DELFT UNIVERSITY OF TECHNOLOGY

MASTER THESIS

Proxying Bond Credit Spreads with Machine Learning

Author:
Giulio BACCI DI CAPACI

Supervisor:
Prof. Dr. Ir. Cornelis Oosterlee
Dr. Ir. Xinzheng Huang

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in

Applied Mathematics
Department or School Name

August 12, 2020

Declaration of Authorship

I, Giulio BACCI DI CAPACI, declare that this thesis titled, "Proxying Bond Credit Spreads with Machine Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 12/08/2020

DELFT UNIVERSITY OF TECHNOLOGY

Abstract

Faculty of Electrical Engineering, Mathematics and Computer Science
Department or School Name

Master of Science

Proxying Bond Credit Spreads with Machine Learning

by Giulio BACCI DI CAPACI

The bond market is affected by the shortage of liquidity problem, which means that many bonds are not frequently traded. This implies that market data for these bonds are missing. This lack of data represent a problem for financial risk measures such as Value at Risk (VaR). This research provides the framework for the construction of a proxy which replaces the missing data with artificial data such that VaR can be calculated. The data used for the VaR calculation are bond z-spreads, which is a credit spread measure. This research represents an improvement of the current proxy methodologies under different aspects. A major improvement is provided by the usage of machine learning algorithms such as Random Forest, Support Vector Regression and CatBoost which significantly increased the predictive accuracy of the proxy. Another main difference from the current proxy methods relies in the prediction of z-spreads daily changes (shifts), instead of z-spread levels. This modification required a shift types assessment and it has been beneficial both for the proxy performance and for the VaR calculation. The main result of this thesis from a financial and statistical perspective is the theoretical and empirical convergence of the VaR obtained through the proxy with the VaR calculated with real market data.

Acknowledgements

The realization of my thesis research came true thanks to the help of several people. First I would like to express my gratitude to my supervisors: Andrea Fontanari, Katarina Strizencova, Kees Oosterlee and Xinzheng Huang. Without any of you, this thesis would have not been the same, and furthermore, the enjoyment I experienced working on it would not be the same. Each of you brought me different point of view based on your experience and I could learn a lot from that.
Secondly, I want to thank ING bank in Amsterdam for giving me this opportunity to grow from a personal and professional perspective. In particular I am grateful to the entire Model Validation for Market Risk team, where I could learn from litterally everyone from the day I joined as intern in October 2019.
Last but not least, I want to express my immense gratitude to my family and friends for the support they gave me in this period. Writing a master thesis is quite intense, but with the right people next to you, it is nothing but a pleasant journey.

Contents

List of Tables

Chapter 1

Introduction

This thesis aims to provide an improved framework for the analysis of one of the most illiquid markets in finance: the bond market. This problem is called 'shortage of liquidity problem' (Brummelhuis and Luo, 2017), and it refers to those assets that are not frequently traded which results in a lack of data.
This lack of data is a problem for financial institution, as it does not allow for the calculation of pricing and risk measures such as Credit Value Adjustment (CVA) and Value at Risk (VaR), the latter is the main application of this thesis.
The shortage of liquidity problem can be partially solved by the usage of a proxy, in which the missing data points are replaced with artificial data in order to mimic the behaviour of the original asset.
The major requirements, for proxy VaR calculation, imposed by the European Banking Authority (EBA) aim for a conservative proxy such that the risk is mitigated, (EBA, 2016). In this thesis, we focus both on the accuracy of the proxied data and on the estimation of a conservative VaR.
Proxy models for bonds and CDS credit spreads are already wide-spread across financial institution. In 2013, EBA introduced the Intersectional Method, in which missing data points are replaced by the an average of liquid data which belong to the same categorical features, such as: credit rating, region and sector, (EBA, 2013). Shortly after the publication of the Intersectional Method, Nomura bank provided a second and more sophisticated method for proxing CDS credit spreads, the Cross-Sectional Method, (Chourdakis et al., 2013). The latter is based on a multidimensional regression across credit rating, region, industry sector and seniority. It effectively reduced many of the problems related to the Intersectional Method, such as stability, robustness and consistency. For this reason we decided to adopt the Cross-Sectional Method as our benchmark model for proxying bond z-spreads which is a measure of credit spread.
Our analysis takes the Cross-Sectional method as starting point and develops it on different aspects. One of the main improvements is offered by machine learning (ML) algorithms, which in recent years are becoming more and more popular in the financial industry. ML algortihms allow to model complex non-linear behaviour which cannot be captured by linear regression. Furthermore, the selection of the categorical features of the model, such as credit rating, region, seniority and their interaction terms can be automatically modelled through machine learning algorithms. Within this thesis we selected 3 different ML algorithms: Random Forest and CatBoost, which are decision trees based methods and Support Vector Regression, which belongs to the Support Vector Machines (SVM).
Another major improvements from the benchmark model is obtained by the addition of extra categorical features, such as currency, time to maturity and market indicator.

The main application of this research is the VaR calculation. In particular, the methodology used for VaR calculation is the historical VaR (HVaR). A crucial aspect in order to perform HVaR calculation is the modelling of credit spreads daily changes which are commonly referred as 'shifts'. Popular choices for modelling daily changes are absolute and relative shifts. Throughout this thesis an extensive assessment of the most suitable shift type is performed. The shift types assessment is the starting point of this thesis. This because, in this research, differently from the current proxy methods, we aim to directly proxy the shifts of the bond credit spreads, i.e. the daily changes in the z-spread of bonds. For this reason, the choice of the shift type is of fundamental importance for the construction and calibration of the proxy model itself.
This analysis has been carried out on a data-set containing 8119 unique bonds across a two years time-frame, from 21^{st} August 2017 to the 20^{th} of August 2019. More than 50% of the z-spreads in the data-set is missing, this clearly outlines the shortage of liquidity problem.
The most challenging task in this research is the maximization of the credit spreads proxy accuracy together with the replication of a conservative VaR, which translate into a proxy that is both accurate and allows for large volatility in the predicted z-spreads in order to mimic the fluctuations of real z-spreads data. This often turned into a trade-off between high accuracy and realistic volatility, however we provided solutions for achieving both with a satisfying outcome.
Another important limit of existing proxy methodologies relies on the fact that the proxy model is estimated day-by-day using only the information available on each single day. The main part of this thesis follows this procedure as it turned out to be simple and effective. However, historical data can provide key insights on the behaviour of bond z-spreads and for this reason we investigated possible further developments incorporating bond credit spreads information across time. This is done, in this research, by the usage of autoregressive (AR) models and it provided a strong foundation for the study of more complex time-varying models, in which information from past and future can be used in order to achieve a even more sophisticated proxy.
This thesis is organized as follows. In Chapter 2, we provide background information about bonds such as pricing approach and an extensive description of bond z-spread. Afterwards, we present Value at Risk and the methodology of its application to this thesis: the historical VaR. This is followed by a description of the current proxy methodologies and the restrictions that these imply, together with the targets of machine learning in this framework and the reasons why ML algorithms could overcome most of the current limitations.
Chapter 3 presents the data-set that has been used for this research and the preprocessing steps that have been performed before the start of the main analysis. Also an exploratory analysis on the processed data is presented. This contains visual representations that can provide clear insights about the data structure and the bond features.
In Chapter 4, all the methodologies used through the thesis are described. First, those related to the shift types assessment, such as backtesting methods and econometric tests. Then, we present the credit spreads proxy methodology, starting from the different evaluation metrics, to the different machine learning procedures and finally the complete model framework.
The results and the model optimization are presented in Chapter 5. Again, we first provide the results for the shift types assessment and then the performance of the bond credit spreads proxy. Of particular relevance for this thesis, from a financial

and statistical point of view, is the VaR comparison at the end of this chapter.
Finally in Chapter 6, we provide a supplementary analysis on the possible further developments of this model. This part aims to include past information of bond z-spreads by means of an autoregressive model.
The last chapter of the thesis, Chapter 7, is dedicated to the conclusions drawn from our research.

Chapter 2

Background

This chapter introduces the fundamental blocks that are needed in order to understand the framework of this thesis project. In order, we present the object in examination (bond z-spreads), the application of the project (historical Value at Risk) and the current methodologies which are used to tackle similar problems. These methods are simpler than those presented in the rest of the thesis and therefore are useful to clearly highlight the problem statement and the limitations of the current methodologies.

2.1 Bond

Bonds are debt securities issued by public authorities, credit institutions or companies to investors in order to raise capital.
A bond issuer (borrower) is obliged to pay a coupon, which is an interest rate, in consecutive time intervals of generally 6 months to the buyer of the bond (lender). Bond instruments have a maturity date, in which the principal, or face value of the bond, is paid back to the bond owner. The interest rate or coupon is a percentage of the face value and the bond can be mathematically described as a collection of cash flows composed by: a series of coupon payments which length depends on the life of the bond (the time to maturity) and the periodicity of the payments and finally a maturity payment that is the sum of a coupon and the face value.
Below we examine the main features of a bond.

Issuer: The nature of the issuer impacts the way the bond is considered in the market and therefore its value. Government bonds usually have the lowest risk and thus are considered of higher quality. The creditworthiness of the issuer determines the credit rating, which is one important feature of bonds, e.g., 'AAA' is the highest quality rating and it is often assigned to trustable treasury bonds.

Coupon rate and face value: The face value is the amount that the issuer pays back to the buyer of the bond at the maturity date. It is also called maturity value, redemption value or par value.
The coupon rate is the annual interest rate paid to the bondholder and is a percentage of the face value. Coupons are issued in form of periodic payments. All bonds imply periodic interest payments except zero-coupon bonds. The latter allows the lender for a positive interest since it is sold below its face value.

Time to maturity: This is the number of years after which the issuer will pay back the obligation. The time to maturity strongly influences the bond's yield as well as the volatility of the bond price.

2.1.1 Bond Pricing

Here, we present the "traditional" approach to bond pricing: in practice more sophisticated techniques are used, but for the scope of this introduction to bonds and z-spreads the following approach is satisfying.
The price of a bond is equal to the present value of its cash flows, therefore we first need to know the bond's cash flows in order to determine the appropriate interest rate at which we can discount the cash flows. Then the price of the bond can be calculated.
Bond's cash flows are coupons that are paid during the life of the bond,together with the final redemption payment. The coupon payments for conventional bonds are made annually, semi-annually or quarterly. If the coupon is semi-annually provided, for example, half of the coupon is paid as interest every six months.
The interest rate used to discount a bond's cash flows (discount rate) is the rate required by the bondholder. It is also known as bond's yield. The bond's yield is determined by the market and the price demanded by investors to buy it.
The fair price of a bond is the present value of all its cash flows. Hence, when pricing a bond we need to calculate the present value of all coupon interest payments and of the maturity payment, and sum these. The price of a conventional semi-annual bond can be given by:

$$P = \frac{C/2}{(1+\frac{1}{2}r)} + \frac{C/2}{(1+\frac{1}{2}r)^2} + \frac{C/2}{(1+\frac{1}{2}r)^3} + ... + \frac{C/2}{(1+\frac{1}{2}r)^{2N}} + \frac{M}{(1+\frac{1}{2}r)^{2N}} \tag{2.1}$$

Where P is the price of the bond, C is the annual coupon payment, that is divided by two because the bond issues semi-annual payments, r is the discount rate, N is the number of years to maturity, therefore a semi-annually paying bond has $2N$ interest periods, M is the face value or maturity payment. For more details see Choudhry, 2003.

2.1.2 Z-Spread

The z-spread is used by analysts and investors to capture discrepancies in bond prices. The z-spread is the parallel shift over the zero-coupon Treasury curve in order to equate the discounted cash flows of the bond to its market value. It is a premium to compensate bond holders for taking credit risk.
A coupon bond can be thought of as a collection of zero-coupon bonds, where each coupon is a small zero-coupon bond that matures when the bondholder receives the coupon. Each one of these bond cash flows is discounted using its own particular yield to maturity from the spot rate Treasury curve or zero-coupon Treasury curve. The z-spread is the number added to all different yields to maturity in the equation such that the bond price equals the market value of the bond. This can be understood from the following equation:

$$\begin{aligned} P &= \frac{C_1/m}{(1+\frac{1}{m}(T_1+Z))} + \frac{C_2/m}{(1+\frac{1}{m}(T_2+Z))^2} + ... + \frac{C_{mN}/m}{(1+\frac{1}{m}(T_{mN}+Z))^{mN}} + \frac{M}{(1+\frac{1}{m}(T_{mN}+Z))^{mN}} \\ &= MV \end{aligned} \tag{2.2}$$

In this equation the discount rate is adjusted with the z-spread. Here, m is the frequency of coupon payments, that before was 2, Z is the z-spread, T_i is the proper yield on the spot rate Treasury curve and MV is the market value of the bond. Calculating the Z-spread is an iterative computation as it is the same number that needs to be plugged-in all cash flows to get the equality between market price and theoretical price. It may be the case that the bond earns a lower yield than the zero-coupon Treasury curve, in that case the Z-spread will be negative. For more information about the z-spread see Choudhry, 2006.
To summarize: the z-spread measures the additional return, earned by the owner of a bond comparing to the benchmark return, which is using the zero-coupon Treasury curve. A higher z-spread implies a major potential profit for the buyer of the bond, as the distance from the spot rate Treasury curve increases, but it also carries higher risk.

2.2 Value at Risk

In this section we introduce the risk measure Value at Risk (VaR). The bond history generated by the bond credit spreads proxy in this thesis will be used to calculate VaR. VaR has been introduced in RiskMetrics^{TM} by JP Morgan in 1994 and it is the main market risk measure used in banking, i.e. it is used to measure the risk derived by fluctuations of prices in the market.
VaR measures the worst expected loss that a company may have with a specified confidence level over a time period under normal market conditions. The confidence level and the time period are specified by the user. We can define VaR more formally as:

Let $\boldsymbol{X}$ be a profit and loss distribution, The VaR at level $\alpha \in (0,1)$ is the smallest number y such that the probability that $\boldsymbol{Y} := \boldsymbol{-X}$ does not exceed y is at least $1-\alpha$, i.e., $VaR_\alpha(\boldsymbol{X})$ is the $(1-\alpha)$-quantile of $\boldsymbol{Y}$:

$$VaR_\alpha(X) = -inf\{x \in \mathbb{R} : F_X(x) > \alpha\}, \tag{2.3}$$

where $F_X(x)$ is the cumulative distribution of the random variable $\boldsymbol{X}$.

For example if a daily VaR is 100,000 for a 95% confidence level, it means that during that day there is a 95% probability that the loss will be smaller than 100,000.
VaR is calculated within a given confidence interval that is typically 95% or 99% .
The main assumption in VaR models is that the distribution of future price changes will follow past variations. There are three different methods for computing VaR and these are: the variance/covariance method, Monte Carlo simulation and historical simulation. For more information about VaR see Choudhry, 2003.
In the following section we introduce the historical simulation method, as the shifts predicted by our credit spreads proxy are used to compute historical VaR.

2.2.1 Historical VaR

Historical simulation method for calculating VaR (HVaR) has the general advantage of relying on very few assumptions. The main assumptions required for the variance/covariance method, i.e. returns are normally distributed and constant correlation, are not required. The method makes very few assumptions about the market price process generating the portfolio's returns. It simply assumes that market price

changes in the future are drawn from the same empirical distribution as the market price changes generated by the historical data, i.e., price changes are independent identically distributed (i.i.d). The HVaR model computes potential losses using historical returns in the risk factors and therefore captures behaviours different from the normal distribution. Since the risk factor returns, used for calculating HVaR, are actual past movement, the correlations are also past correlations.
The prerequisite for HVaR calculations is that the risk factor market data in the historical period must be complete and from here comes the need of a proxy to fill-in the missing data points. The number of simulated scenarios for each HVaR calculation is 260. This number of scenarios implies that there must be market data available for 261 business days prior to the current date. VaR is computed to measure projected risk for 1 and 10 days into the future. When calculating a 1 day VaR, all scenarios must represent a 1 day market movement.
The method employed to calculate the scenarios is therefore to calculate the differences between market data (shifts) on each historical date and create a complete set of shifts. These shifts are then applied to the current market data to create the new scenarios which represents different potential market data variations (ING, 2018b). The choice of the shifts type for the underlying risk factor, i.e. Shift Type Assessment, is of great importance for this task and it is examined in this thesis.
The HVaR methodology for generating scenarios consists of the following steps:

1: Define a vector $\boldsymbol{X}$ consisting of $n + 1$ days of daily observations $\boldsymbol{X} = \{x_0, x_1, ..., x_n\}$ where n usually equals 260, x_0 is the oldest observation and x_n the most recent.

2: Calculate n shifts defined in absolute, relative or others terms depending on the risk factor. Here we present the (displaced) relative shifts, absolute shifts and arcsinh shifts.

Description	Shift type	Scenario value
Absolute shift	$\Delta x_i = [x_{i+1} - x_i] \cdot \sqrt{N}$	$x_i^{scn} = x_n + \Delta x_i$
(Displaced) Relative shift	$\Delta x_i = [\frac{x_{i+1} - x_i}{x_i + a}] \cdot \sqrt{N}$	$x_i^{scn} = x_n(1 + \Delta x_i)$
ArcSinH shift	$\Delta x_i = [sinh^{-1}\frac{x_{i+1}}{b} - sinh^{-1}\frac{x_i}{b}] \cdot \sqrt{N}$	$x_i^{scn} = b \cdot sinh[\Delta x_i + sinh^{-1}\frac{x_n}{b}]$

Here, N is the holding period (one or ten days), a is the displacement factor for the relative shifts (when $a = 0$ we are using relative shifts), b is a parameter that can be opportunely tuned for implementing arcsinh shifts and x_n is the current market value for the risk factor x, i.e. the z-spread.

3: Apply the proper shift type to the risk factor's current market value. This will generate a distribution of 260 possible scenarios, among which VaR is calculated.

2.3 Current Proxy Methodologies

This section describes the two main bond proxy methods currently used by banks. The first method has been proposed by the European Banking Authority (EBA) in

relation to Credit Valuation Adjustment (CVA), (EBA, 2013). When the underlying credit spread is not observable in the market, EBA proposed to average data of liquid names (names that are traded and therefore data are available) across the same rating, region and sector to proxy spreads of not observable names. This method is defined as Intersection method.
A second and preferable method to the aforementioned one is the Cross-section approach described in Chourdakis et al., 2013. This method is based on a multi-dimensional regression across rating, region and industry sector and avoids many of the stability, robustness and consistency problems related with the Intersection method proposed by EBA.

2.3.1 Intersection Method

In the intersection methodology proposed by EBA in EBA, 2013, the liquid entities that belong to the same region, sector and rating are grouped together, this procedure is called bucketing. Then the proxy spread of an illiquid entity i is defined as:

$$S_i^{proxy} = \frac{1}{N}\sum_{j=1}^{N} S(j) \tag{2.4}$$

Here, $N \geq 1$ is the number of liquid names in the same rating, region, sector as entity i and $S(j)$ is the spread level of these. The main problem with the Intersection method is that there are buckets with few or no entities for the same rating, region and sector and therefore the proxy for that combination of entities cannot be calculated.
Another major issue with the Intersection method is historical instability of the proxy spreads. When a bond changes bucket, due to a rating migration for example, the proxy spreads for that bucket will have a jump. Especially, in case the bucket has a few entities, the jumps will be severe.

2.3.2 Cross-Section Method

The Cross-Section Method (CSM) introduced by Chourdakis et al., 2013 assumes that the proxy spread for a given entity is the product of five risk factors: a global factor, a sector factor, a region factor, a rating one and one that accounts for the seniority. Therefore, the model can be described by a log-linear regression relationship:

$$logS_i^{proxy} = \beta_0 + \sum_{j=1}^{N_{sec}} \beta_j^{sec} I_j^{sec}(i) + \sum_{k=1}^{N_{reg}} \beta_k^{reg} I_k^{reg}(i) + \sum_{l=1}^{N_{rat}} \beta_l^{rat} I_l^{rat}(i) + \sum_{m=1}^{N_{sen}} \beta_m^{sen} I_m^{sen}(i) \tag{2.5}$$

Here, $N_{sec}, N_{reg}, N_{rat}, N_{sen}$ represent respectively the number of available entities in the same sector, region, rating and seniority of the obligor i. Similarly $\beta_{sec}, \beta_{reg}, \beta_{rat}, \beta_{sen}$ are the regression coefficients for those categories. The $I(i)$ are indicator functions called also dummy-variable, e.g., I_j^{sec} will be one if entity i is in sector j, zero otherwise.
The beta coefficient are derived by Ordinary Least Squares estimation. Once the regression parameters are estimated, the model will provide proxies for the unobserved entities.
The CSM overcomes many of the issues presented in the Intersection method. Here

for an empty bucket, if there exists at least one liquid spread in a certain sector, region, rating or seniority, a proxy spread is derived. At the same time, CSM provides more stable proxy spreads over time.
Two other major improvements of the CSM are the monotonicity in the rating and the granularity of categories. The former refers to the fact that among the considered features, the rating is the strongest indicator for spreads. A higher rating, as explained in Section 2.1.2, implies a smaller spread and viceversa. CSM provided proxies that are generally monotonic in time w.r.t rating, e.g., 'AAA' rated bonds are below 'AA' for same region, sector and seniority, whereas for the Intersection method, this did not generally hold.
When choosing sector, region and rating categories, a problem called granularity of categories arises. The problem is a trade-off in which, if the categories are too thin, there will be too few liquid entities in some categories, while if categories are too broad, some important information can be lost. CSM can group categories in a finer way comparing to the Intersection method as it considers the categories separately instead of their intersection.
For all these reasons we chose the CSM as a starting point of our analysis and as a benchmark for the challenger models.
It is important to stress that, just like the Intersection method, CSM provides estimation by using data available on the same day. Hence, it does not look at the history of the spreads for the estimation.

2.4 Current Methods Restrictions and Targets of ML

In this section we underline the limitations introduced by the current methods and how Machine Learning algorithms may be able to overcome them. In particular we focus, first, on how we can produce better estimates within the daily regression, i.e. fitting the data separately day-by-day. And secondly on how including a multiple time-structure can improve the model. Since the CSM significantly improved the IM in terms of performance, we mainly focus on the limitations of the CSM to be addressed.

Nonlinearity: As we showed in the previous section, CSM relies on the assumption of a (log)-linear relationship between the target variables and its covariates.The logarithmic transformation was used in Chourdakis et al., 2013 to stabilize the variance of CDS z-spreads but, anyway, it implies a linear relationship between the log-spreads and the categorical features.
This model lacks both in capture the non-linear contributions of each categorical features and especially the interaction terms between these two. The interaction between categorical features was addressed in the IM method but it led to the granularity of categories issue mentioned in the previous section.
Machine Learning algorithms in general, are well known for their ability to learn complex non-linear functions and capturing interaction terms between categorical features without neglecting the singular linear or nonlinear terms introduced by each covariate. Taking into account these nonlinearities allows one to better replicate bond z-spread distributions and achieve better accuracy.

Time Structure: The aforementioned proxy methodologies do not consider any time structure in their models. Every day is analysed independently from the other days

and this can clearly be seen as a waste of information. Including the historical information of every bond is a fundamental additional feature for proxying bond credit spreads.
Machine Learning algorithms can help discovering similar patterns and behaviours among different bonds. This means that not only, past and future information for each singular bond can be used to fill-in missing data points, but also past, present and future information of similar bonds for which we have data.

Chapter 3

Data Framework and Exploratory Analysis

In this section we present the data set that has been used and explain the pre-processing steps and the selection of the categorical features that are used in the credit spreads proxy.
Afterwards, we explore the processed data set and describe its features. In this part, several graphs are presented in order to provide visual intuition about the data structure.

3.1 Data

The pricing data used in this thesis, i.e. the z-spreads, are obtained from the IHS MarkIT Generic Bond File, whereas the static data, the categories, are taken from various data sources: IHS MarkIT Bond Reference Data file, GRID and Bloomberg Credit Risk file. The use of more data providers for the static data aims to have categorical data as complete as possible.
The pricing and static data are then mapped and combined in order to identify the features of the quoted z-spreads used as input data for the proxy. The z-spreads of every issuer are then mapped with 7 categorical features: credit rating, region, sector, seniority, tenor, currency and country. More details about the mapping will follow.
For this analysis we consider corporate bonds and government bonds. Among corporate bonds we excluded callable and putable bonds due to the uncertainty around the maturity date which affects the tenor bucket for the regression.

Data Transformations

A useful technique in explanatory data analysis is the application of data transformation such as the logarithmic transformation, as performed by Chourdakis et al., 2013, for CDS spreads. However, our empirical analysis shows that the quality of the proxy results did not significantly improve running the regression on transformed data compared to the raw z-spreads, therefore we decided to not perform data transformation for what concerns the cross-sectional regression. For the Machine Learning algorithms implementation, the data have been standardized and this is discussed in Chapter 4.
However the following transformations have been implemented and then not applied, also because the main focus of this thesis concerns the prediction of shifts and not the levels:

Log Transformation:

$$z^{transf} = log(\frac{z+a}{b}) \tag{3.1}$$

Arcsinh Transformation:

$$z^{transf} = arcsinh(\frac{z+a}{b}) \tag{3.2}$$

Where z is the z-spread level, a is a displacement parameter and b a scaling parameter

Data Cleaning

The data cleaning procedure followed the indication of ING, 2018a technical documentation. Two data cleaning procedures have been performed. Firstly, we removed callable and putable bonds as previously mentioned, together with bonds that had matured before the final date in examination plus 30 days. This is done to avoid the extra-volatility of the spreads that bonds close to maturity experience. Defaulted bonds are also excluded.
The second layer of the data cleaning process aims at removing idiosyncratic bias from specific issuers. This means that in case multiple bonds of the same issuer and the same categorical features are present, we select only one. The selection criterion is the maturity date, i.e. we choose the bond that is closest to the respective tenor bucket. The following table is an example of this procedure.

Multiple Spreads Selection

Date	ISIN	Issuer	Maturity	Exact Tenor	Difference from 3Y Tenor Bucket	Selected
16/04/18	USG8200TAB64	China Petrochimical Corp	03/05/21	3.05	0.05	Y
16/04/18	USG8200TAG51	China Petrochimical Corp	29/09/21	3.46	0.46	N
16/04/18	USG8200TAA28	China Petrochimical Corp %	12/04/22	3.99	0.99	N
16/04/18	USG8200TAA03	China Petrochimical Corp %	12/04/22	3.99	0.99	N

TABLE 3.1: Procedure for the selection among multiple bonds with same issuer and categorical features.

In Table 3.1, there were 4 bonds that have the same categorical features. Only one has been selected in order to avoid an idiosyncratic bias to the regression coefficients. The selected bond is the one closer to the tenor bucket 3Y.

Input Data Structure

In this subsection, the categorical structure framework is presented. The buckets proposed in this part are clustered buckets, which means that more categories are grouped together.
Following Chourdakis et al., 2013 we select the categorical variables: credit rating, region, sector and seniority. In order to improve fitting we added extra categories such as: tenor, currency and a developed/emerging market indicator. The granularity of the categories and their meaning are presented in the following, Table 3.1.

Categorical Variables Framework			
Category	Description	Buckets	# of Buckets
Rating	Credit rating class	AAA, AA, A, BBB, BB, B, CCC	7
Region	Clustered area in which the bond belongs	Northwest Europe, Northern America, Oceania, AroundAfrica, Asia, Southern Europe, Latin America, Supranational	8
Sector	Operating area of bond issuer	Basics, CommTech, Consumer, Financial, Government Related, Government	6
Seniority	Order of repayment in case of default	Secured, Senior Unsecured,New Senior Non-Preferred, Subordinate, Junior Subordinate	5
Tenor	Time to maturity date	1Y, 2Y, 3Y, 5Y, 7Y, 10Y, 20Y, 30Y, 100Y	9
Currency	Currency of the quoted bond	EUR, USD, GBP, Others	4
EM/DM	Economic market classification	Emerging Market, Developed Market	2

TABLE 3.2: Description and bucketing procedure of each categorical feature in the data-frame.

For what concerns the EM/DM discrimination we followed the indications in FTSE, 2020, for the rest we kept the same structure as in ING, 2018a The table reports 41 unique categorical levels. Overall our data structure will be composed by $T = 522$ days (two years in business days), $N = 8119$ unique bonds ISIN, $M = 8$ variables: 7 categorical variables and 1 continuous variables, i.e. the z-spread. This data set can be better thought of as a 3D structure of dimension given by $(T \times N \times M)$.

3.2 Exploratory Data Analysis

The data-frame we just described is composed by $(522 \times 8119) = 4.229.999$ z-spreads. Among these 1.990.566 have been observed, which means that 52.9% of the data is missing. In figure 3.1, we provide an overview of the percentage of liquid entities across the time-span. The number of entities with complete historical data is 438 (entities with 1 or 2 missing data points are considered complete across our analysis). The number of available bonds starts around 46% in the summer of 2017, to reach its peak around 50% in the summer of 2018, to finally decrease at 45% at the end of the analysed period.The fluctuations are not large so the quality of our proxy

should be quite homogeneous across the 2 years in exam.

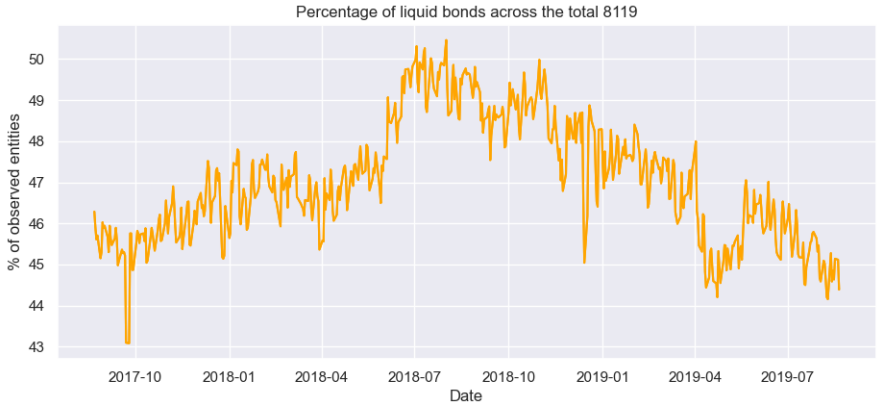


FIGURE 3.1: Percentage of available entities in the period from 21-08-2017 to 20-08-2019

To give more intuition about the liquidity of our buckets, in figure 3.2, we show the number of available entities across different buckets.
Figure 3.2 gives also an indication about which categories are included in the benchmark bucket. In fact, one problem in Chourdakis et al., 2013, was the missing prevention of multi-collinearity. Multi-collinearity happens when one or more predictor variables are a linear combination of other predictor variables. A practical solution to avoid this problem is to set a benchmark bucket of categorical levels, these levels are not part of the parameters estimated during the regression but they compose the intercept. Including the intercept and indicators for the others $n - 1$ levels of a categorical variable is called "reference level coding". More details on this and the one-hot encoding procedure can be found in Chapter 4.
It is common in reference level coding to use the largest category as the reference. Therefore, the categorical levels: rating 'A', region 'Northwest Europe', seniority 'SNRFOR' and sector 'Financial' are part of our benchmark bucket. More details are provided in the methodology section.
Figure 3.2 clearly presents the problem of empty buckets explained in the Intersectional method. In fact, in the figure there are some categorical levels with very few observations, e.g. rating:'CCC', regions: 'Latin America' and 'Around Africa', and these most likely produce empty buckets or buckets with very few spreads that are strongly affected by idiosyncratic bias, which is undesirable since the aim of the proxy is to incorporate only systematic risk.
Figure 3.3 shows averages of z-spreads for different categories, it highlights how different categorical levels affect the z-spread of bonds. Notice that to obtain the spreads in bps we need to multiply by a factor of 10 000. Another drawback of the Intersectional method and partially of the Cross Sectional method is the non monotonicity of the proxy with respect to the credit rating, i.e. a higher credit rating should always correspond to a lower z-spread. In figure 3.3, we see this property holds not only for credit rating but also for different market indicators. It is noticeable that also for different regions and tenors monotonicity holds for a considerable part of the time-span.
For what concerns, the rating class monotonicity does not hold only for the lowest rating levels 'CCC' and 'B'. The reason for this can be found in figure 3.2 where a

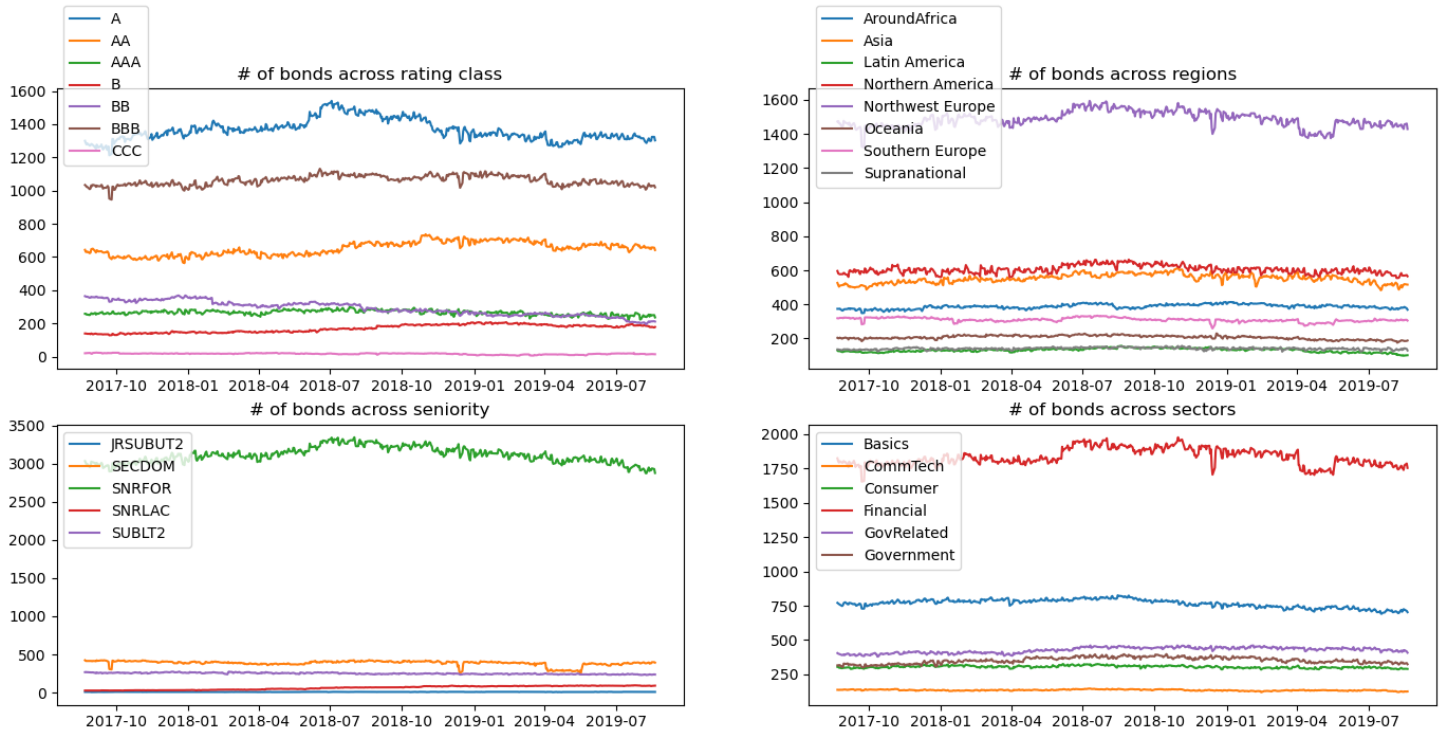


FIGURE 3.2: Number of available entities for different categorical features in the period from 21-08-2017 to 20-08-2019

really small number of 'CCC' rated bonds has been recovered and therefore the average z-spread fluctuations are large.

The top right sub-figure shows the z-spread mean across different regions and we can see that regions are associated with different quality of bonds. The clustered regions 'Around Africa' and 'Latin America' account for much higher z-spread, which unveil a higher risk for these products and a greater volatility, which is explained also by the low number of observed entities for these categorical levels. The lowest z-spread is obtained by 'Supranational' bonds, which are defined as those issued by entities formed by two or more central governments to promote economic development for the member countries. Sovereign bonds include sovereign guaranteed securities with an explicit government guarantee or support from the sovereign, principal or state governments and are therefore of greater quality.

The lower left sub-figure shows the net difference between z-spreads of emerging and developed markets. The categorical feature market indicator was not considered in Chourdakis et al., 2013. However, this graph shows that it is a really powerful discriminant, the two spread averages here, follow a really similar path, which indicates that systematic risk is captured rather than idiosyncratic bias, with the emerging market spreads being approximately 3 times larger than the developed market.

In the last sub-figure we can see that z-spreads are almost monotonic across different tenors, and a longer time to maturity corresponds a higher spread. Here, the '100 years' category level which includes bonds that have 65 or more years before expiring has very large fluctuations because of shortage of liquidity and even some time points with no data at all.

In this figure we presented 4 of the 7 categorical variables that we selected and already from this visual representation it is clear that these features yield strong discriminatory power and therefore are suitable as explanatory variables.

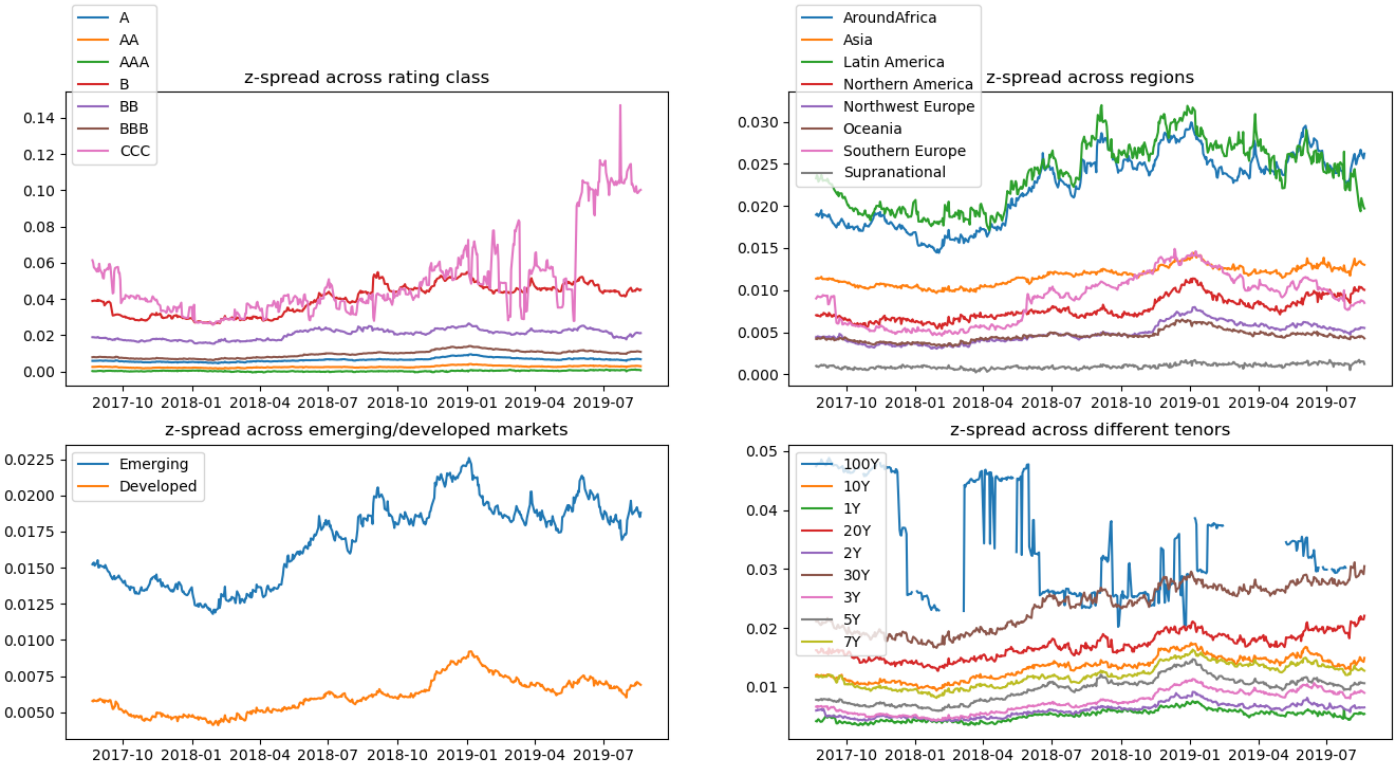


FIGURE 3.3: Mean of the z-spread across different categorical levels in the period from 21-08-2017 to 20-08-2019

Data Distribution

In order to better understand the behaviour of the z-spreads across our explanatory variables, we present the distribution of the z-spread mean across same category bonds and of the absolute shifted z-spread mean through violin plots. Violin plots are similar to box plots but additionally show the probability density of the data values, in our case the density of the z-spread averages across different categorical levels, smoothed by a kernel density estimator.
Figures 3.4 and 3.5 show the distribution of the z-spread means across different rating, region,seniority and market indicators. In particular for the rating classes we can observe that there is a net difference in the distribution of the spreads. Figure 3.3 already highlighted the difference in terms of z-spread mean across rating classes, but with a violin plot we can better understand how the probability distribution changes, in particular how the volatility is affected by the credit rating.
In the sub-figure below of 3.4 it is shown how the z-spreads of clustered regions 'Around Africa' and 'Latin America' are significantly more volatile than the other categorical levels, while on the other hand 'Supranational' bond z-spreads are really stable and clustered slightly above the zero level.
In the sub-figure below, the distributions of the means across different seniorities are presented. These distributions are more homogeneous among each other in terms of variance. The tick black bar that is more visible here represents the interquantile range, which means that 50% of the data points are contained in that range and the white dot corresponds to the median.
In the last sub-figure, distributions of the z-spread mean for emerging and developed markets are presented. It is noticeable how the developed market observations are generally more concentrated around the median, as described by the black bar which is definitely shorter than the one for emerging markets. The distributions just presented are not stationary and heavily skewed, which makes them less suitable for comparison purposes.
A more visually insightful plot is the one of the absolute shifts of z-spread means in figures 3.6 and 3.7. As mentioned in the introduction, the target of this thesis is to focus on proxying the z-spread shifts or daily changes, instead of proxying the z-spread levels. Therefore it is important to understand the behaviour of the z-spread shifts. We decided to present the absolute shifts, but similar plots are drawn for the other shift types in the analysis, i.e. displaced relative and arcsinh shifts.
In contrast to the z-spreads level, the z-spread shifts are stationary with zero mean and are characterized by a bell-shaped distribution, which makes the graphic interpretation more clear. The different characteristics across categorical levels are still noticeable.
Starting from the top sub-figure in figure 3.6, we see how the width of the shift distribution increases for lower rating classes. This feature is reflected in the proxy, the proxy is more robust in predicting higher rating classes as more bonds for these classes are observed and the intra-level variance is smaller.
Region-wise distributions are significantly different. A wider spread distribution is noticed for regions with fewer observed entities and lower credit rating, like 'Around Africa' and 'Latin America'. The more stable distributions are those of regions 'Northwest Europe' and 'Oceania' together with 'Supranational' bonds which as we discussed are considered the least subject to risk exposure. More volatile are the distribution of bonds from 'Asia', 'Southern Europe' and 'Northern America' with the last one presenting a noticeable negative skewness.
In figure 3.7, it is remarkable how shift volatility is much more clustered for secured

entities and gets wider for lower seniority, i.e. later order of repayment in case of default. This intuition was not seen from the z-spread mean violin plot as z-spread levels can vary a lot within the same seniority class. The z-spread shift mean violin plot, instead, shows how observations are narrowly focused around zero, i.e. small daily changes, for secured entities, whereas a wider dispersion is observed for the others seniority classes.
The bottom sub-figure of figure 3.7, presents the distributions of the mean absolute shifts for emerging and developed markets. It is noticeable how daily changes of bonds from emerging markets are bigger in magnitude comparing to the developed markets, this results in a net difference in the distribution variances.
All the considerations in this section provide a general understanding on the behaviour of z-spreads and mostly set the targets for the construction of a more realistic proxy for bond credit spreads.

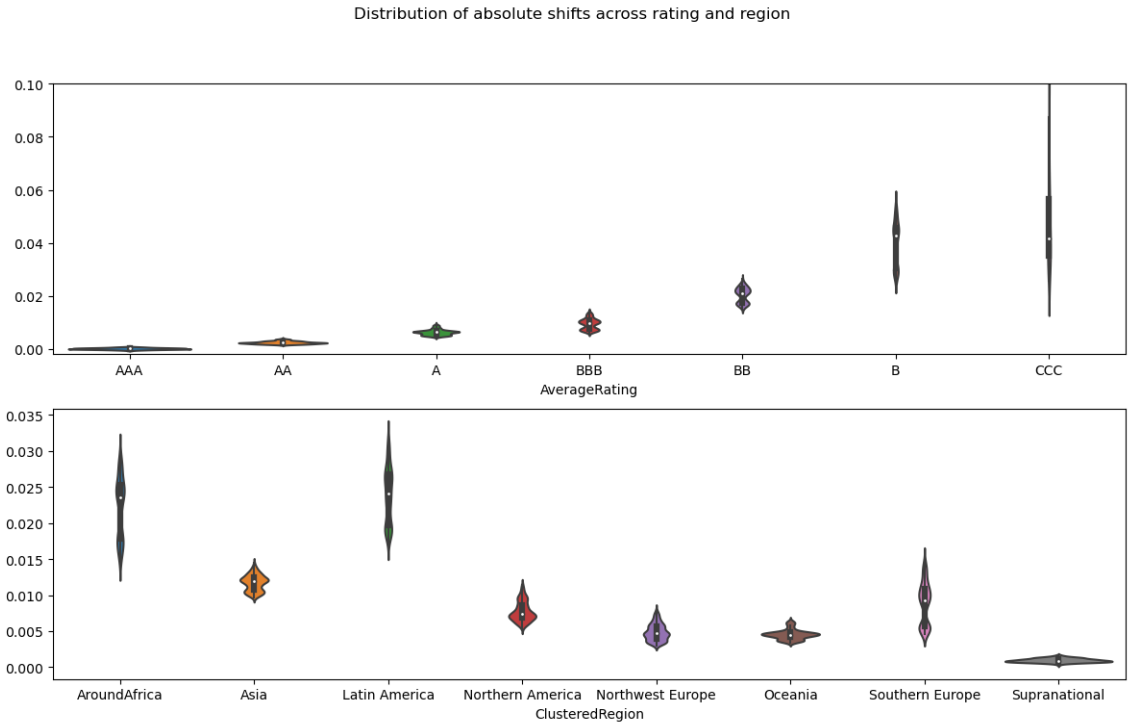


FIGURE 3.4: Violin plots of the z-spread mean across different rating classes and regions.

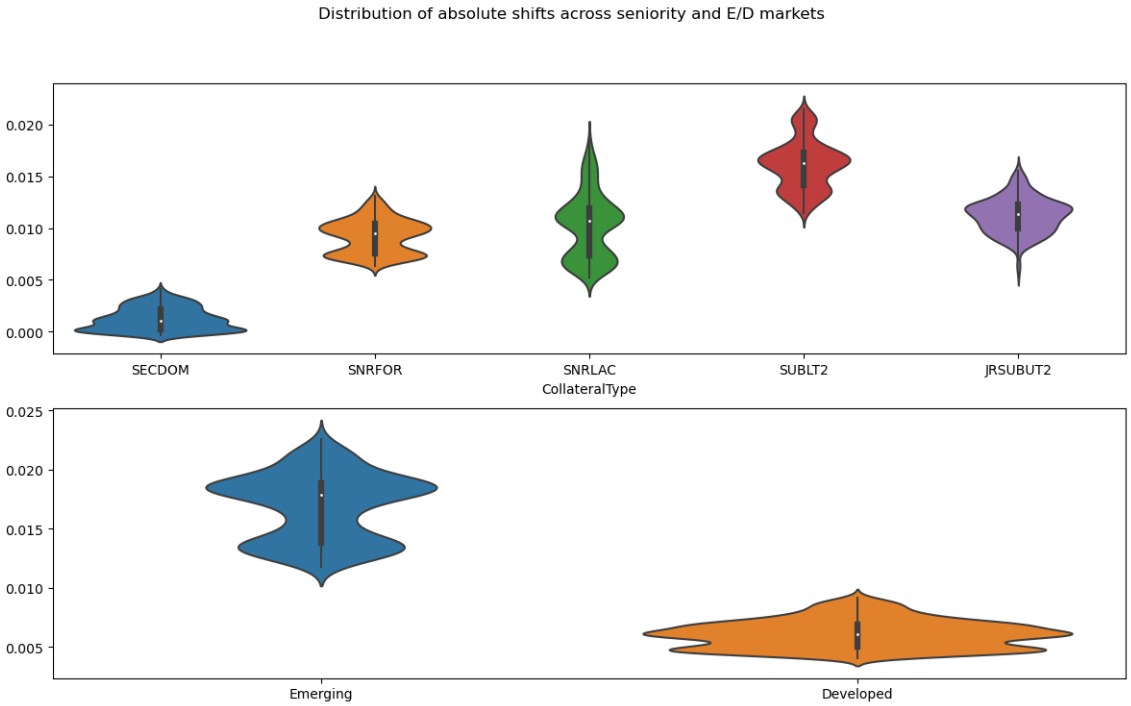


FIGURE 3.5: Violin plots of the z-spread mean across different seniority levels and market indicators.

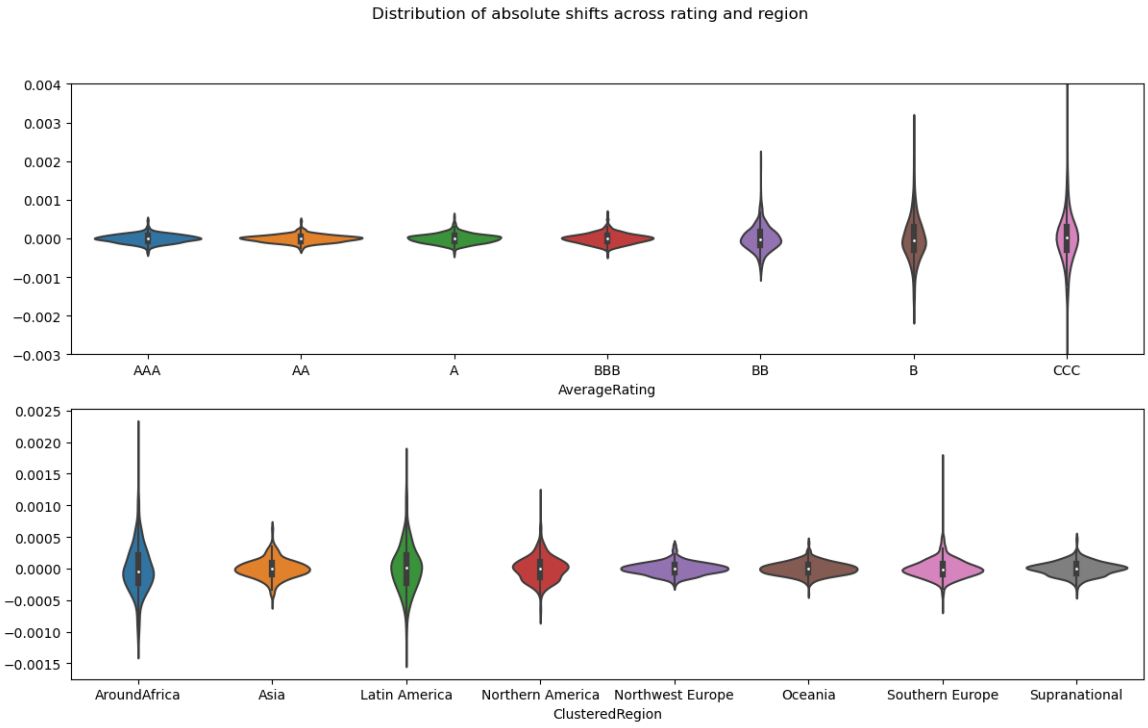


FIGURE 3.6: Violin plots of the absolute shifts of z-spread mean across different rating classes and regions. The y-axis of the top figure is cut to better show the probability distributions.

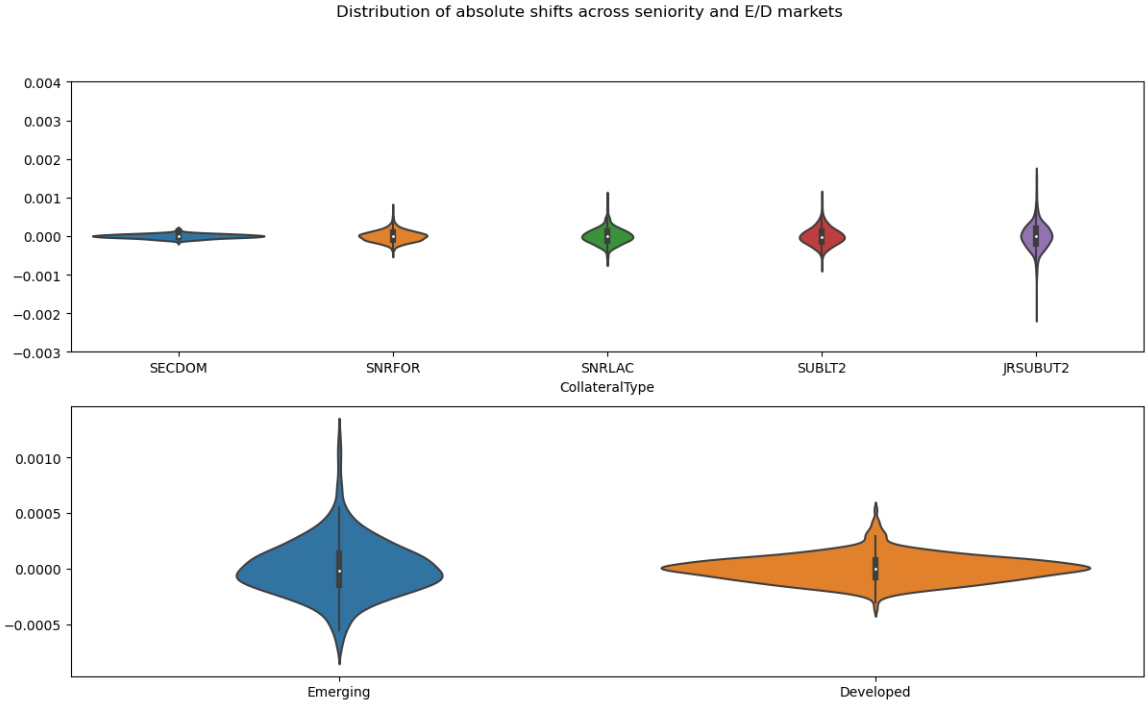


FIGURE 3.7: Violin plots of the absolute shifts of z-spread mean across different seniority levels and market indicators.

Chapter 4

Methodology

This chapter describes the methodologies that are used in this thesis in order to develop the bond credit spreads proxy. The first part concerns the Shift Type Assessment. It is not the main topic of this thesis but it is fundamental in order to correctly implement historical VaR, which is the final application of our proxy. Meanwhile, the second part is dedicated specifically to the construction of the credit spread proxy.

4.1 Shift Type Assessment Methodology

A crucial step in the HVaR calculation is the computation of the daily historical changes (shifts) in the risk factors. There are several ways in which shifts can be computed. For what concerns bond z-spreads, it was common for banks using relative shifts. However, recent ECB guidelines recommend the use of absolute shifts or a mixed approach such as the hyperbolic inverse sine shifts (arcsinh shifts) or displaced relative shifts. In this section the aforementioned shift types will be described together with the criteria to select the most suitable shift type. These criteria concern the implicit statistical assumptions that underline the HVaR model which depend on the chosen shift type, together with back-testing results and econometric tests.

Absolute Shifts: The use of absolute shifts is theoretically justified when the risk factor under examination has the same distribution over time, with constant mean and variance. This means that the dynamics of the risk factor are given by a random walk process:

$$X_t = \phi + X_{t-1} + \epsilon_t, \qquad \epsilon_t \sim i.i.d.(0, \sigma^2). \tag{4.1}$$

Here, $\{X_t\ , t \leq 0\}$ is the process that describes the risk factor over time, with X_0 given, ϕ is the drift and ϵ_t is a random noise of zero mean, variance σ^2 and unspecified distribution.
Absolute shifts determine the 260 scenarios for tomorrow's risk factors as follows:

$$x_{t+1}^{scn\ j} = x_t + \Delta x_{t+1-j}\ , \qquad j = 1, 2, ..., 260, \tag{4.2}$$

where $\Delta x_{t+1} = x_{t+1} - x_t$. There are two reasons why absolute shiftsHhVaR is the correct approach for risk factors described as in equation 4.1. First, the distribution of the risk factor changes coincides with the distribution of the historically observed absolute shifts, i.e.,

$$\Delta x_{t+1} \sim \Delta x_{t+1-j}\ , \qquad j = 1, 2, ..., 260. \tag{4.3}$$

And second, the independence assumption, i.e., for a random walk process, the day-to-day changes are independently distributed over time.

Relative Shifts: The use of relative shifts for 1-day hVaR scenario generation is theoretically justified if the dynamics of the risk factor are given by the following heteroskedastic random walk process:

$$X_t = \phi + X_{t-1} + \epsilon_t, \qquad \epsilon_t = \sigma X_{t-1} Z_t, \qquad Z_t \sim i.i.d.(0,1), \tag{4.4}$$

with X_0 given.
This means that the risk factor has constant mean and standard deviation proportional to the current level of the process.
In order to show that relative shifts are the correct approach for the dynamics in 4.4, we look at the dynamics of the shifts:

$$\Delta X_s | \mathcal{F}_{s-1} \sim (\phi, \sigma^2 X_{s-1}^2), \qquad \forall s \geq 0, \tag{4.5}$$

where $\{\mathcal{F}_t, \ t \geq 0\}$ is the natural filtration that contains $\{X_t, \ t \geq 0\}$ and $\Delta X_t = X_t - X_{t-1}$ as before. Now, since the variance of the absolute risk factor shift changes over time, we can divide the centered shift by the current process level to have:

$$\frac{\Delta X_s - \phi}{X_{s-1}} | \mathcal{F}_{s-1} \sim (0, \sigma^2), \qquad \forall s \geq 0. \tag{4.6}$$

The random variables above are i.i.d. and the value of the risk factor tomorrow can be written as:

$$\begin{aligned} x_{t+1} &= x_t + \Delta x_{t+1} \\ &= x_t + \phi + x_t \left(\frac{\Delta x_{t+1} - \phi}{x_t} \right), \qquad \forall t \geq 0. \end{aligned} \tag{4.7}$$

Then, the correct historically generated scenarios are given by:

$$x_{t+1}^{scn\ j} = x_t + \phi + x_t \left(\frac{\Delta x_{t+1-j} - \phi}{x_{t-j}} \right), \qquad j = 1, 2, ..., 260, \tag{4.8}$$

and when there is no drift, i.e. the mean is zero, we see that relative shifts are the correct approach to model this type of process:

$$\begin{aligned} x_{t+1}^{scn\ j} &= x_t + x_t \left(\frac{\Delta x_{t+1-j}}{x_{t-j}} \right) \\ &= x_t \left(1 + \left(\frac{\Delta x_{t+1-j}}{x_{t-j}} \right) \right) \\ &= x_t \left(1 + \Delta x_{t+1-j}^{rel} \right), \qquad j = 1, 2, ..., 260. \end{aligned} \tag{4.9}$$

Relative shifts are typically used when the risk factor has large absolute values and does not cross the zero level. This is not the case for z-spreads as these can take negative values.

Arcsinh Shifts: Arcsinh shifts are included in the category of shifts that the ECB defines as mixed type. The mixed type is a generalization of absolute and relative shifts. Arcsinh shifts are based on the inverse hyperbolic sine transformation

and behave like absolute shifts for risk factors close to zero and like relative shifts for distant levels.
The arcsinh shift is defined by:

$$\Delta x_{t+1}^{arcsinh} = arcsinh(\frac{x_{t+1}}{b}) - arcsinh(\frac{x_t}{b}), \tag{4.10}$$

where $b > 0$ is a constant scaling factor. An arcsinh shift behaves like an absolute shift for levels that are much lower than b, i.e., $x_t << b$, and as relative shifts for high levels: $x_t >> b$. This is because the arcsinh is a strictly increasing function which has a linear asymptotic for small arguments and a logarithmic asymptotic for large arguments. Therefore, parameter b determines the transition level between the regions of small and large levels of the risk factor and it has to be opportunely tuned. The simulation equation for this type of shift is given by:

$$x_{t+1}^{scn\ j} = b\ sinh(\Delta x_{t+1-j}^{arcsinh} + arcsinh(\frac{x_t}{b})), \qquad j = 1, 2, ..., 260. \tag{4.11}$$

Displaced Relative Shifts: Displaced relative shifts are also a mixed type of shift between absolute and relative.
The displaced historical simulation model is designed to handle negative and close-to-zero risk factors. This is an issue of recent and major interest to the financial sector, both from a regulatory and financial institution perspective, especially in light of observed negative values for bond yields and interest rate spread time series. In historical simulation a common approach is to consider log returns (which are relative changes) given that the risk factors remain positive (Fries, Nigbur, and Seeger, 2017). However, for spreads, i.e., quantities that are by definition differences, e.g. differences of interest rates, it is important to realize that a relative change may not make sense. Because spreads may become negative and therefore they have a vertical asymptotic for values of spreads that are close to zero. This is visually described in Chapter 5.
Displaced relative changes interpolate between absolute and relative changes. The displaced relative shift is defined as:

$$\begin{aligned} \Delta x_{t+1}^{disp} &= \frac{(x_{t+1} + a) - (x_t + a)}{(x_t + a)} \\ &= \frac{x_{t+1} - x_t}{(x_t + a)}. \end{aligned} \tag{4.12}$$

The formula above means that we apply relative shifts on a displaced variable $x_t + a$ where $a > 0$ is a displacement parameter. The formula for the historical simulation is easily derived and it shows how this type of shifts is defined as an interpolation

between absolute and relative changes:

$$\begin{aligned} x_{t+1}^{scn\ j} &= (x_t + a)(1 + \Delta x_{t+1-j}^{disp}) - a \\ &= x_t + (x_t + a)\frac{x_{t+1-j} - x_{t-j}}{(x_{t-j} + a)} \\ &= x_t\big(1 + \frac{x_{t+1-j} - x_{t-j}}{(x_{t-j} + a)}\big) + \frac{a}{(x_{t-j} + a)}(x_{t+1-j} - x_{t-j}) \\ &= x_t\big(1 + \frac{x_{t+1-j} - x_{t-j}}{x_{t-j}}\big)\frac{x_{t-j}}{(x_{t-j} + a)} + (x_t + (x_{t+1-j} - x_{t-j}))\frac{a}{(x_{t-j} + a)} \\ &= \gamma\left[x_t\big(1 + \frac{x_{t+1-j} - x_{t-j}}{x_{t-j}}\big)\right] + (1 - \gamma)\left[x_t + (x_{t+1-j} - x_{t-j})\right] \\ &= \gamma\left[x_t\big(1 + \Delta x_{t+1-j}^{rel}\big)\right] + (1 - \gamma)\left[x_t + \Delta x_{t+1-j}^{abs}\right], \qquad j = 1, 2, ..., 260, \end{aligned} \tag{4.13}$$

where $\gamma = \frac{x_{t-j}}{(x_{t-j}+a)}$.
Equation (4.13) shows that displaced relative shifts form is a linear interpolation between absolute and relative changes applied to the process without displacement. It is easy to check that we can recover the two limiting cases $a = 0$ for relative shifts and $a = \infty$ for absolute shifts. Similarly, as with the arcsinh shift, we have a parameter that has to be opportunely tuned.

4.1.1 Performance Testing

The new market risk calculation in the Fundamental Review of the Trading Book (FRTB) framework is based on Expected Shortfall. However, there is not a widely accepted methodology to backtest Expected Shortfall. Therefore, we choose to adopt VaR backtesting which is required as backtesting methodology in the FRTB framework and it is the same methodology that is going to be used within the bond credit spreads proxy.
To measure the risk factor return performance under each shift type we used two standard backtesting methods: the Likelihood Ratio (LR) statistics for unconditional coverage and LR statistics for statistical coverage. Furthermore we performed three types of econometric tests for each shift type.

Backtesting Methods

In the context of risk management, backtesting is a typical way to measure the performance of a model generated P&L distribution in comparison to the P&L observed in the market. Shift type selection is an essential part of scenarios generation and directly contributes to the P&L predictions. The testing strategy relies on comparing daily risk factor shifts predicted by the model with the real ones to assess if the model captures the underlying dynamics.
The P&L for a risk factor x on a day $t + 1$ is defined as:

$$P\&L_{t+1} = x_{t+1} - x_t, \tag{4.14}$$

whereas the 260 scenarios described for example in equation (4.2) generate a forecast P&L distribution:

$$P\&L_{t+1}^{scn\ j} = x_{t+1}^{scn\ j} - x_t, \qquad j = 1, 2, ..., 260, \tag{4.15}$$

VaR(99%) is calculated as linear interpolation between the second and third lowest P&L values. For completeness we calculated also VaR(1%) similarly such that both tails are tested. This because, in a trade, both long and short positions of the risk factor could be held.
Since forecast P&L is based on 260 previous observations, VaR backtesting needs one year of data, i.e. 260 business days, for calibration. This means that the first year of data is not used for backtesting, but it is only used to calibrate the VaR model.
In the following we provide a detailed description of these two tests.

LR statistics for Unconditional Coverage: This test allows to check if the number of exceptions, i.e. the observations that are more extreme than our VaR estimation, agrees with the expected number predicted by the model. The test is assuming that exceptions occur independently over time and to check the correctness of this assumption we have the LR statistics for a statistical independence test.
We define a hit sequence as an indicator variable $I_t(\alpha)$, which takes the value 1 if the $P\&L$ on day t is more extreme than the VaR prediction, i.e. the loss is bigger than the VaR(99%) or the profit is higher than the VaR(1%) estimate:

$$I_t(\alpha) \begin{cases} 1 \;\; if \;\; P\&L_t > VaR_t(\alpha), \\ 0 \;\; if \;\; P\&L_t \leq VaR_t(\alpha). \end{cases} \tag{4.16}$$

Here α is the significance level and it is set at 1% and the confidence level is $1-\alpha$. This is because the definition of the $P\&L$ given in (4.14) implies positive values of the $P\&L$ for profits and negative for losses. Now, in order to define the likelihood ratio, we define $\hat{\alpha} = \frac{N_e}{N}$ as the percentage of violations observed, where N is the number of days in the backtesting window and N_e is the number of exceptions which is calculated as:

$$N_e = \sum_{t=0}^{N} I_t(\alpha). \tag{4.17}$$

We can now define the likelihood ratio statistics for unconditional coverage LR_{UC} and take the $-2log(LR_{UC})$ which is approximately centrally chi-squared distributed with one degree of freedom, see Romano and Lehmann, 2005 for more details:

$$LR_{UC} = -2log\left(\frac{(1-\alpha)^{N-N_e}\alpha_e^N}{(1-\hat{\alpha})^{N-N_e}\hat{\alpha}^{N_e}}\right) \sim \chi^2(1,0). \tag{4.18}$$

The value of the LR_{UC} is tested against a $\chi^2(1,0)$ distribution and values larger than 3.84, which correspond to a p-value of 5% indicate statistically significant differences between the realized and expected number of tail events.

LR statistics for Statistical Independence: This test checks whether VaR exceptions occur independently over time, which is another important feature for a reliable hVaR model. A failed test indicates that the exceptions tend to cluster, i.e. exceptions are not homogeneously distributed over time.
This test is again a likelihood ratio and it looks for unusually frequent consecutive exceptions, e.g. $I_t(\alpha) = I_{t+1}(\alpha) = 1$.
We define N_{ij} as the number of cases in which $I_t(\alpha) = j$ and $I_{t+1}(\alpha) = i$. Here, for example, N_{11} is the number of consecutive hVaR violations.

Then, we define:

$$\begin{aligned}\hat{\alpha}_{01} &= \frac{N_{01}}{N_{00}+N_{01}},\\ \hat{\alpha}_{11} &= \frac{N_{11}}{N_{10}+N_{11}}.\end{aligned} \tag{4.19}$$

If our null hypothesis is true, i.e. $Pr(I_t = 1 \mid I_{t-1} = 1) = Pr(I_t = 1 \mid I_{t-1} = 0)$, we would have that the same holds for the estimators of these two probabilities: $\hat{\alpha}_{01} \sim \hat{\alpha}_{11}$, see Christoffersen, 1998 for more details.
We can then set the likelihood ratio statistics for statistical independence LR_{IND} in a similar fashion as before and take $-2log(LR_{IND})$ that is approximately centrally chi-squared distributed with one degree of freedom:

$$LR_{IND} = -2log\Big(\frac{(1-\hat{\alpha})^{N-N_e}\hat{\alpha}^{N_e}}{(1-\hat{\alpha}_{01})^{N_{00}}\,\hat{\alpha}_{01}^{N_{01}}\,(1-\hat{\alpha}_{11})^{N_{10}}\,\hat{\alpha}_{11}^{N_{11}}}\Big) \sim \chi^2(1,0). \tag{4.20}$$

Again we reject at the 5% significance level the null-hypothesis that exceptions are homogeneously distributed over time if $LR_{IND} > 3.84$, which is the 95% quantile of the $\chi^2(1,0)$ distribution.

Econometric Tests

Several econometric tests can be performed on the historical data to investigate if the evolution of the transformed risk factors, e.g. absolute shifts, arcsinh shifts, can be reasonably described by a random walk process and to understand why a certain shift type can be a better fit for the underlying distribution. These types of tests are performed for every bond as for the backtesting tests.
Below a brief description of the scope of each test is provided:

White Test: It is the most important econometric test for our scope and it tests the null hypothesis that the error term is homoschedastic in a regression model, i.e. the variance of the errors is constant with respect to the independent variable. In particular, in this assessment we are regressing the z-spread shifts on the levels. We test if the shifts of the risk factor have a variance term which is correlated with the risk factor level. In that case the null-hypothesis is rejected. Otherwise, the result of the test would support the presence of a constant variance in the shift term and therefore the assumption that shifts are i.i.d (independent identically distributed). This is a key component of the HVaR procedure because the previous 260 transformed risk factors or shifts are used to predict new possible scenarios and hence they should not depend on the level of the risk factor. The White test performs an auxiliary regression, which means that after the primary regression of the shifts with the z-spreads level as independent variable there is a second regression, where the dependent variable is the squared of the residuals from the primary model and the explanatory variables are the z-spread levels, the square levels and a constant vector. One then inspects the R^2 coefficient of the auxiliary regression. The Lagrange Multiplier (LM) test statistic is the product of the R^2 value and sample size and it follows a chi-squared distribution, with degrees of freedom equal to P - 1, where P is the number of estimated parameters in the auxiliary regression, which in our case is $3-1=2$, i.e.,

$$LM = nR^2 \sim \chi^2(2,0), \tag{4.21}$$

with n as the sample size. A p-value < 0.05 allows us to reject the null hypothesis that shifts are homoschedastic with respect to the levels at a significance level of 5% .

Breush-Pagan (BP) test: An alternative to the White test is the Breusch–Pagan test, the Breusch-Pagan test is designed to detect only linear forms of heteroskedasticity. It can be described in the same way as the White test, but it does not include squared levels of the z-spread in the explanatory variables.
In this assessment it is used to confirm the result of the White test, which is the main driver to tune the shifts parameters, the procedure is explained in Chapter 5.

Augmented Dickey-fuller (ADF) and Phillips Perron (PP) tests: ADF and PP tests allow us to test the null hypothesis that a unit root is present in the sample. The Phillips-Perron test is robust with respect to unspecified autocorrelation and heteroschedasticity in the process. Therefore, the PP test has been performed on the levels when relative (and displaced relative) shifts were used, while ADF has been used when absolute and arcsinh shifts were calculated. Unit root tests assess the null hypothesis that the transformed risk factor has a random-walk behaviour. The failure to reject the null hypothesis will be in support of our random walk assumption. Both ADF and PP are designed in this assessment to verify whether we should simulate future risk factor values by applying historical shifts on top of the current risk factor level or otherwise, for instance, by directly using historical risk factor levels as a simulated future risk factor value (when the null hypothesis is rejected).

4.2 Credit Spreads Proxy Methodology

4.2.1 Evaluation Criteria

In order to compare and evaluate the models presented in this thesis we need to establish the most suitable performance measures.
For every type of mathematical problem there is a different evaluation framework which is more appropriate. In this case we are tackling a regression model. In regression problems the target variable y is a continuous output of its predictors X. Regression models include algorithms such as Linear Regression, Decision Tree, Random Forest, SVM, Gradient Boosting. The most relevant and widely used metrics that can be used to evaluate a regression model performance are described below.

RMSE: The Root Mean Squared Error represents the sample standard deviation of the residuals i.e., the differences between predicted values $\hat{y}$ and observed values y. It measures the average error given by the model in predicting the outcome for an observation. The RMSE can be mathematically defined as:

$$RMSE := \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \tag{4.22}$$

where N is the number of observations of the target variable y. Lower values of RMSE indicate a better fit. RMSE is a measure of how accurately the model predicts the response.

R^2: The R-squared coefficient is the proportion of the variance in the dependent variable y that is predictable from the independent variables X. It is known as the coefficient of determination. It is a statistical measure of how close the data is to the fitted regression line. The R^2 coefficient is a powerful tool to understand how well the independent variables explain the variance in the model and it is calculated as:

$$R^2 := 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2}, \tag{4.23}$$

where $\bar{y}_i$ is the mean of the observed target variable y. The maximum value for the coefficient of determination is 1. An R^2 coefficient close to 1 indicates that the model explains well the variability of the response data around its mean. On the other side, a coefficient of determination close to 0 implies that our model performs similarly as if we only took the mean of the observable variable y and therefore the model performs poorly.

The aforementioned metrics are sensible to the inclusion of additional variables in the model, even though those variables do not add significant contribution in explaining the outcome. This means that adding additional variables in the model will always increase the R^2 coefficient and decrease the RMSE. Therefore, we need something to penalize the complexity of the model.
Information criteria provide an analytical technique for scoring and choosing among candidate models in which models are scored both on their performance and on their complexity.

AIC: The Akaike Information Criterion is calculated using Maximum Likelihood Estimation (MLE) and is derived from frequentist probability.
Since in regression problems using MLE and minimizing the Mean Squared Error (MSE) leads to the same result, we can calculate AIC as in Gordon, 2015:

$$\begin{aligned} AIC &:= 2K + N\, log(MSE) \\ &= 2K + N\, log(\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2), \end{aligned} \tag{4.24}$$

where K is the number of independent variables and N the number of observations. AIC deals with the trade-off between the goodness of the fit of the model and the simplicity of the model. Generally, the model with the lowest AIC value is preferred. AIC is composed by two terms: one is the error that we want to minimize (goodness of the fit), the other is a penalty for the number of independent variables used in the model (simplicity). AIC has been particularly useful to evaluate if including additional features within the same model could provide significant improvements.

Other evaluation criteria that must be taken into account are those more related to the financial model than the regression problem itself.
From EBA, 2016 article 44 part 2: "*The institution's proxy selected does not underestimate the volatility of the missing risk factor*" which in this case is the Z-spread.
Therefore, we focused on the following properties for the construction of the proxy.

Conservative: The proxy should be appropriately conservative, i.e. does not underestimate the volatility of the proxied risk factors.

Similarity: The shifts or daily changes of the available spread series and the shifts of the proxy should be comparable which can be assessed in two parts: the correlation between the data series and the proxy should be high, the level of the volatility should be similar.

We focused on the correlation between shifts instead of levels because with levels we most likely have spurious correlation since the values for the level are not centered around zero whereas the shifts have zero mean. Therefore, a higher correlation between shifts implies a model that better approaches reality.
The other 2 measures that are used to measure the performance of our model are:

Correlation: Since we are dealing with shifts and not directly with z-spread levels, correlation turns out to be a useful tool to assess the quality of our proxy.
The measure used throughout the analysis is the Pearson correlation coefficient, which measures linear correlation between the two variables: observed target variable $y = \{y_1, ..., y_N\}$ and predicted variable $\hat{y} = \{\hat{y_1}, ..., \hat{y_N}$. Pearson's coefficient is given by the covariance of the two variables divided by the product of their standard deviations:

$$corr_{y,\hat{y}} := \frac{cov(y, \hat{y})}{\sigma_y \, \sigma_{\hat{y}}.} \tag{4.25}$$

Standard Deviation Spread: As indicated from EBA, the standard deviation of the proxy estimates $\sigma_{\hat{y}}$ should not be lower than the true standard deviation of the sample σ_y. In practice the proxied variables fluctuations calculated with our regression approaches are necessary smaller than the true ones. This is not a problem as we are interested in capturing the average market risk, or systematic risk and not the specific counterparty risk, i.e. idiosyncratic risk.
However, knowing the Standard Deviation Spread: $\Delta\sigma := \sigma_y - \sigma_{\hat{y}}$ gives an indication of the magnitude of the volatility underestimation, which can then be used in order to simulate idiosyncratic risk for VaR calculations.

4.2.2 K-Fold Cross Validation

A standard procedure for evaluating Machine Learning models is to split the data into training and test sets. The model is trained on the training set and then evaluated on the test set using performance evaluation metrics as those described in the previous section. The classic approach is to do a 80% / 20% split between training and testing set. This procedure drastically reduces the amount of out-of-sample data available for performance testing which can be a problem if the data set is not large enough.
Since our research is about missing data proxying, it is necessary to use all the data available without wastage. Data is often limited and assessing the model performance only on an independent sub-sample of the data-set is not ideal to test if the model correctly generalized the data. Generalization error can be better measured when the model is fitted and tested on multiple independent sub-samples of the original data-set.
K-Fold Cross Validation provides a solution to this issue by randomly splitting the data into K subsets and using every subset once as testing set during the procedure. The $K-1$ subsets are used to fit the model which is then evaluated on the excluded

subset. The procedure is repeated K times and then the whole data-set is obtained as an out of sample prediction. In this way the model performance is tested over the whole data-set.
The K-Fold Cross Validation procedure is illustrated in Figure 4.1 below for K=5. The choice of the value of K in the cross validation is associated with a bias-variance

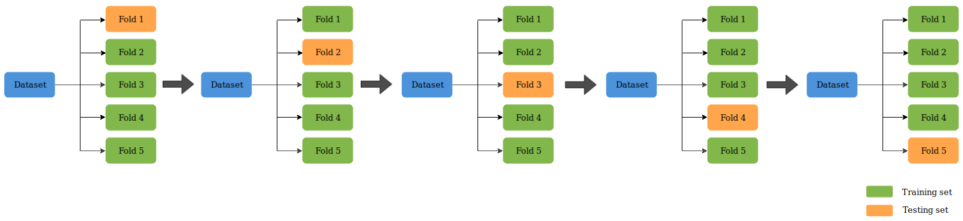


FIGURE 4.1: K-Fold Cross Validation for K=5 in yellow the test set and green the training sets.

trade-off. Typical choice are to choose K between 5 and 10 as these values have been shown empirically to yield test error rate estimates that suffer neither from high bias nor high variance, James et al., 2013.
Once applied the K-Fold Cross Validation procedure, the error metrics defined before, such as RMSE, R-squared coefficient and Pearson correlation coefficient, can be calculated on each of the K In-Sample and Out-Sample sets.
Any of the metrics described before can be defined as: $h(y, \hat{y})$ and the K-Fold Cross-Validation metrics for the in and out samples are given by:

$$\begin{aligned} h_{in}^{CV}(k) &:= \frac{1}{k}\sum_{i=1}^{k} h(y_{in}^{i}, \hat{y}_{in}^{i}), \\ h_{out}^{CV}(k) &:= \frac{1}{k}\sum_{i=1}^{k} h(y_{out}^{i}, \hat{y}_{out}^{i}), \end{aligned} \tag{4.26}$$

where y_{in}^{i} are the elements of the i^{th} fold that are used for training the algorithm.
During our analysis a K-Fold Cross Validation with $K = 10$ improved the Out-of-Sample accuracy comparing to other standard values for K such as K=5. Therefore, all the models presented in this thesis use 10 folds for the cross validation.

4.2.3 Different Machine Learning Procedures

In this subsection and the coming ones we introduce the machine learning algorithms that are used in this thesis through a waterfall approach. Starting from the more general categorizations of machine learning algorithms we narrow the focus on those sub-categories that are relevant for this thesis.
The main feature that characterizes machine learning algorithms is learning from experience. The learning happens when the algorithm is fed with a large data set (training set) and the algorithm can use the information present in this data set to train itself. This process can be done in several ways, but two main branches can be identified that yield very different approaches: unsupervised learning and supervised learning.

Unsupervised Learning
Unsupervised learning is a machine learning algorithm used to draw inferences from a data set of input data without labeled responses, i.e. without a given target variable. The goal of unsupervised learning is to extract similarities and recurrent patterns within the data set and then cluster the data according to these findings.

Supervised Learning
In supervised learning algorithms the output variable is known. This means that there is a mapping of input and output data that are fed to the algorithm such that it can learn this mapping and apply it to general new input data.
It is called supervised learning because the process of an algorithm learning from the training data set can be thought as a teacher supervising the learning process.
The problem we are dealing with is a supervised learning problem, in which the input data are the categorical variables, e.g. credit rating, region, and the output data are the shifts (absolute,displaced relative or arcsinh) of the z-spreads.
Supervised learning problems can be further grouped into two type of problems: regression and classification problems.

Classification Problems
In classification problems the task is to approximate a mapping function of the input variables X, to a discrete class of labels $f(X) = y \in \{y_1, ..y_n\}$.
An example of binary classification, which is a classification problem with two class labels, is to assess if it would be safe for a financial institution to grant a mortgage. The training input set in this case would be a set of categorical and continuous variables associated to the borrower like age, employment type, marital status and the training output set would be a binary variable indicating whether the mortgage was finally repaid.
In our case, since z-spread shifts can in principle assume any value and therefore do not belong to a discrete set, we are not dealing with a classification problem.

Regression Problems
The bond credit spreads proxy belongs to the class of regression problems, in which the output variable can take continuous values. A regression problem with multiple input variables, as the 7 different categories we presented, is often called a multivariate regression problem. For this type of problem, since the output variable is a quantity, i.e. a real number, there is a vast tool-kit of evaluation metrics that can be used such as those described in Section 4.2.1. The simplest and most popular regression algorithm is linear (or multiple linear) regression, like the model

used in Chourdakis et al., 2013 which is the benchmark model for this thesis. In multiple linear regression, several explanatory variables are used to predict the outcome of a target variable. In our analysis the explanatory variables $x_{1,i}, ...x_{k,i}$ are the categorical variables such as rating, region or tenor for the i^{th} bond and y_i is the shifted z-spread of the same bond. The model in this case for k explanatory variables can be formulated as:

$$y_i = \beta_0 + \beta_1 x_{1,i} + ... + \beta_k x_{k,i} + \epsilon, \tag{4.27}$$

where β_0 is the intercept, β_j is the slope coefficient of the j^{th} category and ϵ is the error term or residual. Equation (4.27) assumes that there is a linear relationship between the dependent variables and the independent variable, i.e. it does not consider nonlinearities in the model. Others assumptions of linear regression models are that the independent variables are not highly correlated between each other, the y_i observations are selected independently and randomly from the population and the residuals are normally distributed with zero mean.
The residual $\epsilon_i = y_i - \hat{y}_i$, is the difference between the true value of the dependent variable and the predicted one. Linear regression estimates the parameters β such that the sum of the squared residuals $SSR := \sum_{i=1}^{n} \epsilon_i^2$ is minimized, where n is the number of data points. This procedure is called Ordinary Least Squares approximation (OLS).
The vector containing the β coefficients is therefore estimated as:

$$\widehat{\beta} = (X'\ X)^{-1} X'\ y. \tag{4.28}$$

Under the assumption that the error term has constant variance, the residual variance can be estimated as:

$$\widehat{\sigma_\epsilon} = \frac{SSR}{n-k-1}. \tag{4.29}$$

Therefore in a multiple linear regression model the error terms are assumed to be independent identically distributed as $\epsilon_i \sim \mathcal{N}(0, \widehat{\sigma_\epsilon})$.
Multiple linear regression is the beginning of our analysis. All more complex algorithms that are presented in this thesis: *Catboost*, *Random Forest* and *Support Vector Machines*, are described in the following subsections and still belong to the regression problem algorithms.

4.2.4 Ensemble Learning

Ensemble learning models make predictions based on several different models. By the combination of individual models, often called 'weak learners', the ensemble model is more flexible. Therefore, it allows for a lower bias and also it results in a lower variance.
The goal of learning algorithms is to approximate an underlying function f that maps the matrix of features X to a label or a continuous output y. Ordinary machine learning algorithms search in a space of possible functions, called hypothesis, to find the one $\hat{f}$ that optimally approximate the unknown function f.
Ensemble models can overcome three major problems of ordinary learning algorithms: the statistical problem, the computational problem and the representation problem.
The statistical problem is present when the algorithm searches a space of functions or hypothesis that is too vast for the number of training data available. In this kind of situation, there could be many different functions that have the same performance in terms of RMSE, R^2 coefficient or others error metrics. The algorithm has to select one among these and the choice might not be optimal for a new data set. A learning algorithm that presents this type of issue is said to have high variance. Ensemble learning overcomes this issue incorporating all these functions in a 'democratic' voting system.
The computational problem emerges when the algorithm cannot select the best approximating function in the function space. For example with neural networks and decision tree algorithms it is computationally cumbersome to find the function that best fits the training data, therefore heuristic methods like gradient descent have to be implemented. These methods can get stuck in local minima and hence fail the search for the best function. An algorithm that exhibits this problem is described as having 'computational variance'. Within ensemble learning, a weighted combination of different local minima reduces the risk of selecting the wrong local minimum.
The representation problem arises when the function space does not contain good approximations of f. An algorithm that suffers from the representation problem is said to have high bias. Within ensemble learning, a weighted sum of these approximation functions can expand the function space such that the algorithm may be able to obtain a more accurate approximation function $\hat{f}$.
This is why ensemble methods can reduce both bias and variance of learning algorithms, this has empirically been shown (Liu and Yao, 1999).
Two most popular ensemble methods are *bagging*, which includes Random Forest and *boosting* that includes Catboost.

Bagging

Bagging stands for bootstrap aggregating and it is one of the simplest and most intuitive ensemble learning based algorithms. The diversity of classifiers in bagging is obtained through bootstrapped samples from the training data. Different training data are randomly drawn, with replacement, from the entire training set. Each data subset is used to train a different decision tree, which is an algorithm that only contains conditional control statements. These individual trees or weak learners are then combined by taking the majority vote of their decisions in case of classification problems, or by averaging the outputs in case of regression problems to obtain the output of the ensemble model.
Bagging considers homogeneous weak learners and learns independently from each

other in parallel combining their output. In this way, bagging not only contributes to reduce the variance, but it can help avoiding overfitting as well.

Boosting

Similarly as bagging methods, boosting methods build a family of weak models that are aggregated to obtain a strong learner that outperforms them. However, while bagging mainly aims for reducing the variance, boosting fits multiple weak learners sequentially and not independently. Each model in the sequence is fitted giving more importance to the observations for which the previous weak learners performed worse, i.e. wrong classification for classification problems and high RMSE for example in regression problems. Because of this process the stronger learner has, not only a lower variance like with bagging method, but specially a lower bias.
Since boosting is mainly focused on reducing bias, the base models considered for boosting are generally models with low variance and high bias. This choice is motivated by the fact that weak learners with low variance and high bias are generally less computationally expensive for fitting as they have fewer degrees of freedom for the parameterization. In fact, since computations to fit different models are not done in parallel, unlike in bagging, it is computationally expensive to fit sequentially several complex models.

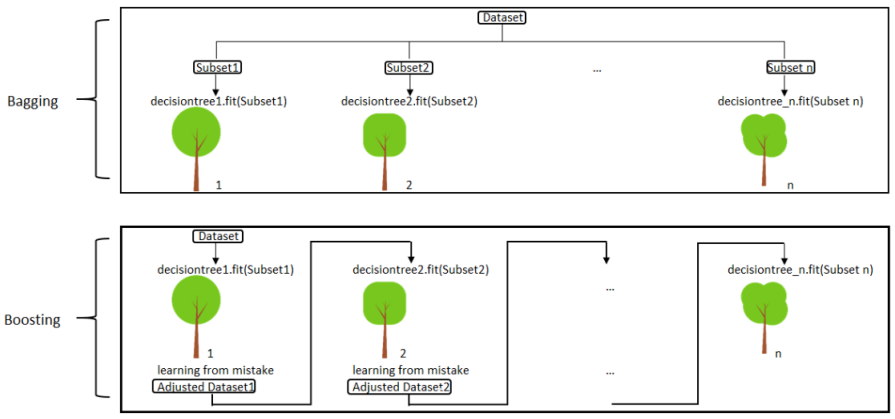


FIGURE 4.2: Illustration of the bagging and boosting procedures

In Figure 4.2, the bagging and boosting procedures are shown. Both ensemble methods use N learners from 1 data set, but while in bagging these are built independently, boosting adds new models that focus on improving what was not captured in the previous models. Both methods generate several training data sets by bootstrapping, but boosting determines weights for the data to tip the scales in favor of the most difficult cases. Also the final decision for both methods is taken by averaging the N learners but in boosting the weights are not equally distributed.
Overall boosting and bagging significantly reduce the variance and provide higher stability. Boosting also reduces the bias but it is more prone to over-fitting.

4.2.5 Random Forest

The random forest method is an ensemble model using bagging as ensemble method and decision trees as individual model.

A decision tree is an algorithm which uses a recursive procedure to divide sample observations of the response variable into sub-groups based on a set of input features. When the belonging sub-space of each observation is determined, the mean of the elements in each sub-group is taken to make predictions. These predictions rely on the homogeneity of the target variables in each subset. These subsets are the result of the splits in the data which aim to optimize the loss function in each group. In case of regression problems, such as the one we are dealing with, the response variable is continuous and the loss function is the mean squared error (MSE). This procedure is often referred to as reducing the tree's impurity.

Why Decision Trees

Decision trees are non-parametric models, henceforth they can model arbitrarily complex nonlinear relations between the dependent and independent variables, whereas standard linear regression, for example, can not handle nonlinearities. Furthermore, non-parametric models make fewer assumptions about the data generating process and do not assume posteriori requirements such as normality of the residuals. In addition, decision trees are capable of handling categorical data such as those provided for this analysis and automatically implement features selection. Decision trees are robust concerning numerical instabilities which enables us to not remove features with little variance and provide estimates for missing data. Decision trees are also easy to interpret which makes them attractive for explanatory purposes. The conditional control statements, which are if-then rules are also quite fast comparing to other machine learning algorithms.
On the other hand, the major drawback of decision trees is the tendency to overfit the training set which can lead to poor out of sample performance. A key task to avoid overfitting is hyperparameters tuning which is discussed in the following.

Classification And Regression Trees

The first tree-based algorithm to include statistical theory is the Classification And Regression Trees (CART) method, introduced by Breiman et al., 1984. The random forest algorithm uses a slightly modified version of CART to construct the singular decision trees that are used in its ensemble.
Suppose that we have a training set with N observations: $(x_i, y_i), i = 1, ..., N$, with $x_i = (x_{i,1}, ..., x_{i,k})$ where k is the number of features in the explanatory variable for the target variable y_i. N in our case is the number of bonds available on a certain day.
The CART algorithm selects, for example, the variable x_1 for every observation i and tries to find recursive cut points(Hastie, Tibshirani, and Friedman, 2009). This procedure is referred to as 'splitting variable' . The CART algorithm finds recursive cut points by splitting $x_j, j = 1, ...k$ in a way that the MSE is minimized. This is called 'splitting criterion' and defines the subset of the predictor x_j that are sent to the right node $\mathcal{N}_\mathcal{R}$ or the left node $\mathcal{N}_\mathcal{L}$. These children nodes together with the cut point threshold s_j are defined as:

$$\mathcal{N}_\mathcal{L}(j, s_j) = \{(x, y) : x_j \leq s_j\} \ \textit{and} \ \mathcal{N}_\mathcal{R}(j, s_j) = \{(x, y) : x_j > s_j\}. \tag{4.30}$$

This CART algorithm grows a tree by selecting among all possible splitting variables and splitting criterion pairs (j, s_j) , the optimal pair $(j*, s_j*)$ such that the MSE at the node is minimized, i.e. the node impurity is minimized. In a regression tree, the

responses in the node $\mathcal{N}$ are modeled using a constant and under MSE loss, this constant is estimated as the mean of the target variables in the node:

$$\hat{c}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} y_i. \tag{4.31}$$

Hence, the CART grows a regression tree by splitting a mother node $\mathcal{N}_\mathcal{M}$ on the splitting variable and a splitting criterion pair $(j*, s_j*)$ such as:

$$(j^*, s_j^*) = argmin_{(j,s_j)} \Big[\sum_{x_j \in \mathcal{N}_\mathcal{L}(j,s_j)} (y_i - \hat{c}(\mathcal{N}_\mathcal{L}))^2 + \sum_{x_j \in \mathcal{N}_\mathcal{R}(j,s_j)} (y_i - \hat{c}(\mathcal{N}_\mathcal{R}))^2 \Big]. \tag{4.32}$$

After the optimal split has been selected, the process is repeated for new children nodes.

Random Forest Regression

Introduced by Breiman, 2001, the random forests method is an ensemble learning algorithm that overcome the tendency of individual trees to overfit the training set. This is carried out through the use of bagging and a slight modification of the CART algorithm and generates a large collection of independent decision trees. Each subset of the data set obtained with bootstrapping is used to train a different CART, which produces an ensemble of different models. The final prediction is then given by an average across all the predictions obtained by the singular CART. This allows for a more robust method than using a single CART in which both variance and overfitting are significantly reduced. The random forest algorithm not only draws a random subset of the training data set but also picks a random subset of features (randomized trees) instead of using all the available ones like in the Cross Sectional model.
In this way, the algorithm is able to build different bucket classes from the predefined (rating, region,sector etc.) buckets. This can be seen in Figure 4.3, where two different decision trees are both capable of producing an estimate for a bond that belongs to the (AAA, Northwest Europe, financial) bucket. The prediction for the bonds belonging to this bucket is calculated in two steps. First, the mean of the z-spreads shifts in the leafs is calculated and these are $\sum_1$ and $\sum_2$ in the figure up to $\sum_m$ where m is the number of decision trees. Then the final prediction is given by the average of all the $\sum$ that contain at least one instance of an AAA rated bond listed in Northwest Europe in the financial sector.
Random forest regression aims at estimating $\hat{f}$ that better approximates the unknown function f such that $f(x) = y$ given the data $\mathcal{D}$.
The random forest predictor consists of a collection of randomized base regression trees $\{ \hat{f}_{tree}(x, \Theta_m, \mathcal{D}\ ), m \leq 1\}$, where $\{\Theta_m\}_{m=1}^M$ are i.i.d. outputs of a randomizing variable Θ that represents the bagging within $\mathcal{D}$ and the random selection of the features vector in $x_1, ...x_k$ to build the randomized tree m (Biau, 2012). We denote the random sample instances drawn given the random object Θ_m from the data set $\mathcal{D}$ as $\hat{\mathcal{D}}(\Theta_m)$. The random trees are aggregated to calculate the mean regression estimate across every CART. We define the leaves of the m-th regression tree as $\mathcal{L}_m(x_1, ..x_k; \Theta_m, \mathcal{D})$, which denotes the leaf constructed in the m-th tree by only using

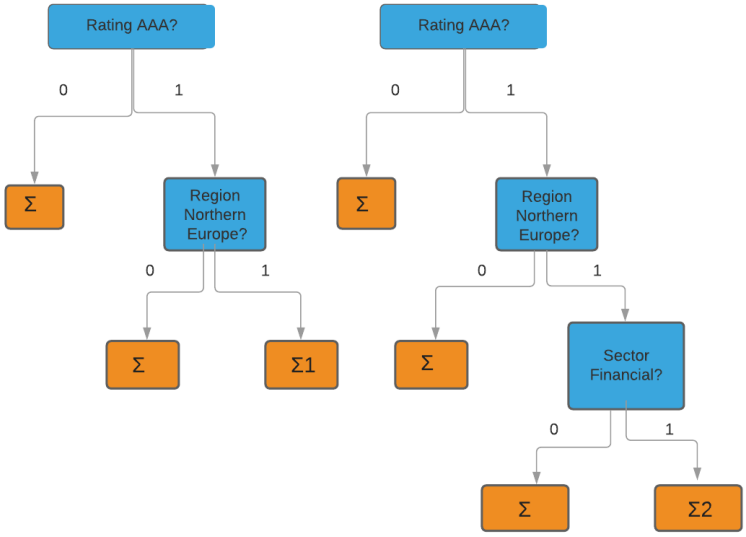


FIGURE 4.3: Two different random decision trees both produce a proxy spread for the bucket: (AAA, Northwest Europe, Financial)

$x_n \in x$ and $\hat{\mathcal{D}}(\Theta_m)$. Then, the prediction of the m-th tree is given by:

$$\hat{f}_{tree}(x, \Theta_m, \mathcal{D}) = \sum_{i \in \hat{\mathcal{D}}(\Theta_m)} \frac{y_i \mathbb{1}\{x_i \in \mathcal{L}_m(x_1, ..x_k; \Theta_m, \mathcal{D})\}}{|x_i \in \mathcal{L}_m(x_1, ..x_k; \Theta_m, \mathcal{D})|}, \tag{4.33}$$

where $\mathcal{L}_m$ is the leaf to which the features of the data point i belong. The numerator contains the sum of the target variables for the x_i that belong to the leaf $\mathcal{L}_m$, while the denominator is the total number of data points in the leaf.
Finally the output of the random forest regression is obtained by taking the mean across all the random trees for any sample observation in $\mathcal{D}$ which belong to any leaf $\mathcal{L}_m$:

$$\hat{f}(x, \Theta, \mathcal{D}) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}_{tree}(x, \Theta_m, \mathcal{D}), \tag{4.34}$$

where M is the number of trees in the forest and x is a predictor vector for the target variable y. This estimator is asymptotically consistent, i.e. it is justified by the law of large numbers for $M \to \infty$, see Biau, 2012.

Random Forest Hyperparameters

In this subsection we present the hyper-parameters that have to be manually tuned in order to optimize the random forest algorithm.

Number of Estimators: It is the number of trees in the forest which we defined as M. A larger number of forests reduces the overfitting issue of singular trees but it also slows down the training process.

Maximum Depth: It is the maximum number of splits allowed in each CART.

Minimum Samples Split: It is the minimum number of data points required to have a splitting node.

Minimum Samples Leaf:It represents the minimum number of bonds required in each leaf. It controls the trade-off between having too many bonds in each leaf, i.e. low accuracy and having a small number of bonds in each leaf, which can lead to overfitting.

Bootstrapping: It allows to choose whether bootstrapping on the training data set has to be performed or not.

The procedure used for the selection of the hyperparameters is explained in a later section.

4.2.6 Gradient Boosting Regression

Gradient boosting, which was introduced by J. H. Friedman (2001), combines two different techniques, one is boosting that we previously described, which is combined with the gradient descent method, also known as the steepest descent method. The idea of averaging the weak learners, that can be CART, resembles what we mentioned for the random forest. However, differently from random forests, the weights of the weak learners are not equally distributed, i.e.,

$$\hat{f}(x, \Theta, \mathcal{D}, \beta\ ) = \sum_{m=1}^{M} \beta_m\ \hat{f}_{tree}(x, \Theta_m, \mathcal{D}\ ). \tag{4.35}$$

Another major difference between boosting and bagging algorithms is that instead of building independent trees, the CART are built sequentially. This allows to minimize the loss function according to the previous trees in the sequence. In case of regression problems the loss function can be the least-squares loss:

$$\mathcal{L}(y, \hat{f}(x)) = (y - \hat{f}(x))^2, \quad -\frac{\partial \mathcal{L}}{\partial \hat{f}} = 2\ (y - \hat{f}(x))\hat{f}'(x) \tag{4.36}$$

Since the stronger learner $\hat{f}(x, \Theta, \mathcal{D}, \beta\ )$ is composed of the M weak learners $\hat{f}_{tree}(x, \Theta_m, \mathcal{D})$, the loss function has to be minimized in the space of the parameters $\{\beta, \Theta\} = \{(\beta_1, \Theta_1), ...(\beta_M, \Theta_M)\}$. This optimization can be performed by using the iterative "greedy" forward stage additive modeling approach, see Friedman, 2001. The idea is to optimize the parameters according to the loss function of the trees that were previous in the sequence:

$$(\beta_m^*, \Theta_m^*) = argmin_{(\beta,\Theta)}\ \mathcal{L}\Big(y, \hat{f}_{m-1}(x) + \beta \hat{f}_{tree}(x, \Theta\ )\Big), \tag{4.37}$$

which results in:

$$\begin{aligned} \hat{f}_m(x) &= \hat{f}_{m-1}(x) + \beta_m \hat{f}_{tree}(x, \Theta_m\ ) \\ &= \sum_{j=1}^{m-1} \beta_j \hat{f}_{tree}(x, \Theta_j\ ) + \beta_m \hat{f}_{tree}(x, \Theta_m\ ) \\ &= \sum_{j=1}^{m} \beta_j \hat{f}_{tree}(x, \Theta_j\ ). \end{aligned} \tag{4.38}$$

Here $\hat{f}_m(x)$ is the ensemble learner built from the previous m trees and $\hat{f}_{tree}(x, \Theta_m)$ is the m-th weak learner or the m-th decision tree.

Gradient Descent Optimization

The second important step in gradient boosting is to combine the previous boosting algorithm with gradient descent iterations. The update for iteration m with the usage of gradient descent is given by:

$$\hat{f}_m(x) = \hat{f}_{m-1}(x) - \beta_m \frac{\partial \mathcal{L}(y, \hat{f}_{m-1}(x))}{\partial \hat{f}_{m-1}(x))}. \tag{4.39}$$

The optimal weight β_m can be obtained by minimizing the loss with respect to the decision tree m:

$$\beta_m = argmin_\beta \, \mathcal{L}\left(y, \hat{f}_{m-1}(x) - \beta_m \frac{\partial \mathcal{L}(y, \hat{f}_{m-1}(x))}{\partial \hat{f}_{m-1}(x))}\right). \tag{4.40}$$

However, from (4.38) and (4.39) we have the following constraint for the m-th decision tree:

$$\hat{f}_{tree}(x, \Theta_m) = \frac{\partial \mathcal{L}(y, \hat{f}_{m-1}(x))}{\partial \hat{f}_{m-1}(x))}. \tag{4.41}$$

This can be obtained by tuning Θ_m in order to select the weak learner $\hat{f}_{tree}(x, \Theta_m)$ that resembles the highest decrease of the loss:

$$\Theta_m = argmin_\Theta \, \mathcal{L}\left(\hat{f}_{tree}(x, \Theta), \frac{\partial \mathcal{L}(y, \hat{f}_{m-1}(x))}{\partial \hat{f}_{m-1}(x))}\right). \tag{4.42}$$

The loss function here is the quadratic loss, as every regression learner can be fitted via quadratic loss and solving it is numerically efficient.

CatBoost Regressor

Among the several gradient boosting algorithms, we selected the CatBoost algorithm, which successfully handles categorical features and outperforms existing publicly available implementations of gradient boosting in terms of quality on a set of popular publicly available data sets (Dorogush, Ershov, and Gulin, 2018).
Categorical features consist of a discrete set of values called categories that can not be used directly in binary decision trees. Therefore, it is common to convert categorical features in numerical values at the preprocessing time. For features with low cardinality, like those considered in this thesis, it is appropriate to use so-called one-hot encoding, which converts the categories into binary variables. The one-hot encoding procedure is described in Section 4.2.8.
When the cardinality of the features is high, CatBoost uses an efficient strategy which reduces overfitting by performing a random permutation of the data set, more information can be found in Dorogush, Ershov, and Gulin, 2018.
CatBoost also effectively implements feature combinations. The feature combinations methodology is one of the main reasons machine learning algorithms outperform linear regression. An example is considering 'AAA' together with Financial

sector and Northwest Europe as a single feature. However, the number of combinations grows exponentially with the number of categorical features in the data set and it is often not possible to consider them all in one algorithm. CatBoost considers combinations in a greedy way, i.e., no combinations are considered for the first split of the decision tree, but in the next split all the combinations and the categorical features in the current tree are combined with all the categorical features in the data set.
In gradient boosting algorithms, each new tree is built to approximate the gradients of the current model. This generally leads to over-fitting, because the gradients used at each step are estimated using the same data points the current model is built on. This causes a shift of the distribution of the estimated gradients in comparison with the true distribution of the gradients which leads to over-fiting.
CatBoost proposes a modification of standard gradient boosting algorithms that does not suffer from this prediction shift. This is done by not using the target variable of the current step in the gradient estimation and it is comprehensively described in Prokhorenkova et al., 2018.

4.2.7 Support Vector Machines

In machine learning, Support Vector Machines (SVM) are nonparametric supervised learning models for both classification and regression problems. SVM were orginally introduced in Cortes and Vapnik, 1995 to solve classification problems. Support Vector Regression (SVR) was introduced later in order to extend the same concept to continuous target variables.
The main idea of SVM is to apply a linear model to the training data but in a higher-dimensional space, such that the model can still capture nonlinearities between dependent and independent variables.
The higher-dimensional model is a hyperplane, SVM builds a set of hyperplanes that are used as decision boundaries between two groups and the best hyperplane is the one that maximizes the distance between the margins, i.e. the distance between the two closest elements of each group. The data points that are closest to the boundaries are called support vectors.
Similarly to the ensemble learning models that we discussed, SVM models are capable to learn complex nonlinear functions and do not rely on any posteriori assumption. Another point of strength of SVM is the possibility to apply regularization to prevent over-fitting.
In Figure 4.4 an illustration of SVM in a very simple case is presented. The goal is to classify the two features, circles and triangles. In the first sub-figure this is simply done in the two-dimensional space and the hyperplane is the line that maximizes the margins. In the second sub-figure it is not possible to separate the two tags with a single line. Therefore, a third dimension is added in the third sub-figure, which is the z-axis such that the two classes can be separated by a hyperplane, i.e. a two-dimensional plane. The final step is transforming the hyperplane back to the original plane by means of a kernel such that a non linear hyperplane is obtained.
This is the idea that underlines the SVM procedure both for classification and regression problems. The latter is explained in more detail in the following.

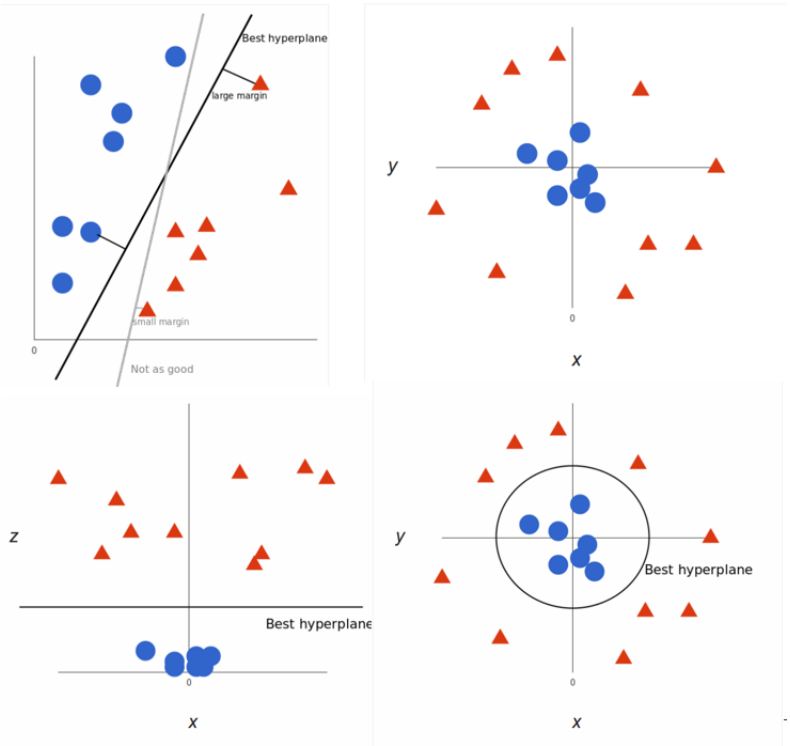


FIGURE 4.4: Explanation of the SVM procedure for a binary classification problem. In the top-left figure, the best hyperplane that separates the two classes is a line. In the top-right figure, a nonlinear hyperplane is required. This is done in the bottom figures by means of a kernel that maps the features space in a higher space in order to separate the two classes and then maps it back to the original features space.

Support Vector Regression (SVR)

For a given data set $\mathcal{D} = \{(x_i, y_i), i = 1, ..., N\}$, where $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,k})$ is the vector containing the k features for the i-th bond and y_i is the z-spread shift of the i-th bond, the goal of SVR is to learn a function $f(x)$ that does not deviate from y by a value greater than a threshold ϵ. This means that SVR uses a symmetrical loss function, which equally penalizes high and low misestimates.
The generalization from SVM to SVR is therefore obtained by introducing this ϵ-insensitive region around the function, called ϵ-tube. The optimization problem transforms from finding the best hyperplane that separates the different classes, to find the ϵ-tube that best approximates the continuous valued function in terms of prediction error and model complexity (Awad and Khanna, 2015).
The simplest case is the linear one:

$$f(x_i) = \beta_0 + x_i'\beta, \quad with \quad \beta \in \chi\,, \beta_0 \in \mathcal{R}, \tag{4.43}$$

where χ is the input features space that contains the vector β. From Hastie, Tibshirani, and Friedman, 2009, the data feature $x_i'\beta$ is called the margin and β_0 denotes the intercept. The problem is a minimization problem of the squared L_2 norm of β:

$$argmin_{(\beta,\beta_0)} \frac{1}{2}||\beta||^2, \tag{4.44}$$

with the aforementioned constraint:

$$|y_i - (\beta_0 + x_i'\beta|) \leq \epsilon \quad \forall i. \tag{4.45}$$

Often, there is no such function $f(x)$ capable of satisfying this requirement for every i, this is also called a hard-margin.
In order to overcome this impediment, the soft-margin concept has been introduced in Smola and Schölkopf, 2004. This is done by the introduction of slack variables ξ_i and ξ_i^* which allow for larger regression residuals and the problem formulation can be stated as:

$$argmin_{(\beta,\beta_0,\xi_i,\xi_i^*)} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*), \tag{4.46}$$

which is now subject to the 'softer' constraints:

$$\begin{aligned} y_i - (\beta_0 + x_i'\beta|) &\leq \epsilon + \xi_i \quad \forall i, \\ (\beta_0 + x_i'\beta|) - y_i &\leq \epsilon + \xi_i^* \quad \forall i, \\ \xi_i\xi_i^* &\leq 0 \quad \forall i. \end{aligned} \tag{4.47}$$

Here the constant $C > 0$ is the cost parameter and determines the trade-off between the flatness of f and the amount up to which deviations larger than ϵ are tolerated. The soft margin concept is visually displayed in Figure 4.5.
Similarly to what we showed for classification problems in SVM, sometimes a linear

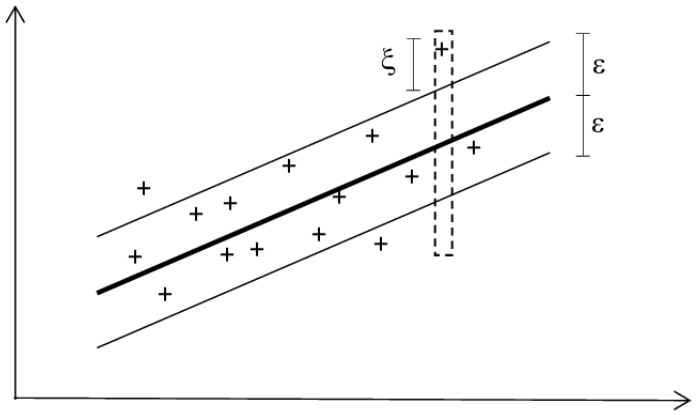


FIGURE 4.5: Illustration of the soft margin concept. The figure shows the ϵ-tube and one of the slack variables ξ

model cannot properly handle the regression problem. In this case SVR fits a curve instead of a line by using a nonlinear kernel $\phi(x)$ that maps the dependent variable into a higher-dimensional space.
Following Platt et al., 1999, we consider the Lagrangian dual formulation of Equation (4.46) because it is simpler to optimize. To do this, the Lagrangian multipliers α_i and α_i^* are introduced for every i, and this leads to the minimization quadratic

problem:

$$argmax_{(\alpha_i,\alpha_i^*)} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^*-\alpha_i)((\alpha_j^*-\alpha_j)H(x_i,x_j)+\epsilon\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)-\sum_{i=1}^{N}y_i(\alpha_i^*-\alpha_i). \tag{4.48}$$

Here $H(x_i,x_j)=\phi(x_i)'\,\phi(x_j)$, and the problem is defined by the contraints:

$$\begin{aligned}\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)&=0,\\ 0\leq\alpha_i,\alpha_i^*&\leq C \quad \forall i,\\ \alpha_i\alpha_i^*&=0 \quad \forall i.\end{aligned} \tag{4.49}$$

In Smola and Schölkopf, 2004, it is shown that β can be written as:

$$\beta=\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)\phi(x_i). \tag{4.50}$$

In addition to the constraints in (4.49), a set of Karush-Kuhn-Tucker (KKT) constraints is required and these are given by:

$$\begin{aligned}\alpha_i(\epsilon+\xi_i-y_i+f(x_i))&=0 \quad \forall i,\\ \alpha_i^*(\epsilon+\xi_i^*-y_i-f(x_i))&=0 \quad \forall i,\\ \xi_i(C-\alpha_i)&=0 \quad \forall i,\\ \xi_i^*(C-\alpha_i^*)&=0 \quad \forall i.\end{aligned} \tag{4.51}$$

The function to be trained is now:

$$f(x)=\beta_0+\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)H(x_i,x). \tag{4.52}$$

If the model is just the linear model, $H(x,x)$ is the dot product xx'. The KKT constraints imply that when $\alpha_i=C$, the observations with these multipliers are located outside the ϵ-tube . The dominance of the multipliers is controlled by the parameter C which performs regularization. Observations inside the ϵ-tube do not require regularization and therefore α_i,α_i^* vanishes for $|y_i-(\beta_0+x_i'\beta|)\leq\epsilon$. Observations with a vanishing multiplier are called Support Vectors.

SVR Hyperparameters

Here we present the three main hyperparameters that are necessary to use SVR:

ϵ: It is the threshold below which no penalty is given to the error. It is the radius of what is called ϵ-tube. A large ϵ means that the regression would be less accurate as large errors are not penalized, i.e. under-fit. Conversely, is ϵ is too small, the number of support vectors increases and this leads to over-fit.

C: the regularization parameter C allows to tune the hardness or softness of the margin transition. When C is low, margin samples are less penalized . A larger C improves the training error but it carries the risk of losing generalization and

leads to over-fitting.

Kernel: The choice of the kernel $H(x, x)$ can be considered as an hyperparameter since it has to be decided from the data scientist. Here we present four popular choices according to Hastie, Tibshirani, and Friedman, 2009:

- Linear: $H(x, x) = x'x$
- k-th Degree Polynomial: $H(x, x) = (\theta + x'x)^k$
- Gaussian Radial Basis Function: $H(x, x) = exp(-\gamma||x - x'||^2)$
- Sigmoid $H(x, x) = tanh(\kappa_1 x'x + \kappa_2)$

The mapping $\phi(x)$ does not require to be explicitly computed. This is called the 'kernel trick'. In fact, the kernel function $H(x, x)$ inner product is applied in the transformed space (Hastie, Tibshirani, and Friedman, 2009).
Also the parameters of the different kernels are hyperparameters. These depend on the kernel choice and therefore are not singularly discussed.

4.2.8 Models Framework and Data Processing

In this subsection, the general framework common to all the algorithms is presented. This includes the modified cross-sectional model structure, the procedure that transforms the categorical features into numerical values together with the standardization of the dependent variable, i.e. the z-spread shift.
Afterwards we present the criteria for filtering the outliers from the fitting of the model that significantly improves the accuracy.

Improved Cross-Sectional Model

As already mentioned in Section 3.1, the cross-sectional model proposed for CDSs in Chourdakis et al., 2013 has been improved by the addiction of three extra categorical features. These are *Tenor*, *Currency* and *Market Indicator*, the latter is a binary variable that discriminates between developed and emerging markets. Together with the existing categorical variables *Rating*, *Region*, *Sector* and *Seniority*, the improved cross-sectional model accounts for 7 categorical features. The inclusion of all these additional factors improved the accuracy of the proxy together with the others metrics such as the correlation, R^2 coefficient, AIC and BIC scores. In order to avoid the possible multicollinearity problem presented in Chourdakis et al., 2013, we define a benchmark intercept. This means that we remove one categorical level from each feature. These categorical levels form a benchmark bucket which is the intercept of the regression model. A common choice in statistical analysis is to pick the levels with the highest number of observations as benchmark. Our benchmark bucket is formed by the categorical levels: [*Rating*: A, *Region*: Northwest Europe, *Sector*: Financial, *Seniority*: Senior Unsecured Debt (SNRFOR), *Tenor*: 1 year, *Currency*: Euro, *Market Indicator*: Developed].
The improved cross-sectional method to proxy bond z-spreads can be described by

the linear relationship:

$$\Delta z_i = \beta_{benchmark} + \sum_{a=1}^{N_{Rat}-1} \beta_a^{Rat} I_a^{Rat}(i) + \sum_{b=1}^{N_{Reg}-1} \beta_b^{Reg} I_b^{Reg}(i) + \sum_{c=1}^{N_{Sec}-1} \beta_c^{Sec} I_c^{Sec}(i) + \sum_{d=1}^{N_{Sen}-1} \beta_d^{Sen} I_d^{Sen}(i) + \sum_{e=1}^{N_{Ten}-1} \beta_e^{Ten} I_e^{Ten}(i) + \sum_{f=1}^{N_{Cur}-1} \beta_f^{Cur} I_f^{Cur}(i) + \beta^{Eme} I^{Eme}(i) + \epsilon_i. \tag{4.53}$$

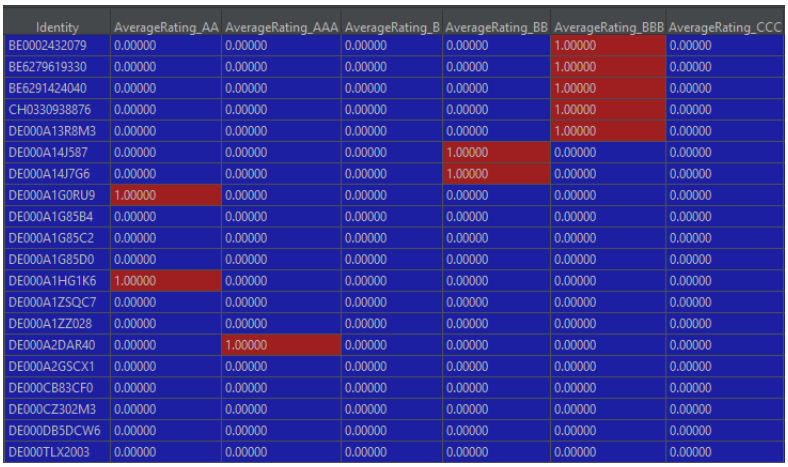
The interpretation of the beta coefficients here is slightly different from the cross-secctional model. In fact, since we are using a benchmark bucket, the beta coefficients represent the changes in the z-spread shift when moving from the benchmark categorical level to the selected level.
The summation on each category contains one instance less than the cross-sectional method because that instance is included in the benchmark bucket. This is also why the last categorical feature 'market indicator', which is a binary variable, has only one term in the model.
For what concerns the machine learning algorithms that we presented, the general framework is the same. Even though the machine learning algorithms do not use a linear regression model, the same benchmark bucket and categorical features are used.

One-Hot Encoding

Many machine learning algorithms cannot operate on categorical data directly. They require all input variables and output variables to be numerical values. Since the number of categorical levels we are dealing with is not particularly high (34 categorical levels after the exclusion of the benchmark bucket), it is reasonable to use one-hot encoding. One-hot encoding and label encoding are the two most popular ways to work with categorical data. However, in label encoding, to each label is assigned a number, e.g. 'Developed market' is marked with 1 and 'Emerging market' with 2. In this way the two categories have a natural ordered relationship which wrongly influences the model construction so it is not desirable.
With one-hot encoding, instead, categories are binary represented as dummy variables. This means that instead of having 7 columns in the X matrix of the categorical features, we have 34 columns, one for each categorical level and each entry is filled with a 1 if the categorical level of the bond corresponds with the respective column, and it is zero otherwise.
For example, if a bond is categorized as :['BBB', 'South-west Europe', 'Consumer', 'Junior Subordinated','2 years to maturity', 'Euro', 'Emerging'], its features matrix X is filled with ones in these aforementioned 7 columns and zeros in the other 27 columns. An example of a one-hot encoded matrix is presented in figure 4.6. Only the rating part is shown, in order to keep the figure readable. Here the identity column refers to the bond ISIN and so it uniquely represents every bond. The other columns are those that encode the rating. The rows in which all entries are zero are those bonds belonging to the benchmark bucket, i.e. 'A' rated bonds.

MinMax Scaler

Many machine learning algorithms perform better when numerical input variables are scaled to a standard range. For what concerns the explanatory variables X, since we used one-hot encoding, no scaling is required as the 'dummy' variables can take



Identity	AverageRating_AA	AverageRating_AAA	AverageRating_B	AverageRating_BB	AverageRating_BBB	AverageRating_CCC
BE0002432079	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
BE6279619330	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
BE6291424040	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
CH0330938876	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
DE000A13R8M3	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000
DE000A14J587	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000
DE000A14J7G6	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000
DE000A1G0RU9	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1G85B4	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1G85C2	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1G85D0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1HG1K6	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1ZSQC7	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A1ZZ028	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000A2DAR40	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000
DE000A2GSCX1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000CB83CF0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000CZ302M3	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000DB5DCW6	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
DE000TLX2003	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

FIGURE 4.6: One-hot encoding matrix for rating columns. Each row corresponds to a different bond.

values $x \in \{0, 1\}$.
We are interested, instead, in standardizing the target variable y as z-spread shifts are clustered in a really small range around zero and this is problematic for both SVR and RF.
The MinMaxScaler is probably the most famous scaling algorithm, and it allows SVR and RF to work properly. It is described by the following formula for a target variable y_i referred to the i-th bond:

$$y_i^{scaled} = \frac{y_i - min(y)}{max(y) - min(y)}. \tag{4.54}$$

The MinMaxScaler shrinks the range in the interval: $y_i^{scaling} \in [0, 1]$. This scaler works better than others specially when the standard deviation is small, as in our case.
The scaled z-spread shifts are then used for the model fit and prediction and afterwards an inverse transformation is applied to measure the performance on the original scale. For what concerns Linear Regression and CatBoost, the scaling is not necessary and does not provide any improvement/deterioration. Anyway, MinMax scaling is applied to all the algorithms to maintain the same common framework.
However, this method is sensitive to outliers and this issue is tackled in the following section.

Outliers Filtering

Linear regression and the machine learning algorithms that are used in this thesis are dramatically sensitive to outliers (Géron, 2019). In Chapter 5 we show that removing outliers from the fitting of the model significantly improves the quality of the proxied z-spreads shifts in term of accuracy, correlation and R^2 coefficient.
It is important to underline that outliers are only removed from the fitting and not from the prediction of the model, i.e. the bond proxy predicts z-spread shifts also for the outliers and these are included in the performance measurements. The reason why this procedure is justified is that we want the proxy to be as accurate as possible and modelling jumps is not the objective.
The objective of the proxy is to model systematic risk and therefore to capture the

general behaviour of the bond in examination. Outliers are considered as idiosyncratic risk which is not in scope of the proxy model. The inclusion of outliers in the fitting of the model only provides noise that reduces the accuracy of the proxy. The analysis has been carried out with and without outliers filtering in the fitting of the model and the results are shown in Chapter 5.
We considered as outliers and removed from the fitting of our model those observation that fall outside 3 standard deviations from the mean of the z-spread shift.
The major drawback derived from this approach is that our proxy Value at Risk is generally underestimated. In Chapter 5, this problem is discussed in more detail and a possible solution is provided.

4.2.9 Hyperparameters Selection Procedure

As mentioned in the previous subsections, machine learning algorithms require hyperparameters tuning. In contrast with model parameters, which are learned by the algorithm, hyperparameters are set a-priori in order to configure the model and control the learning process.
The configuration of hyperparameters is a challenging task within machine learning applications. In fact, the goal is to find an optimal combination among the hyperparameters set, such that the loss function is minimized. The resulting performance of the model is strongly influenced by the hyperparameters choice. The objective is to generalize the model performance in order to obtain an out of sample performance which is comparable with the in sample or training set performance.

Grid Search vs Random Search

The traditional technique used for hyperparameters tuning is the Grid Search strategy. Grid Search sequentially inspects all the possible combination defined by a manually set up hyperparameters set.
Grid Search performance is assessed on the training set. In our case, since K-Fold Cross-Validation is applied, the average loss across the validation samples is used to rank each hyperparameter set. This quantity is often referred as generalization error and in this case is the average RMSE is computed across the K folds. However, Grid Search suffers from the known problem called curse of dimensionality (Hastie, Tibshirani, and Friedman, 2009). This is a significant drawback as the number of hyperparameter combinations grows exponentially. This makes the process really cumbersome even if the number of parameters to tune is relatively low.
In order to overcome this issue, the Random Search strategy has been proposed in Bergstra and Bengio, 2012. In this paper, Bergstra and Bengio show empirically and theoretically that random search is more efficient for parameter optimization than grid search. This is because only few hyperparameters actually affect the performance for a given data set and finding the optimal values of these hyperparameters has more impact than obtaining the optimal combination for all hyperparameters.
Within Random Search, the combinations of hyperparameters are randomly selected. The number of random draws for each hyperparameter value is previously set by the user.
In Figure 4.7, we show a visual demonstration that Random Search searches over a larger space of hyperparameters combinations given the same computational budget.

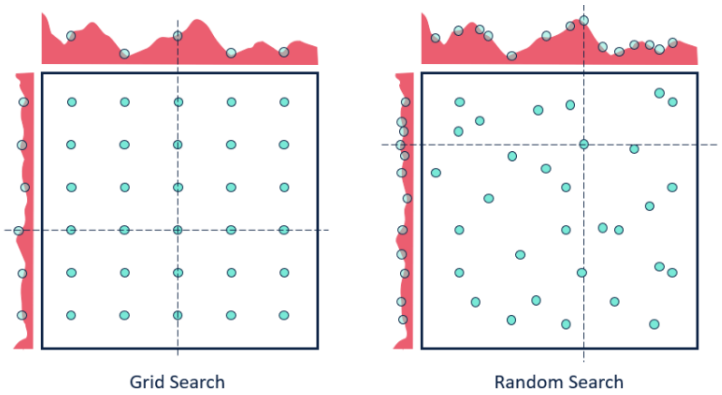


FIGURE 4.7: Illustration of the Grid Search and Random Search strategies for 2 parameters.

Two Steps Selection Strategy

Since the model is calibrated daily, running the hyperparameters search for every day would require excessive time and would lose generality. Therefore, we applied the following two steps procedure.
First, 5 days across the entire time span (2 years) have been homogeneously selected and we ran the randomized grid search on those days. The grid has been empirically adapted over a range of several possible values for each hyperparameter. Every hyperparameter value has been drawn 100 times in the Random Search process.
Secondly, the resulting 5 best hyperparameters combinations from the Random Search strategy have been implemented over the whole time span. Finally, the combination that best performed over the 2 years under examination has been used as final hyperparameter set. More details about the analysed parameters and the selected ones are given in Chapter 5.

Chapter 5

Results and Models Optimization

This chapter presents the framework used to build the models together with the results from the Shit Types Assessment and the Credit Spreads Proxy. Similarly as we did in Chapter 4, the first part of the chapter concerns the Shift Type Assessment, while the second part is dedicated to the Credit Spreads Proxy.

5.1 Shift Types Assessment

In this section we visualize the different shift types and assess the criteria to perform the parameter turning for arcsinh and displaced relative shifts. Afterwards, we present a comparison of the different shift type performance results.

5.1.1 Parameters Tuning

As we explained in the Shift Type Assessment Methodology, Section 4.1, the mixed type shifts that we proposed, i.e. arsinh shifts and displaced relative shifts, both need one parameter to be tuned.
The main assumption of Historical VaR is that price changes are i.i.d and therefore we tune the shift types parameters in order to better satisfy this assumption. In particular, the risk factor shift should not depend on the risk factor level.
This means that the "correct" type of shifts should be homoschedastic with respect to the level, i.e. shifts should be homogeneously distributed along the z-spread level line. This is not the case for relative shifts as we can easily see from Figure 5.1. Note that the y-axis has been cut in order to show the distribution of the points but shifts explode to infinity for levels that approach 0. The behaviour of relative shifts is clearly undesirable for this type of product as it generates completely unrealistic scenarios.
Definitely more in line with the HVaR assumption of i.i.d. is the behaviour of the absolute shifts as can be seen from Figure 5.2.
For the previous figures and for all tests performed in this section, we used the 438 bonds for which we have complete history out of the total 8432 bonds.
A standard procedure in shift type assessments is to tune shift types parameters based on Unconditional Coverage or other backtesting procedures. However, for the period under consideration which runs from August 2017 to August 2019, HVaR generally underestimates the risk. This means that the number of observed exceedances, i.e., the number of times an observation is more extreme than predicted by our VaR model, is higher than expected.
This underestimation is due to the nature of historical VaR and it can be summarized as "the past does not predict the future", in fact HVaR's only assumption is that future scenarios are drawn from the past, in this case, 260 observations that are supposed to be identically independently distributed. This is clearly a strong

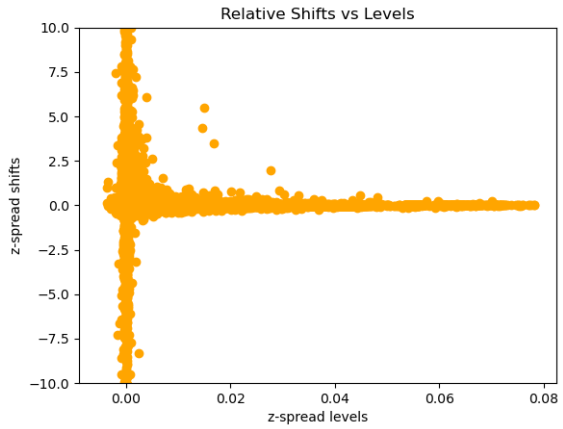


FIGURE 5.1: Scatter plot of relative changes (on the y axis) against levels of the z-spread (on the x axis)

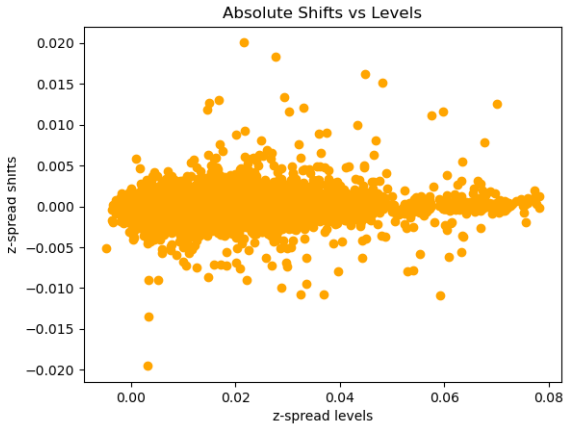


FIGURE 5.2: Scatter plot of absolute changes (on the y axis) against levels of the z-spread (on the x axis)

assumption and, in the period as the one under study, in which the volatility of z-spreads and of z-spread shifts sharply increase, it leads to an underestimation of the risk.
This can be understood from Figure 5.3, in which we plot the standard deviation of the z-spread and z-spread daily changes (or absolute shifts) with a moving window of 260 days, which is exactly where for HVaR we expect constant values.
This volatility increase is the main driver for the HVaR underestimation and also the reason why the Unconditional Coverage test should not be used in order to tune the parameters of arcsinh and displaced relative shifts.
The reason is that Unconditional Coverage (and other backtesting procedures) reward the unrealistic behaviour of relative shifts because it allows for a smaller number of exceedances compared to the absolute shifts. In this way it "compensates" the higher number of exceedances observed. Keep in mind that the parameter choice for the arcsinh and displaced relative shifts make the shift closer to absolute or relative shifts. In line with the aforementioned findings, we decided to perform a linear regression on the modulus of the shifts with respect to the levels and use the regression

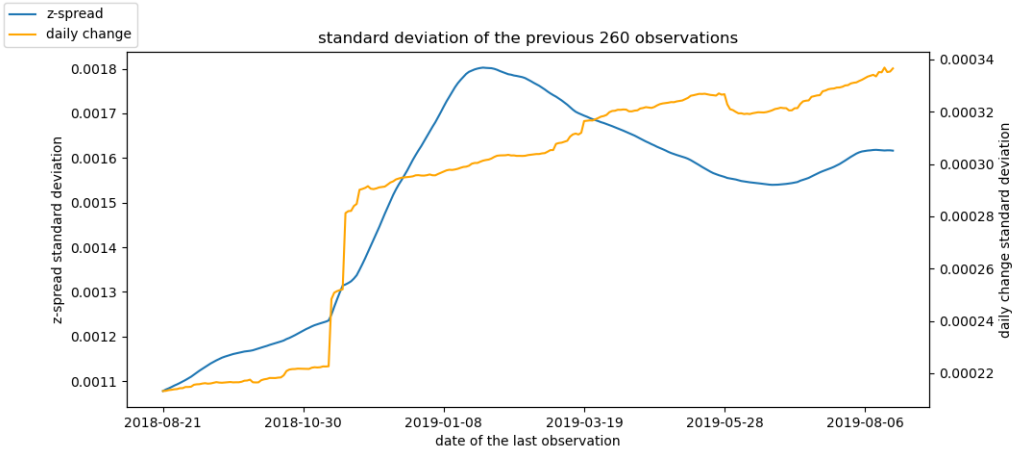


FIGURE 5.3: In blue the standard deviation of z-spreads level over the last 260 days from the data point on the x axis, z-spread level on the left vertical columns. In orange, the daily change standard deviation, which refers to the right vertical axis.

coefficient of the z-spread level as the criterion to optimally select the parameters for the arcsinh and displaced relative changes. This means that we choose the parameters for the shifts such that we obtain a zero regression coefficient for the dependent variable which is the z-spread level.
In fact, the general problem with absolute shifts is that they tend to systematically increase in modulus with the increase of the levels. This can be observed from Figure 5.4 where the regression line has a slightly positive coefficient. The choice of

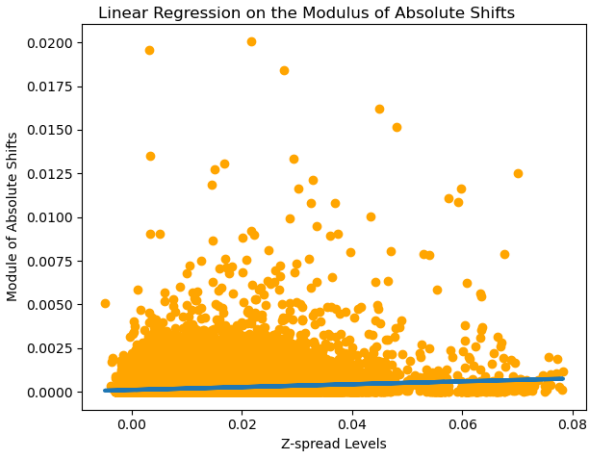


FIGURE 5.4: Fitted regression line on the modulus of absolute shifts with respect to the z-spread levels

using the modulus of the shifts is given by the fact that, since shifts have zero mean, having negative and positive values results for both absolute and relative shifts in a zero coefficient regression line. With the modulus function we can easily show why relative shifts are definitely not recommended (Figure 5.5) and we have a measure to assess the parameters for the mixed shift types.
The measure we choose is to select the parameter for the shift type such that the

FIGURE 5.5: Fitted regression line on the modulus of relative shifts with respect to the z-spread levels, note that the y axis is cut, but shifts explode to infinity.

regression line has a zero coefficient, which means, since shifts have mean equal to zero, that on average we removed linear dependence from shifts and levels. In formulas, we define the dependent variable:

$$\begin{aligned} y_{i,\,t} &= \Delta x_{i,\,t} \\ &= f\big(x_{i,\,t-1}\,, x_{i,\,t},\ \gamma\big). \end{aligned} \tag{5.1}$$

Where $y_{i,t}$ can be any shift type (in this case is either displaced relative or arcsinh) for the bond i at time t and γ is the parameter we want to tune. Then, the dependent variable is just the z-spread level: $X_{i,\,t} = x_{i,\,t}$. Grouping together the 438 bonds with full history and 2 years of data (521 days) we obtain the Ordinary Least Squares regression line :

$$\begin{aligned} &y = \beta_0 + \beta_1 X, \\ &y = \{y_{(0,\,0)}, y_{(0,\,1)}, ..., y_{(1,\,0)}, ..., y_{(438,\,521)}\}, \\ &X = \{X_{(0,\,0)}, X_{(0,\,1)}, ..., X_{(1,\,0)}, ..., X_{(438,\,521)}\}. \end{aligned} \tag{5.2}$$

From here we tune the parameter γ in Equation (5.1), for each shift type such that the linear dependence coefficient β_1 goes to zero. This procedure is in line with the HVaR model and it is shown in Figure 5.6.
It is important to note from the figure that the β_1 coefficient goes to zero for large values of the parameter, but this is just due to scaling reasons. After a certain threshold we are actually using absolute shifts, just scaled by a factor $1/\gamma$. For displaced relative shifts we have:

$$\lim_{\gamma\to\infty} \Delta x_{t+1}^{disp} = \frac{x_{t+1} - x_t}{(x_t + \gamma)} \sim \frac{x_{t+1} - x_t}{\gamma} = \frac{\Delta x_{t+1}}{\gamma}. \tag{5.3}$$

For the arcsinh shifts, by keeping in mind that the arcsin hyperbolic function behaves like a linear function when the argument is close to zero, we obtain:

$$\lim_{\gamma\to\infty} \Delta x_{t+1}^{arcsinh} = arcsinh(\frac{x_{t+1}}{\gamma}) - arcsinh(\frac{x_t}{\gamma}) \sim \frac{x_{t+1}}{\gamma} - \frac{x_t}{\gamma} = \frac{\Delta x_{t+1}}{\gamma}. \tag{5.4}$$

The β_1 coefficient approaches zero at the limit because the magnitude of the shifts is inversely proportional to the magnitude of the parameter, while the z-spread levels remain unchanged.

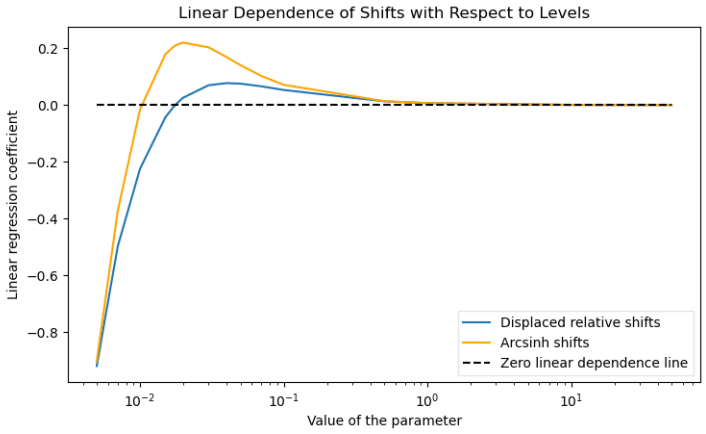


FIGURE 5.6: On the y axis the value of the linear regression coefficient β_1 and on the x axis the shift parameter γ

Following the aforementioned method we set the parameters for the displaced relative and arcsinh shifts respectively to $a = 0.0147845$ and $b = 0.0108211$, we use two different variable names such that it is easier to recall each of them.
In figures 5.7 and 5.8 the resulting distribution of the parameterized shifts versus levels is shown. Again, we can notice a higher dispersion of shifts values for lower levels, but this is due to a much larger number of observations for lower levels.

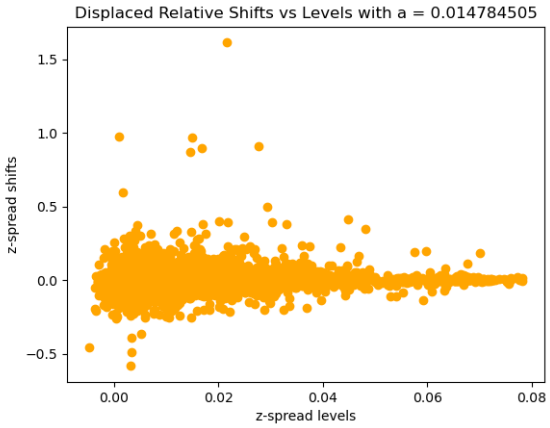


FIGURE 5.7: Scatter plot of displaced relative changes (on the y axis) against levels of the z-spread (on the x axis)

5.1.2 Testing Results

In this section we present the testing results among the three shift types that are considered for our analysis. The main comparison about shift types is going to

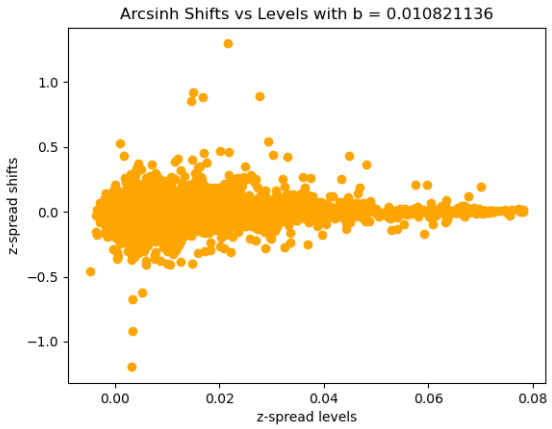


FIGURE 5.8: Scatter plot of arcsin hyperbolic changes (on the y axis) against levels of the z-spread (on the x axis)

be presented with the proxy, however, the results presented in this section are the standard tests for banks.
We perform the statistical test in this section across the 438 bonds with full data history and we report the result of these tests following the ING, 2020 internal documentation. In order to summarize the information collected among these tests we report the percentage of non rejected tests with a significance level of 5% Even though we mentioned that we cannot rely on the Unconditional Coverage test due to the significant increase of standard deviation across z-spreads levels in the 2 years period under examination, we report it for completeness together with the Statistical Independence test in the backtesting results.

Backtesting Results			
	Absolute Shifts	Displaced Relative Shifts	Arcsinh Shifts
Expected # of Exceedances	5.14	5.14	5.14
Observed # of Exceedances	8.89	8.26	8.74
Unconditional Coverage	72.8 %	76.7 %	74.6 %
Statistical Independence	74.8 %	76.2 %	74.1 %

TABLE 5.1: Results of backtesting tests. The percentages in the table are percentages of non-rejected test with a significance level of 5%.

The expected and observed number of excedancees is averaged across the 438 bond with full time history. For the reasons explained in the previous section the number of observed excedancees is always significantly higher than expected. The results of these tests are comparable as expected. Displaced relative shifts slightly

outperform the other two shift types.
For what concerns the econometric tests, we adopt the same strategy that we used for backtesting tests and we present the percentage of non rejected tests across the set of 438 bonds with full data.

Econometric Tests Results			
	Absolute Shifts	Displaced Relative Shifts	Arcsinh Shifts
White Test	16.5 %	24.5 %	19.2 %
BP test	19.1 %	27.3 %	22.4 %
ADF and PP test on levels	95.1 %	87.8 %	95.1 %
ADF and PP test on shifts	0 %	0 %	0 %

TABLE 5.2: Results of econometric tests. The percentages in the table are percentages of non-rejected test with a significance level of 5%.

The homoschedasticity assumption tested by the White test and the Breush-Pagan test are violated in general. This is commonly observed in practice given that the real-world data series are generally more complicated than a simple random walk. Nevertheless, the mixed shifts are generally less frequently rejected in these homoschedasticity tests, as we expect from our parameters tuning.
From the Augmented Dickey-fuller (ADF) and Phillips Perron (PP) test results, we find it reasonable to assume that the z-spread level has a unit root in place, which means that it has a random walk type behaviour, i.e. it is non-stationary. The performed unit root test on levels is the ADF in case we use absolute or arcsinh shifts and PP in case of displaced relative. Overall, it mainly confirms that the risk factor follows the theoretically correct random walk dynamics. This means that it is correct to simulate future risk factor values by applying historical shifts on top of the current risk factor level. The ADF and PP test results on the shifts are the same across all the shift types and it always rejects the null hypothesis that there is a unit root, i.e., shifts are stationary. These unit root tests do not help to discriminate between different shifts but confirm that the modelling procedure is correct.
The presented results are quite similar across the different shift types and this allows us to consider all of them in the z-spreads proxy implementation and have a second comparison.

5.2 Credit Spreads Proxy

In this section we present the main results of this thesis. The chosen combinations of hyperparameters for the machine learning algorithms are first presented. Afterwards, we present the performance of the bond credit spreads proxy with and without outliers in the fitting. Finally, a comparison of the true VaR, i.e. the VaR obtained with the real data, against the VaR given by the proxied data is provided.

5.2.1 Hyperparameter Optimization

As mentioned in Section 4.2.9, finding the optimal hyperparameters combination is essential in machine learning algorithms. Therefore following the two-steps procedure explained in Section 4.2.9, we first selected 5 days and tune the hyperparameters on these days by random search.
In order to perform the random search, we define a finite set of reasonable values for each hyperparameter. For a continous hyperparameter θ, we construct the grid as follows: $\theta \in [a, b]$ where $|a|, |b| < \infty$ with step size s. Through the step size we determine the number of equally spaced elements (in linear or logarithmic scale) in the grid.
In the following subsections we specify the grids for the machine learning algorithms and we present the result of the two-steps procedure. As previously mentioned, the first step is the application of the random search across 5 days homogeneously selected across the two years in examination: (14-11-2017, 03-04-2018, 21-08-2018, 08-01-2019, 28-05-2019). Afterwards, the models are ranked based on the performance on the validation set and the best hyperparameters configuration is picked for each day.
The second step involves testing the 5 best hyperparameter combinations across the whole time-frame and selecting the one that provides the best performance.

Optimizing Random Forest

For what concerns the optimization of the random forest algorithm the following hyperparameters have to be tuned: number of trees, maximum depth, minimum samples split, minimum samples leafs, bootstrapping. The meaning and implications of these hyperparameters are explained in Section 4.2.5.
The Table 5.3 below presents the grid of reasonable hyperparameters for the random forest. The choice of the extremes for the grid has been tested by earlier experiments.

Hyperparameter	Grid	Step Size
# of Trees	[200, 2000]	100
Max Depth	[10, 110]	10
Min Samples Split	[2, 10]	1
Min Samples Leafs	[1, 4	1
Bootstrap	[True, False]	-

TABLE 5.3: Grid and step size for the random forest hyperparameters

The chosen hyperparameter combinations in terms of RMSE are presented in the Table 5.4 below for the 5 days under examination. These 5 combinations are then tested across the covered 2 years period.

Date	# of Trees	Max Depth	Min Samples Split	Min Samples Leafs	Bootstrap
14-11-2017	400	60	10	2	True
03-04-2018	200	20	10	4	False
21-08-2018	1200	10	5	4	False
08-01-2019	600	110	10	2	True
28-05-2019	1800	50	5	4	True

TABLE 5.4: Selected hyperparameters for the 5 tested days

Across these 5 combinations the first one slightly outperforms the others when tested over the full 2 years period. It is important to mention that the obtained accuracy is comparable across these combinations and therefore we reckon it to be sufficient to have the preliminary study across these 5 days only.
For clarity, the selected hyperparameter combinations are presented below.

Selected Hyperparameters for Random Forest

Number of Trees: 400.
Maximum Depth: 60.
Minimum Samples Split: 10.
Min Samples Leafs: 2.
Bootstrap: True.

Optimizing CatBoost

The CatBoost regressor offers a flexible tool that automatically selects the best hyperparameters set composed of the number of trees, learning rate and tree depth. This is done automatically every day, therefore no single combination of hyperparameters is required.
Anyway, the two-steps procedure used for the other machine learning algorithms has been implemented for CatBoost as well. However, the result of the aforementioned automatic feature provides a better performance than the two-steps procedure. The time required to implement these automatic features does not significantly increase comparing to the usage of predetermined hyperparameters, therefore it has been used across the full 2 years period.

Optimizing Support Vector Regression

In SVR we implemented the two-steps procedure for the selection of the best hyperparameter combinations. The hyperparameters that we need to tune are: ϵ, C and the kernel type. Details on these hyperparameters are given in Section 4.2.7.
The Table 5.5 presents the grid of reasonable hyperparameters for the SVR algorithm. As for the random forest, the choice of the extremes for the grid has been

tested by earlier experiments.

Hyperparameter	Grid	Step Size
ϵ	$[0.0001, 10]$	log_{10} scale
C	$[0.001, 1000]$	log_{10} scale
Kernel	Linear, Polynomial, RBF, Sigmoid	-

TABLE 5.5: Grid and step size for the SVR hyperparameters

In Table 5.6 we show the best hyperparameters combination for the 5 days under examination. As previously, these combinations are tested over the full 2 years period. What we mentioned only holds for similar periods, as two years is a relatively short amount of time and it is characterized by a specific behaviour.

Date	ϵ	C	Kernel
14-11-2017	0.1	10	RBF
03-04-2018	0.01	1	RBF
21-08-2018	0.01	1	RBF
08-01-2019	0.1	100	RBF
28-05-2019	0.001	0.1	Polynomial

TABLE 5.6: Selected hyperparameters for the 5 tested days

Most of the combinations are similar however, the Gaussian Radial Basis Function (RBF) seems to outperform the other kernels in the majority of cases. The second (and third) combination are the ones that outperform the others when tested across the 2 years period. The final choice of hyperparameters is shown below.

Selected Hyperparameters for SVR

ϵ: 0.01.
C: 1.
Kernel: Gaussian Radial Basis Function.

5.2.2 Proxy Performance

In order to measure the performance of our proxy for bond credit spreads we used the various metrics presented in Section 4.2.1.
We do not report the results obtained with AIC and BIC as these have been used to check that adding new explanatory variables, e.g. tenor, currency and market indicator, always decreased these coefficients.
The results obtained across the three different shift types are significant, absolute and arcsinh shifts outperform displaced relative shifts. In this section we report only the results for absolute shifts across different algorithms. The results for the others shift types can be found in Appendix A.
The most significant error metrics are reported in this section and these are: RMSE, the R^2 coefficient and the correlation, respectively measured on the training and test sets. For what concerns the standard deviation, of the proxy results, it systematically increase by the addition of new explanatory variables. With the 7 used explanatory variables, our proxy obtained a standard deviation measure σ that is approximately half of the one obtained with the real data. However, the benchmark model, i.e. the intersectional method with 4 regressors obtained a standard deviation that is approximately one fourth of the true one. This result is in favour of a more realistic proxy given by the addition of new information.
The tables in this section present the results obtained by the benchmark model, i.e. the original cross-sectional model, compared with the 4 proposed algorithms: the improved-cross sectional model, which uses linear regression (LR), the random forest algorithm (RF), the CatBoost algorithm (CAT) and the support vector regression algorithm (SVR). See Table 5.7.

Performance of different algorithms on absolute shifts						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	4.6e-4	4.6e-4	0.041	0.013	21.2%	16.7%
LR	4.3e-4	4.4e-4	0.108	0.082	31.4 %	28.9%
RF	4.0e-4	4.1e-4	0.272	0.117	56.6 %	33.8%
SVR	4.0e-4	4.1e-4	0.159	0.110	39.5%	33.7%
CAT	4.1e-4	4.3e-4	0.122	0.096	53.8%	29.5%

TABLE 5.7: Performance metrics for the various ML algorithms across the 2 years under examination with the usage of absolute shifts.

It is noticeable that all the proposed algorithms significantly outperform the benchmark model, which is the original cross-sectional model, in all the considered metrics.
Among the 4 proposed algorithms, random forest and support vector regression are those that perform best. This can be seen from the performance on the out of sample set, which is unbiased. CatBoost and the random forest algorithm show some over-fitting even after the hyperparameter tuning. However, despite the tendency to overfit the dataset, the random forest algorithm is the one that provides the best performance on the test set.
As mentioned in Section 4.2.8, removing outliers from the fitting of the model, drastically improved the proxy performance. Outliers are part of idiosyncratic risk and since the proxy is supposed to replicate systematic risk, the presence of outliers

behaves like noise for the proxy as these do not represent the average behaviour of the market.
Table 5.8 below shows a significant boost in the proxy accuracy due to the removal of outliers from the fitting of the regression model.
This strategy comes with a trade-off between accuracy in terms of the predicted shift and accuracy in terms of standard deviations. In fact, removing outliers from the fitting increased the accuracy in terms of RMSE, R^2 coefficient and correlation, however, the proxy without these outliers is more conservative and the VaR calculation is underestimated. This problem is tackled in the next subsection.

Performance on absolute shifts without outliers in the fitting						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	2.3e-4	2.4e-4	0.095	0.076	28.0%	25.4%
LR	2.1e-4	2.2e-4	0.237	0.216	47.1 %	45.0%
RF	1.9e-4	2.1e-4	0.413	0.277	65.8 %	51.4%
SVR	2.0e-4	2.1e-4	0.320	0.280	56.2%	51.7%
CAT	2.3e-4	2.3e-4	0.268	0.242	51.4%	46.3%

TABLE 5.8: Performance metrics of ML algorithms and benchmark model across the 2 years under examination with the usage of absolute shifts after remove from fitting the outliers. The removed outliers are those far more than 3 standard deviations from the mean.

The improvement of the performance obtained by this outlier filtering is remarkable: the RMSE is approximately halved, the R^2 coefficient is almost tripled and the correlation significantly increased for all the algorithms considered.
As in the previous scenario, the random forest and support vector machine algorithms are those that better perform on the out of sample test.
Similarly to the Table 5.7, the correlation generally doubled in the comparison with the benchmark model, which is a highly satisfactory result. Overall, the results obtained both in the scenarios with and without outliers are remarkable as they present a net improvement with respect to the current proxy model for bond credit spreads.

5.2.3 VaR Comparison

The main application of the bond credit spreads proxy is to approximate the VaR that one would obtain with the real data. In order to test it, we selected the bonds with full history, which are 438, and calculate the VaR obtained for these bonds together with the VaR obtained by our proxy.
Our claim is that, for a large portfolio of bonds, the VaR obtained with real data and the VaR obtained with proxied data should converge. In order to mathematically express this reasoning we decompose the portfolio risk in a systematic risk component, which resembles market risk and an idiosyncratic risk component, which is the individual risk of each counterparty and we assume it to be independent from the others. Our proxy is supposed to model systematic risk and should not incorporate idiosyncratic risk.
We define $Y_i := \Delta z_i$ as the change in the z-spread of the i-th bond on a certain date. Then Y_i can be modelled as:

$$\begin{aligned} Y_i &= Y_i^{syst} + Y_i^{idio}, \quad \forall i, \\ Y_i^{syst} &= \beta_0 + \sum_{k=1}^{N_{features}} \sum_{j=1}^{N_k} \beta_j^k I_j^k(i), \quad \forall i, \\ Y_i^{idio} &\sim_{i.i.d} \mathcal{N}(0, \sigma_i^{idio}), \quad \forall i. \end{aligned} \tag{5.5}$$

The result of our proxy, here reported in case of modelling with linear regression, is Y_i^{syst}, the systematic risk component. The equation for the systematic risk component model is explained in detail in Section 4.2.8. Here, it is summarized first in a summation on the number of features, e.g. seniority, currency etc, and then an internal summation on the levels within each category, e.g. EUR, USD, GBP.
The indicator function takes the value one when the i-th bond is in the same categorical level of the j-th level. β_0 is the benchmark bucket that has been explained in detail in the aforementioned section. In order to proceed with our modelling, we need to explain the relation between z-spread and VaR.

VaR and Z-spread

We recall that the theoretical price of a zero coupon bond and the z-spread are linked by the following relationship:

$$\begin{aligned} P(t_0) &= FVe^{-r_{(T-t_0)}(T-t_0)}, \\ MV(t_0) &= FVe^{-(r+z)_{(T-t_0)}(T-t_0)}. \end{aligned} \tag{5.6}$$

Where $P(t_0)$ is the theoretical price of a risk-free bond at time t_0, FV is the face value of the bond, $MV(t_0)$ is the market value, r is the zero coupon rate taken from the treasury yield curve and z is the z-spread.
Then in order to calculate HVaR we need n observations of MV in order to generate the n scenarios. $\Delta MV = \{MV_0, MV_1, ..., MV_n\}$.
We can derive the change in MV of the bond by shocking z, i.e. by a sensitivity

analysis on the variation of z:

$$\begin{aligned}\Delta MV &= FVe^{-(r+z)_{(T-t)_0)}(T-t_0)}(e^{-\Delta z(T-t_0)}-1)\\ &= \frac{\partial MV}{\partial z}(\frac{1-e^{-\Delta z(T-t_0)}}{T-t_0})\\ &= \frac{\partial MV}{\partial z}(\Delta z + o(\Delta z)^2)\\ &\approx \frac{\partial MV}{\partial z}(\Delta z).\end{aligned} \tag{5.7}$$

Where in the last equation a first-order Taylor approximation has been used. From here it can be seen that the variation in the market value of a bond is proportional to the variation in the z-spread.
Now we consider a multi-bond portfolio composed of N_{Bonds}, in which the i-th bond is weighted by ω_i. The sum of the weights in the portfolio should be finite and for simplicity we set it to one, i.e. $\sum_{i=1}^{N_{Bonds}} \omega_i = 1$. We can then define the P&L for this portfolio as the sum of the daily change in the market value of every i-th bond weighted by the respective ω_i:

$$P\&L(t) = \sum_{i=1}^{N_{Bonds}} \omega_i \Delta MV_i. \tag{5.8}$$

As we previously showed in (5.7), this is proportional to the variation in the z-spread of each bond: $P\&L(t) \propto \sum_{i=1}^{N_{Bonds}} \omega_i \Delta z_i = \sum_{i=1}^{N_{Bonds}} \omega_i Y_i$.

Now we consider an infinitely large portfolio such that $\lim_{N_{Bonds}\to\infty} \omega_i = 0$. An infinitely large portfolio is not a realistic assumption but it serves for our scope, in fact, now we can invoke the Law of Large Numbers and apply it to our P&L:

$$\begin{aligned}\lim_{N_{Bonds}\to\infty} \sum_{i=1}^{N_{Bonds}} \omega_i Y_i &= \lim_{N_{Bonds}\to\infty} \sum_{i=1}^{N_{Bonds}} \omega_i Y_i^{syst} + \lim_{N_{Bonds}\to\infty} \sum_{i=1}^{N_{Bonds}} \omega_i Y_i^{idio}\\ &= \mathbf{E}[Y^{syst}] + 0.\end{aligned} \tag{5.9}$$

This means that for an infinitely large portfolio, the P&L and therefore also the VaR of the real-data portfolio converges to the deterministic value that is the systematic risk contribution, which is exactly the result of our proxy. This result is of crucial importance and it means that for VaR purposes the approach is correct in the limit of an infinite number of bonds.
In this analysis the number of bonds that have full z-spread history is 438 and in the following we show that this number might be sufficient to have convergence between the 'true' VaR and the 'proxy' VaR.

VaR comparison for bonds with full history

Following the previous reasoning, we can calculate the VaR for our portfolio composed of 438 bonds in which we assume that the weights are homogeneously distributed across the bonds. The VaR resulting from the calculation is only proportional to the actual VaR, but it serves for our purpose since we are interested in a comparison of the same portfolio with real and proxied data and therefore the scaling factor is not necessary.

In Figure 5.9, a plot of the VaR calculated in the two years period for each algorithm is shown together with the plot of the 'true' VaR in purple.

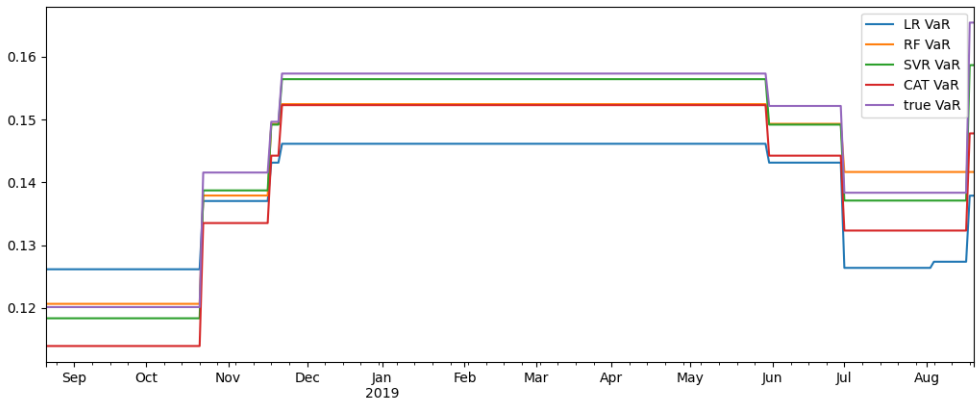


FIGURE 5.9: VaR comparison in the period from September 2017 to September 2019, between the real data and proxied data. The proxied data are calculated with Linear Regression (blue), Random Forest (orange), Support Vector Regression (green) and CatBoost (red).

Similarly as in the previous section: the random forest (orange line) and support vector regression (green line) are those that better resemble the behaviour of the 'true' VaR. Generally, this plot shows a very satisfying result, all the algorithms similarly replicate the VaR obtained with real data, despite the fact that we are not considering idiosyncratic risk. This is in favour of the previous reasoning: for a large number of bonds the VaR is given by the systematic risk contribution.
This is a remarkable result for this thesis, it basically confirms that the approach used is theoretically and empirically correct if the purpose is the VaR calculation and the portfolio is large enough.
Neglecting idiosyncratic risk introduces an underestimation error in the 'proxy' VaR, which converges to zero for an increasing number of bonds. In Figure 5.10, we plot the mean percentage underestimation error (UE) for an increasing number of bonds considered in the portfolio. The aforementioned error is calculated as follows:

$$UE = \frac{1}{T}\sum_{i=1}^{T}\frac{VaR_i^{True} - VaR_i^{Proxy}}{VaR_i^{True}}. \tag{5.10}$$

Here T is the number of days in examination, because the VaR is calculated daily and the error is averaged on the time span. The VaR_i^{True} and VaR_i^{Proxy} are calculated across the 438 bonds for every day.
The plot shows that the underestimation error (UE) approaches zero for an increasing number of bonds and the random forest (orange line) and support vector regression (green line) are the faster in this convergence with just a 1% underestimation error, against linear regression and CatBoost that reach approximately 5%.
Again this result confirms our claim. However, in this calculation outliers have not been filtered out from the fitting and therefore the accuracy obtained with the proxy is lower as seen in Section 5.2.2.

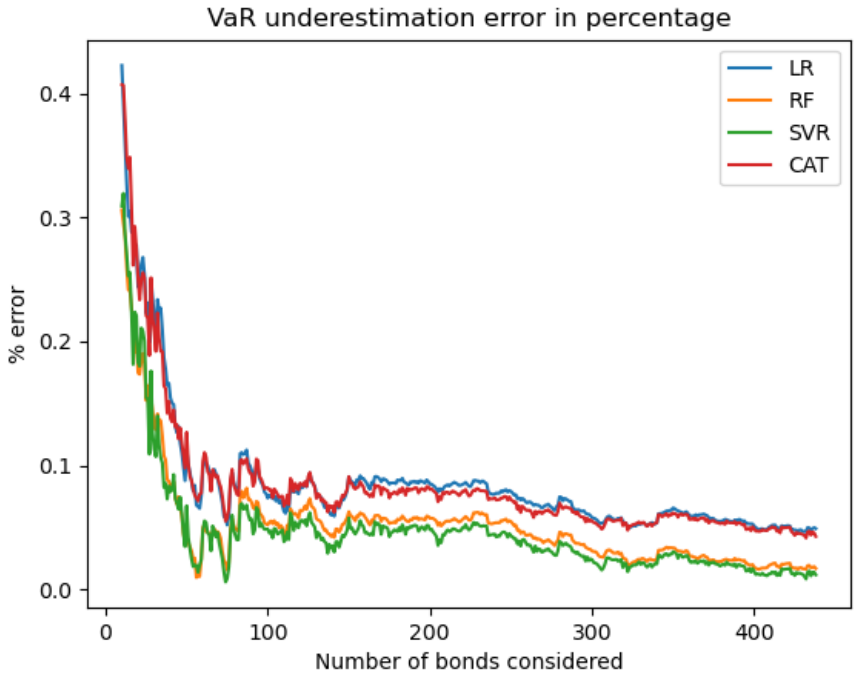


FIGURE 5.10: VaR underestimation error in function of an increasing number of bonds for different ML algorithms. The error is calculated as a percentage of the error mean across time.

VaR comparison for bonds with full history and outliers filtering

Whenever outliers are filtered out from the fitting of the proxy model, the VaR obtained suffers from a stronger underestimation. This is due to the fact that removing the more extreme observations produces a less conservative VaR. This is not in contradiction with the previous reasoning. In fact, the VaR should still converge to the true one, but it requires a larger number of bonds. Figure 5.11, shows the VaR calculation across the 2 years period after removing outliers from the fitting of the model. It is easily noticeable that the result is quite different from the previous case. The underestimation worsened, as expected, and it is the case specially for random forest and support vector regression which better performed in the previous scenario.
The removal of outliers from the fitting strongly affects the performance of random forest and support vector regression algorithms. Whereas CatBoost and Linear Regression performance is less penalized, in terms of VaR underestimation, by the removal of the outliers.

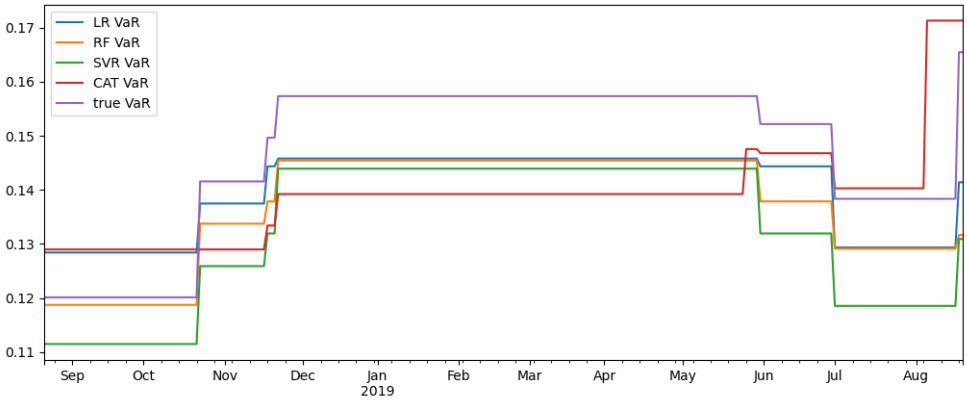


FIGURE 5.11: VaR comparison in the period from September 2017 to September 2019, between the real data and proxied data after removing outliers from the fitting. The proxied data are calculated with Linear Regression (blue), Random Forest (orange), Support Vector Regression (green) and CatBoost (red).

Similarly as before, we plot the mean percentage underestimation error for an increasing number of bonds in Figure 5.12. The plot shows a drastically higher underestimation for random forest and specially for support vector regression algorithms. The performance of CatBoost and linear regression is almost not affected by the outliers removal in terms of underestimation error. This result implies that, despite the fact that support vector regression and random forest perform better in term of accuracy, the usage of linear regression or CatBoost is recommended for VaR calculation in case of outliers filtering.

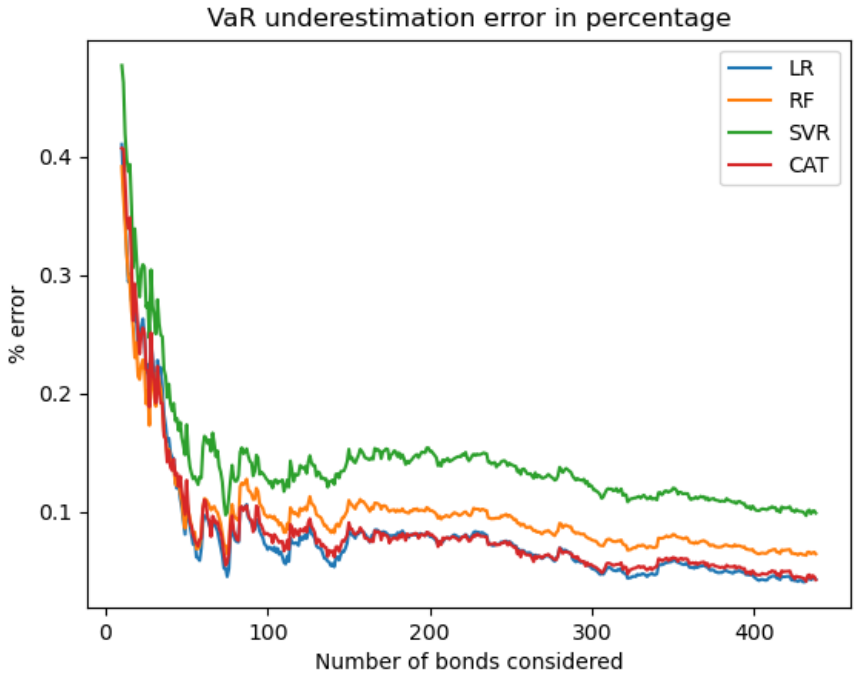


FIGURE 5.12: VaR underestimation error in function of an increasing number of bonds for different ML algorithms after removing outliers from the fitting. The error is calculated as a percentage of the error mean across time.

Idiosyncratic Risk Simulation

Another possibility that allows to improve the convergence of our 'proxy' VaR to the 'true' VaR is to simulate idiosyncratic risk.
For an infinite number of bonds this procedure has no effect as we showed with the law of large numbers, but for a countable portfolio, it speeds up convergence.
Following our previous assumption that idiosyncratic risk is independently normally distributed across bonds we decided to simulate it by looking at the main bond categorical feature: the rating class.
The procedure we follow in order to simulate idiosyncratic risk is to apply Gaussian noise by first calculating the mean standard deviation for different rating classes σ_{rating}, e.g. σ_{AAA} is the mean standard deviation of 'AAA' rated bond z-spread shifts. Secondly, for every predicted time series, the difference between the time series standard deviation and the standard deviation related to the respective rating class is calculated:

$$\begin{aligned} \Delta\sigma_i &= \sigma_{rating}(i) - \sigma_i, \qquad i = 1, ..., N_{Bonds} \\ \Delta\sigma_i &= \max(0, \Delta\sigma_i). \end{aligned} \tag{5.11}$$

Then we simulate a multivariate normal matrix $Q \sim N(0, \Delta\sigma_i)$ with N_{Bonds} columns and T rows, where T is the number of days in exam which is 520.
Finally, this multivariate Gaussian noise is added to the output prediction of our proxy (in which outliers are filtered from the fitting).

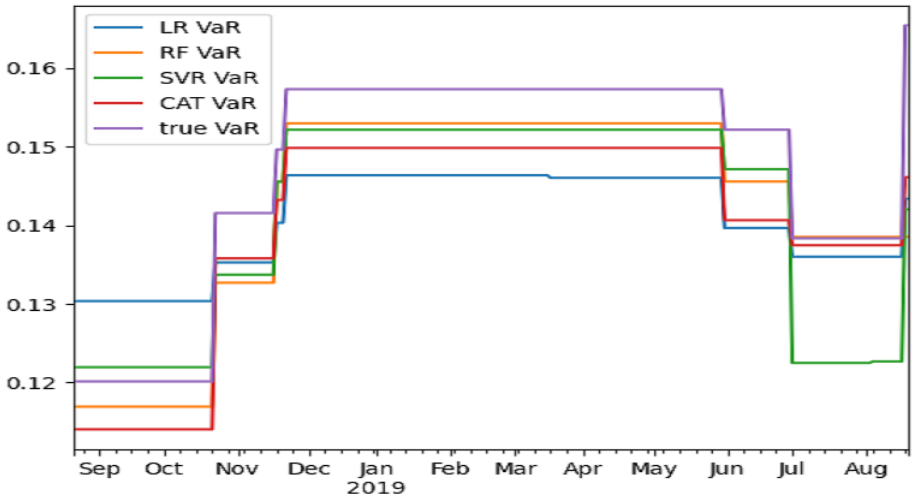


FIGURE 5.13: VaR comparison in the period from September 2017 to September 2019, between the real data and proxied data after removing outliers from the fitting and the addition of idiosyncratic risk simulation. The proxied data are calculated with Linear Regression (blue), Random Forest (orange), Support Vector Regression (green) and CatBoost (red).

Figure 5.13 shows a net improvement in the performance of random forest and support vector regression after the inclusion of the simulated idiosyncratic risk contribution. The underestimation effect is still heavier than the scenario with the outliers included in the fitting, but the improvement is remarkable.
Similarly as we done previously, the mean percentage underestimation error is shown in Figure 5.14.
This method is not robust for a small number of bonds ($n < 200$), but it gains stability for a larger portfolio. This means that a realistic portfolio that accounts for a number of bonds larger than 200 can strongly benefit from this approach as it generally decreases the underestimation error even for a medium-sized portfolio.
Overall, it is not straightforward to assess which strategy is the best one, as it really depends on the size of the considered portfolio and the main application of the proxy. In general this idiosyncratic risk simulation allows for a great accuracy, as the noise is applied only afterwards, for VaR purposes where the underestimation error is reduced.

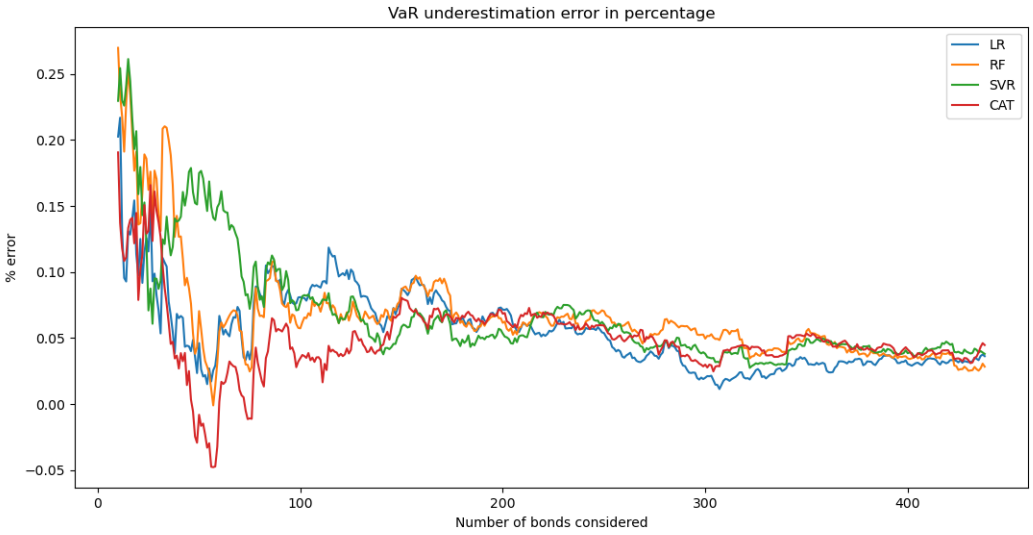


FIGURE 5.14: VaR underestimation error in function of an increasing number of bonds for different ML algorithms after removing outliers from the fitting and adding simulated idiosyncratic risk. The error is calculated as a percentage of the error mean across time.

Chapter 6

Further Developments

In this chapter some further developments of the bond credit spreads proxy are discussed. The main room for improvement relies in including time information, which was not considered in the actual state of the proxy.
In fact, the proxy built so far is fitted with daily z-spread shifts and therefore, it uses only the information available on a single day across the bonds to proxy missing data points.
Time information can be included in our proxy by means of an auto-regressive (AR) model. This can be done if bond z-spread time series present some significant autocorrelation as it is shown in the following section.
The auto-regressive model implementation can be a great support for this analysis. However, the results obtained so far by the 'daily' proxy are satisfactory and therefore, the goal of this chapter is to combine the two model without losing the achievements obtained with the previous model.
In this chapter we are only considering the 438 bonds that have full history. For this reason the reported performance of the 'daily' proxy is different from the previous section. Also, the analysis is presented only on absolute shifts. Anyway, similar conclusions can be drawn for the other suggested shift types.

6.1 Bond Z-Spread Shift Autocorrelation

In order to implement the AR model, we first need to verify the presence of autocorrelation in the bond z-spread shifts. The autocorrelation is derived from the autocovariance, which is nothing more than the covariance computed for a time-series variable. It is the covariance of y_t with itself in the past, i.e.

$$\begin{aligned} \gamma_y(s,t) &= cov(y_s, y_t) \quad with \;\; 0 < t < s, \\ \gamma_y(h) &= cov(y_{t+h}, y_t). \end{aligned} \tag{6.1}$$

Given the autocovariance, we can define autocorrelation as:

$$\rho_y(s,t) = \frac{\gamma_y(s,t)}{\sqrt{\gamma_y(s,s)\gamma_y(t,t)}} \quad with \;\; 0 < t < s. \tag{6.2}$$

Autocorrelation can be verified by an autocorrelation function (ACF) plot. The ACF is the function that describes how the autocorrelation varies with time lags. For the bond z-spreads time series, one lag correspond to one day.
The ACF plot for a single bond is shown in Figure 6.1. This plot correspond to one out of the 438 bonds with full history analysed in this section.
It is clearly infeasible to analyse one by one each of the 438 bond z-spread shift time series. Therefore, a larger scale approach has been implemented. This is the

Ljung–Box test, which has been introduced by Greta M. Ljung and George E. P. Box in 1978. A brief overview of the test methodology and target is presented in the following, while a more extensive explanation can be found in Ljung and Box, 1978.

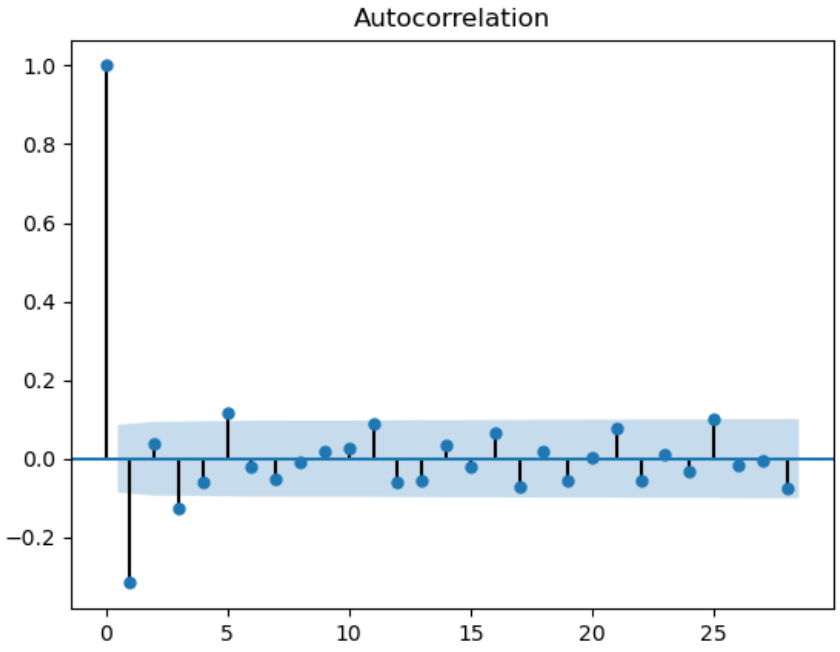


FIGURE 6.1: On the y axis the value of the autocorrelation for a single bond with respect to the lag that is shown on the x axis.

The Ljung-Box test is a statistical test for checking if for a certain number of lags the time-series exhibits autocorrelations different from zero. The null-hypothesis H_0 assumes that data are independently distributed and therefore the correlation is zero. Conversely, the alternative hypothesis H_1 is that data are not independently distributed, i.e., data exhibit serial correlation.
The test statistic of the Ljung-Box test is defined by:

$$Q = n(n+2)\sum_{k=1}^{h}\frac{\hat{\rho}_k^2}{n-k}, \tag{6.3}$$

where n is the sample size, h is the number of tested lags and $\hat{\rho}_k$ is the autocorrelation at lag k. Under the null-hypothesis the test statistic Q follows a chi-squared distribution with h degrees of freedom, i.e. $Q \sim \chi^2(h, 0)$.
The Ljung-Box test has been implemented on all the 438 bonds with $h = 5$ lags, which means that all the lags up to 5 are tested. The results of the test, i.e. the p-values, are summarized in Figure 6.2.
The histogram shows that for more than 350, out of 438 bonds, the null hypothesis is rejected at a significance level of 5%, i.e. serial correlation is present in the majority of the z-spread shift time-series. The result obtained with the Ljung-Box test implies the presence of an autoregressive component in the bond z-spread shift time-series. Therefore, it opens the door for an autoregressive (AR) model implementation.

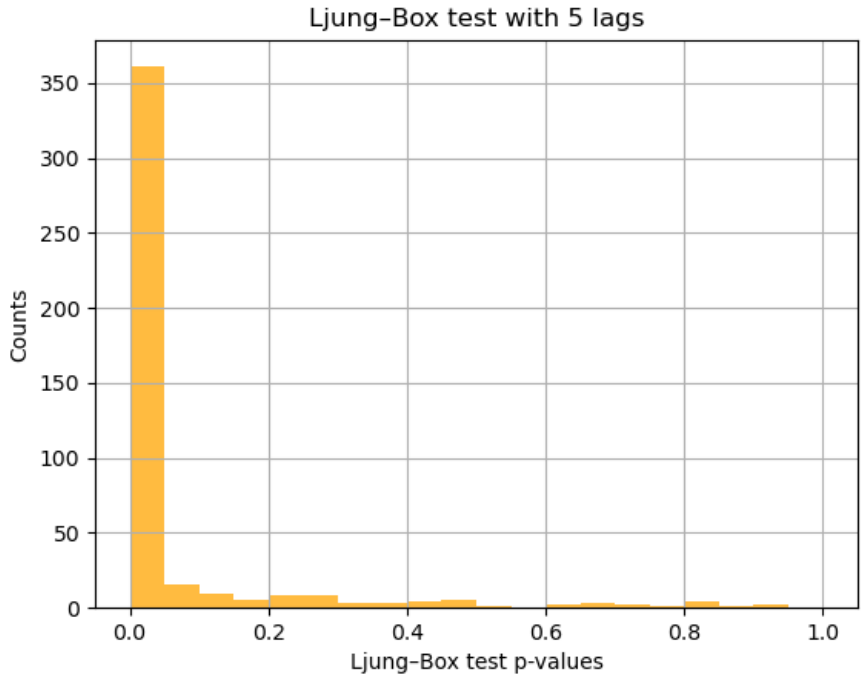


FIGURE 6.2: Histogram showing the result of the Ljung-Box test. On the x axis the p-values obtained for each bond and on the y axis the number of bonds contained in each bin.

6.2 Bond Z-Spread Shift Autoregressive Model

After verifying the existance of autocorrelation in bond z-spread shifts, an AR model has been built. The chosen number of autoregressive lags is 5, same as those tested in the previous section. An autoregressive model predicts the future behaviour based on the past behaviour. It is basically a linear regression of the current data against $h = 5$ past values of the same time-series.
An autoregressive model of order 5 is often written with the notation: $AR(5)$. This model is defined as:

$$Y_t = c + \sum_{i=1}^{h=5} \phi_i Y_{t-i} + \epsilon_t. \tag{6.4}$$

Where c is a constant, ϕ_i is the parameter related to the i-th lag and ϵ_t is white noise. The model has been fit sequentially on each of the 438 bonds with full-history. In order to have a realistic framework for our prediction we applied a 'mask' on the 438 bonds time-series such that these include missing values as well. This has been done by randomly selecting 438 bonds time-series containing at least 80% of the data and apply this mask on the 438 full-history bonds. Therefore, the complete data-set has been used for the fitting but a mask has been applied for the prediction. Bond z-spread shift time-series are stationary, therefore no train-test split has been applied in this exploratory part.
The performance obtained by the $AR(5)$ together with that obtained with our previous proxy is presented in the following table. For what concerns the proxy we present the performance of the random forest algorithm without outliers in the

fitting, which together with support vector regression provided the best results in terms of accuracy.

Performance of $AR(5)$ model and RF proxy algorithm					
	RMSE	R^2	Cor	AIC	BIC
$AR(5)$	2.4e-4	0.07	25.7 %	-7333	-7304
RF	2.0e-4	0.29	60.2%	-9024	-7192

TABLE 6.1: Performance metrics of $AR(5)$ model and Random Forest proxy algorithm for the 438 bonds with full-history.All metrics presented in this table are calculated as mean across the 438 time-series.

It is clear from Table 6.2 that the 'daily' proxy significantly outperforms the autoregressive model. The only metric in which $AR(5)$ model performed better than the proxy is the BIC and this is due to the higher penalization on the number of parameters in the BIC calculation.
We can definitely say that the 'daily' proxy is a better model than the $AR(5)$, but an opportune combination of the two models could yield a better performance.

6.3 Bond Z-Spread Shift Mixed Model

The purpose of this section is to combine the results of the two model into a single mixed model that outperform the previous two.
We define $Y_i^{proxy}(t)$ as the z-spread shift of the i-th bond at time t calculated by means of the daily proxy. Conversely, we define $Y_i^{AR}(t)$ as the z-spread shift calculated with the autoregressive model with 5 lags.
Two possible approaches have been analysed in order to combine the result of the two models: unconstrained linear regression (URL) and LASSO regression, i.e. constrained linear regression. In the following these two approaches are described.

Unconstrained Linear Regression
This method consists in estimating the parameters β_1 and β_2, defined by the following equation, by means of linear regression.

$$Y^{mixed} = \beta_1 Y^{proxy} + \beta_2 Y^{AR}, \tag{6.5}$$

where Y^{proxy} and Y^{AR} are vectors containing all bonds z-spread shifts across the two years under examination. No constraint has been imposed on the parameters β_1 and β_2. The resulting parameters are $\beta_1 = 1.13$ and $\beta_2 = 0.76$

Lasso Regression
Least Absolute Shrinkage and Selection Operator formulated in Tibshirani, 1996, allows to perform linear regression with a regularization term. The parameters are defined by the following equation:

$$(\beta_1, \beta_2) = argmin_\beta[(Y^{True} - \beta_1 Y^{proxy})^2 + (Y^{True} - \beta_2 Y^{AR})^2 + \alpha(|\beta_1| + |\beta_2|], \tag{6.6}$$

in which α can be opportunely tuned such that the model equation results in:

$$Y^{mixed} = \beta_1 Y^{proxy} + \beta_2 Y^{AR} \quad with \;\; \beta_1 + \beta_2 = 1. \tag{6.7}$$

The parameters resulting from the constrained linear regression are $\beta_1 = 0.81$ and $\beta_2 = 0.19$. Fixing the sum of the two parameters to be one has been proposed because of its interpretation but it is not mandatory.
The performance obtained with these two mixed models is presented below.

Performance of the mixed model with URL and LASSO					
	RMSE	R^2	Cor	AIC	BIC
ULR	1.9e-4	0.32	63.0%	-9057	-9049
LASSO	2.0e-4	0.29	62.1%	-9028	-9020

TABLE 6.2: Performance metrics of the mixed model with unconstrained linear regression and LASSO approaches. All metrics presented in this table are calculated as mean across the 438 time-series.

Table 6.3, shows that unconditional linear regression slightly overcome LASSO approach as a mixing procedure for the autoregressive and proxy models. This is related to the issue of standard deviation underestimation presented in Chapter 5. In fact, since both models are generally conservative, a mixed approach that increase the standard deviation of the model is preferred.
Both mixed models improved the stand-alone performance of the 'daily' proxy which is a satisfactory result.
The methods used and the results obtained in this section need to be formalized and aim to be a suggestion for possible further improvements of the 'daily'proxy model.

Chapter 7

Conclusions

In this chapter, we summarize the main aspects of our research and draw conclusions from the obtained achievements.
We start from the shift types assessment, which is the foundation of our proxy methodology. The shift types assessment has a crucial role in the main application of this thesis which is the historical Value at Risk (HVaR). Many financial institutions adopted the usage of relative shifts for bond z-spreads modelling. In our research we showed this to be an undesirable choice since relative shifts have a vertical asymptote for value of the z-spread that are close to 0. This violates the main assumption of HVaR, i.e. market price changes are identically independently distributed (i.i.d).
To overcome this issue we provided a methodology in which, by analysing the behaviour of the modulus of the z-spread shifts in relation with the z-spread level, we set the shift fluctuations to be homogeneously distributed on the different z-spread levels. This resulted in the proposal of three different possible shift types: absolute, displaced relative and arcsinh shifts. These provide similar results and each one fits well with the HVaR foundation. During the course of the analysis we mainly applied absolute shifts as these provide a more simple and intuitive framework and do not require parameters tuning.
Once the choice of the shift type has been made, we can start calibrating our model on top on that. The reason why the shift type choice is crucial for the credit spreads proxy goes beyond the HVaR application. In fact, one of the main differences between the approach of this research and the previous proxy models such as the Intersectional Method introduced in EBA, 2013 and the Cross-Sectional Method presented in (Chourdakis et al., 2013), is that we decided to directly proxy the shifts or daily changes in the z-spread value instead of the z-spread values themselves.
This choice provided better performances and it also fits better with the HVaR application, as in HVaR the shifts are directly used to generate the possible future scenarios for the z-spread values.
The Cross-Sectional Method has been used in this thesis as benchmark model. The reason for this is that it significantly overcome many of the limitation of the Intersectional Method and furthermore its application is wide-spread across financial institutions. However, the Cross-Sectional Method provides quite poor predictive accuracy. This has been significantly improved by the inclusion of extra regressors in our proxy model, such as currency, time to maturity and market indicator, which is a binary variable that discriminates between emerging and developed market. In Chapter 3, we visually showed that these extra regressors provide useful information to more completely represent the bond features. Another major improvement in terms of predictive accuracy relies in the inclusion of non-linearties and interacting terms in the regression model by means of different machine learning algorithms such as Random Forest (RF), CatBoost and Support Vector Regression(SVR). Among these three algorithms the RF and SVR are those that better performed. However,

the performance of each of the proposed algorithm significantly outperformed the benchmark model.
A crucial step in the enhancement of the proxy accuracy was found in the removal of outliers from the fitting of the proxy model. The removal of the outliers from the fitting was justified by the aim of the proxy, which is modelling and replicating systematic risk, i.e. the average behaviour of the market. Outliers are part of idiosyncratic risk which is the risk incorporated by a specific entity and therefore it is not supposed to be modelled by the bond credit spreads proxy. Specially, after the removal of outliers from the fitting, the performance of RF and SVR obtained a striking performance, the correlation more than doubled with respect to the benchmark model and the R^2 coefficient was almost 4 times higher.
The main result of this thesis from a financial and statistical point of view, relies in the VaR comparison between the 'true VaR' and the 'proxy VaR', i.e. the VaR obtained with real data and the VaR obtained with proxied data on the same portfolio composition. We theoretically proved that neglecting idiosyncratic risk does not provide drawbacks in terms of VaR calculation if the portfolio is infinitely large. And we empirically proved that for a sufficiently large portfolio the VaR underestimation is small, up to 1% for RF and SVR by considering a portfolio composed by 438 bonds. This result is of vital importance for this thesis as it basically confirms the correctness of our approach for VaR calculation purposes, i.e. the VaR of the real-data portfolio converges to a deterministic value which is given by the systematic risk component that is what our proxy model aims to predict.
The performed outliers filtering in the model fitting brought an impressive improvement in the accuracy of the proxy, but it generated a smaller predicted volatility which resulted in a significant underestimation of the predicted VaR. This has been compensated by an idiosyncratic risk simulation that can be adopted for the VaR calculation only. However, the removal of outliers from the fitting, necessarily decrease the ability of the proxy to predict large fluctuations in the z-spread shifts. The inclusion of a idiosyncratic noise component improved the aforementioned issue resulting in a VaR underestimation clustered around 5% for a portfolio composed by 438 bonds.
This completes the main part of our research, i.e. the construction and application of a bond credit spreads proxy, in order to significantly improve the current methodologies for illiquid markets and partially solved the shortage of liquidity problem. It is necessary to mention that our analysis concerned a 2 years period from August 2017 to August 2019, which is a relatively short amount of time for financial analysis. Therefore, our conclusions hold for similar periods that are characterized by a specific market behaviour.
Our research expanded beyond the improvement of the 'daily' proxy, which refers to the fact that the existing proxy methodology and those that we presented are fitted day-by-day and use only the information available on each single day to make predictions. This is one of the main limitation of existing proxy methodology, the contribution brought by information across time should not be ignored in order to obtain a more sophisticated and effective proxy. For this reason in the further developments section, Chapter 6, we showed the presence of autoregressive components in the bond z-spreads shifts. This opened the doors to a vast majority of possible approaches. Among these, we focused on a simple one, the construction of an autoregressive (AR) model with 5 time lags. The AR model by itself did not obtained a performance comparable with the 'daily' proxy that uses ML algorithms, however we showed that a combination of the two models can indeed improve the performance of the single models.

Overall, the research provided satisfying results. The main goal of the project, which was the realization of an improved proxy framework that can be easily applied by financial institution, has been accomplished. On top of that, we provided an extensive research on the best shift types for bond z-spreads, which is an uncovered topic in the financial research documentation.
Finally we shed light on a new possible approach for credit spreads proxying, which relies in the inclusion of historical data (potentially from past and future) for the modelling and prediction of the bond z-spreads. Our analysis on this topic aims to be an introduction for a more sophisticated and formal procedure that can dramatically increase the potential of future proxy.
We believe that the new insights provided by this research can be a significant benefit for the financial industry with relevant content also for the scientific community.

Appendix A

Performance of Other Shift Types

A.1 Performance of Arcsinh Shifts

Performance of different algorithms on arcsinh shifts						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	2.92e-2	2.98e-2	0.034	0.021	18.0%	15.3%
LR	2.68e-2	2.73e-2	0.110	0.104	31.3 %	30.6%
RF	2.54e-2	2.64e-2	0.265	0.126	54.8 %	34.6%
SVR	2.61e-2	2.64e-2	0.165	0.128	39.8%	35.1%
CAT	2.61e-2	2.74e-2	0.213	0.108	50.4%	31.4%

TABLE A.1: Performance metrics for the various ML algorithms across the 2 years under examination with the usage of arcsinh shifts.

Performance on arcsinh shifts without outliers in the fitting						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	1.62e-2	1.63e-2	0.078	0.061	27.7%	24.7%
LR	1.46e-2	1.47e-2	0.234	0.216	46.8 %	45.1%
RF	1.28e-2	1.40e-2	0.406	0.282	65.3 %	52.0%
SVR	1.34e-2	1.37e-2	0.323	0.285	56.3%	52.4%
CAT	1.54e-2	1.57e-2	0.243	0.226	52.0%	47.0%

TABLE A.2: Performance metrics for the various ML algorithms across the 2 years under examination with the usage of arcsinh shifts after removing outliers (3 or more standard deviations far from the mean) from the fitting.

A.2 Performance of Displaced Relative Shifts

Performance of different algorithms on displaced relative shifts						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	2.09e-2	2.10e-2	0.035	0.004	17.9%	14.9%
LR	2.01e-2	2.00e-2	0.104	0.082	30.4 %	29.6%
RF	1.84e-2	1.85e-2	0.255	0.100	53.9 %	33.2%
SVR	1.97e-2	1.84e-2	0.152	0.106	38.0%	33.6%
CAT	1.87e-2	1.89e-2	0.203	0.098	50.2%	30.1%

TABLE A.3: Performance metrics for the various ML algorithms across the 2 years under examination with the usage of displaced relative shifts.

Performance on displaced relative shifts without outliers in the fitting						
	Train RMSE	Test RMSE	Train R^2	Test R^2	Train Cor	Test Cor
benchmark	1.08e-2	1.09e-2	0.076	0.060	26.8%	24.3%
LR	0.97e-2	0.98e-2	0.224	0.207	45.9 %	44.1%
RF	0.86e-2	0.94e-2	0.398	0.270	64.9 %	51.0%
SVR	0.92e-2	0.94e-2	0.312	0.275	55.3%	51.4%
CAT	1.04e-2	1.07e-2	0.243	0.226	52.0%	47.0%

TABLE A.4: Performance metrics for the various ML algorithms across the 2 years under examination with the usage of displaced relative shifts after removing outliers (3 or more standard deviations far from the mean) from the fitting.

Bibliography

Awad, Mariette and Rahul Khanna (2015). "Support vector regression". In: *Efficient learning machines*. Springer, pp. 67–80.

Bergstra, James and Yoshua Bengio (2012). "Random search for hyper-parameter optimization". In: *The Journal of Machine Learning Research* 13.1, pp. 281–305.

Biau, GÃŠrard (2012). "Analysis of a random forests model". In: *Journal of Machine Learning Research* 13.Apr, pp. 1063–1095.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Breiman, Leo et al. (1984). "Classification and regression trees. Wadsworth Int". In: *Group* 37.15, pp. 237–251.

Brummelhuis, Raymond and Zhongmin Luo (2017). "Cds rate construction methods by Machine Learning Techniques". In: *Available at SSRN 2967184*.

Choudhry, Moorad (2003). *Bond and money markets: strategy, trading, analysis*. Butterworth-Heinemann.

— (2006). "Revisiting the credit default swap basis: further analysis of the cash and synthetic credit market differential". In: *The Journal of Structured Finance* 11.4, pp. 21–32.

Chourdakis, Kyriakos et al. (2013). "A cross-section across CVA". In: *Nomura. Available at Nomura: http://www. nomura. com/resources/europe/pdfs/cva-crosssection. pdf*.

Christoffersen, Peter F (1998). "Evaluating interval forecasts". In: *International economic review*, pp. 841–862.

Cortes, Corinna and Vladimir Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297.

Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin (2018). "CatBoost: gradient boosting with categorical features support". In: *arXiv preprint arXiv:1810.11363*.

EBA (2013). *Technical standards in relation with credit value adjustment*.

— (2016). *EBA Final draft Regulatory Technical Standards*.

Friedman, Jerome H (2001). "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics*, pp. 1189–1232.

Fries, Christian P, Tobias Nigbur, and Norman Seeger (2017). "Displaced relative changes in historical simulation: Application to risk measures of interest rates with phases of negative rates". In: *Journal of Empirical Finance* 42, pp. 175–198.

FTSE (2020). *FTSE Equity Country Classification Process*.

Géron, Aurélien (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Gordon (2015). *Regression Analysis for the Social Sciences*, p. 201.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

ING (2018a). *Credit Proxy Methodology Specifications*.

— (2018b). *Historical Value at Risk Policy*.

— (2020). "HVaR Risk factor shift type methods". In:

James, Gareth et al. (2013). *An introduction to statistical learning*. Vol. 112. Springer.

Liu, Yong and Xin Yao (1999). "Ensemble learning via negative correlation". In: *Neural networks* 12.10, pp. 1399–1404.

Ljung, Greta M and George EP Box (1978). "On a measure of lack of fit in time series models". In: *Biometrika* 65.2, pp. 297–303.

Platt, John et al. (1999). "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". In: *Advances in large margin classifiers* 10.3, pp. 61–74.

Prokhorenkova, Liudmila et al. (2018). "CatBoost: unbiased boosting with categorical features". In: *Advances in neural information processing systems*, pp. 6638–6648.

Romano, Joseph P and EL Lehmann (2005). *Testing statistical hypotheses*. Springer Berlin.

Smola, Alex J and Bernhard Schölkopf (2004). "A tutorial on support vector regression". In: *Statistics and computing* 14.3, pp. 199–222.

Tibshirani, Robert (1996). "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1, pp. 267–288.