POLITECNICO DI MILANO
School of Industrial and Information Engineering
Master of Science Degree in Mathematical Engineering
Applied Statistics

Estimating counterparty credit risk fusing traditional and alternative data

Supervisor: Prof. Danilo Ardagna
External Tutor: Dott. Matteo Bregonzio

Master Thesis
Danilo Parisi
862767

Academic Year 2017-2018

Gli uomini passano, ma le idee restano.
Restano le loro tensioni morali
e continueranno a camminare
sulle gambe di altri uomini.
Giovanni Falcone

Contents

List of Figures

List of Tables

Abstract

This thesis proposes an innovative methodology for estimating company's credit risk; in specific it studies counterpart risk exploiting a data driven approach combined with alternative data. Counterparty risk is a well know problem within the finance domain; practically, it evaluates the risk that the counterparty will not live up to its contractual obligation.

This work derives from a curricular internship at *3rdPLACE*, which is a company that offers solutions and services in the field of intelligence applied to digital data, and it was commissioned by a company that supports financial institutions, large, medium and small businesses, insurance companies, public administrations and professionals in effective credit management.

The project consist in creating a machine learning algorithm that allows the prediction of companies default. The dataset they provided to us are composed by Italian companies registered at the national Companies House, half of them are available the balance sheet, while the other half we do not have this information. Within this project, the goal involves developing an innovative credit risk estimator designed to work on medium, small and very small Italian companies not quoted on exchange, using a methodology that is purely data-driven with the technologies of machine learning. Differently from listed company where clear and transparent information is publicly available, in this project we tackle also a challenge where information is scarce, not standardized.

Furthermore it has been proposed to create a new feature, in addition to those provided by the client company: the digital score. It measures the company's presence, performance and effectiveness on the web and integrates it into the initial classification problem of default. The end users of this innovative method could be the rating agencies that deal with financial risk, financial institutions and banks.

Sommario

La presente tesi propone una metodologia innovativa per la stima del rischio di credito dell'azienda, in particolare studia il rischio di controparte sfruttando un approccio guidato puramente dai dati combinato con dati alternativi. Il rischio di controparte è un problema ben noto nell'ambito della finanza; in pratica, valuta il rischio che la controparte non rispetti i propri obblighi contrattuali.

Questo lavoro deriva da uno stage curriculare presso *3rdPLACE*, un'azienda che offre soluzioni e servizi nel campo dell'intelligence applicata ai dati digitali, ed è stata commissionata da una società che supporta istituzioni finanziarie, grandi, medie e piccole imprese, compagnie assicurative, pubbliche amministrazioni e professionisti nella gestione efficace del credito.

Il progetto consiste nella creazione di un algoritmo di apprendimento automatico che consente di prevedere le aziende inadempienti. Il set di dati che ci hanno fornito sono composti da società italiane registrate presso la Camera di Commercio. Metà di essi hanno disponibile un bilancio finanziario, mentre per l'altra metà queste informazioni non sono disponibili. All'interno di questo progetto l'obiettivo prevede lo sviluppo di un innovativo stimatore del rischio di credito progettato per lavorare su medie, piccole e piccolissime aziende italiane non quotate in borsa, utilizzando una metodologia puramente guidata dai dati con le tecnologie del machine learning. A differenza della società quotata in cui sono disponibili informazioni chiare e trasparenti, in questo progetto affrontiamo anche una sfida in cui le informazioni sono scarse, non standardizzate.

Inoltre, è stato proposto di creare una nuova feature: il digital score. Esso misura la presenza, le prestazioni e l'efficacia dell'azienda sul web e la integra nel problema di classificazione iniziale di default. Gli utenti finali di questo metodo innovativo potrebbero essere le agenzie di rating che si occupano di rischio finanziario, istituzioni finanziarie e banche.

Chapter 1

Introduction

The present work derives from a curricular internship at **3rdPLACE S.R.L.** 3rdPLACE[1] is a privately owned company founded by senior managers of Google. Using Artificial Intelligence proprietary technologies, 3rdPLACE supports organizations in converting data about users and customers into business insights to improve decision making, actions and operational results.

3rdPLACE is trusted by Nestlè, Euronics, BNL-BNP Paribas, IBS, Amplifon, UniCredit & many others.

This thesis was commissioned by a client company [2], which is part of a group listed on Milan Stock Exchange (STAR segment). The group provides a wide range of business services including Digital Trust, Credit Information & Management, and Sales & Marketing Services. Worthy to mention that the client is among the three leading Italian operators within its sector.

The offered services of the client company are crucial to customers in anticipate business choices, developing new and more complete investments, and expanding and enriching existing businesses. Normally, the services are sold to banks, financial institutions, investment funds and private companies.

Within the client company research & development agenda, there is strong attention to alternative data and data-driven approaches. For those two reasons we have been engaged to 1) collect and analyse a novel set of alternative data coming from the internet and 2) develop a machine learning algorithm capable of estimating conterparty risk by fusing traditional data with the novel set of alternative data.

[1] `http://3rdplace.com/`

[2] For privacy reasons it is not possible to disclose the name of the company that commissioned this work, due to a contractual constrains.

Thesis Goals & Motivation

The client company, as a method of estimating credit risk, uses a proprietary score approach based on economic and statistical analysis combined with personal experience. Based on our initial assessment, the main limitation involves the use of only few variables, of order of few tens, as predictors, very few for a complex problem as the credit risk prediction. Another limitation is that this model is static. It does not automatically update itself with the incoming of new data, but only when the prediction error becomes substantial. Overall, it is a quite rigid method.

For this reason they have contacted 3rdPLACE for studying and trying to enhance their model. The model they use has the following performance: an accuracy of 0.945, a precision of 0.155 a recall of 0.129 and an F1-measure of 0.141. These are the target performance for the model we will develop trough the present thesis, and the goal is to improve them and provide to the client company a model that performs better.

Within this project the goal involves developing an innovative credit risk estimator designed to work on medium, small and very small Italian companies not quoted on exchange, using a methodology that is purely data-driven, which exploits machine learning technologies. Differently from listed company where clear and transparent information is publicly available, in this project we tackle a challenge where information is scarce, not standardized.

Given the growing importance and effectiveness of alternative data such as digital information collected from the internet, it has been proposed to create a new feature, the **digital score**, which measures the company's presence, performance and effectiveness on the web and integrates it into the initial classification problem of default. Web data have already been interesting for E-commerce portals, media companies, research firms and data scientists, and therefore could be useful for the present problem.

In this thesis, machine learning models are developed for the prediction of companies' default. In particular, gradient boosting, support vector machines and neural networks are used. For their development a priori information are not used, as one of the goals is to create a classifier that is completely data-driven. Gradient boosting will prove to be the most suitable model and with better performance for this dataset. It will have a better than 26% F1-measure compared to the model used by the client company: the latter model has a score of 0.141, while gradient boosting has an F1-measure of 0.179.

The end users of this innovative method could be the rating agencies that deal with financial risk, financial institutions and banks.

Overview

The present work is composed by the following parts.

Chapter 2 outlines the state of art. The first part introduces the concept of credit risk and briefly discusses the existent methods for credit score evaluation. In the second part, in a slightly more detailed way, we study more in deep the supervised learning models that will be used in the present thesis: gradient tree boosting, support vector machine and neural networks. In the final part of this chapter we discuss about the technique of principal component analysis.

Chapter 3 provides a description of the financial dataset the client company provided to us. We do some descriptive analysis and discuss about features preprocessing. We describe how the observations of the dataset are distributed in Italy, through the use of geographical maps. Finally, a principal component analysis is performed for 2-D visualization of the dataset.

Chapter 4 describes the construction of the web crawler and how we download the digital features of each company. Then we discuss how we combine them to get the digital score which should measure the presence and the performance of a company on internet. Next we study briefly the distribution of the digital features.

Chapter 5 is dedicated to the developments of the models for predicting the default of a company. In this Chapter is therefore presented all the experimental results, for the prediction problem. Moreover, the final part of the chapter compares the client company model with the one trained in the present thesis.

Finally, Chapter 6 draws conclusions and outlines future developments.

Chapter 2

State of art

This Chapter reviews the state of the art and the main concepts utilised in this work. The chapter structure is the following: Section 2.1 discusses the setting on which the present work is organised: financial risk and credit scoring are briefly introduced. Section 2.2 describes the machine learning algorithms used, their mathematical formulation and how they work. In details, gradient tree boosting, support vector machines and neural networks are reviewed. Later, the popular metrics for classification performance evaluation are introduced. Finally, Section 2.3 reviews the theory of principal components analysis.

2.1 Credit Scoring

Financial Risk is one of the major concerns within the banking sector. Risk can be referred as the chances of having an unexpected or negative outcome. In practice, any action or activity that leads to loss of any type can be termed as risk. In specific, financial risk encompasses many types of risks such as company's capital structure, financing and the finance industry; hence it could cause financial losses. Financial risk is indeed a priority for every business and it can be classified into various types[1], as we can see in Figure 2.1:

Market risk involves the risk of changing conditions in the specific marketplace in which a company competes for business.

Credit risk is the risk that businesses incur by extending credit to customers. It can also refer to the company's own credit risk with suppliers. A business

[1] `https://www.investopedia.com/terms/f/financialrisk.asp`

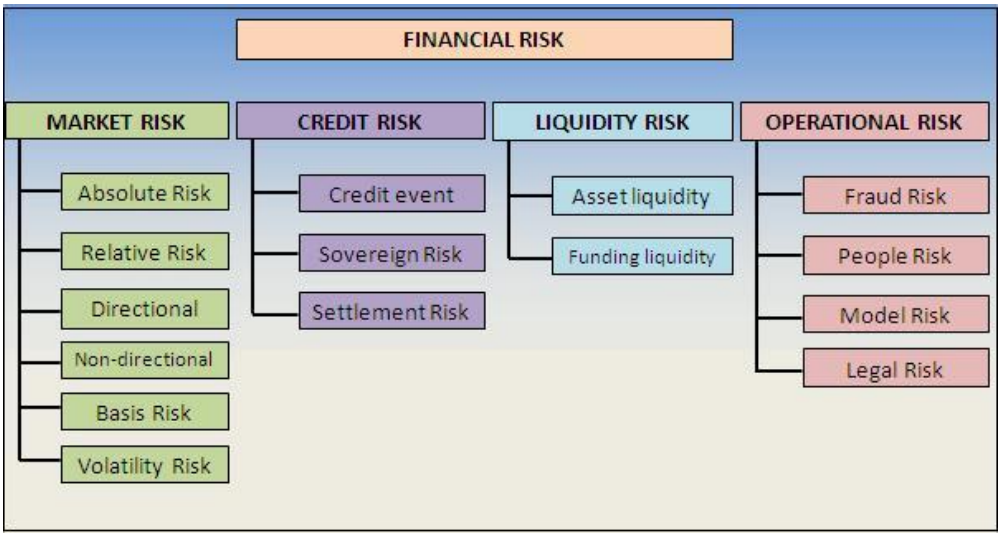


Figure 2.1: Financial risk. **Reference**:
`https://www.simplilearn.com/financial-risk-and-types-rar131-article`

takes a financial risk when it provides financing of purchases to its customers, due to the possibility that a customer may default on payment.

Liquidity risk arises out from the inability to execute transactions. Liquidity risk can be classified into Asset Liquidity Risk and Funding Liquidity Risk. The first, for example, arises either due to insufficient buyers or sellers against sell orders and buy orders respectively.

Operational risk refers to the various risks that can arise from a company's ordinary business activities. The operational risk category includes lawsuits, fraud risk, personnel problems and business model risk, which is the risk that a company's models of marketing and growth plans may prove to be inaccurate or inadequate.

In this work we only deal with the credit risk, and more specifically in counterparty risk, which is the risk to each party of a contract that the counterparty will not live up to its contractual obligations.

Credit scoring is a quantitative measure of counterparty risk. Connected to a financial transaction, it is widely used for evaluating business loans applications, allowing to predict the probability that a subject will default or can repay the loan. The meaning of credit scoring is to assign scores to the characteristics of debt and borrowers and historical default and other loss experienced as an indication of the risk level of the borrower. The aim of the credit score model is to build a single aggregated risk indicator for a set of risk factors. Initially, it was used in the context of consumer credit, and since small and medium-sized companies can be assimilated to individuals, these statistical tools are also used to measure the riskiness of this type of loans.

In the past, the decision to invest or lend money was based on subjective considerations, and personal relationships between the applicant and the loan holder. In recent times, more sophisticated procedures have been studied in order to minimize risks. The main idea involved processing customer's information in order to compute a score that reflects the creditworthiness of the counterpart. One of the benefits of having a standardized score consist in been able of comparing opportunities and mitigate risks.

Using historical data and statistical techniques, the method produce a score which measure the risk of a subject requesting a loan and, it can be used to rank the applicants in terms of risk. After having computed the risk, the interest of the loan is changed, offering a lower interest to the counterpart with less risk and vice versa. Applicant's monthly income, outstanding debt, financial assets, years of life are all potential factors that may relate to loan performance and they are part of the explanatory variables.

The first scoring models were developed in the 1940s, but only in the next decade it began to spread the first consulting firms such as **Fair, Isaac Corporation (FICO)**[2], and nowadays these methods are widely used for consumer lending and also for business. Scoring tools improves the ability to discriminate against deserving and non-deserving customers, leading to a reduction in losses and an increase in the ability to manage a high volume of requests and financial products [18].

Credit scoring system can be developed using several statistical methods. The most used are linear probability models, which assumes a linear relationship between the default and the counterpart profile, the features, logit models, in which the probability is logistically distributed, probit models, in which the cumulative probability is assumed to follow a Normal distribution, and discriminant analysis models that do not model directly the probability but divide the applicants in high and low-risk classes.

According to Fair, Isaac Corporation, 50 or 60 variables might be considered when developing a typical model, but only few of them, up to 12 are the most predictive features. The accuracy of a credit scoring system will depend on the care with which it is developed. The data, on which the system is based, need to be a rich sample of both well-performing and poorly performing loans. They should be up to date, and the models need to be re-estimated frequently to ensure that changes in the relationships between potential factors and loan performance are captured. It is important mentioning that, if a bank starts using an automated scoring model, it must ensure that the new applicants behave similarly to those on which the model was built; otherwise, they may not accurately predict the behavior of these new applicants [15].

[2]FICO is a leading analytics software company that use Big Data and mathematical algorithms to predict consumer behavior. FICO® Score is the standard measure of consumer credit risk in the United States, and it allowed to have credit more widely available around the world.

There have been many studies for the creation of a credit score. On the Italian scene, the project involving the University of Trieste, the Industrial Observatory of Sardinia (OSSIND, Cagliari), the Institute for Economic Studies and Analysis (ISAE, Rome) and the Research Institute on the Enterprise and the territory (CERIS, Turin) was very important and worthy of mention, in which the risk of the financial markets and the country risk were taken into consideration in addition to the risk of corporate default [7].

In addition to the already discussed credit scoring, there are also other methods to model the risk of default, and they can be summarized as follows:

- **Rating System.** They are models used by rating agencies, like Moody's and Standard & Poor's, to analyze the credit quality of a single company. Each agency uses a different evaluation and the resulting models assign a rating in relation to the weight that each factor considered has in the model.

- **Option Pricing.** This approach is based on the option pricing model developed by Black and Scholes and Merton in the 1970s. According to this methodology, the insolvency of a firm occurs when the value of the company's activities is lower than its debts and liabilities.

  A first approach is to model debt as an European call option

  $$E_t = (A_t - X)^+ \tag{2.1}$$

  where E_t is the value of the firm's equity, X is the debt and A_t is the value of its assets at time t, which, in this setting, it is assumed that follows a geometric Brownian motion with mean rate of return on the assets μ and volatility σ

  $$dA_t = \mu A_t dt + \sigma A_t dW_t. \tag{2.2}$$

  Without going too much in detail, using the well-known formula of Black-Scholes, the default probability at time t is

  $$P_t[A_t \leq X] = \mathcal{N}(-d_2^p) \tag{2.3}$$

  where $d_2^p = \frac{\log(A_t/K)+\mu+\sigma^2/2(T-t)}{\sigma\sqrt{T-t}}$, and $\mathcal{N}$ is the Normal cumulative distribution.

  The advantage related to the simplicity of this model is opposed by a series of limitations in the hypotheses. First of all, we consider the default only at the expiry of the option: there are various models that relax this hypothesis, but, in doing so, we no longer have available closed formulas. The second is that the equity develops according to a Brownian model that often is not verified in reality, and it is also assumed tradeable, when it is not even directly observable [14].

Statistical models, in particular logistic regression, and artificial intelligence models (AI) are the most important methods for credit scoring. The prediction accuracies of the statistical models, however, are usually not high, therefore recently artificial intelligence techniques start growing to be used.

One of the model used to the problems related to business insolvency are represented by the application of artificial neural networks, especially since the 90s. These models are inspired by research in the biological field and in particular those based on the structure of the brain. The Italian project, mentioned before, summarized the main contributions of the economic-financial literature, focusing on the strengths and weaknesses of neural networks in relation to the analysis of the risk of insolvency [7].

Gradient boosting method is, also, an alternative to the traditional method such as the logistic regression, which reported better performance in different scenarios. It is a very robust method that has high flexibility for solving classification problems. Experimental studies using real world data sets have demonstrated that the classification and regression trees and neural networks outperform the traditional credit scoring models in terms of predictive accuracy and type II errors [10].

2.2 Supervised Learning

Machine learning is concerned with developing algorithms that learn from experience, build models of the environment from the acquired knowledge, and use these models for prediction. Experience, here, refers to past information available to the learner, the data [16]. Supervised learning is the machine learning task of learning a function that maps an input to an output. Each example is a pair consisting of an input object, the features or predictor variables $\mathbf{x} = (x_1, \ldots, x_p)$, and a desired output value, the target or response variable t.
More in detail, let define our dataset as $\mathcal{D} = \left\{(\mathbf{x}_i, t_i)^N \mid \mathbf{x}_i \in \mathbb{R}^p, t_i \in \mathbb{R}, N = |\mathcal{D}|\right\}$, the N observations of input-output pairs, and assume that exists a relationship between the features and the target defined as:

$$t = f(\mathbf{x}) + \epsilon \tag{2.4}$$

where ϵ is the error terms with $\mathbb{E}[\epsilon] = 0$ and $Var[\epsilon] = \sigma^2$.

The result of running the machine learning algorithm can be expressed as a function $y(\mathbf{x})$ which takes an input $\mathbf{x}$ and generates an output y, that predicts the target t. The precise form of the function $y(\mathbf{x})$ is determined during the training phase, also known as the learning phase, on the basis of the data. The inferred function $y(\mathbf{x})$ can be used for mapping new examples.

In a classification problem, like the one we will face in this thesis, t is a class variable which takes finite values $1, \ldots, K$, or in other words each input vector

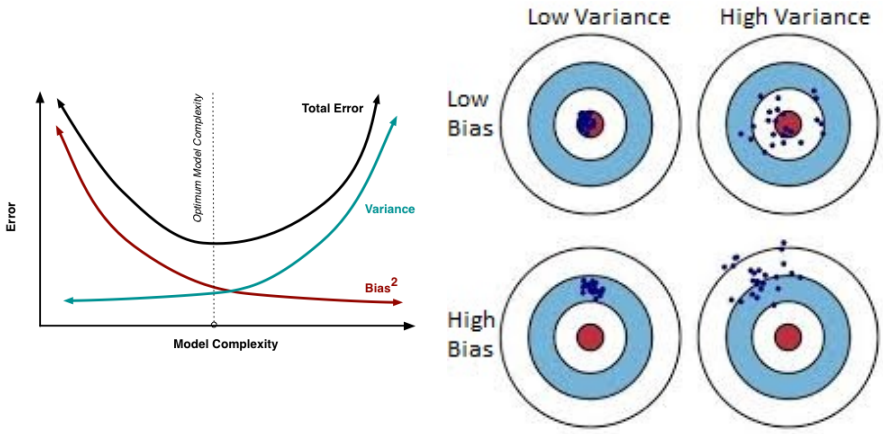


Figure 2.2: Bias-Variance tradeoff. On the left we can see that if one term reduce, the other will increase and vice versa. The best model is the one that reduce the total error, which is the sum of the bias, variance and the irreducible error. Essentially bias is how removed a model's predictions are from correctness, while variance is the degree to which these predictions vary between model iterations, on right side of the figure.

$\mathbf{x}_i$ is assigned to one of K discrete classes $\mathcal{C}_k$. To predict discrete class labels we need to use therefore non-linear function $y(\mathbf{x})$.

When the target variable is predicted, the main causes of difference in the actual and predicted values are **noise**, **variance** and **bias**, in literature this is known as *Bias-Variance tradeoff* (Figure 2.2). The expected square error on an unseen sample $\mathbf{x}$ between the target and the predicted value can be split in three different terms

$$\begin{aligned}\mathbb{E}\left[(t - y(\mathbf{x}))^2\right] &= \mathbb{E}\left[t^2 + y(\mathbf{x})^2 - 2ty(\mathbf{x})\right] \\ &= \underbrace{Var\left[t\right]}_{Noise} + \underbrace{Var\left[y(\mathbf{x})\right]}_{Variance} + \underbrace{\mathbb{E}\left[f(\mathbf{x}) - y(\mathbf{x})\right]}_{Bias^2}.\end{aligned} \tag{2.5}$$

From equation (2.5) we can identify:

- The noise σ, the irreducible error. It forms a lower bound on the expected error on unseen sample;
- The variance term, which measures the difference between what you learn from a particular data set $y(\mathbf{x})$ and what you expect to learn $\mathbb{E}\left[y(\mathbf{x})\right]$. It decreases with simpler model or with more samples;
- The bias, which measures the difference between the truth $f(\mathbf{x})$ and what you expect to learn $\mathbb{E}\left[y(\mathbf{x})\right]$. It decrease with more complex models.

In other words the bias is an error from erroneous assumptions in the learning algorithm: high value can cause an algorithm to miss the relevant relations between

features and target outputs (underfitting). Whereas the variance is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

It is possible, with ensemble methods, to reduce the variance in Eq. (2.5) without increasing the bias or, vice versa, to reduce the bias without increasing the variance. The noise is the constant term and it cannot be decreased: it is irreducible. The basic idea of ensemble models is to learn several models and combine them: many different predictors that are trying to predict the same target variable will perform a better job than any single predictor alone [20]. Ensembling techniques are further classified into Bagging and Boosting (Figure 2.3 and Figure 2.4).

Bagging average models by building many independent learners and combine them using some model averaging techniques. (e.g. weighted average, majority vote or normal average). Bagging reduces variance due to averaging and typically helps when it is applied to an overfitted base model or with high dependency on the training data. Bagging almost always improves performance when the learning algorithm is unstable. An algorithm is unstable if small changes in the training set cause large changes in the learned classifier (e.g. Decision trees, regression trees, linear regression, neural networks). It does not reduce the bias term. Example of bagging ensemble is **Random Forest models**.

Boosting sequentially train weak learners, each of them have a performance that on any train set is slightly better than chance prediction. This technique employs the logic in which the subsequent predictors learn from the mistakes of the previous predictors. Boosting reduces both bias and variance. By focusing on poor predictions and trying to model them better in the next iteration, it reduces bias; and by taking a weighted average of many weak models, the final model has lower variance than each of the weak models. Boosting, however, does not help to avoid overfitting. **Gradient Boosting** is an example of boosting algorithm.

Ensemble classifiers can deal with missing data and imbalance classes, performing better classification accuracy.

2.2.1 Gradient Tree Boosting

In this work one of the main models used is gradient tree boosting, an ensemble method which is widely used nowadays and is a winning algorithm for many challenges[3] such as the Netflix prize [2].

[3]https://github.com/dmlc/xgboost/tree/master/demo

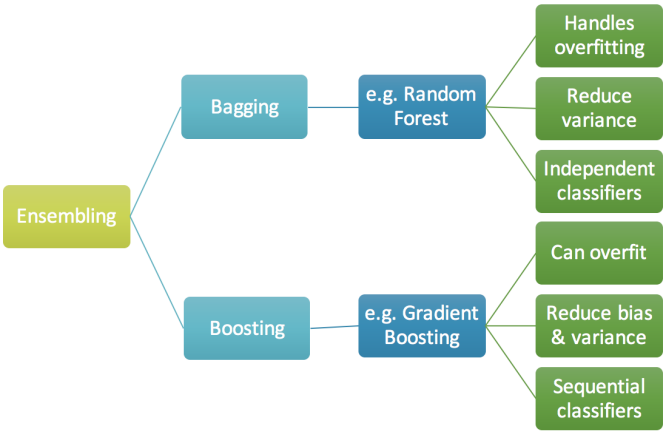


Figure 2.3: Bagging and boosting are ensemble techniques. They use voting and combine models of the same type. Random forest is an example of bagging, while gradient boosting is a boosting algorithm.

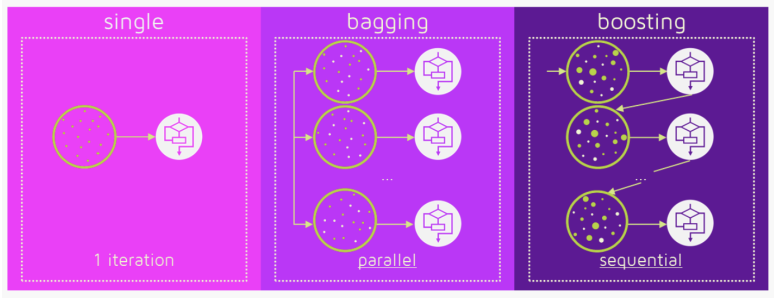


Figure 2.4: Bagging (independent models) & Boosting (sequential models). While the training stage is parallel for bagging (i.e., each model is built independently), boosting builds the new learner in a sequential way. In boosting algorithms each classifier is trained on data, taking into account the previous classifiers success. After each training step, the weights are redistributed. Misclassified data increases its weights to emphasise the most difficult cases. In this way, subsequent learners will focus on them during their training. **Reference**:
`https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/`

We review briefly gradient tree boosting model, a supervised algorithm for classification tasks. Gradient boosting involves three elements.

1. **Loss Function.** Its shape depends on the type of problem considered: in regression it may use a squared error ($L(\theta) = \sum_i (t_i - y_i)^2$) and in classification a logarithm loss ($L(\theta) = \sum_i [t_i \ln(1 + e^{-y_i}) + (1 - t_i) \ln(1 + e^{y_i})]$).

2. **Weak learner.** Decision trees are used as the weak learner in gradient boosting, which are constructed in a greedy manner. The splits points are chosen on the value of a score, like Gini, or to minimize the loss. It is common to constrain the weak learners in specific ways, such as a maximum number of layers, nodes, splits or leaf nodes.

3. **Additive model.** The weak learner are added one at a time, and the existing ones are not changed. A gradient descent procedure is used to minimize the loss when adding trees.

To find the best parameter θ we need to define the objective function that measures the performance of the model

$$obj(\theta) = L(\theta) + \Omega(\theta). \tag{2.6}$$

The first term is the Loss function, while the second one is the regularization term which controls the complexity of the model avoiding overfitting. In classification problems widely used are the logistic loss function, equation (2.7), and the binary crossentropy, in equation (2.8)

$$L(\theta) = \sum_{i=0}^{N} \left[t_i \log(1 + e^{-y_i}) + (1 - t_i) \log(1 + e^{y_i}) \right] \tag{2.7}$$

$$L(\theta) = -\frac{1}{N} \sum_{i=0}^{N} \left[t_i \log y_i + (1 - t_i) \log(1 - y_i) \right] \tag{2.8}$$

where y_i is the predicted class for t_i.

XGBoost, which model is tree ensembles, is a set of classification and regression trees (CART). In CART a score is associated with each of the leafs, and the predictions are obtained summing the score of multiple trees together.

Let us change the notation from here on, defining the target value as y and the prediction as $\hat{y}$, so that we do not get confused with the time step t of the algorithm.

In details, let $\mathcal{F} = \left\{ (f(\mathbf{x}) = w_{q(\mathbf{x})} | q : \mathbb{R}^m \to T, w \in \mathbb{R}^T \right\}$ be the set of all possible regression trees, q maps each example $\mathbf{x}$ to the index of trees, and T is the number of leaves of the tree. Tree ensemble model uses K additive functions to

predict the output, where each f_k corresponds to an independent tree structure q and leaf weights w, as shown in Figure 2.5.

The prediction of a data point $\mathbf{x}_i$ is given by

$$\hat{y}_i = \sum_{k=1}^{K} f_k(\mathbf{x}_i), \quad f_k \in \mathcal{F} \tag{2.9}$$

where f is the regression tree

$$f(\mathbf{x}_i) = w_{q(\mathbf{x}_i)}. \tag{2.10}$$

Therefore the objective function to optimize can be written as

$$obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.11}$$

Again, l is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i; whereas the second term Ω penalizes the complexity of the model. In XGBoost [6] the complexity is defined as

$$\Omega(f) = \gamma T + 1/2\lambda \|w\|^2. \tag{2.12}$$

Gradient boosting uses additive models: having fixed what it has been learned, it is added greedily a new tree f_t that most improves the model according to equation (2.11), one by one. The prediction value $\hat{y}_i^{(t)}$ at time step t is given by

$$\begin{aligned} \hat{y}_i^{(0)} &= 0 \\ \hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(\mathbf{x}_i) = \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i) \end{aligned} \tag{2.13}$$

The objective to optimize at time step t, considering the mean square error as the loss function and removing the constant term, becomes

$$obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.14a}$$

$$= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)\right) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.14b}$$

Second-order approximation can be used to quickly optimize the objective in the general setting

$$obj \simeq \sum_{i=1}^{n} \left[l(y_i - \hat{y}_i^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i) \right] + \Omega(f_k) + const \tag{2.15a}$$

$$\simeq \sum_{i=1}^{n} \left[g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{2.15b}$$

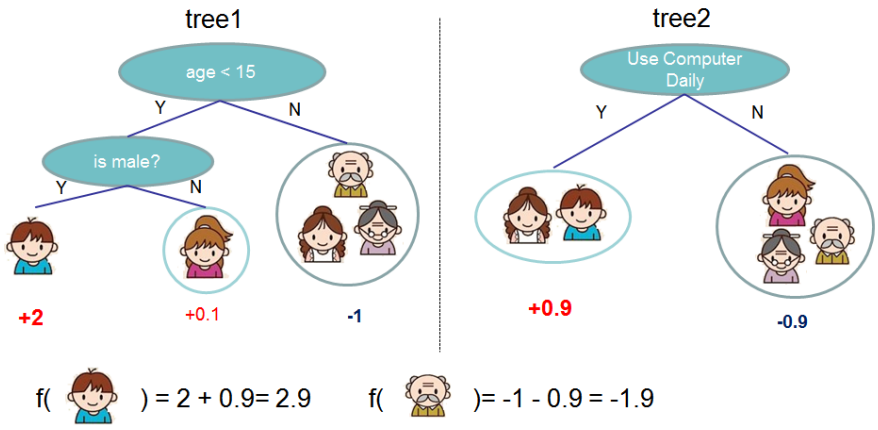


Figure 2.5: Tree Ensenble Model. The prediction scores of each individual tree are summed up to get the final score. The Figure is an example of regression, in which it is used 2 additive functions to predict the output [6].

where $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ are first and second order gradient statistics on the loss function. After reformalizing the tree model and using (2.10), we can rewrite equation (2.15b) as

$$\begin{aligned} obj^{(t)} &\simeq \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T \\ &= \sum_{j=1}^{T} [G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2] + \gamma T \end{aligned} \tag{2.16}$$

where $I_j = \{i | q(\mathbf{x}_i) = j\}$ is the set of data points assigned to the j-th leaf. Finally, the optimal weight w_j^* of leaf j for a given structure $q(\mathbf{x})$ and the best objective reduction can be computed by

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{2.17}$$

$$obj^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T. \tag{2.18}$$

Equation (2.18) measures how good is a tree and it is used as a scoring function: ideally we would enumerate all possible trees and select the best one. This is impossible in practice, so we iteratively adds branches one at the time if the score gain increases:

$$Gain = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma \tag{2.19}$$

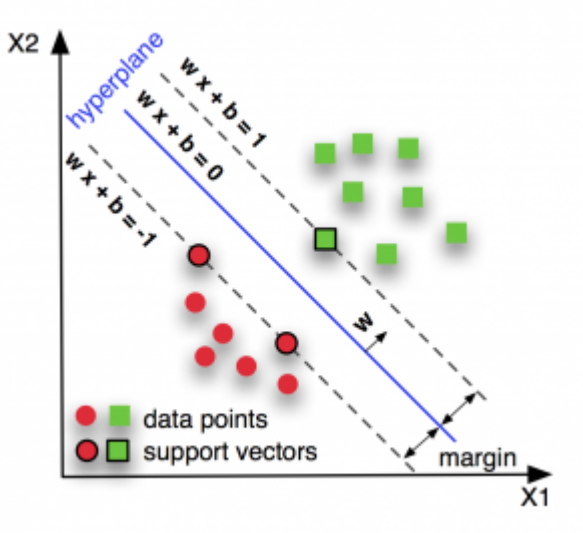


Figure 2.6: Margin and equations of the hyperplanes for a canonical maximum-margin hyperplane. The marginal hyperplanes are represented by dashed lines on the figure. Support vectors are the observations that lie on the maximum margin hyperplanes [3].

This formula[4] is usually used in practice for evaluating the split candidates, and it is very powerful and efficiently in practice.

XGboost[5] is an implementation of a scalable machine learning system for tree boosting, and aims to provide a scalable, portable and distributed gradient boosting (GBM, GBRT, GBDT) library. It is available as an open source package for many programming languages: C++, Java, Python, R. It is used for supervised learning problems, where the covariates $\mathbf{x}_i$ are used to predict the target variable y_i which may take continuous or categorical values. XGBoost runs more than ten times faster with respect for other existing solutions on a single machine and it scales well for billions of data [6, 5].

2.2.2 Support Vector Machines

A Support Vector Machine (SVM) is a discriminative supervised classifier formally defined by a separating hyperplane. In other words, given labelled training data, the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side [16].

An SVM model is a representation of the examples, or observations, as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New observations are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall, as we can see in Figure 2.6. In addition to performing linear

[4]The best split according this formula is founded by enumerating all possible splits (*exact greedy algorithm*). Other variants of exact greedy algorithm approximate the best split point and they are useful when data do not fit entirely into memory. For more details see [6].

[5]`https://github.com/dmlc/xgboost`

classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Support vector machines became a very popular algorithm some years ago for solving problems such as classification, regression, and novelty detection [3]. Some common applications[6] of SVM are: face detection, text and hypertext categorization, classification of images, bio-informatics [1].

An important property of support vector machines is that the determination of the model parameters corresponds to a convex optimization problem, and so any local solution is also a global optimum. The key concepts of support vector machines are [17]:

1. **Maximum margin hyperplane**: to find a linear classifier through optimization;
2. **Kernel trick**: to expand up from linear classifier to a non-linear one;
3. **Soft-margin**: to extend SVM to cases in which the data are not linearly separable.

The SVM originated from the Optical Separating Hyperplane (OSH) developed by Vapnik et al. in the 1960s was extended to a nonlinear classifier combined with kernels in the 1990s [19]. The SVM builds up a classifier that basically identifies two classes. It requires additional techniques such as a combination of multiple SVMs to build up a multi class classifier [13].

Support vector machines approach the problem through the concept of margin, which is defined to be the smallest distance between the decision boundary and any of the samples, and then the decision boundary is chosen to be the one for which the margin is maximized. SVM therefore faces the dual representation with kernel functions, which avoids having to work explicitly in feature space.

First, let us start with the assumption that the training data set is linearly separable. In this framework the training set consist in N input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$, with the corresponding target values $t_1, \ldots, t_N$, and new data points $\mathbf{x}$ are classified according to the sign of $y(\mathbf{x})$. We focus on a two-class classification problem in which the linear models are of the form

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{2.20}$$

where $\phi(\mathbf{x})$ is the basis function and denotes a fixed feature-space transformation. In models where there is a single input variable x, polynomial functions of the form $\phi_j(x) = x^j$ are widely used. There are many other possible choices for the basis functions, as we can see in Figure 2.7, for example the Gaussian basis function

$$\phi_j(x) = \exp\left\{-\frac{(x - \mu_j)^2}{2s^2}\right\} \tag{2.21}$$

[6] `https://data-flair.training/blogs/applications-of-svm/`

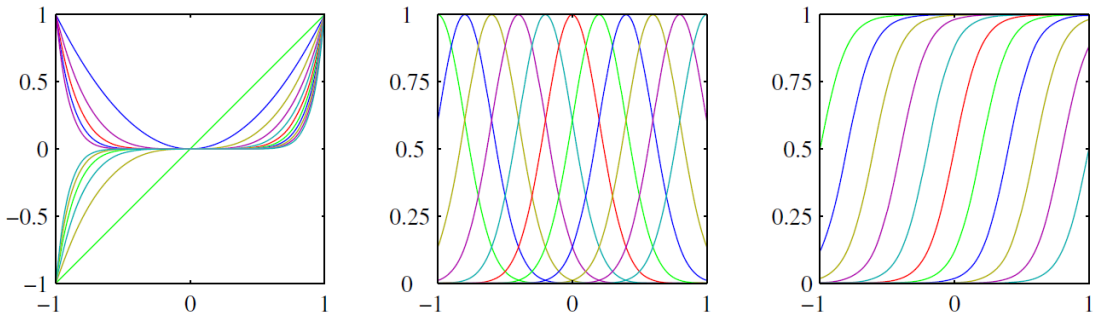


Figure 2.7: Example of basis functions, showing polynomials on the left, Gaussian of the form (2.21) *in the centre, and sigmoidal of the form* (2.22) *on the right.*

where μ_j govern the locations of the basis functions in input space, and the parameter s governs their spatial scale. Another possibility is the sigmoidal basis function of the form

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right) \tag{2.22}$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)}. \tag{2.23}$$

The hypothesis of data linearly separable means that exists at least one choice of the parameters $\mathbf{w}$ and b such that equation (2.20) satisfies $y(\mathbf{x}_n) > 0$ for points having $t_n = +1$ and $y(\mathbf{x}_n) < 0$ for points having $t_n = -1$, so at the end for all points in the training data it is satisfied $t_n y(\mathbf{x}_n) > 0$.

The decision surface is defined as $y(\mathbf{x}) = 0$ and the distance of a point $\mathbf{x}$ to it is given by

$$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \frac{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)}{\|\mathbf{w}\|}. \tag{2.24}$$

The margin is given by the perpendicular distance of the closest point $\mathbf{x}_n$, hence the maximum margin solution is founded by solving

$$\mathbf{w}^* = \arg\max_{\mathbf{w},b}\left\{\frac{1}{\|w\|}\min_n\left[t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)\right]\right\}. \tag{2.25}$$

Direct solution is complex, so this problem is converted into an equivalent one, easier to solve in which, fixed the margin, the weights are minimized [7]

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}} && \frac{1}{2}\|w\|^2 \\ &\text{subject to} && t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) \geq 1, \text{ for } n = 1, \ldots, N. \end{aligned} \tag{2.26}$$

[7] Note that rescaling $\mathbf{w} \to \kappa\mathbf{w}$ and $b \to \kappa b$ does not change the distance of any points to the decision surface. This extra freedom is used to set $t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) = 1$.

This is a quadratic programming problem in which we are trying to minimize a quadratic function subject to a set of linear inequality constraints, and can be solved using the Lagrangian multipliers $\alpha_n \geq 0$, with one multiplier α_n for each of the constrain in (2.26), giving the Lagrangian function:

$$L(\mathbf{x}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|w\|^2 - \alpha_n \left\{ t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1 \right\}. \tag{2.27}$$

Setting the derivatives of $L(\mathbf{x}, b, \boldsymbol{\alpha})$ equal to zero with respect to $\mathbf{x}$ and b we obtain the following condition

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n t_n \phi(\mathbf{x}_n) \tag{2.28}$$

$$0 = \sum_{n=1}^{N} \alpha_n t_n \tag{2.29}$$

and at the end of the computation, after having eliminated $\mathbf{x}$ and b from the Lagrangian function we obtain the dual representation of the maximum margin problem of the form

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{maximize} \quad & L(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \\ \text{subject to} \quad & \alpha_n \geq 0, \text{ for } n = 1, \ldots, N \\ & \sum_{n=1}^{N} \alpha_n t_n = 0 \end{aligned} \tag{2.30}$$

where the kernel function is defined by $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}')$.

All the points that satisfy $\alpha_n > 0$ are called support vectors and they lie on the maximum margin hyperplanes in feature space[8]: they satisfy $t_n y(\mathbf{x}_n) = 1$, as illustrated in Figure 2.6. This property is central in this setting because a significant proportion of data points can be discarded and only the support vectors are retained.

In order to classify new data points using the model just trained, we express (2.20) in terms of $\boldsymbol{\alpha}$ and the kernel function, by substituting $\mathbf{w}$ using (2.28) to give

$$y(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n t_n k(\mathbf{x_n}, \mathbf{x_m}) + b. \tag{2.31}$$

[8]Every data point satisfy either $\alpha = 0$ or $t_n y(\mathbf{x}_n) = 1$. This is a consequence of the KKT conditions of the constrained optimization problem (2.30).

Any data point for which $\alpha = 0$ will not appear in (2.31) and then plays no role in making predictions for new data points. Thus we can rewrite (2.31) in a simpler form

$$y(\mathbf{x}) = \sum_{n \in S} \alpha_n t_n k(\mathbf{x_n}, \mathbf{x_m}) + b. \tag{2.32}$$

where S denotes the set of indices of support vectors and N_S is their total number. By noting that any support vector x_n satisfies $t_n y(\mathbf{x}_n) = 1$, we can then determine the threshold parameter b substituting it in equation (2.32) to give

$$b = \frac{1}{N_S} \sum_{n \in S} \left(t_n - \sum_{m \in S} \alpha_m t_m k(\mathbf{x_n}, \mathbf{x_m}) \right) \tag{2.33}$$

where N_S is the total number of support vectors.

Overlapping class and soft margin

The assumption that the training data is linearly separable in the feature space $\phi(\mathbf{x})$ is often not truthful, and this can lead to poor performance. We therefore allow some of the training data points to be misclassified, introducing slack variables ξ_n, where $n = 1, \ldots, N$, with one slack variable for each training data point. They are defined by

$$\begin{aligned} \xi_n &= 0 && \text{for points correctly classified} \\ \xi_n &= |t_n - y(\mathbf{x}_n)| && \text{otherwise.} \end{aligned} \tag{2.34}$$

Thus, on the decision boundary $y(\mathbf{x}_n) = 0$ we have $\xi_n = 1$, points for which $0 \leq \xi_n \leq 1$ lie inside the margin but are on the correct side of the decision boundary, and points with $\xi_n \geq 1$ are misclassified (Figure 2.8).

Our goal is now to optimize the margin while soft penalizing points that lie on the wrong side of the margin boundary. Thus in this setting the problem becomes

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2} \|w\|^2 + C \sum_{n=1}^{N} \xi_n \\ \text{subject to} \quad & t_n(\mathbf{w}^T \phi(\mathbf{x}_n) + b) \geq 1 - \xi_n, \quad \text{for } n = 1, \ldots, N \\ & \xi_n \geq 0 \qquad\qquad\qquad\qquad\quad \text{for } n = 1, \ldots, N \end{aligned} \tag{2.35}$$

where $C > 0$ controls the trade-off between the slack variable penalty and the margin, then ultimately the bias-variance trade-off, and it is chosen by cross validation. Again, after having introduced the Lagrangian multipliers and done all

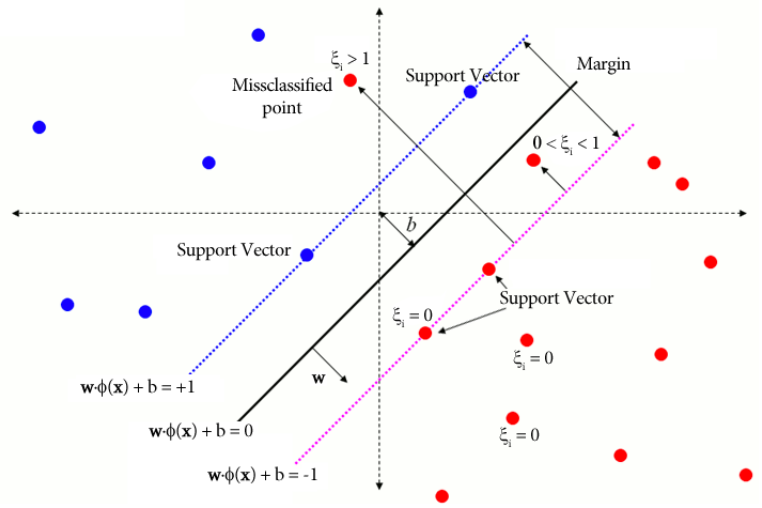


Figure 2.8: The purpose of slack variables explained through a simple sketch. The respective slack variable ξ_i is zero if the observation is located on the correct side of the hyperplane and nothing is changed. The ξ_i is greater than zero if its distance from the separating hyperplane is lower than the distance of support vectors. **Reference**: `http://svm.michalhaltuf.cz/support-vector-machines`.

the computation like before, we end up to the following maximum margin problem

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{maximize} \quad & L(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \\ \text{subject to} \quad & 0 \leq \alpha_n \leq C, \text{ for } n = 1, \ldots, N \\ & \sum_{n=1}^{N} \alpha_n t_n = 0 \end{aligned} \tag{2.36}$$

which is identical to the separable case, except for the constrains. As before, the data points that satisfy $\alpha_n = 0$ do not contribute to the predictive model, while the remaining ones are the support vectors. If $\alpha_n < C$ the point lies on the margin, whereas if $\alpha_n = C$ it lies inside the margin and can be either correctly classified ($\xi_n \leq 1$) or misclassified ($\xi_n > 1$). Finally, all the data points must satisfy $t_n y(\mathbf{x}_n) = 1 - \xi_n$.

Kernel trick

Above we have already meet the kernel functions without explain what they were. A kernel is a similarity function that we, as the domain experts, provide to a machine learning algorithm. Consider the typical machine learning pipeline: we take our dataset, compute features through some preprocessing steps, and we feed these feature vectors and labels into a learning algorithm. Kernels offer an alternative. Instead of defining a slew of features, we define a single kernel

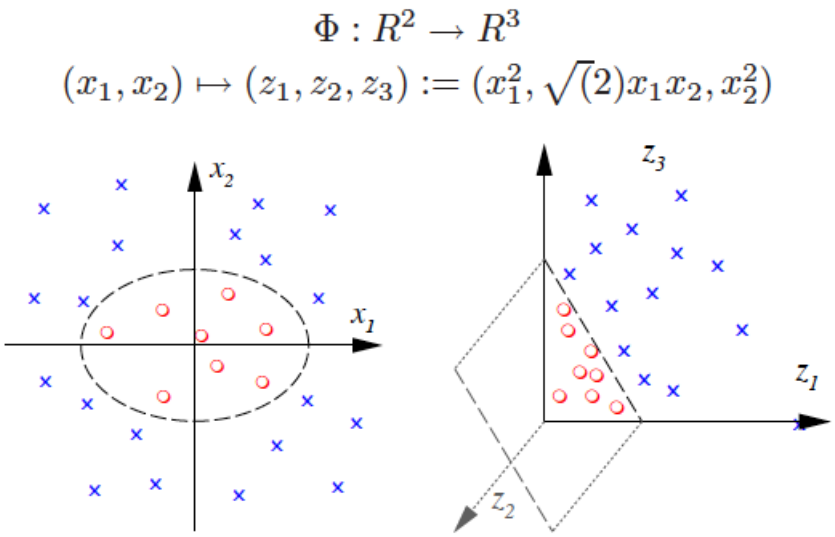


Figure 2.9: Often data is far from linear and the datasets are not separable. To handle this situation, kernels are used to non-linearly map the input data to a higher dimensional space. The dimensionality of $\phi(\mathbf{x})$ can be very large, while the kernel function is simply a dot product and hence is a scalar value. **Reference**: `https://courses.cs.ut.ee/2011/graphmining/Main/KernelMethodsForGraphs`.

function to compute similarity between observations, and we provide it to the learning algorithm, together with the observations and labels.

Support Vector Machines are an application of kernel methods, a class of algorithms for pattern analysis. For many algorithms that solve these tasks, the data in raw representation have to be explicitly transformed into feature vector representation via a feature space mapping $\phi(\mathbf{x})$. In contrast, kernel methods require only a user-specified kernel, a similarity function over pairs of data points in raw representation. Furthermore, many linear parametric models based on $\phi(\mathbf{x})$ can be recast into an equivalent dual representation where the kernel function is given by the relation

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}'). \tag{2.37}$$

The concept of a kernel formulated as an inner product in a feature space allows to build extensions of many algorithms by making use of the *kernel trick*, Figure 2.9. The general idea is that, if we have an algorithm where the input vector $\mathbf{x}$ enters only in the form of scalar products, then we can replace the product with some other choice of kernel.

In order to use kernel substitution we need to be able to construct valid kernel functions. One approach is to choose a feature space mapping $\phi(\mathbf{x})$ and then find the corresponding kernel by using (2.37), as illustrated in Figure 2.10. An alternative approach is to construct the kernel directly, which should correspond

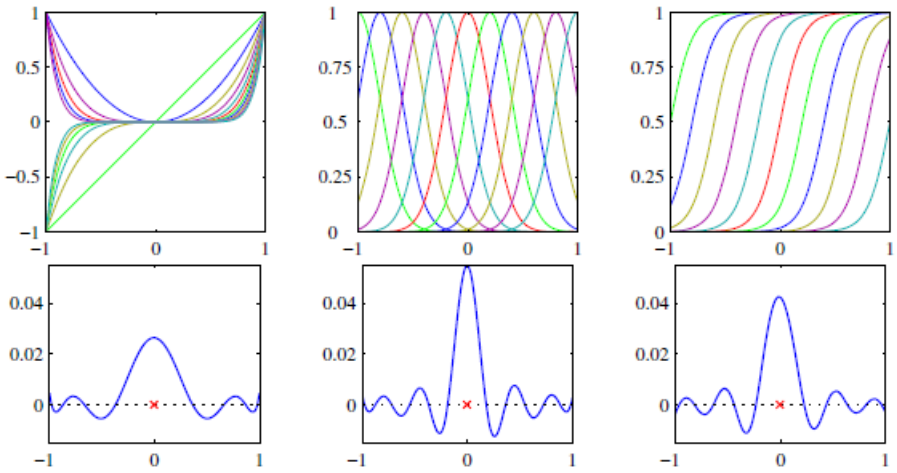


Figure 2.10: Construction of kernel functions starting form a corresponding set of basis functions. The upper plot shows the polynomials, on the left column, Gaussian, centre column, and sigmoidal, right column. The lower plot shows the corresponding kernel function $k(x, x')$ defined by (2.37), and plotted as a function of x for $x' = 0$.

to a scalar product in some feature space which could be also infinite dimensional, avoiding to construct the function $\phi(\mathbf{x})$ directly.

A necessary and sufficient condition for a function $k(\mathbf{x}, \mathbf{x}')$ to be a valid kernel is that the Gram matrix K, whose elements are given by $k(\mathbf{x}_n, \mathbf{x}_m)$, should be positive semidefinite for all possible choices of the set $\{\mathbf{x}_n\}$.

Another powerful technique is to build new kernels out of simpler ones as building blocks. Suppose $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$ are valid kernels, then the following new kernels are also valid:

- $k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}')$;
- $k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}')$;
- $k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}'))$;
- $k(\mathbf{x}, \mathbf{x}') = \exp(k_1(\mathbf{x}, \mathbf{x}'))$;
- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$;
- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$;
- $k(\mathbf{x}, \mathbf{x}') = k_3(\phi(\mathbf{x}), \phi(\mathbf{x}'))$;

where $c < 0$ is a constant, $f(\cdot)$ is any function, $q(\cdot)$ is a polynomial with non-negative coefficients, $\phi(\mathbf{x}')$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$ and $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$. A simple valid kernel is the linear one $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$, from here when can build almost any complex valid kernel.

In this thesis we will use, in Chapter 5 when we build the SVM model for the default classification of the companies, both the linear and the Gaussian kernels.

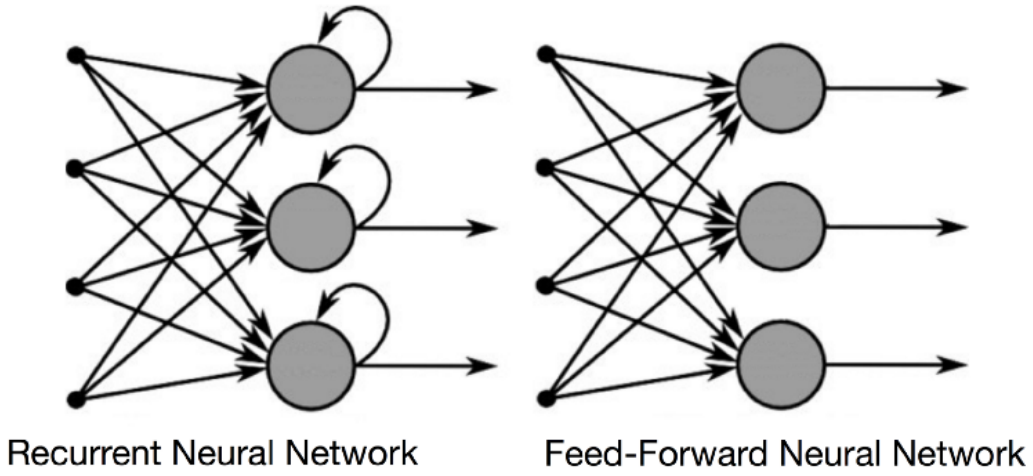


Figure 2.11: Difference in the information flow between a recurrent neural network and a feed-forward neural network.

2.2.3 Neural Networks

A neural network (NNW), also known as net, is a mathematical representation inspired by the human brain and its ability to adapt on the basis of the inflow of new information. Mathematically, NNW is a non-linear optimization tool [8, 12]. The most successful model of this type in the context of pattern recognition is the feed-forward neural network, also known as the multilayer perceptron (MLP), and it is especially suitable for classification. The network consists of one input layer, one or more hidden layers and one output layer, each consisting of several neurons. The goal of a feed-forward network is to approximate the function f, as defined in equation (2.4), which represents the relationship between the input variables and the target [9].

One of the major drawback of NNWs is their lack of explanation capability. While they can achieve a high prediction accuracy rate, the reasoning behind why and how the decision was reached is not available.

In the classification problem, $f(\mathbf{x})$ maps an input $\mathbf{x}$ to a category t. A feed-forward network defines a mapping $y = f(\mathbf{x}; \theta)$ and learns the value of the parameters θ that results in the best function approximation. These models are called feed-forward because information flows through the function being evaluated from $\mathbf{x}$, through the intermediate computations used to define f, and finally to the output y. There are no feedback connections in which outputs of the model are fed back into itself. When feed-forward neural networks are extended to include feedback connections, they are called recurrent neural networks, as we can see in

Figure 2.11.

Feed-forward neural networks are called networks because they are typically represented by composing together many different functions. The model is associated with a directed acyclic graph describing how the functions are composed together. For example, we might have three functions $f^{(1)}$, $f^{(2)}$ and $f^{(3)}$ connected in a chain, to form $f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x})))$. These chain structures are the most commonly used in neural networks. In this case, $f^{(1)}$ is called the first layer of the network or the input layers, $f^{(2)}$ is called the second layer, and so on. The overall length of the chain gives the depth of the model. The final layer of a feed-forward network is called the output layer. All the layers between the input and the output are called hidden layers.

Let us consider more in detail the functional form of the network model: we consider a single-hidden-layer network for simplicity without any loss of generality. The linear models for regression and classification are based on linear combinations of fixed nonlinear basis functions $\phi_j(\mathbf{x})$ and take the form

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right) \tag{2.38}$$

where $f(\cdot)$ is a nonlinear activation function in the case of classification and is the identity in the case of regression. Neural networks use basis functions that follow the same form as (2.38), so that each basis function is itself a nonlinear function of a linear combination of the inputs, where the coefficients in the linear combination are adaptive parameters. We allow therefore the basis functions $\phi_j(\mathbf{x})$ to depend on parameters which can be adjusted, along with the coefficients $\{w_j\}$, during training. The basic neural network model, therefore, can be described as a series of functional transformations.

First, we construct M linear combinations of the input variables $x_1, \ldots, x_D$ in the form

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \qquad j = 1, \ldots, M \tag{2.39}$$

where the superscript (1) indicates that the corresponding parameters are in the first layer of the network. The parameters $w_{ji}^{(1)}$ are the weights and the parameters $w_{j0}^{(1)}$ are the biases. The quantities a_j are known as activations. Each of them is then transformed using a differentiable, nonlinear activation function $h(\cdot)$ to give

$$z_j = h(a_j). \tag{2.40}$$

These quantities correspond to the outputs of the basis functions in (2.38) that, in the context of neural networks, are called hidden units.

The design of hidden units is an extremely active area of research and does not yet have many definitive guiding theoretical principles. Rectified linear units

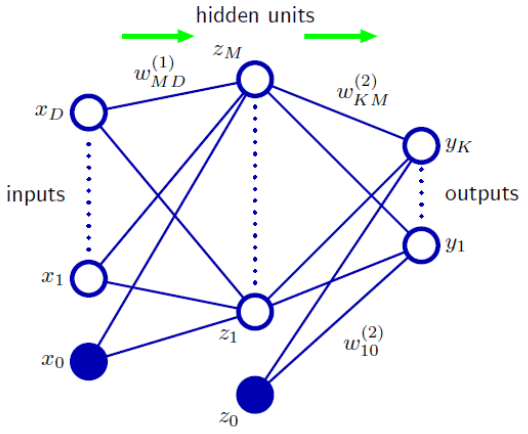


Figure 2.12: Network diagram for a single-hidden-layer network. The input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes, in which the bias parameters are denoted by links coming from additional input and hidden variables x_0 and z_0. Arrows denote the direction of information flow through the network during forward propagation.

(ReLU) are an excellent default choice of hidden unit. Many other types of hidden units are available. Some of the hidden units used in practise are not actually differentiable at all input points. For example, the rectified linear function $h(a) = \max\{0, a\}$ is not differentiable at $a = 0$. In practice, the training algorithm still performs well enough for these models.

The output of the hidden units $\{z_j\}$ in (2.40) are again linearly combined to give output unit activations

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \qquad k = 1, \ldots, K \tag{2.41}$$

where K is the total number of outputs. This transformation corresponds to the second layer of the network, and again the $w_{k0}^{(2)}$ are bias parameters. Finally, the output unit activations are transformed using an appropriate activation function to give a set of network outputs y_k.

The choice of activation function is determined by the nature of the data. Thus for standard regression problems, the activation function is the identity so that $y_k = a_k$. Similarly, for multiple binary classification problems, each output unit activation is transformed using a logistic sigmoid function so that

$$y_k = \sigma(a_k) \tag{2.42}$$

where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}. \tag{2.43}$$

We can combine these various stages to give the overall network function that, for sigmoidal output unit activation functions, takes the form

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right) \tag{2.44}$$

or in a more compact way

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right) \tag{2.45}$$

where the bias parameter in (2.39) is absorbed into the set of weight parameters by defining an additional input variable x_0 whose value is clamped at $x_0 = 1$, and similarly the second-layer bias is absorbed into the second-layer weights. Figure 2.12 shows the two stages of processing.

Error Backpropagation

The weight vector w is founded by minimizing a chosen function error $E(w)$, or loss function

$$\nabla E(\mathbf{w}) = 0. \tag{2.46}$$

The error function typically has a highly nonlinear dependence on the weights and bias parameters, so there is no hope of finding an analytical solution to the equation (2.46), and then we resort to iterative numerical procedures. One of the simplest approach to find w is to use gradient descent optimization

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)}) \tag{2.47}$$

where the parameter $\eta > 0$ is the learning rate. This algorithm makes use of gradient information and therefore require that, after each update, the value of $\nabla E(\mathbf{w})$ is evaluated at the new weight vector $\mathbf{w}^{(\tau+1)}$.

An efficient technique for evaluating the gradient of an error function $E(\mathbf{w})$ for a feed-forward neural network is the *error backpropagation*. First, we note that many error functions of practical interest, for instance those defined in (2.7), comprise a sum of terms, one for each data point in the training set, so that

$$E(\mathbf{w}) = \sum_{i=1}^{N} E_n(\mathbf{w}) \tag{2.48}$$

therefore is sufficient to consider the problem of evaluating $\nabla E_n(\mathbf{w})$ for one such term in the error function.

In a general feed-forward network, each unit computes a weighted sum of its inputs of the form

$$a_j = \sum_j w_{ji} z_i \tag{2.49}$$

where z_i is the activation of a unit, or input, that sends a connection to unit j, w_{ji} is the weight associated with that connection, and the bias is included in the summation. The sum in (2.49) is transformed by a nonlinear activation function $h(\cdot)$ to give the activation z_j of unit j in the form

$$z_j = h(a_j). \tag{2.50}$$

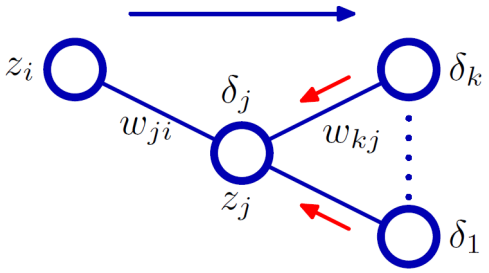


Figure 2.13: Illustration of the calculation of δ_j for hidden unit j by backpropagation of the δ's from those units k to which unit j sends connections.

For each pattern in the training set, we shall suppose that we have supplied the corresponding input vector to the network and calculated the activations of all of the hidden and output units in the network by successive application of (2.49) and (2.50). This process is often called forward propagation.

Now we evaluate the derivative of E_n with respect to a weight w_{ji}. First, we note that E_n depends on the weight w_{ji} only via the summed input a_j to unit j. Applying the chain rule for partial derivatives we have

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}}. \tag{2.51}$$

Using (2.49) we can write

$$\frac{\partial a_j}{\partial w_{ji}} = z_i \tag{2.52}$$

and introducing the notation

$$\delta_j = \frac{\partial E_n}{\partial a_j} \tag{2.53}$$

we obtain

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i. \tag{2.54}$$

Thus, in order to evaluate the derivatives, we need only to calculate the value of δ_j for each hidden and output unit in the network, and then apply (2.54).

For the output units, we have

$$\delta_k = y_k - t_k \tag{2.55}$$

and for evaluating the δ's for hidden units, we again make use of the chain rule for partial derivatives

$$\delta_j = \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j} \tag{2.56}$$

where the sum runs over all units k to which unit j sends connections. After having substitute the definition of δ and having made some calculation we end up with the *backpropagation* formula

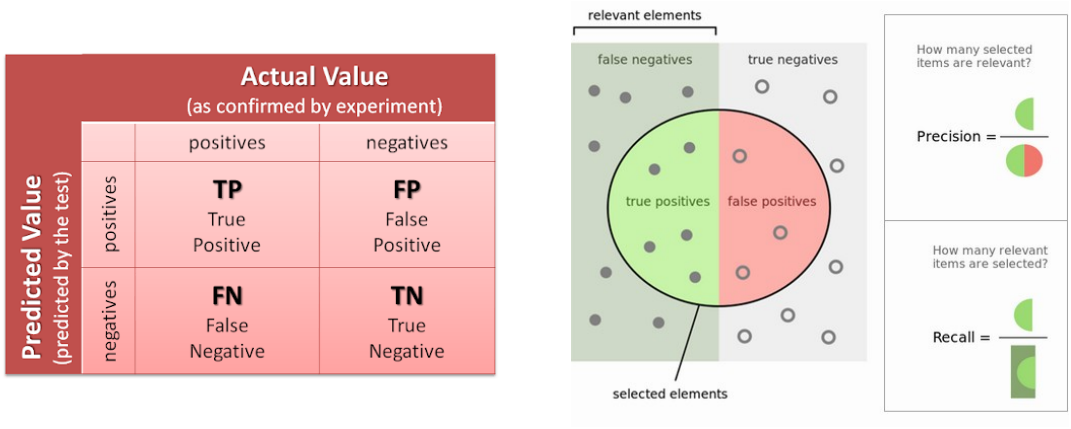


Figure 2.14: Confusion Matrix in a two classes problem. It is a specific table layout that allows visualization of the performance of an algorithm

$$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k \tag{2.57}$$

which tells us that the value of δ for a particular hidden unit can be obtained by propagating the δ's backwards from units higher up in the network, as illustrated in Figure 2.13.

2.2.4 Performance Evaluation

For evaluating the performance of a given classifier, several metrics are available to measures specific aspects of the model. Most of them are computed from the confusion matrix, which summarizes the prediction results on a classification task. In practice, a binary classifier, such as the one developed in this thesis, can make two types of errors: it can incorrectly assign an individual who defaults to the no default category, or it can incorrectly assign an individual who does not default to the default category. It is often of interest to determine which of these two types of errors are being made, and a confusion matrix is a convenient way to display this information. This matrix can be used for 2-class problems where it is very easy to understand, and it can easily be applied to problems with 3 or more class values, by adding more rows and columns to the confusion matrix.

In Figure 2.14 we can see a confusion matrix for 2-class problem. We can assign the event row and column as "positive" and the no-event row as "negative". The top row of the table corresponds to samples predicted to be events, $\hat{y} = 1$. Some are predicted correctly (the true positives, or TP) while others are inaccurately classified (false positives or FP). Similarly, the second row contains the predicted negatives with true negatives (TN) and false negatives (FN).

From those we can compute the most commonly used measures in a classification problem, and, following, it is reported their formula in case of two classes

problem in which they assume a simple form.

- **accuracy:** percentage of data correctly classified ($a = \frac{TP+TN}{N}$);
- **precision:** percentage of positive classifications that are correct ($p = \frac{TP}{TP+FP}$)
- **recall:** percentage of positive elements that have been classified as positive ($r = \frac{TP}{TP+FN}$);
- **F1-measure:** harmonic mean of precision and recall ($F_1 = \frac{2pr}{p+r}$).

Often, a classifier output a score value for the positive class for each point. Typically, a binary classifier chooses some positive score threshold ρ, and classifies all points with score above ρ as positive, with the remaining points classified as negative. However, such a threshold is likely to be somewhat arbitrary.

The measures enumerated before depend on the threshold chosen in the classifier, because it changes the number of data classified as positive or negative. To choose the best threshold ρ, and to compare also different models, normally the ROC curve is analysed when there are two classes. To draw it the true positive rate $\left(TPR = \frac{TP}{TP+FN}\right)$, equivalent to *sensitivity*, and false positive rate $\left(FPR = \frac{FP}{FP+TN}\right)$, equivalent to $1 - specifity$, are needed, as function of some classifier parameter, which usually is the threshold. Each prediction result or instance of a confusion matrix represents one point in the ROC space. As per Figure 2.15, the perfect prediction correspond to the point of coordinates (0,1) representing 100% sensitivity (no false negatives) and 100% specificity (no false positives), while the random guess would give a point along a diagonal line. The diagonal divides the ROC space: points above it represent good classification results, whereas points below the line are bad results (worse than random). Note that the output of a consistently bad predictor could simply be inverted to obtain a good one.

A ROC curve is the most commonly used way to visualize the performance of a binary classifier; from ROC it can be extrapolated AUC (area under the curve), which summarizes the performance in a single number. It is essentially the probability that the model will classify a positive example extracted higher than a negative one. A strength of the ROC is that it is insensitive to unbalanced classes, as the problem we are going to analyze in the present thesis [3, 21], therefore the area under the curve will be used for measuring the model performances.

Most real-world classification problems display some level of class imbalance, which is when each class does not make up an equal portion of the data-set. For example, suppose we have two classes – A and B. Class A is 90% of our data-set and class B is the other 10%, but we are most interested in identifying instances of class B. We can reach an accuracy of 90% by simply predicting class A every time, but this provides a useless classifier for our intended use case. Instead, a properly calibrated method may achieve a lower accuracy, but, for example, would

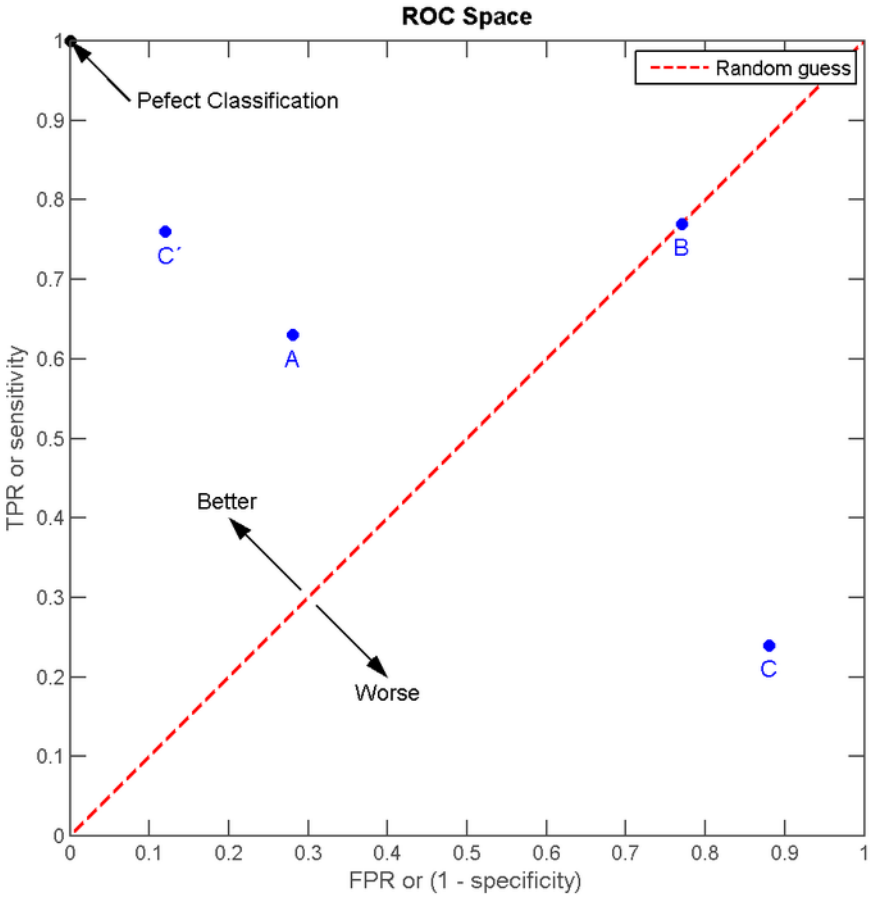


Figure 2.15: The ROC space and the plot of the four classifiers. All points that lie in the diagonal red line are the the random classifier, like B; above it there are good classifiers, whereas under the line we have bad results. Note that a bad classifier, like C, can become good if we invert the predicted class, C'. In this case the resulting classifier is better than A.

have a substantially higher F1-measure. One of the simplest ways to address the class imbalance, and that it will be used in the present work, is to simply provide a weight for each class, which specifies the cost of misclassifying an instance of one class to another, and places more emphasis on the minority classes such that the end result is a classifier which can learn equally from all classes. A common scheme for this is to have the cost equal to the inverse of the proportion of the data-set that the class makes up. This increases the penalization as the class size decreases.

2.2.5 Best method

In general, there is no overall "best" method. What is the best will depend on the details of the problem, the data structure, the characteristics used, the extent to which it is possible to separate the classes by using those characteristics and the objective of the classification (overall misclassification rate, cost-weighted misclassification rate, bad risk rate among those accepted, some measure of profitability, etc.). The various methods are often very comparable in results. Often, there is no superior method for diverse data sets.

Classification accuracy is only one aspect of performance. Others include the speed of classification, the speed with which a scorecard can be revised and the ease of understanding of the classification method and why it has reached its conclusion. As far as the speed of classification goes, an instant decision is much more appealing to a potential borrower than is having to wait for several days.

Classification methods which are easy to understand, such as regression, nearest neighbours and tree based methods, are much more appealing, both to users and to clients, than methods that are essentially black boxes, as neural networks or support vector machines with non-linear kernel.

In general, if one has a good understanding of the data and the problem, then methods that makes use of this understanding might be expected to perform better. In credit scoring, where people have been constructing scorecards on similar data for several decades, there is a solid understanding. This might go some way towards explaining why neural networks, support vectors machines and other algorithms that are not easy to understand have not been adopted as regular production systems in this sector, despite the fact that banks have been experimenting with them for several years [4].

2.3 Principal Component Analysis

Principal component analysis (PCA) is fundamentally a dimensionality reduction algorithm, but it can also be useful as a tool for visualization, for noise filtering, and for feature extraction and engineering. It is concerned with explaining the variance-covariance structure of a set of variables through a linear combination of

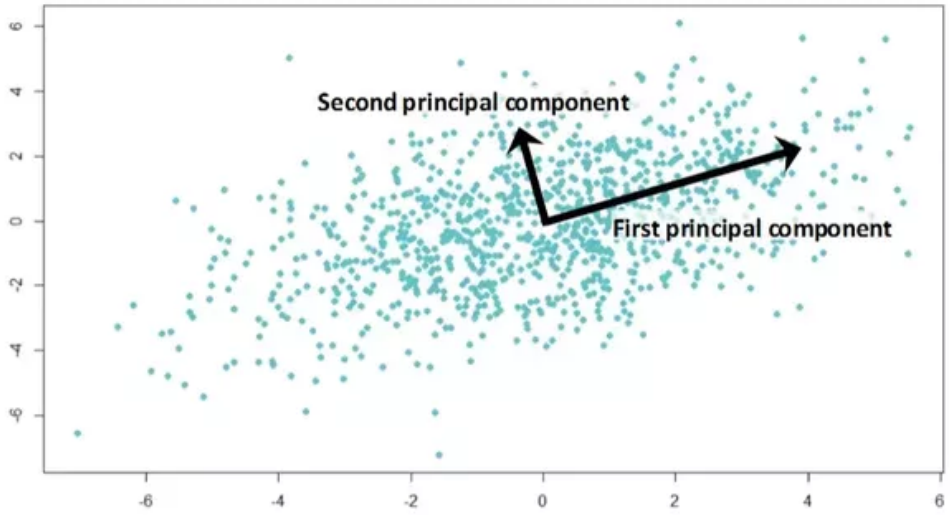


Figure 2.16: A principal component analysis ca be considered as a rotation of the axes of the original variable coordinate system to new orthogonal axes, such that the new axes coincide with direction of maximum variation of the original observations.

them [11]. It is a technique that seeks a r-dimensional basis that best captures the variance in the data. The direction with the largest projected variance is called the first principal component, the orthogonal direction that captures the second largest projected variance is called the second principal component, and so on; the direction of the first principal component is also the one that minimizes the mean squared error.

Algebraically, principal components are particular linear combination of the p random features $X_1, \ldots, X_p$. Geometrically, these combination represent the selection of a new coordinate system obtaining by rotating the original system with $X_1, \ldots, X_p$ as the coordinate axes: they represent the directions with maximum variability (Figure 2.16).

Principal components depend solely on the covariance matrix Σ of $X_1, \ldots, X_p$. Let the random variable $X' = [X_1, \ldots, X_p]$ have the covariance matrix Σ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$ and their respective eigenvectors $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_p$. Consider the linear combination

$$\begin{aligned} Y_1 &= \mathbf{a}_1'\mathbf{X} \\ Y_2 &= \mathbf{a}_2'\mathbf{X} \\ &\vdots \\ Y_p &= \mathbf{a}_p'\mathbf{X}, \end{aligned} \tag{2.58}$$

and their variance and covariance are

$$Var(Y_i) = \mathbf{a}_i' \Sigma \mathbf{a}_i \quad i = 1, \ldots, p \tag{2.59}$$

$$Cov(Y_i, Y_k) = \mathbf{a}_i' \Sigma \mathbf{a}_k \quad i, k = 1, \ldots, p. \tag{2.60}$$

The principal components are those uncorrelated linear combination of $Y_1, \ldots, Y_p$, whose variance in (2.59) are as large as possible. The first principal component Y_1 is the linear combination of with maximum variance; the second principal component Y_2 is then the linear combination that has maximum variance, and is orthogonal to Y_1, i.e. $Cov(Y_1, Y_2) = 0$, and so on.

It turns out that the coefficients of the principal component Y_i are the elements of the eigenvector $\mathbf{e}_i$, i.e. $\mathbf{a}_i = \mathbf{e}_i$, and its variance is equal to the $i-th$ eigenvalue

$$Var(Y_i) = Var(\mathbf{e}_i' \mathbf{X}) = \lambda_i. \tag{2.61}$$

The proportion of variation explained by the $i-th$ principal component is then defined to be the eigenvalue for that component divided by the sum of the eigenvalues. In other words, the $i-th$ principal component explains the following proportion of the total variation

$$\frac{\lambda_i}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}. \tag{2.62}$$

A related quantity is the proportion of variation explained by the first k principal components. This would be the sum of the first k eigenvalues divided by its total variation

$$\frac{\lambda_i + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}. \tag{2.63}$$

If the proportion of variation explained by the first k principal components is large then not much information is lost by considering only the first k principal components.

There is always the question of how many components to retain, and there is no definitive answer to this question. To determine the appropriate number of components, we look for an elbow (bend) in the plot of the proportion of variation. The number of components is taken to be the point at which the remaining proportion of variance are relatively small and all about the same size, or equivalently as we can see in Figure 2.17, the cumulative variance is above some threshold, like for example 80% or 90%. Such dimensionality reduction can be a very useful step for visualising and processing high-dimensional datasets, while still retaining as much of the variance in the dataset as possible. For example, selecting $k = 2$ and keeping only the first two principal components finds the two-dimensional plane through the high-dimensional dataset in which the data is most spread out.

PCA is effected by scale so it is needed to standardize the data before applying it: one approach is to normalize each feature to have zero mean and unit variance.

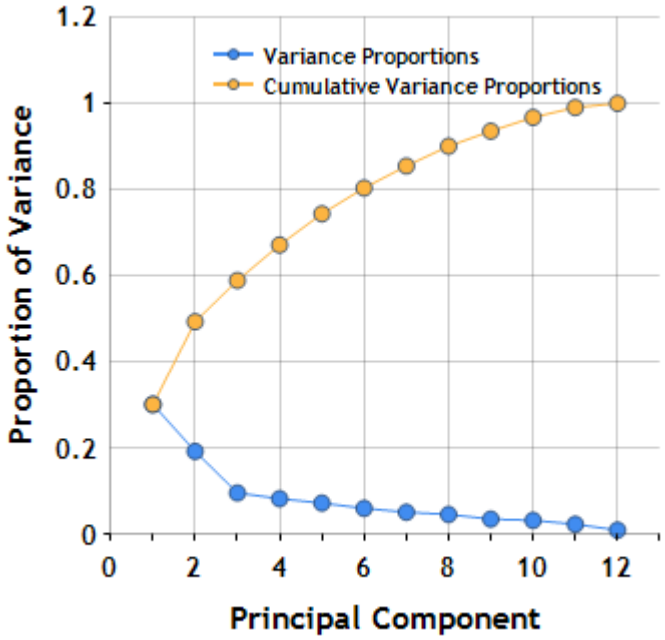


Figure 2.17: Typically, principal component analysis is used as dimension reduction. In the plot we can see that the first two principal components explain about 50% of the variance with respect to the original 12 features, and keeping 6 components we explain 80% of the variance. In this case, we can take the first 6 principal components for dimensionality reduction. **Reference**: `https://www.centerspace.net/clustering-analysis-part-i-principal-component-analysis-pca`.

Chapter 3

Financial Dataset Description

The financial dataset used in this thesis contains information regarding Italian companies registered at national Companies House (Camera di Commercio). The dataset is divided into several files: the company registry, the balance sheets, and the details of the exponents. Each company is identified by a unique index named RK, and it is labelled as Bad or Good based on whether it defaulted or not. The current Chapter is structured as following: Section 3.1 describes in details the financial dataset and the feature engineering approaches developed. Section 3.2 presents the results of principal component analysis.

3.1 Dataset Description

The dataset contains a total of 137143 labelled companies; the labelling describes whether the company is in default or not. They were extracted on December 2017: the companies failed before that date are labelled as bad; while the others are considered good. However, among the latter there could be cases of companies recently defaulted but not yet communicated or updated within Camera di Commercio. The data contains 5436 bad (defaulted) companies, 4% , while the other are good, as we can see in Figure 3.2. Based on that, the dataset appears imbalanced where Good samples are far more numerous compared to Bad. This peculiarity requires to be taken into consideration when developing our predictive algorithm. To handle this problem we adjust the class weight so the minority class gains in importance because its errors are considered more costly than those of the other class.

RK	DENOMINAZIONE	DATA INIZIO ATTIVITA	COMUNE	LONGITUDINE	LATITUDINE	FlagBad
412	SITAV S.P.A.	20-OTT-2004	MILANO	9.180	45.467	NaN
450	ATENIX ELECTRONIC ENGINEERING S.R.L.	02-GEN-1998	MASON VICENTINO	11.613	45.723	NaN
460	CENTRO ZANZARIERE DI BARRETTA MAURO	05-MAR-2001	GAGGIANO	9.031	45.406	NaN
527	CONFEZIONE LA FORTUNA DI QIU ZUJIN	01-NOV-2011	FUCECCHIO	10.812	43.718	NaN
531	MOTOSTORES DI INNOCENTI ANDREA	07-APR-2008	TALAMONA	9.613	46.142	NaN
547	GI.NOVA. SUD S.R.L.	13-GEN-1987	PALERMO	13.347	38.099	Y
551	MUNGO LEONARDO	01-APR-2002	CROTONE	17.121	39.077	NaN
601	ICOPOWER S.R.L.	14-GEN-201	MILANO	9.149	45.472	NaN
666	BISSOLO CASA S.R.L.	15-LUG-1998	GAMBELLARA	11.351	45.432	NaN
795	ICE NICE S.R.L	03-NOV-201	MONTEFIORE CONCA	12.645	43.903	Y

Table 3.1: Master file head for a subset of features. It contains general informations of a company like name, address, starting activity date, geographical information and the $FlagBad$, the target variable. Null Value of $FlagBad$ means that the company is good, while if it is Y the firm is bad. RK is the index wherewith a company is identified across the files. Here it is show only a subset of features, the dimension of the file is: 135395 rows and 34 columns.

Figure 3.1: Scatter plot Master file, for numerical features and it shows that there are no clear trends in the data.

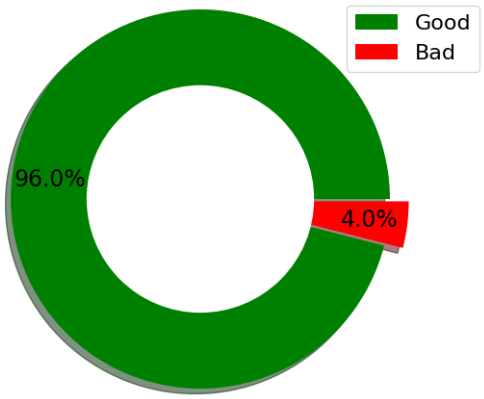


Figure 3.2: Bad and good companies pie chart. The dataset is unbalanced.

The data contained in the dataset, before being used in the various classification models, was subject to preprocessing. In this phase we focused a lot on feature extraction, and features construction, through the combination of variables. Now we turn our attention on the description of the dataset and at the same time we give the basic idea of how new features were built in the preprocessing step.

The Master file contains general information of companies, like name, address, date of the beginning and eventually ending of the activity. In Table 3.1 we can see a subset of features contained in the file, while in Figure 3.1 we can see a scatter plot of the numerical features. It contains also latitude longitude and the target feature $FlagBad$. We have here 135395 observations of companies and 34 features. The full list of variables are in Appendix A.1.

The balance sheet file contains the business balance of companies. We have 440847 observations in the file, and 36 features: the financial data. Only 62918, 48% of total companies, have available the balance sheet, while for the other 71699 we do not have these features. This will therefore involve the construction of different models for companies, depending on the presence or absence of the balance sheet. In Table 3.2 a subset of features contained in the balance sheet file is shown, while the whole variables are described in Appendix A.2.

The main part of the work of feature engineering is done within the business balance data, which were provided from the beginning of 2008 to the end of 2017; the features that are created here can be grouped in the following macro areas:

1. **Hist features**. It is the value of a feature in the last period.
   E.g. $ROE||prev\_n\_years$ is the value of ROE in the n-th year before the last financial statements available in the balance sheet.

2. **Diff features**. Each information in the balance sheet is compared with the same in the previous period, which could be month, year or quarter, both as an absolute difference and as a percentage. E.g. $ROE||abs\_diff||periods\_n$ means the difference with sign, of ROE between the last balance sheet and

RK	DATA CHIUSURA BILANCIO	ROS	ROI	LIQUIDITA	TEMPI MEDI PAGAMENTO	DEBITI	TOTALE ATTIVO
1000026	31/12/2008	0.94	0.9	0.81	112.01	1682816	3417700
1000026	31/12/2009	1.04	0.89	0.63	137.15	2519908	4246962
1000026	31/12/2010	3.77	3.88	0.65	109.15	3296008	5199433
1000026	31/12/2011	5.64	5.85	0.68	70.6	3262116	5287889
1000026	31/12/2012	7.44	8.07	0.77	115.61	3630028	5905571
1000026	31/12/2013	8.01	9.43	0.82	90.61	3017105	5433126
1000026	31/12/2014	5.24	6.51	0.74	92.84	3282373	5796097
1000026	31/12/2015	3.73	4.39	0.76	114.54	4139678	6673411
1000026	31/12/2016	1.96	2.14	0.72	NaN	5426132	8043461
10000533	31/12/2013	10.91	10.8	1.19	126.53	1067756	1383961

Table 3.2: Balance sheet file head for a subset of features. This file is available only for almost 50% of companies in the dataset. It contains economic information of companies like ROS, ROI, net assets and debts. It is composed by 440847 rows and 36 features, in this table we see only a subset of them.

the previous $n-th$ one, while $ROE||pct\_change||periods\_n$ measure how much it changes in percentage.

3. **Is-null features**. Some variables have some field that is null: we do not know if this missing value is because some errors in the transcription of the datum or the firm voluntarily do not provide it: these variables have the suffix is_null.

Furthermore, the variable indicating the delay between the balance sheet closing date and the day of its publication in the Chamber of Commerce has been created, with the suffix $lag\_deposito$; and that one which indicates how many balance sheets are published in each year, $num\_bilanci$: there are firms that publish them even quarterly. These variables are also combined with each other: e.g. $ROE||abs\_diff||periods\_4||prev\_02\_years$ is the absolute difference of values of ROE between the balance sheet of two years before the last one, the balance sheet published in 2017, and four previous periods; in other words it is the absolute difference of the ROE of 2015, 2 years before 2017, compared to 2011, 6 years before 2017.

Employee file contains information about the number of workers in the firm, from the beginning of the activity, updated every trimester from 2015 to 2017. It contains 931466 rows and 6 columns. The first 10 rows of the file are shown in Table 3.3, and in Figure 3.3 a scatter plot of the file is shown. Here the absolute difference and the percentage change are also used for the features building of the Employee variables.

In Ten-member file there is the list of the first ten partners who own the highest share, with also general information like name, address and VAT number. This file contains 127344 observations and 8 features gathered from 2015 to 2017, and the first 10 rows are shown in Table 3.4 for a subset of columns. Lots of companies have a single partner who controls a firm or two that have 50% of shares, as we see in Figure 3.4. The full features contained in Ten-member file are described in Appendix A.4. Starting from it, we create variables indicating if a partner who has a share grater than $x\%$ exists where $x \in [0, 100]$, how

RK	ANNO	TRIMESTRE	DIPENDENTI	INDIPENDENTI	TOTALE
13	2016	04	4	0	4
13	2016	04	4	0	4
13	2016	03	3	0	3
13	2016	02	4	0	4
13	2016	01	5	0	5
13	2015	04	6	0	6
13	2015	03	7	0	7
13	2015	02	7	0	7
71	2017	01	24	0	24
71	2016	04	23	0	23

Table 3.3: Employee file head. In this file there are information about the workers of firms, reported in each quarter from 2008 to 2017. There are the number of employees, workers, apprentices, employees, managers, and the independent employees, members, directors.

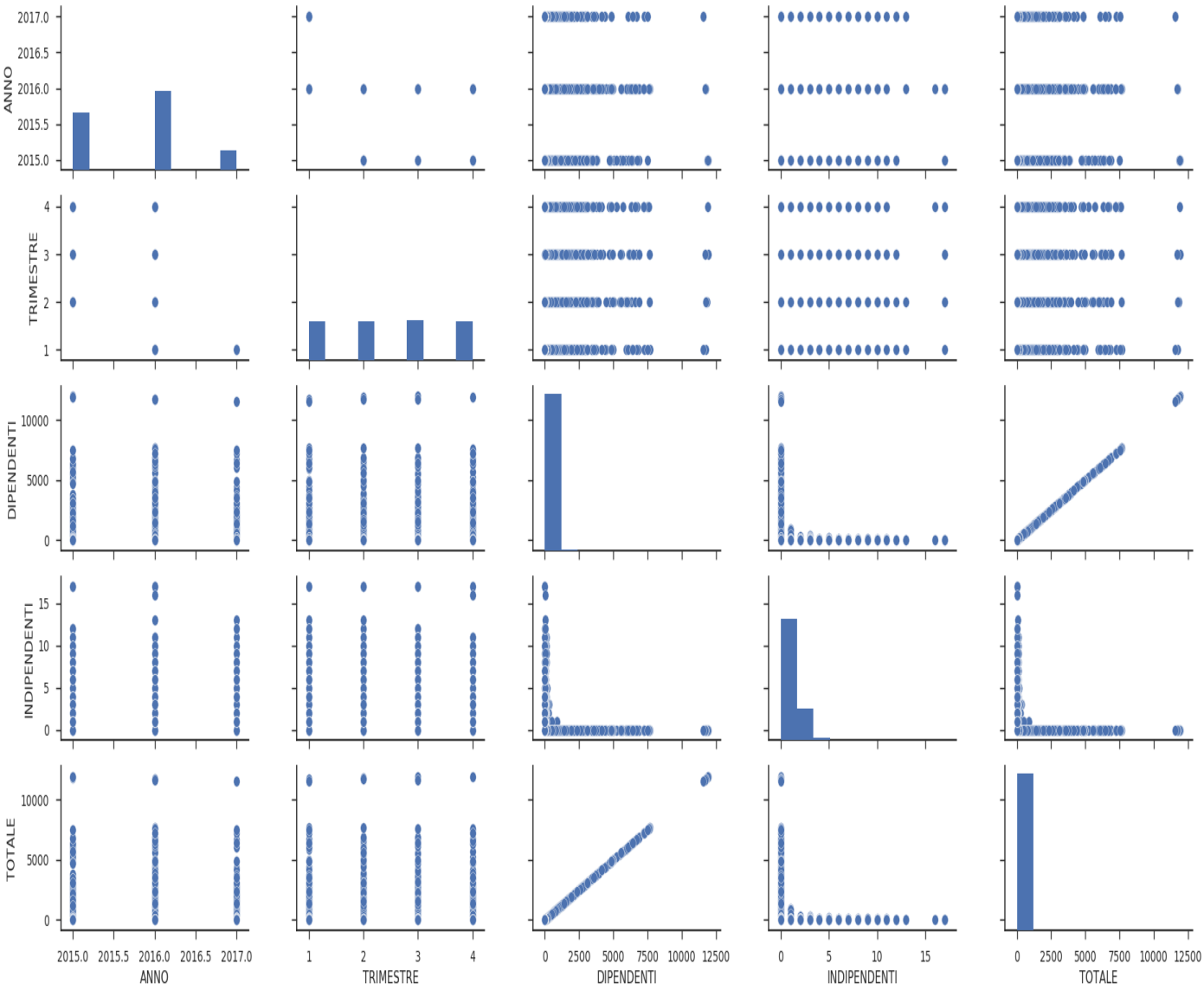


Figure 3.3: Scatter plot Employee file. We can see that this file reports all the dipendets every quarter from 2015 to 2017. Most companies havea low number of employees, while the number of independet employees are less than 5 in most cases.

many of them exceed that quota and the ratio of them: so for example we have *num_soci_dieci_quota_ge_30_percent* counts how many members have a quote of shares grater than 30%; or the ratio of members that have a share grater than 30% compared to the number of members, *ratio_soci_dieci_quota_ge_30_percent*.

Ten-exponent contains information about the management of firms, like name and role, as we can see in Table 3.5. The file is composed by 124717 observations and 8 features. We computed the mean and the variance of the age of the mangers, how many positions are held by different people, and if they live in the same city.

In Local units file there are information of the address of the various units of a firm. In Table 3.6 we can see a subset of the whole features in it. The file is composed by 1253379 rows and 17 columns, and they are described in Appendix A.6. From these variables the number of local units is computed, and if they are located in the same province. The feature $TIPOLOCALIZZAZIONE$ in this dataset is always equal to UL so in will not be take into consideration in the future.

After the preprocessing, the total features available are 1782 for companies in which we have a balance sheet file and 85 for those that do not have it. The dataset for the construction and validation of the model, is immediately split into two parts: the training set, including 80% of the observations, and the remaining part, the test set, for the final evaluation of the constructed model. Therefore the training set consists of 50333 of the companies where the balance sheet file is available, and 57359 of those with the absence of it, while the test set is composed by 12585 companies with a balance sheet and 14340 for the other group

In addition to those features we are going to add one more: the digital score which is a compact variable that summarize the firm presence on the web, as we are going to discuss in Chapter 4. The list of the features are described in Appendix A.

RK	CODICE FISCALE	DENOMINAZIONE	COMUNE	PROVINCIA	PERCENTUALE QUOTA
2760083	LXRCLT76D49G224H	LUXARDO CARLOTTA	NaN	NaN	11.11
1375035	MGNGTN67L24I459B	MIGIANI AGOSTINO	SASSOCORVARO	PS	25.00
1272882	DSNSCR69D10H501I	DI SANTO OSCAR	ROMA	RM	25.33
1627986	MBRVNT78H60H501A	IMBROGLINI VALENTINA	ROMA	RM	11.99
183300	CHNNRC59M14B157Y	CHINI ENRICO	NaN	NaN	17.39
350143	MSCSLV66C67I690K	MASCESE SILVIA	SESTO SAN GIOVANNI	23.75	
722270	03328990969	NESPOLI GROUP S.P.A.		100	
689653	01966260356	PREDIERI GROUP S.R.L.		46.11	
8136818	MRSVTR49M18I373C	MARSON VITTORIO	SAN STINO DI LIVENZA	VE	16.00
4108365	SGNTMS79D18B157N	SIGNAROLI THOMAS	MONTIRONE	BS	12.82 1

Table 3.4: Ten-member file head for a subset of features. Here we have a list of the first ten partners, its general information like name, city in which they live and the percentage of shares they hold. In this table there are listed only a subset of the whole features of the file, which are 8, while the number of information are 127344.

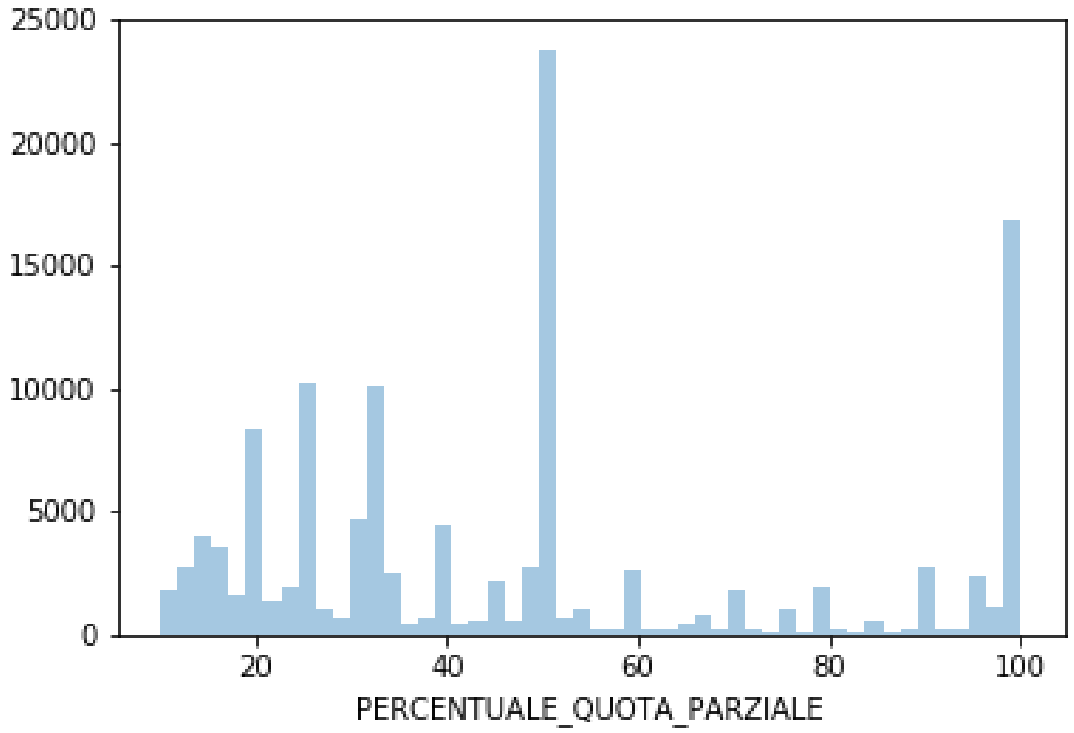


Figure 3.4: Histogram partners share percentage. Many companies have a single member or two, as the bar is hight in 50 and 100.

RK	CODICEFISCALE	NAME	CODICE	CARICA	DATA	LUOGO	SESSO
13	LPRPQL61T23G190K	ALIPERTA PASQUALE	DT	DIRETTORE TECNICO	1961-12-23	G190	M
13	VTLFMN69D68I438I	VITOLO FILOMENA	AUN	AMMINISTRATORE UNICO	1969-04-28	I438	F
71	PRDMTT68H15F205T	PARODI MATTEO	AUN	AMMINISTRATORE UNICO	1968-06-15	F205	M
105	CHRMRS62M69G337X	CHIERICI MARISA	SOA	SOCIO AMMINISTRATORE	1962-08-29	G337	F
105	MNRMRA63H08G337Q	MANARA MAURO	SOA	SOCIO AMMINISTRATORE	1963-08-06	G337	M
110	FNTCHR92A71G843N	FONTANELLI CHIARA	TIT	TITOLARE FIRMATARIO	31/01/1992	G843	F
127	FLPFNC77C27I726O	FILIPPINI FRANCESCO	VPA	VICE PRESIDENTE DEL CONSIGLIO D'AMMINISTRAZIONE	27/03/1977	I726	M
127	FLPGCM73L30I726Q	FILIPPINI GIACOMO	CON	CONSIGLIERE	30/07/1973	I726	M
127	FNTGPR62L12F605W	FANETTI GIAMPIERO	PCA	PRESIDENTE CONSIGLIO AMMINISTRAZIONE	12/07/1962	F605	M
127	FNTRRT70P28F605C	FANETTI ROBERTO	CON	CONSIGLIERE	28/09/1970	F605	M

Table 3.5: Ten-exponent file head for a subset of features. It describes the management of firms, their role, the sex and the date when they start to work there. The file is composed by 124717 rows and 8 columns.

RK	CODICE FISCALE	TIPO LOCALIZZAZIONE	TIPO UNITA LOCALE	DATA ISCRIZIONE	COMUNE	CAP	PROVINCIA
1000026	00600750376	UL	DEP	13-GIU-2012	RECANATI	62019	MC
1000026	00600750376	UL	U	06-MAG-1972	BOLOGNA	40138	BO
1000171	03249830617	UL	SO	16-MAG-2006	DRAGONI	81010	CE
1000195	01884650746	UL	UA	12-MAG-2000	BRINDISI	72100	BR
1000198	02530220967	UL	CAP	19-GIU-2013	VILLASANTA	20852	MB
1000198	02530220967	UL	ULO	21-NOV-1995	SESTO SAN GIOVANNI	20099	MI
1000207	07883080637	UL	DEP	21-MAR-2002	MELITO DI NAPOLI	80017	NaN
1000207	07883080637	UL	DEP	21-MAR-2002	MELITO DI NAPOLI	80017	NaN
1000261	03306650965	UL	LB	15-NOV-2001	PADERNO DUGNANO	20037	MI
1000271	02243710841	UL	MA	26-GIU-2014	SAVIANO	80039	NaN

Table 3.6: Local units file head for a subset of features. It contains information about the units which is composed a firm, their city, their sector, bank, assurance or industrial. It is composed by 1253379 rows and 17 columns.

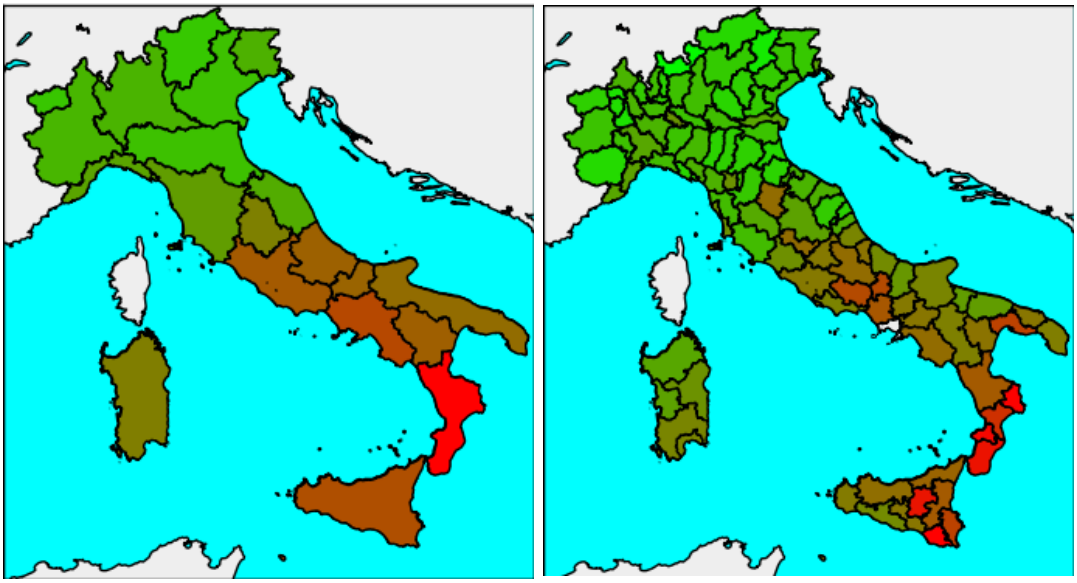


Figure 3.5: Distribution of Companies in Italy. The color indicate the average of bads in the country. Red color means higher percentage of companies in default.

3.1.1 Geographical Distribution

The financial database contains a subset of Italian companies and it preserve their real distribution in Italy. As per current socio-economic Italian situation, the dataset, therefore, reflects the fact that more companies are localised in the north and less in the centre and even less in the south. The percentage of bad companies is higher in the centre and south as per Figure 3.5.

In more details we can say that many of the companies are distributed in northern Italy, in particular in the regions of Lombardy, Veneto and Emila-Romagna: in fact these areas are considered the engine of Italy.

The regions of southern Italy in which there are more good companies are Puglia, Sicily and Campania, which are also those with the highest population density in that area. Naples is the only province where we have no observations.

Figure 3.5 clearly shows Italy is split in two because of the strong gap between the south and the center north. In support of this, all the northern regions have a low percentage of defaulted companies, and in particular Trentino is the best region with a rate of only 1.9%. The South instead, in addition to having few companies compared to the north, has a high concentration of bad companies, and among all Calabria stands out with a rate of 8% of them. This situation is also found by analysing the various provinces of Italy. All those in the north are good, with the only exception of Arezzo which is the worst in the area; while the bad ones are mainly concentrated in southern Italy with some exceptions, especially

in Puglia and Sardinia. The worst provinces are the Calabrian provinces with Reggio Calabria, Vibo Valencia and Crotone, and the Sicilian ones with Enna and Ragusa.

In the dataset are present also foreigns firms, but they are negligible with the presence of only 44. In Figure 3.6 and Table 3.8 the number of good and bad companies for each region is reported.

Now, let us go into more detail and consider the situation of a single region, for example we can concentrate and analyse the actual conditions of Sicily. As discussed before, we can say that Enna and Ragusa are the worst provinces where there is a greater concentration of failed companies, clearly shown by the Figure 3.7. Instead it is clear that in the western part of Sicily, like Palermo and Trapani, there is a significant number of good and productive companies, with a low ratio between goods and bads compared to the other areas of the island.

Therefore the western zone has more resources unlike the eastern part, with the exception of Messina. Particular is the case of Catania, which being a prestigious city for the Sicilian economy and culture, the sample in reference shows a low percentage of observations related to the effective economic value of the province.

Region	Bad	Good	Region	Bad	Good
Abruzzo	169	2926	Piemonte	195	7443
Basilicata	68	1145	Puglia	525	9729
Calabria	442	4601	Sardegna	126	2656
Campania	362	5348	Sicilia	404	6204
Emilia-Romagna	295	12755	Toscana	331	9212
Friuli-Venezia Giulia	67	2354	Trentino-Alto Adige	56	2840
Lazio	757	12500	Umbria	81	1755
Liguria	106	3250	Valle d'Aosta	6	285
Lombardia	622	22506	Veneto	275	12255
Marche	111	3656	ESTERO	1	43
Molise	34	597			

Table 3.8: Number of good and bad companies in the Italian regions.

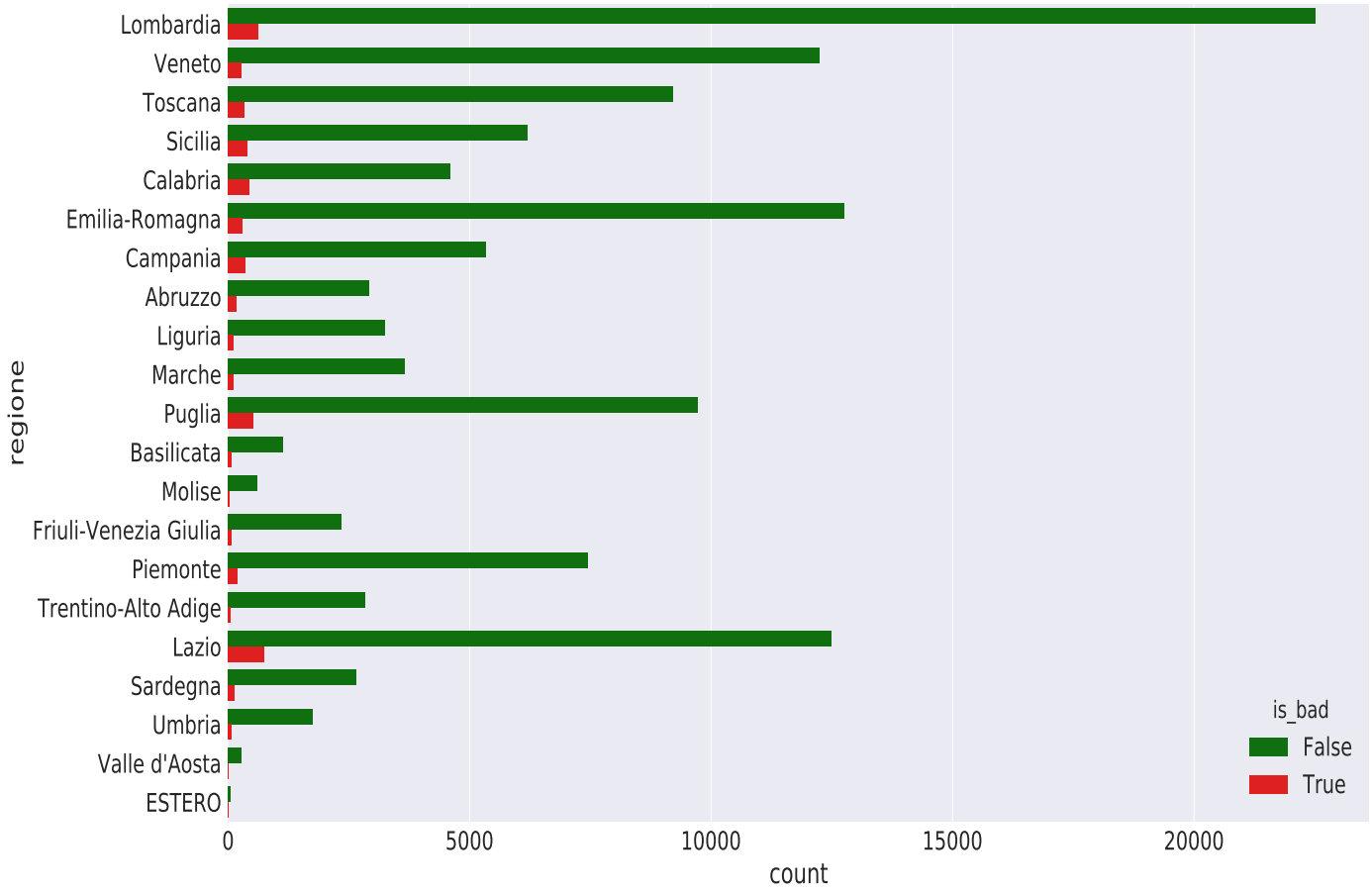
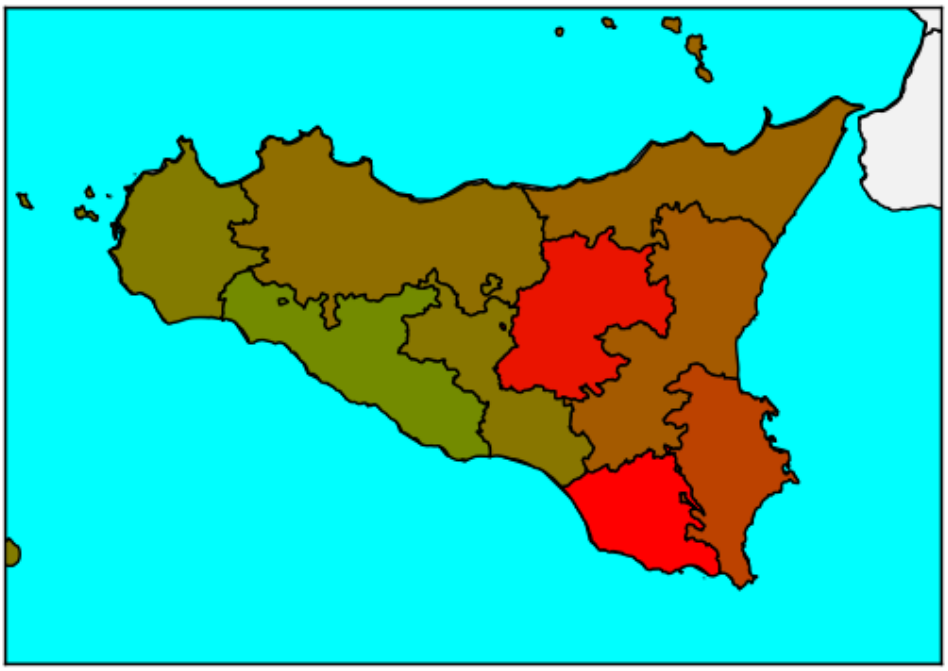
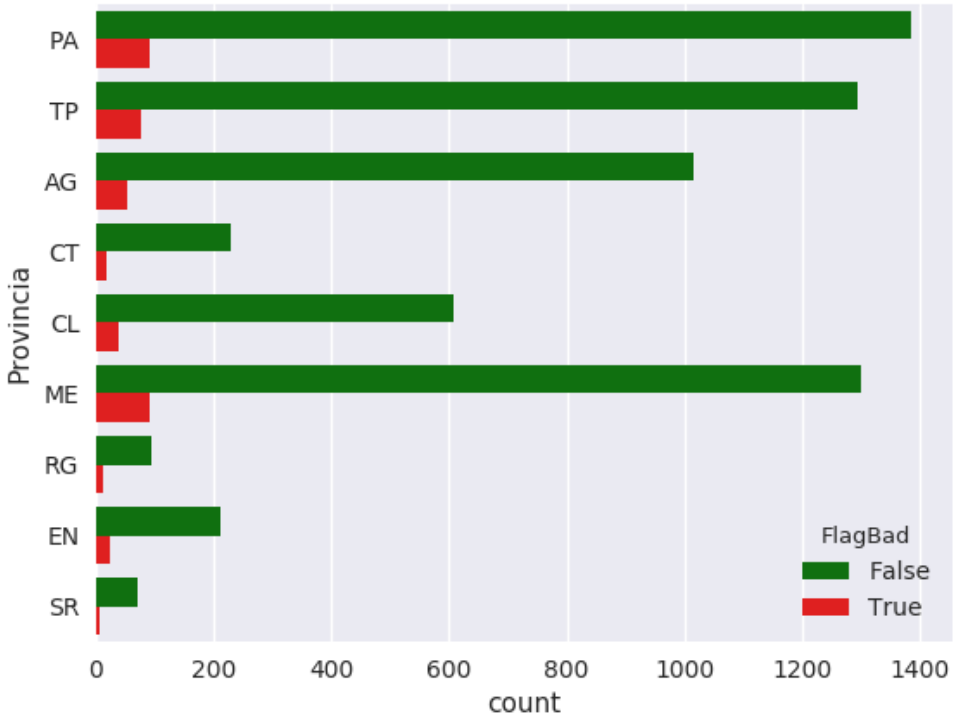


Figure 3.6: Companies countplot for each region, grouped by FlagBad: the green colour represent good companies, while the red is for the bad ones. Lombardia is the most represented region, and in general the companies are distributed mainly in the northern, including Veneto, Emilia-Romagna and Toscana. Outside this area Lazio has a high number of firms, both good and bad. South Italy, as well known historically, is not an attractive area for business.



(a) Companies distribution map in Sicily



(b) Count-plot for the Sicilian provinces

Figure 3.7: Companies distribution in Sicily. The colour indicate the average of bads in the country. Red means higher percentage of companies in default: Enna and Ragusa are the worst province in Sicily. The firms are concentrated more in the west cost than the eastern part, with the exception of the province of Messina.

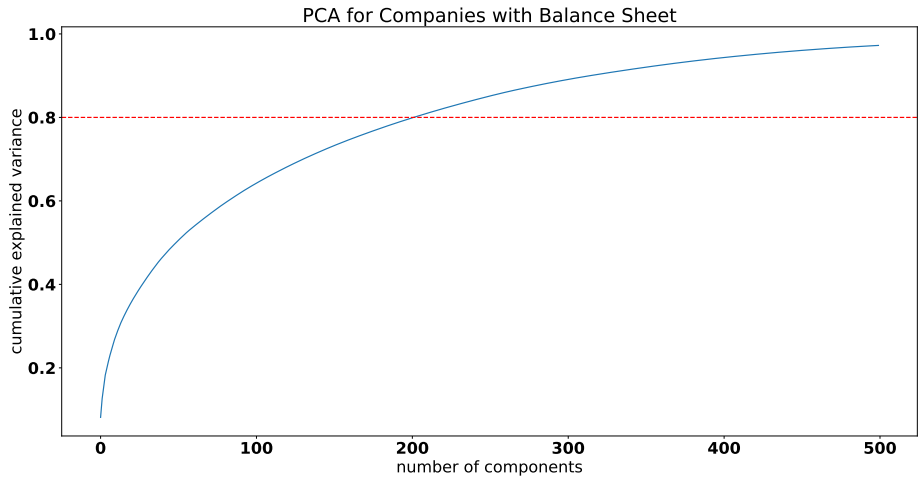


Figure 3.8: Explained variance of the first 500 principal component for companies that have a balance sheet.

3.2 Principal Component Analysis

In this section linear data reduction is performed for the purpose of data visualisation, using the technique of principal component analysis (PCA). PCA is effected by scale so we need to standardize the data before applying it: one approach, which is followed here, is to normalize each feature to have zero mean and unit variance.

Our dataset contains lots of missing values, so in order to apply this technique, they are imputed to the median for the numerical features and to the mode for the discrete ones. Eliminating the observations which contains missing data is not a good a idea because more than 90% of the total observations contains null values.

3.2.1 Companies with a balance sheet

First, the principal component analysis is applied to the companies that have a balance sheet: after the feature engineering, the features we have for them are 1781. As we can see in Figure 3.8, to be able to explain 80% of the variance of the dataset we need 200 principal components. Moreover, the curve is smooth, so there is no clear evidence on how many principal components are to be used.

Looking in more detail the first three principal components, which alone explain only 15% of the total variance in the dataset, we can not give them a precise interpretation. The first principal component (PC) is represented by a linear combination between the total assets and liabilities, the fixed and the net assets of the previous two and three years: they are the main features that make up the PC.

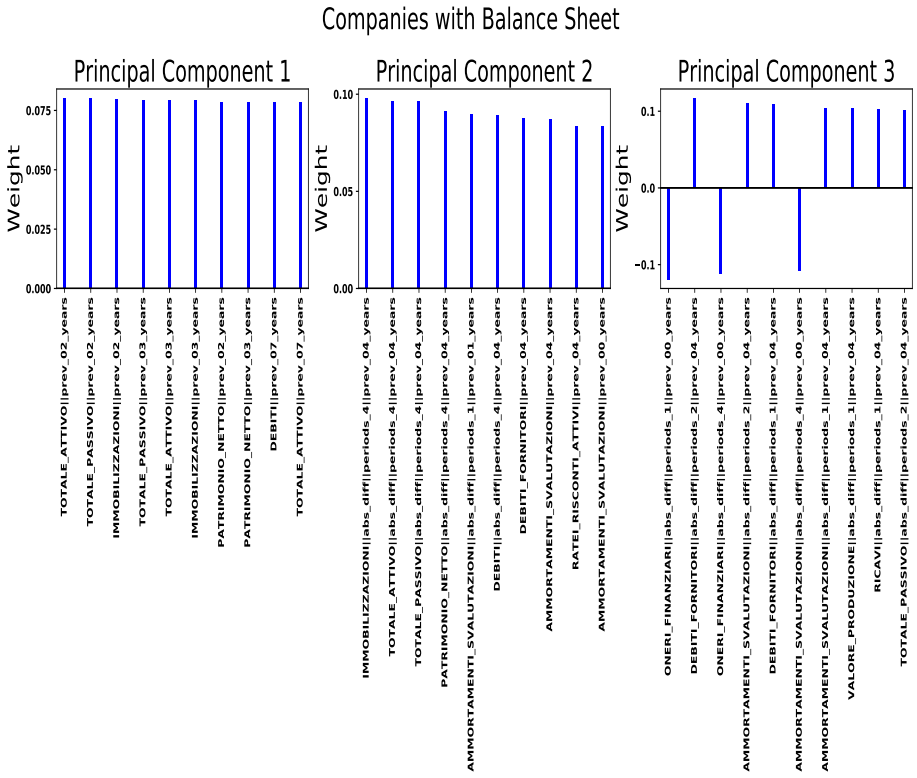


Figure 3.9: Loadings for the first tree principal components. In the plot it is showed the top ten features that contribute more for composing the PC.

The main variables that form the second principal component are the difference in absolute value of the fixed, total and net assets and liabilities in the previous four years. Finally, the third component consists of the combination, with alternative weights positive and negative, of the charges, debts and amortization.

If we plot the data in two dimensions, using the first two principal components, there is no clear distinctions between the good and bad companies, as we can see in Fig. 3.10. In order to distinguish them better, we need other principal components, or other techniques.

3.2.2 Companies without a balance sheet

The same techniques have been used for companies without a balance sheet. In this case, only 85 features are available for the analysis, and as seen in Figure 3.11, using the first 20 components allows explaining more than 80% of the variance.

The first principal component consists in the combination of ratio quotes and numbers of ten members, which shares are grater than 60%. The second one is composed by the ratio of ten members less than 50% , while the third component

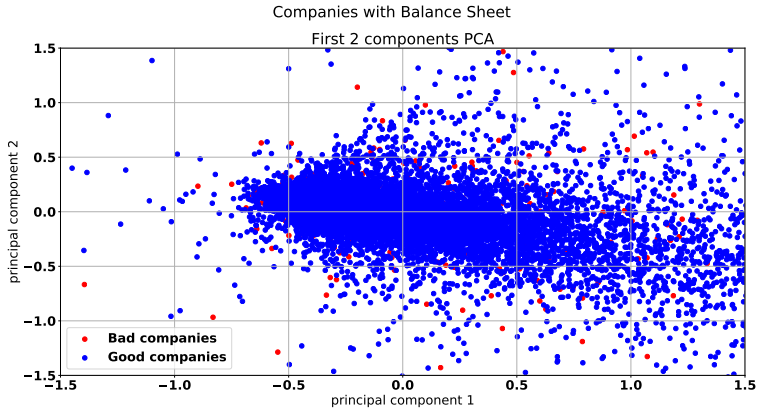


Figure 3.10: 2D-visualization for companies that have a balance sheet. The blue dots represent the good companies, while the red are the bads.

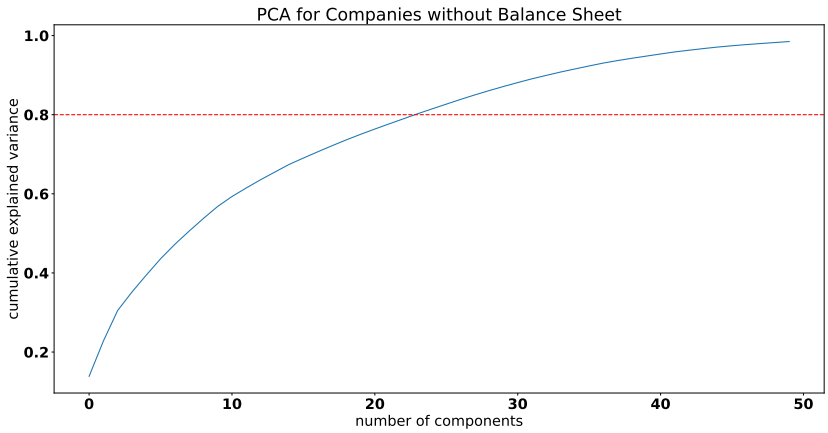


Figure 3.11: Explained variance for the principal component for companies that do not have a balance sheet.

is a combination of activity start date and registration date (Figure 3.12). If we pick up the first 3 we can explain 30% of variance of the dataset. Furthermore, a scatter plot of the first two components is shown in Figure 3.13. Even in this case there is no clear distinction between the two classes of companies. The data are sparse and they are not concentrated around the mean, that in this case is zero because they were standardized.

At the end, we can say that principal component analysis is not very useful for the feature selection, and it will not be used in the machine learning models. Indeed, the explained variance plots are smooth and there is no clear elbow that indicates how many components we have to select. Moreover, selecting the first k components that explain more than 80% of variance is not a good idea, because we need to consider too features.

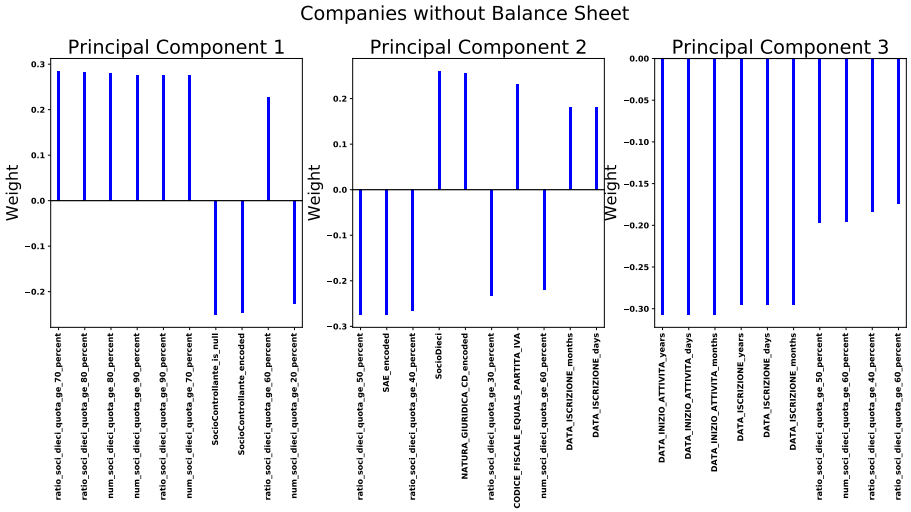


Figure 3.12: Loadings for the first 3 principal components of the companies without a balance sheet. In the plot it is showed the top ten features that contribute more for composing the PC.

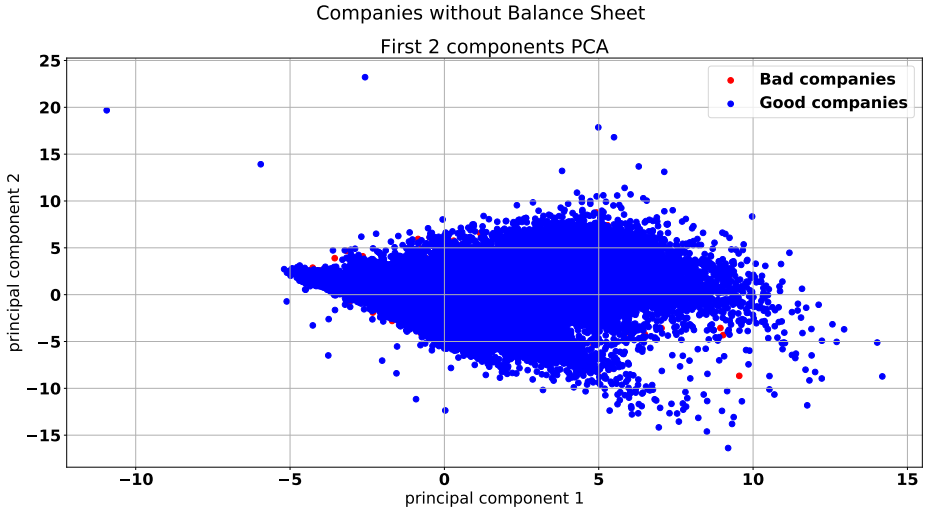


Figure 3.13: 2D-visualization for companies that do not have a balance sheet. The blue dots represent the good companies, while the red are the bad.

Chapter 4

Digital Score

This chapter describes how our novel alternative dataset based on digital sources is constructed and how we extract a digital score to measure risk. The structure of this Chapter is the following: Section 4.1 describes the web crawler employed in the digital data collection. Section 4.2 discusses the distribution and the strengths of the digital score.

4.0.1 Introduction

The digital score is a single numeric value that measures the presence and the performance of a company on internet. It analyses the presence of the company on the web and whether the presence is solid i.e. contains information about that company, if visible on search engines, and the sentiment of the user's reviews. Effectively, a high value of the score means a good presence and visibility on the web.

The main intuition behind digital score is the fact that the solidity of a company could be monitored by regularly observing its digital profile. In this thesis digital profile is defined as a combination of web driven features such as: availability of a web page, presence on Linkedin , presence on Facebook, and a combination of presence and reviews on Google My Businesses. The digital score was obtained by scraping Google and firms websites, through the construction of a web crawler, then, the various information obtained were combined and summarized in a single value through some selected weights. In the digital score we are considering the following information: Google ranking algorithm, for the visibility of a firm, corporate information, measuring if a firm site is in accordance to Italian Civil code, social media and reviews by users. Finally, the score has a range that goes from 0 to 100: low values mean that the digital profile is not good, while high values

are for companies that have a good profile on the web.

The work starts searching the name of the firms on a search engine; in the present work Google was used. We got the urls of the possible sites and other information that can be extrapolated from the search. For each url, finally, we study its html page in order to associate it with the company and to obtain information from it.

The data on websites are generally not available to download easily and can only be accessed by using a web browser. Most of the sites do not provide the functionality to save their data, some of them give APIs to users but often there are restrictions on them and are not reliable enough. The information could be extracted using a web crawler or spider. It is a software that connects on the Internet, fetch the web page through its url and download its content and data. These techniques are also known as web scraping, web harvesting or web data extraction.

Finally, we must say that Google and other sites, do not allow us to extrapolate large amounts of data and they stop the crawler from collecting data as soon as they identify it. In general, a web crawler can be identified by:

1. **Unusual traffic/high download rate**, that is, it performs many requests from a single client or IP address in a very short time;

2. **Repetitive tasks performed on the website**, this is based on the assumption that a human user will not perform the same repetitive tasks all the time.

3. **Detection through honeypots** which are usually links not visible to a human user but only to a spider. When a scraper/spider tries to access the link, then the site stops it.

The User Agent rotation is the best solution for avoiding the block of the crawler, as every request made from a web browser contains a user-agent header and using the same consistently leads to the detection of a bot[1].

4.1 Dataset Creation

This Section focuses on the description of the web crawler and on the building of the digital score from the features gathered with the software. More in detail, Subsection 4.1.1 summarizes the problems that have arisen when we have built the crawler and, then, describes the sites we have considered and the information we have downloaded from there. Subsection 4.1.2 describes the assumption that we have taken for the formula of the digital score.

[1]Rotating the IP or making the crawler slower are other widely used methods that prevent the detection of a bot. **ScrapeHero** indicates the main strategies on how to avoid being stuck during scraping

4.1.1 Web Crawler

The aim of the web crawler is to obtain new features and data from internet and that will be integrated into the dataset the client company provided to us. These new data we will use, together with the financial data, when we build the models in Chapter 5 and hence they take part for the default classification problem.

Given a company of the financial dataset we want to compute its digital score. The dataset, the client company provided to us, contains name, address, city and VAT number of firms; from those we do a search on Google starting from a simple query, name and Vat number, and then going to strengthen it, adding further information of that company, like the city and the address if no url can be associate to the firm. The output of the crawler is the list of the first ten urls found for each company and the html page of the search. In Figure 4.1 we can see the output of Google search for 3rdPLACE and the main information that we can obtain from it.

There were many problems that have arisen in the building of the web crawler. First, our dataset, composed by real-world data, contains lots of missing data and imperfections even in the name of the firm. In particular many companies have as their name also some comments, acronyms, name of all members, etc. This leads to a deterioration of the performance of the crawler, that in many cases the search has very few results and in the worst cases it has none. To overcame these difficulties it is important to do some cleaning work on the name of firms and to perform the search several times, improving significantly its performance and have at the end better results. Another problem that we have faced was that, as mentioned before, Google and also some websites block the crawler in downloading data. This was overcome by rotating the user agent of the crawler.

Now, we discuss more in details what features we got from the crawler. First, from Google search we obtained the following new features:

- **row count:** how many times an URL appears in the same given search of a company;
- **total count:** how many times the URL appears in all the searches performed in the whole dataset;
- **position:** first occurrence position of the URL when searching that company.

Gathering these information, we are implicitly including Google ranking algorithm, PageRank, which measure the importance of website pages. It works by counting the number and quality of links to a page to determine a rough estimate[2] of how important the website is.

[2]The underlying assumption is that more important websites are likely to receive more links from other websites. For more details see `https://web.archive.org/web/20111104131332/https://www.google.com/competition/howgooglesearchworks.html`

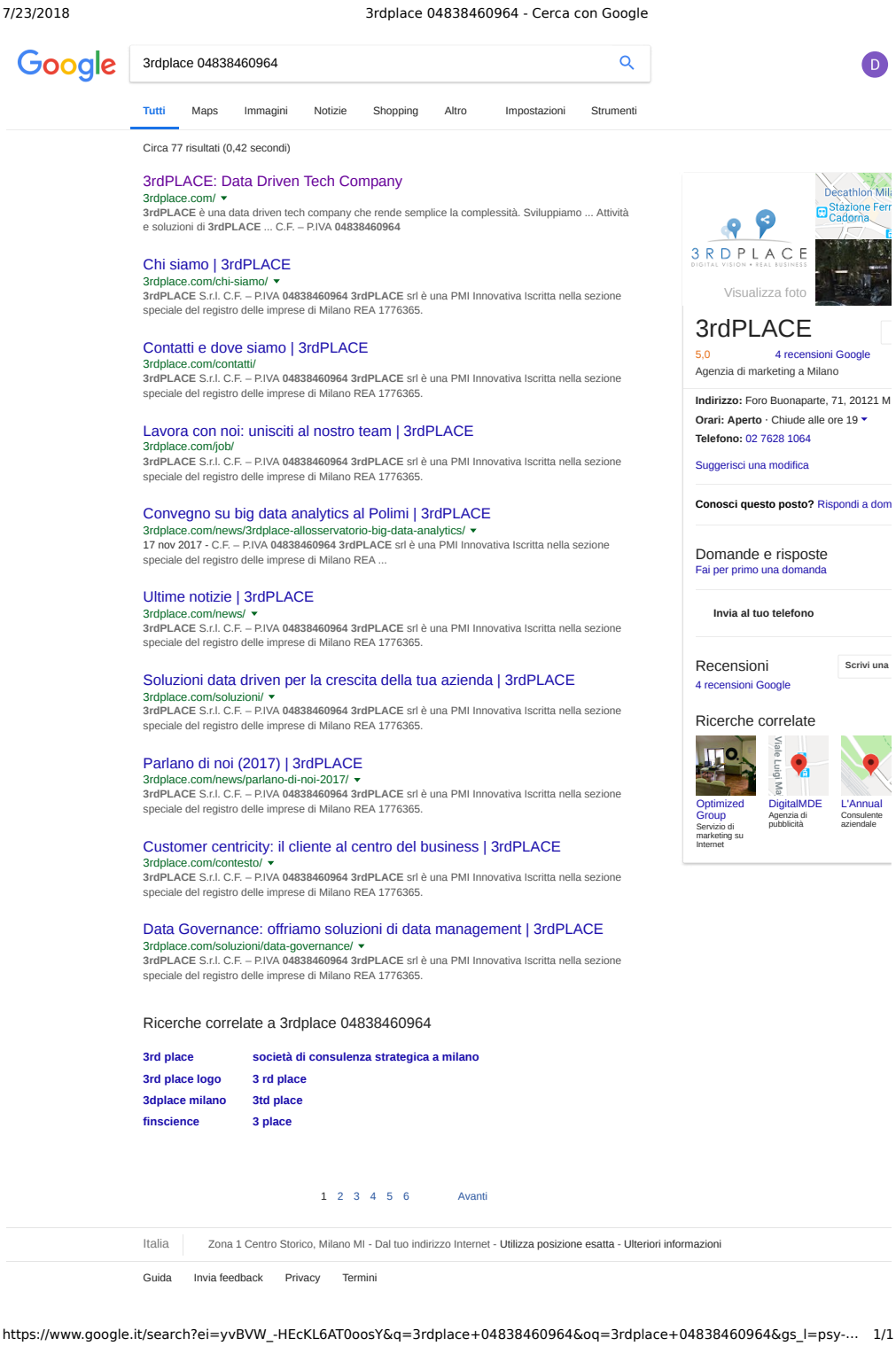


Figure 4.1: 3rdPLACE *Google result. In this specific case we see that all the url founded are from 3rdPLACE, hence we have* $row\_count = 10$, $position = 1$. *It is not possible to compute* $total\_count$ *because we need the search of all companies. In the Figure we can also see on the right the review values from the users, which is 5, and their number, 4.*

Feature	Description
has webpage	indicate whether the company has a site
row count	how many times the site appears in Google Search
total count	how many times the site appears in all the searches
position	first occurrence position of the site in Google Search
corporate informations	presence of name, address, VAT number, PEC, CAP, city.
presence site in box My business	boolean indicating whether the link in My Business to the site is present
social networks	link to main social networks if present in the homepage (Facebook, Instagram, Twitter, Google+, Pinterest, Youtube, Linkedin)
value reviews	quality of reviews in My Business
numbers reviews	numbers of reviews in My Business

Table 4.2: List of features gathered on the web. The first four are extracted when searching the company in the search engine. Corporate informations and social media come from the firm site and, finally, value and numbers of reviews are obtained from Google My Business.

Let us focusing on the website, the corporate data should be present in the homepage. The article 2250 of the Italian Civil code imposes to the companies to publish in their website legal information. Business name, Vat Number and fiscal code, PEC or certified mail and the address must be in their site. Ideally each company's site has these information, however it is estimated that two out of three of them do not report the contents required by Italian and European legislation concerning the publication of web content and electronic commerce. The presence of corporate informations means that the company takes care to keep its site in good condition, and could be an indicator of how good the business is. The data collected are:

- **corporate informations**: presence of name, address, VAT number, PEC, CAP, city;

- **social media**: link for Facebook, Twitter, Linkedin, Google+, Instagram, Youtube, Pinterest if they are present in the homepage.

Finally, we turn our attention on Google My Business. It integrates Google search with Google Maps and Google+, as a tool for search, selection, promotion, information and review of professional activities. Its presence means that the firm invests on online presence. The numbers and value of the reviews of users are the most important features here. In Table 4.2 there is a description of the informations gathered following this procedure which will be at the end combined and summarized in the digital score.

Feature	Weight	Feature	Weight
row count	1	review score	1.3
total count	0.5	presence site in my business	1.4
position	1	Facebook	0.8
numbers of information	1.3	Linkedin	0.8
has website	1.6	Instagram	0.7
Youtube	0.4	Twitter	0.5
Google+	0.3	Pinterest	0.3

Table 4.5: Weights used in the average mean of (4.2) *for computing the digital score. It naturally measures the presence on the web of the firms, hence if it has a website contributes more in the score. Then there are the number of the legal informations, the reviews of the users and if it has Google My Business. The less important is Google+ for its decline and a low user basin.*

4.1.2 Digital Score Formula

The digital score is a combination, through a normalized weighted average, of the features described in Table 4.2, an its range goes from 0 to 100. Before showing the actual formula we describe the assumptions that were assumed for the importance of the features and for the choice of their weights.

First, the position of the first occurrence of a site in search and the number of time it appears are more important than the number of time the same site appears in all the searches. There could be companies that have similar names and a company's site may be present in the list of urls of the other company, increasing the value of *total count*.

Then, there are different social media and some of them could be more important for business. Facebook, for its huge numbers of users, and Linkedin, for its importance in the professional network, are nowadays the social media more widely used and useful for business. Instagram is more known than Pinterest; Youtube may be an added value, while Google+ is now in decline and has a low user basin for companies.

Finally, the value and the numbers of reviews in My Business should be considered a whole, the *review score*. There may be company that has only one review with 5 value, and other with hundred of reviews but with very low value. Moreover for having the same scale it should be considered a logarithm scale for the numbers of reviews: its range is from 0 to thousands, while the values are from 0 to 5. The review score is computed as:

$$review\ score = (value - \overline{value}) \cdot \log(numbers + 1) \tag{4.1}$$

where $value$ is the review value of the company, $\overline{value}$ is the sample mean of the review value across the companies, and $numbers$ is the number of reviews the firms got. In equation (4.1) we add 1 to the numbers value in the logarithm function because if a company does not have review than its score is not $-\infty$.

Putting all together, the score of a company is computed as weighted sum of the digital information x_j

$$digital\ score = \sum_j \frac{w_j x_j}{\sum_k w_k} \tag{4.2}$$

where the weights w_j are chosen following the hypothesis developed before, and the actual value for them are shown in Table 4.5. If a firm is not present on the web and there are no digital information the digital score is set to 0.

We need to highlight that the weights used in this formula come from subjective and prior considerations. It is a first and naive attempt to create a digital score for the digital identity of the companies. In Chapter 5 all the models are built using both the digital score, as defined in equation (4.2), and with all the features got with the web crawler, without combining them. The comparison of the two results can help us to understand and, successively, tune better the weights of the digital score, in future developments.

4.2 Digital Score Distribution

We have succeeded in creating a digital identity at 30% of the companies contained in the sample considered: there are 42639 companies that have a digital score different from 0. In Figure 4.2 we can see the pie charts for the digital score. Of those that do not have a digital identity, many are small businesses that have neither Google MyBusiness nor a website, as they are not strictly useful for their business, such as bakeries, supermarkets, pharmacies and small shops. However, it must be said that we have not been able to correctly associate a website with some companies; one reasons could be that Google search did not produce any results, because their name contained comments or abbreviations. Nevertheless, the number of firm with a digital score can certainly be increased with the improvement of the procedure.

In this section we will study how the digital score is distributed in the dataset, tanking into consideration only the sub-sample without null value of this feature.

Looking at Figure 4.4, we can immediately see that, even if there is not a clear distinction between the two classes, the good and bad companies, there is a certain difference between them, in the mode, in the shape of the distribution: the latter group have a more right-focused distribution. The average for the good companies is 3 points higher than for the bads, thus indicating their best state of health on the web: the average of healthy companies is 29.1, while those in default is 26.5 (Figure 4.3). This correlation with the FlagBad will be useful, as we will see, also for the classification model. Moreover, the digital score range for companies in default is more limited and with values that go up to eighty.

It is also interesting to see how user reviews on MyBusiness change for good and bad companies. It is recalled that this score is computed as the product of

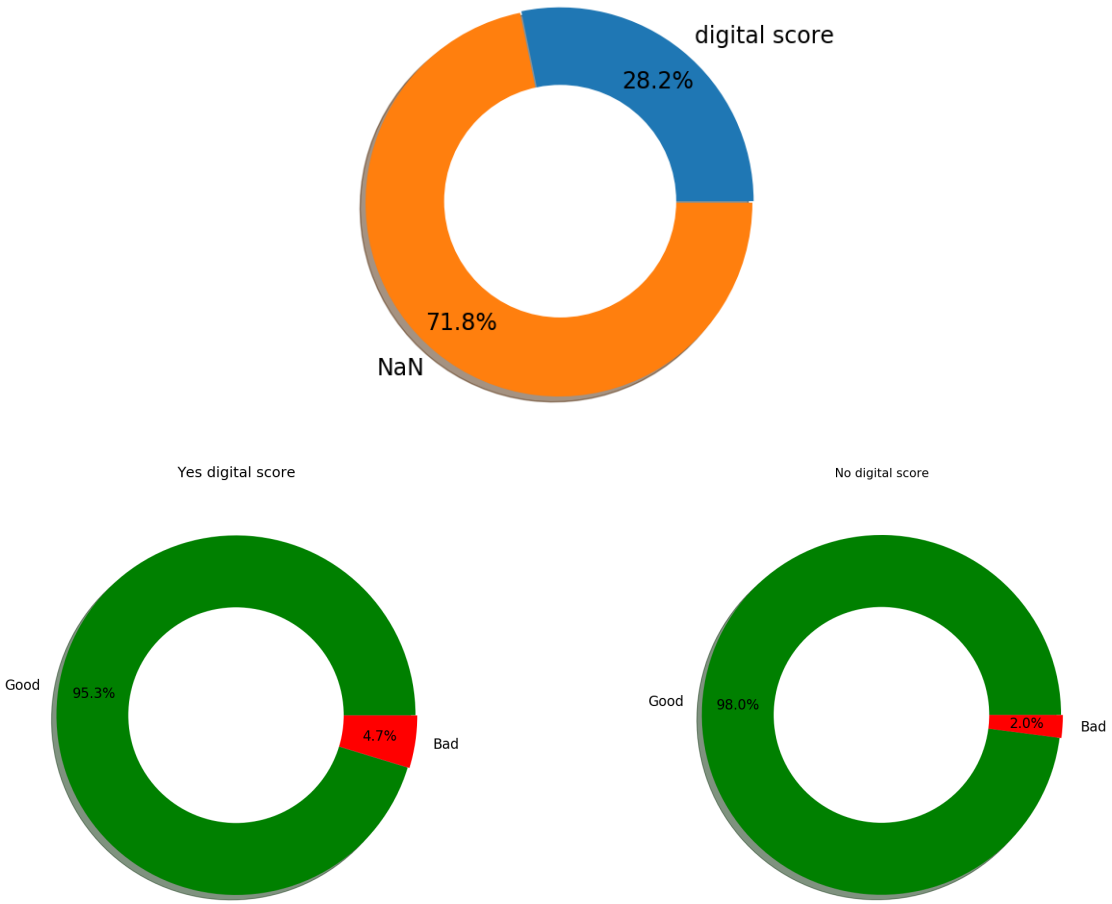


Figure 4.2: On top of the Figure is shown the digital identity distribution. 42639 companies have a digital identity ($digital\_score \neq 0$), and between them (plot on bottom left) 4.7% are bads, while the others are good. On bottom right we can see the distribution of good and bad for companies that do not have a digital identity ($digital\_score = NaN$. If a companies does not have a digital score, then is set to 0.

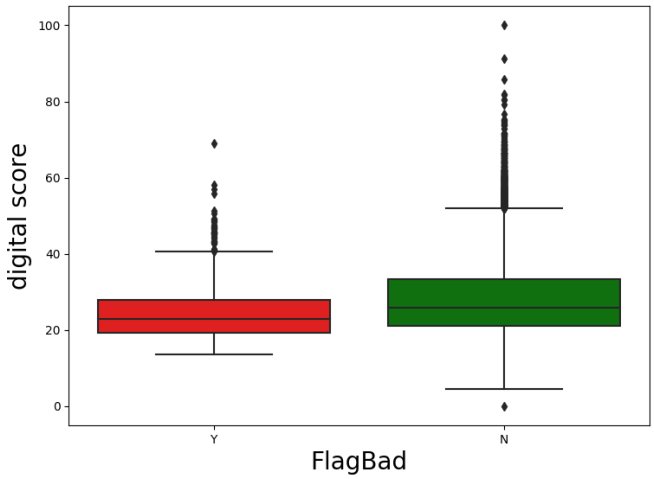


Figure 4.3: Boxplot digital score. The good companies, in green, have in average a higher digital score than the bad companies, in red

the value times their number as described in equation (4.1). The review score takes value from -7.5 to 15 for the good companies, while for the bad ones, its range goes from -3.2 to 13.2. Therefore, the good companies achieve grater values in module of review score. Even in this case, although there is no clear distinction between the two classes, one can see how there is a certain difference between them, as we can see in Figure 4.5. Again the shapes are quite similar, but the red distribution, which represent the reviews for the companies in default, has a mean and a range lower than the distribution of companies healthy.

Let's look in more detail at how the digital score is distributed compared to the regions of Italy, in Figure 4.6 and Figure 4.7. In general, the digital score has a similar average in all regions, which oscillates between 27 and 30. The foreign companies have a digital score whose mean is 24, however there are very few data to compare it with the other ones. The region that has more firms with an online presence is, as expected, Lombardy, confirming the fact that the most innovative and most digitized companies in the country are concentrated there. The high number is also a consequence of the high concentration of companies. Nevertheless the average value of the digital score is not as expected and this leads us to reflect and should be studied more in deep: it is one of the lowest, even if, as mentioned, the value is very close to the others. All the south Italy and the islands perform good: the companies in those regions have a positive value of the digital score, and Sardinia, despite being an agricultural region and breeding, reaches the highest value ever, also because of the low number of companies on the web.

The Italian map resulting from these analyses, in Figure 4.8 is completely different from the one in which it showed in page 45 (Figure 3.5) which plots how healthy and default companies were distributed. From the latter it was shown how the firms of the north were healthier and more robust than the southern ones, and where there was greater concentration. Here, however, although the

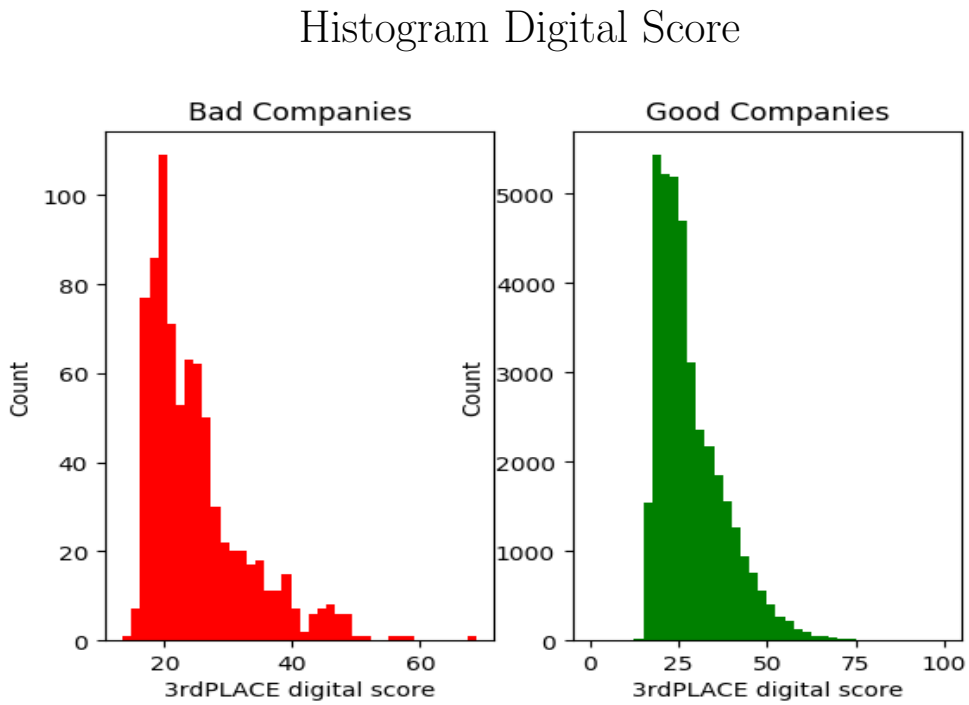


Figure 4.4: Histogram digital score for good, on the left, and bad companies, on the right. You can see that the Bad companies, on the left plot in red, have a more left-focused distribution with respect to the good ones, in blue: the mean of the Bads is less than the Goods.

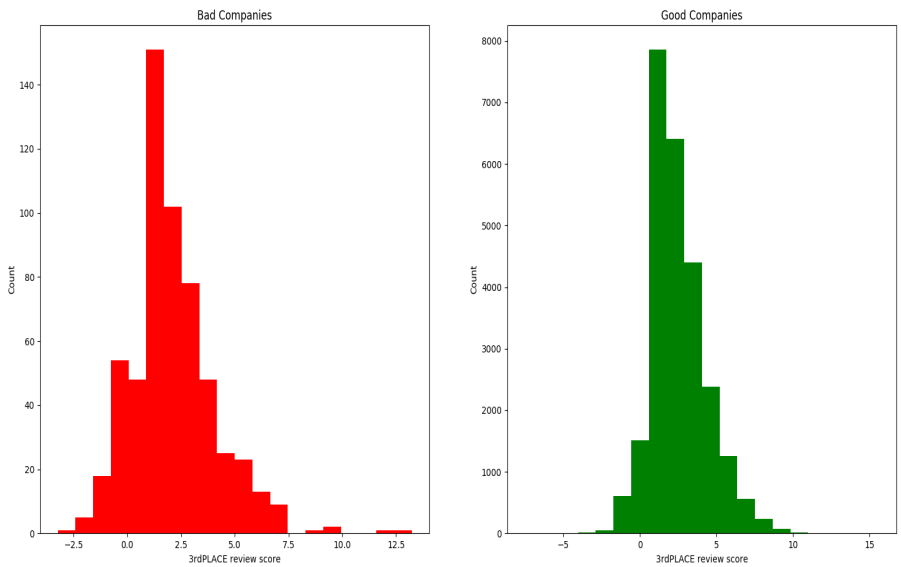


Figure 4.5: Histogram reviews score for good, in blue, and bad companies, in red. It is obvious, from the figure, that the first group get in mean a good score for the reviews: hight value and lots of ratings. We can see that the good companies achieve grater values in module of review score. The Reviews score is is incorporated in the digital score, and it is a component that has a very large weight inside it.

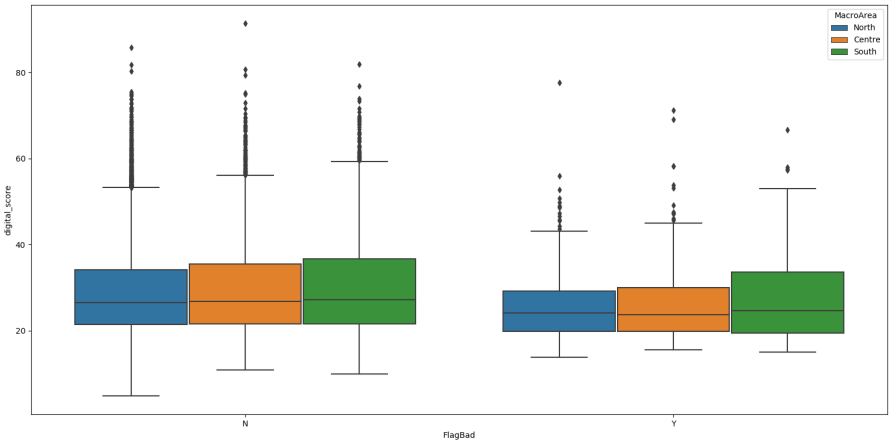


Figure 4.6: Boxplot Digital Score grouped by Macro Area. Blue is the north, orange the center and green the south. The three macro area are grouped by FlagBad: in left there are companies healthy, which looks that have a digital score higher in all the are than the respectiveness of the bad companies, grouped on the right side.

number of companies that have a digital score is greater in the north, similarly to the greater number of healthy companies in this region, they have a lower value in the digital score.

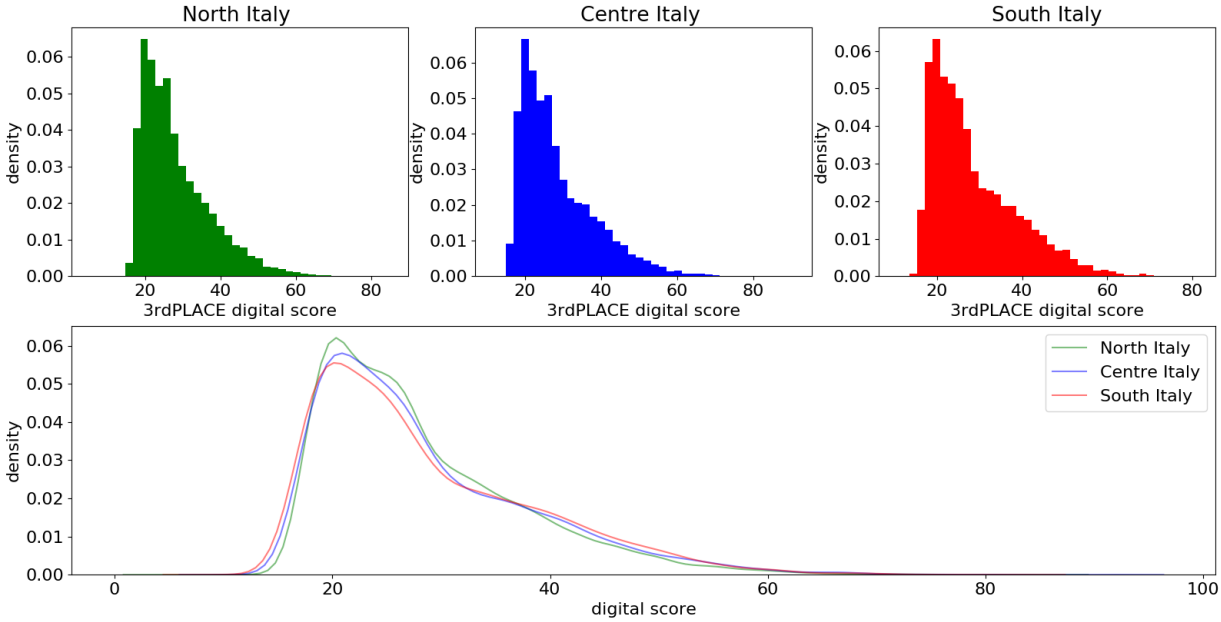


Figure 4.7: On the top we can see the density plot of digital score grouped by macro area. On the bottom, the densities are juxtaposed for better see if there are some differences.

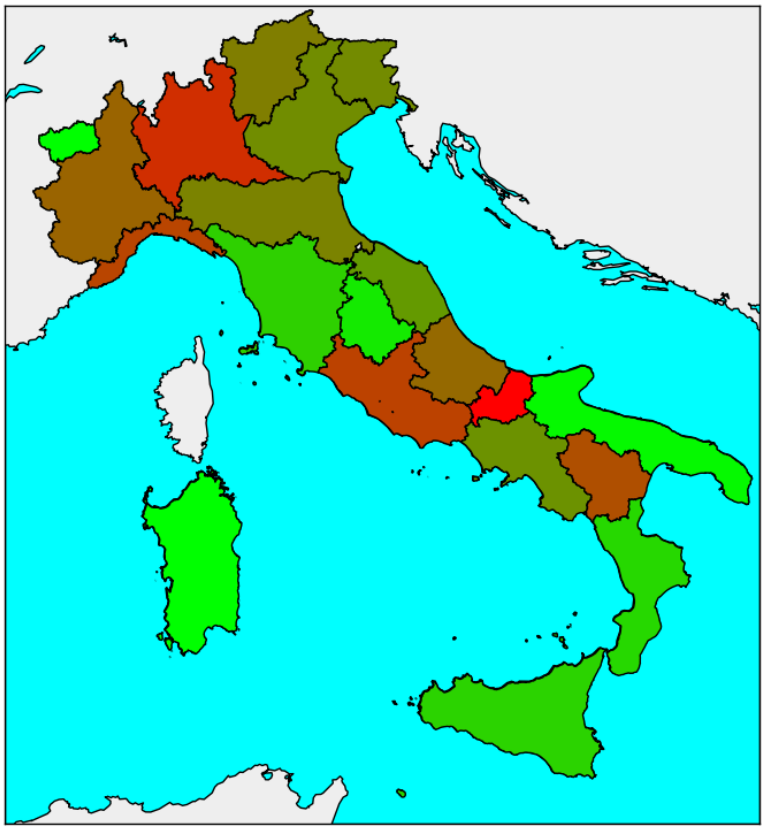


Figure 4.8: Geographical distribution of Digital Score. Regions that have low value of the digital score is colored in red, while the green indicates that its value is higher. Despite the different colors , the mean of the score is quite equal in all the regions, around 29.

Region	Count	mean	Region	Count	mean
Abruzzo	826	28.882	Piemonte	2614	28.831
Basilicata	262	28.624	Puglia	2158	30.352
Calabria	846	30.019	Sardegna	593	30.451
Campania	1272	29.302	Sicilia	1396	29.970
Emilia-Romagna	4685	29.146	Toscana	3139	29.935
Friuli-Venezia Giulia	1059	29.222	Trentino-Alto Adige	1071	29.093
Lazio	3162	28.506	Umbria	653	30.192
Liguria	750	28.534	Valle d'Aosta	85	30.414
Lombardia	9461	28.282	Veneto	5313	29.244
Marche	1420	29.380	ESTERO	13	24.616
Molise	150	27.826			

Table 4.6: Digital score count and mean in each region. Lombardia is the area with the highest concentration of firms on internet but with the one of the smallest mean value. ESTERO, the foreign countries, have a very low mean and numbers.

Chapter 5

Classification Model

This Chapter contains the experimental results of this thesis and describes how the classifiers are trained for predicting the default of companies. The data were split in two groups: 80% of the observations are part of the training set and the other 20% are in the test set. This spilt was done both for companies that have a balance sheet and for those that do not have it. Therefore, the training set and the test set consist respectively in 50333 and 12585 companies that have a balance sheet file, and in 57359 and 14340 companies that do not have it. On 80% of the data in the training set, then, 5-fold cross-validation is used for doing model selection and for considering the best values of the hyper parameters. Each model is built both for companies that have a balance sheet and for those that do not have that information. The machine learning models works also for companies that are not present on the web, and they do not have a digital identity. In this case, indeed, the digital score is zero, meaning that there are no information from the web.

Section 5.1 shows the performance of the client company model. In Section 5.2 gradient tree boosting models are fitted. First we use all the financial features, and hence all the data the client company provided to us, then we include the digital score and finally we use all the digital features we have downloaded without combining them. This section is used also for feature selection for the other classifiers we built. In Section 5.3 support vector machines are trained with the digital score and then with the digital score disaggregated, with all the features that compose it. Section 5.4 describes neural networks classifiers and how was chosen the hyper-parameters. We use both the digital score, and all the features that compose it. Finally, Section 5.5 summarizes the performance of the three models built and compare them to the client model.

Model	Budget	Accuracy	Precision	Recall	F1
Client	Y + N	0.945	0.155	0.129	0.141

Table 5.1: Performance of client company. This model is considered as a black box because we do not know any assumption that they take for building it. These performance is both for companies that have a balance sheet and for those that do not have it. They do not provide us the area under the curve, a measure that instead we use as a reference.

5.1 Performance Client Model

Before training any model, we recap here the performance of the client model. We do not have any information about it, and therefore we consider it as a black box model. In Table 5.1 we can see the performance of the model that the client company use. This model works for both companies that publish a balance sheet, and for those that do not have the file containing the financial information of the firm. They provide us the accuracy, precision, recall and F1-measure, which are 0.945, 0.155, 0.129 and 0.141 respectively. These are the target performance for the model that we develop in this Chapter, and, therefore, the goal is to improve them and provide to the client company a model that performs better.

For model selection and for hyper-parameters optimization, we do not use these measures, instead we use the area under the curve, AUC, even if they do not give us this measure for their model. One reason of having selected this measure is because the performance of the client model were provided to us at the end of the project, thus the comparison between their model and the one we are going to train in this Chapter is done after having chosen our best one. Furthermore, as we have said in Chapter 2, the ROC curve is the most commonly used way to visualize the performance of a binary classifier, and from it we can compute the area under the curve, which summarises the performance in a single number. We also use the AUC score as a reference measure because it is insensitive to unbalanced classes.

5.2 Gradient Tree Boosting

In this section the Gradient tree boosting (XGBoost) models are fitted for the classification of companies default. The models are split in two groups, one for companies for which we have a balance sheet and the other for those without it. For each group, moreover, we first fit a model with all features that the client company provided to us, then we add the features downloaded with the web crawler, both using the digital score as single value and using all the variables that compose it. The final model will be the combination of the best model for each group.

Settings	Value	Settings	Value
n_estimators	35	subsample	0.75
objective	binary:logistic	max_depth	4, 5
min_child_weight	11	silent	1
learning rate	0.07, 0.08	colsample_bytree	0.7
missing	np.nan	seed	1337
n_jobs	-1	scale_pos_weight	(len(y) - y.sum()) / y.sum()

Table 5.2: Settings for XGBoost classifier. The multiple parameters, like the depth of a tree, are chosen according to 5-fold cross validation. Furthermore, different models are built, depending on the presence or absence of the digital score and the balance sheet: in each model a backward feature selection is performed.

It is used the *xgboost* [1] python package, and in particular *XGBClassifier*. All the variables which have more than 70% of null values are not considered in the models and they are dropped out. In Table 5.2 we can see the settings used for training *xgboost*, and in particular those worthy of explanations are:

- **n_estimators**: number of boosted trees to fit;
- **learnig_rate**: boosting learning rate;
- **objective**: specify the learning task and the corresponding learning objective or a custom objective function to be used;
- **max_depth**: maximum tree depth for base learners;
- **silent**: to print messages;
- **subsample**: subsample ratio of the training instance;
- **colsample_bytree**: subsample ratio of columns when constructing each tree;
- **scale_pos_weight**: balancing of positive and negative weights.

During the run of XGBoost a model reduction is done using the feature importance. All the features that have an importance less than a certain threshold (here it was chosen 6%) are dropped out and a new model is fitted with the remaining ones. The importance of a variable in a tree is computed according to some gain score, here the Gini impurity, when splitting a leaf. As will be seen more clearly later, this policy reduces the overfitting on the data with the consequent decreasing of the prediction error, and an increasing of the performances.

[1] `http://xgboost.readthedocs.io`

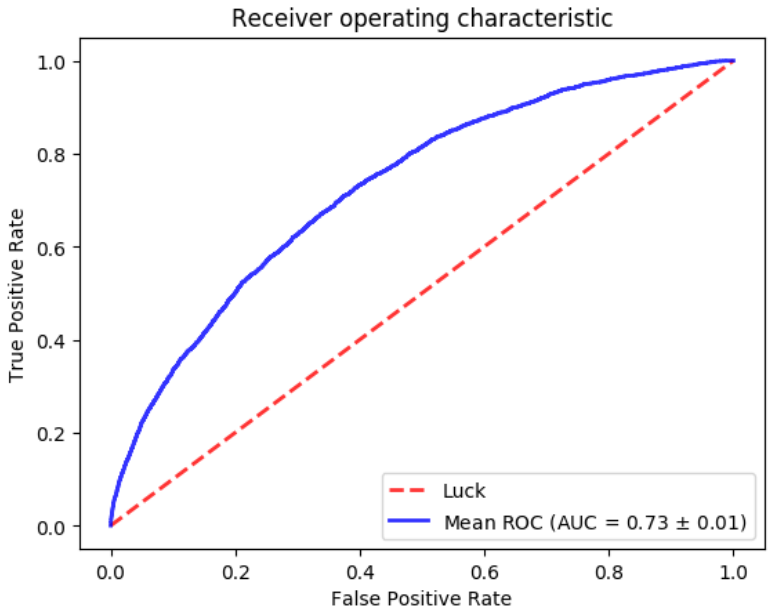


Figure 5.1: ROC curve of the best Gradient tree boosting model without the information of the balance sheet and with the presence of the digital score. In the red dot line it is represented the ROC curve for the random model, therefore the model built has an applicative validity. The area under the curve is 0.73, and therefore this model performs better than the random guess classifier.

Finally, the best model will be chosen through the area under the ROC curve, AUC, and given the same, by the smaller number of variables involved.

For building the models the dataset was split, as already mentioned above, in the training and test set. The latter will be used for the final model evaluation of the performance, that we will discuss in Section 5.5.

Absence of a Balance Sheet

In our dataset, companies that do not have a balance sheet represent 52% of the total, with 71699 observations. Of those, after the splitting of the training and test set, we consider 57359 firms for fitting models for companies that do not have a balance sheet. In this section we are going to create a classifier for them, first with only the data the client company provided us and then integrating also the digital score we developed so far. Moreover, we fit a model that uses all the features that compose the digital score, in order to prove if its weights were chosen in a good manner and the intuition was correct or not.

As we are not using any information on the financial situation of the companies the features available without consider the digital score are only 85 for building the classifiers, and they became 86 with the digital score and 99 if we consider the features that compose it. After applying the feature selection, the variables

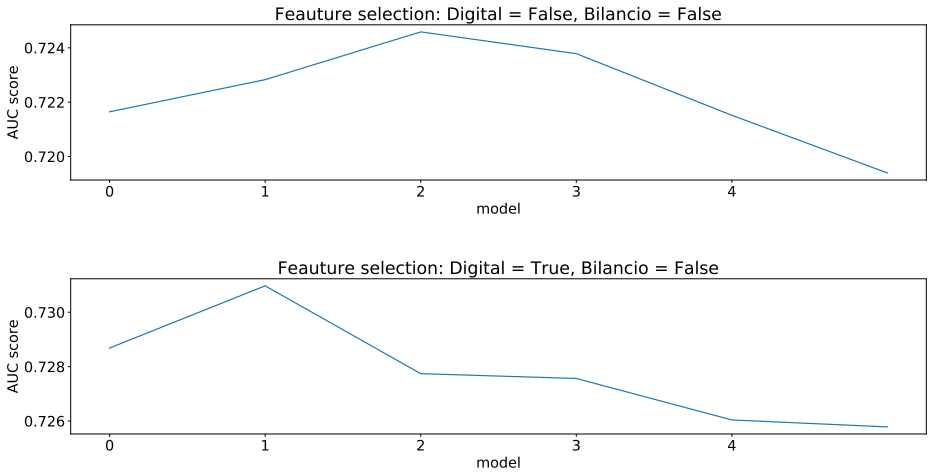


Figure 5.2: Model Selection XGboost for models without the Balance sheet. In the coordinates there is the step model, computed with the policy of backward selection of the features, while in the ordinates there is the area under the curve AUC for each model: on the top the classifier without the digital score, on the bottom with the presence of the digital score. We can see that in a single o few steps the gradient boosting achieves the best perfomance, which are 0.724 without the digital score feature and 0.731 with its presence.

really important are around 30 for all the models, and they were selected through the policy of backward feature elimination,.

In Table 5.3 we can see the steps of the backward feature selection for choosing the best model, and the top ten important variables for the Gradient Tree Boosting classifier without using the digital score. The best model is satisfactory with the area under the curve of 0.72, which is much higher than the 0.5 of the random model, with a precision of 0.07 and a recall of 0.62. The features that take part into the model are only 29. The difference between registration date on the business register and the starting date of its activity is the most influential variable in the gradient tree boosting model. Also their absolute value are important as they are in the fifth and sixth position. The second most important feature is ATECO 2017 which is a classification of the economic activities from ISTAT. The geographical position, latitude and longitude, are also important and in addition to those we have also the city in which the firm works.

Next we fit the classifier using the digital score. As we can see in Table 5.3, the model achieves higher performance than the previous one, even if the improvement is not so clear: we have an AUC score of 0.73, using 37 features. The precision and the recall of the best model are 0.078 and 0.63 respectively, while the ROC curve is shown in Figure 5.1. The optimal model is achieved in few steps, as we can see in Figure 5.2, but it uses more features than the model without the digital score. In this specific case, it is not very important because the input features are few. The digital score is on the top ten of the importance feature list, meaning

XGB model - no budget, no digital score

Step	Number of features	AUC	Precision	Recall
0	85	0.72164	0.07614	0.61074
1	35	0.72283	0.07577	0.60798
2	**29**	**0.72459**	**0.07712**	**0.62407**
3	26	0.72379	0.07751	0.60661
4	24	0.72152	0.07633	0.61533
5	22	0.71939	0.07519	0.61258

	Feature	μ	σ
1	DATA_ISCRIZIONE_INIZIO_ATTIVITA_days_diff	0.0688	0.00518
2	ATECO07_CD_encoded	0.0676	0.00714
3	LATITUDINE_DD	0.0674	0.00538
4	LONGITUDINE_DD	0.0673	0.00776
5	DATA_ISCRIZIONE_days	0.0643	0.00326
6	DATA_INIZIO_ATTIVITA_days	0.0627	0.00392
7	COMUNE_DESC_encoded	0.0463	0.00478
8	esponenti_dieci_DATA_days_mean	0.0455	0.00521
9	RAE_encoded	0.0441	0.00600
10	INDIRIZZO_CIVICO_COMUNE_encoded	0.0423	0.00397

Table 5.3: Gradient Tree boosting model summary for companies which do not have a balance sheet, without the digital score. Above there are the steps of the backward feature selection, below the top ten feature importance list of the best model, the one obtained in the 2nd step. Next to each feature there is the mean of importance in the model and the standard deviation.

that contains some information for the prediction of companies default.

This first attempt to measure the presence and the users opinions of a firm on the web and the importance that have in the the conterpart risk problem, encourages us to continue to develop the score.

As seen also in the previous Chapter, the companies that are good have a lot of interest in taking care of their image on internet and also receive good reviews from users. Therefore a high value of the digital score implies a greater probability that the firm is good. The top ten most influential features are almost the same with the top one is the latitude of a company, while the longitude is also important but in fourth position. Also the registration and the staring dates of its activity are influential, as ATECO 2017 and RAE. A feature that was not present in the previous top ten list, without using the digital score, is the CAP of the firm.

It was decided next not to consider the digital score as a single feature, but all the ones that composed it, in order to validate it or, if the new model performs better, to recalibrate the weights in the digital formula. Table 5.6 shows the results of the third classifier. We have an improvement on the AUC score, meaning that, considering the digital score as a single value for gradient tree boosting, we loose some information and hence we have worst performance. The precision and the recall of the classifier are 0.077 and 0.63. Again the most influential features are the seniority of firms, registration and starting activity dates, the geographical one with latitude, longitude, CAP and city, and ATECO 2017. No digital variables are on the top feature list and the first one, the review score, is in the 33rd position.

XGB model - no budget, yes digital score

Step	Number of features	AUC	Precision	Recall
0	86	0.72868	0.07622	0.62269
1	**37**	**0.73097**	**0.07823**	**0.635094**
2	30	0.72774	0.07808	0.63280
3	26	0.72756	0.07735	0.62270
4	23	0.72604	0.07711	0.61856
5	21	0.72578	0.07660	0.619015

	Feature	μ	σ
1	LATITUDINE_DD	0.0635	0.00659
2	DATA_INIZIO_ATTIVITA_days	0.0627	0.00712
3	DATA_ISCRIZIONE_INIZIO_ATTIVITA_days_diff	0.0624	0.00257
4	LONGITUDINE_DD	0.0586	0.00546
5	ATECO07_CD_encoded	0.0579	0.00429
6	DATA_ISCRIZIONE_days	0.0570	0.00322
7	esponenti_dieci_DATA_days_mean	0.0439	0.00372
8	RAE_encoded	0.0428	0.00394
9	CAP_encoded	0.0426	0.00230
10	**DIGITAL_digital_score**	0.0411	0.00297

Table 5.4: Summary of the Gradient Tree boosting model for companies which do not have a balance sheet, including the digital score. Above there are the steps of the backward feature selection, below the top ten feature importance list of the best model, which is in the 1st step. Next to each feature there is the mean of importance in the model and the standard deviation. The digital score is the 10th most influential feature of the best model, indicating that this variable is useful in the prediction of default for companies.

XGB model - no budget, digital score disaggregated

Step	Number of features	AUC	Precision	Recall
0	99	0.73254	0.07820	0.63509
1	41	0.73363	0.16319	0.63831
2	**33**	**0.73416**	**0.07744**	**0.63831**
3	28	0.73279	0.07779	0.63831
4	25	0.72869	0.07588	0.62085
5	23	0.73111	0.07691	0.63371
6	21	0.73006	0.07702	0.62682
7	20	0.72197	0.07496	0.61947

	Feature	μ	σ
1	DATA_INIZIO_ATTIVITA_days	0.0665	0.00498
2	LATITUDINE_DD	0.0644	0.00191
3	DATA_ISCRIZIONE_INIZIO_ATTIVITA_days_diff	0.0622	0.00315
4	ATECO07_CD_encoded	0.0620	0.00253
5	LONGITUDINE_DD	0.0590	0.00362
6	DATA_ISCRIZIONE_days	0.0573	0.00198
7	esponenti_dieci_DATA_days_mean	0.0447	0.00552
8	CAP_encoded	0.0437	0.00436
9	RAE_encoded	0.0422	0.00433
10	COMUNE_DESC_encoded	0.0411	0.00309
⋮	⋮	⋮	⋮
33	review_score	0.0254	0.00315

Table 5.6: Summary of the gradient tree boosting model for companies which do not have a balance sheet, with the digital score disaggregated. Above it is shown the the backward feature selection steps, and below the top ten feature importance list of the best model, which is in the 2nd step. Next to each feature there is the mean of importance in the model and the standard deviation. The most influential feature which composed the digital score we can find in the 33th position and there are none in the top ten most influential.

The three best models built so far have in common most of the ten more influential variables, with few exceptions. The geographical position, latitude and longitude, as we expected, are an important indicators for the Gradient Boosting: there are more virtuous areas where it is convenient to operate and carry out activities, and those less, concentrated mainly in the south. This is also confirmed by *CAP_encoded*, which are the first four digits, representing the province in which the business operates. The selected model for companies that do not have a balance sheet is therefore the one fitted with all the digital features, containing 33 variables (Table 5.6).

The last classifier, with all the features from the web without combining them, with the setting discussed before, is the selected model for companies that do not have a balance sheet, and, from now on, we refer to it when we discuss about gradient tree boosting.

Now, we discuss briefly on the choice of the weights for computing the digital score in equation (4.2), through the comparison between them and the ones obtained from gradient boosting. On one hand, the weights of the score was chosen with prior and subjective consideration, in Table 4.5; on the other, the importance of a feature, provided by gradient boosting, can be used as weights and depends solely by the algorithm and data. We can, therefore, compare the relative weights and the rank importance of the features for the default classification. In this discussion we use the classifier in Table 5.6 and step 0, which does not use the digital score as a single feature but use all the features that compose it. The weights are computing according to the training algorithm and they are not fixed a priori. Furthermore, we first normalize both the prior and the gradient boosting (XGB) weights, such that their sum is 1 respectively.

As we can see in Table 5.7, there are three different values. Gradient boosting, indeed, assigns more weights to *review score* and $numbers\ of\ information$ compared to what was assigned a priori; while *has webpage* has no influence on the prediction. The other features, instead, have weights comparable with those assigned a priori.

The weights that gradient boosting assigns to the digital features depend on the model, and if we change its input features, the weights change as well. This discussion highlights the differences between the a priori weights, with those obtained from gradient tree boosting, and that could be useful in the future development of the digital score.

Feature	**Rank**	**Digital weight**	**XGB weigth**
review score	18	0.1092	0.3031
numbers of information	20	0.1092	0.2523
total count	31	0.0420	0.1143
row count	34	0.0840	0.1025
presence site in my business	35	0.1176	0.0925
position	37	0.0840	0.0681
Google+	63	0.0252	0.0139
Facebook	66	0.0672	0.0138
Youtube	67	0.0336	0.0138
Linkedin	71	0.0672	0.0116
Instagram	76	0.0588	0.0070
Twitter	77	0.0420	0.0047
Pinterest	80	0.0252	0.0023
has webpage	88	0.1344	0

Table 5.7: Comparison between the digital and the gradient tree boosting (XGB) weights normalized. The Table shows that the algorithm gives to $review\ score$ an higher importance than the digital formula, while it does not give any importance to $has\ webpage$. XGB weights refer to the mean importance of the feature in the gradient tree boosting model for companies that do not have a balance sheet, and with all the feature (model in Table 5.6 and step 0). The rank column gives the rank importance of the feature within the XGB model.

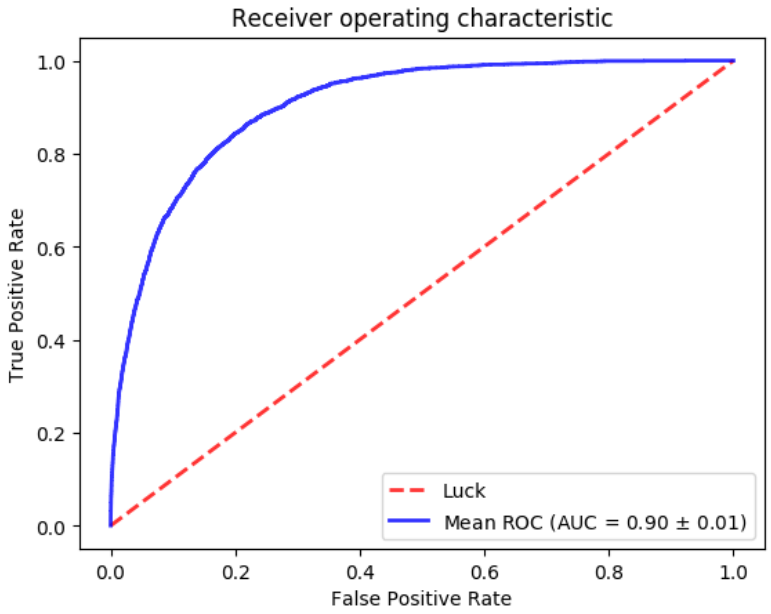


Figure 5.3: ROC curve the best model chosen in step 13 with the presence of the Digital score. With the use of a hundred of features we obtain a model with an area under the curve of 0.90, an excellent result compared with the model built with the absence of balance sheet. The informations stored there are therefore very important for saying if a firm is good or bad.

Presence of Balance Sheet

In this section gradient boosting tree model is built for firms that have a balance sheet, which represent about half of the data we hold, with 62918 records: again 80% of them are used in the training and validation set. The features linked to the financial statements represent 95% of the all attributes. Indeed, if before we could make use of only 100 features at the beginning of the backward feature selection, here 1782 variables are available and most of them are the result of the feature preprocessing.

We fit three models: one without the use of the digital score, the other introducing it in the model, and then with the digital score disaggregated, that is with the features that compose it. So we have availability of 1783 features with the introduction of the digital score and 1796 features for the model with the digital score disaggregated.

In Table 5.9 there is the summary of the model building with the backward feature selection, without using the features coming from internet. We can see that the best model uses 78 features and achieve an AUC score above 0.9, higher compared with those for companies without a balance sheet, and a precision and recall of 0.16 and 0.8 respectively. The economic situation of a firm hence is really useful for predicting the default, as we can expect, and the gain is about 0.2

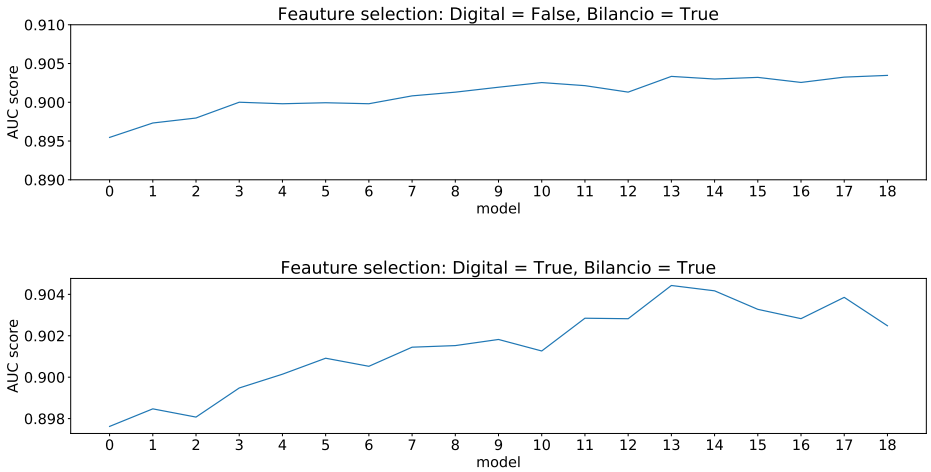


Figure 5.4: Model Selection XGboost: companies with financial features. In the coordinates there is the step model, computed with the policy of backward selection of the features, while in the ordinates there is the area under the curve AUC for each model: on the top there the digital score is not used, on the bottom it is used as input. To achieve the best performance we need more steps than the previous models which do not consider the balance sheet features. The best performance are 0.903 without the digital data and 0.904 including it.

points in the AUC score. As for companies for which we do not have the financial features, the starting activity date is the most influential feature, meaning that the seniority of companies are strongly correlated with the default.
The activity starting date and the geographical information, longitude and latitude, are also in the top ten feature importance list. The presence of the activity starting date in this list can be explain by the fact that the most critical days for a company happen with the beginning of its activity, when it still has to create contacts with customers and receive trust from them. The other features come all from the economic situation of a company, mainly in the last two years. We have self-covering of fixed assets in second position, and then there are the interest, the net asset and the asset turnover that are very influential. The delay between the closure of the balance and its publication, $lag\_deposito\_days$, of the last 2 year is also meaningful for the prediction problem.

After the first classifier, a gradient tree boosting model is fitted using the digital score. In Table 5.9 there is the summary of that model, while in Figure 5.3 we can see the ROC curve for the best setting. With the introduction of the digital score we have an improvement in the area under the curve, even if only slightly. The best model uses 116 features, a bit more than with the previous one, while the best performance is achieved in 13 steps, before with respect to the model that do not use the digital score (Figure 5.4). The important fact here is that even in this case, the digital score is in the top ten most influential features, in the 7th position, hence in somehow, it is correlated with the default of a company.

XGB model - yes budget, no digital score

Step	Number of features	AUC	Precision	Recall
0	1782	0.89547	0.16546	0.76382
1	1013	0.89733	0.16198	0.79118
2	694	0.89797	0.16066	0.78118
3	539	0.90001	0.16449	0.78275
4	431	0.89981	0.16317	0.79432
5	351	0.89994	0.16561	0.78223
6	295	0.89981	0.16332	0.79748
7	252	0.90083	0.18224	0.73700
8	224	0.90130	0.18146	0.73752
9	200	0.90194	0.16135	0.80011
10	175	0.90254	0.16580	0.79432
11	156	0.90214	0.18781	0.73278
12	141	0.90131	0.16122	0.80327
13	126	0.90334	0.18312	0.75540
14	114	0.90300	0.16219	0.80222
15	102	0.90321	0.16578	0.80326
16	93	0.90256	0.17933	0.75645
17	85	0.90325	0.16484	0.80065
18	**78**	**0.90347**	**0.16026**	**0.80853**

	Feature	μ	σ
1	DATA_INIZIO_ATTIVITA_days	0.0244	0.00278
2	AUTOCOPERTURA_IMMOBILIZZAZIONI——prev_00_years	0.0209	0.00535
3	LONGITUDINE_DD	0.0195	0.00195
4	COPERTURA_INTERESSI——prev_00_years	0.0189	0.00159
5	lag_deposito_days——prev_02_years	0.0180	0.00221
6	PATRIMONIO_NETTO——prev_00_years	0.0175	0.00141
7	LATITUDINE_DD	0.0172	0.00164
8	STATO_ATTIVITA_CD_encoded	0.0161	0.00162
9	DATA_ISCRIZIONE_days	0.0158	0.00177
10	ASSET_TURNOVER——prev_00_years	0.0158	0.00203

Table 5.9: Summary of the Gradient Tree boosting model for companies that have a balance sheet, without using the digital score. Above it is shown the backward feature selection steps, and below the top ten feature importance list of the best model, which is in the 18th step. Next to each feature there is the mean importance in the model and the standard deviation. For companies that have the financial features we achieve an area under the curve above 0.9, meaning that they are very important for the default prediction.

Model	Budget	AUC	Precision	Recall	F1
Gradient tree boosting	Y	0.905	0.185	0.746	0.296
Gradient tree boosting	N	0.734	0.077	0.638	0.137

Table 5.10: Gradient tree boosting performance. The values refers to the best model trained. Both the best model with the presence and absence of a balance sheet, use all the features that come from internet without combining them.

There is an improvement in the AUC score for the model that uses the score based on the information founded in internet with respect to the previous one, and again this represent a valid motivation in the use and development of this methodology. The precision of the best model is 0.17 and its recall is 0.78. The starting activity date is the most influential feature and then we have the geographical information, longitude and latitude, and the economic feature of the last 3 previous years. The average payment times, in particular, appears three times, one for the year in which was gathered the data, one in the previous year and then in the 3 years before. Self-covering of fixed assets and *lag_deposito_days* are again very import features.

Finally a third model is fitted, using the digital score disaggregated. Table 5.14 describes the steps of backward feature selection and the ten variables more important for the model which has the best performance considering the area under the curve of the ROC curve. This latter model, with the digital score disaggregated, has better performance than the previous one, even if only slightly, with an AUC of 0.905, precision of 0.18 and recall of 0.74. The first feature that composed the score is *row count* on the 90th position, while the feature that are in the top ten list are almost the same as the ones in the previous models. We can see how the seniority of a company is a very influential factor, with $DATA\_INIZIO\_ATTIVITA\_days$ as the most influential feature for predict the default of a company.

Moreover it seems that also the type of activity and the area where it is located, with the *longitude*, affect the FlagBad. With regard to the attributes of the financial statements, the fixed assets $AUTOCOPERTURA\_IMMOBILIZZAZIONI$, the interests, $COPERTURA\_INTERESSI$, of the current year and average payment times of the previous three years stand out.

Finally, in Table 5.10 we summarize the performance of gradient tree boosting, for both companies for which we have their balance sheet and for those that do not have it. In both cases it is better not to set a priori the weights for the digital data, as done in equation 4.2, but to let the model to choose the best weights, and hence to consider all the information coming from internet.

XGB model - yes budget, yes digital score

Step	Number of features	AUC	Precision	Recall
0	1783	0.89762	0.16895	0.76815
1	960	0.89847	0.16651	0.76998
2	675	0.89807	0.17247	0.75625
3	514	0.89948	0.16627	0.78172
4	409	0.90014	0.17247	0.76588
5	339	0.90092	0.16773	0.77591
6	284	0.90053	0.16934	0.77972
7	242	0.90145	0.17248	0.76879
8	210	0.90153	0.16877	0.78360
9	184	0.90182	0.17261	0.77057
10	163	0.90127	0.16639	0.78055
11	145	0.90285	0.17139	0.78287
12	129	0.90282	0.16736	0.78759
13	**116**	**0.90443**	**0.17159**	**0.78702**
14	105	0.90417	0.17120	0.77982
15	95	0.90328	0.17066	0.78452
16	87	0.90283	0.16545	0.79943
17	80	0.90385	0.18578	0.74796
18	73	0.90248	0.16973	0.79205

	Feature	μ	σ
1	DATA_INIZIO_ATTIVITA_days	0.0194	0.00285
2	AUTOCOPERTURA_IMMOBILIZZAZIONI——prev_00_years	0.0164	0.00291
3	LONGITUDINE_DD	0.0162	0.00279
4	STATO_ATTIVITA_CD_encoded	0.0157	0.00141
5	TEMPI_MEDI_PAGAMENTO——prev_03_years	0.0136	0.00241
6	LATITUDINE_DD	0.0134	0.00296
7	**DIGITAL_digital_score**	0.0129	0.00161
8	TEMPI_MEDI_PAGAMENTO——prev_01_years	0.0127	0.00256
9	lag_deposito_days——prev_00_years	0.0127	0.00093
10	TEMPI_MEDI_PAGAMENTO——prev_00_years	0.0127	0.00119

Table 5.12: Summary of the Gradient Tree boosting model for companies that have a balance sheet, using the digital score. Above it is shown the the backward feature selection steps, and below the top ten feature importance list of the best model, which is in the 18th step. Next to each feature there is the mean of importance in the model and the standard deviation.The digital score appears in the 7th position of the most influential features. The score hence is is used by the model for predicting if a company is good or not.

XGB model - yes budget, digital score disaggregated

Step	Number of features	AUC	Precision	Recall
0	1796	0.89707	0.16627	0.76802
1	1021	0.89857	0.16319	0.78380
2	710	0.89884	0.16645	0.77275
3	551	0.90082	0.16316	0.77539
4	439	0.90118	0.16316	0.79117
5	353	0.90039	0.16623	0.77538
6	300	0.90125	0.16180	0.79485
7	256	0.90211	0.16839	0.79011
8	224	0.90270	0.16322	0.80011
9	196	0.90289	0.90289	0.80274
10	173	0.90395	0.16818	0.79432
11	154	0.90280	0.16579	0.78801
12	138	0.90365	0.16698	0.79485
13	124	0.90477	0.16611	0.79642
14	112	0.90480	0.16405	0.80641
15	**100**	**0.90529**	**0.18563**	**0.74645**
16	91	0.90417	0.16708	0.79642
17	83	0.90379	0.16537	0.79958
18	76	0.90449	0.18171	0.74541

	Feature	μ	σ
1	DATA_INIZIO_ATTIVITA_days	0.0200	0.00316
2	AUTOCOPERTURA_IMMOBILIZZAZIONI——prev_00_years	0.0188	0.00140
3	LONGITUDINE_DD	0.0175	0.00143
4	PATRIMONIO_NETTO——prev_00_years	0.0167	0.00295
5	STATO_ATTIVITA_CD_encoded	0.0162	0.00142
6	ROI——prev_00_years	0.0153	0.00188
7	TEMPI_MEDI_PAGAMENTO——prev_03_years	0.0152	0.00152
8	lag_deposito_days——prev_02_years	0.0151	0.00130
9	TEMPI_MEDI_PAGAMENTO——prev_00_years	0.0150	0.00438
10	max_lag_deposito_days	0.0148	0.00206
⋮	⋮	⋮	⋮
90	row_count	0.00485	0.000788

Table 5.14: Summary of the Gradient Tree boosting model for companies that have a balance sheet, using the variables that compose the digital score. Above it is shown the the backward feature selection steps, and below the top ten feature importance list of the best model, which is in the 18th step. Next to each feature there is the mean of importance in the model and the standard deviation. The first feature that are part of the digital score appears in the 90th position of the most influential feature list. The digital variables then are not so important if picked up one by one.

Setting	Value
kernel	linear, RBF
C	0.1, 1, 10
γ	0.1, 1, 10

Table 5.15: Settings for the GridSearchCV SVM classifier. The hyper-parameters C and γ are chosen with 5-fold cross-validation. Furthermore, two different models are built, depending on the presence or absence of the balance sheet features.

5.3 Support Vector Machine

In this Section we report the results we achieved by developing support vector machines (SVM) for the classification problem of companies default, a completely different approach compared to what was done previously. SVM do not work with datasets that have missing values, so before starting training the model a further preprocessing and imputing phase is performed. There are various choices that can be adopted, each of them with its own pros and cons. Here we attribute missing values that are part of numeric features with the median of that variable, while for the discrete ones the mode was chosen. Often missing data possess within them information and this will be lost through the use of this technique.

Below linear and radial basis function (RBF) kernels are used for training the SVM model. Both the choice of the optimal hyper-parameters, the cost and the gamma parameters, and the evaluation of the performance for the best model, have been computed, as happened previously, with 5-fold cross-validation. Ideally, to obtain the optimal setting we have to consider all possible values for the hyper-parameters and choose the best one. This approach is not feasible in reality for support vector machines, as it would require a training time of days or weeks. What it is normally done is to consider few values on a logarithmic scale, and this approach will be followed in this work.

Support vector machines are implemented in the *Scikit-learn*[2] python package, in the SVM module. A grid search is done for choosing the best model using the parameters that are shown in Table 5.15; while the area under the curve AUC is the selected measure for the evaluation and comparison of the various models.

The gamma parameter γ of SVM defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. It can be seen as the inverse of the radius of influence of samples selected by the model as support vectors, and it is a parameter that take part only using the radial basis function kernel.

The cost C parameter, instead, trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors.

SVM requires a great deal of time for fitting and evaluating it, therefore the selection of a subset of features, that are important for the classification problem,

[2] `http://scikit-learn.org/stable`

is a need. For example, the evaluation of a SVM model with a hundred features requires more than 6 hours. A benefit of using ensembles of decision tree methods like gradient boosting is that they can automatically provide estimates of feature importance from a trained predictive model, which can be used for feature selection. We will see that this approach applied to neural networks is valid, and within them we get performance comparable to gradient tree boosting, as we will discuss in the next Section,.

An attempt to use another feature selection algorithm has been carried out, such that the selected features do not depend on gradient boosting and are optimal for support vector machines algorithm. From Table 5.14, we see that the features used by the XGBoost optimal model, for companies that have a balance sheet, are 100. We can assume, therefore, that another feature selection algorithm can at most select 150 features, while their minimum number could be 50. What has been attempted is to use the *Sequential Feature Selector* [3] (SFS) algoritmh for selecting the best features for the classification problem. In the worst case, the time that the algorithm needs for selecting a subset of important features is given by

$$t_{max} = (N - n + 1)\ CV\ t_N \tag{5.1}$$

where, in our case, $N = 150$ is the maximum number of features considered by the algorithm, $n = 50$ is the minimum numbers of features, $CV = 5$ is the numbers of folds in the cross-validation, and t_N is the running time of the algorithm to choose exactly 150 features (worst case), with no cross-validation. It turned out that t_N is more than 15 hours, which makes the use of SFS impracticable. Therefore we use the features that have been selected by gradient tree boosting models.

As previously done, two groups of models are trained: one only for companies in which a balance sheet file is not available and the other for those that have it. For each of them, we have other two models, which depend on the presence of the digital score or if they use all the features coming from internet.

First, we discuss the models of the first group, where the financial statement is not available. The model with the presence of the digital score uses 37 features: they are the relevant input for the optimal gradient boosting tree with the digital score. In Table 5.16 we can see that linear kernels performs in general better than the Gaussian ones. The best model indeed is linear with a cost of misclassification $C = 0.1$, and it is composed by 44419 support vectors divided in the following parts: 42727 support vectors are good companies, while 1692 are the bad ones. The area under the curve we get is 0.69, while the precision and the recall that we achieve is 0.063 and 0.672 respectively.

With the digital score separated instead, in Table 5.17, where we have not

[3] It is an implementation of sequential feature algorithms (SFAs), a family of greedy search algorithms, that have been developed as a suboptimal solution to the computationally often not feasible exhaustive search. `https://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/`

SVM model - companies with no balance sheet

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
RBF	0.1	0.1	0.65835	0.06849	0.38419
	0.1	1	0.55306	0.03743	0.00321
	0.1	10	0.52949	0.0	0.1
	1	0.1	0.59419	0.05467	0.10294
	1	1	0.54559	0.02337	0.00137
	1	10	0.52952	0.0	0.0
	10	0.1	0.56718	0.05310	0.04411
	10	1	0.54563	0.01904	0.00091
	10	10	0.52948	0.0	0.0

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
Linear	0.1	·	0.69029	0.06318	0.67187
	1	·	0,69004	0.06305	0.67095
	10	·	0,68988	0.06290	6704963

Table 5.16: SVM model selection without financial data. The RBF kernel seems to perform worst than the linear one. The best parameters for the SVM classifier for data that do not provide any budget feature are linear kernel and cost of misclassification $C = 0.1$. The best model has 44419 support vectors, 42727 are good companies, while the others, 1692, are companies in default.

SVM model - no budget, digital score disaggregated

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
RBF	0.1	0.1	0.69160	066315	0.40854
	0.1	1	0.54514	0.07352	0.42191
	0.1	10	0.51859	0.06394	0.16122
	1	0.1	0.58388	0.05326	0.14108
	1	1	0.53353	0.02528	0.02297
	1	10			
	10	0.1	0.55755	0.049017	0.06433
	10	1	0.53212	0.024363	0.01838
	10	10			
Kernel	Cost C	Gamma γ	AUC	Precision	Recall
Linear	0.1	·	0.69161	0.06348	0.67463
	1	·	0,69152	0.06330	0.67371
	10	·	0,69152	0.06332	0.67417

Table 5.17: SVM model selection without financial data, with digital score disaggregated. The linear kernel performs better in this problem, which with a Cost $C = 0.1$ the model achieves an AUC of 0.69. The model uses the selected features when we performed gradient tree boosting.

Model	Budget	AUC	Precision	Recall	F1
Support vector machines	Y	0.821	0.122	0.687	0.207
Support vector machines	N	0.692	0.063	0.674	0.115

Table 5.18: Support vector machines performance in the training set, computed with 5-fold cross-validation.

fixed a priori their weights, we achieve performance that are a little bit better than before, with an AUC of 0.691. Even in this case the linear kernel performs better with respect to the radial basis function. For the purpose of the classification problem, it is better to have all the digital features coming from internet separate and not as a single score. In this case, support vector machines have as input 33 features, selected as before by the gradient boosting model. The performance of SVM however is lower than XGBoost of about five points. Here the AUC does not achieve 0.70 while with the previous model it was 0.73.

Now we apply support vector machines to firms that publish their balance sheet and we make use of those features in the model: we have 116 features in this case. Table 5.19 summarizes the results of the hyper-parameters choice in SVM classifier, considering the digital score as input.

In this case the difference of using Gaussian or linear kernels is not so high, but anyway the latter still performs better. The best model uses the linear kernel with $C = 1$, and it achieves an area under the curve of 0.82 a precision of 0.12 and a recall of 0.68. The support vectors are 28443 for the good companies and 1140 for the bad ones.

Finally, we fit the SVM classifier considering also the digital score disaggregated, in Table 5.20, in which 100 features are used. In this case, unlike what we had seen before, this model that uses all the features coming from internet, with the digital score disaggregated, we achieve lower performance. The AUC reached is 0.81, with a linear kernel and a cost of misclassification $C = 10$.

With SVM, then, it is better to have a digital score as a single measure. The weights of equation (4.2) that composed it seems to perform good in this specific case. However, support vectors machines are not satisfactory, and the difference in the AUC score between gradient tree boosting and SVM models is even greater here for data that have a balance sheet respect to those in which we do not have it. If the first classifier achieved 0.90, here support vector machines have an area under the curve of 0.82, almost 0.10 lower. The performance summary are reported in Table 5.18, for all the companies in the dataset, both for those that have a balance sheet and for those that do not have it.

SVM model - companies with a balance sheet

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
RBF	0.1	0.1	0.81688	0.10646	0.74961
	0.1	1	0.71205	0.09072	0.01736
	0.1	10	0.57878	0.0	0.0
	1	0.1	0.77810	0.12189	0.29617
	1	1	0.68687	0.08956	0.00525
	1	10	0.58026	0.0	0.0
	10	0.1	0.75197	0.126465	0.13309
	10	1	0.68376	0.06838	0.00315
	10	10	0.58029	0.0	0.0
Kernel	Cost C	Gamma γ	AUC	Precision	Recall
Linear	0.1	·	0.81349	0.11561	0.68228
	1	·	0.82186	0.12173	0.68701
	10	·	0,82136	0.12314	0.68700

Table 5.19: SVM model selection with financial features. The RBF kernel seems to perform worst than the linear one. The best parameters for the SVM model for companies for which we have the balance sheet feature are linear kernel and cost of misclassification $C = 0.1$. The best model has 44419 support vectors, 42727 are good companies, while the others, 1692, are companies in default.

SVM model - yes budget data, digital score disaggregated

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
RBF	0.1	0.1	0.81185	0.10772	0.73540
	0.1	1	0.70854	0.09059	0.07259
	0.1	10	0.58117	0.0	0.0
	1	0.1	0.77246	0.12238	0.36349
	1	1	0.68295	0.10304	0.01315
	1	10	0.58096	0.0	0.0
	10	0.1	0.74764	0.12167	0.170959
	10	1	0.66818	0.11381	0.008942
	10	10	0.58060	0.0	0.0

Kernel	Cost C	Gamma γ	AUC	Precision	Recall
Linear	0.1	·	0.81007	0.11890	0.65492
	1	·	0.81774	0.12475	0.65282
	10	·	0.81966	0.12168	0.66596

Table 5.20: SVM model selection with financial data and digital score disaggregated. The linear kernel performs better in this problem, which with a Cost $C = 10$ the model achieves an AUC of 0.819. The model uses the selected features when we performed gradient tree boosting.

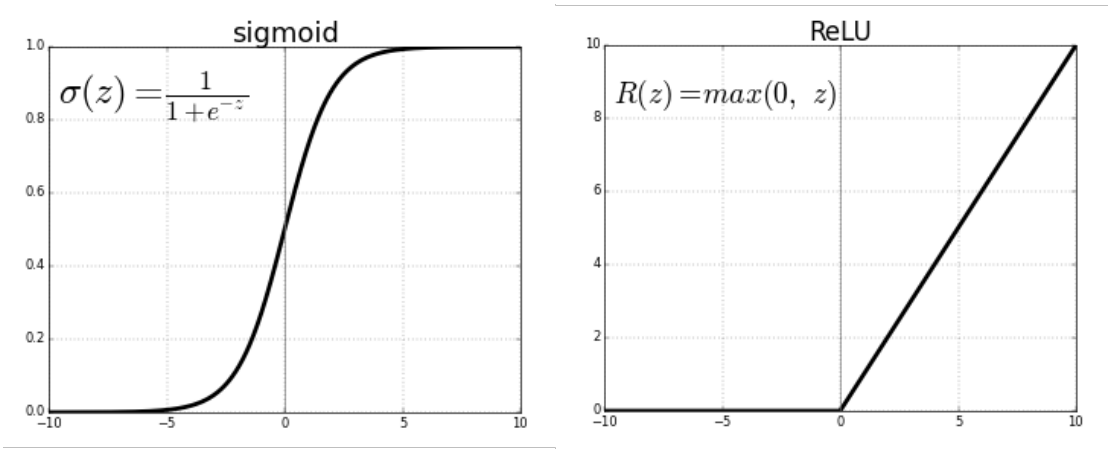


Figure 5.5: Activation functions chosen in neural networks model. The ReLU function is used in the hidden units, and the sigmoid in the output unit.

5.4 Neural Networks

In this section neural network classifiers (NET) are trained for predicting companies default. As happened with support vectors machines, they cannot work with missing data, hence the same procedure of preprocessing and imputing phase is done. We replace a missing value with the median of that feature if it is numeric and with the mode if it is discrete. Furthermore, the input features of the neural networks are the ones selected from the gradient tree boosting model, as happened with support vector machines.

In this Section, we fit two different models: one for companies that have a balance sheet and the other for the firms that do not have it. For each model, as done with SVM, we study the presence of the digital score, and the presence of all the features that compose it. The model adopted is feed-forward with dense hidden units.

The algorithm of NET classifier used in this section is in *keras*[4] python package. The number of hyper-parameters to tune is huge in a neural networks, and there is no clear theory for the optimization of them. In the tuning phase, we need to choose the following parameters: the loss function, the layers, the number of hidden units, the activation function, the epochs, the batch size and the regularization term. Considering all possible combination of them is not feasible in practice because it would require a training phase of days or weeks and this is not possible. Here we follow what in general is done in the creation of a neural network, going to consider only a subset of the previous parameters. Then, we choose them through a grid search, which is the simplest algorithm for hyper-parameters optimization. There are better choice than this method, however with this naive technique we achieve good results, hence in the following this solution is adopted.

[4] `https://keras.io/`

NET model - no financial features, with the digital score

Hidden layers	Hidden unit list	AUC	Precision	Recall
2	200 – 100	0.711	0.070	0.636
3	300 – 200 – 100	0.713	0.071	0.640
4	400 – 300 – 200 – 100	0.717	0.073	0.643
4	500 – 400 – 200 – 100	0.714	0.074	0.619
5	600 – 400 – 200 – 100 – 10	0.713	0.076	0.592
6	600 – 400 – 300 – 200 – 100 – 50	0.710	0.079	0.534
7	700 – 600 – 500 – 400 – 300 – 200 – 100	0.711	0.084	0.469
8	800 – 700 – 600 – 500 – 400 – 300 – 200 – 100	0.708	0.069	0.651

Table 5.21: Net hyper-parameters selection, companies without a balance sheet, with the use of the digital score. The best model has 5 hidden units, and achieves an AUC of 0.717.

NET model - no financial features, digital score disaggregated

Hidden layers	Hidden unit list	AUC	Precision	Recall
2	200 – 100	0.709	0.068	0.661
3	300 – 200 – 100	0.710	0.070	0.619
4	400 – 300 – 200 – 100	0.711	0.069	0.653
4	500 – 400 – 200 – 100	0.713	0.071	0.623
5	600 – 400 – 200 – 100 – 10	0.710	0.073	0.596
6	600 – 400 – 300 – 200 – 100 – 50	0.709	0.074	0.590
7	700 – 600 – 500 – 400 – 300 – 200 – 100	0.709	0.085	0.487
8	800 – 700 – 600 – 500 – 400 – 300 – 200 – 100	0.705	0.075	0.565

Table 5.22: NET hyper-parameters selection, companies without a balance sheet, model with digital score disaggregated. The best model is highlight in gray and has 4 hidden units, plus the input and output ones.

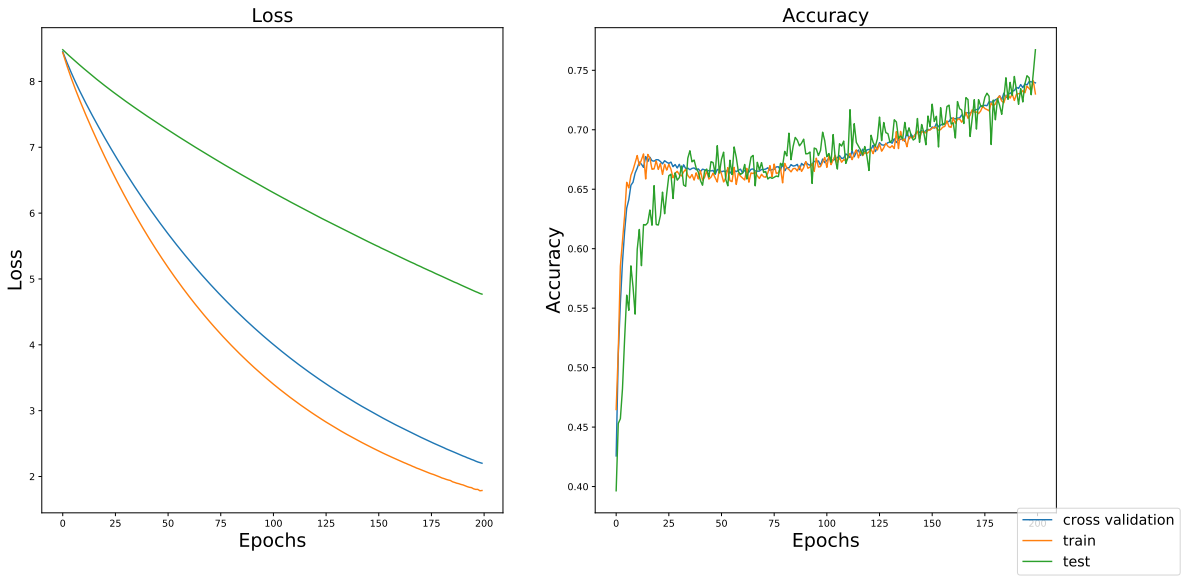


Figure 5.6: Neural networks loss and accuracy for companies that do not have a balance sheet, with the best setting. On the left the loss is reported for the net that was highlighted in grey in Table 5.21. In orange we have the loss computed in the training set; in blue the 5-fold cross validation average; and in green there are the loss in the test set. On the right side of the Figure is a plotted the accuracy of the model.

The loss function used for tuning our neural networks is the *binary crossentropy* described in equation (2.8). As activation function for the output layer the sigmoid is selected, while for the hidden units the $ReLU$ function is adopted (Figure 5.5).

These choices are widely used in the deep learning community for a binary classification problem, whereas the number of layers and the number of hidden units we keep as our hyper-parameters to select and optimize, according to the value of AUC. For evaluating the models 5-fold cross-validation is also used, as previously done with support vectors machines and gradient tree boosting.

The batch size and the numbers of epochs, then, are set to 2000 and 200 respectively. The batch size is the number of training examples in one forward/-backward pass. The higher the batch size, the more memory space is needed. The advantage of considering a batch size smaller than the size of a dataset is that it requires less memory, because we train a network using less number of samples at a time. Furthermore, networks train faster with mini-batches because we update weights after each propagation. On the other side, the smaller the batch the less accurate estimate of the gradient we have. The epochs, instead, refer to how many times an entire dataset passes forward and backward through the neural network. Often we are using a limited dataset and for optimizing the learning and the graph we use a gradient descent which is an iterative process. Updating the weights with single pass or one epoch, then, is not enough for fitting

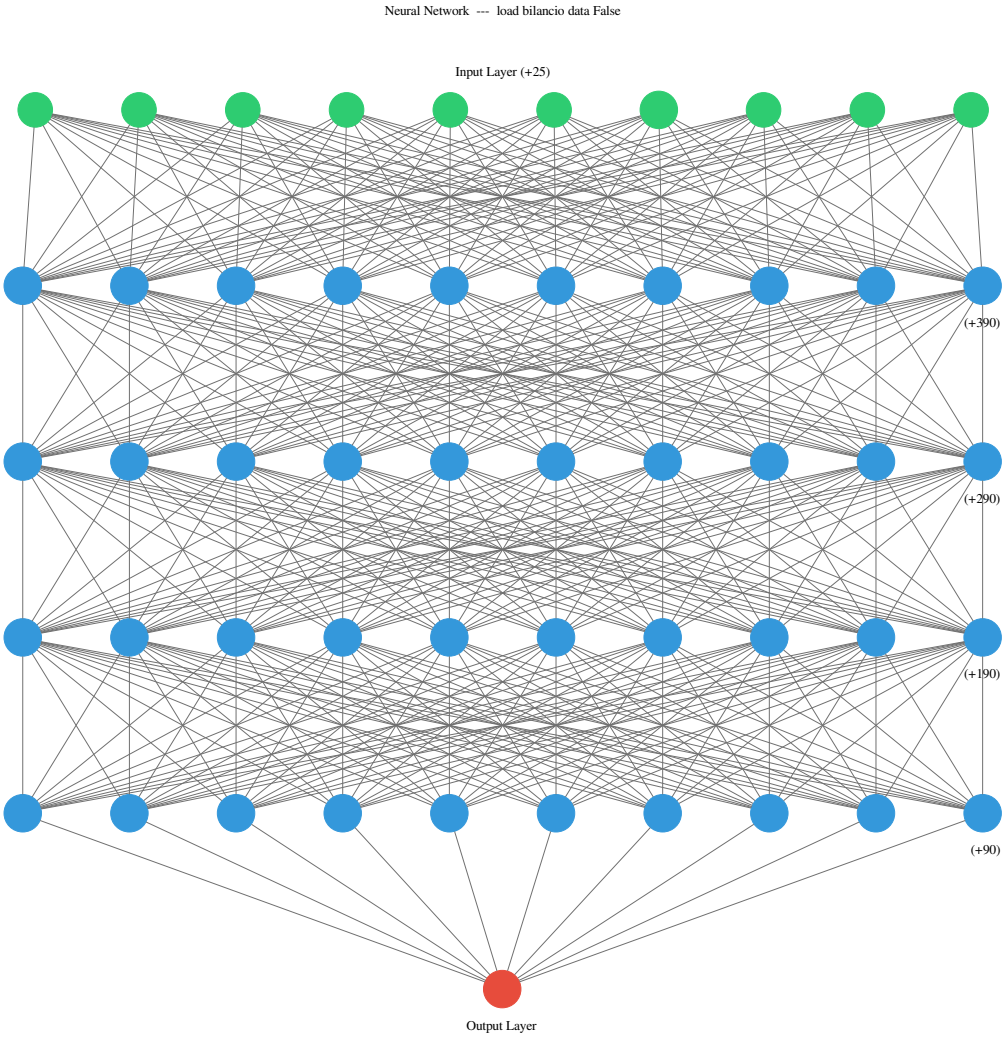


Figure 5.7: Graph neural network for companies without a balance sheet. The model have four layers, and the number of hidden units are 500, 250, 100 for the firs, second, third and fourth layers respectively

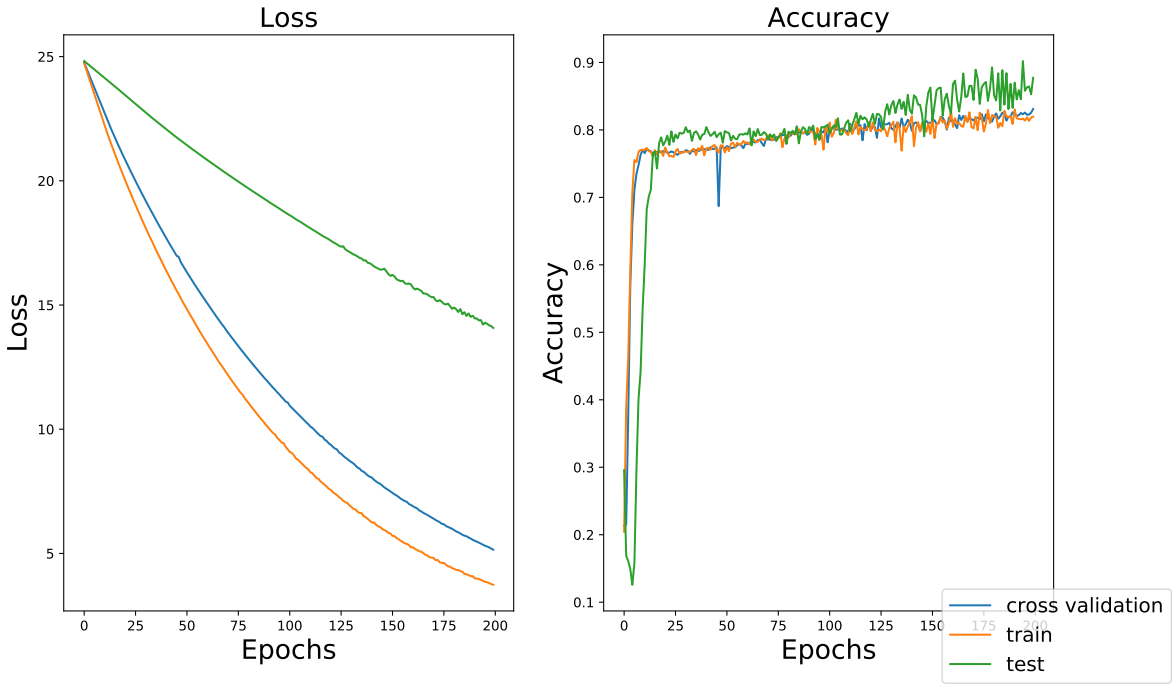


Figure 5.8: Neural networks loss and accuracy for companies that have a balance sheet. On the left the loss is reported for the net that was highlighted in grey in Table 5.24. In orange we have the loss computed in the training set; in blue the 5-fold cross validation average; and in green there are the loss in the test set. On the right side of the Figure is a plotted the accuracy of the model.

a neural network. As the number of epochs increases, more number of times the weight are changed in the neural network and the networks go from underfitting to overfitting the dataset.

The overfitting causes the validation and test performance of neural networks to be suboptimal, as it happens with the other classifiers. To avoid this, a regularization term is incorporated in the loss function that the network optimizes, which is the Frobenius norm of the weights matrix of the networks. The relative weight λ assigned to regularization in the weighted-sum objective is choosen to be $\lambda = 0.01$. Adding this type of regularization penalizes weights for being large in magnitude by contributing to the cost quadratically with it. This reduces the flexibility of the classifier, thereby reducing the overfitting phenomenon. The regularization parameter λ has to be optimized based on the error obtained with 5-fold cross-validation, and this increase the training time of the model.

First, we fit neural networks for companies for which we do not have a balance sheet, with the presence of the digital score. In Table 5.21 we can see the process of parameters optimization. The best neural network has 4 hidden layers composed by 400, 300, 200, 100 units, from the input to the output layers. This model achieves an area under the curve of 0.717 which is lower than the one got with

Model	Budget	AUC	Precision	Recall	F1
Neural Networks	Y	0.851	0.106	0.799	0.187
Neural Networks	N	0.717	0.073	0.643	0.131

Table 5.23: Neural networks performance summary in the training set, computed with 5 cross-validation when we have tuned the models.

gradient tree boosting model: it was $AUC = 0.73$. The summary representation of the graph of this model is shown in Figure 5.7, and the loss and accuracy history for each epoch is shown in Figure 5.6, computed in the training and test set, and with 5-fold cross-validation. This classifier has a precision of 0.073 and a recall of 0.643. The neural networks summary, that use all the features that compose the digital score, is seen in Table 5.22. We can see that the area under the curve in the best setting stays beyond the one with the digital score, so in this case having the score as a single numeric value leads to better performances.

Then, we turn our attention to companies for which the balance sheet is available. In Table 5.24 we see the optimization of the hyper-parameters for the neural networks that use the digital score. We see that with 7 hidden units we achieve an AUC of 0.851, a precision of 0.106 and a recall of 0.799. Loss and accuracy history for each epoch are plotted in Figure 5.8, in the training and test set and with 5-fold cross-validation, while neural network graph is in Figure 5.9. The presence of the digital score as a single value performs better even in this case as we can see in Table 5.25, where using all the features separated we obtain less performance: $AUC = 0.848$.

Finally, the summary of neural networks is shown in Table 5.23: the performance are computed in the training set using 5-fold cross-validation. NETs performs better with the digital score that we built, and this give us confidence for future developments. However, they perform worse than gradient tree boosting.

NET model - yes balance sheet, with digital score

Hidden layers	Hidden unit list	AUC	Precision	Recall
2	200 – 100	0.836	0.111	0.751
3	300 – 200 – 100	0.841	0.112	0.754
4	400 – 300 – 200 – 100	0.842	0.114	0.769
4	500 – 400 – 200 – 100	0.847	0.118	0.752
5	600 – 400 – 200 – 100 – 10	0.848	0.119	0.762
6	600 – 400 – 300 – 200 – 100 – 50	0.848	0.123	0.747
7	700 – 600 – 500 – 400 – 300 – 200 – 100	0.851	0.106	0.799
8	800 – 700 – 600 – 500 – 400 – 300 – 200 – 100	0.849	0.119	0.757

Table 5.24: NET hyper-parameters selection, companies that have a balance sheet, model with the digital score. The best model has 7 hidden layers, and a decreasing numbers of hidden units from 700 to 100, plus the output layer which has 1 hidden unit. The AUC is 0.851, so the performance is lower than gradient tree boosting.

NET model - yes budget features, digital score disaggregated

Hidden layers	Hidden unit list	AUC	Precision	Recall
2	200 – 100	0.838	0.112	0.746
3	300 – 200 – 100	0.842	0.112	0.761
4	400 – 300 – 200 – 100	0.844	0.119	0.739
4	500 – 400 – 200 – 100	0.846	0.117	0.756
5	600 – 400 – 200 – 100 – 10	0.848	0.111	0.776
6	600 – 400 – 300 – 200 – 100 – 50	0.844	0.126	0.719
7	700 – 600 – 500 – 400 – 300 – 200 – 100	0.846	0.139	0.680
8	800 – 700 – 600 – 500 – 400 – 300 – 200 – 100	0.841	0.141	0.680

Table 5.25: NET hyper-parameters selection, companies that have a balance sheet, model with the digital score disaggregated. The best model has 5 hidden layers composed by 600, 400, 200, 100, 10 units. This model has lower performance than the one we get with the digital score as a single value.

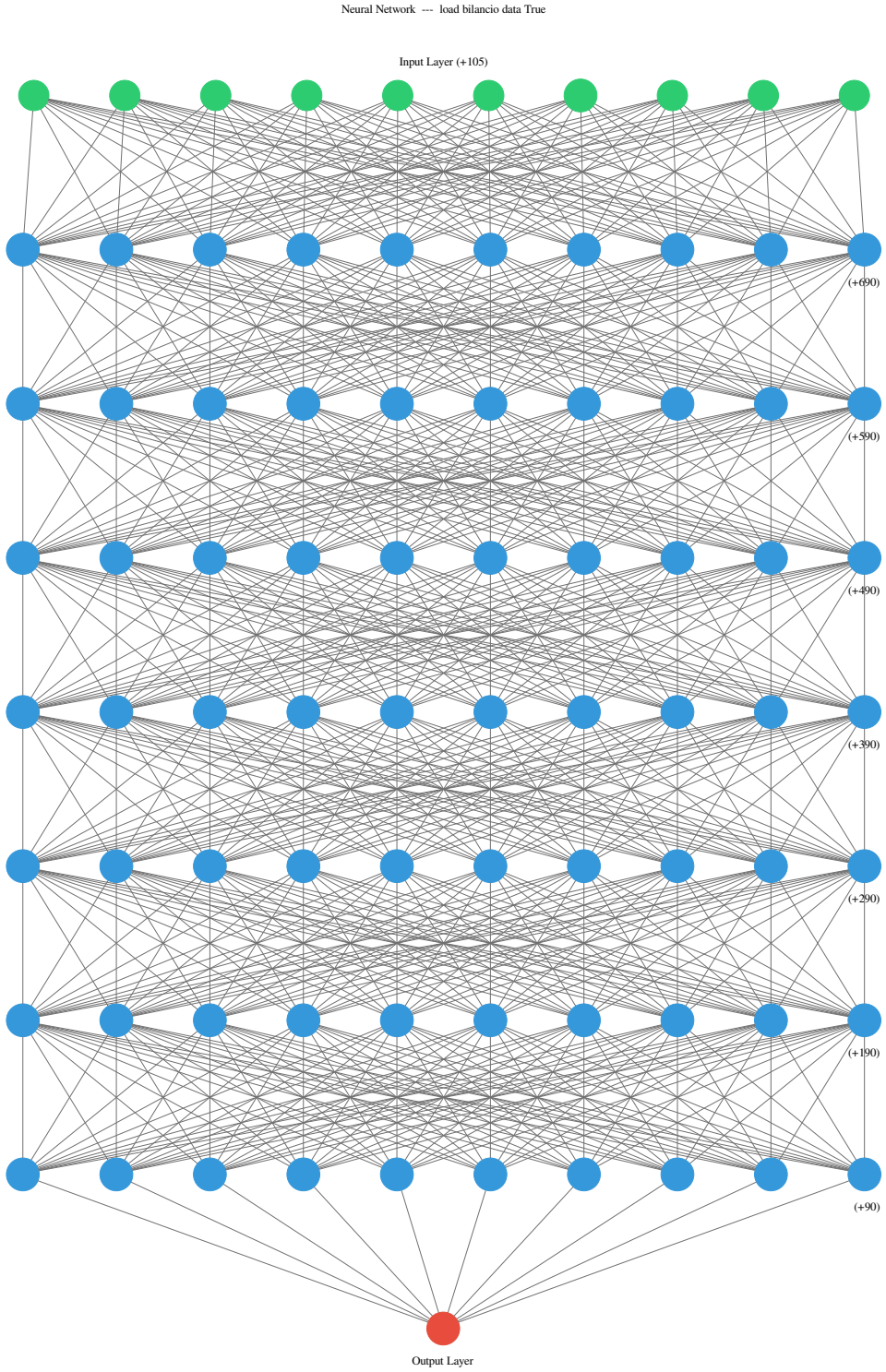


Figure 5.9: Graph neural network for companies for which is available a balance sheet. The model has 7 layers, and the number of hidden units are deceasing from 700 to 100 going from the input to the output layers.

Balance sheet	Model	AUC	Precision	Recall	F1
Y	Gradient tree boosting	0.905	0.185	0.746	0.296
	Support vector machines	0.821	0.122	0.687	0.207
	Neural Networks	0.851	0.106	0.799	0.187
N	Gradient tree boosting	0.734	0.077	0.638	0.137
	Support vector machines	0.692	0.063	0.674	0.115
	Neural Networks	0.717	0.073	0.643	0.131

Table 5.26: Models performances summary on training set. These values are computed with 5-fold cross-validation when we have tuned the models. We see that both for companies that have a balance sheet and for those that do not have it the best classifier is gradient boosting.

5.5 Models Comparison

This Section recaps the models trained during this Chapter, and summarizes their performance. We have built gradient tree boosting, support vector machines and neural networks models. We have also discussed about the performance of the classifiers: they were obtained using 5-fold cross-validation on the 80% of the training part of the dataset. The area under the ROC curve was used as a reference measure for selecting the best model.

In Table 5.26 we can see the performance of the models built, obtained using 5-fold cross-validation in the training-validation set, both for companies that have a balance sheet and for those that we do not have this information. Gradient tree boosting classifier turned out to be the best model for companies that have a balance sheet: it achieved an AUC of 0.905. The best model is gradient tree boosting even for companies that do not have a balance sheet file: in this case the area under the curve is 0.734. Neural networks and support vector machines do not perform well in both cases, having performance smaller than XGBoost model. Table 5.26 reports also the precision, recall and F1 for each model built.

Therefore, we select gradient tree boosting as the final 3rdPLACE model. Table 5.27 shows the final evaluation of the models, computed in the test set. As we can expect, they are lower than the ones obtained in the training part of the dataset, however, especially for companies that have a balance sheet, we achieve good results. Table 5.27 shows also the model performance of the client company. They do not provide us the area under the curve of their classifier, so in order to compare them we refer to F1-measure, which consider both the precision and the recall. 3rdPLACE model has an F1-measure of 0.179, greater than the client company, 0.141, with an improvement of 26.95%. The confusion matrix of the final model is in Figure 5.10, it was built considering both companies that have a balance sheet and the ones that do not have it.

Gradient tree boosting is a probabilistic models: it returns the probability of companies default. The client company provided to us the probability of their

Model	Budget	AUC	Accuracy	Precision	Recall	F1
Gradient tree boosting	Y	0.808	0.819	0.148	0.796	0.249
Gradient tree boosting	N	0.670	0.695	0.077	0.643	0.138
3rdPLACE	Y + N	0.735	0.753	0.103	0.714	0.179
Client	Y + N	·	0.945	0.155	0.129	0.141

Table 5.27: Performance of client company and 3rdPLACE models. 3rdPLACE model is the combination of gradient tree boosting, for all the companies that are in the dataset: both for those that have a balance sheet, and for those that do not have the balance sheet. The test set was used for computing the performance of 3rdPLACE model. Compared to client model, we have better recall and F1.

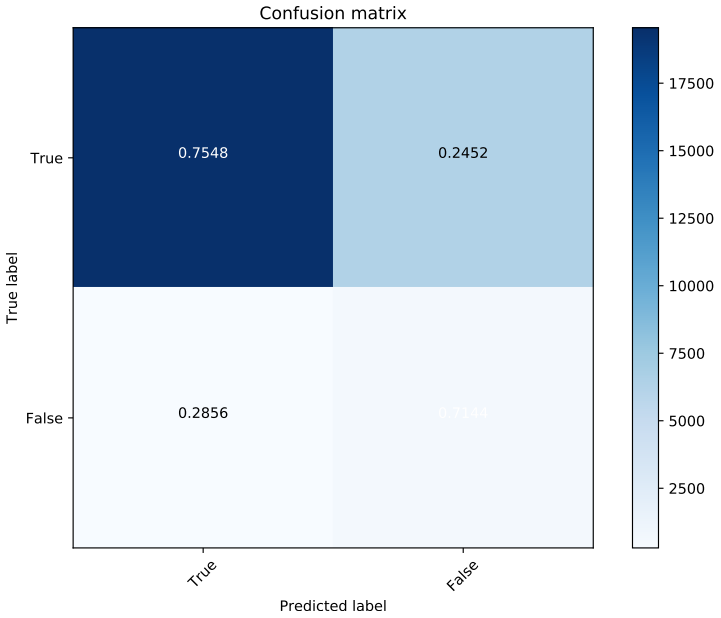


Figure 5.10: Confusion Matrix of gradient tree boosting in test set. It was built considering all companies having both a balance sheet and not.

model that a company is good, which is essentially the inverse of the probability to be in default. In order to compare them we do a simple linear transformation. In particular, the probability of being in good health for a company $\mathbb{P}(good)$ can be computed by

$$\mathbb{P}(good) = 1 - \mathbb{P}(bad) \tag{5.2}$$

where $\mathbb{P}(bad)$ is the default probability of a company. In addition to this they provided to us also how much credit they lend to companies, and this information is available only for firms that are not in default.

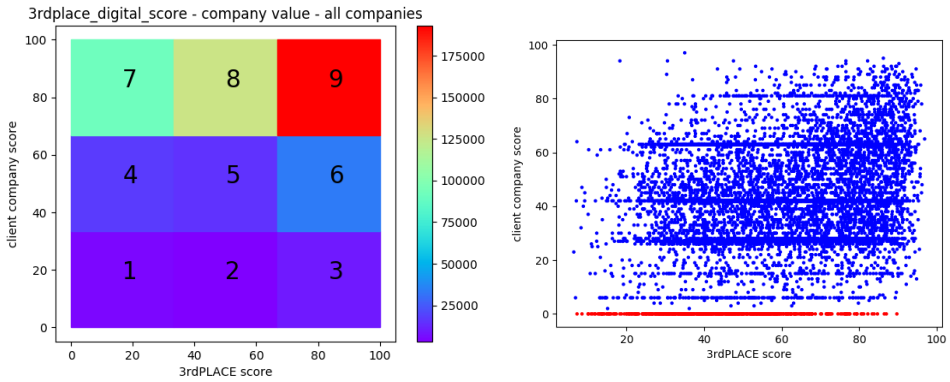
In Figure 5.11 there is a scatter plot for the probabilities of being good for the two models and they refer to companies which do not have a balance sheet. The most critical parts of the plot are the squares 7 and 3, because the two classifiers have different outputs. In the square 7 the client model says that the companies are good while the 3rdPLACE one says that they are bad; while in the square 3 it is the other way around, in which the 3rdPLACE model predicts that the companies that are in there are good, while the client model says that they are bad.

The firms that are in the square 3 are 974 and they have a mean value, the value of the loan gave to them, of 9 k€, and the sum of values of 9217 k€. In the square 7, instead, the values are higher with an average of 91k€ and the total sum of 2107 k€; however there are only 23 companies in that square. The higher values can be explained by the fact that the client company have higher score for firms in squares 7 than in 3, so they lend more money to them.

Figure 5.12, instead refers to companies that have a balance sheet. In this case square 3 of the scatter plot is composed by 1529 firms, with a total value of loan lent of 40256 k€, with a mean value for each company of 26 k€. In square 7 instead, there are 23 companies with total value of 3556 k€ and mean value of 154 k€. The higher values can be explain in the same way as before.

The two scatter plots can help the client: they can be used as a warning if a company is in the square 3 or 7, or in other words, if the client model predicts a different probability value of being good than the 3rdPLACE model.

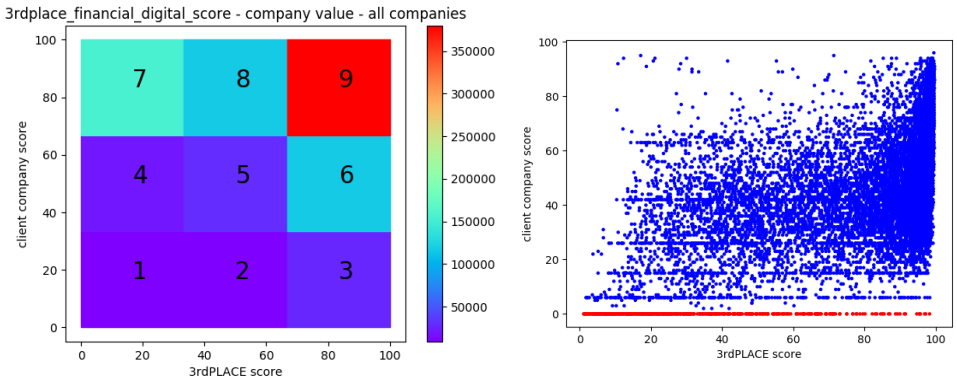
Companies without a balance sheet



face	count	mean value (k€)	total value (k€)
1	247	4.906	1212
2	1792	3.037	5444
3	974	9.463	9217
4	497	17.814	8854
5	4965	15.447	76695
6	3656	33.868	123825
7	23	91.608	2107
8	232	126.051	29244
9	634	193.391	122610

Figure 5.11: Comparison client and 3rdPLACE models for companies without a balance sheet. Low values of the scores mean that the company have a low probability that is good. Count is the number of companies that is that square of the scatter plot. The mean and the total values refers to good firms: they refer to how much money the client lent to them.

Companies with a balance sheet



face	count	mean value (k€)	total value (k€)
1	530	9.164	4857
2	896	8.618	7722
3	1529	26.326	40253
4	336	19.919	6693
5	1229	29.163	35842
6	5156	116.558	600976
7	23	154.739	3559
8	68	116.279	7907
9	2000	380.165	760330

Figure 5.12: Comparison client and 3rdPLACE models for companies with a balance sheet. Low values of the scores mean that the company have a low probability that is good. Count is the number of companies that is that square of the scatter plot. The mean and the total values refers to good firms: they refer to how much money the client company lent to them.

Chapter 6

Conclusions

This thesis proposed an innovative methodology for estimating company's credit risk; in specific, it studied counterpart risk exploiting a data driven approach combined with alternative data. Counterparty risk is a well know problem within the finance domain; practically, it evaluates the risk that the counterparty will not live up to its contractual obligation. The present work was developed during the internship in *3rdPLACE*, which is a company that offers solutions and services in the field of intelligence applied to digital data. The project was commissioned by a client company listed on the STAR segment of the Milan Stock Exchange, and that supports financial institutions, large, medium and small businesses, insurance companies, public administrations and professionals in effective credit management. The aim of the project was to build a classifier, which predicts if a company is in default or bad, or if it is in good health.

The project consisted in creating machine learning models that allow predicting companies default. The dataset they provided to us was composed by Italian companies registered at the national Companies House, for half of them the balance sheet was available, while the other half we did not have this information. The goal involved developing an innovative credit risk estimator designed to work on medium, small and very small Italian companies not quoted on exchange, using a methodology that is purely data-driven with the technologies of machine learning. Furthermore it was proposed to create a new feature, in addition to those provided by the client company: the digital score. It measures the company's presence, performance and effectiveness on the web and integrates it into the initial classification problem of default. The client company, at the end of the study, provided to us the performance of their models, and this was used as a benchmark through the whole thesis in the models development. The client wanted to compare and update their model.

The dataset the client company provided to us, the financial dataset, contained information regarding Italian companies registered at national Companies House (Camera di Commercio). It consisted in 135395 companies from all the regions of Italy. The dataset was divided into several files: the company registry, the balance sheets, and the details of the exponents. Each company was labelled as Bad or Good based on whether it defaulted or not. The balance sheet file was available for almost 50% of them, 62918 companies, while for the other 71699 we did not have these feature. The financial dataset consisted, therefore, in 103 features for the companies in the first group and 67 features for companies that do not have a balance sheet. We added to them features coming from internet through the use of a web crawler. The digital score was a combination of these information.

The models that were considered and trained in this work are gradient tree boosting, support vector machines and neural networks. They were built for companies that have a balance sheet and for those that do not have it. Furthermore, during the training of the models we have considered both the digital score as a single value and all the features that composed it, to compare if there was some difference in using one feature or all of them for the default problem. The performance of those models are reported below: these measure were computed in the training set and with 5-fold cross-validation. Gradient tree boosting achieved an area under the curve (AUC) of 0.905 and 0.734, a precision of 0.185 and 0.077, a recall of 0.746 and 0.683, an F1-measure of 0.296 and 0.137 respectively for companies that have a balance sheet file and for those that do not have it. Both of them used all the features downloaded by the web crawler without combining them in a single score. Support vector machines had respectively an AUC of 0.821 and 0.692, a precision of 0.122 and 0.063, a recall of 0.687 and 0.674, an F1-measure of 0.207 and 0.115. SVM for companies that have a balance sheet used the digital score, while the other model had as input all the digital features that compose it. Finally, neural networks had the following performance: an AUC of 0.851 and 0.717, a precision of 0.106 and 0.073, a recall of 0.799 and 0.643, an F1-measure of 0.187 and 0.131 respectively. Neural networks used the digital score in input.

In light of this and having chosen the area under the curve the measure to optimize, gradient tree boosting was selected as the final model. Its performance in the test set were an area under the ROC curve of 0.735, an accuracy of 0.753, a precision of 0.103, a recall of 0.714, an F1-measure of 0.179. These measures were computed considering all the companies.

After having chosen the model and evaluated it, the client company gave us their performance for comparing their model with ours. The performance of their classifier were the following: an accuracy of 0.945, a precision of 0.155, a recall of 0.129 and an F1-measure of 0.141. They did not provide us their AUC, then the F1-measure was chosen as a measure to compare, because it is an harmonic average between precision and recall. The accuracy in this case was misleading for

the presence of unbalanced class. It turned out that gradient tree boosting model had an improvement on the F1-measure of 26.95% respect to the client model.

Therefore, the goal of this thesis was achieved by the following fact. First, by building a classifier, which predicts if a company is in default or bad, that performs better than the client company. Secondly, by creating a new feature, the digital score, in addition to those provided by the client company, which measures the company's presence, performance and effectiveness on the web and integrates it into the initial classification problem of default.

6.1 Future Developments

The work done in this thesis was appreciated by both 3rdPLACE and the client company that commissioned the assignment. For the former a new business channel has opened up, and it has led to an expansion of its business through the involvement of some client companies that consider the method described in the present thesis to be remarkable and worthy of consideration, in order to apply it to their business.

There are several companies that, having realized the results obtained with this methodology, are becoming interested in Digital score and risk prediction of a subject, as they want to improve their traditional models which in many cases are not based on machine learning and more generally on Artificial Intelligence. One of them in particular, deals with the management of trade receivables, the operational and financial management of companies in industry, commerce and services. It is very similar to the client company activity with which the present work was carried out, but unlike the latter, it is active in the world of Invoicing trading, which very briefly deals with the purchase of invoices. The companies sell some of their invoices on specialized web platforms or to intermediaries, and investors buy them by paying immediately a down payment, generally equal to 80-90% of the nominal value of the credit; when the invoice is paid, the investor collects the countervalue and, at the same time, pays the balance to the transferor company. There are many advantages for businesses and investors in using invoice trading. Companies can then decide based on the needs of the moment which of their invoices to give and which not, without any predetermined commitment, and they can access therefore finance quickly, while investors are able to earn good returns on their money through a diversified portfolio. In this context the model developed so far would be very useful in this field, going to predict the risk of an invoice, i.e. the probability that it will be not paid, and therefore its value for an investor.

The client company that commissioned the work remained satisfied as well, with the resulting credit scoring model and the use of digital data. The model developed has a significantly higher performance than its traditional currently used classifier. Therefore, they are interested in updating it, and have already

asked 3rdPLACE to apply it to all the companies present in their database, about 6 million. In light of this there will certainly be a second step to keep improving the proposed work, both increasing the speed of the code, so as to implement it in a larger set of data; and generalizing it, so that it can be applied to other realities.

Among the future works there will certainly be the enrichment and improvement of the digital score with new features that will influence its value, or with different choice of the weights, that could be set with some algorithm.

A first step in this direction would be the monitoring of social networks, and in particular Facebook and Linkedin, which certainly represent important channels to get know by the public and the company's customers. Taking as a reference Facebook, a firm that has many followers, and among them the presence of other important and recognized companies, or that clearly shows all the information necessary to contact and reach it, or even if it publishes continuously and regularly marketing posts, could all be important signals that indicate if its business is solid and in good health. Moreover, this score represents only a snapshot of a company website, so it is a somewhat static value, which depends on the specific period in which the measurement took place. An added value for this measure would be to have also a temporal component. In particular, it is important to observe and measure how the site of a company, the reviews of its customers and its social media changes over time, and therefore, as all the variables that make up the digital score fluctuate. It is not necessary to recalculate the digital score every day, but it would be sufficient to measure it every quarter, half year or even annually.

Another factor that can be improved is the association of the company with its website or social network. Currently this association takes place through a proprietary 3rdPLACE algorithm, developed previously by other colleagues, but which can be re-trained using also this dataset. Basically what can be done is to increase the training data for the association algorithm, with a consequent improvement in performance and accuracy.

Appendices

Appendix A

Files

File Master

Italian Variable	English Variable	Description
RK		company identification number
CODICE FISCALE	fiscal code	code that unequivocally identify the member
PARTITA IVA	vat code	code that unequivocally identify the member that carries out relevant activity for VAT purpose
CCIAA		identify the Chamber of Commerce in which it is registered
NREA		identify the position in the Chamber of Commerce
DENOMINAZIONE	denomination	name of the company
CODICE NATURA GIURIDICA	legal nature code	identify the legal nature of the company
CODICE STATO ATTIVITÀ	code activity status	identify if it is active, inactive, ceased, suspended or in inscription
CODICE TIPO LOCALIZZAZIONE	type of location	if it refers on the local or legal office
PROGRESSIVO LOCALIZZAZIONE	progressive location	
TIPO UNITÀ LOCALE	type local unit	type of activity of the company
DATA ISCRIZIONE	inscription date	date of registration at the Chamber of Commerce
DATA INIZIO ATTIVITÀ	starting activity date	date of starting activity
DATA DI CESSAZIONE	termination date	date of termination activity
CODICE TOPONIMO	toponym code	code of the address
INDIRIZZO	address	address name
CIVICO	street number	street number
DESCRIZIONE COMUNE	city	
CODICE COMUNE	city code	
CAP	postal code	
CODICE PROVINCIA	province	
CODICE NAZIONE	state	state if it is foreign
CODICE ATECO 2007	Ateco code	economic activity classification
CODICE SAE	sae activity	economic activity sectors or subgroups
CODICE RAE	rae activity	economic activity branch
FLAGBAD		identify if it is bad

NUMERO TRASFERIMENTI	number transfer	number of transfer od legal office from one Chamberof Commerce to another one
SOCIO DIECI	ten member	indicates the presence of a member with 10% of the ownership of the shares
SOCIO CONTROLLANTE	controlling member	indicates the presence of a member with 50% of the ownership of the shares
UNITÀ LOCALI	local units	number of secondary headquarters
PEC		certified mail
FRAZIONE	hamlet	

Table A.1: File Master

File Balance Sheet

Italian Variable	English Variable	Description
RK		company identification number
ID BILANCIO		balance sheet identification number
RATING FINANZIARIO	financial rating	rating computed from data in the balance sheet
COMPARTO	sector	belonging sector
DATA CHIUSURA BILANCIO	closing date	
DATA DEPOSITO	deposit date	
PROCEDURE VOLONTARIE APERTE	procedures voluntarily opened	number of procedures that the company has voluntarily opened
INDICE ROS		
INDICE ROI		
INDICE ROE		
UTILE SU FATTURATO	profit	
COPERTURA INTERESSI	interest coverage	
AUTOCOPERTURA IMMOBILIZZAZIONI	self-covering of fixed assets	
LIQUIDITÀ	liquid assets	
DISPONIBILITA	availability	
TEMPI MEDI DI PAGAMENTO	average payment times	
INDEBITAMENTO	debt	
ASSET TURNOVER		
VALORE PRODUZIONE	production value	
COSTI PRODUZIONE	production costs	
RICAVI	revenues	
TOTALE ATTIVO	total assets	
UTILE PERDITA	profit loss	
PATRIMONIO NETTO	net assets	
AMMORTAMENTI SVALUTAZIONI	depreciation and amortization	
ONERI FINANZIARI	financial charges	
IMMOBILIZZAZIONI	assets	
CIRCOLANTE	working capital	
RATEI RISCONTI ATTIVI	accrued expenses and deferred income	
RIMANENZE	inventory	
DEBITI	debts	
DEBITI OLTRE	debts and other payables	
RATEI RISCONTI PASSIVI	accrued expenses and deferred income	
DEBITI FORNITORI		
MESI BILANCIO	months balcance sheet	duration in months of the financial year to which the financial statements refer

TOTALE PASSIVO	total liabilities
LONGITUDINE	longitude
LATITUDINE	latitude

Table A.2: File Balance Sheet

File Employee

Italian Variable	English Variable	Description
RK		company identification number
ANNO	year	year in which the number of employees refers
TRIMESTRE	quarter	quarter in which the number of employees refers
DIPENDENTI	employee	number of employees (workers, apprentices, employees, managers, etc.)
INDIPENDENTI	independent employees	number of independent employees (members, directors etc.) is identical
TOTALE	total	employee + independent employees

Table A.3: File Employee

File Ten-member

Italian Variable	English Variable	Description
RK		company identification number
CODICE_FISCALE	fiscal code	
DENOMINAZIONE	name	
COMUNE	city	
PROVINCIA	province	
VIA	street	
N_CIVICO	house number	
PERCENTUALE_QUOTA_PARZIALE	percentage partial share	it is the sum of the rights of the member within the company

Table A.4: File Employee

File Ten-exponent

Italian Variable	English Variable	Description
RK		company identification number
CODICE_FISCALE	fiscal code	
NAME		
CODICE	code	it is an acronym that identifies the type of charge of the exponent
CARICA	charge	it is the decoding of the charge code
DATA	date	it is the date of birth of the subject
LUOGO	place of birth	it is the place of birth of the subject expressed with the land registry code obtained from the fiscal code

SESSO	sex	indicates the sex of the subject

Table A.5: File Ten-exponent

File Local units

Italian Variable	English Variable	Description
RK		company identification number
CODICE FISCALE	fiscal code	code that unequivocally identify the member
CCIAA		identify the Chamber of Commerce in which it is registered
NREA		identify the position in the Chamber of Commerce
TIPO_LOCALIZZAZIONE_CD	charge	it is the decoding of the charge code
PROGRESSIVO LOCALIZZAZIONE	progressive location	
TIPO UNITÀ LOCALE	type local unit	type of activity of the company
DATA ISCRIZIONE	inscription date	date of registration at the Chamber of Commerce
TOPONIMO_CD	toponym code	code of the address
INDIRIZZO	address	address name
CIVICO	street number	street number
COMUNE_DESC	city	
COMUNE_CD	city code	
CAP	postal code	
PROVINCIA_CD	province	
NAZIONE_CD	state	state if it is foreign
ATECO07_	Ateco code	economic activity classification

Table A.6: File Local units

Bibliography

[1] L. Barghout, *Spatial-taxon information granules as used in iterative fuzzy-decision-making for image segmentation*, in Granular Computing and Decision-Making, Springer, 2015, pp. 285–318.

[2] J. Bennett, S. Lanning, et al., *The netflix prize*, in Proceedings of KDD cup and workshop, vol. 2007, New York, NY, USA, 2007, p. 35.

[3] C. M. Bishop, *Pattern recognition and machine learning*, Springer Science & Business Media, 2006.

[4] C. Bolton et al., *Logistic regression and its application in credit scoring*, PhD thesis, Citeseer.

[5] J. Brownlee, *A gentle introduction to the gradient boosting algorithm for machine learning*, Machine Learning Mastery. Nov, 9 (2016). `https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/`.

[6] T. Chen and C. Guestrin, *Xgboost: A scalable tree boosting system*, in Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM, 2016, pp. 785–794.

[7] N. D'Annunzio and G. Falavigna, *Modelli di analisi e previsione del rischio di insolvenza: una prospettiva delle metodologie applicate*, CSeris-Cnr, 2004.

[8] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*, " O'Reilly Media, Inc.", 2017.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. `http://www.deeplearningbook.org`.

[10] H. Ince and B. Aktan, *A comparison of data mining techniques for credit scoring in banking: A managerial perspective*, Journal of Business Economics and Management, 10 (2009), pp. 233–240.

[11] R. A. Johnson and D. Wichern, *Applied multivariate statistical analysis*, Pearson Education, Inc, 2007.

[12] N. Ketkar et al., *Deep Learning with Python*, Springer, 2017.

[13] T. Kurita, *Support vector machine and generalization*, Journal of Advanced Computational Intelligence and Intelligent Informatics, 8 (2004).

[14] S. Lin, J. Ansell, and G. Andreeva, *Merton models or credit scoring: modelling default of a small business*, University of Edinburgh Management School, 2007.

[15] L. J. Mester et al., *What's the point of credit scoring?*, Business review, 3 (1997), pp. 3–16.

[16] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*, MIT press, 2012.

[17] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*, Cambridge university press, 2014.

[18] E. Stanghellini, *Introduzione ai metodi statistici per il credit scoring*, Springer Science & Business Media, 2009.

[19] V. Vapnik, *The nature of statistical learning theory*, Springer science & business media, 2013.

[20] XRISTICA, *What is the difference between bagging and boosting?*, (2016). `https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/`.

[21] M. J. Zaki, W. Meira Jr, and W. Meira, *Data mining and analysis: fundamental concepts and algorithms*, Cambridge University Press, 2014.

Ringraziamenti

Volevo ringraziare chi mi ha dato forza in questi anni univeristari e mi è stato vicino durante la stesura della tesi.
In primis, ringrazio la mia famiglia: i miei genitori e mio fratello Eugenio. Mi hanno sempre sostenuto e creduto in me: è grazie al loro incoraggiamento se oggi sono riuscito a raggiungere questo traguardo. Sono stati di aiuto in tutti questi anni di studio.
Voglio ringraziare in modo speciale Rosellina, la mia ragazza. È stata al mio fianco in ogni momento, sostenendomi nei momenti più bui e difficili e festeggiando nei momenti di gioia. È riuscita a starmi accanto anche nella lontananza, e dopo ogni difficoltà ci siamo rialzati e rafforzati sempre di più. Sei stata un punto fisso per me. Grazie per ciò che abbiamo condiviso.
Ringrazio tutti i membri del team Datalysm di 3rdPLACE per avermi aiutato e consigliato attivamente in questi mesi. In particolare Matteo per aver creduto nel progetto e per essere sempre stato disponibile quando avevo bisogno di consigli e chiarimenti. Il suo lavoro di rilettura della tesi è stato molto prezioso.
Sono grato al prof. Ardagna che ha accettato di essere il mio relatore. I suoi consigli di come migliorare il lavoro svolto, le tecniche e i vari step che si devono usare nel costruire correttemente i modelli, la usa pazienza nel supervisionare il lavoro sono stati importanti.
A tutti i miei amici. Agli amici che ho lasciato in Sicilia, a coloro che si trovano in altre città per studio o per lavoro. Agli amici di università e agli amici di Erasmus per i quali si è condiviso gioie e dolori per lo studio: sono stati partecipi a delle esperienze che rimarranno per sempre impresse nel cuore, e che mi hanno fatto crescere tantissimo.
Tra tutti, un grazie sentito va a John che, nonostante ci vediamo pochi giorni l'anno, la nostra amicizia è rimasta invariata. È sempre bello confrontarsi con te e avere un punto di vista differente, condividere momenti, avventure e serate.

Tutti voi avete contribuito a farmi diventare quello che sono oggi, condividendo con me gioie, sacrifici e successi. Grazie!