Imperial College London

Department of Mathematics

Machine learning for directional movement prediction of US corporate bond indices

Author: Andrea IANNUCCI (CID: 02298338)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2022-2023

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Acknowledgements

I would like to express my sincere gratitude to Professor Thomas Cass for his invaluable guidance, support, and insights throughout the completion of this project.

I am deeply appreciative of the precious contributions made by the members of the Fixed Income Quant team at Janus Henderson, particularly Hervé Biancotto and Baris Oztop. Their kindness, consistent support, and constructive feedback were pivotal to the successful completion of this thesis.

I am also indebted to my friends and family for their unwavering support and encouragement. Their unwavering motivation and companionship served as a constant source of inspiration throughout this unforgettable journey.

Abstract

Precise directional predictions in financial markets are essential for optimizing portfolio allocation and executing effective long-term portfolio rebalancing strategies. While extensive research has delved into this area within the realms of stock indices, futures, and foreign exchange markets, this thesis introduces a fresh perspective focused on directional prediction for corporate bond indices. Leveraging the latest insights in factor investing within corporate bond markets, this study applies an array of different machine learning algorithms and data analysis techniques specifically tailored for sequential data to predict directional movements in duration-hedged corporate bond indices. As a result of this comprehensive analysis, the study develops four distinct models. These models not only demonstrate accurate directional predictions for the eleven corporate bond indices under examination but also effectively capture the associated returns.

Contents

List of Figures

List of Tables

Introduction

Directional prediction of market indices is a ubiquitous problem in finance, which has been researched extensively by academics and practitioners. This interest arises from the significant positive impact that accurate predictions can have on optimal portfolio allocation and on the schedule of long term portfolio rebalancing. Moreover, the ability to anticipate these movements offers valuable insights about the expected returns of portfolios that have exposure to a particular index.

One of the pioneering results in this field is attributable to Fama in [1]. In this work the author concludes that knowledge of the time series of the price and other publicly available information regarding a specific security provides no substantial insight for determining the distribution of future prices. While this concept, famously referred to as the 'Efficient Market Hypothesis', was predominant among academics, recent years have seen a surge in articles that challenge this notion.

Illustrative of these opposing viewpoints are the articles by Shiller [2] as well as DeBondt and Thaler [3] that explore short-run momentum as a predictive factor of future positive stock returns. Furthermore, the tendencies of stocks to yield higher returns during specific months of the year, as discussed by Rozeff and Kinney [4] and Dyl [5], alongside predictive patterns utilized in technical analysis, exemplified by the work of Lo, Mamaysky, and Wang [6], collectively underscore the potential predictability of the distribution of future prices.

These new findings renewed the interest in market movements detection through statistical approaches. Tay and Cao in [2] leverage returns computed across different time horizons to train a Radial basis function kernel Support Vector Machine to forecast future prices of five distinct future contracts: Standard&Poor's 500 stock index futures, United States 30-year government bond, United States 10-year government bond, German 10-year government bond and French government stock index futures. The authors then proceed to compare the results of this methodology with those obtained using a neural network, finding that the SVM model demonstrates a better performance.

A similar analysis is repeated by Wang in [3], who employs Principal Component Analysis to reduce the dimensionality of the data before training the classification model. The outcome of this analysis is a set of SVM models with an accuracy consistently higher than 60% in predicting directional daily movements of the Korean composite stock price index as well as the constituents of the Hang Seng index market (a stock market index in Hong-Kong) between 2002 and 2012. The predictive factors explored in this study are past daily returns for the constituents of the index, the S&P 500 index price and the exchange rate of US dollars to Korean Won and to Hong Kong Dollar.

In [4], Novak and Velušček undertake the prediction of daily high price movements for 370 stocks in the S&P500 index, spanning the period from 2005 to 2013. They train different kernel SVM models on sliding windows of the following time series: daily open price, daily high price, daily low price, close price and intraday volume. The most effective model in their analysis, the radial basis function SVM, exhibited an accuracy score ranging from 53.54% to 64.11% across the set of stocks.

One of the most comprehensive analysis in this context, which considers ten distinct data mining techniques, is described in [5]. The authors examine the daily change of closing prices of the Hang Seng index by training the models on open price, daily high price, daily low price, S&P 500 index price, and exchange rate of US dollars to Hong Kong dollar. The best performing method is again the radial basis function kernel SVM, that records a precision greater than 80% both on the test set and the validation set.

In recent times, the investigation of factors capable of explaining the cross-section of excess returns has extended its reach to encompass corporate bond markets. One of the most renowned works that investigate this aspect is a paper by Israel, Palhares and Richardson [6]. In this paper, the authors employ long-short portfolio analysis to study duration hedged monthly excess returns for US Investment Grade and US High yield bond. The results suggest statistically significant positive risk premia for Option Adjusted Spread, equity momentum and value, where this last factor measures how cheap a bond is by comparing its spread to its default probability.

These findings find partial validation in a later examination of the cross-sectional returns of monthly duration-hedged US Investment Grade and US High Yield bonds' excess returns, conducted by Henke in [7]. The outcome of this analysis is that equity momentum and value have statistically significant explanatory power, however the evidence regarding the Option Adjusted Spread factor, the quality factor (determined as a score from 0 to 9, assessing the strength of the fundamentals of a company) and the value factor remains inconclusive.

Building upon these findings, this thesis will provide a prospective on weekly directional movement prediction of duration-hedged sectors indices in the US Investment Grade corporate bond markets. This will be achieved through the application of supervised classification techniques and kernel methods specifically tailored for sequential data.

Outline of the thesis

The thesis is organized as follows: In Chapter 1, Support Vector Machines, Logistic regression model and Gradient boosting trees techniques are presented and analyzed. In the following chapter, the fundamental concepts of kernel methods are elaborated upon, with a focus on the global alignment kernel and the signature kernel as examples of kernel methods tailored for time series. Chapter 3 presents a comprehensive overview of data and the methodology alognside the obtained results.

Chapter 1

Classification Techniques: SVM, Logistic regression and XGBoost

Consider a dataset comprising observations and their associated labels $(x, y) = \{(x_1, y_1), ..., (x_n, y_n)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{1, ..., k\}$, $i = 1, ..., n$, a classification algorithm aims at determining a function $f : \mathbb{R}^d \to \{1, ..., k\}$ that assigns to each observation its corresponding label. When the number of labels is equal to two, the classification problem is also called binary classification problem. Additionally, a variation of this problem involves predicting the likelihood or probability of a data point being associated with a specific label.

Even for this variation, the core objective remains consistent: to obtain a robust model that exhibits the capability to accurately predict the labels for a given observation, even previously unseen ones.

The aim of this chapter is to introduce four classification methodologies and their properties: Logistic regression, Support Vector Machines, Classification trees, and Boosting Trees.

1.1 Support Vector Machines

Support Vector Machines (SVM) are a supervised statistical learning models introduced by Vapnik in [8]. In a binary classification setting, SVM find an hyperplane separating the training data in two groups corresponding to each label.

More specifically, let $(x, y) = \{(x_1, y_1), (x_2, y_2), ...(x_n, y_n)\} \in \mathbb{R}^d \times \{-1, 1\}$ be the data set, a separating hyperplane is an hyperplane $\langle \omega, x \rangle + b = 0$, $\omega \in \mathbb{R}^d$ and $b \in \mathbb{R}$, such that

$$y_i(\langle \omega, x_i \rangle + b) \geq 1 \text{ for every } i = 1, ..., n. \tag{1.1.1}$$

It is easy to prove that the distance between the two closest points with different labels is given by $\frac{2}{||\omega||}$, which corresponds to twice the distance between the points and the hyperplane. This last distance is referred to as the margin and the points that are the closest to the separating hyperplane are called the support vectors.

Since the model should separate the points with different labels as much as possible, the Support Vector Machine classifiers corresponds to the linear decision function $f(x) := \text{sign}(\langle \omega, x \rangle + b)$ that maximizes the margin.

From this observation and the constraint 1.1.1 the problem of finding the optimal separating hyperplane can be formalized as the following constrained minimization problem

$$\begin{cases} \min ||\omega||_2^2 \\ \quad \text{st} \\ y_i(\langle \omega, x_i \rangle + b) \geq 1 \text{ for every } i = 1, ..., n. \end{cases} \tag{1.1.2}$$

The SVM works well when there is a clear margin of separation between classes and is more effective in high dimensional spaces, characteristics that make this method especially suited for datasets where the number of dimensions is bigger than the number of samples. Moreover, given the reduced number of parameters that are needed to describe it, the model is relatively memory efficient.

However SVM is not suited for data sets with more samples than dimensions, with noisy data or in situations where the data points are not exactly separable by any hyperplane.
In cases where most of the data points are linearly separable, for example when observations are affected by measurement errors or in the presence of outliers, this last issue may by solved by relaxing the constraint 1.1.1 to allow for some violations of the decision boundary. Such modification of the SVM model is also known as soft margin SVM whereas the original SVM model is often called hard margin SVM.
This is achieved by introducing a vector of parameters (called slack variables) $\zeta = (\zeta_1, ..., \zeta_n)$ where each component $\zeta_i \geq 0$. This turns the constraint 1.1.1 into:

$$y_i(\langle \omega, x_i \rangle + b) \geq 1 - \zeta_i \text{ for every } i = 1, ..., n.$$

However, for large enough values for every ζ_i, this constraint is always automatically satisfied, therefore the sum of the components of this last vector needs to be included in the objective function. With this remark in mind, the minimization problem 1.1.2, adapted for the soft margin SVM is

$$\begin{cases} \min\limits_{\omega,\zeta} ||\omega||_2^2 + C\sum_{i=1}^n \zeta_i \\ \quad \text{st} \\ \ y_i(\langle \omega, x_i \rangle + b) \geq 1 - \zeta_i \text{ for every } i = 1, ..., n. \\ C > 0, \quad \zeta_i \geq 0 \text{ for every } i = 1, ..., n. \end{cases} \tag{1.1.3}$$

With C being a hyper parameter of the model that is determined during the cross-validation.
From this last expression, the last optimization problem can be understood as a generalization of the original hard margin SVM, for which every slack variable was zero.
Another property of the SVM model, is that it is possible to show that the parameters of the soft margin SVM (and consequently of the hard margin SVM) are linear combinations of the data points. The advantage provided by this reformulation will become apparent in the next chapter, which introduces kernel methods, that allow linear models like the SVM have non-linear decision boundaries.
The first step in proving this result is to define the generalized Lagrangian for soft SVM problem

$$\mathcal{L}(\omega, b, \alpha, \zeta, \beta, \mu) = \frac{1}{2}||\omega||_2^2 + C\sum_{i=1}^n \zeta_i - \sum_{i=1}^n \alpha_i\big(y_i(\langle \omega, x_i \rangle + b) - 1 + \zeta_i\big) - \sum_{i=1}^n \mu_i\zeta_i$$

And notice how, if all the constraints are satisfied, then

$$\max_{\alpha \geq 0, \mu \geq 0} \mathcal{L}(\omega, b, \alpha, \zeta, \beta, \mu) = \frac{1}{2}||\omega||_2^2 + C\sum_{i=1}^n \zeta_i$$

(where for a vector $v \in \mathbb{R}^n$, $v > 0$ is intended as $v_i > 0$, i=1,..., n), but when one at least one of the constraints is not satisfied then

$$\max_{\alpha \geq 0, \mu \geq 0} \mathcal{L}(\omega, b, \alpha, \zeta, \beta, \mu) = \infty$$

So that the problem 1.1.3 can be rewritten as the following min max problem, called primal problem

$$\min_{\substack{\omega, b; \\ \zeta \geq 0}} \max_{\alpha \geq 0, \mu \geq 0} \mathcal{L}(\omega, b, \alpha, \zeta, \beta, \mu)$$

Now, for the soft margin SVM, define the following max min problem, called the dual problem

$$\max_{\alpha \geq 0, \mu \geq 0} \min_{\substack{\omega, b; \\ \zeta \geq 0}} \mathcal{L}(\omega, b, \alpha, \zeta, \beta, \mu)$$

Under certain conditions the equality between the primal and dual problem can be established.

Theorem 1.1.1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ and $g_i(x) : \mathbb{R}^d \to \mathbb{R}$ $i = 1, ..., n$ be convex functions and $h_j(x) : \mathbb{R}^d \to \mathbb{R}$ $j = 1, ..., m$ be affine functions.*
Consider the following convex optimization problem

$$\begin{cases} \min_x f(x) \\ st \\ g_i(x) \leq 0 \quad i = 1...n \\ h_j(x) = 0 \quad j = 1....m \end{cases}$$

If the inequality constraints are feasible, i.e. there exists a value for x such that $g_i(x) < 0$ for $i = 1, ..., n$, then the primal and the dual problem associated to the convex optimization problem have the same solution.

Proof. see chapter 5 in [9]. □

Since the inequality constraints for the problem at hand are linear functions, then the feasibility condition is automatically satisfied, so the solution of the primal and dual problem coincide.
Then, following the derivation in [10], for a fixed α, minimizing the Lagrangian by setting the gradient to zero and simplifying, allows to obtain

$$\begin{cases} \max\limits_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle \\ 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

And express ω as a linear combination of the observations, $\omega = \sum_{j=1}^{n} y_j \alpha_j x_j$.
This implies that the problem of finding ω and b (which can be easily recovered from the expression for the decision function of the SVM), can be reformulated to only depend on the linear combinations of the dot products of the data points.

1.2 Logistic regression

The logistic regression (or logit) is a parametric statistical model that describes non deterministic binary outcomes under the assumption of the existence of a linear relationship between a set of variables and the log-odds of these outcomes.
More precisely, consider the set of observations and response variables $(x, y) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, with $x_i \in \mathbb{R}^d \times \{1, -1\}$ for $i = 1, ..., n$. The logistic regression models the set of response variables y_i conditioned to x_i as a collection of independent random variables with distribution $y_i | x_i \sim Bernoulli(p_i)$, $pi \in (0, 1)$.
A second assumption is that the logarithm of the odds ratio conditioned to the independent variables is a linear combination of the independent variables plus a constant

$$\log \frac{\mathbb{P}(y_i|x_i)}{1 - \mathbb{P}(y_i|x_i)} = \log \frac{p_i}{1 - p_i} = \beta_0 + \langle \beta, x_i \rangle \quad \text{for any } x_i \in x$$

So that the probability of the outcome 1 for the i-th data point is given by

$$p_i = \frac{1}{1 + \exp^{-(\beta_0 + \langle \beta, x_i \rangle)}}$$

This implies that the model predicts outcome 1 if the $p_i > 0.5$ and 0 otherwise.
The function $\sigma : \mathbb{R} \to \mathbb{R}$, $\sigma(x) = \frac{1}{1+\exp^{-x}}$ is ubiquitous in classification problems and is known by the names of logistic or sigmoid function.
Similarly to other statistical models, the parameters β_0, β are commonly estimated through the maximum-likelihood approach. The expression for the conditional likelihood can be derived easily from the hypotheses made above, in fact, starting from the conditional independence and the assumption on the conditional distribution for the response variables and using next the expression for the log odds it is obtained that

$$\begin{aligned} \mathcal{L}(\beta_0, \beta) &= \prod_{\{i|y_i=-1\}} (1 - p_i) \prod_{\{i|y_i=1\}} p_i \\ &= \prod_{\{i|y_i=-1\}} \left(1 - \frac{1}{1 + \exp^{-(\beta_0 + \langle \beta, x_i \rangle)}}\right) \prod_{\{i|y_i=1\}} \left(\frac{1}{1 + \exp^{-(\beta_0 + \langle \beta, x_i \rangle)}}\right) \end{aligned}$$

So that the parameters of the model can be found by maximizing the conditional log-likelihood

$$\begin{aligned}\ell(\beta_0,\beta) &= \log \mathcal{L}(\beta_0,\beta)\\ &= \sum_{\{i|y_i=-1\}} \log\left(1-\frac{1}{1+\exp^{-(\beta_0+\langle\beta,x_i\rangle)}}\right)+\sum_{\{i|y_i=1\}}\log\left(\frac{1}{1+\exp^{-(\beta_0+\langle\beta,x_i\rangle)}}\right)\\ &= \sum_{\{i|y_i=-1\}} \log\left(\frac{1}{1+\exp^{(\beta_0+\langle\beta,x_i\rangle)}}\right)+\sum_{\{i|y_i=1\}}\log\left(\frac{1}{1+\exp^{-(\beta_0+\langle\beta,x_i\rangle)}}\right)\\ &= \sum_{i=1}^{n}\log\left(\frac{1}{1+\exp^{-y_i(\beta_0+\langle\beta,x_i\rangle)}}\right)\end{aligned}$$

It is easily seen how the solution to this minimization problem is not unique in general, in fact, if the columns of the matrix having the values of the observations as rows are not linearly independent, then there are infinitely many solution for β, just like it happens for the linear regression model. Another problem this model can suffer from is overfitting, that is when the model fits the data too closely and is not able to classify accurately new data. This is a problem that can arise when d is large compared to n. A possible solution is to add a L^2 regularization term to the log-likelihood to the expression above, so that the estimated values of β_0 and the components of β are shrinked towards zero, resulting in a simpler model.
The maximization problem for the regularized logistic regression can be formalized as follows

$$\beta^* = \arg\max_{\beta\in\mathbb{R}} \ell(\beta) + C||\beta||_2^2 \quad C>0 \tag{1.2.1}$$

Where, abusing the notation, the term β_0 was included in the vector β that now becomes $\beta = (\beta_0, \beta_1, ..., \beta_k)$ and each independent variable takes the form $x_i = (1, x_{i,1}, ..., x_{i,k})$. Since it provides better readability, this notation will be maintained until the end of this section. Another advantage of introducing the L^2 regularization term is that the minimization problem has a unique solution since the Hessian matrix of the conditional likelihood is negative definite. In fact, as shown in [11]

$$\frac{\partial^2\mathcal{L}}{\partial\beta\partial\beta^T} = -\sum_{i=1}^{n}\frac{\exp(\langle\beta,x_i\rangle)}{\big(1+\exp(\langle\beta,x_i\rangle)\big)^2}x_ix_i^T - \lambda\mathbb{1}_{k+1,k+1} = -xWx^T - \lambda\mathbb{1}_{k+1,k+1}$$

Where x is the matrix with columns $(x_1, ..., x_n)$ and W indicates the diagonal matrix with positive entries

$$W = \operatorname{diag}\left(\frac{\sigma(\langle x_1,\beta\rangle)}{\big(1+\exp(\langle\beta,x_1\rangle\big)}, ..., \frac{\sigma(\langle x_n,\beta\rangle)}{\big(1+\exp(\langle\beta,x_n\rangle\big)}\right)$$

Moreover, anticipating the discussion of the chapter, it is possible to show that even for this model, the parameters given by a linear combination the observations and the problem can be rephrased to be only dependent on dot products of the observations. This result was obtained by Jaakkola and Haussler in [12]. Following the steps presented in [13], the main idea is to take advantage of the inequality

$$\log(\sigma(x)) \le cx - c\log(c) - (1-c)\log(1-c)$$

which, when used in 1.2.1, allows to recover a tight upper bound for the original likelihood maximization problem. Furthermore, it is possible to show that this upper bound has the same stationary points as original problem.

$$\ell(\beta)+C||\beta||_2^2 \le \sum_{i=1}^{n} c_iy_i\langle\beta,x_i\rangle - c_i\log(c_i) - (1-c_i)\log(1-c_i) + C||\beta||_2^2 \quad c_i\in(0,1) \quad i=1,...,n$$

Similarly to the SVM model, after finding the minimizing value for the weights β, the original problem is solved by minimizing over the c_i's. More concretely, setting the gradient with respect to β of the last expression equal to zero yields

$$\frac{\partial\ell}{\partial\beta} = \sum_{i=1}^{n} y_ic_ix_i - 2C\beta = 0_{d+1}$$

$$\implies \beta = \frac{1}{2C}\sum_{i=1}^{n} c_iy_ix_i$$

Then the solution to the maximization problem is obtained as the solution of

$$\begin{cases} \min \frac{1}{2C} \sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j y_i y_j x_i x_j - c_i \log(c_i) - (1-c_i)\log(1-c_i) \\ \quad \text{st} \\ 0 < c_i < 1 \text{ for every } i = 1, ..., n. \end{cases}$$

However, there is no known closed formula for the coefficients of the logistic regression, both for this last problem and for the original regularized likelihood maximization problem. This means that these parameters need to be estimated via numerical methods such as Newton method or the gradient descent methods. For more information about the different numerical methods applicable to this estimation problem refer to [13] and [14] .
Before discussing the next model it is worth mentioning the advantages and disadvantages of choosing this model for supervised classification tasks.
The main advantages of the logistic regression are the ease of training and interpretation, the probabilistic output and the fact that the model can perform well on small datasets compared to more complex machine learning techniques. On the other hand, the assumption of linearity between log odds and observations might be too restrictive in some cases. As anticipated in the previous section, this issue will be addressed in the second chapter where a possible solution is presented.

1.3 XGBoost

1.3.1 Classification trees

Definition 1.3.1. Let (x, y) be set of observations and their associated labels $(x, y) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where $x_i = (x_{i,1}, ..., x_{i,d}) \in \mathbb{R}^d$, $y_i = \{-1, 1\}$ $i = 1, ..., n$. A classification tree is a supervised classification algorithm that partitions $\mathbb{R}^d$ in a set of M disjoint regions $\mathcal{R} = \{R_1, R_2, ..., R_M\}$, such that, every a point $x \in \mathbb{R}^d$ is associated to its label according to the rule $f(x) = \sum_{i=1}^{M} c_i \mathbb{1}_{\{x \in R_i\}}$ with $c_i \in \{-1, 1\}$ for all $i = 1, ..., M$.

The partition $\mathcal{R}$ is determined via a recursive greedy algorithm that minimizes the sum of the classification error at each step by partitioning each region in two disjoint regions until a stopping rule is satisfied. The classification error is measured by a loss function $\ell : \mathbb{R}^d \times \{-1, 1\} \to [0, \infty)$. For a given element of the partition R_i, for which the proportion of observation labeled with k is indicated as $\hat{p}_{i,k}$, the following functions can be used as the loss functions to evaluate the quality of a split:

- Gini index: $1 - \hat{p}_{i,1}^2 - \hat{p}_{i,-1}^2$
- Cross Entropy: $-\hat{p}_{i,1} \log_2 \hat{p}_{i,1} - \hat{p}_{i,-1} \log_2 \hat{p}_{i,-1}$
- Exponential Loss: $\sum\limits_{x_i \in R_i} \exp\big(-y_i f(x_i)\big)$
- Miss-classification error: $1 - \max\{\hat{p}_{i,1}, \hat{p}_{i,-1}\}$

Assuming a stopping rule on the classification error threshold for each separate region, the set of labels and the partitions are identified as follows:

1. Input: The singelton $\mathcal{D} = \{\{\mathbb{R}^d, (x, y)\}\}$, a value ϵ corresponding to the maximum error threshold for each leaf.
2. Create an empty set $\mathcal{R}$ that will contain the final partition
3. Associate a label c_i to every set $\{R_i, (x_i, y_i)\}$ in $\mathcal{D}$, with $(x_i, y_i) = \{(x_{i_1}, y_{i_1}), ..., (x_{i_k}, y_{i_k})\}$, defined as the modal label for this last vector.
4. For each region R_i in $\mathcal{D}$, find the two regions R_1, R_2 of the form $R_1 = \{z \in R_i | z_j > \beta\}$, $R_2 = \{z \in R_i | z_j \leq \beta\}$, $\beta \in \mathbb{R}$, j=1,...,d, such that the loss $\sum_{x_i \in R_1} \ell(x_i, y_i) + \sum_{x_i \in R_2} \ell(x_i, y_i)$ is minimized.
   If $\{R_l, \{(x_i, y_i) | x_i \in R_l\}\}$ with l=1,2 satisfies the stopping rule, add this region to $\mathcal{R}$, otherwise add it to $\mathcal{D}$. Remove R_i from $\mathcal{D}$.
5. If $\mathcal{D}$ is empty, output $\mathcal{R}$, otherwise go to step 2

In addition to the error threshold, alternative stopping criteria can be applied to this algorithm. For instance, one alternative could be setting a threshold on the minimum number of observations that can belong to a region, or establishing a rule on the maximum percentage of observations with identical labels that can be associated with a single region. It's important to note that these alternative stopping rules do not fundamentally alter the main steps of the algorithm previously described.
The main advantages of a binary tree classifier are ease of interpretation and training and also the fact that there is no need to normalize the data prior to the training phase. The major drawback of the classification trees is their high variance: a small perturbation of the data can cause major variations to the structure of the tree. This is due to the greedy nature of the algorithm that partitions the regions. A consequence of this fact is that decision trees may not be suited for classification tasks involving financial time series due the presence of noise.

1.3.2 Gradient boosting trees

Many techniques have been introduced during the years to mitigate the problem of high variance in classification trees, one of the most popular class of algorithms introduced to overcome this issue is known as "gradient boosting trees".
The idea behind these techniques is to combine numerous "weak" learners, which are classifiers that do slightly better than a random classifier. The output is determined by a voting system where each of these 'weak' learners contributes to the output with a weight assigned to their contribution.
One of the most famous implementations of gradient boosting trees is XGBoost, which, since its release in 2014, has been one of the most popular libraries used in machine learning competitions (for more details, see the following survey [15] on Kaggle, one of the biggest platforms for data science competitions).
In the context of binary classification, the XGBoost classifier models the probability of the observation $x \in \mathbb{R}^d$ to be associated to the label 1 as

$$f(x) := \sigma\big(\sum_{k=1}^{K} f_k(x)\big)$$

where σ is the sigmoid function and f_k are distinct classification trees.
Similarly to a standard classification tree, this model can be parametrizedd by the partition $R = \{R_1, ..., R_M\}$ and the weights $c = \{c_1, ..., c_M\}$ associated to each region. However, since the different tree classifiers can have different weights, the label associated to a partition by a single classifier is allowed to be a real number and not just the predicted value for the label.
Just like tree classifiers, the XGBoost algorithm employs a greedy algorithm that minimizes the classification loss at every step in the training phase of the model.
More precisely, following the procedure adopted in the original article by Chen and Guestrin [16], given a set of independent variables and the associated response variables $(x, y) = \{(x_1, y_1), ..., (x_n, y_n)\}$ with $x_i = (x_{i,1}, ..., x_{i,d}) \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, define the objective to be minimized for the (k-1)-th step of the algorithm $\mathcal{L}^{k-1} : \mathbb{R}^d \times \{-1, 1\} \to [0, \infty)$ as

$$\mathcal{L}^{k-1}(x, y) = \sum_{i=1}^{n} \ell(f^{k-1}(x_i), y_i) + \gamma m + \sum_{j=1}^{k-1} \frac{1}{2}\lambda ||c^{(j)}||_2^2$$

where $f^{k-1} = \sum_{j=1}^{k-1} f_j$, is the tree boosted model at the (k-1)-th step of the training cycle, ℓ is the loss function, defined as in the previous section and $c^{(j)}$ denotes the vector of weights associated to the j-th tree.
Assuming that the loss is a twice differentiable function, the boosted model for the k-th step is obtained by adding a new learner f_k to f^{k-1}, so that the quadratic approximation of the objective

is minimized. This concretely translates to minimizing the following expression

$$\begin{aligned}
\mathcal{L}^k(x,y) &= \sum_{i=1}^{n} \ell(f^k(x_i), y_i) + \gamma M + \frac{\lambda}{2}||c^{(k)}||_2^2 \\
&= \sum_{i=1}^{n} \ell(f^{k-1}(x_i) + f_k(x_i), y_i) + \gamma M + \frac{1}{2}\lambda||c^{(k)}||_2^2 \\
&\approx \sum_{i=1}^{n} \ell(f^{k-1}(x_i), y_i) + \frac{\partial \ell(f^{k-1}(x_i), y_i)}{\partial f^{k-1}} f_k(x_i) + \frac{1}{2}\frac{\partial^2 \ell(f^{k-1}(x_i), y_i)}{\partial (f^{k-1}(x_i))^2} f_k^2(x_i) + \gamma M + \frac{\lambda}{2}||c^{(k)}||_2^2 \\
&= \sum_{m=1}^{M} \sum_{\{i|x_i \in R_m\}} \ell(f^{k-1}(x_i), y_i) + \frac{\partial \ell(f^{k-1}(x_i), y_i)}{\partial f^{k-1}(x_i)} c_m^{(k)} + \frac{1}{2}\frac{\partial^2 \ell(f^{k-1}(x_i), y_i)}{\partial (f^{k-1}(x_i))^2} (c_m^{(k)})^2 + \frac{\lambda}{2}(c_m^{(k)})^2 + \gamma M
\end{aligned}$$

Where in the last step the definition of the labels c_m was used.
Define now the quantities $\ell_1(x_i) = \frac{\partial \ell(f^{k-1}(x_i), y_i)}{\partial f^{k-1}(x_i)}$ and $\ell_2(x_i) = \frac{\partial^2 \ell(f^{k-1}(x_i), y_i)}{\partial (f^{k-1}(x_i))^2}$, then the optimal value for the label c_m of the newly added classifier is given by

$$c_m = -\frac{\sum_{\{i|x_i \in R_m\}} \ell_1(x_i)}{\sum_{\{i|x_i \in R_m\}} \ell_2(x_i) + \frac{\lambda}{2}}$$

Plugging this value in the formula for the loss $\mathcal{L}^k(x,y)$ on a single region R_m yields

$$\mathcal{L}^k(x,y,R_m) = -\frac{1}{2}\sum_{m=1}^{M} \frac{\left(\sum_{\{i|x_i \in R_m\}} \ell_1(x_i)\right)^2}{\sum_{\{i|x_i \in R_m\}} \ell_2(x_i) + \frac{\lambda}{2}} + \gamma T$$

which implies that if R_m is split into two disjoint regions R_{m_1} and R_{m_2}, the decrease in loss (or gain) by adding the new learner is then

$$\Delta\mathcal{L} = \mathcal{L}^k(x,y,R_{m_1}) + \mathcal{L}^k(x,y,R_{m_2}) - \mathcal{L}^{k-1}(x,y,R_m)$$

The optimal way of splitting each R_m, which determines the partition used by the new learner, is found by searching the maximum reduction of the loss over all possible division of such region. The algorithm for this last part goes as follows

1. Input: the set I $= \{i|x_i \in R_m\}$ with R_m a generic region.
2. Initialize the variables gain=0, $G = \sum_{\{i|x_i \in R_m\}} \frac{\partial \ell(f^{k-1}(x_i), y_i)}{\partial f^{k-1}(x_i)}$ and $H = \sum_{\{i|x_i \in R_m\}} \frac{\partial^2 \ell(f^{k-1}(x_i), y_i)}{\partial (f^{k-1})^2}$
3. Fix $k \in \{1, ..., m\}$, initialize the variables $G_L = 0$ and $H_L = 0$. Sort the set I by the values $x_{k,\cdot}$. Now for each value $x_{i,k}$ define the update G_L and H_L by the following rule

   $$G_L = G_L + \frac{\partial \ell(f^{k-1}(x_i), y_i)}{\partial f^{k-1}(x_i)} \quad H_L = H_L + \frac{\partial^2 \ell(f^{k-1}(x_i), y_i)}{\partial (f^{k-1})^2}$$

   define $G_R = G - G_L$ and $H_R = H - H_L$.
   The gain of this configuration is now given by

   $$\text{gain config} = \frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{G^2}{H + \lambda}$$

   if gain config > gain, then save this configuration and set gain = gain config
4. Repeat the previous step for all $k = 1, ..., m$
5. Output: The two regions $R_{m,1}$ and $R_{m,2}$ that produce the maximum gain.

In addition to the previously described process, it is possible to select from a set of hyperparameters that can be fine-tuned to achieve an optimal balance between bias and variance, thereby preventing overfitting and controlling various aspects of the training procedure. Some of these hyperparameters include:

- **Number of trees**
- **Learning rate**: A value between in [0, 1] that regulates step size shrinkage applied to each feature weight to prevent overfitting
- **Max depth**: The maximum depth of a tree, the higher this value, the higher the more likely the model is to overfit
- **Column sample by tree**: Subsample ratio of columns that is taken when constructing a new tree
- **Column sample by level**: Subsample ratio of columns taken when a new level of the tree is reached.
- **L1 regularization** term for the weights of the trees
- **L2 regularization** term for the weights of the trees

As in the other methods, this section is concluded by listing advantages and shortcomings of the XGBoost model. The main advantages of the XGBoost model are its flexibility in handling different type of inputs, the great accuracy even in the case of nonlinear relations between the features and the targets and its efficiency in terms of speed and memory usage compared to other gradient boosting techniques. The two main disadvantages are the fact that XGBoost is a black-box method making it difficult to interpret the result and the wide range of parameters that need to be carefully tuned to achieve optimal performance.

Chapter 2

Kernel methods for time series

As previously anticipated, this chapter introduces kernel methods, a class of machine learning algorithms designed to enhance traditional linear classifiers allowing them to be applied to problems where the data in not linearly separable. The main idea behind kernel methods is to use a function $\phi : V \to H$, called feature map, that maps the input from the data space V to a higher-dimensional inner product space H, sometimes called feature space. This transformation may render the data linearly separable in the feature space, allowing successful application of linear classifiers.
Associated to ϕ is the so called kernel $K : V \times V \to \mathbb{R}$ which is a symmetric map defined as $K(x, y) : \langle \phi(x), \phi(y) \rangle_H$. From the dual formulation for the SVM and logistic regression problems, presented in the previous chapter, it is evident that the solution of their related convex optimization problems only depends on the dot products of the data points. This implies that, when using kernel methods with these classification algorithms, instead of explicitly performing the feature space transformation, which could be computationally expensive or even infeasible in very high-dimensional spaces, it is possible to work directly in the original space using the kernel function. This is commonly known as "kernel trick" and its applicability extends to a considerable array of machine learning algorithms, as shown in detail in the reference [17].
The chapter is structured as follows:

- **Introduction to Kernels and Reproducing Kernel Hilbert Spaces**: this section presents the foundational concepts of kernels, Reproducing Kernel Hilbert Spaces (RKHS), and introduce the Representer Theorem. This theorem demonstrates that solutions to a wide range of optimization problems can be expressed as a finite linear combinations of kernel evaluations at sample points.

- **Kernel Properties for Sequentially Ordered Data**: This part of the chapter focuses on the properties of two specific kernels tailored for sequentially ordered data: the Global Alignment Kernel (GAK) and the signature kernel

The chapter begins by introducing the class of kernel that are considered for most of the results.

Definition 2.0.1. Given any set of points $\{x_1, ..., x_n\}_{n \in \mathbb{N}}$ that belongs to the data space V and a kernel map $K : V \times V \to \mathbb{R}$, the matrix

$$G := (K(x_i, x_j))_{i,j} \qquad i, j = 1, ..., n$$

is called the Gram matrix.

Definition 2.0.2 (Positive semidefinte kernels)**.** Let V be the data space and $K : V \times V \to \mathbb{R}$ be a kernel. If for any set of points $\{x_1, ..., x_n\}_{n \in \mathbb{N}} \subset V$, the Gram matrix associated to K is a positive semidefinite matrix then K is said to be a positive semidefinite kernel.

Example 2.0.1. *A trivial example of a positive semidefinite kernel is the linear kernel, also known as "non-kernel". This kernel is generated by the feature map $\phi(x) = x$ and corresponds to the dot product on the data space V*

$$K(x, y) = \langle x, y \rangle_V$$

Positive semidefinite kernels are related to Reproducing Kernel Hilbert Spaces (RKHS), which were first proposed by Aronszajn in [18]. A Reproducing Kernel Hilbert Space is a Hilbert space of functions characterized by the property that the value of a function at any point is equal to the inner product of that function and a particular kernel, called the reproducing kernel.

Definition 2.0.3. Let H be a Hilbert space of functions defined on set V. A reproducing kernel of H is a map $K : V \times V \to \mathbb{R}$ such that

1. The function $K(\cdot, x) : V \to \mathbb{R}$ is in H for every $x \in X$
2. $f(x) = \langle f, K(\cdot, x)\rangle_H$ for every $f \in H$ and for every $x \in X$

The first step towards the definition of the RKHS is the definition of the evaluation functional

Definition 2.0.4. Let H be a Hilbert space of functions on a set V, the evaluation functional is defined as the linear functional

$$L_x : f \to f(x) \quad \text{for every } f \in H$$

that associates to each function f in H its value at $x \in V$

Definition 2.0.5 (Reproducing Kernel Hilbert Space)**.** Given an Hilbert space of functions H defined on set V, if the evaluation functional L_x is continuous, that is, if for every $x \in V$ there is a value $C(x) \geq 0$ such that for all $f \in H$

$$|f(x)| \leq C(x)||f||_H$$

Then the space H is called a reproducing kernel Hilbert space

The following theorem, known as Riesz representation theorem, is a fundamental result in the theory of Hilbert space and establishes a connection between continuous linear functionals and evaluation kernels.

Theorem 2.0.6 (Riesz representation theorem)**.** *Let H be an Hilbert space on a set V, then for every continuous linear functional L and any $f \in H$ there exists a unique $\phi \in H$ such that $L(f) = \langle f, \phi\rangle_H$.*

Consider the RKHS H over a set V, define the function $\phi_x : V \to \mathbb{R}$ to be the element of H associated to the evaluation functional at x by the Riesz representation theorem.
By setting $K(\cdot, x) := \phi_x(\cdot)$, it is possible to recover an expression a reproducing kernel of H, in fact, the following holds

1. $K(\cdot, x) = \phi_x(\cdot) \in H$ for every $x \in V$
2. $f(x) = \langle f, \phi_x(\cdot)\rangle_H = \langle f, K(\cdot, x)\rangle_H$

Notice also that, since $K(x, \cdot) \in H$ for every $x \in V$, from the previous construction

$$\begin{aligned} K(x, y) &= \langle K(\cdot, x), K(\cdot, y)\rangle_H \\ &= \langle \phi_x(\cdot), \phi_y(\cdot)\rangle_H \end{aligned}$$

Which also allows to identify the map ϕ_x obtained via the Riesz theorem with the feature map ϕ that was introduced as the beginning of the chapter.
Another fundamental contribution to the theory of Hilbert spaces is given by the Moora-Aronszajn theorem, which guarantees that every positive semidefinite kernel K can be associated to a unique RKHS.

Theorem 2.0.7 (Moora-Aronszajn Theorem [18])**.** *Given a positive semidefinite kernel $K : V \times V \to \mathbb{R}$, then there exists a unique RKHS H, such that, for any $f \in H$ and $x \in V$*

$$f(x) = \langle K(x, \cdot), f\rangle_H$$

Leveraging these results for the RKHS it is possible to give a proof of the representer theorem.

Theorem 2.0.8. *Let $K : V \times V \to \mathbb{R}$ a semipositive defined kernel and H the RKHS associated to K. Given $\{(x_1, y_1), ..., (x_n, y_n)\}_{n \in \mathbb{N}} \subset V \times \mathbb{R}$, an arbitrary function $\ell : (V \times \mathbb{R}^2)^n \to \mathbb{R}$ and a strictly increasing function $g : [0, \infty) \to \mathbb{R}$ then the minimiser*

$$f^* = arg \min_{f \in H} \ell((x_1, y_1, f(x_1)), ..., (x_n, y_n, f(x_n))) + g(||f||_H^2)$$

if it exists, belongs to $Span\{K(x_i, \cdot)|i = 1, ..., n\}$

Proof. Let $S = Span\{K(x_i, \cdot)|i = 1, ..., n\}$ and for any $f \in H$ denote by f_S the projection of f on S and by $f_{S^\perp}$ the projection of f on $S^\perp$. From the definition of S, f_S is of the form

$$f_S = \sum_{i=1}^{n} c_i K(x_i, \cdot) \text{ where } c_i \in \mathbb{R},\ i = 1, ..., n$$

The reproducing property of the kernel $K(x_j, \cdot)$ for every $j = 1, ..., n$ allows to write

$$\begin{aligned} f(x_i) &= \langle f, K(x_i, \cdot)\rangle_H \\ &= \langle \sum_{i=1}^{n} c_i K(x_i, \cdot) + f_{S^\perp}, K(x_j, \cdot)\rangle_H \\ &= \langle \sum_{i=1}^{n} c_i K(x_i, \cdot), K(x_j, \cdot)\rangle_H \\ &= \langle \sum_{i=1}^{n} c_i K(x_i, x_j)\rangle_H \end{aligned}$$

Where in the second step the projection theorem was used, which guarantees that $H = S \oplus S^\perp$ (S is a finite dimensional linear space, therefore it is closed) and in the last step the reproducing property of $K(x_j, \cdot)$ and the fact that $K(x_i, \cdot) \in H$ for every $i = 1, ..., n$.
Now, using this last identity, the projection theorem and recalling that g is strictly increasing

$$\begin{aligned} &\ell((x_1, y_1, f(x_1)), ..., (x_n, y_n, f(x_n))) + g(||f||_H^2) \\ &= \ell((x_1, y_1, f_S(x_1)), ..., (x_n, y_n, f_S(x_n))) + g(||f||_H^2) \\ &= \ell((x_1, y_1, f_S(x_1)), ..., (x_n, y_n, f_S(x_n))) + g(||f_S||_H^2 + ||f_{S^\perp}||_H^2) \\ &\geq \ell((x_1, y_1, f_S(x_1)), ..., (x_n, y_n, f_S(x_n))) + g(||f_S||_H^2) \end{aligned}$$

Which implies that the minimiser is reached when $f \in S$. □

It is evident that SVM and logistic regression classifiers are solutions to optimization problems that satisfy the assumptions outlined in theorem 2.0.8. Consequently, when using positive semidefinite kernels, the optimal predictors for these two methods can be expressed through a linear combination of kernel functions computed at specific data points.

2.0.1 Kernel approximation methods

The computation of the Gram matrix for a set of n distinct k-dimensional data points comes with substantial computational demands. It incurs a space complexity of $O(n^2)$ and a time complexity of $O(n^2k)$ for many computations of interest, such as matrix inversion or eigendecomposition.
These complexities pose a significant challenge for kernel methods when dealing with large datasets, making efficient scaling difficult. To address this challenge, several methods have been proposed to approximate the Gram matrix efficiently. Among these, three popular techniques stand out: Choleski decomposition [19], Randomized Fourier Features [20], that approximates the feature map for shift invariant kernels, and the Nyström method.
The latter is based on a result from the theory of integral equations obtained by Nyström in [21] that was popularized by Williams and Seeger in [22] for applications to Gram matrices.

This method uses a sample of m columns from the Gram matrix $K \in \mathbb{R}^{n\times n}$ to construct its approximation, $\tilde{K} \in \mathbb{R}^{n\times n}$. This last matrix has rank k, with $k << n$ and can be obtained using SVD decomposition on the original matrix. The storage cost of this new matrix is $O(nm)$ and it requires $O(nkm)$ operations to be computed.
More precisely, let $K \in \mathbb{R}^{n\times n}$ be a symmetric semidefinite matrix with $rank(K) = r$. The SVD decomposition of K allows to recover that $K = U\Sigma U^T$ where $U \in \mathbb{R}^{n\times r}$ and $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r)$ with $\sigma_i > 0$ for every $i = 1, ..., r$. From the SVD decomposition it is also possible to write the

pseudo-inverse of the matrix K as $K^+ = U\Sigma^{-1}U^T$, where $\Sigma^{-1} = \text{diag}(\sigma_1^{-1}, ..., \sigma_r^{-1})$.
The method works as follows: suppose without loss of generality that the sampled columns are the first m columns of K, and their corresponding singular values are ordered in decreasing order, then, using block notation write K as

$$K = \begin{bmatrix} W & K_{2,1}^T \\ K_{2,1} & K_{2,2} \end{bmatrix}$$

with $W \in \mathbb{R}^{m\times m}$, $K_{2,1} \in \mathbb{R}^{n-m\times m}$, and define the matrix containing the sampled columns as $C := \begin{bmatrix} W \\ K_{2,1} \end{bmatrix}$.
The SVD decomposition of W is given by $W = U_1\Sigma_m U_1^T$ where $U = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$, with $U_1 \in \mathbb{R}^{m\times m}$ and $\Sigma_m = \text{diag}(\sigma_1, ..., \sigma_m)$.
Following Williams in [22], the optimal rank k approximation of W with respect to the Frobenius norm is $\tilde{W} = \tilde{U}_1\tilde{\Sigma}\tilde{U}_1^T$ where the columns of the components are given by:

$$\tilde{\sigma_i} = \frac{n}{m}\sigma_i \qquad i = 1, ..., k$$
$$\tilde{u_i} = \sqrt{\frac{m}{n}}\frac{1}{\sigma_i}Cu_i \qquad i = 1, ..., k$$

with u_i being the i-th column of U_1.
The Nyström approximation of K is then

$$\tilde{K} = C\tilde{W}^+C^T \tag{2.0.1}$$

This matrix requires a space complexity of $O(nm)$ to store C, whilst the time complexity of the computation is given by $O(kmn + m^2k) = O(kmn)$ since the SVD decomposition requires $O(m^2k)$ and the matrix multiplication in 2.0.1 requires $O(kmn)$ operations.
Moreover, if $k = n$, as shown in [22], the Nyström approximation reproduces perfectly all the blocks of K except for the block $K_{2,2}$. In fact, in this case, the block representation of $\tilde{K}$ is given by

$$\tilde{K} = \begin{bmatrix} W & K_{2,1}^T \\ K_{2,1} & K_{2,1}W^+K_{2,1} \end{bmatrix}$$

2.1 Radial basis function kernel

The most popular kernel used in classification tasks is the radial basis function kernel, which can be defined as follows

Definition 2.1.1. Let $\sigma \in \mathbb{R}^+$, $x = \{x_0, ..., x_{n-1}\}$ and $y = \{y_0, ..., y_{n-1}\}$ be two elements in $\mathbb{R}^n$, then the radial basis function kernel is defined as

$$K(x, y) = e^{-\frac{1}{2\sigma^2}\sum_{k=0}^{n-1}(x_i - y_i)^2}$$

The radial basis function kernel is a decreasing function of the euclidean distance between the two points. It can be interpreted as a measure of similarity between points: the closer two points are, the larger the value this function takes. The value σ regulates how much the distance between the points affects the value of the kernel function. The higher the value of sigma the more impact the distance between the points is going to have on the value of the kernel. This kernel offers a first example of an alternative to distances as a means of measuring similarity or dissimilarity between points. In the next section this concept will be explored more in depth by introducing a class of symmetric maps that can be used in place of distances in many classification tasks.
The next step in analyzing the properties of this kernel is to show that it is a positive semidefinite kernel. This result will ensure compatibility with all the results introduced in the previous part of this chapter.

Proposition 2.1.2. *The radial basis function kernel is positive semidefinite*

Proof. The result follow easily from the following identity for the Gaussian density function

$$e^{-\frac{1}{2\sigma^2}(x-y)^2} = \frac{\sqrt{2\pi\sigma^2}}{2}\int_{-\infty}^{\infty} e^{-\frac{1}{\sigma^2}(z-x)^2} e^{-\frac{1}{\sigma^2}(y-z)^2} dz \qquad x, y \in \mathbb{R}$$

In fact, from the definition of positive semidefinite kernel, for a collection any set of points $\{x_0, ..., x_{n-1}\}_{n\in\mathbb{N}} \subset \mathbb{R}^n$ and any set $\{c_1, ..., c_n\}_{n\in\mathbb{N}} \subset \mathbb{R}$ the following holds

$$\begin{aligned}
\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} c_i c_j e^{-\frac{(x_i-x_j)^2}{2\sigma^2}} &= \frac{\sqrt{(2\pi\sigma^2)^d}}{2^d}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} c_i c_j \int_{\mathbb{R}^d} e^{-\frac{(z-x_j)^2}{\sigma^2}} e^{-\frac{(x_i-z)^2}{\sigma^2}} dz \\
&= \frac{\sqrt{(2\pi\sigma^2)^d}}{2^d}\int_{\mathbb{R}^d}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} c_i c_j e^{-\frac{(z-x_j)^2}{\sigma^2}} e^{-\frac{(z-x_i)^2}{\sigma^2}} dz \\
&= \frac{\sqrt{(2\pi\sigma^2)^d}}{2^d}\left|\left|\sum_{i=0}^{n-1} c_i e^{-\frac{(z-x_i)^2}{\sigma^2}}\right|\right|_{L^2} \geq 0
\end{aligned}$$

which concludes the proof □

Moreover, it is possible to recover an explicit form for the feature map generating the radial basis function kernel. As it will be shown in the next lines, the feature map $\phi : \mathbb{R}^n \to H$ maps $\mathbb{R}^n$ to H, where H is the Hilbert space l^2, which is the space of square summable sequences taking values in $\mathbb{R}$ equipped with the inner product

$$\langle x, y\rangle = \sum_{u=1}^{\infty} x_u y_u$$

For simplicity, consider two points x and y in $\mathbb{R}$, then the radial basis function kernel

$$\begin{aligned}
K(x, y) &= e^{-\frac{1}{2\sigma^2}(x-y)^2} \\
&= e^{-\frac{1}{2\sigma^2}(x^2+y^2-2xy)} \\
&= e^{-\frac{1}{2\sigma^2}(x^2+y^2)}\sum_{k=0}^{\infty}\frac{x^k y^k}{\sigma^{2k}k!}
\end{aligned}$$

Where the last step follows from the series expansion of the exponential function. Then a feature map generating the radial basis function kernel for points in $\mathbb{R}$ is

$$\phi(x) = e^{-\frac{1}{2\sigma^2}x^2}\left(1, \sqrt{\frac{1}{1}}\frac{x}{\sigma}, \sqrt{\frac{1}{2}}\frac{x^2}{\sigma^2}, \sqrt{\frac{1}{3!}}\frac{x^3}{\sigma^3}, ...\right)$$

The case where x, $y \in \mathbb{R}^n$ just requires a simple modification of this last proof and is presented in detail in [23].

2.2 The DTW distance and the global alignment kernel

As previously mentioned, the following section introduces a class of symmetric maps, called dissimilarity measures, that can be used to assess the similarity between points. Dissimilarity measures provide more flexibility compared to traditional distances while remaining compatible with most machine learning classification techniques. Before defining dissimilarity measures, we shall recall the definition of distance

Definition 2.2.1 (Distance)**.** Let X be a non empty set. A distance $d : X \times X \to \mathbb{R}$ is a map that satisfies

1. $d(x, y) \geq 0$ for every x, y in X
2. $d(x, y) = 0$ if and only if $x = y$

3. $d(x, y) = d(y, x)$ for every x, y in X
4. $d(x, y) \leq d(x, z) + d(z, y)$ for every x, y and z in X

In classification tasks, the distance can be used as a measurement of dissimilarity between two elements in a space. However, even for classification methods based on dissimilarity measurements, properties (2) and (4) are not essential. For example in time series analysis it could be desirable to have a symmetric map d on a set X, $d : X \times X \to [0, \infty]$ such that if two time series x and y take values in X and are equal up to time reparametrization, then $d(x, y) = 0$. It becomes evident that a map defined in such a manner has the potential to violate conditions (2) and (4), yet it can still possess considerable utility within certain contexts.
Motivated by this fact, by relaxing this two conditions, following Bock in [24], a dissimilarity measure can be defined as follows

Definition 2.2.2 (Dissimilarity measure)**.** Let X be a non empty set. A distance $d : X \times X \to \mathbb{R}$ is a map that satisfies

1. $d(x, y) \geq 0$ for every x, y in X and $d(x, y) = 0$ if $x = y$ (positivity)
2. $d(x, y) = d(y, x)$ for every x, y in X (symmetry)

2.2.1 The DTW distance

An example of a dissimilarity measure is the Dynamic Time Warping distance (DTW distance), introduced independently by Vintsyuk [25] and Sakoe and Chiba in [26] for classification tasks in speech recognition.
The DTW distance is a dissimilarity measure for time series that extends a distance by considering a set of time reparametrizations and defining the dissimilarity as the minimal distance between the two time series among all possible time reparametrization. This unique property has found applications in scenarios where both a time series and its time reparametrization are categorized as belonging to the same class. Examples of applications where the DTW has produced state of the art results are music and signal processing ([27] and [28]), finance ([29]) and speech recognition ([30], [31]).

Definition 2.2.3 (DTW distance)**.** Let V be a finite dimensional vector space and the partitions $\mathcal{T} = \{t_0 < ... < t_{n-1}\}$ and $\mathcal{S} = \{s_0 < ... < s_{m-1}\}$ of $[t_0, t_{n-1}]$ and $[s_0, s_{m-1}] \subset \mathbb{R}_0^+$ respectively. Given two time series $x = \{(t_i, x_i)\}_{(t_i, x_i) \in V \times \mathcal{T}}$ and $y = \{(s_i, y_i)\}_{(s_i, y_i) \in V \times \mathcal{S}}$ and a distance d, the dynamic time warping distance is defined as

$$DTW(x, y) = \min_{\pi \in \Pi(x,y)} \sum_{(i,j) \in \pi} d(x_i, y_j) \tag{2.2.1}$$

where $\Pi(x, y)$ is the set of all the collections of pairs, $\pi = ((i_0, j_0), ..., (i_K, j_K))$, called warping path, that satisfy

1. $\pi_0 = (0, 0)$ and $\pi_K = (n - 1, m - 1)$ (boundary constraint)
2. If $\pi_l = (i_l, j_l)$ and $\pi_{l+1} = (i_{l+1}, j_{l+1})$ then $i_l \leq i_{l+1} \leq i_l + 1$, $j_l \leq j_{l+1} \leq j_l + 1$ and $(i_{l+1}, j_{l+1}) \neq (i_l, j_l)$ (monotonicity and continuity constraint)

It is evident from the definition above that the DTW distance is well defined even when the two time series that have different lengths. This further enhances the flexibility of this dissimilarity measure when it is compared to distances.
However, the flexibility provided by the DTW distance comes with the caveat of a large number of admissible time reparametrizations that must be taken into account during the DTW computation. Given two time series of equal length n, the number of admissible paths is in fact given by the $n - th$ Delannoy number D(n) (refer to [32] for more information about the Delannoy numbers). This leads to a possible number of reparametrizations which has asymptotic

$$D(n) \approx \frac{(3 + 2\sqrt{2})^n}{\sqrt{4\pi n(3\sqrt{2} - 4)}}(1 + O(\frac{1}{n}))$$

that would make the computation unfeasible for large enough n.
The solution to this problem comes from the constraints on the admissible paths, that allow the computation of the DTW distance to be formulated as a dynamic programming problem requiring space and time complexity of the order $O(mn)$ if m and n are of the same order of magnitude.
In fact denoting by $x^u = (x_0, ..., x_u)$ the truncated time series at time u, from the expression 2.2.1, using first the boundary constraint and next the monotonicity and continuity constraints, it's possible to obtain the following dynamic programming problem.

$$\begin{aligned}\mathrm{DTW}(x^u, y^v) &= \min_{\pi \in \Pi(x^u, y^v)} \sum_{(i,j)\in\pi} d(x_{t_i}, y_{t_j}) \\ &= d(x_u, y_v) + \min(\mathrm{DTW}(x^{u-1}, y^{v-1}), \mathrm{DTW}(x^u, y^{v-1}), \mathrm{DTW}(x^{u-1}, y^v))\end{aligned}$$

However, for longer time series, a quadratic time complexity may still result problematic. A possible solution to reduce computational time is to include additional constraint in the set of admissible paths, modifying the dissimilarity measure in a way that the resulting dissimilarity measure will be invariant just for local reparametrizations. Examples of additional constraint are

- **The Sakoe-Chiba band**, introduced in [26], is a constraint that requires the distance between the coordinates of the warping path to be less than a certain value, more precisely $\max(||i - d||, ||i - d||) \leq B$ where $(i, j) \in \pi$ and π is an admissible path according to the definition 2.2.3, d is the segment connecting the first point in π to the last point in π and $B \in \mathbb{N}$.

- **The Itakura parallelogram** , introduced in [33], that imposes a maximum slope of the warping path $\max(\frac{i}{j}, \frac{j}{i}, \frac{n-i}{m-j}, \frac{m-j}{n-i}) \leq P$ where $(i, j) \in \pi$ and π is an admissible path according to the definition 2.2.3 for two time series with length n and m respectively and $P \geq \frac{m}{n}$

This additional constraints not only reduce computational time but have been shown to improve the performance of classifiers in certain classification tasks, for example in [26] for speech recognition and in [34] on an array of different classification tasks.

2.2.2 The global alignment kernel

The Global Alignment Kernel (GAK) is class of kernels proposed by Cuturi et al. in [35]. These kernels are specifically designed for time series data and are based on the DTW distance.

Definition 2.2.4 (Global alignment kernel)**.** Let $\mathcal{T} = \{t_0 < ... < t_{n-1}\}$ and $\mathcal{S} = \{s_0 < ... < s_{m-1}\}$. Consider two time series taking values in $\mathbb{R}^n$, $x = \{(t_i, x_i)\}_{t_i \in \mathcal{T}}$ and $y = \{(s_i, y_i)\}_{s_i \in \mathcal{S}}$, a positive semi definite kernel θ defined on $\mathbb{R}^n$. Denote the set of all warping sequences between x and y as $\Pi(x, y)$. Then the global alignment kernel induced by θ is defined as

$$\begin{aligned}K(x, y) &= \sum_{\pi \in \Pi(x,y)} e^{-\sum\limits_{(i,j)\in\pi(x,y)} \theta(x_i, y_j)} \\ &= \sum_{\pi \in \Pi(x,y)} \prod_{l=1}^{|\pi|} k(x_{\pi_1(l)}, y_{\pi_2(l)})\end{aligned}$$

where $x_{\pi_1(\cdot)}$ is the reparametrization of the path x determined by the warping path π, defined as in 2.2.1 and $k(x_{\pi_1(l)}, y_{\pi_2(l)}) = e^{-\theta(x_{\pi_1(l)}, y_{\pi_2(l)})}$

Similarly to the DTW distance, leveraging the constraints on the warping paths coordinates, the computation of the global alignment kernel can be formulated as a dynamic programming problem with time complexity $O(mn)$, where m and n denote the lengths of the involved time series. In fact, denoting by x^u the time series x truncated at time t_u and using first the boundary constraint and next the monotonocity and continuity constraints yields

$$
\begin{aligned}
K(x^u, y^v) &= \sum_{\pi \in \Pi(x^u, y^v)} e^{-\sum\limits_{(i,j)\in\pi(x^u,y^v)} \theta(x_i,y_j)} \\
&= \sum_{\pi \in \Pi(x^u, y^v)} e^{-\theta(x_u,y_v) + \big(-\sum\limits_{(i-1,j)\in\pi(x^{u-1},y^v)} \theta(x_i,y_j) \;-\; \sum\limits_{(i-1,j)\in\pi(x^{u-1},y^{v-1})} \theta(x_i,y_j) \;-\; \sum\limits_{(i-1,j)\in\pi(x^{u},y^{v-1})} \theta(x_i,y_j)\big)} \\
&= \big(K(x^u, y^{v-1}) + K(x^{u-1}, y^v) + K(x^{u-1}, y^{v-1})\big) k(x_u, y_v)
\end{aligned}
$$

Moreover, Theorem 1 in [35] provides sufficient conditions for the global alignment kernel to be positive definite, so that this kernel satisfies the conditions required by the representer theorem

Theorem 2.2.5. *Let $\mathcal{T} = \{t_0 < ... < t_{n-1}\}$ and $\mathcal{S} = \{s_0 < ... < s_{m-1}\}$. Consider two time series taking values in $\mathbb{R}^n$, $x = \{(t_i, x_{t_i})\}_{t_i \in \mathcal{T}}$ and $y = \{(s_i, y_{s_i})\}_{s_i \in \mathcal{S}}$, a positive semidefinite kernel θ defined on $\mathbb{R}^n$, and let $\Pi(x, y)$ be the set of all warping sequences between x and y. Let K be the global alignment kernel on the path x and y and k as in 2.2.4, if $\frac{k}{1+k}$ is positive definite kernel, then the global alignment kernel K induced by θ is positive definite.*

By using theorem it is possible to show that the GAK induced by the squared Euclidean distance is in fact a semipositive definite kernel. This result follows from the relation between a class of functions known as completely monotone functions and radial kernels, which are a class of kernels whose value depends on the squared distance between the two points the kernel is evaluated at.

Definition 2.2.6. An infinite differentiable function $f : [0, \infty) \to \mathbb{R}$ is said to be a completely monotone if $(-1)^n f^{(n)}(x) \geq 0$ for any $n = 0, 1, ...$ and $x \in (0, \infty)$

Bernstein's theorem [36] relates completely monotone function and Laplace transforms of non-negative finite Borel measures on $[0, \infty)$.

Theorem 2.2.7 (Bernstein's theorem). *A function $f : [0, \infty) \to \mathbb{R}$ is completely monotone if and only if $f(x) = \int_0^\infty e^{-tx} d\alpha(t)$ for every $x \in (0, \infty)$ where α is a non-negative finite Borel measure on $[0, \infty)$.*

The last result that will be used is a contribution of Schoenberg [37] that provides the aforementioned link between completely monotone functions and positive semidefinite kernels

Lemma 2.2.8. *Let f be a completely monotone function then the radial kernel $K(x, y) = f(||x - y||^2)$ is a positive semidefinite kernel.*

Proposition 2.2.9. *The global alignment kernel induced by the squared Euclidean distance is positive semidefinite*

Proof. Starting from theorem 2.2.2, for any x, $y \in \mathbb{R}$

$$
\begin{aligned}
\frac{k}{k+1}(x, y) &= \frac{e^{-\frac{1}{2\sigma^2}||x-y||^2}}{1 + e^{-\frac{1}{2\sigma^2}||x-y||^2}} \\
&= \frac{1}{1 + e^{\frac{1}{2\sigma^2}||x-y||^2}} \\
&= \int_0^\infty e^{-t(e^{\frac{1}{2\sigma^2}||x-y||^2})} e^{-t} dt
\end{aligned}
$$

Now Bernstein's theorem implies that the function $f(||x - y||^2) = \frac{k}{k+1}(\sqrt{2\sigma^2}x, \sqrt{2\sigma^2}y)$ is a completely monotone function. This allows to conclude the proof by making use of 2.2.8 and from rescaling x and y. □

2.3 Signature of a path

The signature of a path is a way to encode the information of a path through an infinite sequence of iterated integrals. Its properties were first studied by Chen [38] and it was then used by Lyons in the context of stochastic analysis and controlled differential equations playing a pivotal role in rough paths theory [39]. In recent years signatures based machine learning methods have been successfully applied to a wide range of problems involving sequential data. For example by Gyurko and Lyons [40] for directional futures price prediction using order book data, Lemercier et al. [41] for distribution regression on sequential data and Yang et al.[42] for handwriting recognition with a recurrent neural network combined with signature features.

The initial steps toward formally defining the signature of a path involve introducing the specific class of paths that needs to be considered and establishing the definition of the tensor product.

Definition 2.3.1. Let (V, $||\cdot||_V$) be a normed space, $[s,t] \subset \mathbb{R}^+$ and $p \geq 1$ a real number. The p-variation of a path $x : [s,t] \to V$ is given by

$$||x||_{p-var,[s,t]} = \left(\sup_{\mathcal{D} \subset [s,t]} \sum_{t_i \in \mathcal{D}} ||x_{t_{i+1}} - x_{t_i}||_V^p \right)^{\frac{1}{p}} \tag{2.3.1}$$

Where the supremum is taken over all partitions of $\mathcal{D}$ of [s,t].

If $||x||_{p-var,[s,t]}$ is finite, then x is said to be a path of finite p-variation. If x has finite 1-variation, it is said to be a bounded variation function.
In the following part of this section the class of continuous paths of finite p-variation $x : [s,t] \to V$, with $[s,t] \subset [0,\infty)$ and V a Banach space, will be denoted by $C_p([s,t], V)$.

Definition 2.3.2. Let $V_1, V_2 ... V_n$ and Z be vector spaces, the tensor product of $V_1, V_2 ... V_n$ is a vector space $V_1 \otimes V_2 \otimes ... \otimes V_n$ with a bilinear map $\phi : V_1 \times V_2 \times ... \times V_n \to V_1 \otimes V_2 \otimes ... \otimes V_n$, that satisfies the universal property for which, for every bilinear map $f : V_1 \times V_2 \times ... \times V_n \to Z$ there exists a unique linear map $\tilde{f} : V_1 \otimes V_2 \otimes ... \otimes V_n \to Z$, such that $f = \tilde{f} \circ \phi$

Definition 2.3.3. Let $x \in C_p([s,t], V)$, with $1 \leq p < 2$, the k-fold iterated integral of x over $[s,t]$ is

$$S^k(x)_{s,t} = \int_{s<t_1<t_2<...<t_k<t} dx_{t_1} \otimes ... \otimes dx_{t_k} \in V^{\otimes k}$$

$V^{\otimes k} = \underbrace{V \otimes V \otimes V ... \otimes V}_{\text{k times}}$ and with the convention that $S(x)^0_{s,t} = 1$ and $V^{\otimes 0} = \mathbb{R}$.

The iterated integral of the path x can be understood in the sense of Young and therefore is well defined for any path of finite p-variation, $1 \leq p < 2$ (for a reference on Young integral, see chapter 1 of [43]).
As anticipated, the signature is an infinite sequence of these iterated integrals.

Definition 2.3.4 (Signature of a path)**.** Let $x \in C_p([s,t], V)$, with $1 \leq p < 2$, the signature of x over $[u,v] \subset [s,t]$ is given by the collection of iterated integrals

$$S(x)_{u,v} = (1, S^1(x)_{u,v}, ..., S^n(x)_{u,v}, ...) \in \prod_{k=0}^{\infty} V^{\otimes k}$$

It is possible to equip the space $\prod_{k=0}^{\infty} V^{\otimes k}$ with the operations of sum, product, and natural action of $\mathbb{R}$ defined in the following way: for any two elements $u = (u_0, u_1, ...)$ and $v = (v_0, v_1, ...)$ of $\prod_{k=0}^{\infty} V^{\otimes k}$ and $\lambda \in \mathbb{R}$ then

$$v + u := (u_0 + v_0, u_1 + v_1, ...)$$

$$u \otimes v := (w_0, w_1, ...) \text{ with } w_k = \sum_{i=0}^{k} u_i \otimes w_{k-i} \quad \text{for any } k \geq 0$$

$$\lambda u := (\lambda u_0, \lambda u_1, ...)$$

The space $\prod_{k=0}^{\infty} V^{\otimes k}$ endowed with these operations is a real non-commutative unital algebra with unit element $\mathbf{1} = (1, 0, 0, ...)$, and is denoted $T((V))$. The space $T((V))$ can be understood as the space of formal series of tensors.
Alongside $T((V))$, its subalgebra $T(V) = \bigoplus_{i=0}^{\infty} V^{\otimes i}$ is extremely relevant for the applications of

signatures to kernel methods described later in this section.
Since the signature can contain an infinite number non-zero elements, many practical applications that use them, a necessary step of "truncation" at some level k is required. For a given path $x \in C_p([s,t],V)$, $1 \leq p < 2$, the truncated signature is just the element of $T(V)$ defined as $S^{(n)}_{s,t}(x) = (1, S^1(x)_{s,t}, ..., S^k(x)_{s,t}, 0, 0, ...)$.
The next proposition allows to recover an upper bound on the approximation error committed when the signature is approximated by a truncated signature.

Lemma 2.3.5. *(Exponential decay) Let $x \in C_1([s,t],V)$ then the k-th term of the signature of x satisfies*

$$S^k(x)_{s,t} \leq \frac{||x||^k_{1-var}}{k!} \quad \textit{for every } k \in \mathbb{N}$$

Proof. See Proposition 2.2 in [43] □

This proposition implies that the approximation error committed when truncating the signature signature decays factorially.
Given that the signature is originally defined for continuous paths, when dealing with observed paths sampled at discrete time intervals, it is possible to concatenate the discrete points to form a continuous path. The resulting continuous path can be defined as follows:

Definition 2.3.6. Let $x : [r,s] \to V$ and $y : [s,t] \to V$ be two paths, the concatenated path $x * y : [r,t] \to V$ is defined as

$$(x * y)_u = \begin{cases} x_u & v \in [r,s] \\ y_u + x_s - y_s & v \in (s,t] \end{cases}$$

Another property of the signatures that makes them suited for sequential data is the ease of computation for concatenated paths, in fact, extending the result of Chen in [38], it is possible compute recursively the signature of the resulting path as the tensor product of the signatures of the concatenated paths.

Theorem 2.3.7. *Let $x \in C_1([r,s],V)$ and $y \in C_1([s,t],V)$ be two continuous paths, then $S(x * y)_{r,t} = S(x)_{r,s} \otimes S(y)_{s,t}$*

Proof. Fix $k \in \mathbb{N}$, then

$$\begin{aligned}
&S^k(x*y)_{r,u} = \\
&= \int_{s<u_1<u_2<...<u_k<t} d(x*y)_{u_1} \otimes ... \otimes d(x*y)_{u_k} \\
&= \sum_{i=0}^{k} \int_{s<u_1<u_2<...<u_i<t<u_{i+1}<..<u_k<t} d(x*y)_{u_1} \otimes ... \otimes d(x*y)_{u_k} \\
&= \sum_{i=0}^{k} \left(\int_{s<u_1<u_2<...<u_i<t} d(x*y)_{u_1} \otimes ... \otimes d(x*y)_{u_i} \right) \otimes \left(\int_{t<u_{i+1}<..<u_k<t} d(x*y)_{u_{i+1}} \otimes ... \otimes d(x*y)_{u_k} \right) \\
&= \sum_{i=0}^{k} \left(\int_{s<u_1<u_2<...<u_i<t} dx_{u_1} \otimes ... \otimes dx_{u_i} \right) \otimes \left(\int_{t<u_{i+1}<..<u_k<t} dy_{u_{i+1}} \otimes ... \otimes dy_{u_k} \right) \\
&= \sum_{i=0}^{k} S^i(y)_{r,s} \otimes S^{k-i}(x)_{s,t}
\end{aligned}$$

Where in the third step Fubini‘s theorem was used and the last step follows from the definition of the concatenation of paths. □

Furthermore, employing this proposition, it is possible to show that when linear interpolation is utilized to interpolate a time series, the signature can be expressed as a product of tensor exponentials.

Proposition 2.3.8. *Let x be a piece-wise linear path $x = \{(t_i, x_{t_i})\}_{t_i \in \mathcal{T}}$ with $\mathcal{T} = \{t_0 < ... < t_{n-1} < t_n\}$, then $S(x)_{s,t} = \exp_{\otimes}(x(t_1) - x(t_0)) \exp_{\otimes}(x(t_2) - x(t_1))... \exp_{\otimes}(x(t_n) - x(t_{n-1}))$ where $exp_{\otimes}(x) = \sum_{k=0}^{\infty} \frac{x^{\otimes k}}{k!}$*

Proof. Consider the fist segment the path, $x : [t_0, t_1] \to V$, then the k-th element of the signature is given by

$$\begin{aligned} S^k(x)_{t_0,t_1} &= \int_{t_0<u_1<...<u_k<t_1} \frac{(x_{t_1} - x_{t_0})^{\otimes k}}{(t_1 - t_0)^k} du_1...du_k \\ &= \frac{(x_{t_1} - x_{t_0})^{\otimes k}}{k!} \end{aligned}$$

Which concludes the proof □

Additionally the signature of a path is invariant for time reparametrizations. This characteristic is highly desirable in classification tasks that prioritize the sequential arrangement of path points rather than their specific time parametrization.

Lemma 2.3.9 (Reparameterization invariance)**.** *Consider a path $x \in C_p([s,t], V)$ with $1 \leq p < 2$, and a continuous non-decreasing surjection $\gamma : [u,v] \to [s,t]$ with $[u,v] \subset [0,\infty)$, then $S(x \circ \gamma)_{u,v} = S(x)_{s,t}$.*

Proof. See proposition 7.10 in [44] □

Definition 2.3.10. Let $\mathcal{T} = \{t_0 < ... < t_{n-1} < t_n\}$ and $x = \{(t_i, x_{t_i})\}_{t_i \in \mathcal{T}}$ be a path taking values in $\mathbb{R}^d$, then the lead lag transform of x is given by the process $\hat{x} : \mathcal{T} \to \mathbb{R}^{2d}$ defined as follows

$$\hat{x}_t = \begin{cases} (x_{t_i}, x_{t_{i-1}}) & t \neq t_0 \\ (x_{t_0}, x_{t_0}) & t = t_0 \end{cases}$$

This transformation, first introduced by Chevyrev and Kormilitzin [45], is referred to as the lead-lag transform and will be employed in the subsequent chapters. As shown in [45] (pg 25-27), the lead-lag transform contains direct information about the quadratic variation of a path, which holds particular significance for financial time series of prices.

2.3.1 The signature kernel

In this last part of this section it will be shown how the signature of a path can be utilized as a feature map to define a semipositive kernel that possesses all the properties described at the beginning of the chapter. The first step is to define a suitable tensor product on $T(V)$.

Definition 2.3.11. Let H_1 and H_2 be two Hilbert spaces equipped with the inner products $\langle \cdot, \cdot \rangle_{H_1}$ and $\langle \cdot, \cdot \rangle_{H_2}$ respectively. Then the Hilbert-Schmidt inner product between any $x_1 \otimes x_2$ and $y_1 \otimes y_2$ in the space $H_1 \otimes H_2$ is defined as

$$\langle x_1 \otimes x_2, y_1 \otimes y_2 \rangle_{H_1 \otimes H_2} = \langle x_1, y_1 \rangle_{H_1} \langle x_2, y_2 \rangle_{H_2}$$

It is worth noticing that the from the completion theorem, the space obtained by completing the space $H_1 \otimes H_2$ with respect to this inner product is again an Hilbert space.
Now, following [46], using the Hilbert-Schmidt product of Hilbert spaces, it the inner product for the subalgebra $T(V)$ is defined as follows.

Definition 2.3.12. Let V be a space endowed with a inner product, let u and v be elements of $T(V)$, then the inner product $\langle \cdot, \cdot \rangle_{T(V)}$ is defined as

$$\langle u, v \rangle_{T(V)} = \sum_{i=0}^{n} \langle u_k, v_k \rangle_{V^{\otimes k}}$$

Where $\langle \cdot, \cdot \rangle_{V^{\otimes k}}$ is the Hilbert-Schmidt product on $V^{\otimes k}$ and u_k indicates the k-th component of u.

Denote by $\overline{T(V)}$ Hilbert space obtained by completing the space $T(V)$ with respect to the inner product on $T(V)$. Then it follows from the factorial decay property of the signature that $S(x) \in \overline{T(V)}$ for every path $x \in C_1([s,t],V)$. In fact, from the exponential decay property

$$\begin{aligned}||S(x)_{s,t}||^2_{T(V)} &= \sum_{k=0}^{\infty} ||S^k(x)_{s,t}||^2_{V^{\otimes k}} \\ &\leq \sum_{k=0}^{\infty} \frac{||x||^{2k}_{1-var}}{k!} < \infty\end{aligned}$$

Leveraging this and the definition 2.3.12 it follows that the kernel defined as

$$\begin{aligned}K(x,y) &= \langle S(x)_{s,t}, S_{s,t}(y)\rangle_{T(V)} \\ &= \sum_{i=0}^{n} \langle S^k(x)_{s,t}, S^k(y)_{s,t}\rangle_{V^{\otimes k}}\end{aligned}$$

is positive definite for any $x, y \in C_1([s,t],V)$.
In fact, consider a collection of paths $\{x_1, ..., x_n\} \subset C_1([s,t],V)$, for any set of constants $\{c_1, ..., c_n\} \subset \mathbb{R}$, then

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} c_j c_i \langle S(x_i)_{s,t}, S(x_j)_{s,t}\rangle_{T(V)} &= \langle \sum_{i=1}^{n} c_i S(x_i)_{s,t}, \sum_{j=1}^{n} c_j S(x_j)_{s,t}\rangle_{T(V)} \\ &= \left|\left| \sum_{i=1}^{n} c_i S(x_i)_{s,t} \right|\right|^2_{T(V)} \geq 0\end{aligned}$$

In light of these results, as anticipated, the signature map can now be understood as a feature map, mapping paths from $C_1([s,t],V)$ to the Hilbert space $\overline{T(V)}$ and its associated kernel, being positive definite, enjoys all the properties presented at the beginning of this chapter.
Moreover, Cass et al. have shown in [47] that this signature kernel satisfies an hyperbolic partial differential equation, allowing the kernel to be computed as numerical solution of a PDE.

Theorem 2.3.13. *Let V be a Banach space endowed with an inner product and let $x \in C^1([a,b],V)$ and $y \in C^1([c,d],V)$. The signature kernel $k_{x,y}$ for every $s \in [a,b]$ and $t \in [c,d]$ is a solution of the following linear, second order, hyperbolic partial differential equation*

$$\frac{\partial k_{x,y}}{\partial x \partial y}(s,t) = \langle \dot{x}_s, \dot{y}_t\rangle_V \quad k_{x,y}(a,\cdot) = k_{x,y}(\cdot,b) = 1$$

Proof. See Theorem 2.5 in [47] □

Chapter 3

Directional movement prediction for corporate bond indices

3.1 Data description

3.1.1 Corporate bond indices

The data employed in this study to analyze weekly directional movements of duration hedged corporate bond indices consists of proprietary daily data. These data points were sampled at the close of each business day, spanning from July 10, 2010, to May 30, 2023. The dataframes are constructed from the industry-level 3 and industry-level 2 Bloomberg-Barclays US Investment Grade bond index (LUACTRUU Index), which collectively encompassed 7,658 distinct issues as of August 2023, with a total market value of 6,193 billion USD.
The criteria that regulate the composition of this index, which are detailed in [48] and [49] are the following:

- **Currency**: the index includes only bonds with principal and interest payments denominated in USD
- **Sector of the issuer**: the LUACTRUU index comprises bonds issued by companies in the industrial, utility, and financial sectors. This level of classification represents Industry Level 2. Subsequently, the sub-indices obtained through this aggregation are further categorized into two additional levels: Industry Level 3 and Industry Level 4. The table below outlines the first two levels of the Bloomberg Global Sector Classification Scheme (BCLASS), which serves as the standard used by the Bloomberg-Barclays index to classify bonds based on the issuer's sector.
- **Credit quality of the issuer**: measurement of the ability of the issuer to meet specific financial obligations. Credit ratings, which assess this ability, are provided by the following agencies: Standard & Poor's, Moody's Investor Services and Fitch IBCA. To qualify as an investment-grade bond, a bond must hold a credit rating higher or equivalent to BBB- for Standard & Poor's, Baa3 from Moody's, BBB from Fitch. If at least one of those agencies covers a particular bond issue, then the most conservative modal rating is used. In cases where bond-level ratings are unavailable, issuer-level ratings or expected ratings (ratings assigned before receiving the final documentation from the issuer) may be employed as alternatives.
- **Amount outstanding**: only issues that have an amount outstanding greater than 300mln USD are considered, this criterion is referred to as "liquidity" rule.
- **Time to maturity**: interval of time from the current date to the principal repayment date, only issues with at least one year of maturity are eligible to be included in the LUACTRUU index.
- **Market of issue and placement type**: that indicates whether a bond is publicly registered, exempt from registration or privately placed. It also indicates the type of investor the product is being marketed and sold to: local investors, non-local investors or globally offered. Only publicly registered issues are covered whilst there are no restriction regarding the type of investor the product is marketed to.
- **Taxability**: only fully taxable issues are included in the indices.

- **Level subordination** of the investor‘s claim within the capital structure of the issuer. Both senior and subordinated bonds are included in the corporate indices.
- **Coupon type**: Bonds that convert from fixed to floating rate, including fixed-to-float perpetual, will exit the index one year prior to conversion to floating-rate. Fixed-rate perpetuals are not included.
- **Embedded options**: both option-free bonds, putable and callable bonds are eligible to be included in the indices.

The eligible bonds participate to the index proportionally to their market value at the beginning of each month, when the index is rebalanced. Moreover, the value of the index takes into account intra-month cash flows from interest and principal payments which are not reinvested at a short-term reinvestment rate between rebalance dates but are reinvested in the index at the beginning of the next rebalancing period. According to the Bloomberg-Barclays methodology ([48]), the market value weighting facilitates the replication by investors as it reflects market capacity and liquidity of a particular issue.
Historically Industrials have represented 58% of the market value of the LUACTRUU, followed by Financial Institutions with 32% and Utility with 10% (source: [49]).
The forthcoming analysis will focus on specific sectors, namely the Industry level 2 Utility and Financial Services sectors, as well as all Industry level 3 sectors categorized under the broader Industrial classification at Industry level 2.

Industry level 1	Industry level 2	Industry level 3
Corporate	Industrial	Basic Industry
		Capital Goods
		Communications
		Consumer Cyclical
		Consumer Non-Cyclical
		Energy
		Technology
		Transportation
		Other Industrial
	Utility	Electric
		Natural Gas
		Other Utility
	Financial Institutions	Banking
		Brokerage, Asset Managers, Exchanges
		Finance Companies
		Insurance
		REITS
		Other Financial

Table 3.1: Industry level 1, Industry level 2 and Industry level 3 sectors in LUACTRUU index according to the BCLASS standard

3.1.2 Variables description

The variables used in this study are the following:

Daily total return

Total return is the standard measure of bond return, it is calculated as the sum of return from interest accrual and payments (coupon return), security price movements (price return) and scheduled and unscheduled payments of principal (paydown return). More precisely, considering a time interval [s,t]

$$\begin{aligned}
&\text{Price return} = \frac{\text{Price}_t - \text{Price}_s}{\text{Price}_s + \text{Accrued Interest}_s}\\
&\text{Coupon return} = \frac{\text{Accrued Interest}_t - \text{Accrued Interest}_s + \text{Interest Payment}}{\text{Price}_s + \text{Accrued Interest}_s}\\
&\text{Paydown Return} = \text{Principal Repayment} \times \frac{100 - \text{Price}_t - \text{Price}_s}{\text{Price}_s + \text{Accrued Interest}_s}
\end{aligned}$$

Where $\text{Price}_{(.)}$ is used to indicate the clean price, Interest Payment is the sum of all interest payments in the period [s,t] and Principal Repayment is actual principal payment expressed as a percentage of par divided by the par amount outstanding at the beginning of the period.
The total return for the index is computed as the weighted average of the total returns of the constituents, using as weight the market value of the security at the beginning of the period divided by the market value of the index at the beginning of the period.

Excess returns

The excess return of a bond serves as a basis for comparing the total return of a risky bond to the return of the treasury asset class. To achieve this, a portfolio of treasuries, that has a duration that matches perfectly the duration of the bond, is constructed at the start of each month. The exact methodology adopted to compute the duration hedging is discussed in great detail in the Appendix 3 of [48]. For time spans longer than a month, the common practice is to compute excess returns by compounding the total returns of the bond and comparing them to the compounded returns of a risk-free portfolio of treasuries. This approach is adopted because excess returns over a period, when compounded, may not accurately reflect the difference between the risky bond and the riskless portfolio over the longer period.
Similarly to the total return, the excess return for the index is computed as the weighted average of excess returns of the constituents, using the market value of the security at the beginning of the period divided by the market value of the index at the beginning of the period as weights.

US 2 year and 10 year Treasury yield

These variables allow to gauge changes in the slope of the yield curve. Historically, whenever the yield curve has inverted, that is when the yield of the 10Y treasury falls below the yield of the 2Y treasury , a recession followed (for a more in depth discussion see [50]). It is widely believed that the more pronounced the yield curve inversion, the higher the likelihood of an impending recession. As a result, this variable serves as a valuable macroeconomic indicator for the current phase of the business cycle, which can have a significant impact on some sectors. In fact, sectors like consumer cyclical and financial services tend to be highly sensitive to the current phase of the business cycle, on the other hand, defensive sectors such as utilities are less affected by extreme fluctuations.

Returns of the S&P500 index

The interaction between the stock and corporate bond markets is a well-documented phenomenon in the literature. In one of the most renowned works on this topic, as highlighted by Ilmanen in [51], the observed empirical lead-lag patterns between the stock market and bond markets are thoroughly examined. The interpretation suggests that a weak equity market can lead to the implementation of quantitative easing measures, that can result in an increase of bond prices. On the other hand, strong performance in the stock market is usually followed by lower returns in the bond market compared to other periods of the business cycle.
Given these characteristics, it is plausible to consider the returns of the S&P500 index as predictive indicator of sector performance in the corporate bond market.

Equity momentum

This variable measures the change in stock price of the parent company of the issuer over a period of time.
Numerous publications, including [52], [6], [7] have empirically demonstrated that equity momentum serves as a statistically significant predictor for forecasting the cross-section of future excess returns for corporate bonds.
This phenomenon is elaborated upon in [52] by Gebhardt, Hvidkjaer, and Swaminathan. In their work, the authors present evidence of momentum spillover from equities to investment-grade corporate bonds issued by the same firms. The proposed explanation for this phenomenon is tied to the trend observed in companies with strong equity momentum to exhibit credit rating improvements in the near future, leading to the anticipation of reduced default risk.
The equity momentum for a corporate bond over a period $[t-k,t]$ is calculated as

$$EquityMomentum_t(k) = \frac{ewm_{\ell,\alpha}(P_t)}{ewm_{\ell,\alpha}(P_{t-k})} \tag{3.1.1}$$

with $ewm_{\ell,\alpha}(P_t)$ being the exponential smoothing average of length ℓ for the price P_t of the parent's company stock at time t, with a smoothing factor of α.
The choice of smoothing the prices comes from a need to mitigate the microstructural effects that may affect this indicator in a non negligible way, especially when the momentum is calculated on a short time window.
The decision to smooth the prices arises from the necessity to mitigate the impact of microstructural effects, which can significantly influence this indicator, especially when calculating momentum over a short time window. In the calculation of sector equity momentum, each bond is initially associated with its corresponding equity parent (bonds without equity parents are excluded, although more than 90% of issuers have an equity parent). The equity momentum for each issuer is then computed based on its parent's equity performance. Subsequently, the sector's equity momentum is determined by taking a weighted average of the parent equity momenta. The weighting factors are based on the total outstanding amount for each issuer relative to the total outstanding amount within that sector at the beginning of each month.

Yield to worst

The yield to worst is a measure of the return an investor can realize on a callable bond. Assume a callable bond has the following redemption dates $\mathcal{T} = \{t_0 < t_1 < ... < t_{n-1}\}$, the yield to worst is the lowest return in the set of returns $\{yield_{t_i}\}_{t_i\in\mathcal{T}}$ where for all $i = 1, ..., n-1$ and value $yield_{t_i}$ is the return the investor would obtain if the bond is redeemed at time t_i, under the assumption that the issuer does not default before T. The yield to worst of a portfolios of bond is calculated by using a weighted average of yield to worst of the components, with weights equal to the proportion of the value of each bond with respect to the value of the portfolio.

Option Adjusted Spread

The Option Adjusted Spread (OAS) is the constant spread that needs to be added to the benchmark's yield curve to match the current market price of a bond with its discounted cash flows, considering any embedded options. Similarly to the spread, this metric provides a measure of the compensation an investor receives over a benchmark security for bearing credit risk, however, only the Option Adjusted Spread (OAS) enables the comparison of bonds or portfolios of bonds that may have distinct redemption structures.
More precisely, suppose the market is arbitrage-free and complete, take a callable bond with face value F that pays $\{C_1, ..., C_K\}$ at times $\mathcal{T} = \{t \leq t_1 < ... < t_K = T\}$ if the option is not exercised. Fix a filtered probability space $(\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{P})$ and denote the risk-free measure as $\mathbb{Q}$ (which exists given the hypothesis on the market), the risk-free short rate by r_t, and $\mathcal{T}_{[t,T]}$ as the set of all stopping times taking values in $\mathcal{T}$.
Then, the option adjusted spread is the value $\gamma \in \mathbb{R}$ such that the following equality between the market price $P_t(r)$ and the discounted cash flows holds

$$P_t(r) = \inf_{\tau\in\mathcal{T}_{[t,T]}} \sum_{k=1}^{K} \mathbb{E}_{\mathbb{Q}}\left[C_k \mathbb{1}_{\{\tau>t_k\}} e^{-\int_t^{t_k}(r_s+\gamma)ds} \middle| r_t = r\right] + F\mathbb{E}_{\mathbb{Q}}\left[\mathbb{1}_{\{\tau<T\}} e^{-\int_t^{T}(r_s+\gamma)ds} \middle| r_t = r\right]$$

The computation of the OAS is usually carried out using Montecarlo methods. There are many ways to compute the OAS for a portfolio of callable bonds, however, to ensure comparability between individual securities and portfolios the standard practice is to discount the aggregated payments of each security composing of the portfolio on every path.

Option Adjusted Duration

The Option Adjusted Duration (OAD) measures the sensitivity of the price of a bond to changes in interest rates, taking into account existing embedded options. This allows to compare the interest rate risk of bonds that may present different redemption structures.
More precisely, suppose all the hypothesis on the structure of the market and the bond made in the previous section about the OAS hold. For a callable bond with an OAS denoted as γ and a price at time t determined by the function $P_t(r, \gamma)$, where $P_t(r, \gamma)$ is assumed to be both positive and differentiable with respect to the interest rate r, the OAD of the bond at time t is defined as:

$$OAD = \frac{1}{P_t(r, \gamma)} \frac{\partial P_t(r, \gamma)}{\partial r}$$

Similarly to the, OAS, the computation of the OAD is usually carried out using Montecarlo methods and for a portfolio of bonds, the payments of each security composing of the portfolio are aggregated on every path during the simulation.

3.1.3 Data Processing

This section outlines the steps involved in creating four distinct dataframes, each intended for training different models to predict weekly directional movements of excess returns for the corporate bond sectors. In all versions of the dataframe, the short rate model utilized for calculating OAS and OAD is based on a lognormal model. The price is obtained by compounding total returns of the index. Additionally, for the calculation of momentum, the exponential smoothing process with sequence length of 10 observations and a smoothing factor of 0.3 is applied uniformly across all versions.
For the first dataframe, stores the term of a signature transform truncated at level 3, for the linearly interpolated time series in 3.1.2, computed using a 6 months daily rolling window. Before the signature transform is computed, each time series is first smoothed using a 10 day exponential smoothing average with smoothing factor of 0.3 and is then standardized.

$$\{5 \text{ day total returns}_t^{(lead-lag)}, \text{SPX10 day returns}_t, \frac{\text{USD2Y yield}_t}{\text{USD10Y yield}_t}\} \tag{3.1.2}$$

where t is measured in days and the lead-lag transform is computed according to the definition 2.3.10.
Additionally, for each day, the following variables are added to the dataframe

- 10 day Option Adjusted Spread percentage change
- 1, 3 and 6 months equity momentum
- 10 day yield to worst percentage change

The target for a date t is computed as the excess return of each the sector over the next 5 days.
This process results in 11 different dataframes (one for each sector) with 90 features each and will be used to train, validate and test the XGBoost model.
The second version, that will be used by to train logistic regression and SVM models, uses the signature transform truncated at level 3 and computed for 6 months daily rolling windows for each of the following time series, which are smoothed using a 10 days exponential smoothing average with smoothing factor 0.3

$$\{5 \text{ day total returns}_t^{(lead-lag)}, \text{SPX 10 day returns}_t, \frac{\text{USD2Y yield}_t}{\text{USD10Y yield}_t}\}$$

Similarly to the previous dataframe, for each day, the following variables are added to the dataframe

- 10 day Option Adjusted Spread percentage change

- 1, 3 and 6 months equity momentum
- 10 day yield to worst percentage change
- OAD

This process results in an additional 11 different dataframes, one for each sector, of 91 features each.
A third version, used to compute the Nystrom approximation of the global alignment kernel, contains a 6 months daily rolling window for the following time series

$$\{\text{5 day total returns}_t, \text{SPX 10 day returns}_t, \frac{\text{USD2Y yield}}{\text{USD10Y yield}_t}, \frac{\text{OAS}_t}{\text{OAS}_{t-10}}, \text{Equity Momentumt(1M)}_t,$$
$$\text{Equity Momentumt(3M)}_t, \text{Equity Momentumt(6M)}_t, \frac{\text{yield to worst}_t}{\text{yield to worst}_{t-10}}\}$$

where each 6 month window that constitutes the time series is standardized before computing the kernel.
Finally, the last version, used to compute the signature kernel, with the Python package [47], is constituted by the standardized time series of

$$\{\text{5 day total returns}_t^{(lead-lag)}, \text{SPX 10 day returns}_t, \frac{\text{USD2Y yield}_t}{\text{USD10Y yield}_t}, \frac{\text{OAS}_t}{\text{OAS}_{t-10}}, \text{Equity Momentumt(1M)}_t,$$
$$\text{Equity Momentumt(3M)}_t, \text{Equity Momentumt(6M)}_t, \frac{\text{yield to worst}_t}{\text{yield to worst}_{t-10}}\}$$

3.2 Performance metrics

Confusion matrix

In a supervised binary classification task, the confusion matrix is a 2x2 matrix that records the value of true positives, false positives, true negatives and false negatives. The table below illustrates the structure of this table for the current classification problem

	Actual positive return	Actual negative return
Predicted positive return	True positive (TP)	False positive (FP)
Predicted negative return	False negative (FN)	True negative (TN)

Table 3.2: Confusion matrix matrix for directional predictions

A false negative is also known as a type I error, a false positive as a type II error. Related to these errors are the notions of sensitivity and precision.
The sensitivity of a classifier is an estimate of how likely prediction of the negative class is correct.

$$\text{Sensitivity} = \frac{TN}{FN + TN}$$

The precision of a classifier is an estimate of how likely the prediction of the positive class is correct.

$$\text{Precision} = \frac{TP}{FP + TP}$$

Accuracy score

The accuracy score the most popular metric used in classification tasks and corresponds to the percentage of correct predictions. The accuracy score is given by the formula:

$$\text{Accuracy score} = \frac{TP + TN}{TP + TN + FP + FN}$$

Balanced accuracy score

The balanced accuracy is defined as the average between the sensitivity and the specificity. By taking into account both of these measures it can be a more accurate measure of performance in the case that the dataset is unbalanced.

$$\text{Balanced Accuracy Score} = \frac{1}{2}\left(\frac{TP}{FP+TP} + \frac{TN}{FN+TN}\right)$$

Matthews correlation coefficient

The Matthew coefficient is a correlation coefficient between the observed and predicted classifications

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

This score ranges from [-1, 1], the value -1 is reached in the case of perfect missclassification and the value +1 in the case of perfect classification, if the MCC = 0 then the classification method has the same expected score as a completely random classifier.
Like the previous score, the Matthews Accuracy score is a robust accuracy score for unbalanced dataframes.

Distribution of returns

To achieve a better evaluation of the classifier's performance on the test set, it is essential not only to calculate the prediction accuracy but also to examine the distribution of returns generated by following the classifier's predictions. Specifically, this involves analyzing the returns distribution of a strategy in which the index is either bought when the classifier predicts positive excess returns or sold short when the classifier predicts negative returns.
Since the purpose of the classifier is not to actively define a strategy but to help with the timing of market activity for an existing long term strategy and provide insight on portfolio allocation choices, the following simplifications are not restricting:

- There are no transaction costs
- It is possible to borrow capital at a zero interest rate
- It is possible to short sell the bonds composing the indices
- It is possible to trade the bonds that compose the indices at every time, without liquidity restrictions.

The metrics that will be used to compare the different distributions of returns obtained by the different classifiers are: annualized average excess return, annualized standard deviation of the excess returns, skewness of the distribution and finally the information ratio.

3.3 Sliding window cross validation

The method used to validate the different models is the sliding window cross validationn, which is a validation model specifically designed for the validation of machine learning techniques for time series.
Given a fixed a training window length L, the data $\{x_1, ..., x_n\}$ is divided in contiguous blocks of length k, starting from x_L. The model is initially trained on the first L data points and validated on the first block, then the windows index shifts by k and the model is updated (or retrained in the case of logistic regression and SVM) using the data in the current window while being validated on the successive block. This process iterates until the final block is reached. The sliding window cross validation procedure is repeated to identify hyper parameters that yield the lowest average loss across the validation blocks without overfitting the training data.

Sliding window cross validation offers notable benefits compared to conventional validation methods. Its key advantage lies in its consideration of temporal dependencies within time series data. By employing sequential blocks for training and validation, this technique ensures that evaluations are conducted on sequences similar to those encountered in when the model is applied. Moreover sliding window cross validation provides insight into how well a model generalizes across different segments of the time series, allowing to compute the average and the standard deviation of the accuracy score across time. This is crucial for avoiding overfitting and ensuring that the model's predictions are reliable for unseen data.

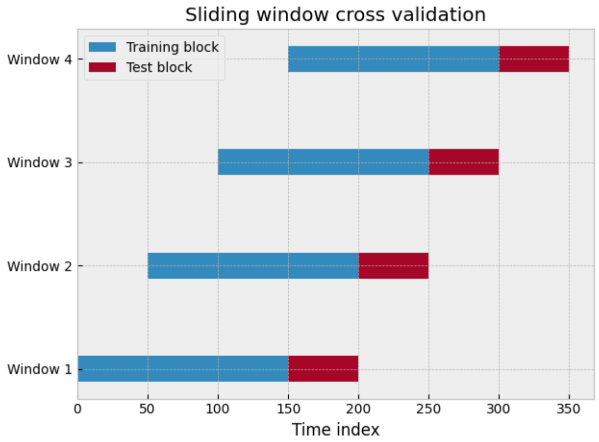


Figure 3.1: Sliding window cross validation with four validation blocks

3.4 Results

The dataset is divided into three subsets: training and validation sets together comprise 70% of the entire data span, ranging from 10-07-2010 to 20-10-2019. The remaining 30% is allocated to the test set, covering the period from 20-10-2019 to 30-05-2023. Each of the models undergoes a grid search, followed by a manual fine-tuning process, aimed at identifying the best set of hyperparameters.
Specifically, XGBoost is fine-tuned to minimize binary cross-entropy and maximize accuracy, while SVM and logistic regression are optimized to find a compromise between maximizing accuracy, MCC and balanced accuracy.
Once this is achieved, the model is tested and the obtained performance scores for the out of sample data are compared to the performance scores obtained for the validation data.
Moreover, using the predictions obtained in this last step, the annualized excess returns, annualized standard deviation, skewness and information ratio are calculated as described in 3.2 and compared to the annualized excess returns, annualized standard deviation, skewness and information ratio obtained from a buy and hold strategy to further assess the performance of the model.
The following sections describe the results for each of the trained model.

3.4.1 XGBoost model

The elevent XGBoost models are incrementally trained using sliding windows comprising 150, 200 or 250 observations. The validation and test blocks are contiguous blocks of 12 data points, with shifts of 28 points between successive validation blocks.
The grid search is performed separately for every sector on the following set of hyperparameters

hyperparameter	values
number of trees	{100, 150, 250}
learning rate	{0.01, 0.05, 0.1, 0.2, 0.3}
max depth	{3, 4, 5}
column sample by tree	{0.8, 0.9}
column sample by level	{0.8, 0.9}
L1 regularization constant	{0.1, 0.3, 0.5}
L2 regularization constant	{0.1, 0.3, 0.5}

Table 3.3: XGBoost model hyperameters grid

The XGBoost models achieved an average accuracy of 55.3% on the validation set, with an average collective accuracy standard deviation of 17% . The sectors where this type of classifier showed the best performance in capturing directional movements are the Utility and Basic Industry sectors, achieving an accuracy of 59.1%. In contrast, the classifier exhibited lower accuracy in the Capital Goods sector, with a performance of 52.5%. The balanced accuracy values in the validation set indicate that the XGBoost classifiers are equally proficient at predicting both upward and downward movements. This finding is further supported by the MCC scores.
In the test set, the average accuracy stands at 56.3%, accompanied by an average standard deviation of 17%. Notably, the sector where the XGBoost demonstrates a higher proficiency in capturing directional movements is Capital Goods, achieving an accuracy of 63.5%. On the other hand, the sector with the lowest performance in terms of accuracy is Consumer Cyclical, with a score of 52%. The table below details the values for the scores for the different XGBoost classifiers.

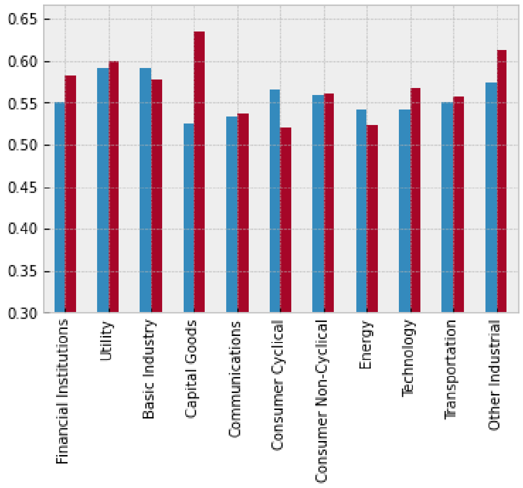


Figure 3.2: Accuracy scores for the XGBoost model for the validation set (in blue) and for the test set (in red)

Following the models' recommendation results in average excess returns that are consistently higher than the returns obtained through a buy and hold strategy, while maintaining a similar standard deviation. However, by examining the plots in the Appendix A.4, for the Consumer Cyclical, Consumer Non-Cyclical and Other Industrial sectors, there are periods where the excess returns obtained by the model are lower than returns obtained through a Buy and Hold strategy. Conversely, for the remaining eight sectors the XGBoost models consistently obtains a higher excess returns compared to the Buy and Hold stategy.
The best results are obtained for the Energy sector (9.33% annualized average excess returns) followed by Technology with an improvement of +6.17% in terms of annualized averaged excess returns. The smallest improvement in excess returns over the benchmark strategy, with an annualized average excess return increase of +0.47% per year, are seen in the Consumer Cyclical and Capital Goods sectors.
With the exceptions of the Capital Goods and the Other Industrial sector, the returns distribution distribution typically exhibits a rightward skew compared to the returns of the Buy and Hold strategy. This suggests that the models efficaciously identifies outliers in the distribution of each sector.

	XGBoost Classifier				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	1.73	6.04	-0.93	0.29	0.52	6.04	-1.34	0.09
Utility	4.22	7.50	-1.51	0.565	0.32	7.47	-2.52	0.043
Basic Industry	2.46	7.30	-1.66	0.338	1.21	7.28	-2.34	0.163
Capital Goods	1.56	6.66	-2.42	0.234	1.09	6.67	-2.37	0.163
Communications	4.04	8.44	2.21	0.371	0.72	8.46	-0.38	0.085
Consumer Cyclical	0.88	7.28	1.63	0.128	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	1.82	6.88	-0.643	0.264	1.07	6.89	-0.580	0.155
Energy	9.46	11.1	0.917	0.851	0.13	11.2	-3.84	0.011
Technology	7.04	5.82	3.05	1.21	0.87	5.92	-0.468	0.147
Transportation	3.92	8.18	-0.978	0.479	0.92	8.20	-1.24	0.113
Other Industrial	1.67	6.99	-5.27	0.240	0.032	6.99	-5.27	0.005

Table 3.4: Comparison between the XGBoost classifier and Buy and Hold average annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

3.4.2 Support Vector Machines

The Support Vector Machine models are trained on windows of 50, 60, or 80 data points, and the window size is chosen independently to better adapt to each sector. Validation and testing are performed on blocks of 12 data points. In a sliding window cross-validation setup, these windows are shifted at intervals of 28 days. The initial choice of the positive real value $\frac{1}{C}$, which determines the regularization parameter C for each model, is independently searched for each model version from the following list: $\{0.001, 0.01, 0.05, 0.07, 0.1, 0.3, 0.5, 0.7, 1, 5, 10, 20, 50, 100\}$. Subsequently, manual selection is employed to further optimize the value of C.

SVM + signatures

The first set of SVM models is based on the second version of the dataframes described in the previous section. These models, each tailored to a specific sector, achieve an average accuracy of 53.9%, with an accuracy standard deviation of 18.2%.
On the validation set, the models that demonstrate the highest accuracy are those predicting directional movements in the Capital Goods and Financial Institutions sectors, achieving scores of 56.3% and 55.3%, respectively. Conversely, the model's accuracy in predicting directional movements for the Other Industrial sector is the lowest, at 51.3%. The consistently positive MCC score indicates that each model consistently outperforms a random classifier. Additionally, the average balanced accuracy score of 52.9% hints at the models' consistent performance in both upward and downward movement classifications.
The models collectively achieve an average performance of 56.3% in the test set. The top performing sectors are Capital Goods and Utility, with a respective accuracy of 63.5% and 59.2% . The accuracy standard deviation on the test is is comparable to the accuracy standard deviation on the validation set, with an average of 17.1%.
Furthermore, the choice of kernel results in an average MCC score of 5.1% and a balanced accuracy of 54.9%. These metrics reaffirm the same conclusions drawn from the training set, highlighting the models' consistent performance.

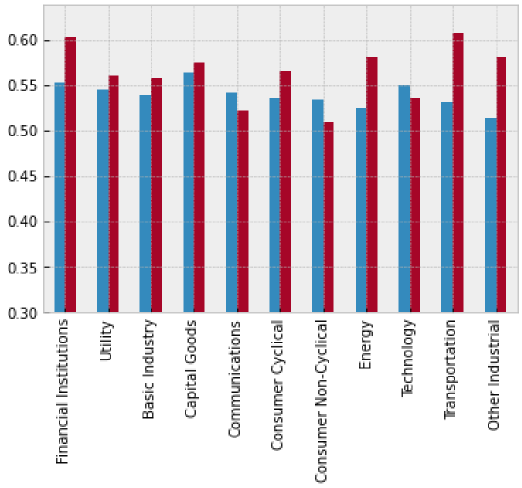


Figure 3.3: Accuracy scores for the SVM+ signature model for the validation set (in blue) and for the test set (in red)

The annualized excess returns obtained following the prediction of the SVM model are consistently higher than the ones obtained through the Buy and Hold strategy, all while maintaining a comparable annualized standard deviation. This results in a higher annualized information ratio. Notably, the distribution of excess returns skews to the right compared to the distribution of the Buy and Hold strategy.
Among the sectors, Basic Industry records the best improvement over the benchmark strategy, with an information ratio of 1.133 compared to the benchmark's 0.163. Additionally, the models associated to the Energy, Utility and Other Industrial sectors also show noticeable improvement of returns over the benchmark strategy. On the other hand, the Consumer Non-Cyclical sector records the lowest improvement over the benchmark strategy, with the SVM predictor achieving an average annualized excess return of 1.20 compared to the 1.07. It is worth noticing that the models associated to Consumer Cyclical and Consumer Non-Cyclical and the Transportation sector exhibit periods were the excess returns obtained by following the models' recommendations are lower than a simple Buy and Hold strategy.

	SVM + signature				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	3.71	6.02	1.05	0.616	0.52	6.04	-1.34	0.09
Utility	4.13	7.47	3.60	0.578	0.32	7.47	-2.52	0.043
Basic Industry	8.18	7.30	2.71	1.133	1.21	7.28	-2.34	0.163
Capital Goods	1.68	6.66	-2.33	0.253	1.09	6.67	-2.37	0.163
Communications	3.29	8.44	2.11	0.390	0.72	8.46	-0.38	0.085
Consumer Cyclical	2.92	7.27	-0.43	0.401	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	1.20	6.89	-0.518	0.174	1.07	6.89	-0.580	0.155
Energy	5.78	11.2	-0.097	0.518	0.13	11.2	-3.84	0.011
Technology	1.06	5.92	1.90	0.180	0.87	5.92	-0.468	0.147
Transportation	4.90	8.18	-0.257	0.599	0.92	8.20	-1.24	0.113
Other Industrial	3.64	6.95	-2.99	0.524	0.032	6.99	-5.27	0.005

Table 3.5: Comparison between the SVM + signature classifier and Buy and Hold average annualized excess returns, annualized standard deviation of excess returns, skewness of the excess returns and annualized information ratio

SVM+sig kernel

The second version of the SVM models, utilizing the signature kernel, demonstrates a collective average accuracy of 53.1%, which is lower than that of the previous set of models. Furthermore, the models show a comparatively higher average standard deviation in accuracy, standing at 20% In this case, the sectors achieving the highest accuracy in the validation set are Financial Services (57.2%) and Other Industrial (54.7%), while Energy lags behind with the lowest accuracy, 50.2%. The MCC score stands at 9.8%, and the average balanced accuracy score is 50.6%. These metrics suggest a relatively lower performance for this model compared to the previous version.
Consistent with the previously discussed models, the average accuracy in the test set surpasses that of the validation set, reaching 53.6%. Notably, the standard deviation for accuracy is lower in the test set, measuring at 18.4%. However, it remains higher than the collective average observed for the SVM+signature model Surprisingly, the model achieves its highest accuracy in the Energy sector (58.5%) followed by Transportation (56.8%). In contrast, the sector with the lowest accuracy is Communication, recording a score of 50.3%
The average MCC score and balanced accuracy are on average 2.6% and 51.6%.

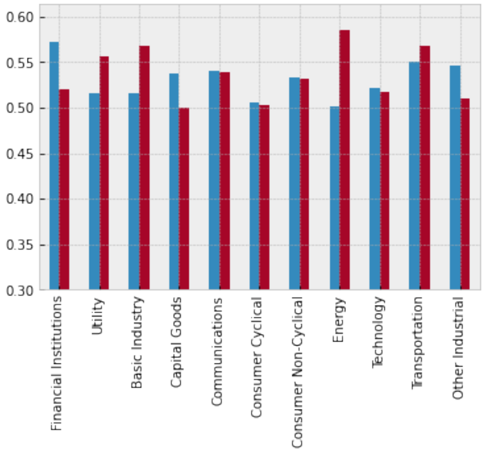


Figure 3.4: Accuracy scores for the SVM+sig kernel model for the validation set (in blue) and for the test set (in red)

The SVM + signature kernel consistently outperforms the Buy and Hold strategy, while incurring in a similar amount or lower volatility for every sector. This consequently implies a better annualized information ratio.
By examining at the plots in Appendix A.4, it is possible to notice how this model achieves the best performance across all models for the Transportation sector where the recorded information ratio is 0.734. Additionally, it remains highly competitive for the Other Industrial sector with an information ratio of 0.640. However, it is worth noticing that, for Financial Institutions, Consumer Cyclical and Consumer Non-Cyclical, the excess returns obtained by the SVM+ sig kernel model are lower than the returns obtained by the Buy and Hold strategy during certain periods.

	SVM + sig kernel				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	0.71	6.04	-1.15	0.117	0.52	6.04	-1.34	0.09
Utility	3.46	7.47	2.32	0.464	0.32	7.47	-2.52	0.043
Basic Industry	2.25	5.78	-0.80	0.390	1.21	7.28	-2.34	0.163
Capital Goods	2.94	6.67	-2.43	0.442	1.09	6.67	-2.37	0.163
Communications	3.38	7.30	0.007	0.463	0.72	8.46	-0.38	0.085
Consumer Cyclical	1.52	7.28	-1.81	0.210	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	1.10	6.71	0.560	0.164	1.07	6.89	-0.580	0.155
Energy	7.29	11.1	3.76	0.655	0.13	11.2	-3.84	0.011
Technology	1.90	5.78	0.285	0.329	0.87	5.92	-0.468	0.147
Transportation	5.99	8.15	0.970	0.734	0.92	8.20	-1.24	0.113
Other Industrial	4.39	6.87	5.30	0.640	0.032	6.99	-5.27	0.005

Table 3.6: Comparison between the SVM + sig kernel classifier and Buy and Hold average annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

SVM+Global Alignment Kernel

The SVM with Global Alignment Kernel (SVM+GAK) achieves the best average accuracy score across sectors on the validation set among all the trained models versions, reaching 55.6%. However, it exhibits the highest average accuracy standard deviation among the SVM models, standing at 21%.

The MCC scores and balanced accuracy score are recorded at 50.4% and 1.2%, respectively. Notably, for many of the models, the MCC score is negative, indicating that in some cases, some model associated to the distinct sectors perform worse than a random classifier. In contrast to the previously discussed models, the SVM+GAK model exhibits a lower average accuracy in the test set compared to the validation set (53.4% compared to 55.6%). This decline in accuracy is accompanied by a decrease in the standard deviation of accuracy, which has an average 19.8% across sectors. The sectors where the models achieves the model is the most accurate in predicting directional movements are Energy and Capital Goods, with an accuracy of 58.0% and 57.3% respectively, Technology achieves the worst performance with just 50.0% accuracy.

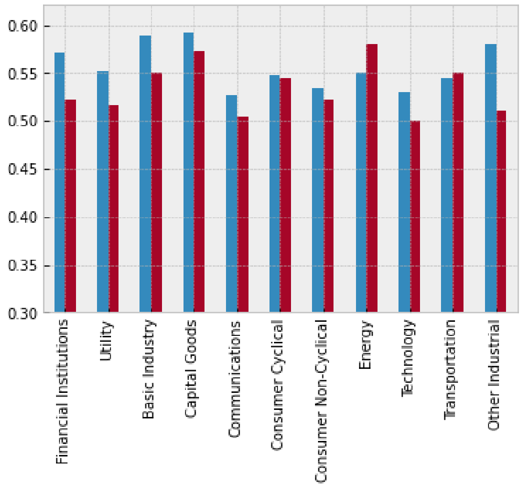


Figure 3.5: Accuracy scores for the SVM+GAK model for the validation set (in blue) and for the test set (in red)

Despite achieving strong accuracy results, a majority of the MCC scores are negative, and the balanced accuracy scores deviate significantly from the overall accuracy. This suggests that the SVM+GAK model struggles to equally recognize both upward and downward movements for most sectors.

These findings are reinforced by the plots in A.4, which clearly illustrate that the SVM+GAK model predominantly predicts upward movements in the majority of the test set. Furthermore, the average annualized excess returns consistently fall short of those obtained through a Buy and Hold strategy. This indicates that despite its high accuracy, the model fails to capture returns effectively, making it unusable for the task at hand.

	SVM + GAK				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	-1.31	6.04	-1.20	-0.217	0.52	6.04	-1.34	0.09
Utility	-1.22	7.50	-2.37	-0.164	0.32	7.47	-2.52	0.043
Basic Industry	-0.44	7.22	-2.25	-0.061	1.21	7.28	-2.34	0.163
Capital Goods	0.14	6.65	-2.32	0.021	1.09	6.67	-2.37	0.163
Communications	-0.88	8.45	-0.225	-0.105	0.72	8.46	-0.38	0.085
Consumer Cyclical	-1.14	7.28	-1.64	-0.156	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	-0.86	6.89	-0.393	-0.125	1.07	6.89	-0.580	0.155
Energy	0.40	11.2	-3.75	0.036	0.13	11.2	-3.84	0.011
Technology	-1.15	5.91	-0.283	-0.195	0.87	5.92	-0.468	0.147
Transportation	-0.71	8.20	-1.08	-0.086	0.92	8.20	-1.24	0.113
Other Industrial	-2.06	6.98	-5.17	-0.296	0.032	6.99	-5.27	0.005

Table 3.7: Comparison between the SVM + GAK classifier and Buy and Hold average annualized excess returns, annualized standard deviation of excess returns, skewness of the excess returns and annualized information ratio

3.4.3 Logistic regression

The Logistic regression models are trained on a window of 80, 100, or 130 points, with the choice of window size tailored independently for each model to better suit the characteristics of each sector. These models are then validated and tested on blocks of 12 points, with shifts of 28 days between successive windows in the sliding window cross-validation setup.

Following an analoguous methodology as the one followed for the tuning of the SVM model, the initial choice of the positive real value $\frac{1}{C}$, which determines the L^2 regularization parameter C for each model, is independently searched for each model version from the following list: $\{0.001, 0.005, 0.01, 0.05, 0.07, 0.1, 0.3, 0.5, 0.7, 1, 5, 10, 20, 50, 100, 150, 200\}$. Subsequently, manual selection is employed to further optimize the value of C.

logit+signature

The first model, a logistic regression + signature, attains an average accuracy of 53.5% on the validation set, with a accuracy standard deviation that collectively averages 19% across the different sectors. The best-performing models are found in the Financial Institutions, Transportation, and Basic Industry sectors, achieving accuracies of 56.4%, 55.0%, and 55.0%, respectively. The lowest accuracy is obtained for the model predicting the movements of the Energy sector, with a score of 51%.

The average MCC score is 0.057 and the average balanced accuracy is 55%, suggesting that the performance of the models is consistent when predicting both movement directions.

In the test set the model has a collective average accuracy of 56.2% with an accuracy standard deviation across sectors of 18.8%. The best performing models are the one predicting movements in the Transportation and Financial Services sectors, achieving accuracies of 60.8% and 60.2%, respectively. The model with the least accuracy is the one associated to the Consumer Non cyclical sector, with a score of 50.9%. The balanced accuracy and MCC scores confirm the models' good performance across most of the sectors, with an average of 54.9% and 9.9%, respectively

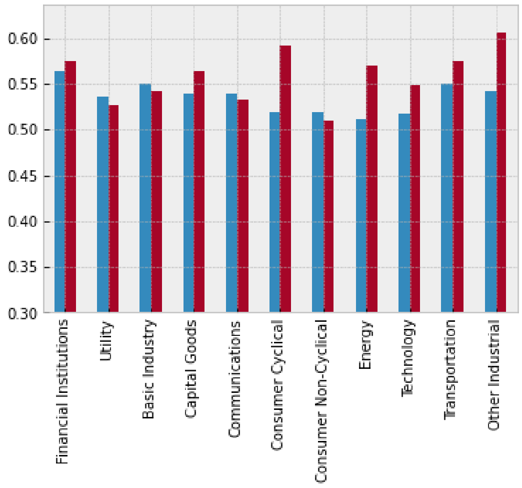


Figure 3.6: Accuracy scores for the logit+signature model for the validation set (in blue) and for the test set (in red)

The models consistently outperforms the benchmark strategy, yielding higher annualized excess returns and information ratios. As observed in previous models, the skewness of excess returns from the models' directional predictions is greater than the skewness of the Buy and Hold strategy. The models that exhibit substantial performance improvements are the one for the Basic Industry (+5.99% annualized averaged excess returns), Communications (+4.39%), and Other Industrial (+4.92%) sectors. Conversely, the Consumer Non-Cyclical sector shows the most consistent performance improvements, albeit modest, with a +0.43% annualized averaged excess return. Furthermore, the model maintains a standard deviation comparable to the benchmark strategy, resulting in consistently superior information ratios compared to the benchmark strategy. Furthermore, the logit+signature model stands out as the top performer in terms of excess returns for both the

Consumer Cyclical and Consumer Non-Cyclical sectors. These two sectors are characterized by extended periods during which all other models consistently yield lower excess returns compared to the benchmark strategy.

	logit + signature				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	3.27	6.03	1.07	0.544	0.52	6.04	-1.34	0.09
Utility	2.35	7.49	3.34	0.315	0.32	7.47	-2.52	0.043
Basic Industry	4.87	7.24	2.73	1.061	1.21	7.28	-2.34	0.163
Capital Goods	1.52	6.66	1.30	0.227	1.09	6.67	-2.37	0.163
Communications	5.70	8.42	2.10	0.667	0.72	8.46	-0.38	0.085
Consumer Cyclical	3.62	7.26	-0.468	0.499	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	1.63	6.88	2.280	0.236	1.07	6.89	-0.580	0.155
Energy	0.99	11.2	-0.371	0.088	0.13	11.2	-3.84	0.011
Technology	1.85	5.91	1.87	0.312	0.87	5.92	-0.468	0.147
Transportation	3.49	8.20	-0.177	0.426	0.92	8.20	-1.24	0.113
Other Industrial	3.64	6.95	-3.00	0.524	0.032	6.99	-5.27	0.005

Table 3.8: Comparison between the logit + signature classifier and Buy and Hold average annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

Logit+signature kernel

Contrary to the SVM + signature kernel models, the logistic regression + signature kernel models fail to consistently obtain a satisfactory accuracy in the test set. For instance, it fails to surpass a 50% accuracy score in the test set for the Technology and Consumer Non-Cyclical sectors.
In the validation set, the best performing sectors are Basic Industry and Consumer cyclical, achieving 58% and 54.7% accuracy respectively.
The MCC score, with an average of -2.2%, and the balanced accuracy score, averaging 49.1%, confirm that collectively the models struggle to capture sector movements even in the validation set.
The good performance in the Basic Industry and Consumer Cyclical sectors observed in the validation set carries over to the test set, with respective accuracy scores of 59.7% and 57.9%. The standard deviation in the test set falls within the range of values seen in other models analyzed so far, at 19.7%. Unlike the training set, the balanced accuracy and MCC score values in the test set suggest strong performance in all sectors achieving an accuracy greater than 50%.

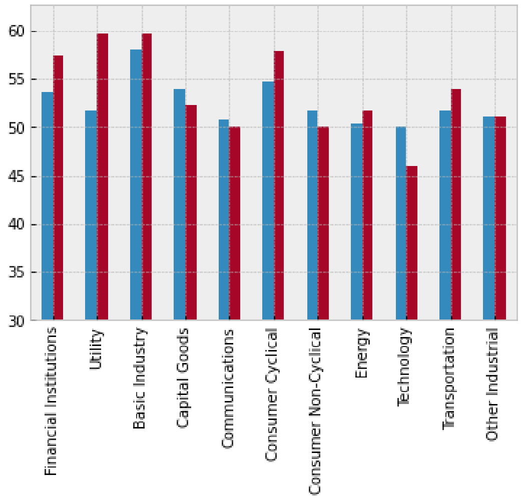


Figure 3.7: Accuracy scores for the logit+sig kernel model for the validation set (in blue) and for the test set (in red)

Considering the results obtained for the accuracy of the model, it's not surprising to see that the model associated to Consumer Non-Cyclical obtains negative excess returns over the testing period. However, the model for the Transportation sector, despite the good out of sample performance, fails to capture efficaciously the excess returns, which when annualized amount to -3.50%. Conversely, the Technology sector, despite being the worst performer in terms of out-of-sample accuracy, manages to achieve +2.23% annualized excess returns. Additionally, the logit + sig kernel models are the best models for Financial Institutions, Capital Goods and the Energy sectors.

	logit + sig kernel				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	5.24	6.00	1.84	0.874	0.52	6.04	-1.34	0.09
Utility	3.58	7.48	3.22	0.479	0.32	7.47	-2.52	0.043
Basic Industry	2.12	7.28	2.43	0.668	1.21	7.28	-2.34	0.163
Capital Goods	2.74	6.67	2.91	-0.020	1.09	6.67	-2.37	0.163
Communications	1.80	8.45	-0.225	-0.213	0.72	8.46	-0.38	0.085
Consumer Cyclical	0.89	7.28	-1.78	0.123	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	-3.84	6.88	0.49	-0.125	1.07	6.89	-0.580	0.155
Energy	18.51	11.2	4.13	1.68	0.13	11.2	-3.84	0.011
Technology	2.23	5.91	1.20	0.378	0.87	5.92	-0.468	0.147
Transportation	-3.50	8.20	-1.07	-0.427	0.92	8.20	-1.24	0.113
Other Industrial	2.75	6.97	4.98	0.394	0.032	6.99	-5.27	0.005

Table 3.9: Comparison between the logit + sig kernel classifier and Buy and Hold average annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

Logit+Global Alignment Kernel

The last set of models, logistic regression + Global Alignment Kernel (logit+GAK), achieve a collective accuracy of 55.5% on the validation set. Among sectors, Basic Industry and Transportation record the highest accuracy, at 58.9% and 58.3%, respectively, while Communication performs the worst with just 50.8% accuracy. The average standard deviation for the accuracy score in the training set is 21%, indicating variability in performance across sectors. However, despite the relatively high accuracy score, the balanced accuracy, which averages 50%, suggests that the model does not predict upward and downward movements with equal accuracy.

In the test set, the models achieve a collective accuracy of 55.2%, with a standard deviation of 19.4%. Transportation and Basic Industry continue to be the sectors for which the model achieves the highest accuracy, both at 58.5%, while the lowest accuracy is obtained for Technology and Communications, both at 51.7%. Similarly to the validation set, the MCC scores for the test data are mostly negative, indicating that despite the high accuracy, the model does not consistently outperform a random classifier for many of the covered sectors.

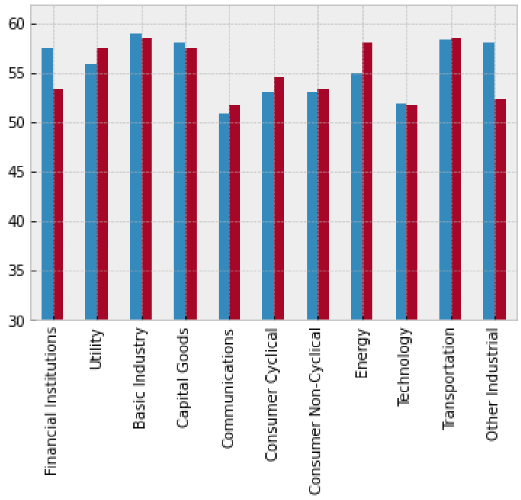


Figure 3.8: Accuracy scores for the logit+GAK model for the validation set (in blue) and for the test set (in red)

Despite achieving results for performance metrics that are consistent with the other models, the logit+GAK model fails to outperform the buy and hold strategy in terms of annualized average excess return.

By examining the plots in A.4, it becomes apparent that the models' prediction largely favor upward movements for the indices throughout most of the period. This observation aligns with the findings from the MCC score and the balanced accuracy, which suggested that the model did not

consistently outperform a random classifier.
Of particular note is the unexpected result for the Transportation sector. Despite the model achieving an accuracy higher than 58% on both the validation and test sets, it fails to capture returns effectively, resulting in annualized average excess returns that do not surpass the Buy and Hold strategy.

	logit+GAK				Buy & Hold			
	avg excess returns (%)	std (%)	skewness	information ratio	avg excess returns (%)	std (%)	skewness	information ratio
Financial Services	-1.74	6.04	1.84	-0.289	0.52	6.04	-1.34	0.09
Utility	-2.32	7.49	-2.43	-0.310	0.32	7.47	-2.52	0.043
Basic Industry	1.19	7.30	-2.35	0.163	1.21	7.28	-2.34	0.163
Capital Goods	-0.13	6.66	-2.30	0.413	1.09	6.67	-2.37	0.163
Communications	-0.89	8.45	-0.225	-0.105	0.72	8.46	-0.38	0.085
Consumer Cyclical	-1.04	7.28	-1.69	-0.143	0.41	7.28	-1.77	0.056
Consumer Non-Cyclical	-0.86	6.89	-0.393	-0.558	1.07	6.89	-0.580	0.155
Energy	-0.427	11.0	-3.82	-0.038	0.13	11.2	-3.84	0.011
Technology	0.85	5.92	-0.358	0.144	0.87	5.92	-0.468	0.147
Transportation	-0.71	8.20	-0.985	-0.086	0.92	8.20	-1.24	0.113
Other Industrial	-1.72	6.97	-5.17	-0.246	0.032	6.99	-5.27	0.005

Table 3.10: Comparison between the logit + GAK classifier and Buy and Hold average annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

Chapter 4

Conclusion

This thesis delves into the weekly prediction of movements in duration-hedged corporate bond sector indices. This is achieved through different models: XGBoost, Support Vector Machine and logistic regression. For the last two models application of global alignment and signature kernels to the Support Vector Machine and the logistic regression model are explored and compared.

The initial sections of this thesis introduce and carefully analyze these models, highlighting their properties and strengths. Subsequently, the focus shifts to a comprehensive exploration of kernel methods, encompassing their properties. Within this context, special attention is given to two examples of kernel designed for the application to sequential data: of the global alignment kernel and the signature kernel, both of which are examined in detail.

Moving forward, the thesis presents the empirical data and conducts a systematic evaluation of the diverse models' performances. The outcome of this evaluation reveals that among the models considered, the SVM model + GAK achieves the highest average accuracy across sectors in predicting directional movements (55.6%) . However, a critical observation arises from this analysis. While the global alignment kernels exhibit promising prediction accuracy, they fall short in effectively capturing returns, thereby compromising the overall reliability of the models. The second model in terms of average accuracy is the XGBoost model (55.3%). This model records a good performance in terms of captured returns, showing consistently better returns than the simple Buy and Hold strategy for most of the sectors. However the model that shows the most consistent improvement over the benchmark strategy is the logistic regression model + signature, which scores an average accuracy of (53.5%). Similarly to this last model, the SVM+signature shows an average accuracy of directional predictions of 53.9% and shows consistently outperforms the benchmark strategy. The last model to obtain satisfactory results is the SVM+sig kernel model, that, despite not showing results to the level of the previous model, still consistently outperforms the Buy and Hold strategy for most of the sectors.

The current project offers several promising avenues for further exploration. One avenue involves the inclusion of longer time horizons for the used time series. This broader dataset not only may improve the ability of the XGBoost model to generalize by encompassing diverse market cycles and behaviors, but also alleviates overfitting concerns that arose during this project in many occasions. Consequently, this expansion enables the utilization of the signature transform with higher levels before truncation, the incorporation of additional variables into the signature, and the integration of simple neural network models. Moreover, an increasing availability of data facilitates the creation of comprehensive test sets, allowing for more comprehensive assessment of the models' performance across various market scenarios. Another compelling path towards improvement lies in the incorporation of a wider set of variables, tailored to the unique characteristics of each sector. For example, variables like the 3-month treasury yield for Financial Institutions or the US Treasury H15 Constant Maturity 5-year real yield curve rates for sectors like Consumer Cyclical, Consumer Non-Cyclical, and Financial Institutions could provide valuable insights. This approach extends to considering not only new time series data, but also experimenting with different lengths for the time series used for the kernels, diverse window shifts for distinct sectors, and variations in length and smoothing techniques for each dataframe.

Additionally, exploring alternative signature kernels, as the ones presented by Cass et al. in [46], presents an opportunity for further developments. These alternate kernels may potentially yield improved results, especially when combined with higher dyadic orders. These orders determine the discretization of the grid utilized in the PDE solver, consequently regulating the precision of the signature kernel's approximation. To address potential computational constraints that could occur even for low value of the dyadic order when using these alternative kernels, a lower-rank approximation strategy could be applied in such cases, effectively mitigating computational costs.

Appendix A

Performance metrics

A.1 Performance metrics - validation data

Financial services	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	55	16.3	1.95	51.1
SVM + signature	52.7	20.1	8.45	54.3
SVM + GAK	57.5	21.3	-1.34	49.5
SVM + sig kernel	57.2	19.5	1.14	50.5
logit + signature	56.3	19.8	9.0	54.5
logit + GAK	57.5	21.3	-2.60	49.1
logit + sig kernel	53.6	20.2	-0.87	49.6

Utility	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	59.2	22.5	1.44	57.4
SVM + signature	54.4	16.1	8.82	54.4
SVM + GAK	55.3	21.8	2.93	51.2
SVM + sig kernel	51.7	22.8	-0.36	49.8
logit + signature	53.6	17.4	6.77	53.4
logit + GAK	55.8	21.7	4.18	51.8
logit + sig kernel	51.7	22.1	8.38	49.5

Basic Industry	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	59.1	13.7	11.5	56.6
SVM + signature	53.9	18.79	7.40	53.7
SVM + GAK	58.9	22.0	6.25	52.5
SVM + sig kernel	51.7	24.0	0.56	50.3
logit + signature	55	20.1	7.46	53.8
logit + GAK	58.9	22.0	6.25	52.5
logit + sig kernel	58.0	18.2	19.1	59.4

Capital Goods	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	52.5	16.7	2.37	51.3
SVM + signature	56.4	20.5	6.97	53.1
SVM + GAK	59.7	20.9	5.0	51.7
SVM + sig kernel	53.9	22.9	5.08	52.6
logit + signature	53.9	18.0	7.21	53.6
logit + GAK	58.1	21.6	1.21	50.4
logit + sig kernel	53.9	13.6	46.0	52.3

Communications	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	53.3	13.5	0.046	50.3
SVM + signature	54.2	18.1	8.25	54.1
SVM + GAK	52.8	21.4	-0.38	49.8
SVM + sig kernel	54.2	22.4	7.25	53.6
logit + signature	53.9	19.7	7.91	54.0
logit + GAK	50.0	21.6	-5.63	47.4
logit + sig kernel	50.8	20.6	-0.12	49.9

Consumer Cyclical	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	56.7	17.8	1.4	57.1
SVM + signature	53.6	22.5	6.88	53.4
SVM + GAK	54.7	20.1	1.17	50.4
SVM + sig kernel	50.8	17.4	1.54	50.8
logit + signature	51.9	22.0	3.17	51.6
logit + GAK	53.1	20.4	-1.88	49.2
logit + sig kernel	54.7	19.0	-5.55	47.3

Consumer Non-Cyclical	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	55.8	17.5	12.1	56.0
SVM + signature	53.3	17.6	6.25	53.1
SVM + GAK	53.1	18.1	-0.54	49.8
SVM + sig kernel	53.3	19.8	3.80	51.8
logit + signature	51.9	20.9	3.66	51.8
logit + GAK	53.1	18.1	-0.54	49.8
logit + sig kernel	51.7	16.4	21.5	51.1

Energy	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	54.2	18.3	6.83	53.5
SVM + signature	52.5	17.4	4.41	52.2
SVM + GAK	55.0	22.1	3.31	51.4
SVM + sig kernel	50.3	24.4	-0.22	49.9
logit + signature	51.1	16.4	1.67	50.8
logit + GAK	55.0	22.1	3.00	51.2
logit + sig kernel	50.3	20.5	-6.15	46.9

Technology	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	54.1	18.0	2.7	51.4
SVM + signature	55.0	17.2	0.94	50.34
SVM + GAK	53.1	21.4	-2.1	49.1
SVM + sig kernel	52.2	23.5	-4.18	48.2
logit + signature	51.7	20.17	1.28	50.64
logit + GAK	51.9	21.6	-5.27	47.7
logit + sig kernel	50.3	19.1	-10.6	44.7

Transportation	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	55.0	20.4	7.8	54.1
SVM + signature	55.0	17.6	0.9	50.3
SVM + GAK	54.4	23.7	-1.02	49.6
SVM + sig kernel	55	22.3	1.87	50.9
logit + signature	51.7	20.2	1.3	50.6
logit + GAK	58.3	22.7	3.03	51.3
logit + sig kernel	51.7	22.0	-4.16	47.9

Other Industrial	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	55.2	14.1	1.85	51.0
SVM + signature	51.4	14.0	1.8	50.9
SVM + GAK	58.1	18.3	7.35	53.3
SVM + sig kernel	54.4	17.9	-5.7	48.2
logit + signature	54.1	16.9	6.5	53.2
logit + GAK	58.1	18.6	6.98	53.1
logit + sig kernel	51.4	18.1	-1.70	49.2

Table A.1: Performance metrics for the classifiers - validation data

A.2 Performance metrics - test data

Financial services	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	58.3	17.2	6.1	53.5
SVM + signature	60.2	18.9	19.7	60.0
SVM + GAK	52.3	20.5	-3.48	48.5
SVM + sig kernel	52.2	16.8	1.53	50-8
logit + signature	57.5	20.7	14.0	57.0
logit + GAK	53.4	20.4	-4.56	48.4
logit + sig kernel	57.4	19.8	15.1	57.6

Utility	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	59.9	15.7	19.6	59.7
SVM + signature	56.0	24.1	9.7	54.8
SVM + GAK	51.7	20.4	-4.56	47.9
SVM + sig kernel	55.7	18.4	11.3	55.7
logit + signature	52.7	24.6	3.26	51.6
logit + GAK	57.4	19.1	5.63	52.5
logit + sig kernel	59.7	22.1	8.38	54.2

Basic Industry	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	57.8	17.3	0.45	48.5
SVM + signature	55.7	18.3	9.09	54.6
SVM + GAK	55.1	23.1	-1.89	49.2
SVM + sig kernel	56.8	21.6	10.8	55.4
logit + signature	54.2	19.2	6.5	53.2
logit + GAK	58.5	22.1	1.1	50.4
logit + sig kernel	59.6	20.6	12.6	56.3

Capital Goods	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	63.5	15.0	15.0	59.5
SVM + signature	57.3	17.9	0.34	50.1
SVM + GAK	57.4	17.5	1.46	50.4
SVM + sig kernel	50.0	18.8	-1.39	49.3
logit + signature	56.4	17.1	9.64	54.8
logit + GAK	57.4	17.5	-1.76	49.6
logit + sig kernel	52.3	13.6	4.60	52.3

Communications	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	53.6	19.0	0.74	50.4
SVM + signature	52.1	18.5	2.82	51.4
SVM + GAK	50.6	20.11	-5.1	48.0
SVM + sig kernel	54.0	18.6	1.22	50.2
logit + signature	53.3	19.6	5.90	52.9
logit + GAK	51.7	20.0	-3.47	48.8
logit + sig kernel	50.3	22.0	-5.70	47.1

Consumer Cyclical	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	52.1	16.3	13.1	49.6
SVM + signature	56.4	18.6	11.7	55.8
SVM + GAK	54.5	19.0	2.0	50.8
SVM + sig kernel	48.3	20.4	1.12	50.6
logit + signature	59.2	20.8	17.4	58.7
logit + GAK	54.5	19.0	-1.2	49.6
logit + sig kernel	57.9	17.9	16.37	58.3

Consumer Non-Cyclical	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	56.1	18.8	0.98	55.9
SVM + signature	50.9	17.8	1.15	50.6
SVM + GAK	52.3	17.1	-1.61	49.4
SVM + sig kernel	54.0	18.2	8.74	54.4
logit + signature	50.9	17.8	1.02	50.5
logit + GAK	53.4	16.9	-1.08	49.7
logit + sig kernel	50.0	17.8	2.91	51.5

Energy	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	52.4	18.1	3.3	51.6
SVM + signature	58.1	17.8	14.4	57.2
SVM + GAK	58.5	18.6	6.35	51.4
SVM + sig kernel	58.5	16.2	6.54	51.6
logit + signature	57.0	22.1	12.8	56.3
logit + GAK	58.0	18.7	3.52	50.7
logit + sig kernel	51.7	20.4	5.13	52.6

Technology	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	56.8	18.5	0.9	55.1
SVM + signature	53.5	13.2	6.65	53.2
SVM + GAK	50.6	17.9	-6.79	47.7
SVM + sig kernel	51.7	16.6	3.97	52.0
logit + signature	54.8	21.61	9.20	54.6
logit + GAK	51.7	17.8	-3.48	48.8
logit + sig kernel	46.0	18.8	-3.11	48.4

Transportation	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	55.7	17.9	4.36	52.3
SVM + signature	60.8	0.24	20.6	60.3
SVM + GAK	55.1	20.5	-6.56	48.1
SVM + sig kernel	56.8	17.2	-3.84	49.1
logit + signature	57.3	24.8	14.6	57.3
logit + GAK	58.5	19.3	0.00	50.0
logit + sig kernel	54.0	21.8	8.87	54.5

Other Industrial	Accuracy(%)	Accuracy std (%)	MCC(%)	Balanced accuracy (%)
XGBoost	61.3	17.54	14.1	57.9
SVM + signature	57.9	17.4	12.5	56.3
SVM + GAK	50.0	23.5	-8.96	46.2
SVM + sig kernel	51.1	23.8	-8.5	46.
logit + signature	60.6	17.4	17.6	58.8
logit + GAK	52.3	23.4	7.9	47.3
logit + sig kernel	51.1	22.9	-2.17	48.9

Table A.2: Performance metrics for the classifiers - test data

A.3 Excess returns by sector

Financial Services	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.52	6.04	-1.34	0.09
XGBoost	1.73	6.04	-0.93	0.29
SVM + signature	3.71	6.02	1.05	0.616
SVM + GAK	-1.31	6.04	-1.20	-0.217
SVM + sig kernel	0.71	6.04	-1.15	0.117
logit + signature	3.27	6.03	1.07	0.544
logit + GAK	-1.74	6.04	-1.21	-0.289
logit + sig kernel	5.24	6.00	1.84	0.874

Utility	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.32	7.47	-2.52	0.043
XGBoost	4.22	7.50	-1.51	0.565
SVM + signature	4.13	7.47	3.60	0.578
SVM + GAK	-1.22	7.50	-2.37	-0.164
SVM + sig kernel	3.46	7.47	2.32	0.464
logit + signature	2.35	7.49	3.34	0.315
logit + GAK	-2.32	7.49	-2.43	-0.310
logit + sig kernel	3.58	7.48	3.22	0.479

Basic Industry	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	1.21	7.28	-2.34	0.163
XGBoost	2.46	7.30	-1.66	0.338
SVM + signature	8.18	7.30	2.71	1.133
SVM + GAK	-0.44	7.22	-2.25	-0.061
SVM + sig kernel	2.25	5.78	-0.80	0.390
logit + signature	4.87	7.24	2.73	1.061
logit + GAK	1.19	7.30	-2.35	0.163
logit + sig kernel	2.12	7.28	2.43	0.668

Capital Goods	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	1.09	6.67	-2.37	0.163
XGBoost	1.56	6.66	-2.42	0.234
SVM + signature	1.68	6.66	-2.33	0.253
SVM + GAK	0.14	6.65	-2.32	0.021
SVM + sig kernel	2.94	6.67	-2.43	0.442
logit + signature	1.52	6.66	1.30	0.227
logit + GAK	-0.13	6.66	-2.30	0.413
logit + sig kernel	2.74	6.67	2.91	-0.020

Communications	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.72	8.46	-0.38	0.085
XGBoost	4.04	8.44	2.21	2.21
SVM + signature	3.29	8.44	2.11	0.390
SVM + GAK	-0.88	8.45	-0.225	-0.105
SVM + sig kernel	3.38	7.30	0.007	0.463
logit + signature	5.70	8.42	2.10	0.677
logit + GAK	-0.89	8.45	-0.225	-0.105
logit + sig kernel	1.80	8.45	-0.225	0.213

Consumer Cyclical	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.41	7.28	-1.77	0.056
XGBoost	0.880	7.28	1.64	0.128
SVM + signature	2.92	7.27	-0.43	0.401
SVM + GAK	-1.14	7.28	-1.64	-0.156
SVM + sig kernel	1.52	7.28	-1.81	0.210
logit + signature	3.62	7.26	-0.468	0.499
logit + GAK	-1.04	7.28	-1.69	-0.143
logit + sig kernel	0.89	7.28	-1.78	0.123

Consumer Non-Cyclical	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	1.07	6.89	-0.580	0.155
XGBoost	1.82	6.88	-0.643	0.264
SVM + signature	1.20	6.89	-0.518	0.174
SVM + GAK	-0.86	6.89	-0.393	-0.125
SVM + sig kernel	1.10	6.71	0.560	0.164
logit + signature	1.63	6.88	2.280	0.236
logit + GAK	-0.86	6.89	-0.393	-0.558
logit + sig kernel	-3.84	6.88	0.499	-0.125

Energy	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.13	11.2	-3.84	0.011
XGBoost	9.46	11.1	0.917	0.851
SVM + signature	5.78	11.2	-0.097	0.518
SVM + GAK	0.40	11.2	-3.75	0.036
SVM + sig kernel	7.29	11.1	3.76	0.655
logit + signature	0.99	11.2	-0.371	0.088
logit + GAK	-0.427	11.0	-3.82	-0.038
logit + sig kernel	18.51	11.2	4.13	1.68

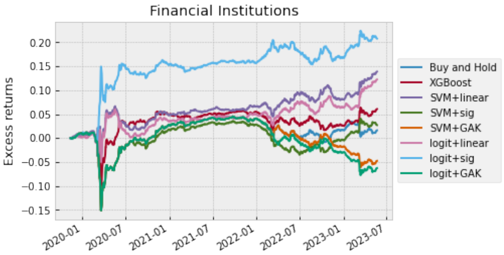
Technology	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.87	5.92	-0.468	0.147
XGBoost	7.04	5.82	3.05	1.21
SVM + signature	1.06	5.92	1.90	0.180
SVM + GAK	-1.15	5.91	-0.283	-0.195
SVM + sig kernel	1.90	5.78	0.285	0.329
logit + signature	1.85	5.91	1.87	0.312
logit + GAK	0.85	5.92	-0.358	0.144
logit + sig kernel	2.23	5.91	1.20	0.378

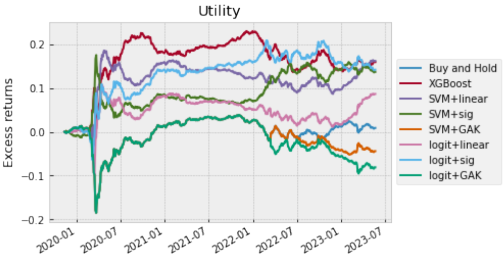
Transportation	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.92	8.20	-1.24	0.113
XGBoost	3.92	8.18	-0.978	0.479
SVM + signature	4.90	8.18	-0.257	0.599
SVM + GAK	-0.71	8.20	-1.08	-0.086
SVM + sig kernel	5.99	8.15	0.970	0.734
logit + signature	3.49	8.20	-0.177	0.426
logit + GAK	-0.71	8.20	-0.985	-0.086
logit + sig kernel	-3.50	8.20	-1.07	-0.427

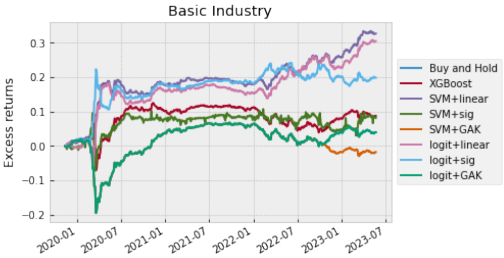
Other Industrial	Avg excess returns (%)	Std (%)	Skewness	Information ratio
Buy & hold	0.033	6.99	-5.27	0.005
XGBoost	1.67	6.99	-5.27	0.240
SVM + signature	3.64	6.95	-2.99	0.524
SVM + GAK	-2.06	6.98	-5.17	-0.296
SVM + sig kernel	4.39	6.87	5.30	0.640
logit + signature	3.64	6.95	-3.00	0.524
logit + GAK	-1.72	6.97	-5.17	-0.246
logit + sig kernel	2.75	6.97	4.98	0.394

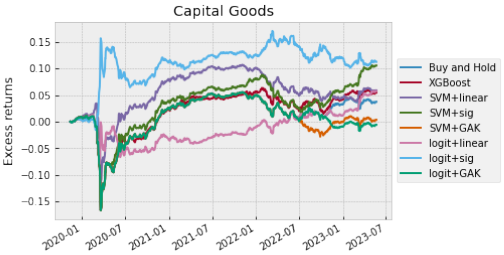
Table A.3: Comparison between the different classifiers in terms of annualized excess returns, annualized standard deviation of the excess returns, skewness of the excess returns and annualized information ratio

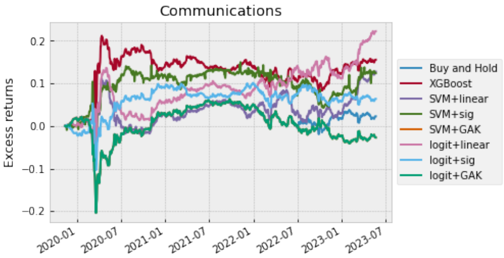
A.4 Excess returns by sector - plots

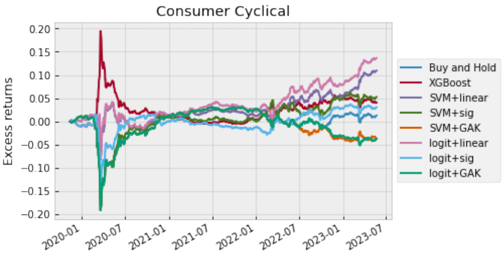


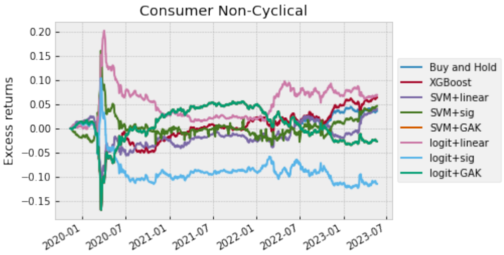


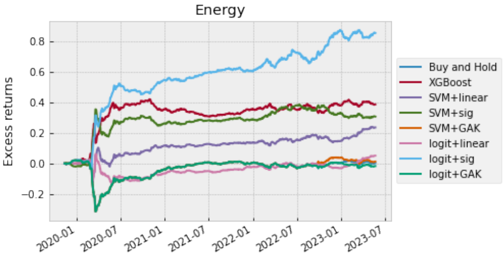


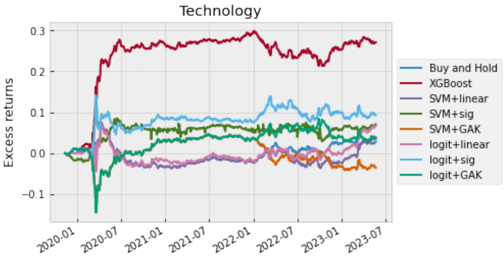


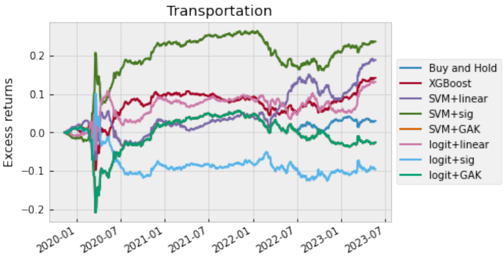


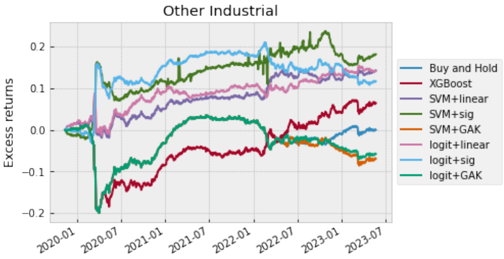












Bibliography

[1] Eugene F Fama. The behavior of stock-market prices. *The journal of Business*, 38(1):34–105, 1965.

[2] Francis E. H. Tay and Lijuan Cao. Application of support vector machines in financial time series forecasting. *Omega*, 29(4):309–317, 2001.

[3] Yanshan Wang. Stock price direction prediction by directly using prices data: an empirical study on the kospi and hsi. *International Journal of Business Intelligence and Data Mining*, 9(2):145–160, 2014.

[4] Marija Gorenc Novak and Dejan Velušček. Prediction of stock price movement based on daily high prices. *Quantitative Finance*, 16(5):793–826, 2016.

[5] Phichhang Ou, Hengshan Wang, et al. Prediction of stock market index movement by ten data mining techniques. *Modern Applied Science*, 3(12):28–42, 2009.

[6] Ronen Israel, Diogo Palhares, and Scott A Richardson. Common factors in corporate bond returns. *Forthcoming in the Journal of Investment Management*, 2017.

[7] Harald Henke, Hendrik Kaufmann, Philip Messow, and Jieyan Fang-Klingler. Factor investing in credit. *The Journal of Beta Investment Strategies*, 11(1):33–51, 2020.

[8] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152.

[9] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[10] Ingo Steinwart and Andreas Christmann. *Support vector machines.* Springer Science & Business Media, 2008.

[11] Wessel N van Wieringen. Lecture notes on ridge regression. *arXiv preprint arXiv:1509.09169*, 2015.

[12] Tommi S Jaakkola and David Haussler. Probabilistic kernel regression models. In *Seventh International Workshop on Artificial Intelligence and Statistics.* PMLR, 1999.

[13] Thomas P Minka. A comparison of numerical optimizers for logistic regression. *Unpublished draft*, pages 1–18, 2003.

[14] Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85:41–75, 2011.

[15] Kaggle. State of data science and machine learning 2021.

[16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[17] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. Kernel methods in machine learning. 2008.

[18] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

[19] Francis R Bach and Michael I Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd international conference on Machine learning*, pages 33–40, 2005.

[20] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[21] Evert J Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. 1930.

[22] Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Advances in neural information processing systems*, 13, 2000.

[23] Andrew Cotter, Joseph Keshet, and Nathan Srebro. Explicit approximations of the gaussian kernel. *arXiv preprint arXiv:1109.4603*, 2011.

[24] Hans-Hermann Bock and Edwin Diday. *Analysis of symbolic data: exploratory methods for extracting statistical information from complex data*. Springer Science & Business Media, 1999.

[25] Taras K Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968.

[26] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[27] Meinard Müller. Dtw-based motion comparison and retrieval. *Information Retrieval for Music and Motion*, pages 211–226, 2007.

[28] Meinard Müller, Henning Mattes, and Frank Kurth. An efficient multiscale approach to audio synchronization. In *ISMIR*, volume 546, pages 192–197. Citeseer, 2006.

[29] Yangqi Li and Taihua Hu. Dynamic time warping application for financial pattern recognition. *Available at SSRN 3658339*, 2020.

[30] Jorge C Lucero, Kevin G Munhall, Vincent L Gracco, and James O Ramsay. On the registration of time and the patterning of speech movements. *Journal of Speech, Language, and Hearing Research*, 40(5):1111–1117, 1997.

[31] Laura L Koenig, Jorge C Lucero, and Elizabeth Perlman. Speech production variability in fricatives of children and adults: Results of functional data analysis. *The Journal of the Acoustical Society of America*, 124(5):3158–3170, 2008.

[32] Eric W. Weisstein. Delannoy number. From MathWorld—A Wolfram Web Resource.

[33] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on acoustics, speech, and signal processing*, 23(1):67–72, 1975.

[34] E Keogh and A Ratanamahatana. Everything you know about dynamic time warping is wrong. In *3rd Workshop on Mining Temporal and Sequential Data, in conjunction with 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD-2004), Seattle, WA*, volume 1, pages 1–11, 2004.

[35] Marco Cuturi, Jean-Philippe Vert, Oystein Birkenes, and Tomoko Matsui. A kernel for time series based on global alignments. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 2, pages II–413. IEEE, 2007.

[36] Serge Bernstein. Sur les fonctions absolument monotones. *Acta Mathematica*, 52(1):1–66, 1929.

[37] Isaac J Schoenberg. Metric spaces and completely monotone functions. *Annals of Mathematics*, pages 811–841, 1938.

[38] Kuo-Tsai Chen. Integration of paths, geometric invariants and a generalized baker-hausdorff formula. *Annals of Mathematics*, 65(1):163–178, 1957.

[39] Terry J Lyons. Differential equations driven by rough signals. *Revista Matemática Iberoamericana*, 14(2):215–310, 1998.

[40] Lajos Gergely Gyurkó, Terry Lyons, Mark Kontkowski, and Jonathan Field. Extracting information from the signature of a financial data stream. *arXiv preprint arXiv:1307.7244*, 2013.

[41] Maud Lemercier, Cristopher Salvi, Theodoros Damoulas, Edwin Bonilla, and Terry Lyons. Distribution regression for sequential data. In *International Conference on Artificial Intelligence and Statistics*, pages 3754–3762. PMLR, 2021.

[42] Weixin Yang, Lianwen Jin, and Manfei Liu. Deepwriterid: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, 31(2):45–53, 2016.

[43] Terry J Lyons, Michael Caruana, and Thierry Lévy. *Differential equations driven by rough paths*. Springer, 2007.

[44] Peter K Friz and Nicolas B Victoir. *Multidimensional stochastic processes as rough paths: theory and applications*, volume 120. Cambridge University Press, 2010.

[45] Ilya Chevyrev and Andrey Kormilitzin. A primer on the signature method in machine learning. *ArXiv*, abs/1603.03788, 2016.

[46] Thomas Cass, Terry Lyons, and Xingcheng Xu. General signature kernels. *arXiv preprint arXiv:2107.00447*, 2021.

[47] Cristopher Salvi, Thomas Cass, James Foster, Terry Lyons, and Weixin Yang. The signature kernel is the solution of a goursat pde. *SIAM Journal on Mathematics of Data Science*, 3(3):873–899, 2021.

[48] Bloomberg L.P. Bloomberg barclays methodology.

[49] Bloomberg L.P. Us corporate index factsheet.

[50] Luca Benzoni, Olena Chyruk, and David Kelley. Why does the yield-curve slope predict recessions? *Available at SSRN 3271363*, 2018.

[51] Antti Ilmanen. Stock-bond correlations. *The Journal of Fixed Income*, 13(2):55, 2003.

[52] William R Gebhardt, Soeren Hvidkjaer, and Bhaskaran Swaminathan. Stock and bond market interaction: Does momentum spill over? *Journal of Financial Economics*, 75(3):651–690, 2005.