# UNIVERSITA' DEGLI STUDI DI PADOVA


**DIPARTIMENTO DI SCIENZE ECONOMICHE ED AZIENDALI "M. FANNO"**

**CORSO DI LAUREA MAGISTRALE IN ECONOMICS AND FINANCE**


**TESI DI LAUREA**

## "Bankruptcy prediction models: A comparative analysis"


**RELATORE:**

**CH.MO PROF. MICHELE FABRIZI**

**LAUREANDO: NICOLÒ GREGGIO**

**MATRICOLA N. 2002794**


**ANNO ACCADEMICO 2021 – 2022**

Dichiaro di aver preso visione del "Regolamento antiplagio" approvato dal Consiglio del Dipartimento di Scienze Economiche e Aziendali e, consapevole delle conseguenze derivanti da dichiarazioni mendaci, dichiaro che il presente lavoro non è già stato sottoposto, in tutto o in parte, per il conseguimento di un titolo accademico in altre Università italiane o straniere. Dichiaro inoltre che tutte le fonti utilizzate per la realizzazione del presente lavoro, inclusi i materiali digitali, sono state correttamente citate nel corpo del testo e nella sezione 'Riferimenti bibliografici'.

*I hereby declare that I have read and understood the "Anti-plagiarism rules and regulations" approved by the Council of the Department of Economics and Management and I am aware of the consequences of making false statements. I declare that this piece of work has not been previously submitted – either fully or partially – for fulfilling the requirements of an academic degree, whether in Italy or abroad. Furthermore, I declare that the references used for this work – including the digital materials – have been appropriately cited and acknowledged in the text and in the section 'References'.*

Firma (signature) 

**INDEX**

# INTRODUCTION

In modern times, globalization has led to a more dynamic and competitive framework worldwide, inducing higher volatility in the economic system. That causes a decrease in the stability of the environment where firms operate, bringing to a habitat where companies increasingly interchange with each other, ceasing and arising most frequently. Furthermore, the interconnection between countries has reached a level in which distant events in certain sectors have a heavy impact on other business areas of different faraway nations. As proof of those statements, three crises have occurred in the last two decades. In such circumstances, the bankruptcy prediction field is a valid quantitative instrument to foresee distress situations on time. It assists in decreasing the operational risk for credit intermediaries and making responses to bankruptcy conditions timelier, re-establishing a more robust financial system and generally a more efficient and effective allocation of resources. From another viewpoint, it may be a tool for managers, but also stakeholders and investors that must select whether to give credit or not to a company. In practice, the area of bankruptcy prediction has two correlated objectives of the research. The first one is to identify those variables and trends that exhibit the corporate default, analysing with different methods which ratio is the best one to detect failures. In this sense, a general result is that there are a few different adequate ratios, but their performances depend on the sectors and period of time taken into consideration. The second one is to build models and find the best one to classify the observations, which is the main topic of this dissertation. Practically, this branch of research focuses on creating new models which have higher performances compared to the older ones or applying new preprocessing techniques to have less raw data and better results from the algorithms.

What has to be clear is that these models can only support the classification task and the creditworthiness system of firms and financial intermediaries. The results showed that the sophisticated machine learning (ML) models are those that, with some bias, classify adequately the instances. The recall score of bankruptcy firms is extraordinarily high given the small number of variables. The best class of models reaches near 83% of correct classification for failed firms at one year and then 75% and 73% for respectively two- and three-years distances. However, the lack of precision is the main issue in the formulation proposed in this thesis. This is a critical point for the performance of a model and it means that a non-neglectable number of firms that will be solvent in the future are defined as bankrupt by the algorithms. This can turn into a self-fulfilling prophecy. For example, the rating firms that use these models can wrongly classify a firm as insolvent and affect the path of that company. Even if it is solid, it could fail because of the wrong assessment of the rating firms and the relative withdrawal from

the credit system. These models are quite powerful and useful in the rating assessment, but the creativity and the human knowledge of an expert continue to be the first requirement to properly classified and evaluate the financial soundness of the firms. The credit risk analysis cannot be composed automatically and only of the results that came out from these models, but they have to be set in a more articulated process checked by a specialist.

This thesis has the goal to evaluate the different models in a small variables setting and in as realistic as possible situation to see if the ML algorithms are worth to be implemented or if the statistical models like the logit one have already adequate performances in the bankruptcy prediction field. In fact, ML algorithms are typically used in problems where the explanatory variables are hundreds and the issues are much more complex, a case scenario far from that of the bankruptcy prediction where in the literature only a few ratios are able to identify adequately the bankruptcy of firms, and the outcome is a simple Boolean one. The main question that this dissertation tries to answer is if these ML models are useful or not in a situation of lack of variables and of high imbalance of the classes in the dataset. As it will be shown the results of the recent ML algorithm clearly outmatch all the other models and the SMOTE and Tomek link procedure partially solve the real problem of the low frequency of the bankruptcy class.

To be more specific, in the empirical part of this thesis, the author has retrieved the dataset from the Orbis database of eight European Union countries. Instead, the procedures and models have been implemented by coding in the Python language through the Jupyter Notebook application. The population is composed of 3870 bankrupt firms and 15276 nonbankrupt ones. The observations go from 2013 to 2020 and the failures of the bankruptcy companies are in 2019, 2020 and 2021. This means that more or less a total of 150000 observations have been taken. The considerable number of elements allows to compare the model and retrieve results in the most general situation possible compared to using a smaller dataset as done in most of the cases in the literature. To be clear, the definition of the bankrupt sample was specified in a rigorous way, applying some constraints to the database to retrieve firms that failed only in the specified period. To construct the nonbankrupt sample, initially, the failed firms were distinguished by country of origin and economic activity. Subsequently, the nonbankrupt sample was built by maintaining for each country and activity a certain proportion of observations compared to the failed ones. In other words, the author avoids including in the nonbankrupt sample country or business activity that were not in the bankrupt sample, in this way the results are solid and do not depend on the ability of the models to distinguish a certain country from another one or a business area to another one. The choice of using only three explanatory variables and not including a fourth or fifth one is due to the fact that the Orbis database is composed of a quite

substantial number of incomplete observations. It was not possible to collect more specific ratios and the author decided to take the first three variables from the most used ones in the literature. However, this is something already seen in literature and done for example by Brédart et al. (2022) that take only three variables for their analysis reaching an accuracy score of 84% using a Neural Network, a Support Vector Machine and an XGBoost model. Moreover, this procedure is coherent with the purpose of this thesis: "compare the performance abilities of different ML and statistic models on the most feasible real basis optic". In this way, an upper level of difficulty for the models is settled, given the shortage of information. For the same purpose, the author decided not to use any matching procedure like the Propensity Score Matching (PSM). This is a methodology that often has the opposite intended effect, as explained in the literature. It is not consistent with the goal of the dissertation and the results would end up being artificially increased by a procedure that can only be done ex-post and practically picks data in a way to better distinguish the two categories. All the techniques adopted in this thesis can be done ex-ante without being aware of the status of the observations. Due to that logical limit, to not affect the results, only that last type of methodology was be performed as data preprocessing. These techniques were executed to solve the following three issues:

- **Multicollinearity**. It is a statistical phenomenon in which some ratios may be linearly predicted from others. Beyond a certain correlation limit, the multicollinearity problem could occur. Even if the three ratios used in the models are taken in groups of three consecutive years, for example to predict at one year for the firms that should fail in 2021 the explanatory variables are those of 2020, 2019 and 2018, the correlation matrix shows that the multicollinearity issue is only partially present for some years and ratios. However, this issue does not decrease the classification ability or reliability of the models taken as a whole. The only bias is regarding the individual predictors' scores that may vary haphazardly in reaction to insignificant changes in the data. Moreover, given the aim of the research, the multicollinearity issue is not a severe problem for the implementation and comparison of the models. Anyway, the Principal Component Analysis (PCA) procedure was conducted to analyse and verify if this approach can effectively increase the stability and performance of the models. The results are clear and in line with the literature, the multicollinearity issue is not significant in the bankruptcy prediction field and given the slightly decrease performance results with the PCA procedure and the fact that all the models implement some types of regularization, the author decided to perform the models without that methodology but tackled the

problem using the parameters which will be explained in detail in the following chapters.

- **Minority class problem**. The evidence that defaults are rare events implies that the number of bankrupt firms is normally much lower than the number of nonbankrupt ones (Japkowicz and Stephen, 2002). That scarcity affects the model's ability to discriminate the minority group ending up having very low accuracy scores for that class. Instead of picking a smaller number of nonbankruptcy firms to solve the issue, as done in the literature, which would lead to meaningless and unreal results, the author has adopted the SMOTE and Tomek link methodology. This is a well-known approach in data science used to artificially increase the number of bankrupt firms only in the training phase. This enables the algorithms to train themselves on more homogeneous data, increasing exponentially their classification ability of bankruptcy firms and the score in the testing phase. The price of the huge increase in the recall score and in the decrease of the Type I error, is the increase of the Type II error and the decrease of the precision of the model. This means that the algorithms will classify extraordinary well the firms that effectively will go bankrupt but at the same time, they will wrongly classify as bankrupt a significant number of firms that instead will continue to operate leading to the just seen self-fulfilling prophecy problem. This dissertation will better explain the innovation, the potentiality and the theoretical background of this new procedure in the bankruptcy prediction field, however, given the cost, some precautions must be taken in this regard.
- **Model optimization**. To compare the models at their best, the hyperparameter tuning methodology has been performed. This procedure finds the best settings, for a given metric, of each model. It is a quite computational and time-demanding method, but it is necessary to fine-tune the models. Essentially, the parameters that are not estimated by the training phase but are set a priori in a standard way are changed through the hyperparameter tuning procedure to increase the model performances on a prespecified score. The author has arbitrarily decided the metric because there is no mention, in the bankruptcy prediction field, of this methodology and given the specificity of the problem, the data science bibliography is not useful. After having tested different scores as the precision, the F1 and F2, the chosen one is the recall value of bankrupt cases (class 0). This means that the optimal hyperparameters will be those that decrease the Type I error. The search procedure has been done using two approaches: a randomized one and a grid one. After setting the parameter space where the two techniques must find the best configurations, the first one randomly searches from the distribution of the

defined parameter values. The second one, rather, is a more specific search that computes each of the possible combinations given the parameter space. Practically, for each model, firstly, it has been set a random search to decrease the space where the best parameters should be found, and subsequently, through the grid search, the optimal parameters were found. Even with this approach to decrease the time required, in some cases, the search takes hours and in the case of the support vector machine (SVM) it takes an entire day. However, this procedure led to better recall of failed firms' score performances in almost all the models.

The dissertation is organized as follows. Chapter 1 is a literature review about the bankruptcy models more used, starting with the statistical one and going on with the first and then complex machine learning algorithms. In light of what follows, this first part explains theoretically and mathematically the main characteristics of all the models that will be executed. In Chapter 2, the research method description, the dataset origin and composition together with the choice of the predictors and the relative descriptive statistics on the sample are explained. In this part, the multicollinearity issue, the minority problem and the hyperparameter tuning process are elucidated. To conclude, Chapter 4 reports the results of the empirical analysis and comparisons of the different algorithms executed one, two and three years prior the failure, highlighting the remarkable results and the critiques of the procedure and model implemented. More precisely in this dissertation eight models are performed and compared through Python, namely: multiple discriminant analysis (MDA), Logistic Regression, k-Nearest-Neighbour (kNN), Support Vector Machines (SVM), Neural Network (NN), Random Forest (RF), XGBoost and CatBoost.

## CHAPETER 1: REVIEW OF THE LITERATURE

The increasing importance of credit assessment for institutional subjects and investors, in general, has a clear effect on the increased number of publications on bankruptcy prediction, as reported by Farias, Martínez and Martín-Cervantes (2021). The authors, based on a sample of 588 articles in the period between 1954 and 2020, showed that the number of publications has grown exponentially, particularly in the last 20 years.

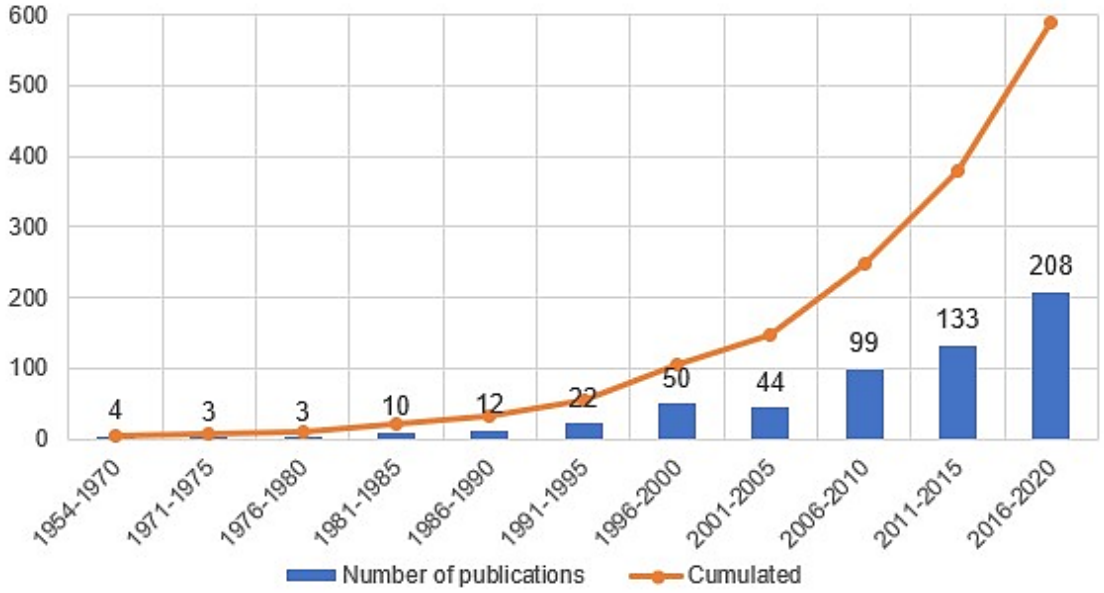


Figure 1: Evolution over time of published articles. Source: Farias et al. (2021).

From an academic point of view, this jump is manifested by the multiple theories and approaches that research has applied to explain this phenomenon. Considering the social point of view, the growth is pointed out by the interest exhibited by institutions to set up regulations that seek to develop public policies that allow better business performances. Examples of that are the three Basel regulations that have been made starting from 1988 to maintain the financial system stability at the economy-wide level. The main elements that have increased the necessity for a more precise model to predict bankruptcies are the facing of several crises in the last decades, such as the Dot-com bubble, the Sub-prime crisis and the Covid crisis. Indeed, financial failure affects company survival and imposes inflated costs on bankers and other creditors, who may only partially recover their investments (Wruck, 1990). For these reasons, the bankruptcy prediction field has become year after year the subject of studies that draw not only from accounting and corporate finance but also from data science and informatics with the use of machine learning techniques. The next paragraphs proceed as follows: in the first one, a review of the most important statistic models since the 30s, and in the second one, the analysis

from the easier to the more advanced data mining models derived by the use of data analysis methodologies.

## 1.1 THE FIRST STEPS INTO A NEW ECONOMIC ISSUE – THE STATISTICAL APPROACH

The story of bankruptcy prediction starts in 1930 with a bulletin published by the Bureau of Business Research (BBR) and goes on with the studies of a few authors until the formulation of the first idea of a model with Beaver in 1962. These initial researches focus their analysis on indices, retrieved from the data available in public financial statements, on small samples of failed and non-failed companies. The authors try to compare those ratios, ranking the prediction ability of these variables to anticipate possible future failures.

Initially, the BBR analysed 24 ratios of 29 firms to detect common distinctive characteristics for bankruptcy. The ratios of each non-failed firm were compared with the average ratios of the failed ones, exhibiting eight good indicators to determine the "growing of weakness". In particular, the authors suggest taking into account the working capital to total assets ratio. PitzPatrick, in 1932, compared 13 ratios of 19 bankrupt and nonbankrupt firms. He discovered that healthy firms show a very favourable ratio, instead, bankrupt firms suffer from unfavourable ones compared to a "standard ratio". Important factors to detect failure were the net worth to debt and the profits to net worth ratio. A more composite analysis, started from the study of BBR, was made by Smith and Winakor (1935). They collect the ratios of 183 failed firms from various gamma of industry and attest the capacity of working capital to total asset ratio to be a good predictor. A noticeable work was also that of Merwin, in 1942. He analysed small manufacturing firms and reported that when he compared bankrupt and nonbankrupt firms, the failing one showed some warning signal four or five years before the failure. He also discovered that the net working capital to total assets ratio, the current ratio and the net worth to total debt ratio are significant indicators of business failure. Significant is also the analysis of Chudson, in 1945, that try to find a "normal pattern" in the financial structure of the firms. He ends up with the result that there was no "normal pattern" in financial structure in general at the economy-wide level. However, "inside a specific industry, size and profitability groups there is a clustering of ratios". Even if Chudson does not develop a bankruptcy prediction model, he indicates that a model developed for general application across industries may not be as appropriate as industry-specific models. The last author of the early stage of the bankruptcy prediction field is Jackendoff (1962). Ratios of profitable and unprofitable firms are the subject of his analysis. He described how the current ratio and net working capital to total assets are

higher for profitable firms than the unprofitable ones. He also found that the debt-to-worth ratio is lower for profitable firms than for unprofitable ones (Bellovary, Giacomino and Akers, 2007).

Like these first studies also **Beaver (1966)** analysed the mean values of 30 ratios of 79 failed and 79 non-failed firms in different industries according to asset size and industry obtained from Moody's Industrial Manual between 1954 and 1964 period. The ratios were selected based on the frequency in the literature and on a "cash-flow" preference. As shown in table 1, the 30 ratios were classified into 6 groups.

| | |
|---|---|
| GROUP I (CASH-FLOW RATIOS)<br>1. Cash flow to sales<br>2. Cash flow to total assets<br>3. Cash flow to net worth<br>4. Cash flow to total debt<br>GROUP II (NET-INCOME RATIOS)<br>1. Net income to sales<br>2. Net income to total assets<br>3. Net income to net worth<br>4. Net income to total debt<br>GROUP III (DEBT TO TOTAL-ASSET RATIOS)<br>1. Current liabilities to total assets<br>2. Long-term liabilities to total assets<br>3. Current plus long-term liabilities to total assets<br>4. Current plus long-term plus preferred stock to total assets<br>GROUP IV (LIQUID-ASSET TO TOTAL-ASSET RATIOS)<br>1. Cash to total assets<br>2. Quick assets to total assets<br>3. Current assets to total assets<br>4. Working capital to total assets | GROUP V (LIQUID-ASSET TO CURRENT DEBT RATIOS)<br>1. Cash to current liabilities<br>2. Quick assets to current liabilities<br>3. Current ratio (current assets to current liabilities)<br>GROUP VI (TURNOVER RATIOS)<br>1. Cash to sales<br>2. Accounts receivable to sales<br>3. Inventory to sales<br>4. Quick assets to sales<br>5. Current assets to sales<br>6. Working capital to sales<br>7. Net worth to sales<br>8. Total assets to sales<br>9. Cash interval (cash to fund expenditures for operations)<br>10. Defensive interval (defensive assets to fund expenditures for operations)<br>11. No-credit interval (defensive assets minus current liabilities to fund expenditures for operations) |

Table 1: Lists of ratios assessed. Source: Beaver (1962).

However, the main difference between him and the authors discussed above is that he tested the abilities to define bankruptcy and nonbankruptcy for each ratio through a classification test that used a dichotomous prediction, the so-called univariate discriminant analysis. After having set the optimal cut-off, which minimizes the incorrect prediction, the firm will be classified as failed if the firm's ratio is below the cut-off, or nonfailed if the ratio is above (in the case of total debt to total-assets ratio that is reversed). Beaver found that the ratio with the highest predictive ability is net income to total debt with 92% of accuracy one year prior to failure, followed by net income to sales with 91% and net income to net worth, cash flow to total debt, and cash flow to total assets. Beavers, in his conclusion, reports an intrinsic problem in this topic, the selection bias, that is the fact that to detect the financial "illness" of firms researchers

used ratios. So, they probably have been many firms whose illnesses were detected before failure occurred through the ratios. If the management has been capable to restore a wellness environment the sample of failed firms will incorporate firms whose illnesses were not detectable making the sample biased. Beaver also suggests switching from a univariate analysis to a multiratio analysis, so using at the same time more than one ratio, in the optic that they may have higher predictive ability than single ratios.

**Altman (1968)** took this suggestion and made a multiple discriminant analysis (MDA), developing a five-factor model to predict the bankruptcy of manufacturing firms. He studied 66 companies, half of which filled the Chapter X of the National Bankruptcy Act in the period between 1946 and 1965. He computed a discriminant coefficient for five independent variables, after that he combined the coefficient with the ratio into a specific score, the so-called "Z-score", which is the optimal cut-off value that divided the bankruptcies and non-bankruptcies companies. In particular, if the firm has a Z-value greater than 2.99 it is classified as "non-bankrupt", instead if it falls in the area under 1.81 it is a bankrupt one. In the region between 1.81 and 2.99, there is the "zone of ignorance" or "grey area" and the model is vulnerable to misclassification. The discriminant function was:

$$Z= 0.012\ X_1+0.014\ X_2+0.033\ X_3+0.006\ X_4+0.999\ X_5 \qquad (1)$$

Where the five ratios were chosen on the basis of their popularity in the literature and on their "potential relevance to the study" and they are:

- $X_1$= Working Capital / Total Assets

It compares the net liquid assets to the total assets of the firm. The working capital is the difference between current assets and current liabilities, thus this ratio determines the short-term company's solvency.

- $X_2$= Retained Earnings / Total Assets

It measures the accumulated profitability over a company's total assets. Younger companies may not have good retained earnings to total assets as they do not yet reach profit potential, instead, they can be in a loss position. The ratio implicitly takes into account the age of the firm. Ceteris paribus captures the fact that young firms have more probability to go into default.

- $X_3$= Earnings Before Interest and Taxes / Total Assets

It ascertains how productive the firm is at using assets to generate profits, regardless of tax and leverage factors.

- $X_4$= Market Value of Equity / Book Value of Total Debt

It compares the market value of equity to the book value of total liabilities. This measures how much the firm's assets can decline in value before the liabilities exceed the assets and the firm becomes insolvent. It includes a market dimension in the analysis, so it can be done only for quoted firms. Due to that, Altman (2000) revisited the model parameters and recalculated the coefficients. He replaced the current $X_4$ with the book value on the total liabilities ratio.

- $X_5$= Sales / Total Assets

It measures the efficiency of a company in managing its assets to generate revenue. Even if the F-test[1] conducted by Altman on these five independent variables indicates that this last factor does not show a significant value, it had been retained for its unique relationship to other variables and because it has the second higher contribution value in the rank to explain the overall discriminating ability of the model in a univariate discriminant analysis.

Altman was the first to use the confusion matrix, also called "accuracy matrix", as a way to display and to synthetize the correct and incorrect classification power of a model.

| Actual | Predicted Group 1 | Predicted Group 2 | Per cent Correct | Per cent Error | n |
|---|---|---|---|---|---|
| Group 1 | 31 | 2 | 94 | 6 | 33 |
| Group 2 | 1 | 32 | 97 | 3 | 33 |
| | | | 95 | 5 | 66 |

Table 2: Initial sample empirical results at 1 year. Source: Altman (1968).

In table 2 Altman shows the classification result for the initial sample of 66 firms. The two errors are those highlighted. In the first raw there is the Type I error, which happens when a firm is wrongly classified as solvent even if it will go bankrupt. Instead, in the second raw there is the Type II error, which is when a firm is incorrectly classified as insolvent even if it will not go into bankruptcy. In other terms, the first one is the amount of money lost from lending to a firm that goes bankrupt while the second one is the opportunity cost of not lending to a firm that does not go bankrupt. Of course, these two errors have different weights in a bankruptcy prediction model: the first one has a more severe effect given that for lenders is much more

[1] An F-test is any statistical test in which the test statistic has an F-distribution under the null hypothesis. In this case, the test relates the difference between the mean values of the ratios in each group to the variability of values of the ratio within each group

damaging to give money to a future bankrupt firm rather than do not profit from a healthy firm. Ceteris paribus, a model that minimizes the first error is the better one.

| Year Prior to Bankruptcy | Hits | Misses | Per cent Correct |
|---|---|---|---|
| 1st n = 33 | 31 | 2 | 95 |
| 2nd n = 32 | 23 | 9 | 72 |
| 3rd n = 29 | 14 | 15 | 48 |
| 4th n = 28 | 8 | 20 | 29 |
| 5th n = 25 | 9 | 16 | 36 |

Table 3: Initial sample five-year predictive accuracy of the MDA model. Source: Altman (1968).

Table 3 shows that in the first year the accuracy of the model is very high with a correct prediction of 95% on the sample. However, it also shows that the results decrease over time as in Beaver's univariate study. After the second year, the MDA model incorrectly classifies the majority of the sample firms. Recalling that 50% accuracy is the "toss of a coin" probability of doing a correct classification, the MDA model becomes useless with 42% of accuracy at three years.

As Altman said: "While a subset of variables is effective in the initial sample, there is no guarantee that it will be effective for the population in general". Hence, it has to be performed an analysis of the accuracy of the model setup using other samples to verify if it works well with other data. Altman did this by taking a smaller (sixteen firms) subset of the original sample. This is an error as explained in the article of Grice and Dugan in 2001. They suggest that the accuracy test should be made in a more rigorous way using a hold-out sample test. A hold-out sample is a random sample from a data set that is retained and not used in the model fitting process. It operates by splitting the data into a training set, used to train the model, and a test set, used to estimate the loss of the trained model in "unseen" data. This gives an unbiased assessment of how well the model might perform if applied to new data (Devroye and Wagner, 1979). However, the potential upward bias of the hold-out test is mitigated only if there are the following characteristics:

1) The initial sample and the hold-out sample should belong to substantially different periods of time.
2) The hold-out sample should come from a different set of industries and not from a subset of the original sample.
3) The hold-out should be large and proportional to actual bankruptcy rates.

Altman did also two other validation tests using a small sample of firms taken outside the initial sample. The first one utilises a sample of 25 bankrupt firms with an asset-size range similar to the initial bankrupt group. The result at one year was quite good, with a 96% accuracy using the coefficient parameters set in equation 1. As explained before this type of test is biased and does not offer an accurate test. The second one, instead, was computed using a new sample of 66 firms, half of which were from the year 1958 and a half from 1961. They are taken without making any consideration of their asset size, the only criteria are that over 65% of these firms had suffered two-three years of negative profits. The result is less good with "only" 79% of accuracy at one year. This test compared to the first one quite satisfies all the criteria of Grice and Dugan and can be a sufficient way to verify the classification accuracy of the model. It is clear how taking the initial specification and using that with different samples reduce the performance of the model.

Another milestone in the bankruptcy prediction models literature is **Ohlson (1980)**. He decided to reject the MDA technique to use another type of approach. Ohlson pointed out that previous studies have the following problems:

- They imposed some statistical conditions as the distributional properties of the predictors. For example, the MDA requires that for both firms' categories the variance-covariance matrices of the predictors should be equal. Another requirement is the normality distribution of the explanatory predictors, the ratios.
- The Z-score has little intuitive interpretation. It is simply an ordinal ranking, so to make some classification tasks an adequate prior probabilistic partition of the payoff is previously needed. In other words, a cut-off value is required, in Altman (1968) was 1.81 and 2.99, but it cannot be known a priori if it is the best range to minimize the misclassification.
- The "matching" procedures used in MDA are based on criteria such as size and industry but the effect of this methodology on the final result is not intuitive nor certain. To figure it out, instead of making these procedures before his analysis, Ohlson included these variables as predictors.

Worthy to be mentioned and to be taken into consideration for the subsequent empirical part of this document are the critiques that Ohlson made in the introduction of his paper in 1980:

- He stressed that previous studies collect data for failed firms from Moody's Manual. The problem here was that the reports do not indicate whether the company goes into bankruptcy prior to or after the date of release. It is a timing issue. Most of the studies before Ohlson assumed that a report is available at the end of the fiscal year. If the firm

fills bankruptcy before the release of the financial statement but after the end of the fiscal year date, there is the possibility that the statement already included information about the default. An example is the Italian regulation that obliges the shareholders to approve the financial statements annually within 120 or 180 days from the end of the fiscal year. Within 30 days from the date of approval, the administrator shall deposit it in the relevant register of company. If the model uses biased financial statements, the results will be contaminated. An adequate procedure to overcome that is the one of Ohlson, he filters every statement that has this problem to clean the dataset from possible bias.

- Another useful highlight must be made in the binary partition of the possible outcomes. The simple dichotomous between the failed and the non-failed company is a heuristic approximation. Blazy and Combier (1997) offered a synthesis of the causes of financial distress: accidental, financial, managerial, market problems but also macroeconomic, strategic and production structures. To make a reliable analysis in which there is a comparison of bankruptcy prediction models between each other, researchers must use a pure legal definition of bankruptcy taking into account possible relevant changes in the regulation that happen over the years. This crude approximation is the only way to establish which of the models is the best predictor.
- The last issue is the period of the analysis. To make an effective comparison between models, the researcher should assess the precision of their model compared to the others using the same historical periods. If it is not done so, it is assumed the stationarity of the model over time. This hypothesis is something that can be excluded watching to the fact that corporate failure rises sharply during economic recessions (Lev, 1974). In 1984, Mensah show that: i) inflation; ii) interest rates and credit availability; iii) business cycle (recession/expansion phase) lead to an overall instability of the coefficients of the model. Mensah suggests estimating the model coefficient as close as possible to the prediction period.

Ohlson decided to use the logistic regression because of its power when the dependent variable is categorical, so when it can take a limited number of values (as bankrupt/nonbankrupt), but also because he wants to take into account potential non-linear effects between the independent and dependent variable and for the fact that the logit model does not require statistical assumptions. When there is a dichotomous characteristic, it has to be created a boundary between the values that are mapped as 0 or 1. In linear regression, the output is a continuous variable, so there is not any limit because the values may go from $-\infty$ to $+\infty$ as shown in figure

2. Instead, the logistic regression model outputs probabilities with a sigmoid function that gives an "S-shaped" curve, which can classify observations only as 0 or 1. Because of that, the logistic model should be preferred rather than linear regression, so the author of this dissertation expects to have higher performance using the logit model rather than the MDA one in the empirical analysis of this thesis.

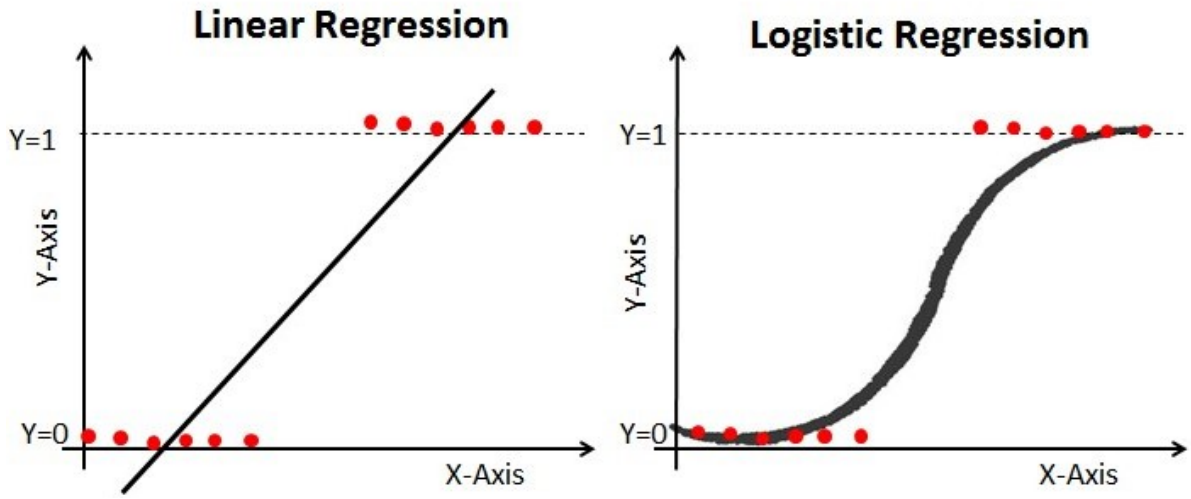


Figure 2: Difference between linear and logistic regression.

Through the cumulative logistic function, the logit model is performed. Employing the cumulative logistic function, it can be obtained the probability of a company going bankrupt (PL):

$$P_{L_a} = \frac{1}{1 + e^{-(Z_{L_a})}} = \frac{e^{Z_{L_a}}}{1 + e^{Z_{L_a}}} \tag{2}$$

Where:

$$Z_{L_a} = \beta_1 x_{1_a} + \beta_2 x_{2_a} + \cdots + \beta_n X_{n_a} \tag{2.1}$$

Hosmer and Lemeshow (2013) defined the logit function as:

$$logit\left(P_{L_a}\right) = ln\left(\frac{P_{L_a}}{1 - P_{L_a}}\right) = f(x, \beta) = \beta^T x \tag{2.2}$$

where $\beta = (\beta_0, \beta_1, \beta_2, \ldots \beta_k)$ is the vector of the coefficients and $\beta^T x$ is the transposed vector. The relationship $\frac{P_{L_a}}{1-P_{L_a}}$ is called odds ratio, which is the probability that the default event happens on the probability of non-default. The $ln$ of this relationship denotes logit transformation, this is a way to normalize the distribution function of the y-axis. The unknown coefficients $\beta_i$ are estimated from the data using the maximum log-likelihood method:

$$l(\beta) = \sum_{i=1}^{N} \{y_i \, ln(P_{L_a}) + (1 - y_i) \, ln(1 - P_{L_a})\} \tag{2.3}$$

In order to find the optimal coefficients $\beta_i$and thus to find the best fit sigmoid curve (figure 2) in the range of the dichotomous output 0/1, the maximum likelihood estimator maximizes the likelihood of getting the desired output values given a cost function that is called Maximum Likelihood Estimation (MLE) function. In the bankruptcy prediction case, the MLE in formula (2.3), maximizes the likelihood of being classified as bankrupt ($y_i$=1) and nonbankrupt ($y_i$=0) through $P_{L_a}$and so the cut-off value $Z_{L_a}$. On the framework of the bankruptcy prediction models, the cut-off score settled defines if a company is classified as failing or non-failing. Higher cut-offs classify more firms as bankrupt, instead lower cut-offs grade more firms as nonbankrupt. As already highlighted, the emphasis should be on the minimization of Type I error, failing company classified as non-failing, and Type II error, non-failing company classified as failing.

Ohlson collected data in the period from 1970 to 1976 for industrial firms quoted in some exchange or OTC market. The sample was composed of 2058 non-bankrupt firms and 105 bankrupt firms. He estimated a logit model using an intercept and nine variables, taken on the basis of their quote frequency in the literature:

1) SIZE = log (total assets/GNP price-level index) with base value in 1968.
2) TLTA = total liabilities divided by total assets.
3) WCTA = working capital divided by total assets.
4) CLCA = current liabilities divided by current assets.
5) OENEG = a dummy variable that is one if total liabilities exceed total assets, zero otherwise.
6) NITA = net income divided by total assets.
7) FUTL = funds provided by operations divided by total liabilities.
8) INTWO = a dummy variable that is one if net income is negative for at least two years, zero otherwise.
9) CHIN = $(NI_t - NI_{t-1})/(|NI_t| + |NI_{t-1}|)$ and $NI_t$ is the net income for the most recent year.

| | Likelihood Ratio Index | Percent Correctly Predicted |
|---|---|---|
| Model 1 | 0.8388 | 96.12 |
| Model 2 | 0.7970 | 95.55 |
| Model 3 | 0.719 | 92.84 |

Table 4: Prediction results using logit model. Source: Ohlson (1980).

Table 4 reports the result prediction of bankruptcy before the end of the first year (Model 1), second year (Model 2) and third year (Model 3). The score confirms the already seen trend of decrease in accuracy increasing the time horizon of the prediction. Ohlson notices that great relevance should be attributed to the size parameter whose value greatly affects the final predictive outcome. Logically, an immediate explanation for this relates to the fact that greater-sized companies have more to deplete before actually filing for bankruptcy. However, the emphasis on the size variable given by the model could also be due to a third, not better known, element linked to size. In principle, as the author points out reporting the results, a possible explanation could be connected to the belonging to stock exchanges or OTC markets. He thus suggests further research to include variables like equity prices and their trends. The model made by Ohlson seems to beat the Z-score of Altman

The last pillar of the statistic bankruptcy prediction field is the probit model. **Zmijewski (1984)** was the first one to formally defined it in this discipline. He defined the population of the analysis as all industry firms listed on the American and New York Stock Exchange from 1972 to 1978 with a Standard Industrial Classification (SIC) code of less than 6000. The data were collected from the SEC 10K report of Cornell University using 40 bankrupt and 800 nonbankrupt firms. Zmijewski, as Ohlson did, used a legal definition of bankruptcy. He recognizes a bankrupt firm if it records a bankruptcy petition in this period of time, and nonbankruptcy otherwise.

Probit regression is often compared with the Logit ones because of some similarities in the way of how the model analyses the relationship between dependent and independent variables. However, the process through which the models are created is different. Assuming the binary dependent variable takes value 0 if the firm is bankrupt and 1 otherwise. The probit model is given as:

$$P = 1 - \Phi(-x, \beta) = \Phi(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \qquad (3)$$

Where Φ is the cumulative distribution function of the standard normal distribution:

$$\Phi(x, \beta) = \int_{-\infty}^{x,\beta} \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} dx \qquad (3.1)$$

Zmijewski selected only three explanatory variables on the basis of the previous literature. Obtaining the so-called X-score:

$$Zmijewski \; = \; H \; = \; -4.3 - 4.5X_1 + 5.7X_2 + 0.004X_3 \qquad (3.2)$$

Where:

1) $X_1$= net income to total assets (ROA)
2) $X_2$= total liabilities to total assets (financial leverage)
3) $X_3$= current assets to current liabilities (liquidity)

The choice of the independent variable was criticized by Shumway (2001). He states that the model of Zmijewski is de facto only a single variable one, because the variables $X_1$ is strongly correlated with $X_2$ ($p = 0.40$)[2]. Shumway argues that the model of Zmijewski (1984) does not have strong predictive power for bankruptcy having high rates of Type I error. One probable cause can be attributed, according to Grice and Dugan (2003), to the fact that Zmijewski did not perform an analysis to find the best ratios to use. However, he obtained quite incredible results in terms of accuracy, near 99% for the training sample. But he did not show the results on the hold-out sample. One of the main advantages of the probit function, like the logit one, is that the function maps the value between 0 and 1, instead of the Altman model where the probabilities can wrongly lie outside the dichotomous space. Zmijewski (1984) classified firms as bankrupt because they do not have complete data or because the model predicts a probability of default that is greater than 0.5. Instead, firms with complete data and with the model that predicts a probability of default less than 0.5 are classified as nonbankrupt.

Zmijewski examined the effect on the bankruptcy prediction model due to nonrandom samples. An exogenous random sampling is one in which observations are randomly drawn and then their characteristic, dependent and independent variables, are observed. Instead, in the field of bankruptcy prediction, initially, the researchers observe the variables and then draw the sample. This methodology creates a "choice-based" sample that is in contrast with the random sampling assumptions and leads to having asymptotically biased parameter and probability estimates. Specifically, Zmijewski analyses two types of bias caused by estimating the model on nonrandom samples.

The first bias is due to the fact that researchers "oversampling" bankrupt firms. Due to the extremely low-frequency rate of firms that goes to bankruptcy compared to the overall population of companies. Graph 3 tells us that bankruptcy is relatively very small compared to the number of firms that continue to be going concern. In the US the number of firms was 7,4 million according to the bureau's "County Business Patterns: 2010" report. Recalling also that bankruptcy is a subset of the more general financial distress situation, in reality the frequency

[2] It's the Pearson Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:1 indicates a strong positive relationship; -1 indicates a strong negative relationship and A result of zero indicates no relationship at all.

rate never exceeds the first percentage point. Zmijewski showed through the Pearson correlation coefficients between frequency rate and group error rates are clear proof of the existence of a choice-based sample bias.

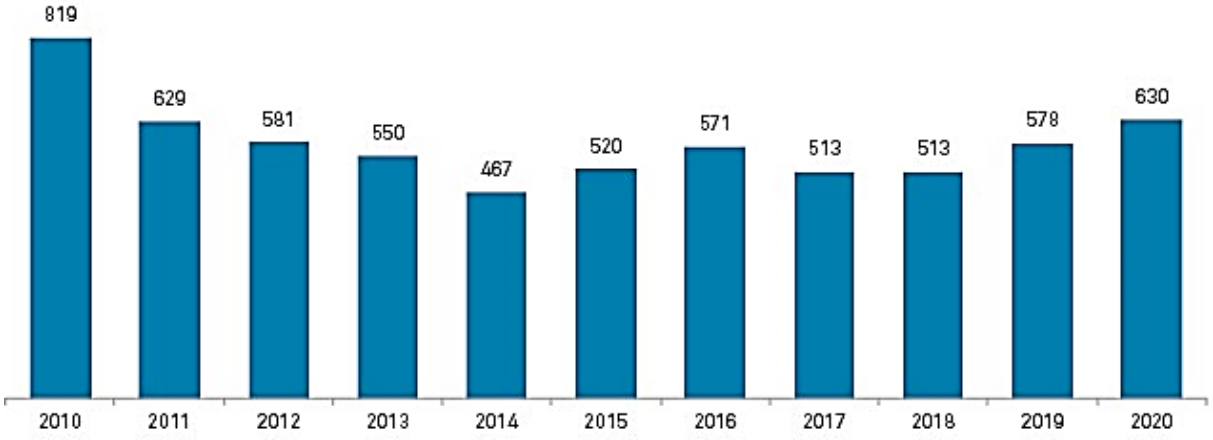


Figure 3: US bankruptcies from 2010 to 2020. Source: S&P Global Market Intelligence.

The negative coefficient for distressed firms means that higher distressed firm sample frequency rates cause lower distressed firm estimated error rates. The positive coefficient for nonbankrupt firms means that higher bankrupt firm sample frequency rates cause higher nonbankrupt firm estimated error rates. The choice-based sample bias is analysed by Zmijewski by comparing the estimates of a probit bankruptcy prediction model and an adjusted probit model. This last sample used the weighted exogenous sample maximum likelihood (WESML) technique to adjust the parameters as shown in the following equation:

$$L^* = [POP/SAMP] \sum_j (B)\, ln[\Phi(H)] \tag{3.3}$$
$$+ [(1 - POP)/(1 - SAMP)] \sum_j (1 - B) ln[1 - \Phi(H)]$$

Where: POP is equal to the number of bankrupt firms divided by the number of firms in the population, and SAMP is equal to the number of bankrupt firms in the sample divided by the number of firms in the sample. The results obtained by Zmijewki proved the presence of the bias and also the methodology to remove that. Despite that, Zmijewki found that the bias did not change the statistical inference and the overall accuracy rates for the bankruptcy prediction model.

The second bias is due to the fact that researchers use “complete data” samples. Substantially bankruptcy firms often do not present a complete set of data. Even if a random sample observation is taken in the population, if there are incomplete data observations, the estimated

parameters and probability will be asymptotically biased (Heckman 1979, Poirier 1980 and Maddala 1983). The distressed firms are the ones that will have incomplete data with a higher probability, in other words, the probability of bankruptcy and of incomplete data have positive covariance. If it is looking to estimate a model using only observation with complete data, it will be committed an error misrepresenting the population probabilities. Since observations with high probabilities of bankruptcy have, on average, low probabilities to be selected because they have less probability to have complete data, the odd of bankruptcy is less than the odd of bankruptcy chosen randomly from the population. The sample selection bias is analysed by Zmijewski by comparing the estimates of a probit bankruptcy prediction model and a bivariate probit model. This last model is composed of two probit equations: the bankruptcy and the complete data equations. The bivariate probit model constrained itself to have the complete set of information for the overall equation. As for the first case, the results proved the presence of the bias, but the author excludes the possibility that the sample selection bias reduced the overall accuracy rates and the statistical inferences.

In 2004, **Hillegeist, Keating, Cram and Lundstedt** were the first to build a rigorous model that used the concept stated by Black and Sholes (1973) and Merton (1974). The authors create a new branch that, before the arrival of machine learning techniques, seems to be the best one in terms of prediction power and minimization of errors. This dissertation cannot be exempt from briefly reviewing the progenitor of the contingency claim models in bankruptcy prediction which are in the middle between the statistic model and the data mining model.

The authors start by explaining the reason why previous models are not reliable:

- The fact that financial statements are based on a principle of going concern[3], can reduce the ability of financial statements indicator to provide information capable to predict future bankruptcy.
- Financial statements are for construction affected by conservatism accounting that understated asset values relative to their market values, in particular for fixed assets and intangibles.
- The fact that financial statements are released approximatively at the end of the year, they are slow indicators of alarm for the potential distress conditions. It is something that the decision-maker cannot use to determine continuously the health condition of the firm.

[3] IAS 1 contains guidance related to the going concern principle, that the entity will remain in business for the foreseeable future and will not be forced to halt operations and liquidate its assets.

- Accounting-based bankruptcy prediction models lack to incorporate a measure of asset volatility that catches the probability that the firm's asset value will decline beyond a value that makes it impossible to repay the firm's debt. The likelihood of bankruptcy, ceteris paribus, increases with the increase of volatility.

Market-based bankruptcy probability model, instead, relies on continuously updated available information, which makes it possible to constantly assess the optimal rate of interest related to a defined firm's situation. It is a more advanced model because rather than only predicting future failure, it also evaluates the creditworthiness of the company.

The Black-Sholes-Merton idea is that equity can be seen as a call option on the value of the firm's asset. If the value of the assets is less than the face value of the liabilities, equity holders will exercise their option at time T when the debt matures paying the strike price of the call that is equal to the face value of the firm's liabilities. Alternatively, if the assets value is not as much as the firm's debt, equity holders will not exercise their option and the firm will go bankrupt. The strongest assumption of the model is that all the debt must mature in one year. This concept is necessary for the model because the BSM idea can be used only to evaluate European options. Where European-style options may be exercised only at expiration, while American-style options may be exercised at any time before the option expires. The BSM equation was corrected to take into account the stream of dividends that the firm pays to the equity holders:

$$V_E = V_A e^{-\delta T} N(d_1) - X e^{-rT} N(d_2) + \left(1 - e^{-\delta T}\right) V_A \tag{4}$$

Where:

$$d_1 = \frac{ln[V_A/X] + (r - \delta + (\sigma_A^2/2))T}{\sigma_A \sqrt{T}} \tag{4.1}$$

$$d_2 = d_1 - \sigma_A \sqrt{T} = \frac{ln[V_A/X] + (r - \delta - (\sigma_A^2/2))T}{\sigma_A \sqrt{T}} \tag{4.2}$$

And where N(·) are standard cumulative normal distributions; $V_E, V_A$ and $X$ are respectively the current market equity value, asset value and the face value of debt maturing at time T; $r$ and $\delta$ are the continuously compounded risk-free rate and the dividend rate; $\sigma_A$ is the standard deviation of asset returns. In equations (4) the term $V_A e^{-\delta T}$ express the reduction in the asset value due to dividends distribution up to time T. The term $\left(1 - e^{-\delta T}\right) V_A$ is an extension to the traditional equation because the dividends received by the equity holders have to be taken into account.

In the BSM model, the probability of bankruptcy for a firm is the odd that the market value of assets is less than the value of the liabilities at time maturity T, so that $V_A(T) < X$. Given that μ is the continuously compounded expected return on assets, the natural log of future assets values is assumed to be distributed normally by the BSM model:

$$ln\, V_A(t) \sim N[lnV_A + \left(\mu - \delta - \frac{\sigma_A^2}{2}\right)t, \sigma_A^2 t] \tag{4.3}$$

Finally, the probability of bankruptcy is the following as defined by McDonald (2002, p.604):

$$\mathrm{N}(-\frac{ln(V_A/X) + (\mu - \delta - (\sigma_A^2/2))T}{\sigma_A\sqrt{T}}) = \text{Prob of bankruptcy in BSM} \tag{4.4}$$

Equation (4.4) shows that the probability of default is related to how much the current value of the firm's assets and the firm's liabilities are close to each other $ln(V_A/X)$ adjusted for the expected growth in asset value $(\mu - \delta - (\sigma_A^2/2))$, correct for the dividend and relative asset volatility.

The authors took the data from Moody's Default Risk Services' Corporate Default database and SDC Platinum's Corporate Restructurings database of only industrial firms between 1980 and 2000. The sample was composed by 756 bankrupt firms and 13547 nonbankrupt firms. The bankruptcy status is defined in a legal way, so only when the firm fills Chapter 11[4].

| | Z-Score | Z-Score$^u$ | O-Score | O-Score$^u$ | BSM-Score |
|---|---|---|---|---|---|
| Constant | −4.95*** | −1.12*** | −5.44*** | −1.47*** | −3.77*** |
| AnnualRate | 0.75*** | 0.81*** | 0.83*** | 0.79*** | 0.54*** |
| Z-Score | 0.18*** | | | | |
| Z-Score$^u$ | | 0.88*** | | | |
| O-Score | | | 0.21*** | | |
| O-Score$^u$ | | | | 0.82*** | |
| BSM-Score | | | | | 0.27*** |
| Log Likelihood | −3972 | −4024 | −3881 | −3831 | −3728 |
| Pseudo-$R^2$ | 0.07 | 0.06 | 0.09 | 0.10 | 0.12 |
| Observations | 78,100 | 78,100 | 78,100 | 78,100 | 78,100 |

Table 5: Performance comparison between the models. Source: Hillegeist et al. (2004).

In table 5, the authors have compared five models defined through a discrete hazard regression. Instead of the traditional way of using a model that only contains one firm-year observation for each firm in a "single-period model", the authors follow the suggestions of Beck et al. (1998)

[4] The Bankruptcy Reform Act 1978 combined Chapter X,XI and XII into a single Chapter 11.

and Shumway (2001). Using a logit model as an example, they start from the formula (2) of the ordinary logit model and implement a discrete hazard regression:

$$P_{i,t} = \frac{e^{\alpha(t)+\boldsymbol{X_{i,t}\beta}}}{1 + e^{\alpha(t)+\boldsymbol{X_{i,t}\beta}}} \tag{4.5}$$

Where the subscript $t$ displays that the model uses multiple-year observations for each firm i. The authors do the same process for the Altman model too. In table 5, the two different specifications of both the model of Altman and Ohlson are due to the fact that the authors want to compare the accuracy using the coefficient derived from the small old data (respectively the data used by Altman and Ohlson in their paper) and the more recent one.

Both the log-likelihood and the pseudo-$R^2$ are used to evaluate the "goodness of fit" in regression models with categorical dependent variables. The log-likelihood function is used to retrieve the maximum likelihood estimator of the parameter of interest, in other terms by finding the parameter that maximizes the log-likelihood of the observed sample. Instead, the pseudo-$R^2$ does the same job that the $R^2$ metric makes in the field of ordinary least square (OLS). Table 5 states that the O-score model outperforms the Z-score one in predictive accuracy, this can be seen through both the log likelihood and the pseudo-$R^2$ measures. What is surprising is that the Altman model based on small and old data outperforms the same model recalculated. Moreover, table 5 states that the BSM model outmatch the others having a pseudo-$R^2$ 20% larger than the O-score and two times bigger than the recalculated Z-score. The result suggests to use the BSM model instead of Z-score and O-score to assess the probability of default. Hillegeist et al. (2004) showed that by using market-based parameters and a contingency claim analysis higher accuracy levels can be reached compared to the usual accounting-based model. Despite this analysis, as the market-based field in general, suffers of a great problem. This is characterized by the fact that only publicly-traded firms can be taken into account. This means that all the other populations of businesses that are not listed cannot be included in the BSM model and should be subject to other less performant models.

## 1.2 A THEORETICAL ANALYSIS OF THE MACHINE LEARNING MODELS IMPLEMENTED

Starting from the first machine learning (ML) models, the field of bankruptcy prediction model became more and more in turmoil as shown in figure 1 by Farias, Martínez and Martín-Cervantes (2021). This second part of the dissertation, about the state-of-art in bankruptcy prediction, cannot be implemented chronologically because of the huge number of researchers. Instead of watching the works of every author present in literature, this section will focus on the main models implemented. However, the starting point should be the definition of machine learning and other broad concepts like artificial intelligence and deep learning to clarify the framework.

The founding event of artificial intelligence as a discipline of study was the Dartmouth Summer Research Project on Artificial Intelligence in 1956, also called the Dartmouth workshop. Eight mathematicians and scientists suggested that computers could be programmed to think and reason: “that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it” (John McCarthy et al., 1956). They call this capacity “artificial intelligence”. In simple words, AI is a broad concept focused on automating intellectual assignments usually accomplished by humans, instead, machine learning and deep learning are specific approaches used to reach that achievement. Indeed, AI’s objectives are to hardcode (explicitly writing) rules for every possible scenario at the wider level in a particular domain of interest, maximizing the likelihood of achieving success. A modern example of AI is self-driving cars. Although AI is skilled at finding the solution to clearly defined logical problems, it fails for tasks that require higher-level pattern recognition with a high rate of precision, as in disciplines like classifications of bankruptcy firms. These more specific works are where machine and deep learning methods perform well. Machine learning is the ability of systems to learn from specific problem training data to make autonomous the building model process and solve related tasks (Janiesch, Zschech and Heinrich, 2021). Meanwhile, deep learning is a mere subset of machine learning. The principal difference concerns how each algorithm learns and how much data each type of algorithm uses. Deep learning is an evolution of machine learning because it automates much of the extraction process, reducing some of the manual human intervention required. It is also more prompt to the use of large data sets, that is why it is also called "scalable machine learning". This skill is particularly interesting because allows using unstructured data. Compared to the structured data (quantitative data) that are highly organized and easily decipherable, unstructured data (qualitative data) cannot be processed and analysed via conventional data tools and methods.

Unstructured data are information that are not arranged according to a pre-set data model or schema, and therefore cannot be stored in a traditional relational database. From an IDC's research[5], the unstructured data will be 80% of global data in 2025. Classical machine learning models are more dependent on human intercession. Experts have the duty to determine the hierarchy of features that has to be understood and cannot relate too much to unstructured data that cannot be easily handled by the non-deep models. Instead, deep machine learning does not necessarily require a labeled dataset. It can ingest unstructured data in its raw form (like text, formulas and images), and it can automatically determine and distinguish a set of features.

Machine learning models are traditionally classified into three macro-categories:

- Supervised learning is applied when the data are in the form of input variables and labelled output target values. The algorithm learns the mapping function from the input to the output. The availability of large-scale labeled data samples makes it an expensive approach for tasks where data is scarce. These approaches can be broadly divided into two main categories:
    1. Classification; the output variable has a finite number of categories. ML techniques try to identify the characteristics that indicate the group to which each case belongs. It is done by examining already classified data and finding a predictive pattern. However, an expert must classify a sample of the database, and then this labeled data will be used to create the model. In the end, the algorithm will be applied to the entire database and the accuracy value will be computed on the so-called hold-out dataset.
    2. Regression; the output variable is a real or a continuous value. Regression problems try to forecast from existing values what other values will be. The simplest case is the linear regression one, but there are more complex methodologies like CART (Classification and Regression Trees) and LASSO (Least Absolute Shrinkage and Selection Operator). The first one is a decision tree algorithm and the second one is a regularization equation useful to choose the more adequate explanatory variables.
- Unsupervised learning is applied when the data are available only in the form of input and there are no corresponding output variables. Such algorithms model the underlying patterns in the data in order to learn more about their characteristics. One type of unsupervised algorithm is clustering. In this technique, inherent groups in the data are

[5] International Data Corporation (IDC) is one of the bigger global providers of market intelligence, advisory services, and events for the information technology, telecommunications, and consumer technology markets.

discovered and then used to predict output for unseen inputs. An example of this technique would be to predict customer purchasing behaviour.

- Reinforcement learning is applied when the task at hand is to make a sequence of decisions towards a final reward. During the learning process, an artificial agent gets either rewards or penalties for the actions it performs. Its goal is to maximize the total reward. Examples include learning agents to play computer games or performing robotics tasks with an end goal.

As shown by Chaochao et al. (2022) in their analysis, the machine learning universe is incredibly huge. For our goals, the review of the ML models will be set in a way to explain only the methodologies that later on will be used in the empirical part of this dissertation, that are, however, represent almost all the models present in the supervised learning literature from the older to the new ones.

- **K-Nearest Neighbour**

The k-Nearest Neighbours (kNN) was proposed by Evelyn and Hodges in 1951 and elaborated by Cover and Hart in 1967. It is a supervised nonparametric machine learning algorithm used to solve classification problems. Nonparametric means that it makes no strong assumptions about the form of the mapping function, that is the function that an algorithm learns from the training data. Indeed, kNN is one of the lazy-learning algorithms. The model works by storing all training data and it waits until the test data are added, without establishing a learning model (Haneen et al. 2019). Hence, the model requires law computation time during the training phase, but more data processing power during the test phase. Moreover, no distribution assumption needs to be made in k-NN. Thus, kNN could be the best choice for any classification study that involves little or no prior knowledge about the distribution of the data. The simple idea behind the functioning is that the model classifies an unlabelled test sample by identifying (k) nearest

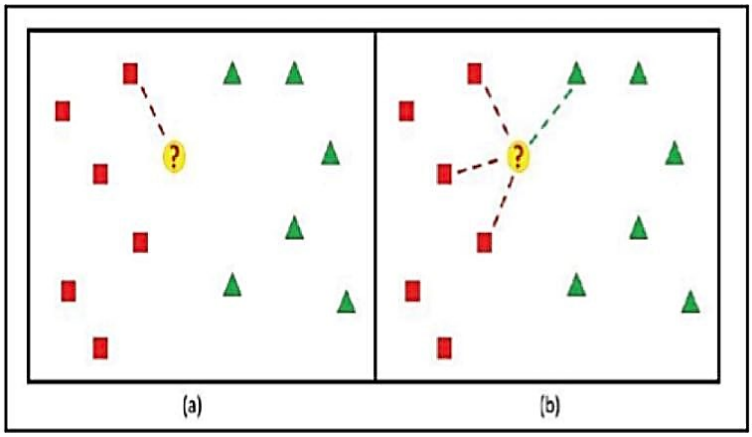


Figure 4: the k-NN decision rule. Source: Imandoust and Bolandraftar, Mohammad (2013).

neighbours observations to the sample and uses those neighbours to identify the class of the unseen sample (Kataria, Aman and Singh, 2013). The nearest neighbour (NN) is the simplest form of kNN, that is when k=1. Each object is classified considering its adjacent labeled observation. Figure 4 shows two kNN models with decision rules respectively k=1 and k=4 for a sample with two classes. Using the simple NN, figure 4(a), an unknown sample is classified by using only one known bordering sample. Instead, in figure 4(b) the closest four samples are considered to define the unknown one. In these fictitious cases the performance and the output choice do not change for different k levels, instead in reality, the choice of k is one of the determinants of the efficiency of the model with the choice of the distance metric to apply. Hence, the first element that affects the kNN performance is the sensitivity of the selection of the neighbourhood size k. If k is very small the local estimation could be poor given the data scarcity or the unreliable and wrong data. To correct that problem the number of k that the model has to find to define the unknown observation can be increased. If this is done too much, the estimates could become over smoothing and the performance will decrease. The problem is to define a suitable neighbourhood size k that maximizes the classification performance of kNN. The fact is that there are no pre-defined statistical methods to find the best value of k. The only way is a cross-validation procedure in which a random k value is initialized and the kNN model is computed on the validation sample, initially choosing a small value of k and then increasing that value. To choose the best k, a graph is derived between the error rate and k. Then the value k which gives the minimum error rate is selected (Guo et al. 2003).

The second element that affects the kNN performance is the specific distance measure that the algorithm should use when it defines the distances between each of the training data and the test sample. In 2013 Kataria, Aman and Singh analysed the three best-used methodologies in this field, that are:

a) Euclidean distance:

$$ED(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

The ED is the distance between two points, in Euclidean space, it is the length of a line segment connecting them. It can be calculated from the Cartesian coordinates of the points using the Pythagorean theorem. In consequence, sometimes is called the Pythagorean distance.

b) Cityblock or Taxicab distance:

$$MD(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

The MD, also known as Taxicab distance or rectilinear distance, was invented by Hermann Minkowski in the 19th-century. This distance represents the sum of the absolute differences between the opposite values in vectors.

c) Cosine distance:

$$CosD(x,y) = 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt[2]{\sum_{i=1}^{n} x_i^2}\,\sqrt[2]{\sum_{i=1}^{n} y_i^2}}$$

The CosD, also called angular distance, is obtained from the cosine similarity that measures the angle between two vectors and it can take values between -1, which denotes an exact opposite match, and +1 which indicates two equal vectors. Finally, CosD is obtained by subtracting the cosine similarity from 1 (Haneen et al. 2019). The authors found that Euclidean distance has a higher efficiency compared to the other ones. However, they pointed out that the kNN classifier has limitations such as great calculation complexity and so the slowness of the algorithm, a huge dependence on the training set and there is no weight difference between classes. The kNN algorithms give the same importance to every neighbour, assuming perfectly defined boundaries between classes, that in reality are not. Recently to enhance the precision of kNN several approaches have been implemented, the one with the more encouraging results is the Fuzzy-kNN. This last version of the kNN model calculates a fuzzy degree of membership of each observation to the classes of the problem. The result is that it ends up with smoother boundaries in the space on the class boundary (Maillo et al. 2017). It did so by assigning to all the instances a membership value in each class rather than a binary outcome of “bankruptcy” or “nonbankruptcy”. Opposite to the simple kNN, with the degree of membership, the points closer to the query participate with a larger value to the definition of the membership compared to far away neighbours. In 2011, Chen et al. practically combine the FkNN model, in which the parameters k (number of classes) and m (the fuzzy strength parameter that weights each neighbour’s contribution), with the continuous Particle Swarm Optimization (PSO). The latter is a computational method that optimizes iteratively trying to find: i) the most discriminant features; ii) the best size of k and m given a measure of quality as the AUC-ROC curve (Area Under the Curve and Receiver Operating Characteristics). The results state how the FkNN-PSO model performs significantly better compared to the other state-of-art classifiers in terms of sensitivity and specificity, being a quite fast algorithm. However, the authors required future investigation to find if the model performs as well as in their research when applied to bigger financial datasets.

- **Support Vector Machine**

The Support Vector Machine (SVM) was developed by Cortes and Vapnik (1995). It is a supervised learning algorithm generally used for classification problems but that can be rearranged also for the regression ones. The main idea behind the model is that based on the labeled data in the training phase, the algorithm can find the optimal hyperplane which can be utilised to classify new data points. Considering a one-dimension line in which lies some points belonging to two classes, a single point could divide the classes. In a two-dimension, the point becomes a line and in a three-dimensions, a plane is required to divide the space. Hence, in high-dimension space, the hyperplane is the line that separates the classes (Noble, 2006).

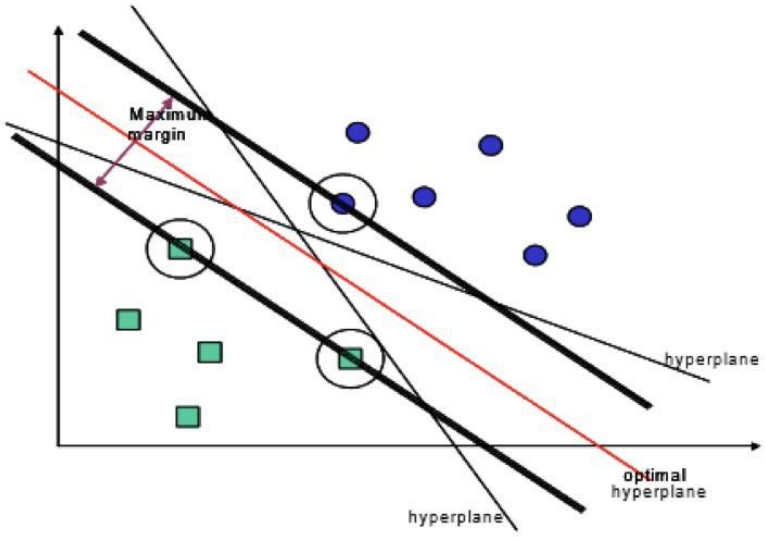


Figure 5: the optimal hyperplane. Source: Kotsiantis (2007).

Roughly speaking, there is not only one hyperplane that divides the space. The SVM algorithm chooses the one that maximises the margin and thereby creating the largest possible distance between the separating hyperplane and the instances. Once the optimal hyperplane is found, support vector points are the ones that lie on its margin as shown in figure 5. All other training data are essentially useless for determining class boundaries. Given the nonlinearity and misclassification of real datasets of instances, many times the SVM may not be able to divide the space so easily. To handle these cases, a soft margin can be included to allow some points to be inside the margin of the separating hyperplane. To control the numbers of violations in the training test derived from the soft margin an error upper bound variable is introduced. However, even introducing a soft margin, most real problems involve nonseparable data. The kernel function allows to find a solution to the inseparability problem, mapping the data onto a higher-dimensional space, also called transformed feature space, in which the optimal hyperplane is found. As Genton et al. (2001) described, the selection of the best kernel function

is a problem that can be addressed only through a cross-validation procedure. The empirical part of this dissertation will verify the best SVM model between at least two popular kernels present in the literature:

- Polynomial Kernel $= K(x, x_i) = (x * x_i + 1)^d$

Where d is the degree of the polynomial, the polynomial kernel is an efficient and accurate function for linear data.

- Gaussian Radial Basis Function (RBF) $= K(x, x_i) = e^{\left(-\frac{1}{\delta^2}(x-x_i)^2\right)}$

Where the value of $\delta^2$ varies from 0 to 1 and usually 0.1 is the most favourited in literature, especially in the case of non-linear data. Worthy to be mentioned is the research done by Shin, Lee and Kim (2005). They state that SVM has a few attractive characteristics compared to other machine learning techniques like Neural Network (NN). The first one is due to the fact that SVM follows the structural risk minimization principle (SRM) and so it has superior generalization performance. Given the problem of overfitting, when a model becomes strongly tailored to the training dataset and generalizes poorly to new data. The SRM principle reduces this problem by balancing the model's complexity against its performance at fitting the training data using a trade-off parameter upper bound C which is the regularization term. According to Tay and Cao (2001), a small value for C would under-fit the training data because the weight placed on the training data is too small thus resulting in small values of prediction accuracy on both the training and validation sets while a large value for C would overfit the training data. The second one is the fact that since the training of SVM is the solution of constrained quadratic programming (QP), the SVM guarantees the existence of a global and unique optimal solution. Instead, the traditional empirical risk minimization (ERM) used by NN models, optimizes the weights of NN in a way that the sum of square error is minimized along the steepest slope of the error surface, the result from training may be massively multimodal, leading to non-unique solutions, and be in danger of getting stuck in a local solution. The last one is that the SVM technique is flexible from a mathematical viewpoint and it is also quite easy to interpretable the logic behind it. The NN models are not so easy, they are most of the time black boxes. Shin, Lee and Kim (2005) evaluate 2320 medium-size manufacturing firms, half of which filed for bankruptcy and half did not. They took the data from the Korea Credit Guarantee Fund from 1996 to 1999. They initially select from 250 financial ratios the first 52 most significant ones through a t-test. Then, the authors select 10 ratios using an MDA stepwise method. They use an RBF kernel to take into account the nonlinearity of the variables. Finally, they perform the analysis of different values of upper bound C and kernel parameter $\delta^2$ to find the best one,

comparing the RBF-SVM model with a Back Propagation Neural Network (BPN). The result confirms another statement of Tay and Cao (2001) so that a small value of $\delta^2$ would over-fit the training data and so while the accuracy on the training set increases as $\delta^2$ increases; on the other hand, the accuracy on the validation set shows a tendency to decrease with increasing $\delta^2$. The best performance occurs with C=75 and $\delta^2$=25%. Finally, the last important conclusion they reach is that SVM has higher prediction accuracy than BPN when the training size becomes smaller, while for large dataset size the results of the two models are quite the same. SVM should be choose as the preferable model for small dataset.

To improve the accuracy and precision of the SVM model recent research proposed a sophisticated fuzzy-SVM (FSVM) algorithm. As in the case of the kNN model, it is introduced a fuzzy degree of membership of each observation to the two classes of the problem. The simple idea is that if an input observation is ascertained as an outlier, it is not taken into account and that should reduce the error term. This is done mathematically through the fuzzy membership $s_i$ that can be regarded as the attitude of the corresponding training point toward one class in the classification problem and the value $1 - s_i$ like the attitude of meaningless (Lin and Wang, 2002). Chaudhuri and De (2011) took the data from the Division of Corporate Finance of Securities and Exchange Commission (SEC) of 100 large firms, half of which fill Chapter 11 of the US bankruptcy code in 2001 and 2002. They compute 32 financial ratios and perform an analysis taking respectively only the first 6, 12, 30 and all the 32 variables and compare the performance of the models. Founding that the best result for the pure SVM is obtained with C=50 and $\delta^2$=10% using only the first six ranked ratios through a t-test. They also found that FSVM outperforms all the other models, respectively a Logit, NN, pure SVM and a genetic algorithm with the SVM (GA-SVM).

| | Sensitivity | Specificity | Overall accuracy % |
|---|---|---|---|
| Method | Accuracy % | Accuracy % | |
| PNN | 76 | 88 | 82 |
| FSVM | 98 | 90 | 94 |

Table 6: Results of PNN and FSVN classification. Source: Chaudhuri and De (2011).

To be clear in table 6 the sensitivity score is equal to the ratio between the number of true positive classified cases on the train sample population, instead specificity is equal to the ratio between the number of true negative classified cases on the train sample population. Overall accuracy is an average value between sensitivity and specificity. Table 6 demonstrates the power of the FSVM model in a clustering problem, comparing it with a NN model which is a

specific type of neural network that implements a statistical algorithm called kernel discriminate analysis.

- **Neural Network**

Artificial Neural Networks (NN) are brain-inspired models that mimic basic biological systems. The first NN was proposed by McCulloch and Pitts in 1943. The simpler NN model is constructed by only one hidden layer called perceptron that computes the sum of weighted inputs and outputs through an adjustable threshold. Perceptrons are able to classify only linear separable instances. Given the nonlinearity of the majority of real phenomena, it is preferable to use a Multilayer Perceptron model (MPL) also known as Deep Feedforward networks. An MPL is composed of several layers of nodes. The first one is the lowest layer or input layer where information is received. The last and higher layer is the output layer where the solution to the problem is obtained (bankruptcy/nonbankruptcy). Between the input and the output layer, there are one or more hidden layers that are composed of a certain number of neurons or nodes connected to the subsequent synapses like in the human brain.

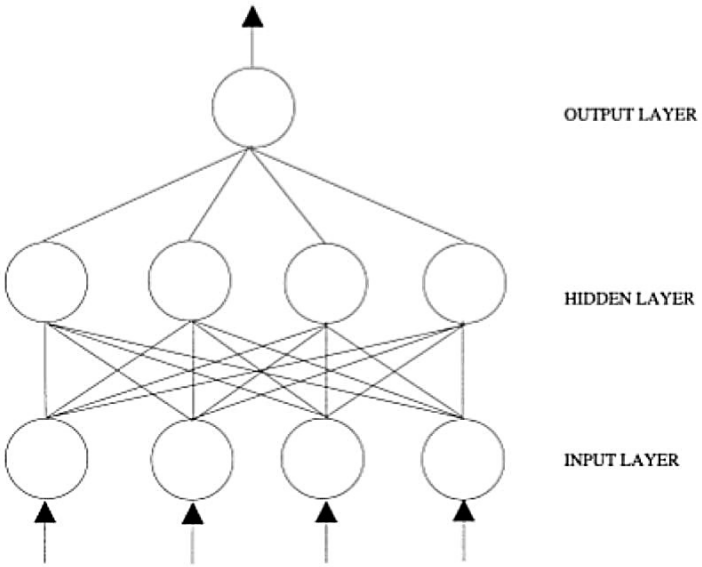


Figure 6: A typical MPL. Source: Zhang, Patuwo and Hu (1998).

Like other machine learning techniques, there are at least two phases: the training phase and the test phase. In the training one, the set of data is filled into the input nodes. The activation values of the input nodes are weighted and accumulated at each node in the first hidden layer. The total is then transformed by an activation function into the node's activation value. It in turn becomes an input into the nodes in the next layer, until eventually the output activation values are found. The training algorithm is used to find the weights that minimize some overall

error measures such as the sum of squared errors, $SSE = \sum_i^n e_i^2$, or mean squared error, $MSE = \frac{1}{n}\sum_i^n e_i^2$. After the network achieved a desired level of development, in the test phase new instances input reached the input nodes and there are propagated forward through the network to determine the ultimate accuracy of the network. What is crucial to reach the higher possible accuracy level is the definition of the following variables of the model:

- The numbers of hidden layers and hidden notes. These are the elements that allow the network to capture the pattern in the data and to perform the complicated nonlinear mapping. Theoretical publications had showed that a single hidden layer is sufficient for many complex problems (Cybenko, 1989; Hornik et al.,1989). Even so, only one hidden layer could necessitate a large number of hidden nodes and this is unattractive because it causes a reduction in the generalization ability of the network. Hence, the most frequent and preferable choice is a two hidden layer network (Barron, 1994). Increasing the depth of the network with more than two hidden layers does not provide any improvement (Zhang, 1994). The issue of determining the number of nodes is subject to several rules of thumb as using "2n+1", "2n" and "n/2" where n is the number of input nodes. However, Zhang, Patuwo and Hu (1998) have found that networks with the same number of hidden notes and input nodes are the ones with higher accuracy.
- The number of input nodes. It should be equal to the number of explanatory variables, in bankruptcy prediction should be equal to the number of ratios chosen to solve the classification problem.
- The interconnection of the nodes in the layers. Technically, the network is fully connected if one layer is fully linked to all nodes in the next subsequent layer apart for the output layer as in figure 6. However, it is not the unique solution but there are other possible choices as sparsely connected networks and direct connections between input and output nodes.

To better understand the general characteristics of neural networks and to have an idea of what the author has done in the empirical part of this dissertation, particularly why some functions are chosen compared to other ones, it has to be done a review of two other elements:

- The activation function.

It is sometimes called the "transfer function". It determines how the weighted sum of the input is converted into an output from a node or nodes in a layer of the network. It can be linear on nonlinear and the choice of the type of activation function has a large impact on the performance

of the neural network. Accurately speaking, the more suitable activation functions are bounded, monotonically increasing and differentiable. The popular ones are:

1) The linear function: $f(x) = x$.
2) The sigmoid (logistic) function: $f(x) = \frac{e^x}{e^x+1}$.
3) The hyperbolic tangent (tanh) function: $f(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$
4) The radial basis function: $R_i(\boldsymbol{x}) = e^{(-\frac{\|x-u_i\|^2}{2\sigma_i^2})}$

Where $R_i(\boldsymbol{x})$ is the radial basis function in the $i-$ th receptive field unit, $\sigma_i$ is the spread width and $u_{\mathrm{i}}$ is the center of that unit and is a vector with the same dimension as $\boldsymbol{x}$. Furthermore, the output node performs the weighted sum associated with each receptive field (Tseng and Hu, 2010). The last activation function allows to build a different type of NN with compared to the multilayer perception one (MPL). This sophisticated model is called radial basis function network (RBFN), where each hidden node has its own radial basis function $R_i(\boldsymbol{x})$. The difference between MLP and RBFN is due to the fact that in the former one each connection has its own weight, while in an RBFN, there are no connection weights between the input layer and the hidden layer. However, in general, a network can have more than one activation function for different nodes in the same or diverse layers. There is no proof of the superiority of one to another for the hidden nodes. Conventionally, the logistic activation function is more suitable for the output nodes for binary classification problems.

- The training algorithm

The NN training is an unconstraint nonlinear minimization problem where the training algorithm has the purpose of finding the weights of the network and iteratively modifying them to minimize the error term of the sample. Even if there are many different training algorithms all of these do not guarantee to find a global optimal solution for the problem. In practice, they all suffer from the local optimal problem and they can only find the best local optimal solution. The most popular and widely used training algorithm is the Back Propagation (BP). It does the following six steps:

1) Gives a training sample to the network.
2) Calculates the error in each output node derived from the difference between the output of the network and the desired one from the sample.
3) Calculates the local error, so what each neuron should have been and how much it should have been lower or higher to reach the desired output through a scaling factor.

4) Corrects the weights of each neuron given the analysis made before.
5) Gives responsibility also to the nodes at the previous level, ranking for higher weights in a backward process.
6) Repeats the steps until the best minimum local error is obtained.

Mathematically, the general rule to update weights is:

$$\Delta W_{ji} = \eta \delta_j O_i$$

Where $\eta$ is the so-called learning rate that defines the step size in the gradient descent search. Where the gradient measures the change in all weights with regard to the change in error and descent is the action of going downwards. The gradient descent algorithm (BP) searches the parameters' values that minimize the cost function towards a local minimum. This can be done because a convex activation function is imposed. However, high value of $\eta$ enables the model to move faster towards the target but it increases the possibility of never reaching that target because it causes network oscillation in the weight space.

Worthy to be mentioned are the researches of Tseng and Hu (2010) and Zhang (2017). The former study makes an analysis using: backpropagation multilayer perception (MPL), radial basis function network and two types of logit model. They took the data from DATASTREAM and FT EXTEL Company Research of 904 UK public companies, 353 of which failed between 1985 and 1994 but only 32 had sufficient financial information and are included in the final sample. The explanatory variables selected are only three: working capital over operational expenditure, after-tax profit over total assets and change ($\Delta$) in cash over total liabilities. The learning parameter $\eta$ were set for both the network equal to 0.05 and they imposed as stopping rules that the algorithms will arrest after 200000 epochs. Both the network was settled with only one hidden layer and 15 hidden nodes. The results were quite unexpected. In the training sample, the two NNs outperform the logit models, but in the validation test, only the RBFN outperforms the two logit models while the MPL one has the tinier accuracy level. That is maybe due to a problem of overfitting and demonstrates that the radial basis function network has superior generalization power compared to a simple MPL. However, to make an exhaustive analysis the number of hidden layers should be increased to see if the performance of the NNs increases. The second research is significant in the NN field because it verifies the increase in the accuracy of the network due to the adoption of a dropout method. It is a technique that temporarily removes randomly hidden or visible nodes in the network. It decreases the amount of computation power needed and the possibility to overfit in the learning phase, obtaining poor results in the test phase. The author collected the dataset of 250 companies from a previous

analysis published by Sebastian Tomczak, a researcher at the University of Science and Technology in Poland. He compares a NN with and without the dropout method, with a Random Forest and a kNN algorithm. He uses a genetic algorithm (GA) to choose the best ratios from the available data. For the sake of brevity, GAs are general-purpose search and optimization procedures, they are searchers' heuristic that mimics the process of natural evolution (Bateni and Asghari, 2020). In literature, they are used to select the best explanatory variables (ratios) for the bankruptcy problem. The author found that with a dropout rate of 0.3 and training the NN on 150 of the total 250 companies, the NN reaches 99,5% accuracy one year ahead in the hold-out sample.

- **Random Forest**

Random Forest (RF) is a type of ensemble machine learning algorithm first developed in a proper model by Breiman (2001). An ensemble ML is composed of a set of independently trained classifiers whose predictions are combined when the comprehensive model has to classify new instances. The brick of the random forest, but also of other models such as the Extreme Gradient Boosting (XGB) and the CatBoost, is the Decision Tree (DT) model formally known as Classification and Regression Tree (CART). It is a weak or basic classifier, so a learner that performs slightly better than a random guess. It is a nonparametric supervised machine learning where the branches of the tree are derived from the data features. It is for definition a quite comprehensible model but this is balanced by a quite low accuracy performance and suffering from overfitting of the training data.

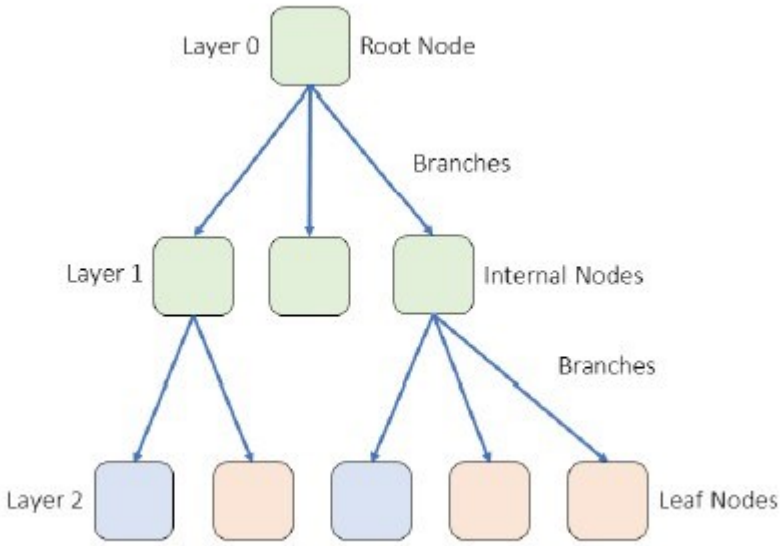


Figure 7: A standard decision tree.

Figure 7 shows graphically how a decision tree model is arranged. The given dataset is partitioned recursively into two groups based on of certain criteria until a predefined stopping condition. There are three types of nodes: the root node is the first one and is the feature that divides in a better way the training set. The points where decisions are taken and the tree ends are called leaf nodes. In the middle, the points where the subgroup is broken to a new sub-group or a leaf node are known as internal nodes. This last type is also called decision node because it is where the nodes dived further on the basis of the best attribute of the subgroup. The split of a node to another takes place only on a certain threshold value of one characteristic based on purity. The decision tree algorithm split the node where it finds the best homogeneity for the subgroup. It is called a pure node a sub-node that contains only elements that have the same and unique class (for example only bankruptcy or only nonbankruptcy), so it has the maximum value of homogeneity. To be clear, the attributes are the explanatory variables, so in the bankruptcy field the ratios of the firms. The choice of the best attribute and threshold value couple arises on the basis of an algorithm that will give the purest nodes. In literature the more utilized ones are:

- Information gain or entropy: $E = -\sum_{i=1}^{c} p_i * \log(p_i)$

Where $c$ is the number of classes or attributes and the prior probability of each given class is $p_i$. In this way, the DT model maximizes the information gained at every division of the tree (Kotsiantis, 2007). Entropy is a measure of information that reveal the disorder of the attributes. The optimum split is chosen between the feature with less entropy, that is at the maximum level when the probability of the two classes is identical. Instead, a node is pure when the entropy has its minimum value, 0, and this node will not be split again.

- Gini index: $G = 1 - \sum_{i=1}^{c} p_i^2$

The Gini impurity indicates the frequency at which the elements of the dataset will be mislabelled when they are randomly labeled. The minimum value is 0 when the node is pure. As for entropy, the optimum split is defined by the features with the higher Gini Index. The literature, however, has pointed out that there is not a single superior method (Murthy, 1998). What is certain from previous studies is that the choice of the best algorithms to find purity does not solve the problem of overfitting and the subsequent low general predictive accuracy of the DT model. There are a few methodologies used to partially solve these problems, all of them try in a different way to build several different trees based on a subsets dataset.

The first procedure is called Bagging or Bootstrap Aggregation (Breiman, 1996). It randomly creates new subsets of data through sampling with replacements from the training dataset. Then

it constructs and makes learning as many DT models as the number of subsets. For example, given training size $t$, the bagging algorithm generates a certain number of new samples by selecting $t$ times one of the $t$ observations randomly from the training sample. The characteristic of this methodology is that all observations have an equal likelihood to be selected and some training combinations may not be constructed, usually, 1/3 of the sample observations are not picked. A bagged ensemble classifier will allocate new observations by asking each decision tree to classify them and it will choose the one with the higher number of votes. It gains from a high frequency of disagreement between the single decision trees. However, Breiman warns that with a small initial dataset the procedure could fall into less accurate performance because each tree sees only a smaller dataset of the already small comprehensive one (Maclin and Opitz, 1998). The intrinsic problem with this procedure is that decision trees embedded in the model may be quite similar and therefore have high correlations, decreasing the general accuracy of the bagging classifier. The random forest (RF) algorithm made by Breiman (2001) modifies the way in which the decision trees are constructed in the bootstrap procedure decorrelating the trees. DTs are built not taking every explanatory variable in the sample but considering only a smaller set of predictors in different combinations. For the sake of clarity, if the ratios of our dataset are the return on investment (ROI), the debt-to-equity ratio and the cash to total asset ratio, each tree can only consider two or fewer ratios mixing them in different ways. In literacy, this is known as random subset projection or random feature projection (Lee et al., 2019). The hyper-parameter $m$, the optimal number of variables for the DTs in the random forest model, is usually chosen using some rule of thumb, given that for very large samples the cross-validation procedure is time-consuming. In classification problems, $m$ is picked between 1 and the square root of the number of the parameters available.

The high accuracy performance of the RF model has been verified in the literature. Barboza, Kimura and Altman (2017) showed how RF outperforms all the other classifiers, even the bagging one. They retrieve 11 financial data on American and Canadian companies from 1985 to 2013 analysing more than 10000 firm-year observations. The authors found that RF has 87% of accuracy in the testing sample, compared to the 85% of the bagging model and 86% of the boosting model, while the statistic and naïve ML models perform poorly, with the best of them an SVM- RBF model reaching only 75% of accuracy. The superiority is also evident watching the AUC-ROC rate and Type I and II errors. A more recent and comprehensive analysis was made by Jabeur et al. (2021). They compared RF with other 8 models: MDA, logistic regression, SVM, two different types of NN, extreme gradient boosting and CatBoost. The authors took a sample of French firms from the Orbis database taking 18 ratios to capture the

profitability, liquidity and solvency situations of the observations. Comparing the result obtained they found that the bosting model as XGB and CatBoost outperform the other, however, the RF model perform quite well obtaining the third position in term of accuracy (ACC) rate of about 94% in a one-year horizon. Worthy to be mentioned is the fact that RF reaches 74% of ACC rate in a three-year horizon, overcoming also the more sophisticated boosting model being the more stable model over time.

- **Extreme Gradient Boosting and CatBoost**

The second procedure used to answer the weakness of the decision tree model is called Boosting (Freund and Schapire, 1996). This method, like the bagging one, builds a series of classifiers. The main difference with the former is that instead of randomly picking a given number of observations from the training data sample and constricting a series of DTs, the boosting algorithm builds a series of DTs depending on the performance of the previous trees in the series. Hence, the classifiers will concentrate on the sample of observations that are more frequently incorrectly predicted by previous trees. In other words, like bagging, given training size $t$, the bagging algorithm generates a certain number of new samples by selecting $t$ times one of the $t$ observations conditionally to how often that observation was misclassified by previous classifiers.

In literature, the first two types of boosting classifiers are: Arching (Breiman 1996b) and AdaBoost (Freund and Schapire, 1996). Initially, they both set the probability of randomly take the observations and then recompute that probability iteratively for each tree built and added to the ensemble classifier. In AdaBoost, the probabilities are computed in the following way: if the sum of the misclassified observations probabilities of the trained classifier $C_k$ is $\epsilon_k$. Therefore, the likelihood for the next trial is generated from the probabilities of incorrectly classify instances $C_k$ multiplied by a factor $\beta_k = (1 - \epsilon_k)/\epsilon_k$. The result should be normalized to obtain a sum equal to 1. The peculiarity is that AdaBoosting used a weighted voting rule, multiplying each answer of the classifiers $C_1, \dots C_k$ to their relative coefficients in logarithm $\log(\beta_1), \dots \log(\beta_k)$. In Arching the probabilities are renovated in a different way on contrary to AdaBoosting, this methodology uses an unweighted voting rule. Setting $m_i$ as the number of times an observation was classified wrongly by the previous $K$ classifiers. The likelihood $p_i$ for selecting the observation $i$ into the classifier $K + 1$ in the training set is:

$$p_i = \frac{1 + m_i^4}{\sum_{j=1}^{N}(1 + m_i^4)}$$

Maclin and Opitz (1998) performed an analysis that compares the bagging and the two boosting models. Worthy to recall is that these methodologies do not mandatory need a decision tree as building model, in fact, the author proposed a dual analysis comparing the model using both a DT and NN models. The authors used 23 datasets from the UCI repository which is a publicly available library to test and perform ML analysis. The results state that bagging always performs better than single NN and DT models. On another hand, boosting results are more volatile. Arcing sometimes performs worse than the simple single classifier and other times it dominates both the single and the bagging classifiers. With AdaBoost, the variability is even more emphasized. Freund and Schapire (1996) in their analysis proposed the problem of overfitting the training dataset as an explanation of this behaviour. This reasoning seems to be correct particularly for the case of AdaBoost because the weighting vote rule increases the odds of overfitting while an unweighted scheme generally does not lead to this type of problem (Sollich and Krogh 1996). Another explanation of this path is that the methodology used by AdaBoost to update the probabilities could accentuate noisy observations. To clarify, noise examples are observations whose features were wrongly inserted into the dataset. Finally, Maclin and Opitz (1998) stated that bagging and boosting models construct on DT or NN perform quite in the same way.

The next evolution step in ensemble models is Gradient Boosting Machine (GBM), originally derived by Friedman (2001). It is the combination between the AdaBoost model and the weighted minimization procedure analogous to the gradient descent search in a neural network. The main idea behind this model is to build base decision tree learners that are maximally correlated with the negative gradient of the loss function related to the whole ensemble (Natekin and Knoll, 2013). The purpose of Gradient Boosting classifiers is to minimize the loss function which is the difference between the actual class value of the training dataset and the predicted ones. This is different from what happens in AdaBoost that the weights of the previous trees are adjusted when new trees are included. In GBM the weights of the previous trees remain unchanged each time a new tree is connected to the ensemble model. The key element for the popularity of the GB is that the model has high flexibility and customizability. The first element that researchers can freely define is the type of loss function. In case of classification problems, the standard and most used are the Bernoulli and AdaBoost loss function:

1) $\Psi(y, f)_{Bern} = \log\left(1 + e^{-2\bar{y}f}\right)$
2) $\Psi(y, f)_{Ada} = e^{-\bar{y}f}$

Where the notation $\bar{y}f$ means the number of correct discriminated observations. Natekin and Knoll (2013) showed that with both functions the GBM model can overfit the data but given the nearly linear impact of outliers in the Bernoulli loss function, it is usually less sensitive to this type of problem and so it is our best standard choice. Concerning the overfitting problem, several different approaches were used in the literature in the GBM framework. The simpler one is subsampling. It is a procedure of regularization that introduce some randomness into the fitting process. At each learning step the DT is used to fit only a random part of the training dataset. As a standard way to proceed, the sample observations are taken without replacement but it is also possible to follow a bootstrap approach. The subsampling parameter is known as "bag fraction". In literature for a big number of observations in the dataset the default value is 0.5, which means sampling using only 50% of data at each step. This changes the paradigm to what in literature is called "Ordered boosting". The second procedure, which is the most used one, is shrinkage. As the learning rate parameter in NN, in GBM it is used to reduce the effect of each additional tree. It reduces the size of incremental steps and it is usually indicated by the hyperparameter $\lambda$. The smaller $\lambda$ is, the higher is the generalization accuracy obtained by the GBM model. The cost of this upgrade in accuracy is to increase the number of iterations $M$ required to find the optimal loss minimum. However, this procedure affects positively also the single DTs in the ensemble model increasing their capacity to capture more continuity in modelled effect, having a smoothing effect. Hence, it is profitable to train GBMs that have small $\lambda$ values (Friedman, 2001; Buhlmann, 2006). Correlated with $\lambda$ is the early stopping approach. If the ensemble model is stopped at the corresponding number of trees predefined, the minimum could be never reached by the algorithm. Different level of $\lambda$ required different number of iterations $M$ to reach the global minimum. The most frequent approach is the cross-validation procedure, given that there is not a simplified approach rule.

In the last few years, the implementation of GBM models in data science and various field has led to the birth of even more sophisticated ensemble models. The first one is the XGBoost (Chen and Guestrin, 2016). It is a refined version of a gradient boosting model. It is focused on increasing the scalability of the model, decreasing the storage space requirement and increasing the quickness of the training process compared to a simple GBM. The main difference with the latter model is that XGBoost added a variable in the loss function to control the complexity of each decision tree avoiding overfitting of the training data and increasing the general accuracy of the model. Mathematically, the control factor is usually defined in the following way:

$$\Omega(h) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$

Where $w$ are the scores of the leaves and T is the number of leaves. Including this loss function in the split criteria of each decision tree reduces the branching of trees leading to a pre pruning strategy. The value of $\gamma$ controls the minimum loss reduction gain required to break the node. Higher is $\gamma$ simpler will be the DTs of the model. However, as the other GBM, the researchers can set also other parameters to avoid overfitting as reducing the deepness of the trees (number of leaf nodes) or setting a low shrinkage and the other regularization variables (Bentéjac, Csörgő and Martínez-Muñoz, 2021).

The last important model in this field is the CatBoost one (Prokhorenkova et al., 2018). Its name come from two words "Categorical" and "Boosting". Its peculiarity is the ability to automatically pre-process categorical features like for example gender, name, and nation. For the goal of this dissertation, however, the main advantages of this model derived from the fact that during the learning process of the DTs, CatBoost has oblivious symmetric trees. This means that all trees will have at the same level the same number of nodes that test the same predictor (ratios). This has a dual favourable effect: the first one is the lowering of overfitting and reduces the need for extensive hyperparameter tuning, the second one is the speed and scalability particularly required when researchers deal with big dataset.

The CatBoost model in the bankruptcy prediction field has been examined only by Jabeur et al. (2021). Given the remarkable results obtained by the authors, this dissertation should verify if that boosting models outperform the other ML and statistic models. In terms of generalized accuracy, the author of this dissertation expects that this will be the model that will have the higher performance in the following empirical part.

| | Differences (%) | | |
|---|---|---|---|
| | Y-1 | Y-2 | Y-3 |
| CatBoost -DA | 3.9** | 9.9* | 8.9 |
| CatBoost -LR | 7.5*** | 6.6 | 2.0 |
| CatBoost -SVM | 4.3*** | 5.6** | 8.3** |
| CatBoost -NN | 4.6*** | 16.4*** | 9.6*** |
| CatBoost -RF | 6.3*** | 5.7 | 2.5 |
| CatBoost-GBM | 4.4*** | 13.7*** | 4.6 |
| CatBoost-DNN | 48.8*** | 36.8*** | 26.3*** |
| CatBoost -XGBoost | 6.3*** | 6.3*** | 4.9 |

Notes: ***, **, and * indicate a significant coefficient at the 1% level, 5% level, and 10% level, respectively.

Table 7: Differences (%) between AUC achieved with CatBoost and 8 other models. Source: Jabeur et al. (2021).

Table 7 shows the potentiality and improvement that CatBoost reached compared to all the other 8 models in the analysis of Jabeur et al. (2021). Impressive is the high difference in performance in the one-year horizon, near 50% compared to Deep Neural Network and 6% compared to the XGBoost model. The results remain excellent in the two- and three-year horizons too, being the best models in all of them in the AUC-ROC score. However, as it will be explained in the following chapter, the AUC-ROC measure has some lack in defining the discriminant ability of the models. Anyway, one of the interests of this paper is to evaluate whether this model will work well even when the explanatory variables (ratios) are few or if other models described above perform better in such situations.

# CHAPTER 2: RESEARCH METHOD DESCRIPTION

The empirical part of this dissertation will implement several reviewed models from the literature, comparing them and verifying which one is the best for predicting at one, two and three years prior to the failure. The author decides to examine eight models from the most famous and accurate ones, particularly it will be analysed: the multiple discriminant analysis (MDA) or Z-score, the Logit Regression, the k-Nearest Neighbour, the Support Vector Machine, the Neural Network, the Random Forest, the XGBoost and the CatBoost models. The data sample is composed of European Union data of 3870 bankrupt firms and 15276 nonbankrupt firms, for a total of 19146. The observations are taken from 2013 to 2020, rounding up eight years of balance sheet data. It has been decided to use such a huge data sample to make the analysis as realistic as possible, retrieving a large number of observations but a small number of explanatory variables. Only three explanatory variables that performed well in literature to distinguish between the bankruptcy and nonbankruptcy outcomes were acquired. The data sample comes from the Orbis database, it contains financial and business information on about 200 million companies worldwide and it is used specially to gather data of non-listed companies. To make a real comparison between the models, it has been decided not to perform any type of ex-post matching procedure that could fictitiously increase the performance of the models. In this chapter, it will be firstly discussed the composition and the general characteristic of the data used to perform the models. It will also be explained how the problem of the imbalance of the dataset and how the hyperparameters and variables analysed in the first chapter of the dissertation are chosen.

## 2.1 SAMPLE ORIGIN AND COMPOSITION

On preliminary analysis, it has been decided to take into consideration various countries in the EU and several different activities given the high number of firms observations without a sufficient amount of data in the Orbis database. Firstly, it has been taken the data of the firms that will go bankrupt respectively in 2019, 2020 and 2021. To make the successive analysis more robust, all the observations with at least one NaN (Not a Number) variable were cancelled. In this way, the data observations for the bankrupt firms in 2019 are 1124, for the bankrupt firms in 2020 are 1282 and for 2021 they are 1464. Table 8 explains how the sample data has been collected. To avoid any potential inconsistency between observation characteristics, all the bankrupt firms were selected and obtained imposing the constraint on the research in the Orbis database shown in table 8. The first 3 constraints define that the bankruptcy sample has collected only data from the European Union for all the available business activity from 2013

to 2020. Imposing constraint number 4, the level of observations with a missing value has been reduced. However, what is important in defining the belonging on the sample of the bankrupt firm for a specific year refers to conditions 5 and 6.

| Search Strategy for 2021 bankruptcy firms | |
|---|---|
| Search Step: | |
| 1. Status | Failed |
| 2. Geographic area | European Union |
| 3. Available balance sheet years | 2013,2014,2015,2016,2017,2018,2019,2020 |
| 4. Updates | From 01/01/2013 : New annual balance sheet |
| 5. Last available balance sheet year | 2020 |
| 6. Updates | From 01/01/2022 : Change of status in bankruptcy proceedings |
| 7. Quoted/Unquoted firms | Unquoted firms |
| Boolean search | 1 and 2 and 3 and 4 and 5 not 6 and 7 |

Table 8: Example of search strategy made for bankrupt firms.

Imposing the former condition, the observations space was reduced to all the bankrupt firms that have as last balance sheet financial year the year prior to the one that the model should predict. For example, firms that will go bankrupt in 2021 should have as last available balance sheet the one of 2020. However, this condition alone does not imply logically collecting only firms that failed in 2021. To do so, another condition should be imposed, the sixth one, that asks the database to exclude all the firms that go into bankruptcy the year following the considered one. This has been done using a Boolean condition different from the previous AND condition that find all of the search terms. Through the NOT condition that eliminates items that contain the specified term, it has been possible to remove all the firms that filed for bankruptcy in the example case from the first January of 2022. Hence, the data collection and definition for the bankruptcy firms has been made in a strict legal term. A company is declared failed only if it filled the bankruptcy procedure in the considered period of time. In this way, it has been possible to define in the best feasible way which time the data refers to. Hence, the results of this dissertation will be on data of firms that failed without any doubt in the prespecified year. From the same theoretical viewpoint, the collection of the nonbankrupt firms has been made. The only practical difference is that condition 6 was imposed logically for the complete period of time. Practically, it was asked to the database to collect only firms that have not suffered any legal status change for the nonbankrupt data sample. This gives the case to infer that all the nonfailed firms are in a healthy status and do not suffer any financial distress.

Another significant point to make a solid-based analysis derives from the fact that both bankrupt and nonbankrupt firms should contain approximately the same number of firms in terms of geographic area and business activity. Hence, the two-sample has been balanced, and the data collected are in equilibrium between them. Theoretically, this implies that the results of the performed models do not depend on the ability to detect some intrinsic and unknown characteristics of a specific market in a specific country. In other words, the fact that the two data samples contain the same country and business activity implies that the analysis depends less on white noise in the observations. Firstly, it has been collected all the observations related to the bankrupt subset and it has been defined specifically the number of firms in all the countries through the ISO country code, distinguishing them for the business activity through the NACE code as shown by table 9.

| BANKRUPTCY FIRMS COMPOSITION BY COUNTRY AND BUSINESS ACTIVITY | SE | IT | BE | ES | FI | FR | NL | LU | SUM |
|---|---|---|---|---|---|---|---|---|---|
| Agriculture, forestry and fishing | 14 | 12 | 10 | 3 | 7 | 0 | 1 | 0 | 47 |
| Mining and quarrying | 1 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 5 |
| Manufacturing | 121 | 240 | 78 | 35 | 25 | 3 | 2 | 0 | 504 |
| Electricity, gas, steam and air conditioning supply | 3 | 4 | 2 | 1 | 0 | 1 | 0 | 0 | 11 |
| Water supply; sewerage, waste management and remediation activities | 2 | 7 | 1 | 0 | 1 | 1 | 0 | 0 | 12 |
| Construction | 235 | 234 | 192 | 33 | 50 | 6 | 1 | 0 | 751 |
| Wholesale and retail trade; repair of motor vehicles and motorcycles | 343 | 221 | 214 | 40 | 38 | 4 | 1 | 1 | 862 |
| Transportation and storage | 89 | 38 | 32 | 7 | 22 | 2 | 0 | 0 | 190 |
| Accommodation and food service activities | 115 | 74 | 123 | 12 | 16 | 5 | 0 | 0 | 345 |
| Information and communication | 47 | 32 | 29 | 3 | 3 | 0 | 0 | 0 | 114 |
| Financial and insurance activities | 22 | 10 | 17 | 1 | 4 | 2 | 1 | 0 | 57 |
| Real estate activities | 42 | 164 | 36 | 7 | 8 | 6 | 0 | 0 | 263 |
| Professional, scientific and technical activities | 158 | 54 | 73 | 6 | 13 | 2 | 3 | 0 | 309 |
| Administrative and support service activities | 91 | 41 | 39 | 6 | 8 | 3 | 0 | 0 | 188 |
| Education | 21 | 6 | 4 | 1 | 3 | 2 | 0 | 0 | 37 |
| Human health and social work activities | 35 | 7 | 8 | 0 | 4 | 0 | 0 | 0 | 54 |
| Arts, entertainment and recreation | 25 | 17 | 7 | 3 | 3 | 0 | 0 | 0 | 55 |
| Other service activities | 31 | 7 | 22 | 4 | 0 | 2 | 0 | 0 | 66 |
| Total sum | 1395 | 1170 | 887 | 164 | 205 | 39 | 9 | 1 | 3870 |

Table 9: Distribution of bankrupt firms by country and business activity.

For the sake of clarity, the ISO country code is a standard defining code for the countries' names. Specifically, there are 8 countries in the bankruptcy sample: Sweden (SE), Italy (IT), Belgium (BE), Spain (ES), Finland (FI), France (FR), Netherlands (NL) and Luxembourg (LU). Instead, the business activities are defined by following the broad structure of NACE Rev. 2

made by Eurostat[6]. Due to the fact that not even a firm went bankrupt on public administration and defence, compulsory social security, activities of households as employers, undifferentiated goods and services-producing activities of households for own use and activities of extraterritorial organizations and bodies; these activities are not displayed in tables 9 and 10.

| NONBANKRUPTCY FIRMS COMPOSITION BY COUNTRY AND BUSINESS ACTIVITY | SE | IT | BE | ES | FI | FR | NL | LU | SUM |
|---|---|---|---|---|---|---|---|---|---|
| Agriculture, forestry and fishing | 48 | 60 | 35 | 5 | 33 | 0 | 5 | 0 | 186 |
| Mining and quarrying | 5 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 15 |
| Manufacturing | 570 | 1125 | 390 | 61 | 111 | 13 | 10 | 0 | 2280 |
| Electricity, gas, steam and air conditioning supply | 15 | 20 | 10 | 3 | 0 | 5 | 0 | 0 | 53 |
| Water supply; sewerage, waste management and remediation activities | 10 | 35 | 5 | 0 | 5 | 5 | 0 | 0 | 60 |
| Construction | 678 | 813 | 704 | 20 | 210 | 30 | 5 | 0 | 2460 |
| Wholesale and retail trade; repair of motor vehicles and motorcycles | 1018 | 711 | 787 | 139 | 103 | 13 | 5 | 5 | 2781 |
| Transportation and storage | 370 | 175 | 145 | 10 | 110 | 10 | 0 | 0 | 820 |
| Accommodation and food service activities | 525 | 340 | 615 | 5 | 77 | 24 | 0 | 0 | 1586 |
| Information and communication | 235 | 140 | 143 | 17 | 13 | 0 | 0 | 0 | 548 |
| Financial and insurance activities | 110 | 50 | 85 | 0 | 20 | 10 | 5 | 0 | 280 |
| Real estate activities | 197 | 578 | 180 | 14 | 40 | 23 | 0 | 0 | 1032 |
| Professional, scientific and technical activities | 707 | 270 | 345 | 43 | 65 | 10 | 15 | 0 | 1455 |
| Administrative and support service activities | 403 | 175 | 145 | 15 | 40 | 15 | 0 | 0 | 793 |
| Education | 103 | 30 | 20 | 3 | 13 | 8 | 0 | 0 | 177 |
| Human health and social work activities | 145 | 33 | 43 | 0 | 17 | 0 | 0 | 0 | 238 |
| Arts, entertainment and recreation | 113 | 84 | 35 | 8 | 14 | 0 | 0 | 0 | 254 |
| Other service activities | 114 | 31 | 103 | 1 | 0 | 9 | 0 | 0 | 258 |
| Total sum | 5366 | 4680 | 3790 | 344 | 871 | 175 | 45 | 5 | 15276 |

Table 10: Distribution of nonbankrupt firms by country and business activity.

Secondly, it has been randomly collected all the observations for the nonbankruptcy firms, in a way that for each combination of country and business activity, a certain number of firms has been taken. It has been arbitrary decided to maintain a ratio of one to five between the two samples. This means that for one firm that goes bankrupt there should be five firms that continue to operate as shown in table 10. Divergences from this ratio are due to the fact that all the observations with at least one NaN value have been removed. Furthermore, the ratio was decided to maintain a feasible number of observations, given that to perform quite complicated machine learning models a high computation power is needed. An increase in the observations

[6] The Statistical Classification of Economic Activities in the European Community (NACE) is the industry-standard classification system used in the European Union. The NACE Rev. 2 regulation is the current version and it was adopted in December 2006. The scheme used to define precisely the various activities come from page 59 of the following pdf: https://ec.europa.eu/eurostat/documents/3859598/5902521/KS-RA-07-015-EN.PDF.

could lead to an exponential increase in the time required for the tuning of the hyperparameters and for the training of the models. Table 9 suggests some important hints about how bankruptcy firms are distributed in the sample of EU countries. The ones that have suffered more from the Covid crisis are Sweden, Italy and Belgium. A minor impact was the one that hit France, Spain and Finland, instead, Netherlands and Luxembourg have a neglectable effect on their economies. The data give us a clear view of the difference in the bankruptcy phenomenon between countries. It defined where the crisis has struck strongly and where the economic framework is more prone to declare bankruptcy instead of other forms of financial restructuring. For example, in Italy, the crisis and the lockdown have caused a GDP decrease of 8,9% in 2020[7]. That explains why there is such a high number of Italian firms in the EU sample in 2020. In the same table, it can be retrieved other information evaluating the data on sectors. Those who drive more this negative path are the manufacturing, the construction and the wholesale and retail trade and the repair of motor vehicles and motorcycles sectors. Only a few of them have not suffered a significant change as the electricity, gas, steam and air conditioning supply, the water supply, sewerage and waste management and remediation activities. All the other sectors with different degrees have a moderate rate of bankruptcy. This point of view shows the significance of some sectors that are the vital points of the European economic system like the retail and construction ones. On the other hand, the data points out the grade of chaoticity of these markets that are maybe for construction more inclined to high rate of changes of firms. The ones with fewer bankrupt firms, instead, are those with a more mature and stable market and with a smaller number of firms and so markets with less competition within.

| BANKRUPTCY FIRMS COMPOSITION BY COUNTRY AND YEAR OF FAILURE | 2021 | 2020 | 2019 | SUM |
|---|---|---|---|---|
| Sweden (SW) | 382 | 615 | 398 | 1395 |
| Italy (IT) | 484 | 258 | 428 | 1170 |
| Belgium (BE) | 390 | 300 | 197 | 887 |
| Spain (ES) | 82 | 33 | 49 | 164 |
| Finland (FI) | 96 | 68 | 41 | 205 |
| France (FR) | 28 | 7 | 4 | 39 |
| Netherlands (NL) | 2 | 1 | 6 | 9 |
| Luxembourg (LU) | 0 | 0 | 1 | 1 |
| Total sum | 1464 | 1282 | 1124 | 3870 |

Table 11: Distribution of bankrupt firms by country and year of failure.

[7] Italy was not the worst one in term of GDP decreases during the Covid Crisis: Sweden (-2,9%), Belgium (-5,7%), Spain (-10,8%), Finland (-2,8%), France (-7,9%), Netherlands (-8,2%), Luxembourg (-1,8%).

Table 11 shows the effect of the Covid crisis on the 8 countries in the sample. All of them, except for the Netherlands and Luxembourg, have seen an increase of bankruptcy cases from 2019. Most of the time the spike in the number of bankrupt cases is in 2021, apart from Sweden. It is interesting to see that even if for example Italy had an increase in GDP of 6,5% in 2021 and the relaxing of Covid containment measures, in Italy but also in Belgium, Finland and Spain there was a huge increase in bankruptcy cases from the previous year. The distribution of the bankruptcy cases does not affect the ability to capture them by the proposed model, hence, these types of reasoning are only companions and introductions to better understand the sample. Table 12 is the last chart that shows something about the sample composition. It gives a viewpoint on the different business activities and the change in the cases of bankruptcy year by year. As shown in table 11 by countries, also watching through the NACE code the same pattern comes over again. Usually, the greater number of cases is not as expected in 2020 but in the following year. The only three sectors that suffer more in 2020 maybe have this different habit because of their natural need for people that are free to move and that are not in lockdown. For example, the transportation and storage given to Covid have been in a sort of stasis due to the fact that the big trade with China, the EU and US was frozen.

| BANKRUPTCY FIRMS COMPOSITION BY BUSINESS ACTIVITY AND YEAR OF FAILURE | 2021 | 2020 | 2019 | SUM |
|---|---|---|---|---|
| Agriculture, forestry and fishing | 18 | 14 | 15 | 47 |
| Mining and quarrying | 2 | 0 | 3 | 5 |
| Manufacturing | 221 | 141 | 142 | 504 |
| Electricity, gas, steam and air conditioning supply | 5 | 4 | 2 | 11 |
| Water supply; sewerage, waste management and remediation activities | 5 | 3 | 4 | 12 |
| Construction | 281 | 230 | 240 | 751 |
| Wholesale and retail trade; repair of motor vehicles and motorcycles | 319 | 299 | 244 | 862 |
| Transportation and storage | 61 | 79 | 50 | 190 |
| Accommodation and food service activities | 135 | 117 | 93 | 345 |
| Information and communication | 43 | 31 | 40 | 114 |
| Financial and insurance activities | 23 | 20 | 14 | 57 |
| Real estate activities | 94 | 73 | 96 | 263 |
| Professional, scientific and technical activities | 110 | 108 | 91 | 309 |
| Administrative and support service activities | 67 | 92 | 29 | 188 |
| Education | 11 | 14 | 12 | 37 |
| Human health and social work activities | 16 | 18 | 20 | 54 |
| Arts, entertainment and recreation | 29 | 16 | 10 | 55 |
| Other service activities | 24 | 23 | 19 | 66 |
| Total sum | 1464 | 1282 | 1124 | 3870 |

Table 12: Distribution of bankrupt firms by business activity and year of failure.

## 2.2 DESCRIPTIVE ANALYSIS AND DATA PREPROCESSING

The aim of this part of the dissertation is to provide the last feature description of the explanatory variables and to analyse some standard ways to manipulate the data. It has to be specified that to make a coherent analysis of the two-sample group it has been decided to show the variable until 2018. In other words, given that not all firms in the bankruptcy subgroup fail in 2021, but 1124 go bankrupt in 2019 and 1282 in 2020, to compare the two groups in the same years, both have been restricted. To evaluate the models, it has been decided to take only three explanatory variables. This choice was made to balance the problem of lack of data derived from the Orbis database and mainly to follow the idea of parsimony and economy of the models. A good model is the one that given a feasible amount of data has the best results. The ability of a model to perform well with small data is a characteristic that can be attractive for real bankrupt problems. For sure, a model that performed well with small variables can only increase its score with more variables. It has been decided to use the approach followed by Brédart (2014) which showed a prediction accuracy of his models near 84% one year before the failure. His analysis was based on 3 categories of explanatory variables based on:

- **Profitability** through the return on assets ratio, $ROA_t = Net\ Income\ _t / Total\ Asset_t$. The assumption is that a company whose profits decrease over time is expected to have a high probability of facing financial difficulties. In fact, the profitability generated by the firm is one of the primary criteria for the granting of credit by financial institutions.
- **Liquidity** through the current ratio, $CR_t = Current\ Assets_t / Current\ Liabilities_t$. Intuitively, it is assumed that higher levels of liquidity will influence positively the survival odds of businesses. Indeed, companies in distress usually have low liquidity and concerns to meet their commitments. In this situation, causing more financial troubles, banks may tighten credit conditions to prevent future risk of failure. A liquidity ratio is an inevitable financial indicator as it assesses the level of the funds available to deal with various short-term situations.
- **Solvency** through the solvency ratio, $SR_t = Equity_t / Total\ Assets_t$. The creditworthiness of the company is the ability to repay its debts and the financial burden. In this field, its financial structure may play a role in the risk of bankruptcy. When it encounters financial troubles, causing debts and financial burdens, the probability of default of the company increases dramatically.

| Explanatory variables | count | mean | std | min | 0,25 | 0,5 | 0,75 | max |
|---|---|---|---|---|---|---|---|---|
| ROA 2018 | 3870 | -1,752 | 46,603 | -2257,000 | -0,238 | -0,030 | 0,016 | 76,667 |
| ROA 2017 | 3870 | -0,283 | 3,392 | -118,167 | -0,139 | -0,006 | 0,032 | 29,000 |
| ROA 2016 | 3870 | -0,136 | 2,584 | -142,400 | -0,092 | 0,000 | 0,045 | 23,000 |
| ROA 2015 | 3870 | -0,483 | 23,865 | -1481,500 | -0,070 | 0,001 | 0,053 | 38,125 |
| ROA 2014 | 3870 | -0,076 | 0,943 | -36,200 | -0,065 | 0,002 | 0,053 | 15,526 |
| ROA 2013 | 3870 | -0,642 | 32,315 | -2006,667 | -0,068 | 0,000 | 0,052 | 5,581 |
| CR 2018 | 3870 | 2,806 | 25,156 | 0,000 | 0,466 | 1,000 | 1,555 | 1252,000 |
| CR 2017 | 3870 | 2,572 | 17,873 | -2,667 | 0,587 | 1,080 | 1,693 | 666,667 |
| CR 2016 | 3870 | 2,435 | 20,404 | 0,000 | 0,684 | 1,119 | 1,761 | 1198,333 |
| CR 2015 | 3870 | 3,890 | 107,526 | -0,085 | 0,751 | 1,160 | 1,798 | 6673,333 |
| CR 2014 | 3870 | 2,939 | 34,293 | 0,000 | 0,765 | 1,166 | 1,838 | 2027,273 |
| CR 2013 | 3870 | 3,620 | 66,248 | -1,377 | 0,793 | 1,163 | 1,823 | 4045,167 |
| SR 2018 | 3870 | -3,437 | 50,147 | -2131,500 | -0,458 | 0,055 | 0,258 | 1,595 |
| SR 2017 | 3870 | -2,126 | 49,960 | -2187,500 | -0,152 | 0,093 | 0,314 | 1,750 |
| SR 2016 | 3870 | -1,549 | 56,393 | -3409,000 | -0,043 | 0,117 | 0,333 | 1,571 |
| SR 2015 | 3870 | -1,021 | 29,987 | -1756,500 | 0,007 | 0,132 | 0,342 | 1,000 |
| SR 2014 | 3870 | -0,267 | 4,478 | -153,250 | 0,016 | 0,140 | 0,358 | 1,000 |
| SR 2013 | 3870 | -0,506 | 21,357 | -1288,667 | 0,022 | 0,143 | 0,363 | 1,545 |

Table 13: Descriptive records of bankrupt firms.

Tables 13 and 14 show the descriptive statistic based on the two-sample. These tables were retrieved using the python function from the pandas library called "*.describe*". It calculates some statistical data like percentile, mean and std of the explanatory variables. Comparing the two tables it can be pointed out that the mean values of all the independent variables are strictly higher for the nonbankrupt firms compare to the bankrupt ones. This result validates the Brédart (2014) statement: the higher the ratios of a firm, the lower the probability that the firm will go bankrupt. Concerning the std of the two subsets, the trend for the ROA and SR exhibits a much lower value for the nonbankruptcy sample, instead, this is reversely considering the CR. This means that data for the former ratios in the nonbankruptcy group are clustered near the mean. Instead, in the bankruptcy sample data are not grouped near the average for those two ratios. This means that the ROA and SR have a high discriminant value. On the contrary, such high values of std for the CR in the nonbankruptcy group suggest that not such high discern ability due to the high volatility of that ratio should be expected as shown in table 14. The other values define the percentile and the median of the distribution of the explanatory variables. The differences are remarkable for all the ratios, this means that the distributions of the ratios of the two subsamples are not too much overlapped with each other. In other words, there is class separability between the two categories. However, this distance is less considerable on CR due to the high variance. This suggests, again, that in the univariate discriminant analysis it will be found performance values that are quite worse for this ratio compared to the others. The last elements are the min and the max values that for completeness show the two extreme values

for both distributions. Looking at these two values it is clear that at some point the distributions of the ratios overlapped each other and so the distributions at some given values intersect each other.

| Explanatory variables | count | mean | std | min | 0,25 | 0,5 | 0,75 | max |
|---|---|---|---|---|---|---|---|---|
| ROA 2018 | 15276 | 0,067 | 0,132 | -3,822 | 0,013 | 0,052 | 0,110 | 1,930 |
| ROA 2017 | 15276 | 0,070 | 0,284 | -7,590 | 0,013 | 0,051 | 0,108 | 25,059 |
| ROA 2016 | 15276 | 0,062 | 0,191 | -14,973 | 0,012 | 0,050 | 0,106 | 2,787 |
| ROA 2015 | 15276 | 0,056 | 0,293 | -20,670 | 0,009 | 0,047 | 0,103 | 2,823 |
| ROA 2014 | 15276 | 0,053 | 0,216 | -10,500 | 0,007 | 0,044 | 0,100 | 6,693 |
| ROA 2013 | 15276 | 0,047 | 0,275 | -9,585 | 0,004 | 0,039 | 0,095 | 19,500 |
| CR 2018 | 15276 | 3,718 | 50,970 | -8,608 | 1,117 | 1,485 | 2,177 | 4241,000 |
| CR 2017 | 15276 | 9,342 | 286,540 | -0,351 | 1,119 | 1,495 | 2,185 | 23260,000 |
| CR 2016 | 15276 | 16,475 | 904,720 | -6,941 | 1,112 | 1,478 | 2,200 | 107344,000 |
| CR 2015 | 15276 | 17,967 | 623,142 | -6,536 | 1,098 | 1,464 | 2,174 | 50522,000 |
| CR 2014 | 15276 | 12,725 | 417,583 | -358,000 | 1,080 | 1,443 | 2,142 | 33133,833 |
| CR 2013 | 15276 | 11,659 | 368,430 | -46,222 | 1,068 | 1,428 | 2,118 | 26012,000 |
| SR 2018 | 15276 | 0,376 | 0,687 | -58,678 | 0,218 | 0,381 | 0,555 | 1,000 |
| SR 2017 | 15276 | 0,373 | 0,675 | -55,373 | 0,215 | 0,379 | 0,554 | 0,999 |
| SR 2016 | 15276 | 0,328 | 3,702 | -351,084 | 0,212 | 0,377 | 0,558 | 1,000 |
| SR 2015 | 15276 | 0,330 | 2,545 | -195,718 | 0,204 | 0,371 | 0,554 | 3,977 |
| SR 2014 | 15276 | 0,338 | 1,793 | -188,252 | 0,192 | 0,360 | 0,546 | 1,000 |
| SR 2013 | 15276 | 0,396 | 7,972 | -218,312 | 0,180 | 0,349 | 0,535 | 959,300 |

Table 14: Descriptive records of bankrupt firms.

It has been decided to not take into account any matching procedure, to avoid any fictitious increase in the performance of the model. As explained by King and Nielsen (2019): “propensity score matching (PSM) often accomplishes the opposite of its intended goal, thus increasing imbalance, inefficiency, model dependence, and bias”. When it has to collect the sample of the nonbankrupt firms it is not fair to do that in a way to better distinguish the two categories. It is common to believe that a random sample collection is better than the PSM procedure. In other terms, any type of match on the performance in profitability, liquidity and solvency is purposely avoided to let its implications and drivers be directly embedded in prediction models, but also because the PSM procedure can be done only ex-post. This means that in reality it is something that cannot be done ex-ante without knowing the status of the observations and so it would not be coherent with the purpose of this dissertation: “make a comparison of models in the most possible real environment”. Following the purpose of evaluating the models in the realest environment possible, it has been decided to perform ex-ante two standard data manipulation procedures that can be done without any knowledge of the category of the observation. Usually, to scale numerical data prior to the modelling phase, data analysts used two popular techniques:

- Normalization was performed using the scikit-learn function on python called: "*preprocessing.Normalizer"*. It scales each explanatory variable alone to the range between 0 and 1. Given that the scale and the distribution of the data could be drawn from domains that can be different for each variable and that explanatory variables may have different units or magnitude that, in turn, may mean that the variables have different scales. These dissimilarities in the scales may increase the complexity of the problem being modelled. For example, if there are large input values, the model trained can result in learning large weight values, making it unstable. This means that it could suffer from poor performance during the testing phase. Of course, the difference in scale does not influence all machine learning algorithms in the same way. The algorithms that fit a model using a weighted sum of input variables are the major affected ones, like Linear and Logistic regression and Artificial Neural Networks. Also, algorithms that use distance measures between examples are affected, such as k-Nearest Neighbours and Support Vector Machines. Instead, some algorithms are unaffected by the scale of numerical input variables, most notably decision trees and ensembles of trees, like Random Forest. Rather for boosting methods like XGBoost and CatBoost a target variable with a large spread of values may result in large error gradient values causing weight values to change dramatically, making the learning process unstable.

- Standardization was performed using the scikit-learn function on python called: "*preprocessing.StandardScaler*". It scales each explanatory variable one by one subtracting the mean and dividing by the standard deviation in a manner to shift the distribution to have a standard Gaussian distribution with zero mean and a standard deviation of one. The act of subtracting the mean from the data is referred to as centering, while dividing by the standard deviation is called scaling. Due to that, the method is known as "center scaling". Standardization has to be done, like normalization, and it is required in some machine learning algorithms. For instance, many estimators used in the objective function of a learning algorithm, such as the RBF kernel, the Support Vector Machines and the L1 and L2 regularizers of the linear models, assume that all features are cantered around 0 and have variance in the same order. If a feature has a variance that has orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

## 2.3 PRINCIPAL COMPONENT ANALYSIS AND CORRELATION MATRIX

To complete the descriptive examination of the explanatory variables and define the statistical relations that occur between themselves over the various years, it may be useful to compare the correlation matrix between the two-sample group. This is particularly important to define if the indexes included in the models carried out the same underlying information. In other words, if the ratios have the same behaviour and share some relationship. After a certain correlation threshold multicollinearity might occur. It can lead to skewed or misleading results when a researcher or analyst attempts to determine how well each independent variable can be used most effectively to predict or understand the dependent variable in a statistical model. Lieberman and Morris (2014) analyse the effect of multicollinearity on classification problems like the bankruptcy prediction one. Their result depends on the goals of the analysis, if prediction accuracy is the aim of the research, then multicollinearity is not relevant. On the other hand, if attention does not regard model prediction accuracy, but the allocation of variable importance within the model, then, of course, multicollinearity tends to confound such judgment. If interest is in both goals, the researcher may need to contemplate their relative importance. However, it seems that what appear to be general warnings regarding multicollinearity in classification problems should at least be more precisely expanded in consideration of these alternate modelling goals.

To be compliant with the literature this dissertation takes into account the problem of multicollinearity through the Principal Component Analysis (PCA) procedure and compares if this approach can affectively increase the stability and performances of the models. In detail, it is a linear dimensionality reduction applying a singular value decomposition of the data to draw them into a lower-dimensional space increasing interpretability but at the same time minimizing information loss. To recall a little bit of linear algebra, the singular value decomposition (SVD) is a factorization of a real or complex matrix. It generalizes the eigendecomposition of a square normal matrix with an orthonormal eigenbasis matrix. In simpler terms, SVD is a matrix decomposition method that reduces a matrix to its building parts in order to make certain subsequent matrix calculations simpler. In this dissertation, the SVD uses the LAPACK implementation of the full SVD or a randomized truncated SVD by the method of Halko et al. (2009) to retrieve explanatory variables having fewer correlations between them. Through the scikit-learn python function: "*.PCA*", the SVD method has been performed. The results, however, are similar to what Lieberman and Morris (2014) found, PCA does not increase and in some years and for certain models on the contrary slightly decreases the accuracy, the precision and the recall, as shown in appendix 1. However, as will be explained in detail in the

tuning of the hyperparameter of the model section, some of them have some variables already made some types of regularizations. Given these reasons, it has been decided not to carry out such technique.

Even if multicollinearity does not reduce the predictive power or reliability of the model as a whole, to make a complete description of the sample data the analysis of the correlation matrices will be produced. To be coherent with the following section and the following chapter 3, it has been decided to show the result of the correlation matrix of the dataset composed of the bankrupt firms in 2021 and the complete sample of nonbankrupt ones. Nevertheless, for the sake of completeness, all the other correlation matrices will be shown in appendix 2.

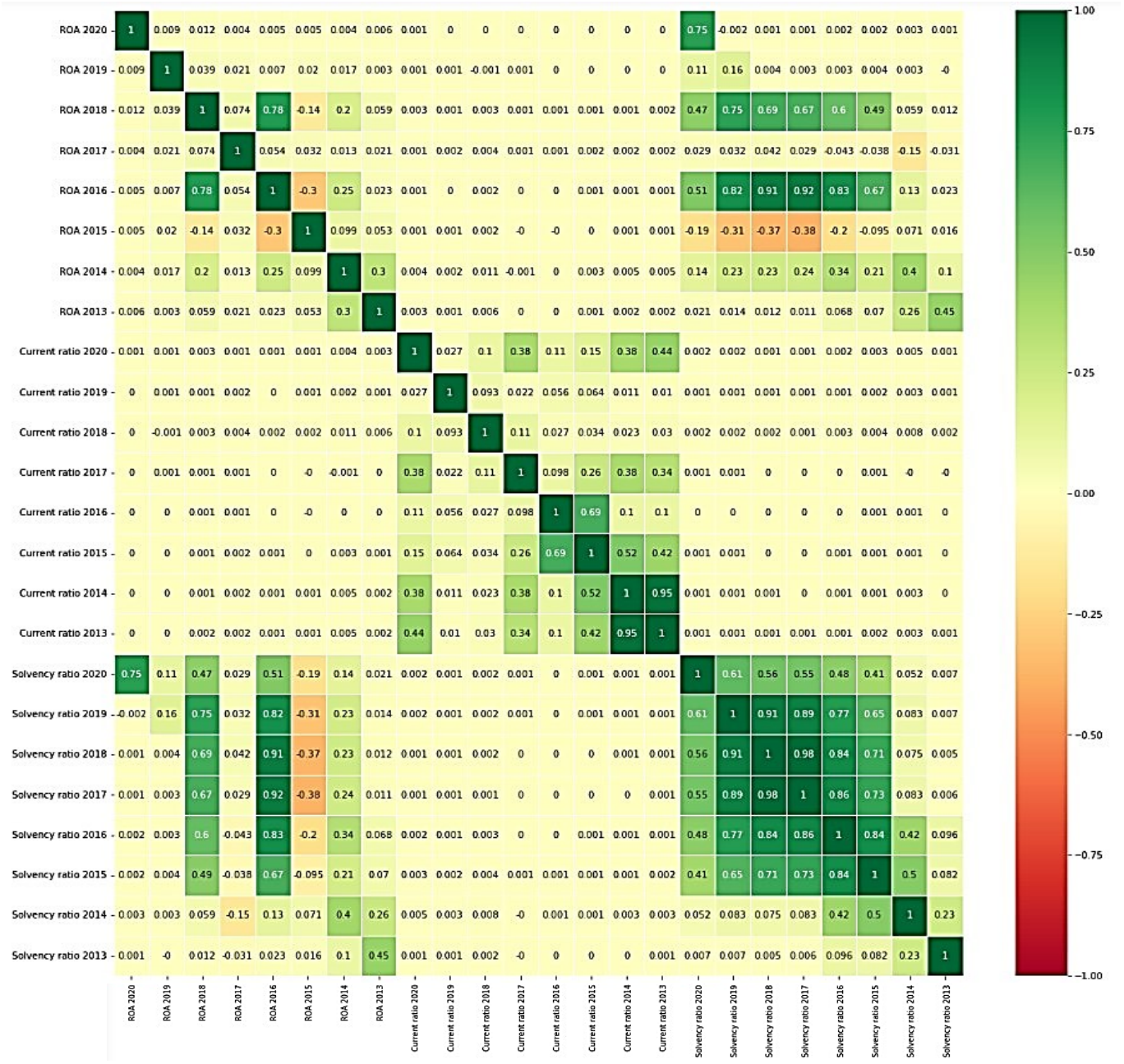


Figure 8: Correlation Matrix for dataset of firms in 2021.

Figure 8 is the correlation matrix of such dataset and it is computed using the pandas python function: "*.corr*". As with all the correlation matrices, it is sufficient to observe the upper or the lower triangle of the matrix. Given the fact that the explanatory variables have been standardized and normalized, the correlation values can flow from -1 to +1. Recalling also that the population Pearson correlation coefficient is described as $\rho(X,Y) = covariance\ (X,Y)/\sigma_X\sigma_Y$ and $covariance(X,Y) = \sum(x-\bar{x})(y-\bar{y})/n$. Hence, given that the correlation of a variable to himself is the variance and the std of a variable to the std of the same variable is the variance, the diagonal of figure 8 should report $\rho = 1$. Figure 8 shows the normalized correlations in the bankrupt sample are in general quite low. Watching the correlation between ROA variables, by taking into consideration different years, they are very low recalling that they are the correlation values of the same lagged ratio. Instead, the correlation between ROA and CR is near 0, this is quite useful for adding new information for the classification problem and mitigating the quite high correlation between ROA and SR. Unsurprisingly, the correlation values of CR compared in different years have a higher value. Finally, as expected the correlation between ROA and SR is high because these two ratios share the same denominator "$Total\ Assets_t$". In general figure 8 draws a picture of explanatory variables that are quite not so correlated except for some cases. Hence, the approach of taking three consecutive years of ratios will not be a severe problem in terms of multicollinearity.

## 2.4 UNIVARIATE LOGISTIC REGRESSION

This section reviews how the univariate logistic regressions were carried out. To be consistent with the discussion of the results in this and in the following chapter, the univariate results showed are the ones related to the failed firms in 2021 and the complete sample of nonbankrupt firms. In other words, the ratio results will be relative to eight years from 2020 to 2013. This choice is made to not add too many tables that can be confusing, however, the results of the dataset of the 2019 and 2020 groups will be included in appendix 3 and are similar to the one of 2021. To be clear, the univariate logistic regression is the performing of a logistic regression in which the dichotomous dependent variable "Bankrupt or NonBankrupt" that can take a value of 0 for failed firms and 1 otherwise is explained by only one ratio at a time. Of course, the evaluation of the performance is made on a separate sample data, as previously explained the sample is divided into two datasets: the training and the testing ones. In the first one, the model learns how to fit the data, in the second one the trained model is applied and the result is retrieved. In this dissertation, the training sample is composed of 80% of the total sample and therefore the testing one is composed of 20%. Crucial are the measures of efficiency and

performance. Given the imbalanced sample classification problem, some corrections to what has been shown as performance measures used in literature have to be made. However, all the measures involve four concepts:

- True Positive (TP) that is the number of bankrupt firms correctly classified as failed.
- True Negative (TN) that is the number of nonbankrupt firms correctly classifies as solvent.
- False Positive (FP) that is the number of nonbankrupt firms wrongly classified as failed, also known as Type II error in literature.
- False Negative (FN) that is the number of bankrupt firms wrongly classified as solvent, also known as Type I error in literature.

All these basic concepts can be immediately seen from the Confusion Matrix or Error Matrix as shown in table 2 in the first chapter of this dissertation. In the interest of clarity, in this analysis the first row refers to the bankrupt firm also called class 0, the sum of TP and FN gives the total number of failed firms in the sample. On the contrary, the second row shows the comprehensive number of nonfailed firms also recall as class 1 summing TN and FP. The diagonal gives the correctly classified observations, and so the better model is the one with the higher numbers in the diagonal. However, from these concepts, a lot of more specific scores can be built. The famous ones are:

- **Accuracy** = (TP+TN)/ (TP+TN+FP+FN).

It is the percentage of correctly predicted firms to the total number of firms in the sample. This is the main score used in the literature on the bankruptcy prediction field. However, this is a partial and misleading score that depends on the balancing of the sample. For example, a model that is tested on a sample composed of 90 nonfailed firms and only 10 failed ones that score 90% of accuracy it is not necessarily a good algorithm. This is because even a model that correctly classifies all the nonfailed firms but no one of the failed ones achieves 90% of accuracy level.

- **AUC-ROC curve**

The Area Under the Curve and Receiver Operating Characteristics is a graphical plot that illustrates the performance of a binary classifier as its discrimination threshold is varied. It is built by plotting the fraction of TP out of the total positives that is equal to the True Positive Rate (TPR) versus the fraction of FP out of the total negatives that is equal to the False Positive Rate (FPR), at various threshold settings. TPR is also known as sensitivity, and FPR is one

minus the specificity. The AUC-ROC curve will give an immediate idea of the general performance of the model as it will be shown in chapter 3, but as the accuracy score, it is incapable to give precise information in case of unbalanced data. In those cases, an AUC-ROC curve could look pretty decent while misclassifying most or all of the minority class.

- **Precision** (Specificity) = TP / (TP+FP).

It is the percentage of correctly predicted positive observations over the total predicted positive. In other terms, it is the ability of the classifier not to label as positive the observations that are actually negative. In the scikit-learn library, there is the possibility to compute the unweighted or the weighted precision. The former makes an average between the precision score obtained from class 0 and class 1. The second one, instead, makes the same but weighting the average. Due to the fact that bankruptcy prediction is an unbalanced problem and that the prediction of class 0, the failed one, is more important than the others, it has been decided not to use the weighted value. Moreover, in python, there is the possibility to compute the value for the specific classes. So as shown in table 15 and then in chapter 3, the value of each class precision performance will be specified.

- **Recall** (Sensitivity) = TP / (TP+FN)

It is the percentage of correctly predicted instances to all the observations in that class. The recall value states the ability of a classifier to find all the positive samples. In the bankrupt prediction framework, it is the most important score to verify if a model can well predict the failure event. As the precision one, it will be not used the weighted value but the unweighted one and will be specified the score for each type of class.

- **F1** score = 2*(Recall *Precision) / (Recall + Precision) = TP / (TP + 0.5(FP +FN)

It is the harmonic average of Precision and Recall. The relative contribution of precision and recall to the F1 score are equal, this means that F1 assumes that Type I and II have the same impact. Of course, this is not the case for bankruptcy prediction problems but it can be a valid score to understand the general performances. As the precision and recall score, the unweighted value will be used.

- **F2** score = TP / (TP + 0.2FP +0.8FN)

It is similar to the F1 score but it gives more attention to minimizing FN than minimizing FP. It is more important to identify bankrupt firms than nonbankrupt, so FN should have more weight compared to FP. However, it is not defined in literature the optimal weight to give for recall compared to precision. In this dissertation, it has been assumed that FN has two times the

importance of FP. Hence, the F2 score has been chosen but it could have used the F3 or F4 scores, which give three or four times higher weights for the recall score compared to the precision one.

Confusion Matrix:
[[ 61 272]
[ 5 3010]]

Classification Report:

| | precision | recall | f1-score | f2-score | support |
|---|---|---|---|---|---|
| 0 | 0.92 | 0.18 | 0.31 | 0.21 | 333 |
| 1 | 0.92 | 1.00 | 0.96 | 0.98 | 3015 |
| accuracy | | | 0.92 | | 3348 |
| macro avg | 0.92 | 0.59 | 0.63 | 0.59 | 3348 |
| weighted avg | 0.92 | 0.92 | 0.89 | 0.90 | 3348 |

Table 15: Confusion matrix and classification report of ROA 2020.

In favour of clarity, it is worth providing an example of how the score measures that have just been seen can be calculated practically. Table 15 shows the confusion matrix and the classification report of the univariate logistic regression of the ROA 2020 ratio. Looking at the confusion matrix it can clearly see that the number of nonbankrupt firms correctly classified (TN) is 3010 while only 61 of the bankrupt firms are correctly classified (TP). That means that the ratio wrongly classifies as default only 5 solvent firms (FP or Type II error) whereas the number of default firms incorrectly classifies as solvent is 272 (FN or Type I error). As explained above, the correct classified observations are on the diagonal and the classification report "0" defines the bankrupt firms and "1" the nonfailed ones. The total number of the observations is summarized in the last column (support), in fact, 333 are the bankrupt firms and 3015 the nonbankrupt ones. The accuracy score to measure the overall performance is equal to (61+3010)/3348 ≈ 92%. This means that ROA 2020 alone can classify correctly 92% of the firms. Watching figure 9 the same positive conclusion can be derived from the AUC-ROC curve having a quite high value of 89%. However, the results are not so good, as explained by these partial scores. The precision value for the bankrupt firms is computed as 61/(61+5) ≈ 92%, and for the nonbankruptcy firms as 3010/(3010+272)≈ 92%. For obvious causes, the unweighted and weighted are 92%. This means that when the model finds a bankrupt case, it is quite sure that it is correct, in fact only 8% of the time it is incorrect. The recall value for the bankrupt firms is computed as 61/(61+272) ≈ 18%, and for the nonbankruptcy firms as 3010/(5+3010) ≈ 100%.

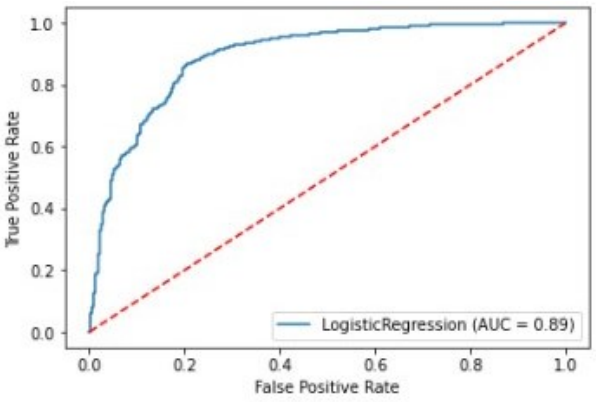


Figure 9: AUC-ROC curve ROA 2020.

The unweighted sum is calculated as (0.18+1)/2 ≈ 59% and the weighted one as (0.18*333+1*3015)/3348 ≈ 92%. Hence, the model quite always successfully classifies the nonbankrupt firms but 82% of the time it is not able to detect the bankrupt ones. This is quite a constant pattern in very imbalanced datasets, this is due to the fact that given the high imbalance of the sample, the model cannot learn adequately how to discern the bankrupt cases. This problem will be approached in the next section through the SMOTE and Tomek link approach. The F1 score for the bankrupt firms is simply define as 2*(0.18*0.92)/(0.18+0.92) ≈ 31% and for the nonbankrupt ones as 2*(0.92*1)/(0.92+1) ≈ 96%. And as the usual way, the unweighted and weighted values are 63% and 89%. This score will be useful to detect the more balanced model from the general viewpoint. Finally, the F2 score for the bankruptcy firms is defined as 61/(61+0.2*5+0.8*272) ≈ 21% and for the nonbankruptcy as 3010/(3010+0.2*272+0.8*5) ≈ 98%. The unweighted and weighted values are 59% and 90%. The higher the weight gives to the recall value, the more the F2 score becomes similar to the recall value of the specific class. This score will give a more unbalanced statistic for the models compared to the simple F1 score and will be one of the crucial scores to choose the best model. To conclude, given these reasons, it has been decided to show in this section and in chapter 3, for the general scores, only the unweighted ones also plotting a separated table to specifically analyse the precision and recall for each class type. Tables 16 and 17 summarize the univariate logistic regression for the 24 ratios.

| Ratio | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| ROA 2020 | 0.917264 | 0.920683 | 0.590762 | 0.630888 | 0.599568 |
| ROA 2019 | 0.907407 | 0.78957 | 0.565255 | 0.589801 | 0.56973 |
| ROA 2018 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| ROA 2017 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| ROA 2016 | 0.901135 | 0.763398 | 0.50701 | 0.48862 | 0.498289 |
| ROA 2015 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| ROA 2014 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| ROA 2013 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| CR 2020 | 0.898148 | 0.533723 | 0.501345 | 0.478951 | 0.491672 |
| CR 2019 | 0.899642 | 0.450224 | 0.499502 | 0.473585 | 0.488802 |
| CR 2018 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| CR 2017 | 0.897551 | 0.45012 | 0.498342 | 0.473005 | 0.487888 |
| CR 2016 | 0.900239 | 0.450254 | 0.499834 | 0.47375 | 0.489063 |
| CR 2015 | 0.899044 | 0.450194 | 0.499171 | 0.473419 | 0.488541 |
| CR 2014 | 0.898447 | 0.450165 | 0.498839 | 0.473254 | 0.48828 |
| CR 2013 | 0.896655 | 0.450075 | 0.497844 | 0.472756 | 0.487496 |
| SR 2020 | 0.918757 | 0.943494 | 0.594263 | 0.636697 | 0.603682 |
| SR 2019 | 0.910394 | 0.881151 | 0.558899 | 0.581514 | 0.562068 |
| SR 2018 | 0.90233 | 0.888698 | 0.510345 | 0.4948 | 0.502345 |
| SR 2017 | 0.900836 | 0.950403 | 0.501502 | 0.476902 | 0.491101 |
| SR 2016 | 0.901135 | 0.825658 | 0.504339 | 0.482876 | 0.494772 |
| SR 2015 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| SR 2014 | 0.900538 | 0.450269 | 0.5 | 0.473833 | 0.489194 |
| SR 2013 | 0.900239 | 0.450254 | 0.499834 | 0.47375 | 0.489063 |

Table 16: Unweighted Accuracy, Precision, Recall, F1 and F2 scores for all ratios.

The first one shows the general unweighted scores. Not surprising is the fact that the scores decrease in their performances moving away from the bankruptcy year apart from the accuracy one that, for the reasons previously analysed, is poor in detecting differences among the predictors. However, with some exceptions, all the scores decrease over time. The ROA ratio in predicting failure of 2021 suffers from a huge decrease in performance switching from 2019 to 2018, but for all the other years it is able to maintain a stable value in the scores. The CR trend is quite similar but the performance scores are quite inferior compared to the ROA ones, particularly for the first two years of the recall, precision, F1 and F2 scores. Finally, the SR is the superior one, it performs better than the ROA ratio until 2015. Its supremacy is due to the little higher ability to correctly distinguish the two categories probably for the better separability of the classes in the data with the SR.

Table 17 specifically analyses the precision and recall scores for each of the two classes. All the ratios show extraordinary ability to predict the nonbankrupt firms, with values that are between the 90% and the 100% of performance. On the contrary, bankrupt firms are poorly detected. Devastating is also the effect of going behind in time. For example the best ratio, the SR, three years prior to the potential failure in 2021, so in 2018, it is able to detect only 2% of the failed company. The inferior one, the CR, two years before failure, in 2019, it is not capable to detect this type of firms. The higher precision is something useless given that the ratios can only detect at maximum the 18% of the actually failed firms.

| Ratio | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| ROA 2020 | 0.924242 | 0.183183 | 0.917124 | 0.998342 |
| ROA 2019 | 0.666667 | 0.138138 | 0.912473 | 0.992371 |
| ROA 2018 | 0.0 | 0.0 | 0.900538 | 1.0 |
| ROA 2017 | 0.0 | 0.0 | 0.900538 | 1.0 |
| ROA 2016 | 0.625 | 0.015015 | 0.901796 | 0.999005 |
| ROA 2015 | 0.0 | 0.0 | 0.900538 | 1.0 |
| ROA 2014 | 0.0 | 0.0 | 0.900538 | 1.0 |
| ROA 2013 | 0.0 | 0.0 | 0.900538 | 1.0 |
| CR 2020 | 0.166667 | 0.006006 | 0.900779 | 0.996683 |
| CR 2019 | 0.0 | 0.0 | 0.900448 | 0.999005 |
| CR 2018 | 0.0 | 0.0 | 0.900538 | 1.0 |
| CR 2017 | 0.0 | 0.0 | 0.90024 | 0.996683 |
| CR 2016 | 0.0 | 0.0 | 0.900508 | 0.999668 |
| CR 2015 | 0.0 | 0.0 | 0.900389 | 0.998342 |
| CR 2014 | 0.0 | 0.0 | 0.900329 | 0.997678 |
| CR 2013 | 0.0 | 0.0 | 0.90015 | 0.995688 |
| SR 2020 | 0.969231 | 0.189189 | 0.917758 | 0.999337 |
| SR 2019 | 0.851064 | 0.12012 | 0.911239 | 0.997678 |
| SR 2018 | 0.875 | 0.021021 | 0.902395 | 0.999668 |
| SR 2017 | 1.0 | 0.003003 | 0.900807 | 1.0 |
| SR 2016 | 0.75 | 0.009009 | 0.901316 | 0.999668 |
| SR 2015 | 0.0 | 0.0 | 0.900538 | 1.0 |
| SR 2014 | 0.0 | 0.0 | 0.900538 | 1.0 |
| SR 2013 | 0.0 | 0.0 | 0.900508 | 0.999668 |

Table 17: Precision and Recall scores for the two classes.

## 2.5 DEALING WITH IMBALANCED DATASETS: SMOTE AND TOMEK LINK

The results seen in the univariate logistic analysis are due to the fact that the sample is highly unbalanced. This is something that the author has purposely tried to reproduce. With the idea of performing and comparing models on the most feasible real basis optic. As explained in the literature part, in reality, in the third world economies the failure rates are around a maximum of 1%. This means that in the worst-case scenario a developed country can have that in a year 1% of the firms failed or suffered financial trouble. It has no sense to collect and analyse data that are balanced or with small unbalance. On the other hand, having a strong unbalance causes a decrease in the performance of the models. As recalled previously, a model learns from a training dataset, if these data have only a few observations of a specific category the model will understand in a poor way how to allocate the minority class. This is a serious issue when the information in the data sample from the minority class is more important, as in the bankruptcy prediction case. One of the favourite approaches in data science to solve imbalanced data sample problems is to oversample the minority class or undersample the majority one. These methodologies, however, have their vulnerability. In the standard oversampling method, the idea is to randomly multiplicate some observations from the minority class to equalize the number of instances per class. However, this procedure is not able to add any new information to the original instances dataset. On the other hand, the vanilla undersampling procedure is runned by eliminating randomly some observations from the majority class losing some information from the original instances. The undersampling techniques improve runtime and storage problems, on the contrary, the oversampling techniques worsen execution time and storage requirements. To offset these weaknesses, the data science literature has established two sophisticated techniques that have been contemporaneously used in this dissertation. These two procedures are:

- **SMOTE**

Developed by Chawla et al. (2002) the Synthetic Minority Oversampling Technique (SMOTE) artificially expands the number of minority class instances by generating synthetic examples. In other terms, this procedure creates extra training data by performing certain operations on real data. The mathematical viewpoint of the SMOTE takes the bases from the k-nearest neighbour technique. Given the initial sample $x_i$, the new balanced sample $x_{new}$ will be generated considering its k-nearest neighbour. For example, taking k=3 the relative nearest-neighbours are incorporated in the blue circle as illustrated in figure 10. To be precise, in this dissertation the standard k=5 is chosen. Subsequently, one of these nearest-neighbours $x_{zi}$ is chosen, and a part of the new sample is created as:

$$x_{new} = x_i + \lambda(x_{zi} - x_i)$$

Where $\lambda$ randomly flows in the range between 0 and 1. The interpolation creates a part of the new sample on the imaginary line between $x_i$ and $x_{zi}$ as shown in figure 10.

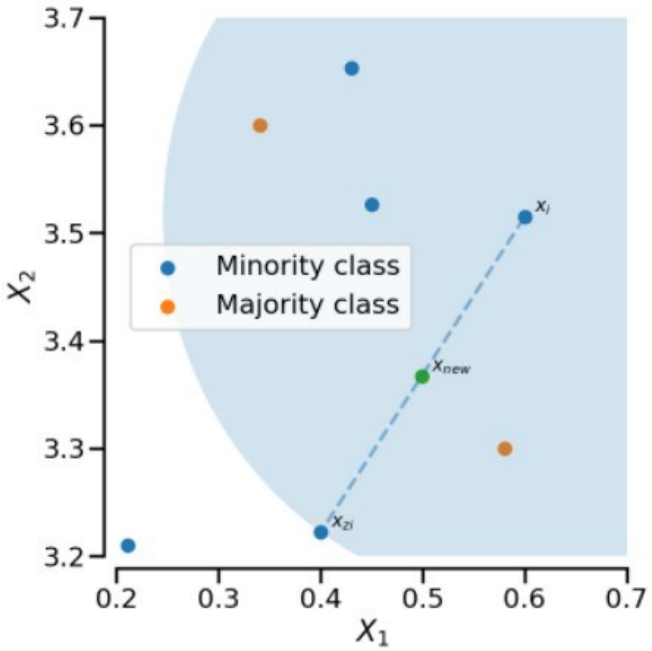


Figure 10: Illustration of the SMOOTE. Source: scikit-learn library.

The procedure can be performed through the function "*imblearn.over_sampling.SMOTE*" in the scikit-learn library in python. Setting the parameter 'sampling_strategy', which controls which class has to be oversampled as figure 10 shown. Fixing that equal to 'minority', the synthetic over-sampling operates to induce the classifier to build larger decision regions that contain nearby minority class observations. The SMOTE procedure provides more related minority class samples to learn from, thus allowing the model to cut broader decision regions, leading to more coverage of the minority class. Indeed, the results show that the SMOTE approach exponentially improves the recall of the bankrupt firm class by the classifiers. On the other hand, the oversampling procedure of the training dataset causes a partial decrease in the precision of the bankrupt firms of the models as will be shown later in chapter 3.

- **TOMEK link**

Developed by Tomek (1976) mathematically a Tomek's link between two samples of different classes $x$ and $y$ is determinate such that for any sample $z$:

$$d(x,y) < d(x,z)\ and\ d(x,y) < d(y,z)$$

Where $d(\cdot)$ is the distance between the two samples. In other terms, a Tomek's link exists if the two samples are the nearest neighbours of each other. Figure 11 shows how a Tomek's link is illustrated by highlighting the samples of interest in green.

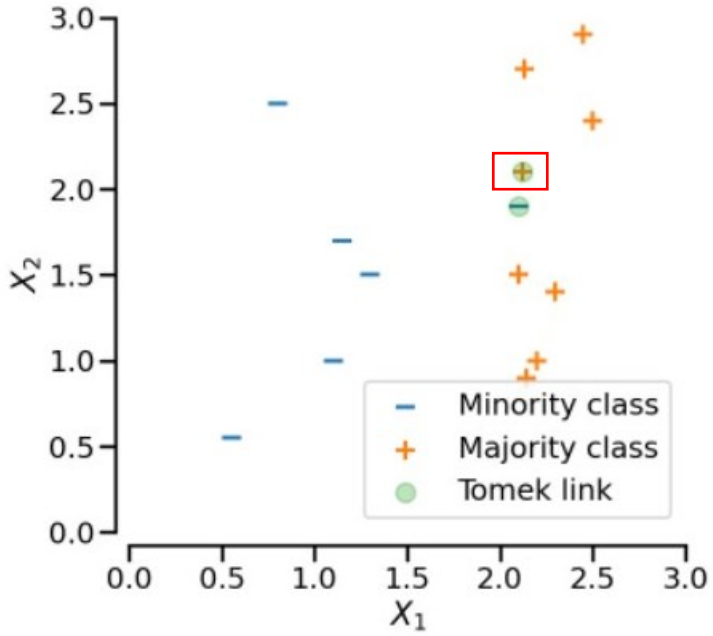


Figure 11: Illustration of the TOMEK link. Source: scikit-learn library.

Through the function "under_sampling.TomekLinks" in the scikit-learn library in python, there is the possibility to define the parameter 'sampling_strategy' that controls which sample of the link will be removed. If the standard 'sampling_strategy' is settled equal to 'auto', only the sample from the majority class will be removed. In this analysis, it has been decided to not change this parameter. There are two reasons for this. First, in an imbalanced dataset, the minority class examples may be too valuable to be wasted, especially if the minority class is extremely underrepresented. Second, the recall score of the minority class is frequently of greater importance than precision, so it is worth retaining some noisy data in order to avoid losing rare cases. Graphically, the Tomek link with this characteristic will cancel all the observations of interest in green with the plus symbol as it is shown in figure 11.

- **SMOTE + TOMEK link**

Both procedures were performed because SMOTE alone can generate noisy samples by interpolating new points between marginal outliers and inliers. This issue can be solved by cleaning the space resulting from over-sampling. In this regard, Tomek's link is a quite good clearing method that has been added to the pipeline after applying SMOTE over-sampling to obtain a cleaner space. This was done through the function "*imblearn.combine.SMOTETomek*" in the scikit-learn library in python. Comparing the result obtained in chapter 3 with the one in

appendix 4 may give a clear overview of the incredible difference in performance between a model with and without the SMOTE and Tomek link techniques. To be consistent with the other part of the analysis, figure 12 shows how the number of firms in the training dataset used to predict bankrupt firms in 2021 changes by applying these two methodologies simultaneously. These procedures rebalanced the observations giving the possibility of a training phase of the models done more in equilibrium.

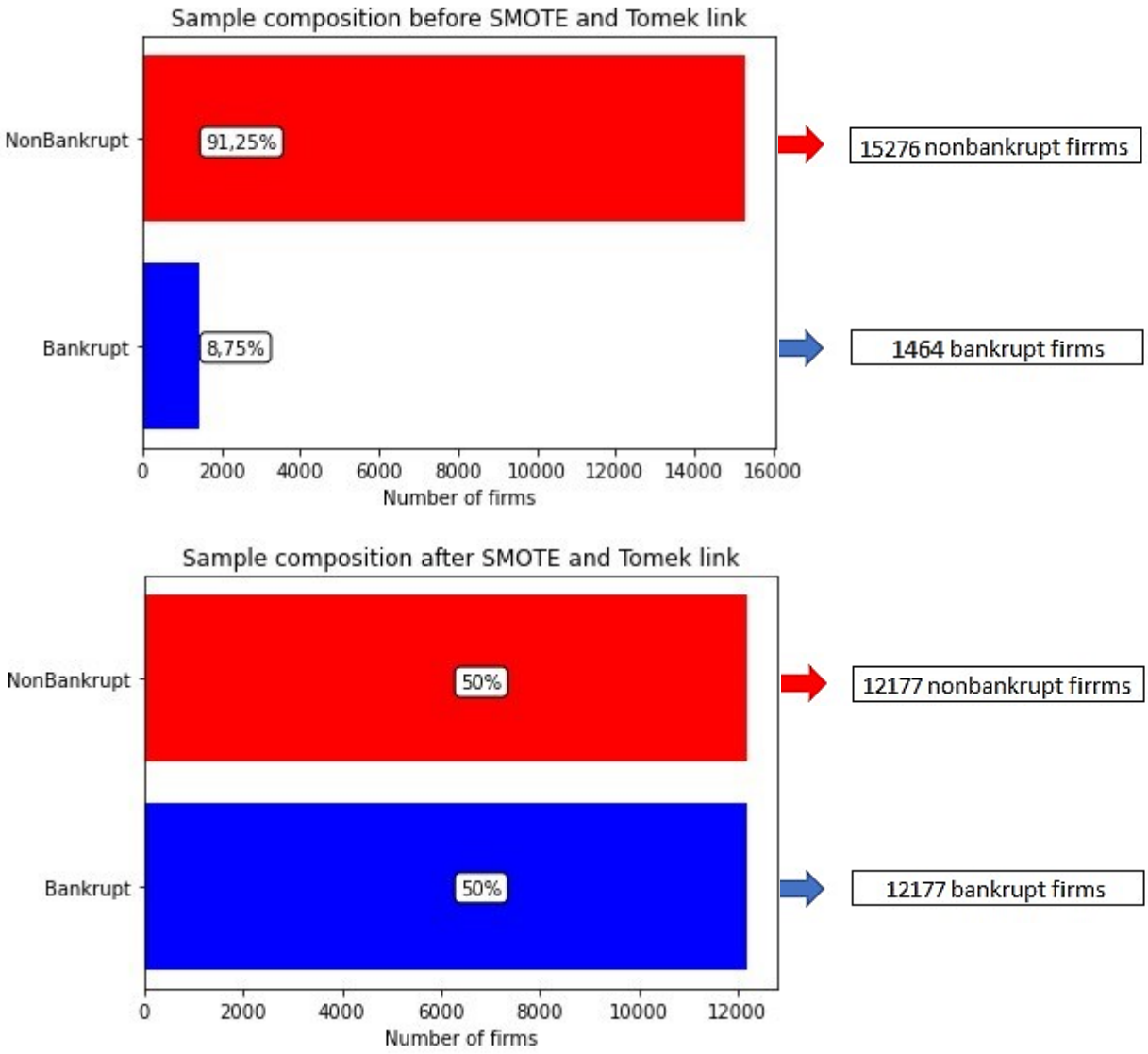


Figure 12: Sample composition before and after SMOTE and Tomek link.

## 2.6 HYPERPARAMETER TUNING

In this section, it will review how the parameters of models that will be executed in the following chapter have been chosen and designed. This is one of the more important and extremely time and computational consuming procedures that have to be made by a data analyst. As explained in the literature part of this dissertation there are many combinations and variables of a single model. The fundamental to make a complete analysis is to exploit all the potential of each model and this can be done only by changing the models' composition. In computer science, the process in which the right combination of hyperparameters that maximize the performance of the model is settled is called hyperparameter tuning or hyperparameter optimization. In the interest of clarity, it is necessary to understand the differences between a parameter and a hyperparameter. The first ones are the parameters that are estimated by the model in the phase of the training. Some examples are the weights of the artificial neural network, the criteria to split the tree in the random forest model and the gradient descendent weight in XGBoost. Instead, the model hyperparameters are the parameters that are not estimated by the model in the training phase. A few examples are the learning rate in the artificial neural network, the soft margin regularization parameter in the support vector machine and the k values for the k-nearest neighbour model. To synthesize, whereas the parameters define how to transform input data into the actual output, the hyperparameters specify the structure of the model. The hyperparameter tuning process practically works by running multiple trials on the training sample. Each test is a complete execution of the training of the model with the hyperparameters that change until all the defined combinations are computed. The number and the types of these combinations are manually predetermined using a parameter space matrix. This process, once finished, gives the set of the hyperparameter values that are best suited for the model to reach the higher possible value into a prespecified performance score, as one of those analysed in the part on the univariate logistic regression. Formally the hyperparameter tuning needs the following elements:

- an estimator or the model;
- a parameter space;
- a method for searching or sampling candidates;
- a cross-validation scheme;
- a score function.

In this dissertation, two approaches to parameter search have been used: a randomized search and a grid search. The first one randomly searches over the parameter space, where each set of

each trial is sampled from a distribution over the defined parameter values. This has two main benefits over an exhaustive search: i) the number of iterations can be chosen regardless of the number of parameters and possible values through the function 'n_iter', setting a computation limit that helps when the parameter space is huge and so high is the time required for the search, ii) adding parameters that do not influence the performance does not increase the time for the search. Practically, for each model, at first, it has been set a random search to decrease the parameter space where the best parameters should be found, and subsequently, through a more specific grid search the optimal parameter were found. Given the type of python library used to perform the model employed in this empirical analysis, two random searches have been applied. The first is the one from the scikit-learn library through the function "*RandomizedSearchCV*" and the second derives from the catboost library through the function ".*randomized_search*". The catboost one is quite interesting because it is able to perform in real-time a graphic representation of the analysed parameter space as shown in figure 13. In the second phase, the grid search has been used to better define the hyperparameter that maximizes the performance score. The author decided to proceed in this way due to the high time required if using only an exhaustive search. In some cases, even with the procedure implemented to reduce the parameter space, the grid search takes hours to find the best settings for some sophisticated models such as the XGBoost one and in the case of the support vector machine (SVM), it takes an entire day. This is something expected given the cross-validation scheme used and the number of parameters analysed.

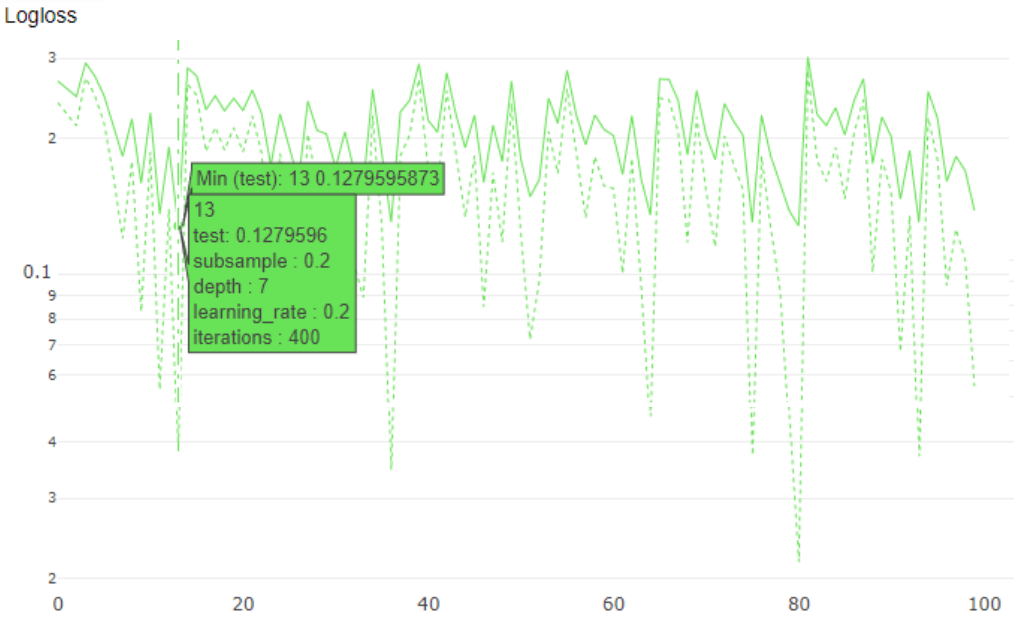


Figure 13: Example of randomized search with 100 iterations.

Before reviewing what is, in data science, the cross-validation procedure, it is necessary to describe clearly why it is mandatory to split the data into training, validation and test sets. The final aim for any machine learning model is to learn from examples in such a manner that the model is able to generalize the learning to new observations which it has not yet seen. At the fundamental level, a model should be trained on a subset of the dataset, holding out the remaining data to test the model's ability to classify new instances. Due to that, the dataset is divided into training and test. In the hyperparameter tuning process, the same need occur. When various model architectures have to be evaluated, the model's ability to generalize to unseen data is what has to be evaluated. However, if the model architecture will be fitted and evaluated to the testing data the ability to truly evaluate how the model performs on unseen data is lost. This is sometimes referred to as 'data leakage' because the test set can leak into the model, and evaluation metrics no longer report on generalization performance. To solve this problem, yet another part of the dataset can be held out as a so-called 'validation set': training proceeds on the training set, after which evaluation is done on the validation set, and when the experiment seems to be successful, final evaluation can be done on the test set. However, by partitioning the available data into three sets, the number of samples which can be used for learning the model is drastically reduced, and the results can depend on a particular random choice for the pair of train and validation sets. A solution to this problem is a procedure called cross-validation (CV). A test set should still be held out for final evaluation, but the validation set is no longer needed when doing CV. In the approach used, called k-fold CV, the training set is split into k smaller sets. The following procedure is repeated for each of the k folds: a model is trained using k-1 of the folds as training data; the resulting model is validated on the remaining part of the data. The performance measure reported by k-fold cross-validation will be the average of

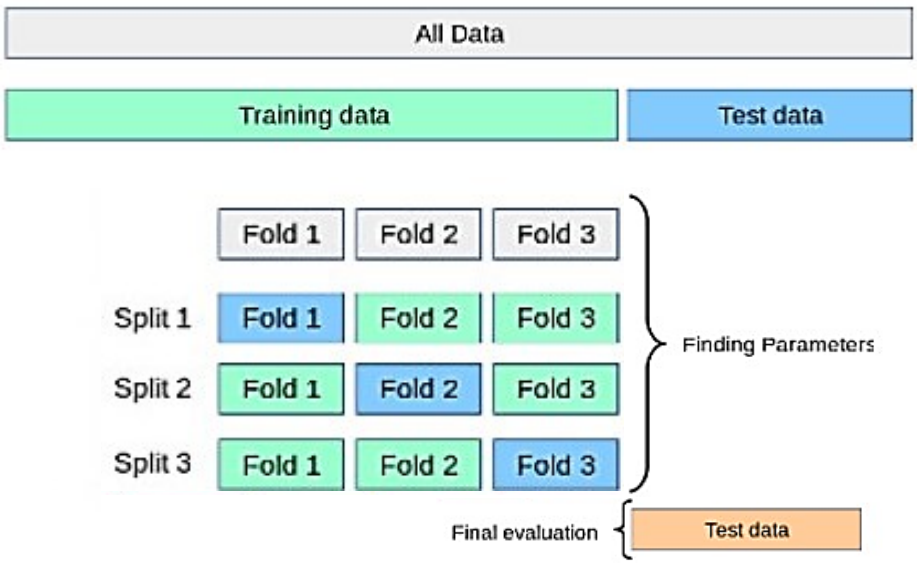


Figure 14: Illustration of the cross-validation procedure. Source: scikit-learn library.

the values computed in the loop. This approach can be computationally expensive but does not waste too much data as does the case when fixing an arbitrary validation set. Figure 14 explains graphically what has been done in this dissertation. The set was randomly divided into 80% for the training phase and 20% for the testing one. A 3-CV was implemented through the parameter 'cv' in the grid and random search functions and the optimal hyperparameter to a specific performance score are found.

Finally, the tuning of the hyperparameters was defined depending on a specific performance metric. Recalling that this analysis has the aim to compare and find the best model to predict future bankruptcy, it has been decided to tune the parameter on the recall score of the class 0 observations. This means that the model hyperparameters found are the optimal ones to classify the higher possible number of bankruptcy firms. This was done in optics of reducing the value of the Type I error and making models that adequately classify the bankruptcy cases. The price of this performance score and procedure, as it will be clear in the following chapter, is that the precision will decrease and so the Type II error will slightly increase. However, as it will be shown this way collected a high rate of correct nonbankruptcy classification and a huge increase in the recall of the failing ones. Other performance measures have been implemented like F1 and F2, but given the theoretical viewpoint of this research and the quite superior result, the recall class 0 score was the chosen one. Figure 15 tries to give a clear idea of what was the code to implement the hyperparameter tuning through a grid search. It is an example of what was done for the XGBoost model. The algorithm made a 3-CV procedure for each of 1440 candidates, for a total of 4320 fits and it finds the best hyperparameters that increase at the maximum possible level the recall of the bankrupt firms. Those settings will be then used in chapter 3 to compute the definitive test performances and to compare and find the best model.

```
# Calibrate the parameters - Grid search for XGBoost model
parameters = {'booster': ['gbtree'],
                'learning_rate': [0.05,0.1, 0.2, 0.25, 0.3],
                  'subsample': [0.2,0.3,0.4,0.5,0.6,0.7],
                    'max_depth': [5,6,7,8],
                      'reg_lambda':[0.95,1,1.01],
                        'colsample_bytree' : [0.7,0.8,0.9,1],
                          'random_state' : [100]
             }

xgbo = XGBClassifier()
recall_score = make_scorer(recall_score, pos_label=0, average='binary')

Grid_CBC = GridSearchCV(estimator=xgbo, param_grid = parameters, cv = 3,
                        n_jobs=-1, scoring = recall_score, refit=True, verbose= 10)
Grid_CBC.fit(x_train, y_train)
```

Figure 15: Example of a hyperparameter tuning code.

## 2.7 MODELS IMPLEMENTATION AND PARAMETERS OVERVIEW

This last part of the chapter will review each parameter space for each of the eight models implemented. For the sake of brevity, it will be quickly specified the difference between the various potential choices and the optimal one, obtained through the hyperparameter tuning process. However, this part should be seen as a unicum with the part related to the review of the literature, in which the model settings and their theoretical evolution were explained. This is because many concepts were specified more in detail in that part of this dissertation.

### I. Multiple Discriminant Analysis

It was performed through the scikit-learn python library, using the function "*.LinearDiscriminantAnalysis*". The tuning process compared the following hyperparameter:

- *Solver*; it is the algorithm used for the classification problem. The Singular Value Decomposition 'svd' is the predefined one. It does not compute the covariance matrix, therefore this solver is suggested for data with a large number of features. The Least Square 'lsqr' solver is an efficient algorithm that only works for classification problems. It needs to explicitly compute the covariance matrix Σ, and it supports the shrinkage parameter. This solver calculates the coefficients $w_k = \Sigma^{-1} u_k$ by solving for $\sum w = \mu_k$, thus avoiding the direct computation of the inverse $\Sigma^{-1}$. The Eigen 'eigen' solver is based on the maximization of the between-class scatter to within-class scatter ratio. It can be used for both classification and regression problems and it supports the shrinkage process. However, it needs to calculate the covariance matrix, due to that it might not be suitable for situations with a high number of features.
- *Shrinkage*; it is a form of regularization used to improve the estimation of covariance matrices in situations where the number of training samples is small compared to the number of features. In this scenario, the empirical sample covariance is a weak estimator, and shrinkage improves the generalization performance of the classifier. In linear discriminant analysis (LDA) the shrinkage parameter can be used by setting it equal to 'auto'. This automatically defines the optimal shrinkage parameter in a mathematical way following the lemma introduced by Ledoit and Wolf (2017). Recall that the shrinkage procedure only works when the solver parameter is settled equal to 'lsqr' or 'eigen'.
- *Store_covariance*; it is a bool parameter that if set as 'True' makes the algorithm explicitly calculate the weighted within-class covariance matrix when solver is settled as 'svd'. The matrix is always computed and stored for the other solvers. From the tuning process, the best parameters to increase the recall of the bankrupt firms are 'lsqr' as solver, 'auto' as regularization term and 'True' for the covariance storing.

### II. Logit Regression

It was performed through the scikit-learn python library using the function ".*LogisticRegression*". The hyperparameters space was composed by:

- *Penalty*; it is a regularization term and relies on the type of solver used for the regression. The choices are: ' $\ell_2$', ' $\ell_1$' and 'elasticnet'. The first one is also called Ridge regression and it is the optimization problem of solving the following function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log\left(\exp\left(-y_i(X_i^T w + c)\right) + 1\right).$$

Similarly, the '$\ell_1$' regularized logistic regression solve the cost function below:

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log\left(\exp\left(-y_i(X_i^T w + c)\right) + 1\right).$$

Finally, the Elastic-Net regularization is a combination of these two and $\rho$ controls the intensity of '$\ell_1$' and '$\ell_2$' and it minimizes the following cost function:

$$\min_{w,c} \frac{1-\rho}{2} w^T w + \rho \|w\|_1 + C \sum_{i=1}^{n} \log\left(\exp\left(-y_i(X_i^T w + c)\right) + 1\right)$$

- *C*; it is the inverse of regularization strength and it could be any value greater than 0. Like in support vector machines C=1 is the standard value, however, smaller values specify stronger regularization.
- *Solver*; it is the algorithm to use in the optimization problem. The solver 'liblinear' uses a coordinate descent (CD) algorithm and it converges faster for some high-dimensional data. The 'lbfgs', 'sag' and 'newton-cg' solvers only support '$\ell_2$' regularization or no regularization. The 'sag' solver uses stochastic average gradient descent. It is faster than other solvers for large datasets when both the number of samples and the number of features are considerable. The 'saga' solver is a version of 'sag' that also supports the '$\ell_1$' and 'elasticnet' penalties. The 'lbfgs' is a maximization algorithm that approximates the Broyden, Fletcher, Goldfarb and Shanno method, which belongs to quasi-Newton ones. The 'lbfgs' solver is recommended to use for small datasets but for larger datasets its performance suffers.
- *Class_weight*; it can be settled as 'balanced' or 'none' which is the default option. It defines the weights associated with classes. If not given, all classes are supposed to have weight one. The 'balanced' mode uses the values of y to automatically revise weights inversely proportional to class frequencies in the input data. In this way the model takes into account the imbalances of the sample, increasing the weights for the bankrupt firms. From the tuning

process the optimal parameters are '$\ell_1$' for the penalty term, C=0.5, 'liblinear' for the solver and 'balanced' as 'class_weight'.

### III. k-Nearest Neighbour

It was performed through the scikit-learn python library using the function ".*KNeighborsClassifier*". The tuning procedure was made on:

- *N_neighbors*; it is the number of neighbours (k) to use for each query classification. The optimal choice of the value k is highly data-dependent, in general, a larger k represses the effects of noise but makes the classification boundaries less distinct and accurate.
- *Metric*; it is the distance metric to use for the tree. Recalling the literature part of the dissertation and given that cosine distance is not available in the scikit-learn library, the Euclidean, the Cityblock and the Minkowski ones are compared. This last distance metric is defined as follows:

$$(\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$$

If p=1 this metrics is equal to the Cityblock distance and if p=2 it is equal to the Euclidean one.

- *Weights*; it is the weight function (the fuzzy parameter) that assigns each neighbour a given value. There are two choices: 'uniform', all points in each neighbourhood are weighted equally and it will perform a nonfuzzy kNN or 'distance', the points are weighted by the inverse of their distance. In this case, closer neighbours of a query point will have a greater influence than neighbours which are further away. This hyperparameter defines if a fuzzy-kNN will be performed or not. From the parameter search the optimal value are k=3, 'euclidean' as metric and 'uniform' as weights. The fact that the hyperparameter process displays that a uniform approach is better than the fuzzy one is surprising but reflects the data-dependent of this parameter.

### IV. Support Vector Machine

It was performed through the scikit-learn python library using the function ".*SVC*". The hyperparameters were chosen on the following elements:

- *Kernel*; it is the function that allows to find a solution to the inseparability problem and map the data in a higher-dimension space. As previously examined in the literature part, there are several well-behave functions. Through the random and grid search, it has been analysed which one between the 'linear', 'polynomial' and 'radial gaussian'. Given the high time

required to perform the hyperparameter search, it was not possible to add also the 'sigmoid' kernel.

- *C*; it is the soft margin that allows the algorithm to retain some points to be inside the margin of the separating hyperplane as explained in the literature part. It is a regularization parameter whose strength is inversely proportional to C.
- *Gamma*; it is the delta value in the literature part. It defines how much influence a single training example has. The larger gamma is, the closer other examples must be to be affected. If it is set as 'auto' the gamma value is equal to $1 / (\text{n\_features} * \text{X.var}())$. If it is set as 'scale' is equal to $1 / \text{n\_features}$. Given that X.var() is 1 because the data has been standardized and normalized, 'scale' and 'auto' are the same. From the tuning procedure, the best parameters are 'linear' for the kernel, C=0.1 and 'scale' as gamma.

### V. Neural Network

It was performed through the scikit-learn python library using the function "*.MLPClassifier*". The parameter compared were the following:

- *Hidden_layer_sizes*; it is composed by two number, the first one represents the quantity of hidden layers and the second one the quantity of neurons/nodes of the network. Coherently to what is explained in the literature part, the best number of nodes is equal to the number of explanatory variables, in this analysis is 9. Instead, the best number of hidden layers differs to what found in literature. In fact, the optimal number of hidden layers to maximize the recall of bankrupt firms for the model is much higher. Instead of one or two as said by literature, it has been found that the optimal value is between 80 and 100 hidden layers. This could give the problem of overfitting, but, as will be shown in chapter 3, it seems to work well in this framework.
- *Activation*; it is the transfer function for the hidden layer that determined how the weighted input are converted into the output. The possible choice are: 'logistic' also called sigmoid function which formula is f(x) = 1 / (1 + exp(-x)); 'tanh', a hyperbolic tan function which function is f(x) = tanh(x); 'relu', a rectified linear unit function with function f(x) = max(0, x) and finally 'identity', a linear function f(x) = x.
- *Max_iter*; it is the maximum number of iterations. The solver iterates until convergence or it reaches this number. For stochastic solvers such as the 'sgd' and 'adam' ones, this value determines the number of epochs, so how many times each data point will be used, and not the number of gradient steps.
- *Solver*; it is the training algorithm for weight optimization. There are three main solvers based on a Multilayer Perceptron classifier (MPL): 'lbfgs', an optimizer in the family of quasi-

Newton methods; ‘sgd’ that is a stochastic gradient descent method and ‘adam’ that refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Ba (2014). Mathematically the algorithm updates the parameters weights using the gradient of the loss function as follow:

$$w \Leftarrow w - \eta(\alpha \frac{\partial R(w)}{\partial w} + \frac{\partial Loss}{\partial w})$$

Where $\eta$ is the learning rate and $Loss$ is the loss function used for the network. The default solver ‘adam’ works pretty well on relatively large datasets in terms of both training time and validation score. For small datasets, however, ‘lbfgs’ can converge faster and perform better. From the tuning technique, the best parameters are ‘logistic’ as activation function, 100 hidden layers and 9 nodes, 300 maximum iterations and ‘adam’ as training algorithm.

### VI. Random Forest

It was performed through the scikit-learn python library using the function “.*RandomForestClassifier*”. The hyperparameters space was made by:

- *N_estimators*; it represents the number of trees in the forest. This is a highly data-dependent parameter, in general, the bigger the value the better the estimation until a certain number, but also the longer the time demanded for the computation.
- *Max_features*; it is the number of ratios randomly taken into consideration when looking for the best split. The lower the ‘max_features’ value, the higher the reduction of variance and the higher the increase in bias. The choices available are: if ‘auto’ or ‘sqrt’, then maximum features are equal to $\sqrt{\text{number of features}}$, in this case, equal to 3. If ‘log2’, then max_features is equal to $2 \text{ x } \log(\text{number of features})$.
- *Criterion*; it is the function to measure the quality of a split. Supported criteria are ‘gini’ which is the gini impurity seen in the literature part and ‘entropy’ for the information gain. In the doctrine, it is stated that there is no superior criteria method, however, this analysis will choose the result from the grid search.
- *Max_depth*; it is the maximum depth of the tree. If set as ‘None’, then nodes are expanded until all leaves are pure or until all leaves contain less than the minimum number of samples required to split an internal node which is 2. It can be also settled whatever positive integral number for the maximum number of trees.
- *Bootstrap*; it is a bool parameter that set whether bootstrap samples are used when building trees. If ‘False’, the whole dataset is used to build each tree, if ‘True’ only a small dataset is used.

- *Ccp_alpha*; it is a parameter used for minimal cost-complexity to prune a tree to avoid overfitting with a default value equal to 0.0. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. The complexity parameter is used to define the cost-complexity measure, $R_\alpha(T)$ of a given tree T:

$$R_\alpha(T) = R(T) + \alpha\left|\tilde{T}\right|$$

Where $\left|\tilde{T}\right|$ is the number of terminal nodes $T$ and $R(T)$ is traditionally defined as the total misclassification rate of the output nodes. From the tuning methodology, the best parameters are 'True' for the bootstrap procedure, 0.01 as 'ccp_alpha', 'gini' as criteria for the split, '5' as maximum number of trees, 'max_features' set in 'auto' and 90 for the number of estimators.

### VII. XGBoost

It was performed through the xgboost python library using the function "*.XGBClassifier*". The space of the search was composed by:

- *Booster*; it defines the type of boost used: 'gbtree' and 'dart' use tree-based model while 'gblinear' uses a linear function for each learner in the gradient boosting process. For brevity, the dart booster is an XGBoost with a new method to add dropout procedures derived from the artificial neural network field.
- *Learning_rate*; it defines how much the gradient descendent algorithm can move from the starting point. As explained in the literature part it is called shrinkage and it is a regularization procedure to avoid overfitting.
- *Subsample*; it determines as the bootstrap parameter in RF if and how much sample is used in composing the ensemble model of the training instances. Setting it equal to 0.5 means that XGBoost would randomly sample half of the training data prior to growing trees and this will prevent overfitting. Subsampling will occur once in every boosting iteration. It is also known as 'bag fraction'.
- *Reg_lambda*; it is a L2 regularization term on weights. Increasing this value will make model more conservative, and on the contrary a smaller value makes the model freer to adapt itself to new data.
- *Colsample_bytree*; it is the subsample ratio of columns used when constructing each tree. Subsampling occurs once for every tree constructed. In this way, as occurs in RF, it reduces the similarity and correlations between trees in the ensemble boost model. From the tuning process, the best parameters are 'gbtree' as the type of booster, 0.7 as subsample of columns, 0.05 as learning rate, 5 maximum number of nodes in each tree, 0.95 for the '$\ell_2$' regularization.

### VIII. CatBoost

It was performed through the catboost python library using the function "*.CatBoostClassifier*". Only a new parameter was included compared to the XGBoost hyperparameter space:

- *Scale_pos_weight*; it gives the weight for the incorrect identification of class 1 in binary classification. The value is used as a scalar for the weights of objects from the nonbankrupt firms. A smaller value gives more weight to the bankrupt observations (class 0). In other terms, the algorithm with a high *scale_pos_weight* place more weight on bankrupt firms in the learning phase, inducing the model to find more often the class 0 instances. The fact that the main difference between the XGBoost and CatBoost hyperparameters is the *scale_pos_weight* parameter makes it possible to verify something that is not considered in the literature. The author would like to see if an unbalanced weight model (XGBoost) is better or not than a balance weight one (CatBoost). However, the tuning process states that the best parameters are 0.05 as 'learning_rate', 4 as 'max_depth', '100' number of estimators, subsample rate of 0.2 and scale_pos_weight equal to 0.9.

# CHAPTER 3: RESULTS AND DISCUSSIONS

In this section, the results of the various models will be displayed considering one, two and three years as the distance between the last ratio used to make the prediction and the relevant year. For example, the prediction at one year from the potential failure in 2021 will be done with the ROA, CR and SR from 2018, 2019 and 2020. For the sake of brevity and the similarity of the results, the prediction for the 2020 and 2019 groups will be displayed in appendix 5. As shown in chapter 2 in the univariate logistic regression section, the analysis will be firstly done in a general perspective with an AUC-ROC curve that shows graphically the accuracy performances of the various models. Secondly, the author will exhibit the average result of the main unweighted indicators such as the accuracy, specificity, recall, F1 and F2. Finally, the analysis will go in deep seeing the number of firms correctly specified for the two classes through the recall and specificity of each of these. The choice of the best models will be based on two criteria. The first one refers to comparing the value obtained by the models in the F2 score. The second one concerns the analysis of the recall obtained for bankrupt firms (class 0) in the optic that it is not obvious that the model with the higher value of that score is the best one but what is important is the differential between the best models. If there will be two or three models that have very similar recall class 0 scores but only one of them has the other metrics that are much higher, that will be the best one. This is because even if maybe it is not the higher recall 0 model, such low differences do not justify the fact of picking the other algorithm. In other words, it is preferable to implement a model that performs well and recognizes efficiently the class 0 firms and is balanced in its performances compared to a model that for only some percentage point recall better the bankruptcy firms but has low precision in its classification.

## 3.1 ONE-YEAR DISTANCE RESULTS

The first of the three cases, logically, should be the one with the higher performance compared to the other ones given the closeness to the relevant year. On the other hand, the practical application in the real world seems to be restricted. The ability of a model to well predict bankruptcy only one year before, ceteris paribus, is less significant compared to a model that is able to predict accurately two and even three years in advance. However, for completeness, it is worth to see the maximum and the less real performance measures. Figure 16 displays the AUC-ROC curve for the eight models at 1 year, as explained in the chapter above, this does not provide a correct measure of the ability to predict failure but helps to understand the general ability and clearly explains the big differences in performance between the models. For the

record, in the legend the term 'LinearDiscriminantAnalysis' refers to the MDA model and 'SVC' to the Support Vector Machine, this is due to the function name used in the python code as shown in chapter 2.

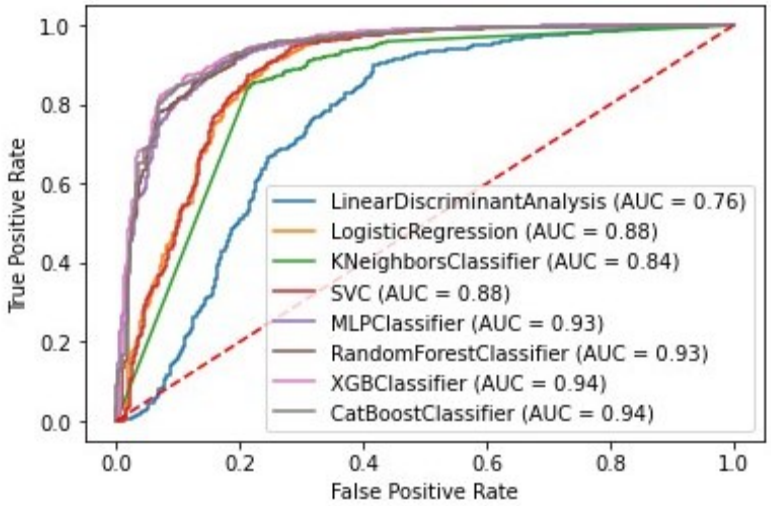


Figure 16: AUC-ROC curve comparison at 1 year.

Practically, there are three groups of models in terms of working behaviour. The first one is composed only of the MDA model, which is the worst one, with a performance of 76%. The second group is composed of the kNN, logit model and SVM. These are less sophisticated and quite old approaches but seem to perform quite well with a maximum of 88%. The best group of models, as also shown in the literature, are the NN, RF, XGBoost and CatBoost with a maximum value of 94%. It is logical to expect that the best model will be in this last group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.90233 | 0.722554 | 0.666617 | 0.688941 | 0.674499 |
| Logit | 0.894265 | 0.724833 | 0.827762 | 0.761488 | 0.796333 |
| KNN | 0.886201 | 0.708244 | 0.799243 | 0.740948 | 0.771728 |
| SVM | 0.919654 | 0.772628 | 0.829837 | 0.797338 | 0.815734 |
| NN | 0.874253 | 0.706517 | 0.860728 | 0.749014 | 0.802671 |
| RF | 0.884707 | 0.717214 | 0.857183 | 0.759867 | 0.808211 |
| XGBoost | 0.900836 | 0.740604 | 0.87148 | 0.784482 | 0.829159 |
| CatBoost | 0.884409 | 0.718073 | 0.863695 | 0.761592 | 0.81195 |

Table 18: Unweighted general performances measures at 1 year.

Table 18 exhibits the unweighted performance measures of the eight models. As explained in chapter 2, these metrics are the simple average between the measures for the two classes. In this way, the imbalance of the sample is taken into account without fictitiously increasing the scores. Except for the accuracy one, all the other scores give an accurate idea of the performances of

the models. The accuracy metric is highly dependent on the sample proportion of the two classes and gives some misleading information. Surprisingly, the precision scores are quite similar between the various model. Also, the old MDA one is quite reliable when classifying the two classes of instances, with SVM that reach the higher score with about 77%. The precision score defines the efficiency of the classification. A high precision model commits a low error when it classifies the two classes. In other words, when it classifies an instance, it is rarely wrong. However, it can happen that a model has high precision but low recall, this means that the model is a poor one and it cannot correctly detect the observations, but when it does it is infrequently wrong. What shows the overwhelming ability of sophisticated machine learning models over others is the recall score. For example, on 100 firms the XGBoost model is the one that more correctly classifies firms, with on average 87 companies compared to only 66 of the MDA one and the 82 of the Logit model. The F1 score establishes the unweighted mean between the precision and the recall metric, showing that except for SVM, the XGBoost reaches the higher value with almost 78%. The F2 score correctly gives twice the emphasis to the recall score such importance in cases of imbalances dataset and again the XGBoost has the higher score with about 83%. To specifically analyse the model scores, it is necessary to distinguish the performances compared to the two classes as shown in table 19. The same pattern explained in chapter 2 is also present in the models, the nonbankrupt firms are classified correctly more than 90% of the time, instead, the bankrupt ones are more poorly detected. The values are much higher than what the univariate discriminant analysis has shown, however, there are still some inadequate values of recall for example from the old models like MDA and KNN. The higher values are those obtained by models like NN, Catboost, XGBoost and RF with nearly 83% of bankrupt firms correctly classified. The higher precision, and so the smaller Type II error is obtained by the XGBoost. The higher recall of class 0 is reached by the NN model. However, given that the difference in recall is too small to prefer NN to XGBoost (the differential is less than 1%), for the most favourable combination of recall and precision for the class 0 firms at 1 year the best model is the XGBoost one.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.512397 | 0.372372 | 0.932711 | 0.960862 |
| Logit | 0.479691 | 0.744745 | 0.969975 | 0.910779 |
| KNN | 0.452756 | 0.690691 | 0.963732 | 0.907794 |
| SVM | 0.577295 | 0.717718 | 0.967962 | 0.941957 |
| NN | 0.432308 | 0.843844 | 0.980726 | 0.877612 |
| RF | 0.455907 | 0.822823 | 0.978522 | 0.891542 |
| XGBoost | 0.500901 | 0.834835 | 0.980308 | 0.908126 |
| CatBoost | 0.455882 | 0.837838 | 0.980263 | 0.889552 |

Table 19: Performances measures for each class at 1 year.

### 3.2 TWO-YEARS DISTANCE RESULTS

In this section, the same performance scores considered in the previous paragraph are examined. This time, however, results relate to three years ratio values taken two years before the relevant year of default. Practically the ROA, CR and SR variables are taken for the years 2019, 2018 and 2017. Figure 17 shows the AUC-ROC curve for the eight models. As expected, the score decreases compared to the one obtained at one year. The chart establishes that there are two types of models: the poorest ones are the kNN, MDA, Logit and SVM and the superior ones are NN, RF, XGBoost and CatBoost. Hence, with high probability, the best model will be in the last group as shown in the paragraph before.

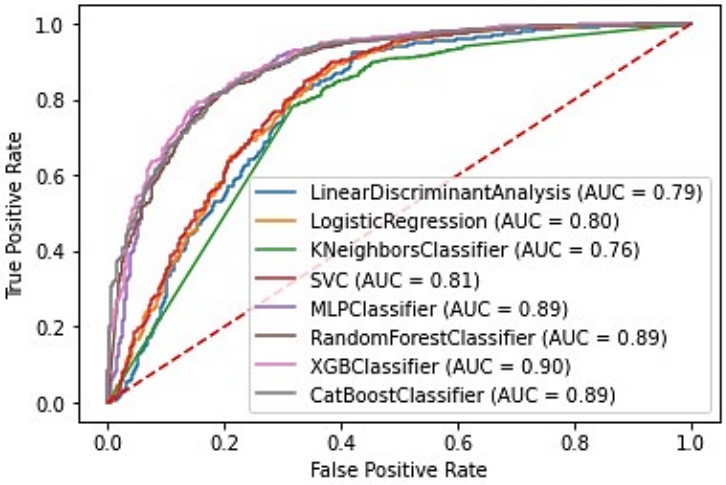


Figure 17: AUC-ROC curve comparison at 2 years.

Table 20 is helpful to understand the performances of the models. Except for the accuracy measure that suffers of biases in imbalance samples, the others are useful to see the average performances in the binomial classification problem. Again unsurprisingly, the scores are decreasing compared to the one year one, the precision values are quite poor with a maximum value of 72% of the MDA model. As explained above, it is not a crucial score to evaluate model

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.90233 | 0.723779 | 0.683981 | 0.701128 | 0.690284 |
| Logit | 0.822581 | 0.634253 | 0.746556 | 0.65869 | 0.698718 |
| KNN | 0.844982 | 0.641691 | 0.721595 | 0.665953 | 0.693833 |
| SVM | 0.83184 | 0.642653 | 0.754368 | 0.669352 | 0.70896 |
| NN | 0.847969 | 0.671935 | 0.814078 | 0.70636 | 0.756333 |
| RF | 0.83632 | 0.66176 | 0.807611 | 0.693363 | 0.744844 |
| XGBoost | 0.858423 | 0.681392 | 0.817211 | 0.717655 | 0.76517 |
| CatBoost | 0.825866 | 0.655459 | 0.808485 | 0.68452 | 0.738616 |

Table 20: Unweighted general performances measures at 2 years.

performance. Instead, the important one is the recall metric. The higher value is reached by the XGBoost with almost 82% value. The F1 score suggests that when the Type I and II errors are equally important the higher value is reached by the XGBoost model. The F2 score implies that when the Type I error is twice as important as the Type II error, the best model remains the XGBoost. Table 21 is the last phase to understand completely the performances of the models. As usual, the precision and recall of the nonbankrupt firms are quite good, even if the values are lower than what shows at 1 year. The recall value of class 0 firms of the sophisticated model remains adequate with the better one that is the CatBoost one with 78%. However, the other refined ML techniques are good with values that are near 76 and 77%. What drastically decreases, is the precision of even these models, with the maximum value reached by the XGBoost one with nearly 40%, except for MDA which however given the very poor recall value is discarded. Given the best unweighted and general scores performances, the small difference in the recall of class 0 compared to the other refined ML algorithms and the higher value in the precision of class 0, the best model remains the XGBoost model.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.511194 | 0.411411 | 0.936364 | 0.956551 |
| Logit | 0.31223 | 0.651652 | 0.956276 | 0.841459 |
| KNN | 0.335106 | 0.567568 | 0.948276 | 0.875622 |
| SVM | 0.327844 | 0.657658 | 0.957463 | 0.851078 |
| NN | 0.372464 | 0.771772 | 0.971407 | 0.856385 |
| RF | 0.352538 | 0.771772 | 0.970981 | 0.843449 |
| XGBoost | 0.391705 | 0.765766 | 0.971079 | 0.868657 |
| CatBoost | 0.338501 | 0.786787 | 0.972416 | 0.830182 |

Table 21: Performances measures for each class at 2 years.

### 3.3 THREE-YEARS DISTANCE RESULTS

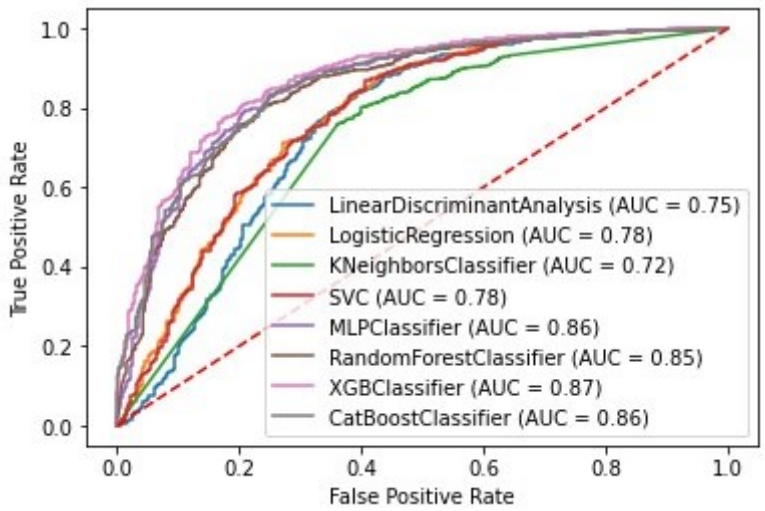


Figure 18: AUC-ROC curve comparison at 3 years.

The analysis will follow the same method as the two previous paragraphs. The results derive from three years ratio values taken three years before the relevant year of default. Hence, the ROA, CR and SR variables are taken from the years 2018, 2017 and 2016. Figure 18 displays that there are two groups of models. However, compared to the cases before, the difference is less remarkable, this means that the superior models are losing performance, and given that the others are quite poor, they marginally lose discriminant ability less quickly than the former ones. Table 22 shows the usual pattern, from the accuracy score it can be observed a general performance decrease given the 3-year bias from the default year. The higher precision score is reached by the XGBoost model with 65%, except for the 70% of MDA one. The recall score has a decreased and a maximum value that remains near the 80% for the XGBoost model which is the best in this score value. The F1 score and particularly the F2 one confirm the usual pattern of the superiority of the most recent ML and in detail of the XGBoost model.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.891876 | 0.692977 | 0.67417 | 0.68286 | 0.677489 |
| Logit | 0.773596 | 0.601592 | 0.715351 | 0.611654 | 0.652958 |
| KNN | 0.823178 | 0.610682 | 0.680104 | 0.629046 | 0.653771 |
| SVM | 0.763441 | 0.598347 | 0.715055 | 0.605103 | 0.647466 |
| NN | 0.789128 | 0.631982 | 0.789422 | 0.648869 | 0.704629 |
| RF | 0.811231 | 0.636795 | 0.773645 | 0.660215 | 0.708958 |
| XGBoost | 0.834827 | 0.65782 | 0.797432 | 0.688418 | 0.737758 |
| CatBoost | 0.792115 | 0.629434 | 0.777724 | 0.647141 | 0.699881 |

Table 22: Unweighted general performances measures at 3 years.

To be precise, table 23 shows the accuracy and recall scores for both classes.The first group of models confirms their poor discriminant power for the bankruptcy group with a higher value near 65%. The second group is able to determine more than 10% better than the former one, with a higher value reached by the NN model. As always, the nonbankrupt group is efficiently recognized by all the algorithms, even if in the second group the better one is the XGBoost

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.451178 | 0.402402 | 0.934775 | 0.945937 |
| Logit | 0.250879 | 0.642643 | 0.952305 | 0.78806 |
| KNN | 0.281619 | 0.501502 | 0.939746 | 0.858706 |
| SVM | 0.243575 | 0.654655 | 0.953119 | 0.775456 |
| NN | 0.292547 | 0.78979 | 0.971417 | 0.789055 |
| RF | 0.309068 | 0.726727 | 0.964522 | 0.820564 |
| XGBoost | 0.347222 | 0.750751 | 0.968417 | 0.844113 |
| CatBoost | 0.291139 | 0.75976 | 0.967729 | 0.795688 |

Table 23: Performances measures for each class at 3 years.

model with a value of 84%. Despite the differential starts to be weakly significant, almost 79% of recall of class 0 for the NN model compared to the 75% of the XGBoost, the other scores are much higher for the XGBoost. For example, the precision of class 0 and recall of class 1 is almost 5% better than the NN values. Given these reasons and due to that XGBoost is the most balanced model as shown by the F2 score, it is the best one to predict failure at 3 years.

# CONCLUSIONS

This dissertation had the aim to compare on a realistic basis the performances of the most common models in the bankruptcy prediction field. The algorithms were chosen for their importance in the literature, starting from the older statistic one to conclude with the ensemble machine learning model. In a preliminary phase, it has been randomly taken the two sample observations in a way to have a high unbalanced dataset as in the real world. To maintain the analysis as realistic as possible and to overcome some problems of lack of some variables due to the Orbis database, it has been decided to use only three ratios as explanatory variables. This was performed in the optic of analysing and comparing the models to understand which one is the best in the absence of information. After some preprocessing procedures and the evaluation of the multicollinearity' issue, it has been dealt with the unbalances of the dataset which is a well-known problem in data science. The author has pushed on this problem because it is something that is barely touched in the bankruptcy prediction literature. The definition of the proportion of the sample is something that significantly affects the performance of the models. To not fictitiously increase these values, it has been decided to make a real basis analysis having almost 9% of bankrupt firms compared to the total population. To solve this problem the SMOTE and Tomek link procedure was implemented on the training dataset. To perform the models at their best the hyperparameter tuning process has been executed. In that way, the models' hyperparameters are calibrated and the best setting solution for each of them is found. The final phase was to perform the eight models and examine their working behaviour one, two and three years prior the failure. The findings are straightforward: the ability of prediction of all the models increases while increasing the proximity to the bankruptcy year; it is possible to observe a decreasing score trend increasing the time prediction distance. Anyway, this is something already seen in the bankruptcy prediction literature. Relating to the models' comparison there is one group of models which were revealed to have the higher performances through the AUC-ROC curve and afterward in the other metrics. These algorithms are the neural network, the random forest, the XGBoost and Catboost. The superiority of these models was verified with the unweighted and specific class scores. However, the task of stating which algorithm is the best one is a subtle and labile problem. Due to the very similar results of these four models, the definition of the best one is a matter of small percentages, however, looking at the decimal, the space of the candidate algorithms can be restricted to the XGBoost and NN algorithms. The final choice and the criteria used are based on the concept of model balance. The first criterion is the higher F2 score, this means that the optimal algorithm is the one with the higher unweighted average between recall and precision of both classes. The second one is

choosing the model with the higher ability to detect failed firms conditional on how the precision of class 0 and recall of class 1 change. In other terms, a model is the best one because it has a high level of discrimination ability for the bankrupt class but at the same time it is precise. This is because if for the higher recall the model requires as price a level of Type II error that is disproportionate, the model should be discarded. The gains in terms of recall and the loss in terms of precision have to be balanced. Of course, this is an arbitrary decision. Another analyst could decide to maximize the recall of bankrupt firms no matter the cost. Practically, this approach can be performed without a doubt in the one and two-year cases, given that the recall difference between XGBoost and the NN is less than 1%, instead, the precision gain using the former model is respectively 7% and 2%. For the three-year case, the choice of a NN model could be rational considering that the differential becomes almost 4%. However, given the high stability in the values of the XGBoost model, and such low precision and so high Type II error rate of the NN algorithm, the author continues to prefer the XGBoost one. Another important point is that the decrease in the recall value for the sophisticated model is quite low. This means that even at three years of distance, the model performs quite well classifying correctly near 75% of the failed firms. This is due to the SMOTE and Tomek link approach that exponentially increases the discriminant power of all the models.

Even if the analysis implemented in this dissertation leads to encouraging results, some shortcomings should be highlighted. The lack of financial statement data and so the construction of models taking into account only three types of ratios is a limit of this examination. Moreover, it should be used not only analytical information that could be transposed into numbers. A good idea to increase the precision could be to add all the qualitative information that are usually discarded in this type of analysis, due to comparability problems. More complete data should be acquired to see how the ability of these models to predict failure changes, increasing the available data. Another significant weakness is the low level of precision. This, as explained in the introduction part, could lead to a self-fulfilling prophecy. Firms that are wrongly classified as failed should go bankrupt because of the incorrect rating of the models. Despite an outstanding bankruptcy firm recall score, it is necessary an improvement in the Type II error rate. That could be reached by increasing the quality and quantity of explanatory variables and including these procedures of ML models on a system of credit rating supervised by experts.

To conclude, ensemble methods like RF, XGBoost and CatBoost and the descendent gradient NN algorithm are very fascinating, and they are models with a lot of potential. These models should be the center of future research in this field because they are able to overcome all the other statistical models and the older machine learning algorithms. To answer the question of

the thesis, it is worth to implement the ML models compared to the naïve and old ones, but we should implement some procedures like the SMOTE and Tomek link to avoid problems derived from the dataset class unbalances. However, this thesis could be continued and extended by increasing the unbalanced of the dataset, reaching a proportion of only 1% of bankrupt firms compared to the total population, to see how the models work in those extreme situations. Another possible research path in this field is testing the models with observations with more ratios and with some type of qualitative information as explanatory variables. In this way, the lack of precision in the more distant years could be balanced and partially solved by reaching an extraordinary level of accuracy.

**References**

Altman, E. (2000). Predicting Financial Distress Of Companies: Revisiting The Z-Score And Zeta. *Handbook of Research Methods and Applications in Empirical Finance*. 5. 10.4337/9780857936097.00027.

Altman, E., I. (2013). Predicting financial distress of companies: revisiting the Z-Score and ZETA® models. In Handbook of *Research Methods and Applications in Empirical Finance*. Cheltenham, UK: Edward Elgar Publishing.

Barboza, F., Kimura, H. & Altman, E. (2017). Machine learning models and bankruptcy prediction, *Expert Systems with Applications*, Volume 83, Pages 405-417, ISSN 0957-4174. Available on: https://doi.org/10.1016/j.eswa.2017.04.006.

Barron, A., R. (1994). Neural Networks: A Review from Statistical Perspective: Comment, *Statistical Science*, vol. 9, no. 1, pp. 33–35. Available: http://www.jstor.org/stable/2246277.

Bateni, L. & Asghari, F. (2020). Bankruptcy Prediction Using Logit and Genetic Algorithm Models: A Comparative Analysis. *Computational Economics*. 1-14. 10.1007/s10614-016-9590-3.

Beck, N., Katz, J., N. & Tucker, R. (1998). Taking Time Seriously: Time-Series-Cross-Section Analysis with a Binary Dependent Variable. *American Journal of Political Science*, 42(4), 1260–1288. Available on: https://doi.org/10.2307/2991857.

Bentéjac, C., Csörgő, A. & Martínez-Muñoz, G (2021). A comparative analysis of gradient boosting algorithms. *Artif. Intell. Rev* 54, 1937–1967. Available on: https://doi.org/10.1007/s10462-020-09896-5.

Blazy, R. & Combier, J. (1997). La défaillance d'entreprise: causes économiques, traitement judiciaire et impact financier.

Breiman, L. (1996). Bagging predictors. *Machine Learning* 24, 123–140. Available on: https://doi.org/10.1007/BF00058655.

Breiman, L. (1996). Bias, Variance, And Arcing Classifiers. *Technical Report* TR 460, UC-Berkeley, CA.

Breiman, L. (2001). Random Forests. *Machine Learning* 45, 5–32. Available on: https://doi.org/10.1023/A:1010933404324.

Bühlmann, P. (2006). Boosting for high-dimensional linear models. *Ann. Statist*. 34 (2) 559 - 583. Available on: https://doi.org/10.1214/009053606000000092.

Chaochao et al. (2022). Innovative Materials Science via Machine Learning. Advanced Functional Materials. 32. 10.1002/adfm.202108044.

Chaudhuri, A. & De, K. (2011). Fuzzy Support Vector Machine for bankruptcy prediction, *Applied Soft Computing*, Volume 11, Issue 2, Pages 2472-2486, ISSN 1568-4946. Available on: https://doi.org/10.1016/j.asoc.2010.10.003.

Chawla, N., Bowyer, K., Hall, L. & Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Intell. Res. (JAIR). 16. 321-357. 10.1613/jair.953.

Chen, T., Guestrin, C. (2016). Xgboost: a scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD* international conference on knowledge discovery and data mining, KDD'16. ACM, New York, pp 785–794.

Choi, R., Y., Coyner, A., S., Kalpathy-Cramer, J., Chiang, M., F. & Campbell, J., P. (2020). Introduction to Machine Learning, Neural Networks, and Deep Learning. *Translational vision science & technology*, 9(2), 14. Available on: https://doi.org/10.1167/tvst.9.2.14.

Cortes, C.& Vapnik, V. (1995). Support-vector networks. Machine Learning 20, 273–297. Available on: https://doi.org/10.1007/BF00994018.

Cover, T. & Hart, P. (1967). Nearest neighbor pattern classification, *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21-27, doi: 10.1109/TIT.1967.1053964.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function, *Math. Control Signal Systems* 2, 303–314. Available on: https://doi.org/10.1007/BF02551274.

Devroye, L. & Wagner, T. Distribution-free performance bounds for potential function rules. In *IEEE Transactions on Information Theory*, vol. 25, no. 5, pp. 601-604, September 1979, doi: 10.1109/TIT.1979.1056087.

Freund, Y. & Schapire, R. (1996). Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning,* 148 156. Morgan Kaufmann.

Friedman, J., H.(2001). Greedy Function Approximation: A Gradient Boosting Machine. The *Annals of Statistics*, 29(5), 1189–1232. Available on: http://www.jstor.org/stable/2699986.

Genton, M., G., He, L. & Liu, X. (2001) Moments of Skew Normal Random Vectors and Their Quadratic Forms. *Statistics & Probability Letters*, 51, 319-325. Available on: http://dx.doi.org/10.1016/S0167-7152(00)00164-4

Grice, J., S. & Dugan, M.,T. (2001) The Limitations of Bankruptcy Prediction Models: Some Cautions for the Researcher. *Review of Quantitative Finance and Accounting* 17, 151–166. Available on: https://doi.org/10.1023/A:1017973604789.

Guo, G., Wang, H., Bell, D., Bi, Y. & Greer, K. (2003). KNN Model-Based Approach in Classification. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds) *On The Move to Meaningful Internet Systems 2003*: CoopIS, DOA, and ODBASE. OTM 2003. Lecture Notes in Computer Science, vol 2888. Springer, Berlin, Heidelberg. Available on: https://doi.org/10.1007/978-3-540-39964-3_62.

Halko, N., Martinsson, P., G. & Tropp, J., A. (2009) Finding Structure with Randomness: Stochastic Algorithms for Constructing Approximate matrix Decompositions. *ACM Technical Reports*, 2009-05. California Institute of Technology , Pasadena, CA. Available on: https://resolver.caltech.edu/CaltechAUTHORS:20111012-111324407.

Haneen et. Al. (2019). *Big Data*. 221-248. Available on: http://doi.org/10.1089/big.2018.0175.

Heckman, J. J. Sample Selection Bias as a Specification Error. *Econometric* 47: 153-61. Available on: https://doi.org/10.2307/1912352.

Hillegeist, S., A., Keating, E., K., Cram, D., P. et al. (2004). Assessing the Probability of Bankruptcy. *Review of Accounting Studies* 9, 5–34 (2004). https://doi.org/10.1023/B:RAST.0000013627.90884.b7.

Ho, T., K. (1995). Random Decision Forest. *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, Montreal, 278-282.

Hornik, K., Stinchcombe, M. & White, H. (1989). Multilayer feedforward networks are universal approximators, *Neural Networks*, Volume 2, Issue 5, Pages 359-366, ISSN 0893-6080. Available on: https://doi.org/10.1016/0893-6080(89)90020-8.

Imandoust, S., B. & Bolandraftar, M. (2013). Application of K-nearest neighbor (KNN) approach for predicting economic events theoretical background. *S B Imandoust et al. Int. Journal of Engineering Research and Applications* 3. 605-610.

Jabeur, B., S., Gharib, C., Mefteh-Wali, S. & Arfi, B., W. (2021). CatBoost model and artificial intelligence techniques for corporate failure prediction. *Technological Forecasting and Social Change*. 166. 120658. 10.1016/j.techfore.2021.120658.

Janiesch, C., Zschech, P. & Heinrich, K. (2021). Machine learning and deep learning, Electronic Markets, Springer; *IIM University of St. Gallen*, vol. 31(3), pages 685-695, September. Available on: https://doi.org/10.1007/s12525-021-00475-2.

Japkowicz, N. & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5): 429-449.

Kataria, A. & Singh, M. (2013). A Review of Data Classification Using K-Nearest Neighbour Algorithm, *International Journal of Emerging Technology and Advanced Engineering*. ISSN 2250-2459, ISO 9001:2008, Volume 3, Issue 6.

King, G. & Nielsen, M. (2019). Why Propensity Scores Should Not Be Used for Matching. *Political Analysis*, 27, 4, Pp. 435-454 Available on: https://tinyurl.com/y5b5yjxo.

Kingma, D. & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations.*

Kotsiantis, S., B. (2007). Supervised Machine Learning: A Review of Classification Techniques. In Proceedings of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, *HCI, Information Retrieval and Pervasive Technologies*. IOS Press, NLD, 3–24.

Ledoit, O. & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365 – 411.

Ledoit, O. & Wolf, M. (2017). Nonlinear shrinkage of the covariance matrix for portfolio selection: Markowitz meets goldilocks. *The Review of Financial Studies*, 30(12):4349–4388.

Lee, T., H., Ullah, A. & Wang, R. (2019). Bootstrap Aggregating and Random Forest, No 201918, Working Papers, *University of California at Riverside*, Department of Economics.

Lev, B. Financial Statement Analysis: A New Approach. Englewood Cliffs, N.J.: *Prentice-Hall*, 1974.

Lieberman, M. & Morris, J. (2014). The Precise Effect of Multicollinearity on Classification Prediction. 40. 5-10. Available on:

https://www.researchgate.net/publication/336561703_The_Precise_Effect_of_Multicollinearity_on_Classification_Prediction.

Lin C., F. & Wang, S., D. (2002). Fuzzy support vector machines, *in IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 464-471, doi: 10.1109/72.991432.

Maclin, R. & Opitz, D. (1998). An Empirical Evaluation of Bagging and Boosting. *Proceedings of the National Conference on Artificial Intelligence*.

Maddala, G., S. (1983). Limited-Dependent and Qualitative Variables in Econometrics. Cambridge: *Cambridge University Press*.

Maillo, J., Luengo, J., García, S., Herrera, F. & Triguero, I. (2017). Exact fuzzy k-nearest neighbor classification for big datasets, *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1-6, doi: 10.1109/FUZZ-IEEE.2017.8015686.

McCulloch, W., S. & Pitts, W., A. (1943). Logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* 5, 115–133. Available on: https://doi.org/10.1007/BF02478259.

Mensah, Y., M. (1984). An Examination of the Stationarity of Multivariate Bankruptcy Prediction Models: A Methodological Study. *Journal of Accounting Research*, 22(1), 380–395. Available on: https://doi.org/10.2307/2490719.

Murthy, S., K. (1998). Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. *Data Mining and Knowledge Discovery* 2, 345–389. Available on: https://doi.org/10.1023/A:1009744630224.

Natekin, A. & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7, 21. Available on: https://doi.org/10.3389/fnbot.2013.00021.

Noble, W. (2006). What is a support vector machine?. *Nature Biotechnology* 24, 1565–1567. Available on: https://doi.org/10.1038/nbt1206-1565.

Poirier, D., J. (1980). Partial Observability in Bivariate Probit Models. *Journal of Econometrics* 12: 209-17.

Prokhorenkova, L. O., Gusev, G., Vorobev, A., Dorogush, A., V. & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features., in Samy Bengio; Hanna M. Wallach; Hugo Larochelle; Kristen Grauman; Nicolò Cesa-Bianchi & Roman Garnett, ed., '*NeurIPS*' , pp. 6639-6649.

Shetty, S., Musa, M. & Brédart., X. (2022). Bankruptcy Prediction Using Machine Learning Techniques. *Journal of Risk and Financial Management* 15, no. 1: 35. Available on: https://doi.org/10.3390/jrfm15010035.

Shi, Y. & Li, X. (2019). An overview of bankruptcy prediction models for corporate firms: A systematic literature review. *Intangible Capital*, 15(2), 114-127. Available on: https://doi.org/10.3926/ic.1354.

Shin, K., S., Lee, T. & Kim, H., J. (2005). An Application of Support Vector Machines in Bankruptcy Prediction Model. *Expert Systems with Applications*. 28. 127-135. 10.1016/j.eswa.2004.08.009.

Shumway, T. (2001). Forecasting Bankruptcy More Accurately: A Simple Hazard Model. *Journal of Business* 74:1, 101–124.

Sollich, P. & Krogh, A. (1996). Learning with ensembles: How over-fitting can be useful. In Touretsky, D.; Mozer, M., and Hasselmo, M., eds., *Advances in Neural Information Processing System,* volume 8, 190-196. Cambridge, MA: MIT Press.

Tay, F., E., H. & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega,* 29, 309–317. Available on: https://doi.org/10.1016/S0305-0483(01)00026-3.

Tomek, I. (1976) Two Modifications of CNN. IEEE Transactions on Systems Man and Communications, 6, 769-772. Available on: http://dx.doi.org/10.1109/TSMC.1976.4309452.

Tseng, F.,M. & H, Y., C. (2010). Comparing four bankruptcy prediction models: Logit, quadratic interval logit, neural and fuzzy neural networks, *Expert Systems with Applications*, Volume 37, Issue 3, Pages 1846-1853, ISSN 0957-4174. Available on: https://doi.org/10.1016/j.eswa.2009.07.081.

Wruck, K. H. (1990). Financial distress, reorganization, and organizational efficiency. *Journal of Financial Economics*, 27(2), 419–444. Available on: https://doi.org/10.1016/0304-405X(90)90063-6.

Zhang, G., Patuwo, B., E. & Hu, M., Y. (1998). Forecasting with artificial neural networks: The state of the art, *International Journal of Forecasting*, Volume 14, Issue 1, 1998, Pages 35-62, ISSN 0169-2070. Available on: https://doi.org/10.1016/S0169-2070(97)00044-7.

Zhang, W., (2017). Machine Learning Approaches to Predicting Company Bankruptcy. *Journal of Financial Risk Management*, 6, 364-374. Available on: https://doi.org/10.4236/jfrm.2017.64026.

Zhang, X. (1994) Time series analysis and prediction by neural networks, *Optimization Methods and Software,* 4:2, 151-170. Available on: https://doi.org/10.1080/10556789408805584.

**Sites**

https://www.appforfinance.com

https://advisory.kpmg.us/articles/2020/going-concern.html#:~:text=Going%20concern%20%E2%80%93%20the%20underlying%20basis,the%20company%20or%20stop%20trading.

https://github.com

https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks

https://www.investopedia.com

https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks

https://www.oreilly.com/library/view/deep-learning/9781491924570/ch01.html

https://www.registroimprese.it/bilancio-d-esercizio

https://www.statistics.com

https://www.spglobal.com/marketintelligence/en/news-insights/latest-news-headlines/us-corporate-bankruptcies-end-2020-at-10-year-high-amid-covid-19-pandemic-61973656

https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb

https://www.youtube.com/StatQuest

https://ec.europa.eu/info/business-economy-euro/economic-performance-and-forecasts/economic-performance-country/italy/economic-forecast-italy_en

**Appendix 1: Results for the 2021 group adding the PCA procedure**

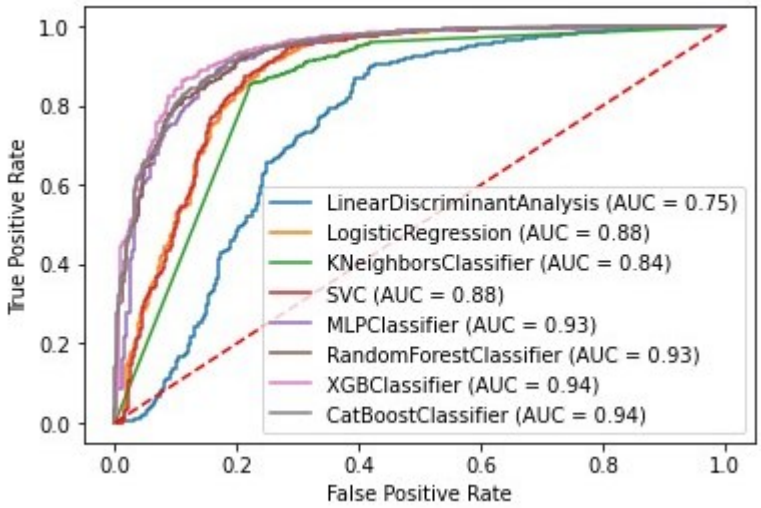


Figure 1: AUC-ROC curve comparison at 1 year with PCA.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.496154 | 0.387387 | 0.933938 | 0.956551 |
| Logit | 0.479691 | 0.744745 | 0.969975 | 0.910779 |
| KNN | 0.45098 | 0.690691 | 0.963707 | 0.907131 |
| SVM | 0.577295 | 0.717718 | 0.967962 | 0.941957 |
| NN | 0.406609 | 0.84985 | 0.981146 | 0.863018 |
| RF | 0.465969 | 0.801802 | 0.976216 | 0.898507 |
| XGBoost | 0.499099 | 0.831832 | 0.97995 | 0.907794 |
| CatBoost | 0.452769 | 0.834835 | 0.979883 | 0.888557 |

Table 1: Unweighted general performances measures at 1 year with PCA.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.89994 | 0.715046 | 0.671969 | 0.690092 | 0.678546 |
| Logit | 0.894265 | 0.724833 | 0.827762 | 0.761488 | 0.796333 |
| KNN | 0.885603 | 0.707344 | 0.798911 | 0.740117 | 0.771115 |
| SVM | 0.919654 | 0.772628 | 0.829837 | 0.797338 | 0.815734 |
| NN | 0.861708 | 0.693878 | 0.856434 | 0.734174 | 0.791022 |
| RF | 0.888889 | 0.721092 | 0.850155 | 0.762578 | 0.806915 |
| XGBoost | 0.900239 | 0.739524 | 0.869813 | 0.783183 | 0.827666 |
| CatBoost | 0.883214 | 0.716326 | 0.861696 | 0.759553 | 0.80986 |

Table 2: Performances measures for each class at 1 year with PCA.

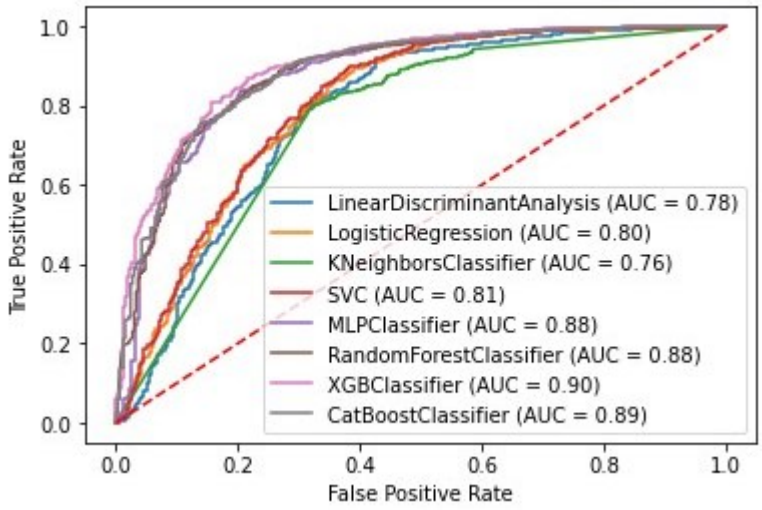


Figure 2: AUC-ROC curve comparison at 2 years with PCA.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.901434 | 0.72104 | 0.684819 | 0.700629 | 0.690672 |
| Logit | 0.821685 | 0.633557 | 0.746058 | 0.657783 | 0.697901 |
| KNN | 0.845281 | 0.639247 | 0.713747 | 0.662451 | 0.688376 |
| SVM | 0.83184 | 0.642653 | 0.754368 | 0.669352 | 0.70896 |
| NN | 0.848865 | 0.671317 | 0.809233 | 0.705489 | 0.753959 |
| RF | 0.853345 | 0.673827 | 0.805042 | 0.708264 | 0.754269 |
| XGBoost | 0.862007 | 0.686785 | 0.82588 | 0.724339 | 0.77293 |
| CatBoost | 0.823775 | 0.653113 | 0.804652 | 0.681424 | 0.735036 |

Table 3: Unweighted general performances measures at 2 years with PCA.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.505495 | 0.414414 | 0.936585 | 0.955224 |
| Logit | 0.310888 | 0.651652 | 0.956226 | 0.840464 |
| KNN | 0.332123 | 0.54955 | 0.946371 | 0.877944 |
| SVM | 0.327844 | 0.657658 | 0.957463 | 0.851078 |
| NN | 0.372607 | 0.75976 | 0.970026 | 0.858706 |
| RF | 0.379205 | 0.744745 | 0.968448 | 0.86534 |
| XGBoost | 0.400616 | 0.780781 | 0.972953 | 0.870978 |
| CatBoost | 0.33462 | 0.780781 | 0.971606 | 0.828524 |

Table 4: Performances measures for each class at 2 years with PCA.

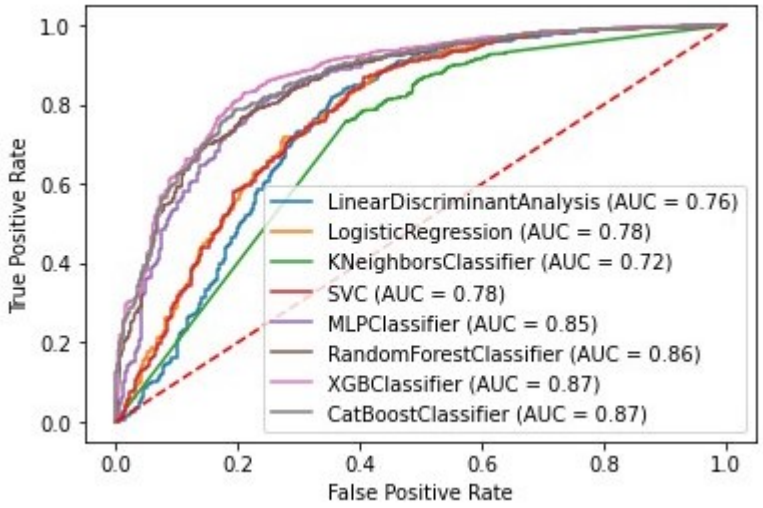


Figure 3: AUC-ROC curve comparison at 3 years with PCA.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.89307 | 0.697517 | 0.681511 | 0.689016 | 0.684402 |
| Logit | 0.773895 | 0.601749 | 0.715517 | 0.61191 | 0.653214 |
| KNN | 0.822879 | 0.610961 | 0.681274 | 0.629426 | 0.654482 |
| SVM | 0.763441 | 0.598347 | 0.715055 | 0.605103 | 0.647466 |
| NN | 0.782855 | 0.626377 | 0.779261 | 0.640902 | 0.69516 |
| RF | 0.799283 | 0.629869 | 0.769683 | 0.649651 | 0.699519 |
| XGBoost | 0.827658 | 0.65564 | 0.805473 | 0.685021 | 0.738003 |
| CatBoost | 0.79092 | 0.633 | 0.790417 | 0.650533 | 0.706288 |

Table 5: Unweighted general performances measures at 3 years with PCA.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.458746 | 0.417417 | 0.936289 | 0.945605 |
| Logit | 0.251174 | 0.642643 | 0.952324 | 0.788391 |
| KNN | 0.281879 | 0.504505 | 0.940044 | 0.858043 |
| SVM | 0.243575 | 0.654655 | 0.953119 | 0.775456 |
| NN | 0.283516 | 0.774775 | 0.969237 | 0.783748 |
| RF | 0.295042 | 0.732733 | 0.964697 | 0.806633 |
| XGBoost | 0.339895 | 0.777778 | 0.971384 | 0.833167 |
| CatBoost | 0.294513 | 0.78979 | 0.971487 | 0.791045 |

Table 6: Performances measures for each class at 3 years with PCA.

## Appendix 2: Correlation matrices for the 2020 and 2019 groups

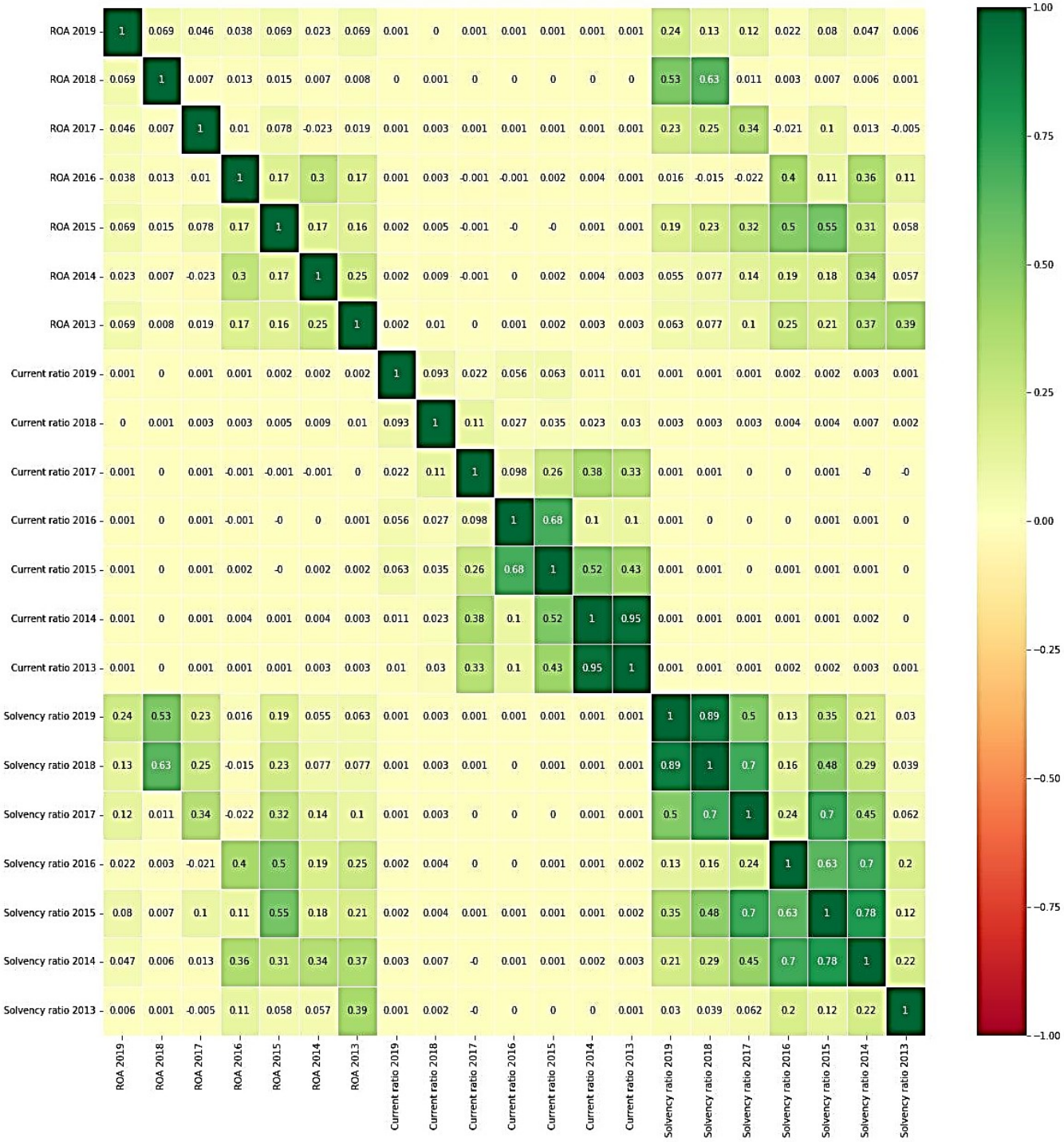


Figure 4: Correlation Matrix for dataset of firms in 2020.

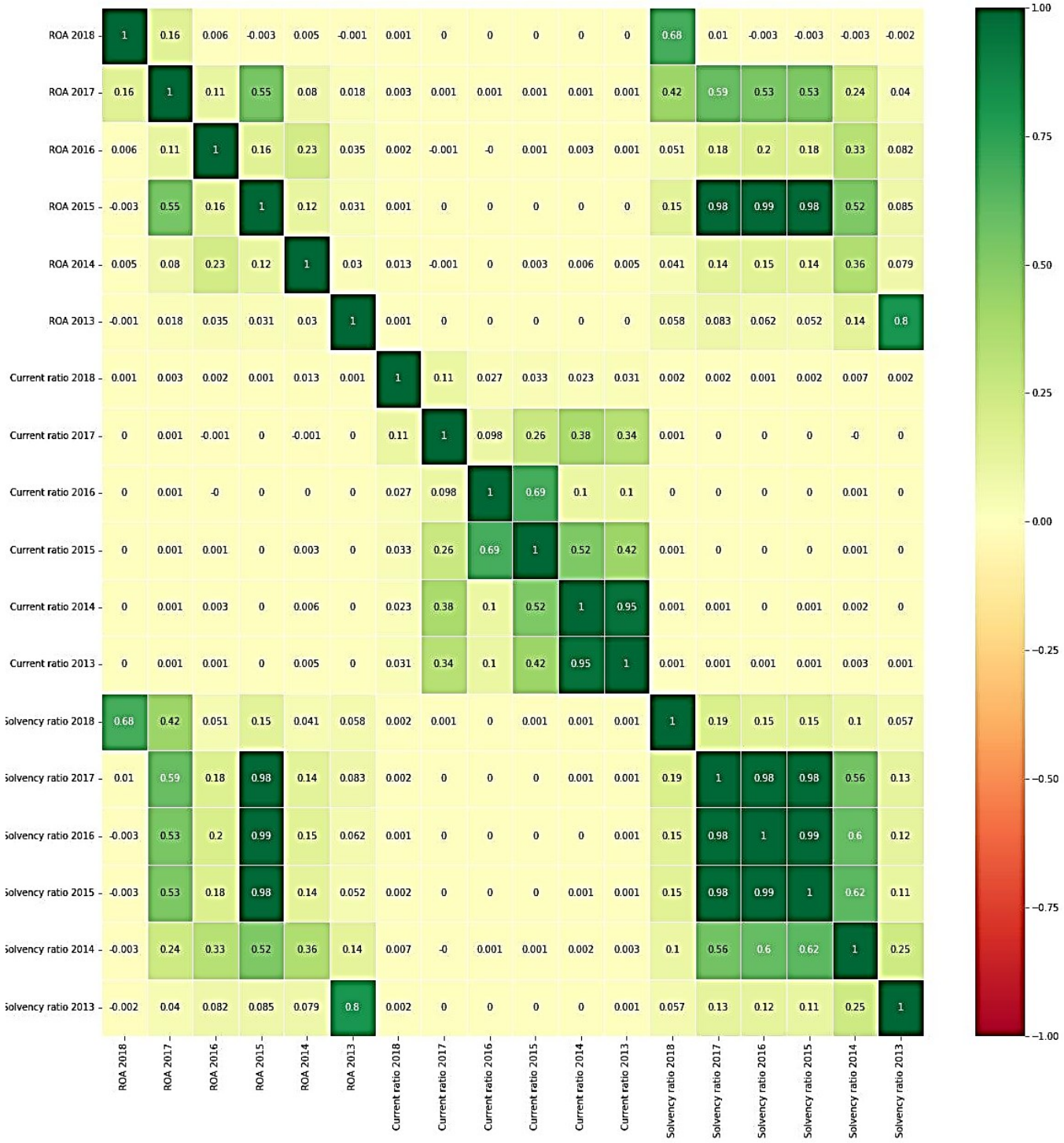


Figure 5: Correlation Matrix for dataset of firms in 2019.

## Appendix 3:Univariate logistic regression for the 2020 and 2019 groups

| Ratio | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| ROA 2019 | 0.931159 | 0.879556 | 0.634643 | 0.687905 | 0.649486 |
| ROA 2018 | 0.913043 | 0.746959 | 0.517576 | 0.512677 | 0.513101 |
| ROA 2017 | 0.913647 | 0.885455 | 0.510144 | 0.497529 | 0.50342 |
| ROA 2016 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| ROA 2015 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| ROA 2014 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| ROA 2013 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| CR 2019 | 0.911534 | 0.581167 | 0.501222 | 0.480242 | 0.49233 |
| CR 2018 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| CR 2017 | 0.909722 | 0.455962 | 0.498676 | 0.476364 | 0.489505 |
| CR 2016 | 0.910628 | 0.456002 | 0.499172 | 0.476612 | 0.489897 |
| CR 2015 | 0.908514 | 0.491748 | 0.499567 | 0.479303 | 0.491005 |
| CR 2014 | 0.90942 | 0.455949 | 0.49851 | 0.476281 | 0.489374 |
| CR 2013 | 0.909118 | 0.497727 | 0.499898 | 0.479491 | 0.49127 |
| SR 2019 | 0.923309 | 0.908677 | 0.571337 | 0.604057 | 0.577942 |
| SR 2018 | 0.916063 | 0.879297 | 0.526995 | 0.529598 | 0.52448 |
| SR 2017 | 0.91244 | 0.956207 | 0.501718 | 0.480525 | 0.492727 |
| SR 2016 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| SR 2015 | 0.91244 | 0.956207 | 0.501718 | 0.480525 | 0.492727 |
| SR 2014 | 0.912138 | 0.456069 | 0.5 | 0.477025 | 0.490549 |
| SR 2013 | 0.91244 | 0.956207 | 0.501718 | 0.480525 | 0.492727 |

Table 7: Unweighted Accuracy, Precision, Recall, F1 and F2 scores for all ratios of the 2020 group.

| Ratio | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| ROA 2019 | 0.824742 | 0.274914 | 0.93437 | 0.994373 |
| ROA 2018 | 0.578947 | 0.037801 | 0.914971 | 0.997352 |
| ROA 2017 | 0.857143 | 0.020619 | 0.913767 | 0.999669 |
| ROA 2016 | 0.0 | 0.0 | 0.912138 | 1.0 |
| ROA 2015 | 0.0 | 0.0 | 0.912138 | 1.0 |
| ROA 2014 | 0.0 | 0.0 | 0.912138 | 1.0 |
| ROA 2013 | 0.0 | 0.0 | 0.912138 | 1.0 |
| CR 2019 | 0.25 | 0.003436 | 0.912334 | 0.999007 |
| CR 2018 | 0.0 | 0.0 | 0.912138 | 1.0 |
| CR 2017 | 0.0 | 0.0 | 0.911925 | 0.997352 |
| CR 2016 | 0.0 | 0.0 | 0.912005 | 0.998345 |
| CR 2015 | 0.071429 | 0.003436 | 0.912068 | 0.995697 |
| CR 2014 | 0.0 | 0.0 | 0.911898 | 0.997021 |
| CR 2013 | 0.083333 | 0.003436 | 0.912121 | 0.996359 |
| SR 2019 | 0.893617 | 0.14433 | 0.923737 | 0.998345 |
| SR 2018 | 0.842105 | 0.054983 | 0.91649 | 0.999007 |
| SR 2017 | 1.0 | 0.003436 | 0.912413 | 1.0 |
| SR 2016 | 0.0 | 0.0 | 0.912138 | 1.0 |
| SR 2015 | 1.0 | 0.003436 | 0.912413 | 1.0 |
| SR 2014 | 0.0 | 0.0 | 0.912138 | 1.0 |
| SR 2013 | 1.0 | 0.003436 | 0.912413 | 1.0 |

Table 8: Precision and Recall scores for the two classes of the 2020 group.

| Ratio | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| ROA 2018 | 0.92439 | 0.914639 | 0.535517 | 0.546669 | 0.53565 |
| ROA 2017 | 0.923476 | 0.793081 | 0.552229 | 0.573774 | 0.555995 |
| ROA 2016 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| ROA 2015 | 0.919512 | 0.745843 | 0.50705 | 0.493714 | 0.50047 |
| ROA 2014 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| ROA 2013 | 0.918902 | 0.626386 | 0.501555 | 0.482593 | 0.493484 |
| CR 2018 | 0.92439 | 0.914639 | 0.535517 | 0.546669 | 0.53565 |
| CR 2017 | 0.923476 | 0.793081 | 0.552229 | 0.573774 | 0.555995 |
| CR 2016 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| CR 2015 | 0.919512 | 0.745843 | 0.50705 | 0.493714 | 0.50047 |
| CR 2014 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| CR 2013 | 0.918902 | 0.626386 | 0.501555 | 0.482593 | 0.493484 |
| SR 2018 | 0.93628 | 0.913117 | 0.619428 | 0.673118 | 0.633712 |
| SR 2017 | 0.923476 | 0.930453 | 0.528136 | 0.533394 | 0.526564 |
| SR 2016 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| SR 2015 | 0.920122 | 0.960024 | 0.50566 | 0.490374 | 0.498514 |
| SR 2014 | 0.919207 | 0.459604 | 0.5 | 0.478952 | 0.491362 |
| SR 2013 | 0.918902 | 0.459591 | 0.499834 | 0.478869 | 0.491232 |

Table 9: Unweighted Accuracy, Precision, Recall, F1 and F2 scores for all ratios of the 2019 group.

| Ratio | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| ROA 2018 | 0.904762 | 0.071698 | 0.924517 | 0.999337 |
| ROA 2017 | 0.659091 | 0.109434 | 0.92707 | 0.995025 |
| ROA 2016 | 0.0 | 0.0 | 0.919207 | 1.0 |
| ROA 2015 | 0.571429 | 0.015094 | 0.920257 | 0.999005 |
| ROA 2014 | 0.0 | 0.0 | 0.919207 | 1.0 |
| ROA 2013 | 0.333333 | 0.003774 | 0.919439 | 0.999337 |
| CR 2018 | 0.904762 | 0.071698 | 0.924517 | 0.999337 |
| CR 2017 | 0.659091 | 0.109434 | 0.92707 | 0.995025 |
| CR 2016 | 0.0 | 0.0 | 0.919207 | 1.0 |
| CR 2015 | 0.571429 | 0.015094 | 0.920257 | 0.999005 |
| CR 2014 | 0.0 | 0.0 | 0.919207 | 1.0 |
| CR 2013 | 0.333333 | 0.003774 | 0.919439 | 0.999337 |
| SR 2018 | 0.888889 | 0.241509 | 0.937344 | 0.997347 |
| SR 2017 | 0.9375 | 0.056604 | 0.923407 | 0.999668 |
| SR 2016 | 0.0 | 0.0 | 0.919207 | 1.0 |
| SR 2015 | 1.0 | 0.011321 | 0.920049 | 1.0 |
| SR 2014 | 0.0 | 0.0 | 0.919207 | 1.0 |
| SR 2013 | 0.0 | 0.0 | 0.919183 | 0.999668 |

Table 10: Precision and Recall scores for the two classes of the 2019 group.

**Appendix 4: Results without SMOTE and Tomek link for the 2021 group**

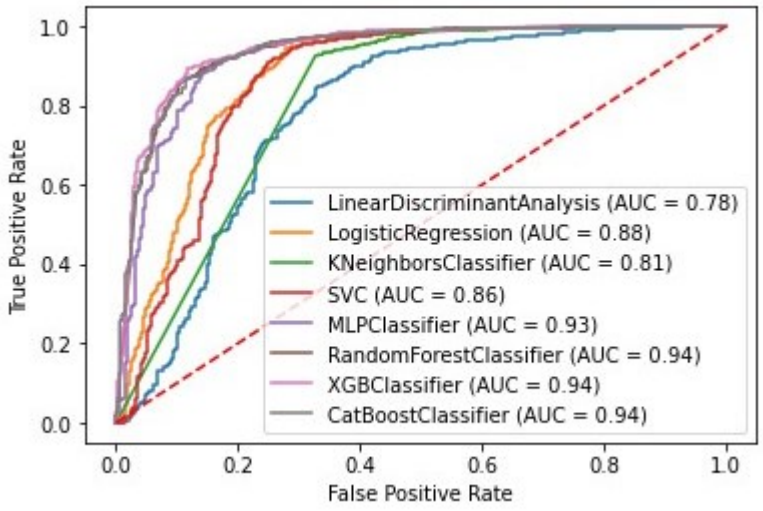


Figure 6: AUC-ROC curve comparison at 1 year without SMOTE and Tomek link.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.905317 | 0.884614 | 0.528031 | 0.528525 | 0.524487 |
| Logit | 0.935185 | 0.934059 | 0.686195 | 0.750247 | 0.705178 |
| KNN | 0.937276 | 0.874451 | 0.738111 | 0.787059 | 0.754621 |
| SVM | 0.932497 | 0.940724 | 0.668675 | 0.731903 | 0.686714 |
| NN | 0.944743 | 0.889044 | 0.775649 | 0.820038 | 0.791236 |
| RF | 0.943548 | 0.879567 | 0.777657 | 0.818496 | 0.792157 |
| XGBoost | 0.948925 | 0.895043 | 0.798005 | 0.837847 | 0.812324 |
| CatBoost | 0.945042 | 0.880127 | 0.787836 | 0.825793 | 0.801486 |

Table 11: Unweighted general performances measures at 1 year without SMOTE and Tomek link.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.863636 | 0.057057 | 0.905592 | 0.999005 |
| Logit | 0.932836 | 0.375375 | 0.935283 | 0.997015 |
| KNN | 0.802956 | 0.489489 | 0.945946 | 0.986733 |
| SVM | 0.94958 | 0.339339 | 0.931867 | 0.99801 |
| NN | 0.824561 | 0.564565 | 0.953526 | 0.986733 |
| RF | 0.805085 | 0.570571 | 0.954049 | 0.984743 |
| XGBoost | 0.831967 | 0.60961 | 0.958119 | 0.986401 |
| CatBoost | 0.804082 | 0.591592 | 0.956171 | 0.98408 |

Table 12: Performances measures for each class at 1 year without SMOTE and Tomek link.

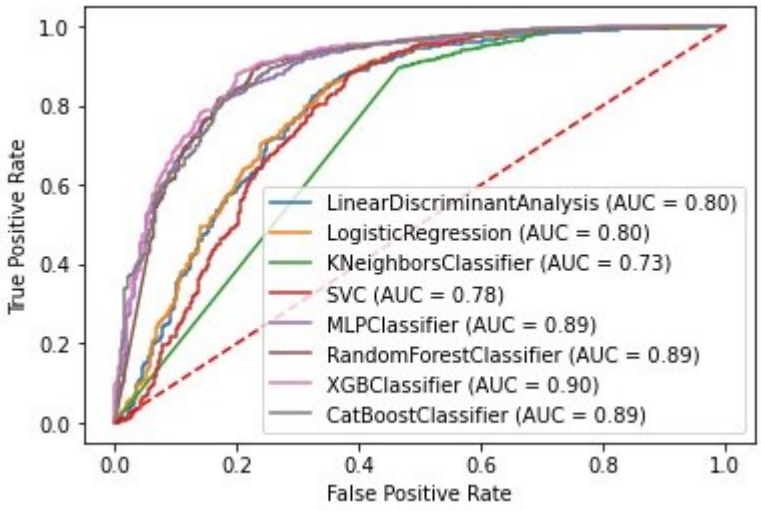


Figure 7: AUC-ROC curve comparison at 2 years without SMOTE and Tomek link.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.907407 | 0.889358 | 0.539877 | 0.549697 | 0.539053 |
| Logit | 0.913381 | 0.883485 | 0.576586 | 0.608971 | 0.582996 |
| KNN | 0.916368 | 0.802665 | 0.643691 | 0.687014 | 0.656599 |
| SVM | 0.910394 | 0.897231 | 0.556228 | 0.57733 | 0.55884 |
| NN | 0.925329 | 0.829643 | 0.694079 | 0.739376 | 0.708849 |
| RF | 0.926225 | 0.869282 | 0.665192 | 0.719055 | 0.681174 |
| XGBoost | 0.925627 | 0.841119 | 0.68356 | 0.732382 | 0.698982 |
| CatBoost | 0.923835 | 0.835611 | 0.674551 | 0.722893 | 0.689611 |

Table 13: Unweighted general performances measures at 2 years without SMOTE and Tomek link.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.870968 | 0.081081 | 0.907748 | 0.998673 |
| Logit | 0.852459 | 0.156156 | 0.914512 | 0.997015 |
| KNN | 0.677852 | 0.303303 | 0.927477 | 0.98408 |
| SVM | 0.883721 | 0.114114 | 0.910741 | 0.998342 |
| NN | 0.721925 | 0.405405 | 0.937362 | 0.982753 |
| RF | 0.807143 | 0.339339 | 0.931421 | 0.991045 |
| XGBoost | 0.747059 | 0.381381 | 0.935179 | 0.985738 |
| CatBoost | 0.737805 | 0.363363 | 0.933417 | 0.985738 |

Table 14: Performances measures for each class at 2 years without SMOTE and Tomek link.

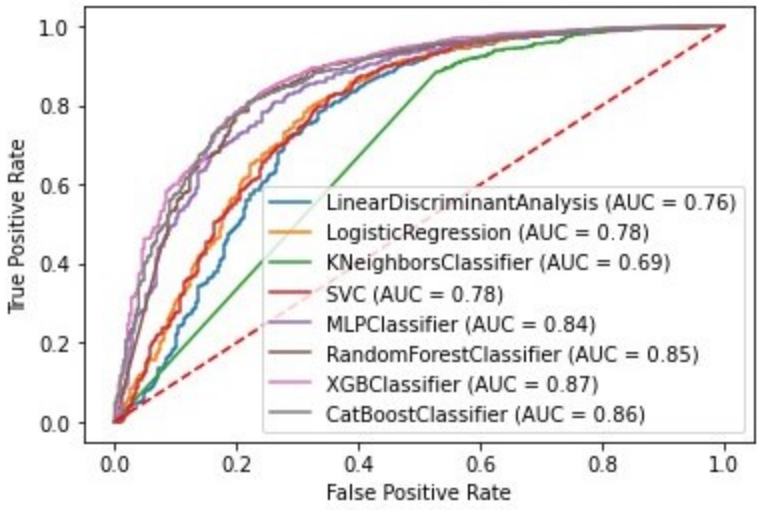


Figure 8: AUC-ROC curve comparison at 3 years without SMOTE and Tomek link.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.907706 | 0.870924 | 0.54405 | 0.556882 | 0.544188 |
| Logit | 0.910394 | 0.8462 | 0.566913 | 0.593553 | 0.571631 |
| KNN | 0.905914 | 0.741628 | 0.61251 | 0.645856 | 0.622091 |
| SVM | 0.905317 | 0.869585 | 0.529367 | 0.531018 | 0.52618 |
| NN | 0.911888 | 0.802626 | 0.599798 | 0.637361 | 0.609263 |
| RF | 0.914875 | 0.823726 | 0.610806 | 0.652624 | 0.621711 |
| XGBoost | 0.918459 | 0.825226 | 0.63951 | 0.685795 | 0.652846 |
| CatBoost | 0.915472 | 0.811186 | 0.62583 | 0.669034 | 0.637902 |

Table 15: Unweighted general performances measures at 3 years without SMOTE and Tomek link.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.833333 | 0.09009 | 0.908514 | 0.99801 |
| Logit | 0.779661 | 0.138138 | 0.912739 | 0.995688 |
| KNN | 0.561644 | 0.246246 | 0.921611 | 0.978773 |
| SVM | 0.833333 | 0.06006 | 0.905836 | 0.998673 |
| NN | 0.686275 | 0.21021 | 0.918977 | 0.989386 |
| RF | 0.726415 | 0.231231 | 0.921036 | 0.990381 |
| XGBoost | 0.723881 | 0.291291 | 0.926571 | 0.987728 |
| CatBoost | 0.698413 | 0.264264 | 0.92396 | 0.987396 |

Table 16: Performances measures for each class at 3 years without SMOTE and Tomek link.

## Appendix 5: Results for the 2020 and 2019 groups

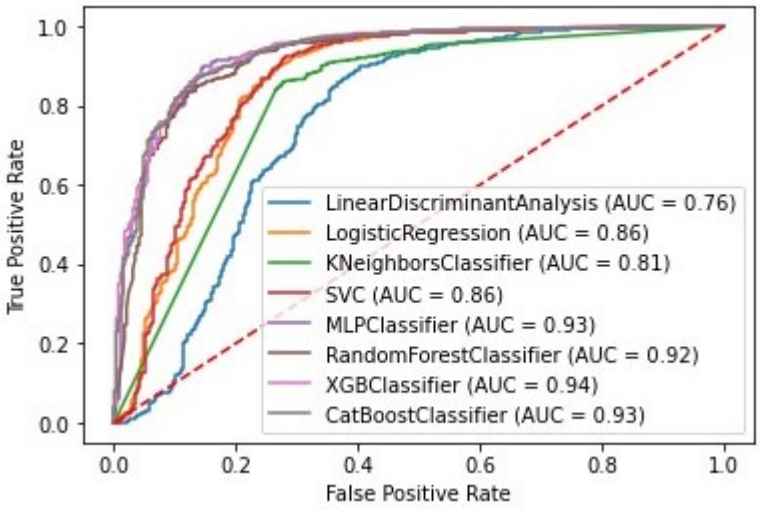


Figure 9: AUC-ROC curve comparison at 1 year of the 2020 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.912138 | 0.722894 | 0.694088 | 0.707134 | 0.699018 |
| Logit | 0.868659 | 0.674183 | 0.813104 | 0.711919 | 0.760202 |
| KNN | 0.885568 | 0.683766 | 0.774238 | 0.715403 | 0.746125 |
| SVM | 0.903986 | 0.720531 | 0.820047 | 0.756963 | 0.790424 |
| NN | 0.876812 | 0.694653 | 0.867259 | 0.739158 | 0.798972 |
| RF | 0.855978 | 0.673259 | 0.852734 | 0.712187 | 0.774527 |
| XGBoost | 0.88285 | 0.700248 | 0.864358 | 0.745281 | 0.802059 |
| CatBoost | 0.866244 | 0.684731 | 0.866124 | 0.727086 | 0.790014 |

Table 17: Unweighted general performances measures at 1 year of the 2020 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.5 | 0.429553 | 0.945787 | 0.958623 |
| Logit | 0.375433 | 0.745704 | 0.972933 | 0.880503 |
| KNN | 0.404348 | 0.639175 | 0.963184 | 0.909302 |
| SVM | 0.469663 | 0.718213 | 0.971399 | 0.92188 |
| NN | 0.404878 | 0.85567 | 0.984427 | 0.878848 |
| RF | 0.363235 | 0.848797 | 0.983283 | 0.85667 |
| XGBoost | 0.417376 | 0.841924 | 0.983119 | 0.886792 |
| CatBoost | 0.384146 | 0.865979 | 0.985316 | 0.866269 |

Table 18: Performances measures for each class at 1 year of the 2020 group.

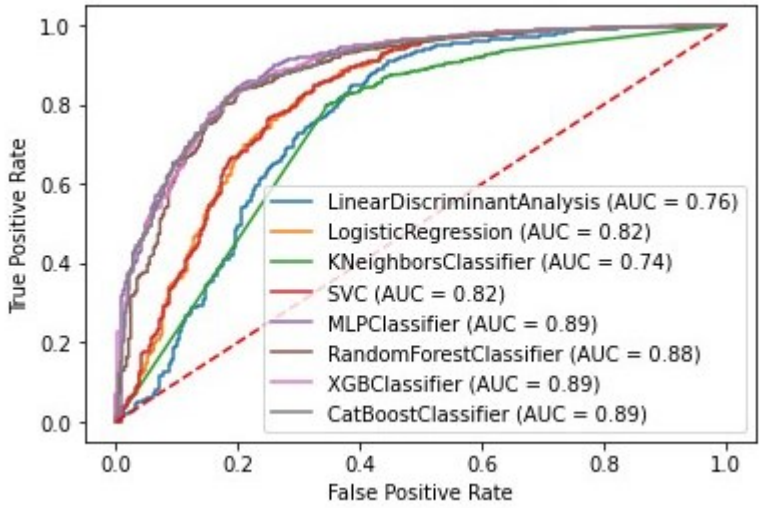


Figure 10: AUC-ROC curve comparison at 2 years of the 2020 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.888285 | 0.675128 | 0.724489 | 0.695228 | 0.711426 |
| Logit | 0.794384 | 0.611497 | 0.755309 | 0.62691 | 0.677364 |
| KNN | 0.841787 | 0.62201 | 0.712975 | 0.645678 | 0.677744 |
| SVM | 0.804046 | 0.615585 | 0.755947 | 0.634148 | 0.683538 |
| NN | 0.828502 | 0.645945 | 0.817487 | 0.675046 | 0.734742 |
| RF | 0.835749 | 0.649122 | 0.813696 | 0.680162 | 0.737577 |
| XGBoost | 0.854771 | 0.661782 | 0.811701 | 0.697251 | 0.749581 |
| CatBoost | 0.833333 | 0.647729 | 0.813925 | 0.678092 | 0.736045 |

Table 19: Unweighted general performances measures at 2 years of the 2020 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.397403 | 0.525773 | 0.952853 | 0.923204 |
| Logit | 0.256858 | 0.707904 | 0.966135 | 0.802714 |
| KNN | 0.290844 | 0.556701 | 0.953176 | 0.869249 |
| SVM | 0.265707 | 0.697595 | 0.965463 | 0.8143 |
| NN | 0.314094 | 0.804124 | 0.977795 | 0.830851 |
| RF | 0.322082 | 0.786942 | 0.976163 | 0.84045 |
| XGBoost | 0.349684 | 0.75945 | 0.973881 | 0.863952 |
| CatBoost | 0.319001 | 0.790378 | 0.976457 | 0.837471 |

Table 20: Performances measures for each class at 2 years of the 2020 group.

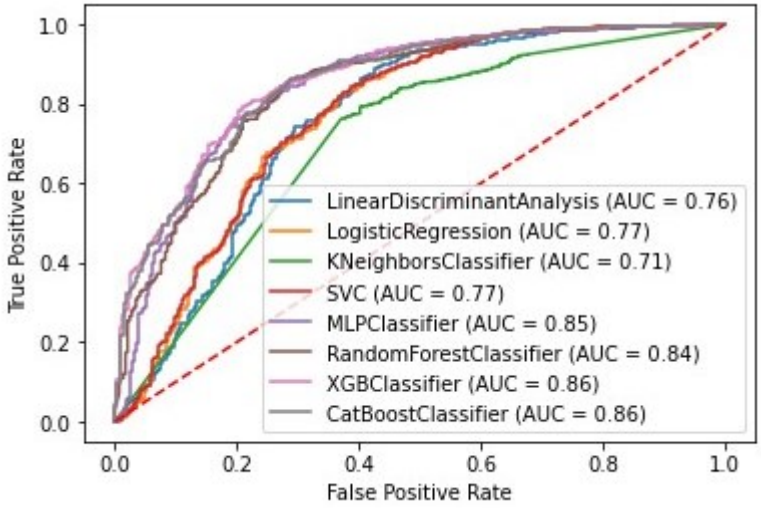


Figure 11: AUC-ROC curve comparison at 3 years of the 2020 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.871679 | 0.65232 | 0.723149 | 0.677104 | 0.70121 |
| Logit | 0.744867 | 0.582508 | 0.709533 | 0.578968 | 0.623438 |
| KNN | 0.82186 | 0.595685 | 0.674103 | 0.612669 | 0.640805 |
| SVM | 0.749698 | 0.58493 | 0.713734 | 0.583279 | 0.628386 |
| NN | 0.811896 | 0.625703 | 0.77733 | 0.647721 | 0.700825 |
| RF | 0.804348 | 0.621545 | 0.774745 | 0.641023 | 0.694591 |
| XGBoost | 0.835447 | 0.640891 | 0.785582 | 0.67003 | 0.720788 |
| CatBoost | 0.806763 | 0.624409 | 0.780727 | 0.644924 | 0.69952 |

Table 21: Unweighted general performances measures at 3 years of the 2020 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.351111 | 0.542955 | 0.953529 | 0.903343 |
| Logit | 0.205945 | 0.666667 | 0.959072 | 0.7524 |
| KNN | 0.245315 | 0.494845 | 0.946055 | 0.85336 |
| SVM | 0.210129 | 0.670103 | 0.959732 | 0.757365 |
| NN | 0.281579 | 0.735395 | 0.969828 | 0.819265 |
| RF | 0.273189 | 0.738832 | 0.969901 | 0.810659 |
| XGBoost | 0.31213 | 0.725086 | 0.969651 | 0.846077 |
| CatBoost | 0.277707 | 0.749141 | 0.971112 | 0.812314 |

Table 22: Performances measures for each class at 3 years of the 2020 group.

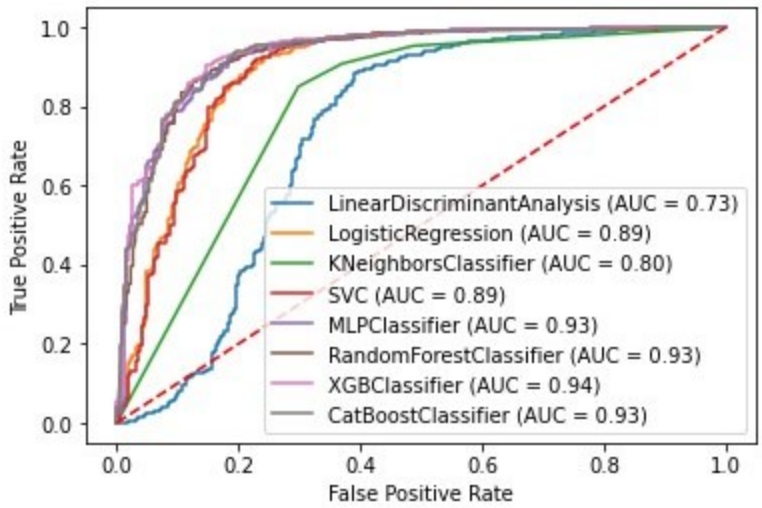


Figure 12: AUC-ROC curve comparison at 1 year of the 2019 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.921037 | 0.731132 | 0.676532 | 0.699062 | 0.68464 |
| Logit | 0.880488 | 0.680725 | 0.835176 | 0.722761 | 0.776138 |
| KNN | 0.885976 | 0.671216 | 0.769323 | 0.703921 | 0.737434 |
| SVM | 0.889024 | 0.690912 | 0.838099 | 0.733998 | 0.784692 |
| NN | 0.889329 | 0.695909 | 0.860637 | 0.741892 | 0.798665 |
| RF | 0.895122 | 0.703797 | 0.865509 | 0.750834 | 0.806455 |
| XGBoost | 0.906402 | 0.720962 | 0.876808 | 0.769715 | 0.823043 |
| CatBoost | 0.889024 | 0.696219 | 0.863913 | 0.742563 | 0.800364 |

Table 23: Unweighted general performances measures at 1 year of the 2019 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.515152 | 0.384906 | 0.947112 | 0.968159 |
| Logit | 0.382625 | 0.781132 | 0.978824 | 0.889221 |
| KNN | 0.376975 | 0.630189 | 0.965456 | 0.908458 |
| SVM | 0.403131 | 0.777358 | 0.978693 | 0.898839 |
| NN | 0.408582 | 0.826415 | 0.983236 | 0.894859 |
| RF | 0.423892 | 0.830189 | 0.983702 | 0.900829 |
| XGBoost | 0.456967 | 0.841509 | 0.984957 | 0.912106 |
| CatBoost | 0.408503 | 0.833962 | 0.983936 | 0.893864 |

Table 24: Performances measures for each class at 1 year of the 2019 group.

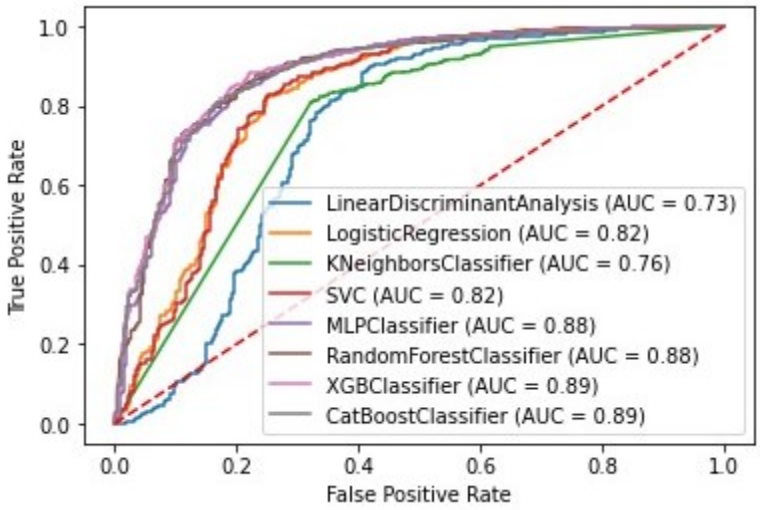


Figure 13: AUC-ROC curve comparison at 2 years of the 2019 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.920732 | 0.729668 | 0.672925 | 0.69608 | 0.68121 |
| Logit | 0.816463 | 0.621909 | 0.784862 | 0.64428 | 0.700528 |
| KNN | 0.856707 | 0.62375 | 0.7121 | 0.648664 | 0.679517 |
| SVM | 0.807622 | 0.61712 | 0.781774 | 0.63635 | 0.692963 |
| NN | 0.841159 | 0.643834 | 0.815504 | 0.675657 | 0.735021 |
| RF | 0.858232 | 0.655899 | 0.816186 | 0.692197 | 0.747814 |
| XGBoost | 0.87439 | 0.673416 | 0.830139 | 0.714086 | 0.768338 |
| CatBoost | 0.850915 | 0.650334 | 0.815648 | 0.684841 | 0.742155 |

Table 25: Unweighted general performances measures at 2 years of the 2019 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.512821 | 0.377358 | 0.946515 | 0.968491 |
| Logit | 0.270123 | 0.74717 | 0.973695 | 0.822554 |
| KNN | 0.291242 | 0.539623 | 0.956257 | 0.884577 |
| SVM | 0.260471 | 0.750943 | 0.973768 | 0.812604 |
| NN | 0.309524 | 0.784906 | 0.978144 | 0.846103 |
| RF | 0.334983 | 0.766038 | 0.976814 | 0.866335 |
| XGBoost | 0.368515 | 0.777358 | 0.978317 | 0.882919 |
| CatBoost | 0.323344 | 0.773585 | 0.977324 | 0.857711 |

Table 26: Performances measures for each class at 2 years of the 2019 group.

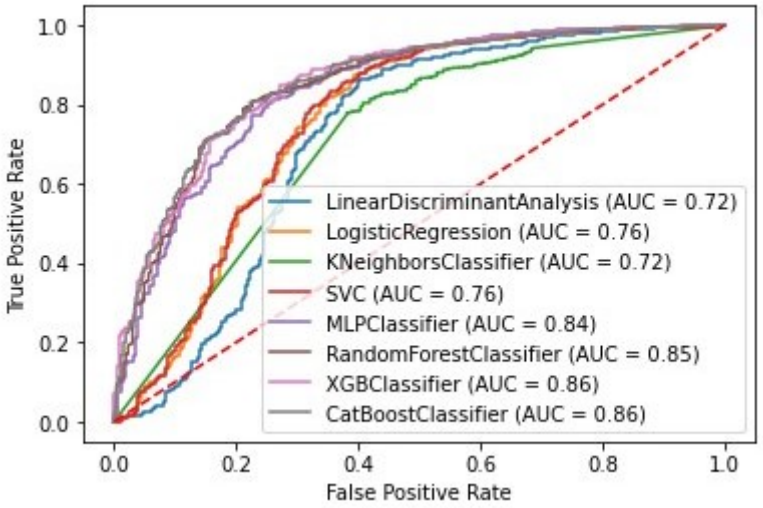


Figure 14: AUC-ROC curve comparison at 3 years of the 2019 group.

| Models | Accuracy | Precision | Recall | F1 | F2 |
|---|---|---|---|---|---|
| MDA | 0.889329 | 0.652287 | 0.688542 | 0.667545 | 0.679318 |
| Logit | 0.777744 | 0.589783 | 0.725939 | 0.596285 | 0.643348 |
| KNN | 0.837195 | 0.597041 | 0.675672 | 0.615751 | 0.643675 |
| SVM | 0.78628 | 0.594355 | 0.732304 | 0.604103 | 0.651904 |
| NN | 0.793598 | 0.608973 | 0.772424 | 0.622838 | 0.678731 |
| RF | 0.82378 | 0.623097 | 0.776795 | 0.647351 | 0.700724 |
| XGBoost | 0.837805 | 0.634076 | 0.787866 | 0.663093 | 0.716578 |
| CatBoost | 0.813415 | 0.620039 | 0.783203 | 0.641313 | 0.697575 |

Table 27: Unweighted general performances measures at 3 years of the 2019 group.

| Models | Precision Class 0 | Recall Class 0 | Precision Class 1 | Recall Class 1 |
|---|---|---|---|---|
| MDA | 0.354167 | 0.449057 | 0.950408 | 0.928027 |
| Logit | 0.215686 | 0.664151 | 0.96388 | 0.787728 |
| KNN | 0.24381 | 0.483019 | 0.950272 | 0.868325 |
| SVM | 0.224051 | 0.667925 | 0.964659 | 0.796683 |
| NN | 0.24505 | 0.74717 | 0.972896 | 0.797678 |
| RF | 0.27482 | 0.720755 | 0.971373 | 0.832836 |
| XGBoost | 0.295559 | 0.728302 | 0.972592 | 0.84743 |
| CatBoost | 0.266487 | 0.74717 | 0.973591 | 0.819237 |

Table 28: Performances measures for each class at 3 years of the 2019 group.