:

$$H_{\mathcal{F}}(n) = \log \mathbb{E}(|\mathcal{F}_{Z_{1:n}}|).$$

It enables the derivation of the following generalization bound.

$$\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) \leq 2\exp\Big(H_{\mathcal{F}}(2n) - \frac{n\varepsilon^2}{8}\Big).$$

The proof is based once again on the symmetrization lemma and also on the use of Rademacher variables. These latter are iid random variables, denoted $\sigma_1, \ldots, \sigma_n$, also independent from the samples $Z_{1:n}$ and $Z^{'}_{1:n}$. Besides, they check: $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = \frac{1}{2}$. We have then:

$$\begin{aligned}\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) &\leq 2\mathbb{P}(\sup_{f\in\mathcal{F}} |R_n(f) - R^{'}_n(f)| > \frac{\varepsilon}{2}) \\ &= 2\mathbb{P}(\sup_{f\in\mathcal{F}_{Z_{1:n}, Z^{'}_{1:n}}} |R_n(f) - R^{'}_n(f)| > \frac{\varepsilon}{2}) \\ &= 2\mathbb{E}\mathbb{P}_{\sigma}(\sup_{f\in\mathcal{F}_{Z_{1:n}, Z^{'}_{1:n}}} \frac{1}{n}|\sum_{i=1}^{n} \sigma_i(\mathbb{1}_{\{f(X_i)\neq Y_i\}} - \mathbb{1}_{\{f(X_i)\neq Y_i\}})| > \frac{\varepsilon}{2}).\end{aligned}$$

Indeed, the variables $\frac{1}{n}\sum_{i=1}^{n}(\mathbb{1}_{\{f(X_i)\neq Y_i\}} - \mathbb{1}_{\{f(X_i)\neq Y_i\}})$ and $\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\mathbb{1}_{\{f(X_i)\neq Y_i\}} - \mathbb{1}_{\{f(X_i)\neq Y_i\}})$ are identically distributed. Applying the Hoeffding's inequality yields the result.

$$\begin{aligned}\mathbb{P}(\sup_{f\in\mathcal{F}} |R(f) - R_n(f)| > \varepsilon) &\leq 2\mathbb{E}\Big(|\mathcal{F}_{Z_{1:n}, Z'_{1:n}}| \sup_{f} \mathbb{P}\Big(\frac{1}{n}|\sum_{i=1}^{n} \sigma_i(\mathbb{1}_{\{f(X_i)\neq Y_i\}} - \mathbb{1}_{\{f(X_i)\neq Y_i\}})| > \frac{\varepsilon}{2})\Big)\Big) \\ &\leq 2\mathbb{E}\Big(|\mathcal{F}_{Z_{1:n}, Z'_{1:n}}| \exp(-\frac{n\varepsilon^2}{8})\Big).\end{aligned}$$

Another capacity measure for classes of functions is based on the Rademacher variables introduced above and referred to as Rademacher averages [Men03, BBL04, BBL05]. The Rademacher average $\mathcal{R}(\mathcal{G})$ of a given class of function $\mathcal{G}$ is defined as

$$\mathcal{R}(\mathcal{G}) = \mathbb{E}\sup_{f\in\mathcal{G}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(Z_i),$$

while its empirical counterpart is

$$\mathcal{R}_n(\mathcal{G}) = \mathbb{E}_\sigma \sup_{f \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(Z_i).$$

We use them to get the next distribution-dependent generalization bounds in the classification context.

Theorem 5. *(Barteltt, Boucheron, Lugosi [BBL02], Koltchinskii, Panchenko [KP04]): $\forall \delta > 0$, with probability at least $1 - \delta$, we have, $\forall f \in \mathcal{F}$:*

$$R(f) \leq R_n(f) + \mathcal{R}(\mathcal{F}) + \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}.$$

Besides, with probability at least $1 - \delta$, $\forall f \in \mathcal{F}$,

$$R(f) \leq R_n(f) + \mathcal{R}_n(\mathcal{F}) + \sqrt{\frac{2\log(\frac{2}{\delta})}{n}}.$$

The proof is based on symmetrization and convexity arguments and also on the following concentration inequality.

Theorem 6. *(McDiarmid or Bounded Differences or Hoeffding-Azuma inequality [McD89]). Let $G : \mathcal{Z}^n \longrightarrow \mathbb{R}$ be a function of $Z = (Z_1, \ldots, Z_n)$ which satisfies $\forall i$, $\forall z_1, \ldots, z_n, z_i^{'} \in \mathcal{Z}$,*

$$\sup_{z_1,\ldots,z_n,z_i^{'}} |G(z_1, \ldots, z_i, \ldots, z_n) - G(z_1, \ldots, z_i^{'}, \ldots, z^n)| \geq c_i,$$

then $\forall \varepsilon > 0$,

$$\mathbb{P}(|G(Z) - \mathbb{E}(G(Z))| > \varepsilon) \exp\left(- \frac{2\varepsilon^2}{\sum_{i=1}^{n} c_i^2} \right).$$

Furthermore, these generalization bounds are tighter those obtained by the previous capacity measures. To see that, we need to introduce the concept of covering numbers. Given a distance d, we define the covering number $N(\mathcal{F}, \varepsilon, d)$ as the minimal number of open balls (with respect to the metric d) of radius $\varepsilon > 0$ needed to cover the class of functions $\mathcal{F}$ [Men03]. We have the following upper bound on empirical Rademacher averages.

Theorem 7. *(Dudley [Dud, Sud, vdVW96, Dud99]):*

$$\mathcal{R}_n(\mathcal{F}) \leq 12 \int_0^{+\infty} \sqrt{\frac{\log N(\mathcal{F}, \varepsilon, L^2(\mu_n))}{n}} d\varepsilon,$$

where the $L^2(\mu_n)$ norm is defined as $\|f\|^2_{L^2(\mu_n)} = \frac{1}{n} \sum_{i=1}^{n} f(X_i)^2$.

Besides, a tight VC dimension bound on covering numbers is provided by the next result.

Theorem 8. *(Haussler [Hau95]): let $\mathcal{F}$ be a class of functions with finite VC dimension d. Then, $\forall \varepsilon > 0$, $\forall X_1, \ldots, X_n \in \mathcal{X}$,*

$$N(\mathcal{F}, \varepsilon, L^2(\mu_n)) \leq Cd(\frac{4e}{\varepsilon})^d,$$

where C is a positive constant.

Combining these two last inequalities leads to

$$\mathcal{R}_n(\mathcal{F}) \leq C\sqrt{\frac{d}{n}},$$

which yields an improved generalization bound.

Many capacity measures can thus be used to bound the estimation error $(R(\hat{f}_n) - R(f^\star_{\mathcal{F}}))$. However, regarding the approximation error $(R(f^\star_{\mathcal{F}}) - R^\star)$, we have no guarantees that the optimal model $f^\star$ is contained in the set of functions $\mathcal{F}$. A possible solution is the Structural Risk Minimization (SRM) approach [VC74, Vap82], whose following version can be found in [DGL96].

Theorem 9. *(Structural Risk Minimisation (Vapnik-Chervonenkis [VC74], Vapnik [Vap82]): assume that $\mathcal{F}_1, \mathcal{F}_2, \ldots$ is sequence of classes of decision functions such that for any distribution P of (X, Y),*

$$\lim_{n \longrightarrow +\infty} R(f^\star_{\mathcal{F}_n})) = R^\star,$$

and that the VC dimensions $VC(\mathcal{F}_1), VC(\mathcal{F}_2), \ldots$ are all finite. If

$$k_n \longrightarrow +\infty, \qquad \text{and} \qquad \frac{VC(\mathcal{F}_{k_n} \log n)}{n} \underset{n \to \infty}{\longrightarrow} 0,$$

then the classifier $\hat{f}_n$ which minimizes the empirical risk over the class of functions $\mathcal{F}^\star_{k_n}$ is strongly consistent, i.e.

$$\lim_{n \longrightarrow +\infty} R(\hat{f}_n) = R^\star, \textit{with probability 1.}$$

In other words, the estimation and approximation errors of the classifier $\hat{f}_n$ converge to 0.

The theorem supposes that the approximation error converges to 0, but it does not propose a method for constructing the classes $(\mathcal{F}_n)_{1 \leq n < +\infty}$ checking this condition nor a method to compute the sequence $(k_n)_{1 \leq n < +\infty}$. Besides, minimizing the empirical risk can be computationally very difficult, as it is not "smooth enough" [BBL05, Vay06].

We consider now classifiers of the form

$$f(x) = 2\,\mathrm{sgn}(g(x)) - 1 = \begin{cases} 1 & \text{if } g(x) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

where $g : \mathcal{X} \longrightarrow \mathbb{R}$ is a real-valued function. We denote L the probability of error of g and L_n its empirical version:

$$L(g) = \mathbb{P}(\mathrm{sgn}(g(X)) \neq Y) = \mathbb{P}(f(X) \neq Y) = R(f)$$

and

$$L_n(g) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}(\mathbb{1}_{\mathrm{sgn}(g(X_i)) \neq Y_i}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}(\mathbb{1}_{f(X_i) \neq Y_i}) = R_n(f).$$

We introduce moreover the cost function $\phi : \mathbb{R} \longrightarrow \mathbb{R}_+$ such that $\phi(x) \geq \mathbb{1}_{]0,+\infty[}(x)$. The cost function is a surrogate loss function which allows us to introduce the following surrogate empirical risk as well:

$$A(g) = \mathbb{E}(-\phi(g(X)Y)) \qquad \text{and} \qquad A_n(g) = \frac{1}{n} \sum_{i=1}^{n} -\phi(g(X_i)Y_i).$$

Obviously, we have:

$$L(g) \leq A(g) \qquad \text{and} \qquad L_n(g) \leq A_n(g).$$

We have then all the ingredients to deduce, using the McDiarmid's inequality and arguments of convexity, a general upper bound on the classifier performance [BBL05].

Theorem 10. *("Margin-based performance bound" [BBL05]): assume that the class of functions $\mathcal{G}$ and the cost function ϕ check the following conditions:*

- *there exists $B > 0$ such that*

$$\sup_{g \in \mathcal{G}, (x,y) \in \mathcal{X} \times \mathcal{Y}} \phi(-g(x)y) < B.$$

- *the function ϕ is Lipschitz, i.e. there exists a constant L_ϕ such that*

$$|\phi(u) - \phi(v)| \leq |u - v|, \qquad \forall u, v \in \mathbb{R}.$$

Then with probability at least $1 - \delta$,

$$L(\hat{g}_n) \leq A(\hat{g}_n) \leq A_n(\hat{g}_n) + 2L_\phi \mathbb{E}(\mathcal{R}_n(\mathcal{G}(X_{1:n})) + B\sqrt{\frac{2\log\frac{1}{\delta}}{n}}.$$

The performance of the classifier $\hat{g}_n$ is thus controlled by the empiric Rademacher average of the class of functions $\mathcal{G}$.

This very general setting enables the introduction of two important types of classifier construction methods: the ensemble methods and the kernel methods. Both of them state explicit classes of candidate functions for classifiers.

For example, ensemble methods, such as Boosting or Bagging, choose classifiers in classes of the form:

$$\mathcal{G}_\lambda = \left\{ g(x) = \sum_{i=1}^{N} c_i b_i(x), N \in \mathbb{N}, \sum_{i=1}^{N} |c_i| \leq \lambda, b_1, \ldots, b_N \in \mathcal{B} \right\},$$

where $\mathcal{B}$ is called the class of base classifiers and contains only functions defined on $\mathcal{X}$ and taking values in $\{-1, +1\}$. Using subadditivity properties of Rademacher averages, Hoeffding's inequality and Sauer's lemma, we get the following upper bound on the empiric Rademacher average of the class $\mathcal{G}_\lambda$ [BBL05]:

$$\mathcal{R}_n(\mathcal{G}_\lambda(X_{1:n})) = \lambda \mathcal{R}_n(\mathcal{B}(X_{1:n})) \leq \lambda\sqrt{\frac{2VC(\mathcal{B})\log(n+1)}{n}}.$$

Pugging it into the result of the previous theorem implies that the probability of error $L(\hat{g}_n)$ of any function $\hat{g}_n \in \mathcal{G}_\lambda$ checks with probability at least $1 - \delta$:

$$L(\hat{g}_n) \leq A_n(\hat{g}_n) + 2L_\phi\sqrt{\frac{2VC(\mathcal{B})\log(n+1)}{n}} + B\sqrt{\frac{2\log\frac{1}{\delta}}{n}}.$$

Note particularly that the VC dimension of the base class $\mathcal{B}$, which appears in the right-hand side, is typically much smaller than the VC dimension of the class $\mathcal{G}_\lambda$ [BBL05]. This is therefore a strong result, which is tempered by the replacement of the empirical probability of error $L_n(\hat{g}_n)$ by the surrogate empirical risk $A_n(\hat{g}_n)$. Still, it is possible to bound $A_n(\hat{g}_n)$ by a quantity related to $L_n(\hat{g}_n)$, simply by choosing well the cost function ϕ. Fix thus $\gamma > 0$ and consider

$$\phi(x) = \begin{cases} 0 & \text{if } x \leq -\gamma \\ 1 & \text{if } x \geq 0 \\ 1 + \frac{x}{\gamma} & \text{otherwise} \end{cases}$$

Observe that $\phi(x) \geq \mathbb{1}_{]0,+\infty[}(x)$, $B = 1$ and $L_\phi = \frac{1}{\gamma}$. Hence, ϕ is a valid cost function. We have also $\phi(x) \leq \mathbb{1}_{]-\gamma,+\infty[}(x)$. This implies:

$$A_n(g) = \frac{1}{n}\sum_{i=1}^{n} -\phi(g(X_i)Y_i) \leq L_n^\gamma(g),$$

where $L_n^\gamma(g) = \sum_{i=1}^{n} \mathbb{1}_{]-\infty,\gamma[}(g(X_i)Y_i))$. Intuitively, $L_n^\gamma(g)$ counts the number of pairs which are either misclassified or well classified, but with a too small confidence, or "margin" γ [BBL05]. Finally, we have the margin-based performance bound: $\forall\gamma > 0$, with probability at least $1 - \delta$:

$$L(\hat{g}_n) \leq L_n^\gamma(\hat{g}_n) + 2\frac{\lambda}{\gamma}\sqrt{\frac{2VC(\mathcal{B})\log(n+1)}{n}} + B\sqrt{\frac{2\log\frac{1}{\delta}}{n}}.$$

Likewise, kernel methods such as Support Vector Machines aim at choosing classifiers in specific constrained classes of functions. They are of the form:

$$\mathcal{G}_\lambda = \left\{ g(x) = \sum_{i=1}^{N} c_i K(x_i, x), N \in \mathbb{N}, \sum_{i,j=1}^{N} c_i c_j K(x_i, x_j) \leq \lambda^2, x_1, \dots, x_N \in \mathcal{X} \right\},$$

where $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is a positive definite kernel function. It is again possible to derive a margin-based performance bound using similar arguments and properties of Kernel Hilbert spaces. We state it without proof: with probability at least $1 - \delta$:

$$L(\hat{g}_n) \leq L_n^\gamma(\hat{g}_n) + 2\frac{\lambda}{\gamma n}\sqrt{\sum_{i=1}^{n} k(X_i, X_i)} + \sqrt{\frac{2\log\frac{2}{\delta}}{n}}.$$

We make now additional assumptions on the cost function ϕ. Like in [Vay06], we suppose that $\phi : \mathbb{R} \longrightarrow \mathbb{R}_+$ is strictly convex, differentiable, strictly increasing with $\phi(0) = 1$, $\lim_{x \longrightarrow -\infty} \phi(x) = 0$. Indeed, as we will see, minimizing the convex risk $A(g) = \mathbb{E}(\phi(-Y g(X)))$ is not only computationally interesting, but also benefits from useful theoretical properties. First, it leads to an optimal function $g^\star$ which is related to the Bayes classifier $f^\star(x) = 2\mathbb{1}_{\eta(x) > \frac{1}{2}} - 1$ such that $\text{sgn}(g^\star) = f^\star$. The proof is based on convexity arguments [BBL05, Vay06, LV04]. Thus, for the following usual cost functions, the corresponding optimal functions are easily computed:

- the exponential cost $\phi(x) = \exp(x)$, used in the procedure AdaBoost, yields the optimal function $g^\star(x) = \frac{1}{2}\log(\frac{\eta(x)}{1-\eta(x)})$.
- the logit cost $\phi(x) = \log_2(1 + \exp(x))$ is associated to $g^\star(x) = \log(\frac{\eta(x)}{1-\eta(x)})$.
- the hinge loss $\phi(x) = (1 + x)_+$, used in Support Vector Machines, directly induces the Bayes classifier $f^\star$. Note however that this latter cost function does not satisfy the aforementioned assumptions as it is not strictly convex nor differentiable.

In case of the exponential and logit costs, under specific technical assumptions [Zha04, LV04], convexity arguments lead to the inequality valid for $g_n \mathcal{G}_\lambda$,

$$L(g_n) - L^\star \leq 2\sqrt{2}(A(g_n) - A^\star)^{\frac{1}{2}}.$$

Therefore, using previous results, we get:

$$\begin{aligned} L(g_n) - L^\star &\leq 2\sqrt{2}(A(g_n) - \inf_{g \in \mathcal{G}_\lambda} A(g))^{\frac{1}{2}} + 2\sqrt{2}(\inf_{g \in \mathcal{G}_\lambda} A(g) - A^\star)^{\frac{1}{2}} \\ &\leq 4\sqrt{2}\left(2L_\phi \lambda \sqrt{\frac{2VC(\mathcal{B})\log(n+1)}{n}} + B\sqrt{\frac{2\log\frac{1}{\delta}}{n}}\right)^{\frac{1}{2}} + 2\sqrt{2}(\inf_{g \in \mathcal{G}_\lambda} A(g) - A^\star)^{\frac{1}{2}}, \end{aligned}$$

with probability at least $1 - \delta$. Note that the last term of the right-hand side is the approximation error, which equals 0 if λ is chosen sufficiently large. It is difficult to describe further and more precisely this error as the approximation properties of the class $\mathcal{G}_\lambda$ are not yet well understood. Note that a similar result can be obtained for the hinge loss [BBL05]. Further informations concerning the approximation properties of kernel classes and the universal consistency of Support Vector Machines can be found in [Ste01, Ste02, Ste05].

Other frameworks for deriving tight generalization bounds have been developed and are under current investigation. The Probably Approximately Correct (PAC) - Bayesian analysis of statistical learning is one of them. In this approach, in the classification context, the set of candidate functions $\mathcal{F}$ is randomized. We denote π the (fixed) prior distribution associated to the set $\mathcal{F}$. We aim at choosing from the data a posterior distribution $\hat{\rho}_n$ over the set $\mathcal{F}$. The true risk and the empirical risks are thus denoted:

$$\mathbb{E}_\rho(R(f)) = \int R(f)d\rho(f) \qquad \text{and} \qquad \mathbb{E}_\rho(R_n(f)) = \int R_n(f)d\rho(f),$$

where ρ stands for any posterior distribution.
The following PAC-Bayesian bound, historically first expressed in [McA99], states that with probability at least $1-\delta$, for any posterior distribution ρ, we have:

$$\mathbb{E}_\rho(R(f)) - \mathbb{E}_\rho(R_n(f)) \leq \sqrt{\frac{K(\rho,\pi) + \log(2n) + \log(\frac{1}{\delta})}{2n-1}},$$

where $K(\rho,\pi)$ is the Kullback-Leibler divergence between the distributions ρ and π: $K(\rho,\pi) = \mathbb{E}_\rho(\log(\frac{\rho}{\pi}))$ if ρ admits a density with respect to π, denoted $\frac{\rho}{\pi}$, and $K(\rho,\pi) = +\infty$ otherwise. The proof of this bound is based on convexity inequalities, properties of the Kullback-Leibler divergence and on the Hoeffding's inequality [LSM01, See03].
A similar proof leads to the following tighter bound [LSM01, See03]:

$$K(\mathbb{E}_\rho(R(f)), \mathbb{E}_\rho(R_n(f))) \leq \frac{K(\rho,\pi) + \log(\frac{2\sqrt{n}}{\delta})}{n},$$

where, with a slight abuse of notation, $K(\mathbb{E}_\rho(R(f)), \mathbb{E}_\rho(R_n(f)))$ denotes the Kullback- Leibler divergence between the Bernoulli distributions of respective parameters $\mathbb{E}_\rho(R(f))$ and $\mathbb{E}_\rho(R_n(f))$. Note that other bounds and further explanations can be found in [Cat03, Cat07, Zha06b, Zha06a, Aud].

This brief and partial summary of statistical learning theory highlights that algorithms for constructing classifiers (in the binary setup) exist and are reliable under specific assumptions. Particularly, proofs of consistency and rates of convergence for generalization bounds are based on sophisticated probabilistic tools, mainly concentration inequalities (applied to empirical processes in a functional space) and symmetrization techniques. Classically formulated, these methods and results require the assumption that the observations are independent and identically distributed. In the next subsection, we review the challenges we are confronted with when applying learning methods to financial data, which are not iid.

I.2.2 Challenges and Extensions of the Classic Learning Framework

The non-iid case is a burning issue in statistical learning theory. Particularly, learning with stationary (or identically distributed) dependent series has attracted increasing interest. It is very popular to study such time series under different types of asymptotic independence conditions, called mixing conditions [MR10, MSS11a], introduced hereafter. Let $Z_{1:\infty} = (Z_t)_{1 \leq t < +\infty}$ be a stationary process. Let

$$\sigma_1^n = \sigma(Z_1, \dots, Z_n)$$

be the sigma-algebra generated by the first n random variables $Z_1, \dots, Z_n$, and

$$\sigma_n^\infty = \sigma(Z_n, \dots)$$

the sigma-algebra generated by the infinite set of random variables $(Z_t)_{t \leq n}$. The dependence between past and future events defined with such algebras can de quantified with the following coefficients

α, β, ϕ - expressed as in [Yu94]:

$$\begin{aligned}
\alpha(k) &= \sup_{\substack{n\in\mathbb{N} \\ A\in\sigma_1^n \\ B\in\sigma_{k+n}^\infty}} |P(A\cap B) - P(A)P(B)| \\
\beta(k) &= \sup_{\substack{n\in\mathbb{N} \\ \{A_i\}\text{ partition in }\sigma_1^n \\ \{B_j\}\text{ partition in }\sigma_{k+n}^\infty}} \frac{1}{2}\sum_{i=1}^{I}\sum_{j=1}^{J} |P(A_i\cap B_j) - P(A_i)P(B_j)| \\
\phi(k) &= \sup_{\substack{n\in\mathbb{N} \\ A\in\sigma_1^n \\ B\in\sigma_{k+n}^\infty}} |P(A|B) - P(A)|.
\end{aligned}$$

We say that the process $Z_{1:\infty}$ is:

- α-mixing or strongly mixing if
$$\alpha(k) \underset{k\to\infty}{\longrightarrow} 0.$$
- β-mixing or absolutely regular if
$$\beta(k) \underset{k\to\infty}{\longrightarrow} 0.$$
- ϕ-mixing if
$$\phi(k) \underset{k\to\infty}{\longrightarrow} 0.$$

As noted in [Bra05, Dou94], $\alpha(k) \leq \beta(k) \leq \phi(k), \forall k \in \mathbb{N}$, which means that phi-mixing implies beta-mixing, which in turn implies alpha-mixing.

According to [Vid03], β-mixing is "just the right" assumption for deriving generalization bounds in the non- iid case. To see why, suppose that the process $Z_{1:\infty}$ is stationary and β-mixing. The celebrated following technique [Yu94, Ber27] enables the construction of new process $\Xi_{1:\infty}$ made of independent blocks, which have the same distribution as finite sequences of $Z_{1:\infty}$. As in [Yu94], we consider the sample $Z_{1:n} = (Z_1, \ldots, Z_n)$ and define $m_n \in \mathbb{N}$ and $\mu_n \in \mathbb{N}$ such that $2m_n\mu_n = n$. We divide then the sample $Z_{1:n}$ into $2\mu_n$ blocks of each length m_n as follows:

$$\begin{aligned}
Z_{2(u-1)m_n+1:(2u-1)m_n} &= (Z_t)_{2(u-1)m_n+1\leq t\leq (2u-1)m_n} \\
Z_{(2u-1)m_n+1:2um_n} &= (Z_t)_{(2u-1)m_n+1\leq t\leq 2um_n}
\end{aligned}$$

where $1 \leq u \leq \mu_n$. Focusing only on the sequence of blocks $\mathcal{Z}_{m_n} = (Z_{2(u-1)m_n+1:(2u-1)m_n})_{1\leq u\leq \mu_n}$, we define the corresponding sequence of iid blocks $\Xi_{m_n} = (\Xi_{2(u-1)m_n+1:(2u-1)m_n})_{1\leq u\leq\mu_n}$ such that the sequence Ξ_{m_n} is independent of $Z_{1:n}$ but such that each block $\Xi_{2(u-1)m_n+1:(2u-1)m_n}$ has the same distribution as the block $Z_{2(u-1)m_n+1:(2u-1)m_n}$ from the original sequence:

$$\Xi_{2(u-1)m_n+1:(2u-1)m_n} \overset{\mathcal{L}}{=} Z_{2(u-1)m_n+1:(2u-1)m_n} \overset{\mathcal{L}}{=} Z_{1:m_n}.$$

Because of the mixing assumption on $Z_{1:\infty}$, the dependence between the blocks $Z_{2(u-1)m_n+1:(2u-1)m_n}$ decreases as the lag m_n separating the blocks increases. Hence the distribution of the two sequences $\mathcal{Z}_{m_n}$ and Ξ_{m_n} become closer and closer to each other. This is highlighted by the following lemma.

Lemma 11. *(Lemma 4.1 in [Yu94]): for any measurable function ϕ with respect to the process $Z_{1:\infty}$ uniformly bounded by M,*

$$|\mathbb{E}(\phi) - \tilde{\mathbb{E}}(\phi)| \leq M(\mu_n - 1)\beta(m_n) \leq M(\mu_n - 1)\phi(m_n),$$

where the first expectation $\mathbb{E}(\phi)$ is computed with the distribution of $Z_{1:\infty}$, while the second one $\tilde{\mathbb{E}}(\phi)$ is computed using the distribution of $\Xi_{1:\infty}$.

Used in conjunction with the independent blocks construction technique, this lemma is the key to apply methods for deriving generalization bounds in the iid case - such as symmetrization techniques and concentration inequalities - to the β-mixing case (and also to the ϕ-mixing case) [Yu94, KV09, MR10, MSS11a]. For example, (non- parametric) generalization bounds are obtained in [Mei00] via an extension of the method of the structural risk minimization approach suggested by Vapnik to the β-mixing case. In this case again, the consistency of regularized Boosting methods is proven in [LKS06]. Besides, Rademacher complexity-based bounds are derived in [MR08], while stability-based bounds are derived in [MR10] for β-mixing and ϕ-mixing sequences. These latter bounds are shown to be useful to analyze the properties of various kernel regularization-based and relative entropy-based regularization algorithms (such as Support Vector Regression, Kernel Ridge Regression, and Support Vector Machines), when used with mixing inputs. Finally, PAC-Bayes generalization bounds for classifiers and scoring functions are obtained in [RSS10] using an approach exploiting the notion of fractional covers of graphs [SU97]. This approach enables the representation of data dependencies with a graph and the splitting of a process into subsets of independent random variables via a decomposition of this graph. It applies then usual iid PAC-Bayes bounds in the β-mixing case. Besides, the problem of making predictions at a fixed horizon $h \in \mathbb{N}$ on β- mixing time series is studied in [MSS11b] from a learning point of view. The goal is still to find a prediction function $\widehat{f}_n = \widehat{f}_n(Z_1, \ldots, Z_n)$, although in [MSS11b] both classification and regression problems are considered, i.e. $\widehat{f}_n$ can be binary or real-valued. The distinction between past and future events appears clearly in the expressions of the true and of the empirical risks:

$$R(f) = \mathbb{E}(Q(\tau_h(Z), f) \qquad \text{and} \qquad R_n(f) = \frac{1}{n}\sum_{t=1}^{n} Q(Z_{t+h}, f).$$

where $\tau_h(Z) = (Z_{t+h})_{t\geq 1}$. Generalization error bounds are provided for stationary univariate autoregressive (AR) models. The proof is based first on generalization bounds obtained in [MR08] for stationary β-mixing sequences. These bounds rely on Gaussian complexity, which is similar to Rademacher complexity, except that the random variables $\sigma_1, \ldots, \sigma_n$ are no longer binary and uniformly distributed but real-valued and Gaussian [BM03]. Secondly, it is shown that the Gaussian complexity of a specific order AR model can be relevantly bounded. A strategy based on the Structural Risk Minimization approach is then proposed to select the best model, and is illustrated by predicting interest rate movements. This work builds a very interesting bridge between statistical learning theory and econometric models. A practical limitation is that the data structure dependence is usually unknown, which is an obstacle to the computation of the generalization bounds. Actually, this limitation concerns in most extensions of the statistical learning theory to the β-mixing case (in all the aforementioned ones at least) and motivates the method for estimating β-mixing rates (from data) presented in [MSS11a].

Now, there are interesting points of view apart β-mixing. For instance, the consistency of Support Vector Machines is proven in [SHS09] for processes satisfying a certain weak law of large numbers, incidentally for α-mixing (non-stationary) processes, generalizing results obtained in [Ste02, Ste05]. Besides, the consistency of Support Vector Machines using Gaussian Radial Basis Function (RBF) kernels is also proven for a class of processes exhibiting a sufficiently fast decay of correlations - a rather general class of dynamical process. The approach is based on the use of bounds on the sequence of correlations, on the properties of Gaussian RBF kernels, and on a specific concentration inequality involving the sequence of correlations. In [MSS11c], stationary and dependent time series models are studied. Rademacher complexity based generalization bounds for such models are derived. The approach is based on concentration inequalities for dependent data due to Van de Geer and on a symmetrization technique close to those developed in [RST10b, RST10a] for online learning.

There is thus theoretical support for extending the standard statistical learning theory to cases where data are dependent and stationary. Indeed, kernel and ensemble methods have been extensively and successfully applied to the problem of predicting movement of financial series [TC01, VGSB$^+$01, CG02, TC02, jK03, HW06a, HW06b, USC06, CM08, HW08, YSC08, LD08, FRD09, FHST10, WZ10, HW10, Hua11]. Besides, very recent works show how the tools and techniques of statistical learning theory can be used to analyze econometric models and online learning [RST10b, RST10a]. Such connections also open perspectives to learn with non-stationarities. Particularly, the sub-field of online learning called prediction of individual sequences [CBL06, Sto05, Sto11] proposes a relevant framework for dealing with non-stationary time series [SJC11]. It investigates the problem of sequentially computing a forecast $\hat{z}_t = \hat{z}_t(z_1, \ldots, z_{t-1}) \in \mathcal{Z}_f$ given the past observations $z_1, z_2, \ldots, z_{t-1} \in \mathcal{Z}$. Unlike in statistical learning theory, no assumption is made on the mechanism generating these data samples, which is not supposed to be a stochastic process. The notion of risk is therefore not defined and the constructions specific to statistical learning theory that lead to a prediction function of low risk are here not applicable. Instead, the performance of a forecasting strategy S is assessed via a cumulative loss, computed iteratively using a loss function $l : \mathcal{Z}_f \times \mathcal{Z} \longrightarrow \mathbb{R}$ during n rounds of prediction, and a finite class of experts or forecasters $f_{k,t} = f_{k,t}(z_1, \ldots, z_{t-1}) \in \mathcal{Z}_f$, where $1 \leq k \leq q$. The goal is to ensure that the cumulative loss is close to that of the best forecaster in the class. In other words, we want a forecasting strategy of low regret, where regret refers to the difference between the two cumulative losses. A natural setting of prediction of individual sequences is to study forecasting strategies based on weighted averages of experts:

$$\hat{z}_t = \sum_{k=1}^{q} \hat{w}_{k,t} f_{k,t},$$

where

$$\hat{w}_{k,t} \geq 0 \qquad \forall k \in \{1, \ldots, q\}, \qquad \text{and} \sum_{k=1}^{q} \hat{w}_{k,t} = 1.$$

Therefore, using notations of [Sto11], we define the cumulative losses of the strategy S and of the strategy associated to w as:

$$\begin{aligned} \hat{L}_n(S) &= \sum_{t=1}^{n} l(\hat{z}_t, z_t) = \sum_{t=1}^{n} l\Big(\sum_{k=1}^{q} \hat{w}_{k,t} f_{k,t}, z_t\Big), \\ L_n(w) &= \sum_{t=1}^{n} l\Big(\sum_{k=1}^{q} w_{k,t} f_{k,t}, z_t\Big). \end{aligned}$$

The regret to minimize is:

$$R_n(S) = \hat{L}_n(S) - \inf_{w} L_n(w).$$

In this setting, forecasting strategies are determined by the iterative computation of $\hat{w}_t$. We aim therefore at finding relevant ways of updating $\hat{w}_t$, which ensure tight bounds on the regret $R_n(S)$. Such theoretical guarantees are particularly provided by strategies based on exponentially-weighted average of experts [Vov90, CBL06, Sto05]. In practice, they are employed to model stationary ergodic times series [OW10] as well as non- stationary ones. For instance, they can be used to adapt to stationarity changes [SJC11, HS09], putting increasingly more weight on new patterns (i.e. on the best up-to-date forecasters) and less on the old ones, which may be not as efficient or even counterproductive. It is thus not surprising that the concepts of prediction of individual sequences are very popular to tackle the problem of designing investment strategies [Cov91, HSSW98, BEYG00, SL05, Gyö06b, AHKS06].

Complex time series can also be modelled using Dynamic Bayesian Networks (DBNs) [Gha97, FMR98, Mur02], which are extensions of Bayesian Networks (BNs) to the temporal setting. A Bayesian Network (G, Θ) is a probabilistic graphical model which uses the Directed Acyclic Graph (DAG) $G = (V, E)$ (refer to the appendix for a definition of DAGs) to represent the joint probability distribution of the finite set of random variables $V = (Z^1, \ldots, Z^p)$. Directed edges $e = Z^k Z^l \in E$ indicate a causal dependence of the child node $Z^l \in De(Z^k)$ toward its parent $Z^k \in Pa(Z^l)$, where $De(Z^k)$ is the set of descendents of Z^k and $Pa(Z^l)$ the set of parents of Z^l. Besides, a Bayesian

Network satisfies the local Markov property, which states that each variable Z^k is independent of its non-descendents given its parents in G [Whi90, Lau96, Pea00, KF09]:

$$Z^k \perp Z^L | Pa(Z^k), \quad \forall Z^L \subset V \setminus De(Z^k).$$

Note that we have the inclusion $Pa(Z^k) \subset V \setminus De(Z^k)$ because the graph G is acyclic. Additionally, using these conditional independences and the Bayes rule (hence the name Bayesian Networks), the joint probability distribution of $(Z^1, \ldots, Z^p)$ can be factorized as follows:

$$P(Z^1, \ldots, Z^p) = \prod_{k=1}^{p} P(Z^k | Pa(Z^k)),$$

where $P(Z^k|Pa(Z^k) \in \Theta$, which is the set of Θ of parametrized conditional distributions. This factorized form is particularly helpful for inference, i.e. to compute the conditional probability $P(Z^K|Z^L)$ of a set of query variables Z^K given a set of observed variables Z^L. The variable elimination algorithm finds the factorized form of this conditional probability using all variables Z^M connected and disjoint to Z^K and Z^L. It then sequentially considers each variable $Z^m \in Z^M$, collects each factor in $P(Z^K|Z^L)$ that includes Z^m, and creates a new factor by summing out Z^m. A drawback of this procedure is that it is performed given a query. Hence it is not computationally efficient to compute a large number of marginals. Instead, the junction tree algorithms are based on the construction of a junction tree, which is a maximum spanning tree connecting clusters of nodes. It satisfies specific properties which ensure, via message passing strategies, the computation of all marginals [Lau96, CLDS99]. Note that these two methods perform exact inference. In case of massive data sets, it can be necessary to use instead approximate inference methods, such as variational methods [JGJS99], Monte Carlo sampling and Markov chain Monte Carlo (MCMC) methods (particularly the Gibbs sampling and the MetropolisŨHastings algorithm) [Mac98], loopy belief propagation [Pea88, Wei00], etc. In practice, the Bayesian Network is not known. Its structure as well as its parameters have to be learnt, the first task being much harder than the second one [Mur98]). Dealing with missing values and with hidden nodes is also a problem. That is why authors usually distinguish between four cases, regarding whether the structure is known or unknown and the observability is full or partial [Mur98]. In case of known structure and full observability, a natural solution is to maximise the log-likelihood $L(\Theta|Z_{1:n})$ of the training set $Z_{1:n} = \Big((Z_1^1, \ldots, Z_1^p), \ldots, (Z_n^1, \ldots, Z_n^p)\Big)$:

$$L(\Theta|Z_{1:n}) = \sum_{i=1}^{n} \sum_{k=1}^{p} \log P(Z_i^k | Pa(Z_i^k)).$$

If the structure is known and the observability partial, a possible strategy consists in finding a (locally) optimal maximum likelihood estimate of the distribution parameters via the Expectation Maximization (EM) algorithm [Mur98, BG07, GY06a]. In the two last cases, structure learning may be achieved as explained in [Sch10, Scu10] by search and score methods. They perform model selection in the space of DAGs via the optimization of a given criterion, usually the BIC score [HGC95]. Other possible strategies for structure learning are constraint-based methods, which aim first at pruning DAGs edges and second at determining the directions of the remaining edges [VP90, SG, Scu10]. Finally, hybrid methods, which combine both search and score and constraint-based techniques, seem to be the most popular strategies [FNP99, SNMM07, Sch10].
Consider now the sequence of random vectors $Z_{1:\infty} = (Z_t)_{1 \leq t < +\infty}$. A Dynamic Bayesian Network is defined as the pair of Bayesian Networks $(B_1, B_{\rightarrow})$ [Mur02]. The BN B_1 particularly specifies the prior distribution $P(Z_1)$, while $B_{\rightarrow}$ introduces the directed acyclic graph model formed by the variables $Z_t^1, \ldots, Z_t^p$ and by their respective parents $Pa(Z_t^1), \ldots, Pa(Z_t^p)$. It also defines the conditional probability $P(Z_t|Z_{t-1})$ as follows:

$$P(Z_t|Z_{t-1}) = \prod_{k=1}^{p} P(Z_t^k | Pa(Z_t^k)).$$

Note that $B_{\rightarrow}$ is therefore a two-slice temporal Bayesian Network (2TBN). The nodes in the first slice are not parametrized. On the other hand, each node in the second slice is associated to a conditional probability distribution, which specifies $P(Z_t^k|Pa(Z_t^k))$, $\forall t > 1$ [Mur02]. The joint distribution of the DBN is computed by unrolling the 2TBN to get the n time slices:

$$P(Z_1^n) = \prod_{t=1}^{n} \prod_{k=1}^{p} P(Z_t^k|Pa(Z_t^k)).$$

The DBN structure is rich and flexible. State-Space Models (SSMs), including for example Hidden Markov Models (HMMs) [Rab89, CMR05] and Kalman Filter Models (KFMs) [RG99], and many other sophisticated dynamic models are particular instances of DBNs [Mur02]. The first interest of a DBN is that it provides a compact representation of temporal causal links via the directed acyclic graph of its 2TBN. For example, encoding factorial HMMs as DBNs requires exponentially fewer parameters [Mur02]. Additionally, the DBN structure can be used to solve (more) efficiently inference problems, i.e. to compute marginals of the type $P(Z_t^k|Z_{1:\tau})$ with algorithms of low complexity (these problems being referred to as prediction if $\tau < t$, filtering if $\tau = t$ and smoothing if $\tau > t$). It is shown in [Mur02] that exact inference for DBNs can be quickly executed via backward and forward passes implementing the junction tree algorithm. Besides, approximate inference can be performed offline using standard methods, such as loopy belief propagation (which includes the Boyen-Koller algorithm [BK98] and the Factored Frontier algorithm [MW01]) or MCMC methods, as well as online by plugging DBNs in particle filtering algorithms [Mur02]. Finally, the methods for learning DBS are essentially those which have been developed for learning BNs and graphical models [Mur02, Sch10]. Last but not least, DBNs, in their original form, are supposed to be stationary. Their parameters and their structure remain unchanged through time. However, non-stationary effects can be modelled by incorporating in DBNs the ability for parameter change [BNI98, PRM00, GH00, PHW01, MPR05, IHS06, ORBD08, FSJW08, GKE10] and/or structure change [PADF02, TL04, Fea06, XM07, GHFX07, WZSS08, RH08, GH09, SKX09, KSX09, KSAX09, FSX09, WKY$^+$11]. DBNs, mostly under the form of KFMs and HMMs, have been widely used in finance [BH, ME07] and econometrics.

Non-stationarities can also be studied in the framework of transfer learning. Indeed, in this sub-field of machine learning, the training data and the test data are not supposed to be identically distributed nor even to belong to the same feature space [PY10]. Transfer learning focuses then on knowledge transfer from the training set to the test set. This transfer can be considered under different settings and completed via various approaches. These distinctions are made explicit hereafter using the concepts of domain and of task introduced in [PY10] in the context of supervised learning. The domain $\mathcal{D} = \{\mathcal{X}, P_{X_{1:n}}\}$ consists thus of the space $\mathcal{X}$ and of the distribution $P_{X_{1:n}}$ of the random variables $X_{1:n} = (X_1, \ldots, X_n)$. The task $\mathcal{T} = \{\mathcal{Y}, P_{Y_{1:n}|X_{1:n}}\}$ is made of the space $\mathcal{Y}$ and of the conditional distribution $P_{Y_{1:n}|X_{1:n}}$ of the labels $Y_{1:n} = (Y_1, \ldots, Y_n)$ given $X_{1:n}$. The training - or source - set $\Big((X_{S_1}, Y_{S_1}), \ldots, (X_{S_{n_S}}, Y_{S_{n_S}})\Big)$ is thus composed of the corresponding source domain $\mathcal{D}_S$ and of the learning task $\mathcal{T}_S$, while the test - or target - set $\Big((X_{T_1}, Y_{T_1}), \ldots, (X_{T_{n_T}}, Y_{T_{n_T}})\Big)$ corresponds to the source domain $\mathcal{D}_T$ and to the learning task $\mathcal{T}_T$. The following proper definition of transfer learning is proposed in [PY10].

Définition 12. *(Definition 1(transfer learning) [PY10]): transfer learning aims at learning a function f_T from the training set $\mathcal{D}_S$ and $\mathcal{T}_S$, which yields a low prediction error on the test set $\mathcal{D}_T$ and $\mathcal{T}_T$.*

This definition enables the distinction between different types of transfer learning. For instance, in the setting of inductive transfer learning, the target learning task $\mathcal{T}_T$ is supposed to be different, but related, to the source learning task $\mathcal{T}_S$.

Définition 13. *(Definition 2(inductive transfer learning) [PY10]): inductive transfer learning is transfer learning with condition $\mathcal{T}_S \neq \mathcal{T}_T$.*

This type of transfer learning is closely related to multi-task learning which aims at learning the knowledge common to multiple tasks in order to improve the learning of each specific task [Thr96, Car97, Bax00, BDS03]. On the other hand, in the setting of transductive transfer learning, the source and the target tasks are supposed to be the same, whereas the source and the target domains are assumed to be different.

Définition 14. *(Definition 3(transductive transfer learning) [PY10]): transductive transfer learning is transfer learning with conditions $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$.*

The condition $\mathcal{D}_S \neq \mathcal{D}_T$ can indicate that the spaces $\mathcal{X}_S$ and $\mathcal{X}_T$ are different or that $\mathcal{X}_S = \mathcal{X}_T$, while the distribution of the training and of the test sets are different in the sense that $P_{X_{1:S_n}} \neq P_{X_{1:T_n}}$. This latter case particularly includes domain adaptation [IM06, BMP06, BDBCP07], where the spaces $\mathcal{X}_S$ and $\mathcal{X}_T$ are identical and where the training data $\Big((X_{S_1}, Y_{S_1}), \dots, (X_{S_{n_S}}, Y_{S_{n_S}})\Big)$ is made of two particular subsets. The distribution of the first - usually large - subset is different but related to the distribution of the test set $\Big((X_{T_1}, Y_{T_1}), \dots, (X_{T_{n_T}}, Y_{T_{n_T}})\Big)$, while the second - usually much smaller - subset has the same distribution as the test set. When there is no second subset, we speak of learning under sample selection bias [Hec79, Zad04] or covariate shift [Shi00]. Semi-supervised learning also manages two training subsets, where the first one, often large, is unlabeled, while the second one is small and labeled. The idea is that the unlabeled training data can be used to improve the learning of the prediction function [Zhu05]. Finally, the last type of transfer learning is unsupervised transfer learning, naturally defined as follows.

Définition 15. *(Definition 4(unsupervised transfer learning) [PY10]): unsupervised transfer learning is transfer learning with condition $\mathcal{T}_S \neq \mathcal{T}_T$, while $(Y_{S_1}, \dots, Y_{S_{n_S}})$ and $(Y_{T_1}, \dots, Y_{T_{n_T}})$ are not observable.*

Besides, an important aspect of transfer learning problems is to determine the form of the knowledge to transfer. For instance, only a limited part of the training set can be relevant to learn the test set distribution, the other part of the training set being useless or even harmful. A solution is then to develop strategies for identifying this relevant part of the training set and increasing its influence in the learning process [DYXY07, JZ07, HSG$^+$07, BBS07, QCSSL09]. Another example of problem is to find the most appropriate parametric model based representation [LP04, EP04, BCW08] or feature based representation [Jeb04, BMP06, BDP07, AMPY07, AEP08] of data in order to enhance the similarity between the two data sets and to improve the learning performances. Note that these different approaches are not specific to a particular setting.

We consider transfer learning as a relevant framework for developing exploratory analyses of non-iid time series. It is indeed well designed to determine the conditions, limits and opportunities of any type of knowledge transfer from a given training set to a specific test set. For instance, we believe that it can help to prevent overfitting, which occurs when a too sophisticated numerical model is unable to generalize. Likewise, it can instigate methods for controlling data snooping, which refers to the bias induced by a systematic (re)use of the same training data for inference or model selection and which can lead to artificially good modelling results [LM90, STW99, Whi00, CG06]. These two problems, typical of model construction approaches with non-stationary data, can be seen as misleading knowledge transfer attempts, possibly of too complex or misspecified prediction functions. In order to avoid such pitfalls and to develop an efficient transfer learning approach, we present a specific methodology based on model automatic construction, backtesting and backtesting interpretation in the next section.

I.3 Research Methodology

The non-stationarity and time dependence of financial series are the main obstacles to a straightforward application of machine learning techniques to market analysis. Indeed, the classic statistical learning theory has been developed in the iid case. Although there exist many relevant extensions to various cases where the data are identically distributed and somehow time dependent, the case where data are non-stationary and time dependent is still difficult to address. Practical applications can thus suffer from problems due to overfitting and data snooping. We have developed therefore a stepwise research methodology for constructing reliably trend predictive numerical models. This methodology is conducted by an experimentation protocol, made of four successive components - or modules executed one after another. It is also based on a new statistical model for clustering and classification, actually an extension of the classic statistical model for classification. This model and the experimentation protocol are both introduced in the next subsection. Besides, the sequential (re-)execution of the four modules has led to the development of a database construction procedure, based on the UniCart algorithm. It has also motivated the statement of a new statistical model for clustering and classification and the design of a new transfer learning procedure, called Relabeled Nearest Neighbors (RNN).

I.3.1 Presentation of the Experimentation Protocol

Learning trend predictive numerical models requires first to find a time series representation method capable of turning financial series into more homogeneous, interpretable and economically meaningful features. In parallel, we need a machine learning approach, which yields precise and robust prediction functions even when trained with (possibly still) time dependent and non-stationary features. Finally, these prediction functions have to be evaluated and analyzed via backtesting plans. It is important to understand their decision mechanism, for example to highlight the role of each feature or to estimate the probability of realization of any prediction. These interpretation functionalities help to understand better the statistical nature of the data, and to improve the database and model construction methods as well. In addition, they enable comparisons with other opinions, results, or methods in economics and computational finance, which is also a way to limit data snooping. In that perspective, it is useful to automate as much as possible the steps of database construction and of model selection. The backtesting results are then all the most reliable, while further interpretation and enhancements are fully justified.

These requirements for learning efficiently trend predictive numerical models compose the four modules of the experimentation protocol (cf. figure I.14). The first module, entitled *Data Observation and Modeling Choices*, serves to define general modeling choices and objectives. We consider the following set of p finite financial series:

$$D = (D_t^1, \dots, D_t^p)_{t=1,\dots,n_D}.$$

By convention, D^1 is the target variable, i.e. the variable for which we want to compute predictions. Besides, $D^2, \dots, D^p$ are the explanatory variables. Next, we choose the setup, distinguishing between classification, ranking or regression. In other words, before turning the set D into the set of labeled features $Z = (X, Y)$ (as done in the next module), we decide on the nature of the prediction task. Hence we determine the set of labels $\mathcal{Y}$. We have $\mathcal{Y} = \mathbb{R}$ in case of regression and $\mathcal{Y} = \{-1, +1\}$ in case of (binary) classification. The choice of the setup is particularly important, as these prediction objectives are of unequal difficulty. Predicting real labels is indeed more difficult and more ambitious than predicting binary ones, while the difficulty of ranking (i.e. ordering the database objects according to their propensity of being of positive labels) stands in between. Besides, the classification and ranking setups naturally enable more interpretation functionalities, such as the computation of probabilities of realization associated to predictions. Assumptions on the data distribution and on the data generating process are also formulated in this module. For example, we can suppose that the distribution $\mathbb{P}_D$ is Gaussian or that the process generating the $i-$th series $(D_t^i)_{1\leq t\leq n_D}$ is an AR or an ARMA process. Finally, the time ranges for the training set and for the test set are defined as well. They are respectively denoted $\{1, \dots, n_{D_{trn}}\}$ and $\{n_{D_{trn}} + 1, \dots, n_{D_{trn}} + n_{D_{test}}\}$, and satisfy: $n_{D_{trn}} + n_{D_{test}} = n_D$.

The second module, *Database Construction*, turns the explanatory and target series D into features and labels $Z = (X, Y)$ via an user chosen time series representation ϕ. We have:

$$\begin{array}{ccc} D & \overset{\phi_X}{\rightarrow} & X \\ D & \overset{\phi_Y}{\rightarrow} & Y. \end{array}$$

Labels are defined as movements of the target series at an user specified horizon and scale. The specification of the horizon and of the scale necessarily reduce slightly the number of objects in both the training set and the test set. We denote thus the labeled objects of the training set:

$$(X_{trn_1}, Y_{trn_1}), \dots, (X_{trn_{n_{trn}}}, Y_{trn_{n_{trn}}}),$$

where $n_{trn} \leq n_{D_{trn}}$. Note that for clarity, when there cannot be any confusion between the training set and the test set, the training set is simply referred to

$$(X_1, Y_1), \dots, (X_n, Y_n),$$

where $n := n_{trn} \leq n_{D_{trn}}$. Likewise, the labeled objects of the test set are denoted

$$(X_{test_1}, Y_{test_1}), \dots, (X_{test_{n_{test}}}, Y_{test_{n_{test}}}),$$

where $n_{test} \leq n_{D_{test}}$. Besides, note that the number d of features is usually larger than the number p of explanatory variables, as there is more than one computed feature per variable. Hence we have $X_{trn}, X_{test} \in \mathcal{X} = \mathbb{R}^d$. The distributions of the features and labels $P_{X_{trn},Y_{trn}}$ and $P_{X_{test},Y_{test}}$ have to be sufficiently similar to ensure the success of learning techniques. The database construction is thoroughly studied in the second chapter.

In the third module, *Model Construction*, we define a statistical model for clustering and classification. We consider the random triplet $(X, Y, R) \sim P$ where the observation vector $X \in \mathcal{X}$ and the label $Y \in \mathcal{Y}$ have already been defined, and where $R \in \{1, \dots, N_R\}$ is a latent variable standing for a specific distribution of the random pair (X, Y). The distribution of that random pair is given by the mixture model

$$(X, Y) \sim \sum_{r=1}^{N_R} c_r P_r,$$

where $c_r = P(R = r)$ is a weight, and where $P_r = P_{(X,Y)|R=r}$ is a non-parametric component distribution also called regime. Each regime P_r is unknown and the number N_R of regimes is also unknown. Note that this model encompasses the classic models for classification and for regression, for which there is only one regime ($N_R = 1$). The goal is then to find a number N_R, a regime attribution function $H : \mathcal{X} \longrightarrow [1, N_R]$ and regime specific prediction functions $(f_r)_{r=1,\dots,N_R}$, where $f_r : \mathcal{X} \longrightarrow \mathcal{Y}$, so that the numerical model defined by the triplet $\left(N_R, H, (f_r)_{r=1,\dots,N_R}\right)$ yields good backtesting performances. This model is further investigated in the third chapter, where methods for finding the triplet $\left(N_R, H, (f_r)_{r=1,\dots,N_R}\right)$ are proposed.

The fourth and last module, *Backtesting and Numerical Results*, consists of two backtesting plans, which assess the precision of the prediction functions produced in the third module. The first one computes a straightforward measure of the prediction error. The second one designs trading rules using predictions made over the test set and computes their relative profit and loss accounts (P&Ls). Backtesting plans are completed by interpretation functionalities helping the statistician to understand the reasons for any regime attribution or prediction. These functionalities can be related to measures of influence of features in the predictions and therefore to variable ranking and selection. They also can give indications about the reliability of the model and about its predictions. They decisively contribute to formulating better data hypotheses and enhancing the time-series representation, database and model construction procedures. The experimentation protocol I.14 emphasizes thus the interactions between these modules. It shows that any innovation in a module influences the further cyclical execution of the modules. To get backtesting results as reliable as possible, it is therefore helpful to limit user intervention in the experimentation protocol and to automate parameter tuning and model selection in the execution of each module. The fourth chapter is devoted to the presentation of the backtesting results.

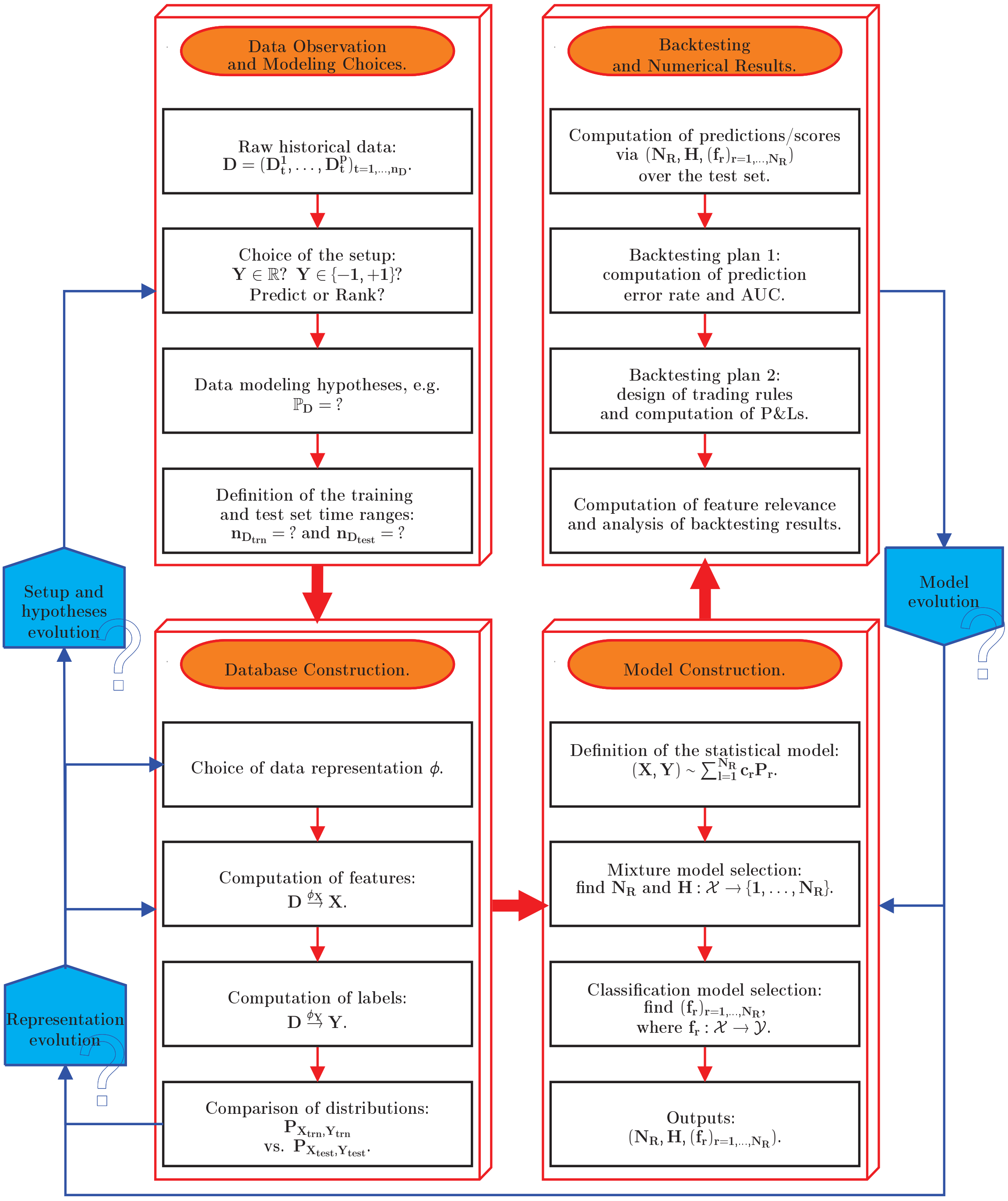

Data Observation and Modeling Choices.
Raw historical data: $\mathbf{D} = (\mathbf{D}^1_t, \ldots, \mathbf{D}^p_t)_{t=1,\ldots,n_D}$.
Choice of the setup: $\mathbf{Y} \in \mathbb{R}$? $\mathbf{Y} \in \{-1, +1\}$? Predict or Rank?
Data modeling hypotheses, e.g. $\mathbb{P}_D = ?$
Definition of the training and test set time ranges: $n_{D_{trn}} = ?$ and $n_{D_{test}} = ?$
Backtesting and Numerical Results.
Computation of predictions/scores via $(\mathbf{N_R}, \mathbf{H}, (\mathbf{f}_r)_{r=1,\ldots,N_R})$ over the test set.
Backtesting plan 1: computation of prediction error rate and AUC.
Backtesting plan 2: design of trading rules and computation of P&Ls.
Computation of feature relevance and analysis of backtesting results.
Setup and hypotheses evolution
?
Model evolution
?
Database Construction.
Choice of data representation ϕ.
Computation of features: $\mathbf{D} \overset{\phi_X}{\to} \mathbf{X}$.
Computation of labels: $\mathbf{D} \overset{\phi_Y}{\to} \mathbf{Y}$.
Comparison of distributions: $\mathbf{P}_{X_{trn},Y_{trn}}$ vs. $\mathbf{P}_{X_{test},Y_{test}}$.
Model Construction.
Definition of the statistical model: $(\mathbf{X}, \mathbf{Y}) \sim \sum_{l=1}^{N_R} c_r \mathbf{P}_r$.
Mixture model selection: find N_R and $\mathbf{H} : \mathcal{X} \to \{1, \ldots, N_R\}$.
Classification model selection: find $(\mathbf{f}_r)_{r=1,\ldots,N_R}$, where $\mathbf{f}_r : \mathcal{X} \to \mathcal{Y}$.
Outputs: $(\mathbf{N_R}, \mathbf{H}, (\mathbf{f}_r)_{r=1,\ldots,N_R})$.
Representation evolution
?

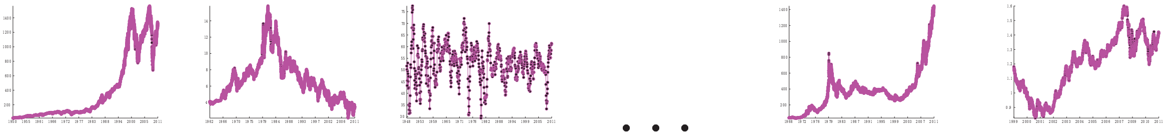
Figure I.14: **Experimentation Protocol and Notations**

I.3.2 Input-Output Diagram of the UniCart/RNN approach

The repeated executions of the modules of the experimentation protocol have led to the design of new procedures. The database construction is thus achieved via UniCart, a multiscale and piecewise approximation method based on the CART algorithm, which is used for the representation of univariate series. In addition, a new transfer (supervised) learning procedure, called Relabeled Nearest Neighbors (RNN), has been developed. This meta-algorithm aims at learning a pseudometric based on the relative importance of features (or of iterative classifiers?) in the prediction. This pseudometric is used to define a hierarchical partitioning structure, which is transferred from the training set to the test set. Finally, the trend predictive numerical models obtained through the UniCart and RNN procedures are evaluated via specific backtesting plans. The first backtesting plan computes the recognition rate and the area under the ROC curve of the RNN classifier. The second one designs trading rules using RNN predictions and scores and calculates their profit and loss accounts (P&Ls). The backtesting plans are also completed by interpretation functionalities. They allows us to evaluate and to improve the model and also to draw conclusions concerning market analysis, that we can compare with mainstream opinions and theories in economics and computational finance. In order to make the reading of the thesis easier, we sum up the UniCart/RNN approach with the following input-output diagram. The modules as well as the flow of the variables between the modules are represented. Note finally that the plan of the next sections is based on the development of each module. The second chapter is devoted to the database construction, the third one to the model construction and the fourth and last one to the presentation and analysis of the backtesting results.

Input-Output Diagram UniCart RNN

Input series: $D = (D_t^1, \ldots, D_t^p)_{t=1,\ldots,n_D}$.



Database Construction via UniCart (Matlab)

- Parameters: $W, w, \tilde{S}, \tilde{K}$.
- Inputs: $D = (D_t^1, \ldots, D_t^p)_{t=1,\ldots,n_D}$.
- Inputs Format: csv file.
- Outputs: $Z = \Big((X_t^1, Y_t^1), \ldots, (X_t^p, Y_t^p)\Big)_{t=1,\ldots,n}$.
- Outputs Format: csv file.

Model Construction via RNN (R)

- Parameters: h, n_{trn}, n_{test}.
- Inputs: $Z_{trn} = \Big((X_{trn_t}^1, Y_{trn_t}^1), \ldots, (X_{trn_t}^p, Y_{trn_t}^p)\Big)_{t=1,\ldots,n_{trn}}$

$Z_{test} = \Big((X_{test_t}^1, Y_{test_t}^1), \ldots, (X_{test_t}^p, Y_{test_t}^p)\Big)_{t=1,\ldots,n_{test}}$

- Inputs Format: csv file.
- Outputs: $\Big(N_R, H, (f_r)_{r=1,\ldots,N_R}\Big)$.
- Outputs Format: R data frames.

Backtesting Results (Excel)

- Parameters: $\theta_{buy}, \theta_{sell}$.
- Inputs: $\Big(f_{H(X_{test_t}^1,\ldots,X_{test_t}^p)}(X_{test_t}^1, \ldots, X_{test_t}^p)\Big)_{t=1,\ldots,n_{test}}$
- Inputs Format: R data frames.
- Outputs: Error Prediction, ROC Curve, P&Ls.
- Outputs Format: csv file.

Chapter II

Construction of Economic and Financial Databases

> Do you wish to rise? Begin by descending. You plan a tower that will pierce the clouds? Lay first the foundation of humility
>
> Saint Augustine

Chapter I focuses on the applicability of machine learning techniques to the construction of trend predictive models for financial variables. It introduces an experimentation protocol, which achieves this construction in four interdependent modules. Particularly, the second module aims a computing the database used to train the numerical model. Ideally, this database is made of features (and labels) homogeneous enough to satisfy the requirements of the statistical learning theory. Besides, these features must be sufficiently intuitive and interpretable to enable further developments and improvements of the overall trend prediction approach. The search for a time series representation, capable of providing a database meeting these objectives, is thus totally justified. This chapter introduces two time series representations. The first one, well-known to quantitative analysts, combines state-space models and the Kalman filter. The second one, called UniCart, is a new multiscale and piecewise approximation method. Based on the CART algorithm, it implements a top-down binary tree structured segmentation. In this chapter, we focus on the UniCart method. We show that it can lead to the design of two types of databases. These databases are made of trend-based features, which are close to the representations used in technical analysis. They can therefore help reproducing automatically the reasoning of technician analysts and market professionals. The plan of the chapter is as follows. Significant economic and financial variables, which play the role of target series and of explanatory series in chapter IV, are introduced in the first section. Basic stylized facts are recalled. The use of sophisticated time series representations is then motivated. Specific criteria for representation methods are highlighted. They evaluate qualitatively the capacity of these methods to turn heterogeneous sequences into databases relevant for modelling. Representation methods meeting these criteria in a satisfactory way are reported. State- space models and Kalman filter as well as the UniCart method are particularly made explicit. The UniCart method is particularly emphasized and illustrated. Its implementation by a binary tree structure based top down procedure is thoroughly documented, and two variants - regression and interpolation - are introduced. The second section is devoted to the construction of UniCart databases. Their principle of construction consists in shifting a sliding window across the time ranges of explanatory variables to compute local features. It is explained in detail. Two types of features, hence two types of databases, are then displayed. The first one is composed of lagged trends, considered at a single chosen scale, and of associated statistical quantities, while the second one is made of multiscale returns of lag 1. Both are extensively illustrated and their stationarities are particularly studied. Finally, we sum up the

Matlab implementation of the construction of UniCart databases with an input-output diagram.

II.1 Representations of Economic and Financial Series

This section aims at introducing time series representations, which are relevant for constructing trend predictive models. In that perspective, key financial variables, used in the fourth chapter as target series, are first introduced, as well as specific explanatory economic and financial variables. Standard stylized facts are recalled. A synthesis of the tasks pursued in this document via a well-chosen time series representation initiates the second subsection. Ideal properties of time series representations for learning efficient and interpretable numerical models are particularly stated and commented. Classical methods, which are likely to check these properties, are then briefly reviewed. We focus especially on the application of the Kalman filter to linear state-space models, what is often used by quantitative analysts to detrend financial series. UniCart, a representation method performing multiscale piecewise approximation, is finally introduced. Its linear version produces configurations close to technical analysis patterns via the computation of linear trends at various scales, which is illustrated. Its implementation according to a top-down binary tree structured segmentation procedure is also thoroughly presented.

II.1.1 Presentation of Key Economic and Financial Variables

As we are interested in predicting the trends of financial variables in general, we need to consider a set of variables which are representative of the diversity of financial data. We propose to select variables from different areas of the financial markets. In the following, we introduce target variables, i.e. variables whose trends, converted to labels, have to be predicted.

1. Stocks:

    - the Daily S&P 500 Index expressed in U.S. Dollars, denoted **SP500** (cf. figures II.1 and II.2) and available at `http://fr.finance.yahoo.com/q/hp?s=^GSPC`. This is a value-weighted index composed of the prices of 500 common stocks, nearly all of them being stocks of the 500 most important companies trading in the American stock markets.

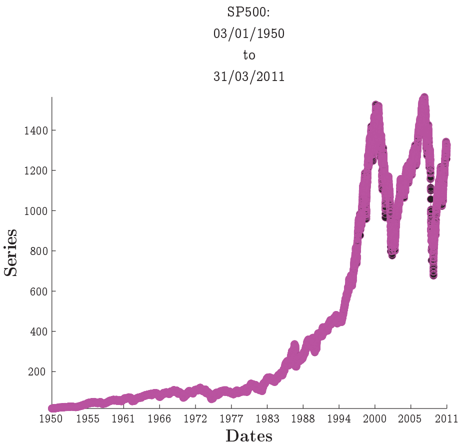


Figure II.1: **SP500 Series**

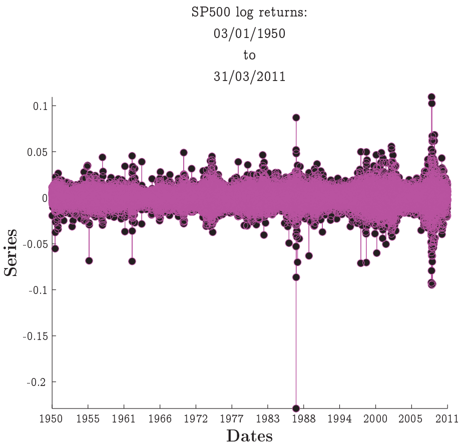


Figure II.2: **SP500 Log Returns**

- the Daily Euro Stoxx 50 Index expressed in Euros, denoted **SX5E** (cf. figures II.3 and II.4) and available at `http://www.stoxx.com/download/historical_values/hbrbcpe.txt`. This index is composed of the prices of 50 stocks, all of them being stocks of European major and sector leader companies.

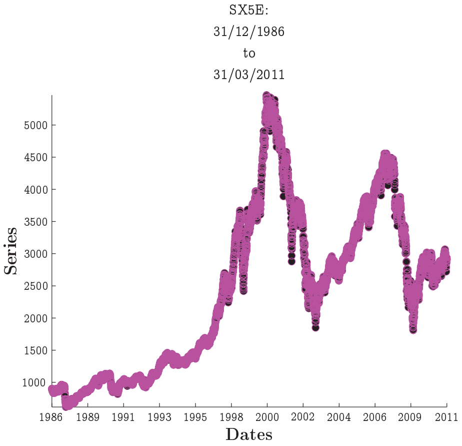


Figure II.3: **SX5E Series**

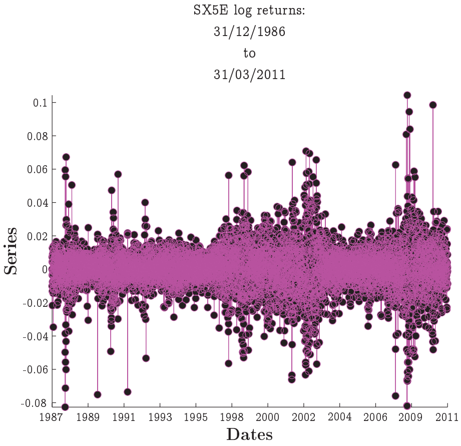


Figure II.4: **SX5E Log Returns**

2. Commodities:

- the Daily Price of Gold (afternoon fixing in London) expressed in U.S. dollars, denoted **GOLD** (cf. figures II.5 and II.6) and available at `http://www.bundesbank.de/statistik/statistik_zeitreihen.php?lang=en&open=devisen&func=list&tr=www_s332_b01015_3`.

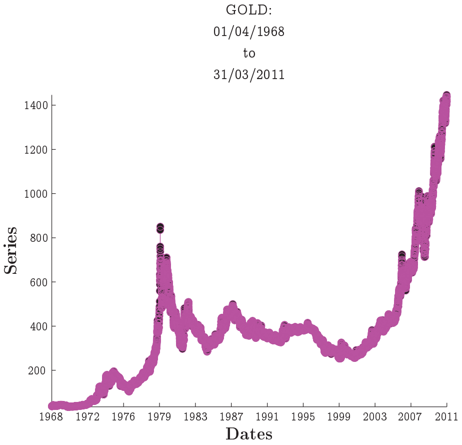


Figure II.5: **GOLD Series**

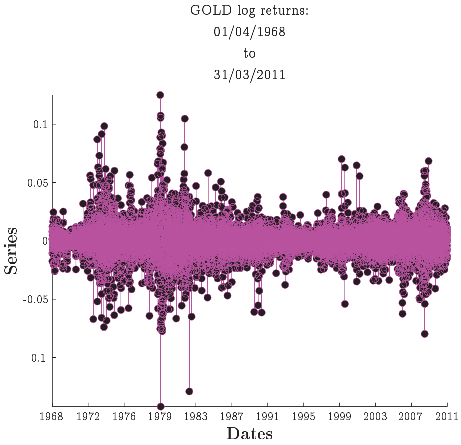


Figure II.6: **GOLD Log Returns**

- the West Texas Intermediate Daily Spot Oil Price expressed in U.S. dollars per barrel, denoted **OIL** (cf. figures II.7 and II.8) and available at `http://www.eia.gov/dnav/pet/hist/LeafHandler.ashx?n=PET&s=RWTC&f=D`. West Texas Intermediate is a light and sweet crude oil. It is used as a benchmark in oil pricing and is the underlying commodity of New York Mercantile Exchange's oil futures contracts.

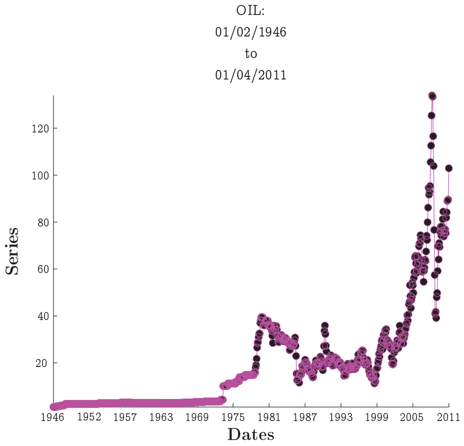


Figure II.7: **OIL Series**

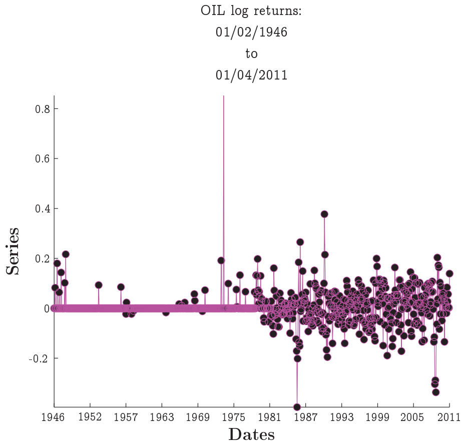


Figure II.8: **OIL Log Returns**

3. Exchange rates:

- the Daily U.S./Euro Foreign Exchange Rate, denoted **EURUSD** (cf. figures II.9 and II.10) and available at `http://research.stlouisfed.org/fred2/series/DEXUSEU`.

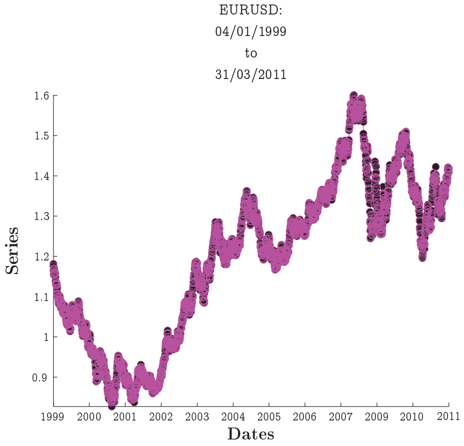


Figure II.9: **EURUSD Series**

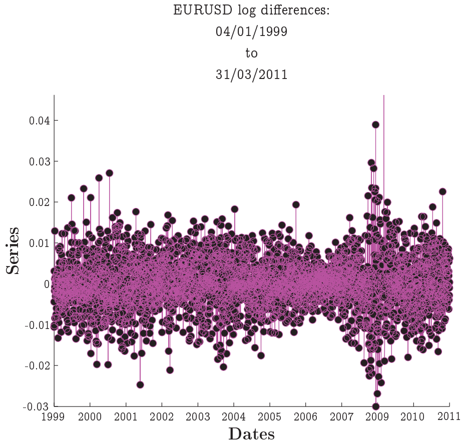


Figure II.10: **EURUSD Log Differences**

- the Daily U.S./U.K. Foreign Exchange Rate, denoted **DEXUSUK** (cf. figures II.11 and II.12) and available at `http://research.stlouisfed.org/fred2/series/DEXUSUK`.

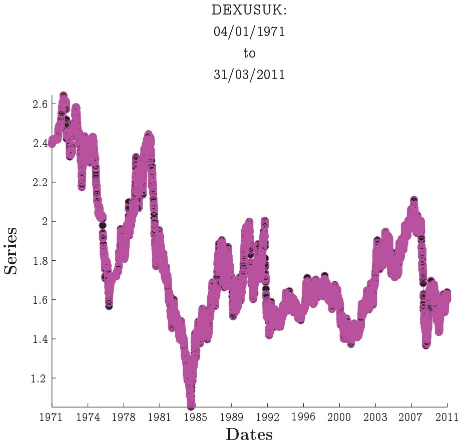


Figure II.11: **DEXUSUK Series**

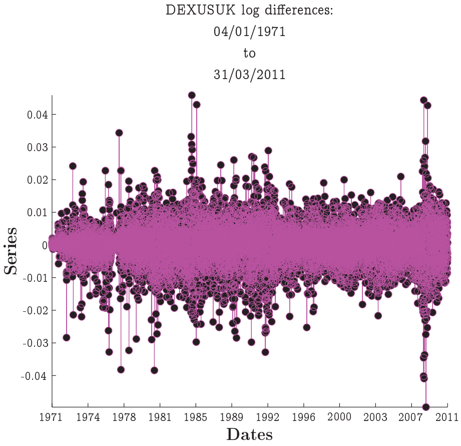


Figure II.12: **DEXUSUK Log Differences**

A specific set of explanatory economic and financial variables is associated with each of these target variables in the fourth chapter of the thesis. Naturally, explanatory and target variables are not exclusive sets. Any target variable can play the role of explanatory variable of itself or of any other target variable. As an example, the set of explanatory variables for **SP500** includes **OIL** and **GOLD**. It is completed by the following significant U.S. economic variables, that we mention and plot to illustrate the diversity of economic and financial data.

- the Daily U.S. 10-Year Treasury Constant Maturity Rate, denoted **DGS10** (cf. figures II.13 and II.14) and available at `http://research.stlouisfed.org/fred2/series/DGS10`. As explained in section I.1, treasury securities are debt financing instruments issued by governments. Different maturities are available. Treasury bills refer to less than one year maturity and do not pay interest until their maturity. Treasury bonds have a maturity of ten or twenty years and have a coupon payment every six months.

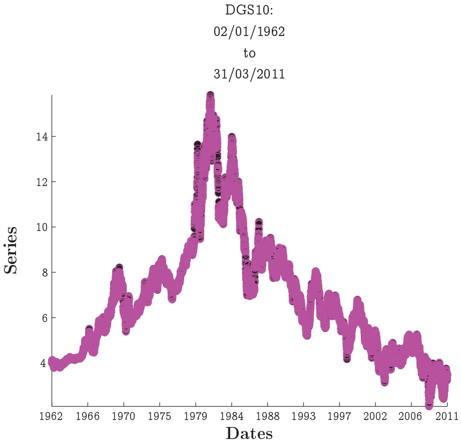


Figure II.13: **DGS10 Series**

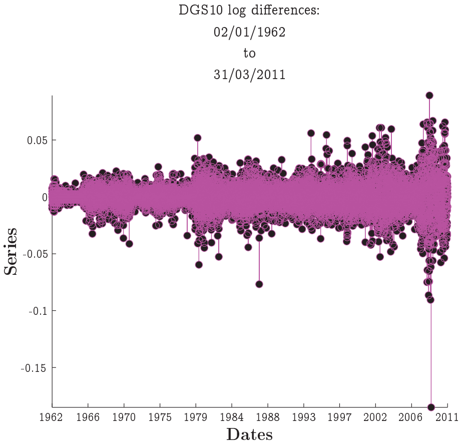


Figure II.14: **DGS10 Log Differences**

- the US 3-Month Treasury Bill, denoted **DTB3** (cf. figures II.15 and II.16) and available at `http://research.stlouisfed.org/fred2/series/DTB3`.

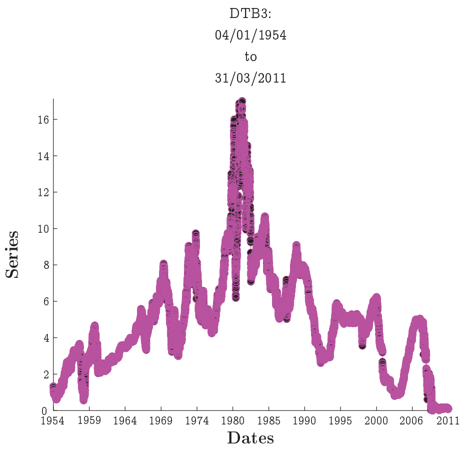


Figure II.15: **DTB3 Series**

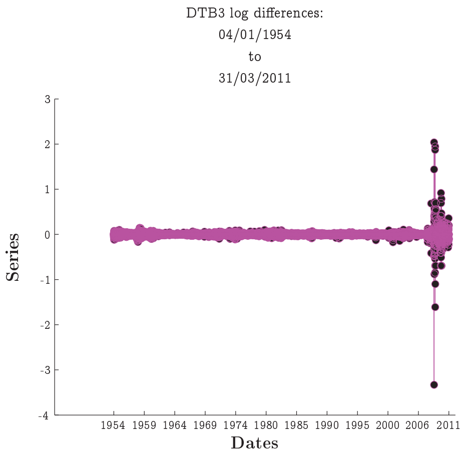


Figure II.16: **DTB3 Log Differences**

- the S&P 500 Price-to-Earnings Ratio, denoted **PER** (cf. figures II.17 and II.18) and available at `http://www.econ.yale.edu/~shiller/data.htm`. The P/E ratio of a stock is a measure of the price paid for a share relative to the annual net income or profit earned by the firm per share. In the case of a market index such as the S&P 500, a weighted average of all stock constituents is computed, by calculating each stock's underlying market cap (price multiplied by number of shares in issue) to give the total value in terms of market capitalization for the whole market index.

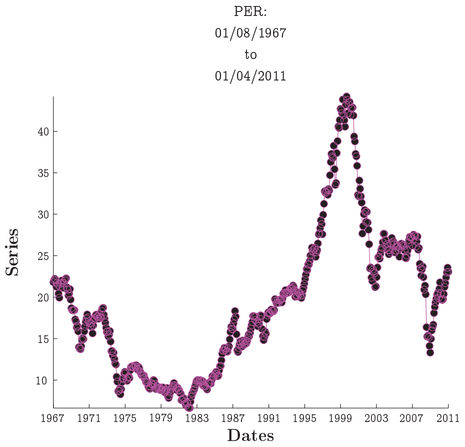


Figure II.17: **PER Series**

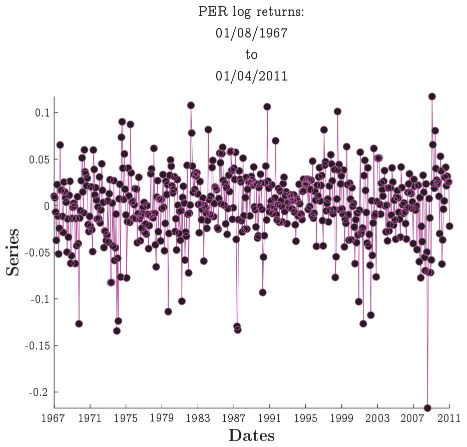


Figure II.18: **PER Log Returns**

- the US Unemployment Rate, denoted **UNEMPLOY** (cf. figures II.19 and II.20) and available at `http://research.stlouisfed.org/fred2/series/UNEMPLOY`.

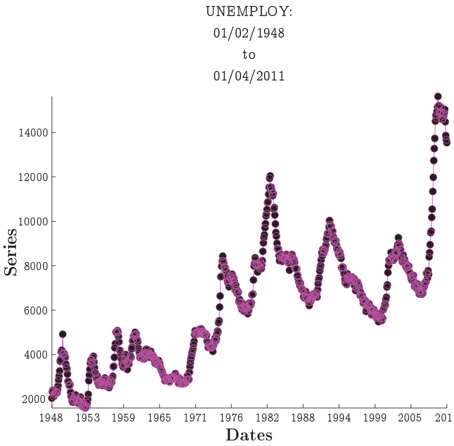


Figure II.19: **UNEMPLOY Series**

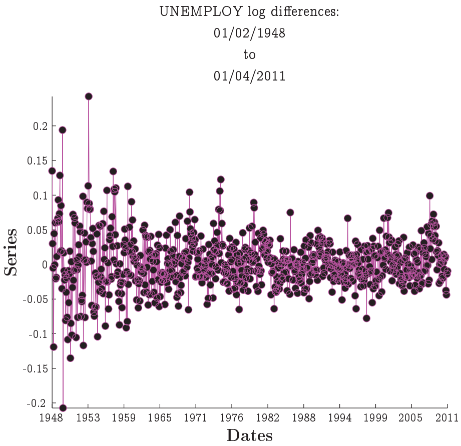


Figure II.20: **UNEMPLOY Log Differences**

- the NAPM, ISM Manufacturing, Purchasing Managers' Composite Index, denoted **NAPM** (cf. figures II.21 and II.22) and available at `http://research.stlouisfed.org/fred2/series/NAPM`. Released by the Institute for Supply Management every month, this economic indicator is a composite index that is based on five major indicators: new orders, inventory levels, production, supplier deliveries, and the employment environment. It is the best indicator of factory production and it is popular for detecting inflationary pressure as well as manufacturing economic activity, both of which investors pay close attention to.

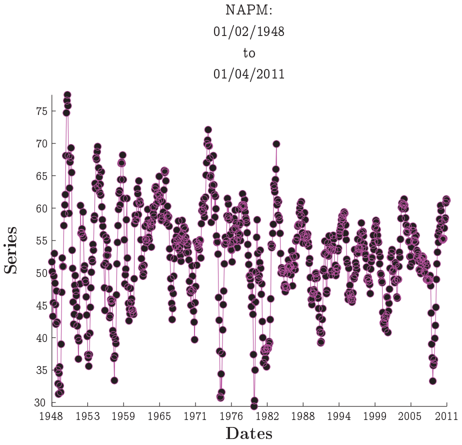


Figure II.21: **NAPM Series**

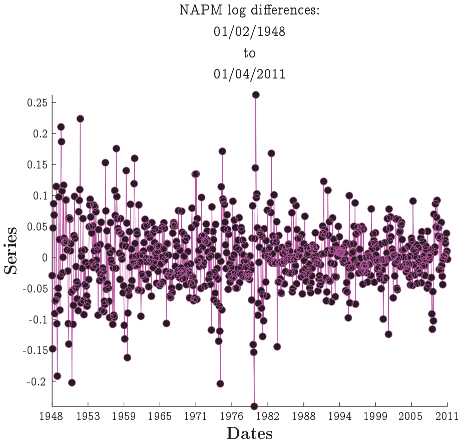


Figure II.22: **NAPM Log Differences**

- the Personal Income, denoted **INCOME** (cf. figures II.23 and II.24) and available at `http://research.stlouisfed.org/fred2/series/PI`.

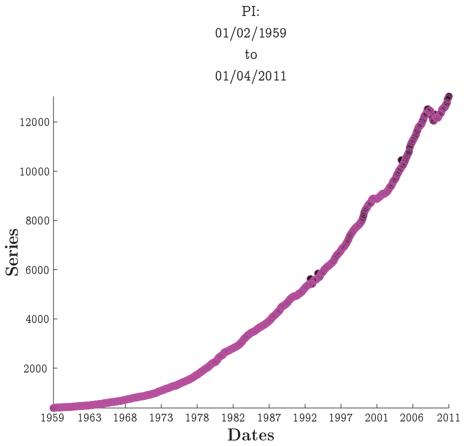


Figure II.23: **INCOME Series**

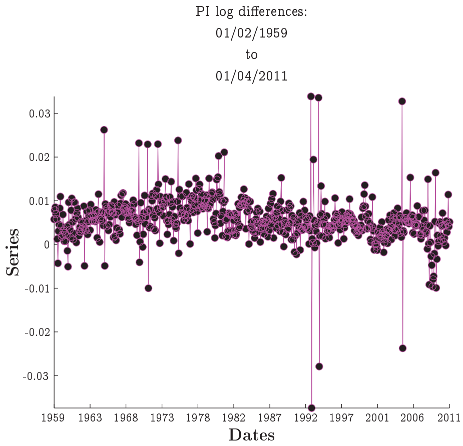


Figure II.24: **INCOME Log Differences**

- the US Population, denoted **POP** (cf. figures II.25 and II.26) and available at `http://research.stlouisfed.org/fred2/series/POP`.

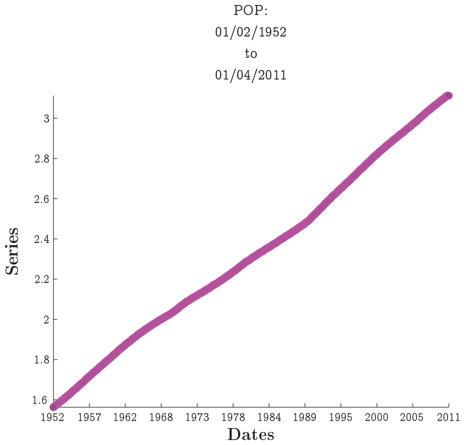


Figure II.25: **POP Series**

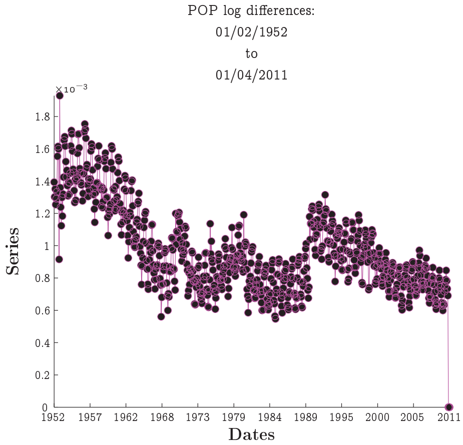


Figure II.26: **POP Log Differences**

- the Consumer Price Index for All Urban Consumers, denoted **CPIAUCSL** (cf. figures II.27 and II.28) and available at `http://research.stlouisfed.org/fred2/series/CPIAUCSL`. This variable is an inflation indicator.

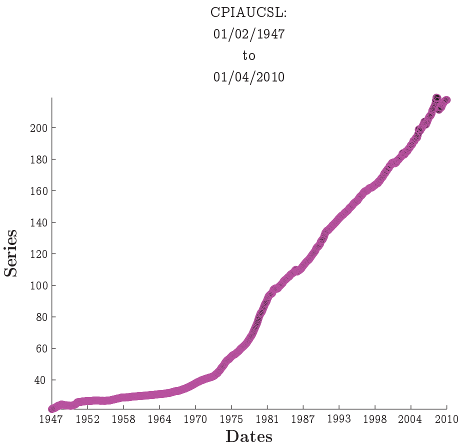


Figure II.27: **CPIAUCSL Series**

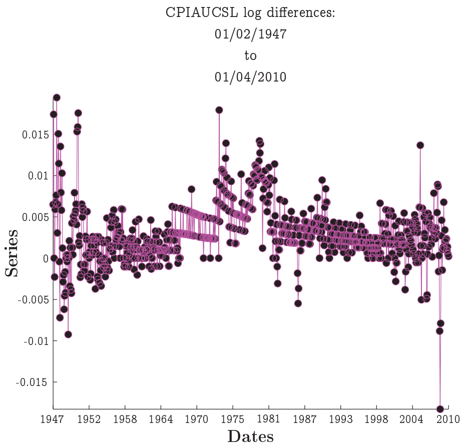


Figure II.28: **CPIAUCSL Log Differences**

- the 30-Year Conventional Mortgage Rate, denoted **MORTG** (cf. figures II.29 and II.30) and available at `http://research.stlouisfed.org/fred2/series/MORTG`.

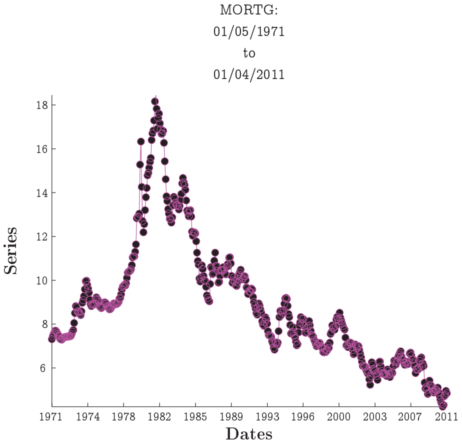


Figure II.29: **MORTG Series**

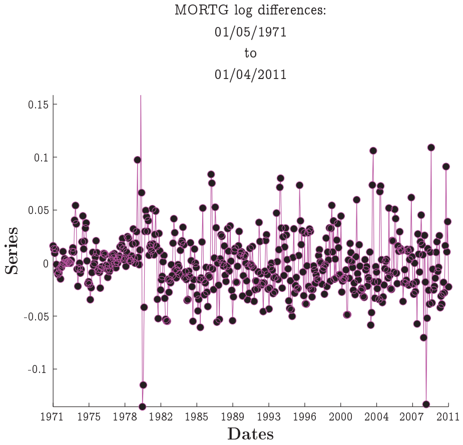


Figure II.30: **MORTG Log Differences**

Additional explanatory variables are used in this document for building trend predictive numerical models for the target variables **SX5E**, **GOLD**, **OIL**, **EURUSD**, **DEXUSUK**.

- the CAC40 French Stock Market Index, denoted **CAC40**, available at `http://finance.yahoo.com/q?s=^FCHI`.
- the DAX German Stock Market Index, denoted **DAX**, available at `http://finance.yahoo.com/q?s=^GDAXI`.
- the Euro Area 10-Year Government Benchmark Bond Yield, denoted **EURGS10**, available at `http://sdw.ecb.europa.eu/quickview.do?SERIES_KEY=143.FM.M.U2.EUR.4F.BB.U2_10Y.YLD`.

- the daily German 10-Year Treasury Constant Maturity Rate, denoted **BUND**, available at http://www.bundesbank.de/statistik/statistik_zeitreihen.en.php?lang=en&open=zinsen\&func=row\&tr=WT4608.
- the 3-Month Eurodollar Deposit Rate, denoted **DED3**, available at http://research.stlouisfed.org/fred2/series/DED3.
- the Daily Chine/U.S. Foreign Exchange Rate, denoted **DEXCHUS**, available at http://research.stlouisfed.org/fred2/series/DEXCHUS.
- the Daily Hong Kong/U.S. Foreign Exchange Rate, denoted **DEXHKUS**, available at http://research.stlouisfed.org/fred2/series/DEXHKUS.
- the Daily India/U.S. Foreign Exchange Rate, denoted **DEXINUS**, available at http://research.stlouisfed.org/fred2/series/DEXINUS.
- the Daily Japan/U.S. Foreign Exchange Rate, denoted **DEXJPUS**, available at http://research.stlouisfed.org/fred2/series/DEXJPUS.
- the Daily Switzerland/U.S. Foreign Exchange Rate, denoted **DEXSZUS**, available at http://research.stlouisfed.org/fred2/series/DEXSZUS.
- the Weekly U.S. Ending Stocks of Crude Oil and Petroleum Products, denoted **OILSTOCKS**, available at http://www.eia.gov/dnav/pet/hist/LeafHandler.ashx?n=PET&s=WTTSTUS1&f=W.
- the Total Industry Capacity Utilization, denoted **TCU**, available at http://research.stlouisfed.org/fred2/series/TCU.

Economic and financial variables are usually highly noisy, serially correlated and nonstationary (cf. to subsection I.1.4). This temporal heterogeneity, specific to each series, is usually first addressed by computing straightforward transformations, such as returns, logarithmic returns (as systematically plotted previously), indices or rates. These transformations yield new representations for each series. Their interest is that they have more interesting statistical properties than raw samples. Particularly, they are closer to stationary signals [Tsa05]. They are thus the first step to the development of econometric models which can for example reproduce the effects of volatility clustering (such as conditional heteroscedastic models) or capture abrupt changes (such as regime-switching models). Likewise, homogeneous features are needed to develop trend predictive models. Obviously, the vector formed by any target variable and its corresponding explanatory variables is not only temporally heterogeneous (because of the temporal heterogeneities of each variable), but also spatially heterogeneous. Indeed, these variables have different historical ranges and are available at different sampling rates: daily for financial data and monthly or quarterly for economic data. The latter ones are often lagged and/or revised. Because of these temporal and spatial heterogeneities, the relevant scales of observation may be very different from one variable to another. A time series representation capable of addressing these intra and inter heterogeneities is thus required. Its relevance will be first evaluated by the performance of its associated trend predictive model, i.e. via the backtesting plans of the experimentation protocol. It will also be assessed by the propensity of the database and model construction procedures to be improved, i.e. by the richness of the informations provided by interpretation functionalities. Note that the results obtained via a time series representation, which is close to the representation methods used by financial practitioners, are likely to be more easily interpreted. In the next subsection, the use of time series representations for developing trend predictive models is thoroughly motivated. The properties of the ideal representation method are particularly listed.

II.1.2 Stakes of Time Series Representation

Time series representations are a subject of great interest in the signal processing, machine learning and data mining communities. In this document, they are employed to accomplish or to make possible the following tasks - made explicit in [KCHP93, KK03, LKLC03, LE07, Reb07]:

- compression/approximation: it consists in turning a time series of length n into a set of k associated features, where $k \ll n$, so that the new sparse representation provides a reconstruction of the series, usually optimal according to a distance.
- indexing: in a database, it aims at retrieving the time series which is most similar to a target time series, according to a similarity criterion or a distance.
- clustering: it aims at computing a grouping structure for the time series of a database. This structure can be a set of clusters or a dendrogram (a nested set of clusters). It is determined using a similarity criterion or a distance.
- classification/regression: it aims at predicting the label of each time series of a test set via a classifier built using a training set. The labels can belong to a finite set (this is classification), or can be real (this is regression).

Specific properties are required to meet these objectives. From a signal processing point of view, it is first particularly relevant to provide a representation method whose properties of approximation can handle with precision the noise inherent in each series, existing potentially at different scales and varying over time [Mal99]. From a statistical learning point of view, what matters is to work with stationary features. We expect therefore a homogenization effect from the representation method in order to make plausible the hypothesis of identical distribution (cf. subsection I.2.1). Note also that the labels are predefined in many applications concerned with classification and regression tasks. This is not the case when constructing trend predictive models. Actually, a specific procedure for extracting features as well as another one for extracting trends, which play the role of labels, are necessary. Ideally, both would be provided by the same time series representation method. Another essential point is the robustness of the representation, which would ensure that similar economic or financial series have close representing features. We are also interested in working with features analogous the representations used by financial practitioners and experts. Our conclusions concerning the validity of the constructed models and the relevance of specific features would then be easily comparable to their opinions and theories. Particularly, multiscale features would help to question the concept of trend, introduced in technical analysis as a series of consecutive ascending or descending bottoms and tops [BB02, Mur99], and its prediction ability. We also have to deal with the curse of dimensionality [HTF01]. This refers to the exponential increase volume caused by adding dimensions to a Euclidean space. In a high dimensional space, all the points are "far away" from each other. In other words, too many features would be counterproductive when considering clustering and classification tasks. Admittedly, the compression ability of the representation method is not an absolute priority as the new learning procedures resist to the curse of dimensionality well [Vay06]. However, regarding the representation and model interpretability, it would definitely be extremely convenient. To sum up, the ideal time series representation would provide multiscale, homogeneous, interpretable (from an economic point of view), robust and parsimonious features as well as labels.

Time series representations suitable for the tasks of compression, indexing, clustering and prediction, and a priori capable of handling heterogeneous multivariate series are briefly reviewed here. We focus first on harmonic analysis methods. They aim at providing a sparse representation of any given finite series (of finite energy) via a decomposition over a chosen family of elementary functions [Mal99]. This family is usually called a dictionary. Elementary functions are referred to as atoms. Dictionaries can consist of a single orthonormal basis or can be made of redundant families, which improve the sparsity of the series representation. Fourier analysis is thus based on the orthonormal basis made of sinusoidal waves. Any function of finite energy and of finite support can be represented as a sum of such elementary functions. Fourier analysis is well adapted to the representation of uniform regular functions but fails to provide a sparse representation when confronted with non stationary series [Mal99]. On the other hand, wavelet bases, which are orthonormal bases made of atoms localized both in time and in frequency [Mey87, Dau88, Mal], yield parsimonious representations of piecewise regular functions containing transient perturbations. They also define a multiscale (the coined term being multiresolution) representation of series. Besides, constructing dictionaries specifically adapted to the series is also possible. Wavelet packet (orthonormal) bases provide atoms whose frequency localization is optimally determined according to a cost function [CMW92, CW92]. Likewise, local cosine (orthonormal) bases are computed so that their time localization is optimal. Finding sparse adaptive representations in redundant dictionaries is finally a popular alternate solution, illustrated by the development of matching pursuit algorithms [MZ93] and basis pursuit algorithms [CDS01]. Such procedures have been extensively used to decompose and filter financial series more efficiently [RZ97, Gre98, Ram99], sometimes in conjunction with econometric models [Cap02]. Different concepts of local stationarity have also been introduced to study non- stationary time series more accurately [Koz96, Dah97, MPZ98, NvSK00, FBvS03, DMvS98, lsc, CS03, FSR06].

Singular spectrum analysis [BK86a, BK86b, GNZ01] is a radically different time series representation which provides filtering, forecasting and detection of structural change facilities. The filtering or analysis step aims at decomposing any given one-dimensional series of interest into additive components of various resolutions, distinguishing between a long term trend, oscillatory and noise components. The analysis step actually consists of two steps:

- the decomposition step turns the original one-dimensional series of interest into a set of lagged subseries, which forms the trajectory matrix M. The singular value decomposition of the trajectory matrix provides a decomposition into elementary matrices, each of them being associated with an eigenvector of $MM^\top$.
- the reconstruction step regroups the elementary matrices into a set of new matrices. Each of them is transformed into a new series, which is an additive component of the original series. Criteria and conditions for separability between trend and oscillatory components, as well as specific corresponding procedures, are proposed in [GNZ01].

Once trend components have been extracted, we consider the linear subspace defined by the eigenvectors associated with each trend. It is used to compute a linear recurrent formula between the components of the time series [GNZ01]. This formula allows future values of the time series to be forecast. Structural changes are said to occur when the time series is no longer governed by the formula. Detection procedures are proposed in [GNZ01]. Although singular spectrum analysis is a very recent technique, mostly employed to handle climatic, meteorological and geophysical time series, it has also been successfully used for modeling and predicting financial series [TWW02, HSZ09].

The data mining point of view is also extremely interesting as it provides interpretable and intuitive time series representations. Piecewise Aggregate Approximation (PAA), which actually refer to piecewise constant approximation, is one of them. Time series are thus summarized by a set of consecutive constant segments, whose length is user defined [YF00] or adaptively determined [CKMP02]. Such techniques can also be used as a transformation preceding a discretization step which turns the time series into a consecutive set of equiprobable symbols. This latter technique, called Symbolic Aggregate approXimation (SAX), is a very popular method in data mining [LKLC03]. Besides, Piecewise Linear Approximation (PLA) techniques are also largely employed. They aim at summarizing time series via linear interpolation or linear regression, and are usually implemented by bottom-up or top-down procedures [KCHP93, KCHP01]. Such data mining techniques are currently more and more used to represent and analyze financial times series [cFlCLmN08, cFlCyKmN08, cFlCLmN07, HA07, cFlCmN06].

Now, econometric models (cf. subsection I.1.4) remain the most common representation techniques of financial series. Actually, they are not applied to the series themselves but systematically to the returns or logarithmic returns of the series, which are closer to stationary signals. The next subsection therefore focuses on linear state-space models and on the Kalman filter, which enables to perform various inferences tasks on this broad class of models.

II.1.3 State-Space Models and Kalman Filter

State-space models are defined by a set of two equations: the observation equation II.1 and the state equation II.2 [RG99, DK01, Mur02]. We set the following notations: $O_t \in \mathbb{R}^m$ is an **o**bserved random vector, while $U_t \in \mathbb{R}^p$ is a hidden (or **u**nobserved) random vector. In addition, Π_t and Γ_t are real matrices of size respectively $m \times p$ and $p \times p$ and have to be specified. Finally, σ_{V_t} and σ_{W_t} are the observation and evolution covariance matrices of size respectively $m \times m$ and $p \times p$. The following linear state space model is defined:

$$\begin{cases} O_t = \Pi_t U_t + V_t, \qquad V_t \sim \mathcal{N}(0, \sigma_{V_t}), & \text{(II.1)} \\ U_t = \Gamma_t U_{t-1} + W_t, \qquad W_t \sim \mathcal{N}(0, \sigma_{W_t}), & \text{(II.2)} \\ U_0 \sim \mathcal{N}(m_0, \sigma_{V_0}), \end{cases}$$

In the application of the fourth section, we assume that $\sigma_{V_t} = \sigma_V$ and $\sigma_{W_t} = \sigma_W$, for any t. Then the observation and evolution covariance matrices σ_V and σ_W are the only parameters of the model. They are estimated from available data using maximum likelihood, thanks to the L-BFGS-B quasi-Newton optimization algorithm (cf. [LN89, RHBZ95]).

The Kalman filter is an algorithm employed for recursively estimating the internal state of the process U_t given the sequence of noisy observations $O_{1:\tau} = (O_1, \ldots, O_\tau)$ [Kal60, KB61, RG99, Mur02]. This inferring task is called filtering if $\tau = t$, smoothing if $\tau > t$ and predicting if $\tau < t$. We denote by $\hat{U}_{t|\tau}$ the estimate of the state at time t given observations up to and including time τ. Besides, $C_{t|\tau}$ is the associated error **c**ovariance matrix and Id is the identity matrix. The Kalman filter can be summed up by the system of equations:

$$\begin{cases} \hat{U}_{t|t-1} = \Gamma_t \hat{U}_{t-1|t-1}, & \text{(II.3)} \\ C_{t|t-1} = \Gamma_t C_{t-1|t-1} \Gamma_t^\top + \sigma_{W_{t-1}}, \\ I_{r_t} = O_t - \Pi_t \hat{U}_{t|t-1}, & \text{(II.4)} \\ S_t = \Pi_t C_{t|t-1} \Pi_t^\top + \sigma_{V_t}, \\ K_t = C_{t|t-1} \Pi_t^\top S_t^{-1}, \\ \hat{U}_{t|t} = \hat{U}_{t|t-1} + K_t I_{r_t}, \\ C_{t|t} = (Id - K_t \Pi_t) C_{t|t-1}, \end{cases}$$

The equation (II.3) computes the predicted state at step t, while the equation (II.4) the innovation residual. Therefore, the observed signal O_t can be decomposed into a predicted filtered component and an innovative residual. This is the time series representation provided by the Kalman filter applied to state-space models. It is often used by quantitative analysts to detrend financial time series. The next subsection is based on multiscale piecewise linear approximation, which is maybe closer to the concerns and intuitions of both technical analysists and economists.

II.1.4 UniCart

Methodology and Notations. UniCart, which shares close ties with the Classification And Regression Trees (CART) algorithm [BFOS84], is a method we have developed. It provides a time series representation based on multiscale piecewise approximation. It consists of top-down binary-tree structured procedure, completed by a best-scale criterion. As with CART, the procedure is declined in a tree-growing and a tree-pruning phase. Before stating explicitly the goal of UniCart, we need to introduce notations and concepts. They are very similar to those recalled in the appendix for tree classifiers. Consider a finite real series $d = (d_t)_{t \in I}$, where $I = \{1, \ldots, |I|\}$. UniCart recursively partitions the set of indices I of the series d. It produces a finite set of disjoint subsets $(I_j)_{1 \leq j \leq J}$

such that their union forms the set I, i.e.

$$I = \bigcup_{j=1}^{|I|} I_j.$$

We define this recursive partition as the couple $(tr, \tilde{tr})$, where tr is a rooted binary tree $tr = (V, A)$ and where $\tilde{tr}$ is a function of the type:

$$\tilde{tr} = \begin{cases} I \times V & \longrightarrow & \mathcal{P}_I \\ (i, v) & \longmapsto & \tilde{tr}(i, v), \end{cases}$$

$\mathcal{P}_I$ being the set of partitions of I. The function $\tilde{tr}$ satisfies the following conditions.

- The root $1 \in V$ of the tree tr checks $\tilde{tr}(I, 1) = I$.
- Each node $v \in V$ is associated with a subset $\tilde{tr}(I, v)$ such that its child nodes are associated respectively with $\tilde{tr}(I, v_L)$ and $\tilde{tr}(I, v_R)$, which are disjoint subsets checking:

$$\tilde{tr}(I, v) = \tilde{tr}(I, v_L) \bigcup \tilde{tr}(I, v_R).$$

We choose the recursive partition $(tr, \tilde{tr})$ in a set of candidate functions, made of finite intersections of splits. We define a **split** or **splitting** rule I_{sp} as a finite subset of I of the type

$$I_{sp} = \{1, \ldots, t_{sp}\}.$$

The set of finite combinations of splits is denoted Sp. Additionally, there exists $I_{sp_v} \in Sp$ such that:

$$\begin{array}{rcl} \tilde{tr}(I, v_L) & = & I_{sp_v} \cap \tilde{tr}(I, v), \\ \tilde{tr}(I, v_R) & = & I^c_{sp_v} \cap \tilde{tr}(I, v). \end{array}$$

Finally, we define the approximation function f as a function of the type:

$$f = \begin{cases} I & \longrightarrow & \mathbb{R} \\ t & \longmapsto & f(t). \end{cases}$$

We call regression tree the triplet $(tr, \tilde{tr}, f)$. Denoting $\tilde{V}$ the set of the leaves of the tree tr, the value $l(t)$ attributed to t by the regression tree $(tr, \tilde{tr}, f)$ has to be close to the value d_t. It is given by the formula:

$$l(t) = f\left(\bigcup_{v \in \tilde{V}} \{t \in \tilde{tr}(I, v\} \right).$$

Goal. We can now state the goal of the UniCart procedure. Referring to the appendix, it aims at minimizing the tree impurity $Q(tr, \tilde{tr}, f)$. With a slight abuse of notation, we denote the impurity criterion: $Q(tr) := Q(tr, \tilde{tr}, f)$. We recall that the tree impurity $Q(tr)$ is the sum of the impurities $Q(v)$ of the nodes $v \in V$, which can be written:

$$Q(tr) = \sum_{v \in \tilde{V}} Q(v).$$

Referring to the appendix again, minimizing the impurity $Q(tr)$ can be achieved through the iterative maximization of the quantity

$$\Delta Q(sp, v) = Q(v) - Q(v_L) - Q(v_R),$$

called the goodness of split. The best split sp_* is the split in Sp which most decreases $\Delta Q(sp, v)$. We have:

$$\begin{aligned}\Delta Q(sp, v) &= Q(v) - Q(v_L) - Q(v_R),\\ \Delta Q(sp_*, v) &= \max_{sp \in Sp} \Delta Q(sp, v).\end{aligned}$$

The recursive partition stops as soon as a stopping criterion is fulfilled, i.e. when the subsets of are of a (user-)required size in our implementation.

Besides, selecting the most suitable level of segmentation is possible through the minimization of a cost-complexity function. We define the cost-complexity function $Q_\alpha(tr)$ (actually the same as the function defined for the CART algorithm [BFOS84]):

$$Q_\alpha(tr) = Q(tr) + \alpha|\tilde{V}|.$$

Minimal cost-complexity pruning (tuning parameter α) is classically done using k-fold cross-validation. This criterion allows the best scale in the tree structured linear decomposition to be found, that is the ensemble of terminal nodes $\tilde{V}$ giving the best adapted approximation indexed in the tree.

UniCart Pseudocode. The parameters of the UniCart procedure are the series $d = (d)_{t=1,\dots,|I|}$ and the parameter w, which defines the minimal node size under which the segmentation stops. The steps for executing the UniCart procedure are recapitulated in the following pseudocode.

```
1. Initialization:
    • $v = 0$, $nextv = 1$.
    • $V = \{1\}, A = \emptyset, tr = (V, A), \tilde{tr}(I, 1) = I$.
2. While $v < nextv$
    • If $|\tilde{tr}(I, v)| \geq w$
        - Compute $\Delta Q(sp, v) = Q(v) - Q(v_L) - Q(v_R)$, $\forall sp \in Sp$.
        - Compute $\Delta Q(sp_*, v) = \max_{sp \in Sp_v} \Delta Q(sp, v)$.
        - Creation of $v_L$ the left child node of $v$, corresponding to the split $sp_*$. We have:
          $\begin{cases} \tilde{tr}(I, v_L) &= I_{sp_*} \cap \tilde{tr}(I, v), \\ V &= V \bigcup \{v_L\}, \\ A &= A \bigcup \{v v_L\}. \end{cases}$
        - Creation of $v_R$ the right child node of $v$, corresponding to the split $sp_*$. We have:
          $\begin{cases} \tilde{tr}(I, v_R) &= I^c_{sp_*} \cap \tilde{tr}(I, v), \\ V &= V \bigcup \{v_R\}, \\ A &= A \bigcup \{v v_R\}. \end{cases}$
        - Update $nextv = nextv + 2$.
    • Update $v$: $v = v + 1$.
3. Outputs: $(tr, \tilde{tr}, f)$.
```

Definition of the Impurity. We propose to work with the impurity criterion based on the norm l^2. We define, $\forall v \in V$,

$$Q(v) = \frac{|\tilde{tr}(I,v)|}{|I|} \sum_{\tau \in \tilde{tr}(I,v)} (d_\tau - a_v - T_v \times \tau)^2.$$

This choice for the impurity Q defines the linear UniCart. The definition of a_v and T_v leads to two different versions of the linear UniCART. Version II.5 returns a multiscale linear approximation purely based on regression, which implies a gap between consecutive trends. On the other hand, version II.6 refers to an interpolation approach and produces a continuous approximation:

1. at each node v, compute a_v and T_v so that

$$(a_v, T_v) = \underset{a\in\mathbb{R}, T\in\mathbb{R}}{\arg\min} \sum_{\tau \in \tilde{tr}(I,v)} (d_\tau - a - T \times \tau)^2. \tag{II.5}$$

Denoting

$$\begin{aligned} \tau_{min,v} &= \min\{\tau, \tau \in \tilde{tr}(I,v)\}, \\ \tau_{max,v} &= \max\{\tau, \tau \in \tilde{tr}(I,v)\}, \\ \Delta_v &= (\tau_{max,v} - \tau_{min,v} + 1), \\ M_v &= \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & \Delta_v \end{pmatrix}, \end{aligned}$$

we have:

$$\begin{aligned} T_v &= (M_v^\top M_v)^{-1} M_v (d_\tau)_{\tau \in \tilde{tr}(I,v)}, \\ a_v &= d_{\tau_{max,v}} - T_v \tau_{max,v}. \end{aligned}$$

2. at each node v, considering the first and last time indices of $\tilde{tr}(I,v)$, $\tau_{min,v}$ and $\tau_{max,v}$, we have the following relationship:

$$\begin{aligned} d_{\tau_{min,v}} &= a_v + T_v \tau_{min,v}, \\ d_{\tau_{max,v}} &= a_v + T_v \tau_{max,v}, \end{aligned}$$

which implies

$$\begin{aligned} a_v &= \frac{d_{\tau_{min,v}} \tau_{max,v} - d_{\tau_{max,v}} \tau_{min,v}}{\tau_{max,v} - \tau_{min,v}}, \\ T_v &= \frac{d_{\tau_{max,v}} - d_{\tau_{min,v}}}{\tau_{max,v} - \tau_{min,v}}. \end{aligned} \tag{II.6}$$

In the rest of the document, we uniquely focus on version II.6 of the linear UniCart, while still referring to UniCart. Particularly, the linear trends mentioned in the following are the interpolated linear trends $(T_v)_{v\in V}$. Their computation (hence performed as in II.6) is illustrated in the next paragraph.

Illustration. UniCart is applied to the case of the daily **SP500** considered on a local window of $W = 1000$ open days. This window encompasses the period from 01/05/1971 to 24/01/1974. The minimal node size under which the segmentation stops is $w = 5$days. The number of samples used in the tree-pruning phase equals 50. This latter phase is quite time consuming. It is also superfluous if our main concern is to detect the most important trend changes in the time window. In such a case, the choice of the best scale is rather controlled by the minimal node size than by the pruning procedure. Figures II.31, II.32, II.33, II.34, II.35 and II.36 illustrate the flexibility and the simplicity of the method.

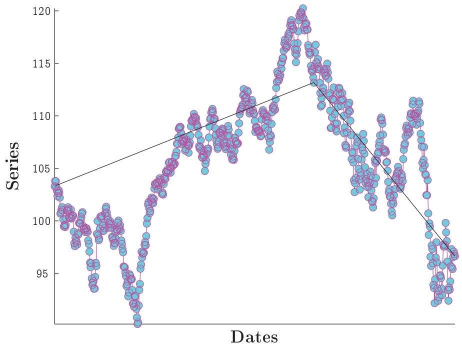


Figure II.31: **Rough Scale Segmentation**

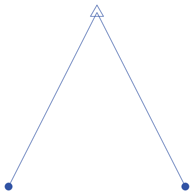


Figure II.32: **Rough Scale Tree**

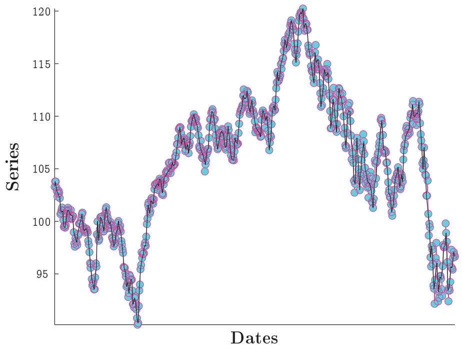


Figure II.33: **Finest Scale Segmentation**

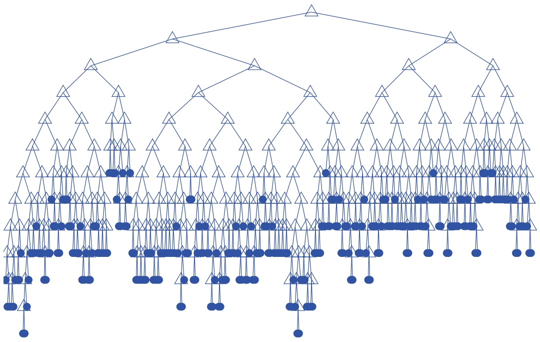


Figure II.34: **Finest Scale Tree**

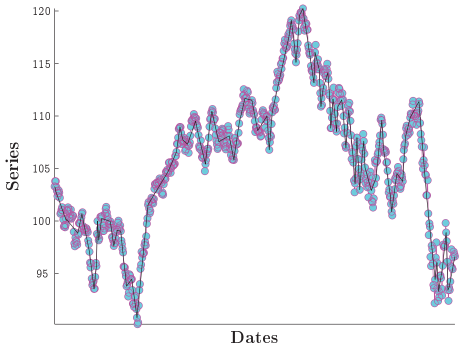


Figure II.35: **Best Scale Segmentation**

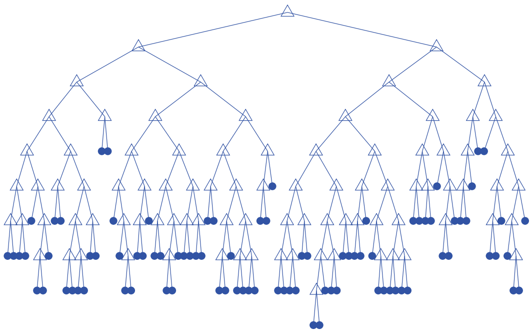


Figure II.36: **Best Scale Tree**

As a conclusion concerning the UniCart procedure, note that there are actually many ways of using the results of the multiscale approximation to compute features, which are not only necessarily linear trends. In the next section, we focus particularly on two types of databases.

II.2 Construction of UniCart Databases

The UniCart procedure does not directly produce features and labels for training numerical models. Instead, it delivers a multiscale representation of finite series. This section introduces methods for transforming this multiscale representation into features and labels. It is organized as follows. It presents first the intangible database construction principle, which consists of applying UniCart to the explanatory variables considered over a sliding window. Two types of databases are then introduced. Each of them contains a family of interpretable and intuitive explanatory features, computed via a specific processing of the UniCart representation. The statistical characteristics of these features, particularly their homogeneity and their multiscale nature, are illustrated and compared via application examples.

II.2.1 Principle of Construction

Methodology and Notations - Features Extraction. The database construction is based on a sliding time window, which moves across the historical base along the time range of the target variable according to its sampling step. At each step, a windowed series is extracted for each explanatory variable and for the target variable by considering each of these variables in a time range corresponding to the time window. UniCart is then applied over each windowed series. It produces a multiscale linear segmentation. Note that the variables do not have to be sampled at the same frequency. We introduce the following notations. As in subsection I.3.1, we consider the set of p finite financial series:

$$D = (D_t^1, \dots, D_t^p)_{t=1,\dots,n_D}.$$

By convention, D^1 is the target variable. Besides, $D^2, \dots, D^p$ are the explanatory variables. Let

$$W \ll n_D$$

be the size of the sliding time window. As in subsection II.1.4, the minimal subset size below which the segmentation stops is denoted

$$w < W.$$

For each variable i and each sample date t, the windowed series $(D_t^i)_{t-W+1\leq u\leq t}$ is converted via the extraction features function ϕ_X to a set of features $x_t^i = (x_{t,v}^i)_{v\in V_t^i}$. This set is indexed by the nodes of the UniCart tree $tr_t^i = (V_t^i, A_t^i)$. For simplicity, the corresponding partition and approximation functions are denoted respectively $\tilde{tr}_t^i(v) := \tilde{tr}_t^i(\{t-W+1,\dots,t\}, v)$ and $f_{t,v}^i$. We have:

$$(D_t^i)_{t-W+1\leq u\leq t} \overset{\phi_X}{\to} x_t^i = (x_{t,v}^i)_{v\in V_t^i}.$$

Yet, handling all the features of each tree would be computationally infeasible and not necessarily relevant for learning efficient models. Besides, the number of features may not be constant from a given time window to another, which could be a problem for the database construction. It is therefore wiser to consider only a subset of $x_t^i = (x_{t,v}^i)_{v\in V_t^i}$. Our solution is to re-index the set of features x_t^i according to their vertical position in the tree, referred to as the scale

$$s = s(i, t, W, w) \leq S(i, t, W, w),$$

and according to their horizontal position, referred to as the lag

$$k = k(s, i, t, W, w) \leq K(s, i, t, W, w).$$

Note that $S(i,t,W,w)$ is the depth of the tree tr_t^i and $K(s,i,t,W,w)$ the number of nodes at each level s. We select then for each set of features $x_t^i = (x_{t,v}^i)_{v\in V_t^i}$ a maximal number of scales and, for each scale, an associated maximal number of lags. With a slight abuse of notation, we still denote it:

$$x_t^i = (x_{t,v}^i)_{v\in V_t^i} = (x_{t,s,k}^i)_{s\leq S, k\leq K}.$$

Likewise, any variable indexed by the nodes $v \in V$ are equivalently indexed by the couple (s,k), with $s \leq S$ and $k \leq K$.

Methodology and Notations - Label Extraction. Among features, linear trends are now denoted

$$T_t^i = (T_{t,s,k}^i) \subset x_t^i.$$

They naturally allow computation of binary labels. The user is invited to select a scale $s^\star$. The most recent lag at scale $s^\star$ is denoted $k^\star$, while the horizon of prediction is h. The binary labels are then:

$$y_t(h) = \mathrm{sgn}(T_{t+h,s^\star,k^\star}^1 - T_{t,s^\star,k^\star}^1).$$

As with features, this computation is denoted:

$$(D_t^i)_{t-W+1\leq u\leq t} \overset{\phi_Y}{\to} y_t(h).$$

Basically, it means that the user chooses for himself the scale at which labels are computed. Finally, The UniCart database is composed of both features and labels, which is written:

$$(x_t^1, \dots, x_t^p, y_t(h))_{W\leq t\leq n_D-h}$$

Pseudocode - Features and Label Extraction. To sum up, the database construction embeds the windowed series

$$(D_t^i)_{1\leq i\leq p, t-W+1\leq u\leq t}$$

indexed by

$$W \leq t \leq n_D - h$$

into the sequence of features

$$(x_t^i)_{1\leq i\leq p, W\leq t\leq n_D-h},$$

where

$$x_t^i = (x_{t,s,k}^i)_{s\leq S,k\leq K}.$$

This sequence of features is completed by the labels

$$(y_t(h))_{W\leq t\leq n_D-h},$$

where

$$y_t(h) = \mathrm{sgn}(T_{t+h,s^\star,k^\star}^1 - T_{t,s^\star,k^\star}^1).$$

where $s^\star$ is user-defined, $k^\star$ is the most recent lag at scale $s^\star$ and h is the horizon of prediction. The database is thus expressed as:

$$(x_t, y_t(h)),$$

where

$$W \leq t \leq n_D - h.$$

The following pseudocode is then straightforward:

```
1. While $W \leq t \leq n_D - h$,
    • Compute:
        $$(D_t^i)_{t-W+1\leq u\leq t} \overset{\phi_X}{\to} x_t^i = (x_{t,s,k}^i)_{s\leq S,k\leq K}.$$
```

- *Compute:*
$$(D_t^i)_{t-W+1\leq u\leq t} \overset{\phi_Y}{\to} y_t(h).$$
- *Update t: $t = t+1$.*

2. *Outputs:* $(x_t, y_t(h))_{W\leq t\leq n_D-h}$.

With a slight abuse of notation, we write the neat relation stated in the experimentation protocol (cf. subsection I.3.1):
$$\begin{cases} D \overset{\phi_X}{\to} x \\ D \overset{\phi_Y}{\to} y. \end{cases}$$
In the next subsection, two methods for computing the features x are made explicit.

II.2.2 Two Types of Databases

Multiscale Trend based Returns vs. Single-scale Trend based Features. Consider as previously the windowed series $(D_t^i)_{t-W+1\leq u\leq t}$. As mentioned earlier, it is necessary that the number of features remain constant over the shifts of the sliding window. Therefore, the two computing features methods, which are now introduced, require that the user enter both a maximal number of scales $\tilde{S}$ and a maximal number of lags for each scale, denoted $\tilde{K} = \tilde{K}(s)$. Only the features whose scale s and lag k check:
$$\begin{cases} s & \leq \tilde{S} \\ k & \leq \tilde{K} \end{cases}$$
are taken into consideration. Note also that the number of scales $S(i,t,W,w)$ is not necessarily larger than the maximal number of scales $\tilde{S}$. This remark holds true for $K(i,t,W,w,s)$ and $\tilde{K}$. By convention, if $S(i,t,W,w) < \tilde{S}$ or $K(i,t,W,w,s) < \tilde{K}$, the available features are selected, while the remaining features are all virtually set to 0. In practice, $\tilde{S}$ and $\tilde{K}$ are small. We distinguish between two different types of databases:

- a first type of database $(x_t, y_t(h))_{W\leq t\leq n_D-h}$ is composed of features, which are all considered at the finest scale available $s^\star$. The features x_t^i are made of the linear trends $(T^i_{t,s^\star,k})_{1\leq k\leq \tilde{K}}$ and of corresponding statistical quantities. For $1 \leq k \leq \tilde{K}$, these latter are the duration $Dur^i_{t,s^\star,k}$ of the trend $T^i_{t,s^\star,k}$, the variance $Var^i_{t,s^\star,k}$ of the series over the trend, the skewness $Skew^i_{t,s^\star,k}$, the kurtosis $Kurt^i_{t,s^\star,k}$ as well as the minimal and maximal distances $Min^i_{t,s^\star,k}$ and $Max^i_{t,s^\star,k}$ of the series to the trend. To define these quantities, we denote $\mathbb{V}ar$, $\mathbb{S}kew$, $\mathbb{K}urt$, $\mathbb{M}in$ and $\mathbb{M}ax$ the usual operators computing respectively the variance, the skewness, the kurtosis, the minimum and the maximum of a given random variable. The corresponding formulas are:
$$\begin{cases} Dur^i_{t,s^\star,k} & = \left|\tilde{tr}_t(s^\star,k)^i\right| = \tau^i_{max,s^\star,k} - \tau^i_{min,s^\star,k} + 1 \\ Var^i_{t,s^\star,k} & = \mathbb{V}ar_{\tau\in\tilde{tr}_t(s^\star,k)^i}\left[D^i_\tau - a^i_{t,s^\star,k} - T^i_{t,s^\star,k}\times\tau\right] \\ Skew^i_{t,s^\star,k} & = \mathbb{S}kew_{\tau\in\tilde{tr}_t(s^\star,k)^i}\left[D^i_\tau - a^i_{t,s^\star,k} - T^i_{t,s^\star,k}\times\tau\right]\mathbb{1}_{Var^i_{t,s^\star,k}>0} \\ Kurt^i_{t,s^\star,k} & = \mathbb{K}urt_{\tau\in\tilde{tr}_t(s^\star,k)^i}\left[D^i_\tau - a^i_{t,s^\star,k} - T^i_{t,s^\star,k}\times\tau\right]\mathbb{1}_{Var^i_{t,s^\star,k}>0} \\ Min^i_{t,s^\star,k} & = \mathbb{M}in_{\tau\in\tilde{tr}_t(s^\star,k)^i}\left[D^i_\tau - a^i_{t,s^\star,k} - T^i_{t,s^\star,k}\times\tau\right] \\ Max^i_{t,s^\star,k} & = \mathbb{M}ax_{\tau\in\tilde{tr}_t(s^\star,k)^i}\left[D^i_\tau - a^i_{t,s^\star,k} - T^i_{t,s^\star,k}\times\tau\right]. \end{cases}$$
This type of database requires the user to choose a number $\tilde{K}$ of most recent lags to select at the scale $s^\star$.

- a second type of database is made of returns exclusively, considered at the scales $\{1, \ldots, \tilde{S}\}$ and for the most recent lag 1 only. For $1 \leq s \leq \tilde{S}$, these latter are the returns $Ret^i_{t,s,k}$, the ratios $Sh^i_{t,s,k}$ and the ratios $So^i_{t,s,k}$. The corresponding formulas are:

$$\begin{cases} Ret^i_{t,s,k} & = \left[\dfrac{D^i_{\tau_{max,s,k}}}{D^i_{\tau_{min,s,k}}}\right]^{\frac{250}{Dur^i_{t,s,k}}} - 1 \\ \Sigma^i_{t,s,k} & = \sqrt{\mathbb{V}ar_{\tau \in \tilde{tr}_t(s,k)^i}\left[\dfrac{D^i_{\tau+1,s,k}}{D^i_{\tau,s,k}} - 1\right]} \\ Sh^i_{t,s,k} & = \dfrac{Ret^i_{t,s,k}}{\sqrt{250}\Sigma^i_{t,s,k}}, \\ Ret^i_{t,s,k,0} & = \left[\dfrac{D^i_{\tau_{max,s,k}}}{D^i_{\tau_{min,s,k}}}\right]^{\frac{1}{Dur^i_{t,s,k}}} - 1 \\ DR^i_{t,s,k} & = \dfrac{250}{Dur^i_{t,s,k}} \sum_{\tau \in \tilde{tr}_t(s,k)^i} \left[\dfrac{D^i_{\tau+1,s,k}}{D^i_{\tau,s,k}}\right]^2 \mathbb{1}_{\frac{D^i_{\tau+1,s,k}}{D^i_{\tau,s,k}} < 0} \\ So^i_{t,s,k} & = \dfrac{Ret^i_{t,s,k,0}}{DR^i_{t,s,k}}. \end{cases}$$

The returns $Sh^i_{t,s,k}$ and $So^i_{t,s,k}$ are actually returns divided by a measure of noise, being respectively inspired of Sharpe ratios and Sortino ratios. This database is composed of multiscale returns, usually more stationary than the linear trends $T^i_{t,s,k}$.

The computation of the two databases is illustrated in the next paragraph.

Construction of the two Databases: Illustrations. The principle of construction of the two databases is now made clear using the set of explanatory variables presented in subsection II.1.1. As explained in the fourth chapter, this set has been constituted in the perspective of predicting the trends of the variable **SP500**. All these variables have not been available since the same date and at the same frequency. For example, **SP500** has been sampled at a daily rate since 03/01/1950 (following the convention dd/mm/yyyy), whereas **MORTG** is a monthly variable that has been available since 01/04/1971. Despite these differences, as explained in the previous paragraphs, the principle of trend determination and collection is alike for all series.

In practice, in our implementation, a sliding time window is defined for each variable of the set. As each series is likely to start from a different date, the sliding time windows start from the latest of them all. This defines a start date, which corresponds to $u = 1$. The end date corresponds to the date at which the value D^1_W of the target variable **SP500** is available. The shifts of the windows across the series are coordinated. They are guided by the shift of the window of the target variable **SP500**. When this leading window moves along the time range of **SP500**, it defines a specific time sub-part. The latest date of this sub-part, which is the current end date, defines the positioning of the other windows. Their relative end dates must be less recent than or as recent as the leading window latest date.

We display the following graphs to give a clear idea of the sliding windows' relative positioning at the beginning of the features collection. The first sliding windows of two daily variables, **SP500** and **DGS10**, and of two monthly variables, **UNEMPLOY** and **PER**, are thus reproduced in the figures II.37, II.38, II.39 and II.40. The size of the sliding windows is fixed at $W = 1000$ days. We choose a very broad window so that it is easier to see on graphs. The start date of all "first" sliding windows, that is the latest date among the first dates of each variable time range, is 01/05/1971. The last date of the **SP500** time window is 24/01/1974, i.e. $W = 1000$ days after 01/05/1971. It means that the last dates of all other sliding windows are less recent or as recent as 24/01/1974.

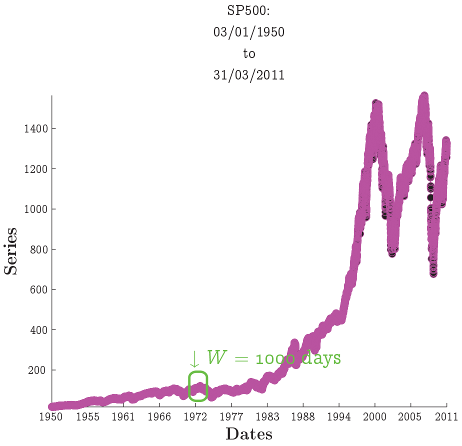


Figure II.37: **SP500 - W**

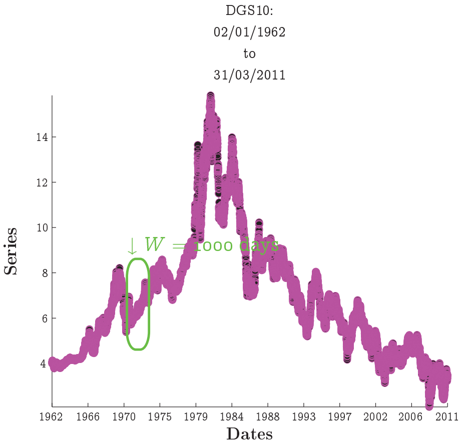


Figure II.38: **DGS10 - W**

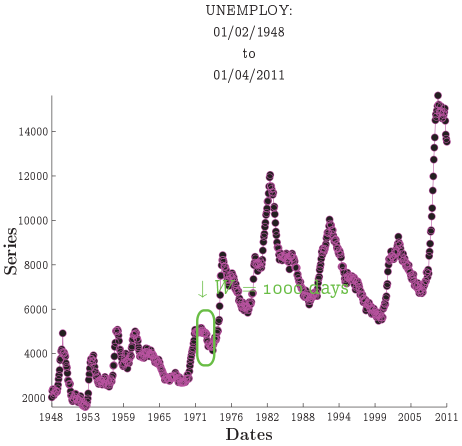


Figure II.39: **UNEMPLOY - W**

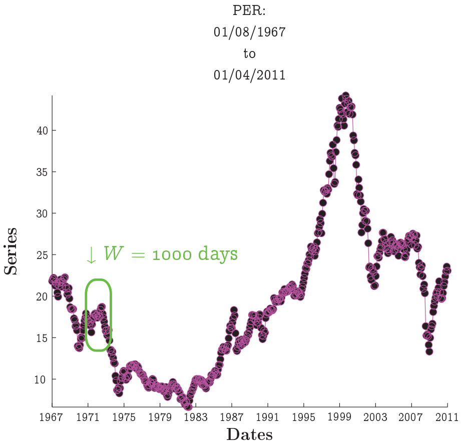


Figure II.40: **PER - W**

Once the relative positioning of the sliding windows has been completed, the UniCart segmentation procedure is applied to each of them. In the case of the first type of database, the $\tilde{K}$ lagged features of the finest scale are selected. The next graphs zoom on the previous sliding windows (cf. figures II.41, II.43, II.45 and II.47). Figures II.42, II.44, II.46 and II.48 show the UniCart approximation produced on this time window for the variables **SP500**, **DGS10**, **UNEMPLOY** and **PER** with the following parameters:

$$\begin{cases} W & = 1000\,\text{days} \\ w & = 100\,\text{days} \\ \tilde{S} & = s^{\star}\,\text{scale levels} \\ \tilde{K} & = 5\,\text{consecutive trend lags.} \end{cases}$$

Besides, figures II.49, II.50, II.51, II.52 represent the features obtained for these variables over the whole historical range (i.e. via the iterated application of UniCart on the sliding window, which is shifted over this range).

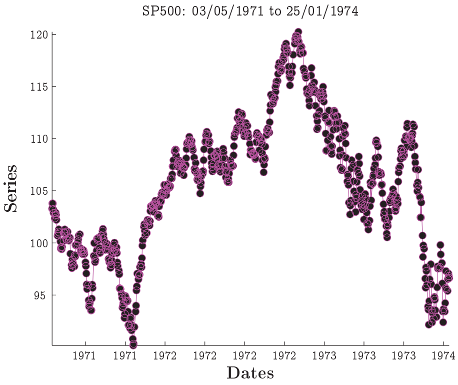


Figure II.41: **SP500 Zoom on W**

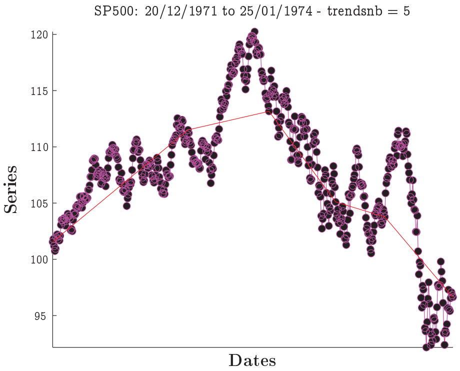


Figure II.42: **SP500 - UniCart on W**

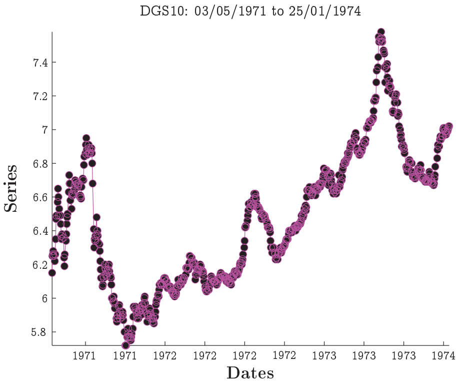


Figure II.43: **DGS10 Zoom on W**

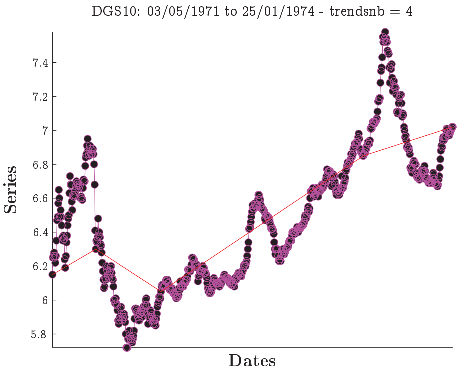


Figure II.44: **DGS10 - UniCart on W**

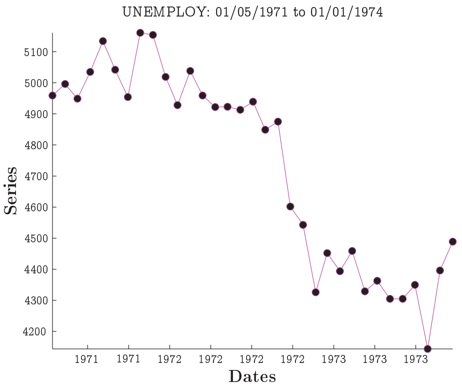


Figure II.45: **UNEMPLOY Zoom on W**

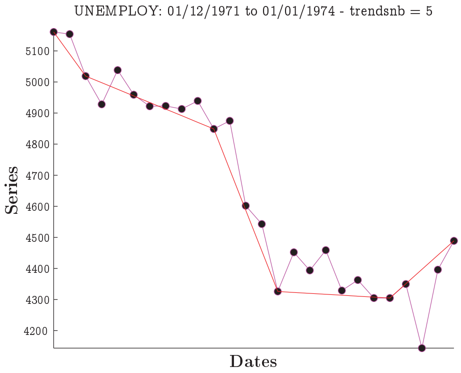


Figure II.46: **UNEMPLOY - UniCart on W**

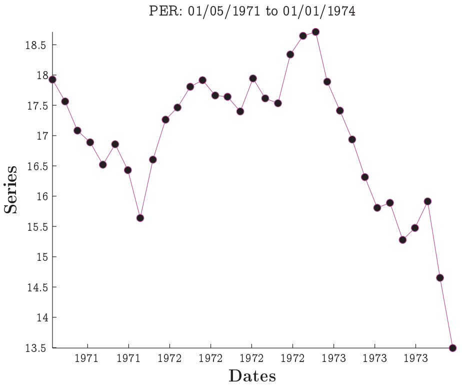


Figure II.47: **PER Zoom on W**

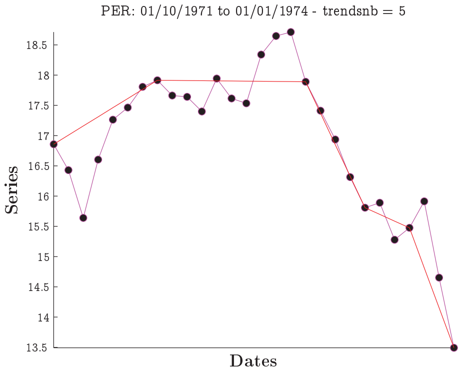
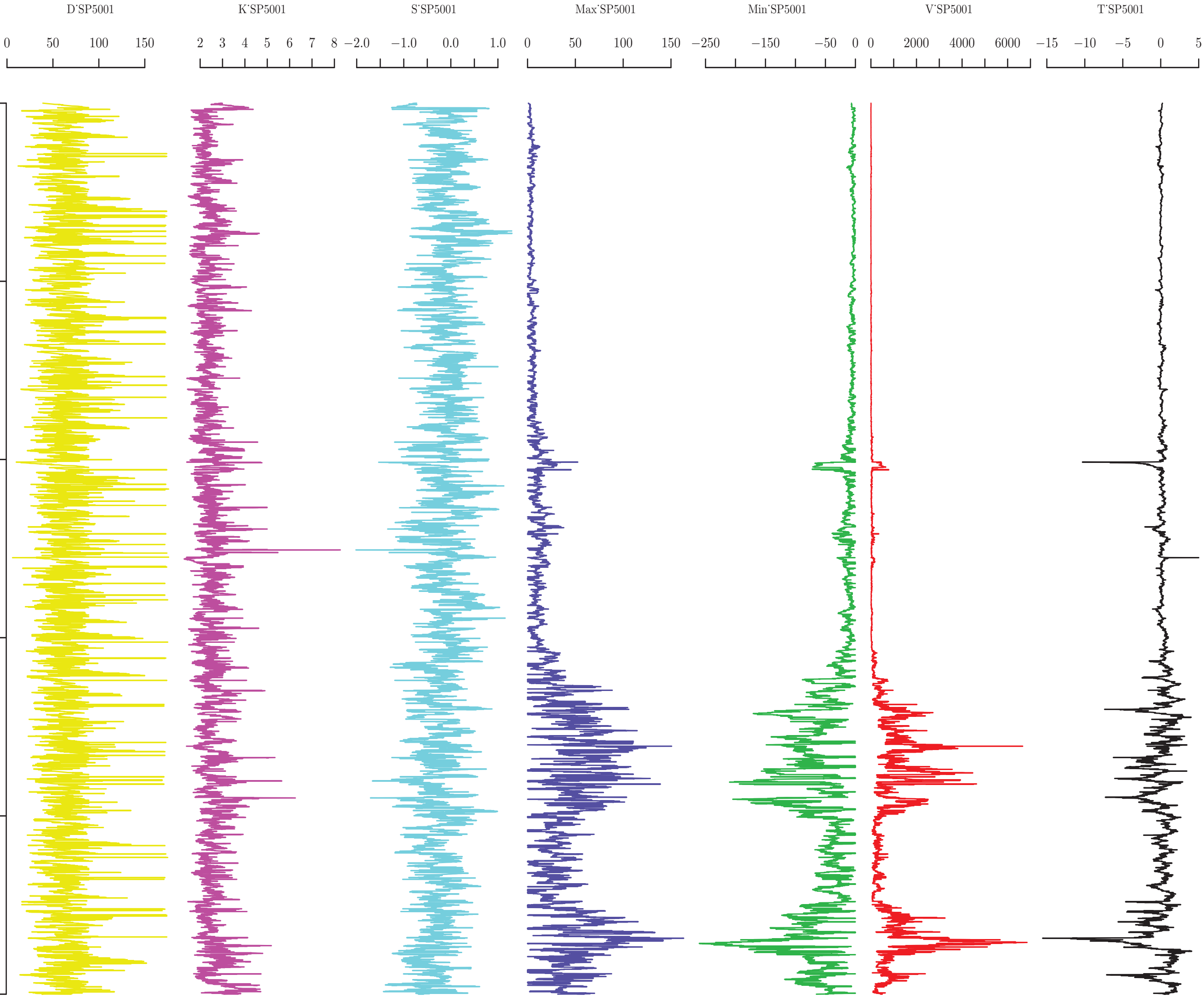
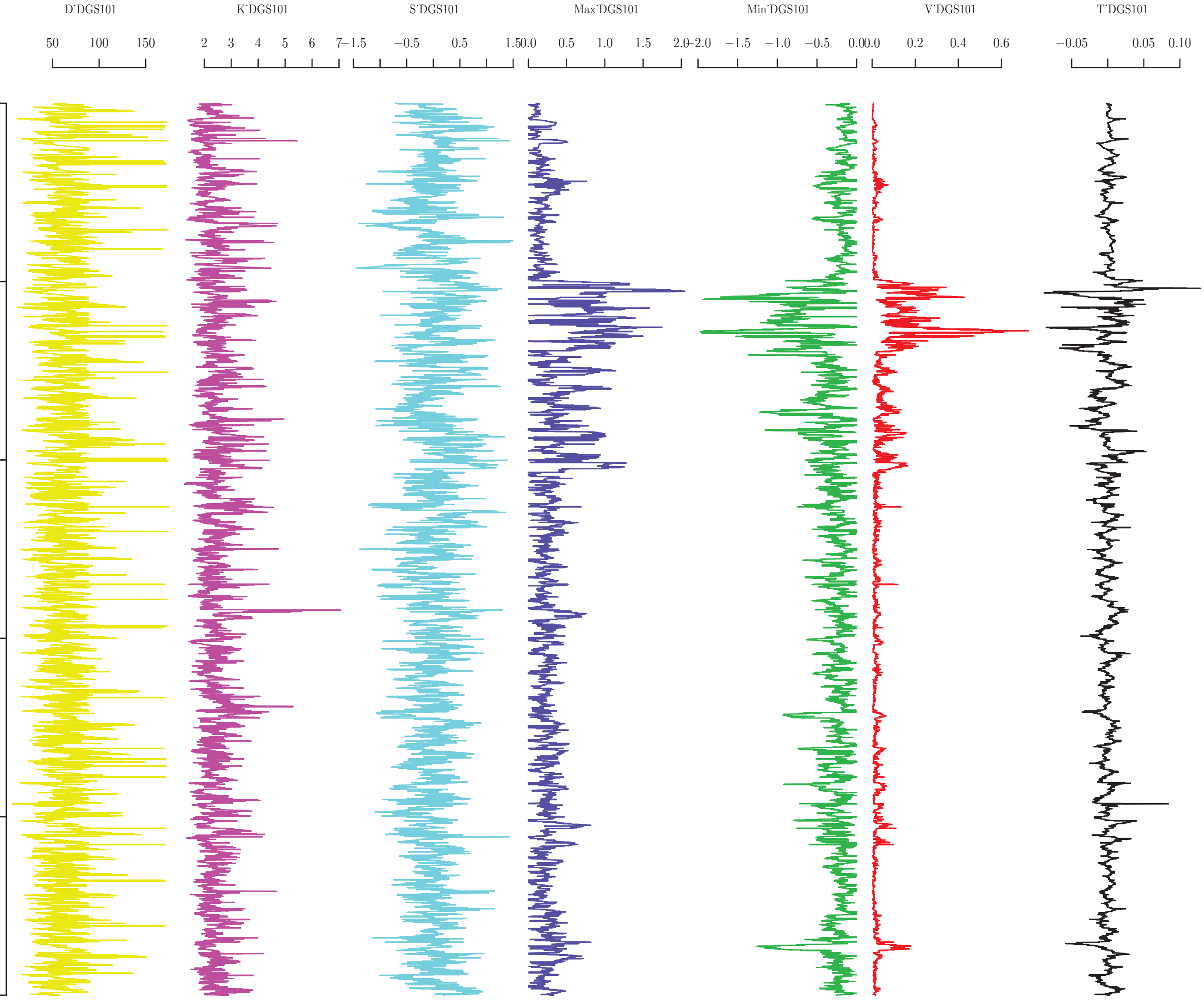
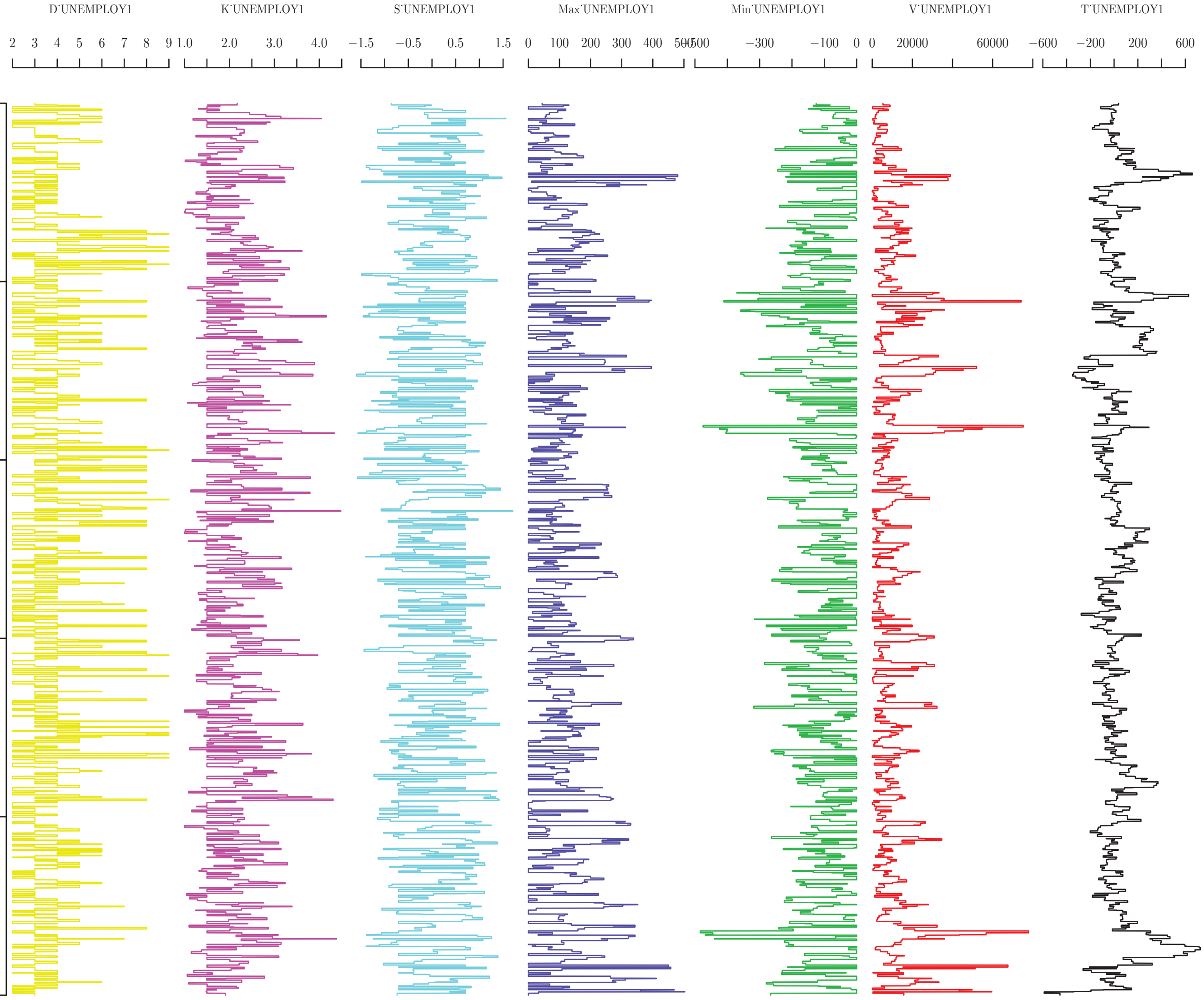
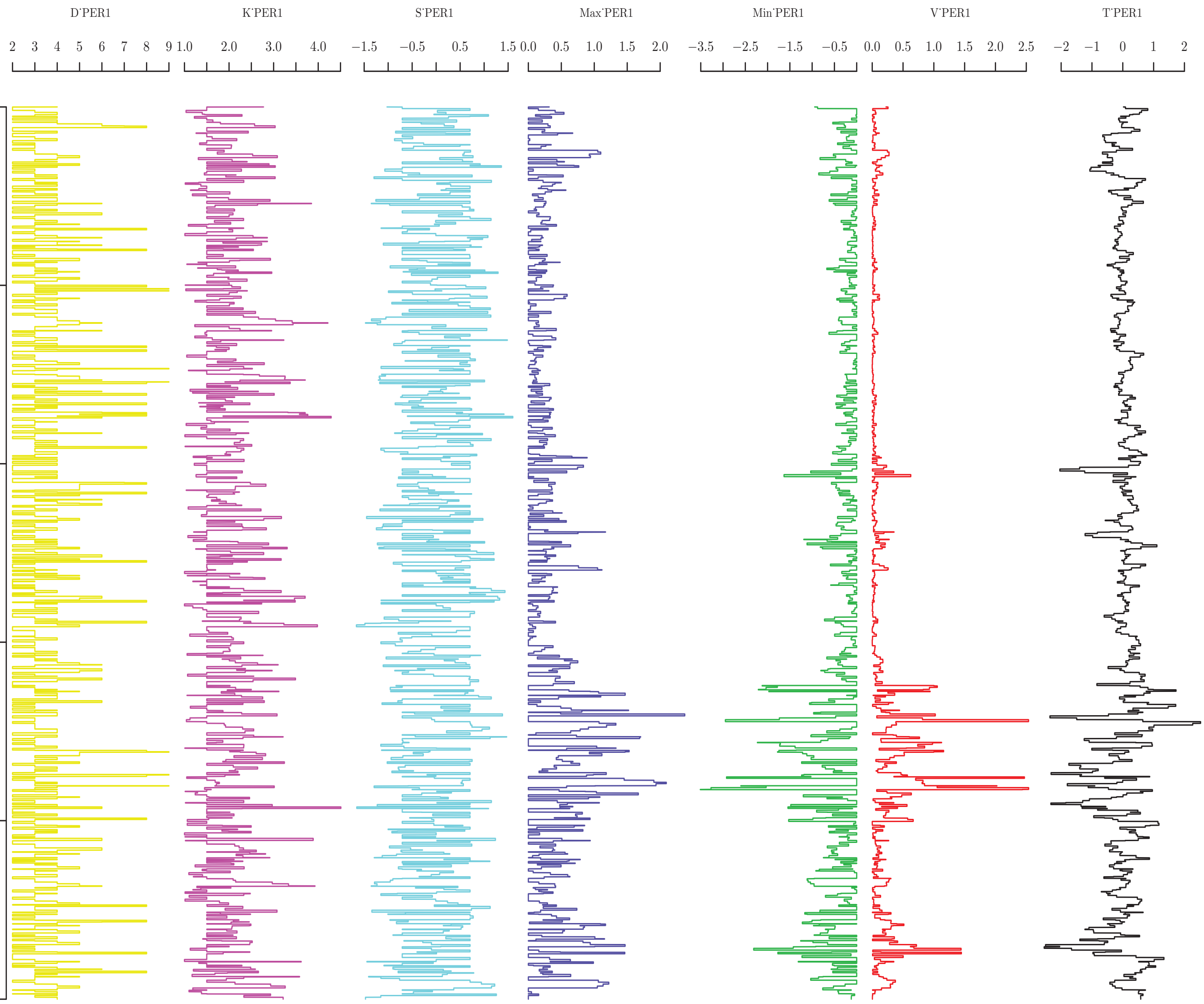


Figure II.48: **PER - UniCart on W**


D˙SP5001
0
50
100
150
K˙SP5001
2
3
4
5
6
7
8
S˙SP5001
−2.0
−1.0
0.0
1.0
Max˙SP5001
0
50
100
150
Min˙SP5001
−250
−150
−50
0
V˙SP5001
0
2000
4000
6000
T˙SP5001
−15
−10
−5
0
5


D˙DGS101
50 100 150
K˙DGS101
2 3 4 5 6 7
S˙DGS101
−1.5 −0.5 0.5 1.5
Max˙DGS101
0.0 0.5 1.0 1.5 2.0
Min˙DGS101
−2.0 −1.5 −1.0 −0.5 0.0
V˙DGS101
0.0 0.2 0.4 0.6
T˙DGS101
−0.05 0.05 0.10


D˙UNEMPLOY1
K˙UNEMPLOY1
S˙UNEMPLOY1
Max˙UNEMPLOY1
Min˙UNEMPLOY1
V˙UNEMPLOY1
T˙UNEMPLOY1
2
3
4
5
6
7
8
9
1.0
2.0
3.0
4.0
−1.5
−0.5
0.5
1.5
0
100
200
300
400
500
−500
−300
−100
0
0
20000
60000
−600
−200
200
600


D˙PER1
2
3
4
5
6
7
8
9
K˙PER1
1.0
2.0
3.0
4.0
S˙PER1
−1.5
−0.5
0.5
1.5
Max˙PER1
0.0
0.5
1.0
1.5
2.0
Min˙PER1
−3.5
−2.5
−1.5
−0.5
V˙PER1
0.0
0.5
1.0
1.5
2.0
2.5
T˙PER1
−2
−1
0
1
2

In the case of the second type of database, features of lag 1 and of scales $\{1, \dots, \tilde{S}\}$ are selected. Again, the next graphs zoom on the previous sliding windows. They show the results of the UniCart segmentation performed on **SP500** uniquely (cf. figures II.54, II.55, II.56, II.57 and II.58) with the following parameters:

$$\begin{cases} W & = 1000 \text{ days} \\ w & = 30 \text{ days} \\ \tilde{S} & = 5 \text{ scale levels} \\ \tilde{K} & = 1 \text{ trend lag.} \end{cases}$$

As an example, figure II.59 finally displays the **SP500** features computed over the whole historical range.

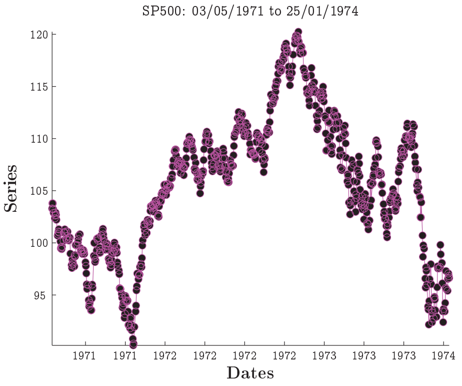


Figure II.53: **SP500 Zoom on W**

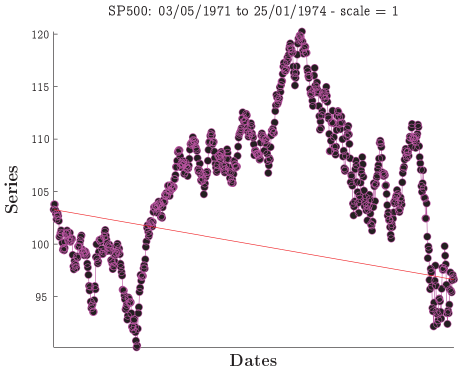


Figure II.54: $s = 1, k = 1$

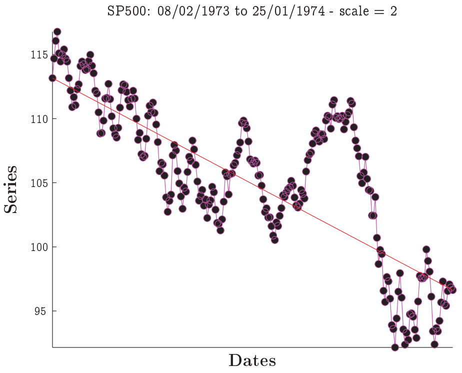


Figure II.55: $s = 2, k = 1$

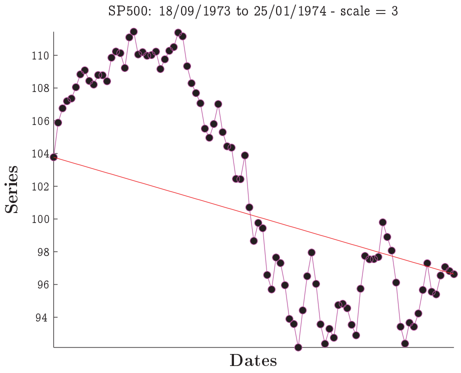


Figure II.56: $s = 3, k = 1$

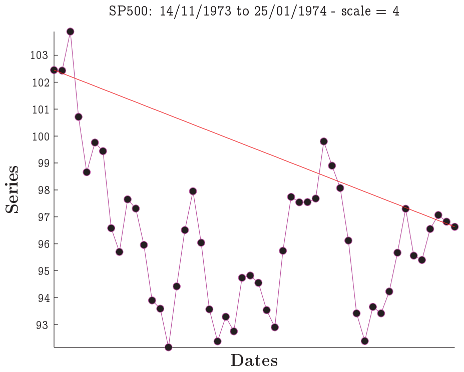


Figure II.57: $s = 4, k = 1$

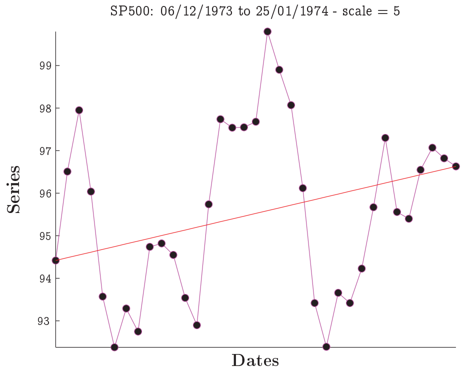
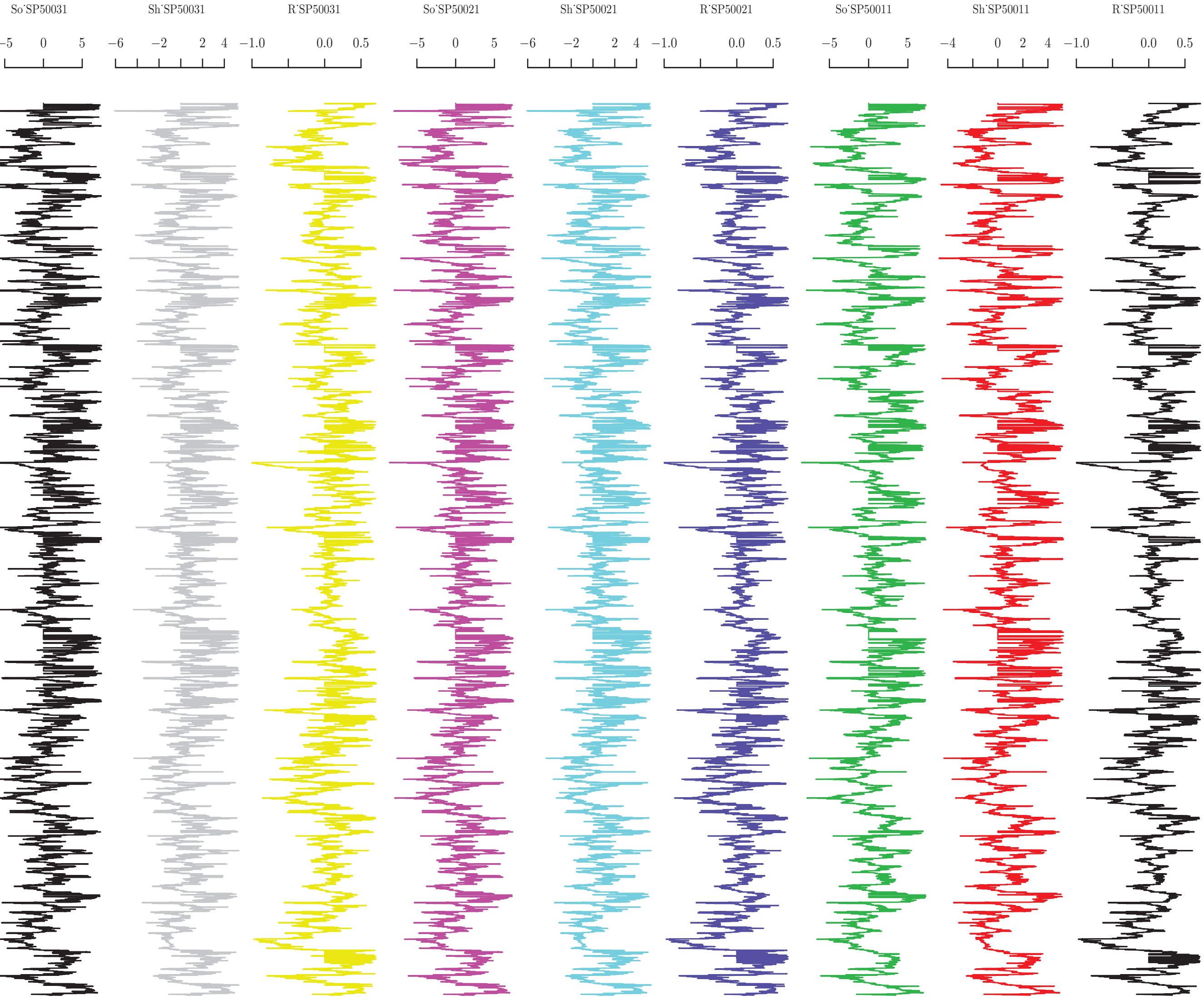


Figure II.58: $s = 5, k = 1$

The two types of databases depend on the size W of the sliding window, on the minimal segment size $w < W$, on a limited number number $\tilde{S}$ of scale levels as well as on a limited number $\tilde{K}$ of trend lags, and on the horizon h of prediction. They are illustrated here with parameter values, which enable a clear introduction to the database construction task. In the applications of the fourth chapter, the size of the sliding window is reduced to $W = 250$ days. The stationarity of the features of the two types of databases is studied in the next paragraph.


So˙SP50031
Sh˙SP50031
R˙SP50031
So˙SP50021
Sh˙SP50021
R˙SP50021
So˙SP50011
Sh˙SP50011
R˙SP50011
−5 0 5
−6 −2 2 4
−1.0 0.0 0.5
−5 0 5
−6 −2 2 4
−1.0 0.0 0.5
−5 0 5
−4 0 2 4
−1.0 0.0 0.5

Stationarity of the Features in two Types of Databases. Once a choice has been made about the type of the database, and the database has been computed, any classical machine learning procedure can be used to train the prediction function. Yet, a last and important prerequisite concerns the stationarity of the features. We anticipate here the backtesting results of chapter IV. Each of the two types of **SP500** UniCart databases has been split into a training set, made of features corresponding to the historic range from 01/05/1971 to 31/03/2006, and into a test set, which starts at 31/03/2006 and ends at 31/03/2011. The boxplots II.60, II.61, II.62 and II.63 graphically compare the quartiles of specific features' distributions for the training set and the test set of the first database. They show that some of these features are not homogeneous. Particularly, the **SP500** linear trends do not have the same distribution support. This is a problem for learning an accurate numerical model because, as underlined in subsection IV.2.2 of chapter IV, this feature plays an important role in the prediction. In consequence, the influence of this feature has motivated the introduction of more homogeneous - and trend-related - features, hence the creation of the second type of database. The boxplot II.64 highlights that these return-based features are much more stationary. In addition, the TreeRank based method for testing the homogeneity of two samples [CVD09] has been applied to our problem of comparison of features' distributions. The results of this test confirm this graphically observed increase in stationarity but also reject the homogeneity hypothesis even at low levels. A model construction procedure capable of handling non- stationarities is therefore needed. A new transfer learning algorithm, called Relabeled Nearest Neighbors, is thus introduced in the subsection III.3 of the next chapter.

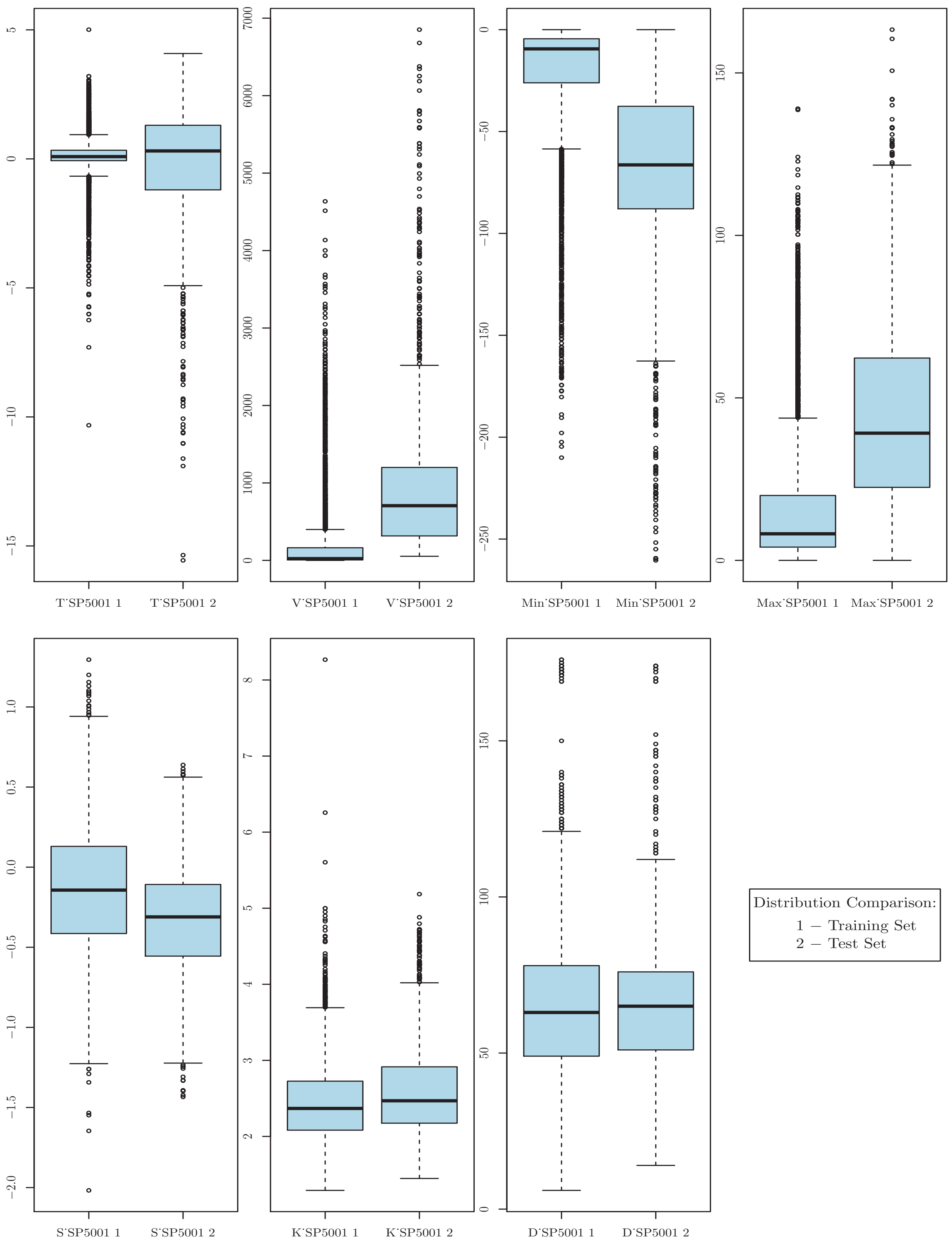


Figure II.60: **Comparison of SP500 (lag 1) Features Distributions, 1st DB**

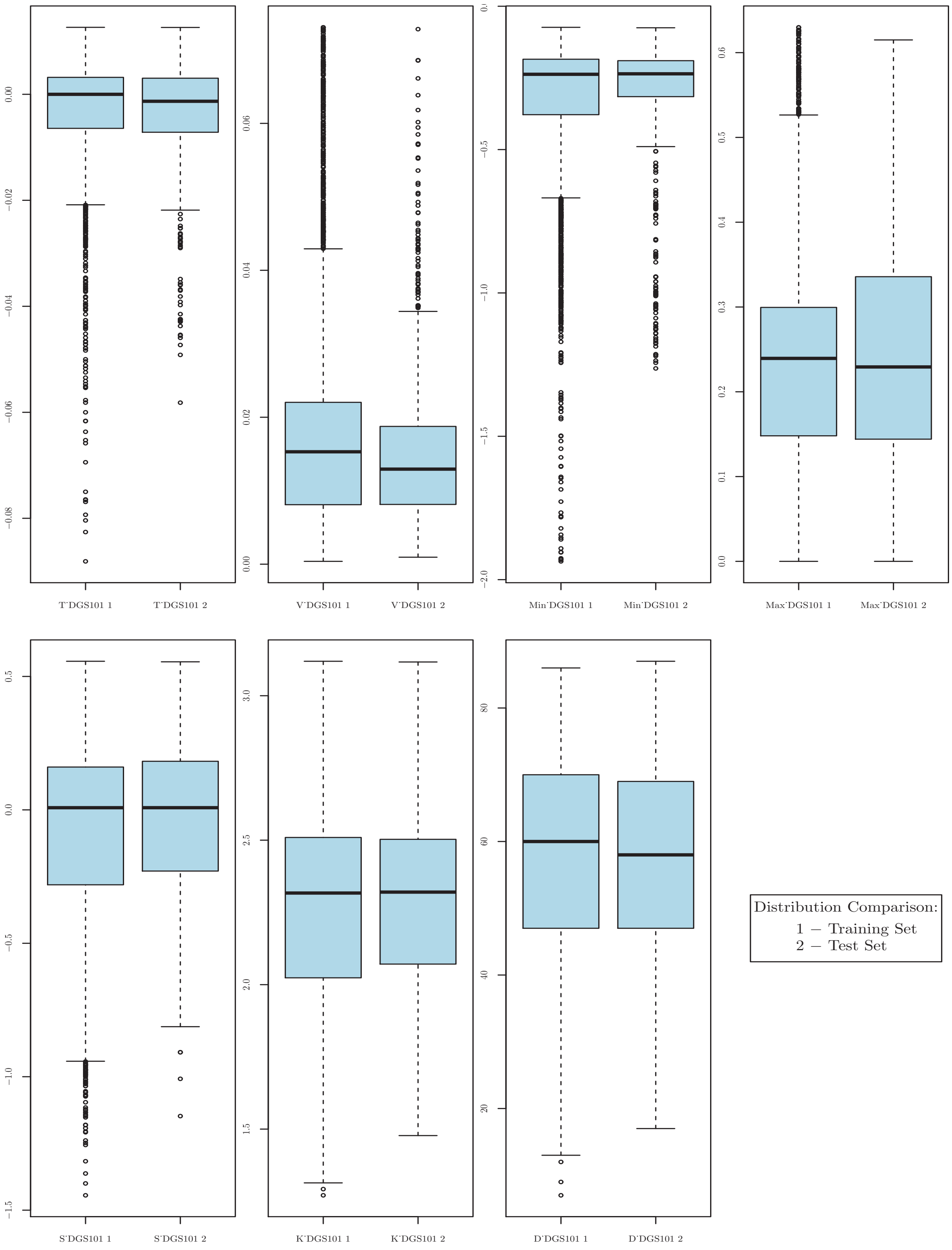


Figure II.61: **Comparison of DGS10 (lag 1) Features Distributions, 1st DB**

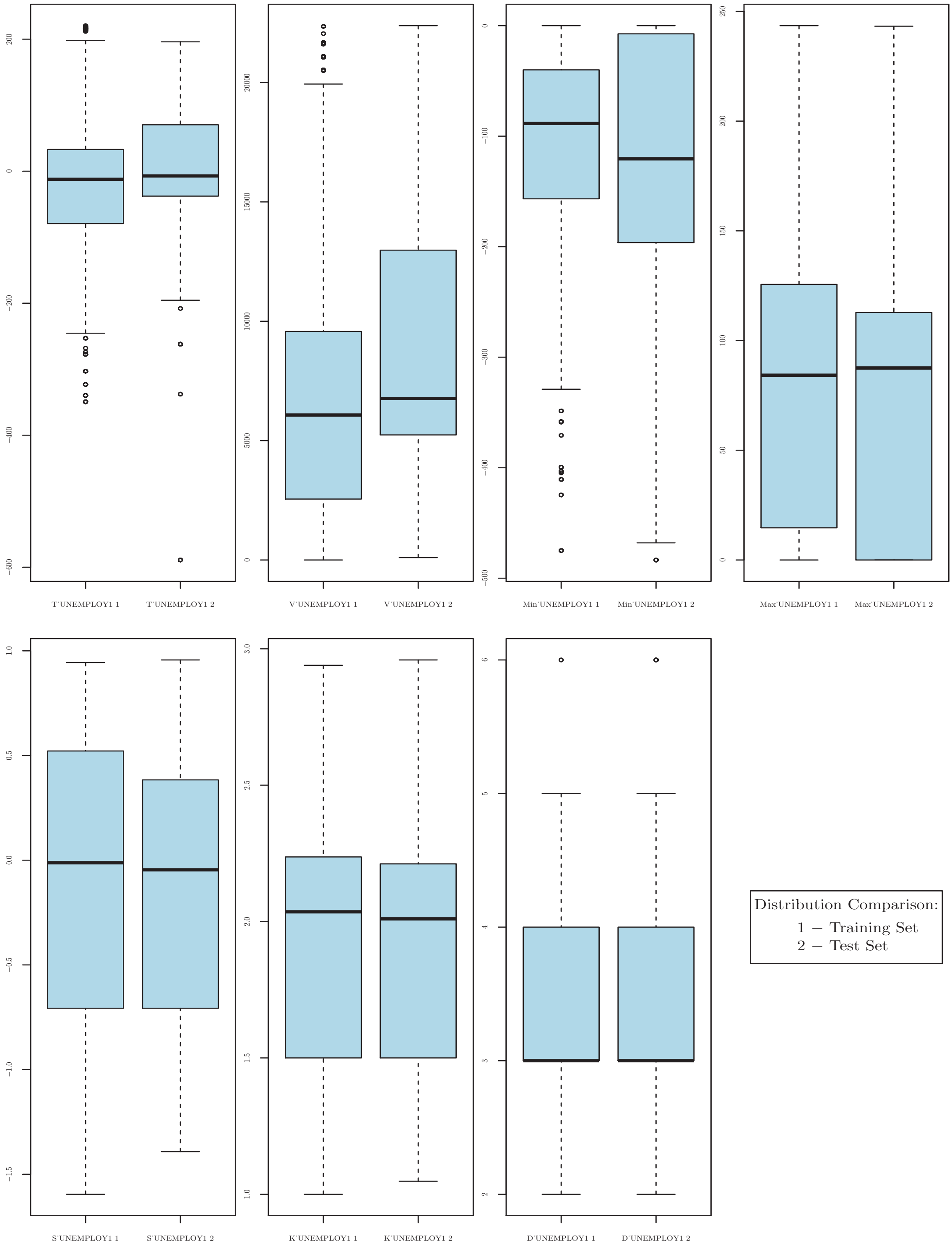


Figure II.62: **Comparison of UNEMPLOY (lag 1) Features Distributions, 1^{st} DB**

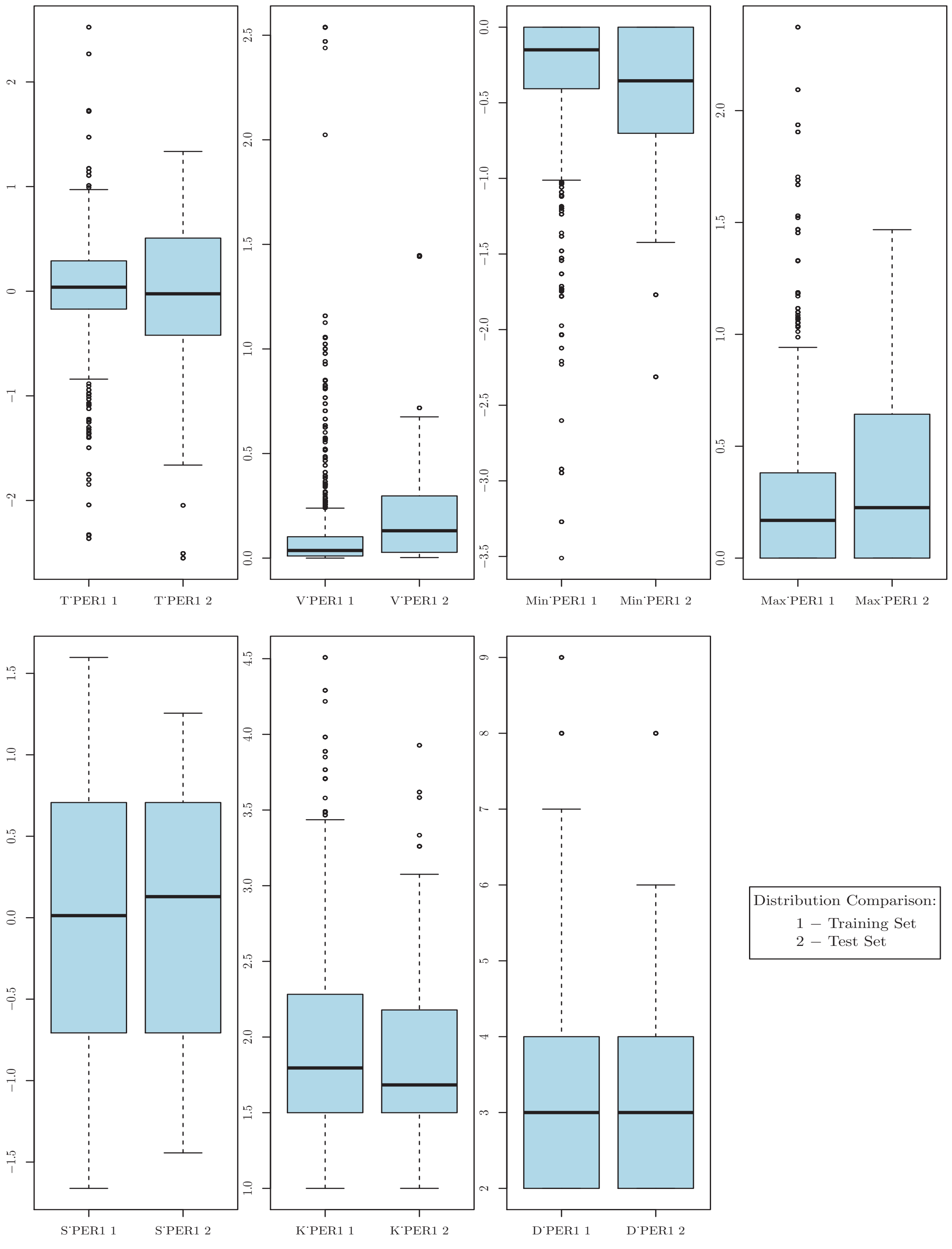


Figure II.63: **Comparison of PER (lag 1) Features Distributions, 1st DB**

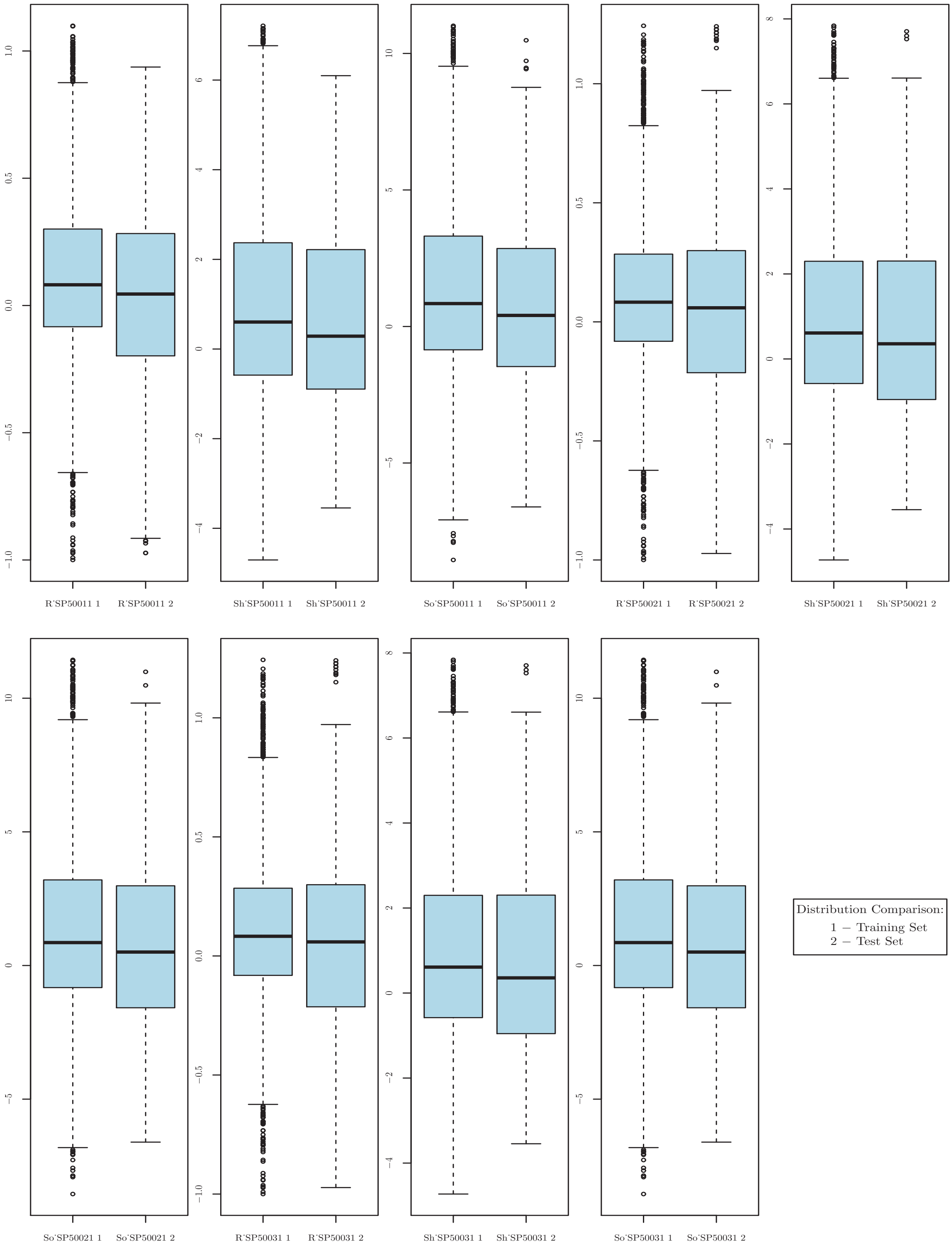
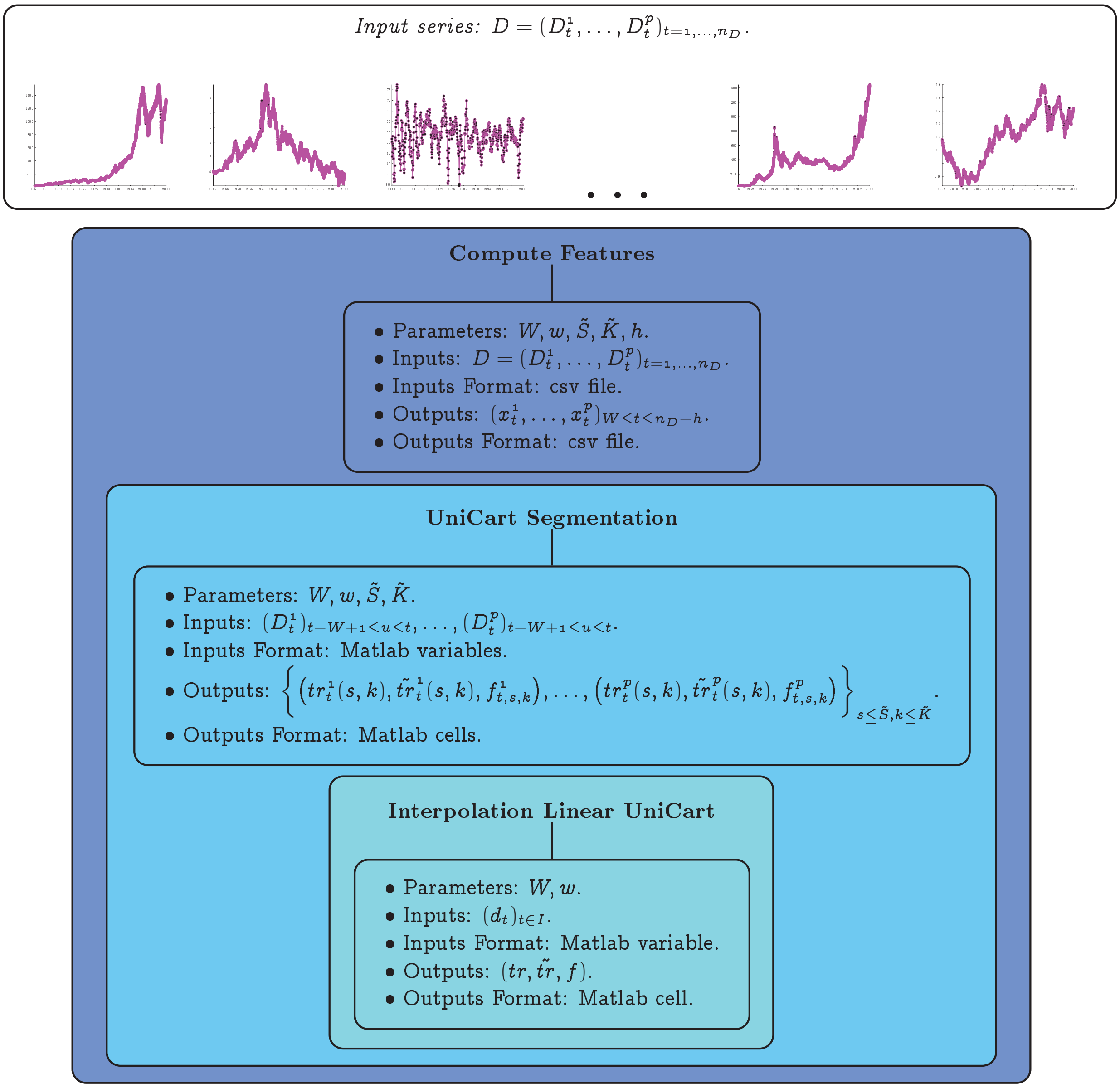


Figure II.64: **Comparison of SP500 (lag 1) Features Distributions, 2^{nd} DB**

II.2.3 Input-Output Diagram of the UniCart Database Construction Procedure

The following simplified diagram sums up the Matlab implementation of the UniCart database construction procedure. It consists of three nested modules. The first one, entitled *Interpolation Linear UniCart*, turns any given finite series $(d_t)_{t\in I}$ into the UniCart approximation tree $(tr, \tilde{tr}, f)$. The second module, *UniCart Segmentation*, calls the first module to produce the linear approximations $\left(tr_t^i(s,k), \tilde{tr}_t^i(s,k), f_{t,s,k}^i\right)$ indexed by the couple (s,k) of scale and lag for each windowed series $(D_t^i)_{t-W+1\leq u\leq t}$. The third module, *Compute Features*, uses the second module to compute the set of features $x_t^i = (x_{t,s,k}^i)_{s\leq S,k\leq K}$ associated with the approximations $\left(tr_t^i(s,k), \tilde{tr}_t^i(s,k), f_{t,s,k}^i\right)$. The features x_t^i are collected for all variables $D_t^1, \ldots, D_t^p$ over the historic range $W \leq t \leq n_D - h$.

Input-Output Diagram UniCart



Input series: $D = (D_t^1, \ldots, D_t^p)_{t=1,\ldots,n_D}$.

Compute Features

- Parameters: $W, w, \tilde{S}, \tilde{K}, h$.
- Inputs: $D = (D_t^1, \ldots, D_t^p)_{t=1,\ldots,n_D}$.
- Inputs Format: csv file.
- Outputs: $(x_t^1, \ldots, x_t^p)_{W\leq t\leq n_D-h}$.
- Outputs Format: csv file.

UniCart Segmentation

- Parameters: $W, w, \tilde{S}, \tilde{K}$.
- Inputs: $(D_t^1)_{t-W+1\leq u\leq t}, \ldots, (D_t^p)_{t-W+1\leq u\leq t}$.
- Inputs Format: Matlab variables.
- Outputs: $\left\{\left(tr_t^1(s,k), \tilde{tr}_t^1(s,k), f_{t,s,k}^1\right), \ldots, \left(tr_t^p(s,k), \tilde{tr}_t^p(s,k), f_{t,s,k}^p\right)\right\}_{s\leq\tilde{S},k\leq\tilde{K}}$.
- Outputs Format: Matlab cells.

Interpolation Linear UniCart

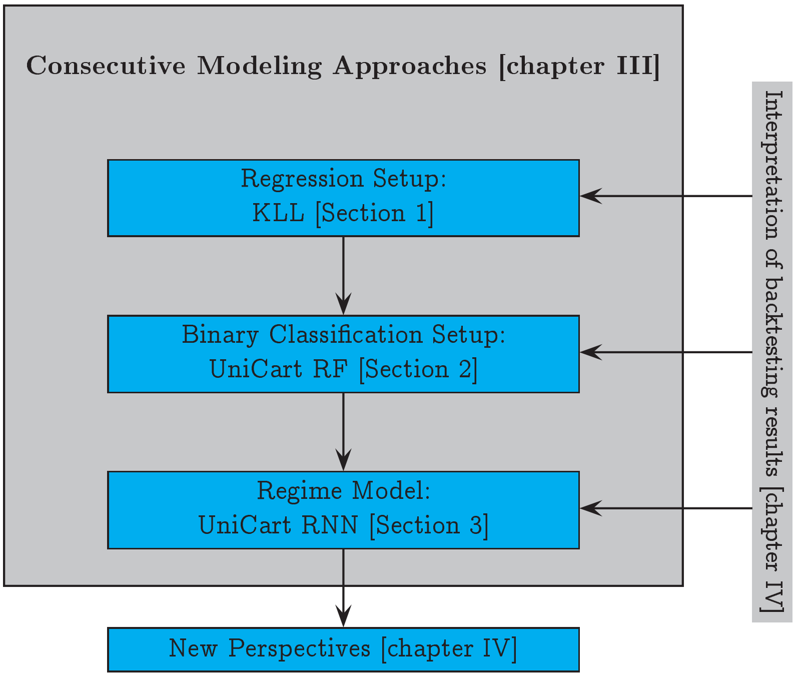
- Parameters: W, w.
- Inputs: $(d_t)_{t\in I}$.
- Inputs Format: Matlab variable.
- Outputs: $(tr, \tilde{tr}, f)$.
- Outputs Format: Matlab cell.

Chapter III

Construction of Trend Predictive Models

> And Now for Something Completely Different
>
> Monty Python

Chapter II is devoted to the construction of databases relevant for learning and improving trend predictive numerical models. The goal of this chapter is precisely to present our research, oriented by the experimentation protocol, for the most suitable setup and model construction procedure. Three different modeling approaches have thus been consecutively developed and shaped: first linear model selection in the regression setup, then classic learning in the binary classification setup, and finally transfer learning with a non-parametric regime model. They correspond respectively to the section 1, 2 and 3 of this chapter. The switch from one approach to another is thoroughly motivated in chapter IV. Particularly, the interpretability, the robustness and the quality of the parameter tuning of the procedures, specific to each successive approach, are shown to increase significantly. The regression setup, somehow the most conventional in financial modeling, is investigated in section 1 via the Kalman LagLasso (KLL) approach. Strong hypotheses are formulated on data, which enable the statement of a linear relation between the (Kalman) filtered target variable and (Kalman) filtered and lagged factors. These factors are monthly sampled variables (hence both economic and financial factors are sampled at the same frequency). Model selection is performed by a new procedure called LagLasso. LagLasso is a Lasso-type procedure which includes selection of lags for individual factors. The backtesting results of chapter IV prove that this selection step may suffer from lack of robustness, and requires a heavy parameter tuning procedure. Particularly, the model does not provide a fair idea of the relevance of explanatory variables. Finally, the lack of precision of the predictions has convinced us that the setup may be too ambitious and inappropriate. The binary classification setup, introduced in section 2, provides a more realistic targeted level of precision for the predictions. The approach developed in this setup consists in learning the distribution of UniCart features and labels via tree aggregation based procedures, such as Random Forests (RF). Many interpretation functionalities are made available. For instance, probabilities of realization, associated with predictions delivered by the numerical model, greatly help in the designing of efficient trading rules. The RF procedure also ranks features according to their relative predictive ability. It emphasizes thus the dominant influence in the prediction of the trend-related features of the target variable. However, a strong overfitting effect is observed. The performance of tree classifiers dramatically decrease in case of a too small value of the minimal node size. In other words, tree classifiers of a too large size do not generalize well. We consider this overfitting incident as a transfer learning failure. More precisely, the partition structure of any large tree classifier cannot be successfully transferred from the training set to the test set. To produce a partitioning structure, which would be transferable, the non-parametric regime model stated in subsection I.3.1 is introduced in section 3. The regimes as well as the number of regimes are learnt using a new transfer learning procedure called Relabeled Nearest Neighbors (RNN). This meta-algorithm requires a generator of hierarchical partitioning rules, e.g. any tree aggregation procedure, and a clustering method, e.g. any hierarchical or spectral clustering method. It is composed of three steps. It uses first partitioning rules, constructed on the training set, to produce a pseudometric. Secondly, a hierarchical dyadic partition of the training set is computed using the pseudometric and the clustering method. We call this step relabeling as the terminal clusters allow new labels, i.e. regimes, to be defined. These new labels are additionally associated with scores, which have the property to rank the objects of the training set according to their relative proximity (i.e. according to their propensity of being labelled +1). The third and final step reduces to a straightforward 1-nearest neighbor classification. For any object in the test, the pseudometric indicates its closest object in the training set. The regime and score of this latter are thus attributed to the test set object. RNN is proven in chapter IV to yield very good and stable results. It notably exhibits an outstanding reduction of overfitting compared to the RF procedure for our cases of study. To complete this presentation, another of our procedure, which has led to the development of the RNN procedure, is described in appendix. It is called Supervised Boosted Nearest Neighbors (SBNN) and is, like RNN, specifically designed to learn different regimes. It introduces the new notion of clusterizer, actually a distance-based function indicating whether two objects are closed to each other or not. The SBNN procedure trains specific classifiers, called local classifiers, using clusters of objects defined by the clusterizers. The underlying idea of the procedure is to use the boosting philosophy to simultaneously aggregate clusterizers and local classifiers in order to get more precise classifiers. Unfortunately, the computational difficulties of this approach make it hardly usable, even if heuristics to reduce that computation cost have been developed. Specific parameters must also be specified. We preferred therefore to use the RNN procedure.


Consecutive Modeling Approaches [chapter III]
Regression Setup:
KLL [Section 1]
Binary Classification Setup:
UniCart RF [Section 2]
Regime Model:
UniCart RNN [Section 3]
New Perspectives [chapter IV]
Interpretation of backtesting results [chapter IV]

III.1 Kalman LagLasso

The goal of the Kalman LagLasso approach, developed in the regression setup, is to predict the variations of a target variable at a given horizon. This section first states restricting hypotheses, which motivate the introduction of a specific temporal regression model. This model defines a linear relationship between the Kalman innovation residual of the target variable and lagged and Kalman innovation residuals of the explanatory variables. To estimate the parameters of the model, classical linear model selection procedures are reviewed. A new procedure, called LagLasso, is introduced. It consists in selecting the most significant lagged residuals under the constraint that only one lag per variable can be chosen. The LagLasso implementation is finally described in detail.

III.1.1 Statement of the KLL Temporal Regression Model

The following modeling choices and hypotheses are made. We work in the regression setup and consider that the label is the variation of a target variable at a given horizon. We assume that the label can be decomposed into an inner component, which is due solely to the target variable itself, and an exogenous component, which can be explained by a small subset of factors. We also suppose that the influence of each of these factors can change over time and can be lagged. Finally, the label is supposed to be particularly sensitive to unexpected variations of the factors, that is to say to large deviations of the explanatory variables from their own trend. To meet these hypotheses, the explanatory variables and the target variable are filtered via a linear state-space model, which is specified for each of them. Their innovation residuals are computed with the Kalman algorithm (cf. equation II.4 in subsection II.1.3). Formally, we observe the random pair $(\tilde{X}_t, \tilde{Y}_t)$ over time $t = 1, \dots, n$. We consider $\tilde{Y}_t$ as being the residual of the target variable and $\tilde{X}_t$ as the random vector of dimension p whose components are residuals of explanatory variables $(\tilde{X}_t^1, \dots, \tilde{X}_t^p)$. We aim at predicting $(\tilde{Y}_t)_{t=n+1,\dots,n+h}$, where the integer $h \geq 1$ is the prediction horizon. The following regression model is introduced (where all variables are supposed to be centered hence there is no constant term):

$$\tilde{Y}_{t+h} = \sum_{i=1}^{p} \beta_i \tilde{X}^i_{t-\sigma(i)}. \tag{III.1}$$

The lag of the i^{th} variable is an integer denoted $\sigma(i)$ and checks $0 \leq \sigma(i) \leq \Sigma$. The regression coefficients are represented by the real vector $\beta = (\beta_1, \dots, \beta_d)$. Lags and regression coefficients are unknown parameters of the model and have to be estimated. A model selection procedure is therefore needed. In the next subsection, the regression problem and the linear regression estimation model are recalled, as well as classic procedures of model selection in linear regression.

III.1.2 Model Selection in the Regression Setup

Regression Problem and Linear Regression Estimation Model. As in subsection I.2.1, the random vector $Z = (X, Y) \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, distributed according to P, stands for the data. The random vector $X = (X^1, \dots, X^p) \in \mathcal{X}$ represents the factors (in our case the residuals of explanatory economic and financial variables). Besides, the label $Y \in \mathcal{Y}$ is real, i.e. $\mathcal{Y} = \mathbb{R}$. The loss function is:

$$Q(Z, f) = L(Y, f(X)) = |Y - f(X)|^2.$$

The true risk and the empirical risk are expressed as follows.

$$R(f) = \mathbb{E}(|Y - f(X)|^2) = \int |y - f(x)|^2 dP(x, y), \qquad R_n(f) = \frac{1}{n} \sum_{i=1}^{n} |Y_i - f(X_i)|^2.$$

Using the observations $Z_{1:n} = (Z_1, \dots, Z_n)$, we aim at finding a numerical model $\widehat{f}_n = \widehat{f}_n(Z_1, \dots, Z_n)$ of low prediction error in the set of candidate functions $\mathcal{F}$. We consider the linear regression problem, i.e. the particular case where the candidate functions $f \in \mathcal{F}$ are linear:

$$f(x) = f_\beta(x) = x^\top \beta, \qquad x \in \mathbb{R}^p,$$

where $\beta = (\beta_1, \dots, \beta_p) \in \mathbb{R}^p$ and $x_{1:n} = (x_1, \dots, x_n)$ are realizations of the random vectors $X_{1:n} = (X_1, \dots, X_n)$. Once again, all the variables are supposed to be centered. The least square estimator $\widehat{f}_{\widehat{\beta}^{ls}}$ is obtained via the application of the ERM strategy over the set $\mathcal{F}$ of linear functions.

$$\widehat{\beta}^{ls}(Z_{1:n}) = (X_{1:n}^\top X_{1:n})^{-1} X_{1:n}^\top Y_{1:n}.$$

In the case of linear regression estimation in the classical paradigm, we suppose that the labels Y follow a model with additive and Gaussian noise:

$$Y = X\beta + \varepsilon,$$

where $\varepsilon \backsim \mathcal{N}(0, \sigma_\varepsilon^2)$. The least square estimator is unbiased:

$$\mathbb{E}(\widehat{\beta}^{ls}) = \beta,$$

and the Gauss-Markov theorem [HTF01] shows that it has the smallest variance among all linear unbiased estimators.

Regularization Methods: Ridge Regression and the Lasso. As pointed out in subsection I.2.1, the risk can be expressed as a trade-off between bias and variance:

$$R(f) = \mathbb{E}_X\left(var(Y|X) + var\big(f_\beta(X)\big) + bias\big(f_\beta(X)\big)\right).$$

This decomposition shows that including many variables naturally provides a lower bias, but usually also a higher variance. On the contrary, having fewer variables helps reducing the variance of the estimator but also produces a higher bias. Introducing a small bias in the estimation of β might lead to a substantial decrease in variance and hence to a smaller prediction error. A strategy to control the variance term is to add a penalty on β. This is called shrinkage or regularization [HTF01]. For instance, the Ridge Regression is a regularization method, which uses L^2 penalty [HTF01]. We have:

$$\widehat{\beta}_\lambda^{ridge}(Z_{1:n}) = \underset{\beta \in \mathbb{R}^p}{\arg\min} \, ||Y_{1:n} - X_{1:n}^\top \beta||^2 + \lambda \sum_{j=1}^{p} |\beta_j|^2, \ \lambda \geq 0,$$

where $||.\beta||^2$ is the L^2 norm of $\mathbb{R}^p$. The Ridge Regression delivers a path of solutions $\widehat{\beta}_\lambda^{ridge}$ indexed by the parameter λ:

$$\widehat{\beta}_\lambda^{ridge}(Z_{1:n}) = \left(X_{1:n} X_{1:n}^\top + \lambda Id\right)^{-1} X_{1:n}^\top Y_{1:n}.$$

To select a single λ, that is a single Ridge solution, cross-validation or Mallows' C_p are the most popular criteria. Another celebrated regularization method is the Lasso (acronym for Least Absolute Shrinkage and Selection Operator), which uses L^1 penalty [BET04]. We have:

$$\widehat{\beta}_\lambda^{lasso}(Z_{1:n}) = \underset{\beta \in \mathbb{R}^p}{\arg\min} \, ||Y_{1:n} - X_{1:n}^\top \beta||^2 + \lambda \sum_{j=1}^{p} |\beta_j|, \ \lambda \geq 0.$$

Compared to the L^2 penalty, the L^1 penalty adds sparsity. This means that, for any λ value, a smaller subset of factors are selected for computing the estimator $\widehat{\beta}_\lambda^{lasso}(Z_{1:n})$ than for computing the estimator $\widehat{\beta}_\lambda^{ridge}(Z_{1:n})$. This is an interesting interpretation property. However, unlike Ridge Regression, Lasso solutions have no closed form. Quadratic programming techniques can be used to compute the Lasso estimator but the Lars (acronym for Least Angle Regression algorithm, the "S" suggesting "Lasso" and "Stagewise" [BET04]) is a much simpler and computationally more efficient strategy [BET04].

Lars Pseudocode. The Lars algorithm is an iterative procedure of variable selection generalizing the concept of bisector in a multidimensional framework [BET04, BEW07]. Its pseudocode is as follows. Note that it is written with $x_{1:n} = (x_1, \dots, x_n)$, realizations of the random variables $X_{1:n} = (X_1, \dots, X_n)$ and that the set of iteratively selected variables is called the active set.

```
1. Standardize the factors $x_i$ to have mean 0 and variance 1. Initialisation: $r = y - \overline{y}$, $\beta_i = 0, \forall i$.
2. Find the factors $x_j$ most correlated with $r$.
3. Move $\beta_j$ from 0 toward its least-squares coefficient $\langle x_j, r\rangle$, until some other competitor $x_k$, $k \neq j$, has as much correlation with the current residual as $x_j$ has.
4. Move $(\beta_j, \beta_k)$ in the direction defined by their joint least squares coefficient of the current residual on $(x_j, x_k)$, until some other competitor $x_l$ has as much correlation with the current residual, i.e.:
   $< x_l, r >=< x_k, r >=< x_j, r >$.
5. If a non-zero coefficient hits zero, drop it from the active set, reinclude the variable in the inactive set and recompute the current joint least squares direction.
6. Eliminate the variable $j$ from the inactive set. Continue until $p$ variables are entered.
```

Under a slight modification, the Lars algorithm yields all Lasso solutions [BET04]. Finally, the Lasso sparse selection of factors is computable via the Lars algorithm. It may lead to both a variance reduction of the estimator $\widehat{\beta}$ - hence to more accurate predictions - and to a gain in terms of interpretability. However, the Lasso cannot be used straightforwardly to estimate the parameters of the model III.1. Indeed, the Lasso problem does not take into account the constraint of selecting a single lag per variable. The next subsection is devoted to the presentation of the LagLasso problem and to the introduction of a new algorithm for solving it.

III.1.3 LagLasso: Introduction and Application

The LagLasso model. We introduce specific notations for the lagged vectors.

$$X_{i,\sigma} = (X^1_{i-\sigma}, \dots, X^p_{i-\sigma}),$$

where $1 \leq i \leq n$ and $1 \leq \sigma \leq \Sigma$. In addition, the vector $X_{i,1:\Sigma}$ regroups the lagged vectors in a single one:

$$X_{i,1:\Sigma} = (X^1_i, \dots, X^1_{i-\Sigma}, \dots, X^p_i, \dots, X^p_{i-\Sigma}).$$

Naturally, $X_{1+\Sigma:n+\Sigma,1:\Sigma}$ is the matrix made of these lagged vectors

$$X_{1+\Sigma:n+\Sigma,1:\Sigma} = (X^1_i, \dots, X^1_{i-\Sigma}, \dots, X^p_i, \dots, X^p_{i-\Sigma})_{1+\Sigma \leq i \leq n+\Sigma}.$$

Likewise, we use a double index for $\beta_{i,\sigma} \in \mathbb{R}^{p\times\Sigma}$ to account for the variables and the lags. We consider the problem:

$$\widehat{\beta}^{laglasso}_{\lambda}(Z_{(1+\Sigma):n+\Sigma,(1:\Sigma)}) = \arg\min_{\beta\in\mathbb{R}^{p\times\Sigma}} ||Y_{(1+\Sigma):(n+\Sigma)} - X^{\top}_{(1+\Sigma):(n+\Sigma),1:\Sigma}\beta||^2 + \lambda\sum_{j=1}^{p}\sum_{\sigma=1}^{\Sigma}|\beta_{j,\sigma}|,\ \lambda \geq 0.$$

Besides, we choose the following type of candidate functions:

$$f_\beta = \left\{ \begin{array}{lll} \mathbb{R}^{p\times\Sigma} & \longrightarrow & \mathbb{R} \\ x & \longmapsto & \sum_{i=1}^{p}\sum_{\sigma=1}^{\Sigma}\beta_{i,\sigma}x_{i,\sigma}, \end{array} \right.$$

such that $\exists!\sigma(i), \beta_{i,\sigma(i)} \neq 0$. We propose to solve this problem via the procedure explained in the next paragraph.

The LagLasso solution. This problem of lag identification is different from the problem stated in [CC08], where a Lars algorithm specifically designed for time series is introduced. Each variable is then represented by a matrix made of its lagged realizations: this algorithm manages to select iterative blocks of lags corresponding to a single variable instead of single lags corresponding each to a variable. On the other hand, we aim not only at building a competitive prediction method for time series but also at clearly stating the problem of lag identification. The idea consists in writing a variant of the Lars/Lasso algorithm, where both a variable and a lag are selected at each step, all the other lags of the variables being then eliminated from the possible further selections. By analogy with the Lars algorithm, the set of iteratively selected lagged variables is called the active set. We call this variant the LagLasso procedure. Its steps of the LagLasso are the following:

1. *Choose* $\mathrm{lag}^{\max}$ *and* $\mathrm{lag}^{\min}$*:* $\sigma_i \in [\mathrm{lag}^{\min}, lag^{max}], \forall i$.
2. *Standardize the factors* $x_{i,\sigma}$ *to have mean* 0 *and variance* 1*. Initialisation:* $r = y - \overline{y}$, $\beta_{i,\sigma} = 0, \forall i, \sigma$.
3. *Find the factors* $x_{j,\sigma}$ *most correlated with* r.
4. *Move* $\beta_{j,\sigma}$ *from* 0 *toward its least-squares coefficient* $\langle x_{j,\sigma}, r\rangle$*, until some other competitor* $x_{k,\tau}$*,* $k \neq j$*, has as much correlation with the current residual as* $x_{j,\sigma}$ *has.*
5. *Move* $(\beta_{j,\sigma}, \beta_{k,\tau})$ *in the direction defined by their joint least squares coefficient of the current residual on* $(x_{j,\sigma}, x_{k,\tau})$*, until some other competitor* $x_{l,v}$ *has as much correlation with the current residual, i.e.:*
   $< x_{l,v}, r >=< x_{k,\tau}, r >=< x_{j,\sigma}, r >$.
6. *If a non-zero coefficient hits zero, drop it from the active set, reinclude the variable and all its lags in the inactive set and recompute the current joint least squares direction.*
7. *Eliminate all the lags corresponding to variable* j *from the inactive set. Continue until* p *variables are entered.*

In addition, as for the Ridge and Lasso regressions, both C_p-type and cross-validation stopping criteria are implemented to select a single step in this iterative process, hence a single vector $\widehat{\beta}$. Reconsidering the statistical model III.1, the model selection is performed straightforwardly via LagLasso. It provides the main interpretation functionality of the Kalman LagLasso procedure, as it enables a ranking of the explanatory variables according to their importance. Unfortunately, backtesting results, provided in the next chapter, show the parameter tuning difficulties of Kalman LagLasso, which are directly linked to the appreciating of the relevance of the explanatory variables. Besides, the precision of the Kalman LagLasso predictions casts a serious doubt on the appropriateness of the setup. We switch therefore to the binary classification setup, which is less demanding regarding the precision of the predictions and possibly richer in terms of interpretation functionalities.

III.2 Tree Aggregation Procedures for Binary Classification

This section is devoted to the presentation of the most popular tree aggregation procedures in the binary classification setup. These procedures are extensively used in the data mining and machine learning communities mainly because of their interpretability [BFOS84, RM08]. This section therefore first introduces tree classifiers. The CART algorithm for learning tree classifiers is particularly made explicit. Secondly, the aggregation techniques of Boosting and of Bagging are presented. We explain how both of them produce very efficient classifiers by combining weak classifiers. We also show how they lead to the design of specific tree aggregation procedures, such as Boosted Trees, Bagged Trees and Random Forests (RF). The interpretation functionalities of these procedures are finally highlighted.

III.2.1 Introduction to Tree Classifiers

Tree classifiers are regarded as interpretable classifiers, and often referred to as "weak" classifiers, i.e. classifiers with a low predictive power. They are also sophisticated mathematical objects, whose definition borrows from both machine learning and graph theory [Big93, Bol98, CG01, BFOS84, RM08]. In this subsection, tree classifiers are first presented as a specific set of candidate functions for binary classification. The CART procedure for growing tree classifiers is then thoroughly recalled.

Tree Classifiers: Candidate Functions for Binary Classification. Following the notations of the subsection I.2.1, the objects of the database are realizations of the random vector $(X^1, \dots, X^p) \in \mathcal{X}$, where $\mathcal{X}$ is a Borel space. The labels of the objects are represented by the binary variable $Y \in \mathcal{Y} = \{-1, 1\}$. Tree classifiers are now rigorously defined using the notions of recursive partition, splitting rule and rule of label attribution. We call a partition of $\mathcal{X}$ a finite set of disjoint subsets $\left(\mathcal{X}_i\right)_{i \in I}$ such that their union form the space $\mathcal{X}$, i.e.

$$\mathcal{X} = \bigcup_{i=1}^{|I|} \mathcal{X}_i.$$

A recursive partition of the space $\mathcal{X}$ is defined as the couple $(tr, \tilde{tr})$, where $tr = (V, A)$ is a rooted binary tree and where $\tilde{tr}$ is a function of the type:

$$\tilde{tr} = \begin{cases} \mathcal{X} \times V & \longrightarrow & \mathcal{B}(\mathcal{X}) \\ (X, v) & \longmapsto & \tilde{tr}(X, v) \end{cases}$$

which satisfies the following conditions.

- The root $1 \in V$ of the tree checks $\tilde{tr}(\mathcal{X}, 1) = \mathcal{X}$.
- Each node $v \in V$ is associated with a subset $\tilde{tr}(\mathcal{X}, v)$ such that its child nodes are associated respectively with $\tilde{tr}(\mathcal{X}, v_L)$ and $\tilde{tr}(\mathcal{X}, v_R)$, which are disjoint subsets checking:

$$\tilde{tr}(\mathcal{X}, v) = \tilde{tr}(\mathcal{X}, v_L) \bigcup \tilde{tr}(\mathcal{X}, v_R).$$

We denote $\tilde{V}$ the set of the leaves of the tree tr. The leaves $v \in \tilde{V}_{sub}$ of any subtree $tr_{sub} = (V_{sub}, A_{sub}) \subseteq tr$ define a partition $\{\tilde{tr}_{sub}(\mathcal{X}, v)\}_{\{v \in \tilde{V}_{sub}\}}$ of $\mathcal{X}$. Usually, the recursive partitions associated with tree classifiers are defined as finite intersections of splits. We define a **split** or **splitting rule** sp as the inverse image of an unbounded interval under a single explanatory variable. It is typically of the type:

$$sp = X_i^{-1}(] - \infty, a]), \; a \in \mathbb{R}, \; i \in \{1, \dots, p\}.$$

The real value a is called the threshold. The set of finite combinations of splits is denoted $Sp \subset \mathcal{B}(\mathcal{X})$. Additionally, we suppose that there exists $sp_v \in Sp$ such as:

$$\begin{array}{rcl} \tilde{tr}(\mathcal{X}, v_L) & = & sp_v \cap \tilde{tr}(\mathcal{X}, v), \\ \tilde{tr}(\mathcal{X}, v_R) & = & sp_v^c \cap \tilde{tr}(\mathcal{X}, v). \end{array}$$

Each subset of the partition is thus an element of the set Sp. The last ingredient to define a tree classifier is the rule of label attribution. We call rule of label attribution any function of the type:

$$f = \begin{cases} \mathcal{B}(\mathcal{X}) & \longrightarrow & \mathcal{Y} \\ A & \longmapsto & f(A). \end{cases}$$

The rule is usually the majority vote computed on the training set. In other words, the label attributed to $A \in \mathcal{B}(\mathcal{X})$ is the label of the largest number of the objects of the training set $x_{1:n}$ contained in A. The majority rule is formally defined as follows.

$$f_M(A) = \arg\max_{j \in \mathcal{Y}} p_r(j|A) = argmax_{j \in \mathcal{Y}} \, p_r(A, j),$$

where

$$\begin{aligned} p_r(j|A) &= \frac{\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathbb{1}_{\{x \in A, y=j\}}}{|A \cap x_{1:n}|}, \\ p_r(j, A) &= \frac{\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathbb{1}_{\{x \in A, y=j\}}}{n}. \end{aligned}$$

A tree classifier is defined by the triplet $(tr, \tilde{tr}, f_M)$. Any object X of the database browses the tree from the root to a leaf. It is oriented to the node associated subset to which it belongs. It is thus associated with a leaf, then with the leaf corresponding subset and finally with the label attributed to the subset via the rule. The label $l(X)$ attributed to X by the tree classifier $(tr, \tilde{tr}, f_M)$ is given by the formula:

$$l(X) = f_M\left(\bigcup_{v \in \tilde{V}} \{X \in \tilde{tr}(\mathcal{X}, v\}\right).$$

Tree classifiers have been introduced as elementary candidate functions for binary classification. The CART algorithm is a procedure which allows such functions to be trained. Its goal is formally stated in the next paragraph.

Goal of the CART algorithm [BFOS84]. The construction of a tree classifier $(tr, \tilde{tr}, f_M)$ is based on the concept of purity. The purity of a tree node is related to the proportion of objects of different labels in its associated subset. A node is said to be purer than another one if its proportion of objects of different labels is lower. When growing a binary tree classifier, the goal is that each couple of child nodes is purer than its parents. To state this goal more formally, the notion of impurity is now introduced. We denote $p_r(v)$ the **pr**oportion of objects of the training set corresponding to the node v. Besides, $p_r(j|v)$, $j \in \mathcal{Y}$, $v \in V$, represents the proportion of these objects, labelled j. Using a slight abuse of notation, we have:

$$\begin{aligned} p_r(v) &:= p_r(\tilde{tr}(\mathcal{X}, v)) = \frac{\sum_{i=1}^{n} \mathbb{1}_{\{x \in \tilde{tr}(\mathcal{X}, v)\}}}{n}, \\ p_r(j|v) &:= p_r(j|\tilde{tr}(\mathcal{X}, v)). \end{aligned}$$

Additionally, the function ϕ_{imp} is said to be an **imp**urity function if it satisfies the following properties.

- ϕ_{imp} is defined on the set $\{(p_1, p_2) : (p_1, p_2) \in \mathbb{R}_+^2, \, p_1 + p_2 = 1\}$.
- ϕ_{imp} has a maximum only at the point $(\frac{1}{2}, \frac{1}{2})$.
- ϕ_{imp} has a minimum at the points $(1, 0)$ and $(0, 1)$.
- ϕ_{imp} is a symmetric function of p_1, p_2.
- ϕ_{imp} is strictly concave.

The node impurity measure Q is defined using this impurity function.

$$Q(v) = p_r(v) q(v),$$

where

$$q = \left\{ \begin{array}{lcl} V & \longrightarrow & \mathbb{R}_+ \\ v & \longmapsto & \phi_{imp}(p_r(-1|v), p_r(+1|v)). \end{array} \right.$$

Therefore, at node v, the impurity is maximal if the two labels are equally represented in the subset $\tilde{tr}(X, v)$. It is on the contrary minimal if all the objects have the same labels. There are various functions ϕ_{imp} hence various impurity measures:

- the misclassification function: $\phi_{imp}(p_1, p_2) = 1 - max(p_1, p_2)$.
- the Gini function: $\phi_{imp}(p_1, p_2) = p_1 \times p_2$.
- the entropy function: $\phi_{imp}(p_1, p_2) = -(p_1 \times \log p_1 + p_2 \times \log p_2)$.
- the Kearns-Mansour function: $\phi_{imp}(p_1, p_2) = \sqrt{p_1 \times p_2}$. *Check it.*

Finally, we call tree impurity the quantity denoted $Q(tr) := Q(tr, \tilde{tr}, f_M)$ with a slight abuse of notation:

$$Q(tr) = \sum_{v \in \tilde{V}} p_r(v) q(v) = \sum_{v \in \tilde{V}} Q(v).$$

The CART algorithm aims precisely at minimizing the tree impurity $Q(tr)$ using the Gini impurity function.

CART Pseudocode [BFOS84]. Minimizing the tree impurity $Q(tr)$ is achieved through the iterative maximization of the quantity

$$\Delta Q(sp, v) = Q(v) - Q(v_L) - Q(v_R),$$

called the goodness of split. We give a slight and intuitive justification for this iterative strategy. Consider the tree $tr^{'}$ obtained from the tree tr by splitting the subset of one of its leaf $v^{'} \in \tilde{V}$ with a split $s_{v'} \in Sp$. Two child nodes $v_L^{'}$ and $v_R^{'}$ are thus added to the leaf $v^{'} \in \tilde{V}$. It is straightforward to notice that

$$Q(tr^{'}) = \sum_{v \in \tilde{V} - \{v'\}} Q(v) + Q(v_L^{'}) + Q(v_R^{'}) = Q(tr) - Q(v^{'}) + Q(v_L^{'}) + Q(v_R^{'}).$$

The strict concavity of ϕ_{imp} implies that, for any node v and any split sp,

$$\Delta Q(sp, v) \geq 0,$$

with equality if and only if $p_r(j|v) = p_r(j|v_L) = p_r(j|v_R)$, $\forall j \in \mathcal{Y}$. The CART algorithm is centered on this result. The iterative maximization of the tree impurity stops as soon as a tree-growing stopping criterion is met - for example when the subset size or the goodness of split reach a minimum threshold, or if the number of nodes exceeds a maximum threshold. The pseudocode is as follows.

```
1. Initialization:
    • $v = 0$, $nextv = 1$.
    • $V = \{1\}, A = \emptyset, tr = (V, A), \tilde{tr}(\mathcal{X}, 1) = \mathcal{X}$.
    • Choose the stopping criterion: $|\tilde{tr}(\mathcal{X}, v)| \leq \theta_{min}$, where $|\tilde{tr}(\mathcal{X}, v)|$ is the size of the
      subset $\tilde{tr}(\mathcal{X}, v)$ and $\theta_{min}$ a minimum threshold.
2. Loop:
    • If the stopping criterion is not fulfilled:
        - Compute $\Delta Q(sp, v) = Q(v) - Q(v_L) - Q(v_R)$, $\forall s \in Sp$.
```

- *Compute* $\Delta Q(sp_*, v) = \max_{sp \in Sp} \Delta Q(sp, v)$.
- *Creation of* v_L *the left child node of* v*, corresponding to the split* sp_**. We have:*

$$\begin{cases} \tilde{tr}(\mathcal{X}, v_L) & = sp_* \cap \tilde{tr}(\mathcal{X}, v), \\ V & = V \bigcup \{v_L\}, \\ A & = A \bigcup \{vv_L\}. \end{cases}$$

- *Creation of* v_R *the right child node of* v*, corresponding to the split* sp_**. We have:*

$$\begin{cases} \tilde{tr}(\mathcal{X}, v_R) & = sp_*^c \cap \tilde{tr}(\mathcal{X}, v), \\ V & = V \bigcup \{v_R\}, \\ A & = A \bigcup \{vv_R\}. \end{cases}$$

- *Update* $nextv = nextv + 2$.

- *Update* v*:* $v = v + 1$.

3. *End of Loop. Outputs:* $(tr, \tilde{tr}, f)$.

Additional information, particularly concerning pruning methodology and consistency results, can be found in [BFOS84]. Other meaningful procedures for the construction of tree classifiers, particularly the ID3 and C4.5 algorithms, are presented in [Qui86, Qui93, Qui96]. A known characteristic of these procedures is their instability [Bre96]. Indeed, it has been observed that small perturbations in the training set may lead to the construction of very different tree classifiers. Procedures of aggregation are precisely methods for improving the variance and the bias of tree classifiers. They are reviewed in the next subsection.

III.2.2 Techniques of Aggregation of Weak Classifiers

Techniques of aggregation (or ensemble learning methods, mixtures of experts, committee of learners, etc) aim at selecting and combining many classifiers, via a weighted or unweighted vote of their predictions, in order to produce a more accurate classifier [FS95, SS99, Vay06, Bre96, FH99, AAG96, Bre01]. The combined classifiers are called weak or base classifiers. Methods for combining weak classifiers which can apply to any type of weak classifiers are usually referred to as meta-algorithms. For example, Boosting [FS95, SS99, Vay06] and Bagging (or Bootstrap Aggregating) [Bre96, FH99] are meta-algorithms, whereas Random Forests (RF)[Bre01] is a bagging-based method which only handles tree classifiers. In this document, we are interested in highly efficient and very interpretable techniques of aggregation. We focus therefore on tree aggregation procedures. In the next subsections, the Bagging and Boosting methods are recalled. Their use in conjunction with tree classifiers is motivated. The Random Forests (RF) procedure is finally introduced as a specific extension of the Bagged Trees strategy.

Boosting and Boosted Trees. Boosting algorithms are deterministic techniques of aggregation of weak classifiers. They follow an iterative strategy. They maintain a probability distribution on the objects of the training set, which is initialized as the uniform distribution and updated at each step to emphasize misclassified objects. Each step consists thus of the computation of a weighted error on the training set, which is minimized via the selection of a weak classifier. The classifier is attributed a weight, inversely proportional to the error. Boosting algorithms produce a weighted linear combination of the weak classifiers. The first boosting algorithm is called Adaboost and was introduced in [Sch90, Fre95, FS95]. We recall here the Adaboost pseudocode as stated in [Vay06] and first introduce the notations. We denote ψ_ι the vector representing the discrete distribution on the objects of the training set at iteration ι and

$$R_n^\iota(f) = \sum_{i=1}^{n} \psi_\iota(i) \mathbb{1}_{\{f(x_i) \neq y_i\}}$$

the weighted error on the training set of weak classifier f. The algorithm consists of the following steps.

1. *Initialization:*
$$\psi_1(i) = \frac{1}{n}, \forall i \in \{1, \dots, n\}.$$

2. *For $\iota = 1, \dots, \iota_{max}$, execute the following procedure.*
   - *Choose weak classifier f_ι approximately minimizing R_n^ι over all $f \in \mathcal{F}$.*
   - *Set $R_n^{\iota,\star} = R_n^\iota(f_\iota)$ and adjust the weight w_ι of the classifier f_ι:*
$$w_\iota = \frac{1}{2} \ln \left(\frac{1 - R_n^{\iota,\star}}{R_n^{\iota,\star}} \right).$$
   - *Update the distribution vector:*
$$\psi_{\iota+1}(i) = \psi_\iota(i) \exp\big(- w_\iota f_\iota(x_i) y_i \big).$$

   Normalize the updated vector.

3. *Output:*
$$F_{\iota_{max}} = \sum_{\iota=1}^{\iota_{max}} w_\iota f_\iota.$$

Adaboost is seen in [FHT98] as an implementation of a gradient descent method to minimize the following surrogate of the empirical risk:

$$A_n(f) = \sum_{i=1}^{n} exp(-y_i f(x_i)).$$

Consistency results, based on convex risk minimization strategies, can be found in [Bre00, MMZ02, LV04, BJM06, LKS06, BT07]. Finally, it is straightforward to boost tree classifiers, although a last parameter to tune is the size of each tree. As very large trees are prone to overfitting, it is usually wiser to use small sized trees. A method proposed in [HTF01] is to restrict all trees to be the same size. The ANOVA (abbreviation for analysis of variance) expansion of any aggregated tree classifier f highlights the degree of interaction between features:

$$f(X) = \sum_{j=1}^{p} f^{(j)}(X^j) + \sum_{j,k=1}^{p} f^{(j,k)}(X^j, X^k) + \sum_{j,k,l=1}^{p} f^{(j,k,l)}(X^j, X^k, X^l) + \dots$$

Indeed, the first term indicates a sum of trees which depends on one feature only. The second term refers to trees which depend on two features exactly, etc. It is claimed in [HTF01] and empirically checked by us (in the next chapter) that low-order interactions are the most relevant. Choosing the optimal size of the trees can thus be achieved by applying a cross-validation strategy to a small number of aggregated tree classifiers of "reasonable" size $J \in \{1, \dots, J_{max}\}$, e.g. $J_{max} = 10$. Naturally, a combination of trees is less interpretable than a single one. However, the feature importance is a functionality that remains available. Indeed, the relative importance of any feature in a given tree classifier is computed by summing the improvements in tree purity gained by the splits of that feature. This measure is easily extended to aggregated tree classifiers by summing the relative importance of each variable over all aggregated trees [Fri00].

Bagging and Bagging Trees. Bagging or Bootstrap Aggregating is random technique of aggregation of weak classifiers. It follows an iterative strategy. At each step, a bootstrap sample of the training set, i.e. a replicate of the training set obtained via resampling with replacement, is generated. A weak classifier is then selected to minimize the error computed on the bootstrap sample. The Bagging procedure produces an equally weighted linear combination of weak classifiers. The Bagging pseudocode is recalled here. The bootstrap sample at iteration ι is denoted $((x'_{1,\iota}, y'_{1,\iota}), \dots, (x'_{n_\iota,\iota}, y'_{n_\iota,\iota}))$. The error on the training set of weak classifier f is written:

$$R^\iota_n(f) = \sum_{i=1}^{n_\iota} \mathbb{1}_{\{f(x'_{i,\iota}) \neq y'_{i,\iota}\}}$$

The steps of the algorithm are the following.

```
1. For $it = 1, \dots, it_{max}$, execute the following procedure.
    • Construct a bootstrap sample of the training set $((x'_{1,\iota}, y_{1,\iota})', \dots, (x'_{n_\iota,\iota}, y'_{n_\iota,\iota}))$.
    • Choose weak classifier $f_\iota$ approximately minimizing $R^\iota_n$ over all $f \in \mathcal{F}$.
2. Output:
    $$F_{\iota_{max}} = \sum_{\iota=1}^{\iota_{max}} \frac{1}{\iota_{max}} f_\iota.$$
```

Bagging has been introduced in [Bre96] to reduce the variance of aggregated weak classifiers (cf. the estimation error as defined in the subsection I.2.1). Unstable classifiers, that intuitively are classifiers whose predictions can change drastically under a slight change in the training set, have been particularly investigated. Bagging has been reported to yield excellent results when applied to unstable weak classifiers such as tree classifiers [Bre96], [FH99]. Instability has been formally defined in [BY02]. Theoretical results regarding the variance reduction effect for unstable classifiers have also been obtained in [BY02]. On the other hand, *in the case of U-statistics*, it was shown in [BS06] that bagging could not always reduce variance, whereas it always increases bias. Additionally, conditions for improving nearest neighbor classifiers through bagging have been discussed in [HS05] and consistency results have been obtained. Bagging nearest neighbor classifiers was finally proven to be universally consistent in [BD10]. Finally, plugging bagging to any procedure for generating tree classifiers is straightforward and referred to as Bagged Trees. The remarks made in the case of the Boosted Trees procedure about the choice of the tree size and the interpretability of the tree classifiers remain true for the Bagged Trees procedure. The Random Forests procedure, which can be seen as an extension of the Bagged Trees procedure, is introduced in the next paragraph.

Random Forests. Random Forests is a random and iterative technique of aggregation of tree classifiers. It is initialized by choosing a number $p' \leq p$, where p is the number of features. At each step, a bootstrap sample of the training set is generated. A tree classifier is grown on this sample so that

- at each node, the best split is determined using a subset of p' variables, which are randomly selected.
- it is not pruned (in some versions, the growing can stop at early stages).

The procedure produces an equally weighted linear combination of tree classifiers. The pseudocode of the algorithm is as follows and uses the same notations as for bagging.

```
1. For $\iota = 1, \dots, \iota_{max}$, execute the following procedure.
    • Choose $p' \leq p$ the number of features to be considered at each node of each tree.
    • Construct a bootstrap sample of the training set $((x'_{1,\iota}, y'_{1,\iota}), \dots, (x'_{n_\iota,\iota}, y'_{n_\iota,\iota}))$.
```

- *Grow the tree classifier f_ι approximately minimizing R_n^ι:*

  (a) at each node, randomly select p' features.

  (b) pick up the best split among the p' features.

- *Use the objects, which were not selected in the bootstrap sample, to estimate the error of the tree classifier by predicting the class of each such object.*

2. Output:

$$F_{\iota_{max}} = \sum_{\iota=1}^{\iota_{max}} \frac{1}{\iota_{max}} f_\iota .$$

Random Forests was introduced in [Bre01] as a combination of bagging and random selection of variables [AAG96], [Ho98], [Die00]. A first consistency result has been obtained in the particular case of quantile regression [Mei06]. Recently, the link between random forests and nearest neighbor methods has been highlighted in [LJ06], [BD10]. By considering random forests as randomized nearest neighbor rules, consistency results have been found [BDL08], [Bia]. Furthermore, Random Forests have many interpretation functionalities. As mentioned in the algorithm description, it generates an internal unbiased estimate of the generalization error as the forest construction progresses, by predicting, for each tree, classes on samples that were not used to grow the tree. It also ranks the features according to their importance in the classification, not only using the improvement in tree purity but also according to their corresponding prediction accuracy. In addition, Random Forests yields a similarity measure by computing, for each pair of objects, their number of occurrences in the same terminal node over all trees. Finally, the RF classifier associates a realization probability with each computed prediction by averaging the predictions made by each tree for each class.

Conclusions. We chose to use Random Forests for its efficiency and its wide range of interpretation functionalities. Results are reported in chapter IV. Particularly, overfitting was observed in specific applications, caused by the choice of a too small terminal node size for tree classifiers (i.e. by too large tree classifiers). This observation has lead us to the design of a new transfer learning procedure, called Relabeled Nearest Neighbors (RNN), and introduced in the next section.

III.3 Relabeled Nearest Neighbours

We aim at constructing partitioning structures for classification, which would be transferable from the training set, denoted $x_{trn_{1:n_{trn}}}$ from now on, to the test set, denoted $x_{test_{1:n_{test}}}$. In this section, this problem is stated in mathematical terms. The algorithm Relabeled Nearest Neighbours (RNN) is introduced to solve it. The following statistical model for clustering and classification is therefore defined. We consider the random triplet $(X, Y, R) \sim P$, where $X \in \mathcal{X}$ is the observation vector, where the label $Y \in \mathcal{Y} = \{-1, +1\}$, and where $R \in \{1, \dots, N_R\}$ is a latent variable standing for a specific distribution of the random pair (X, Y). The distribution of the random pair (X, Y) is given by the mixture model

$$(X, Y) \sim \sum_{r=1}^{N_R} c_r P_r ,$$

where $c_r = P(R = r)$ is a weight, and where $P_r = P_{(X,Y)|R=r}$ is a non-parametric component distribution also called regime. Each regime P_r is unknown and the number N_R of regimes is also unknown. The goal is then to find a number of regimes N_R, a regime attribution function $H : \mathcal{X} \longrightarrow \{1, \dots, N_R\}$ and regime specific prediction functions $(f_r)_{r=1,\dots,N_R}$, where $f_r : \mathcal{X} \longrightarrow \mathcal{Y}$, so that the numerical model defined by the triplet $\left(N_R, H, (f_r)_{r=1,\dots,N_R}\right)$ has a low prediction error.

RNN is a three-step procedure for constructing the numerical model $\left(N_R, H, (f_r)_{r=1,\ldots,N_R}\right)$. The first step is entitled *Computation of the Pseudometric*. It computes the pseudometric $d_{RNN}^{\mathcal{T}_\mathcal{R}}$ by generating an ensemble $\mathcal{T}_\mathcal{R}$ of hierarchical partitioning rules. Each rule is considered as a node weighted graph. The node weight actually reflects the prediction ability of the corresponding split. The figure III.1 proposes an intuitive representation of these rules.

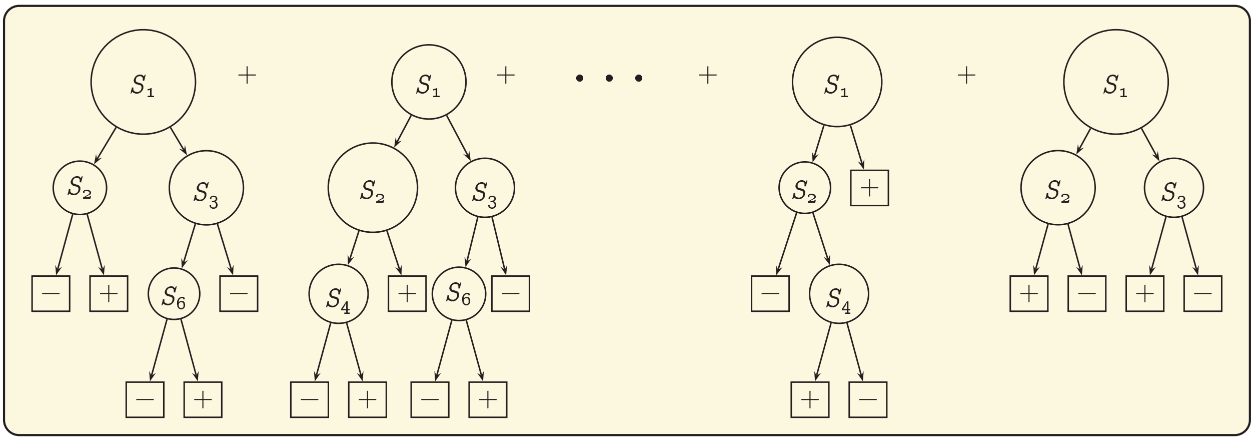


Figure III.1: **Partitioning Rules as Node Weighted Graphs**

The second step, called *Relabeling the Training Set*, produces the hierarchical partitioning rule $(tr_{RNN}, \tilde{tr}_{RNN}, sc_{RNN})$ using the training set, the pseudometric $d_{RNN}^{\mathcal{T}_\mathcal{R}}$ and a clustering method selected by the user. The leaves $\tilde{V}_{RNN}$ of the tree are new labels, which stand for regimes identified in the training set. In other words, the training set is relabeled. These new labels are represented in blue in figure III.2. Besides, the function sc_{RNN} is a scoring function, which induces a partial order relation in the space $\mathcal{X}$. Its computation is based on the tree tr_{RNN} as shown in figure III.2.

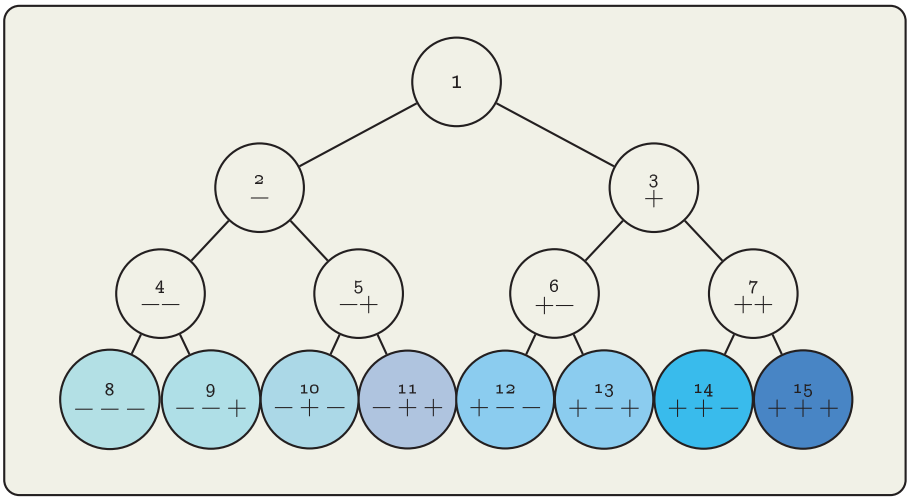


Figure III.2: **Partitioning of the Training Set**

Finally, the third step *1-NN Classification* attributes the new labels and the corresponding scores to the test set. In practice, the distances between a given object in the test set and each object of the training set are evaluated with the pseudometric $d_{RNN}^{\mathcal{T}_\mathcal{R}}$. The attributed label is the label of the closest object in the training set. In the following subsections, we present thoroughly the three steps of the procedure.

III.3.1 Step 1: Computation of the Pseudometric

New Representation of Hierarchical Partitioning Rules. A large hierarchical partitioning rule risks overfitting, while a rule that is too small may not capture all the relevant and available information. The tree-growing step is usually completed by a pruning step whose role is precisely to determine the optimal size of the tree. We propose here another method for constructing robust classifiers with additional functionalities. It aims at using the information provided by the tree-growing step regarding the relative importance of each split. We consider thus hierarchical partitioning rules as node-weighted directed acyclic graphs. Each node weight indicates the prediction ability of the associated split, which is measured by the decrease in the tree impurity. The following figures shows the difference between the classic representation of a hierarchical partitioning rule - figure III.3 - and the one we introduce - figure III.4.

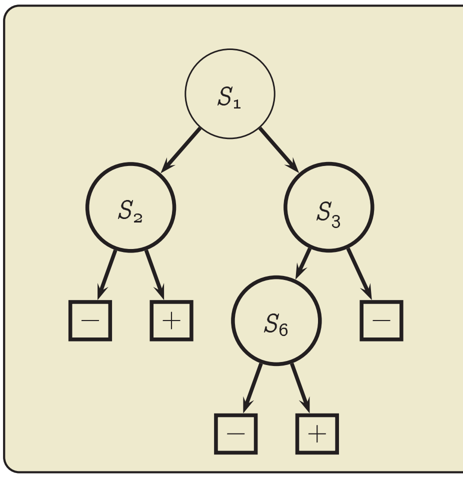


Figure III.3: **Classic Representation**

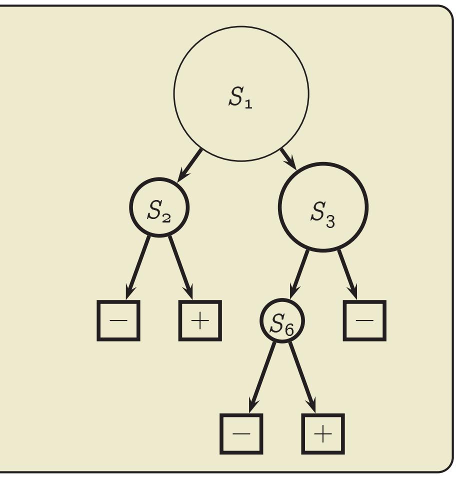


Figure III.4: **New Representation**

Classification Paths Based Pseudometric. Consider the hierarchical partitioning rule $(tr, \tilde{tr}, f)$, where $tr = (V, A)$ and $V = \{1, \ldots, |V|\}$. We denote L^{tr}_{RNN} the operator, which returns the **l**eaf of the tree tr attributed to the object x, once x has been classified by the rule.

$$L^{tr}_{RNN} = \begin{cases} \mathcal{X} & \longrightarrow \quad \{1, \ldots |V|\} \\ x & \longmapsto \quad L^{tr}_{RNN}(x). \end{cases}$$

We also define the operator C^{tr}_{RNN}, which returns the path of nodes visited by the object x when it is classified by the rule. This path of nodes is denoted $\{1 \xrightarrow{\star} L^{tr}_{RNN}(x)\}$.

$$C^{tr}_{RNN} = \begin{cases} \mathcal{X} & \longrightarrow \quad \mathcal{P}(\{1, \ldots, |V|\}) \\ x & \longmapsto \quad \{1 \xrightarrow{\star} L^{tr}_{RNN}(x)\}. \end{cases}$$

We call $C^{tr}_{RNN}(x)$ the **c**lassification path of the object x in the tree tr. Consider now the two objects x and x'. To compare their classification paths in the tree tr, we introduce the operator P^{tr}_{RNN}, which returns the **p**ath $L^{tr}_{RNN}(x) \xrightarrow{\star} L^{tr}_{RNN}(x')$ from the leaf $L^{tr}_{RNN}(x)$ to the leaf $L^{tr}_{RNN}(x')$.

$$P^{tr}_{RNN} = \begin{cases} \mathcal{X} \times \mathcal{X} & \longrightarrow \quad \mathcal{P}(\{1, \ldots, |V|\}) \\ (x, x') & \longmapsto \quad \{L^{tr}_{RNN}(x) \xrightarrow{\star} L^{tr}_{RNN}(x')\}. \end{cases}$$

In other words, $P^{tr}_{RNN}(x, x')$ is the sequence of nodes common to the classification paths of x and x' in tr. Besides, the relation between P^{tr}_{RNN} and C^{tr}_{RNN} is expressed as follows.

$$P^{tr}_{RNN}(x, x') = \left\{ C^{tr}_{RNN}(x) \bigtriangleup C^{tr}_{RNN}(x') \right\} \bigcup \max \left\{ C^{tr}_{RNN}(x) \bigcap C^{tr}_{RNN}(x') \right\},$$

where

$$\left\{C^{tr}_{RNN}(x) \triangle C^{tr}_{RNN}(x')\right\} = \left\{C^{tr}_{RNN}(x) \bigcup C^{tr}_{RNN}(x')\right\} \setminus \left\{C^{tr}_{RNN}(x) \bigcap C^{tr}_{RNN}(x')\right\}.$$

As an example, the classification paths of x and x' are respectively tagged in red and in yellow in the figures III.5 and III.6. In this case, $P^{tr}_{RNN}(x, x') = 1$.

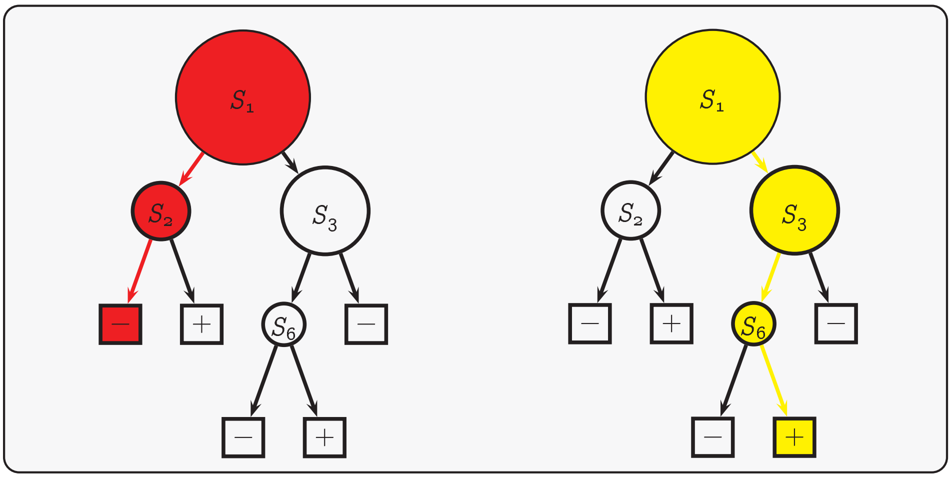


Figure III.5: **Classification Path for x** Figure III.6: **Classification Path for x'**

In addition, we introduce α^{tr} the function returning the weights of the nodes of the tree tr.

$$\alpha^{tr} = \begin{cases} V & \longrightarrow & \mathbb{R}_+ \\ v & \longmapsto & \alpha(v). \end{cases}$$

In the applications in chapter IV, we define α^{tr} using the decreases $(\Delta Q(sp_{v^\star}, v))_{v \in V}$ in the tree impurity defined with the Gini function.

$$\alpha^{tr}(v) = \Delta Q(sp_{v^\star}, v), \forall v \in V.$$

Finally, the pseudometric $d^{tr}_{RNN}(x, x')$ is computed by summing the weights $\alpha^{tr}(v)$ over the sequence of nodes $P^{tr}_{RNN}(x, x')$.

$$d^{tr}_{RNN}(x, x') = \sum \alpha^{tr}(v) \mathbb{1}_{\{v \in P^{tr}_{RNN}(x,x')\}}.$$

Let now $\mathcal{T}_\mathcal{R}$ be an ensemble of hierarchical partitioning rules generated by bagging. Straightforwardly, the pseudometric $d^{\mathcal{T}_\mathcal{R}}_{RNN}$ is the sum of the pseudometrics d^{tr}_{RNN} over all trees tr in the ensemble $\mathcal{T}_\mathcal{R}$.

$$d^{\mathcal{T}_\mathcal{R}}_{RNN}(x, x') = \sum_{(tr, \tilde{tr}, f) \in \mathcal{T}_\mathcal{R}} \alpha^{tr}(v) \mathbb{1}_{\{v \in P^{tr}_{RNN}(x,x')\}}.$$

As a conclusion, defining this pseudometric can also be seen as a method to re-encode the objects of the training set using their classification paths in the ensemble $\mathcal{T}_\mathcal{R}$. Indeed, the distance between the objects x and x' is made via the comparison of their corresponding series of classification paths $\left(C^{tr}_{RNN}(x)\right)_{(tr, \tilde{tr}, f) \in \mathcal{T}_\mathcal{R}}$ and $\left(C^{tr}_{RNN}(x')\right)_{(tr, \tilde{tr}, f) \in \mathcal{T}_\mathcal{R}}$.

III.3.2 Step 2: Relabeling the Training Set

This step aims at computing the ordered hierarchical partitioning rule $(tr_{RNN}, \tilde{tr}_{RNN}, sc_{RNN})$ using the training set, the pseudometric $d^{\mathcal{T}_\mathcal{R}}_{RNN}$ and a user-chosen clustering method. This is achieved in the two complementary sub-steps.

- Apply recursively a user-chosen clustering method to the matrix

$$\left(d^{\mathcal{T}_\mathcal{R}}_{RNN}(x,x')\right)_{x,x'\in x_{trn_{1:n_{trn}}}}.$$

  This produces the tree $tr_{RNN} = (V_{RNN}, A_{RNN})$ and a recursive partition of the training set, denoted $\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)_{v\in V_{RNN}}\right)$. This recursive partition is illustrated by the graphs III.8, III.9, III.10, III.11, III.12 and III.13. Each graph represents a specific subset of the **SP500** training set under the form of a hierarchical partitioning structure called dendrogram. These dendrograms are produced by this sub-step when used with a hierarchical agglomerative clustering method.

- Reorder iteratively the nodes $v \in V_{RNN}$ according to the proportion of positive labels corresponding to the subsets $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_L) \subset x_{trn_{1:n_{trn}}}$ and $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_R) \subset x_{trn_{1:n_{trn}}}$. By convention, the left child node and the right child node of v are respectively denoted v_L and v_R. For simplicity, we also denote $y_{v_L} \subset y_{trn_{1:n_{trn}}}$ and $y_{v_R} \subset y_{trn_{1:n_{trn}}}$ the labels of respectively $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_L)$ and $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_R)$. They satisfy the relation:

$$\frac{|\{i, y_{v_{L_i}} = +1\}|}{|y_{v_L}|} \leq \frac{|\{i, y_{v_{R_i}} = +1\}|}{|y_{v_R}|}.$$

  The score is computed in parallel on the recursive partition of training set. It is defined by the following recurrence relations - illustrated by figure III.7.

$$\begin{cases} sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 1)\right) & = sc_{RNN}\left(x_{trn_{1:n_{trn}}}\right) = \frac{1}{2} \\ sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_L)\right) & = sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right) - \frac{1}{2^{b(v)+1}} \\ sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_R)\right) & = sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right) + \frac{1}{2^{b(v)+1}} \end{cases}$$

  where b returns the level of any node v in the tree tr_{RNN}:

$$b = \begin{cases} \mathbb{N} & \longrightarrow & \mathbb{N} \\ n & \longmapsto & \min\{m \in \mathbb{N}, 2^{m-1} \leq n \leq 2^m - 1\}. \end{cases}$$

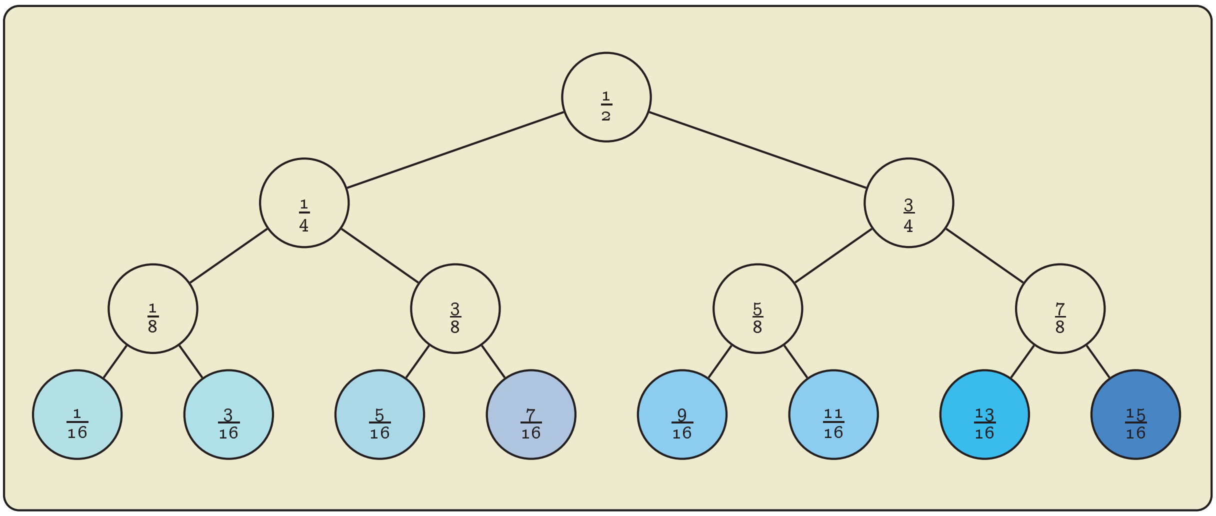


Figure III.7: **Computation of the Score on the Training Set**

The outputs of these combined sub-steps are:

- the leaves $\tilde{V}_{RNN}$ which are considered as new labels of the training set. We say that they relabel the training set. For example, the figure III.14 compares the original labels of the **SP500** training set, where labels $+1$ are represented in blue and labels -1 in green, with the new RNN labels. Note that the recursive partitioning stops only when the size $|\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}})|$ of the subsets equals 1. Hence the large number of new labels.

- the score defined on the training set such that:

$$s(x) = \sum_{v \in C_{RNN}^{tr_{RNN}}(x)} \frac{(-1)^{v-1}}{2^{b(v)}}, \forall x \in x_{trn_{1:n_{trn}}}.$$

The computation of the score of the **SP500** training set is illustrated with figure III.15.

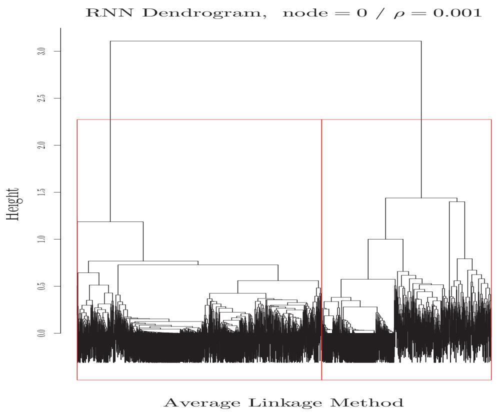


Figure III.8: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 1)$

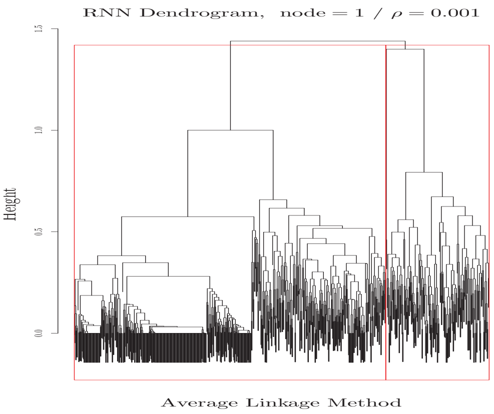


Figure III.9: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 2)$

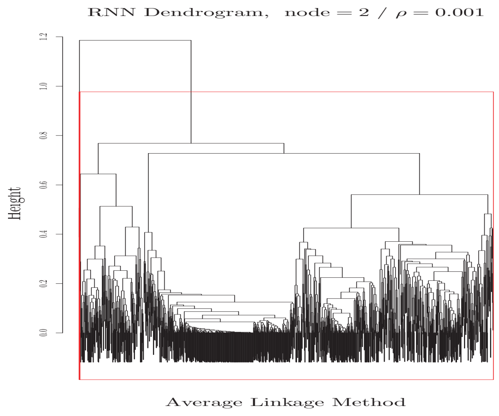


Figure III.10: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 3)$

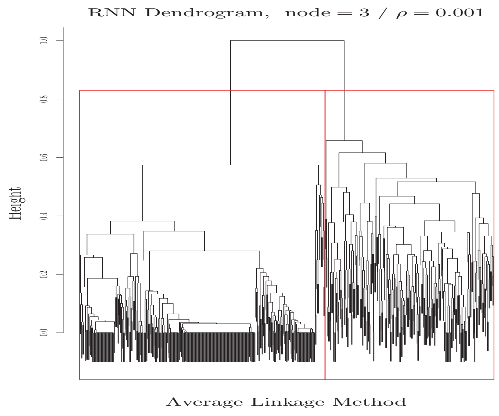


Figure III.11: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 4)$

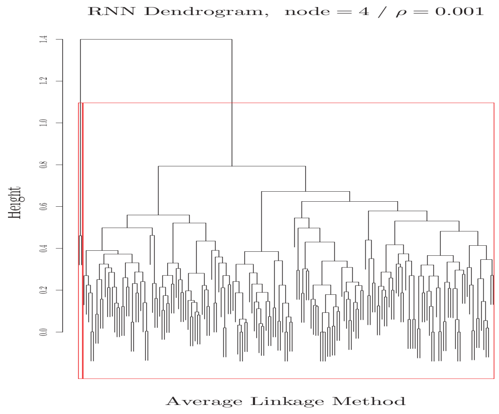


Figure III.12: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 3)$

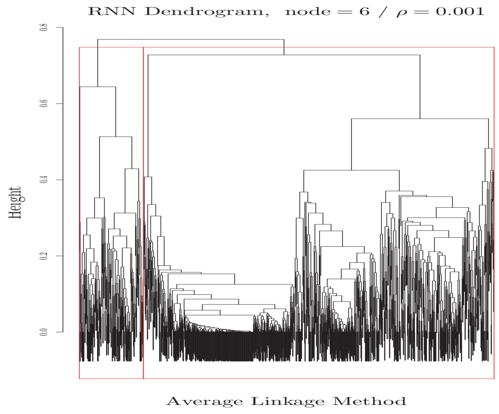


Figure III.13: $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 6)$

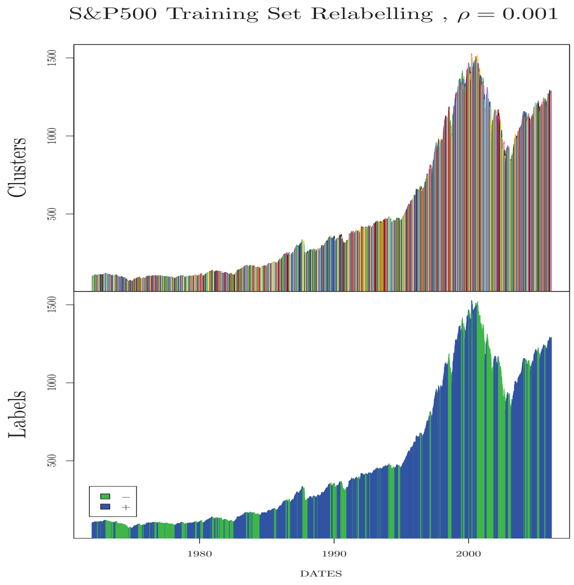


Figure III.14: **SP500 Training Set Relabeling**

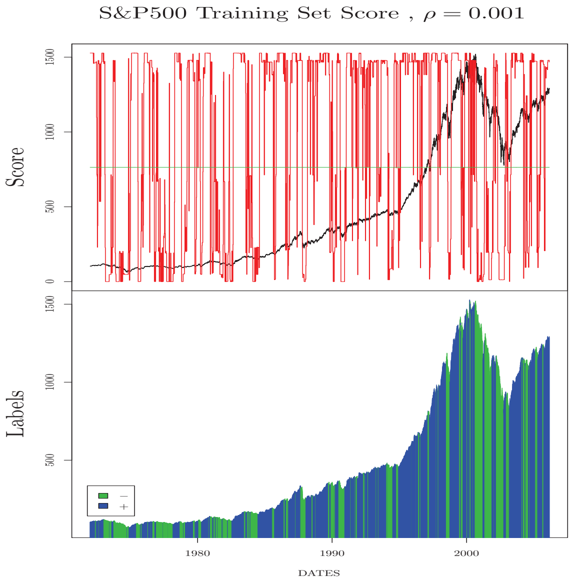


Figure III.15: **Score Computation**

Clustering Methods. A clustering method is required for fulfilling the relabeling task. Generally speaking, clustering is a field of unsupervised learning [HTF01]. It aims at identifying in a database groups of objects or clusters, which are homogeneous according to a given similarity measure. It has inspired many analyses and procedures [Har75, And73, Mur85, Gor99, HTF01]. Yet, the question of its theoretical foundations has emerged only recently [VLBD05, BDvLP06]. Properties of stability, convergence and consistency of various clustering methods are currently under active investigation [vL09, vLBB08, BvL09, CM10, PP11]. We focus here on Hierarchical Agglomerative Clustering (HAC) [MRS08] and Spectral Clustering [vL07].

Hierarchical Agglomerative Clustering. Hierarchical Agglomerative Clustering (HAC) is an iterative bottom-up procedure for merging clusters, which produces a hierarchical partition of data called dendrogram [MRS08]. It requires a linkage criterion, which determines a distance or a similarity measure between two finite sets. The HAC procedure is initialized by considering each object of the training set as a cluster. The most similar pair of clusters is then found iteratively using the linkage criterion and merged into a new cluster. The procedure stops when all the objects of the training set are in the same cluster. A dendrogram is produced, which recapitulates the consecutive cluster agglomerations [CM10, JD88].

The corresponding pseudocode, stated in [MRS08], is reproduced here. We denote $Sim = (Sim(i,j))_{1\leq i\leq n_{trn}, 1\leq j\leq n_{trn}}$ the **sim**ilarity matrix of the training set, where Sim is initialized as follows:

$$Sim(i,j) = d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{trn_j}).$$

Besides, we need to maintain temporary programming variables. Let Sim^{sort} be a vector of vectors initialized as the matrix Sim, where each row $\big(Sim(i,j)\big)_{1\leq j\leq n, j\leq i}$ has been sorted in the increasing order. We denote Sim^{index} the corresponding indices of Sim in the reordered matrix Sim^{sort}. We also denote Cls a set of active **cl**usters, which are updated during the procedure. The output dendrogram is denoted A_{dendro}. Now, we also need to choose a linkage criterion. A **link**age criterion is a function $Link$

$$Link = \begin{cases} \mathcal{P}\{x_{trn_{1:n_{trn}}}\} \times \mathcal{P}\{x_{trn_{1:n_{trn}}}\} & \longrightarrow \quad \mathbb{R}_+ \\ (A,B) & \longmapsto \quad Link(A,B), \end{cases}$$

which depends on a similarity measure that we denote $Dist$. The function $Link$ is usually chosen among the following criteria (this list is non-exhaustive):

$$\begin{cases} \text{the single - linkage criterion :} & Link(A,B) = min_{x \in A,\, x' \in B} Dist(x,x'). \\ \text{the complete - linkage criterion :} & Link(A,B) = max_{x \in A,\, x' \in B} Dist(x,x'). \\ \text{the average linkage criterion :} & Link(A,B) = \frac{1}{|A||B|} \sum_{x \in A, x' \in B} Dist(x,x'). \\ \text{the average group linkage criterion :} & Link(A,B) = \frac{1}{|A \cup B|} \sum_{(x,x') \in (A \cup B)^2} Dist(x,x'). \end{cases}$$

The HAC procedure consists of the following steps.

1. *Initialization:*
   - *for each object* $i \in \{1, \ldots, n_{trn}\}$, $Cls(i) = 1$.
   - *Compute* Sim^{sort} *and* Sim^{index}.
   - $A_{Dendro} = \emptyset$.
2. *For* $i \in \{1, \ldots, n_{trn}\}$,
   - *Find the most similar pair of clusters:* $i^\star = \arg\min_{Cls(i)=1} Sim^{sort}(i,.)$.
   - $j^\star = Sim^{index}(i^\star, \arg\min(Sim^{sort}(i^\star,.)))$.
   - *Update dendrogram:* $A_{dendro} = A_{dendro} \cup \{(i^\star, j^\star)\}$.
   - *Deactivate cluster* j^{star}*:* $Cls(j^\star) = 0$.
   - *Update* Sim^{sort}*.*
     For $ii \in \{1, \ldots, n_{trn}\}$, $Cls(ii) = 1$, $ii \neq i^\star$,
     - *Delete values* $Sim(ii, i^\star)$ *and* $Sim(ii, j^\star)$ *in the vector* $Sim^{sort}(ii,.)$.
     - *Update Sim:* $Sim(ii, i^\star) = Link(Sim(ii, i^\star), Sim(ii, j^\star))$ *and* $Sim(i^\star, ii) = Sim(ii, i^\star)$.
     - *Insert new value* $Sim(ii, i^\star)$ *in the vector* $Sim^{sort}(ii,.)$ *and* $Sim(i^\star, ii)$ *in* $Sim^{sort}(i^\star,.)$*, respecting the increasing order.*
3. *Output* A_{dendro}.

Cutting the dendrogram at a specific level produces a flat clustering in a certain number of clusters. Likewise, it is straightforward to select the cutting point that produces a user-defined number of clusters. These procedures have recently been reformulated in a specific framework, where properties of stability and convergence have been proved [CM10]. Note finally that the second step of the RNN procedure systematically requires two clusters as outputs.

Spectral Clustering. Spectral Clustering aims at producing a user-defined number *cl* of **cl**usters by using the eigenvectors of the Laplacian graphs, computed on the training set. Defining Laplacian graphs requires introducing the following notations. The similarity matrix of the training set is denoted as previously $Sim = \big(Sim(i,j)\big)_{1 \leq i \leq n_{trn}, 1 \leq j \leq n_{trn}}$, where for more simplicity:

$$Sim(i,j) = d^{\mathcal{T_R}}_{RNN}(x_{trn_i}, x_{trn_j}).$$

The diagonal matrix $diag_{Sim} = \big(diag_{Sim}(i,j)\big)_{1 \leq i \leq n_{trn}, 1 \leq j \leq n_{trn}}$, where

$$diag_{Sim}(i,i) = \sum_{j=1}^{n} Sim(i,j).$$

As in [vL07], we distinguish three different Laplacian graphs:

- the unnormalized graph **Lap**lacian, defined as

$$\Delta^0_{lap} = diag_{Sim} - Sim$$

- the normalized graph **Lap**lacians, defined as

$$\begin{cases} \Delta^1_{lap} & = diag_{Sim}^{-\frac{1}{2}} \Delta^0_{lap} diag_{Sim}^{-\frac{1}{2}} = I - diag_{Sim}^{-\frac{1}{2}} Sim\, diag_{Sim}^{-\frac{1}{2}} \\ \Delta^2_{lap} & = diag_{Sim}^{-1} \Delta^0_{lap} = I - diag_{Sim}^{-1} Sim \end{cases}$$

  where I is the identity matrix.

Once a version of the graph Laplacian has been chosen, the projections of the objects of the training set on the first cl eigenvectors are computed. The projected objects are then clustered into cl clusters using K-means. We now reproduce the pseudocode of the spectral clustering procedures as stated in [vL07]. We distinguish between unnormalized spectral clustering and normalized spectral clustering using the graph Laplacians Δ^1_{lap} [NJW01] and Δ^2_{lap} [SM00]. The inputs are the similarity matrix Sim and the number cl_{eig} of clusters.

1. *Compute the graph Laplacian* Δ_{lap}*,* $i \in \{0, 1, 2\}$*.*
2. *Compute the first* cl_{eig} *eigenvectors* $eig^1, \dots, eig^k$ *of* Δ^i_{lap}*.*
3. *Let* $eig \in \mathbb{R}^{n \times k}$ *be the matrix containing the vectors* $eig^1, \dots, eig^k$ *as columns.*
4. *If* $i = 1$*, normalize the rows of eig:* $eig_{ij} \leftarrow \frac{eig_{ij}}{(\sum_{k=1}^n eig_{ik}^2)^{\frac{1}{2}}}$*.*
5. *Define the vectors* $\left(eig_i\right)_{i \in \{1,\dots,n\}}$ *corresponding to the rows of eig.*
6. *Cluster* $\left(eig_i\right)_{i \in \{1,\dots,n\}}$ *into cl clusters using K-means.*
7. *Output the resulting clusters.*

Note finally that properties of convergence of the graph Laplacian have been established in [vLBB08] and [PP11].

III.3.3 Step 3: 1-NN Classification

The hierarchical partitioning rule $(tr_{RNN}, \tilde{tr}_{RNN}, sc_{RNN})$ is computed at step 2 on the training set. It is extended as follows to the space $\mathcal{X}$.

- The recursive partition $\left(\tilde{tr}_{RNN}(\mathcal{X}, v)_{v \in V_{RNN}}\right)$ of the space $\mathcal{X}$ is defined via 1-NN classification applied to the recursive partition $\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)_{v \in V_{RNN}}\right)$ of the training set.

$$\tilde{tr}_{RNN}(\mathcal{X}, v) = \left\{x \in \mathcal{X}, \underset{x' \in \tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)}{\arg\min} d^{\mathcal{T_R}}_{RNN}(x, x') \in \tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right\}.$$

  Particularly, we call regimes the subsets corresponding to the leaves of this rule

$$\left(\tilde{tr}_{RNN}(\mathcal{X}, v)\right)_{v \in \tilde{V}_{RNN}},$$

  which form a partition of $\mathcal{X}$.

- The score sc_{RNN} is likewise straightforwardly extended to a scoring function $\mathcal{X} \longrightarrow [0, 1]$, which induces a partial order relation in the space $\mathcal{X}$.

$$s(x) = \sum_{v \in C^{tr_{RNN}}_{RNN}(x)} \frac{(-1)^{v-1}}{2^{b(v)}}, \forall x \in \mathcal{X},$$

Note that the computation of $s(x)$ relies solely on $C_{RNN}^{tr_{RNN}}(x) = \{1 \overset{\star}{\to} L_{RNN}^{tr}(x)\}$, hence on $L_{RNN}^{tr}(x)$, which is a leaf of the tree tr_{RNN}. In other words, we have the relation:

$$s(x) = s(x'),\ \forall x \in \tilde{tr}(\mathcal{X}, v),\ \forall x' \in \tilde{tr}(x_{trn_{1:n_{trn}}}, v),\ v \in \tilde{V}_{RNN}.$$

The value of the score characterizes therefore the regime.

Finally, the Relabeled Nearest Neighbors procedure is a new meta-algorithm which aims at identifying and at predicting regimes. It is proven in the next chapter that it yields very good and stable results. It notably exhibits an outstanding reduction of overfitting, compared to the Random Forests.

III.3.4 Input-Output Diagram of the RNN Procedure

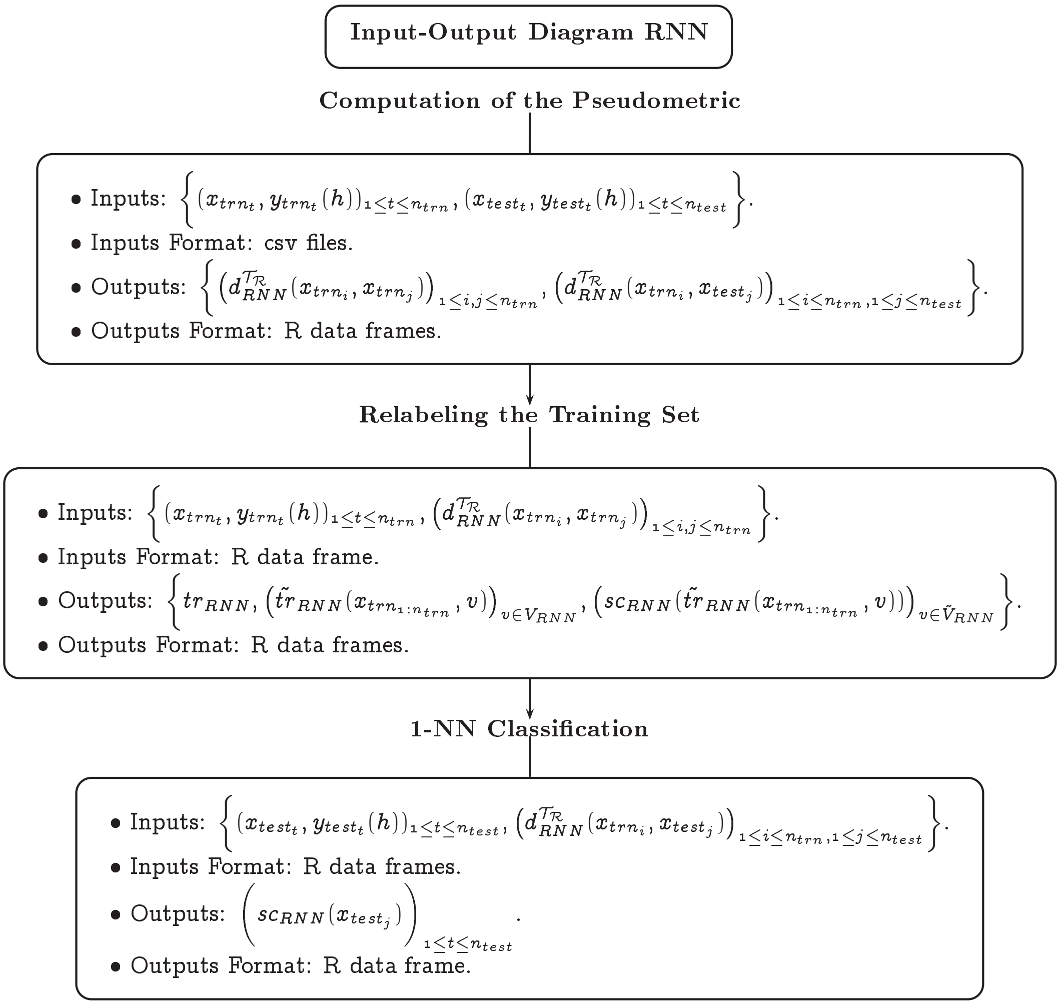
The following diagram recapitulates the three steps of the RNN procedure, implemented with R. In practice, splitting rules are stumps, i.e. tree classifiers of size 3 (one parent node and two child nodes). Besides, the hierarchical partitioning rules of step 1 are tree classifiers generated by bagging. The inputs of the procedure are the labeled training and test sets.

$$\left\{(x_{trn_t}, y_{trn_t}(h))_{1\le t\le n_{trn}}, (x_{test_t}, y_{test_t}(h))_{1\le t\le n_{test}}\right\}.$$

Note that the labels are computed as explained in subsection II.2.1, using the linear trends of the target variable.

$$y_t(h) = \operatorname{sgn}\left(T^1_{t+h,s^\star,k^\star} - T^1_{t,s^\star,k^\star}\right),$$

where $s^\star$ is user-defined, $k^\star$ is the most recent lag at scale $s^\star$ and h is the horizon of prediction.



Input-Output Diagram RNN

Computation of the Pseudometric

- Inputs: $\left\{(x_{trn_t}, y_{trn_t}(h))_{1\leq t\leq n_{trn}}, (x_{test_t}, y_{test_t}(h))_{1\leq t\leq n_{test}}\right\}$.
- Inputs Format: csv files.
- Outputs: $\left\{\left(d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{trn_j})\right)_{1\leq i,j\leq n_{trn}}, \left(d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{test_j})\right)_{1\leq i\leq n_{trn}, 1\leq j\leq n_{test}}\right\}$.
- Outputs Format: R data frames.

Relabeling the Training Set

- Inputs: $\left\{(x_{trn_t}, y_{trn_t}(h))_{1\leq t\leq n_{trn}}, \left(d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{trn_j})\right)_{1\leq i,j\leq n_{trn}}\right\}$.
- Inputs Format: R data frame.
- Outputs: $\left\{tr_{RNN}, \left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right)_{v\in V_{RNN}}, \left(sc_{RNN}(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v))\right)_{v\in \tilde{V}_{RNN}}\right\}$.
- Outputs Format: R data frames.

1-NN Classification

- Inputs: $\left\{(x_{test_t}, y_{test_t}(h))_{1\leq t\leq n_{test}}, \left(d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{test_j})\right)_{1\leq i\leq n_{trn}, 1\leq j\leq n_{test}}\right\}$.
- Inputs Format: R data frames.
- Outputs: $\left(sc_{RNN}(x_{test_j})\right)_{1\leq t\leq n_{test}}$.
- Outputs Format: R data frame.

The pseudocodes corresponding to these three steps are as follows.

Pseudocode - Computation of the Pseudometric.

```
1. Generate the ensemble $\mathcal{T}_{\mathcal{R}}$ of bagged tree classifiers using the training set $(x_{trn_t}, y_{trn_t}(h))_{1 \leq t \leq n_{trn}}$
   and the CART algorithm with the Gini function. We get:
       $$\alpha^{tr}(v) = \Delta Q(sp_{v^\star}, v), \forall v \in V,$$
   where $tr = (V, A) \in \mathcal{T}_{\mathcal{R}}$.
2. Compute the pseudometric on the training set.
       $$d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{trn_j}) = \sum_{(tr,\tilde{tr},f) \in \mathcal{T}_{\mathcal{R}}} \alpha^{tr}(v) \mathbb{1}_{\{v \in P^{tr}_{RNN}(x_{trn_i}, x_{trn_j})\}}.$$
3. Compute the pseudometric between the training set and the test set.
       $$d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{test_j}) = \sum_{(tr,\tilde{tr},f) \in \mathcal{T}_{\mathcal{R}}} \alpha^{tr}(v) \mathbb{1}_{\{v \in P^{tr}_{RNN}(x_{trn_i}, x_{test_j})\}}.$$
4. Outputs:
       $$\left\{ \left( d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{trn_j}) \right)_{1 \leq i,j \leq n_{trn}}, \left( d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{test_j}) \right)_{1 \leq i \leq n_{trn}, 1 \leq j \leq n_{test}} \right\}.$$
```

Pseudocode - Relabeling the Training Set.

```
1. Initialization:
   • $v = 0$, $nextv = 1$.
   • Enter
       $$\begin{cases} V_{RNN} & = \{1\} \\ A_{RNN} & = \emptyset \\ tr_{RNN} & = (V_{RNN}, A_{RNN}) \\ \tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, 1) & = x_{trn_{1:n_{trn}}} \\ sc_{RNN}\left(x_{trn_{1:n_{trn}}}\right) & = \frac{1}{2} \end{cases}$$
   • Choose a clustering method.
2. While $v < nextv$
   • If $|\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)| > 3$
       - Apply the clustering method to $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)$.
         Produce two clusters $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_L)$ and $\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_R)$ such that:
           * the following relation is checked.
               $$\frac{|\{i, y_{v_{L_i}} = +1\}|}{|y_{v_L}|} \leq \frac{|\{i, y_{v_{R_i}} = +1\}|}{|y_{v_R}|}.$$
           * $v_L$ and $v_R$ are respectively the left child node and the right child node of $v$.
               $$\begin{cases} V_{RNN} & = V_{RNN} \bigcup \{v_L\} \bigcup \{v_R\}, \\ A_{RNN} & = A_{RNN} \bigcup \{v v_L\} \bigcup \{v v_R\}. \end{cases}$$
```

```
            - Compute the scores corresponding to these new nodes.
              $$\begin{cases} sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_L)\right) & = sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right) - \frac{1}{2^{b(v)+1}} \\ sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v_R)\right) & = sc_{RNN}\left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right) + \frac{1}{2^{b(v)+1}} \end{cases}$$
            - Update $nextv = nextv + 2$.
        • Update $v$: $v = v + 1$.
    3. Outputs:
        $$\left\{tr_{RNN}, \left(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)\right)_{v \in V_{RNN}}, \left(sc_{RNN}(x_{trn_{1:n_{trn}}}, v)\right)_{v \in \tilde{V}_{RNN}}\right\}.$$
```

Pseudocode - 1-NN Classification.

```
    1. For $j = 1$ to $n_{test}$
    2. Find $i^\star$
        $$i^\star = \underset{i \in [1, n_{trn}]}{\arg\min}\, d^{\mathcal{T}_{\mathcal{R}}}_{RNN}(x_{trn_i}, x_{test_j}).$$
    3. Attribute $v^\star \in \tilde{V}_{RNN}$ to the object $x_{test_j}$ such that $i^\star \in \tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v)$.
    4. Attribute $sc_{RNN}(x_{test_j}) = sc_{RNN}(\tilde{tr}_{RNN}(x_{trn_{1:n_{trn}}}, v^\star))$.
    5. Outputs:
        $$\left(sc_{RNN}(x_{test_j})\right)_{1 \leq j \leq n_{test}}.$$
```

Chapter IV

Backtesting and Numerical Results

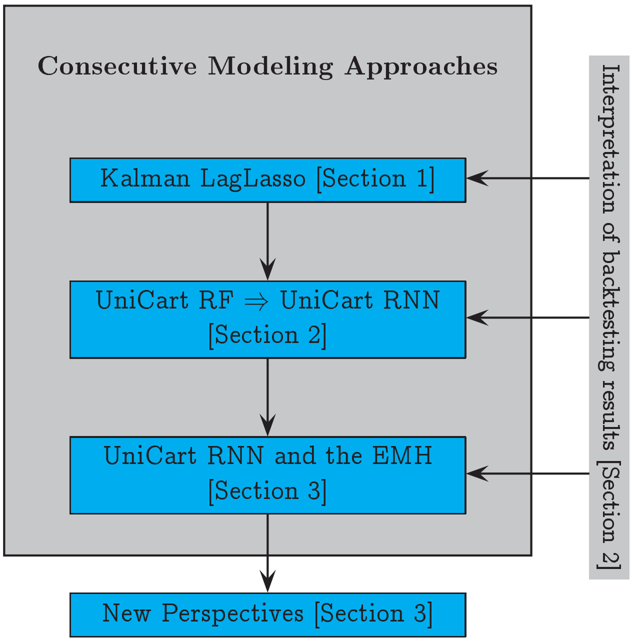
I only believe in statistics that I doctored myself

Winston S. Churchill

Methods of database and model construction are reported in chapters II and III. They have been consecutively developed to meet the improvements suggested by the module *Backtesting and Numerical Results* (cf. the experimentation protocol - figure I.14 in subsection I.3.1). The goal of this chapter is to introduce this module and to present the results obtained with a group of key financial target variables composed of stocks, commodities and exchange rates. The first components of the backtesting module are two generic backtesting plans, which evaluate the accuracy of any trend predictive numerical model over a "significant" time period.

- The first plan computes the model recognition rate of upward and downward trends at a user-chosen horizon and scale.
- The second plan designs trading rules using predictions made over the test set. The underlying idea is the following. Imagine a trader who can decide, at each moment, to sell or to buy one unit of the target variable, or to keep his position unchanged. If the prediction of the next trend is positive, the trader buys; if it is negative, he sells. This trading rule yields a profit and loss account (P&L), which is the cumulated earned money over time.

In addition, interpretation functionalities, such as measures of feature prediction ability and measures of model and prediction reliability, help to analyze the decision mechanism of the numerical model. The decisive role played by these interpretation functionalities in the switch from one modeling approach to another is clarified in this chapter. Another important aspect of the backtesting analysis concerns the dramatic influence of the parameter computation procedures - at each step of the experimentation protocol - on the formulation of conclusion statements. This methodological difficulty is particularly highlighted, as it has led to the design of the Relabeled Nearest Neighbors (RNN) procedure. This procedure has then been used to study the Efficient Market Hypothesis (EMH), what is illustrated. The plan of this chapter is as follows. The first section is devoted to the backtesting analysis of Kalman LagLasso (KLL) predictions of **SP500** monthly variations. These real-valued predictions are calculated using a pool of filtered and lagged explanatory variables, which are considered on a five years sliding window moving across a twenty years historical data range. Results, i.e. trend recognition rates and P&Ls computed over this period, are summed up for different KLL and classic linear models. The selection of lagged variables by specific KLL models during the backtesting is particularly investigated. It is shown that if well-chosen KLL models outperform their competitors, they also suffer from drastic limits concerning prediction accuracy, interpretation functionalities and parameter tuning. The prediction level of precision is therefore lowered in the next sections, as we switch from the regression setup to the binary classification setup. In addition, the UniCart representation procedure is introduced to provide more interpretable and intuitive features and labels. The second section focuses on the construction of trend predictive numerical models using the Random Forests (RF) and the Relabeled Nearest Neighbors (RNN) procedures with the two types of UniCart databases. The interpretation functionalities of both procedures are used to upgrade the decision-making of the trading rules and to identify the most important features. Backtesting analyses are performed over a period of five years for the group of target variables introduced in subsection II.1.1. They highlight the overfitting of Random Forests numerical models, and show that it is due to the size of the tree classifiers. Conversely, they prove that the RNN procedure, which does not require any parameter tuning procedure, is not submitted to this overfitting phenomenon. In the third section, the specific backtesting results of the RNN numerical models are thoroughly studied. The efficient market hypothesis is questioned using RNN numerical models, whose trading decisions have been delayed. In addition, the relabeling of the training set and the 1-NN classification of the test set are meticulously illustrated using the application examples of the second section. New research directions for designing trend predictive models are finally discussed.


Consecutive Modeling Approaches
Kalman LagLasso [Section 1]
UniCart RF ⇒ UniCart RNN [Section 2]
UniCart RNN and the EMH [Section 3]
New Perspectives [Section 3]
Interpretation of backtesting results [Section 2]

IV.1 The Prospective Kalman LagLasso Approach

In this section, we use the Kalman LagLasso (KLL) approach, presented in section III.1, to compute predictions of monthly movements of the **SP500** index. Modeling choices, hypotheses and parameters are specified in the first subsection. The Kalman LagLasso model and the related calibration methods are also very briefly recalled. The second subsection is devoted to the presentation of backtesting results. In practice, different KLL numerical models, corresponding each to a specific value of maximal lag, are calibrated using a pool of factors considered on a time window of five years. Each of them computes a prediction of the variation of the **SP500** index over the next month. A significant number of successive predictions is produced by shifting the time window over the last 20 years of **SP500** monthly variations. Their accuracy is evaluated via the backtesting plans. Recognition rates as well as P&Ls of KLL numerical models are shown to compare favorably to those obtained with classic state-space models. In addition, these results highlight the importance of the Kalman filtering step. They also suggest that the choice of the regression setup may be too ambitious. Besides, straightforward measures of factor importance, based on the relative relevance of the iteratively selected variables, are also computed. They finally point out the limits of the KLL approach in terms of interpretation functionalities.

IV.1.1 SP500 KLL Numerical Models

We work with the modeling choices and hypotheses stated in subsection III.1.1. We aim at predicting the real-valued monthly variation of the **SP500** index at horizon $h = 1$ month. We assume that this variation can be decomposed into an inner component, which is due solely to the **SP500** index, and an exogenous component, which can be explained by a small subset of macro-economic and financial factors. They are picked up among the following US variables, introduced in the subsection II.1.1.

- the Monthly US 10-Year Treasury Constant Maturity Rate (**GS10**).
- the Monthly US 3-Month Treasury Bill (**TB3**).
- the **SP500** Price-to-Earnings Ratio (**PER**).
- the US Unemployment Rate (**UNEMPLOY**).
- the NAPM, ISM Manufacturing, Purchasing Managers' Composite Index (**NAPM**).
- the Disposable Personal Income (**INCOME**).
- the Monthly Price of Gold, in London, afternoon fixing (**GOLD**).
- the Monthly Spot Oil Price, West Texas Intermediate, in dollars per barrel (**OIL**).
- the US Population (**POPULATION**).
- the Consumer Price Index For All Urban Consumers (**CPIAUCSL**).
- the 30-Year Conventional Mortgage Rate (**MORTG**).

Note that these variables need to be chosen carefully. We observed indeed that having too many correlated factors in the database is usually very counter-productive. In addition, we assume that the influence of each of these factors can change over time and can be lagged. Finally, the **SP500** variation is supposed to be particularly sensitive to unexpected variations of the factors, that is to say to large deviations of the explanatory variables from their own trend. The Kalman LagLasso modeling approach is based on these hypotheses. We denote Y_t the **SP500** monthly variations and $X_t = \{X_t^1, \dots, X_t^p\}$ the random vector of dimension p, which represents the monthly variations of the explanatory variables. The variables X_t and Y_t are filtered via a specific linear state-space model, directly characterized by the matrices Γ_t and Π_t of equations II.1 and II.2 in subsection II.1.3. We consider the following very simple models.

- **State-Space Model 1 (SSM1)** also called local level model or Arima(0,1,1): $\Gamma_t = \Pi_t = 1$.
- **State-Space Model 2 (SSM2)** also called local linear trend model or Arima(0,2,2): $\Gamma_t = 1$ and $\Pi_t = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

The innovation residuals of the variables X_t and Y_t are computed with the Kalman filtering algorithm. Referring to the equations II.1 and II.2 in subsection II.1.3, we assume that the observation and evolution covariance matrices σ_{V_t} and σ_{W_t} check:

$$\begin{cases} \sigma_{V_t} &= \sigma_V \\ \sigma_{W_t} &= \sigma_W . \end{cases}$$

They are estimated via the L-BFGS-B quasi-Newton optimization algorithm (cf. [LN89, RHBZ95]), which is a maximum likelihood strategy. Besides, we denote $\tilde{Y}_t$ the residual of the **SP500** monthly variation. The residuals of the variations of the explanatory variables are likewise denoted $\tilde{X}_t = \{\tilde{X}_t^1, \dots, \tilde{X}_t^p\}$. In our case of multivariate time series, one could wonder whether to use in practice a different filter for each individual time series or the same filter. In our implementation, we used one state-space model for the response Y_t and the same state-space model for all factors related to X_t^k, where $k = 1, \dots, p$, for simplicity of use. We observe thus the random pair $(\tilde{X}_t, \tilde{Y}_t)$ over time $t = 1, \dots, n$ and aim at predicting $\tilde{Y}_{n+1}$. The KLL temporal regression model is as in equation III.1 in subsection III.1.1.

$$\tilde{Y}_{t+1} = \sum_{i=1}^{p} \beta_i \tilde{X}^i_{t-\sigma(i)}.$$

The lag of the i^{th} variable is the integer $\sigma(i)$, which checks $0 \leq \sigma(i) \leq \varSigma$. In the next subsection, we focus on four Kalman LagLasso numerical models, corresponding to the maximal lags $\varSigma = 1, 3, 6, 12$. The lags as well as the regression coefficients represented by the real vector $\beta = (\beta_1, \dots, \beta_d)$ are estimated by the LagLasso procedure (cf. subsection III.1.3). The next subsection presents the backtesting plans and results specific to the different Kalman LagLasso models and to classic linear models.

IV.1.2 Backtesting Plans and Results

The backtesting plans aim at evaluating the precision of the Kalman LagLasso numerical models and at exploring further refinements. They are based on the same principle: considering the last 20 years of **SP500** monthly variations, we use a sliding window of five years to make a prediction of the variation of the **SP500** index over the next month. A number of successive predictions at horizon $h = 1$ month are obtained and compared with those computed with classic linear univariate and multivariate models. The following models are thus studied to clarify the role played by the filtering step and by the variable and lag selection step.

- SSM1: $\varGamma_t = \varPi_t = 1$. No explanatory variables are used.
- SSM2: $\varGamma_t = 1$ and $\varPi_t = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$. No explanatory variables are used.
- KLL models for $\varSigma = 1, 3, 6, 12$: the **SP500** only is filtered using SSM2, while all the explanatory variables are filtered using SSM1.
- Regression-based and Lasso-based models: the **SP500** only is filtered using SSM2 while the explanatory variables are not. Variable selection is performed through Regression or Lasso.

The first backtesting plan consists in computing a recognition rate of upward and downward variations of the **SP500** depending on the amplitude of the variation of the index. The results presented in the table IV.1.2 clearly show that the Kalman LagLasso models outperform the other ones. These results are completed by the figures IV.1, IV.2, IV.3 and IV.4. They display the kernel estimated density of the conditional variables $y_t|(\text{sgn}(\tilde{y}_t) = 1)$ and $y_t|(\text{sgn}(\tilde{y}_t) = -1)$, where y_t stand for the observed variations of the **SP500** index and $\tilde{y}_t$ for the predictions computed by KLL models for the different maximal lags. These figures prove that KLL models are particularly efficient for the detection of downward trends of great amplitude. Yet, they perform poorly when confronted with upward trends of great amplitude. The second backtesting plan implements a straightforward trading rule based on the predictions of the **SP500** variations. Imagine a trader who sells or buys one unit of the **SP500** index every month. If the prediction of the model for next month is positive, the trader buys. If it is negative, he sells. At the end of the backtesting period, the gained P&Ls are compared on figure IV.5: State Space Models 1 and 2 are represented in green, Regression and Lasso- based Models in red and Kalman LagLasso Models in black. Note that the gains obtained with LagLasso models are all higher than gains obtained with other models. Besides, State Space Model 2 performs much better than State Space Model 1. This underlines the essential role played by the filtering step. Finally, the importance and predictive ability of each factor in KLL models is analyzed with the following simple indicators. The figures IV.6, IV.7, IV.8 and IV.9 display thus the occurrences of each factor for respectively $\varSigma = 1, 3, 6, 12$ months. These graphs give a hint about the relative importance of the factors but not necessarily about their relative relevance. They are therefore completed by the tables IV.2, IV.3, IV.4 and IV.5. These latter indicate whether the predictions delivered by each factor, i.e. $\{\beta_i \tilde{x}^i_{t-\sigma(i)}\}$, are of the same sign as the targeted labels $\{\tilde{y}_{t+1}\}$. They allow us therefore to distinguish between the number of right predictions and the number of wrong predictions per factor. They show that despite the good results obtained by the KLL numerical models, the recognition rate of every factor is quite poor.

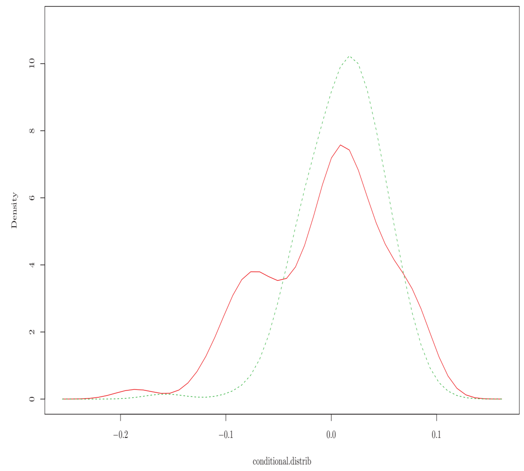


Figure IV.1: $y_t|(\mathrm{sgn}(\tilde{y}_t)), \Sigma = 1$

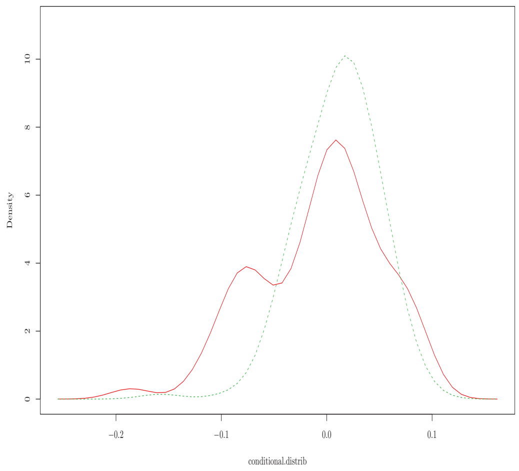


Figure IV.2: $y_t|(\mathrm{sgn}(\tilde{y}_t)), \Sigma = 3$

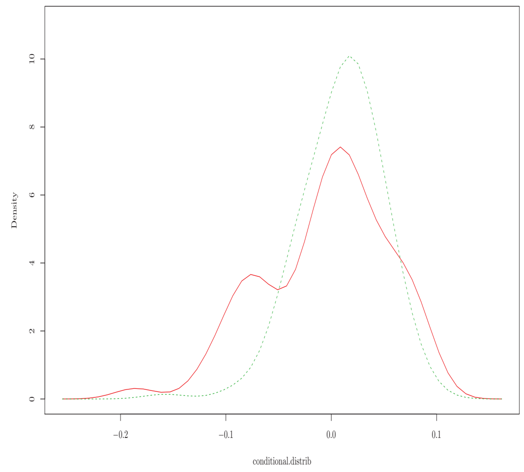


Figure IV.3: $y_t|(\mathrm{sgn}(\tilde{y}_t)), \Sigma = 6$

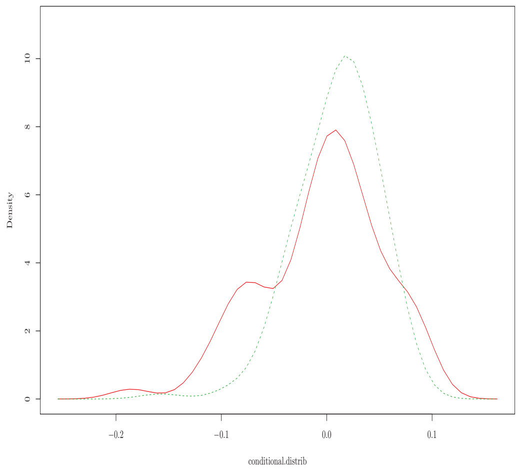
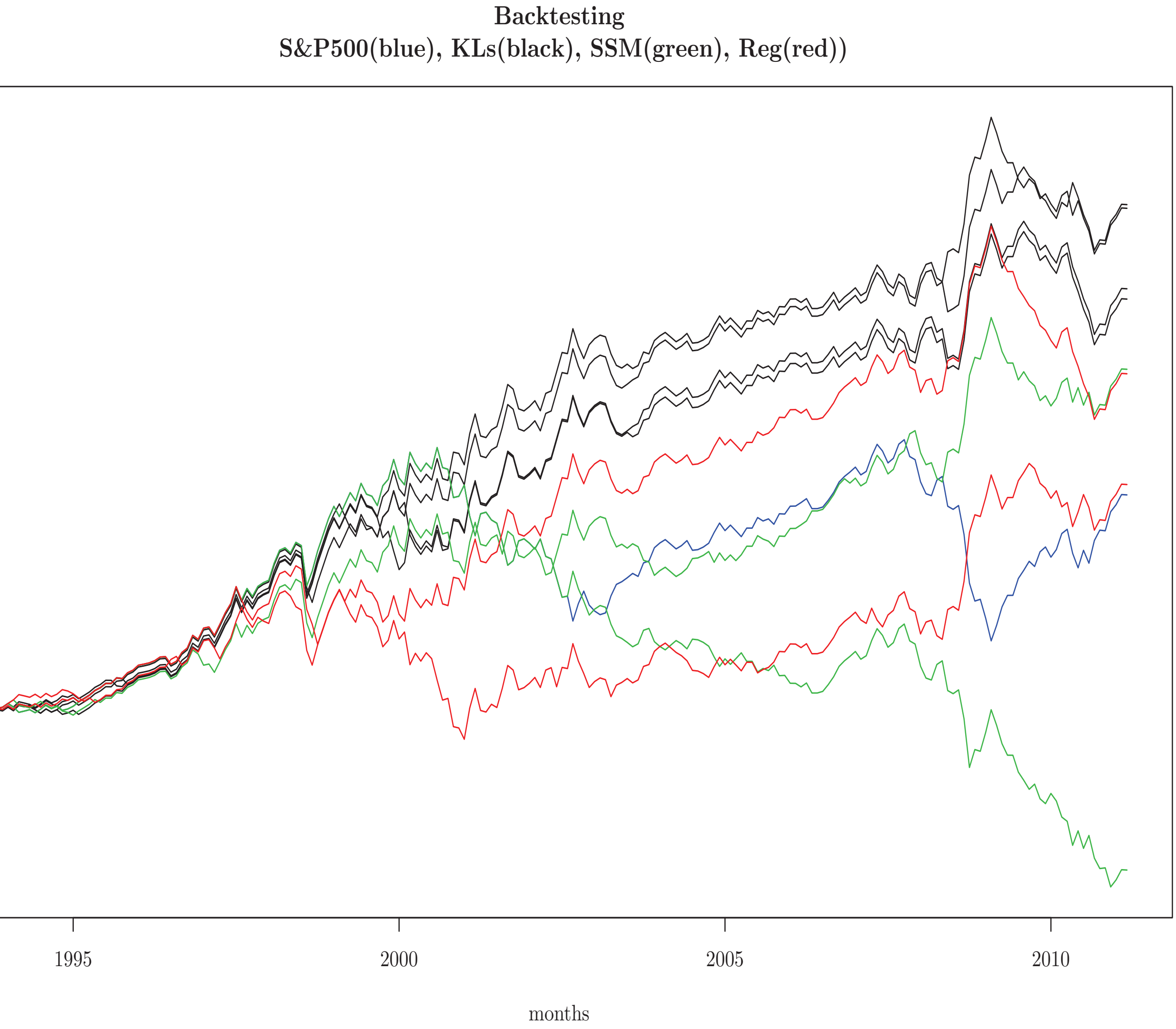


Figure IV.4: $y_t|(\mathrm{sgn}(\tilde{y}_t)), \Sigma = 12$

n o,	**Kalman LagLasso,** $\Sigma = 3$	**Kalman LagLasso,** $\Sigma = 6$	**Kalman LagLasso,** $\Sigma = 12$	**Level Model**	**Local Trend Model**	**Lasso Model**	**Regression Model**
	60%	59%	60%	52.4%	57.6%	57.6%	55.7%
	60.7%	60.1%	60.1%	52.8%	59%	58.4%	56.2%
	62.6%	61.1%	61.1%	50.4%	59.5%	58.8%	58.8%
	63.5%	60.6%	61.5%	50%	58.7%	57.7%	57.7%
	67.1%	62.9%	64.3%	47.1%	55.7%	58.6%	58.6%
	69.1%	65.5%	67.3%	45.4%	56.4%	58.2%	54.5%

Table IV.1: **Recognition Rates of SP500 Upward and Downward Movements**


Backtesting
S&P500(blue), KLs(black), SSM(green), Reg(red))
1995
2000
2005
2010
months

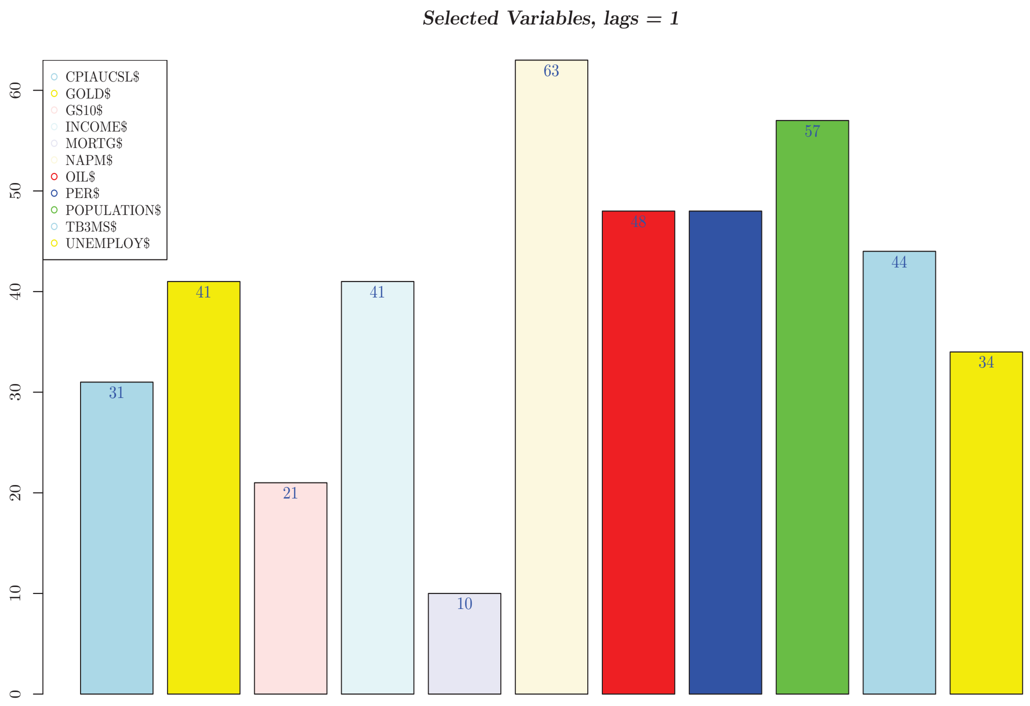


Figure IV.6: **Selected Variables Histogram for KLL, $\Sigma = 1$**

Explanatory Variable	**# Selections**	**# Correct Preds**	**# Wrong Preds**
CPIAUCSL	31	19	12
GOLD	41	20	21
GS10	21	10	11
INCOME	41	20	21
MORTG	10	5	5
NAPM	63	34	29
OIL	48	23	25
PER	48	16	32
POPULATION	57	26	31
TB3MS	44	21	23
UNEMPLOY	34	19	15

Table IV.2: **Correct and Wrong Predictions per Variable for KLL, $\Sigma = 1$**

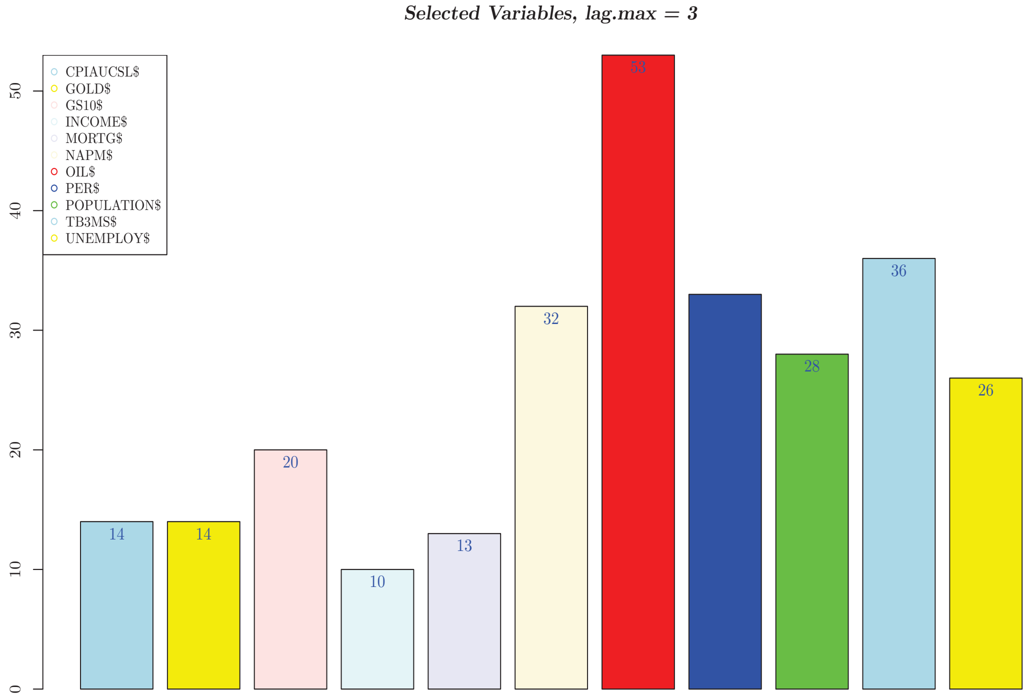


Figure IV.7: **Selected Variables Histogram for KLL, $\Sigma = 3$**

Explanatory Variable	**# Selections**	**# Correct Preds**	**# Wrong Preds**
CPIAUCSL	14	6	8
GOLD	14	4	10
GS10	20	10	10
INCOME	10	4	6
MORTG	13	7	6
NAPM	32	16	16
OIL	53	24	29
PER	33	15	18
POPULATION	28	15	13
TB3MS	36	15	21
UNEMPLOY	26	16	10

Table IV.3: **Correct and Wrong Predictions per Variable for Kalman LagLasso, $\Sigma = 3$**

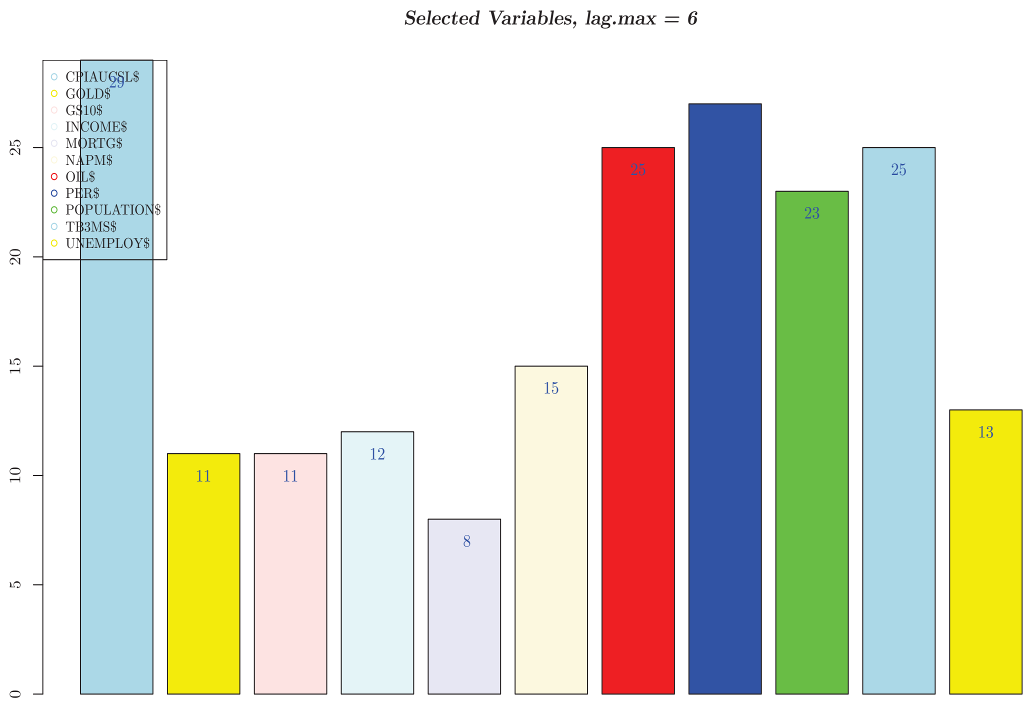


Figure IV.8: **Selected Variables Histogram for KLL, $\Sigma = 6$**

Explanatory Variable	**# Selections**	**# Correct Preds**	**# Wrong Preds**
CPIAUCSL	29	9	20
GOLD	11	3	8
GS10	11	5	6
INCOME	12	6	6
MORTG	8	6	2
NAPM	15	7	8
OIL	25	15	10
PER	27	12	15
POPULATION	23	12	11
TB3MS	25	13	12
UNEMPLOY	13	6	7

Table IV.4: **Correct and Wrong Predictions per Variable for KLL, $\Sigma = 6$**

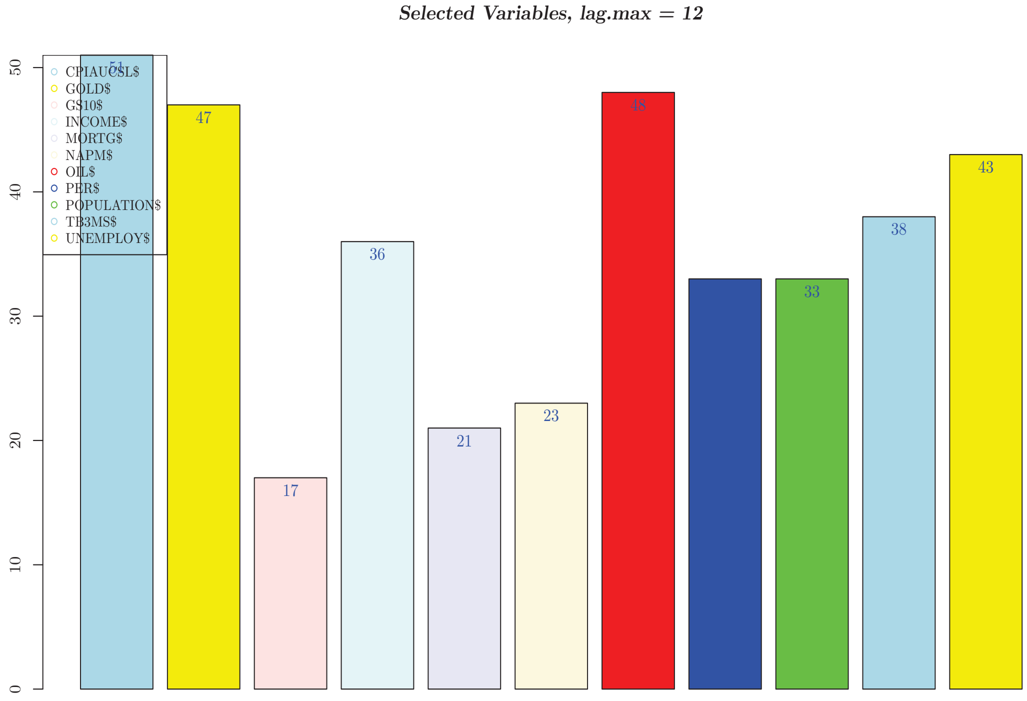


Figure IV.9: **Selected Variables Histogram for KLL, $\Sigma = 12$**

Explanatory Variable	# Selections	# Correct Preds	# Wrong Preds
CPIAUCSL	51	24	27
GOLD	47	23	24
GS10	17	9	8
INCOME	36	20	16
MORTG	21	14	7
NAPM	23	10	13
OIL	48	30	18
PER	33	14	19
POPULATION	33	15	18
TB3MS	38	22	16
UNEMPLOY	43	21	22

Table IV.5: **Correct and Wrong Predictions per Variable for KLL, $\Sigma = 12$**

IV.1.3 Conclusions and Limits of the KLL Approach

Two striking observations can be made concerning the backtesting results.

- Kalman LagLasso numerical models outperform their more classic competitors. Yet, there is a stunning gap between the ambitious goal of predicting real labels and the actual accuracy of the KLL predictions. Indeed, they only yield a recognition rate of 60% in the best case (cf. table IV.1.2).

- The filtering step seems to play a more important role than the variable and lag selection step. First, the tables IV.2, IV.3, IV.4 and IV.5 all point out the poor recognition rate of downward and upward trends per variable. In addition, the results obtained by the Kalman LagLasso numerical models for different maximal lags are very similar. This hardly highlights the influence of lagged variables in the prediction of **SP500** movements. Eventually, the choice of the filtering step is decisive as shown in table IV.1.2 and figure IV.5. These results have the merit of reminding us that good estimating of the current trend of the target variable is of major importance.

The main drawback of the Kalman LagLasso procedure is that it does not provide a method for automatically choosing the most appropriate filter for each variable. This is also an important limit in terms of interpretation functionalities, as we are unable to quantitatively compare the respective relevance of variable filtering and of variable selection. The switch from the regression setup to the binary classification setup is therefore strongly motivated. It first allows us to lower the prediction level of precision, as we aim at predicting binary labels and not real ones. Besides, many learning procedures specific to the binary classification setup benefit from advanced interpretation functionalities, such as a precise analysis of the importance of features. In the next sections, we work in the binary classification setup. We use the UniCart procedure for turning economic and financial variables into interpretable features. This leads to the computation of new databases comprising up-to-date information, even if these variables are not sampled at the same frequency rate. Besides, UniCart features are computed at different scales, which may correspond to different filtering levels. Trend predictive numerical models, trained using the Random Forests (RF) and the Relabeled Nearest Neighbors (RNN) procedures, are compared. Particularly, we show that the RNN procedure is not submitted to an overfitting effect observed with the RF procedure.

IV.2 From the UniCart RF to the UniCart RNN approach

This section aims at evaluating the relevance of the UniCart RF and UniCart RNN approaches. In that perspective, these approaches are applied to the prediction of the trends of the following group of key financial target variables (cf. subsection II.1.1).

1. Stocks: the daily S&P 500 index expressed in U.S. Dollars, denoted **SP500**, and the daily Euro Stoxx 50 Index expressed in Euros, denoted **SX5E**.

2. Commodities: the daily price of gold (afternoon fixing in London) expressed in U.S. dollars, denoted **GOLD**, and the West Texas Intermediate daily spot oil price expressed in U.S. dollars per barrel, denoted **OIL**.

3. Exchange rates: the daily U.S./Euro foreign exchange rate, denoted **EURUSD**, and the daily U.S./U.K. foreign exchange rate, denoted **DEUKUSD**.

The parameters of the numerical models, trained on the two types of UniCart databases, are specified in the first subsection. The backtesting plans and results are thoroughly presented in the second subsection. For each target variable, recognition rates, ROC curves as well as P&Ls of trading rules are computed over the common time period of five years starting at 31/03/2006 and ending at 31/03/2011. The overfitting effect, which occurs for too large UniCart RF classifiers, is particularly emphasized. It is conversely shown that the UniCart RNN numerical models are by nature not submitted to this effect. In parallel, the interpretation functionalities of the RF and RNN procedures are focused on. The role of the scores in the improvement of trading rules is thus highlighted. Finally, the identification of the most predictive features is illustrated for each target variable. The measures of feature prediction ability, computed via the Gini based variable importance functionality, have motivated the design of the Relabeled Nearest Neighbors procedure.

IV.2.1 UniCart RF and RNN Numerical Models

The two types of UniCart databases are used in the backtesting. The first one is made of single-scale trend based features, while the second one is composed of multiscale trend based returns. We recall the definitions introduced in subsection II.2.2.

- The first type of database $[\{x_t, y_t(h)\}]_{W \leq t \leq n_D - h}$ is composed of features, which are all considered at the finest scale available $s^\star$. The features x_t^i are made of the linear trends $\{T^i_{t,s^\star,k}\}_{1 \leq k \leq \tilde{K}}$ and of corresponding statistical quantities. For $1 \leq k \leq \tilde{K}$, these latter are the duration $Dur^i_{t,s^\star,k}$ of the trend $T^i_{t,s^\star,k}$, the variance $Var^i_{t,s^\star,k}$ of the series over the trend, the skewness $Skew^i_{t,s^\star,k}$, the kurtosis $Kurt^i_{t,s^\star,k}$ as well as the minimal and maximal distances $Min^i_{t,s^\star,k}$ and $Max^i_{t,s^\star,k}$ of the series to the trend. In the following backtesting plans, we set the parameters:

$$\begin{cases} W & = 250 \text{ days} \\ w & = 90 \text{ days} \\ h & = 25 \text{ days.} \end{cases}$$

  Besides, we work with a single scale level, denoted $s^\star$, and with $\tilde{K} = 3$ consecutive trend lags. Note that on graphs IV.24, IV.44, IV.64, IV.84, IV.104, IV.124, features are first denoted by their type, where D stands for Dur, V for Var, S for $Skew$ and K for $Kurt$. The name of the variable is then indicated, followed by the lag.

- The second type of database is made of returns exclusively, considered at the scales $\{1, \ldots, \tilde{S}\}$ and for the most recent lag 1 only. For $1 \leq s \leq \tilde{S}$, these latter are the returns $Ret^i_{t,s,k}$, the ratios $Sh^i_{t,s,k}$ and the ratios $So^i_{t,s,k}$. The values of the parameters W, w, h are the same as those entered for the first type of database. Moreover, we consider $\tilde{S} = 3$ consecutively nested scale levels and the most recent trend lag only, i.e. $\tilde{K} = 1$. Note like previously that on graphs IV.25, IV.45, IV.65, IV.85, IV.105, IV.125, features are first denoted by their type, then by their name, scale and lag. The scale $s = 1$ indicates the broadest scale that we aim at predicting, while the scale $s = 3$ is the finest one.

Each UniCart database is then split into a training set and a test set. The distribution of features and labels is learnt on the training set using either the Random Forests (cf. subsection III.2.2) or the Relabeled Nearest Neighbors (cf. section III.3) procedures. More specifically, the RF procedure is used with $it_{max} = 500$ equally weighted tree classifiers. In the following backtesting experiments, we also focus on the minimal size of the subsets of the training set, which are associated with the leaves of the tree classifiers. We express it as a ratio ρ between the number of objects contained in the largest subset and the number of objects in the training set. Hence a high value for ρ indicates that the tree classifiers are small sized. In addition, the RNN procedure manages an ensemble $\mathcal{T}_\mathcal{R}$ of 20 bagged large sized tree classifiers such that $\rho = 0.001$.

IV.2.2 Backtesting Plans and Results

The advanced interpretation functionalities of the RF and the RNN procedure enable the construction of more sophisticated backtesting plans. The UniCart RF numerical models compute indeed probabilities of realizations, which complete the corresponding predictions. Likewise, the UniCart RNN numerical models produce scores, which enable the ranking of the data according to their propensity of being labeled $+1$. The concept of Receiver Operating Characteristic (ROC) provides a functional criterion for evaluating the ranking performance [GS66, CDV11]. The ROC curve of a given scoring function sc is defined as the plot of the true positive rate against the false positive rate:

$$ROC_{sc} = \left\{ \begin{array}{lll} \mathbb{R} & \longrightarrow & [0,1]^2 \\ t & \longmapsto & \{P(sc(X) > t|Y = -1), P(sc(X) > t|Y = +1)\}. \end{array} \right.$$

In case of UniCart RF numerical models, the probability of realization associated with the positive label is used as the input scoring function for the construction of the ROC curve. In the following, we therefore call score this probability of realization. In this subsection, ROC curves complete the classic recognition rates. They are used to visualize in two dimensions the relevance of the ordering induced by scores, which are computed on test sets. Scores are also used to improve the design of trading rules. We define the buying threshold θ_{buy} and the selling threshold θ_{sell}. A buying order for the considered target variable is placed as soon as the score exceeds the buying threshold θ_{buy}. Likewise, a selling order is placed when the score reaches the selling threshold θ_{sell}. In between, the trader holds its position. Besides, orders can be executed with a user-determined delay, that we call the trading rule sensitivity and denote Sy. Expressed in days, this parameter allows us to observe the effect of delaying the order execution on the final value of the P&L. In our experiments, orders are executed as soon as the score exceeds the buying or selling threshold during the number of days indicated by the trading rule sensitivity. Transactions costs are taken into account and set at 1% of transaction amounts. In addition, the P&Ls gained by the trading rules over the test period are annualized. Last but not least, the feature importance is evaluated using the Gini function. This is a very popular functionality, specific to tree classifiers [BFOS84, HTF01]. The backtesting results, detailed in the next subsections, are organized as follows. For each target variable, the recognition rates, the ROC curves, the annualized P&Ls of the trading rules and the measures of feature relevance are displayed for the two types of database. All these results are reported according to the ratio ρ, as this parameter plays a determining role in the exhibition of overfitting. Besides, we illustrate the production of scores over the test set. The P&L gained by the trading rule corresponding to these scores is also represented for well-chosen parameters θ_{buy} and θ_{sell}.

Backtesting Material for each Target Variable

- Explanatory variables
- Size of the training and test sets(corresponding dates)
- Recognition rates of UniCart RNN numerical models - **DB** 1 and 2
- ROC curves of UniCart RNN numerical models - **DB** 1 and 2
- Annualized P&Ls of UniCart RNN trading rules - **DB** 1 and 2
- Feature importance - **DB** 1 and 2
- Illustrative example of scores and P&Ls

SP500 Backtesting Results

The explanatory variables are those used in the subsection IV.1.2 for the construction of Kalman LagLasso numerical models. The only difference is that the financial variables are this time considered daily sampled, while the economic variables remain monthly sampled - as they naturally are. The UniCart databases are thus built with series available at different frequencies from 01/01/1972 to 31/03/2011, and containing up-to-date daily information. The training set encompasses the period from 01/01/1972 to 31/03/2006, while the test set starts at 31/03/2006 and ends at 31/03/2011. The table IV.6 displays the recognition rates obtained for the UniCart RF and UniCart RNN numerical models trained on the first database. It shows that the UniCart RF numerical models perform better for the largest values of the ratio ρ, i.e. for small sized tree classifiers. This table also illustrates obvious overfitting effects, as larger sized trees achieve extremely good results on the training set and perform poorly on the test set. It is finally shown that the RNN procedure, despite using very large tree classifiers, obtains results comparable to the best RF classifiers. Note that the RNN recognition rate of the training set, which is actually relatively low, is very close to the RNN recognition rate of the test set. This emphasizes the absence of overfitting. In addition, the recognition rates obtained using the second database are reported table IV.7. They show that the more homogeneous features of the second database enable a dramatic reduction in overfitting for all UniCart RF numerical models. Note that The RNN numerical model remains very competitive. Besides, the ROC curves, computed with the scores of the test set, globally confirm the observations made on the recognition rates. The figures IV.12, IV.13, IV.14 stress the relatively good performances of the small sized UniCart RF classifiers on the first database, while the figures IV.15, IV.16, and IV.17 underline the overfitting effect. The comparison with the graph IV.10, which represents the ROC curve computed with the RNN score, proves that the RNN model can handle non-stationary features. In addition, the ROC curves corresponding to the UniCart RF classifiers trained on the second database may help refining our analysis. Indeed, the curves displayed by graphs IV.18 and IV.19 indicate rather satisfying results, while the graphs IV.20, IV.21, IV.22, and IV.17 reveal clear perturbations in the convexity of the curves. This is not the case of the RNN ROC curve - cf. figure IV.11. This curve is also radically different from its counterpart of the first database. It shows that the RNN procedure, while capable of dealing with non-stationarities, achieves much better results in case of features having the same support of distribution. Annualized P&Ls of trading rules computed for the first and second databases are then presented. These rules are parametrized first with specific values of the buying and selling thresholds θ_{buy} and θ_{sell}, which check the relation:

$$\theta_{buy} = 1 - \theta_{sell} = \theta.$$

The last parameter is the trading rule sensitivity, which is expressed in days. The table IV.8 focuses on the first database, while the table IV.9 presents results for the second database. Both tables display results, which are consistent with previous recognition rates and ROC curves. In addition, the table IV.9 shows that the trading rules of UniCart/RF numerical models all yield high P&Ls with one notable exception. Trading rules do not thus perform as good as expected for large sized tree classifiers used with high buying thresholds and low selling thresholds. Furthermore, it is clear that the P&Ls obtained with the RNN numerical model are much more stable. Results concerning the UniCart features importance are also reported. This importance is evaluated with the Gini function during the training of the UniCart RF classifier of parameter $\rho = 0.001$. This latter numerical model, like the RNN one, uses the largest tree classifiers taken into consideration in this backtesting analysis. Among the numerical models we tested, it contains the richest information regarding feature relevance. The graph IV.24, computed on the first database, shows thus reliably that the most important feature is the most recent **SP500** linear trend. The graph IV.25, specific to the second database, confirms this result and stresses the dominant role played by the **SP500** returns of the largest scale. Finally, the graph IV.26 displays the scores computed over the test set for the first database. The scores are drawn in blue and the different colors of the **SP500** series stand for the different RNN regimes. It is completed by the graph IV.27, which represents the P&L gained with the trading rule based on these scores for parameters $\theta_{buy} = \theta_{sell} = 0.5$. The graphs IV.28 and IV.29 illustrate the same kind of results for the second database.

		Trn	**Trn** $-$	**Trn** $+$	**Test**	**Test** $-$	**Test** $+$
RF	$\rho = 0.001$	92,48%	88,90%	94,48%	56,28%	4,28%	96,41%
	$\rho = 0.01$	89,38%	86,96%	90,73%	52,55%	23,42%	75,04%
	$\rho = 0.05$	79,77%	80,10%	79,59%	58,79%	49,07%	66,28%
	$\rho = 0.1$	75,52%	72,94%	76,96%	62,67%	44,42%	76,76%
	$\rho = 0.3$	66,23%	76,34%	60,57%	71,98%	70,26%	73,31%
	$\rho = 0.5$	65,97%	75,86%	60,44%	72,15%	70,07%	73,74%
RNN	$\rho = 0.001$	64,29%	77,15%	57,09%	71,58%	70,26%	72,6%

Table IV.6: **SP500 Recognition Rates, 1^{st} DB**

		Trn	**Trn** $-$	**Trn** $+$	**Test**	**Test** $-$	**Test** $+$
RF	$\rho = 0.001$	94,12%	92,87%	94,84%	75,38%	69,45%	80,15%
	$\rho = 0.01$	90,68%	90,34%	90,88%	72,06%	70%	73,72%
	$\rho = 0.05$	81,33%	77,34%	83,65%	71,01%	71,27%	70,80%
	$\rho = 0.1$	75,85%	79,48%	73,73%	72,15%	84,73%	62,04%
	$\rho = 0.3$	74,28%	70,04%	76,74%	74,33%	74,36%	74,31%
	$\rho = 0.5$	74,16%	68,78%	77,29%	74,25%	73,64%	74,74%
RNN	$\rho = 0.001$	73,34%	69,22%	75,73%	74,25%	74,36%	74,16%

Table IV.7: **SP500 Recognition Rates, 2^{nd} DB**

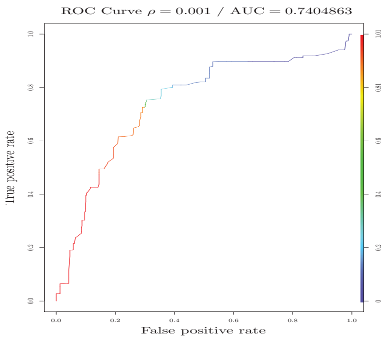


Figure IV.10: **RNN**, $\rho = 0.001$, 1^{st} **DB**

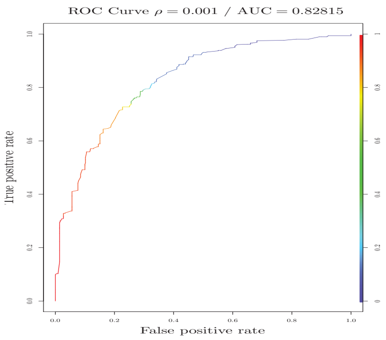


Figure IV.11: **RNN**, $\rho = 0.001$, 2^{nd} **DB**

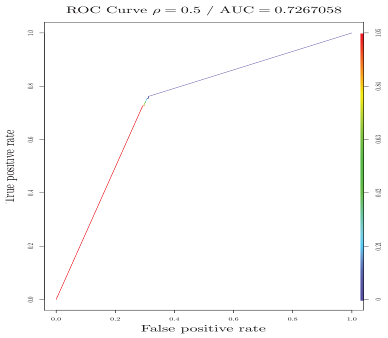


Figure IV.12: **RF**, $\rho = 0.5$, 1^{st} **DB**

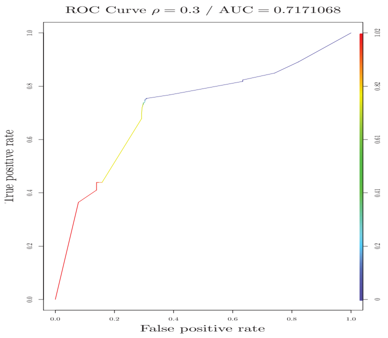


Figure IV.13: **RF**, $\rho = 0.3$, 1^{st} **DB**

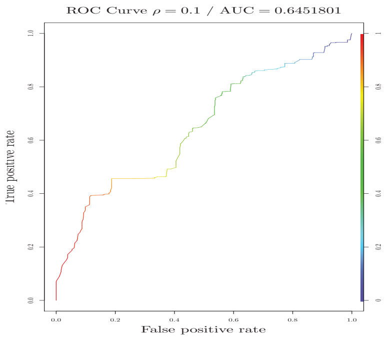


Figure IV.14: **RF**, $\rho = 0.1$, 1^{st} **DB**

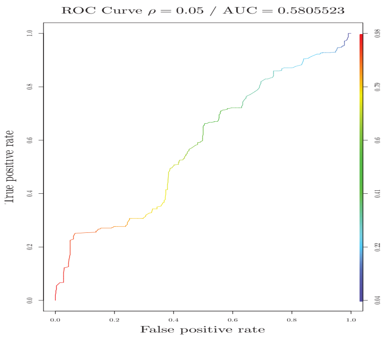


Figure IV.15: **RF**, $\rho = 0.05$, 1^{st} **DB**

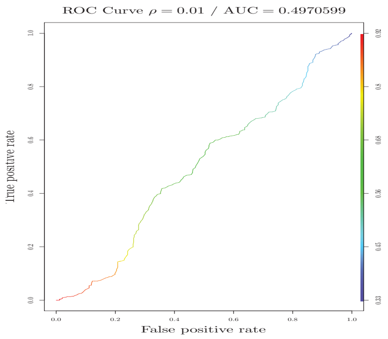


Figure IV.16: **RF**, $\rho = 0.01$, 1^{st} **DB**

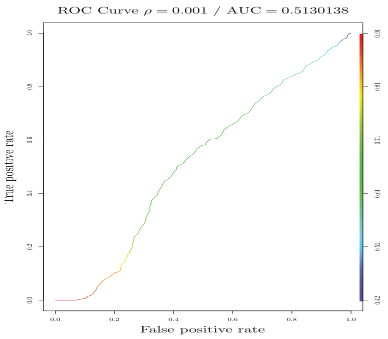


Figure IV.17: **RF**, $\rho = 0.001$, 1^{st} **DB**

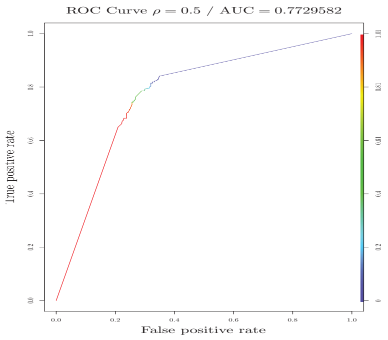


Figure IV.18: **RF**, $\rho = 0.5$, 2^{nd} **DB**

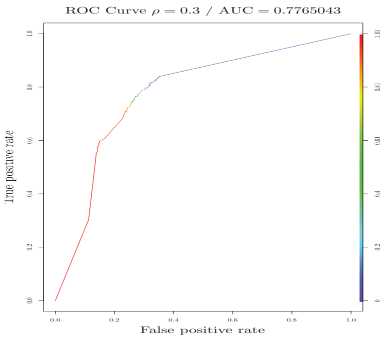


Figure IV.19: **RF**, $\rho = 0.3$, 2^{nd} **DB**

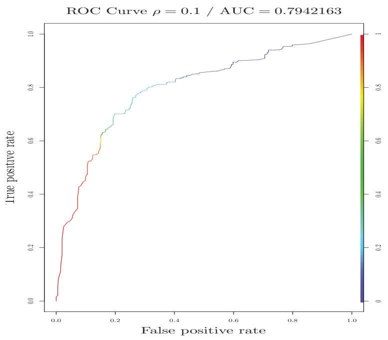


Figure IV.20: **RF**, $\rho = 0.1$, 2^{nd} **DB**

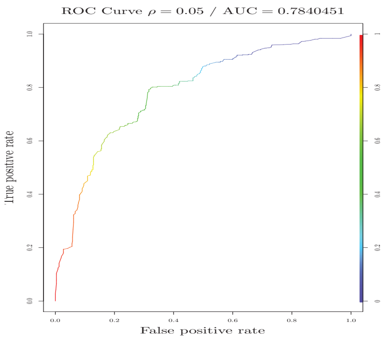


Figure IV.21: **RF**, $\rho = 0.05$, 2^{nd} **DB**

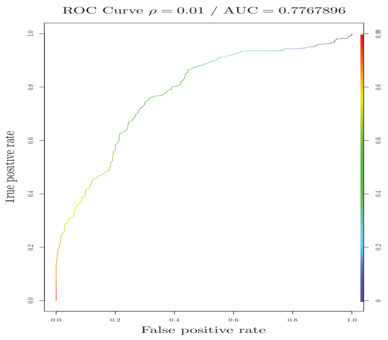


Figure IV.22: **RF**, $\rho = 0.01$, 2^{nd} **DB**

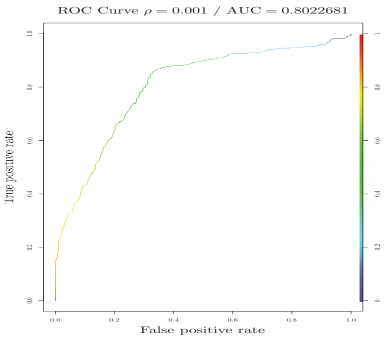


Figure IV.23: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	-1,43%	0,19%	-0,07%	0,12%	7,36%
	$Sy = 1$	-3,33%	0,19%	-0,06%	0,12%	7,36%
	$Sy = 2$	-5,47%	0,19%	-0,01%	-0,17%	7,36%
	$Sy = 3$	-1,60%	0,19%	0,09%	0,12%	6,19%
	$Sy = 4$	-4,92%	0,19%	-0,01%	-0,14%	5,75%
	$Sy = 5$	-2,32%	0,19%	-0,04%	-0,17%	7,36%
RF, $\rho = 0.01$	$Sy = 0$	-9,84%	-4,05%	0,15%	0,11%	-1,28%
	$Sy = 1$	-11,81%	-5,32%	0,15%	0,21%	-1,28%
	$Sy = 2$	-27,58%	-5,97%	0,13%	0,13%	-1,2%
	$Sy = 3$	-17,34%	-4,95%	0,22%	0,12%	-1,2%
	$Sy = 4$	-9,26%	-7,13%	0,22%	0,21%	-1,16%
	$Sy = 5$	-15,59%	-3,61%	0,19%	-0,17%	-1,2%
RF, $\rho = 0.05$	$Sy = 0$	-5,03%	-9,39%	-8,1%	-4,23%	6,68%
	$Sy = 1$	-4,23%	-8,5%	-7,91%	-5,60%	6,30%
	$Sy = 2$	-1,43%	-3,68%	-5,11%	-5,59%	6,46%
	$Sy = 3$	2,30%	-1,9%	-0,33%	-0,99%	5,20%
	$Sy = 4$	2,61%	-3,05%	-4,71%	-5,07%	5,84%
	$Sy = 5$	-6,03%	-6,92%	-7,82%	-7,06%	5,93%
RF, $\rho = 0.1$	$Sy = 0$	-8,95%	-3,99%	-4,46%	2,80%	11,44%
	$Sy = 1$	-7,94%	-3,28%	-2,29%	3,14%	11,27%
	$Sy = 2$	-5,10%	-0,50%	0,49%	3,66%	12,46%
	$Sy = 3$	1,35%	2,20%	3,20%	5,07%	11,76%
	$Sy = 4$	-1,35%	0,27%	0,43%	2,97%	12,53%
	$Sy = 5$	-5,84%	-4,34%	-1,99%	2,41%	13,42%
RF, $\rho = 0.3$	$Sy = 0$	6,27%	6,42%	6,23%	4%	4%
	$Sy = 1$	5,99%	6,97%	7,07%	5,09%	5,09%
	$Sy = 2$	4,76%	5,71%	6,33%	5,37%	5,37%
	$Sy = 3$	6,06%	6,06%	5,85%	4,32%	4,32%
	$Sy = 4$	8,44%	8,44%	8,44%	8,16%	8,16%
	$Sy = 5$	8,81%	9,59%	9,75%	9,04%	9,04%
RF, $\rho = 0.5$	$Sy = 0$	5,67%	5,68%	6,23%	6%	6,07%
	$Sy = 1$	6,60%	7,07%	7,07%	7%	7%
	$Sy = 2$	4,94%	5,89%	6,33%	6,49%	6,49%
	$Sy = 3$	6,44%	5,85%	5,85%	6,37%	6,37%
	$Sy = 4$	8,42%	8,42%	8,44%	8,44%	8,44%
	$Sy = 5$	8,97%	9,75%	9,75%	9,56%	9,56%
RNN, $\rho = 0.001$	$Sy = 0$	7,05%	6,82%	5,89%	5,51%	9,25%
	$Sy = 1$	6,48%	6,41%	6,41%	6,53%	9,63%
	$Sy = 2$	4,82%	4,99%	5,87%	5,67%	9,39%
	$Sy = 3$	5,46%	5,98%	5,98%	6,08%	9,65%
	$Sy = 4$	8,17%	8,17%	8,44%	8,44%	9,26%
	$Sy = 5$	9,59%	9,4%	9,40%	9,45%	10,4%

Table IV.8: **SP500 Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	12,40%	8,59%	9%	8,66%	2,15%
	$Sy = 1$	11,12%	9,15%	10,02%	8,23%	2,22%
	$Sy = 2$	13,01%	11,2%	10,02%	8,14%	1,57%
	$Sy = 3$	13,89%	8,52%	8,45%	7,96%	1,88%
	$Sy = 4$	12,29%	11,11%	9,75%	8,81%	2,05%
	$Sy = 5$	11,35%	12,29%	9,85%	7,82%	1,83%
RF, $\rho = 0.01$	$Sy = 0$	9,15%	6,24%	5,48%	9,08%	5,77%
	$Sy = 1$	8,63%	7,15%	6,47%	8,85%	5,92%
	$Sy = 2$	9,74%	9,14%	7,50%	9%	5,23%
	$Sy = 3$	8,82%	7,19%	4,84%	8,59%	5,41%
	$Sy = 4$	8,76%	7,54%	4,74%	8,60%	6%
	$Sy = 5$	9,21%	11,84%	7,53%	8,30%	5,31%
RF, $\rho = 0.05$	$Sy = 0$	4,53%	6,81%	7,32%	5,70%	1,58%
	$Sy = 1$	4,62%	8,77%	7,67%	5,71%	2,64%
	$Sy = 2$	5,18%	9,04%	8,20%	5,71%	3%
	$Sy = 3$	5,75%	8,26%	7,42%	3,14%	2,11%
	$Sy = 4$	7,52%	9,49%	7,87%	4,10%	4,05%
	$Sy = 5$	7,88%	10,99%	7,31%	4,19%	3,59%
RF, $\rho = 0.1$	$Sy = 0$	10,5%	10,09%	10,63%	10,82%	7,91%
	$Sy = 1$	9,58%	9,47%	10,54%	11,77%	8,54%
	$Sy = 2$	9,1%	8,83%	9,87%	11,01%	10,13%
	$Sy = 3$	8%	8,32%	10,11%	11,82%	8,73%
	$Sy = 4$	8,17%	8,17%	9,52%	10,73%	8,11%
	$Sy = 5$	8,65%	8,66%	9,43%	10,63%	8,43%
RF, $\rho = 0.3$	$Sy = 0$	7,31%	7,45%	8,20%	8,23%	9,10%
	$Sy = 1$	7,18%	7,69%	8,23%	8,40%	8,16%
	$Sy = 2$	5,64%	6,72%	7,15%	7,53%	8,22%
	$Sy = 3$	7,44%	8,21%	9,21%	9,97%	9,28%
	$Sy = 4$	9,15%	9,15%	10,21%	10,21%	8,94%
	$Sy = 5$	9,96%	10,11%	9,13%	9,13%	7,71%
RF, $\rho = 0.5$	$Sy = 0$	6,5%	7,36%	7,37%	8,90%	9,71%
	$Sy = 1$	7,40%	6,79%	7,95%	8,64%	9,08%
	$Sy = 2$	6,45%	6,14%	7,53%	8,93%	8,93%
	$Sy = 3$	7,03%	8,28%	9,97%	10,93%	10,93%
	$Sy = 4$	7,85%	9,88%	10,21%	10,26%	10,26%
	$Sy = 5$	9,91%	9,40%	9,13%	9,02%	9,02%
RNN, $\rho = 0.001$	$Sy = 0$	7,89%	7,12%	7,27%	7,22%	8,39%
	$Sy = 1$	7,18%	6,35%	7,89%	8,49%	8,47%
	$Sy = 2$	5,64%	5,80%	6,48%	7,68%	7,90%
	$Sy = 3$	7,44%	8,99%	9,51%	10,25%	8,26%
	$Sy = 4$	9,15%	10,21%	10,21%	9,70%	8%
	$Sy = 5$	9,96%	9,23%	8,96%	9,13	8,15%

Table IV.9: **SP500 Annualized P&Ls, 2^{nd} DB**

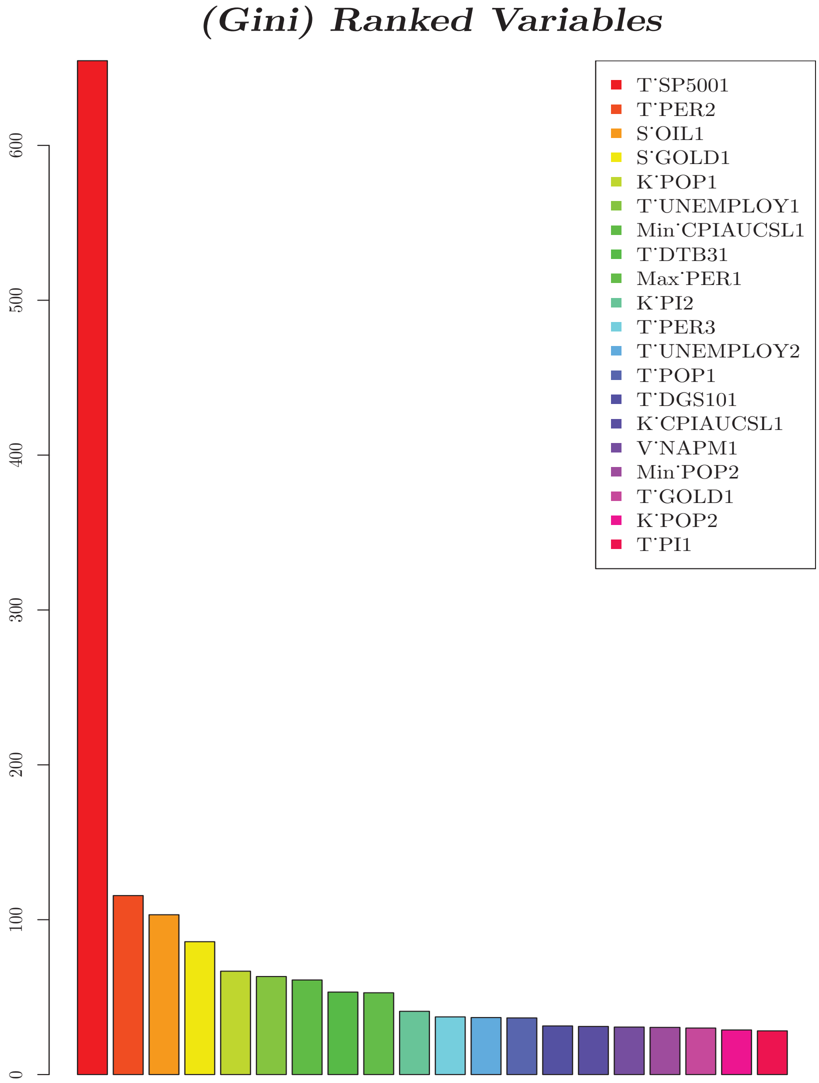


Figure IV.24: **Gini Features Importance, $\rho = 0.001$, 1^{st} DB**

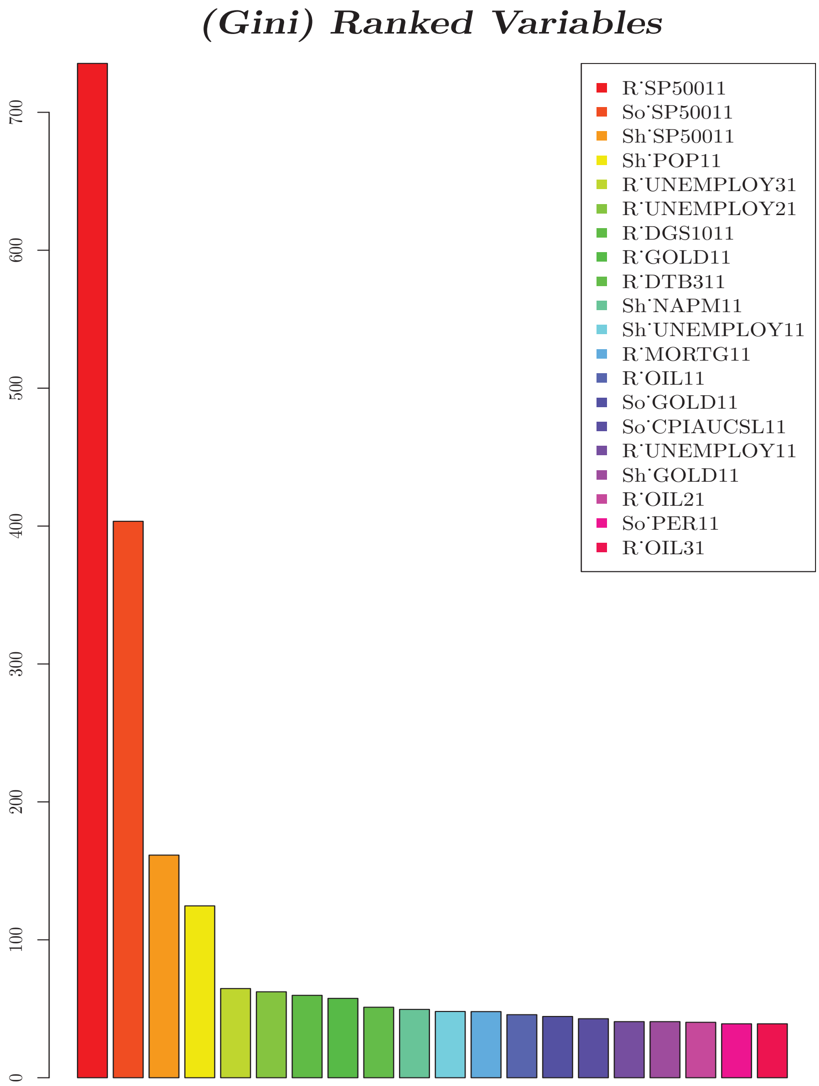


Figure IV.25: **Gini Features Importance, $\rho = 0.001$, 2^{nd} DB**

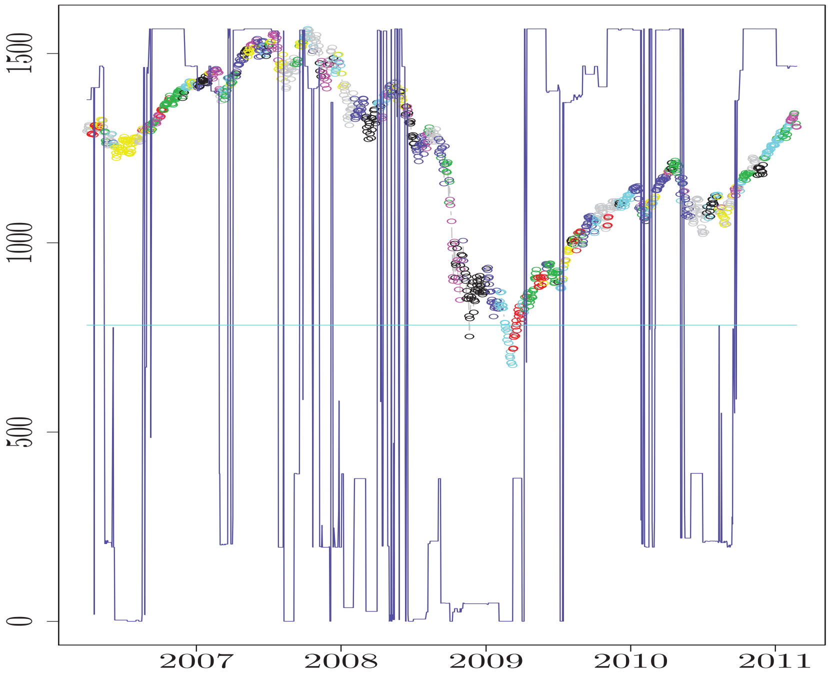


Figure IV.26: **SP500 Scores, 1st DB**

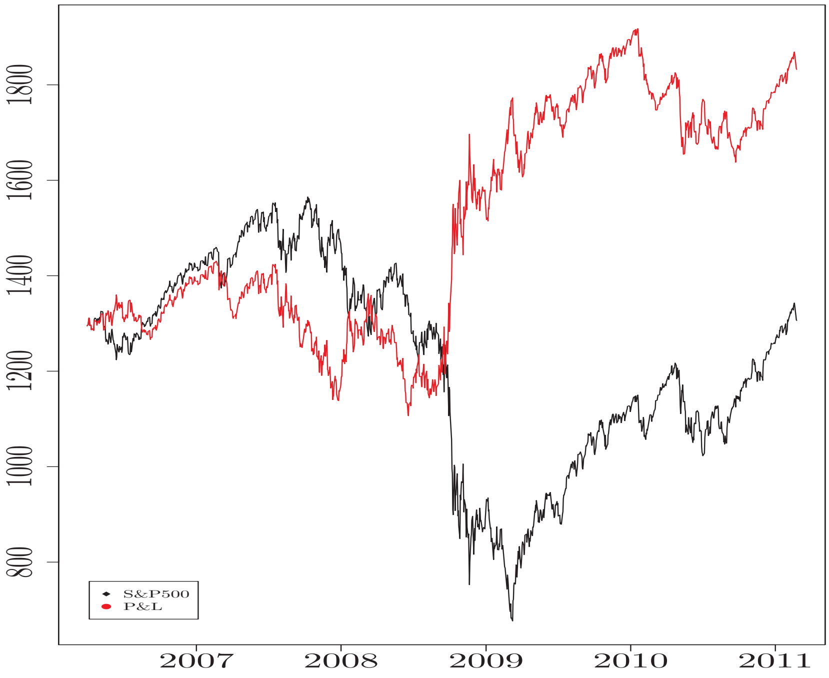


Figure IV.27: **SP500 P&L,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, **1st DB**

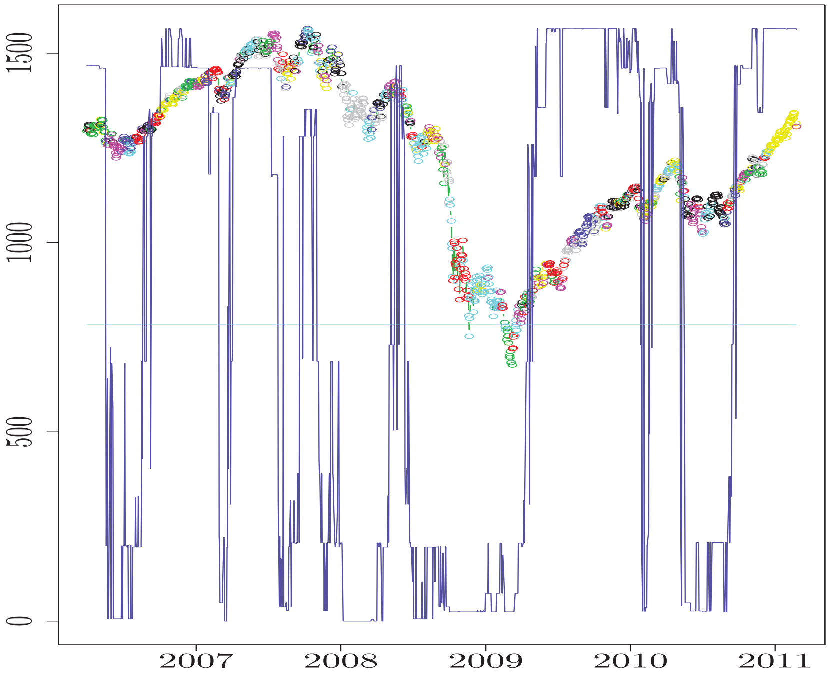


Figure IV.28: **SP500 Scores, 2^{nd} DB**

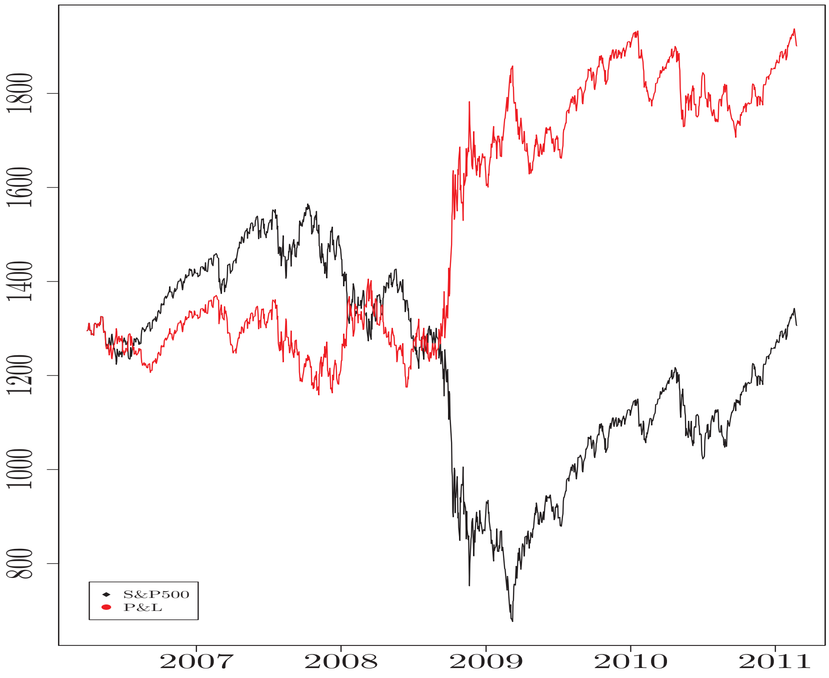


Figure IV.29: **SP500 P&L, $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} DB**

SX5E Backtesting Results

Explanatory Variables	OIL	CAC40	DAX
	GOLD	EURUSD	DEXUKUSD
	DGS10	EURGS10	SP500

Table IV.10: **Data for SX5E**

Training Set	From	05/08/1991	to	31/03/2006
Test Set	From	31/03/2006	to	31/03/2011

Table IV.11: **Size of the Sets**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	91,92%	88,61%	93,90%	51,15%	2,15%	96,63%
	$\rho = 0.01$	91,04%	89,68%	91,86%	47,90%	4,62%	88,06%
	$\rho = 0.05$	82,16%	80,71%	83,03%	47,18%	17,99%	74,27%
	$\rho = 0.1$	77,01%	75,59%	77,87%	42,26%	28,05%	55,44%
	$\rho = 0.3$	70,48%	83,56%	62,64%	59,49%	81,35%	39,20%
	$\rho = 0.5$	71,87%	73,24%	71,04%	68,07%	71,12%	65,24%
RNN	$\rho = 0.001$	70,72%	69,11%	71,68%	68,31%	70,96%	65,85%

Table IV.12: **SX5E Recognition Rates, 1^{st} DB**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	94,77%	92,44%	96,19%	59,73%	48,73%	70,75%
	$\rho = 0.01$	92,85%	91,88%	93,44%	60,29%	53,17%	67,41%
	$\rho = 0.05$	85,20%	84,96%	85,35%	68,71%	68,10%	69,32%
	$\rho = 0.1$	80,59%	78,74%	81,71%	73,23%	78,25%	68,20%
	$\rho = 0.3$	78,56%	81,50%	76,78%	72,68%	81,43%	63,91%
	$\rho = 0.5$	76,08%	82,56%	72,15%	69,50%	83,17%	55,80%
RNN	$\rho = 0.001$	75,79%	80,44%	72,96%	70,06%	83,17%	56,92%

Table IV.13: **SX5E Recognition Rates, 2^{nd} DB**

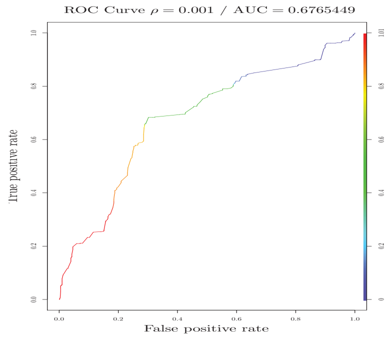


Figure IV.30: **RNN**, $\rho = 0.001$, 1^{st} **DB**

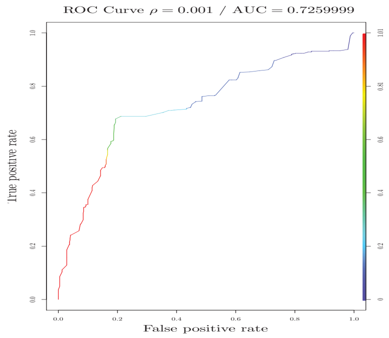


Figure IV.31: **RNN**, $\rho = 0.001$, 2^{nd} **DB**

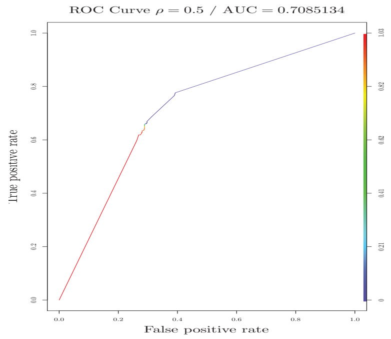


Figure IV.32: **RF**, $\rho = 0.5$, 1^{st} **DB**

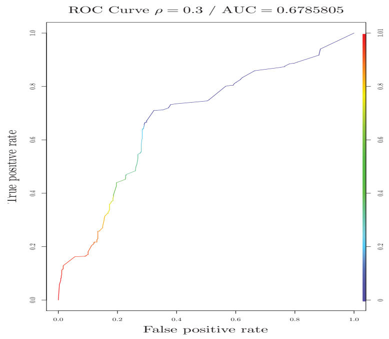


Figure IV.33: **RF**, $\rho = 0.3$, 1^{st} **DB**

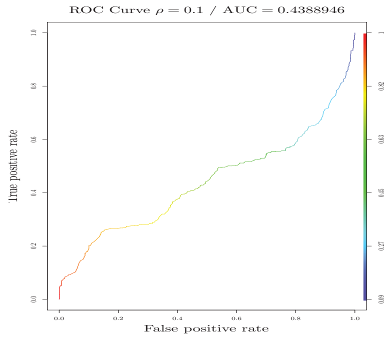


Figure IV.34: **RF**, $\rho = 0.1$, 1^{st} **DB**

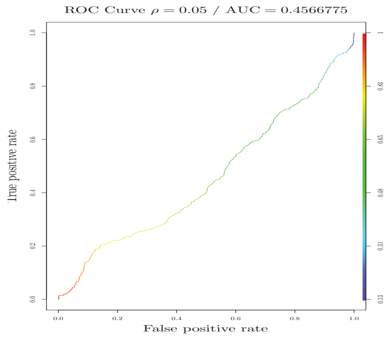


Figure IV.35: **RF**, $\rho = 0.05$, 1^{st} **DB**

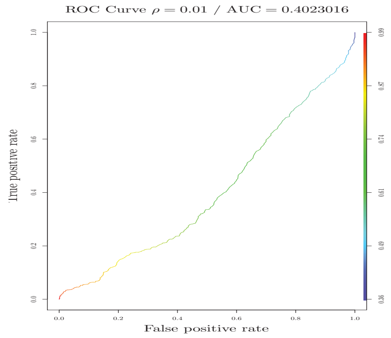


Figure IV.36: **RF**, $\rho = 0.01$, 1^{st} **DB**

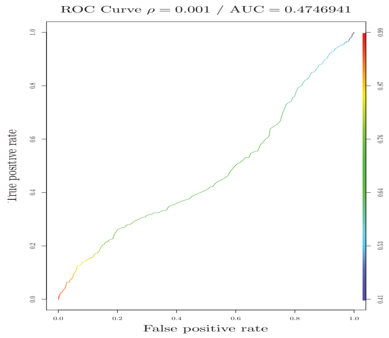


Figure IV.37: **RF**, $\rho = 0.001$, 1^{st} **DB**

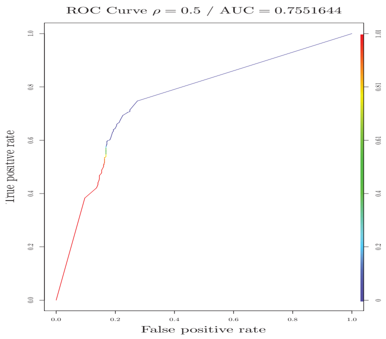


Figure IV.38: **RF**, $\rho = 0.5$, 2^{nd} **DB**

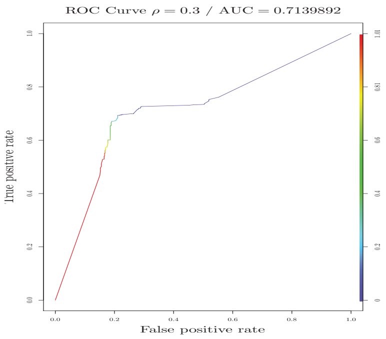


Figure IV.39: **RF**, $\rho = 0.3$, 2^{nd} **DB**

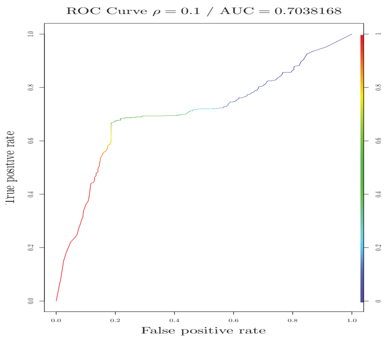


Figure IV.40: **RF**, $\rho = 0.1$, 2^{nd} **DB**

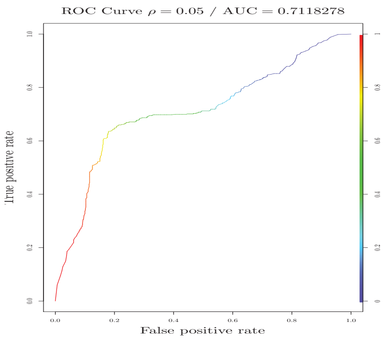


Figure IV.41: **RF**, $\rho = 0.05$, 2^{nd} **DB**

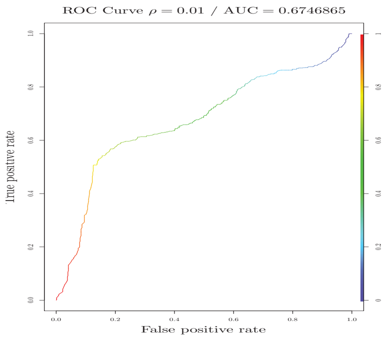


Figure IV.42: **RF**, $\rho = 0.01$, 2^{nd} **DB**

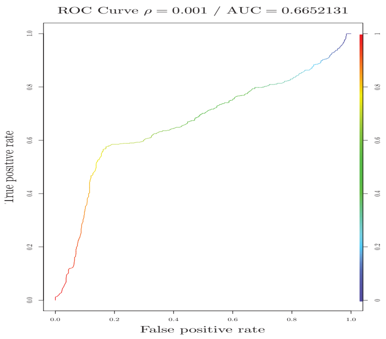


Figure IV.43: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	-9,11%	-5,20%	-5,18%	-5,24%	-4,99%
	$Sy = 1$	-10,62%	-5,20%	-5,18%	-5,24%	-4,99%
	$Sy = 2$	-10,33%	-5,20%	-5,26%	-5,24%	-4,99%
	$Sy = 3$	-10,32%	-5,20%	-5,25%	-5,24%	-4,99%
	$Sy = 4$	-11,91%	-5,20%	-5%	-5,31%	-4,97%
	$Sy = 5$	-12,36%	-5,20%	-5,13%	-5,24%	-4,99%
RF, $\rho = 0.01$	$Sy = 0$	-20,66%	-8,33%	-4,73%	-5,43%	-5,22%
	$Sy = 1$	-19,01%	-5,51%	-4,67%	-5,25%	-5,22%
	$Sy = 2$	-15,27%	-7,12%	-4,73%	-5,31%	-4,94%
	$Sy = 3$	-15,71%	-4,24%	-5,24%	-5,42%	-5%
	$Sy = 4$	-18,68%	-6,27%	-4,67%	-5,31%	-5,32%
	$Sy = 5$	-17,68%	-8,11%	-5,24%	-5,11%	-5,48%
RF, $\rho = 0.05$	$Sy = 0$	-35,98%	-21,41%	-15,01%	-6,43%	-5,43%
	$Sy = 1$	-167,92%	-17,5%	-14,27%	-5,83%	-5,25%
	$Sy = 2$	-179,31%	-10,90%	-14,50%	-6,22%	-5,31%
	$Sy = 3$	-33,29%	-12,02%	-14,97%	-5,70%	-5,42%
	$Sy = 4$	-21,96%	-6,38%	-10,56%	-6,10%	-5,31%
	$Sy = 5$	-24,48%	-15,26%	-12,37%	-6,85%	-5,11%
RF, $\rho = 0.1$	$Sy = 0$	-6,91%	-16,23%	-34,52%	-22,10%	-10,43%
	$Sy = 1$	-8,04%	-11,05%	-26,83%	-21,23%	-10,67%
	$Sy = 2$	-13,20%	-18,47%	-30,30%	-22,88%	-10,60%
	$Sy = 3$	-11,11%	-11,92%	-32,40%	-20,25%	-10,87%
	$Sy = 4$	-9,97%	-11,23%	-24,45%	-21,95%	-9,32%
	$Sy = 5$	-7,16%	-7,92%	-20,90%	-12,51%	-10,08%
RF, $\rho = 0.3$	$Sy = 0$	1,62%	1,64%	0,41%	-2,68%	-6,22%
	$Sy = 1$	0,26%	-0,14%	-2,57%	-5,81%	-7,60%
	$Sy = 2$	2,45%	2,45%	0,07%	-2,20%	-5,32%
	$Sy = 3$	-1,09%	-1,45%	-3,14%	-4,15%	-6,04%
	$Sy = 4$	0,36%	0,36%	-0,64%	-1,70%	-4,76%
	$Sy = 5$	2,38%	2,38%	0,25%	-0,27%	-1,70%
RF, $\rho = 0.5$	$Sy = 0$	4,33%	4,80%	4,84%	4,84%	4,47%
	$Sy = 1$	1,75%	1,38%	1,32%	1,32%	1,3%
	$Sy = 2$	4,30%	4,30%	4,30%	4,30%	4,3%
	$Sy = 3$	2,05%	1,74%	1,44%	1,44%	1,76%
	$Sy = 4$	4,05%	4,05%	3,36%	3,36%	3,36%
	$Sy = 5$	6,10%	6,10%	6,10%	6,10%	6,1%
RNN, $\rho = 0.001$	$Sy = 0$	5,15%	3,44%	4,53%	5,20%	2,5%
	$Sy = 1$	1,32%	2,91%	3,78%	3,78%	5,47%
	$Sy = 2$	4,30%	3,31%	4,65%	5,39%	6,31%
	$Sy = 3$	1,44%	3,26%	3,54%	3,54%	5,08%
	$Sy = 4$	3,36%	4,14%	3,80%	4,22%	7,27%
	$Sy = 5$	6,10%	7,72%	6,80%	6,80%	7,49%

Table IV.14: **SX5E Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	1,08%	1,65%	-0,45%	4,50%	10,21%
	$Sy = 1$	-0,59%	2,61%	-0,89%	3,25%	9,80%
	$Sy = 2$	-1,80%	1,84%	-0,21%	2,51%	9,10%
	$Sy = 3$	2,32%	2,49%	-2,93%	1,17%	9,83%
	$Sy = 4$	-9,48%	-1,71%	-6,18%	0,23%	8,03%
	$Sy = 5$	-1,34%	3,09%	-0,04%	4,24%	8,90%
RF, $\rho = 0.01$	$Sy = 0$	1,99%	-7,68%	3,91%	3,87%	3,13%
	$Sy = 1$	-2,51%	-9,56%	3,22%	2,80%	2,45%
	$Sy = 2$	-3,43%	-11,24%	2,77%	0,01%	3,85%
	$Sy = 3$	0,05%	-6,04%	-2,29%	-0,80%	3,14%
	$Sy = 4$	-10,71%	-15,17%	1,61%	-3,06%	-0,02%
	$Sy = 5$	-1,19%	-3,43%	6,26%	1,84%	4,07%
RF, $\rho = 0.05$	$Sy = 0$	9,17%	8,94%	5,66%	2,95%	3,44%
	$Sy = 1$	4,39%	3,22%	1,53%	1,56%	3,12%
	$Sy = 2$	1,54%	3,11%	0,83%	3,38%	2,63%
	$Sy = 3$	3,81%	-0,56%	-2,51%	-0,30%	1,87%
	$Sy = 4$	1,54%	4,99%	1,46%	3,45%	1,52%
	$Sy = 5$	5,60%	5,74%	3,31%	5,15%	4,18%
RF, $\rho = 0.1$	$Sy = 0$	9,52%	9,24%	7,94%	6,63%	1,89%
	$Sy = 1$	5,65%	5,42%	5,46%	4,50%	3,69%
	$Sy = 2$	4,55%	5,54%	6,21%	4,14%	1,86%
	$Sy = 3$	0,82%	1,61%	2,72%	2,45%	1,04%
	$Sy = 4$	6,75%	8,32%	6,99%	4,41%	0,96%
	$Sy = 5$	6,76%	8,32%	8,60%	6,04%	2,36%
RF, $\rho = 0.3$	$Sy = 0$	5,27%	6,44%	6,09%	5,49%	5,87%
	$Sy = 1$	4,20%	5,91%	6,70%	4,17%	4,74%
	$Sy = 2$	6,08%	6,65%	5,21%	4,92%	4,92%
	$Sy = 3$	4,70%	5,27%	4,38%	1,44%	1,21%
	$Sy = 4$	7,54%	7,13%	6,26%	3,63%	3,69%
	$Sy = 5$	6,43%	6,99%	5,10%	6,08%	6,08%
RF, $\rho = 0.5$	$Sy = 0$	3,92%	3,39%	3,39%	4,24%	4,97%
	$Sy = 1$	5,19%	5,40%	5,40%	4,44%	4,45%
	$Sy = 2$	3,35%	4,19%	4,19%	3,16%	3,16%
	$Sy = 3$	3,30%	4,30%	4,30%	3,36%	3,36%
	$Sy = 4$	4,13%	4,13%	4,13%	4,27%	4,41%
	$Sy = 5$	4,51%	5,52%	5,52%	4,51%	4,51%
RNN, $\rho = 0.001$	$Sy = 0$	3,43%	6,25%	5,07%	7,44%	7,66%
	$Sy = 1$	4,34%	6,20%	4,28%	6,76%	7,01%
	$Sy = 2$	3,35%	5,99%	4,01%	6,06%	6,79%
	$Sy = 3$	3,43%	3,88%	1,39%	4,76%	4,57%
	$Sy = 4$	4,13%	4,70%	1,96%	3,80%	5%
	$Sy = 5$	4,51%	5,66%	5,03%	6,19%	7,08%

Table IV.15: **SX5E Annualized P&Ls, 2^{nd} DB**

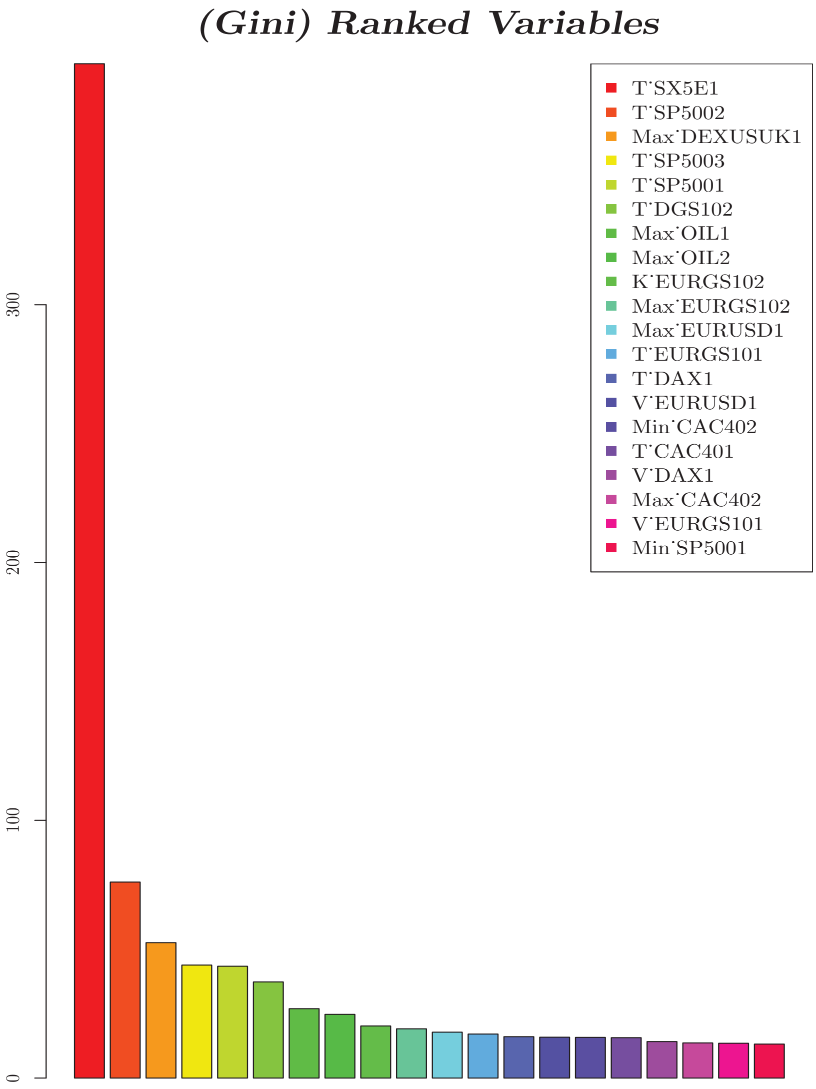


Figure IV.44: **Gini Features Importance, $\rho = 0.001$, 1^{st} DB**

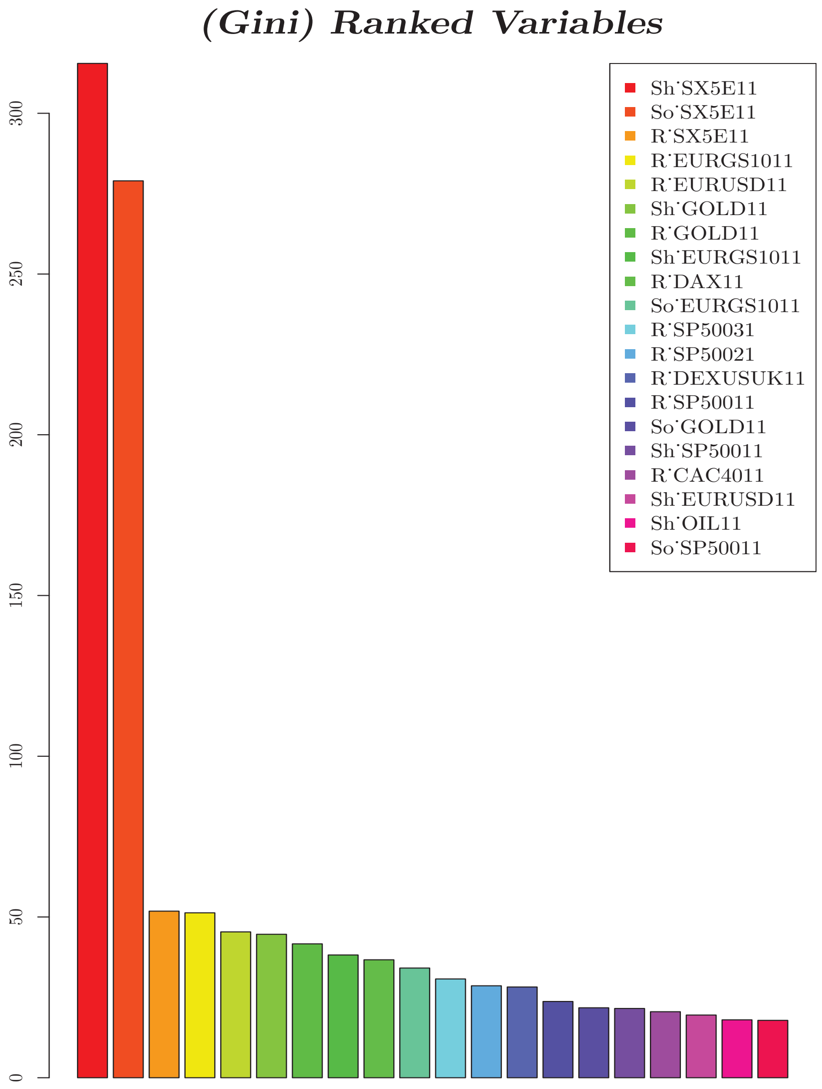


Figure IV.45: **Gini Features Importance, $\rho = 0.001$, 2^{nd} DB**

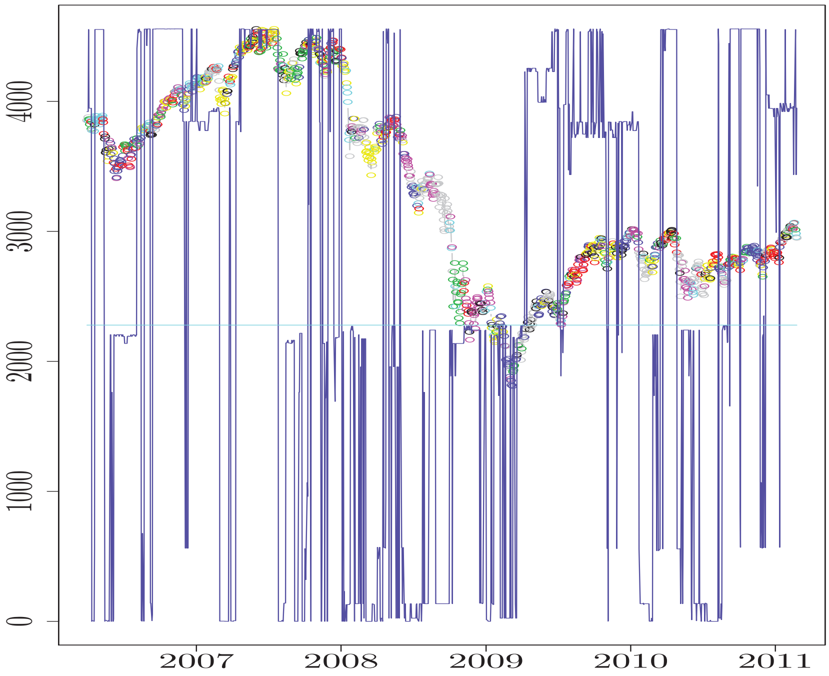


Figure IV.46: **STOXX Scores, 1^{st} DB**

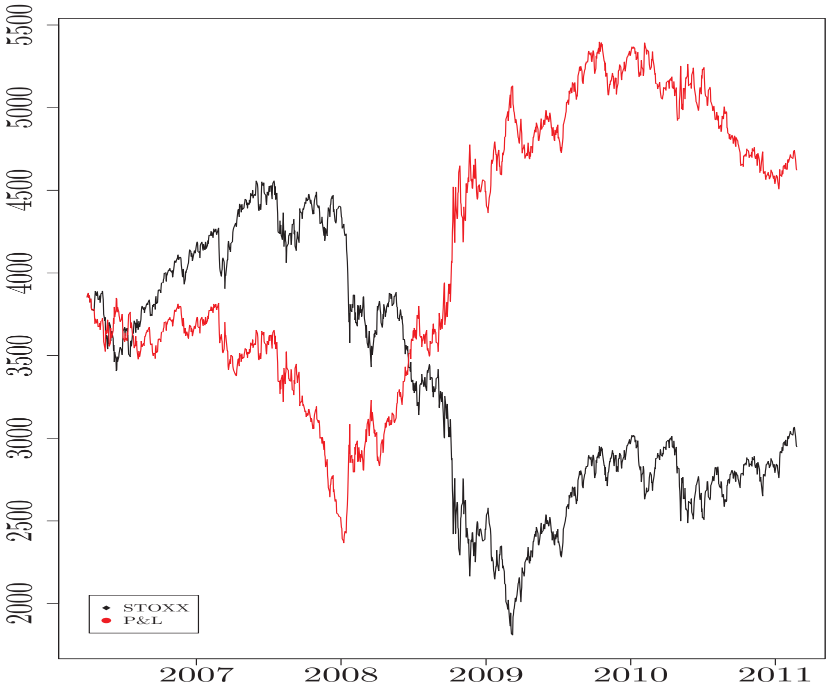


Figure IV.47: **STOXX P&L, $\theta_{buy} = 0.6, \theta_{sell} = 0.4$, 1^{st} DB**

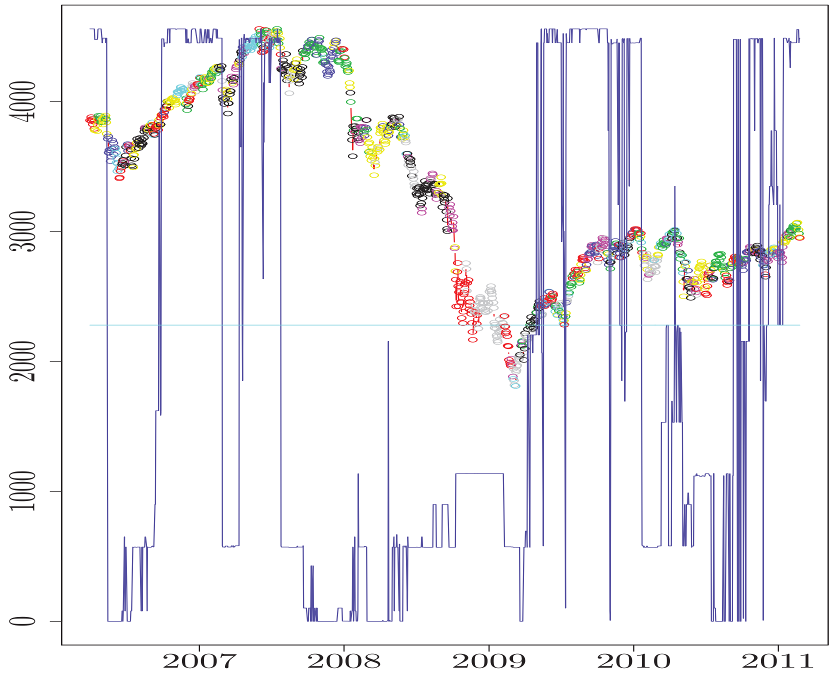


Figure IV.48: **STOXX Scores, 2^{nd} DB**

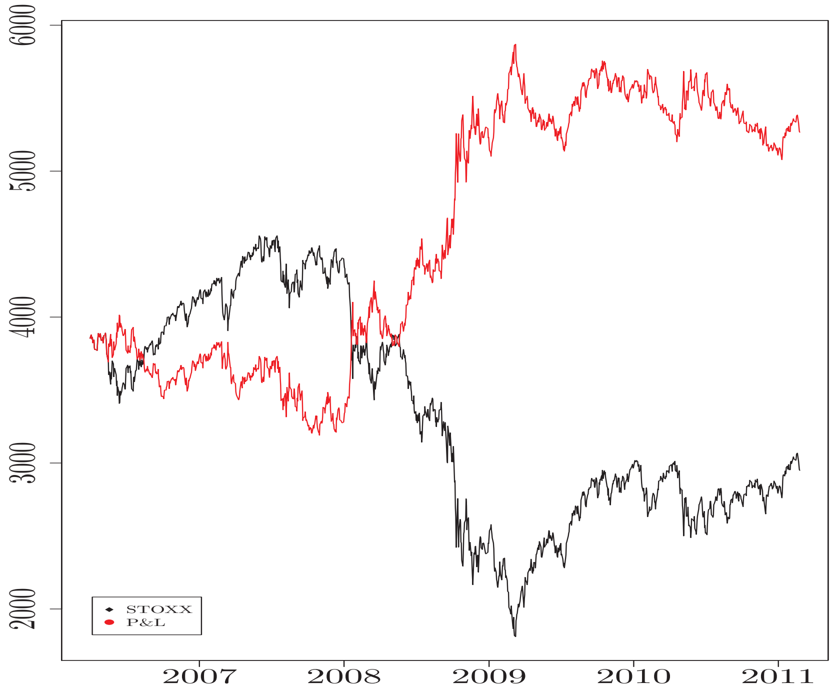


Figure IV.49: **STOXX P&L, $\theta_{buy} = 0.6, \theta_{sell} = 0.4$, 2^{nd} DB**

GOLD Backtesting Results

Explanatory Variables	OIL	DGS10	DTB3	DED3
	SP500	EURGS10	EURUSD	
	DEXCHUS	DEXHKUS	DEXINUS	
	DEXJPUS	DEXSZUS	DEXUSUK	

Table IV.16: **Data for GOLD**

Training Set	From	09/09/1981	to	31/03/2006
Test Set	From	31/03/2006	to	31/03/2011

Table IV.17: **Size of the Sets**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	92,90%	93,59%	92,14%	52,11%	43,66%	55,54%
	$\rho = 0.01$	91,30%	91,56%	91,02%	56,59%	41,13%	62,86%
	$\rho = 0.05$	82,40%	80,58%	84,38%	59,43%	43,10%	66,06%
	$\rho = 0.1$	75,92%	70,67%	81,63%	64,47%	32,39%	77,49%
	$\rho = 0.3$	72,07%	72,76%	71,32%	67,8%	52,11%	74,17%
	$\rho = 0.5$	70,93%	71,64%	70,16%	70%	47,61%	79,09%
RNN	$\rho = 0.001$	69,87%	70,89%	68,76%	70%	47,61%	79,09%

Table IV.18: **GOLD Recognition Rates, 1^{st} DB**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	94,7%	95%	94,36%	73,01%	57,48%	78,97%
	$\rho = 0.01$	91,51%	91,04%	92,05%	75,37%	46,92%	86,28%
	$\rho = 0.05$	83,69%	83,85%	83,50%	77,56%	53,37%	86,84%
	$\rho = 0.1$	78,83%	78,18%	79,58%	76,99%	56,01%	85,04%
	$\rho = 0.3$	75,72%	73,03%	78,81%	77,32%	49,85%	87,85%
	$\rho = 0.5$	75,67%	73,37%	78,32%	77,40%	51,61%	87,29%
RNN	$\rho = 0.001$	74,99%	72,24%	78,14%	76,67%	48,68%	87,40%

Table IV.19: **GOLD Recognition Rates, 2^{nd} DB**

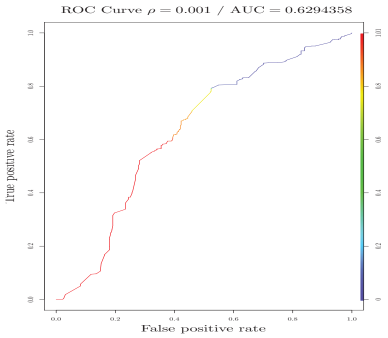


Figure IV.50: **RNN**, $\rho = 0.001$, 1st **DB**

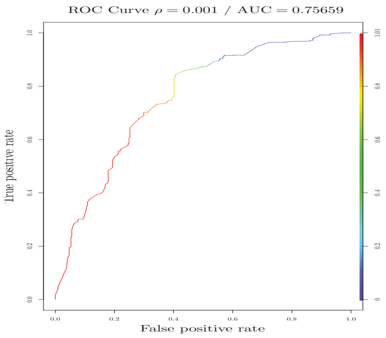


Figure IV.51: **RNN**, $\rho = 0.001$, 2nd **DB**

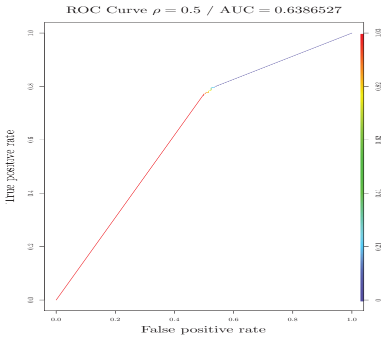


Figure IV.52: **RF**, $\rho = 0.5$, 1st **DB**

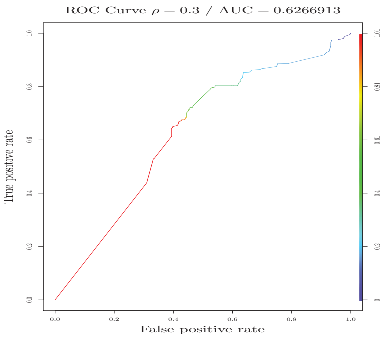


Figure IV.53: **RF**, $\rho = 0.3$, 1st **DB**

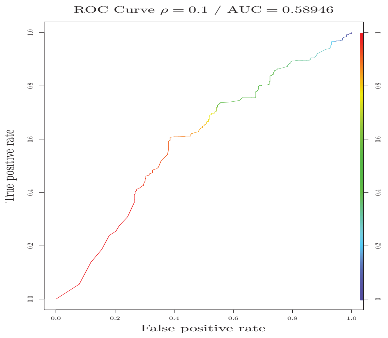


Figure IV.54: **RF**, $\rho = 0.1$, 1st **DB**

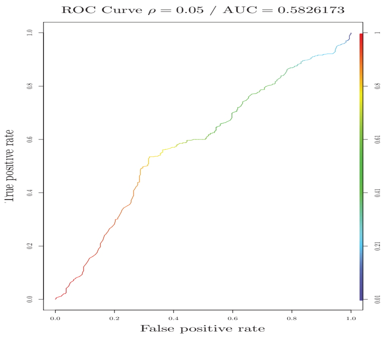


Figure IV.55: **RF**, $\rho = 0.05$, 1st **DB**

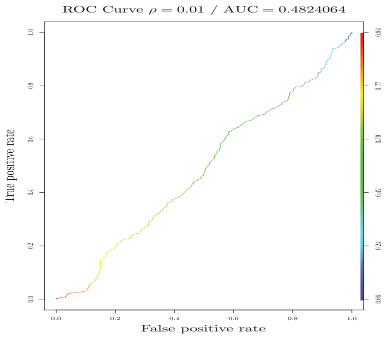


Figure IV.56: **RF**, $\rho = 0.01$, 1st **DB**

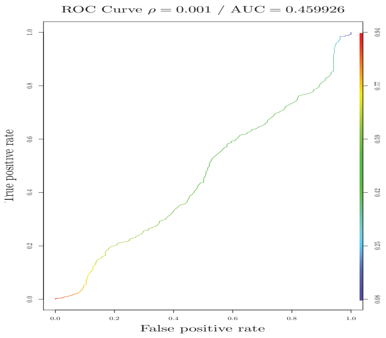


Figure IV.57: **RF**, $\rho = 0.001$, 1st **DB**

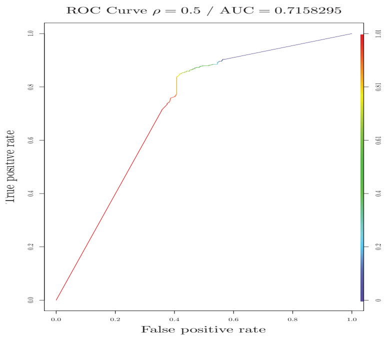


Figure IV.58: **RF**, $\rho = 0.5$, 2^{nd} **DB**

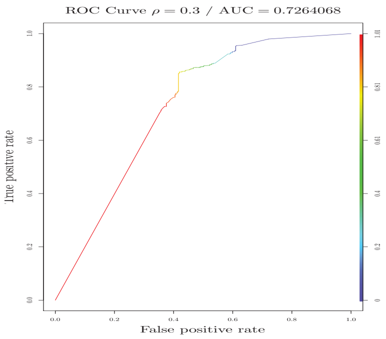


Figure IV.59: **RF**, $\rho = 0.3$, 2^{nd} **DB**

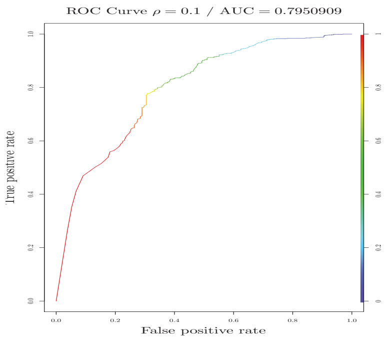


Figure IV.60: **RF**, $\rho = 0.1$, 2^{nd} **DB**

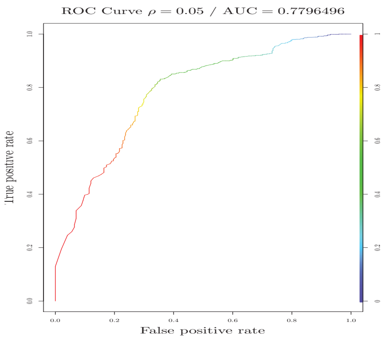


Figure IV.61: **RF**, $\rho = 0.05$, 2^{nd} **DB**

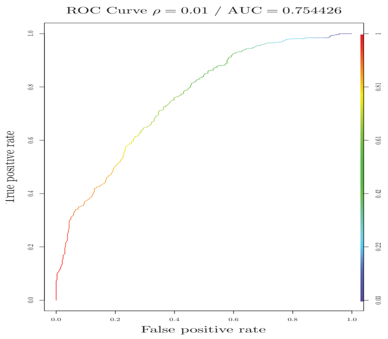


Figure IV.62: **RF**, $\rho = 0.01$, 2^{nd} **DB**

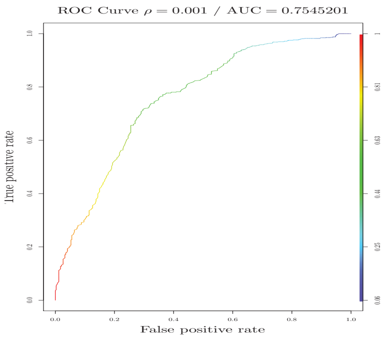


Figure IV.63: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	-6,06%	-1,13%	5,50%	8,3%	-178,63%
	$Sy = 1$	-2,12%	1,41%	6,51%	7,47%	-178,63%
	$Sy = 2$	-7,96%	6,14%	6,54%	8,61%	-178,90%
	$Sy = 3$	2,51%	3,93%	5,48%	7,79%	-178,90%
	$Sy = 4$	-0,79%	3,35%	7,84%	7,08%	-178,63%
	$Sy = 5$	0,59%	12,22%	7,37%	8,77%	-178,90%
RF, $\rho = 0.01$	$Sy = 0$	-3,08%	3,79%	5,08%	7,62%	-177,59%
	$Sy = 1$	4,2%	8,46%	0,67%	7,16%	-175,56%
	$Sy = 2$	6,79%	13,16%	1,94%	7,29%	-175,16%
	$Sy = 3$	6,07%	6,02%	-0,47%	7,48%	-175,56%
	$Sy = 4$	4,67%	3,38%	-1,18%	6,81%	-175,16%
	$Sy = 5$	8,06%	11,40%	0,28%	7,42%	-171,97%
RF, $\rho = 0.05$	$Sy = 0$	7,54%	18,82%	16,14%	16,31%	11,21%
	$Sy = 1$	4,49%	17,90%	13,69%	15,56%	10,54%
	$Sy = 2$	7,60%	14,92%	13,36%	17,46%	11,3%
	$Sy = 3$	8,01%	18,34%	13,48%	13,52%	10,81%
	$Sy = 4$	8,08%	7,95%	10,25%	13,47%	10,1%
	$Sy = 5$	11,03%	15,10%	12,27%	14,65%	11,41%
RF, $\rho = 0.1$	$Sy = 0$	17,03%	19,07%	22,55%	21,15%	19,32%
	$Sy = 1$	13,55%	15,07%	21,58%	20,02%	19,28%
	$Sy = 2$	12,92%	17,97%	21,81%	21,40%	19,11%
	$Sy = 3$	14,95%	15,14%	21,97%	19,05%	18,63%
	$Sy = 4$	5,73%	10,76%	16,80%	15,74%	17,72%
	$Sy = 5$	17%	20,2%	22,43%	18,42%	17,95%
RF, $\rho = 0.3$	$Sy = 0$	18,78%	16,91%	17,57%	15,86%	19,46%
	$Sy = 1$	13,17%	14,56%	14,81%	13,48%	19,20%
	$Sy = 2$	8,95%	13,49%	13,49%	14,62%	20,04%
	$Sy = 3$	15,79%	16,28%	16,28%	13,75%	19,18%
	$Sy = 4$	3,86%	2,39%	3,98%	5,85%	17,29%
	$Sy = 5$	13,56%	12,83%	12,83%	13,51%	18,30%
RF, $\rho = 0.5$	$Sy = 0$	12,42%	12,42%	12,42%	13,11%	11,65%
	$Sy = 1$	9,29%	9,29%	9,29%	9,21%	9,37%
	$Sy = 2$	10,72%	10,72%	10,72%	11,14%	9,04%
	$Sy = 3$	12,72%	12,72%	12,72%	13,34%	12,22%
	$Sy = 4$	-0,49%	-0,49%	-0,49%	-1,96%	-1,62%
	$Sy = 5$	9,92%	9,92%	9,92%	9,71%	7,96%
RNN, $\rho = 0.001$	$Sy = 0$	12,42%	12,42%	12,42%	15,86%	10,23%
	$Sy = 1$	9,29%	9,29%	9,29%	14,20%	10,68%
	$Sy = 2$	10,72%	10,72%	10,72%	13,49%	7,88%
	$Sy = 3$	12,72%	12,72%	12,72%	15,97%	14,05%
	$Sy = 4$	-0,49%	-0,49%	-0,49%	2,03%	-0,64%
	$Sy = 5$	9,92%	9,92%	9,92%	12,83%	6,18%

Table IV.20: **GOLD Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	9,83%	14,73%	13,99%	15,69%	11,28%
	$Sy = 1$	6,22%	11,15%	13,15%	15,66%	11,51%
	$Sy = 2$	2,04%	10,29%	12,87%	13,66%	11,45%
	$Sy = 3$	2,03%	5,58%	14,09%	16,28%	11,51%
	$Sy = 4$	-7,94%	13,82%	14,21%	13,66%	10,10%
	$Sy = 5$	-15,09%	1,90%	13,03%	13,52%	10,33%
RF, $\rho = 0.01$	$Sy = 0$	18,72%	14,68%	18,2%	12,32%	12,96%
	$Sy = 1$	17,92%	10,54%	16,78%	11,29%	13,39%
	$Sy = 2$	14,84%	12,39%	17,54%	9,70%	12,78%
	$Sy = 3$	18,64%	12,90%	17,28%	12,14%	13,68%
	$Sy = 4$	11,78%	8,39%	15,88%	12,77%	12,91%
	$Sy = 5$	5,33%	7,62%	17,65%	11,58%	10,58%
RF, $\rho = 0.05$	$Sy = 0$	20,13%	19,87%	18,29%	6,58%	15,18%
	$Sy = 1$	17,52%	14,30%	15,07%	7,75%	14,88%
	$Sy = 2$	13,56%	18,82%	15,15%	7,15%	15,16%
	$Sy = 3$	16,92%	19,71%	16,06%	5,80%	15,95%
	$Sy = 4$	12,11%	19,45%	13,12%	10,29%	16,16%
	$Sy = 5$	10,91%	13,05%	12,84%	12,57%	14,88%
RF, $\rho = 0.1$	$Sy = 0$	16,36%	9,82%	3,41%	13,95%	13,32%
	$Sy = 1$	7,25%	8,61%	3,67%	13,56%	12,57%
	$Sy = 2$	8,79%	6,93%	5,20%	12,58%	12,46%
	$Sy = 3$	9,25%	12,64%	8,10%	14,42%	12,75%
	$Sy = 4$	12%	16,36%	11,16%	17,65%	14,02%
	$Sy = 5$	5,59%	9,95%	6,91%	15,19%	10,17%
RF, $\rho = 0.3$	$Sy = 0$	12,34%	11,55%	11,40%	11,93%	12,41%
	$Sy = 1$	8,49%	10,09%	9,70%	11,97%	13,14%
	$Sy = 2$	8,25%	10,36%	11,92%	11,36%	12,53%
	$Sy = 3$	9,17%	11,22%	11,24%	11,68%	11,68%
	$Sy = 4$	15,29%	16,58%	14,53%	13,46%	13,18%
	$Sy = 5$	7,30%	9,90%	9,90%	10,23%	12,01%
RF, $\rho = 0.5$	$Sy = 0$	9,77%	10,65%	11,39%	11,62%	11,68%
	$Sy = 1$	4,54%	8,67%	11,03%	11,30%	12,26%
	$Sy = 2$	7,56%	9,25%	11,90%	11,27%	11,79%
	$Sy = 3$	5,70%	10%	10,79%	11,1O%	11,19%
	$Sy = 4$	12,81%	12,74%	14,25%	13,02%	13,28%
	$Sy = 5$	7,13%	9,90%	9,86%	9,66%	11,51%
RNN, $\rho = 0.001$	$Sy = 0$	12%	10,35%	8,71%	8,63%	4,05%
	$Sy = 1$	2,44%	8,67%	7,96%	7,25%	1,84%
	$Sy = 2$	8,96%	9,25%	6,74%	5,07%	2,21%
	$Sy = 3$	8,52%	10%	9,06%	7,55%	4,29%
	$Sy = 4$	13,25%	12,74%	11,81%	11,52%	10,95%
	$Sy = 5$	6,38%	9,9%	4,44%	0,33%	0,84%

Table IV.21: **GOLD Annualized P&Ls, 2^{nd} DB**

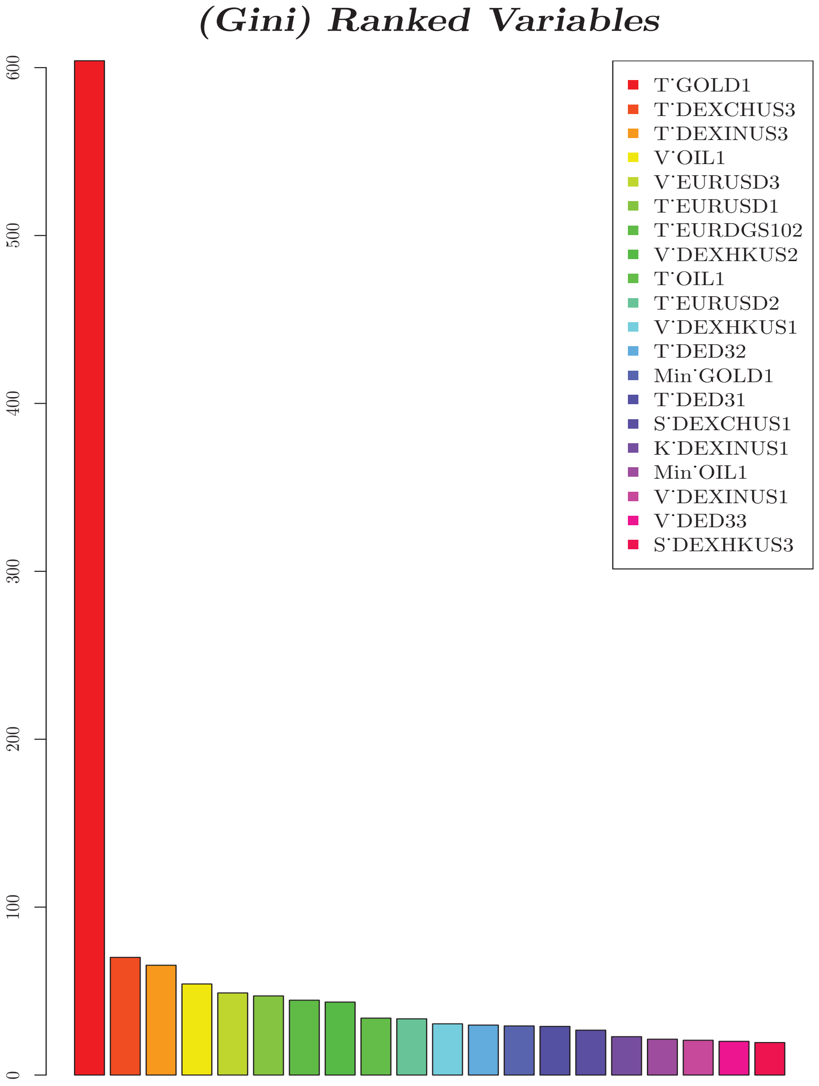


Figure IV.64: **Gini Features Importance, $\rho = 0.001$, 1^{st} DB**

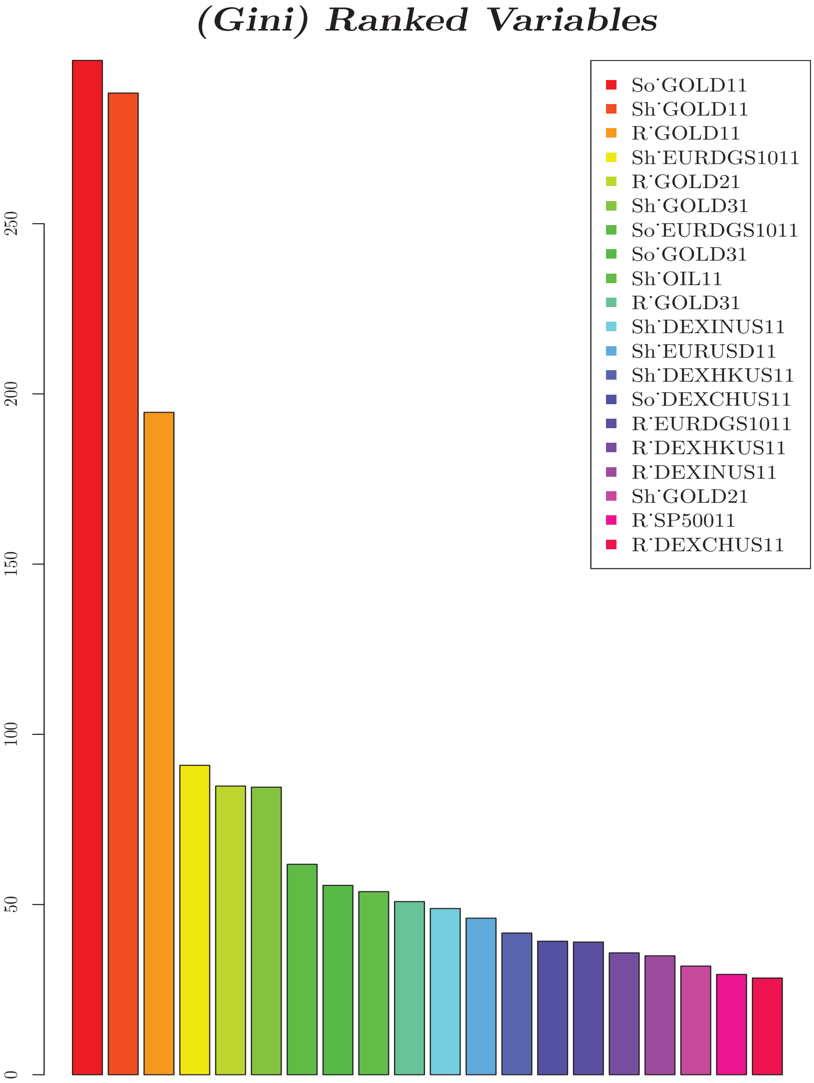


Figure IV.65: **Gini Features Importance, $\rho = 0.001$, 2^{nd} DB**

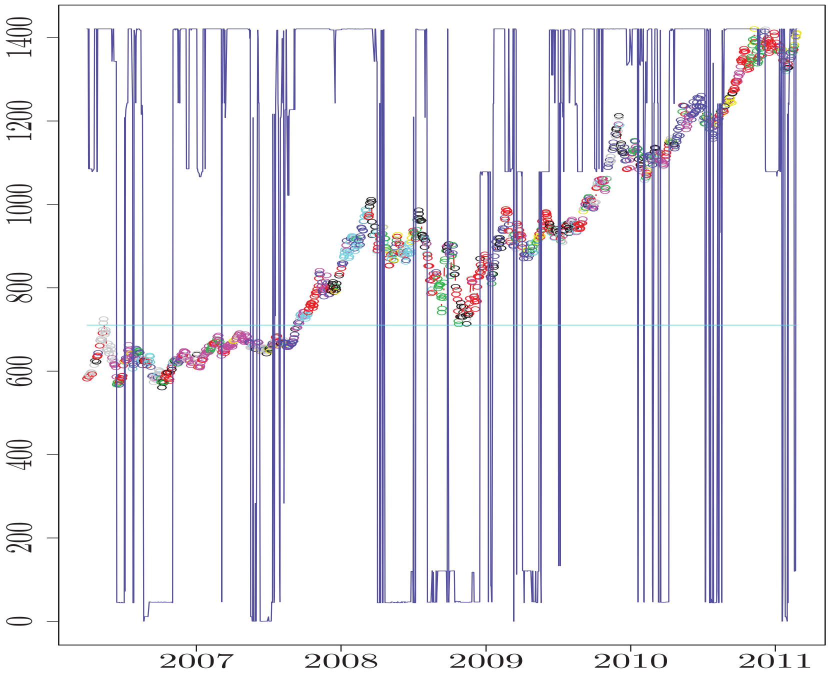


Figure IV.66: **GOLD Scores, 1^{st} DB**

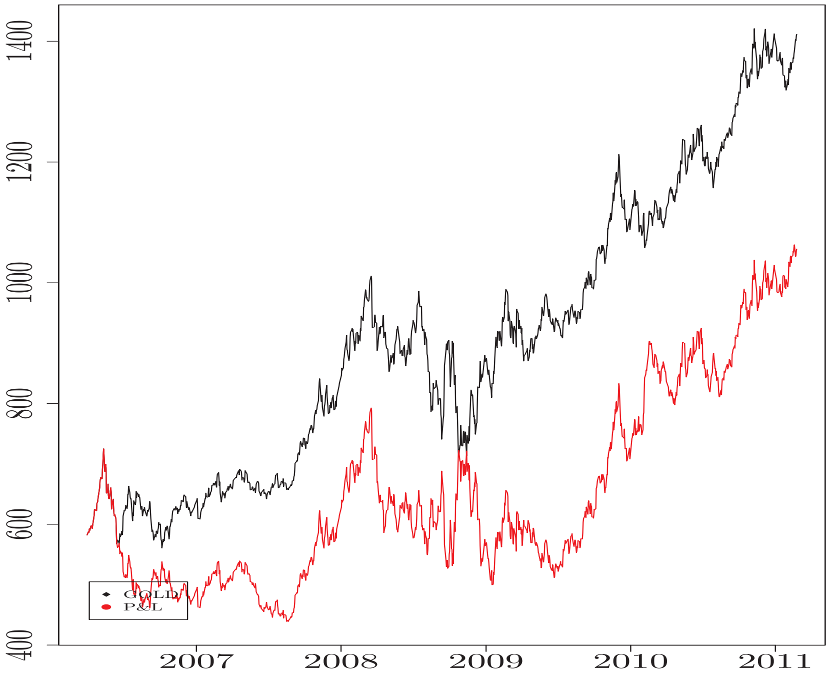


Figure IV.67: **GOLD P&L,** $\theta_{buy} = 0.6, \theta_{sell} = 0.4$, **$1^{st}$ DB**

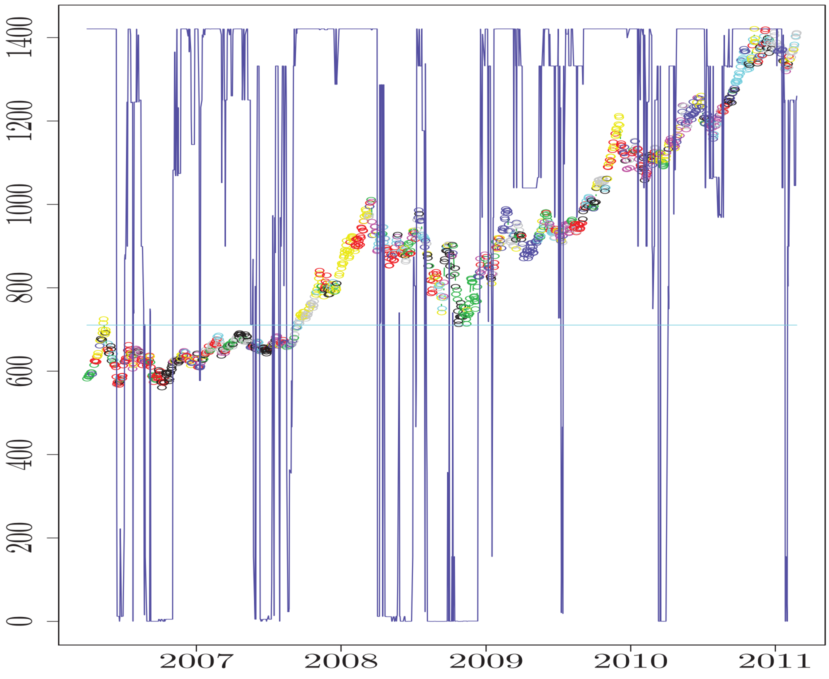


Figure IV.68: **GOLD Scores, 2^{nd} DB**

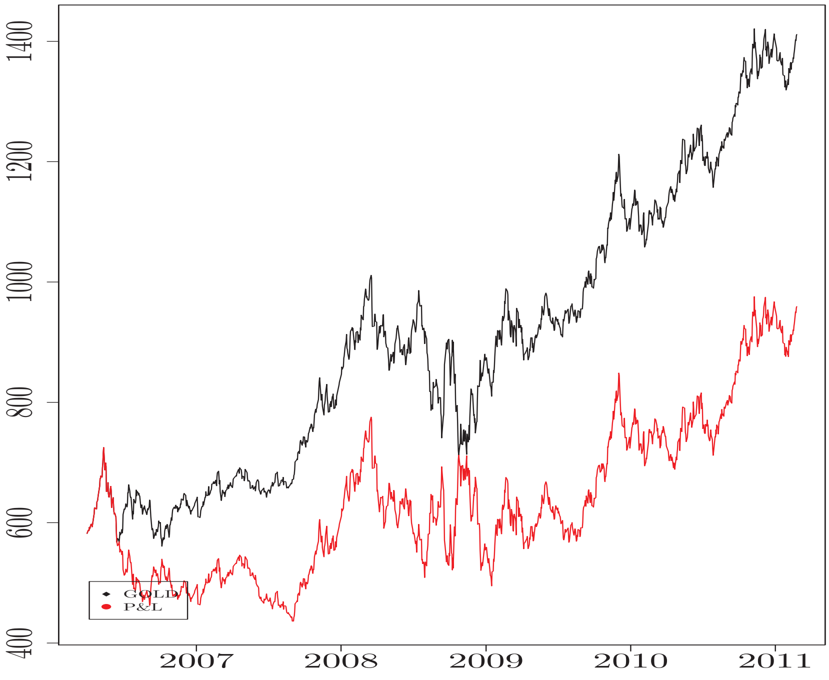


Figure IV.69: **GOLD P&L, $\theta_{buy} = 0.6, \theta_{sell} = 0.4$, 2^{nd} DB**

OIL Backtesting Results

Explanatory Variables	SP500	MORTG	NAPM	TCU
	DEXJPUS	EURUSD	GOLD	
	DGS10	DTB3	OILSTOCKS	

Table IV.22: **Data for OIL**

Training Set	From	12/09/1990	to	31/03/2006
Test Set	From	31/03/2006	to	31/03/2011

Table IV.23: **Size of the Sets**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	91,67%	91,44%	91,85%	65,32%	17,83%	87,01%
	$\rho = 0.01$	90,95%	91,84%	90,22%	65,24%	23,26%	84,42%
	$\rho = 0.05$	83,9%	83,11%	84,55%	62,40%	47,80%	69,07%
	$\rho = 0.1$	76,88%	73,69%	79,49%	64,83%	46,77%	73,08%
	$\rho = 0.3$	69,34%	72,03%	67,13%	74,47%	64,86%	78,87%
	$\rho = 0.5$	67,88%	59,08%	75,09%	78,04%	63,57%	84,65%
RNN	$\rho = 0.001$	67,57%	59,47%	74,20%	78,12%	63,57%	84,77%

Table IV.24: **OIL Recognition Rates, 1^{st} DB**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	95,55%	95,68%	95,43%	66,45%	69,65%	64,90%
	$\rho = 0.01$	93,83%	93,45%	94,18%	64,99%	66,17%	64,42%
	$\rho = 0.05$	85,29%	83,91%	86,57%	66,29%	72,14%	63,46%
	$\rho = 0.1$	80,25%	76,51%	83,74%	76,09%	64,93%	81,49%
	$\rho = 0.3$	75,36%	77,09%	73,74%	79,01%	72,39%	82,21%
	$\rho = 0.5$	73,89%	73,31%	74,44%	81,69%	72,39%	86,18%
RNN	$\rho = 0.001$	74,23%	65,48%	82,40%	83,06%	67,41%	90,62%

Table IV.25: **OIL Recognition Rates, 2^{nd} DB**

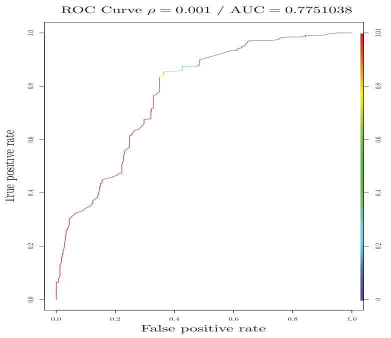


Figure IV.70: **RNN**, $\rho = 0.001$, 1^{st} **DB**

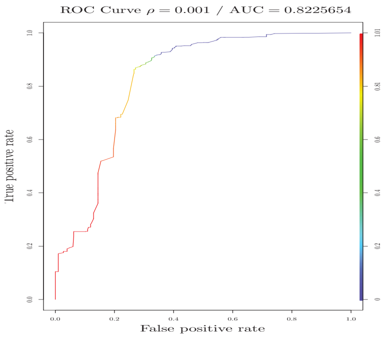


Figure IV.71: **RNN**, $\rho = 0.001$, 2^{nd} **DB**

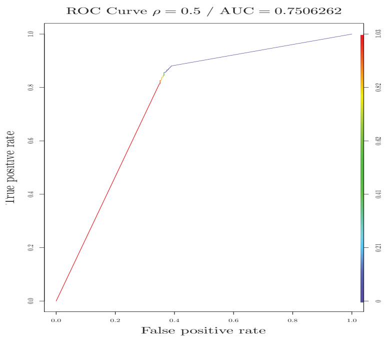


Figure IV.72: **RF**, $\rho = 0.5$, 1^{st} **DB**

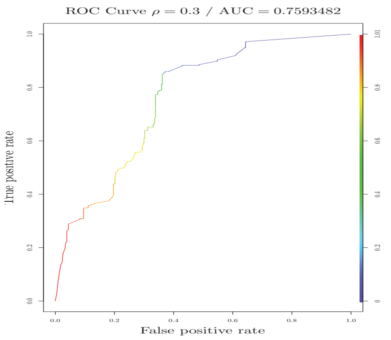


Figure IV.73: **RF**, $\rho = 0.3$, 1^{st} **DB**

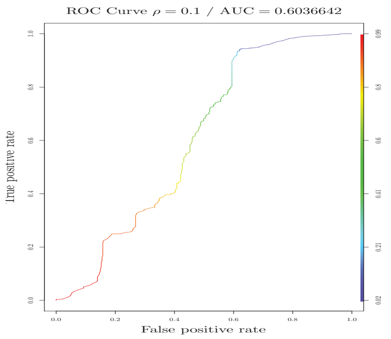


Figure IV.74: **RF**, $\rho = 0.1$, 1^{st} **DB**

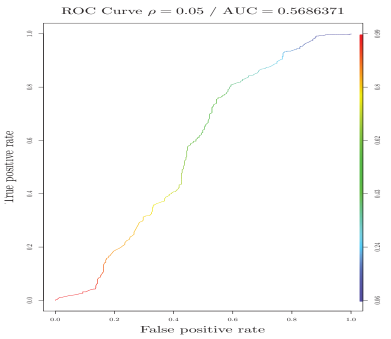


Figure IV.75: **RF**, $\rho = 0.05$, 1^{st} **DB**

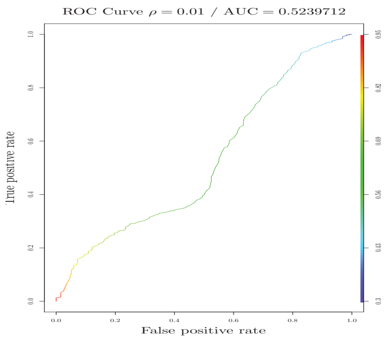


Figure IV.76: **RF**, $\rho = 0.01$, 1^{st} **DB**

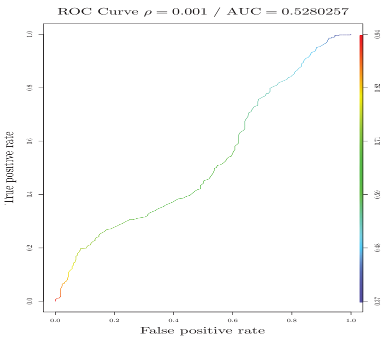


Figure IV.77: **RF**, $\rho = 0.001$, 1^{st} **DB**

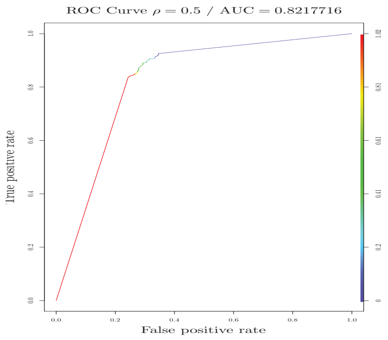


Figure IV.78: **RF**, $\rho = 0.5$, 2^{nd} **DB**

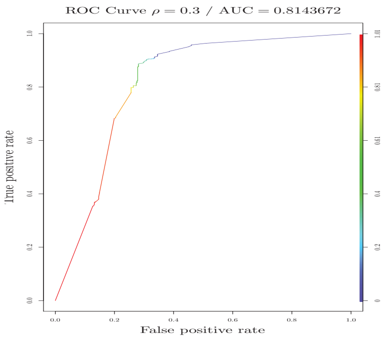


Figure IV.79: **RF**, $\rho = 0.3$, 2^{nd} **DB**

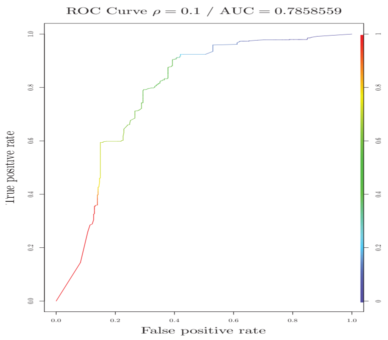


Figure IV.80: **RF**, $\rho = 0.1$, 2^{nd} **DB**

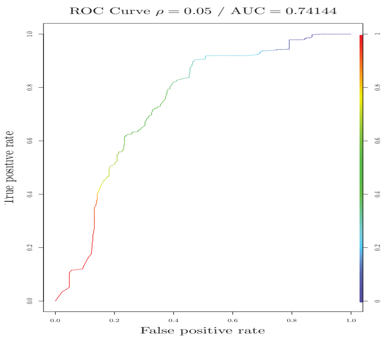


Figure IV.81: **RF**, $\rho = 0.05$, 2^{nd} **DB**

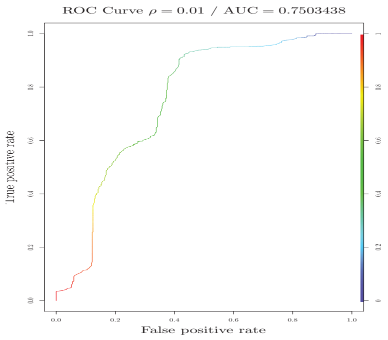


Figure IV.82: **RF**, $\rho = 0.01$, 2^{nd} **DB**

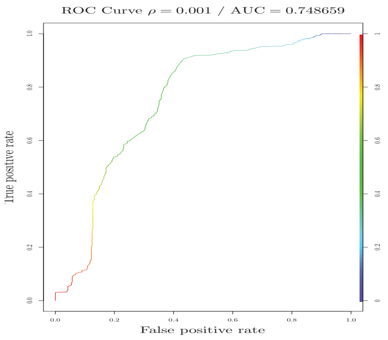


Figure IV.83: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	13,80%	7,72%	6,99%	6,17%	8,33%
	$Sy = 1$	5,02%	8,11%	6,81%	6,17%	8,44%
	$Sy = 2$	6%	8,45%	6,99%	7,06%	8,44%
	$Sy = 3$	-2,57%	8%	6,39%	6,17%	8,44%
	$Sy = 4$	5,55%	9,13%	6,81%	7,06%	8,44%
	$Sy = 5$	11,58%	7,88%	6,39%	7,06%	7,96%
RF, $\rho = 0.01$	$Sy = 0$	12,37%	4,54%	6,99%	6,17%	8,7%
	$Sy = 1$	17,60%	3,31%	6,81%	6,17%	8,69%
	$Sy = 2$	12,26%	1,49%	6,99%	7,06%	8,70%
	$Sy = 3$	4,49%	1,57%	6,39%	6,17%	8,69%
	$Sy = 4$	1,75%	0,82%	6,81%	7,06%	8,70%
	$Sy = 5$	11,32%	2,60%	6,39%	7,06%	8,33%
RF, $\rho = 0.05$	$Sy = 0$	0,50%	18,91%	12,33%	-14,64%	7,97%
	$Sy = 1$	7,71%	21,63%	14,70%	-5,33%	7,23%
	$Sy = 2$	7,10%	23,71%	15,24%	-5,89%	8,60%
	$Sy = 3$	4,32%	13,18%	12,84%	-7,61%	6,99%
	$Sy = 4$	9,01%	14,52%	11,08%	-19,40%	8,5%
	$Sy = 5$	20,96%	22,18%	15,29%	-27,79%	7,63%
RF, $\rho = 0.1$	$Sy = 0$	8,01%	18,57%	14,26%	12,62%	21,10%
	$Sy = 1$	15,16%	20,71%	15,07%	16,12%	20,58%
	$Sy = 2$	9,56%	22,06%	17,69%	17,47%	21,59%
	$Sy = 3$	4,87%	11,48%	6,80%	15,69%	20,79%
	$Sy = 4$	10,41%	17,68%	14,75%	11,90%	21,46%
	$Sy = 5$	19,38%	22,54%	15,09%	14,14%	19,68%
RF, $\rho = 0.3$	$Sy = 0$	18,35%	19,69%	11,67%	4,17%	1,74%
	$Sy = 1$	19,14%	21,20%	11,48%	5,43%	1,67%
	$Sy = 2$	18,69%	19,37%	8,70%	5,69%	1,5%
	$Sy = 3$	7,45%	9,33%	3,77%	-9,80%	-0,23%
	$Sy = 4$	19,49%	19,78%	9,29%	4,74%	4,13%
	$Sy = 5$	16,22%	20,28%	9,91%	10%	7,61%
RF, $\rho = 0.5$	$Sy = 0$	21,70%	21,86%	22,22%	22,29%	23,18%
	$Sy = 1$	22,28%	22,61%	22,90%	22,90%	22,97%
	$Sy = 2$	22,53%	22,27%	22,78%	22,78%	22,67%
	$Sy = 3$	17,05%	17,60%	17,69%	17,69%	17,11%
	$Sy = 4$	22,03%	21,64%	20,91%	20,91%	22,98%
	$Sy = 5$	21,03%	21,03%	20,35%	20,35%	18,87%
RNN, $\rho = 0.001$	$Sy = 0$	22,78%	22,66%	22,82%	22,52%	23,63%
	$Sy = 1$	22,28%	22,09%	22,43%	23,13%	24,67%
	$Sy = 2$	23,41%	23,21%	22,96%	21,41%	21,41%
	$Sy = 3$	17,05%	17,05%	17,60%	18,15%	19,41%
	$Sy = 4$	22,03%	22,03%	21,64%	20%	20,7%
	$Sy = 5$	21,03%	21,94%	21,94%	21,58%	21,58%

Table IV.26: **OIL Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	9,40%	12,74%	14,16%	-6,05%	6,40%
	$Sy = 1$	14,53%	14,43%	15,58%	-8,26%	6,82%
	$Sy = 2$	12,29%	17,33%	17,07%	-7,22%	6,82%
	$Sy = 3$	15,23%	13,72%	14,50%	-7,58%	6,58%
	$Sy = 4$	14,87%	14,49%	17,20%	-4,16%	7,55%
	$Sy = 5$	22,39%	20,37%	19,55%	-5,98%	6,08%
RF, $\rho = 0.01$	$Sy = 0$	13,81%	18,88%	16,63%	20,41%	18,67%
	$Sy = 1$	9,56%	19,42%	17,72%	20,36%	17,05%
	$Sy = 2$	12,77%	20,67%	19,91%	21,60%	16,91%
	$Sy = 3$	3,85%	15,81%	12,12%	15,65%	18,44%
	$Sy = 4$	16,07%	16,68%	18,92%	21,32%	20,12%
	$Sy = 5$	18,33%	20,75%	18,29%	22,01%	16,49%
RF, $\rho = 0.05$	$Sy = 0$	17,84%	25,54%	22,71%	22,23%	22,36%
	$Sy = 1$	21,74%	26,44%	22,79%	21,82%	22,92%
	$Sy = 2$	15,75%	27,34%	23,81%	24,12%	24,44%
	$Sy = 3$	22,46%	25,17%	19,62%	17,36%	18,63%
	$Sy = 4$	23,31%	27,07%	24,04%	24,81%	24,21%
	$Sy = 5$	26,21%	28,38%	23,54%	22,16%	23,85%
RF, $\rho = 0.1$	$Sy = 0$	22,64%	18,31%	25,67%	22,88%	26,04%
	$Sy = 1$	21,13%	19,43%	26,22%	23,27%	25,22%
	$Sy = 2$	21,10%	19,59%	26,87%	24,77%	26,05%
	$Sy = 3$	16,72%	15,67%	21,39%	19,03%	25,45%
	$Sy = 4$	15,33%	16,61%	23,92%	24,81%	26,11%
	$Sy = 5$	16,82%	19,55%	26,12%	24,19%	25,30%
RF, $\rho = 0.3$	$Sy = 0$	18,85%	24,16%	24,59%	25,56%	15,91%
	$Sy = 1$	18,41%	23,58%	24,31%	24,63%	15,96%
	$Sy = 2$	19,09%	24,36%	24,74%	24,38%	17,04%
	$Sy = 3$	16,10%	19,50%	25,18%	23,97%	19,06%
	$Sy = 4$	18,94%	24,77%	26,69%	24,62%	19,25%
	$Sy = 5$	18,80%	22,43%	24,16%	22,66%	19,25%
RF, $\rho = 0.5$	$Sy = 0$	24,54%	24,72%	24,99%	25,56%	23,99%
	$Sy = 1$	24,27%	24,33%	24,51%	24,63%	23,33%
	$Sy = 2$	24,69%	23,76%	24,84%	24,38%	23,86%
	$Sy = 3$	21,96%	21,13%	24,37%	23,97%	23,97%
	$Sy = 4$	23,85%	24,77%	25,42%	24,62%	24,62%
	$Sy = 5$	23,75%	22,43%	22,45%	22,66%	23,41%
RNN, $\rho = 0.001$	$Sy = 0$	26,33%	27,11%	23,70%	24,82%	20,01%
	$Sy = 1$	27,02%	26,94%	23,67%	23,87%	19,14%
	$Sy = 2$	27,56%	27,37%	24,91%	23,86%	18,96%
	$Sy = 3$	27,86%	28,40%	25,77%	23,97%	20,18%
	$Sy = 4$	26,50%	26,85%	23,20%	24,62%	20,57%
	$Sy = 5$	26,67%	26,10%	24,69%	23,41%	18,57%

Table IV.27: **OIL Annualized P&Ls, 2^{nd} DB**

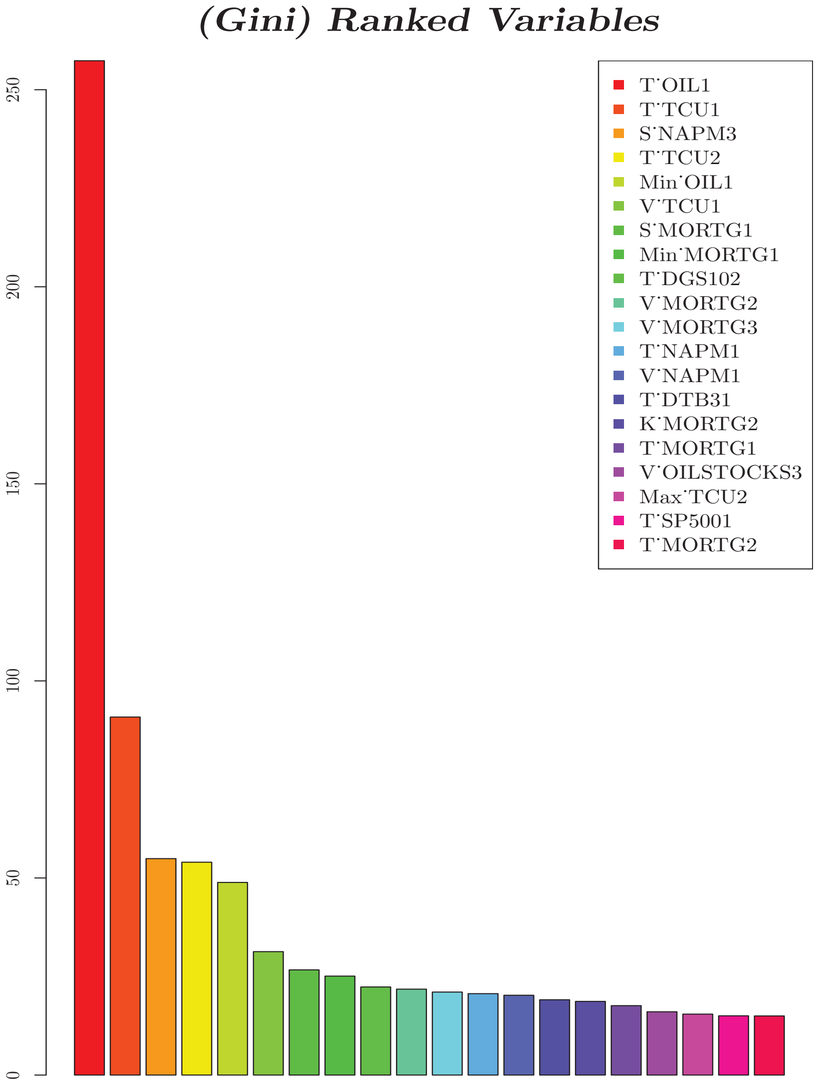


Figure IV.84: **Gini Features Importance,** $\rho = 0.001$, 1^{st} **DB**

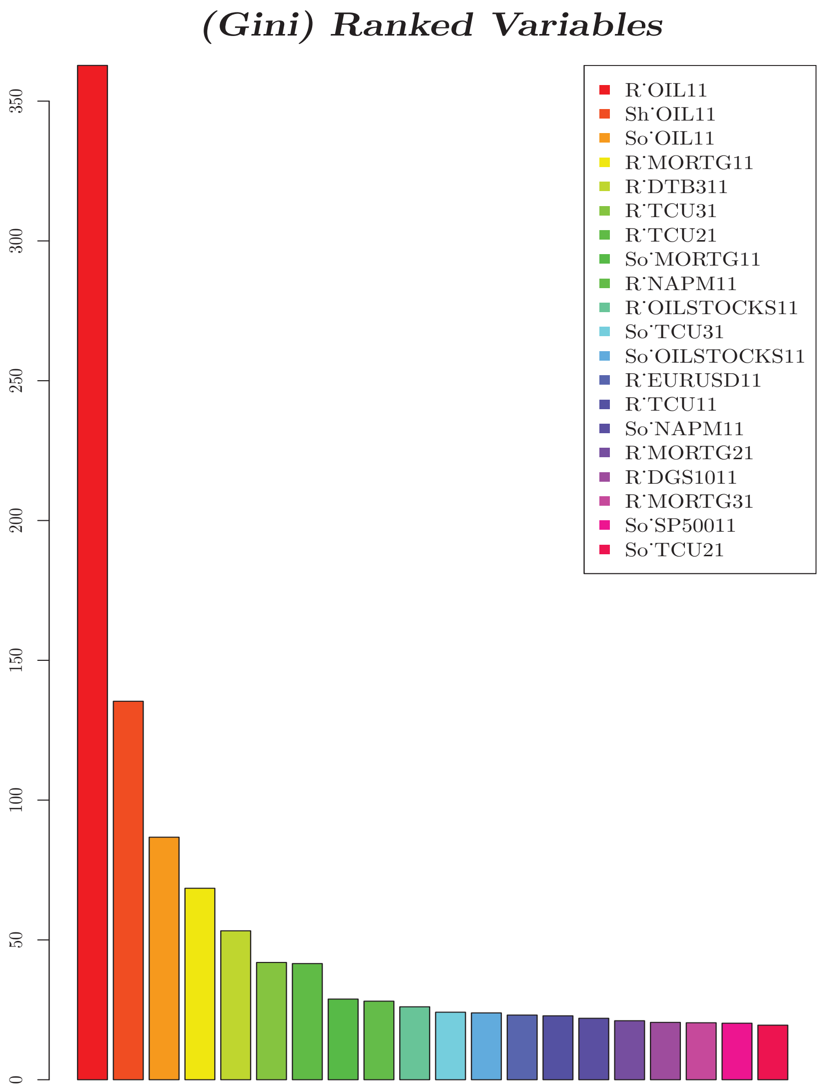


Figure IV.85: **Gini Features Importance,** $\rho = 0.001$, 2^{nd} **DB**

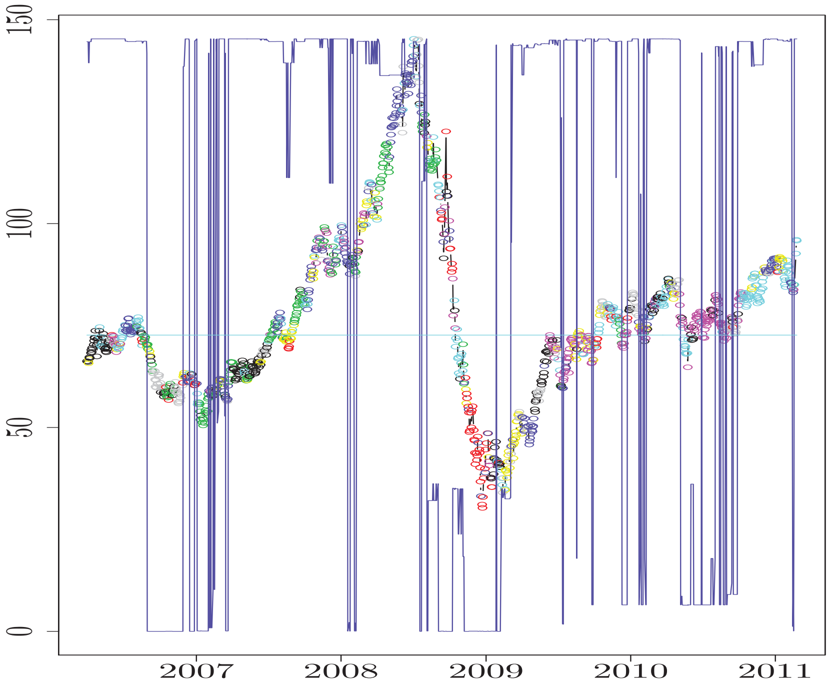


Figure IV.86: **OIL Scores, 1^{st} DB**

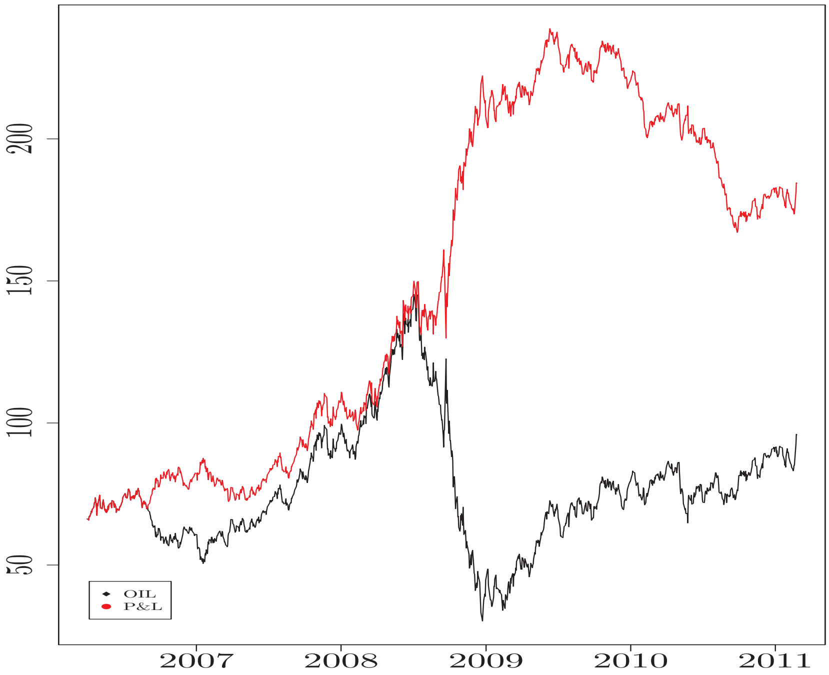


Figure IV.87: **OIL P&L,** $\theta_{buy} = 0.6, \theta_{sell} = 0.4$, 1^{st} **DB**

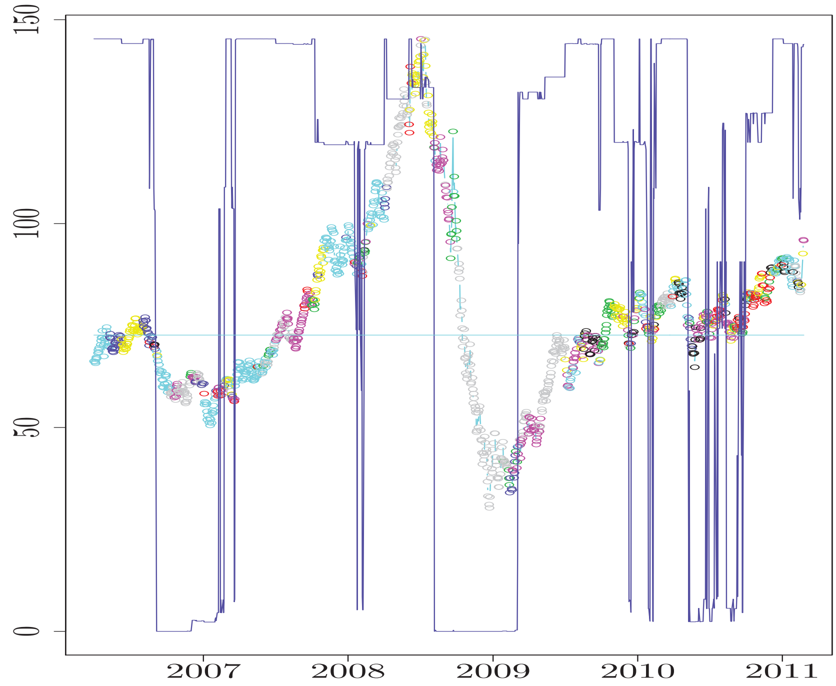


Figure IV.88: **OIL Scores, 2^{nd} DB**

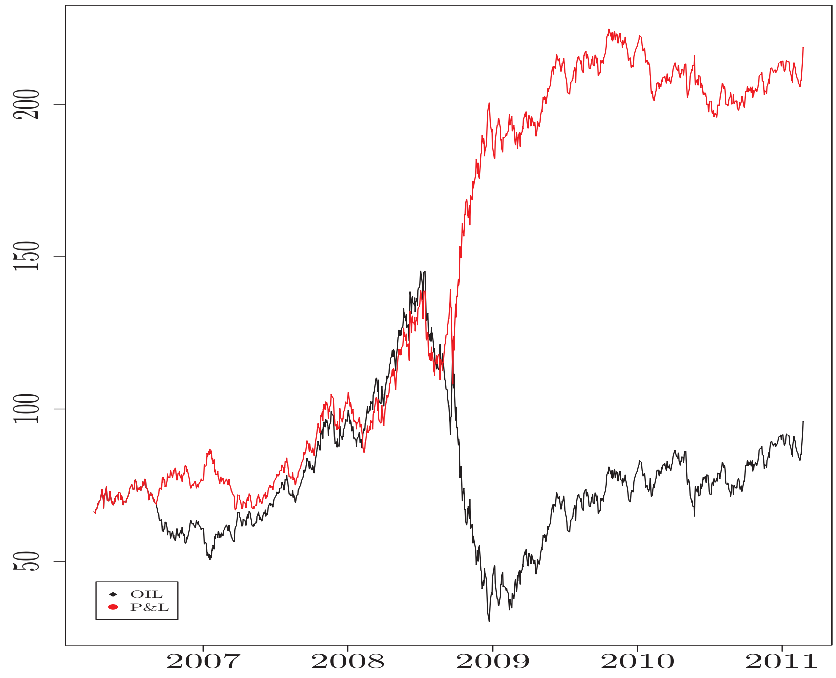


Figure IV.89: **OIL P&L,** $\theta_{buy} = 0.6, \theta_{sell} = 0.4$**,** 2^{nd} **DB**

EURUSD Backtesting Results

Explanatory Variables	SP500	CAC40	DAX	MORTG
	OIL	GOLD	NAPM	
	DGS10	DTB3	DED3	EURDGS10

Table IV.28: **Data for EURUSD**

Training Set	From	05/08/1991	to	31/03/2006
Test Set	From	31/03/2006	to	31/03/2011

Table IV.29: **Size of the Sets**

		Trn	**Trn** $-$	**Trn** $+$	**Test**	**Test** $-$	**Test** $+$
RF	$\rho = 0.001$	93,3%	92,88%	93,73%	47,85%	60,40%	41,87%
	$\rho = 0.01$	92,23%	92,17%	92,29%	34,09%	36,84%	32,78%
	$\rho = 0.05$	87,55%	85,64%	89,52%	53,60%	62,41%	49,40%
	$\rho = 0.1$	79,1%	77,39%	80,87%	71,26%	68,67%	72,49%
	$\rho = 0.3$	71,44%	66,76%	76,26%	81,21%	66,42%	88,28%
	$\rho = 0.5$	71,36%	66,76%	76,10%	81,30%	66,67%	88,28%
RNN	$\rho = 0.001$	70,9%	68,11%	73,77%	81,30%	66,67%	88,28%

Table IV.30: **EURUSD Recognition Rates, 1^{st} DB**

		Trn	**Trn** $-$	**Trn** $+$	**Test**	**Test** $-$	**Test** $+$
RF	$\rho = 0.001$	94,97%	94,73%	95,21%	74,82%	57,27%	81,59%
	$\rho = 0.01$	93%	93,28%	92,70%	76,84%	63,37%	82,04%
	$\rho = 0.05$	84,33%	86,51%	82,06%	77,09%	49,42%	87,77%
	$\rho = 0.1$	81,59%	83,77%	79,33%	77,33%	53,78%	86,42%
	$\rho = 0.3$	76,18%	71,31%	81,23%	84,29%	73,26%	88,55%
	$\rho = 0.5$	76,18%	71,57%	80,95%	84,62%	74,42%	88,55%
RNN	$\rho = 0.001$	75,93%	73,78%	78,16%	84,62%	75,29%	88,22%

Table IV.31: **EURUSD Recognition Rates, 2^{nd} DB**

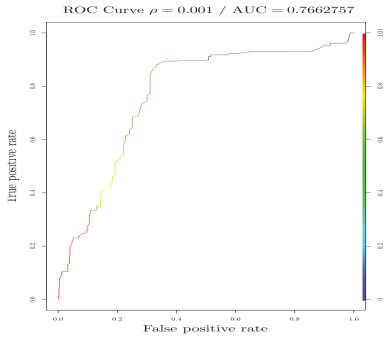


Figure IV.90: **RNN**, $\rho = 0.001$, 1^{st} **DB**

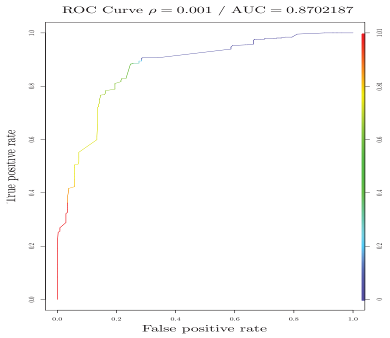


Figure IV.91: **RNN**, $\rho = 0.001$, 2^{nd} **DB**

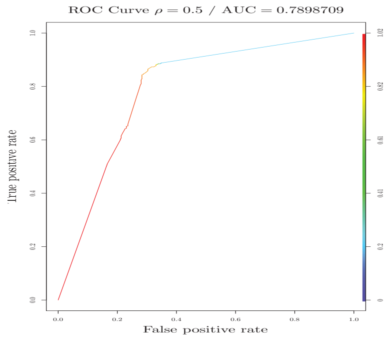


Figure IV.92: **RF**, $\rho = 0.5$, 1^{st} **DB**

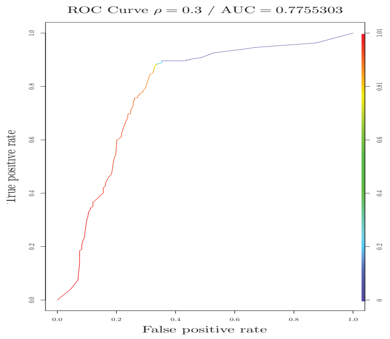


Figure IV.93: **RF**, $\rho = 0.3$, 1^{st} **DB**

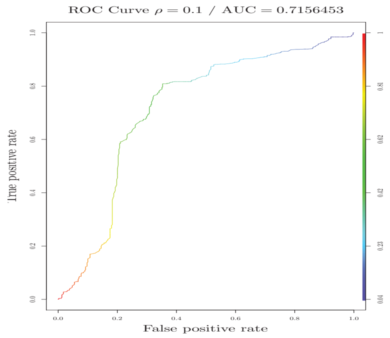


Figure IV.94: **RF**, $\rho = 0.1$, 1^{st} **DB**

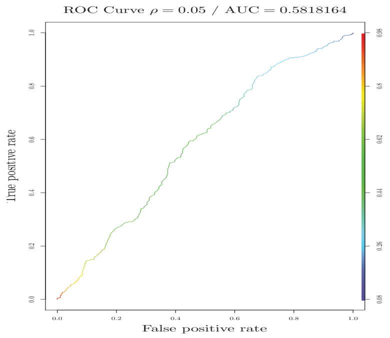


Figure IV.95: **RF**, $\rho = 0.05$, 1^{st} **DB**

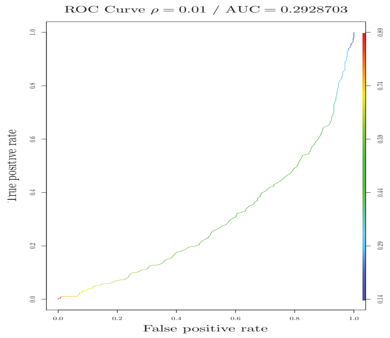


Figure IV.96: **RF**, $\rho = 0.01$, 1^{st} **DB**

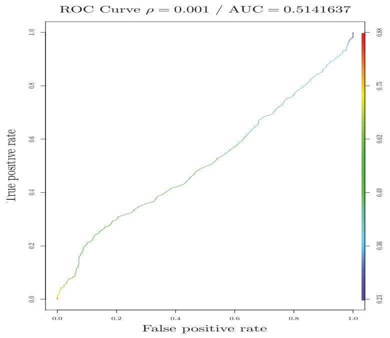


Figure IV.97: **RF**, $\rho = 0.001$, 1^{st} **DB**

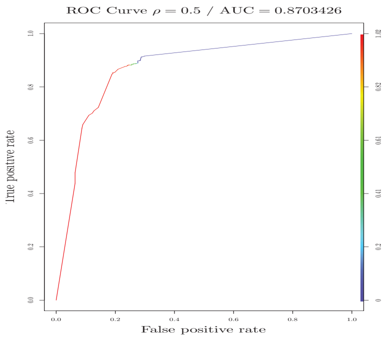


Figure IV.98: **RF**, $\rho = 0.5$, 2^{nd} **DB**

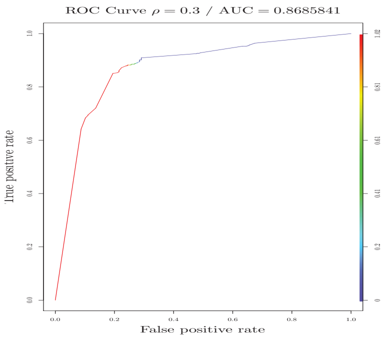


Figure IV.99: **RF**, $\rho = 0.3$, 2^{nd} **DB**

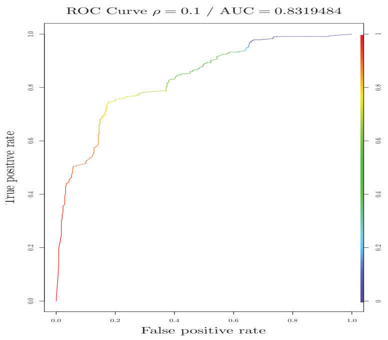


Figure IV.100: **RF**, $\rho = 0.1$, 2^{nd} **DB**

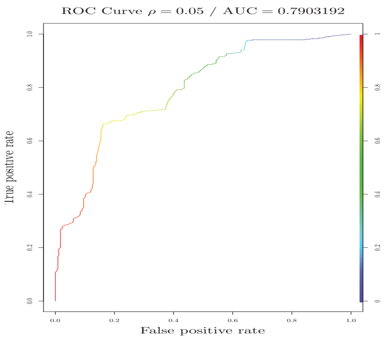


Figure IV.101: **RF**, $\rho = 0.05$, 2^{nd} **DB**

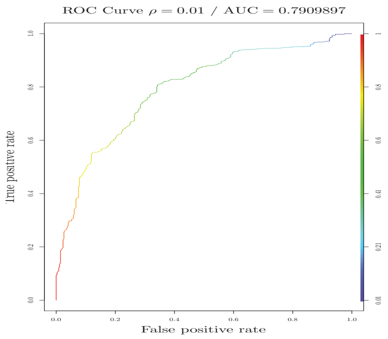


Figure IV.102: **RF**, $\rho = 0.01$, 2^{nd} **DB**

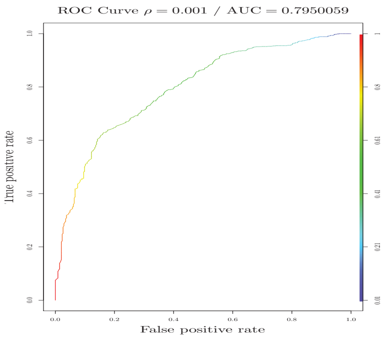


Figure IV.103: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	-5,66%	-6,45%	0,41%	1,48%	-0,03%
	$Sy = 1$	-2,32%	-5,25%	0,63%	1,48%	-0,03%
	$Sy = 2$	-0,82%	-4,04%	0,72%	1,76%	-0,03%
	$Sy = 3$	0,01%	-6,80%	0,53%	1,76%	-0,03%
	$Sy = 4$	-0,84%	-7,19%	0,61%	1,86%	-0,03%
	$Sy = 5$	4,59%	-8,96%	1,54%	1,76%	-0,03%
RF, $\rho = 0.01$	$Sy = 0$	-9,46%	-2,65%	4,49%	9,65%	-0,03%
	$Sy = 1$	-11,49%	-3,54%	3,73%	9,74%	-0,03%
	$Sy = 2$	-13,91%	-2,79%	5,72%	10%	-0,03%
	$Sy = 3$	-16,14%	-4,16%	3,56%	9,97%	-0,03%
	$Sy = 4$	-9,41%	-3,90%	3,90%	9,61%	-0,03%
	$Sy = 5$	-8,14%	-6,78%	5,43%	9,68%	-0,03%
RF, $\rho = 0.05$	$Sy = 0$	-5,08%	-2,18%	0,99%	-0,90%	0,50%
	$Sy = 1$	-7,79%	-4,39%	-1,34%	-1,05%	0,64%
	$Sy = 2$	-6,41%	-3,43%	-1,32%	-1,46%	1,12%
	$Sy = 3$	-6,34%	-2,26%	-2,42%	-1,51%	0,93%
	$Sy = 4$	-7,03%	-2,89%	-2,07%	-1,71%	1,03%
	$Sy = 5$	-3,37%	-2,28%	-2,15%	-0,83%	1,91%
RF, $\rho = 0.1$	$Sy = 0$	3,41%	6,46%	3,18%	0,31%	-1,03%
	$Sy = 1$	3,96%	6,17%	1,30%	-0,57%	-0,59%
	$Sy = 2$	6,03%	7,12%	2,17%	-0,49%	-0,55%
	$Sy = 3$	8,54%	7,20%	1,34%	-0,43%	-1,85%
	$Sy = 4$	6,37%	7,18%	2,92%	-0,16%	-0,98%
	$Sy = 5$	6,81%	7,12%	1,42%	1,22%	-0,17%
RF, $\rho = 0.3$	$Sy = 0$	9,89%	10,12%	10,12%	8,17%	8,33%
	$Sy = 1$	9,45%	9,91%	9,91%	8,13%	6,60%
	$Sy = 2$	7,85%	7,64%	7,64%	6,43%	7,04%
	$Sy = 3$	8,03%	8,48%	8,48%	8,43%	7,29%
	$Sy = 4$	7,68%	7,68%	7,68%	7,68%	6,51%
	$Sy = 5$	6,70%	7,18%	7,18%	7,19%	6,03%
RF, $\rho = 0.5$	$Sy = 0$	10,12%	10,12%	10,12%	9,89%	9,25%
	$Sy = 1$	9,45%	9,91%	9,91%	9,91%	7,67%
	$Sy = 2$	7,85%	7,64%	7,64%	7,60%	7,83%
	$Sy = 3$	8,03%	8,48%	8,48%	8,43%	8,26%
	$Sy = 4$	7,68%	7,68%	7,68%	7,68%	7,74%
	$Sy = 5$	6,70%	7,18%	7,18%	7,19%	6,74%
RNN, $\rho = 0.001$	$Sy = 0$	10,12%	4,39%	5,51%	4,05%	2,13%
	$Sy = 1$	9,45%	4,40%	5,53%	4,28%	2,46%
	$Sy = 2$	7,85%	4,69%	4,56%	3,31%	1,55%
	$Sy = 3$	8,03%	4,32%	4,88%	3,01%	1,12%
	$Sy = 4$	7,68%	3,69%	6,29%	4,86%	2,86%
	$Sy = 5$	6,70%	5,61%	6,45%	4,65%	2,99%

Table IV.32: **EURUSD Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	7,24%	12,39%	8,58%	9,32%	4,77%
	$Sy = 1$	7,45%	11,72%	8,64%	9,60%	4,24%
	$Sy = 2$	8,85%	10,58%	8,04%	9,71%	4,45%
	$Sy = 3$	9,62%	11,57%	8,58%	9,80%	4,47%
	$Sy = 4$	9,16%	10,18%	6,51%	10,08%	4,03%
	$Sy = 5$	6,45%	11,2%	8,26%	10,27%	4,5%
RF, $\rho = 0.01$	$Sy = 0$	8,62%	10,31%	4,83%	10,66%	1,47%
	$Sy = 1$	8,84%	10,24%	5,43%	11,15%	1,14%
	$Sy = 2$	9,64%	8,24%	4,13%	10,38%	1,32%
	$Sy = 3$	11,83%	9,09%	6,27%	10,88%	1,43%
	$Sy = 4$	11,64%	6,55%	6,34%	10,72%	1,99%
	$Sy = 5$	8,21%	9,13%	7,42%	11,05%	3,15%
RF, $\rho = 0.05$	$Sy = 0$	9,21%	10,31%	11,08%	8,61%	7,32%
	$Sy = 1$	10,6%	11,42%	11,50%	10,13%	8,82%
	$Sy = 2$	9,37%	10,56%	11,14%	8,36%	7,10%
	$Sy = 3$	10,61%	11,11%	11,58%	10,37%	8,70%
	$Sy = 4$	11,39%	11,13%	10,34%	8,24%	7,24%
	$Sy = 5$	10,52%	12,20%	12,42%	8,99%	7,30%
RF, $\rho = 0.1$	$Sy = 0$	7,95%	7,95%	7,80%	7,36%	10,27%
	$Sy = 1$	8,93%	8,71%	8,64%	8,69%	9,27%
	$Sy = 2$	6,94%	7,57%	7,54%	6,93%	10,10%
	$Sy = 3$	7,23%	8,15%	7,97%	8,11%	9,23%
	$Sy = 4$	7,92%	6,85%	6,46%	6,43%	7,93%
	$Sy = 5$	8,24%	8,01%	8,30%	8,40%	8,74%
RF, $\rho = 0.3$	$Sy = 0$	10,22%	10,28%	9,74%	8,55%	9,01%
	$Sy = 1$	10,18%	9,96%	9,91%	8,73%	9,16%
	$Sy = 2$	9,29%	7,60%	7,60%	7,03%	7,40%
	$Sy = 3$	8,20%	8,15%	8,43%	7,86%	7,86%
	$Sy = 4$	8,96%	7,68%	6,38%	6,38%	6,76%
	$Sy = 5$	7,13%	7,19%	7,19%	5,62%	6,01%
RF, $\rho = 0.5$	$Sy = 0$	10,51%	10,07%	9,89%	9,74%	8,55%
	$Sy = 1$	9,96%	9,91%	9,91%	9,91%	8,73%
	$Sy = 2$	7,64%	7,50%	7,60%	7,60%	7,03%
	$Sy = 3$	8,20%	8,43%	8,43%	8,43%	7,86%
	$Sy = 4$	7,68%	7,68%	7,68%	6,38%	6,38%
	$Sy = 5$	7,18%	7,19%	7,19%	7,19%	5,62%
RNN, $\rho = 0.001$	$Sy = 0$	9,84%	8,62%	8,70%	2,29%	3,01%
	$Sy = 1$	9,34%	8,14%	8,07%	1,29%	2,64%
	$Sy = 2$	7,85%	7,38%	7,33%	1,32%	2,40%
	$Sy = 3$	8,03%	7,45%	6,20%	0,96%	1,63%
	$Sy = 4$	7,68%	6,38%	4,17%	-0,35%	0,28%
	$Sy = 5$	6,70%	5,10%	5,01%	1,14%	0,53%

Table IV.33: **EURUSD Annualized P&Ls, 2^{nd} DB**

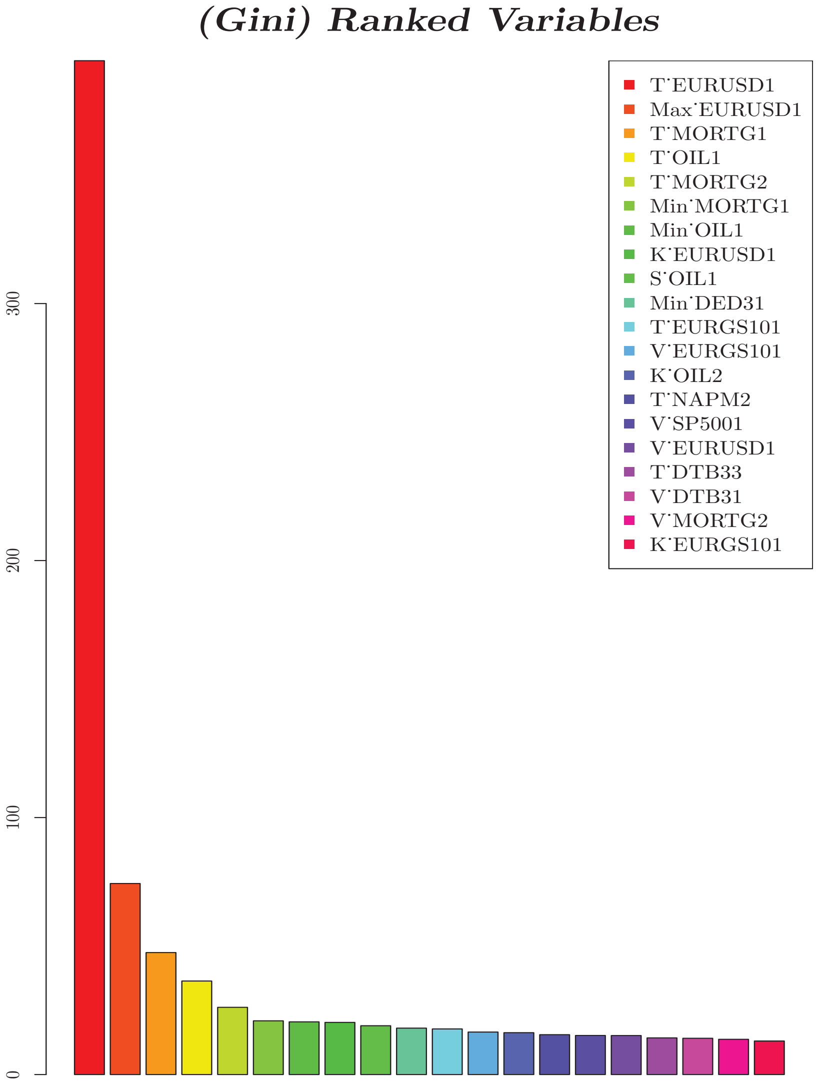


Figure IV.104: **Gini Features Importance, $\rho = 0.001$, 1^{st} DB**

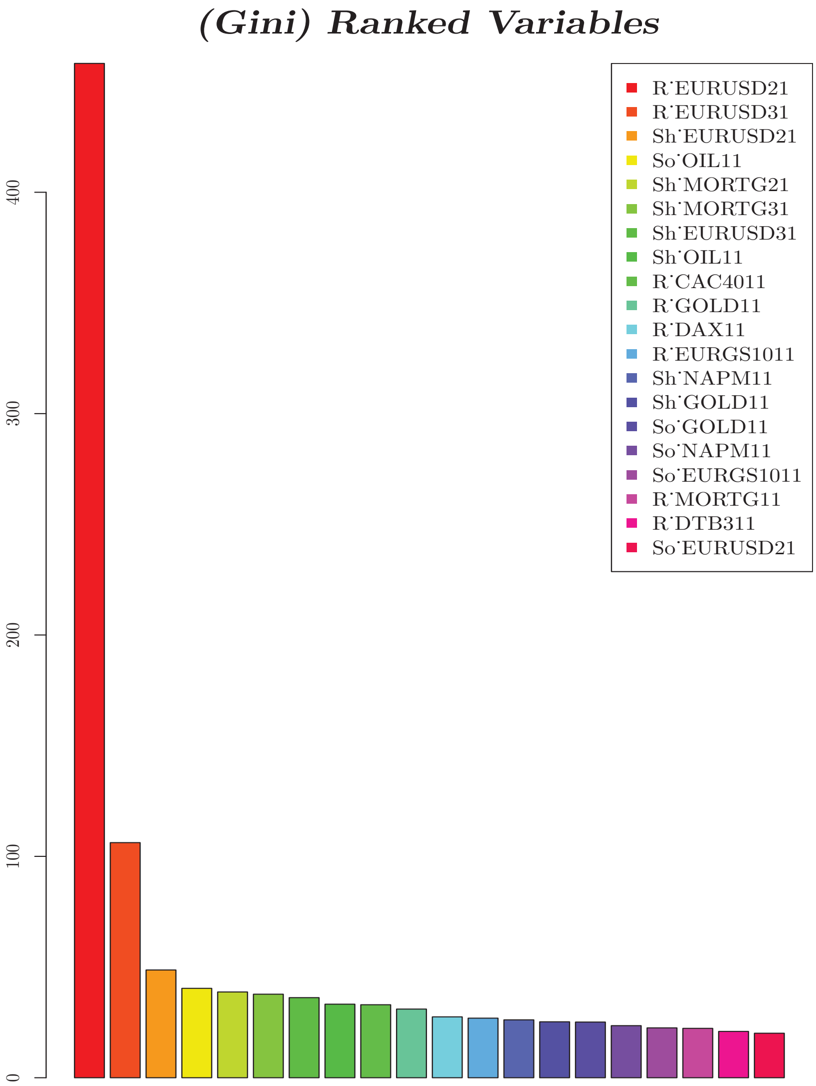


Figure IV.105: **Gini Features Importance**, $\rho = 0.001$, 2^{nd} **DB**

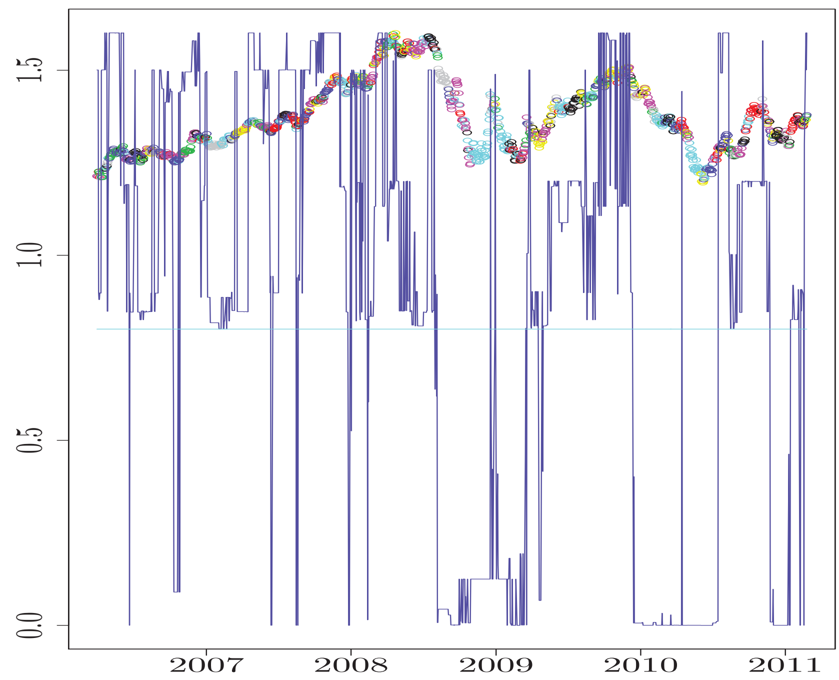


Figure IV.106: **EURUSD Scores, 1^{st} DB**

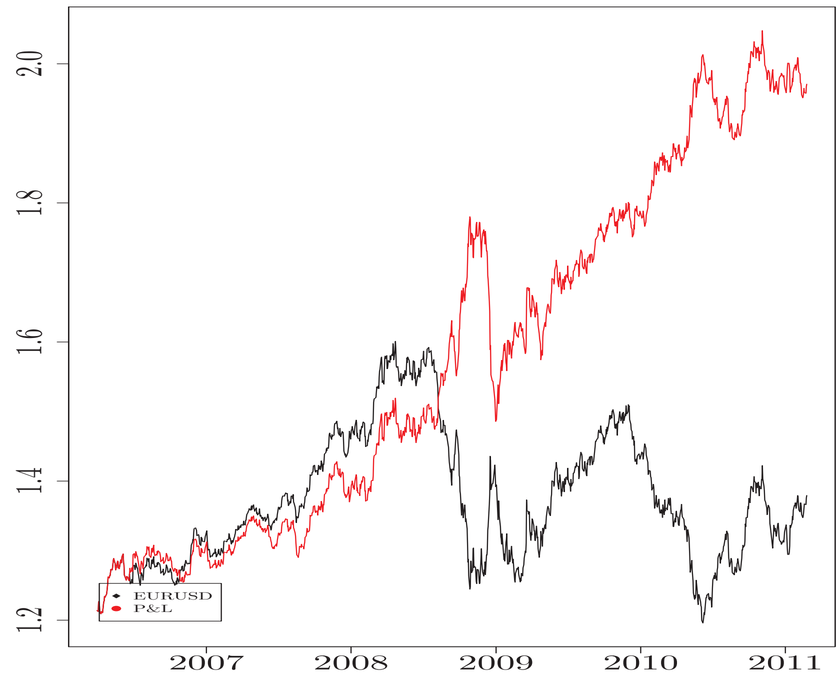


Figure IV.107: **EURUSD P&L, $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} DB**

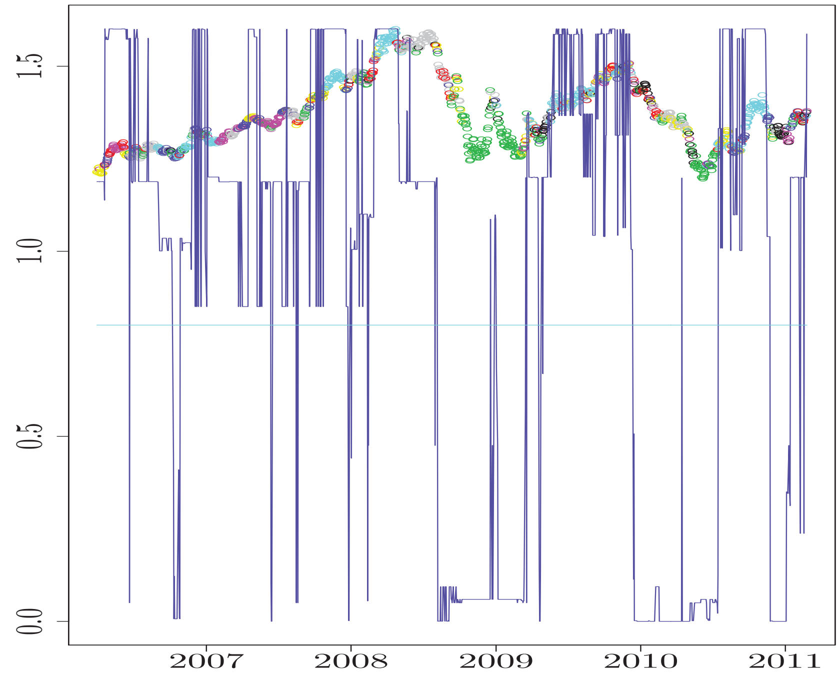


Figure IV.108: **EURUSD Scores, 2^{nd} DB**

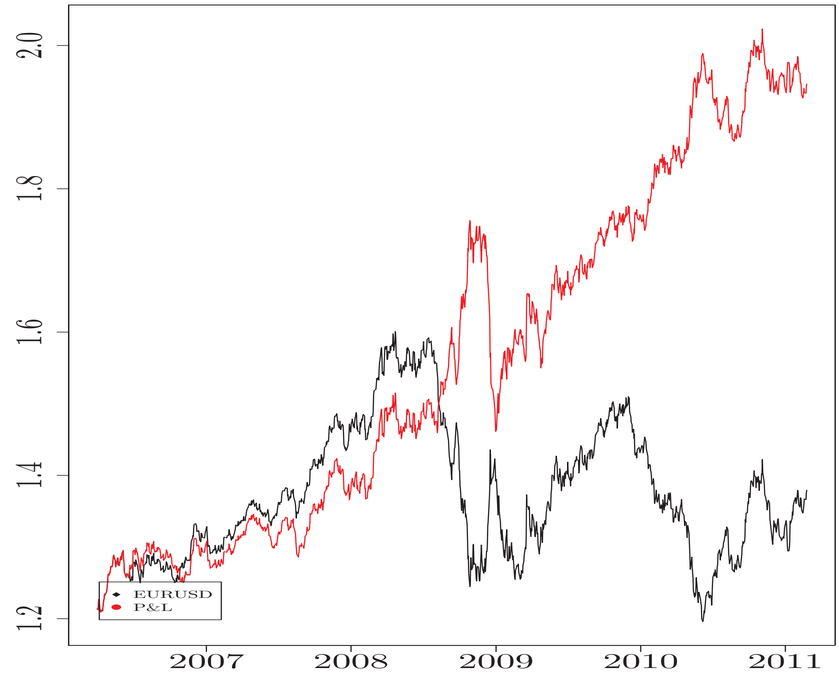


Figure IV.109: **EURUSD P&L,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

DEXUSUK Backtesting Results

Explanatory Variables	SP500	GOLD	EURUSD
	DGS10	EURDGS10	

Table IV.34: **Data for DEXUSUK**

Training Set	From	13/09/1971	to	31/03/2006
Test Set	From	31/03/2006	to	31/03/2011

Table IV.35: **Size of the Sets**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	92,66%	92,64%	92,68%	72,39%	72,41%	72,38%
	$\rho = 0.01$	88,72%	88,55%	88,91%	71,17%	71,23%	71,13%
	$\rho = 0.05$	78,72%	81,75%	75,58%	70,61%	73,97%	68,23%
	$\rho = 0.1$	74,21%	75,48%	72,90%	74,25%	70,45%	76,93%
	$\rho = 0.3$	74,04%	75,32%	72,71%	74,33%	70,65%	76,93%
	$\rho = 0.5$	72,72%	71,91%	73,56%	73,85%	67,91%	78,04%
RNN	$\rho = 0.001$	71,39%	70,09%	72,73%	73,68%	67,51%	78,04%

Table IV.36: **DEXUSUK Recognition Rates, 1^{st} DB**

		Trn	**Trn −**	**Trn +**	**Test**	**Test −**	**Test +**
RF	$\rho = 0.001$	93,81%	94,49%	93,14%	74,98%	68,75%	79,16%
	$\rho = 0.01$	87,80%	88,98%	86,63%	77,81%	73,59%	80,65%
	$\rho = 0.05$	81,62%	84,24%	79,03%	75,71%	70,97%	78,89%
	$\rho = 0.1$	80,06%	78,82%	81,29%	78,95%	70,77%	84,44%
	$\rho = 0.3$	78,11%	76,66%	79,54%	76,92%	70,77%	81,06%
	$\rho = 0.5$	78,10%	76,64%	79,54%	76,92%	70,77%	81,06%
RNN	$\rho = 0.001$	77,21%	75,29%	79,10%	76,84%	70,36%	81,19%

Table IV.37: **DEXUSUK Recognition Rates, 2^{nd} DB**

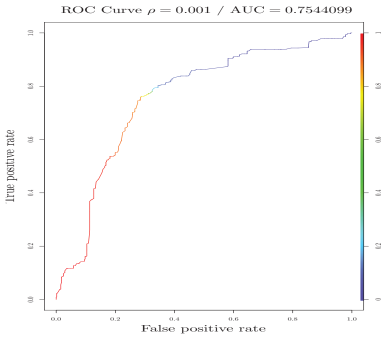


Figure IV.110: **RNN**, $\rho = 0.001$, 1^{st} **DB**

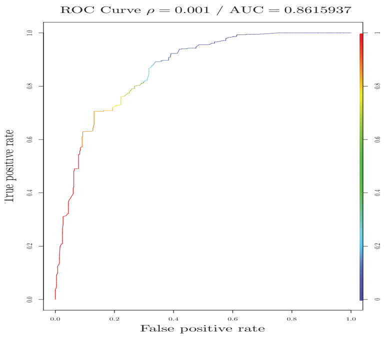


Figure IV.111: **RNN**, $\rho = 0.001$, 2^{nd} **DB**

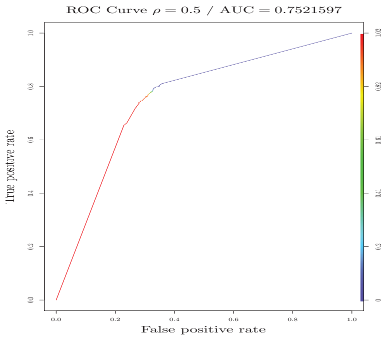


Figure IV.112: **RF**, $\rho = 0.5$, 1^{st} **DB**

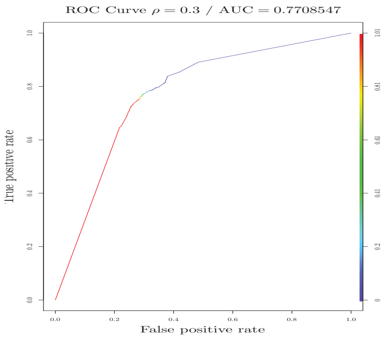


Figure IV.113: **RF**, $\rho = 0.3$, 1^{st} **DB**

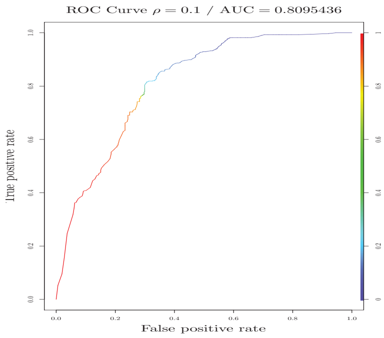


Figure IV.114: **RF**, $\rho = 0.1$, 1^{st} **DB**

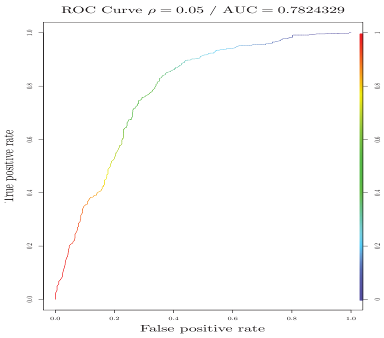


Figure IV.115: **RF**, $\rho = 0.05$, 1^{st} **DB**

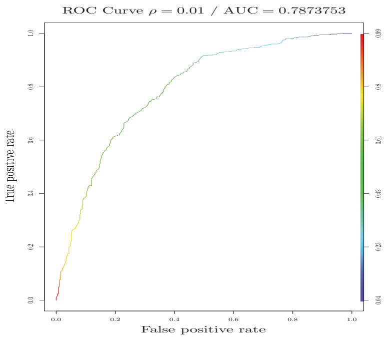


Figure IV.116: **RF**, $\rho = 0.01$, 1^{st} **DB**

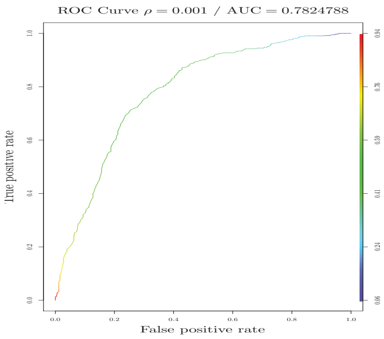


Figure IV.117: **RF**, $\rho = 0.001$, 1^{st} **DB**

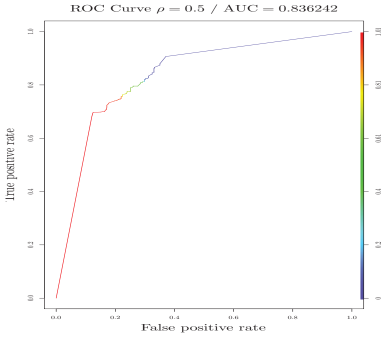


Figure IV.118: **RF**, $\rho = 0.5$, 2^{nd} **DB**

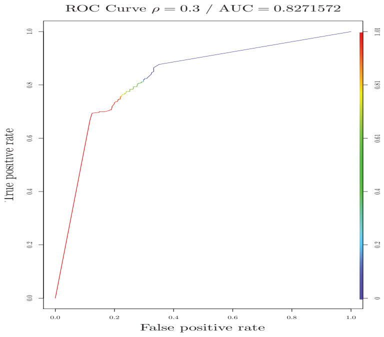


Figure IV.119: **RF**, $\rho = 0.3$, 2^{nd} **DB**

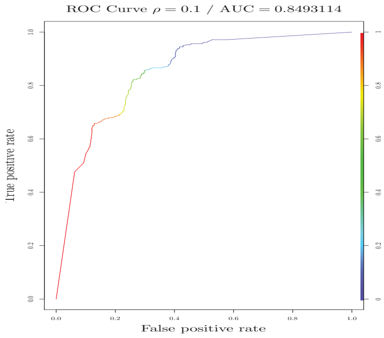


Figure IV.120: **RF**, $\rho = 0.1$, 2^{nd} **DB**

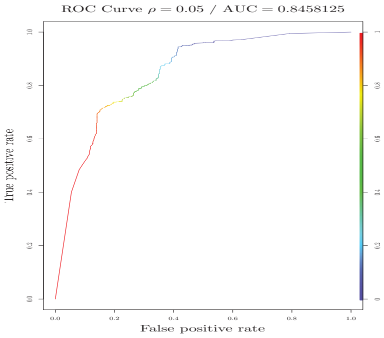


Figure IV.121: **RF**, $\rho = 0.05$, 2^{nd} **DB**

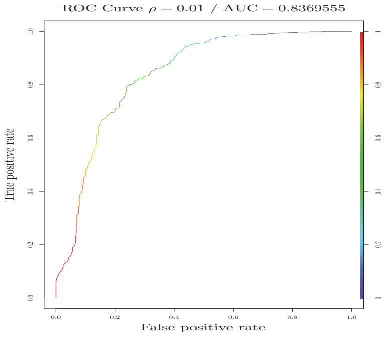


Figure IV.122: **RF**, $\rho = 0.01$, 2^{nd} **DB**

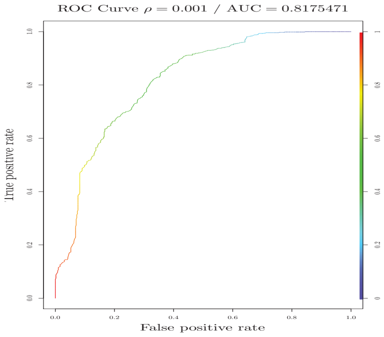


Figure IV.123: **RF**, $\rho = 0.001$, 2^{nd} **DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	4,97%	2,58%	4,23%	5,76%	3,61%
	$Sy = 1$	7,06%	1,46%	3,76%	5,76%	3,62%
	$Sy = 2$	3,91%	3,35%	4,54%	5,68%	3,65%
	$Sy = 3$	7,22%	4,50%	2,79%	5,92%	3,64%
	$Sy = 4$	6,99%	4,66%	3,01%	5,40%	3,63%
	$Sy = 5$	8,97%	0,56%	4,09%	6,20%	3,76%
RF, $\rho = 0.01$	$Sy = 0$	4,51%	6,18%	6,77%	2,36%	6,63%
	$Sy = 1$	3,70%	4,46%	6,07%	1,85%	6,20%
	$Sy = 2$	7,61%	8,20%	7,43%	1,06%	6,59%
	$Sy = 3$	6,46%	5,60%	6,29%	0,63%	6,41%
	$Sy = 4$	7,86%	7,01%	7,44%	-0,52%	5,96%
	$Sy = 5$	7,37%	7,40%	6,06%	0,09%	6,39%
RF, $\rho = 0.05$	$Sy = 0$	1,85%	1,70%	4,69%	5,50%	6,78%
	$Sy = 1$	0,85%	-0,35%	4,63%	4,27%	6,04%
	$Sy = 2$	2,81%	1,84%	4,13%	5,39%	6,23%
	$Sy = 3$	0,26%	-0,44%	3,18%	4,50%	5,59%
	$Sy = 4$	4,38%	3,98%	4,56%	3,83%	4,42%
	$Sy = 5$	2,47%	2,02%	4,27%	4,28%	5%
RF, $\rho = 0.1$	$Sy = 0$	6,30%	6,53%	6,60%	7,02%	3,85%
	$Sy = 1$	4,71%	4,64%	4,69%	5,62%	3,93%
	$Sy = 2$	6,22%	6,22%	5,66%	5,59%	4,50%
	$Sy = 3$	4,94%	4,78%	4,23%	4,66%	1,69%
	$Sy = 4$	5,62%	5,62%	5,62%	5,96%	3,54%
	$Sy = 5$	5,44%	5,44%	4,56%	5,18%	0,28%
RF, $\rho = 0.3$	$Sy = 0$	6,04%	6,43%	6,60%	6,60%	5,93%
	$Sy = 1$	4,71%	4,54%	4,69%	4,69%	4,61%
	$Sy = 2$	6,06%	6,22%	5,66%	5,66%	4,22%
	$Sy = 3$	4,94%	4,78%	4,23%	4,23%	4,23%
	$Sy = 4$	5,59%	5,62%	5,62%	5,62%	5,62%
	$Sy = 5$	5,44%	5,44%	4,56%	4,56%	4,56%
RF, $\rho = 0.5$	$Sy = 0$	4,03%	4,45%	4,63%	4,78%	4,78%
	$Sy = 1$	2,92%	2,74%	2,90%	2,90%	2,90%
	$Sy = 2$	4,06%	4,23%	3,63%	3,63%	3,04%
	$Sy = 3$	4,37%	4,21%	3,65%	3,65%	3,65%
	$Sy = 4$	4,37%	4,40%	4,40%	4,40%	4,40%
	$Sy = 5$	2,41%	2,41%	1,42%	1,42%	1,42%
RNN, $\rho = 0.001$	$Sy = 0$	4,45%	4,63%	4,78%	5,24%	1,15%
	$Sy = 1$	2,74%	2,90%	2,90%	4,05%	1,91%
	$Sy = 2$	4,23%	3,63%	3,63%	4,78%	3,94%
	$Sy = 3$	4,21%	3,65%	3,65%	3,26%	2,26%
	$Sy = 4$	4,40%	4,40%	4,40%	4,85%	4,23%
	$Sy = 5$	2,41%	1,42%	1,42%	4,56%	2,37%

Table IV.38: **DEXUSUK Annualized P&Ls, 1^{st} DB**

		$\theta = 0.5$	$\theta = 0.4$	$\theta = 0.3$	$\theta = 0.2$	$\theta = 0.1$
RF, $\rho = 0.001$	$Sy = 0$	-0,27%	4,76%	4,86%	7,30%	4,54%
	$Sy = 1$	-1,49%	4,87%	3,67%	6,88%	5,01%
	$Sy = 2$	0,11%	4,42%	4%	6,60%	4,59%
	$Sy = 3$	-0,36%	3,32%	3,20%	7,28%	5,22%
	$Sy = 4$	5,62%	4,41%	4,17%	6,42%	5,48%
	$Sy = 5$	1,68%	4,56%	4,59%	6,08%	5,24%
RF, $\rho = 0.01$	$Sy = 0$	5,03%	5,38%	6,08%	6,76%	6,18%
	$Sy = 1$	4,16%	3,21%	4,85%	5,69%	5,52%
	$Sy = 2$	4,90%	4,42%	5,72%	6,13%	5,50%
	$Sy = 3$	2,98%	1,71%	5,51%	5,05%	4,82%
	$Sy = 4$	6,77%	5,77%	5,54%	5,15%	5,23%
	$Sy = 5$	6,16%	5,07%	6,46%	5,94%	5,41%
RF, $\rho = 0.05$	$Sy = 0$	4,54%	4,48%	4,58%	5,12%	9,01%
	$Sy = 1$	2,43%	3,41%	3,71%	3,92%	8,32%
	$Sy = 2$	4,09%	4,40%	5,21%	4,90%	8,61%
	$Sy = 3$	2,47%	2,75%	3,14%	3,62%	8,58%
	$Sy = 4$	4,36%	4,42%	4,66%	4,82%	8,68%
	$Sy = 5$	4,26%	4,44%	4,93%	4,72%	8,79%
RF, $\rho = 0.1$	$Sy = 0$	6,53%	6,58%	5,85%	3,08%	6,41%
	$Sy = 1$	4,77%	4,33%	4,49%	2,29%	5,93%
	$Sy = 2$	4,93%	6,27%	5,81%	3,50%	6,10%
	$Sy = 3$	4,69%	4,84%	4,29%	2,17%	5,86%
	$Sy = 4$	6,25%	6,26%	4,76%	4,44%	7,42%
	$Sy = 5$	4,80%	5,49%	5,65%	4,43%	6,54%
RF, $\rho = 0.3$	$Sy = 0$	3,33%	3,12%	3,78%	3,29%	3,12%
	$Sy = 1$	1,51%	1,93%	2,79%	2,55%	3,05%
	$Sy = 2$	3,84%	3,13%	3,73%	3,92%	3,72%
	$Sy = 3$	3,15%	2,78%	3,18%	3,17%	3,25%
	$Sy = 4$	4,38%	4,38%	4,16%	4,16%	4,33%
	$Sy = 5$	2,33%	1,96%	3,75%	3,75%	3,82%
RF, $\rho = 0.5$	$Sy = 0$	3,33%	3,11%	3,60%	3,51%	2,54%
	$Sy = 1$	1,51%	1,93%	2,79%	2,97%	2,65%
	$Sy = 2$	3,84%	3,71%	3,51%	4,14%	3,28%
	$Sy = 3$	3,15%	2,78%	3,18%	3,27%	2,85%
	$Sy = 4$	4,38%	4,38%	4,16%	4,32%	4,68%
	$Sy = 5$	2,33%	1,96%	3,75%	3,81%	3,38%
RNN, $\rho = 0.001$	$Sy = 0$	3,05%	3,21%	3,39%	1,56%	8,73%
	$Sy = 1$	1,61%	1,51%	2,83%	1,95%	8,93%
	$Sy = 2$	3,29%	3,29%	3,44%	1,24%	9,19%
	$Sy = 3$	2,93%	3,15%	3,96%	1,18%	8,63%
	$Sy = 4$	3,65%	3,65%	4,22%	2,78%	8,93%
	$Sy = 5$	2,33%	2,33%	4,25%	1,04%	8,06%

Table IV.39: **DEXUSUK Annualized P&Ls, 2^{nd} DB**

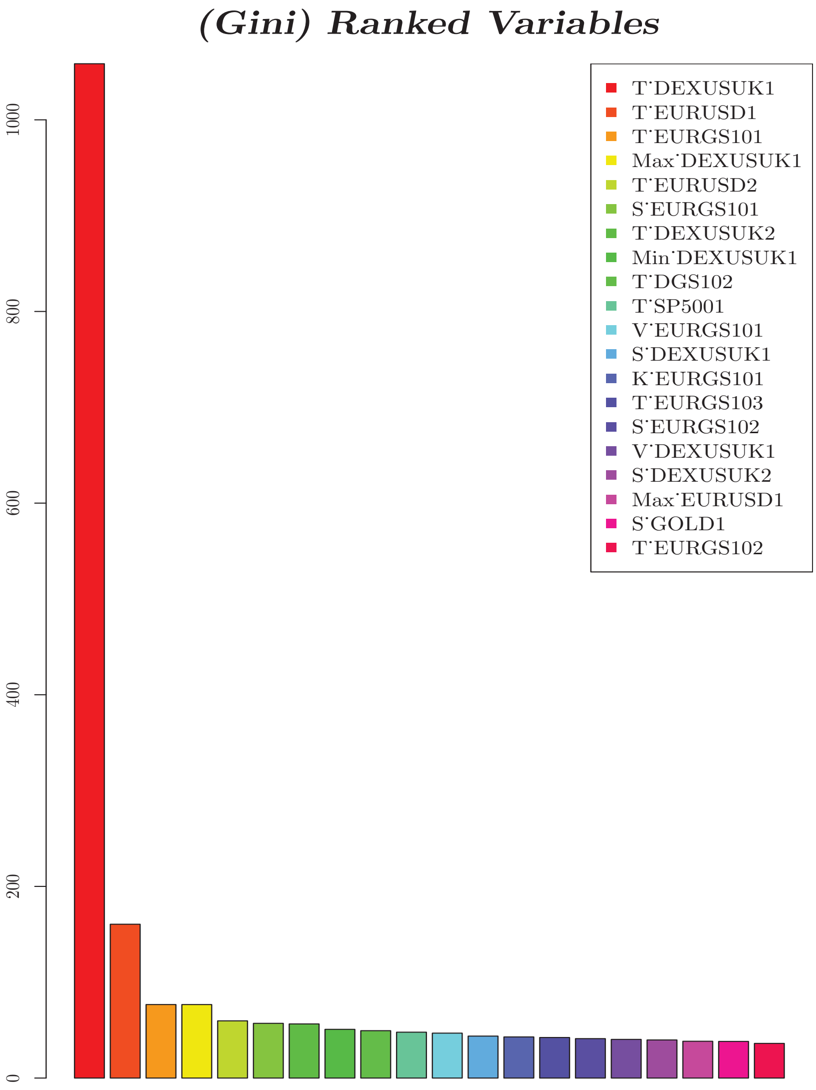


Figure IV.124: **Gini Features Importance, $\rho = 0.001$, 1^{st} DB**

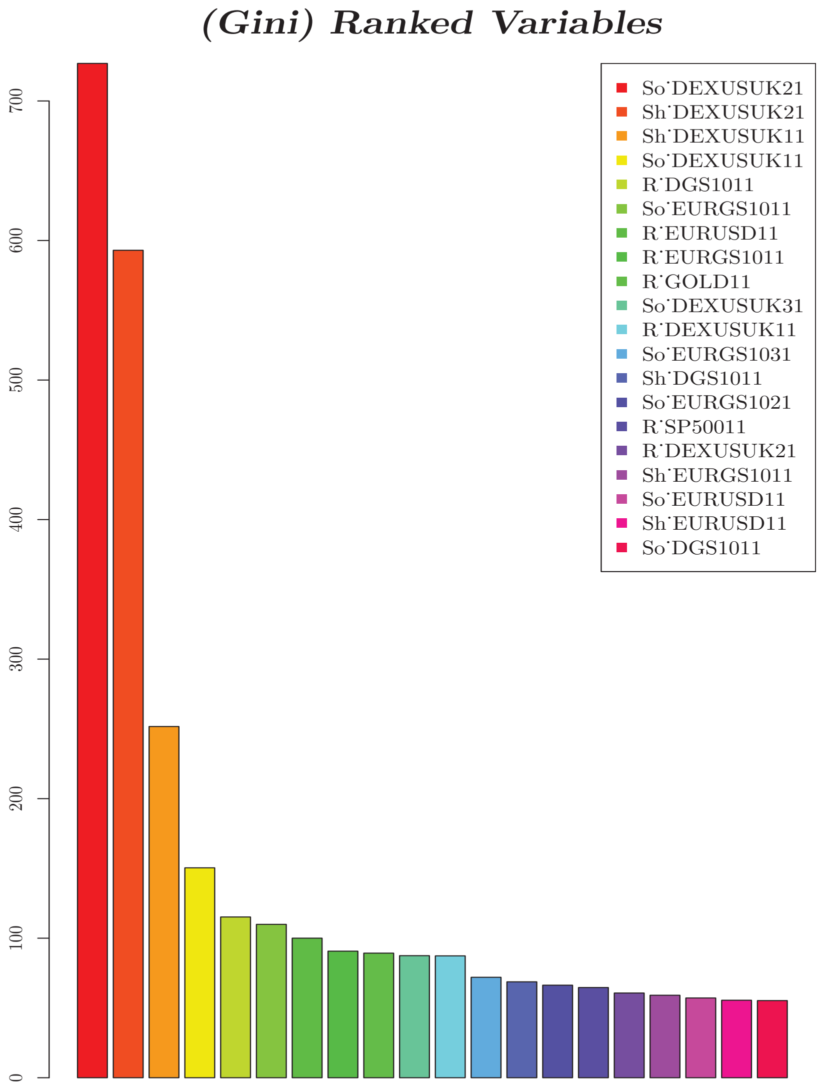


Figure IV.125: **Gini Features Importance, $\rho = 0.001$, 2^{nd} DB**

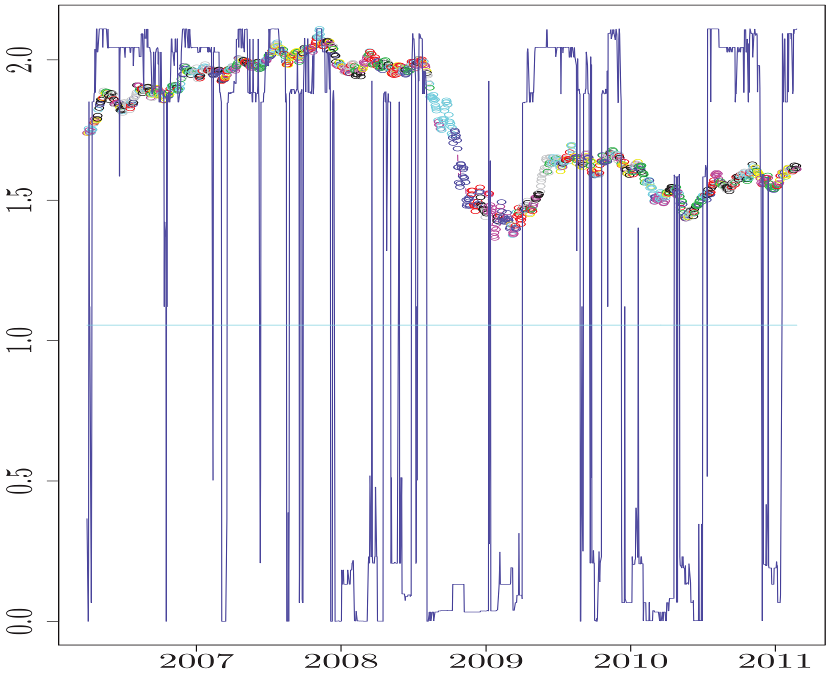


Figure IV.126: **DEXUSUK Scores, 1^{st} DB**

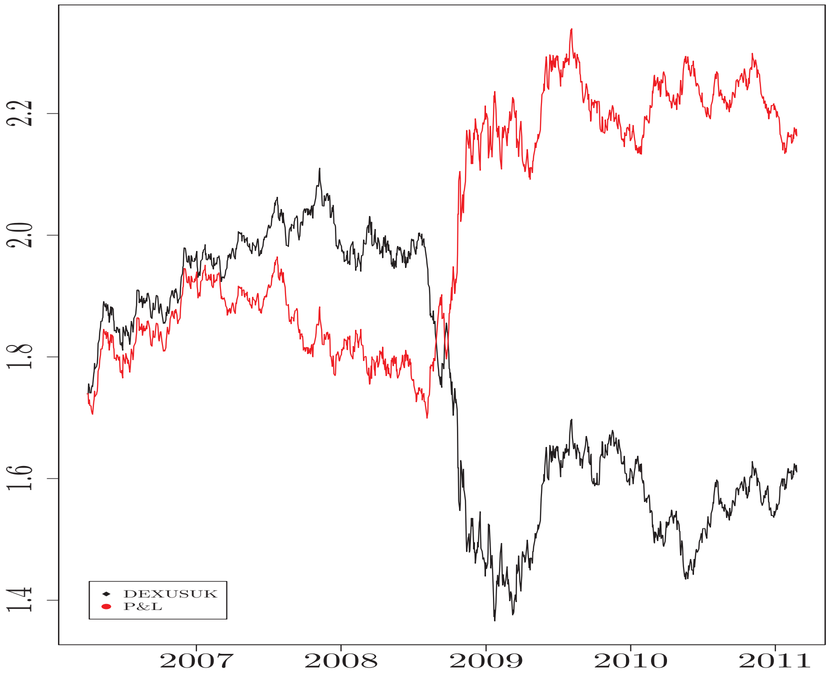


Figure IV.127: **DEXUSUK P&L, $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} DB**

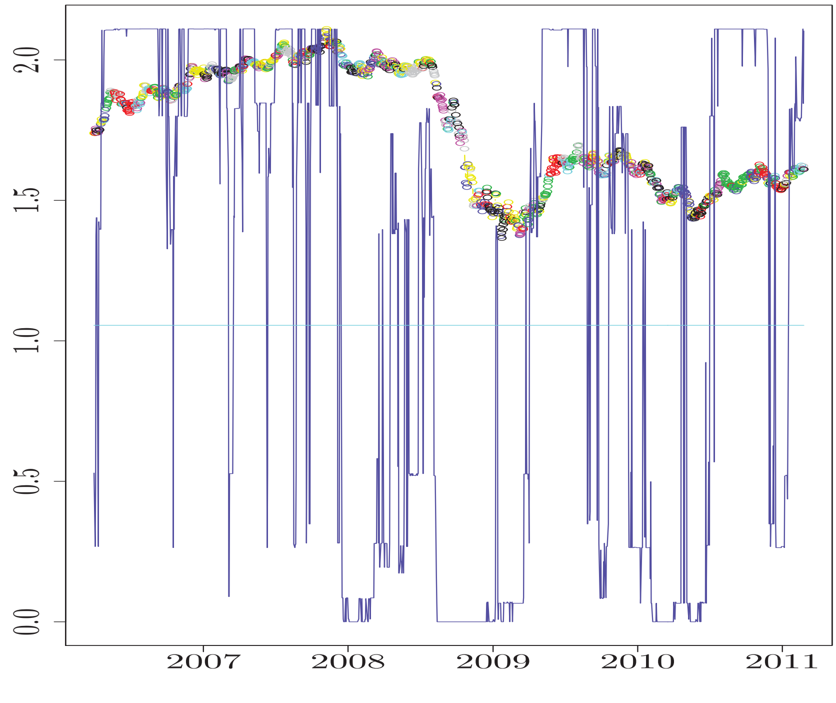


Figure IV.128: **DEXUSUK Scores, 2^{nd} DB**

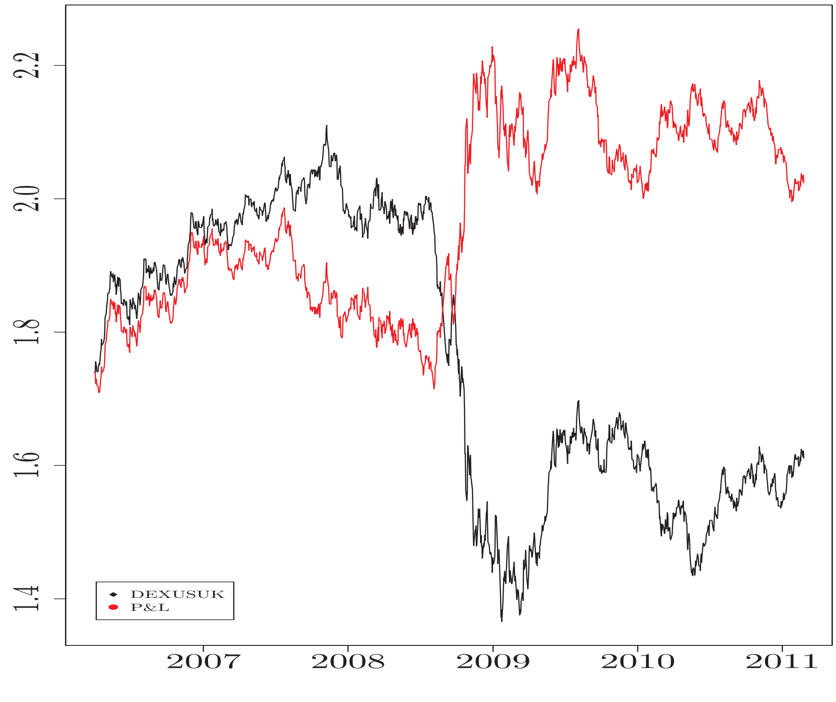


Figure IV.129: **DEXUSUK P&L,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

IV.2.3 Conclusions

RF and RNN numerical models, trained on the two types of UniCart databases using a group of meaningful financial target variables, have been extensively backtested in this subsection. The backtesting results can be summed up in a few points.

1. RF numerical models tend to overfit. The overfitting effect depends on the parameter ρ, which determines the size of the tree classifiers. The higher the value of the parameter ρ, the smaller the size of the tree classifiers. Our experiments show that decreasing gradually the value of ρ usually leads to less and less accurate classifiers.

2. The analysis of feature importance performed on the first type of database highlights that the non-lagged linear trend of the target variable is the most influential feature in the prediction computation. This observation is consistent with the guiding principle of technician analysts: "Follow the trend". Besides, the study of the second type of database shows that returns - of the target variable - computed for the broader scales are the most relevant. These observations emphasize that the role of machine learning in finance is rather to select the scale of observation than the features.

3. RNN numerical models are not submitted to overfitting. The recognition rates and ROC curves of these classifiers underline that the RNN procedure is capable of handling non-stationary features. However, the trading rules based on the predictions of the RNN numerical models are not necessarily highly competitive. Indeed, the UniCart representation also has to be well adapted to the modeled financial series.

Points 1 and 2 have motivated the creation of the Relabeled Nearest Neighbors procedure. The intuition behind was that the importance of features could be taken into account to distinguish between objects of the training set and to draw robust comparisons between the training set and the test set. A decisive advantage of the RNN procedure is that it enables the construction of parameter freed models. We believe that this is essential from a methodological point of view. Indeed, it has allowed us to reliably analyze the link existing between the performance of the trading rules and their sensitivity - measured by the reaction delay parameter. The next section focuses first on the study of the efficient market hypothesis via trading rules based on UniCart RNN numerical models. We propose then further improvements and extensions of the UniCart RNN approach.

IV.3 Further Perspectives for the UniCart RNN Approach

This section is devoted to the analysis of the backtesting results of UniCart RNN numerical models. We believe that this analysis is particularly reliable because it is based on procedures and backtesting plans specifically designed to avoid data snooping. Indeed, the RNN procedure does not require any subroutine for parameter tuning. Hence the UniCart RNN numerical models only depend on the set of UniCart parameters (W, w, h), whose values have been fixed for the whole group of considered target variables and for the two types of databases.

$$\begin{cases} W & = 250 \text{ days} \\ w & = 90 \text{ days} \\ h & = 25 \text{ days.} \end{cases}$$

The fact that the backtesting results of section IV.2, obtained for different target variables with a restricted number of parameters, are consistent with each other gives appropriate credit to the UniCart RNN approach. It also strongly supports the study of the efficient market analysis via the UniCart RNN trading rules, presented in the first subsection. The behaviour of these trading rules suggests an extension of the UniCart RNN approach, which could help improving the stability of the score. This is discussed in the second subsection.

IV.3.1 About the Efficient Market Hypothesis

The aim of this subsection is to question the efficient market hypothesis. In that perspective, we focus on the parameter sensitivity Sy of the UniCart RNN trading rules. This parameter is expressed in days and is used to place buying and selling orders. More specifically, a single buying (resp. selling) order is placed as soon as the RNN score exceeds the buying (resp. selling) threshold during the number of days indicated by the parameter Sy. Note that in our experiments, reported in section IV.2, we can not place more than one order. Therefore, when the score stays above the buying threshold (resp. below the selling threshold), we do not increase the invested money. Instead, we just keep our position. Our method to study the efficient market hypothesis is to observe the P&Ls gained by the UniCart RNN trading rules by making the parameter Sy vary over a wide range of values, from $Sy = 90$ days to $Sy = 1$ day. The graphs IV.130, IV.132, IV.133, IV.134, IV.135 report the annualized P&Ls obtained for the **SP500** index and the first type of UniCart database as well as all the thresholds θ_{buy} and θ_{sell} considered in subsection IV.2.2. The graphs IV.131, IV.137, IV.138, IV.139, IV.140 display the annualized P&Ls specific to the second type of database. Furthermore, the annualized P&Ls computed using the first type of database with parameters $\theta_{buy} = \theta_{sell} = 0.5$ for respectively the variables **STOXX**, **GOLD**, **OIL**, **EURUSD** and **DEXUSUK** are presented by the graphs IV.142, IV.144, IV.146, IV.148 and IV.150. The graphs IV.143, IV.145, IV.147, IV.149 and IV.151 are their counterparts of the second type of database. Note that for clarity and conciseness, all the P&Ls corresponding to all examined parameters θ_{buy} and $\theta_{sell} = 0.5$ are not reproduced in this document. Nonetheless, two decisive observations can be made on all of them.

- The variance of the annualized P&Ls - obtained at the term of the test period - tends to reduce when the sensitivity Sy tends to 1. In other words, the performances of the trading rules are more stable for small sensitivity values.
- The P&Ls are also usually higher for small sensitivity values. The case of **GOLD** is an exception and the contrary is actually observed on graph IV.145. This is explained by the behavior of the price of gold during the test period, which is almost constantly increasing (cf. figure IV.69). The selling orders placed by our trading rules thus almost systematically lead to losses.

These observations may imply that the efficient market hypothesis need a smoother reformulation. Contrarily to what the weak form of the EMH states, markets would then incorporate information "quickly" into prices, and not immediately. This mechanism is however difficult to quantify, all the most that our trading rules do not yield very stable P&Ls even for small sensitivity values. The next and last section suggests improvements of the UniCart RNN approach regarding this stability aspect.

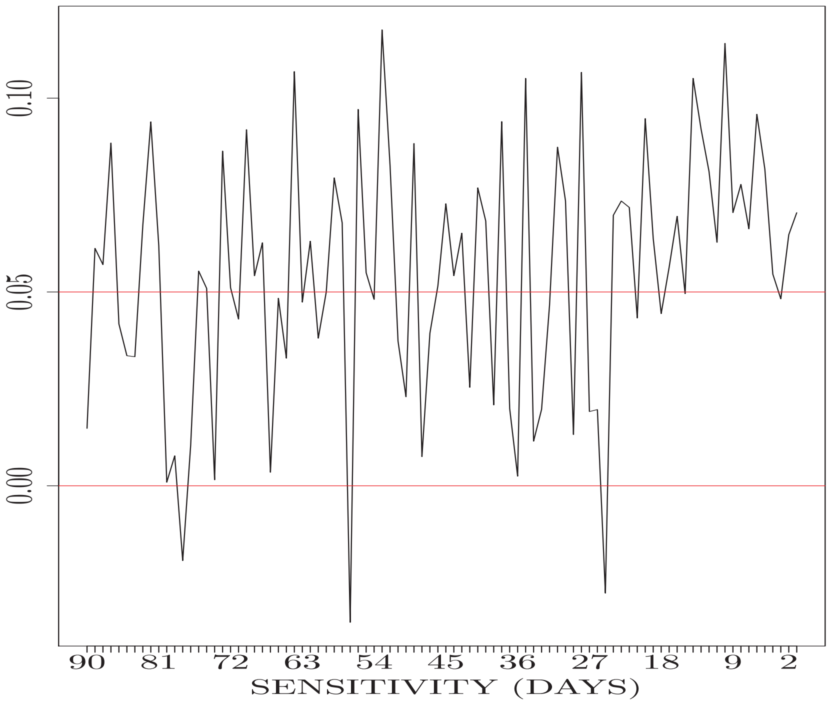


Figure IV.130: **SP500,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

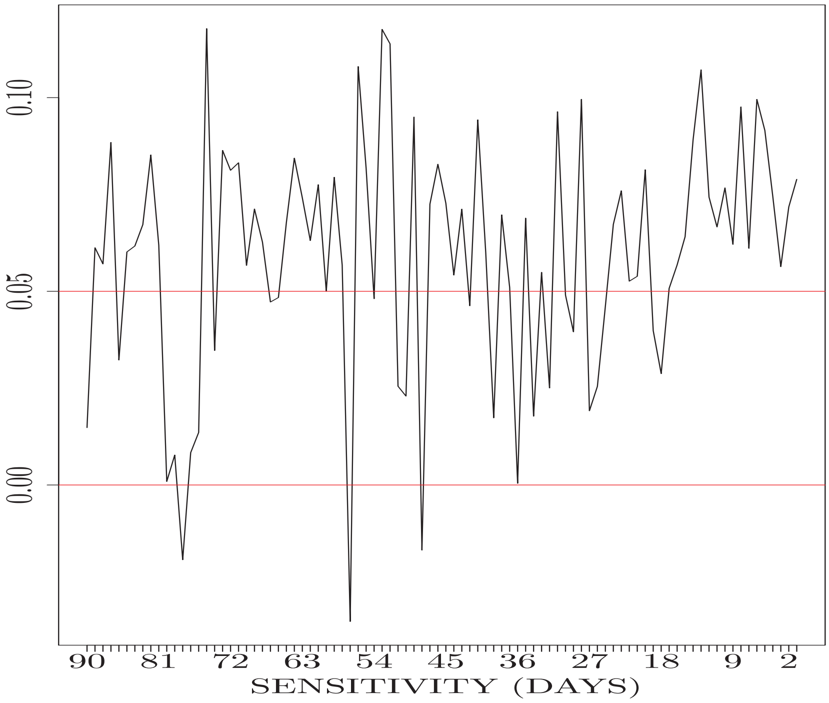


Figure IV.131: **SP500,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

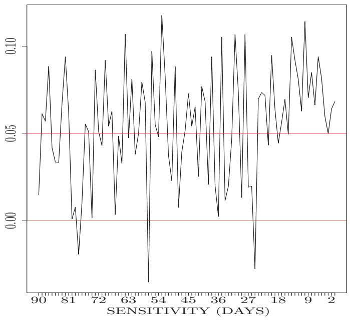


Figure IV.132: $\theta_{buy} = 0.6, \theta_{sell} = 0.4$

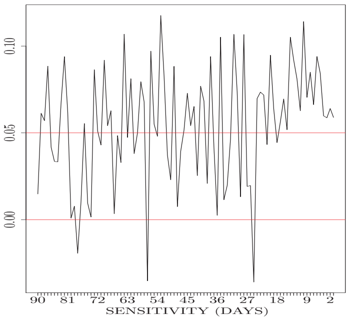


Figure IV.133: $\theta_{buy} = 0.7, \theta_{sell} = 0.3$

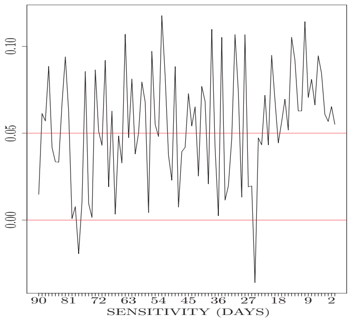


Figure IV.134: $\theta_{buy} = 0.8, \theta_{sell} = 0.2$

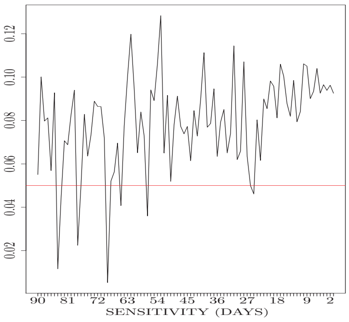


Figure IV.135: $\theta_{buy} = 0.9, \theta_{sell} = 0.1$

Figure IV.136: **SP500,** 1^{st} **DB**

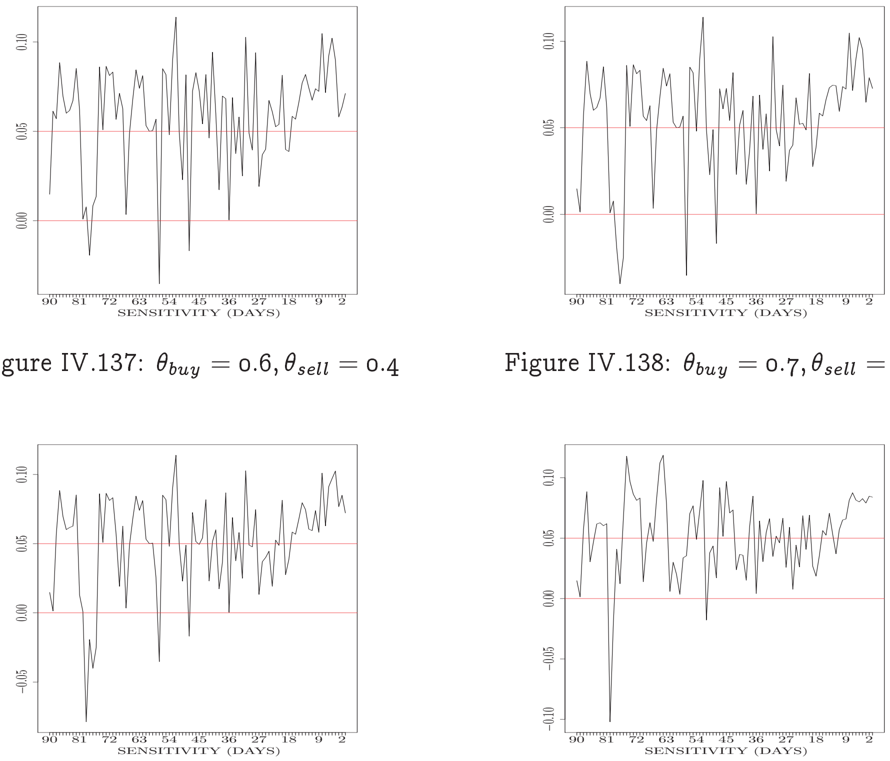


Figure IV.137: $\theta_{buy} = 0.6, \theta_{sell} = 0.4$

Figure IV.138: $\theta_{buy} = 0.7, \theta_{sell} = 0.3$

Figure IV.139: $\theta_{buy} = 0.8, \theta_{sell} = 0.2$

Figure IV.140: $\theta_{buy} = 0.9, \theta_{sell} = 0.1$

Figure IV.141: **SP500, 2^{nd} DB**

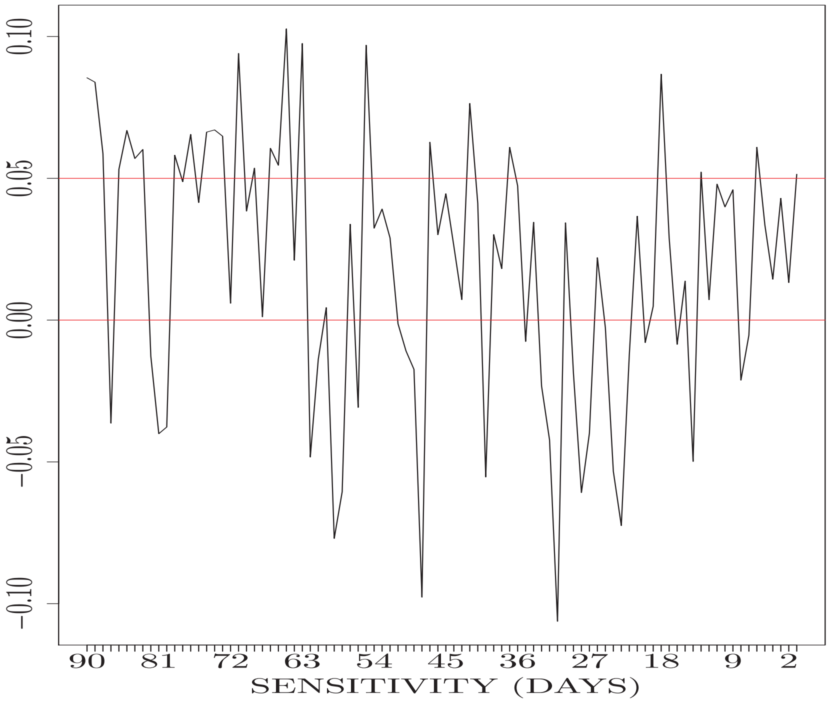


Figure IV.142: **STOXX,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

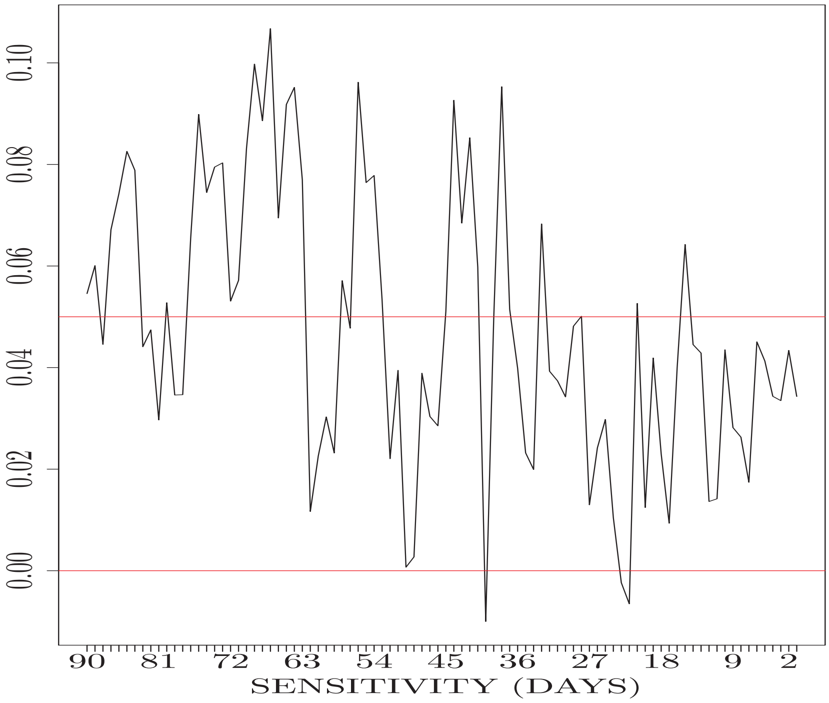


Figure IV.143: **STOXX,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

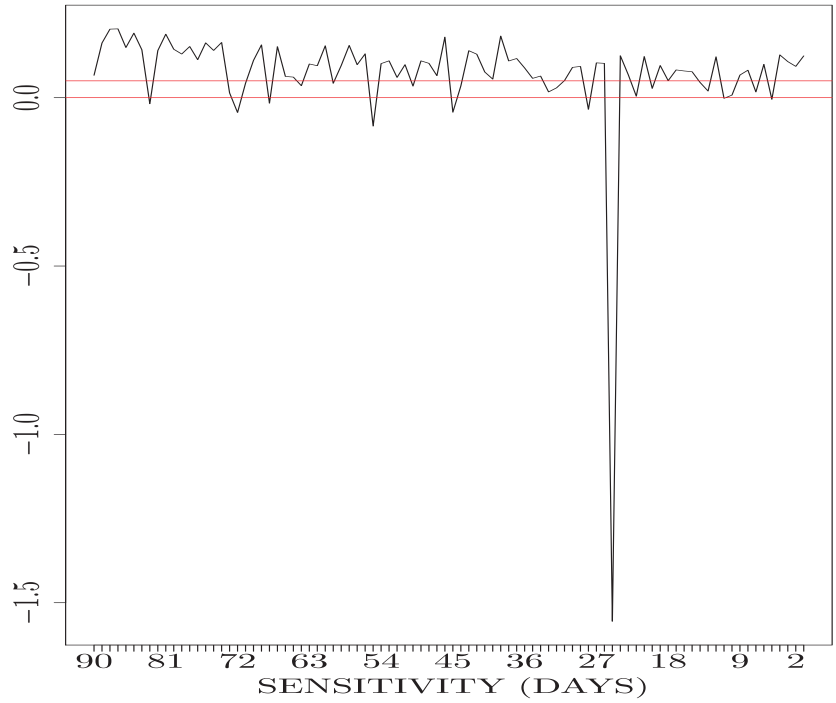


Figure IV.144: **GOLD,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

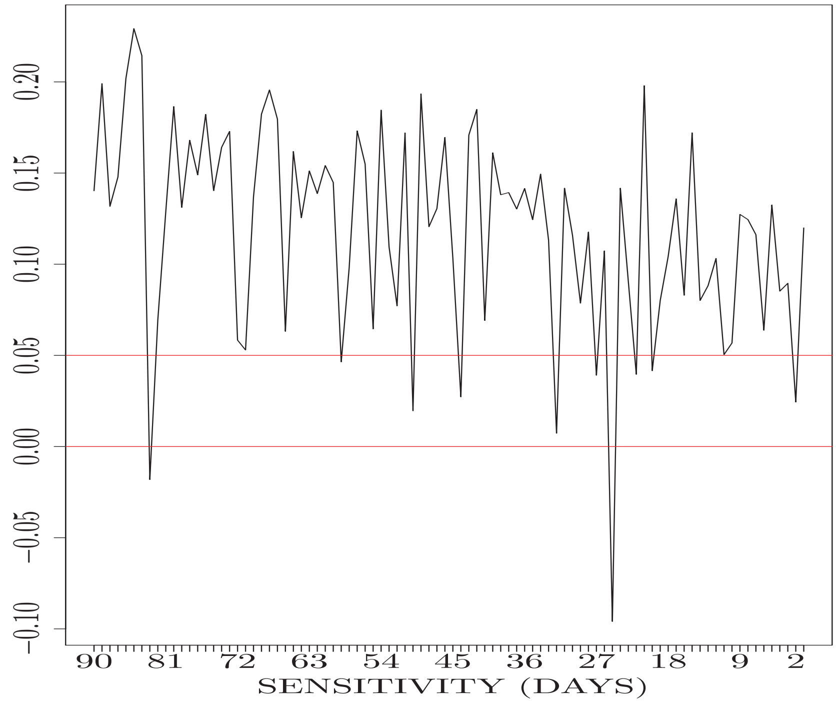


Figure IV.145: **GOLD,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

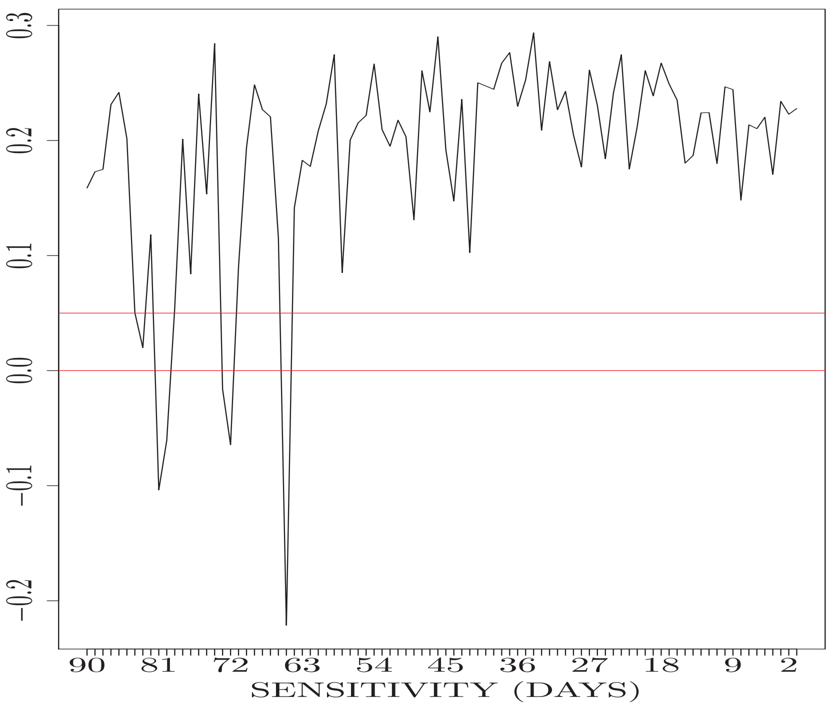


Figure IV.146: **OIL,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

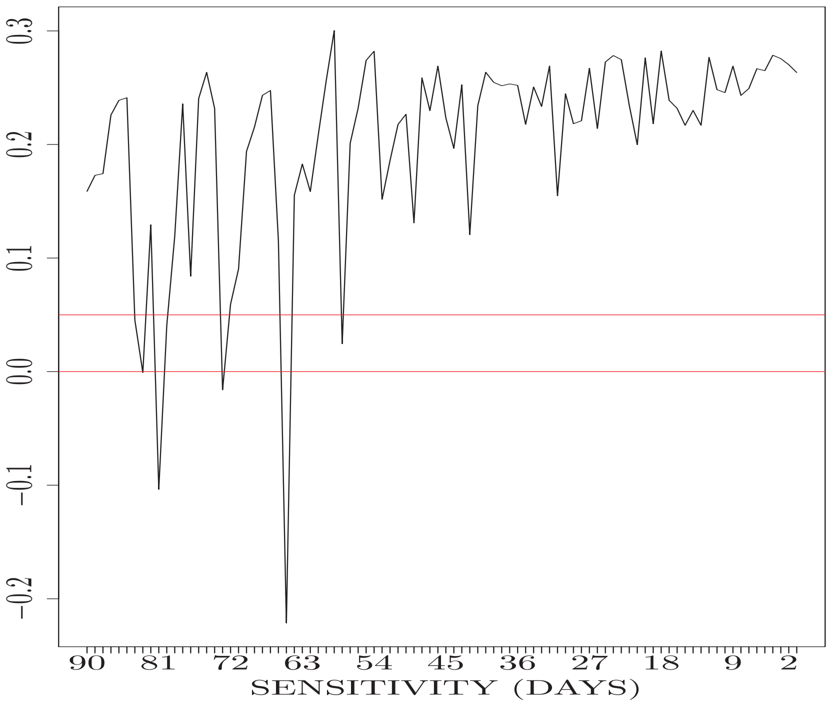


Figure IV.147: **OIL,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

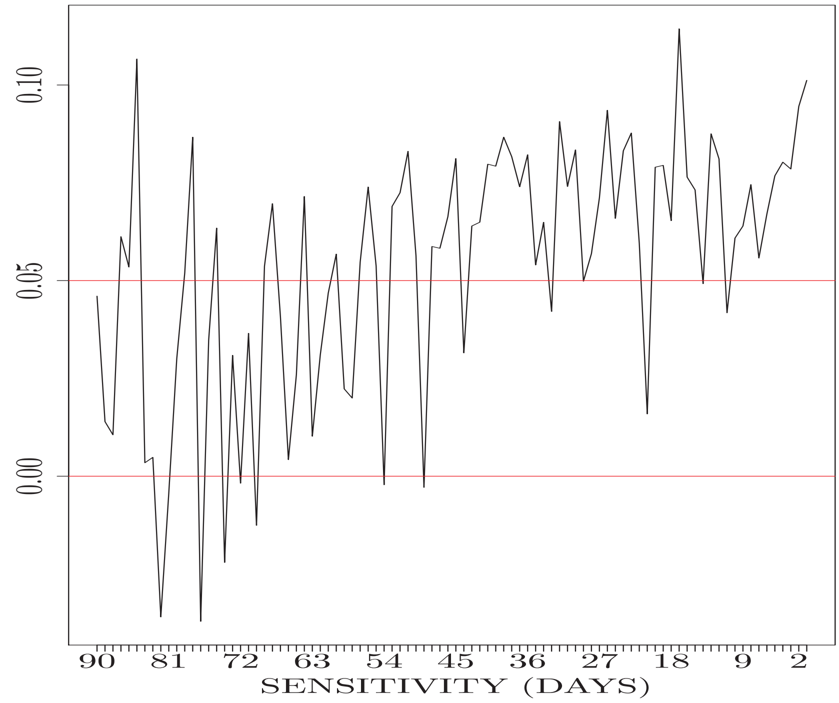


Figure IV.148: **EURUSD,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

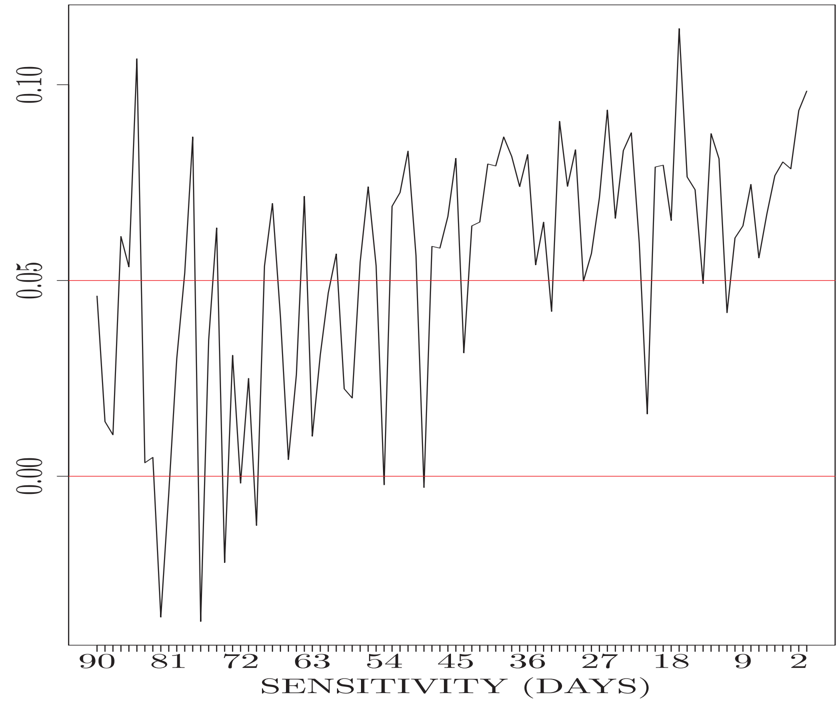


Figure IV.149: **EURUSD,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

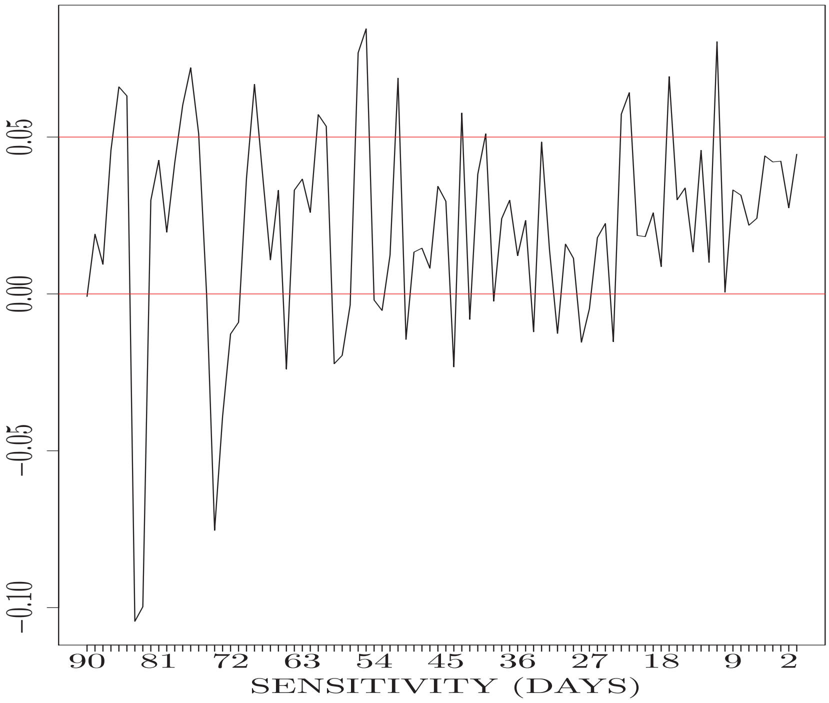


Figure IV.150: **DEXUSUK,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 1^{st} **DB**

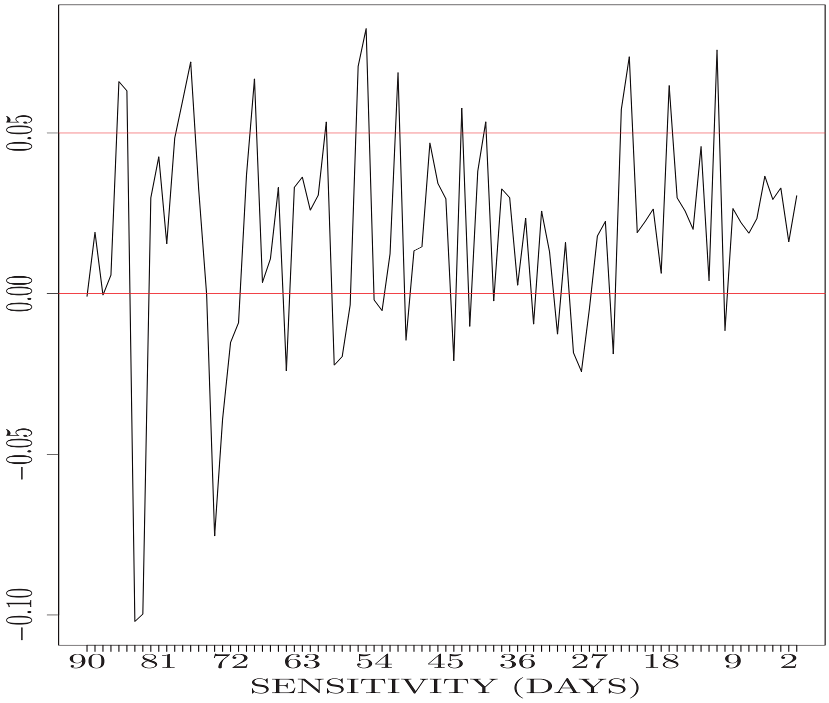


Figure IV.151: **DEXUSUK,** $\theta_{buy} = 0.5, \theta_{sell} = 0.5$, 2^{nd} **DB**

IV.3.2 Improvements for the UniCart RNN Approach

Instability of UniCart RNN numerical models. The subsection IV.3.1 highlights the variance reduction of the annualized P&Ls when the sensitivity decreases, i.e. $Sy \rightarrow 1$. Yet, if the variance of the P&Ls is admittedly reduced for small sensitivity values, it is not necessarily very low. This lack of stability of the UniCart RNN numerical models is addressed here from another point of view. We illustrate the two last steps of the RNN procedure, the relabeling of the training set and the 1-NN classification of the test set, using the first database made for the modeling of the **SP500** index. The graph IV.152 represents thus the original labels of the training set, while the graph IV.153 displays the new labels computed for the training set by the RNN procedure. The scores corresponding to these new labels are plotted on graph IV.154. The new labels, as well as scores, are then transferred via 1-NN classification to the test set, as shown in graphs IV.156 and IV.157. These figures particularly highlight the concentration of the scores around the values 0 and 1. The score clearly identifies broad-scale upward and downward trends, perturbed by transitory variations of large amplitude. These observations are also made on the test set of the **SP500** second database, as underlined by figure IV.28, as well as on the databases of all the other target variables, as emphasized by graphs IV.46, IV.48, IV.66, IV.68, IV.86, IV.88, IV.106, IV.108, IV.126 and IV.128. The new model suggested next paragraph aims at moderating the magnitude of the high-frequency variations of the score.

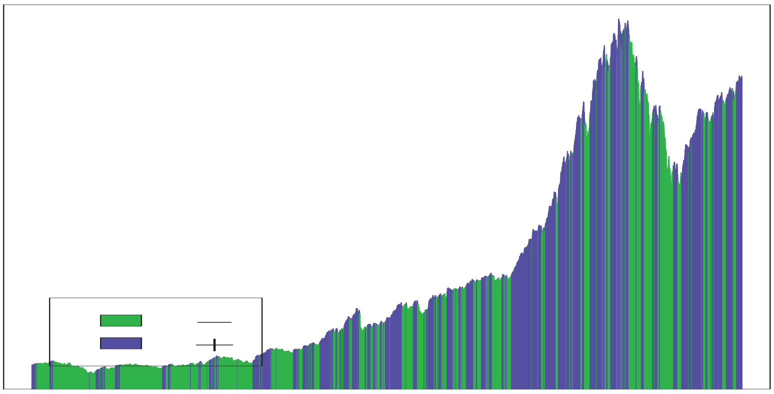


Figure IV.152: **SP500, Trn Labels, 1st DB**

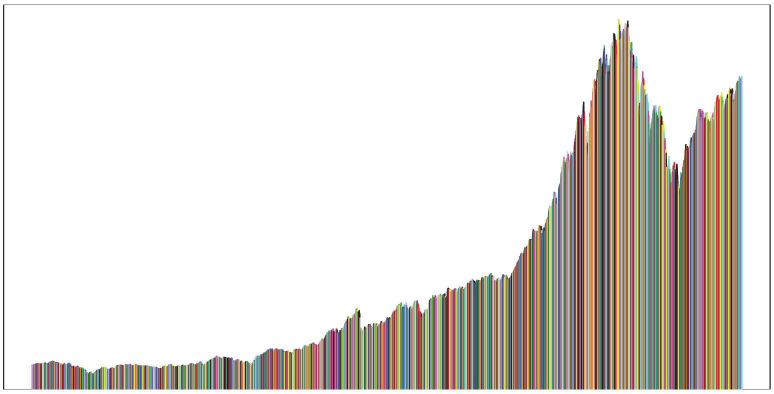


Figure IV.153: **SP500, New Trn Labels, 1st DB**

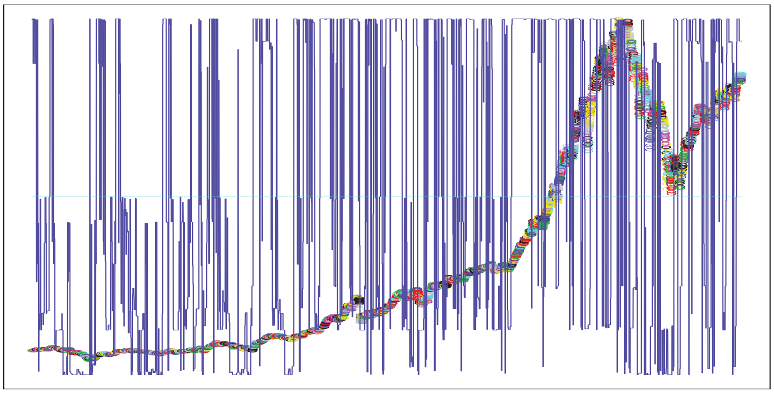


Figure IV.154: **SP500, Trn Scores, 1st DB**

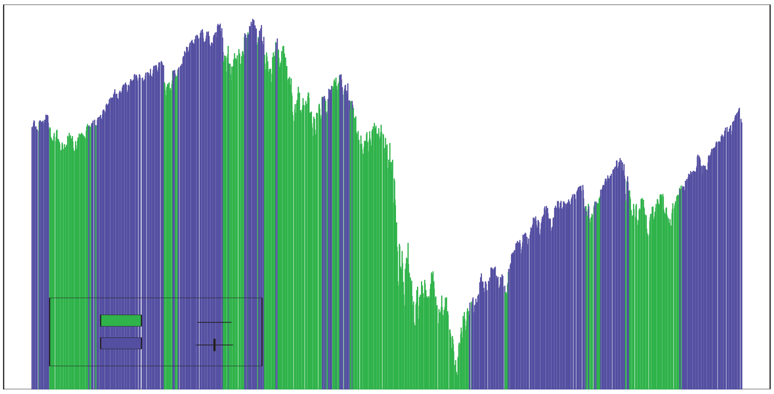


Figure IV.155: **SP500, Test Labels, 1^{st} DB**

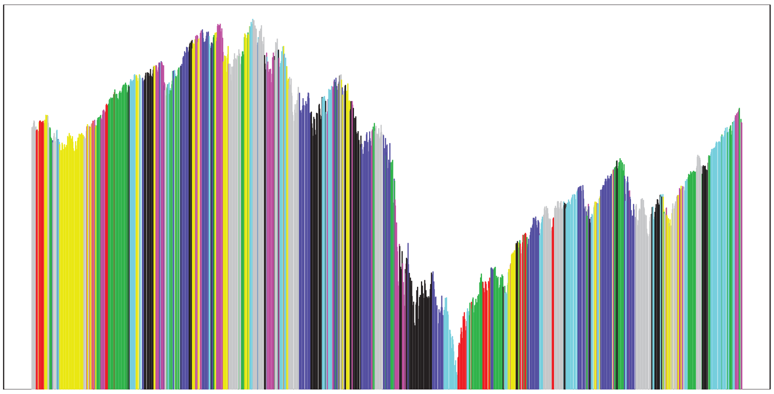


Figure IV.156: **SP500, New Test Labels, 1^{st} DB**

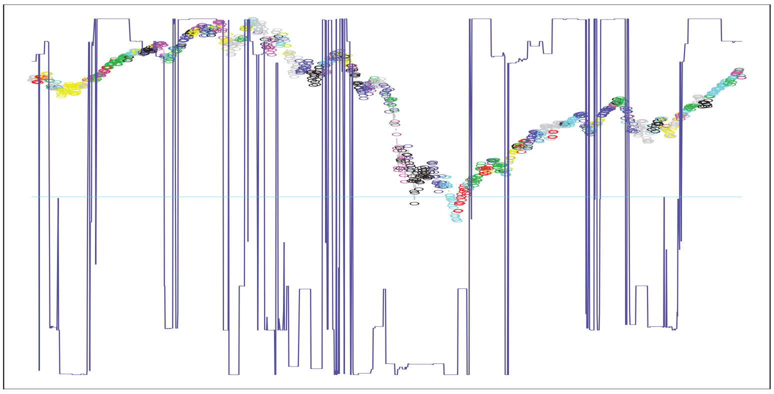


Figure IV.157: **SP500, Test Scores, 1^{st} DB**

Suggestion of a new model. We consider the random process $X \sim P$, where

$$\begin{cases} X = \left(X_\cdot^s\right)_s = \left(X_t^\cdot\right)_t = \left(X_t^s\right)_{t,s} \\ 1 \leq s \leq S \\ t \in \mathbb{N}^\star. \end{cases}$$

We call X a random multiscale process indexed by the time t and by the scale s. We aim at finding the multivariate regression function

$$\begin{cases} f = (f_s)_s \\ 1 \leq s \leq S \end{cases}$$

which leaves the process X (as) invariant (as possible) over various scales s at horizon h. We denote first τ_h the translation operator such that

$$\tau_h\Big(\left(X_\cdot^s\right)_s\Big) = \left(X_{\cdot+h}^s\right)_s.$$

We introduce then the following true and empirical risks.

$$R(f) = \mathbb{E}\bigg(Q\Big(\tau_h\Big(\left(X_\cdot^s\right)_s\Big), f\Big)\bigg) \qquad \text{and} \qquad R_n(f) = \frac{1}{n}\sum_{t=1}^{n} Q\Big(\left(X_{t+h}^s\right)_s, f\Big),$$

where the loss function Q is:

$$Q\Big(\tau_h\Big(\left(X_\cdot^s\right)_s\Big), f\Big) = \sum_{s=1}^{S}\left|f_s\Big(\left(X_\cdot^s\right)_s\Big) - \tau_h\Big(\left(X_\cdot^s\right)_s\Big)\right|^2.$$

Suppose that the process X is made of the UniCart trends or returns of a given target financial variable. Consider that $f = (f_s)_s$ is a multivariate regression tree and that each split sp in the tree f is defined by a single scale $\bar{s}$. The splot sp is typically of the type:

$$sp = X^{\bar{s}^{-1}}(]-\infty, a]),$$

where a is a real threshold. Our problem is then a problem of scale selection, in the sense that we aim at selecting the splits, hence the scales, which most reduce the risk. We believe that this problem, by mixing several scales, may lead to more stable trend predictive numerical models.

We propose to study a RNN-like formulation of this problem. We introduce the following mixture model, inspired by the model stated in section III.3. We consider the random pair process $(X, R) \sim P$, where X is a random multiscale process and where $R = \left(R_t\right)_t \in \{1, \ldots, N_R\}$ is a latent process standing for a specific distribution of the random vector $X_t^\cdot = (X_t^s)_{1 \leq s \leq S}$. The distribution of the random process X is given by the mixture model

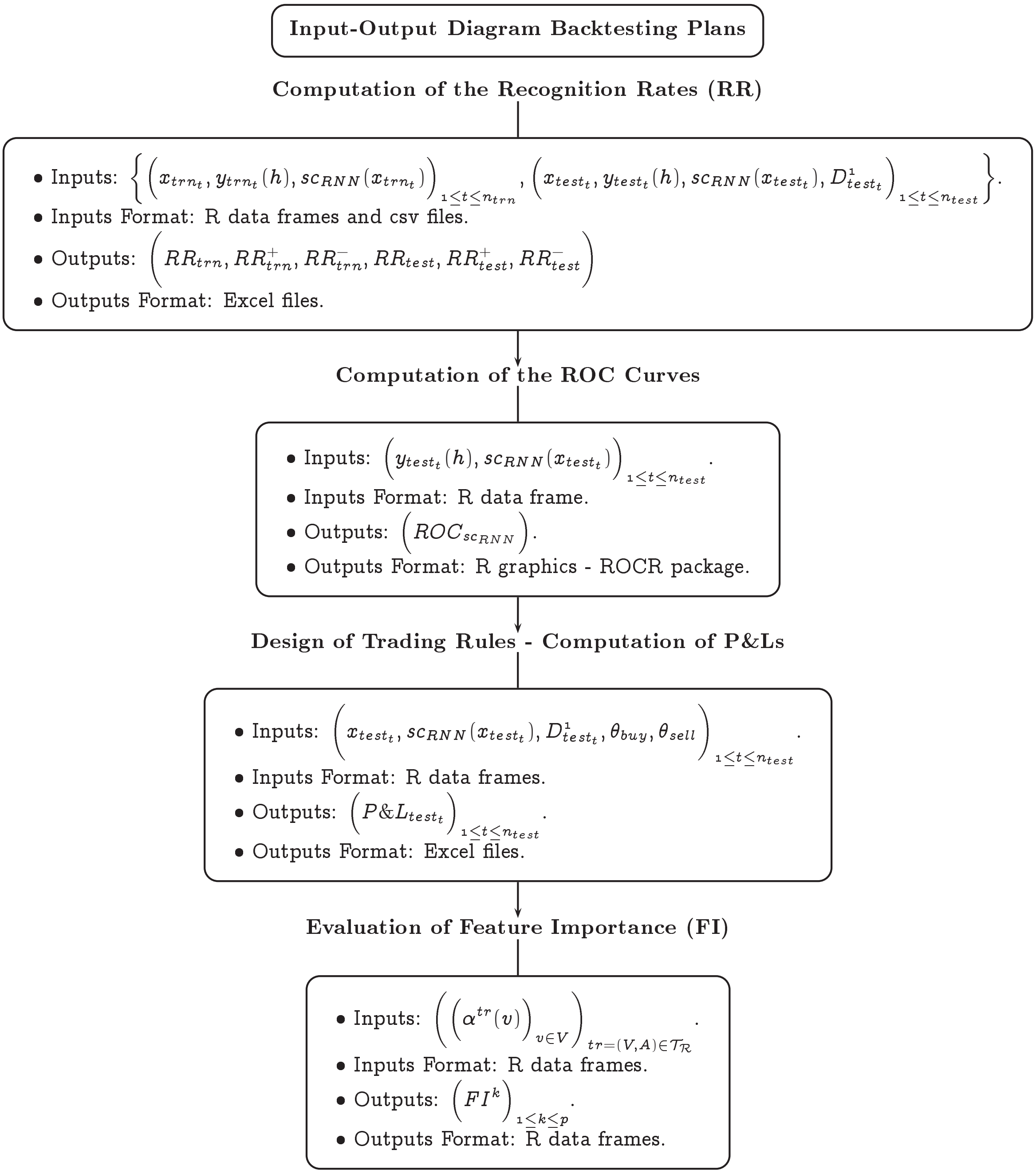
$$X_t^\cdot \sim \sum_{r=1}^{N_R} c_{t,r} P_{t,r},$$

where $c_{t,r} = P(R_t = r)$ is a weight and where $P_{t,r} = P_{X_t^\cdot | R_t = r}$ is a regime. Each regime $\left(P_{t,r}\right)_{1 \leq r \leq N_R}$ is unknown and the number N_R of regimes is also unknown. The goal is then to find a number of regimes N_R, a regime attribution function $H : \mathcal{X} \longrightarrow \{1, \ldots, N_R\}$ and regime specific prediction functions $\left(f_{r,s}\right)_{r,s}$, where $1 \leq r \leq N_R$, $1 \leq s \leq S$ and $f_{r,s} : \mathcal{X} \longrightarrow \mathbb{R}$, so that the numerical model defined by the triplet $\Big(N_R, H, \left(f_{r,s}\right)_{r,s}\Big)$ has a low prediction error. The UniCart RNN approach can not be directly applied to find and to transfer the regimes. Particularly, the problem considered here is a combination of regression problems and not a binary classification problem. Yet, slight changes in the RNN procedure may be sufficient to study this new problem.

IV.3.3 Input-Output Diagram of the Backtesting Plans

The following diagram sums up the steps of backtesting plans, implemented with R and Excel. The target variable is denoted $(D^1_{test_t})_{1\leq t\leq n_{test}}$ and is required to compute the P&Ls of the trading rules. Besides, recognition rates are computed by adopting the convention

$$\begin{cases} sc_{RNN}(x) \geq \frac{1}{2} \Rightarrow "+" \\ sc_{RNN}(x) < \frac{1}{2} \Rightarrow "-". \end{cases}$$



Input-Output Diagram Backtesting Plans

Computation of the Recognition Rates (RR)

- Inputs: $\left\{\left(x_{trn_t}, y_{trn_t}(h), sc_{RNN}(x_{trn_t})\right)_{1\leq t\leq n_{trn}}, \left(x_{test_t}, y_{test_t}(h), sc_{RNN}(x_{test_t}), D^1_{test_t}\right)_{1\leq t\leq n_{test}}\right\}$.
- Inputs Format: R data frames and csv files.
- Outputs: $\left(RR_{trn}, RR^+_{trn}, RR^-_{trn}, RR_{test}, RR^+_{test}, RR^-_{test}\right)$
- Outputs Format: Excel files.

Computation of the ROC Curves

- Inputs: $\left(y_{test_t}(h), sc_{RNN}(x_{test_t})\right)_{1\leq t\leq n_{test}}$.
- Inputs Format: R data frame.
- Outputs: $\left(ROC_{sc_{RNN}}\right)$.
- Outputs Format: R graphics - ROCR package.

Design of Trading Rules - Computation of P&Ls

- Inputs: $\left(x_{test_t}, sc_{RNN}(x_{test_t}), D^1_{test_t}, \theta_{buy}, \theta_{sell}\right)_{1\leq t\leq n_{test}}$.
- Inputs Format: R data frames.
- Outputs: $\left(P\&L_{test_t}\right)_{1\leq t\leq n_{test}}$.
- Outputs Format: Excel files.

Evaluation of Feature Importance (FI)

- Inputs: $\left(\left(\alpha^{tr}(v)\right)_{v\in V}\right)_{tr=(V,A)\in\mathcal{T}_{\mathcal{R}}}$.
- Inputs Format: R data frames.
- Outputs: $\left(FI^k\right)_{1\leq k\leq p}$.
- Outputs Format: R data frames.

Appendix

Supervised Boosted Nearest Neighbors

As indicated in the introduction of chapter III, this subsection presents the Supervised Boosted Nearest Neighbors (SBNN) procedure. Like the RNN procedure, it aims at discovering and learning different regimes in the binary classification setup. It is based on the new concepts of clusterizer and local classifier. We call clusterizer any distance-based function indicating whether two objects of the database are closed from each other. Given an object of the database, any clusterizer defines thus a subset of the training set. We call local classifiers the classifiers indexed by this subset. The SBNN procedure is a boosting-like algorithm, which aggregates a given ensemble of clusterizers and local classifiers to produce more precise classifiers. In the following, we introduce notations and state the SBNN goal and pseudocode. Computational difficulties are emphasized as well.

Notations. We consider the statistical model introduced in subsection III.3. In the SBNN approach, clusterizers cz are used to identify the different regimes. These functions depend on a given distance $Dist$ and on a given number $I \in \mathbb{N}$ of nearest neighbors. They are defined as follows.

$$cz = \left\{ \begin{array}{lll} \mathcal{X} \times x_{trn_{1:n_{trn}}} & \longrightarrow & \{-1, +1\} \\ (x, x_{trn_j}) & \longmapsto & 1 - 2\mathbb{1}_{\left(x_{trn_j} \in NN(x)\right)} \end{array} \right.$$

where

$$NN(x) := NN_I^{Dist}(x)$$

is the set of the I nearest neighbors of the object x in the training set $x_{trn_{1:n_{trn}}}$ according to the distance $Dist$. In the following, we suppose that the clusterizers cz belongs to the finite ensemble $\mathcal{C}$ of clusterizers, which corresponds to a family of $|\mathcal{C}|$ distances. In addition, for more clarity, usual notations are simplified and we have: $x_j := x_{trn_j}$ and $n := n_{trn}$. The local classifiers, corresponding to the object x, are then denoted

$$(f_j)_{x_j \in NN(x)}.$$

We assume finally that the local classifiers $(f_j)_{1 \leq j \leq n}$ belong to the class of functions $\mathcal{F}$.

Goal of the SBNN Procedure and Pseudocode. The idea is to use the boosting philosophy to aggregate clusterizers $(cz_\iota)_{1 \leq \iota \leq \iota_{max}}$ and local classifiers $(f_\iota)_{1 \leq \iota \leq \iota_{max}}$, where $f_\iota = (f_{\iota,j})_{1 \leq j \leq n}$. For each object x_i, we denote by $\psi_{\iota,j}$ the discrete distribution on the objects of the training set at step ι. We consider the weighted empirical error R_n^ι of base clusterizer cz and of local classifiers $(f_j)_{1 \leq j \leq n}$.

$$R_n^\iota(cz, f) = \sum_{i=1}^n \sum_{j=1}^n \psi_{\iota,j}(i) \mathbb{1}_{\{cz(x_i,x_j)=-1, f_j(x_i) \neq y_i\}}.$$

We aim at minimizing the following smoother bound of the weighted empirical error:

$$A_n^\iota(cz, f) = \sum_{i=1}^n \sum_{j=1}^n \psi_{\iota,j}(i) \exp\left(-w_\iota \left(\frac{1 - cz_\iota(x_i, x_j)}{2} f_{\iota,j}(x_i) y_i + \frac{1 + cz_\iota(x_i, x_j)}{2} \right) \right).$$

The idea is to diminish the influence of the object x_j on x_i during the training phase as soon as:

- either x_j is not a neighbor of x_i.
- or the classifier f_{ι,x_j} did predict correctly the label of x_i, i.e. $f_{\iota,j}(x_i) = y_i$.

Conversely, the influence of x_j on x_i is increased if:

- x_j is a neighbor of x_i
- and the classifier f_{ι,x_j} did not predict correctly the label of x_i, i.e. $f_{\iota,j}(x_i) \neq y_i$.

This motivates the following SBNN pseudocode.

1. Initialization:
$$\psi_{1,j}(i) = \frac{1}{n}, \quad \forall i, j \in \{1, \dots, n\}.$$
2. For $\iota = 1, \dots, \iota_{max}$, execute the following procedure.
    - Choose clusterizer cz_ι and local classifiers $f_{\iota,j}$ approximately minimizing R_n^ι over all $cz \in \mathcal{C}$ and $(f_j)_{1 \leq j \leq n} \subset \mathcal{F}$.
    - Set $R_n^{\iota,\star} = R_n^\iota(cz_\iota, f_\iota)$ and adjust the weight w_ι:
    $$w_\iota = \frac{1}{2} \ln \left(\frac{n - R_n^{\iota,\star}}{R_n^{\iota,\star}} \right).$$
    - Update the distribution vector:
    $$\psi_{\iota+1,j}(i) = \psi_{\iota,j}(i) \exp \left(- w_\iota \left(\frac{1 - cz_\iota(x_i, x_j)}{2} f_{\iota,j}(x_i) y_i + \frac{1 + cz_\iota(x_i, x_j)}{2} \right) \right).$$

    Normalize the updated vector.
3. Outputs:
$$\begin{cases} CZ_{\iota_{max}}(x_i, x_j) & = \sum_{\iota=1}^{\iota_{max}} w_\iota \; cz_\iota(x_i, x_j) \\ F_{\iota_{max}}(x_i, x_j) & = \sum_{\iota=1}^{\iota_{max}} w_\iota \; \frac{1 - cz_\iota(x_i, x_j)}{2} \; f_{\iota,j}(x_i). \end{cases}$$

The SBNN prediction of the object x is computed as follows.

$$\begin{cases} j^\star & = \arg\min_{j \in \{1, \dots, n\}} CZ_{\iota_{max}}(x, x_j) \\ F_{\iota_{max}}(x, x_{j^\star}) & = \sum_{\iota=1}^{\iota_{max}} w_\iota \; \frac{1 - cz_\iota(x_i, x_{j^\star})}{2} \; f_{\iota,j^\star}(x). \end{cases}$$

Computational Difficulties. Selecting at each step ι the best clusterizer $cz_\iota \in \mathcal{C}$ and the corresponding local classifiers $f_\iota \subset \mathcal{F}$ requires to compute the local classifiers of each clusterizer $cz \in \mathcal{C}$. This is computationally highly expensive. To reduce this computation cost, a possible strategy is to reduce the number of local classifiers. This can be done by defining clusterizers via **clus**ters of objects instead of neighborhoods of objects. Clusters are computed using the distance $Dist$ and a user-chosen number of clusters cl. We have then:

$$cz = \begin{cases} \mathcal{X} & \times & x_{trn_{1:n_{trn}}} & \longrightarrow & \{-1, +1\} \\ (x & , & x_j) & \longmapsto & 1 - 2\mathbb{1}_{\left(Clus(x_j) = Clus(x)\right)} \end{cases}$$

where

$$Clus(x) := Clus_{cl}^{Dist}(x)$$

is the integer indicating the cluster of the object x. It naturally checks:

$$1 \leq Clus(x) \leq cl.$$

We denote then $f_{cz,Clus(x)}$ the local classifier, corresponding to the object x and to the clusterizer cz. At each iteration ι of the SBNN procedure, cl local classifiers are trained (instead of n_{trn} local classifiers).

Conclusion. The main drawback of the SBNN procedure is that it requires parameters to be entered, either the number I of nearest neighbors or the number cl of clusters. Besides, the family of distances, which defines the ensemble of clusterizers, has to be chosen carefully as it has a strong influence on output classifiers. We preferred therefore to work with the RNN procedure.

Bibliography

[AAG96] Yali Amit, Geman August, and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588, 1996.

[Ach00] Steven B. Achelis. *Technical Analysis from A to Z.* McGraw-Hill, New York, second edition, 2000.

[ACW97] A Abhyankar, L S Copeland, and W Wong. Uncovering nonlinear structure in real-time stock-market indexes: The s&p 500, the dax, the nikkei 225, and the ftse-100. *Journal of Business & Economic Statistics*, 15(1):1–14, 1997.

[AEP08] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex Multi-Task Feature Learning. *Mach. Learn.*, 73(3):243–272, 2008.

[AHKS06] Amit Agarwal, Elad Hazan, Satyen Kale, and Robert E. Schapire. Algorithms for portfolio management based on the newton method. In *Proceedings of the 23rd International Conference on Machine learning*, ICML '06, pages 9–16, New York, NY, USA, 2006. ACM.

[Aka73] Hirotugu Akaike. *Information Theory and an Extension of the Maximum Likelihood Principle*, volume 1, pages 267–281. Akademiai Kiado, 1973.

[AMPY07] Andreas Argyriou, Charles A. Micchelli, Massimiliano Pontil, and Yiming Ying. A spectral regularization framework for multi-task structure learning. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *NIPS*. MIT Press, 2007.

[AN10] Terrence M. Adams and Andrew B. Nobel. Uniform convergence of vapnik-chervonenkis classes under ergodic sampling. *The Annals of Probability*, 38(4):1345–1367, 2010.

[And73] M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, New York, 1973.

[AP03] Peter A. Ammermann and Douglas M. Patterson. The cross-sectional and cross-temporal universality of nonlinear serial dependencies: Evidence from world stock indices and the taiwan stock exchange. *Pacific-Basin Finance Journal*, 11(2):175–195, 2003.

[Aro06] David R. Aronson. *Evidence-Based Technical Analysis: Applying the Scientific Method and Statistical Inference to Trading Signals*. Wiley, Hoboken, NJ, 2006.

[Aud] Jean-Yves Audibert. *PAC-Bayesian Aggregation and Multi-Armed Bandits*. Habilitation Thesis.

[Bac00] Louis Bachelier. Théorie de la spéculation. *Annales Scientifiques de l'Ecole Normale Supérieure*, 3(17), 1900.

[Bax00] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[BB02] Thierry Béchu and Eric Bertrand. *L'Analyse Technique, Pratique et Méthodes*. Economica, 5th edition, 2002.

[BBL02] Peter L. Bartlett, Stéphane Boucheron, and Gábor Lugosi. Model selection and error estimation. *Machine Learning*, 48(1-3):85–113, 2002.

[BBL04] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to Statistical Learning Theory. Springer Series in Statistics, pages 169–207. Springer, 2004.

[BBL05] Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of Classification: a Survey of Some Recent Advances. *ESAIM: P&S*, 9:323–375, 2005.

[BBS07] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 81–88, New York, NY, USA, 2007. ACM.

[BCW08] Edwin V. Bonilla, Kian M. Chai, and Christopher K. I. Williams. Multi-task gaussian process prediction. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA, 2008.

[BD10] Gérard Biau and Luc Devroye. On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification. *J. Multivar. Anal.*, 101:2499–2518, 2010.

[BDBCP07] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of Representations for Domain Adaptation. In *NIPS*, 2007.

[BDL08] Gérard Biau, Luc Devroye, and Gábor Lugosi. Consistency of random forests and other averaging classifiers. *J. Mach. Learn. Res.*, 9:2015–2033, 2008.

[BDP07] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, Bollywood, Boomboxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[BDS03] Shai Ben-David and Reba Schuller. Exploiting task relatedness for mulitple task learning. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *COLT*, volume 2777 of *Lecture Notes in Computer Science*, pages 567–580. Springer, 2003.

[BDvLP06] Shai Ben-David, Ulrike von Luxburg, and Dávid Pál. A Sober Look at Clustering Stability. In *Learning Theory*, volume 4005 of *Lecture Notes in Computer Science*, chapter 4, pages 5–19. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2006.

[Ber27] Sergei N. Bernstein. Sur lŠextension du théorème limite du calcul des probabilités aux sommes de quantités dépendantes. *Mathematische Annalen*, 97:1–59, 1927.

[BET04] I. Johnstone B. Efron, T. Hastie and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–451, 2004.

[BEW07] R. Tishirani B. Efron, J. Taylor and G. Walther. Forward stagewise regression and the monotone lasso. *Electron. J. Statist.*, 1:1–29, 2007.

[BEYG00] Allan Borodin, Ran El-Yaniv, and Vincent Gogan. On the competitive theory and practice of portfolio selection. In *Proceedings of the 4th Latin American Symposium on Theoretical Informatics*, LATIN '00, pages 173–196, London, UK, 2000. Springer-Verlag.

[BFOS84] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth and Brooks, Monterey, CA, 1984.

[BG04] Benoît Bellone and Erwan Gautier. Predicting economic downturns through a financial qualitative hidden markov model. http://bellone.ensae.net/bellonepaper.html, 2004.

[BG07] Irad Ben-Gal. *Bayesian Networks.* John Wiley and Sons, 2007.

[BH] Ramaprasad Bhar and Shigeyuki Hamori.

[Bia] Gérard Biau. Analysis of a Random Forests Model.

[Big93] Norman Biggs. *Algebraic Graph Theory.* Cambridge University Press, 1993.

[BJM06] Peter L. Bartlett, Michael I. Jordan, and Jon D. Mcauliffe. Convexity, Classification, and Risk Bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[BK86a] David S. Broomhead and Gregory P. King. Extracting qualitative dynamics from experimental data. *Physica D: Nonlinear Phenomena*, 20(2-3):217 – 236, 1986.

[BK86b] David S. Broomhead and Gregory P. King. On the Qualitative Analysis of Experimental Dynamical Systems. In S. Sarkar, editor, *Nonlinear Phenomena and Chaos*, pages 113–144. Adam Hilger, Bristol, England, 1986.

[BK98] X. Boyen and D. Koller. Tractable Inference for Complex Stochastic Processes. In *Uncertainty in Artificial Intelligence*, pages 32–42, Madison, Wisconsin, 1998.

[BLL92] William Brock, Josef Lakonishok, and Blake LeBaron. Simple Technical Trading Rules and the Stochastic Properties of Stock Returns. *The Journal of Finance*, 47(5):1731–1764, 1992.

[BM46] Arthur F. Burns and Wesley C. Mitchell. *Measuring Business Cycles.* National Bureau of Economic Research, Inc, 1946.

[BM03] Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2003.

[BMA10] Richard Brealey, Stewart Myers, and Franklin Allen. *Principles of Corporate Finance.* McGraw-Hill/Irwin, 10th edition, 2010.

[BMP06] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia, 2006. Association for Computational Linguistics.

[BNI98] Andrew Blake, Ben North, and Michael Isard. Learning Multi-Class Dynamics. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *NIPS*, pages 389–395. The MIT Press, 1998.

[Bol67] Arthur Hamilton Bolton. *Money and Investment Profits.* Dow Jones Irwin, Inc., Homewood, Ill, 1967.

[Bol86] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.

[Bol98] Bela Bollobas. *Modern Graph Theory.* Springer, 1998.

[Bou02] Jean-Philippe Bouchaud. An Introduction to Statistical Finance. *Physica A: Statistical Mechanics and its Applications*, 313(1-2):238–251, 2002.

[Bou08] Jean-Philippe Bouchaud. Economics Needs a Scientific Revolution. *Nature*, 455(7217):1181, 2008.

[BP70] G. E. P. Box and David A. Pierce. Distribution of Residual Autocorrelations in Autoregressive-Integrated Moving Average Time Series Models. *Journal of the American Statistical Association*, 65(332):1509–1526, 1970.

[BP04] Jean-Philippe Bouchaud and Marc Potters. *Theory of Financial Risk and Derivative Pricing: From Statistical Physics to Risk Management*. Cambridge University Press, 2004.

[Bra05] Richard C. Bradley. Basic properties of strong mixing conditions. a survey and some open questions. *Probability Surveys*, 2:107, 2005.

[Bre96] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[Bre00] Leo Breiman. Some infinite theory for predictor ensembles, 2000.

[Bre01] Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[BS73] Fischer Black and Myron S. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–54, 1973.

[BS06] A. Buja and W. Stuetzle. Observations on bagging. *Statistica Sinica*, 16:323–352, 2006.

[BT07] Peter L. Bartlett and Mikhail Traskin. Adaboost is consistent. *Journal of Machine Learning Research*, 8:2347–2368, 2007.

[BvdG11] Peter Bühlmann and Sara van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications.* Springer Series in Statistics. Springer, 2011.

[BvL09] Sébastien Bubeck and Ulrike von Luxburg. Nearest neighbor clustering: a baseline method for consistent clustering with arbitrary objective functions. *Journal of Machine Learning Research*, 10:657–698, 2009.

[BY02] Peter Bühlmann and Bin Yu. Analyzing Bagging. *Annals of Statistics*, 30:927–961, 2002.

[Cap02] Enrico Capobianco. Multiresolution Approximation for Volatility Processes. *Quantitative Finance*, 2:91–110, 2002.

[Car97] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[Cat03] Olivier Catoni. A pac-bayesian approach to adaptive classification. Technical report, Universités Paris 6 and Paris 7, 2003.

[Cat07] Olivier Catoni. *PAC-Bayesian Supervised Classification: the Thermodynamics of Statistical Learning*, volume 56 of *Lecture Notes-Monograph Series.* IMS, 2007.

[CB08] Nicolas Chapados and Yoshua Bengio. Training graphs of learning modules for sequential data. *ACM Transactions on Knowledge Discovery from Data*, 2008.

[CBL06] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, New York, NY, USA, 2006.

[CC08] S. Gelper C. Croux. Least angle regression for time series forecasting with many predictors. *K.U.Leuven*, pages 1–36, 2008.

[CDS01] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Rev.*, 43:129–159, 2001.

[CDV11] Stéphan Clémençon, Marine Depecker, and Nicolas Vayatis. Adaptive partitioning schemes for bipartite ranking. *Machine Learning*, 83:31–69, April 2011.

[cFlCLmN07] Tak chung Fu, Fu lai Chung, Robert Luk, and Chak man Ng. Stock time series pattern matching: Template-based vs. rule-based approaches. *Engineering Applications of Artificial Intelligence*, 20:347 – 364, 2007.

[cFlCLmN08] Tak chung Fu, Fu lai Chung, Robert Luk, and Chak man Ng. Representing financial time series based on data point importance. *Engineering Applications of Artificial Intelligence*, 21:277 – 300, 2008.

[cFlCmN06] Tak chung Fu, Fu lai Chung, and Chak man Ng. Financial time series segmentation based on specialized binary tree representation. In *DMIN*, pages 3–9, 2006.

[cFlCyKmN08] Tak chung Fu, Fu lai Chung, Ka yan Kwok, and Chak man Ng. Stock time series visualization based on data point importance. *Eng. Appl. Artif. Intell.*, 21:1217–1232, 2008.

[CG01] Gordon F. Royle Chris Godsil. *Algebraic Graph Theory.* Springer, 2001.

[CG02] Lijuan Cao and Qingming Gu. Dynamic support vector machines for non-stationary time series forecasting. *Intelligent Data Analysis*, 6(1):67–83, 2002.

[CG06] Michael Cooper and Huseyin Gulen. Is time-series-based predictability evident in real time? *Journal of Business*, 79(3):1263–1292, 2006.

[Cha98] Marcelle Chauvet. An econometric characterization of business cycle dynamics with factor structure and regime switching. *International Economic Review*, 39(4):969–96, 1998.

[Cha09] Nicolas Chapados. *Sequential Machine Learning Approaches for Portfolio Management.* Dissertation, Université de Montréal, 2009.

[Che96] Lin Chen. Stochastic Mean and Stochastic Volatility – A Three-Factor Model of the Term Structure of Interest Rates and Its Application to the Pricing of Interest Rate Derivatives. 1996.

[CIR85] John C. Cox, Jr Ingersoll, Jonathan E., and Stephen A. Ross. A theory of the term structure of interest rates. *Econometrica*, 53(2):385–407, 1985.

[CKMP02] Kaushik Chakrabarti, Eamonn J. Keogh, Sharad Mehrotra, and Michael J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Trans. Database Syst.*, 27(2):188–228, 2002.

[CLDS99] Robert G. Cowell, Steffen L. Lauritzen, A. Philip David, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1999.

[CM08] Stephan K. Chalup and Andreas Mitschele. Kernel Methods in Finance. pages 655–687. 2008.

[CM10] Gunnar Carlsson and Facundo Mémoli. Characterization, stability and convergence of hierarchical clustering methods. *J. Mach. Learn. Res.*, 11:1425–1470, 2010.

[CMR05] Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[CMW92] Ronald R. Coifman, Yves Meyer, and Mladen V. Wickerhauser. Wavelet analysis and signal processing. In Mary B. Ruskai, Gregory Beylkin, Ronald R. Coifman, Ingrid Daubechies, Stéphane Mallat, Yves Meyer, and Louise Raphael, editors, *Wavelets and Their Applications*, pages 153–178. Jones and Bartlett, Boston, 1992.

[Con01] Rama Cont. Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues. *Quantitative Finance*, 1, 2001.

[Con07] Rama Cont. Volatility Clustering in Financial Markets: Empirical Facts and Agent-Based Models. In Gilles Teyssière and Alan P. Kirman, editors, *Long Memory in Economics*, chapter 10, pages 289–309. Springer Berlin Heidelberg, 2007.

[Cov91] Thomas M. Cover. Universal Portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

[CP10] Marcelle Chauvet and Simon Potter. Business cycle monitoring with structural changes. *International Journal of Forecasting*, 26(4):777–793, 2010.

[CR76] John C. Cox and Stephen A. Ross. The valuation of options for alternative stochastic processes. *Journal of Financial Economics*, 3(1-2):145–166, 1976.

[CS03] Stéphan Clémençon and Skander Slim. Statistical analysis of financial time series under the assumption of local stationarity. THEMA Working Papers 2003-23, THEMA (THéorie Economique, Modélisation et Applications), Université de Cergy-Pontoise, 2003.

[CT08] Rama Cont and Peter Tankov. *Financial Modelling with Jump Processes*. Chapman & Hall/CRC Financial Mathematics Series, 2nd edition, 2008.

[CVD09] Stéphan Clémençon, Nicolas Vayatis, and Marine Depecker. Auc optimization and the two-sample problem. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *NIPS*, pages 360–368. Curran Associates, Inc., 2009.

[CW88] N. H. Chan and C. Z. Wei. Limiting distributions of least squares estimates of unstable autoregressive processes. *Annals of Statistics*, 16:367–401, 1988.

[CW92] Ronald R. Coifman and Mladen V. Wickerhauser. Entropy based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 32:712–718, 1992.

[Dah97] Rainer Dahlhaus. Fitting Time Series Models to Nonstationary Processes. *The Annals of Statistics*, 25(1):1–37, 1997.

[Dau88] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 41:909–996, 1988.

[DGL96] Luc Devroye, Lászlo Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

[Die00] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Mach. Learn.*, 40:139–157, 2000.

[DK01] J. Durbin and S. J. Koopman. *Time Series Analysis by State Space Methods.* Oxford University Press, Oxford, UK, 2001.

[DMvS98] David L. Donoho, Stéphane Mallat, and Rainer von Sachs. Estimating Covariances of Locally Stationary Processes: Rates of Convergence of Best Basis Methods, 1998.

[DNW96] Robert Dittmar, Christopher J Neely, and Paul Weller. Is technical analysis in the foreign exchange market profitable? a genetic programming approach. CEPR Discussion Papers 1480, C.E.P.R. Discussion Papers, 1996.

[Dou94] Paul Doukhan. *Mixing: Properties and Examples (Lecture Notes in Statistics).* Springer, 1 edition, 1994.

[DR96] Francis X. Diebold and Glenn D. Rudebusch. Measuring business cycles: a modern perspective. *The Review of Economics and Statistics*, 78(1):67–77, 1996.

[Dud] R. M. Dudley. The sizes of compact subsets of hilbert space and continuity of gaussian processes.

[Dud99] R. M. Dudley. *Uniform Central Limit Theorems.* 1999.

[DYXY07] Wenyuan Dai, Qiang Yang, Gui R. Xue, and Yong Yu. Boosting for Transfer Learning. In *Proceedings of the 24th International Conference on Machine learning*, ICML '07, pages 193–200, New York, NY, USA, 2007. ACM.

[EMB07] Robert D. Edwards, John Magee, and W.H.C. Bassetti. *Technical Analysis of Stock Trends.* CRC Press, Boca Raton, 9th edition, 2007.

[Eng82] Robert F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica*, 50(4):987–1007, 1982.

[EP04] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 109–117, New York, NY, USA, 2004. ACM.

[Fam65a] Eugene F. Fama. The behavior of stock-market prices. *The Journal of Business*, 38(1):34–105, 1965.

[Fam65b] Eugene F. Fama. Random walks in stock market prices. *Financial Analysts Journal*, 21:55–60, 1965.

[Fam70] Eugene F. Fama. Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, 25(2):383–417, 1970.

[FBvS03] Piotr Fryzlewicz, Sébastien Bellegem, and Rainer von Sachs. Forecasting non-stationary time series by wavelet process modelling. *Annals of the Institute of Statistical Mathematics*, 55(4):737–764, 2003.

[Fea06] Paul Fearnhead. Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16:203–213, 2006.

[FH99] Jerome Friedman and Peter Hall. On bagging and nonlinear estimation. Technical report, Stanford University, 1999.

[FHST10] Tristan Fletcher, Zakria Hussain, and John Shawe-Taylor. Currency Forecasting using Multiple Kernel Learning with Financially Motivated Features. *Networks*, pages 1–6, 2010.

[FHT98] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[FM04] Jerome H. Friedman and Jacqueline J. Meulman. Clustering objects on subsets of attributes. *Journal Of The Royal Statistical Society Series B*, 66(4):815–849, 2004.

[FMJF10] Frank J. Fabozzi, Franco G. Modigliani, Frank J. Jones, and Michael G. Ferri. *Foundations of Financial Markets and Institutions.* Prentice Hall, Boston, Massachusetts, 4th edition, 2010.

[FMR98] Nir Friedman, Kevin P. Murphy, and Stuart J. Russell. Learning the structure of dynamic probabilistic networks. In *UAI'98*, pages 139–147, 1998.

[FNP99] Nir Friedman, Iftach Nachman, and Dana Pe'er. Learning bayesian network structure from massive datasets: The "sparse candidate" algorithm. In Kathryn B. Laskey and Henri Prade, editors, *UAI*, pages 206–215. Morgan Kaufmann, 1999.

[FRD09] Tristan Fletcher, Fabian Redpath, and Joe D'Alessandro. Machine learning in fx carry basket prediction. In S. I. Ao, Len Gelman, David WL Hukins, Andrew Hunter, and A. M. Korsunsky, editors, *Proceedings of the World Congress on Engineering 2009 Vol II, WCE '09, July 1 - 3, 2009, London, U.K.*, Lecture Notes in Engineering and Computer Science, pages 1371–1375. International Association of Engineers, Newswood Limited, 2009.

[Fre95] Yoav Freund. Boosting a weak learning algorithm by majority. *Inf. Comput.*, 121:256–285, 1995.

[FRGMSR00] Fernando Fernandez-Rodriguez, Christian Gonzalez-Martel, and Simon Sosvilla-Rivero. On the profitability of technical trading rules based on artificial neural networks:: Evidence from the madrid stock market. *Economics Letters*, 69(1):89–94, 2000.

[Fri00] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[FS95] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, London, UK, 1995. Springer-Verlag.

[FSJW08] Emily B. Fox, Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. Nonparametric bayesian learning of switching linear dynamical systems. In Koller et al. [KSBB09], pages 457–464.

[FSR06] Piotr Fryzlewicz, Theofanis Sapatinas, and Suhasini Subba Rao. A haar-fisz technique for locally stationary volatility estimation. *Biometrika*, 93(3):687–704, 2006.

[FSX09] Wenjie Fu, Le Song, and Eric P. Xing. Dynamic Mixed Membership Blockmodel for Evolving Networks. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 329–336, New York, NY, USA, 2009. ACM.

[Gay90] Michael E.S. Gayed. *Intermarket Analysis and Investing, Integrating Economic, Fundamental, and Technical Trends.* New York Institute of Finance, New York (N.Y.), 1990.

[GH00] Zoubin Ghahramani and Geoffrey E. Hinton. Variational Learning for Switching State-Space Models. *Neural Computation*, 12:831–864, 2000.

[GH09] Marco Grzegorczyk and Dirk Husmeier. Non-Stationary Continuous Dynamic Bayesian Networks. In *NIPS 2009*, 2009.

[Gha97] Zoubin Ghahramani. Learning dynamic bayesian networks. In C. Lee Giles and Marco Gori, editors, *Summer School on Neural Networks*, volume 1387 of *Lecture Notes in Computer Science*, pages 168–197. Springer, 1997.

[GHFX07] Fan Guo, Steve Hanneke, Wenjie Fu, and Eric P. Xing. Recovering temporally rewiring networks: a model-based approach. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 321–328, New York, NY, USA, 2007. ACM.

[GKE10] Deniz Gencaga, Ercan E. Kuruoglu, and Aysin Ertuzun. Modeling non-gaussian time-varying vector autoregressive processes by particle filtering. *Multidimensional Syst. Signal Process.*, 21:73–85, 2010.

[GNZ01] Nina Golyandina, Vladimir V. Nekrutkin, and Anatoly A. Zhigliavskii. *Analysis of Time Series Structure : SSA and Related Techniques.* Monographs on Statistics and Applied Probability, 90. Chapman & Hall/CRC, 2001.

[Gor99] A. D. Gordon. *Classification.* Monographs in Applied Statistics and Probability. Chapman & Hall / CRC, second edition, 1999.

[Gre98] Seth Greenblatt. Atomic decomposition of financial data. *Computational Economics*, 12(3):275–93, 1998.

[GS66] David M. Green and John A. Swets. *Signal Detection Theory and Psychophysics.* Wiley, New York, 1966.

[GY06a] T. L. Griffiths and A. Yuille. A primer on probabilistic inference. *Trends in Cognitive Sciences*, 10(7), 2006.

[Gyö06b] László Györfi. Nonparametric kernel-based sequential investment strategies. *Mathematical Finance*, 16:2006, 2006.

[HA07] Nguyen Quoc Viet Hung and Duong Tuan Anh. Combining sax and piecewise linear approximation to improve similarity search on financial time series. In *Proceedings of the 2007 International Symposium on Information Technology Convergence*, ISITC '07, pages 58–62, Washington, DC, USA, 2007. IEEE Computer Society.

[Ham89] James D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica*, 57(2):357–84, 1989.

[Ham94] James D. Hamilton. *Time-Series Analysis.* Princeton Univerity Press, Princeton, NJ, 1994.

[Har75] J. Hartigan. *Clustering Algorithms.* John Wiley and Sons, New York, 1975.

[Har11] Campbell R. Harvey. Hypertextual finance glossary. http://www.duke.edu/ charvey/Classes/wpg/bfgloss.htm, 2011.

[Hau95] David Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded vapnik-chervonenkis dimension. *Journal of Combinatorial Theory, Ser. A*, 69(2):217–232, 1995.

[Hec79] James J Heckman. Sample selection bias as a specification error. *Econometrica*, 47(1):153–161, 1979.

[Hes93] Steven L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6(2):327–343, 1993.

[HGC95] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20(3):197–243, 1995.

[HKLW02] Patrick S. Hagan, Deep Kumar, Andrew S. Lesniewski, and Diana E. Woodward. Managing smile risk. *Wilmott Magazine*, 2002.

[Ho98] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20:832–844, 1998.

[Hoe63] Wassily Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[Hos81] J. R. M. Hosking. Fractional Differencing. *Biometrika*, 68(1), 1981.

[HRS94] Andrew Harvey, Esther Ruiz, and Neil Shephard. Multivariate stochastic variance models. *Review of Economic Studies*, 61(2):247–264, 1994.

[HS05] Peter Hall and Richard J. Samworth. Properties of bagged nearest neighbour classifiers. *Journal Of The Royal Statistical Society Series B*, 67(3):363–379, 2005.

[HS09] Elad Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 393–400, New York, NY, USA, 2009. ACM.

[HSG$^+$07] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting Sample Selection Bias by Unlabeled Data. In *Advances in Neural Information Processing Systems 19*, pages 601–608, 2007.

[Hsi91] David A Hsieh. Chaos and nonlinear dynamics: Application to financial markets. *Journal of Finance*, 46(5):1839–1877, 1991.

[HSSW98] David P. Helmbold, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.

[HSZ09] Hossein Hassani, Abdol S. Soofi, and Anatoly A. Zhigljavsky. Predicting Daily Exchange Rate with Singular Spectrum Analysis. *Nonlinear Analysis: Real World Applications*, 2009.

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[Hua11] Shian-Chang Huang. Integrating spectral clustering with wavelet-based kernel partial least square regressions for financial modeling and forecasting. *Applied Mathematics and Computation*, 217(15):6755–6764, 2011.

[HW06a] Shian-Chang Huang and Hsing-Wen Wang. Combining time-scale feature extractions with svms for stock index forecasting. In Irwin King, Jun Wang, Laiwan Chan, and DeLiang L. Wang, editors, *ICONIP (3)*, volume 4234 of *Lecture Notes in Computer Science*, pages 390–399. Springer, 2006.

[HW06b] Shian-Chang Huang and Tung-Kuang Wu. Wavelet-based relevance vector machines for stock index forecasting. In *IJCNN*, pages 603–609. IEEE, 2006.

[HW08] Shian-Chang Huang and Tung-Kuang Wu. Forecasting stock indices with wavelet-based kernel partial least square regressions. In *IJCNN*, pages 1910–1916. IEEE, 2008.

[HW10] Shian-Chang Huang and Tung-Kuang Wu. Integrating recurrent som with wavelet-based kernel partial least square regressions for financial forecasting. *Expert Systems with Applications*, 37(8):5698–5705, 2010.

[IHS06] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 207–216, New York, NY, USA, 2006. ACM.

[IM06] Hal Daumé III and Daniel Marcu. Domain adaptation for statistical classifiers. *J. Artif. Intell. Res. (JAIR)*, 26:101–126, 2006.

[JD88] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[Jeb04] Tony Jebara. Multi-task feature and kernel selection for svms. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, New York, NY, USA, 2004. ACM.

[JGJS99] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[jK03] Kyoung jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

[JPR94] Eric Jacquier, Nicholas G Polson, and Peter E. Rossi. Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics*, 12(4):371–417, 1994.

[JZ07] Jing Jiang and Chengxiang Zhai. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[Kal60] Rudolph Emil Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME, Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[KB61] R. E. Kalman and R. S. Bucy. New Results in Linear Filtering and Prediction Theory. *Transactions of the ASME, Journal of Basic Engineering*, 83(Series D):95–107, 1961.

[KCHP93] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: a survey and novel approach. In *In an Edited Volume, Data mining in Time Series Databases. Published by World Scientific*, pages 1–22, 1993.

[KCHP01] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. An Online Algorithm for Segmenting Time Series. In *Proc. of the IEEE Int. Conf. on Data Mining (ICDM)*, pages 289–296, San Jose, USA, 2001.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques (Adaptive Computation and Machine Learning Series)*. The MIT Press, 1 edition, 2009.

[KK03] Eamonn Keogh and Shruti Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Min. Knowl. Discov.*, 7(4):349–371, 2003.

[KN98] Chang-Jin Kim and Charles R. Nelson. Business cycle turning points, a new coincident index, and tests of duration dependence based on a dynamic factor model with regime switching. *The Review of Economics and Statistics*, 80(2):188–201, 1998.

[KN99] Chang-Jin Kim and Charles R. Nelson. *State-Space Models with Regime Switching*. The MIT Press, Cambridge, 1999.

[Koz96] Werner Kozek. *Matched Weyl-Heisenberg Expansions of Nonstationary Environments*. PhD thesis, NuHAG, University of Vienna, 1996.

[KP04] Vladimir Koltchinskii and Dmitry Panchenko. Rademacher processes and bounding the risk of function learning. *High Dimensional Probability II*, 47:14, 2004.

[KSAX09] Mladen Kolar, Le Song, Amr Ahmed, and Eric P. Xing. Estimating time-varying networks. *Annals of Applied Statistics*, 4:94–123, 2009.

[KSBB09] Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors. *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*. MIT Press, 2009.

[KSX09] Mladen Kolar, Le Song, and Eric Xing. Sparsistent learning of varying-coefficient models with structural changes. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1006–1014. 2009.

[KV09] R.L. Karandikar and M. Vidyasagar. Probably approximately correct learning with beta-mixing input sequences. Technical report, 2009.

[KY95] Myung-Jig Kim and Ji-Sung Yoo. New index of coincident indicators: a multivariate markov switching factor model approach. *Journal of Monetary Economics*, 36(3):607–630, 1995.

[Lau96] Steffen L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[LB78] G. M. Ljung and G. E. P. Box. On a Measure of Lack of Fit in Time Series Models. *Biometrika*, 65(2):297–303, 1978.

[LD08] Ronny Luss and Alexandre D'Aspremont. Predicting abnormal returns from news using text classification. *Arxiv preprint arXiv08092792*, (1994):1–25, 2008.

[LE07] Karamitopoulos Leonidas and Georgios Evangelidis. Current trends in time series representation. In *Proceedings of 11th Panhellenic Conference on Informatics*, pages 217–226, Computer Engineering & Informatics Department of University of Patras, 2007. Greek Computer Society (EPY).

[LeB96] Blake LeBaron. Technical trading rule profitability and foreign exchange intervention. NBER Working Papers 5505, National Bureau of Economic Research, Inc, 1996.

[Lew03] Jonathan Lewellen. Financial management. `http://ocw.mit.edu/courses/sloan-school-of-management/15-414-financial-management-summer-2003/index.htm/`, 2003.

[LGWcs] Adam Krzyzak Laszlo Györfi, Michael Kohler and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. 2002], publisher=Springer New York Inc., series=Springer Series in Statistics,.

[Lin65] John Lintner. The Valuation of Risk Assets and the Selection of Risky Investments in Stock Portfolios and Capital Budgets. *The Review of Economics and Statistics*, 47(1):13–37, 1965.

[LJ06] Yi Lin and Yongho Jeon. Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101:578–590, 2006.

[LKLC03] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[LKS06] A.C. Lozano, S.R. Kulkarni, and R.E. Schapire. Convergence and consistency of regularized boosting algorithms with stationary β-mixing observations. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 819–826. MIT Press, 2006.

[LM90] Andrew W. Lo and A. Craig MacKinlay. Data-snooping biases in tests of financial asset pricing models. *Review of Financial Studies*, 3(3):431–467, 1990.

[LMW00] Andrew W. Lo, Harry Mamaysky, and Jiang Wang. Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance*, 55(4):1705–1765, 2000.

[LN89] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Math. Program.*, 45(3):503–528, 1989.

[LP04] Neil D. Lawrence and John C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, New York, NY, USA, 2004. ACM.

[lsc]

[LSM01] John Langford, Matthias Seeger, and Nimrod Megiddo. An Improved Predictive Accuracy Bound for Averaging Classifiers. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 290–297, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[LV04] Gabor Lugosi and Nicolas Vayatis. On the bayes-risk consistency of regularized boosting methods (with discussion). *Annals of Statistics*, 32(1):30–55, 2004.

[Mac98] David J. C. MacKay. Introduction to monte carlo methods. In *Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models*, pages 175–204, Norwell, MA, USA, 1998. Kluwer Academic Publishers.

[Mal] Stéphane Mallat. Multiresolution approximations and wavelet orthonormal bases in $\mathbf{L}^2(\mathbf{R})$.

[Mal99] Stéphane Mallat. *A Wavelet Tour of Signal Processing, Second Edition (Wavelet Analysis & Its Applications).* Academic Press, 1999.

[Mar52] Harry Markowitz. Portfolio Selection. *The Journal of Finance*, 7(1):77–91, 1952.

[Mar59] Harry Markowitz. *Portfolio Selection: Efficient Diversification of Investments.* New York, London, Sydney: John Wiley & Sons, 1959.

[McA99] David A. McAllester. Pac-bayesian model averaging. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, COLT '99, pages 164–170, New York, NY, USA, 1999. ACM.

[McD89] Colin McDiarmid. On the Method of Bounded Differences. In *Surveys in Combinatorics*, number 141 in London Mathematical Society Lecture Note Series, pages 148–188. Cambridge University Press, 1989.

[ME07] Rogemar S. Mamon and Robert J. Elliott, editors. *Hidden Markov Models in Finance (International Series in Operations Research & Management Science).* Springer, 1 edition, 2007.

[Mei00] Ron Meir. Nonparametric time series prediction through adaptive model selection. *Machine Learning*, 39:5–34, 2000.

[Mei06] Nicolai Meinshausen. Quantile regression forests. *J. Mach. Learn. Res.*, 7:983–999, 2006.

[Men03] Shahar Mendelson. *A Few Notes on Statistical Learning Theory*, pages 1–40. Springer-Verlag New York, Inc., New York, NY, USA, 2003.

[Mer73] Robert C. Merton. Theory of rational option pricing. *Bell Journal of Economics*, 4(1):141–183, 1973.

[Mer92] Robert C. Merton. *Continuous-Time Finance.* Blackwell, Cambridge, Mass. [u.a.], rev. and 1. publ. in paperb. edition, 1992.

[Meu09] Attilio Meucci. Review of Discrete and Continuous Processes in Finance: Theory and Applications. *Social Science Research Network Working Paper Series*, 2009.

[Meu10] Attilio Meucci. Review of statistical arbitrage, cointegration, and multivariate ornstein-uhlenbeck. *Review Literature And Arts Of The Americas*, 2010.

[Meu11] Attilio Meucci. 'p' versus 'q': Differences and commonalities between the two areas of quantitative finance. *GARP Risk Professional*, pages 47–50, 2011.

[Mey87] Yves Meyer. Principe d'incertitude, bases hilbertiennes et algèbres d'opérateurs. In *Sémin. Bourbaki, 38ème année, Vol. 1985/86*, volume 145/146 of *Exp. Astérisque*, pages 209–223. 1987.

[MM00] Virginia B. Morris and Kenneth M. Morris. *Dictionary of Financial Terms (Lightbulb Press).* McGraw-Hill Companies, 2000.

[MMBWon] Robert F. Meigs, Mary A. Meigs, Mark Bettner, and Ray Whittington. *Financial Accounting.* Richard D Irwin, 1997, 9th edition.

[MMZ02] Shie Mannor, Ron Meir, and Tong Zhang. Greedy algorithms for classification - consistency, convergence rates, and adaptivity. *Journal of Machine Learning Research*, 4:2003, 2002.

[MPR05] Eric Moulines, Pierre Priouret, and Francois Roueff. On recursive estimation for time varying autoregressive processes. *ANNALS OF STATISTICS*, 33:2610, 2005.

[MPZ98] Stéphane Mallat, George Papanicolaou, and Zhifeng Zhang. Adaptive Covariance Estimation of Locally Stationary Processes. *The Annals of Statistics*, 26(1):1–47, 1998.

[MR08] Mehryar Mohri and Afshin Rostamizadeh. Rademacher complexity bounds for non-i.i.d. processes. In Koller et al. [KSBB09], pages 1097–1104.

[MR10] Mehryar Mohri and Afshin Rostamizadeh. Stability bounds for stationary &#966;-mixing and &#946;-mixing processes. *J. Mach. Learn. Res.*, 11:789–814, 2010.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.

[MS99] Rosario N. Mantegna and H. Eugene Stanley. *Introduction to Econophysics: Correlations and Complexity in Finance.* Cambridge University Press, 1999.

[MSS11a] Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark J. Schervish. Estimating beta-mixing coefficients. *CoRR*, abs/1103.0941, 2011.

[MSS11b] Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark J. Schervish. Generalization error bounds for stationary autoregressive models. *CoRR*, abs/1103.0942, 2011.

[MSS11c] Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark J. Schervish. Risk bounds for time series without strong mixing. *CoRR*, abs/1106.0730, 2011.

[MT90] Angelo Melino and Stuart M. Turnbull. Pricing foreign currency options with stochastic volatility. *Journal of Econometrics*, 45(1-2):239–265, 1990.

[Mur85] F. Murtagh. *Multidimensional Clustering Algorithms.* Physica-Verlag, Wuerzburg, 1985.

[Mur98] Kevin Murphy. A brief introduction to graphical models and bayesian networks. `http://www.cs.ubc.ca/~murphyk/Bayes/bnintro.html`, 1998.

[Mur99] John J. Murphy. *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications.* New York Institute of Finance, New York, 1999.

[Mur02] Kevin Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* Dissertation, UC Berkeley, 2002.

[MW01] Kevin P. Murphy and Yair Weiss. The factored frontier algorithm for approximate inference in dbns. In Jack S. Breese and Daphne Koller, editors, *UAI*, pages 378–385. Morgan Kaufmann, 2001.

[MZ93] Stéphane Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41:3397–3415, 1993.

[NJW01] A. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an Algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001.

[NvSK00] Guy P. Nason, Rainer von Sachs, and Gerald Kroisandt. Wavelet processes and adaptive estimation of the evolutionary wavelet spectrum. *Journal Of The Royal Statistical Society Series B*, 62(2):271–292, 2000.

[NW99] Christopher J. Neely and Paul A. Weller. Technical trading rules in the european monetary system. *Journal of International Money and Finance*, 18(3):429–458, 1999.

[ORBD08] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Learning and inferring motion patterns using parametric segmental switching linear dynamic systems. *Int. J. Comput. Vision*, 77:103–124, 2008.

[OSon] Arthur O'Sullivan and Steven M. Sheffrin. *Economics: Principles in Action.* Pearson Prentice Hall, 2004, 3rd edition.

[OW10] Alquier Olivier and Pierre Wintenberg. Model selection for weakly dependent time series forecasting. Working Papers 2010-39, Centre de Recherche en Economie et Statistique, 2010.

[PADF02] E. Punskaya, C. Andrieu, A. Doucet, and W. Fitzgerald. Bayesian Curve Fitting using MCMC with Applications to Signal Segmentation. *IEEE Trans. Signal Process*, 50(3):747–758, 2002.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[Pea00] Judea Pearl. *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2000.

[PF01] Robert R. Prechter and Alfred J. Frost. *Elliott Wave Principle: Key to Market Behavior.* John Wiley & Sons, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 1978-2001.

[Phi85] Peter C.B. Phillips. Time series regression with a unit root. Cowles Foundation Discussion Papers 740R, Cowles Foundation for Research in Economics, Yale University, 1985.

[PHW01] R. Prado, G. Huerta, and M. West. Bayesian time-varying autoregressions: Theory, methods and applications. *Resenhas*, 4:405–422, 2001.

[PI04] Cheol-Ho Park and Scott H. Irwin. The profitability of technical analysis: A review. AgMAS Project Research Reports 37487, University of Illinois at Urbana-Champaign, Department of Agricultural and Consumer Economics, 2004.

[PP11] Bruno Pelletier and Pierre Pudlo. Operator norm convergence of spectral clustering on level sets. *J. Mach. Learn. Res.*, 12:385–416, 2011.

[Pri02] Martin J. Pring. *Technical Analysis Explained: The Successful Investor's Guide to Spotting Investment Trends and Turning Points.* McGraw-Hill, New York, 4th edition, 2002.

[PRM00] Vladimir Pavlovic, James M. Rehg, and John MacCormick. Learning switching linear models of human motion. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *NIPS*, pages 981–987. MIT Press, 2000.

[PY10] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[QCSSL09] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning.* The MIT Press, 2009.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Qui93] J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[Qui96] J. Ross Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research (JAIR)*, 4:77–90, 1996.

[Rab89] Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.

[Ram99] James B. Ramsey. The contribution of wavelets to the analysis of economic and financial data. *Philosophical Transactions of The Royal Society A: Mathematical, Physical and Engineering Sciences*, 357:2593–2606, 1999.

[Reb07] Umaa Rebbapragada. Introduction to machine learning research on time series, 2007.

[RG99] Sam Roweis and Zoubin Ghahramani. A unifying review of linear gaussian models. *Neural Comput.*, 11:305–345, 1999.

[RH08] Joshua W. Robinson and Alexander J. Hartemink. Non-Stationary Dynamic Bayesian Networks. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, Léon Bottou, Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS*, pages 1369–1376. MIT Press, 2008.

[RHBZ95] J. Nocedal R. H. Byrd, P. Lu and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, 1995.

[RM08] L. Rokach and O.Z. Maimon. *Data Mining with Decision Trees: Theory and Applications*. Series in Machine Perception and Artificial Intelligence. World Scientific, 2008.

[Rob67] Harry V. Roberts. Statistical versus clinical prediction of the stock market. Technical report, CRSP, University of Chicago, 1967.

[Ros76a] Stephen A. Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(3):341–360, 1976.

[Ros76b] Stephen A. Ross. Return, risk and arbitrage. Rodney L. White Center for Financial Research Working Papers 17-73, Wharton School Rodney L. White Center for Financial Research, 1976.

[RSS10] Liva Ralaivola, Marie Szafranski, and Guillaume Stempfel. Chromatic pac-bayes bounds for non-iid data: Applications to ranking and stationary-mixing processes. *Journal of Machine Learning Research*, 11:1927–1956, 2010.

[RST10a] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning: Beyond regret. *CoRR*, abs/1011.3168, 2010.

[RST10b] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning: Random averages, combinatorial parameters, and learnability. *CoRR*, abs/1006.1138, 2010.

[RZ97] James B. Ramsey and Zhifeng Zhang. The analysis of foreign exchange data using waveform dictionaries. *Journal of Empirical Finance*, 4(4):341–372, 1997.

[SA09] Ingo Steinwart and Marian Anghel. Consistency of support vector machines for forecasting the evolution of an unknown ergodic dynamical system from observations with unknown noise. *Annals of Statistics*, 37:841, 2009.

[Sam65] Paul A. Samuelson. Proof that Properly Anticipated Prices Fluctuate Randomly. *Industrial Management Review*, 6:41–49, 1965.

[Sau72] N. Sauer. On the Density of Families of Sets. *J. Combinatorial Theory Ser. A*, 13:145–147, 1972.

[SC09] Ingo Steinwart and Andreas Christmann. Fast Learning from Non-i.i.d. Observations. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1768–1776. 2009.

[Sch78] Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[Sch90] Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5:197–227, 1990.

[Sch99] Robert E. Schapire. Theoretical views of boosting and applications. In *ATL*, pages 13–25, 1999.

[Sch10] Mark Schmidt. *Graphical Model Structure Learning with* $\updownarrow_1$*-Regularization*. PhD thesis, University of British Columbia, 2010.

[Scu10] Marco Scutari. Learning bayesian networks with the bnlearn r package. *Journal of Statistical Software*, 35:1, 2010.

[See03] Matthias Seeger. Pac-bayesian generalisation error bounds for gaussian process classification. *J. Mach. Learn. Res.*, 3:233–269, 2003.

[Sew08] Martin Sewell. `http://www.technicalanalysis.org.uk/`, 2008.

[Sew11] Martin Sewell. `http://finance.martinsewell.com/stylized-facts/`, 2011.

[SG] P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*.

[Sha63] William F. Sharpe. A Simplified Model for Portfolio Analysis. *Management Science*, 9(2):277–293, 1963.

[Sha64] William F. Sharpe. Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk. *The Journal of Finance*, 19(3):425–442, 1964.

[Sha66] William F. Sharpe. Mutual Fund Performance. *The Journal of Business*, 39(1):119–138, 1966.

[Sha70] William F. Sharpe. *Portfolio Theory and Capital Markets*. McGraw-Hill series in finance. McGraw-Hill, New York, NY [u.a.], 1970.

[She72] Saharon Shelah. A Combinatorial Problem: Stability and Order for Models and Theories in Infinitary Languages. *Pacific Journal of Mathematics*, 41:247–261, 1972.

[Shi00] H. Shimodaira. Improving Predictive Inference under Covariate Shift by Weighting the Log-likelihood Function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

[Shi08] Robert Shiller. Financial markets, lecture 6, efficient markets vs. excess volatility. http://oyc.yale.edu/economics/financial-markets/content/transcripts/transcript-6-efficient-markets- vs.-excess, 2008.

[SHS09] Ingo Steinwart, Don Hush, and Clint Scovel. Learning from dependent observations. *J. Multivariate Analysis*, 100:175–194, 2009.

[SJC11] Cosma Rohilla Shalizi, Abigail Z. Jacobs, and Aaron Clauset. Adapting to non-stationarity with growing expert ensembles. *CoRR*, abs/1103.0949, 2011.

[SKX09] Le Song, Mladen Kolar, and Eric Xing. Time-Varying Dynamic Bayesian Networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1732–1740. 2009.

[SL89] Jose A Scheinkman and Blake LeBaron. Nonlinear dynamics and stock returns. *The Journal of Business*, 62(3):311–337, 1989.

[SL05] Gilles Stoltz and Gábor Lugosi. Internal regret in on-line portfolio selection. *Mach. Learn.*, 59:125–159, 2005.

[SM00] Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[SNMM07] Mark Schmidt, Alexandru Niculescu-Mizil, and Kevin Murphy. Learning graphical model structure using l1-regularization paths. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, pages 1278–1283. AAAI Press, 2007.

[SS99] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

[Ste01] Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

[Ste02] Ingo Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, 18:768–791, 2002.

[Ste05] Ingo Steinwart. Consistency of support vector machines and other regularized kernel machines. *IEEE Trans. Inform. Theory*, 51:128–142, 2005.

[Sto05] Gilles Stoltz. *Incomplete Information and Internal Regret in Prediction of Individual Sequences.* Dissertation, Université Paris-Sud, 2005.

[Sto11] Gilles Stoltz. *Contributions to the Sequential Prediction of Arbitrary Sequences: Applications to the Theory of Repeated Games and Empirical Studies of the Performance of the Aggregation of Experts.* Habilitation Thesis. 2011.

[STW99] Ryan Sullivan, Allan Timmermann, and Halbert White. Data-Snooping, Technical Trading Rule Performance, and the Bootstrap. *The Journal of Finance*, 54(5):1647–1691, 1999.

[SU97] Scheinerman and Ullman. *Fractional Graph Theory.* John Wiley & Sons, 1997.

[Sud] V. N. Sudakov. Gaussian random processes and measures of solid angles in hilbert space.

[SW88] James H. Stock and Mark W. Watson. A probability model of the coincident economic indicators. NBER Working Papers 2772, National Bureau of Economic Research, Inc, 1988.

[SW89] James H. Stock and Mark W. Watson. New indexes of coincident and leading economic indicators. In *NBER Macroeconomics Annual 1989, Volume 4*, pages 351–409. National Bureau of Economic Research, Inc, 1989.

[SW93] James H. Stock and Mark W. Watson. *A Procedure for Predicting Recessions with Leading Indicators: Econometric Issues and Recent Experience*, pages 95–156. University of Chicago Press, 1993.

[SYS$^+$06] Anne V. Smith, Jing Yu, Tom V. Smulders, Alexander J. Hartemink, and Erich D. Jarvis. Computational Inference of Neural Information Flow Networks. *PLoS Computational Biology*, 2, 2006.

[Tay] Stephen J. Taylor. *Asset Price Dynamics, Volatility, and Prediction.* Princeton University Press, Princeton, NJ.

[Tay94] Stephen J. Taylor. Modeling stochastic volatility: A review and comparative study. *Mathematical Finance*, 4(2):183–204, 1994.

[TC01] Francis E. H. Tay and Lijuan Cao. Application of support vector machines in financial time series forecasting. *Omega*, 29(4):309–317, 2001.

[TC02] Francis E. H. Tay and Lijuan Cao. Modified support vector machines in financial time series forecasting. *Neurocomputing*, 48(1-4):847–861, 2002.

[Thr96] Sebastian Thrun. Is Learning the n-th Thing Any Easier than Learning the First? In *Advances in Neural Information Processing Systems*, volume 8, pages 640–646, 1996.

[TK74] Amos Tversky and Daniel Kahneman. Judgment under Uncertainty: Heuristics and Biases. *Science*, 185(4157):1124–1131, 1974.

[TL04] Allan Tucker and Xiaohui Liu. A bayesian network approach to explaining time series with changing structure. *Intell. Data Anal.*, 8:469–480, 2004.

[Tob56] James Tobin. Liquidity preference as behavior towards risk. Cowles Foundation Discussion Papers 14, Cowles Foundation for Research in Economics, Yale University, 1956.

[Tsa05] Ruey S. Tsay. *Analysis of Financial Time Series (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2005.

[Tsy10] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer, 2010.

[TWW02] Dimitrios D. Thomakos, Tao Wang, and Luc T. Wille. Modeling daily realized futures volatility with singular spectrum analysis. *Physica A: Statistical Mechanics and its Applications*, 312:505 – 519, 2002.

[USC06] Christian Ullrich, Detlef Seese, and Stephan K. Chalup. Foreign exchange trading with support vector machines. In Reinhold Decker and Hans-Joachim Lenz, editors, *Advances in Data Analysis, Proceedings of the 30th Annual Conference of the Gesellschaft für Klassifikation e.V., Freie Universität Berlin*, pages 539–546. Springer, 2006.

[Vap82] Vladimir N. Vapnik. *Estimation of Dependences based on Empirical Data (Information Science and Statistics)*. Springer, 1982.

[Vap95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[Vas77] Oldrich Vasicek. An equilibrium characterization of the term structure. *Journal of Financial Economics*, 5(2):177–188, 1977.

[Vay06] Nicolas Vayatis. *Approches Statistiques en Apprentissage: Boosting et Ranking*. Habilitation Thesis. 2006.

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

[VC74] V. N. Vapnik and A. Ya. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

[VC81] V. N. Vapnik and A. Ya. Chervonenkis. The necessary and sufficient conditions for the uniform convergence of the means to their expectations. *Theory of Probability and their Applications*, 26(3):532–555, 1981.

[vdVW96] Aad van der Vaart and Jon Wellner. *Weak Convergence and Empirical Processes*. Springer-Verlag, 1996.

[VGSB$^+$01] T. Van Gestel, J. K. Suykens, D. E. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on Neural Networks*, 12(4):809–821, 2001.

[Vid03] Mathukumalli Vidyasagar. *Learning and Generalization with Application to Neural Networks*. Springer-Verlag, London, 2003.

[vL07] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[vL09] Ulrike von Luxburg. Clustering stability: An overview. *Foundations and Trends in Machine Learning*, 2(3):235–274, 2009.

[vLBB08] U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of Spectral Clustering. *Annals of Statistics*, 36(2):555–586, 2008.

[VLBD05] Ulrike Von Luxburg and Shai Ben-David. Towards a Statistical Theory of Clustering. In *In PASCAL workshop on Statistics and Optimization of Clustering*, 2005.

[Vov90] Volodimir G. Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, COLT '90, pages 371–386, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[VP90] Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, UAI '90, pages 255–270, New York, NY, USA, 1990. Elsevier Science Inc.

[vS11] Ulrike von Luxburg and Bernhard Schölkopf. Statistical learning theory: Models, concepts, and results. In S. Hartmann D. Gabbay and J. Woods, editors, *Handbook for the History of Logic, Vol. 10: Inductive Logic*. Elsevier, 2011.

[WD09] Bartek Wilczyński and Norbert Dojer. BNFinder: Exact and Efficient Method for Learning Bayesian Networks. *Bioinformatics (Oxford, England)*, 25:286–287, 2009.

[Wei00] Yair Weiss. Correctness of Local Probability Propagation in Graphical Models with Loops. *Neural Computation*, 12:1–41, 2000.

[Whi90] Joe Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley Publishing, 1990.

[Whi00] Halbert White. A Reality Check for Data Snooping. *Econometrica*, 68(5):1097–1126, 2000.

[WKY$^+$11] Zhaowen Wang, Ercan E. Kuruoglu, Xiaokang Yang, Yi Xu, and Thomas S. Huang. Time varying dynamic bayesian network for nonstationary events modeling and online inference. *IEEE Transactions on Signal Processing*, 59:1553–1568, 2011.

[WM97] David H. Wolpert and William Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82, 1997.

[Wol96] David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Comput.*, 8:1341–1390, 1996.

[Wol01] David H. Wolpert. The supervised learning no-free-lunch theorems. In *In Proc. 6th Online World Conference on Soft Computing in Industrial Applications*, pages 25–42, 2001.

[WZ10] Li Wang and Ji Zhu. Financial market forecasting using a two-step kernel learning method for the support vector regression. *Annals of Operation Research*, 174(1):103–120, 2010.

[WZSS08] Kaijun Wang, Junying Zhang, Fengshan Shen, and Lingfeng Shi. Adaptive learning of dynamic bayesian networks with changing structures by detecting geometric structures of time series. *Knowl. Inf. Syst.*, 17:121–133, 2008.

[XM07] Xiang Xuan and Kevin Murphy. Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 1055–1062, New York, NY, USA, 2007. ACM.

[YF00] Byoung-Kee Yi and Christos Faloutsos. Fast time sequence indexing for arbitrary lp norms. In *Proceedings of the 26th International Conference on Very Large Data Bases*, VLDB '00, pages 385–394, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[YSC08] Wei Yan, Martin V. Sewell, and Christopher D. Clack. Learning to optimize profits beats predicting returns: Comparing techniques for financial portfolio optimisation. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 1681–1688, New York, NY, USA, 2008. ACM.

[Yu94] Bin Yu. Rates of convergence for empirical processes of stationary mixing sequences. *The Annals of Probability*, 22(1):94–116, 1994.

[Zad04] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the Twenty-First International Conference on Machine learning*, ICML '04, New York, NY, USA, 2004. ACM.

[Zha04] Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 32:56–134, 2004.

[Zha06a] Tong Zhang. From $
epsilon$-entropy to kl-entropy: Analysis of minimum information complexity density estimation. *Annals of Statistics*, 34:2180, 2006.

[Zha06b] Tong Zhang. Information-Theoretic Upper and Lower Bounds for Statistical Estimation. *Information Theory, IEEE Transactions on*, 52(4):1307–1321, April 2006.

[Zhu05] Xiaojin Zhu. Semi-Supervised Learning Literature Survey. Technical report, Computer Sciences, University of Wisconsin-Madison, 2005.