LUISS

Department
of Business and Management

Advancing the Merton Model: The impact of Deep Learning – Based Volatility on Probability of Default.

Professor Pierluigi Murro
Supervisor

Alessandro Micci Battaglini
Candidate

ID Number
759921

Accademic Year 2023 – 2024

Table of Contents

Introduction

Black and Scholes published their work in 1973, followed by Merton in 1974, introducing the basic approach for valuing companies' stock and debt as derivatives on the companies' asset value. Since then, a vast number of different models addressing corporate default has been published, trying to relax the strict assumptions of the Merton model to better suit real-world scenarios.

In the meanwhile, the advances in technology and research, and the increasing availability of data, have created a fertile environment for the growth and rapid spread of Machine Learning across various disciplines, including finance and credit risk.

This work positions itself at the intersection of these developments, aiming to incorporate Machine Learning techniques in the Merton Model, to see if it is possible to enhance the model's performance, without losing its economic meaning. In particular, this work studies the sensitivity of the Merton Model to how the volatility of the stocks is computed, checking if the performance of the model, measured through the area under the Receiver Operating Characteristic (ROC) curve, can be improved by using ML algorithms to compute the volatility of the stocks.

Furthermore, a LSTM model, that uses the same information of the Merton Model, is built to *challenge* the latter and explore if the Machine Learning model can better understand the relationship between the variables and the riskiness of the companies, compared to the Merton Model.

The thesis is organized as follows. The first chapter focuses on reviewing past literature about corporate default models. Three different approaches are taken into

consideration. First, the structural approach, divided into classic models, first-passage models and excursion models. Those models are based on the work of Merton (1974) and Black and Scholes (1973) and consider the companies' stock and debt as derivatives on the companies' asset value. Second, the reduced-form approach. This approach considers that default has an exogenous component and that it is distributed as a Poisson random variable. Following this approach, default cannot be predicted from the market information. Third, the incomplete information approach. It is a hybrid model and has the best features of both structural approach and reduced-form approach. The incomplete information approach shares with the structural approach the definition of default, while shares with reduced-form approach the exogenous component and the fact that default cannot be predicted by market data.

The second chapter aims at describing the methodology and the data used in this work. First, a deeper description of the Merton Model and its equations is conducted, and it is explained how the probability of default is computed. The second section of chapter 2 focuses on the data, how they are gathered and how they are cleaned. Section 3 describes how the Merton Model is applied in practice, the different ways in which the variables (Debt, Market capitalization, Assetvalue, asset returns, volatility of the asset returns, and maturity) are estimated by practitioners, and the way those variables are estimated in this work, with a focus on how volatility of equity is computed (Historical volatility, Exponentially Weighted Moving Average volatility, Deep Learning volatility and Random Features volatility) and used to estimate the volatility of the asset returns. The last section of chapter 2 is dedicated to the LSTM model. First a brief introduction on Artificial Neural Networks is made, and then the architecture of the model used in the thesis follows.

The third and last chapter shows the results.

Follows the conclusion.

Chapter 1

Literature Review

Starting from the last century, a vast amount of literature regarding corporate default prediction models has been published. Academics individuate three different quantitative approaches[1] to analyze credit: the structural approach focuses on the cause of default. It interprets corporate liability as an option on the assets of the company. Some assumptions regarding the dynamics of the company's assets, its capital structure and its debt and shareholders are made, and the company defaults if its assets are considered not sufficient according to some measures. The reduced-form approach instead, focuses on modelling the default rate. The default of a company is not restricted just to the possibility of the firm's assets falling below a certain threshold, but it occurs at an exogenous rate. The incomplete – information approach describes a hybrid model, and it is based on the fact that investors have limited information. In this approach the firm defaults if its value falls below a determined measure (as in the structural approach), however the information that the investors have are lagged or noisy, meaning that they do not know the real value of the company assets, or they do not know the default barrier, or both.

[1] Frank J. Fabozzi: *"The handbook of fixed income securities",* 7th edition.

1. Structural Approach

Kay Giesecke[2] divides the models of the structural approach in three different subsets: classical approach, first-passage approach and excursion approach.

1.1. Classic Models

The bases of the structural approach are rooted in the Black and Scholes (1973)[3] model to price European call and put options. In fact, it considers the corporate liabilities as contingent claims on the assets of the company, while the market value of the latter is considered the main source of uncertainty shaping credit risk.

The classical approach is based on the Merton model (1974)[4]. In his paper Merton considers a company which issues just two classes of claims: equity and debt. Furthermore, Merton assumes i) that the debt is a zero-cupon bond with maturity date T; ii) that the equity is a residual claim; iii) if the payment is not met at maturity, then the bondholders take over the company; iv) the firm cannot issue new senior claims, pay dividends or repurchase its own shares prior to the maturity date.

As a consequence, the value of the company in any point in time t can be written as:

$$V_t = S_t + B_t \tag{1}$$

Where:

- S_t is the value of the equity at time t;

[2] Kay Gieseke: *"Credit risk modeling and valuation: an introduction"*, 2004.
[3] Fisher Black and Miron Scholes: *"The pricing of Options and Corporate Liabilities"*, Journal of political economy, 1973
[4] Robert C. Merton: *"On the pricing of Corporate Debt: The Risk Structure of Interest rate",* Journal of Finance, 1974

- B_t is the value of the bond D discounted at time t.

The classical approach assesses the default of the company only at maturity, the particular path taken by the value V during the time span t to T, where T is the maturity, is not important. What matters in the Merton model is the value of the company at maturity: if at maturity date T, the value of the firm V_T is below D, then the corporation is bankrupt, the bondholders have taken over and the value of the equity is equal to zero. On the other hand, if $V_T > D$, the bondholders are paid, and the value of the equity is equal to $V_T - D$.

Consequently, the value of the debt and equity at maturity can be summarized by the following equations:

$$S(T) = \max\{V_T - D, 0\} \quad (2)$$

$$B(T) = \min\{D, V_T\} = D - \max\{D - V_T, 0\} \quad (3)$$

As it is clear from the above equations, the value of the equity is equivalent to the value of a long European call option with strike price D and maturity T. On the other hand, the value of the bond is equivalent to a portfolio composed of a loan and a short position on a European put with strike price D and maturity T.

Determining the value of equity and credit-risky debt therefore involves pricing European options.

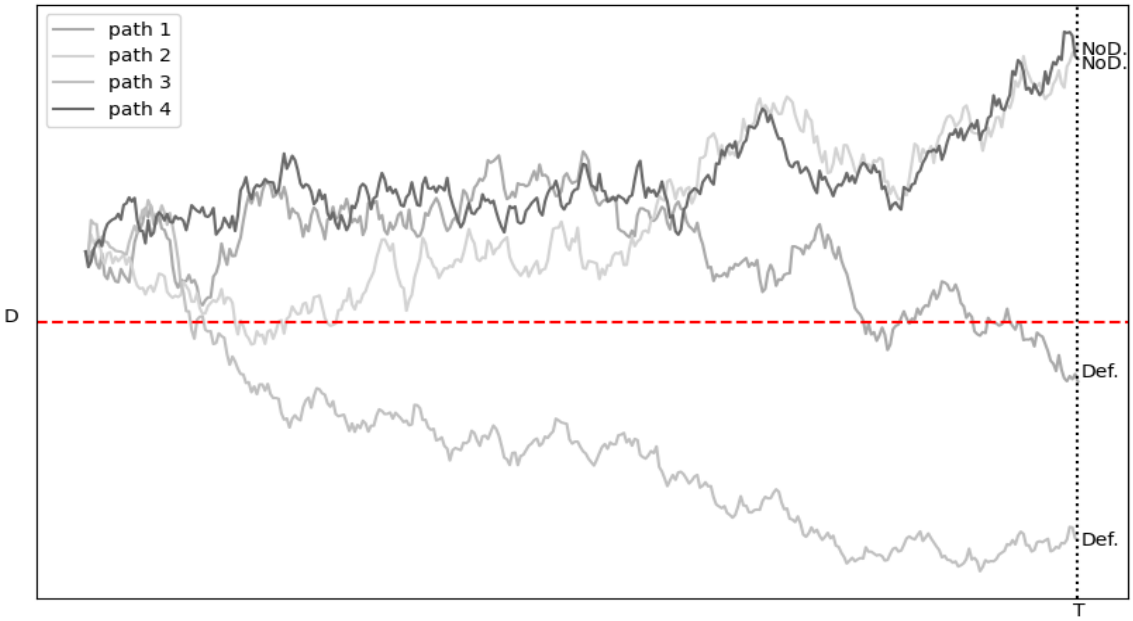

Figure 1: Default in classical approach.

1.2. First-passage Models

The first – passage approach is indeed derived from the work of Black and Cox (1976)[5].

As said above, in the classical approach the path of the value of the company does not matter from time t to maturity T, and it can contract to almost zero without triggering default. However, this is not realistic, since the bondholders usually include safety covenants, contractual provisions which give the debtholders the right to force the reorganization of the company or bankruptcy if its value falls below a certain barrier.

The first-passage approach is a generalization of the classical approach since it allows the firm to default at times different than the debt maturity, that is whenever *V* falls below the barrier level.

[5] Fisher Black and John C. Cox: *"Valuing corporate securities: some effects of bond indenture provisions",* The Journal of finance, 1976.

If we take in consideration $M_t = \min V_s$ for $s \leq t$ as the historical low of the firm values, and consider the default barrier to be equal to H, with H constant and equal to D, then if the value of the company over the time span T – t never falls below the barrier, that is if $M_T > H$, the bondholders receive D and the equity holders will receive $V_T - D$. On the other hand, if over the time span taken into account the value of the firm falls below the barrier, with $M_T < H$, then the bondholders will take over the company, while the shareholders will receive nothing.

The behaviors of both equity and debt, therefore, is similar to that of a European barrier option[6]. More in particular, the equity position is equivalent to a long position in a down-and-out call on firm assets, with a strike price equal to D and barrier value equal to H. The value of the debt is equal to the difference between the value of the firm and the value of the equity.

If, conversely, H, the default barrier, is lower than the face value of the debt D, $H < D$, then we have to take in consideration three different scenarios:

- when $M_T > H$ and $V(T) > D$, then the firm does not default, the debtholders receive D, while the shareholders get the remaining value $V_T - D$;
- when $M_T > H$ and $V_T < D$, then the company defaults at maturity T;
- when $M_T < H$, then the company defaults at time t, before maturity.

It is indeed easy to see how, in the case of the first-passage approach, the probability of default is higher than in the case of the classical approach, since a firm can both default before and at maturity.

[6] John C. Hull and Sankarshan Basu: *"Option, Futures and other derivatives"*, 9th edition.

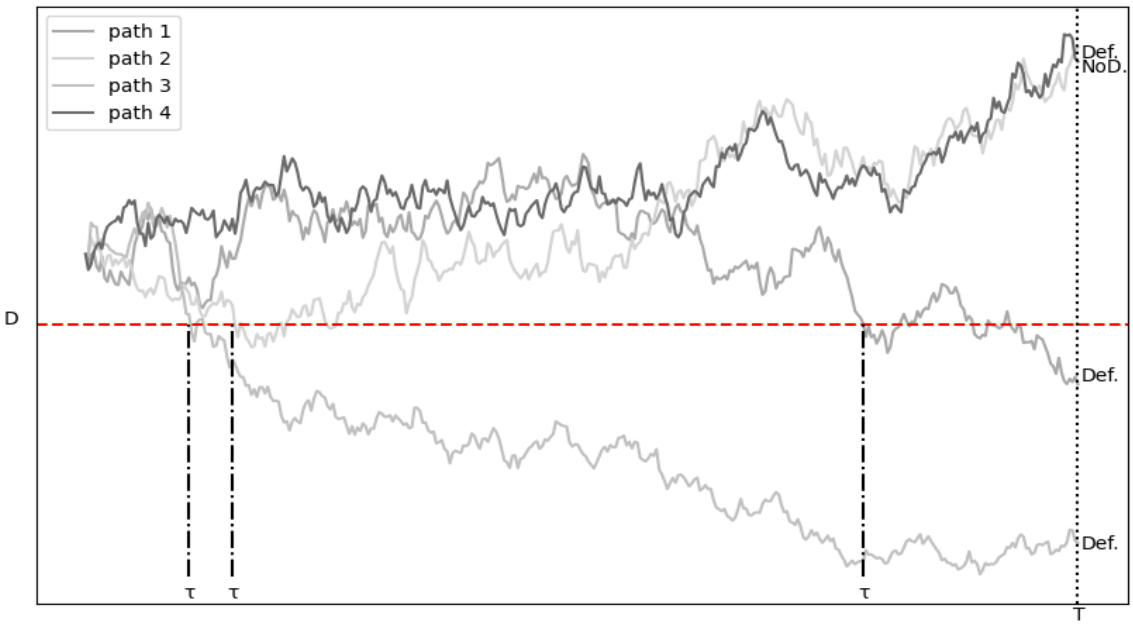

Figure 2: Default in First-passage approach.

1.3. Excursion Models

Differently from the first passage approach, where the debtholders take over the company as soon as the value of the firm falls below the value of the barrier, the excursion approach takes in consideration the time given to the company to reorganize the operations. If the restructuring is successful, then the company does not default and continues to operate. On the other hand, if the restructuring is not successful, then the firm is taken over by the debtholders who liquidate it.

2. Reduced – form Approach

In the reduced – form models, also known as the intensity models, the default of the company happens without warning. Contrary to structural models, default cannot be

monitored using market or fundamental observables, indeed, it has an exogenous component that is independent from all the default free market information.

The reduced – form models start from the idea of describing the default time τ as a first jump time in the Poisson stochastic process. A Poisson process is a type of stochastic process that models the occurrence of events in a finite interval of time. It has two main characteristics that are:

- Memorylessness[7]: it is a characteristic of exponential distribution. Since the times between jumps in a Poisson process are exponential random variables, the time of the next event has the memorylessness property, meaning that it does not depend on any past event.
- Stationary increments: the distribution of the increments depends only on the length of the time interval and not on the position in time.

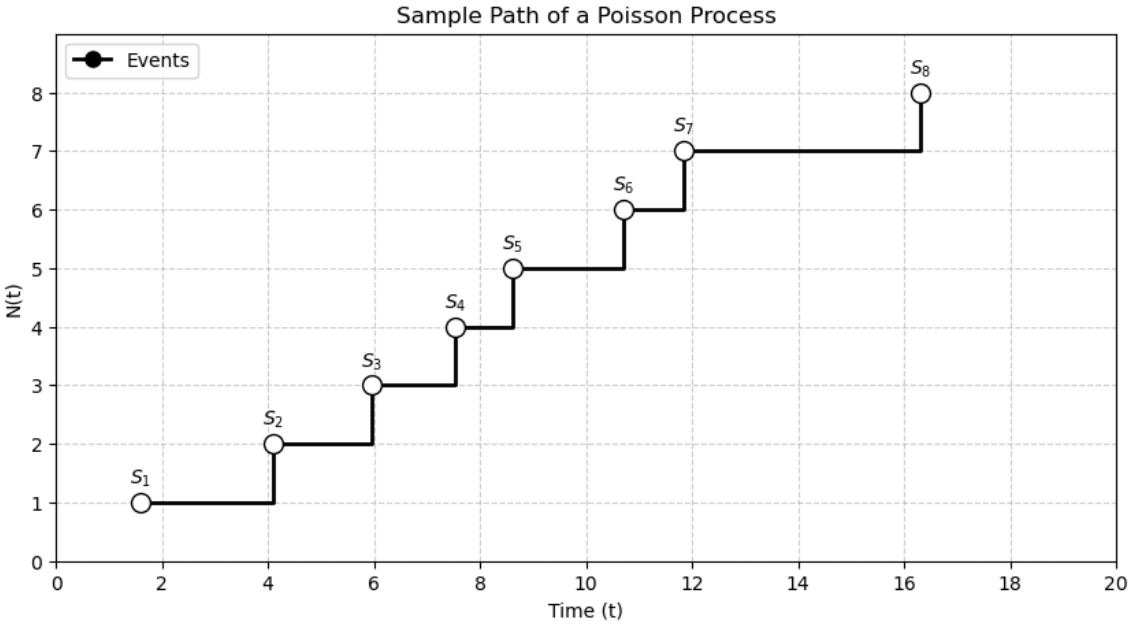

Figure 3: Path of a Poisson process, with S1, S2 ... S_t the arrival times

[7] Steven E. Shreve: *"Stochastic Calculus for Finance II: Continuous – Time model"*.

To calibrate the intensity rate, different types of Poisson processes can be used, from the time homogeneous Poisson processes to the more complex stochastic intensity Poisson processes. Usually,[8] the default intensity rate is described using three different Poisson processes. The simplest one is the time homogeneous Poisson process, that is a *{M_t, t>0} unit – jump increasing, right continuous process with stationary independent increments and null first condition* that is M_0=0. Since the times between jumps are independent and identically distributed as an exponential random variable, and in particular $\gamma\ \tau^1$ is a standard exponential random variable, we have that:

$$Q\{\tau^1 > t\} = e^{-\gamma t} \tag{4}$$

with Q the neutral probability measure.

On the other hand, the probability that the company defaults in the arbitrary small fraction of time *dt*, given that it has not defaulted yet, is equal to γdt.

The second type of Poisson process used in the intensity models is the inhomogeneous Poisson process, and it is particularly useful to extract implied default probabilities from CDS (Credit Default Swaps). An inhomogeneous Poisson Process is a stochastic process with deterministic time – varying intensity rate γ(t). Thus, unlike in the homogeneous Poisson process, where the intensity rate was constant, here the intensity rate changes with time.

If we consider N_t the time inhomogeneous Poisson process, and M_t as the standard Poisson process with intensity rate equal to one, then we have that:

$$N_t = M_{\Gamma(t)} \tag{5}$$

with

$$\Gamma(t) = \int_0^t \gamma_u du \tag{6}$$

[8] Damiano Brigo and Fabio Mercurio: *"Interest rate Models – Theory and Practice"*, and Kay Gieseke: *"Credit Risk modeling and valuation: an introduction"*.

This basically means that a time inhomogeneous Poisson process can be viewed as a standard Poisson process with changed timeline. For a time – inhomogeneous Poisson process the probability of survival is given by the following formula:

$$Q(\tau > t) = exp(-\Gamma(t)) = e^{-\int_0^t \gamma(u)du} \quad (7)$$

The last type of Poisson process usually used to in reduced – form models, or intensity models, is the Cox process, also called the doubly stochastic Poisson process, so called due to the fact that there two different sources of randomness since the intensity is both time – varying and stochastic.

In a doubly stochastic Poisson process the probability of survival is given by the formula:

As can be seen from the formulas 1, 4, 5, survival probabilities and discount factors have the same proprieties, with the intensity rate substituted to the risk – free interest rate. The analogy survival probabilities/discount factor is due to the exponential distribution of jump times, and it is particularly useful since it allows to use much of the interest rates theory in credit default modeling.

3. Incomplete – information Approach

The incomplete-information models are hybrid models that take the best features of the two approaches described above, the structural approach and the reduced-form approach. Incomplete-information models share with structural models the definition of default. The classical definition of default is expressed in Black and Scholes (1973), and Merton (1974), and states that a firm defaults if at debt maturity the value of the company V_T is below the face value of the debt D. Following the definition of default given by the first passage approach, Black and Cox (1976), a company defaults if its value V_t falls below a certain threshold H.

On the other hand, incomplete – information models share with the reduced – form models the inaccessibility of τ, the default time. Indeed, in structural – models, if the assets cannot jump, the default time is predictable. Consequently, model credit spreads go to zero as the debt approaches maturity, meaning that there is no short – term credit risk, and the short term spread between corporate bonds and Treasuries is zero. This result conflicts with reality, where, instead, credit spreads are significantly positive. In the intensity models' default is exogenous and τ is not predictable. This uncertainty causes the investors to ask a premium over the Treasuries yields, since they are facing short – term credit risk.

Incomplete – information models were introduced by Duffie & Lando (2001)[9], and Giesecke (2001)[10].

In Giesecke 2001 the role of the time – dependent revelation of information to investors has been studied. The author models the evolution of the information available to investors through a filtration. A default model is infact described by two elements that are: the default definition and the model filtration $F \subseteq G$ that describes the information that are available to investors at any time. The basic idea behind the incomplete information model developed by Giesecke is that, taking in consideration the default definition given in the first passage approach, the model inputs obtained from corporate Financial Statements, such as company assets and liabilities, may be noisy or lagged.

In a similar vein to Giesecke (2001), Duffie & Lando (2001) also deal with the problem of incomplete information. However, the two authors, in a reduced – form models fashion, give the conditions for the existence of an intensity and compute it, taking into consideration that the incompleteness of the information comes from the assets side. However, as pointed out by Giesecke, there are models with inaccessible time to default but with no intensity, for example first passage models with barrier that is not observable. Thus, in Giesecke (2001) the author provides a generalized reduced – form representation that solves all the incomplete – information models using the *trend* of a default model.

[9] Darrell Duffie and David Lando: *"Term structures of credit spreads with incomplete accounting Information"*, 2001.
[10] Kay Giesecke: *"Default and Information"*, 2001.

Chapter 2

Methodology and Data

The aim of this thesis is to study if it is possible to increase the performance of the Merton Model using Machine Learning techniques, in particular Deep Neural Network. The thesis first studies the sensitivity of the model to how the volatility of the stocks is computed, checking if the performance of the model, measured through the area under the Receiver Operating Characteristic (ROC) curve, can be improved by using ML algorithms to compute the volatility of the stocks. Secondly the thesis checks if a LSTM (Long – Short Term Memory) algorithm can better classify the riskiest companies given the same variables used in the Merton Model.

1. Application of the Merton Model

In this thesis the simplest of the structural models has been used, that is the Merton Model. As explained in chapter 1, section 1.1, the MM implies that the firm has issued just equity and a zero cupon bond, and that it defaults only at maturity if the value of its assets is lower than the value of its debt. Furthermore, the model assumes that the asset value of the firm follows a Geometric Brownian Motion (GBM):

$$dV_a = \mu_a V_a dt + \sigma_a V_a dW \tag{8}$$

The model then applies the Black and Scholes formula to compute the value of the firm's equity as a call option on the company's asset value.

$$E = V_a N(d1) - Ke^{-rT} N(d2) \tag{9}$$

The probability of default is given by the equation:

$$PD = N(-DD) \tag{10}$$

With DD, the distance to default, given by the formula:

$$DD = \frac{\ln\left(\frac{V_a}{D}\right) + [\mu_a - 0.5\sigma_a^2]T}{\sigma_a\sqrt{T}} \tag{11}$$

Neither the assets value of the company, its volatility nor the expected return of the assets can be observed, thus the three variables need to be estimated.

In literature, different methods have been used to compute them. The first method, and also the approach used in this thesis, is the one proposed by Jones et all. (1984)[11], where the company's assets value and its volatility are computed solving a system of two equations. From (6) we know that the value of the equity is equal to the value of a call option with underlying V_a, strike price D, volatility σ_a, interest rate μ_a and maturity T, thus we can write:

$$\mathrm{E} = \mathrm{c}(V_a, D, \sigma_a, T, \mu_a) \tag{12}$$

[11] Philip E. Jones, Scott P. Manson and Eric Rosenfeld: *"Contingent Claims Analysis of Corporate Capital Structures: an empirical Analysis",* 1984.

Using Ito's lemma for equation (9) then we have:

$$dE = \frac{\partial c}{\partial t} dt + \frac{\partial c}{\partial V_a} dV_a + \frac{1}{2} \frac{\partial^2 c}{\partial V_a^2} dV_a^2 \tag{13}$$

Substituting equation (8) in equation (13) we have the following:

$$dE = \frac{\partial c}{\partial t} dt + \frac{\partial c}{\partial V_a} \mu_a V_a dt + \frac{\partial c}{\partial V_a} \sigma_a V_a dW + \frac{1}{2} \frac{\partial^2 c}{\partial V_a^2} \sigma_a^2 V_a^2 dt \tag{14}$$

Since the only non-deterministic term of equation (14) is $\frac{\partial c}{\partial V_a} \sigma_a V_a dW$, we can write:

$$E\sigma_e = \frac{\partial c}{\partial V_a} \sigma_a V_a \tag{15}$$

Thus, solving simultaneously equation (15) and equation (9) we get the company's assets value and its volatility, while the expected return of assets needs to be estimated separately.

In Crosbie and Bohn (2003)[12], the authors point out that the relationship described in equation (15) holds only instantaneously, and that the leverage moves too much for the equation to give a reasonable result. Thus, they implement an iterative process to compute the three unobservable variables. As explained in Bharath and Shumway (2008)[13], the iterative process consists in giving an initial guess for the value of the assets' volatility and substituting it in equation (9) to compute the assets value of the company for every time step. From V_a is then computed the implied log returns of the assets (μ_a), and from them a new estimate of σ_a is derived. The process is carried out until the estimates for σ_a converge, that is until the difference between σ_a^x and σ_a^{x-1} is negligible.

[12] Peter Crosbie and Jeff Bohn: *"Modeling Default Risk"*, 2003.

[13] Sreedhar Bharath and Tyler Shumway: *"Forecasting Default with the Merton Distance to Default Model",* 2008.

A third approach for estimating the assets value, volatility of the assets and expected returns of the assets is the one proposed by Duan (1994)[14], who suggested to use a Maximum Likelihood approach to compute simultaneously the three variables.

2. Data

To evaluate the performance of the models, information about default events of non-financial companies between 2000 and 2023 is collected from the Compustat database. Financial companies (from sic 6000 to 6799) are excluded from the computations due to their particular Financial Statements, characterized by high leverage and different regulations. Using a methodology similar to the one used by Forssbæck and Vilhelmsson (2017)[15], companies are considered defaulted if they were delisted (DLRSN in Compustat database) due to Chapter 11 (code 02 in Compustat), and Chapter 7 (code 03 in Compustat).

Regarding the variables needed in the Merton Model, current liabilities (DLC) is considered as short – term debt, while, for long – term debt the variable DLTT is taken into consideration. Similarly, the total number of shares outstanding is given by the variable CSHO. For the daily prices, the adjusted close price is computed using the following formulas:

$$adj\ close = \frac{price}{adj\ factor} \tag{16}$$

Finally, for the risk-free rate, the DGS1 series from the Federal Reserve is used, that is the 1-year treasury bill rate.

[14] Jin – Chuan Duan: *"Maximum Likelihood estimation using price data of the derivative contract"*, 1994.

[15] J. Forssbæck and A. Vilhelmsson: *"Predicting Default – Merton vs Leland"*, 2017.

For all the companies, 5 years of data prior to the defaulting event, considered as the day of delisting, are taken into account. Companies with missing accounting or equity data are discarded.

In this way, the database of defaulting companies is composed of sixty-nine firms. This group of defaulting companies is then compared to a controlled group of sixty-nine non-defaulting companies that operated in the same period[16].

The short-term debt, long term-debt and shares outstanding data are annual information, while the prices are daily. In order to overcome this problem a cubic spline interpolation was used, so to have a function that smoothly incorporates the changes that yearly data face between one year and the other.

Weekly returns are computed as the sum of the daily log returns using the following formula:

$$r_t = \sum_{i}^{n} r_{i,t} \tag{17}$$

With $r_{i,t}$ the i-th daily return of week t, computed as the log difference of the daily prices, and n the number of trading days during week t. Similarly, the annualized realized volatility at week t is computed as:

$$rv_t = \sqrt{\frac{252}{n} \sum_{i}^{n} r_{i,t}^2} \tag{18}$$

Similarly to Merxe Tuleda and Garry Young (2003)[17], for each company, and for each estimate of volatility used, the PDs (probabilities of default) are computed on a weekly

[16] For a company defaulted in 2001 a non-defaulting firm which operated in the year 1995 – 2000 in the same industry was selected.

[17] Merxe Tuled and Garry Young: *"A Merton-model approach to assessing the default risk of UK public companies"*, 2003.

basis using a look back window of four years. Thus equations (9) and (15) are simultaneously solved to get the assets value and the volatility of assets, that are then used to compute the distance to default and the PD using respectively equation (11) and (10). For each firm, the PDs are computed the 52 weeks preceding the *1 – year to default date*[18] with every estimate of volatility (historical, EWMA, Deep Neural Networks and random features) and then the average of this PDs is used as measure of 1 – year ahead default probability.

	Variables	count	mean	std	min	max
H vol	Historical volatility	69	77.51%	39.23%	27.41%	212.95%
EWMA vol	EWMA volatility	69	107.78%	58.03%	15.29%	246.12%
DNN vol	Deep Neural Network volatility	69	87.55%	42.86%	20.94%	235.21%
RF vol	Random Features volatility	69	85.70%	43.80%	26.99%	219.21%
MKT cap	Market capitalization	69	62.118	96.645	0.484	468.602
D	Debt	69	167.100	322.803	-	1,681
rf	risk - free rate	69	1.86%	1.62%	0.10%	6.11%

Table 1: defaulting companies' descriptive statistics

The descriptive statistics of defaulting companies are shown in the table above, while for non – defaulting companies are presented in the table below.

	Variables	count	mean	std	min	max
H vol	Historical volatility	69	63.66%	66.91%	14.29%	321.98%
EWMA vol	EWMA volatility	69	50.21%	35.48%	18.13%	201.74%
DNN vol	Deep Neural Network volatility	69	62.26%	62.85%	15.01%	320.28%
RF vol	Random Features volatility	69	61.44%	61.27%	14.33%	304.40%
MKT cap	Market capitalization	69	20,275	43,804	0.19	262,825
D	Debt	69	1,363	2,001	-	7,742
rf	risk - free rate	69	1.87%	1.62%	0.10%	6.11%

Table 2: non - defaulting companies' descriptive statistics

[18] If a company defaulted May 2001, the 1 – year to default date is considered 31 December 2000.

3. Variables Estimation

This section contains the refinements that are needed to apply the Merton Model in practice.

3.1. Debt, Expected Returns and Maturity

One of the biggest issues when dealing with the Merton Model in practice is the amount of debt that is relevant to trigger a potential default and its maturity. One of the assumptions of the MM is that the company issues a zero cupon bond with a specified maturity T. However, in practice usually companies have different types of debt with different maturities. Following the literature (Afik et all.[19], Bharath and Shumway, Crosbie and Bohn) the maturity is assumed to be one year. On the other hand, the amount of debt that could trigger default during a one-year time step is a not trivial question. As pointed out by Afik et all., taking total liabilities of the company would be inappropriate when not all of them are due in one year, since the firm could still be solvent even if its assets value falls below the total debt. At the same time considering just short-term liabilities would not be sufficient, since often debtholders put up covenants that force the company to serve them first if the firm's value deteriorates. Crosbie and Bohn highlighted the fact that the value of the assets at which usually companies default lies somewhere between the short – term liabilities and the long – term liabilities. The value of D is therefore given by:

$$D = ST + k \cdot LT \tag{19}$$

Where *k* is a constant between zero and one. In this thesis *k* is chosen to be equal to 0.5.

[19] Zvika Afik, Ohad Arad, Koresh Galil: *"Using Merton model for default prediction: An empirical assessment of selected alternatives"*, 2015.

Since the two equations system has been chosen to derive the assets value and volatility of the assets, the expected returns of the assets needs to be estimated separately. In Campbell et all., to limit the noise, a constant market premium has been chosen, and the expected return of firm's assets is given by:

$$\mu_a = r_f + 6\% \tag{20}$$

In their work Afik et all. take into consideration several alternatives. The first two alternatives use a CAPM approach, the first one with a constant market premium, while the second one with a variable market premium, estimated for the historical excess returns of the index for the previous year. The return of the assets is therefore given by:

$$\mu_a = r_f + \beta_a(r_M - r_f) \tag{21}$$

where β_a is the asset beta. Other alternatives assume simply that the expected assets return is given by the historical equity returns of the previous year, or that it is equal to the maximum between the historical equity returns of the previous year and the risk – free rate, since taking into consideration just the former may bring up some problems due to the fact that they can result negative. In this work, the expected assets return is assumed to be equal to the risk free – rate, thus:

$$\mu_a = r_f \tag{22}$$

3.2. Equity Volatility

The estimation of equity volatility and the sensitivity of the model to changes in how it is computed is the core of this thesis.

In most past literature, to estimate equity volatility is used the historical volatility. However, as pointed out by Afik et all., the use of historical volatility has some drawbacks, the primary one being the fact that it is backward looking estimate, while a forward looking would be preferable.

For this reason, different models to compute future equity volatility for the 52 weeks preceding the *1 – year to default* date have been implemented here and their performances in the MM is compared with that of simple historical volatility. In particular, in addition to historical volatility, three different models to compute σ_e have been implemented here: the Exponential Weighted Moving Average (EWMA) model, the random features model and the Deep Neural Network model.

The EWMA model was first suggested by the RiskMetrics – Technical Document (1996)[20]. The simple recursive formula is given by the following:

$$\sigma_{t+1}^2 = (1-\lambda) r_t^2 + \lambda \sigma_t^2 \tag{23}$$

Where $1 - \lambda$ is the decay parameter, indicating the speed at which the weights decrease, while λ is the persistence parameter, that is how sticky the variance is. Given those definitions, it is easy to understand which characteristics of volatility the EWMA tries to replicate. In general, market volatility has four different properties that a correct volatility measure should replicate. These properties are:

- heteroscedasticity, meaning that volatility changes over time;
- time clustering, meaning that high volatility periods tend to cluster together;
- long – run mean reversion, meaning that volatility tends to come back to its long – run mean;
- asymmetry, that means that market volatility tends to surge more when returns decrease than when they increase.

[20] J. Longerstaey, M. Spencer: *"RiskMetricsTM—Technical Document"*, 1996.

The EWMA model, through its parameters, can replicate two characteristics that are time varying and time clustering.

There are several ways to choose lambda for the EWMA model. In RiskMetrics (1996) the authors suggest using a persistence parameter of 0.94 for daily data and 0.97 for monthly data. Other than that, it is possible to compute λ by using Maximum Likelihood, Minimum Root of the mean squares of forecast errors, matching the desired "half-life" or choosing the desired "effective history".

In this thesis the latter method has been chosen to pick the right lambda. The "effective history" determines how much historical data the model considers relevant to compute future volatility, thus the lower the "effective history" the higher are the weights on recent observations. In particular, in this thesis, the "effective history" (N) is chosen to be equal to seven weeks. The formula to compute λ from the "effective history" is given by the following:

$$\lambda = \frac{N-1}{N+1} \tag{24}$$

Random Features is a Machine Learning technique usually used to approximate a kernel method. Often, the relation between a selected variable and its features is a non – linear relationship, and the value of the selected variable cannot be forecasted using common linear techniques, such as linear regression, ridge regression or lasso regression. Therefore, the main idea behind kernel methods is to map the features of the data into a new feature space where the aforementioned non – linear relationships can be represented in a linear form, and thus detected by simple linear regressions.

$$\phi: \mathbb{R}^d \longrightarrow \mathcal{F} \tag{25}$$

The problem with the kernel method is that it can be computationally expensive, since the dimension of $\mathcal{F}$, the new feature space, can be much higher, or even infinitive, than the original input space.

In their work Rahimi and Recht (2007)[21], in fact explain that methods that use the Kernel matrix, despite quite attractive since they can approximate any function, perform poorly in terms of efficiency if compared to linear models, such as regularized regressions (Ridge regression and Lasso regression). Thus, the authors propose a way to exploit the advantages of both methods by mapping the data into relatively low – dimensional randomized features space, and then use regularized regressions.

In this thesis, to predict volatility, one hundred thousand random features are generated. These features are created by first projecting the data into a lower – dimensional space using random projections drawn from a normal distribution. Then non – linear transformations are applied to the projections. In this case sinusoidal functions are used for non – linear transformations. In particular, fifty thousand random projections are mapped using the *sin* function, and the remaining fifty thousand are mapped using the *cosin* function. Finally, these random features are used as inputs to a ridge regression model to predict the target variable.

The third method used in this work to forecast volatility is a Deep Leaning model[22]. Deep Learning is a branch of Machine Learning that puts the emphasis on learning relations from data. Usually, these relations are learned through artificial neural networks, that are mathematical models inspired by biological neurons and by how animal brains work to perform complex tasks. The first ANNs (Artificial Nural Networks) model was introduced in 1943 by Warren McCulloch and Walter Pits[23]. During the recent years, due to the availability of large dataset to train the models, much larger computational power for the CPUs, and also improvement of training algorithms (introduction of back propagation algorithms), there has been a renewed enthusiasm toward ANNs. Deep learning algorithm can perform many complex tasks by mapping the inputs to the targets thanks to what it learned by observing many examples of inputs and targets. The learning process is carried

[21] Ali Rahimi and Ben Recht: *"Random Features for large – scale Kernel Machine"*, 2007.

[22] More on Machine Learning can be found in T. Hastie, R. Tibshirani and J. Fridman: *"Element of Statistical Learning"* and in A. Géron: *"Hands on Machine Learning with Scikit – Learn, Keras and TensorFlow"*. More focused on Deep Learning is instead F. Chollet: *"Deep Learning with Python"*.

[23] W. McCulloch and W. Pits: *"A Logical Calculus of Ideas Immanent in Nervous Activity"*, 1943.

out thanks to a sequence of data transformations each happening in different layers of the model. A Deep Learning model is infact composed of many layers:

- the first layer is called the input layer, and its dimension depends on the dimension of the data;
- the hidden layers are where the transformations occur through the activation functions. Nonlinear activation functions are used so that complex relationships between the data can be found, and the model adjust the weights so to minimize the objective function (such as MSE for regression or Cross Entropy for classification problems). The number of hidden layers determines the depth of the model, and it can range from a few layers to hundreds or thousands of them. The width of each layer depends on the number of neurons on each layer.
- Finally, the last layer is called the output layer, and it contains the forecasted values. Its size and whether an activation function is applied depend on the type of task the model is performing. For regression tasks, activation functions are typically not used (or a linear activation is applied) to produce continuous values. For classification tasks, the sigmoid function is often applied for binary classification to constrain the output between 0 and 1, while the *softmax* function is used for multi-class classification to produce probabilities for each class.

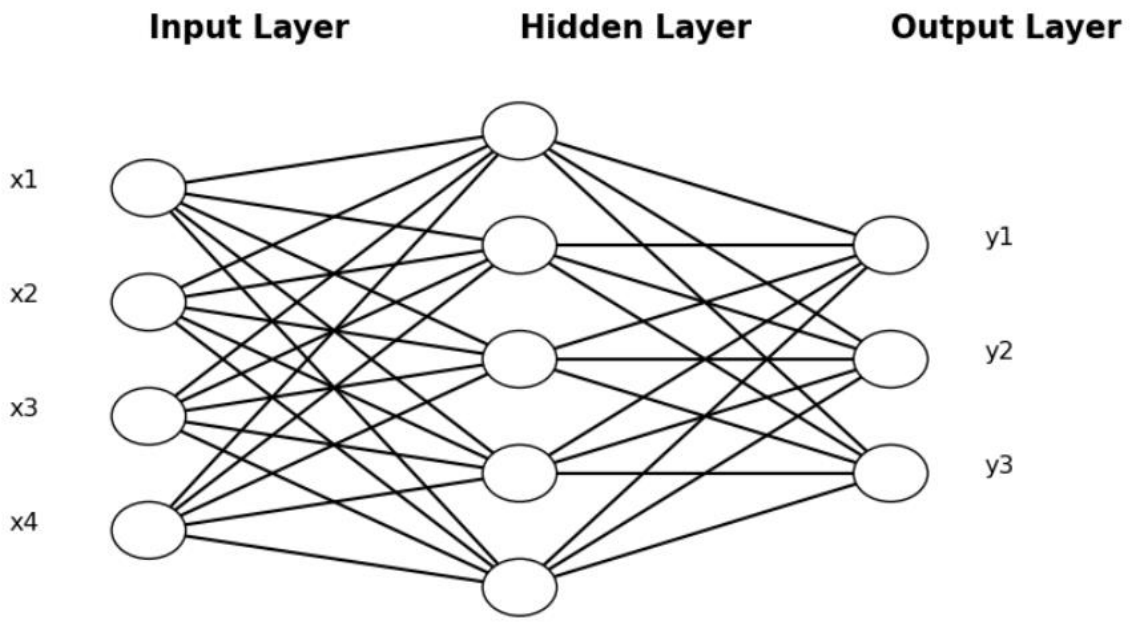

Figure 4: Vanilla ANNs structure

The figure above summarizes the structure of a vanilla (just one hidden layer) Neural Networks. The hidden layer neurons represent a linear combination of the input features, and followed by an activation function that introduces nonlinearity:

$$Z_m = \sigma(\alpha_0 + \alpha_m^T X_m) \tag{26}$$

with σ the activation function. Similarly, the neurons in the output layer are a linear combination of the Z_m from the hidden layer, processed by the output function, g:

$$Y_k(X) = g(\beta_{0k} + \beta_k^T Z) \tag{27}$$

As already mentioned before, in case of regression the output function is usually chosen as the identity function, so that nonfinal transformation is applied.

The figure below shows the output of three among the most common activation functions (Sigmoid, ReLu, and Tanh).

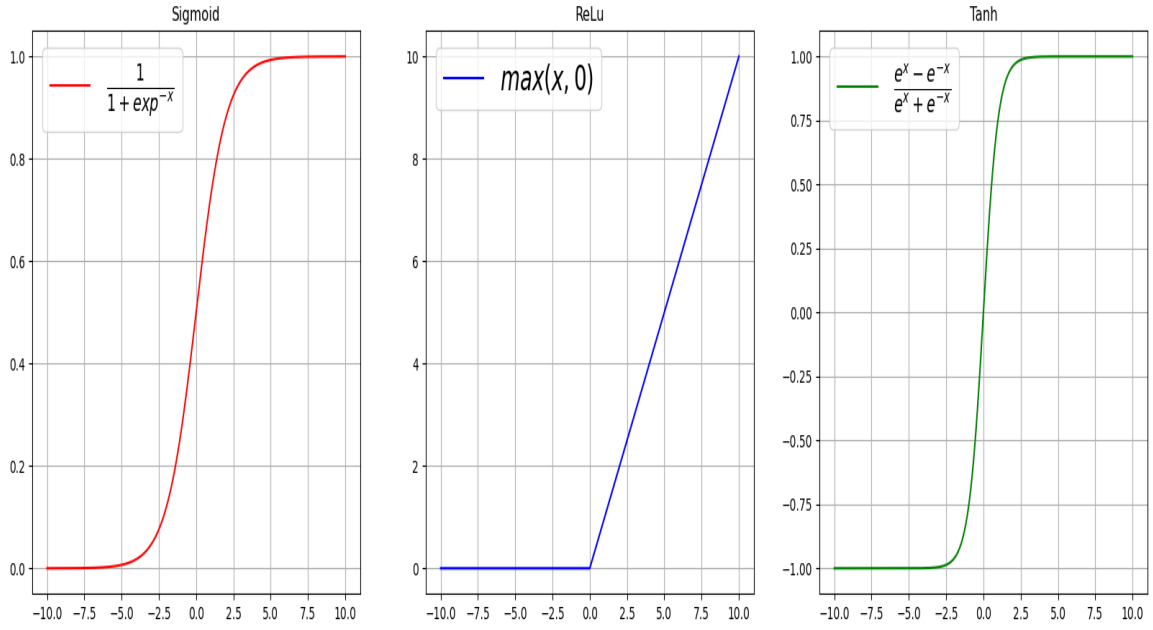

Figure 5: plot of the output of Sigmoid, ReLu and Tanh functions

For this work, past returns and past volatilities are chosen as input features to forecast future weekly volatility for each company. The model is composed of 2 hidden layers and each layer has a width of 64 neurons. The ReLu function ($max\{x, 0\}$) is chosen as activation function, and the model is trained over 200 epochs with and early stopping rule to limit overfitting, and a learning rate of 0.001, that is the default learning rate in the Adam optimizer. Fine tuning of the model hyperparameters is not carried out in this work for two main reasons:

- simplicity and computational constraints, infact, fine tuning each hyperparameter of the model for the 138 companies used in this work would increase the running time and computational cost exponentially;
- risk of overfitting: fine tuning each hyperparameter for each company might cause the model to adapt to specific company data, increasing overfitting and limiting generalization across different companies.

4. Long Short – Term Memory Model

Traditional models, such as the Merton Model in this work, may not detect complicated patterns and relationships between the variables, dependencies that instead can be captured by Machine Learning models. For this reason, in addition to a Merton Model, is also built a LSTM (Long Short – Term Memory) model, fed with the same variables of the MM.

LSTM algorithms are types of Recurrent Neural Networks (RNNs). One of the characteristics of the Deep Learning model described above is that it does not have memory, each input is processed independently, and eventually, the sequential order of the features is ignored. This is in contrast to what happens with biological intelligence, where information is processed incrementally, meaning that past information is updated with new information and inputs. RNNs are built on the same logic. The activations do not flow only in one direction, from input layer to out layers, but the output from the previous examples is also taken into consideration, and in this way RNNs manage to have memory of what happened in the past. This feature makes them particularly suitable for time series analysis.

However, the simple Recurrent Neural Networks face one problem: it suffers from unstable gradient, meaning that it can either explode or vanishing. Hochreiter and Schmidhuber studied the theoretical reasons behind this problem, and they came up with a solution in 1997[24], introducing the Long Short Memory cell. The difference between a LSTM cell and a cell of a simple RNNs is that the state of the former, that is a function of some inputs and the state at the previous time step, is split into two vectors: $h_{(t)}$, that is the short-term state, and $c_{(t)}$, that is the long-term state. In a LSTM cell, the long-term state goes through 2 different gates: the *forget gate*, where it drops some information (some memories), and the *input gate*, where instead other information is added into the vector, that is the starting point for the long – term state at the next step. In addition, after the *input gate*, a copy of the long – term state is processed by a tanh function and passed through another gate, called the *output gate* that produces the short – term state.

[24] S. Hochreiter and J. Schmidhuber: "*Long short-term memory"*, 1997.

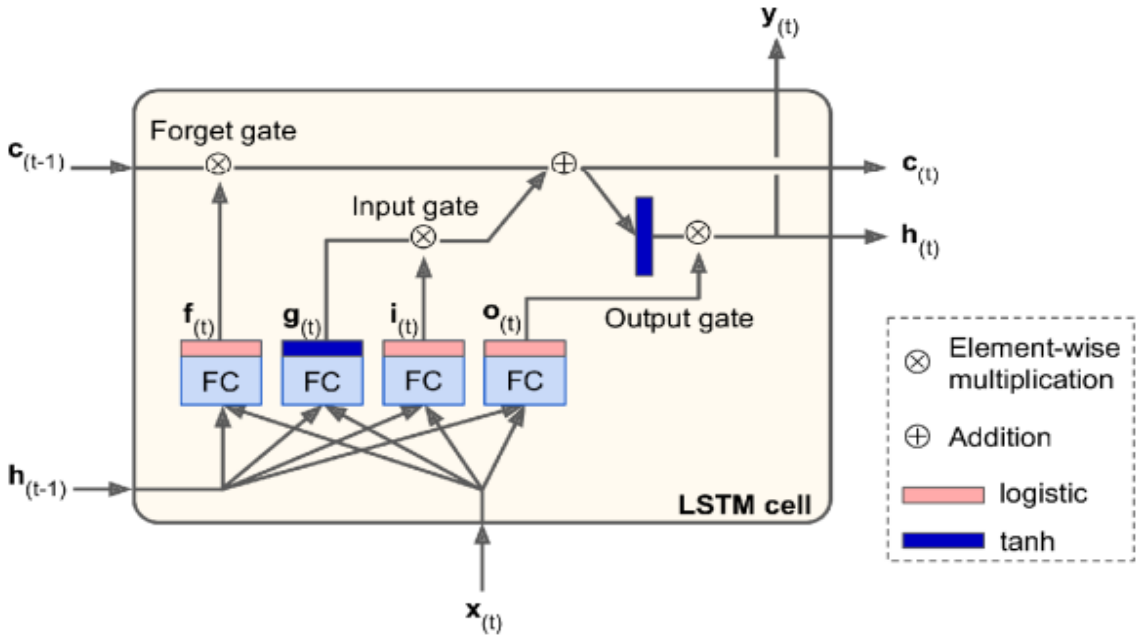

Figure 6: LSTM cell diagram[25]

The figure above shows the typical structure of an LSTM cell. The input vector (x_t) and the short-term state vector (h_{t-1}) are processed by four different layers. The main layer is the one that outputs $\tilde{c}_t$, because it is where the current input and the short-term state are analyzed. The other three layers are used for the *gates*. Since they are sigmoid functions, their output ranges between 0 and 1, thus they decide, respectively, which information should be forgotten (f_t function), which information coming from the main layer should be added to the long-term state c (i_t function), and which information should be sent to the short-term state (o_t function).

In this work a LSTM model is compared to the MM to see if, using the same variables, it can better understand which are the riskiest companies, using a look back window of 52 weeks. Since the number of defaulting and non-defaulting companies is low (138 firms in total), bootstrapping is used to increase the size of the sample to 1000 companies. The model is composed of an input layer with size six (equal to the number of features used), two hidden layers with width equal to four, and one output layer. Differently

[25] The figure was taken from Aurélien Géron: *"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow"*.

from the models used to forecast volatility, the LSTM model will not perform a regression problem, but instead a classification problem. For this reason, a Binary Cross – Entropy (BCE) is used as a objective function, instead of a Mean Squared Error. Similarly to the DNNs model used to predict volatility, the LSTM model is trained over 200 epochs, with a learning rate of 0.001. Furthermore, to limit the impact of randomness and obtain more reliable and stable predictions, the final results are given by the average of the output of ten different seeds. A seed is in fact a fixed starting point of a pseudorandom number generator, and it ensures the reproducibility of an experiment. In the context of LSTM model, a seed is responsible for some random processes, such as weight initialization and data shuffling. Thus, averaging over 10 different seeds limits the sensitivity of the output of the model to outlier (too good or too bad) initializations or data shuffling.

Chapter 3

Results

The aim of this chapter is to show the results of the analysis done in this work. First was evaluated the sensitivity of the Merton Model to changes in how equity volatility is computed. As explained by Afik et all., valuation of the model can be carried out by two different methods. The first one being the *Model's Power*, that is how good is the model at ranking the observations, the second method being the *Model's Calibration*, that is the goodness of the predictions made and how they fit reality. This thesis focuses on the first method; thus, model A would be better than model B if it can better rank the companies based on their riskiness. For this purpose, the ROC curve and AUC are used, since they can evaluate a model without the need for default thresholds. The idea is to do a *horse race* between the four models, if the curve of one model is always above the curves of the other models, then the former is a superior model. If the curves cross, then the Area Under the Curve can be used to evaluate them. Furthermore, the non-parametric test of DeLong[26] is used to check if the AUCs are significantly different.

Secondly, using the same method, is evaluated the performance of the LSTM model against the performance of the Merton Model.

[26] E. R. DeLong, D. M. DeLong and D. L. Clarke-Pearson: *"Comparing the Areas Under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach"*, 1988.

1. Data Visualization

In chapter 2 (Methodology and Data), section 2 (Data), a quick analysis of the variables used was already conducted, and it was possible to gain a first understanding of the dataset and of the major differences between defaulting and non-defaulting companies.

The aim of the section is to go a bit further in the analysis and gain some other insights into the data.

First, since Market capitalization and Debt value plausibly have an important role in whether a company defaults or not, it seems reasonable to plot their relation and how the two variables behave in the two circumstances (defaulting or not defaulting). In the figure below the *scatter plot* between the average Market capitalization and the average Debt value for the two groups is plotted.

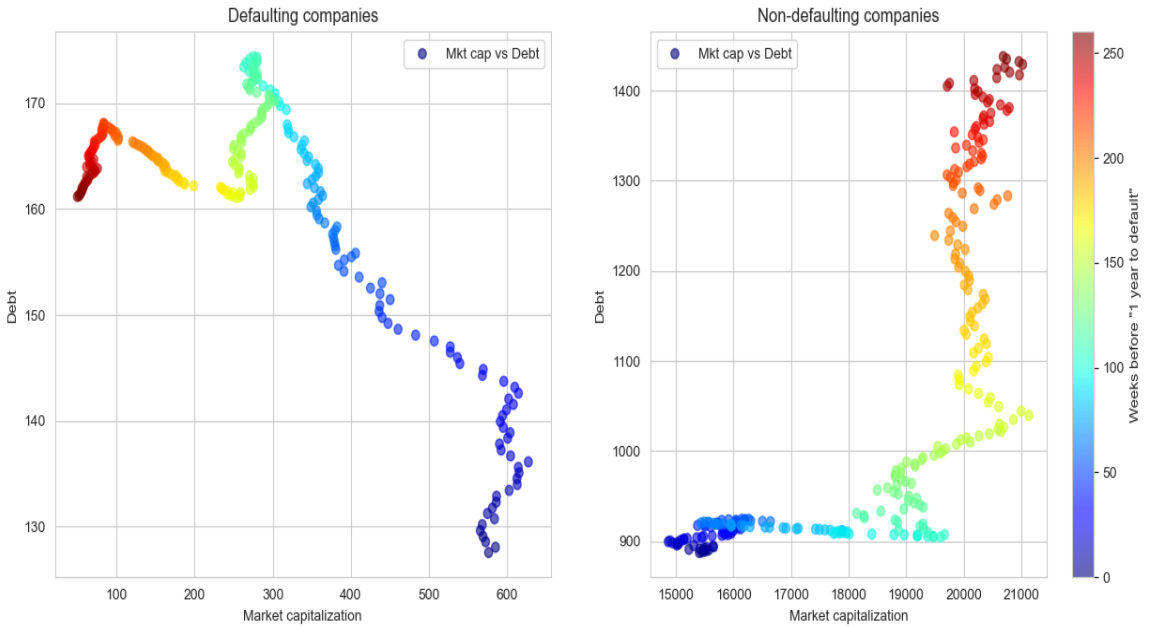

Figure 7: Market Capitalization vs Debt Value - Defaulting vs non-Defaulting companies

From the chart is clear the different behaviors of the two groups: for defaulting companies, as they approach the *1-year to default date*, the market capitalization shrinks, while the debt increases; for non-defaulting companies, instead, both market capitalization and debt follow an upward trend as time passes.

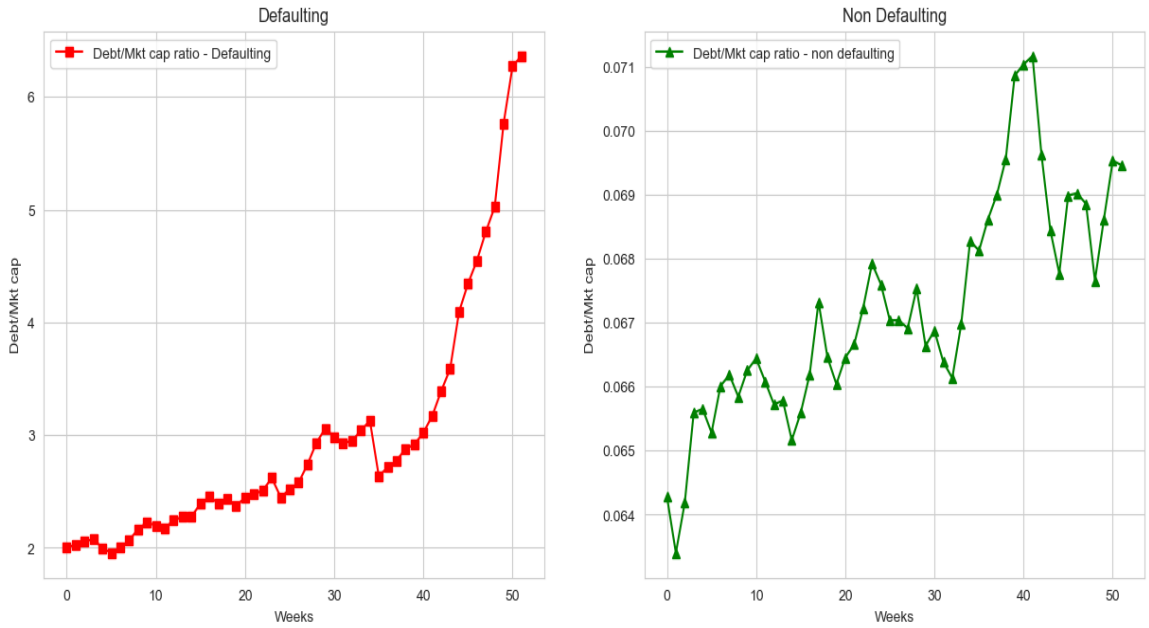

Figure 8: Debt/Mkt cap ratio, 52 weeks before "1-year to default date" - Default vs non-defaulting

Figure 8 shows the behavior of the Debt over Market capitalization ratio of the two different groups. It can be noted that for non-defaulting companies the ratio remains stable during the 52 weeks preceding the *1-year to default date*, while the ratio for defaulting companies shows a threefold increase during the same period. This confirms that both Debt and Market capitalization have an important role for predicting company's default, and thus their ratio can reasonably be used as a benchmark (*simple model*) for both MMs and LSTM model to see if they bring new information and improve on the *simple model*.

2. Merton Models Evaluation

Once that the four different volatilities have been computed, as well as the implied probabilities of default for each estimate of volatility, using the system of two equations, equation (6) and equation (12), and the Distance to Default, equation (7), the work proceeds

by comparing the ROC curves and AUCs of the models against each other and against the *simple model*.

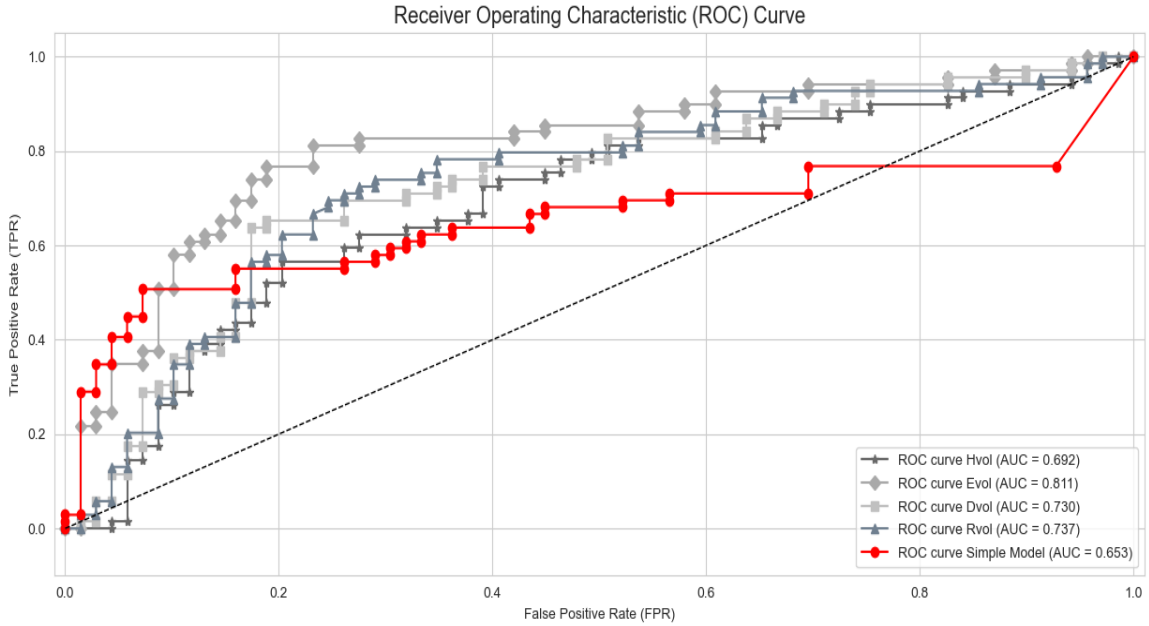

Figure 9: ROC and AUC for MMs against the simple model

Figure 9 shows the ROCs of the Merton Model for different estimates of volatility, historical (Hvol), EWMA (Evol), Deep learning (Dvol) and Random Features (Rvol), plus the ROC of the *simple model*.

There are few aspects that can be noted from the chart above: first, as expected, Debt value and Market capitalization have an important role in predicting whether a company will default or not in the next year, infact, the AUC of the *simple model* is different from 0.5 (the AUC of the 45 degrees line, that is the random model). The *simple model* has good predictive power for low threshold of default, meaning that it can rank the riskiest companies quite well. However, its performance deteriorates when the threshold increases. Second, the MM seems to bring additional information compared to the *simple model*, since their AUCs are greater than the AUC of the *simple model*. Third, forward looking volatilities (Evol, Dvol and Rvol) seem to deliver a greater predictive power than backward looking volatility (Hvol). Furth, the MM that uses Exponentially Weighted Moving Average volatility is the one that has the highest performance.

	Variables	Logpvalues	Pvalues	Corr
Hvol vs Evol	Historical vs EWMA vol.	-4.12	0.00008	0.76
Hvol vs Dvol	Historical vs Deep Learning vol.	-1.43	0.037381	0.92
Hvol vs Rvol	Historical vs Random features vol.	-1.89	0.01274	0.92
Hvol vs Simple Model	Historical vol. vs Debt/Mkt cap	-0.55	0.28206	0.71
Evol vs Dvol	EWMA vs Deep Learning vol.	-3.07	0.000843	0.84
Evol vs Rvol	EWMA vs Random features vol.	-2.65	0.00226	0.83
Evol vs Simple Model	EWMA vol. vs Debt/Mkt cap.	-4.11	0.00008	0.62
Dvol vs Rvol	Deep Learning vs Random features vol.	-0.18	0.666419	0.93
Dvol vs Simple Model	Deep Learning vol. vs Debt/Mkt cap.	-1.42	0.037628	0.69
Rvol vs Simple Model	Random features vol. vs Debt/Mkt cap.	-1.55	0.02792	0.67

Table 3: DeLong non-parametric test - results

The significance of the AUCs can be observed from the table above, that shows the p-values from the non-parametric test of DeLong. The test checks if the AUCs are significantly different; generally, if the p-value is lower than 5%, the areas are considered different.

The table below shows the results of the test of DeLong against an AUC of 0.5, to check if the AUCs are significantly different from the area of the 45 degrees curve.

	Variables	PValues	Zscores
Hvol	Historical volatility	0.00003	4.178
Evol	EWMA volatility	0.00000	8.151
Dvol	Deep learning volatility	0.00000	5.252
Rvol	Random features volatility	0.00000	5.424
Simple Model	Devt/Mkt cap	0.00236	3.041

Table 4: DeLong non-parametric test vs 45 degrees line - results

To better understand why forward-looking volatilities, have a greater predictive power than historical volatility, and why EWMA volatility is the one with the best performance, further analyses are carried out.

The table below shows the volatilities' descriptive statistics. From the table some features can be detected. First, EWMA volatility is the one that better differentiates between Defaulting companies and non-defaulting companies, infact, the 1-year PD average for defaulting group is the highest, while 1-year PD average for non-defaulting group is the lowest.

	PDs Def.	PDs Non Def.	Vol of Vol Def.	Vol of Vol Non-Def.	Mean of Vol Def.	Mean of Vol Non-Def.	Model perf.
Hvol	19.43%	8.57%	0.00%	0.00%	77.51%	63.66%	18
Evol	25.78%	5.23%	68.18%	18.96%	107.78%	50.21%	42
Dvol	21.92%	7.96%	24.30%	13.52%	87.55%	62.26%	37
Rvol	21.56%	7.93%	25.66%	13.64%	85.70%	61.44%	41

Table 5: Volatilities descriptive statistics

This characteristic of EWMA volatility can be observed in other columns of the table, clearly showing that it is the model that best differentiates between the groups. The last column of the table displays the model's performance, measured by the number of times each model achieved the highest R^2 out-of-sample, compared to the other models. Following this definition, the Historical volatility model has the worst performance, followed by the Deep Learning model and Random Features model.

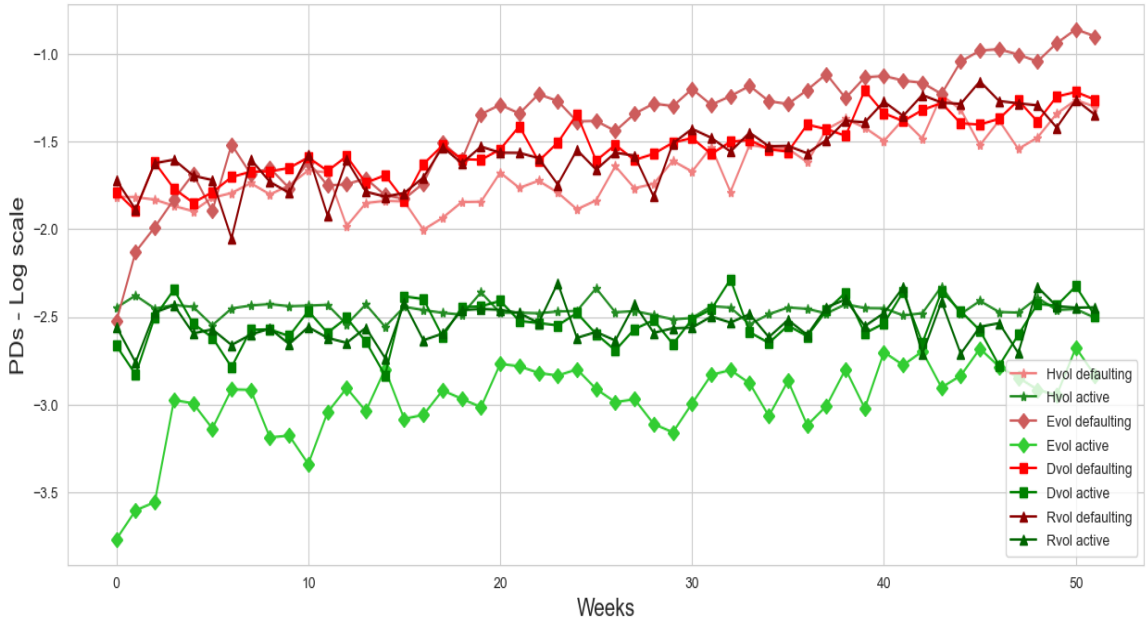

Figure 10: PDs defaulting vs PDs non-defaulting - 52 weeks

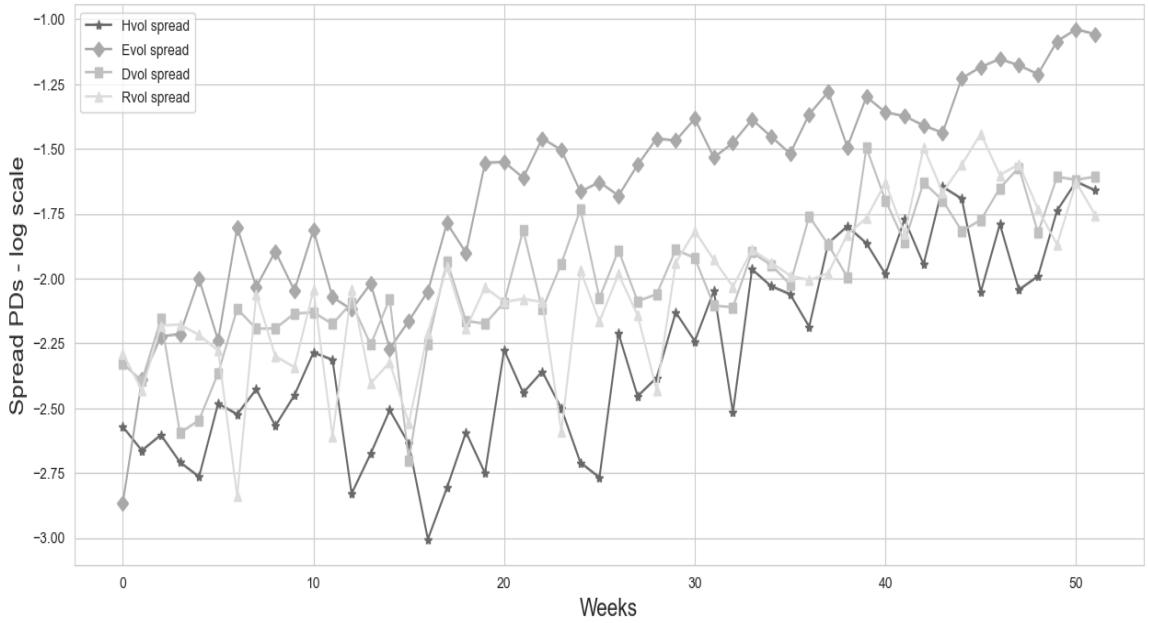

Figure 11: Spread PDs Defaulting vs non-Defaulting - 52 weeks

The two figures above summarize the features described before.

3. LSTM Model Evaluation

This subsection of chapter three is dedicated to the evaluation of the performance of the LSTM model. Even in this case ROC curves and AUCs are used to compare the different models, and the non-parametric test of DeLong is carried out to check if the AUCs are significantly different.

The figure below displays the ROC curves and AUCs of the LSTM model fed with the same variables of the MM and the four estimates of volatility. As explained in chapter 2, subsection 4, for LSTM model bootstrapping is used to increase the size of the sample from 138 to 1000 companies. The model is then trained on 800 companies and tested on the remaining 200.

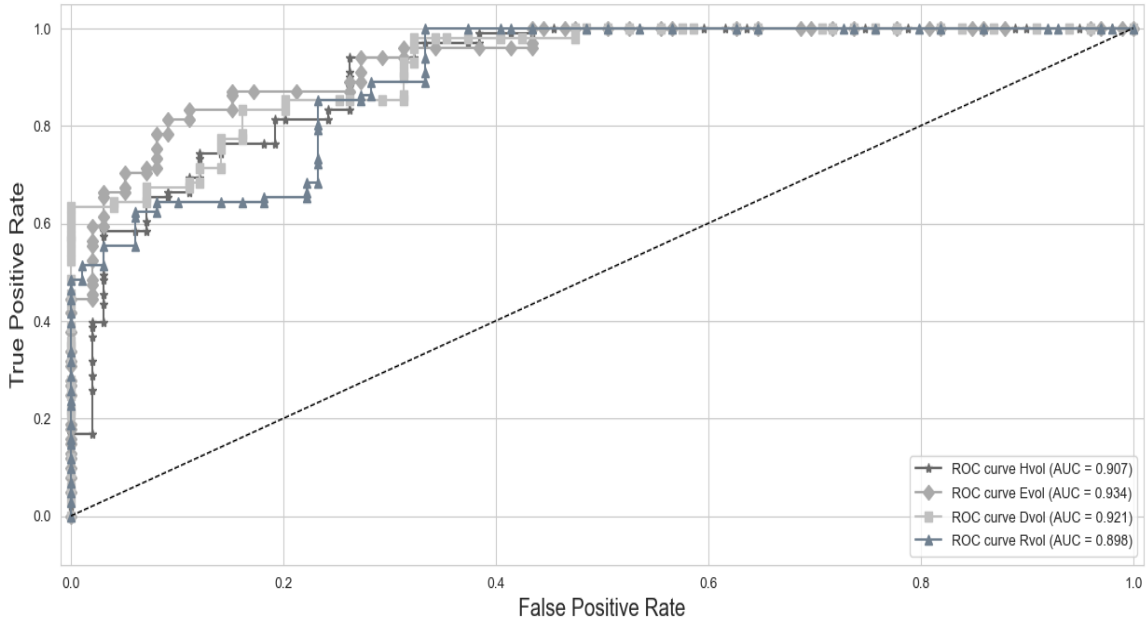

Figure 12: ROC curves and AUCs for LSTM model with different volatility estimates

The table below shows the results of the non-parametric test of DeLong. Despite the LSTM model that uses as feature the EWMA volatility has the highest AUC, it is not significantly different from the others.

	Variables	Logpvalues	Pvalues	Corr
LSTM Hvol vs LSTM Evol	LSTM model Hist. vs EWMA vol.	-0.50	0.31572	-0.03
LSTM Hvol vs LSTM Dvol	LSTM model Hist. vs Deep Learning vol.	-0.32	0.48329	0.51
LSTM Hvol vs LSTM Rvol	LSTM model Hist. vs Random features vol.	-0.28	0.52748	0.73
LSTM Evol vs LSTM Dvol	LSTM model EWMA. vs Deep Learning vol.	-0.27	0.54311	0.18
LSTM Evol vs LSTM Rvol	LSTM model EWMA. vs Random features vol.	-0.88	0.13260	0.18
LSTM Dvol vs LSTM Rvol	LSTM model Deep Learning. vs Random features vol.	-1.08	0.08317	0.77

Table 6: DeLong non-parametric test LSTM models – results

The figure below displays the ROC curves and the AUCs of the LSTM models where different estimates of volatility are used, and of the MMs, where the PDs are computed on the same test sample as the LSTM models.

The non-parametric test of DeLong on table 7 shows that, except for the MM with EWMA volatility and LSTM model with Random features volatility, the LSTM models deliver a significantly higher AUC than the MMs.

This means that the LSTM models generally can better rank companies based on their riskiness compared to a simple Merton Model fed with the same information.

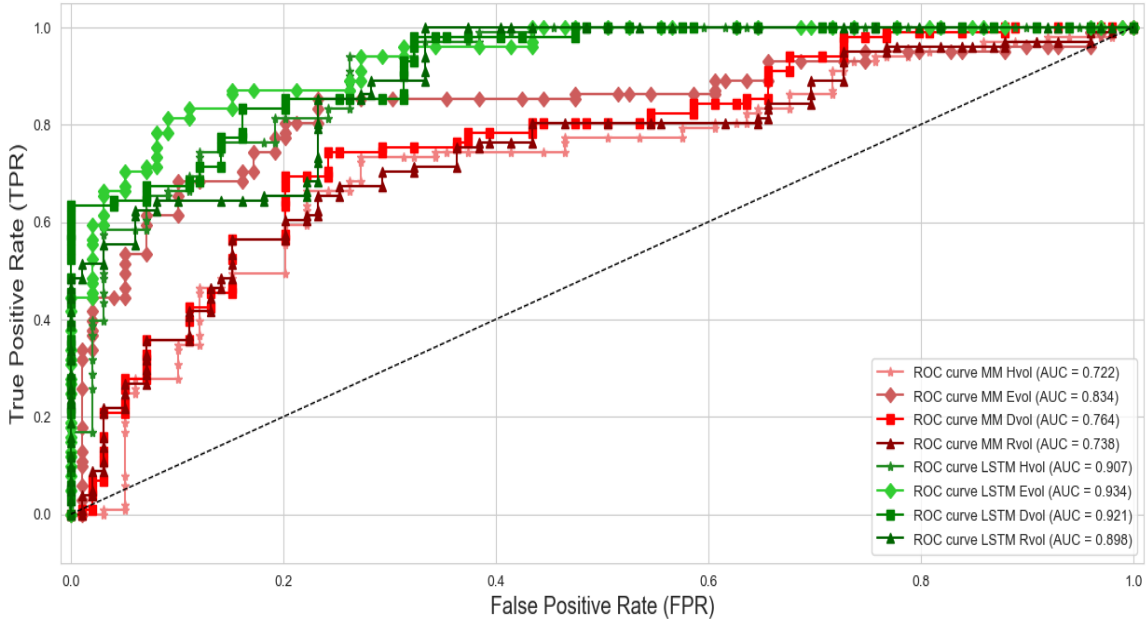

Figure 13: ROC curves and AUCs for MMs and LSTM models

	Variables	Logpvalues	Pvalues	Corr
MM Hvol vs LSTM Hvol	MM Hist. Vs LSTM Hist. vol	-4.86	0.00001	-0.04
MM Hvol vs LSTM Evol	MM Hist. Vs LSTM EWMA. vol	-11.71	0.00000	0.60
MM Hvol vs LSTM Dvol	MM Hist. Vs LSTM DL. vol	-6.51	0.00000	0.12
MM Hvol vs LSTM Rvol	MM Hist. Vs LSTM RF. vol	-4.78	0.00002	0.07
MM Evol vs LSTM Hvol	MM EWMA. Vs LSTM Hist. vol	-1.33	0.04666	-0.04
MM Evol vs LSTM Evol	MM EWMA. Vs LSTM EWMA. vol	-5.05	0.00001	0.69
MM Evol vs LSTM Dvol	MM EWMA. Vs LSTM DL. vol	-1.88	0.01323	-0.01
MM Evol vs LSTM Rvol	MM EWMA. Vs LSTM RF. vol.	-1.14	0.07308	0.05
MM Dvol vs LSTM Hvol	MM DL. Vs LSTM Hist. Vol	-3.62	0.00024	0.02
MM Dvol vs LSTM Evol	MM DL. Vs LSTM EWMA. Vol	-9.55	0.00000	0.63
MM Dvol vs LSTM Dvol	MM DL. Vs LSTM DL. Vol	-4.68	0.00002	0.08
MM Dvol vs LSTM Rvol	MM DL. Vs LSTM RF. Vol	-3.36	0.00044	0.09
MM Rvol vs LSTM Hvol	MM RF. Vs LSTM Hist. Vol	-4.47	0.00003	0.00
MM Rvol vs LSTM Evol	MM RF. Vs LSTM Hist. Vol	-10.95	0.00000	0.61
MM Rvol vs LSTM Dvol	MM RF. Vs LSTM Hist. Vol	-5.81	0.00000	0.10
MM Rvol vs LSTM Rvol	MM RF. Vs LSTM Hist. Vol	-4.25	0.00006	0.08

Table 7: DeLong non-parametric test MM vs LSTM – results

Conclusion

This thesis mainly focuses on the integration of Machine Learning techniques within the Merton Model, with a particular attention on the sensitivity of the MM to how equity volatility is computed and then used in the two-equations system to compute the volatility of the asset returns.

The analyses conducted in this thesis show some interesting features. The first characteristic that seems emerging from this work is that the way in which equity volatility is computed plays an important role in the capacity of the Merton Model to rank companies on their riskiness. Furthermore, the analyses show that a forward-looking volatility seems to be a better choice than a backward-looking volatility, since it tends to deliver a better performance and a higher Area under the ROC curve.

From the analyses, there seems to be a correlation between how well the equity volatility model can predict future volatility and the performance of the Merton Model, infact, the EWMA model, that is the one that better predict future volatility, is also the one that delivers the highest performance when used to compute the volatility of the assets returns. This finding seems to be in line with Occam's Razor and the principle of parsimony (simpler models are usually more effective). However, some aspects that can explain the underperformance of both Random Features model and Deep Learning model, must be taken into consideration. First, ML models are very sensible to the number of observations used in the train set. In this thesis five years of data are taken into account, with the last year is used as test set. Thus, the train set is composed of four years of weekly data, that is

approximately 200 observations, when a number greater than 1000 would have been preferable. Second, in this work hyperparameters tuning is not conducted.

Finally, the LSTM model seems to deliver a better performance compared to the traditional Merton Model, and it can significantly increase the Area under the ROC curve. However, also in this case, the size of the data plays an important role. Infact, bootstrapping was conducted to increase the size of the sample, otherwise would have been impossible to train the LSTM model just with 138 observations. Thus, the traditional Merton Model seems to be a better choice whenever the size of the sample is small.

References

1. Zvika Afik, Ohad Arad, Koresh Galil. “Using Merton Model for Default Prediction: An Empirical Assessment of Selected Alternatives.” *Journal of Empirical Finance,* 2015.
2. Sreedhar Bharath and Tyler Shumway. “Forecasting Default with the Merton Distance to Default Model.” *Review of Accounting Studies,* 2008.
3. Fisher Black and John C. Cox. “Valuing Corporate Securities: Some Effects of Bond Indenture Provisions.” *The Journal of Finance,* Vol. 31, No. 2, 1976.
4. Fisher Black and Myron Scholes. “The Pricing of Options and Corporate Liabilities.” *Journal of Political Economy*, Vol. 81, No. 3, 1973.
5. Damiano Brigo and Fabio Mercurio. *Interest Rate Models – Theory and Practice.* Springer Finance, 2nd Edition, 2006.
6. François Chollet. *Deep Learning with Python.* Manning Publications, 2017.
7. Peter Crosbie and Jeff Bohn. “Modeling Default Risk.” *Moody's KMV,* 2003.
8. E. R. DeLong, D. M. DeLong, and D. L. Clarke-Pearson. “Comparing the Areas Under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach.” *Biometrics*, 1988.
9. Darrell Duffie and David Lando. “Term Structures of Credit Spreads with Incomplete Accounting Information.” *Econometrica,* 2001.
10. Jin-Chuan Duan. “Maximum Likelihood Estimation Using Price Data of the Derivative Contract.” *Mathematical Finance,* Vol. 4, No. 2, 1994.

11. Frank J. Fabozzi. *The Handbook of Fixed Income Securities.* McGraw-Hill, 7th Edition, 2005.
12. J. Forssbæck and A. Vilhelmsson. *Predicting Default – Merton vs. Leland*, 2017.
13. Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.* O'Reilly Media, 2nd Edition, 2019.
14. Kay Giesecke. "Default and Information." *Journal of Financial Economics,* 2001.
15. Kay Giesecke. *Credit Risk Modeling and Valuation: An Introduction.* Springer, 2004.
16. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Springer, 2nd Edition, 2009.
17. S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory." *Neural Computation,* 1997.
18. John C. Hull and Sankarshan Basu. *Options, Futures, and Other Derivatives.* Pearson Education, 9th Edition, 2017.
19. Philip E. Jones, Scott P. Manson, and Eric Rosenfeld. "Contingent Claims Analysis of Corporate Capital Structures: An Empirical Analysis." *Journal of Financial Economics,* 1984.
20. J. Longerstaey and M. Spencer. *RiskMetrics™ – Technical Document.* Morgan Guaranty Trust Company of New York, 1996.
21. Robert C. Merton. "On the Pricing of Corporate Debt: The Risk Structure of Interest Rates." *The Journal of Finance*, Vol. 29, No. 2, 1974.
22. Ali Rahimi and Ben Recht. "Random Features for Large-Scale Kernel Machines." *Neural Information Processing Systems*, 2007.
23. Steven E. Shreve. *Stochastic Calculus for Finance II: Continuous-Time Models.* Springer Finance, 2004.
24. Merxe Tudela and Garry Young. "A Merton-Model Approach to Assessing the Default Risk of UK Public Companies." *Bank of England Working Paper*, 2003.

Appendix A

LSTM Python Code

```
import numpy as np
import pandas as pd
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
import seaborn as sns
import math
import random
import torch
from torch.utils.data import Dataset, DataLoader, TensorDataset
import torch.optim as optim
import torch.nn as nn
import warnings
from tqdm import tqdm
warnings.filterwarnings("ignore")
```

```
class LSTMModel(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers, output_size):
        super(LSTMModel, self).__init__()
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers,
batch_first=True)
        self.fc = nn.Linear(hidden_size, output_size)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        h0 = torch.zeros(num_layers, x.size(0), hidden_size).to(x.device)
        c0 = torch.zeros(num_layers, x.size(0), hidden_size).to(x.device)
        out, _ = self.lstm(x, (h0, c0))
        out = self.fc(out[:, -1, :])  # Get the last time step's output
        out = self.sigmoid(out)
```

```
        return out

    def set_seed(seed_value = 42):
        np.random.seed(seed_value)
        torch.manual_seed(seed_value)
        random.seed(seed_value)
```

```
def train_model(num_epochs: int,
                train_loader: DataLoader,
                criterion,
                optimizer,
                model):
  # Training loop
  for epoch in range(num_epochs):

        for inputs, targets in train_loader:
            # Forward pass
            outputs = model(inputs)
            loss = criterion(outputs, targets)

            # Backward and optimize
            optimizer.zero_grad() # kill old gradients
            loss.backward() # compute new gradients
            optimizer.step() # perform the step of gradient descent

        if (epoch+1) % 20 == 0:
            print(f'Epoch [{epoch+1}/{num_epochs}], Loss:
{loss.item():.4f}')

def get_predictions(loader, model):
    model.eval()
    targets = []
    predictions = []

    with torch.no_grad():
        for inputs, labels in loader:
            outputs = model(inputs)
            targets.extend(labels.numpy())
            predictions.extend(outputs.numpy())

    return np.array(targets).flatten(), np.array(predictions).flatten()
```

```
#function to normalize the signals
def normalization_3d(data: np.array, norm_params: dict = None):
    if norm_params is None:
        data_max = np.max(data, axis = (0, 1))
        data_min = np.min(data, axis = (0, 1))
    else:
        data_max = norm_params['max']
        data_min = norm_params['min']
    data_norm = (data - data_min)/(data_max - data_min) -0.5
    params_used = {
        'max':data_max,
        'min':data_min
    }
    return data_norm, params_used
```

```
def run_LSTM(split: int, # train set vs test set split
             data: np.array, # array with the MM variables for the past 52
weeks
             input_size: int, # number of features
             width: int, # number of neurons
             depth: int, # number of hidden layers
             output_size: int, # output layer size
             seed: int = 42,
             seeds: int = 10
            ):

    signals,_ = normalization_3d(data) # normalization of the features
with min max
    labels = np.array([1]*500 + [0]*500).reshape(-1, 1) # 1 for def and 0
for non def
    #shuffling the companies so not to have alla def companies together
    indices = np.arange(signals.shape[0])
    np.random.seed(42)
    np.random.shuffle(indices)
    signals = signals[indices]
    labels = labels[indices]
    # splitting train set and test set
    train_x, train_y = signals[:split], labels[:split]
    test_x, test_y = signals[split:], labels[split:]
    x_train_tensor, y_train_tensor = torch.tensor(train_x, dtype =
torch.float32), torch.tensor(train_y, dtype = torch.float32)
```

```
    x_test_tensor, y_test_tensor = torch.tensor(test_x, dtype =
torch.float32), torch.tensor(test_y, dtype = torch.float32)
    # Create TensorDatasets and DataLoaders for training and test sets
    train_dataset = TensorDataset(x_train_tensor, y_train_tensor)
    test_dataset = TensorDataset(x_test_tensor, y_test_tensor)
    # Creating the loaders
    train_loader = DataLoader(train_dataset, batch_size=8, shuffle=True)
    test_loader = DataLoader(test_dataset, batch_size=8, shuffle=False)

    # creating the model with the give architecture for each seed from
seeds
    models = [LSTMModel(input_size, width, depth, output_size) for seed in
range(seeds)]
    #Training the model for each seed in seeds
    for seed in range(seeds):
        print(seed)
        LSTMModel.set_seed(seed)
        # Define the loss function and optimizer
        criterion = nn.BCELoss() #Binary Cross Entrpy is used since it is
a classification problem
        optimizer = optim.Adam(models[seed].parameters(), lr=0.001)
        # Train the LSTM model
        train_model(num_epochs=200,
                    train_loader=train_loader,
                    criterion=criterion,
                    optimizer=optimizer,
                    model=models[seed])

    # Get predictions
    train_sample = [get_predictions(train_loader, models[seed]) for seed
in range(seeds)]
    test_sample = [get_predictions(test_loader, models[seed]) for seed in
range(seeds)]

    train_targets = train_sample[0][0]
    test_targets = test_sample[0][0]
    # averaging the predictions from the 10 different seeds to get the
final result
    train_predictions = np.concatenate([train_sample[seed][1].reshape(-1,
1) for seed in range(seeds)], axis=1).mean(1)
    test_predictions = np.concatenate([test_sample[seed][1].reshape(-1, 1)
for seed in range(seeds)], axis=1).mean(1)
```

```
    return test_predictions, test_targets

indexes = [[0,4,8,12,13,14], [1,5,9,12,13,14], [2,6,10,12,13,14],
[3,6,11,12,13,14]] # indexes to take the MM variables
volatilities = ['Hvol', 'Evol', 'Dvol', 'Rvol'] # volatilities types
data_features = np.concatenate([features_def, features_active], axis = 0)
# array where are stored the MM variables for the past 52 weeks
split = int(data_features.shape[0]*0.8)
predictions_array = np.zeros((200, len(volatilities))) # array where to
store the predictions for further analysis

for i in range(len(indexes)):
    idx = indexes[i]
    features_touse = data_features[:,:,idx]
    predictions_array[:, i], test_targets = run_LSTM(split = split,
                                                     data=features_touse,
                                                     input_size=len(idx),
                                                     width = 4, depth = 2,
                                                     output_size = 1)
```

```
# plotting the LSTM results

colors = ['dimgray', 'darkgray', 'silver', 'slategrey']
markers = ['*', 'D', 's', '^']
# iterating for all the columns of the predictions array
for i in range(predictions_array.shape[1]):
    test_predictions = predictions_array[:, i]
    fpr, tpr, _ = roc_curve(test_targets, test_predictions)
    roc_auc = auc(fpr, tpr)
    plt.plot(fpr, tpr, marker = markers[i], color = colors[i],label=f'ROC
curve {volatilities[i]} (AUC = {roc_auc:.3f})')

plt.plot([0, 1], [0, 1], color='black', lw=1, linestyle='--')  # Diagonal
line
plt.xlim([-0.01, 1.01])
plt.ylim([-0.1, 1.05])
plt.xlabel('False Positive Rate', fontsize = 15)
plt.ylabel('True Positive Rate', fontsize = 15)
plt.legend(loc='lower right')
plt.show()
```