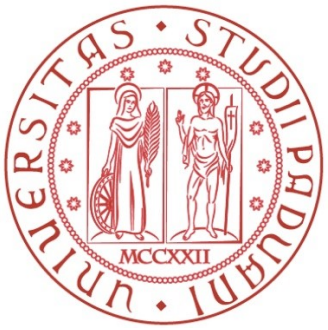



# UNIVERSITÀ DEGLI STUDI DI PADOVA

**DIPARTIMENTO DI SCIENZE ECONOMICHE E AZIENDALI**

**"M. FANNO"**

**CORSO DI LAUREA MAGISTRALE IN**

**ECONOMICS AND FINANCE**

**TESI DI LAUREA**

## BANKRUPTCY PREDICTION: APPLYING STATISTICAL AND MACHINE LEARNING METHODOLOGIES TO PREDICT COMPANIES FAILURE IN VENETO

**RELATORE**

**Chia.mo Prof. MICHELE FABRIZI**

**Laureando: PAOLO PICCOLI**

**Matricola: 1203194**

**Anno Accademico 2019/20**

Il candidato dichiara che il presente lavoro è originale e non è già stato sottoposto, in tutto o in parte, per il conseguimento di un titolo accademico in altre Università italiane o straniere. Il candidato dichiara altresì che tutti i materiali utilizzati durante la preparazione dell'elaborato sono stati indicati nel testo e nella sezione "Bibliografia" e che le eventuali citazioni testuali sono individuabili attraverso l'esplicito richiamo alla pubblicazione originale.

*The candidate declares that the present work is original and has not already been submitted, totally or in part, for the purposes of attaining an academic degree in other Italian or foreign universities. The candidate also declares that all the materials used during the preparation of the thesis have been explicitly indicated in the text and in the section "Bibliography" and that any textual citations can be identified through an explicit reference to the original publication.*

Firma dello studente

## Table of contents

# INTRODUCTION

Bankruptcy prediction studies, aimed at identifying and analysing common patterns of corporate default, have regained popularity among academics during the aftermath of the 2008 great financial crisis. This is mainly due to the objectives that the bankruptcy prediction research field has promised since its inception: lower operational risk borne by lending institutions, wiser investing decision for practitioners, faster reaction to distress conditions, more stable financial system and in general a more efficient and, possibly, effective allocation of resources. In practice, predictions are carried out on the basis of companies' financial indices retrieved from their public statements.

This thesis project finds its primary concern in the development and application of statistical and machine learning based bankruptcy prediction models on firms headquartered in Veneto, region located in the north east of Italy. To pursue it a sample composed of financial statements from 424 firms defaulted between 2013 and 2019 and 29711 sound entities have been employed to train and test six prediction models, namely: Logistic regression, Support Vector Machines, K – Nearest Neighbour, Adaptive Boosting, Decision Tree and Extreme Gradient Boosting. Four relevant conclusions have been reached. First, results indicate that Extreme Gradient Boosting stands as the best performing model with a peaking accuracy of 93%. Further, models show almost no sensitivity to the level of correlation allowed among the applied financial ratios, where the correlation range tested comprises values from 0,3 to 0,9. In addition, Net Income to Total Assets has been identified as the best individual predictor among the 54 financial ratios considered. Finally, the reliability in predictions drop substantially moving from one to two years prediction time while it remains stable between two and three years forecasting period.

The document is structured as follows: chapter 1 reviews the relevant literature developed from the '30s to the present; chapter 2 then analyses firms financial statements, describes the propensity score matching procedure followed to match sound firms with failing ones, describe the composition of all 54 financial ratios employed assessing their individual performance, compute the average correlation among indices and presents the logic and application of prediction models; chapter 3 moreover, presents results elicited from the applications of models on both an internal test set, composed only by Veneto based companies, and an external test set, grouping firms from the whole Italy; finally, chapter 4 hand conclusions out, lists possible future paths of research and close the document with the author comment.

# 1. REVIEW OF THE RELEVANT LITERATURE

Historically, bankruptcy prediction – the exploration of parameters and associated patterns useful for forecasting the probability of corporate failure – is a topic that had concerned investors, lenders and practitioners alike. Indeed, the ability to distinguish companies with solid future perspectives from those most likely to default, is critical to set expectations on returns and thus develop sound strategies. The literature on bankruptcy prediction has started developing at the turn of the 20$^{th}$ century when academics introduced studies regarding ratio analysis (Beaver, 1966). Ratio analysis aims at answering to the need for bankruptcy prediction models primarily looking at accounting and financial ratios. These have, among others, the advantage of being comparable amid companies with divergent absolute parameters (e.g. different revenues, sizes, assets, etc.) and operating in unrelated industry sectors.

In broad terms, the research on bankruptcy prediction based on ratio analysis can be divided in two macro periods. Up to the mid-1960s academics focused on univariate (single factor) studies (Bellovary et al. 2007). The most notable article of this first period is written by Beaver (1966) who is primarily concerned with a formal empirical verification of accounting ratios usefulness for prediction purposes. Thereafter, the attention shifted towards a multivariate approach looking to consider factors, ratios, in their interdependence. Pioneering in this second period is Altman's (1968) paper, still the most popular research among academics, with the first implementation of a multivariate analysis.

Deepening, the second period can be further split into two, mainly overlapping, phases: while the first decades are especially characterised by the implementation of more traditional statistical techniques (e.g. Altman (1968) adopts an MDA, Ohlson (1980) exploit a Logit regression), from the '90s more and more researches begin applying modern machine learning algorithms hoping to overcome previous limitations (Liang et al. 2016).

The next paragraphs are organised as follow: first, a review of the most relevant univariate phase researches is presented; further, the multivariate phase is elaborated splitting between authors applying traditional statistical techniques and those introducing more advanced frameworks; third, the crucial argument of how to define bankruptcy is treated; finally, three minor streams linked with the bankruptcy prediction literature but with different approaches from most academic articles are touched (the use of corporate governance indicators, Black-Scholes-Merton model and macroeconomic variables).

## 1.1 LITERATURE UP TO 1966 – THE UNIVARIATE PHASE

A first noticeable work in the bankruptcy prediction field belongs to the Bureau of Business Research (BBR) which in 1930 published a study analysing 24 ratios from 29 industrial firms. The ratios were compared with their overall average to look for specific trends affecting failing firms. The study highlighted eight ratios as good indicators of firm weaknesses. Moreover, it reports that, among others, working capital to total assets seems to perform particularly well as predictor of failure.

Two years later, FitzPatrick (1932) paper compares 13 ratios of 19 pairs of failed and non-failed firms. He concludes that net worth to debt and net profits to net worth are two crucial ratios to take into account while looking for defaulting patterns. He considers mainly manufacturing sector firms including phonographs and records manufacturers, food production and packaging companies, cotton and woollen factories, steel products manufacturers and adds to that the wholesale merchandise business.

Forward, Smith and Winakor (1935) studied ratios of 183 defaulted firms belonging to various industries in a follow-up analysis to the BBR's publication. They confirmed the importance of the working capital to total assets ratio as clear parameter to determine risk of financial distress.

In 1942, Merwin published a study regarding small manufacturers. He reports that failing firms display signs of weakness starting as early as four or five years before failure, on average. He moreover suggests net working capital to total assets, current ratio (i.e. current assets to current liabilities) and net worth to total debt as most relevant factors in bankruptcy predictions.

Further, Chudson (1945) looks for patterns of companies' financial structure in order to unveil if any factor follows 'normal', repetitive, sequences. The analysis is firstly focused both on the interrelationship of working capital items among each other and the actual role of the current ratio, which was widely considered, at the time, the most powerful accounting figure for prediction purposes. The author concludes that there is no 'normal' pattern on a general level. However, he also acknowledges that there are indications of clustering of ratios within specific set of industry, size and profitability. The study is not directly related to the bankruptcy prediction topic but rather to the interrelation of accounting quantities across firms belonging to different sectors: Manufacturing, Mining, Trade, and Construction. Nonetheless, it provides a relevant contribution to the field. Indeed, the clustering argument indicates that prediction models need to take into consideration the diverging features connoting different economic sectors.

Furthermore, Jakendoff (1962) compared ratios of profitable and unprofitable firms. He reports that current ratio and net working capital to total assets are higher for sound firms compared to weaker companies. Also, debt to worth ratio display lower values in profitable firms.

Finally, to conclude the first period relevant authors, Beaver (1966) compared the mean values of ratios from 79 pairs composed by failed and a non-failed firm belonging to 38 industries. Firms data are retrieved from the Moody's Industrial Manual and pertain to the 1954 to 1964 period. He considers 30 ratios from six broad categories: cash flow, net income, debt to total assets, liquid assets to total assets, liquid assets to current debt and turnover ratios. The study is based on a comparison of means, a dichotomous classification test and an analysis of likelihood ratios. Beaver finds that ratio analysis can be useful in the prediction of failure for at least five years before failure. Specifically, he reports that net income to total debt had the highest predictive ability in the first year prior to failure, 92% accuracy, followed by net income to sales (91%), net income to net worth, cash flow to total debt and cash flow to total assets (90% accuracy).

Interestingly, in his conclusions the author reports that there may have been many details preventing a measurement of the "true" predictive ability of ratios and that "There exists a countless number of arguments regarding the possible biases in the data" (Beaver (1966), p. 101), an issue that affects almost all the literature. He particularly refers to a selection bias: given that ratios are adopted to detect the financial "illness" of a firm, there may be companies whose "illnesses" were detected and cured just before default. Sample including such firms as non-defaulted are biased for any investigation of the usefulness of ratios in detecting bankruptcy early signs. Indeed, a crucial information is missing from the sample related to the actual number of firms that were able to "heal" using ratio analysis and that are thus considered as non-failed firms. This fact may understate the real ability of accounting ratios in forecasting failure.

Beaver also made an important contribution opening the path to the multivariate approach which will then be adopted by academics. In fact, in his "Suggestions for future research" he points out that "it is possible that a multiratio analysis, using several different ratios and/or rates of change in ratios over time, would predict even better than the single ratios" (Beaver (1966), p. 100).

Before passing on to discuss the relevant multivariate based works thereafter, it is worth asserting that since Beaver (1966) there have been other researches based on the univariate approach. Relevant papers include Pinches et al. (1975) and Chen and Shimerda (1981).

Pinches et al. (1975) examined 48 ratios from 221 firms with data coming from the COMPUSTAT[1] data tapes over the 1966-1969 timeframe. They looked for designing a grouping setting for the 48 ratios based on empirical evidences of correlation and informativeness[2]; determining the hierarchical relationship among these empirically based financial ratio groups; and proposing accounting ratios with highest predictive ability. Authors reported that 92% of the common variation among the 48 figures was accounted for in the following 7 groups: return on investment, capital turnover, inventory turnover, financial leverage, receivable turnover, short-term liquidity and cash position ratios. Further, they identified return on investment as the group including the overall most significant predictors, followed by capital turnover, inventory turnover and financial leverage.

Similarly, Chen and Shimerda (1981) had a primary question related to which financial ratios, among the hundreds that can be computed easily from the available financial data, should be analysed to obtain the information for predictive purposes. They looked for rigorous analysis aimed at indicating sound methodologies to select the best ratios avoiding collinearity/high correlation issues. They conducted a principal component analysis of 39 ratios for a total of 1,053 firms with complete data for both total assets and net sales in 1977 included in the COMPUSTAT tape. Authors show that high correlation levels between ratios cause results on predictive abilities to be sample-sensitive and possibly misleading. Moreover, they report a list of highly correlated ratios specifying that the actual selection of the preferable ratio has to be made on an ad hoc basis.

## 1.2 FROM 1966 TO THE 1990S – THE MULTIVARIATE STATISTICAL PHASE

The first and most famous multivariate study was published by Altman in 1968. He examines pairs composed of 66 corporations and 33 manufacturing firms that filed a bankruptcy petition under Chapter X of the US National Bankruptcy Act and an equal number of non-bankrupt companies. Data refer to the 1946-1965 period. Contrary to previous univariate research methodologies, Altman (1968) adopts the Multiple Discriminant Analysis (MDA): "a statistical technique used to classify an observation into one of several a priori groupings dependent upon

[1] Compustat is a database of financial, statistical and market information on active and inactive global companies throughout the world founded in 1962. Product of S&P Global Market Intelligence, which is a division of S&P Global

[2] their final objective was to propose a grouping framework to prevent future analysis to be based upon correlating and thus non-optimal set of accounting ratios.

the observation's individual characteristics" (Altman (1968), p. 590). MDA attempts to derive a linear combination of the characteristics (independent variables) which can best discriminate between the bankrupt, non-bankrupt groups of companies. It does so by computing a discriminant coefficient for each independent variable and subsequently combining them, coefficient and independent variable, into a specific score. All specific scores, attaining to all characteristics, are finally summed up to obtain what Altman (1968) calls "Z-score". A z-score optimum cut-off threshold is then computed to discriminate between bankrupt and non-bankrupt companies.

$$Z = v_1x_1 + v_2x_2 + v_3x_3 + \cdots + v_nx_n ,$$

where $v_1, v_2, v_3, v_n$ = Discriminant coefficients and $x_1, x_2, x_3, x_n$ = Independent variables

The author selects five accounting ratios as independent variables: working capital to total assets, retained earnings to total assets, earnings before interests and taxes to total assets, market value of equity to book value of total debt and sales to total assets.

He concludes that its bankruptcy prediction model is an accurate forecaster of failure up to two years prior to bankruptcy and that the accuracy diminishes substantially as the lead time increases. His methodology resulted in a 95% predictive accuracy for the initial sample one year prior to failure, in a substantially lower 72% accuracy for two years lead time and in only 48% accuracy for three years prior to failure (almost comparable to a random guess).

Since Altman's research, the contribution to the literature has increase dramatically both in terms of number of publications and complexity of approaches applied. Bellovary et al. 2007 count 28 relevant studies in the '70s, 53 during the '80s and more than 70 in the '90s.

Moreover, it is worth mentioning that while during the '70s and '80s papers were concerned with looking for the best 'traditional' statistical approach, starting from the '90s academics began introducing computationally intensive advance statistical methodologies like neural network and machine learning. From that point onwards research focused more and more on the new capabilities brought by these tools and on the comparison between traditional and advanced approaches.

The intention hereafter does not concern with reporting all material publications on bankruptcy prediction but rather to pinpoint and examine those either more influential or pioneering in the introduction of a novel approach or group of companies analysed.

Meyer and Pifer (1970) selected 30 pairs of failed and non-failed banks as original sample and 9 pairs for the hold-out sample for the accuracy testing. They considered only banks closed

(bankrupted) between 1948 and 1965 with at least six years of accounting data available. Authors run a multiple linear regression, with a dummy variable to discriminate between viable and less viable banks, based on 28 operating ratios and 4 balance sheet groupings. Similarly to Altman 1968, they find a sharp decrease in their model predictive ability with three or more years of lead time. They achieved between 67% to 100% accuracy in the holdout sample for failed banks and between 55% to 89% accuracy for non-failed banks. In other terms they were able to achieve low type II error rate, misclassifying failing firms as non-failing, keeping type I error rate, misclassifying non-defaulting for defaulting companies, significantly higher. It is worth adding that type I error is widely considered more dangerous for lenders (Du Jardin (2016)).

Wilcox (1973) based his models on the consideration that a firm financial level (e.g. net worth) is regarded as existing in one (N) of a positive, infinite set of states at any given time t. Further, at the immediate successive time period (t+1), the firm financial level can either decrease to N-1 or increase to N+1. From it, the author develops a binomial model applied to 52 pairs of bankrupted and non-bankrupted industrial firms selected from the Moody's Industrial Manual with at least six years of available accounting data from 1949 to 1971. To compute the year level for N, he collects accounting data about Net Income (including special or extraordinary), cash-only Dividends, Stock issued, Cash (including marketable securities), Current assets, Total assets and Total liabilities. The model accuracy resulted in 94% overall accuracy (true bankrupt and true non-bankrupt) for one year before failure prediction and 90%, 88%, 90% and 76% for 2, 3, 4 and 5 lead years to failure respectively.

Bilderbeek (1977) applied a step wise discriminant framework to classify 58 Dutch companies with available data for the 1950 – 1974 period. Similarly to Altman 1968, he computed the Z score for each firm and looked for the most efficient cut-off point to predict whether the firm considered was headed towards failure or not. The author analysed 20 ratios and ended up selecting five of them all: retained earnings to total assets, added value to total assets, accounts payable to sales, sales to total assets and net profit to equity. Bilderbeek (1977) achieved accuracies ranging from 70% to 80% over the five years lead time taken into account (Altman (1984)). Nonetheless, his step wise discriminant model is one of the first to be adopted in practice by Netherland institutions.

Martin (1977) selects the logit model to conduct a research on early signs of banks failure commissioned by the Federal Reserve of New York. In general terms, the logit model can be thought of as $\Pr(Y_i = 1) = F\,(x_{i1}, x_{i2}, \dots, x_{iM}, b_1, b_2, \dots)$ , the probability that the final outcome (bank belonging to either the defaulting group or non-defaulting one) Y for firm *i* is 1

(defaulting group) is a function of $x_{ij}$, the value of the j-th variable (accounting ratio) for the i-th observation and of all coefficients $b_i$ related to the M explanatory variables. The assumed functional form F is defined, inside the logit model, by the logistic function:

$\Pr(y_i = 1) = \frac{1}{1+e^{-Wi}}$ , i = 1,…, N , where $W = b_0 + \sum_{j=1}^{M} b_j x_{ij}$ is a linear combination of the independent variables and a set of coefficients $B = b_0, b_1, \dots, b_M$ which are to be estimated. The coefficient vector B of this linear combination is not known a priori but must be inferred from the known values of the $x_{ij}$'s and $Y_i$'s. The estimation of coefficients in W can be applied through to the *probit analysis* provided that W is normally distributed due to the Central Limit Theorem. The author not only applied logit model but also compared its performances with those deriving from a linear discriminant analysis (LDA) and a quadratic discriminant analysis (QDA) (both run similarly to Altman (1968) MDA in their basic principles).

Martin identified 58 banks as failed at some point in time between 1970 and 1976 from the FED databases (comprising approximately 5,700 Federal Reserve member banks). He moreover exploited 25 accounting ratios that can be divided in 4 classes: asset risk (e.g. loans to total assets), liquidity, capital adequacy and earnings. The author reported that logit models perform significantly better than linear discriminant analysis in various combinations of year data, lead years to failure and ratios (80% to 90% accuracy achieved by the logit implementation against 60% to 80% for LDA). However, no significant improvement was elicited when comparing logit and MDA results. Martin 1977 also asserts that greater sample need to be considered in subsequent studies to both validate results more precisely and verify normality assumptions on data.

Weinrich (1978), attempted to analyse risk classes, based on six risk-related accounting ratios, in order to predict insolvency. His sample of failed firms comprised of 44 German small and medium size companies, with an average sale of DM 4 million (less than $ 2 million), over the 1969-1975 period (Altman, 1984). Weinrich considers 3 consecutive annual financial statements, from the second to the fourth lead year before failure, without considering the last operating year, closest to the default event. Not including accounting data from the last year of operations preceding bankruptcy represents a marked difference from most of the previous articles and will be deepened later on looking at the work by Ohlson (1980). The author opted for a non-parametric linear discriminant analysis abandoning parametric classification techniques due to the lack of basic assumption (normality, variance- homogeneity of groups, and high correlation amongst the variables). His study achieves 89% accuracy for two years before failure and 84% and 78% for three and four years respectively.

Second only to Altman (1968), the bankruptcy literature has identified Ohlson (1980) as one of the most influential research. In his introduction, Ohlson stresses three critical concepts that set apart his analysis from foregoing works. First, he highlights the uniqueness of the time period taken into consideration when comparing research results: academics should not assess the precision of their models against others when data refers to different historical periods. In other terms, given that the models are generally not robust to macroeconomic and time dependent conditions, Ohlson underlines the necessity of only comparing models with data belonging to the same time period. Indeed, models trained with different historical periods data may, ceteris paribus, display dissimilarities due to, precisely, the historical background. Secondly, in contrast with most of the previous relevant papers, the author avoids collecting accounting figures from the Moody's Manual but rather obtains them from 10-K financial statements as reported at the time. The reason being the advantage of knowing at what point in time the statements were released to the public. Indeed, as mentioned with Weinrich (1978) just above, there might be a timing issue related with the first lead year to bankruptcy: if default occurs in essence before the release moment, then there is a chance that the statement already incorporates information about the default. When this is the case and said statement is included in the model formulation, the final prediction accuracy is seriously biased because of the timing issue. To account for this, Weinrich (1978) avoid the first lead year data, assuming the risk of losing valuable knowledge. Ohlson (1980), on the other hand, filters accounting statements to overcome the timing issue without 'sacrificing' precious information. The third introductory concept has to do with the definition of failure and the parameters that need to be satisfied to allow describing a company as bankrupt. Indeed, there might be critical differences in the definition of default adopted among researches that make essentially useless any comparison between models. For his framework, he proposes a purely legalistic definition based on whether the firm has filed for Chapter X or XI or some other legal notifications indicating bankruptcy proceedings. Such a matter will be further examined in broader terms later in the dedicated section.

Ohlson (1980), in contrast to the majority of previous studies, avoids the use of MDA as proposed by Altman (1968) for three considerations. First, MDA requires specific statistical properties to the predictor's distribution (such as normality and equal variance-covariance matrix for both failed and non-failed) that cannot be given for granted. Secondly, the output of the application of an MDA model is a score with little intuitive interpretation in light of the need to discriminate failing from non-filing firms. Further, if a Bayesian revision process is introduced, starting with the specification of prior probabilities, it will be invalid unless the

same assumptions of the pervious point hold (normality, etc.). Thirdly, the matching procedure applied by previous papers, carried out by considering for instance asset size and industry, seems to be arbitrary and of questionable usefulness. In this regard, Ohlson wonder if a better model might be achieved by directly including the matching variables into the set of predictors.

To overcome the three issues, the author opts for the use of conditional logit analysis. His data sample considers only industrial firms from the 1970-1976 period, with equity traded in either stock exchanges or Over-the-counter markets. The final sample comprises of 105 failed companies and 2058 non-failed entities. Nine ratios are included as predictors: size, log(total assets/GNP price levels index); total liabilities to total assets; working capital to total assets; current liabilities to current assets; first dummy variable, 1 if total liabilities exceeds total assets, 0 otherwise; net income to total assets; funds provided by operations to total liabilities; second dummy variable, 1 if net income was negative for the last two years, 0 otherwise; $\frac{NI_t - NI_{t-1}}{|NI_t| + |NI_{t-1}|}$ where $NI_t$ represent net income for the most recent period t, a measure taken from McKibben (1972). In this setting, Ohlson model reaches 96.12% accuracy for the first lead year, 95.55% for the second and 92.84% for the third. Ohlson notices that great relevance was to be attributed at the size parameter whose value greatly affects the final predictive outcome. Logically, an immediate explanation for this relates with 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 results, said element may be connected to the belonging to stock exchanges or OTC markets. He thus suggests as further research to include variable like equity prices and their trends.

Before continuing with the next relevant historical research, it is now compelling to look at a 1997 paper by Begley, Ming and Watts aimed at investigating the field performances that Altman (1968) and Ohlson (1980) models are actually able to reach. This is of particular interest because many among academics and practitioners frequently adopt such models for determining real world companies' financial distress and, as benchmarks, assessing other frameworks predictive power.

The starting concept of their analysis explains that the original models are typically applied to current data without considering the measurement error that they might introduce due to time discrepancies and the consequent risk of biased results. To give a precise meaning to these discrepancies, authors discuss two main arguments: first, leverage levels play an important role in both Altman (1968) and Ohlson (1980) and this may have an unwanted effect on subsequent

data since starting from the '80s a relatively high corporate debt level began being legally accepted; second, changes of bankruptcy laws in the late '70s allowed for greater strategic use of the default event[3]. For both reasons, results may be deviated by changes not incorporated in the original models and, as a consequence, less accountable in predicting failure. To examine it, they apply both original models and a re-calibrated version[4] of them on data from 165 bankrupt and 3300 non-bankrupt firms belonging to the 1980-1992 period. Begley et al. (1996) report that for what concerns the original models blindly applied to more recent data, they produce higher combined error rates. Interestingly though, Ohlson original model produce a slightly lower (12.4% against 10.8%) type I error on the new data with respect to Ohlson (1980) data. Type II error however increases substantially (17.4% against 26.6%) making the plain original approach unsuitable. Moreover, the re-estimated models show no meaningful improvements compared to original models. Thus, concluding, they underpin the adoption of Ohlson 1980 original model for analysis on more recent data.

One of the possible drawbacks of the Begley et al. (1996) has to do with the fact that Ohlson original paper might already incorporate, at least partially, "knowledge" of the economic conditions at work during the '80s and '90s. In other terms, Ohlson (1980), for its time proximity with the more recent period considered in the study, might be better performing when compared with a much older framework like Altman (1968). If this is true then, authors conclusions should be updated after considering the predictive power of Ohlson (1980) on more recent data (to match the time span intervened between Altman (1968) and the '80s, time to which data belongs). This consideration is left open for further proves. The main point of the research is nonetheless very relevant: historical dynamics may play a significant part in the development of models and thus they cannot be ignored for newer data.

Getting back to the pivotal researches on failure prediction, a peculiar study was carried out by Zimmer (1980). Following Libby (1975), he looked at prediction accuracies achieved by loan officers in executing the task of making annual predictions of corporate failure based on a time series of ratios. Specifically, 30 subjects were selected among loan officers from two Australian major banks. The selection did not follow random patterns, nonetheless the author achieved great variety in subjects characteristics (age, experience, etc.). Each subject was individually provided with 42 real but disguised industrial companies listed on the Sydney Stock Exchange between 1961 and 1977, half of which filed for bankruptcy. Moreover, five accounting ratios

[3] New bankruptcy acts made failure less costly for corporations which began to strategically exploit default in their interests.

[4] The models are run on more recent data to look for re-estimated coefficients and cut-off values.

were identified to be suitable for the analysis: quick assets ratio (in general, cash and cash equivalents to current liabilities); earnings before interest and taxes to total assets; ordinary dividends to ordinary earnings; total debt to gross cash flow; and long term debt to equity. The task was carried out individually and independently with the notion that roughly half of the financial profiles to be examined belonged to defaulted companies. Zimmer (1980) remarks that at a 95% confidence level, loan officers would have needed to correctly indicate at least 27 out 42 predictions for his conclusion to be significantly different from the prediction made by a random guess. He found that the overall accuracy is significantly higher than randomness would imply, with a peak of almost 90% accuracy for predicting failing firms when the loan officer declared to be "very confident" and a trough of 58% accuracy for non-failing firm and "not-very-confident" self-assessment. Though it is clear that the conclusions reported have powerful implications regarding valuable synergies between empirical models and professional's judgment, the author himself calls for further research to verify results in a more realistic setting (e.g. avoiding setting prior probabilities by revealing that half of financial profiles belong to failing firms). Casey (1980), in a fairly similar study, concludes that in such more realistic scenario loan officer predictions are not significantly better than random selection.

Frydman, Altman and Kao (1985) were among the firsts to apply the Recursive Partitioning Algorithm (RPA) and to compare it to an MDA. RPA is a computerized, nonparametric classification technique based on pattern recognition. The model resulting from RPA is in the form of a binary classification tree which assigns objects into selected a priori groups and whose final nodes represents the final classification (Breiman et al. (1984)). Authors compared 200 total firms, 58 bankrupt industrial companies failed during the 1971-1981 period and 142 randomly selected manufacturing and retailing enterprises from the COMPUSTAT database. Further, the analysis relates on 20 accounting ratios deemed valuable by looking at previous researches results. To thoroughly examine the behaviour of the model, authors set a weighting cost function to either avoid type I error (i.e. a bankrupt firm classified as non-bankrupt, generally considered more dangerous in the literature) or type II error (i.e. non-bankrupt firm classified as bankrupt). They report that at all weights considered, RPA significantly outperforms MDA in its predictive ability. Interestingly, at a cost level of 50 (both errors considered equal) the RPA tree uses a specific cash flow to total debt level as cut-off point, which underlines the ratio pivotal predictive ability at classifying failing firms inside the model.

## 1.3 FROM THE 1990S TO PRESENT – THE 'ADVANCED' MODEL PHASE

Thanks to the evolution and accessibility of more reliable and faster computing technologies, new, advanced approaches combining multiple academics fields have been more and more adopted for prediction purposes (Liang et al. 2016).

In 1990 the first relevant authors adopting the Neural Network (NN) technique appeared in the literature. In general terms, neural network architecture can be described as biologically inspired, involving the intricate interconnection of many nodes (the equivalent of brain neurons) through which inputs are transformed into outputs. Once a specific network architecture is defined, the overall network is repeatedly presented with training cases from an estimation sample, and the connection weights between nodes are updated to bring the network outputs closer to the actual target output values. This training process is referred to as network learning. One of the best advantages of neural network modelling is the capability to capture nonlinear processes.

Bell, Ribar and Verchio (1990) were interested in the comparison of a logit model and the prediction accuracy gained with a NN framework composed by a twelve nodes input layer, a six nodes hidden layer and a final output layer. They identified 28 candidate predictor variables using the results of prior research. Variables relate to the following features: size, loan exposure, capital adequacy, asset quality, operating performance, non-operating performance and liquidity. Authors applied logit and neural net models to an estimation sample formed by 102 banks failed in 1985 to be added to 906 non failed (1984 annual financial statement data) and a separate holdout sample containing 131 banks that failed during 1986 on top of 928 non defaulting institutions (1985 annual financial statement data). The conclusions of the study highlight a 69.5% and 97.3% average accuracy in predicting failing and non-failing banks respectively for the logit model. Similar levels are achieved by the NN model as well. Indeed, Bell et al. (1990) stress the fact that the two models are essentially comparable in performances showed. It is however also reported that NN tends to have a significant, though not large, higher precision for what concerns type II error (an average 5% lower rate of error).

Cadden (1991) follows a similar path comparing a backpropagation NN models with an MDA framework applied to three years, prior to failure, accounting data belonging to 59 companies defaulted in the '70s. In addition, data from 59 non-failing firm from the same time period is elicited. He sets up an estimation sample composed of 98 firms (49 from each group) and a testing sample with 10 pairs. Moreover, twelve ratios are included as predicting variables: current assets to current debt, net profits on net sales, net profits on tangible net worth, net

profits on net working capital, net sales on tangible net worth, net sales on net working capital, net sales to inventory, fixed assets to tangible net worth, current debt to tangible net worth, total debt to tangible net worth, inventory to net working capital and current debt to inventory. Results on the estimation sample indicate that while the first-year lead time accuracy is substantially comparable between MDA and NN, with the latter slightly outperforming the former, a significant difference is spotted in the subsequent year predictions. Indeed, if NN almost holds onto the same level of accuracy, MDA drastically decreases in performances going from a maximum of 93.9% on the first year to a minimum of 61.2% accuracy on the third. Even greater divergence in performance is significantly affecting the test sample results. Here in all the three years lead time considered, backpropagation NN overcomes MDA accuracy.

Luoma and Laitinen (1991) apply what is known as proportional hazards (PH), in the form of the semi-parametric PH model defined by Cox (1972), to compare its predictive power to both MDA and Logit models. To understand proportional hazards, it is necessary a brief introduction to survival analysis (SA). Following Gepp and Kumar (2010), SA technique is a dynamic statistical tool used to analyse the time probability until a certain event. Thus, the SA approach to bankruptcy prediction is fundamentally different from the other aforementioned approaches (i.e. MDA, Logit, NN, etc.). While other techniques model default predictions as a classification stance, SA models them considering businesses' datapoints as represented by lifetime distributions. Lifetime distributions can be characterised by a number of descriptor functions, the most common being the survival or hazard function. Survival function S(t) represents the probability that a business will survive past a certain time t, while hazard function h(t) represents the instantaneous rate of failure at a certain time t.

Further, the basic difference between various SA models is the assumptions about the relationship between the hazard (or survival) function and the set of explanatory variables (X). Thus, the general regression formula can be written as $h(t) = g(t, XT\beta)$, where XT is the transpose of X, β is the vector of explanatory variable coefficients (the covariates), t is the time considered and g an arbitrary function. Traditionally, SA has been divided into two main types of regression models. These types are the proportional hazards (PH) and accelerated failure time (AFT) models, both of which have fully parametric and semiparametric versions. Due to its flexibility, the most prominent model applied in business failure field is the semi-parametric PH model defined by Cox (1972).

In such settings, Luoma and Laitinen (1991) selected 36 failed Finnish limited companies and 36 successful counterparts with data from the '80s. Their predictions are made by dividing the businesses into two groups based on their hazard ratios, according to the ratio of failed and non-

failed businesses in the original sample. They report an average 68% accuracy for the PH model for both defaulted and successful companies. While MDA registered 64% and 76% accuracy for bankrupt and non-bankrupt firm respectively (essentially in line with PH performance), Logit model reached an overall average of 71% accuracy showing a slightly higher predictive power with respect to the others. Nevertheless, authors argue that the SA approach is more appropriate and flexible, and, thanks to its time dependent configuration, uses information more valuably. Also, they point out that the empirical under-performance could have been due to the small sample size, an issue that can be overcome with relative ease.

Another relevant methodology was introduced by Tam and Kiang (1992), who studied the impact of machine learning K - Nearest Neighbour (KNN) and Decision Trees, also known as Inductive Dichotomizer 3 (ID3), approaches to the field.

On the one hand, KNN is a distribution-free, non-parametric method for classifying observations into one or several groups based on one or more quantitative variables. Compared to MDA and Logit, its main advantage lies in the possibility of both relaxing the normality assumption and eliminating the functional form required in MDA and logistic regression. The group assignment of an observation is decided by the group assignments of its first k nearest neighbour (hence the name). Using the nearest neighbour decision rule, an observation is assigned to the group to which the majority of its k nearest neighbours belong. This method has the merits of better approximating the sample distribution by dividing the variable space into any arbitrary number of decision regions, with the maximum bounded by the total number of observations.

On the other, ID3 instead of generating a decision rule in the form of a discriminant function, it creates a decision tree that properly classifies the training sample in a recursive manner. It entails a nonbacktracking splitting procedure that recursively partitions a set of subsamples (randomly selected) into disjointed subsets. The subsets obtained are then aggregated to reach the final group classification.

Tam and Kiang (1992) compare KNN and ID3 with MDA, Logit and Backpropagation neural network. They employ data sample consisting two years prior to failure ratios of 59 Texas banks that failed in the period 1985-1987. They claim to have selected only Texas banks to increase datapoints homogeneity. Moreover, these were matched with 59 non-failed entities on the basis of asset size, number of branches, age and charter status. To make sure the models could then be adopted by practitioners, authors selected 19 accounting ratios following CAMEL criteria (Capital, Asset, Management, Equity, and Liquidity) used by the FIDC (Federal Deposit

Insurance Corporation), a US banking system supervisor institution. Findings show that there is no clear best predictive model among those applied: Logit seems to have the highest average accuracy in predicting non-failed banks (95% in one year prior to default and 100% for two years) while KNN manifests the lowest accuracy for prediction concerning failed banks (59% and 80% for respective lead years). Overall though, authors prize the structure of the backpropagation NN framework because of attributes that, in their view, best fit with the needs and challenges that bankruptcy prediction entails and thus suggest further research in this direction.

Bryant (1996) was among the first to apply Case-Based Reasoning (CBR) artificial intelligent methodology to bankruptcy predictions. In general terms, the basic principle underlying CBR refers to the fact that human experts use analogical or experiential reasoning to solve complex problems and to learn from problem-solving experiences. However, in searching their memories, human experts may suffer from primacy (remembering the first thing more vividly) and/or recency (remembering the last thing more vividly) effects. CBR model basically corrects for such biases allowing for a systematic search of a case library (memory) in order to retrieve cases that most closely match the problem at hand. In doing so, CBR relies on sets of independent decision trees. She further suggests CBR for bankruptcy prediction modelling because of its adaptability to articulated set of data, ease of revision/update, comparability with other studies and clarity in results interpretation. Bryant (1996) primary research objective is to verify if CBR can actually be favourably employed for prediction purposes and if so, comparing its performances with Ohlson (1980) nine factors model. To enhance comparability of the predictive accuracy of the two models, the author closely follows Ohlson's logit model sampling procedures. Accordingly, the proportion of bankrupt to nonbankrupt firms roughly attains 1:20, and only manufacturing and industrial firms are included. A random sample consisting of 85 bankrupt and 2,000 nonbankrupt manufacturing and industrial firms from the 1975-1994 period is generated. For each firm, 25 financial ratios found significant in the literature are calculated and included. Three (one per each first three years heading to failure) CBR models are derived using data from 1975-1989. The remaining data is used as holdout sample to validate the three models. Bryant (1996) finds that CBR behaves rather poorly on bankrupt firms: the estimation sample (1975 to 1989 firms) is best predicted in the first year with only 47.3% accuracy while the worst relates to the third year with 39.7% accuracy (lower than a complete random process would perform). On the contrary, CBR algorithms execute rather well in predicting non-bankrupt firms: the lowest accuracy level achieved among the three years stands at 94.8% for the third year. Similar findings connote the holdout sample

accuracies. The conclusion of her study is that Ohlson's logit model have superior predictive accuracy than the CBR. She adds that although there is limited academic research on CBR, her findings cannot support claims of overall CBR superiority to other methods, such as logit. Specifically, in minimizing classification errors of bankrupt firms (type I errors, considered to be the more important than type II), logit far outperforms CBR.

In the same year, Wallrafen, Protzel and Popp publish a work seeking to find more accurate prediction results by combining different models. Specifically, they adopted what is known as "Soft computing"(Zadeh (1994)): a combination of two or more artificial intelligence methods which are capable of pinpoint data patterns looking at seemingly unrelated parameters. In particular, authors decided to analyse the optimization role of Genetic Algorithms applied to a neural network model applied to the prediction task. The advantage for such a decision lies on the fact that neural network learning encounters problems with generalizing its results to unknown cases and this might be avoided by using Genetic Algorithms to pre-emptively select training data.

Genetic Algorithms (GA) can generally be seen as modelling the principles of biological evolution through a four-step cycle: they firstly generate an initial population of potential solutions called individuals; then an evaluation of the fitness of each member of the population is carried out; further, they select promising individuals to be manipulated through genetic operators (mutation, crossover, selection), where a proportionate selection scheme gives individuals with higher fitness a larger chance of being included in the modelling cycle; finally, the manipulation of selected individuals through genetic operators takes place.

Wallrafen et al. (1996) comprise of a 6667 German corporations' dataset, including 2667 entities as hold out sample. Moreover, they employed 73 financial ratios as predicting variable. These financial ratios can be subdivided into eight clusters: capital structure, liquidity, financial strength, profitability, current account turnover, short-term liabilities, repayment behaviour and miscellaneous ratios. For each company at least three data sets from different years are available. Companies are assigned to one of the binary classes "solvent" or "insolvent' based upon their actual historical performance, where 'Insolvency" is defined by specific legal conditions under German statutes. A period of at least 18 months between the date of the last financial statements used for the 'Insolvent" companies and the date insolvency actually occurred, ensures a meaningful time horizon for the prediction. Authors report lower-than-expected results. The GANN methodology (Genetic Algorithms combined with Neural Network) reached at most a 64% predictive accuracy on the testing sample and only after a fairly long time: 295 generations (iterations) which required about four computing days. What

is more, they add that few improvements can be considered with such specific techniques and that further research should look at completing the framework with other more powerful models.

In 1999, Dimitras, Slowinski, Susmaga and Zopounidis published a paper introducing a novel framework, Rough set theory, with the aim of comparing it with the more adopted MDA and Logit models. The rough set philosophy is anchored on the assumption that every object of a conceptualized universe can be associated with some information (data, knowledge). Objects characterized by the same information are thus to be considered indiscernible. The indiscernibility relation generated in this way is the mathematical basis for the rough set theory. Any set of all indiscernible objects, called elementary set, forms a basic element of knowledge about the universe. Any set of objects being a union of some elementary sets is referred to as crisp (precise); otherwise the set is defined as rough (imprecise, vague). Consequently, each rough set has limit cases, i.e. objects which cannot be classified with certainty as members of the contemplated set. Therefore, a rough set can be represented by a pair of crisp sets, the lower and the upper approximation, where the lower approximation consists of all objects which belong to the set with certainty whilst the upper approximation contains objects which belong to the set only with certain probability. From such setting, a classification can be carried out.

Authors collected data from a large number of firms which failed in Greece during the 1986-1990 period. From them, 40 firms belonging to 13 industries were selected and paired with non-failing companies. The healthy firms were chosen among those of the same industry, having similar total assets and number of employees. Furthermore, a second, hold out, testing sample consisting of 19 pairs of entities was collected using a similar approach. For it, however, only firms failed in the 1991-1993 timeframe were considered. Five years (three for the hold-out sample) prior to default financial statements were collected and analysed to identify 28 accounting ratios (mainly suggested by the previous literature). In their findings, authors report that the accuracies reached by the rough set framework were generally better than those obtained by the classical discriminant analysis and logit analysis, although the superiority over logit was not so distinct as that over discriminant analysis. Moreover, their conclusions stress that rough set accuracy measures on the testing sample drop abruptly in the second and third year (from 73.7% to 47.4% and 36.8% respectively) which indicates low reliability in the model aside from one-year lead time predictions. Nonetheless, Dimitras et al. (1999) underline the relevance of their model for predictive purposes asserting that many advantages can be obtain given the similarity between the bankruptcy prediction connotating elements and the working structure of rough sets models: it accepts both quantitative and qualitative attributes; it

contributes to the minimization of the time and cost of the decision making process; it offers transparency of classification decisions, allowing for clearer argumentation; and it takes into account background knowledge of the decision maker.

In 2005, Min and Lee sought to introduce Support Vector Machines approach into the bankruptcy prediction field comparing its performances against MDA, Logit and a three-layer fully connected back-propagation neural networks (BPN).

Support Vector Machines is a machine learning technique conceived by Vapnik (1998). SVM optimisation model is based on the transformation of a mathematical function by another function, called the 'kernel', by which it identifies the greatest distance between the most similar observations that are oppositely classified. It does so by means of a higher space optimal separating hyperplane (OSH) of some specified dimension which is specifically "constructed" and used for clustering purposes. SVM looks to find a special kind of OSH: the maximum margin hyperplane. The maximum margin hyperplane gives the maximum separation between decision classes. The training examples that are closest to the maximum margin hyperplane and thus define the minimum distance between groups identified, are called support vectors. All other training data is essentially irrelevant for defining class boundaries.

Many attractive features make SVM suitable for prediction goals. First, SVM is considered to achieve excellent generalization performance on a wide range of settings, particularly useful when combining differing characteristics. Also, SVM follows the structural risk minimization principle, SRM, which has been shown to be superior to traditional empirical risk minimization, ERM, principle employed by conventional neural networks. SRM minimizes an upper bound of generalization error as opposed to ERM that minimizes the error on training data. Therefore, the solution of SVM may be closer to global optimum while other neural network models tend to fall into local optimal solutions. Third, the technique is broadly acknowledged as easily tractable under a mathematical viewpoint. Finally, overfitting seems to occur much less frequently than in other machine learning approaches.

Min and Lee (2005) evaluate 1888, 944 pairs of bankrupt and non-bankrupt in random order, Korean's small and medium size enterprises with data obtained from the Korea's largest credit guarantee organization. 11 accounting ratios, from an initial set of 38 figures retrived from the literature, are then selected applying a stepwise logistic regression analysis. To fine tune the SVM model, authors implement and compare 4 types of kernels in the study: Linear, Polynomial, Radial Basis Function (RBF) and sigmoid. Among them, the best performing is found to be the RBF kernel with 88% average prediction accuracy in the training data and 83%

in the holdout sample. This results to be also the overall best predictor model. In fact, BNN display a slightly lower accuracy of 85% and 82% respectively, followed by the MDA approach with 78% and 79% accuracy and finally by the logit framework, 79% and 78%.

Finally, it is worth mentioning a paper published by Barboza, Kimura and Altman in (2017). They selected and compared eight among the most relevant framework adopted in the previous bankruptcy prediction literature: Bagging, Boosting, Random Forest (RF), Support Vector Machines with both a linear kernel (SVM-Lin) and a Radial Basis Function kernel (SVM-RBF), Artificial Neural Networks (NN), Logistic regression and MDA.

Bagging, whose name shortens "bootstrap aggregating" is a technique involving independent classifiers that uses portions of the data to then combine them through model averaging, providing the most efficient clustering results (Breiman, (1996)). It creates random new subsets of data through sampling, with replacement, from a given dataset, generating confidence-interval estimates. The final objective of the bagging approach is to reduce class overfitting within the model. Their Bagging algorithm follows Breiman's: first, a random bootstrap set, t, is selected from the parent dataset; second, classifiers, $C_t$, are configured on the dataset from step 1; further, steps 1 and 2 are repeated for t = 1, . . ., T.; finally, each classifier determines a vote $C(x) = T^{-1} \sum_{t=1}^{T} C_t(x)$ where x is the data of each element from the training set. In the last step, the class that receives the largest number of votes is chosen as the classifier for the dataset.

Secondly, the Boosting technique consists of the repeated use of a base prediction rule or function on different sets of the initial set. Boosting builds on other classification schemes and assigns a weight to each training set, which is then incorporated into the model. The data are then reweighted. Boosting can apply the base classifier to find a model that better classifies the set, identified by a lower error rate in the training set. A derived algorithm, AdaBoost ("adaptive boost") has proved powerful for classification prediction. AdaBoost initialises giving equal weights to all observations. Thus, the first sample is uniformly generated from the initial observations. After the training set, Xi, is extracted from X, a classifier Yi is trained on Xi. The error rate is calculated, considering the number of observations inside the training set. The new weight for each observation is based on the effectiveness of the classifier Yi. If the error rate is greater than a random guess, the test set is discarded, and another set is generated with the original weights. Alternatively, if the error rate is satisfactory, the weights of the observation are updated according to the importance of the classifier. These new weights are then used to generate another sample from the initial observations.

Further, the random forest technique (RF) is based on decision tree models, also known as generalised classification and regression trees' (CART). It is particularly robust and allows for the presence of outliers and noise in the training set. Finally, RF identifies the importance of each variable in the classification results. Therefore, it provides not only the classification of observations but also information about the determinants of separation among groups. The RF technique follows an approach similar to bagging, as it repeatedly generates classification functions based on subsets. However, RF randomly selects a subset of characteristics from each node of the tree, avoiding correlation in the bootstrapped sets. The forest is built for several subsets that generate the same number of classification trees. The preferred class is defined by a majority of votes, thus providing more precise forecasts and, more importantly, avoiding data overfitting (Breiman, (2001)).

Authors run the analysis on American and Canadian companies covering the 1985 to 2013 period using Compustat. Furthermore, a subset covering 1985 to 2005 (133 bankrupt and 13300 solvent) was extracted to provide the training set, which included information on 449 companies that filed for bankruptcy during this period as well as information on the same number of non-bankruptcy firms. Insolvent firms in the training set include all companies in the database that filed for bankruptcy during this period and for which financial data were available three years prior to filing. The solvent firms were randomly chosen and were limited to companies that did not file for bankruptcy during the entire period and for which financial data for at least two consecutive years were available. They included variables following Altman (1968) and Carton and Hofer (2006): liquidity (X1), profitability (X2), productivity (X3), leverage (X4), and asset turnover (X5) (Altman, 1968); growth of assets (GA), growth in sales (GS), growth in the number of employees (GE), operational margin (OM), change in return on equity (CROE), and change in price-to-book ratio (CPB) (Carton and Hofer, 2006). As usual in the literature, two kind of accuracy measures are retrieved: Sensitivity, type I error, also called True Positive Ratio, measures the proportion of bankrupt firms correctly classified on the total number considered; Specificity, type II error, also known as True Negative Ratio, measures the proportion of solvent firms correctly classified. For lending purposes, it is firstly necessary to prioritize the minimization of type I error (increase sensitivity) in order to avoid losses (Ohlson (1980)). However, prioritizing type I error also bears the risk of limiting credit access to solid, creditworthy enterprises.

Barboza et al. (2017) report a significant difference in performance between traditional statistical frameworks and the more advanced machine learning approaches. Specifically, looking at the overall accuracies registered in the training sample, it registered the superiority

of the Random Forest approach (87.06% accuracy) even though both boosting and bagging achieve very close levels (86.65% and 85.65% accuracy respectively). Logit model (76%) reaches the mean accuracy of SVM-Lin, SVM-RBF and NN (71.50%, 79.77% and 72.98%) while MDA displays quite poor results with just 52.18% average accuracy.

To conclude the historical section, it is worth reporting Bellovary et al. (2007) conclusions, which, despite their relatively old date of publication, are still valid: a great amount of model has already been suggested and fine-tuned and high levels of accuracy with low number of ratios have been reached (Beaver (1966) reported 92% overall accuracy with just one variable employed). Thus, the challenge should not be now concerned with contrive and introducing new, more powerful frameworks but rather with finding the right paths to put the most promising models into practice.

## 1.4 THE DEFINITION OF BANKRUPTCY

A crucial issue for the identification of prediction models able to classify bankrupt and non-bankrupt entities resides in the definition of bankruptcy itself. In other terms, as put by Ohlson, "one may ask a basic and possibly embarrassing question: why forecast bankruptcy?" (Ohlson (1980), p. 4). The question is all but trivial since behind it lies the need for better understanding what dynamics (losses, inefficiencies, non-suitable market/financial approach, etc.) should be avoided by lenders and practitioners alike while examining companies. Ohlson argument underlines that no obvious answers can be found and that, ultimately, the hurdle reflect the impossibility of reducing firms' complex reality to a binary status: bankrupt and non-bankrupt. Alternatively said, there is not a simple and univocal way to determine exactly when a firm can be said to be defaulted. Most of the researches reported so far, interpret bankruptcy as the legal status attributable whenever some legal condition is recognised and thus processed (the condition being logically dependent on the jurisdiction considered). This incontrovertible legal status, however, can only be contemplated as the 'lower bound' of the bankruptcy definition. The real issue relates with deciding at which point, along the distance between "legally defaulted" and "sound", should be set the discriminant boundary. As Ohlson asserts, empirical studies do not agree on what constitute "failure", with definitions varying significantly and arbitrarily across studies.

Correlated with above is the question on how to realistically interpret prediction model results. In this sense, Beaver (1966) posed an important argument: if accounting ratios are applied to detect financial "illness" of a firm, there may be many companies whose illnesses were detected

before failure occurred. If this is the case and an unknown proportion of them happens to be included in the data of the solvent entities, there may be the risk of overstating the model ability to predict bankruptcy. In other terms, there is again an incomprehension due to the complexity in determining the definition of bankruptcy. If in fact, a precise threshold for default classification was to be identified, Beaver's problem would not stand because the actual proportion of firms saved just before irreversible complications would be known and thus a correct estimation of the model accuracy could be computed. Beaver decided to consider bankruptcy as the inability to repay interest and principal from liabilities due for both simplicity and need to limit as much as possible the drawback just mentioned.

In line with the issue faced by Beaver (1966), Laswad, Kuruppu and Oyelere (2003) looked for prediction model aimed at classifying going concern from non-going concern corporations based on the probability for the firm to be facing liquidity procedures. The assumption behind the article refers to the fact that too many differences are present among bankrupt firms in different countries and that a framework with good generalization ability need to begin from more objective, shared elements. Such elements cannot be found in the definition of bankruptcy found in previous articles. They spot a critical problem in the bankruptcy prediction literature: there are profound differences in the legal determination and rights/obligations granted under bankruptcy acts among countries in the world. For instance, in the US where the insolvency laws are debtor oriented, corporate bankruptcy procedures encourage companies in financial difficulty to continue as going concerns. Therefore, it is possible for companies that file for bankruptcy to reorganise and emerge from bankruptcy, or to merge with another entity as a going concern. This is in contrast to the insolvency procedures in creditor-oriented countries such as the UK, Germany, Australia and New Zealand where liquidation is the most common outcome of corporate insolvency. So, to overcome the problem, they propose a prediction model based on liquidation risk rather than bankruptcy filing (inability to repay principal and interest of some sort of liability). Authors examined a total of 135 from the 1987 – 1993 period using the logit model. They show that their model outperforms Altman (1968) Z-score procedure in the overall accuracy achieved and thus could have great implications in credit-oriented legislations as those mentioned above.

Laswad et al. (2003) stands as an example of how hurdles pertaining the definition of bankruptcy may be limited pursuing alternative solutions. Nonetheless, the prediction inaccuracy that results from it is still to be taken into account as an unknown factor in practice.

## 1.5 OTHER RELEVANT RESEARCHES: CORP. GOVERNANCE INDICATORS

Accounting and financial ratios are not the only type of predictive variables tested in the literature. Following Liang et al. (2016), corporate governance indicators (CGI) also can be useful parameters to predict firms' failure. In their comprehensive examination of the application of prediction models combining accounting ratios and CGI, they define corporate governance as the set of mechanisms, processes and relations by which corporation are controlled and directed by the chief team. Further, these are to be intended as integrated with internal and external control mechanisms allowing shareholders to exercise appropriate oversight on the company to ensure proper profitability levels. In such defined context, many corporate governance indicators (CGIs) have been identified in the literature which have been used for enhancing bankruptcy or financial crisis management. These can be broadly classified into five categories including board structure (e.g. Number of seats on board, number of directors, number of supervisors, etc.), ownership structure (e.g. Shareholding ratio of board, shareholding ratio of advisor, etc.), cash flow rights (e.g. Amount of investments in other enterprises divided by stockholder's equity), key person retained (e.g. Turnover of spokesman within a month, Turnover of CEO within a month), and others. Authors research looks to compare prediction accuracies on financial ratios with and without CGI on five prediction models, namely support vector machines (SVM), k-nearest neighbour (K-NN), naïve Bayes classifier (NB), classification and regression tree (CART), and multilayer perceptron (MLP). Feature selection, to reduce irrelevant or redundant features by selecting more representative features having more discriminatory power over a given dataset, is carried out applying five feature selection methodologies to find the most promising predictors (financial ratios and CGI): stepwise discriminant analysis (SDA); stepwise logistic regression (SLR); t-testing; genetic algorithm (GA); and recursive feature elimination (RFE). Data were collected from the Taiwan Economic Journal for the years 1999–2009. The resultant sample includes companies from the manufacturing industry composed of industrial and electronics companies (346 companies), the service industry composed of shipping, tourism, and retail companies (39 companies), and others (93 companies). Consequently, the collected dataset is composed of 239 bankrupt and 239 nonbankrupt cases, with each company represented by 95 financial ratios and 95 CGIs as the input variables, to be filtered with the feature selection processes. Liang et al. (2016) report that overall Financial ratios show higher predictive ability compared to CGI alone but also that indeed the financial ratios and CGIs combination enhances the predictive power in all models examined. Furthermore, the best performing framework, that combines SDA in

the feature selection process and SVM, achieves an average 81% accuracy with a 16.3% type I error rate.

A similar study was conducted the subsequent year by Elshahat I., Elshahat A. and Rao who compared the introduction of a corporate governance index against Altman's Z-score framework. They include variables such as Board of Director's characteristics, Board Committees, internal control and auditing systems add to the understanding of the firms' corporate governance. Interestingly, they explain that corporate governance can be used as a comprehensive measure for the agency problems that directly affect the firm structure and survival and that in these terms corporate governance indices might be a good proxy for the risk brought about. For a one-year prediction window, authors find no significant differences for bankrupt and non-bankrupt firm accuracy averages between Altman (1968) data and z-score applied to their data. The inclusion of the corporate governance index, however, slightly improves the predictability of the bankrupt firms. Nonetheless, both prediction models, with and without corporate governance index, achieve an overall accuracy of about 69%. This, compared with Altman's original 95% maximum accuracy, seems quite irrelevant even though, as they continue, Altman (1968) has been criticized for inconsistency in results many times in the literature. Thus, it is not always clear what role is actually performed by Corporate governance indices in enhancing models' predictive ability.

## 1.6 OTHER RELEVANT RESEARCHES: BLACK – SCHOLES – MERTON

Another material branch of the bankruptcy prediction literature has to do with the application of market-based measures alone as predictive variable. An example of it is the study conducted by Hillegeist, Keating, Cram and Lundstedt in 2003 in which they compare Altman (1968) Z-score and Ohlson (1980) O-score with an approach based on the Black, Scholes and Merton (BSM) option-pricing theory. Authors assert four main advantages in adopting a market-based model: first, the going-concern principle implies the assumption that firms will not go bankrupt, thus their data might be under/overstated; also, the conservativism principle often causes asset values as reported in financial statements, to be understated relative to their market values; third, accounting-based bankruptcy prediction models fail to incorporate a measure of asset volatility, crucial in in capturing the likelihood that the value of a firm will decline to such an extent that the firm will be unable to repay its debts; finally, accounting ratios can only be computed at financial statements publication (few times a year), with the logic consequence of

risking to timely alert of distress conditions, while market-based variables can be exploited with greater frequency.

BSM starts from the intuition that equity can be viewed as a call option on the value of the firm's assets. Thus, following their pricing model, the equation for valuing equity as a European call option is given by

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

with $d_1 = \frac{\ln\left[\frac{V_A}{X}\right] + \left(r - \delta + \frac{\sigma_A^2}{2}\right) T}{\sigma_A \sqrt{T}}$ and $d_2 = d_1 - \sigma_A \sqrt{t}$

Where $V_E$ is the current value of equity; $V_A$ is current market value of assets; X is the face value of debt maturing at time T; r is the continuously-compounded risk-free rate; δ is the continuous dividend rate expressed n terms of $V_A$; and $\sigma_A$ is the standard deviation of assets returns. Value of equity equation is modified for dividends and reflects that the stream of dividends paid by the firm accrue to the equity holders.

The BSM model assumes that the natural log of future asset values is distributed normally as

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

where $\mu$ is the continuously compounded expected return on assets.

From it, and following McDonald 2002 (p. 604), authors derive the probability that $V_A(T) < X$ that is $BSM - prob = N(-\frac{ln\frac{V_A}{X} + \left(\mu - \delta - \frac{\sigma_A^2}{2}\right) T}{\sigma_A \sqrt{T}})$ which shows that the probability of bankruptcy is a function of the distance between the current value of the firm's assets and the face value of its liabilities $\frac{V_A}{X}$ adjusted for the expected growth in assets values $\left(\mu - \delta - \frac{\sigma_A^2}{2}\right)$ relative to assets volatility $\sigma_A$. Authors included 756 bankrupted (chapter X, XI and XII filings) industrial companies during the 1980 – 2000 period and about 13 500 non-failed enterprises. Data was elicited from Moody's Risk Services' Corporate Default database and is comprised of only publicly traded companies. They find that Ohlson (1980) framework outperforms Altman's (1968) in predictive accuracy reached. However, both show significantly lower predictive power compared with the BSM-prob model. A Vuong test certifies that BSM-prob display higher accuracy at the 1% significance level. Moreover, the pseudo-$R^2$ for BSM-prob (0.12) is 20% larger than for O-score (0.10) and twice as larger compared with Z-score (0.6) confirming the higher propensity of BSM-prob in capturing the probability of bankruptcy.

Hillegeist et al. (2003) attest that higher accuracy levels may be achieved considering market-based parameters. However, the great limit to their analysis, and of the whole market-based prediction branch, is represented by the fact that only publicly traded entities can be taken into consideration. Thus, all small and medium size firms are excluded from the BSM-prob model and need to be included in other frameworks.

## 1.7 OTHER RELEVANT RESEARCHES: MACROECONOMIC PARAMETERS

An additional research stream in the corporate default prediction literature is constituted by the analysis of macroeconomic variables and their relationship with companies' risk of failure. Related to it, an interesting article was published in 2010 by Zhou, Lai and Yen. They considered two sample (S1 and S2) with data obtained from the Fundamentals Annual Dataset of Compustat North America regarding 962 pairs of bankrupt and non-bankrupt companies, S1, and 227 bankrupt and 237 non-bankrupt entities for S2. Firms belong to non-financial sectors like energy, materials, industrial, consumer staples, utilities, automobiles, media and others. Further, data pertains to the 1980-2006 period. 23 financial ratios were selected following the best performing in the previous literature. Moreover, four macroeconomic indicators are also added as proxy for the general economic cycle. These are (US) GDP, to account for the country level trend; Personal Income Index, to take care of any significant fluctuation in aggregate goods and services demand; Consumer Price Index, to include any inflationary effects that might affect corporate operations; and M2 index, which reflect the amount of money supplied to the economy. Macro variables are employed in the form of year-to-year ratios other than absolute values: for instance, GDP for year t is defined as the proportional increment with respect to t-1 level. $GDP_t = \frac{GDP_t - GDP_{t-1}}{GDP_{t-1}}$. Authors, along with other popular models adopted as benchmarks, examine the impact of a popular configuration of the Neural Network framework: the Multilayer Perceptron Neural Network (MLPNN). MLPNN is typically composed of an input layer, one or more hidden layers and an output layer, each consisting of several neurons. MLPNN has the advantage of being relatively easy and versatile in identifying inner patterns. It started to be widely adopted after Hect-Nielsen 1987 who proved that a two hidden layers MLPNN can represent any continuous function mapping.

Authors report that when macroeconomic parameters are included, models in general (MDA, Logit, CART, etc.) perform slightly better, with a maximum overall accuracy delta of just under 2% points for the Logit model. Moreover, the most promising model seems to be MPLNN with an overall accuracy of 78.61% points when including macroeconomic indices. The benefit, as

they stress, coming from macroeconomic figures appears to be slight enough that they themselves call for further research to prove whether their adoption may be cost effective in terms of time spent and complexity added for academics and practitioners.

On a similar path can be traced also the work of Ptak-Chmielewska (2019), who studies the influence of macroeconomic variables on small and medium enterprises (SME) in Poland. He specifically takes into consideration 1,138 SMEs operating in the industry, trade and services sectors, half defaulted in the 2002-2010 period and half successful. 16 ratios considered reliable by the previous literature are selected and successively filtered by a clustering variables procedure to avoid collinearity issues. The resultant six ratios (quick ratio, capital share in assets, current assets turnover, operating profitability of sales, net profitability of equity and inventory turnover) are then applied through the Logit model. The results of the logistic regression are successively confronted with the exact same model enriched with three macroeconomic variables: GDP, inflation rate and unemployment rate. Ptak-Chmielewska reports that the overall classification effectiveness was improved in the model with the macroeconomic variables. However, the benefit is significant in type II error while type I error (the one considered to be more important by most of literature researches) do not show any decrease achieving comparable values (33.4%, 33.7%, without and with macroeconomic parameters). Interestingly, as he points out, the application of the macro variables in the model displays improvements on the average classification of non-bankrupt companies.

Thus, again macroeconomic figures seem to only slightly improve the prediction accuracy in the models applied so far in the literature.

To conclude, a table with all relevant researches presented along with a brief description for each one of them. Only the section about ‘definition of bankruptcy’ is not included for it does not add any information on studies developed on the bankruptcy prediction topic but rather takes a different view angle from those already reported.

| LITERATURE UP TO 1966 – THE UNIVARIATE PHASE | |
|---|---|
| FitzPatrick (1932) | Analyse 24 ratios from 29 industrial firms. The ratios were compared with their overall average to look for specific trends affecting failing firms |
| Smith and Winakor (1935) | Studies individual ratios contribution to bankruptcy prediction on a variety of industry sectors |
| Merwin (1942) | Reports that failing firms display signs of weakness starting as early as four or five years before failure looking at small manufacturers |

| | |
|---|---|
| Chudson (1945) | Looks for patterns of companies' financial structure in order to unveil if any factor follows 'normal', repetitive and visible sequences. |
| Jakendoff (1962) | Compared ratios of profitable and unprofitable firms to identify ratios individual role on bankruptcy |
| Beaver (1966) | Compared the mean values of ratios from 79 pairs composed by failed and a non-failed firm belonging to 38 industries. |
| Pinches et al. (1975) | Determining the hierarchical relationship among 48 empirically based financial ratio and proposing those with highest predictive ability |
| Chen and Shimerda (1981) | Conduct a research on the individual best performing ratios adopting Principal Component Analysis. |
| FROM 1966 TO THE 1990S – THE MULTIVARIATE STATISTICAL PHASE | |
| Altman (1968) | Introduces Multivariate Discriminant Analysis. His research is considered pioneering in the field and is treated as comparison benchmark |
| Meyer and Pifer (1970) | Run a multiple linear regression, with a dummy variable to discriminate between viable and less viable banks, based on 28 operating ratios and 4 balance sheet groupings |
| Wilcox (1973) | Develops a binomial model applied to 52 pairs of bankrupted and non-bankrupted industrial firms selected from the Moody's Industrial Manual |
| Bilderbeek (1977) | Applied a step wise discriminant framework to classify 58 Dutch companies |
| Martin (1977) | Selects the logit model to conduct a research on early signs of banks failure commissioned by the Federal Reserve of New York |
| Weinrich (1978) | Aanalyse risk classes, based on six risk-related accounting ratios, in order to predict insolvency on German SME |
| Ohlson (1980) | Applies Logistic regression acieving 96,12 % accuracy. After Altman (1968) his research is considered pioneering |
| Begley et al. (1997) | Investigating the field performances that Altman (1968) and Ohlson (1980) models are actually able to reach. They find Ohlson to be the best performer |
| Zimmer (1980) | Looked at prediction accuracies achieved by loan officers in executing the task of making annual predictions of corporate failure based on a time series of ratios |
| Frydman (1985) | Among the first to apply Recursive Partitioning Algorithm (RPA) on Compustat firms |

| FROM THE 1990S TO PRESENT – THE 'ADVANCED' MODEL PHASE | |
|---|---|
| Bell et al. (1990) | Introduce Artificial Neural Network for prediction purposes |
| Cadden (1991) | Compares Backpropagation Neural Networks with Multivariate Discriminant Analysis finding the former almost always overcaming the latter |
| Luoma and Laitinen (1991) | Apply Proportional Hazards model, in the settings designed by Cox (1972), on Finnish entities |
| Tam and Kiang (1992) | Compare KNN and ID3 with MDA, Logit and Backpropagation neural network on Texas banks |
| Bryant (1996) | Introduce Case-Base Reasoning and compares results with Ohlson (1989) finding higher accuracy |
| Wallrafen (1996) | Employs 73 ratios as random variables on German corporations through Genetic Artificial Neural Network |
| Dimitras (1999) | Apply Rough Set Theory on Greek companies |
| Min and Lee (2004) | Fine tune Support Vector Machines to 1888 Korean's SME |
| Barboza et al. (2017) | Compare Bagging, Boosting, Random Forest, Support Vector Machines, Artificial Neural Networks, Logistic Regression and Multivariate Discriminant Analysis |
| OTHER RELEVANT RESEARCHES | |
| Liang et al. (2014) | Thoroughly analyse CGIs and build a modle to exploit their predictive power |
| Elshahat (2014) | Combine Altman's Z-scores with corporate governance ratios with mixed results |
| Hillegeist et al. (2003) | Show high accuracies applying Black-Scholes-Merton based approaches for predictions |
| Zhou et al. (2010) | Includes GDP based measure to predict North America firms' bankruptcy and reports higher accuracy through the use of macroeconomic parameters |
| Ptak-Chmielewska (2019) | Analysis of SME in Poland through 16 ratios including macroeconomic parameters |

# 2. DATA DESCRIPTION, ASSESSEMENT AND PREDICTION MODELS

This chapter aims at describing the application of six statistical methodologies, namely Logistic regression, Support Vector Machines, K-Nearest Neighbour, Adaptive Boosting (AdaBoost), Decision Tree and XG Boost, to companies' financial statements in order to predict bankruptcy. More specifically, after computing the relevant financial indices and checking for the correlation among them to avoid multicollinearity issues, the above models are trained and tested on purposely pre-processed data to retrieve the degree of accuracy per each model.

To describe how the project was handled and conducted a first general description of the firms data will be carried out along with the Propensity Score Matching procedure, through which data is filtered to obtain an homogeneous starting point for prediction; secondly, it will be detailed how financial ratios are computed, assessed and built from Italian financial statements; finally, the application of statistical and machine learning methodologies is shown.

## 2.1 DATA DESCRIPTION AND PROPENSITY SCORE MATCHING

The following section will comprise a general and statistical description of the data used throughout the project. Furthermore, the Propensity Score Matching (PSM) procedure that has refined the initial non-defaulting dataset is presented. Specifically, the following topics will be undertaken: first, a description of the defaulting sample is set out; following, the PSM procedure is explained; at last, the description of the filtered non-defaulting firms' sample is detailed.

### 2.1.1 The Defaulting Sample

Before delving into knowing the data exploited in the project, it is worth mentioning that data was treated via Excel, a Microsoft Office software and, more often, with Python, an opensource programming language featuring ease of use and flexibility. Further, included inside the Python framework, Numpy, Pandas and Matplotlib, have been among the packages most frequently applied.

Data comprises 29711 non-defaulting firms and 424 failing entities. All companies are based in the Veneto region (North-east of Italy) but they do not necessarily have their major profits coming from the same geographical area. Ten years of financial statements, from 2009 up to 2018, were elicited from each company. Financial statements include balance sheet and income statements in the Italian form. Data was retrieved from the AIDA database, a software belonging to the Bureau Van Dijk, a Moody's Analytics company. Mentioned database easily enables to filter data from any entity based in Italy and to operate conditional selections, e.g.

geographical selections. For this study, it has been of particular utility the possibility of selecting the definition of failure wanted and applying it as further filter on data.

In this context the definition of failure lays in the Italian 'Concordato preventivo' and 'Procedura concorsuale liquidatoria'. To this respect, 'Concordato preventivo' (Arrangement with creditors) is defined in the Italian Civil Code (art. 2221) as an insolvency procedure granted by law to the commercial entrepreneur who is in a state of irreversible insolvency, but who, at the same time, demonstrates that he possesses certain merit requirements in order to escape the negative consequences of bankruptcy. It consists of a formal agreement between debtor and creditors regarding the methods by which all obligations must be extinguished: the bankrupt debtor has to arrange the full payment of the privileged creditors and the payment of a share of the unsecured creditors. The offer that derives from it must be approved by the majority of creditors involved and ratified by the court.

On the other hand, 'Procedura concorsuale liquidatoria' or simply 'Fallimento' (Failure), is established by the Italian Civil Code (art. 2221) in the event of insolvency from behalf of the commercial entrepreneur in order to ascertain the inability of the same to honour all its debts and the overall debt situation towards the various creditors. Its ultimate aim is that of liquidating all the assets of the company and distribute it according to the criterion of par condicio creditorum (without prejudice to legitimate causes of pre-emption). In order to satisfy the largest possible number of creditors, the entrepreneur's assets can be replenished with appropriate actions, in particular through the bankruptcy revocation.

Each and every company set of data, for both failing and non-failing companies, includes 2455 datapoints spread among ten years of financial statements items and other descriptive elements like business name, VAT number, ATECO classification ('ATtività ECOnomiche', business activities), a six figures code to identify the company business activity, ATECO description, province of the headquarter and number of employees.

Geographically, the defaulted entities are located for the majority in Vicenza, 120 firms, and Padova, 96, provinces while the remaining are distributed as follows: 67 from Verona, 57 from Venice, 16 based in Rovigo province and 10 from Belluno. Here density in population is at play: Rovigo and Belluno are by far the least densely populated areas in Veneto and they thus display lower defaults. An interesting observation comes by looking at the national levels of defaults. In fact, the comparison with other Italian provinces indicates a higher than average rate of default in Veneto region. It is particularly impressive that, overall, the 2009-2018 period lists Vicenza and Padova inside the top five provinces for defaulted companies. This is

remarkable because Padova and Vicenza cannot be placed among the most densely populated areas in the Italian landscape. Such interesting evidence could be due to at least two reasons: the first relates to the physiognomy of the typical company in the north-east area in Italy and the second is linked to the recent failure of two major local Banks.

Related to the former, Venetian companies are for the majority pertaining to what is defined, for total annual revenues, net income, overall value and number of employees, as small and medium enterprises (SME). Understanding the reasons and the implications behind this fact is beyond the scope of this analysis; nonetheless, said enterprises configuration can already explain why higher than average failing rate are, ceteris paribus, affecting this area. Indeed, SME tends to be more subjected to adverse economic conditions and fail more rapidly in higher numbers. Since the argument is being analysed in absolute number of defaults per province, then it comes plausible that the higher occurrences are related to it. Another probable reason, possibly added to the previous one, has to do with the recent default of two local major banks: Veneto Banca and Banca Popolare di Vicenza. The two institutions started displaying difficulties beginning from 2015, which may explain why Padova and Vicenza are so high ranking in these last, ensuing years. In fact, the lack of immediate liquidity granted by the banking partners and, more importantly, the loss of millions of euros in savings in the local economy, may have been leading factors in the default of many of the 424 firms taken into account in the analysis. Both conclusions, however, need further exploration.

To better understand the composition of the defaulting firms, three major parameters can be helpful: Total Assets, EBIT (earnings before interest and taxes), and Net Income. To plot levels of the three measures the year 2009 is selected.This should bring at least two advantages compared with other years: first, many of firms included in the sample began filing for failure from 2013 and 2014, so that considering 2009 limits lack of data and better represents entities conditions; secondly, taking values too close to failure might affect the quality of data itself since it may already be gravely affected from the defaulting status.

The following two charts present a comparison between EBIT in 2009 and Net Income in 2009 (Figure 1) and between the same measures and Total Assets in 2009 (Figure 2). Values are reported in thousands of Euros and grouped per deciles (x-axis). The first histogram quite clearly depicts a struggling condition for eight out of ten deciles of companies. Indeed, the first three deciles show negative EBIT and Net Income while the first eight deciles report either negative or almost zero profits. Only the last two deciles of firms seem to produce positive outcomes both in terms of EBIT and profits. However, it seems clear that even the best performing, among the 2009 results, cannot sustain in the long term, the total levels of Assets.

In fact, focusing on the second chart, it stands the constant huge difference between Assets and EBIT and Profits. The sustainability of assets in the long term is a crucial aspect for firms survival: when a company ability of profiting from its activity cannot produce enough value to replace/develop its assets, it will be obliged to find new way of financing or, over years, declare failure.

Though, the clarity offered by the two charts seems to be sufficient to draw conclusions, it is nonetheless crucial taking them as only a first attempt to describe the defaulting dataset. Indeed, at least two external variables may play a role in hiding more subtle conclusions that cannot be drawn from a chart based on absolute levels. First of all, it is important to bear in mind that a timing element may be distorting conclusions. This has to do with the fact that many of the defaulted companies have filed for default only from 2015 onward. Now, given that, following Ohlson (1980), five years before default a company may still be considered to be sound in its fundamentals on average, last deciles of the distribution might positively affect the whole scenario. Indeed, higher values, belonging to far-from-failure entities (the last deciles), may overestimate the actual levels when they are not considered. Secondly, there might be an historical reason as well: 2009 comes right in the middle of the Great Financial Crisis aftermath which may have caused a decrease in all performance indices in both failing and non-failing corporations.

Figure 2.1 Comparison between EBIT 2009 and Net Income 2009, default dataset

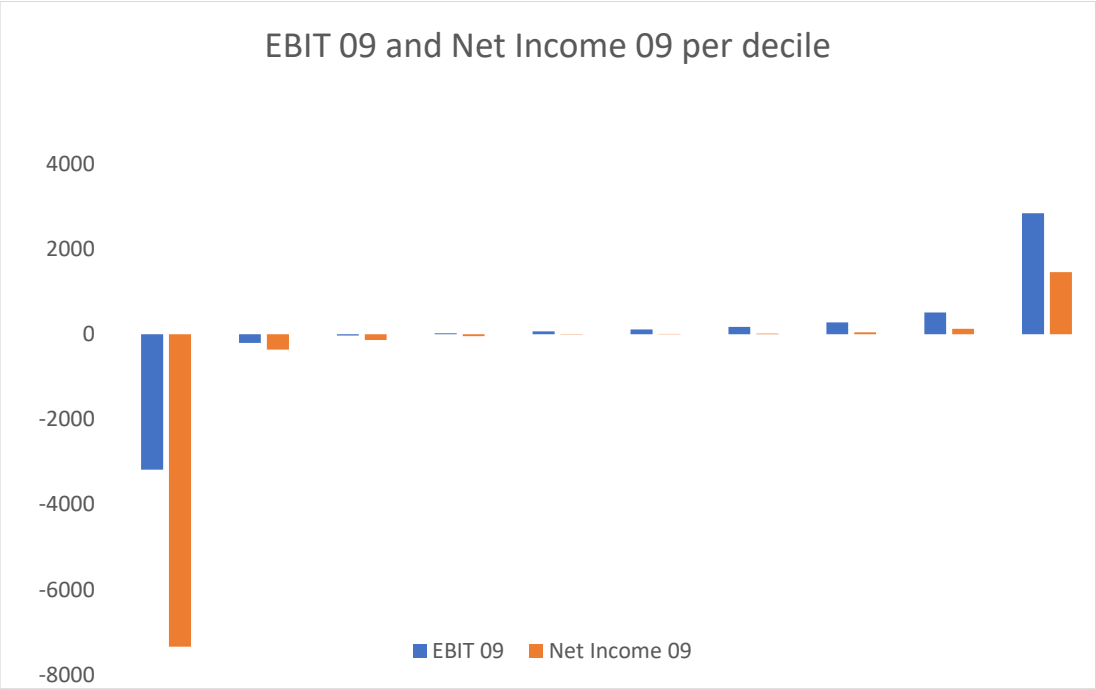
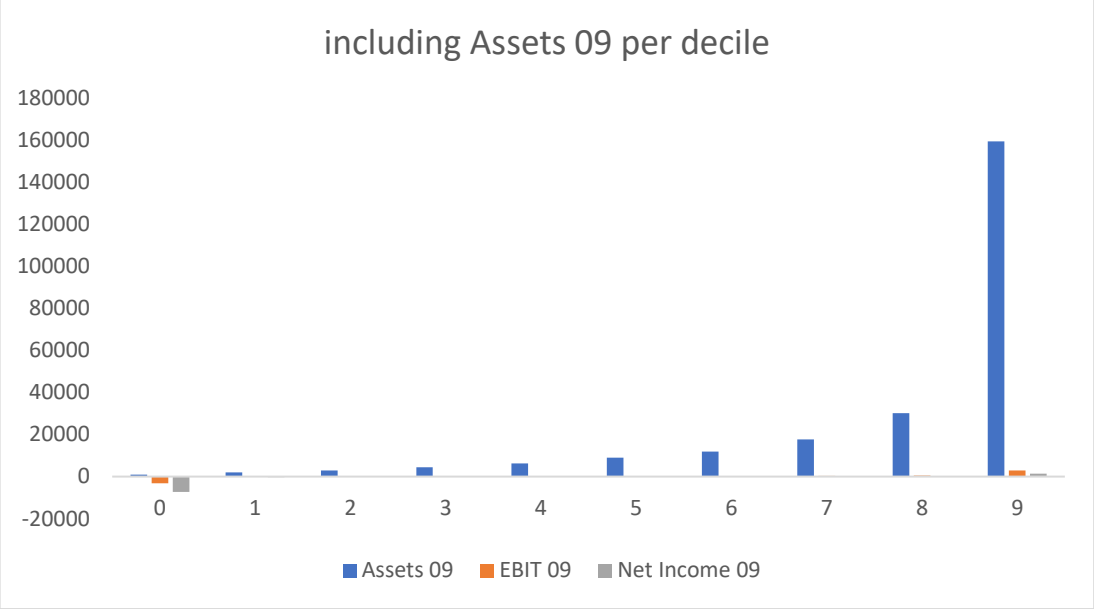


Figure 2.2 Comparison: Total Assets 2009, EBIT 2009 and Net Income 2009, default dataset




The first issue can be tackled by looking at the year-of-default distribution over all defaulted companies. The following table (table 1) display the year of default and the number of defaulted companies for said years.

Table 2.1 Years of default and frequency in the default dataset

| **2013** | **2014** | **2015** | **2016** | **2017** | **2018** | **2019** |
|---|---|---|---|---|---|---|
| 102 | 97 | 67 | 25 | 20 | 38 | 77 |

From Table 1 it can be appreciated that 37.5% (160) of defaulted companies filed for bankruptcy in or after 2016. These should be considered as sound companies in 2009 values for the reason explained before. 31.3% of the reported 37.5% (50 entities) belongs to the last three deciles in Figure 1, indicating that firms defaulting after 2016 seem to be evenly spread through all deciles in their 2009 data. However, if one looks at the mean per decile, related to the last three deciles of companies defaulted between 2016 and 2019 only, the timing issue starts being revealed. Indeed, the mean retrieved for the last decile of 2016-2019 firms reaches for EBIT, 5086,3 (thousands of euros), almost one thousand points above the overall mean of the last decile alone (visible in chart 1). Similarly, the average Net Income stands at 2154,62 (thousands of euros) if computed in the same subsample, almost 400 thousand euros higher than the overall mean. Conversely, the eighth and seventh deciles exhibit a different behaviour: here the average contribution of 2016 onwards defaulted companies is very much in line with the mean value calculated with all defaulted companies. From such evidences on the last three deciles, it can

be concluded that the timing issue affects materially only the nineth decile of the distribution and thus chart 1 overestimates both EBIT and Net Income only in the last decile. Said overestimation is, again, result of choosing 2009 datapoints to describe the sample of the defaulting companies.

On the other hand, the historical reason, linked to understanding whether charts 1 and 2 are effectively representing defaulting companies on average or, on the contrary, macroeconomic factors from the 2009 GFC are at play in the data, can only be resolved by comparing the charts with non-defaulted companies data. This, for sake of simplicity will be carried out after the section related to the Propensity Score Matching procedure through which all 29711 non-defaulting corporations have been filtered to better match the defaulting dataset.

Lastly, to grasp the composition of the defaulting dataset it is worth examining the composition in terms of field of activity. To do so we can rely on the ATECO code which precisely bears the role of defining the sector to which a firm activity belongs to for fiscal purposes. ATECO is commonly composed by six figures that identify a specific industry sector by the following procedure: the first two figures pertain to the division, the third identifies the group, the fourth figure tells the class which is followed by the category and finally (sixth figure) by the subcategory[5]. Each subdivision deepens the specification of the economic sector to which the entity belongs to. For the purposes of this study, only the first figure is considered which provide an indication of the macroeconomic sector of membership. The following table (Table 2) reports the frequency per each ATECO first figure.

Table 2.2 Frequency per each ATECO first figure

| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
|---|---|---|---|---|---|---|---|---|---|
| 3 | 56 | 83 | 27 | 156 | 4 | 67 | 15 | 6 | 0 |

ATECO 0 is composed by agriculture committed firms; ATECO 1 is mainly concerned with the textile, painting and packaging industries; the third macro group is then related to chemical and plastic-linked sectors; ATECO 3 counts a majority of furniture factories; then, the most numerous macro group has to do mainly with real estate related activities; ATECO 5 revolves to editorial communication and logistics; the seventh section is composed by enterprises mainly

[5] To complete the picture there must be mentioned that before any numeric subdivision there is an alfa division, the section, aimed at declaring the macro economic sector in which thee firm operates. The section same information, however, can be found in the first two figure, those considered in the study. These and other details can be found at www.codiceateco.it.

dependent on constructions; ATECO 7 comprises engineering and scientific activities; ATECO 8 includes goods exchanging commercial companies while, finally, ATECO 9, which does not belong to any of the 426 defaulting companies in the dataset, refers to entertainment firms.

### 2.1.2 The Propensity Score Matching procedure

Following Rosenbaum and Rubin (1983) in their examination on propensity scores in observational studies for causal effects, there is a key difference between randomized and non-randomized experiments. If in randomized experiments, the results in the two treatment groups may often be directly compared because their units are likely to be similar, in nonrandomized experiments, such direct comparisons may be misleading because the units exposed to one treatment generally differ systematically from the units exposed to the other treatment. In other terms, any procedure applied to non-randomizable data carries the risk of being inconsistent, biased, due to the precise effect that the impossibility to randomize the sample builds into the model. To avoid this risk it can be helpful computing balancing scores to build a new, pseudo-randomized initial sample. These can be described as functions, b(x), of the observed covariates x such that the conditional distribution of x given b(x) is the same for treated (z = 1) and control (z = 0) units. In this setting, the most trivial balancing score is b(x) = x, what actually happens in randomized experiments. Further, Rosenbaum and Rubin (1983) call the coarsest among the possible balancing score functions 'propensity score' and adduce five theorems whose conclusions may be summarized as follows: (i) The propensity score is a balancing score; (ii) Any score that is 'finer' than the propensity score is a balancing score; moreover, x is the finest balancing score and the propensity score is the coarsest one; (iii) If treatment assignment is strongly ignorable given x, then it is strongly ignorable given any balancing score; (iv) At any value of a balancing score, the difference between the treatment and control means is an unbiased estimate of the average treatment effect at that value of the balancing score if treatment assignment is strongly ignorable. Consequently, with strongly ignorable treatment assignment, pair matching on a balancing score, subclassification on a balancing score and covariance adjustment on a balancing score can all produce unbiased estimates of treatment effects; (v) Using sample estimates of balancing scores can produce sample balance on x.

In this analysis, the propensity score concept conveniently suggests a procedure to filter the non-defaulting dataset making it more aligned with the defaulting sample[6]. Indeed, matching

[6] Rosenbaum and Rubin (1983) were primarily concerned with causal effects in observational studies, a topic not directly link with this thesis. Nonetheless, propensity score matching represents a valuable procedure to identify meaningful matches for defaulting firm on the basis of selected covariates and improve final results.

defaulting and non-defaulting firms on the basis of selected covariates should result in more comparable data sample and thus more reliable prediction results.

To carry out the procedure, there was implemented the PyMatch Python package which deploys a specific instance called Matcher[7]. Matcher makes use of the logistic regression to compute the propensity scores. In general terms, Matcher follows this procedure: it splits defaulting and non-defaulting companies assigning binary values; runs a logistic regression on the basis of the given covariates; it then fine-tunes the relevant threshold from which scores are computed in a random fashion; compute all scores, for both types of firms; and finally, after ranking the obtained scores, search for the closest n non-defaulting matches looping over all defaulted entities. Python code related to this section is available in the appendix 1.

For this project two covariates have been selected: Sales and Equity to total Assets. Sales was reported from the Italian "Ricavi vendite e prestazioni" while Equity to total Assets was computed as the ratio between "Totale Patrimonio Netto" and "Totale Attivo". Such voices have been selected for their relevancy: Sales represents the overall dimension in operations a company is capable of reaching; Equity to total Assets, on the contrary, features the solidity in ownership that connotes a firm (it should indeed be intended as a sort of opposite index of leverage). Ultimately, choosing Sales and Equity to total Assets as covariates, determines matches operated on volume and ownership solidity dimensions. Here any match on the performance in profitability is purposely avoided for its high level of year by year variance and, more importantly, to let its implications and drivers be directly embedded in prediction models.

As it can be seen from the code in appendix 1, only 2009 to 2015 data was loaded to be exploited with PyMatch. This fact relates with the choice of considering only the fifth year prior to default for all defaulted entities. The logic behind such choice looks at implementing the matching procedure avoiding data already affected by a failing condition. On average, as reported above, it is safe considering about five years before default to prevent it. Indeed, matching still-sound firm data brings the advantage of improving the capability of subsequently identifying those parameters that better describe a failing dynamics compared with a successful one. In other terms, examining the historical path followed by companies with similar fundamentals while still both in good shape, should simplify the recognition of the drivers towards default.

[7] PyMatch and its Matcher instance are open source resources available at github.com/benmiroglio/pymatch

Aside from the year-level distinction, only datapoints from the same year could be matched together, an additional subdivision, based on the ATECO code[8], was put in place. For this purpose, ten sub classes were created based on the first figure of the code to give more importance to the business sector in which each firm operates. In this regard, matches could not fall outside the subgroup. At least one disadvantage and one advantage are brought about from this decision. The main disadvantage relates with the low overall number of companies included in the defaulting sample. Indeed, only considering the Veneto region does not guarantee enough datapoints to cover all subgroups. To understand the implication of this disadvantage it is sufficient looking at Table 2. From it, it is immediately clear that subgroup 9, the last reported, has no companies at all. This, in turn, prevents any subgroup 9 non-defaulting firm from been selected in the matching procedure and thus, no final prediction can be truly representative of it. Nonetheless, splitting on the base of the ATECO code increases similarities between the two types and thus refine the final prediction. This happens because increasing similarities, examining only similar sector companies in this case, permits a higher level of precision in the identification of the hidden drivers toward default for a specific enterprise. In other terms, further increasing similarities improves predictions by enhancing the predictive model training procedure.

It is finally worth adding that a number of five non-defaulting entities were decided to be matched with every defaulted firm. Moreover, since companies defaulting years range between 2009 and 2015, it is important to notice that the same non-defaulting entity can be selected for more than one defaulting company not just in the same year, because perhaps it represents the closest score of multiple defaulting datapoints, but also over multiple years.

Before describing the results from the Propensity Score Matching procedure, it is needed to be pointed out that not all of the 426 failing companies were matched through the Matching instance. In fact, only for 395 could be computed propensity scores, while the remaining 31 could not, with the subsequent impossibility of being matched. The reason behind is to be searched among the implications caused by the subdivision implied by the ATECO code summed to the year to year split: where the number of firms per group was not higher than two, the logistic regression could not figure out how to assign scores. To overcome such limitation a final piece of code was supplemented after the Matcher. The 'Manual Matching', as it was titled, proceeds as follows: first a python Data Frame with all the left-out defaulting firms is

[8] As already specified, ATECO first figure does not represent a specific macro business sector per se. Though, the so formed subdivision represents a practical methodology to increase feature similarities among matches.

created; it is then computed a scaling factor ($\frac{Sales}{Equity\ to\ total\ Assets}$) to account for the scale difference between sales and equity to total assets data; after, the distance between the level of defaulting and non-defaulting firms covariates is computed, per each of the two covariates, for all non-defaulting companies; then, the sum of the distance registered from Sales and the scaled distance coming from Equity to Total Assets is carry out per each non-defaulting firm and a rank is established; finally, the five 'closest' non-failing entities, those showing at the top of the rank, are chosen as matches. Of the 31 left-outs however, only 10 found matches with the Manual Matching, all the remaining were discarded for low quality of data reported (they disclosed either zeros or missing values in great proportions).

Thus, a total of 405 failing entities were matched with five non-failing firms for a total of 2430 firms considered for prediction purposes.

To check for the validity of the matches obtained, an Ordinary Least Square regression is run to determine whether a significant difference between the defaulting and non-defaulting groups is in place over the two considered covariates, Sales and Equity to total Assets. To do so, the OLS is configured so that as dependent variable was selected one covariate at the time while as independent variable was selected the dummy constituted by a vector of reflecting the failing, non-failing status of the two groups. If the dummy is determined to be statistically significant at 5% level, then a new OLS regression was run introducing a constant term, and keeping the dummy variable, to control for it. Results indicates that the failing/non-failing dummy is indeed significant at 5% level for Sales, without constant term (Figure 3), p-value at 0,1%, but becomes not statistically significant when the constant term is added, p-value at 6%. Since the p-value concerning the constant term is > 0,001 it can be concluded that it is statistically significant at the 5% level and it is thus appropriate preferring this model to the previous. On the other hand, when running the same OLS regression on Equity to total Assets, both options deliver a not statistically significant dummy with p-values of 12,3% without constant (Figure 4) term and 9,4% with constant term respectively. In this case also the constant term is not statistically significant, p-value at 51,4%.

In either cases, the dummy results to be not statistically significant, which implies that the two groups, failing and non-failing, do not display any difference with respect to either covariate: the Propensity Score Matching procedure has successfully combined data on the basis of the covariates.

Figure 2.3 The result from OLS regression on Sales with and without constant term ('Default' represents the dummy variable, 'const' the constant term).

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| Default | 1.444e+04 | 4202.571 | 3.437 | 0.001 |

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| const | 2.285e+04 | 1821.756 | 12.541 | 0.000 |
| Default | -8402.0379 | 4462.373 | -1.883 | 0.060 |

Figure 2.4 The result from OLS regression on Equity to total Assets with and without constant term

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| Default | -0.2125 | 0.138 | -1.543 | 0.123 |

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| const | 0.0402 | 0.062 | 0.653 | 0.514 |
| Default | -0.2527 | 0.151 | -1.675 | 0.094 |

### 2.1.3 The non-defaulting sample

Since the analysis of the defaulting firms has already been conducted, it is now worth examining general features connotating the non-defaulting sample. Following the same approach applied above, a chart comparing 2009 mean levels of EBIT and Net Income distributed over deciles (Figure 5) and another adding total Assets to the picture (Figure 6), are reported.

After just a quick look, it already become evident an interesting similarity between Figure 5 and Figure 1. Indeed, after a first decile exhibiting high average loss levels in terms of both EBIT and Net Income, the subsequent deciles appear to remain at relatively constant low levels before an upsurge starting from the 9$^{th}$ deciles and boosted in the last percentiles. Curiously though, if the absolute values for EBIT are almost identical between defaulting (Figure 1) and non-defaulting (Figure 5) entities, with physiological differences that can be easily expected, what attract the attention should be the divergence in the levels of Net Income. In fact, almost every decile shows lower profits for the defaulting sample. Moreover, shifting the focus toward the comparison with the average total Assets quantities, again it can be concluded that the condition of the two samples remains almost indistinguishable, not just in terms of trend over deciles but also accounting for absolute values. Strikingly, it actually seems even slightly higher the level of total assets for defaulting companies, on average, than for non-defaulting ones.

These two last considerations, fairly similar EBIT and total Assets and slightly lower levels of profits in a context of similar trend, can be the basis to try formulating an answer to the second issue raised above on the historical role of the GFC and its influence in the data. Observing the four graph, it can be concluded that though it is surely possible that the overall peculiar pattern

affecting all three variables and common to both samples may be resulting from the historical context, the difference in mean Net Income values indicates a clear delta in performance independent from the context per se. Rather, it signals a deterioration in the ability to generate profits of companies inside the defaulting sample with respect to their counterparts, on average, within the historical context of the GFC.

It is finally worth adding a further consideration to the frame. Firstly, Figure 5 first two deciles depict the condition of those firms able to re-establish a flourishing activity and avoid default. It should be noted that these are precisely those companies whose 'sickness' was detected and cured before failure occurs, as in Beaver (1966), with the consequence of making harder the identification of distress drivers and thus carrying the risk of overstating the prediction model reliability.

Figure 2.5 Comparison between EBIT 2009 and Net Income 2009, non-defaulting dataset

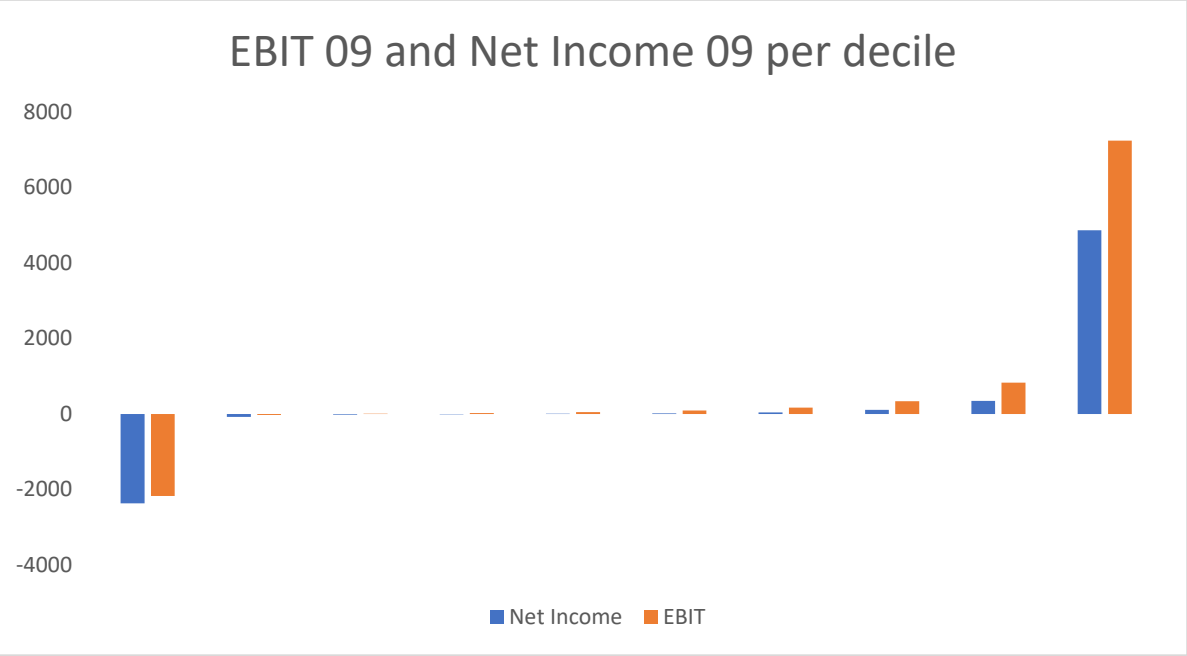
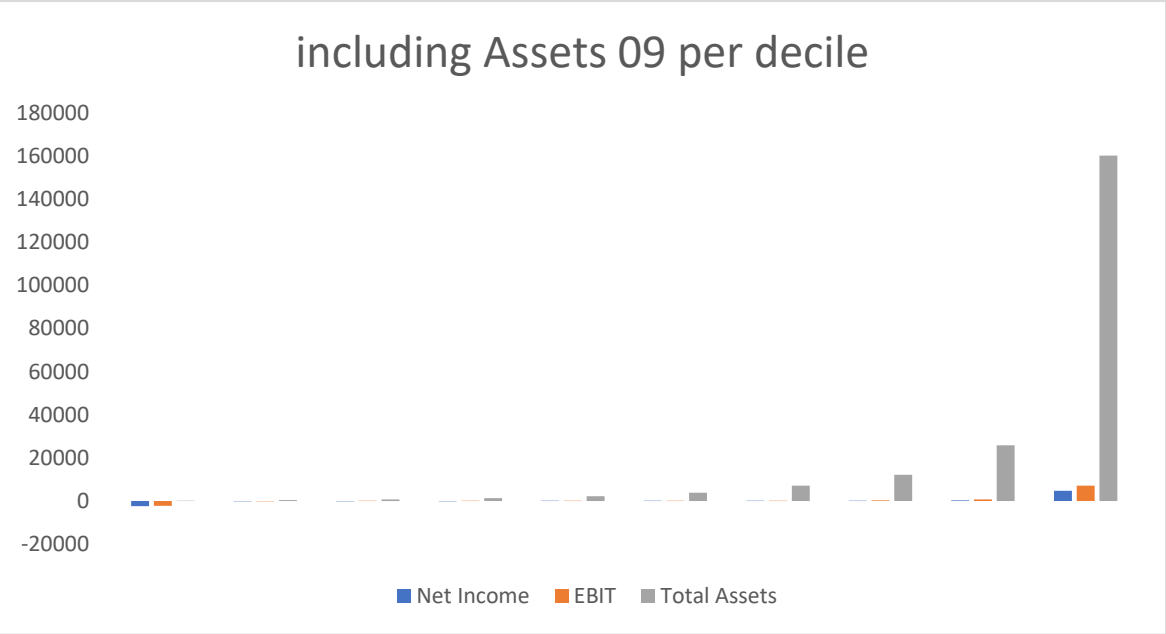


Figure 2.6 Comparison between Total Assets 2009, EBIT 2009 and Net Income 2009, non-defaulting dataset




Two additional features might be examined to deepen the analysis on data from both samples: a comparison of the geographical location of firms and the analysis of the average number of employees. To accomplish the geographical comparison, it is sufficient to integrate to the defaulting entities locations already set out the provinces where non-defaulting firms are headquartered. Table 3 shows the number of firms based in each province for both samples, resulted from the PSM procedure.

Table 2.3 Headquarters of Propensity Score Matched firms by province in Veneto

| **Province** | Belluno | Padova | Vicenza | Venezia | Rovigo | Verona | Treviso |
|---|---|---|---|---|---|---|---|
| **Failing** | 10 | 94 | 113 | 55 | 14 | 63 | 56 |
| **Non-fail.** | 38 | 420 | 478 | 230 | 52 | 431 | 377 |

Table 3 underlines that the vast majority of non-failing enterprises is located inside the provinces of Vicenza, Verona and Padova respectively, followed, not far behind, by Treviso. Except form Verona and Treviso, the results are in line with the failing firms' geographical distribution. Covering all provinces is critical for prediction purposes. Indeed, training models while including all possible conditions, geographical in this case, ensure higher reliability in results when testing predictions on hold-out companies' data. In particular, the presence of enterprises based and operating in more marginal locations like Rovigo and Belluno, guarantees specific parameters related to location of activities, to be indirectly included in the trained model, thus making it more reliable. Nonetheless, questions may arise on the actual number of

firms' data effectively needed to avoid any form of overfitting towards more represented areas. This issue is however not further explored in this study due to the fact that the final frequency per province is not a primary concern when looking for predictions on the Veneto region as whole, given that all provinces are, in any case, represented. In other words, here is not critical a province balanced sample because all Veneto firms are actually assumed to share common basic feature (culture, management style, relation among firms, etc.) regardless from the province of origin.

Secondarily, looking at the average number of employees might better contextualize the data that are going to be used to perform predictions afterward. To analyse it, the 5$^{th}$ year prior to default is taken into consideration for what concerns defaulting firms. The same rule is applied also to non-defaulting companies as follows: for each failing entity, the five Propensity Score Matched sound firms are selected and only the average number of employees belonging to the matching year, is considered (remark that the 5$^{th}$ year is precisely the one exploited in the PSM procedure). Therefore, eventually, only years from 2009 up to 2015 are reported (Figure 7). The chart shows average values for failing, non-failing and all companies together. As expected, blue bars always stand between the orange and the grey ones but closer to the latter given the higher number of occurrences between failing and non-failing entities (1 vs 5). Except for 2010, with lowest levels registered, and 2014, the highest on record, all other years exhibit a fairly similar level of average employment over the enterprises included in this study. Interestingly, the failing sample always display lower number of employees. This may indicate that on the 5$^{th}$ year prior to default, some sign of distress is already at play, which perhaps forces the management to cut costs, and substantially contradicts the belief, true to most of the relevant literature, of firms sounding fundamentals five years before default. However, some other, and possibly more subtle, process might determine the delta employees delineated in Figure 7. Indeed, if a deeper focus is pointed toward the dimension of companies considered, usually SMEs, it comes logically that bigger dimensions companies may misrepresent the average population of failing entities. Larger corporations can in fact be more resilient in hard times with respect to smaller ones due to more capital (assets, liquidity, etc.) to exploit before filing for bankruptcy. This fact may indicate that relatively big firms may endure more than five years before bankruptcy and that are thus to be considered above average inside the framework of the five years rule. Implication of this is that if on the one hand it can still be true that, on average, the 5$^{th}$ year prior to default bears no distress signs, on the other, above average entities may actually be already suffering. Now, given that usually, larger corporations hire more employees, it comes naturally that outliers, bigger than average companies, may be the reason behind the

smaller orange bars. To check for this possibility, Table 4 reports the weight of the last decile on the overall distribution of number of employees for those years reporting higher difference between failing and non-failing entities (2011, 2013, 2014 and 2015). Moreover, it also shows, in 'Delta (absolute)' denominated row, the absolute delta between the average in the last decile and the average computed on all other deciles (in brackets is also present the 1st nine deciles average). Two major observations can be drawn from it. first it is clear that the last decile plays a pivotal role in the determination of the final average number of employees. In this sense, it can be argued that for all four years, the last deciles represent a sort of aggregated outlier, which vastly affects the final outcome: a decrease in the last decile more than proportionally decreases the mean of the whole distribution. Secondly, it should be underlined that the weight seems to be directly correlated with the distance between the orange and grey bars. In fact, where the difference is larger in absolute terms (2014), the weight of the last decile is relatively heavier, whilst it is lighter for closer values (2015). Table 4 cannot be interpreted as a conclusive proof, nonetheless it concisely suggests that few companies may have a determinant role in explaining Figure 7 results. This, in turn, rejects the hypothesis of unreliability over the five years prior to default rule, confirming the literature argument.

Figure 2.7 Number of employees on the 5th year prior to failure

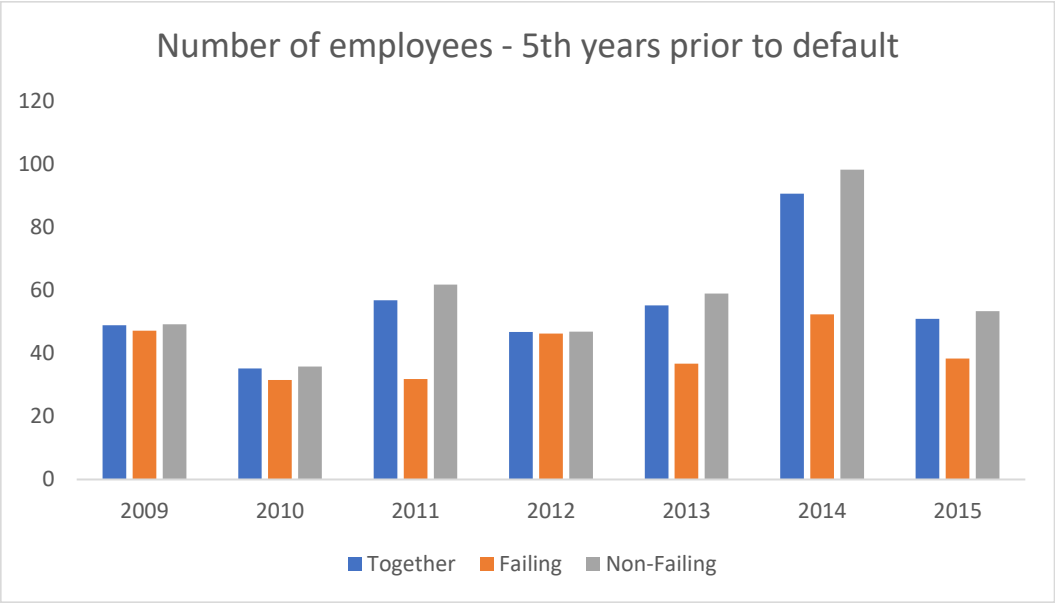



Table 2.4 Weight of last decile on years with greater difference in the number of employees

| **Year** | 2011 | 2013 | 2014 | 2015 |
|---|---|---|---|---|
| **Last decile weight** | 43,4% | 41,2% | 63,8% | 38,4% |
| **Delta (absolute)** | 201 (19) | 119 (24) | 391 (21) | 251 (25) |

## 2.2 FINANCIAL RATIOS

The following section is dedicated to the description of how financial ratios have been built from Italian statements and the successive assessment of their individual quality level. First, the structure of each ratio is described; then, the results from the univariate logistic regression are reported; third, Binning categorization, Weight of Evidence and Information Values valuation are carried out; finally, the examination of ratios interrelations is considered through correlation analysis.

### 2.2.1 Financial Indices Composition

The first crucial step to apply any prediction model has to do with the selection of the parameters that will then be useful for running prediction models. For the purposes of this study, said parameters ought to be found among the various financial indices that can be elicited from financial statements. 54 different ratios have been included, all retrieved from components of either Balance Sheet or Income Statement for each company. Table 5 reports all ratios employed (left-hand side of the table) along with their components (right-hand side). The first 45 have been collected directly from Bellovary et al. 2007 summary research while the last 9 from other authors analysed in the literature dedicated chapter. The table is conceived as follows: each element reported in Italian inside the 'DESCRIPTION' column features the actual quantity elicited from financial statements; when it happens to be marked by an asterisk (*) it is further defined in the bottom attached table. All components find their ultimate definition in Italian financial statements quantities. Moreover, the only difference between upper-cased and lower-cased components is referred to whether the component is an aggregate item (upper-case) in the statement or not (lower-case). For sake of clarity upper- and lower-cased items have been integrally transcribed. Python code tailored for the formation of indices can be found in appendix 2.

Table 2.5 List of ratios employed and their components

| RATIO | DESCRIPTION |
|---|---|
| **Net Income/Total Assets** | UTILE/PERDITA DI ESERCIZIO **/** TOTALE ATTIVO |
| **Total Debt/Total Assets** | TOTALE DEBITI **/** TOTALE ATTIVO |
| **Net Income/Equity** | Utile/perdita di esercizio **/** TOTALE PATRIMONIO NETTO |
| **Total Liabilities/Total Assets** | Total Liabilities* **/** TOTALE ATTIVO |
| **Inventory/Sales** | Var. rimanenze prodotti **/** Ricavi vendite e prestazioni |
| **Operating Income/Total Assets** | RISULTATO OPERATIVO **/** TOTALE ATTIVO |
| **Net Income/Sales** | Utile/perdita di esercizio **/** Ricavi vendite e prestazioni |
| **Long-term debt/Total Assets** | Totale debiti oltre l'esercizio **/** TOTALE ATTIVO |
| **Total liabilities/Equity** | Total Liabilities* **/** TOTALE PATRIMONIO NETTO |

| | |
|---|---|
| **Operating expenses/Operating income** | COSTI DELLA PRODUZIONE **/** RISULTATO OPERATIVO |
| **Current Ratio** | Current Assets* **/** Current Liabilities* |
| **Working Capital/Total Assets** | Net Working Capital * **/** TOTALE ATTIVO |
| **Retained earnings/Total assets** | Utile/perdita a nuovo **/** TOTALE ATTIVO |
| **Current Assets/Total Assets** | Current Assets* **/** TOTALE ATTIVO |
| **Current Liabilities/Total Assets** | Current Liabilities* **/** TOTALE ATTIVO |
| **Current Assets/Sales** | Current Assets* **/** Ricavi vendite e prestazioni |
| **Working Capital/Equity** | Net Working Capital * **/** TOTALE PATRIMONIO NETTO |
| **quick ratio (quick ass/current liabilities)** | Quick Assets* **/** Current Liabilities* |
| **Sales/Total assets** | Ricavi vendite e prestazioni **/** TOTALE ATTIVO |
| **quick assets/Total assets** | Quick Assets* **/** TOTALE ATTIVO |
| **quick assets/Sales** | Quick Assets* **/** Ricavi vendite e prestazioni |
| **EBIT/Total assets** | RISULTATO OPERATIVO **/** TOTALE ATTIVO |
| **EBIT/Interest** | RISULTATO OPERATIVO **/** TOTALE PROVENTI E ONERI FINANZIARI |
| **Working capital/Sales** | Net Working Capital * **/** Ricavi vendite e prestazioni |
| **CFO/Total assets** | Cash Flow from Operations* **/** TOTALE ATTIVO |
| **CFO/Total debt** | Cash Flow from Operations* **/** TOTALE DEBITI |
| **CFO/Sales** | Cash Flow from Operations* **/** Ricavi vendite e prestazioni |
| **CFO/Current Liabilities** | Cash Flow from Operations* **/** Current Liabilities* |
| **CFO/Total liabilities** | Cash Flow from Operations* **/** Total Liabilities* |
| **Cash/Total Assets** | TOT. DISPON. LIQUIDE **/** TOTALE ATTIVO |
| **Equity/Total Assets** | TOTALE PATRIMONIO NETTO **/** TOTALE ATTIVO |
| **Total Debt/Equity** | TOTALE DEBITI **/** TOTALE PATRIMONIO NETTO |
| **Cash/Current Liabilities** | TOT. DISPON. LIQUIDE **/** Current Liabilities* |
| **Equity/Total liabilities** | TOTALE PATRIMONIO NETTO **/** Total Liabilities* |
| **no-credit interval (Current Ass/Daily Operating expenses)** | Current Assets* **/** (COSTI DELLA PRODUZIONE/365) |
| **Asset Turnover** | Ricavi vendite e prestazioni **/** [(TOTALE ATTIVO (t-1) + TOTALE ATTIVO (t)) /2] |
| **Return on Total Asset** | RISULTATO OPERATIVO **/** TOTALE ATTIVO |
| **Ebitda/EBIT** | EBITDA* **/** RISULTATO OPERATIVO |
| **CFO/EBIT** | Cash Flow from Operations* **/** RISULTATO OPERATIVO |
| **Tax Expenses/EBIT** | Totale Imposte sul reddito correnti, differite e anticipate **/** RISULTATO OPERATIVO |
| **Other Revenues/Total Produced Value** | Altri ricavi **/** TOT. VAL. DELLA PRODUZIONE |
| **Cash Flow ratio** | Cash Flow from Operations* **/** Current Liabilities* |
| **Interest Coverage** | EBTDA* **/** TOTALE PROVENTI E ONERI FINANZIARI |
| **Cash Flow from Operations** | Cash Flow from Operations* |
| **log(Total Assets)** | Log (TOTALE ATTIVO) |
| **Turnover Payables** | Cost of Good Sold* x 1,22 **/** (Fornitori entro + Fornitori oltre) |
| **Turnover Receivables** | Cost of Good Sold* x 1,22 **/** (Cred. vs Clienti entro + Cred. vs Clienti oltre) |
| **Turnover Inventory** | (COSTI DELLA PRODUZIONE - Incrementi di immob.) **/** TOTALE RIMANENZE |

| | |
|---|---|
| **Acid Ratio** | (ATTIVO CIRCOLANTE - TOTALE RIMANENZE) **/** Current Liabilities |
| **Net sales/Cash from sales** | Ricavi vendite e prestazioni **/** Cash from Sales* |
| **Sales/Net Account Receivables** | Ricavi vendite e prestazioni **/** Total Customer Receivables* - Svalut. crediti |
| **CFO/Financial Debt** | Cash Flow from Operations* **/** (Banche entro + Banche a lungo + Altri finanziatori entro + Altri finanziatori oltre) |
| **Fixed Charges Cash Coverage** | (Delta Principal* + Totale Oneri finanziari + Cash Flow from Operations*) **/** Current Liabilities* |
| **Fixed Charges EBIT Coverage** | (Delta Principal* + Totale Oneri finanziari + RISULTATO OPERATIVO) **/** Current Liabilities* |
| *** | *** |
| * **Total Liabilities** | TOTALE PASSIVO - TOTALE PATRIMONIO NETTO |
| * **Current Assets** | TOT. DISPON. LIQUIDE + Crediti a breve + CREDITI FIN. A BREVE + TOTALE RIMANENZE |
| * **Current Liabilities** | DEBITI A BREVE + Obblig.ni entro + Obblig.ni convert. entro |
| * **Net Working Capital** | Current Assets - Current Liabilities |
| * **Quick Assets** | Current Assets - TOTALE RIMANENZE |
| * **Cash Flow from Operations** | EBITDA* - Delta Net Working Capital* |
| * **EBITDA** | RISULTATO OPERATIVO + TOT Ammortamenti e svalut. |
| * **Delta Net Working Capital** | Net Working Capital (t) - Net Working Capital (t-1) |
| * **Cost of Good Sold** | Materie prime e consumo + Servizi + Godimento beni di terzi |
| * **Cash from Sales** | Ricavi vendite e prestazioni - Delta Customer Receivables* |
| * **Delta Customer Receivables** | Total Customer Receivables* (t) - Total Customer Receivables* (t-1) |
| * **Total Customer Receivables** | Cred. vs Clienti entro + Cred. vs Clienti oltre |
| * **Delta Principal** | Principal* (t) - Principal * (t-1) |
| * **Principal** | Obblig.ni entro + Obblig.ni oltre + Soci per Finanziamenti entro + Soci per Finanziamenti oltre + Banche entro + Banche a lungo + Altri finanziatori entro + Altri finanziatori oltre + Titoli di credito entro + Titoli di credito oltre |

In general, the most important characteristic of financial indices concerns with the high level of comparability between different companies that they allow. Indeed, the primary reason why they can be adopted for prediction goals, as in this research, is precisely linked to the possibility of uniformly treat all datapoints regardless of the underling dissimilarities (firm structure, indices components absolute levels, etc.). Further, since each single ratio expresses only a specific feature of the data being analysed, it logically follows that the combination and use of multiple indices could be a reliable way to look at multiple characteristics on all companies' data altogether. In other words, the adoption of indices is key for the employment of

multivariate models, able to combine numerous ratios and thus aggregate the (comparable) knowledge they carry. Nevertheless, before applying any multivariate technique, it is essential conducting an examination of both the quality and interconnectedness of ratios. This would bring two main benefits: first, the identification of ratios individual ability to predict bankruptcy can already indicate which items is best to include in the multivariate approach as well as set a minimum level of accuracy that needs to be overcome by multivariate techniques; second, the determination of the correlation level of each financial ratio with all others can clarify the need to select some items against their correlated counterparties to avoid multicollinearity[9] issues. To achieve them, three analysis are pursued: a univariate logistic regression, to look for the individual performance in prediction; a binning categorization followed by the computation of each bin weight of evidence and information value, to further explore each index ability to perform predictions; and an average correlation classification, to, again, prevent any multicollinearities due to indices carrying similar information.

### 2.2.2 Univariate Logistic Regression analysis

For the univariate logistic regression, the average index value of four years prior to default were considered. For non-defaulting firms, the identification of the four years is accomplished by considering as reference year the defaulting year for the matched bankrupt entity. To clarify, the five non-failing enterprises matched with the same failing firm through the Propensity Score procedure, take as reference year the defaulting year of the failing firm. From the reference year, the average value of the index is computed including the previous four years data. Whenever four years were not available[10], only three years were taken into account. Average values, collected in a matrix composed by as many rows as companies and columns as ratios, are then employed to run the univariate logistic regression.

[9] To understand the issue related with multicollinearity few lines from M. P. Allen, 1997, Understanding Regression Analysis, are reported: 'Other things being equal, an independent variable that is very highly correlated with one or more other independent variables will have a relatively large standard error. (…) Multicollinearity exists whenever an independent variable is highly correlated with one or more of the other independent variables in a multiple regression equation. Multicollinearity is a problem because it undermines the statistical significance of an independent variable. Other things being equal, the larger the standard error of a regression coefficient, the less likely it is that this coefficient will be statistically significant.' (p. 176, M. P. Allen, 1997, Understanding Regression Analysis, published by: Springer, Boston, MA).

[10] The unavailability of all four years occurred only to those indices computed through the use of delta components and only for companies with 2013 as reference year. Delta components are those items resulting from the difference between the value of the index at time t and the same index at time t-1. Example of this class of indices can be Cash Flow from Operations which results from EBITDA - Delta Net Working Capital, where the latter, a delta component, is precisely defined as Net Working Capital (t) - Net Working Capital (t-1). These indices lack from the 2009 values because delta components cannot be computed for 2009 (t) since no records are available for 2008, (t-1).

The model entails three steps to be undertaken: data pre-processing; the selection of the relevant parameters (solver function, proportion of train and test sets, etc.); and the definition of the measures of performance.

Data pre-processing, the procedure that takes care of the arrangement of data to ensure model efficiency and reliability in results, has been conducted on outliers and missing values (also known as NaN values inside the Python framework). In practice, pre-processing was handled as follows: first, outliers have been spotted by standardizing all values from the 4 years averages matrix, index by index, through the relation $z = \frac{x-m}{sd}$ (where z is the standardized value, x the initial value, m the mean of all values and sd the standard deviation) and looking for $|z| > 3$; secondly, all spotted outliers are converted into NaN values and the matrix is restored to the initial values; and finally, all missing values are filled with the mean of all other values in the same column (where each column is related to a specific financial index).

Afterwards, the selection of the relevant parameters was performed. In this case, two parameters have been modified from the default settings[11]. First, the 'solver' of the Logistic Regression Class has been set to 'lbfgs' in accordance with the best performing solver for the available samples. Solver is the name of the algorithm portion assigned to carry out the computational task involved with the application of the logistic regression. Lbfgs refers to the limited-memory version of the BFGS, the iterative method for solving unconstrained nonlinear optimization problems based on the work of Broyden, Fletcher, Goldfarb and Shanno[12]. Secondly, the test size proportion was changed to 0.25 to result in a division of 75% of items randomly selected for the training set and 25% kept as hold-out testing sample.

Finally, five measures of performance are considered: a confusion matrix reporting the number of True Negatives, non-defaulting companies correctly predicted, False Negatives, non-defaulting companies predicted as failing, True Positives, failing firms correctly spotted, and False Positives, defaulting companies predicted as non-defaulting (Figure 8 shows the conventional format for confusion matrices); the Recall measure (also known as sensitivity or true positive rate) or the ability of a model to find all the relevant cases within a dataset, is defined as the number of True Positives over the total occurrences of Positives (True Positives + False Negatives); the Precision measure (also known as positive predicted value), or the

[11] The settings in question relates to those provided by Sklearn.linear_model.LogisticRegression, an opensource python based statistical package that can be recovered from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[12] More on this topic can be found on Saputroa and Widyaningsih (2017) paper available at https://aip.scitation.org/doi/pdf/10.1063/1.4995124

ability of a classification model to identify only the relevant data points, is defined as the number of True Positives to the total number of predicted Positives (True Positives + False Positives); the Accuracy measure, entailing the aggregate performance of the model in prediction, is defined as the sum of True Positives and True Negatives to the sum of all Positives and all Negatives (TP + FP + TN + FN), which, in other words, refers to the frequency of the correct predictions; and the Receiving Operating Characteristics Area Under Curve (ROC AUC), equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').

Figure 2.8 the Confusion Matrix conventional format

| | | **Predicted Values** |
|---|---|---|
| **Actual Values** | True Negatives | False Positives |
| | False Negatives | True Positives |

For sake of clarity, only two among the 54 financial ratios will here be presented in their results. All results can however be explored in appendix 3. Also, Python code referred to the univariate examination of indices can be found in the same appendix, right after results. For the univariate logistic regression sklearn opensource package was employed.

The two selected ratios are Net Income to Total Assets (NI-TA) and Working Capital to Equity[13] (WC-NW). The former resulted to be among the best individual performers while the latter displayed lower than average achievements. This difference should be helpful in marking pros and cons of the measures of performance considered.

First of all, both confusion matrices are transcribed (Table 6). Divergencies are immediately visible: if, on the one hand, NI-TA shows 430 True Negatives and 90 True Positives, with a total of 520 correct predictions; WC-NW, on the other only stands at 121 correct predictions with 117 True Positives and only 4 True Negatives. Moreover, although WC-NW seems to perform better than NI-TA in terms of False Negatives (0 against 27 respectively), the opposite situation occurs for the False Positives but at higher order of magnitude (a staggering 487 for WC-NW versus 61 for NI-TA). Overall, the two confusion matrices, who both total 608 cases (the number of cases grouped inside the test set), suggest a situation in which False Negatives are more easily handled while False Positives are missed at much higher frequency by the logit

[13] From now on Equity will be reported as 'Net Worth'. From it 'WC-NW'.

model. In addition, they already convey the intuition of a better performing NI-TA against WC-NW.

Table 2.6 Confusion Matrices for univariate NI-TA and WC-NW.

| **NI-TA** | |
|---|---|
| 430 | 61 |
| 27 | 90 |

| **WC-NW** | |
|---|---|
| 4 | 487 |
| 0 | 117 |

The second performance measure is Recall ($\frac{True\ Positives}{True\ Positives+False\ Negatives}$). NI-TA reaches 0.822497 (82%) as result of the 27 incorrectly predicted False Negatives. WC-NW, on the other hand, overcomes it achieving 100% recall. Logically, zero errors in terms of False Negatives, come at the cost of increasing the rate of False Positives.

Further on, Precision ($\frac{True\ Positives}{True\ Positives+False\ Positives}$) is reported. Here, a reversed scenario is depicted: NI-TA records 0.768473 (77%) while WC-NW only generates a precision of 0,2402464 (24%).

From Recall and Precision measures, again, it can be pointed out that the logit model performs better on the identification of False Negatives than False Positives, which is why Recall outperforms Precision in both indices.

The fourth and most comprehensive measure of performance is Accuracy ($\frac{True\ Posits + True\ Negats}{Negatives + Positives}$). As expected, NI-TA accuracy, at 0.855263 (86%) outruns WC-NW's, at 0.199013 (20%), due to mainly WC-NW inability to correctly classify False Positive cases.

Finally, the last measure of performance is given by the area under curve, where the curve in question is the receiving operating characteristics curve. It charts the False Positive Rate ($\frac{False\ Positives}{False\ Positives+True\ Negatives}$) against the True Positive Rate (the Recall measure). Moreover, a dotted line representing the diagonal is visualized to compare the curve against its halfway level[14]. The comparison against Figure 8 and Figure 9 confirms the conclusion obtained through the accuracy measure. First, the curve for NI-TA increases faster than WC-NW toward high level of TPR while keeping low the FPR. Secondly, NI-TA AUC (0.88) more than doubles WC-NW's, confirming the higher reliability on the logit model applied with NI-TA.

[14] Indeed, a ROC curve precisely lays on the diagonal would imply an AUC of 0.5.

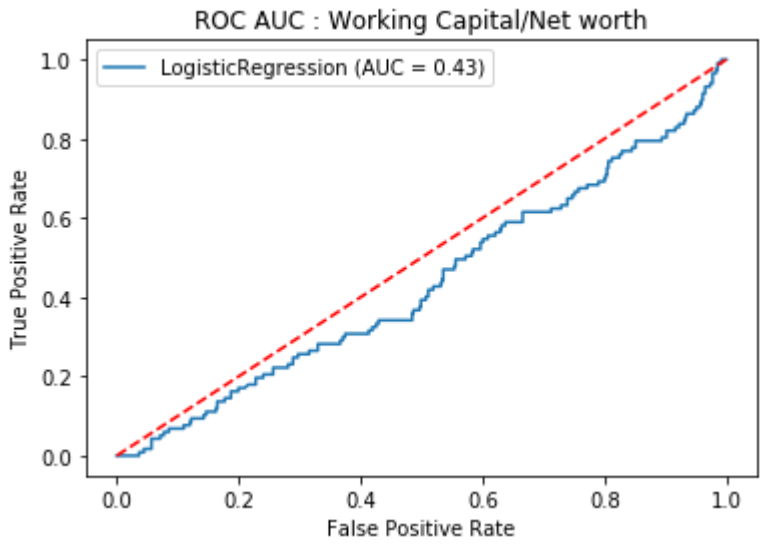



Figure 2.8 ROC AUC for Net Income to Total Assets

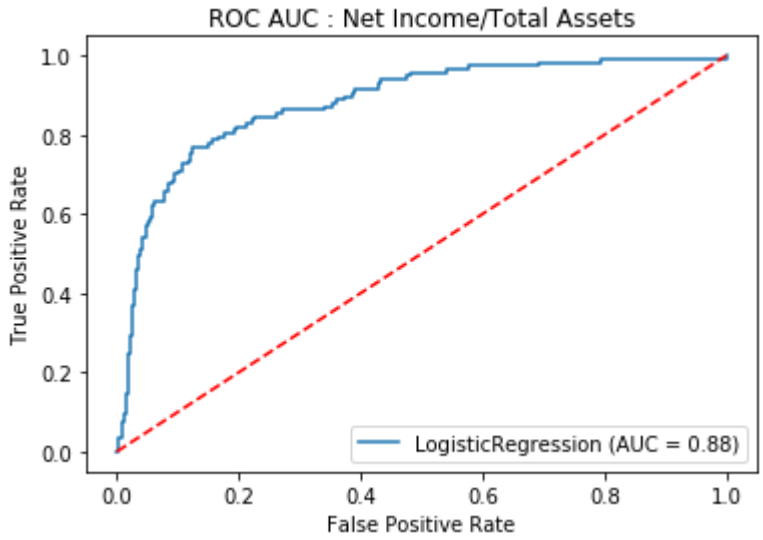



Figure 2.9 ROC AUC for Working Capital to Net Worh

### 2.2.3 Binning, Weight of Evidence and Information Value

In general, binning can be defined as a categorization process aimed at transforming a continuous variable into a small set of groups (also known as bins). It usually serves the purpose of reducing the scale of data being used to better examine the prediction ability of a variable (a financial ratio) inside a model[15], prior to the application of the model itself. Usually, in presence of multiple potential parameters to be included in a multivariate prediction model (as this case

[15] More on binning can be found in Zeng (2014), *A Necessary Condition for a Good Binning Algorithm in Credit Scoring*, available at https://www.researchgate.net/profile/Guoping_Zeng/publication/264455896_A_Necessary_Condition_for_a_Good_Binning_Algorithm_in_Credit_Scoring/links/5675770908aebcdda0e46b34.pdf

entails), binning introduces the sequence of procedures to eventually compute the Information Value through which operate a selection among all variables available[16].

After the creation of bins to summarize the distribution of a specific parameter, the Weight of Evidence (WoE) for each bin can be computed. WoE is defined as ln ($\frac{\%\ of\ non-events}{\%\ of\ events}$), where the numerator indicates the frequency of non-occurrences of the considered bin against all non-occurrences related to the observed variable ($\frac{n\ of\ non-events\ in\ bin_i}{\sum(n\ of\ non-events\ in\ bin_i)}$) and the denominator refers to the frequency of occurrences against all occurrences of the variable ($\frac{n\ of\ events\ in\ bin_i}{\sum(n\ of\ events\ in\ bin_i)}$). Specifically, non-occurrences are to be assigned to the number of non-defaulting firms while occurrences to defaulting ones. The natural logarithm establishes how WoE results should be read: if WoE < 0, then the percentage of default occurrences exceeds the percentage of non-defaults, else the opposite holds[17].

Finally, with the availability of frequency of non-events and events and WoEs, the Information Value (IV) for each bin can be calculated. IV is defined as

$\sum(\%\ non-events_i\ -\ \%\ events_i) * WOE_i$ and represents an aggregate measure of the quality of a certain variable for prediction objectives.

Similarly to the univariate section above, to illustrate binning, WoE and IV only three ratios have been selected: Net Income to Net Worth (NI-NW), CFO to Current Liabilities (CF-CL) and EBIT to Total Assets (EB-TA). Again, all other ratios results, along with Python code applied can be found in appendix 4.

As a first step, binning is applied. In this case bins have been made corresponding to the deciles of the distribution of both indices. Following, for every decile it has been calculated the frequency of failing and non-failing companies and results have been charted in Figure 10, 11 and 12. Charts report on the y-axis the proportion of failing companies and on x-axis the sequence of deciles. What is more, a red dotted line is added as to show the best fitting line for all ten points.

The graphs already represent an initial proxy to assess the quality of the ratio for predictive purposes. Indeed, it can logically be assumed that ratios displaying higher concentration of default events in the first (last) deciles and lower in the last (first) ones, convey the knowledge

[16] In this project, the procedure adopted cannot be technically reminded to the binning concept since no continuous variable is considered. Nonetheless, given the similarity with the binning process, it will continue to be referred to as binning.

[17] Where the rare result of ln(1), is considered to be not interesting.

of higher risk of default for ratio values belonging to the initial (final) part of the distribution. In principle, the bigger the gap between the higher defaults zone and the lower default zone, the more precisely the prediction should result. Instead, whenever the ratio distribution of default frequencies deviates from describe pattern, the prediction model becomes either more complex or less reliable.

In Figure 10, EB-TA does in fact show a behaviour pretty close to the ideal: a high concentration of defaulting cases appears only on the first three deciles while from the fourth onward, a clear decrement in bankruptcy events occur. CF-CL instead, seems to follow a much less linear pattern: firstly, the scale of difference between the high and low frequency zones is much smaller than EB-TA; moreover, the overall trends appears to be more chaotic and thus less interpretable and reliable in prediction. Finally, Figure 12 reports the distribution pattern followed by NI-NW. Although at first glance the model does not seem to follow the ideal pattern, the overall accuracy it registers reaches 75,8%, one of the highest of all recorded via the univariate logit. The reason of it has to be looked for in Figure 12: tough there is not only a single high frequency of default zone, the two displayed are pretty concentrated in the first and last deciles. What is more, the gap between the high frequency zones and the low one is material. In this case it can be concluded that the variable can be highly performing in predictions even though the predictive model cannot follow a linear approach. Accuracies confirm the reasoning: EB-TA, 83,8%; CF-CL, 28,9%; NI-NW, 75,8%.

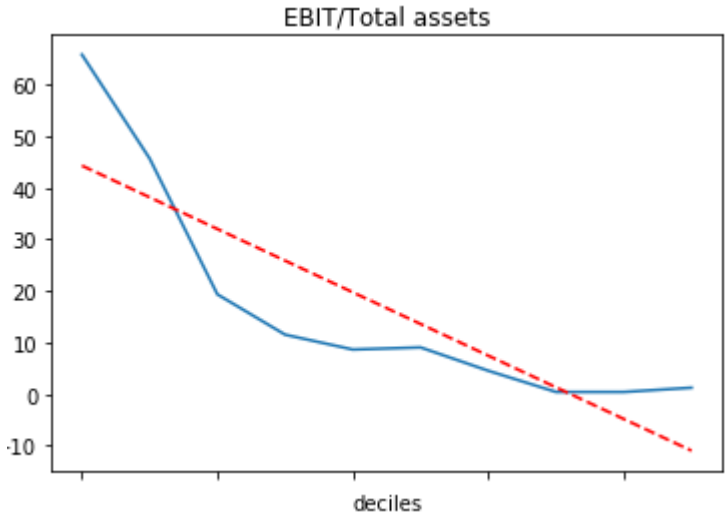



Figure 2.10 Binning chart for EB-TA

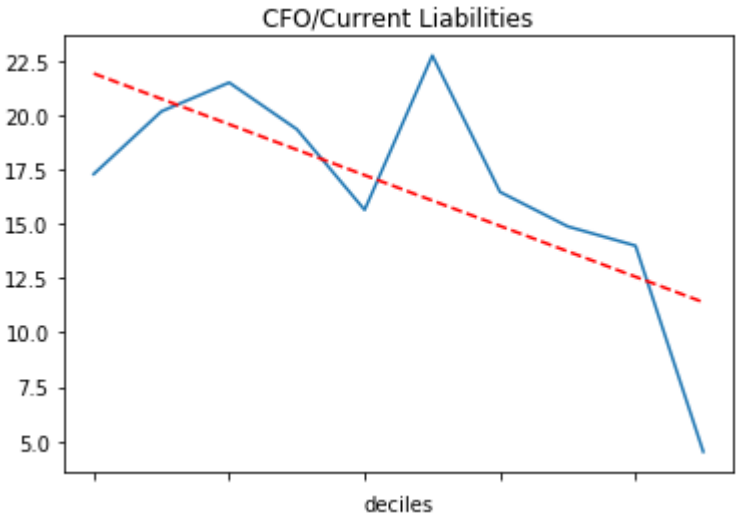



Figure 2.11 Binning chart for CF-CL

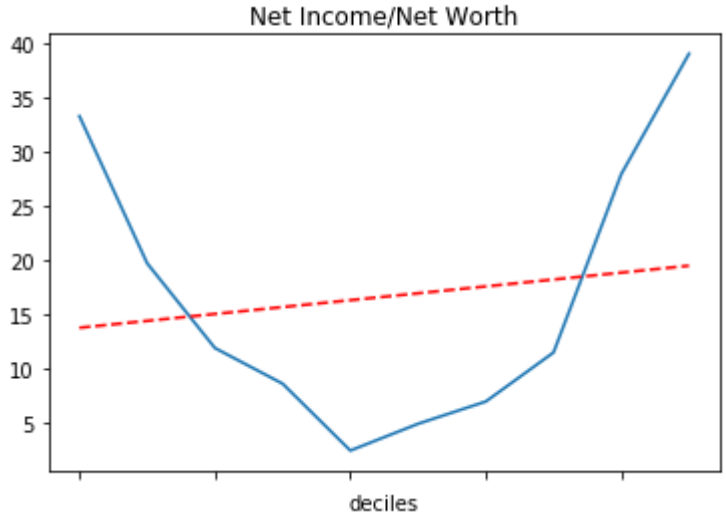



Figure 2.12 Binning chart for NI-NW

From the charts is also clear the role of the line of best fit (dotted red): its slope absolute level can be interpreted as a proxy for eliciting whether the ratio is either linearly reliable or not. In other terms, it signals whether the index can be a 'good' candidate for linear prediction approaches (high absolute slope) or 'bad' one (low absolute slope), where 'bad' indicates either low quality or higher level of complexity for it to elicit valuable knowledge.

After the binning procedure is applied, WoE values are computed and the overall IV is generated. To illustrate it, Table 7 shows the results obtained per each decile considering EB-TA.

Table 2.7 WoE and IV computation procedure results for EB-TA

| Decile | # of non-events | # of events | % of non-events | % of events | WoE | IV |
|---|---|---|---|---|---|---|
| 1 | 83 | 160 | 4,10% | 39,51% | -2,266 | 0,802 |
| 2 | 132 | 111 | 6,52% | 27,41% | -1,436 | 0,300 |
| 3 | 196 | 47 | 9,68% | 11,60% | -0,181 | 0,003 |
| 4 | 215 | 28 | 10,62% | 6,91% | 0,429 | 0,016 |
| 5 | 222 | 21 | 10,96% | 5,19% | 0,749 | 0,043 |
| 6 | 221 | 22 | 10,91% | 5,43% | 0,698 | 0,038 |
| 7 | 232 | 11 | 11,46% | 2,72% | 1,439 | 0,126 |
| 8 | 242 | 1 | 11,95% | 0,25% | 3,879 | 0,454 |
| 9 | 242 | 1 | 11,95% | 0,25% | 3,879 | 0,454 |
| 10 | 240 | 3 | 11,85% | 0,74% | 2,773 | 0,308 |
| **Total** | **2025** | **405** | | | | **2,416** |

After the computation of all financial indices IVs an important issue is to understand how to interpret them. If on the one hand, the benefit brought by IVs can be easily recognized when comparing ratios predictive quality, on the other, it is not immediately comprehensible how to evaluate IV in absolute terms (or, better, relatively to other studies comparable results). Indeed, as pointed out above, the higher the IV score the more performing the index should be. This principle allows for ranking all ratios and run prediction trials on different numbers of ratios always including the best performers. Nonetheless, without any other comparable data, also retrieved outside of this study, nothing can be said by IVs per se[18].

It can be however useful to look at the sorted distribution of the obtained IVs to shed light on its boundaries and relevant thresholds. To do so, Table 8 exhibit the decile of IVs distribution.

Table 2.8 Deciles from the sorted distribution of all financial ratios IVs

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 0,0619 | 0,1018 | 0,1227 | 0,16560 | 0,2395 | 0,44846 | 0,68317 | 0,83129 | 1,73791 | 2,81963 |

### 2.2.4 Correlations among financial indices

The third and final examination undertaken is related with the observation of all interrelations each ratio demonstrates with all others. This is of particular importance to avoid including in

[18] Siddiqi (2006) is one of the few authors found to report thresholds to interpret IVs in the credit scoring field. His results however, cannot be applied to this context for the differences between the treated topics

the predictive model, indices carrying the same fundamental information. In other terms, whenever two or more ratios, belonging to the same firm, exhibit a fairly similar behaviour over time, it logically follows that the knowledge they bear, shares some affinity. Further, such affinity increases with the level of correlation. Then, if a certain correlation threshold is overcome the issue known as multicollinearity might appear, along with the consequences discussed above.

In practice, all correlations have been computed with the Pearson method through the '.corr()' instance available in the Pandas package[19]. Pearson correlation coefficient for a population is defined as $\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$, where the numerator corresponds to the covariance between X and Y variables and the denominator normalizes the numerator through X and Y standard deviations.

All Python code created to compute correlations is available in appendix 5.

To achieve a matrix with all average correlations among indices, the following steps have been executed: first, to ensure a reliable measure, only non-defaulting companies are included in the computation, thus all bankrupt companies' ratios are discarded[20]; second, to account for those indices derived by 'delta components' (explained above), only the period between 2010 and 2018, extremes included, is taken into account; third, a correlation matrix is computed for all remaining firms through the .corr() instance; finally, a comprehensive correlation matrix is built by averaging out all single firms matrices. The comprehensive matrix is thus the average of all correlations computed for each and every company considered.

To complete the picture, it is worth mentioning how missing values and 'inf' values, deriving from ratios with null denominator, were handled during the steps. 'Inf' values have been considered as missing value altogether because of evidences in the initial data. As example of this, 'Inventory/Sales' suffers by the presence of inf values in the ratios belonging to multiple companies and, per each firm, over multiple years. This results from Sales (the Italian 'Ricavi vendite e prestazioni') being null inside financial statements. Now, null Sales over multiple years can either indicate that the company is not set to engage with any kind of customer, typically useful for legal purposes only, or that the values comes from some form of simplified financial statement, usually granted to firms within certain produced values. In both cases it is

[19] More on the .corr() instance can be found inside the Pandas documentation available at https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html

[20] Discarding all defaulting entities' ratios give more reliability to the final correlations since the failing condition may affect the relationship among ratios in ways not dependent to the underline, true relationship.

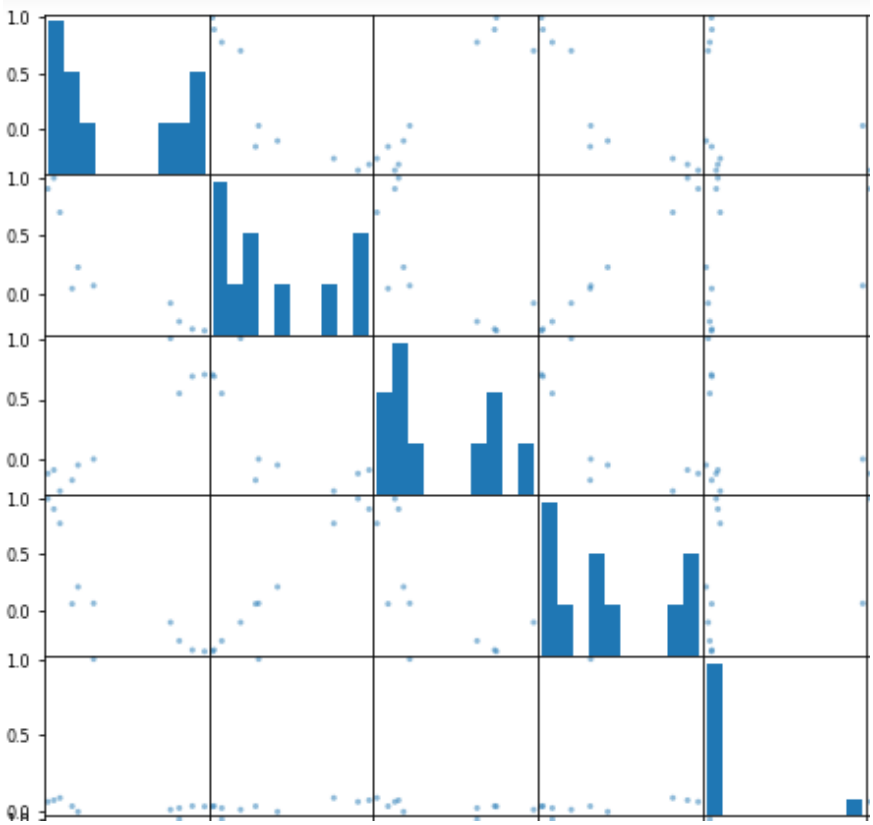
fair considering null values on sales as a missing value. Missing values, on the other hand, have been simply discarded in every step of the process to limit any effect deriving, for example, from filling their values with means or medians from the distribution of each ratio.

To show the results obtained from the correlation Table 9 and Figure 13 report the average correlations and their graphical representation, respectively.

Table 2.9 Average correlation among five selected financial ratios

| | Net Income/Total Assets | Total Debt/Total Assets | Net Income/Net Worth | Total Liabilities/Total Assets | Inventory/Sales |
|---|---|---|---|---|---|
| **Net Income/Total Assets** | 1.000000 | -0.319694 | 0.701873 | -0.372463 | 0.028436 |
| **Total Debt/Total Assets** | -0.319694 | 1.000000 | -0.081887 | 0.906469 | 0.069653 |
| **Net Income/Net Worth** | 0.701873 | -0.081887 | 1.000000 | -0.111642 | 0.007473 |
| **Total Liabilities/Total Assets** | -0.372463 | 0.906469 | -0.111642 | 1.000000 | 0.059160 |
| **Inventory/Sales** | 0.028436 | 0.069653 | 0.007473 | 0.059160 | 1.000000 |

Figure 2.13 Graphical representation of five selected financial ratios average correlation

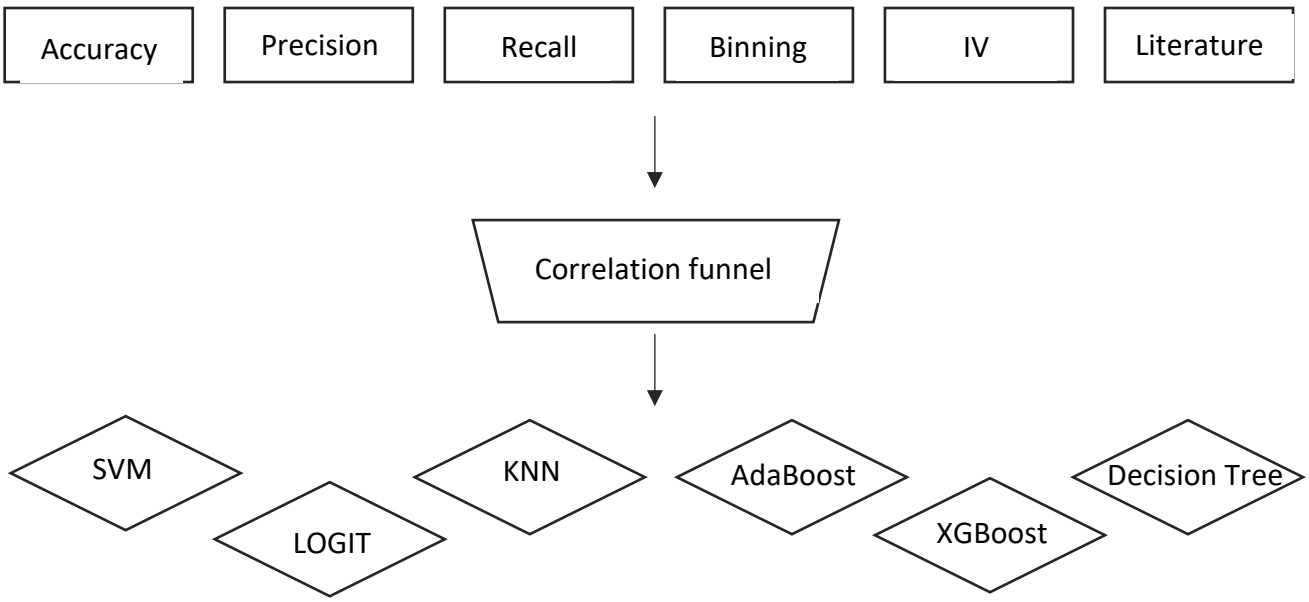


The chart shows the graphical representation of the ten selected financial ratios average correlation with each other (indices have been limited to five only for allowing a clearer representation). Whilst the diagonal, which should report $\rho(X,X) = \frac{cov(X,X)}{\sigma_X \sigma_X} = 1$, given that both numerator and denominator equal to the variance of X, shows the distribution of the index, all other cells display the direction of covariance between X and Y.

## 2.3 PREDICTION MODELS

This section explains the procedure aimed at applying multivariate prediction models. It comprises three main steps: the creation of 'priority lists'; their refinement through the 'correlation funnel'; and the actual application of six prediction models (Logit, Support Vector Machines, K Nearest Neighbour, AdaBoost, Decision Tree and XG Boost). Figure 14 schematizes the steps followed to carry out predictions.

Figure 2.14 Steps toward the application of multivariate prediction models




### 2.3.1 Priority Lists

The first step concerns the creation of priority lists. These are lists resulting from ranking all available indices on the basis of a common parameter. In total, six parameters have been selected: the accuracy measure elicited from the univariate logistic regression, to account for ratios' individual overall ability to predict bankruptcy/non-bankruptcy occurrences; the precision parameter, still from Logit, to include the individual qualification in avoiding mispredictions; recall, again from the univariate Logit, to prioritize those ratios that best recognize defaulted firms; the slope of the line of best fit, obtained through the first step of the binning procedure, to give more importance to those ratios that reveals to be linearly powerful, thus subordinating messy and 'complex'[21] ratios; the Information Values resulting from the

[21] Complex here is used to define those ratios which, in the binning procedure, appear to resemble good predictors only if applied by prediction models able to handle higher dimension than the simplest, linear approach.

Weights of Evidence; and a final list based on the most used and appreciated ratios in the relevant literature observed[22].

In other terms, each list contains the same 54 financial ratios permuted in differing orders. The importance of ranking all ratios in different lists prioritizing them on the basis of one of the six parameters will become clear only after the 'correlation funnel' section and is here introduced. Given the risk of multicollinearity, a selection between over correlating ratios is needed before the application of any multivariate prediction model. It is however not clear a priori what reasoning should be put into practice to implement such selection. To this end, priority lists offer a logical answer to the matter.

In relation to this, it can be argued that setting up only one priority list might restrict the final prediction due to the ranking of ratios. Further, expanding the argument, even all six lists may still be insufficient to represent a sufficient number of outcomes, thus limiting the analysis on the final prediction results. Maximizing the scope of analysis to all possible permutations, a total of $2{,}3\ x\ 10^{72}$ priority lists should be considered. Such vast number of possibilities, however, is too costly in terms of computational power required to run compute and analyse all possible predictions. Moreover, said analysis would out fall the purpose of this study, which is primarily related with the examination of feasible ways to apply prediction models on financial statements belonging to Veneto region enterprises, rather with the observation of theoretically sound models. For these reasons, only the six described priority lists will be considered. All other permutations are left for further study.

Before proceeding, it is worth mentioning one interesting detail: Net Income to Total Assets always ranks first in all lists. This confirms the literature preference for such an item, which is individually able to reach about 85% of accuracy in the univariate logit.

For sake of clarity, all priority lists are reported in appendix 6 along with the code written to build them.

### 2.3.2 Correlation funnel

After priority lists are set up, ratios can be filtered by the 'correlation funnel'. It represents the step concerning with the selection of ratios on the premise of priority lists, to prevent multicollinearity issues.

[22] The Literature priority list is based on the review published by Bellovary et al. (2007).

The code executing the filtering process is structured as follows: first, a correlation threshold is establish; secondly, from the threshold, all combinations of ratios whose absolute value of correlation exceeding it are pinpointed; third, all correlating pairs are looped over as inputs of the *prioritizing* function which has the role of determining the 'winner' and 'loser' ratios per each pair, on the basis of the rank established in the considered priority list; further, all indices not belonging to any correlating pair are added in order at the bottom of the list of winners. This procedure is looped over all six priority lists.

To meaningfully expand the scope of the analysis, the procedure described above is repeated considering multiple correlation values. Specifically, all values in the range from 0.3 to 0.9 with interval of 0.1 are applied as correlation thresholds[23]. This ensures a finer look into the role of the correlation threshold and its effects onto prediction results.

The core of the correlation funnel, once the average correlation matrix is available, is represented by the *prioritizing* function. The function takes as inputs the set correlation threshold, the average correlation matrix and the six priority lists arranged, while returns as output six lists, one per priority, containing only the ratios to be applied inside prediction models. It does so by executing two main body of code: it first determines 'winners' and 'losers' of each identified over correlating pair, through a series of if-else statements; secondly, it composes the final ready-to-use lists, keeping winners and non-paired and discarding losers, checking the appropriateness of it.

The first section, after setting up the loop over the six priority lists and pinpointing those pair of ratios with absolute correlation higher than the threshold, makes use of nine if-else statements. Initially, eight of them check whether the ratios contained in the pair under examination have already been assigned. The logic governing these eight statements is as follow: if any index is found to be losing in any of the pairs it might be belonging to, it is directed to the 'losers' list with no changing option; else, if the index is found to be winning in its correlating pair, then it is provisory directed to the 'winners' list; if, further, a previously winning index is found to be loosing in a second pair, then the first if statement described is executed. The underlying logic is to consider 'victory' provisory, it only holds in the time during which no better ranking index is paired, and 'defeat' permanent. Finally, the nineth if statement, simply regulates the occurrences in which both ratios have not already been directed, chasing the same logic. The second section instead, is undertakes the role of constituting the

[23] Here only positive values of correlation are considered since the *prioritizing* function identifies the over correlating pairs on the basis of the absolute value of their correlation.

final list that will then passed on to the prediction models. In doing so it also checks and rectify for any mistakes committed in the previous section.

All the describe Python code, along with some examples of ready-to-use lists are reported in the appendix.

### 2.3.3 Multivariate Prediction Models

This section relates with the illustration of pre-processing procedure applied to the data before running any prediction and of the models being used to predict defaulting and non-defaulting entities.

Data Pre-processing

The relevant data for predictions is computed, similarly to the univariate Logit, as the average of four years prior to default, per financial index, per each company selected by the PSM. The number of averaging years prior to default decreases to three for those indices without an available 2009 value, only for companies with 2013 as relevant defaulting year[24]. In other terms, from each subgroup created in through the PSM procedure (1 defaulting and 5 non-defaulting firms) is taken the average of four years prior to the defaulting year of the bankrupt firm inside.

The decision of taking the four years prior to default average bears at least two consequences: first, any model thus trained will be performing optimally on a new, hold-out, firm only with a similar data structure as input; what is more, the prediction outputted should be considered effective for a time span of one year. To overcome these limitations, other structures have been implemented and will be described directly looking at prediction results in the next chapter.

Before applying it, the resulting data is pre-processed to comply with the requirements needed to run all six statistical methods. Data pre-processing involves three main steps: handling outliers, filling missing values and scaling variables. The first two steps closely follow the symmetric procedure adopted for the univariate logistic regression. Outliers are initially spotted by standardizing all values (i.e. retrieving $z = \frac{x-m}{sd}$, where z is the standardised value, x the initial value, m the mean of the distribution and sd its standard deviation) and searching for $|z| > 3$. They are then replaced with NaN values (also known as missing values) and the rest of values is brought back to the initial amount. Then, all the missing values resulting are filled with the mean of the newly formed distribution. It is referred to as 'new' distribution since outliers are not anymore included. Finally, in addition to that, feature scaling is carried out. It

[24] As for the univariate logit, for relevant defaulting year is intended either the actual defaulting year, for bankrupt firms or the defaulting year of the matched bankrupt firm, for non-bankrupt companies.

is performed to avoid any issue related with the differences in the absolute levels of the ratios. Indeed, the output from it is a value between 0 and 1: this smooths out the previous possible greater disparities between indices merely deriving from the underlying components and/or structure of indices themselves. Feature scaling is executed through the default version of MinMaxScaler sub package which pertains to the sklearn.preprocessing opensource module[25].

Prediction models

A total of six prediction models were adopted: Logistic regression, Support Vector Machines, K – Nearest Neighbour, AdaBoost, Decision Tree and XGBoost.

Logistic Regression

Following Peng, J. (2002), the central mathematical concept that underlies logistic regression (LR) is the logit, the natural logarithm of an odds ratio, where the odds are usually referred to dichotomous cases. Indeed, LR is generally well suited for describing and testing hypotheses about relationships between a categorical outcome variable and one or more categorical or continuous predictor variables. LR is able to handle dichotomous dependent variables better than, for instance, Ordinary Least Squares regressions transforming y, the dependent variable through the logit function. From this, it occurs that the LR can be described, in its simplest form, by

$\ln(\frac{\pi}{1-\pi}) = \alpha + \beta X$, where $\pi$ is the probability of y happening, $\alpha$ and $\beta$ are coefficients of the regression and X is the independent variable (or group of variables, in the multivariate case).

For the purposes of this study, it was adopted the LogisticRegression sub module from the opensource sklearn.linear_model package[26]. As for the univariate section, little has been fine tuned with respect to the default setting of the module. The solver selected is 'lbfgs', previously explained and the proportion between train and test (hold-out) samples is set to 25%. Moreover, to correct for the defaulting threshold of 50%, through which the testing values are declared bankrupt or non-bankrupt, a new, better threshold is evaluated. Figure 15 reports the code for the new threshold. For it, True Positive Rates and False Positive Rates are computed from the confusion matrix resulting from the 50% limit; their difference is initiated ('J' in Figure 15) and maximised through the argmax function; the new threshold is then identified among all

[25] More information on MinMaxScaler default version can be consulted at https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
[26] More can be found at https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

potential ones; finally, it is applied the old set of dependent variables (Y_pred['new_thrsld'] in Figure 15).

Figure 2.15. Code to retrieve the optimize threshold

```
# get the best threshold
J = tpr - fpr
ix = np.argmax(J)
best_thresh = thresholds[ix]
y_pred = pd.DataFrame(y_pred)
y_pred['new_thrsld'] = np.where(y_pred >= best_thresh, 1, 0)
```

Support Vector Machines

From Cervantes et al. (2019), Support Vector Machines (SVM) was introduced by Vapnik as a kernel-based machine learning model for both classification and regression task. Due to its good theoretical foundations and good generalization capacity, however, in recent years, SVMs have become one of the most used classification methods. In particular, its generalization capacity stands out against other classification models. By generalization is intended the ability of the classifier, the model, to recognise the relevant patterns useful for organizing data into the correct groups. When the model, as an example, is too fit for the training data, the model begins to memorize training data rather than learning to generalize, degrading the generalization ability of the classifier.

SVM carry out classifications through the determination of the 'optimal separation hyperplane'. This is the only separation hyperplane with maximum margin, the distance between the hyperplane and the support vectors. Support vectors, in turn, are those hyperplanes, parallel to the optimal one, identified by the closest datapoints standing at the margin distance.

A key element of the SVM theory is the kernel, critical if the training data are not linearly separable. Indeed, the basic idea in designing non-linear SVMs is to transform the input vectors into vectors of a higher dimensional feature space. The function adopted for the transformation is, precisely, the kernel. There is no unanimous conclusion about which kernel is better or worse for specific applications. For purposes of the study a polynomial kernel has been applied. In general, the polynomial kernel follows: $K(x_i, x_j) = (x_i . x_j + 1)^p$ , where p is the polynomial degree, following from Mercer's condition for function to be implemented as kernels.

To implement SVM in the analysis, the SVC module of the sklearn.svm package has been selected[27]. For it a C parameter of 150 is set and, as mentioned a polynomial kernel selected. C

[27] More on it is available at https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

refers to the Regularization parameter that defines the amount of misclassification allowed by the model. Too high C determines overfitting problems, too low C diminishes the quality of the final classification. C equalling 150 has been appointed as the best performing level.

K – Nearest Neighbour

The intuition underlying Nearest Neighbour Classification is quite straightforward: examples are classified based on the class of their nearest neighbours. In particular, after selecting k, the number of nearest neighbours that will be taken into consideration, the classification is made on the basis of the number of neighbours obtained per each class. As then pointed out in Cunningham, P. and Delany, S. J. (2007), starting from this basic frame the model can be fine tuned to the data being used: the definition of distance may be adjusted, differing weights can be given to classes when determining the classification, etc.

Similarly to above, sklearn.neighbors.KNeighborsClassifier module has been implemented in the study[28]. K is set to 12 given that it showed higher than average performances. All other parameters have been kept in default settings.

AdaBoost

AdaBoost comes from 'Adaptive Boosting' and refers to one of the first practical declinations of the boosting methodology. In general, boosting is an ensamble approach to machine learning based on the idea of creating a highly accurate prediction rule by combining many relatively weak and inaccurate rules. Following Schapire, R. E. in his *Explaining AdaBoost* review, the pseudo code for the AdaBoost algorithm can be summarized as:

Given: $(x1, y1), \ldots, (xm, ym)$ where $xi \in X, yi \in \{-1, +1\}$

Initialize: $D1(i) = 1/m$ for $i = 1, \ldots, m$.

For $t = 1, \ldots, T$ : • Train weak learner using distribution Dt;

• Get weak hypothesis $ht : X \rightarrow \{-1, +1\}$.

• Aim: select $ht$ with low weighted error: $\varepsilon t = Pr_{i\sim Dt}\,[ht(xi)\ 6 \neq yi\ ]$.

• Choose $\alpha t = \frac{1}{2}\ln\left(\frac{1-\varepsilon t}{\varepsilon t}\right)$

[28] More at https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

• Update, for $i = 1, \ldots, m$: $Dt + 1(i) = \frac{Dt(i)\ exp(-\alpha t\ yi\ ht(xi))}{Zt}$, where $Zt$ is a normalization factor (chosen so that $Dt + 1$ will be a distribution).

Output the final hypothesis: $H(x) = sign\ (\sum at\ ht(x))$

Here we are given m labelled training examples $(x1, y1), \ldots, (xm, ym)$ where the $xi$'s belong to some domain X , and the labels $yi \in \{-1, +1\}$. On each round $t = 1, \ldots, T$, a distribution Dt is computed over the m training examples, and a given weak learner is applied to find a weak hypothesis $ht : X \rightarrow \{-1, +1\}$, where the aim of the weak learner is to find a weak hypothesis with low weighted error $\varepsilon t$ relative to $Dt$. The final, combined hypothesis H computes the sign of a weighted combination of weak hypotheses.
Cleared the general functioning of the algorithm, it is critical to deepen the basic feature of the weak learner adopted. Usually, AdaBoost makes use of Decision Stumps (DS). These are the simplest form of decision trees and are only able to carry out classifications entailing one, single independent variable. They are call weak learners because the accuracy deriving from DS would already be optimal if they guessed whichever answer, 1 or 0, is most common in the data. If, for instance, 60% of the examples are 1s, then the model will obtain 60% accuracy just by predicting 1 every time. The main advantage of weak learners like DS is their adaptability to different scenario: their lower than average results are pretty stable over differing datasets and independent variables, even though individually poor. The power of AdaBoost precisely comes from leveraging DS adaptability and does so combining multiple DS results to reach high degrees of accuracy for complex classification tasks.

For purposes of the research, sklearn.ensemble.AdaBoostClassifier module has been applied setting a number of weak learning estimators to 150[29]. This has been selected on the basis of performances.

Decision Tree

Following Maimon and Rokach (2005) a decision tree is a classifier expressed as a recursive partition of the instance space. It consists of nodes that form a rooted tree, meaning that it is a directed tree with a node called 'root' that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces according to a certain

[29] More on the sklearn algorithm can be found at https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

discrete function of the input attributes values. In the simplest and most frequent case, as already seen for Decision Stamps above, each test considers a single attribute, such that the instance space is partitioned according to the attribute's value. In the case of numeric attributes, the condition refers to a range. The process followed by decision trees dealing with multiple predictors can be generally described as follows: First, the root node predictor is chosen by selecting the one displaying lower overall impurity; secondly, all other nodes are linked to the remaining predictors looking at the lowest impurity classification; finally, classification is carried out through the constituted tree. Impurity generally refers to the degree of misclassification reached by a single predictor. Usually, the impurity measure applied is the Gini index, which can be defined as $G = 1 - (\Pr\, outcome\ 1)^2 - (\Pr\, outcome\ 0)^2$ for binary classifications[30]. For instance, to determine the predictor at the root node are executed these steps: a confusion matrix is initially generated (on the testing sample) taking as single independent variable each one of the 54 financial ratios[31]; then the Gini impurity index is computed to all 54 generated confusion matrices; finally the lowest Gini index ratio is selected as root node. These steps are then repeated for all nodes of the tree to establish ratios order among them. Eventually, the constituted tree model is run.

For the implementation of Decision Tree model, the sklearn.tree.DecisionTreeClassifier module is chosen in its default settings[32].

XGBoost

XGboost was developed by Chen and Guestrin in 2016 and stands for "Extreme Gradient Boosting". It belongs to the supervised learning gradient boosted trees family. As with the other ensemble methods, the idea of XGBoost is to combine weak learners such as decision trees into a strong learner. A series of decision trees, of usually constant shape and depth, is created which together form a single predictive model. New learners are trained on the errors, the residuals, of the previous learners so to increase the final predictive power. Put differently, the idea of tree boosting is to add a new tree to the ensemble fit to the residual of the predictions from earlier trees. The residual is typically defined in terms of derivative of the loss function. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient and flexible especially for large datasets. XGBoost supports various objective functions

[30] The probabilities are easily retrieved from the confusion matrix generated.
[31] The example considers all 54 financial ratios. In the study however the actual number of ratios included in the model varies according to the correlation threshold and the relative winner and loser lists.
[32] Any further information can be consulted at https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

including regression, classification and ranking. It recently gained much popularity and attention both among academics and practitioners for its features.

To run it in this project the XGBoost python package was selected, which differently from above, does not belong to sklearn framework[33]. All settings have been kept at their default state: number of estimators at 100 and a binary and logistic objective function.

[33] More can be found at https://xgboost.readthedocs.io/en/latest/python/python_intro.html

# 3. RESULTS

This chapter aims at illustrating the results collected from the application of the six prediction models. Specifically, two sets of results will be outlined: first, the measures of performance retrieved from the testing, hold-out sample (25% randomly selected datapoints from the whole initial sample) and, secondly, the outcomes from applying the same prediction models (i.e. trained on the same training sample as in the previous point implies) on data belonging to companies external to the Veneto region. While the first sets of results logically follow the path already delineated, the second set will be especially useful for examining how the models behave with comparable but not identical samples. This should, for instance, shed light onto the risk of overfitting the models might be exposed to.

As explained in the previous chapter, the starting point to carry out any prediction in this analysis is the average ratios matrix. This, as illustrated in Figure 3.1, is composed by rows referring to each of the companies selected from the Propensity Score Matching procedure and columns representing the 54 financial ratios previously computed. Each cell carries the value of the column ratio and row firm that has been chosen from all firm ratios available in the initial sample. In this case, such value is determined as the average resulting from four years of the specified firm-ratio selected on the basis of the chosen distance from the relevant year[34]. As for example, assuming as chosen distance one year, then for each firm will be associated values of ratios equal to the average of four years starting from one year prior to the relevant year. If 2014 happens to be the relevant year for the examined firm_1, then its ratio_54 value will be equal to the average of the firm_1's ratio_54 in 2013, 2012, 2010 and 2009.

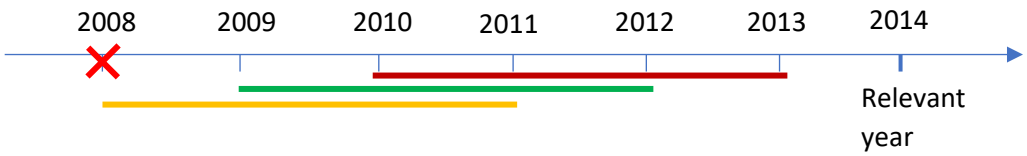
Figure 3.1 Matrix of ratio averages for each firm with stylized computations of ratios averages (the chosen distance from the relevant year is one year).

| | **Ratio_1** | **…** | **Ratio_54** | **Relevant year** |
|---|---|---|---|---|
| **Firm_1** | $\frac{1}{4}$ * F_1-R_1 (2009+2010+2011+2012) | | $\frac{1}{4}$ * F_1-R_54 (2009+2010+2011+2012) | **2014** |
| **…** | | | | |
| **Firm_2430** | $\frac{1}{4}$ * F_2430-R_1 (2014+2015+2016+2017) | | $\frac{1}{4}$ * F_2430-R_54 (2014+2015+2016+2017) | **2018** |

[34] As in the previous chapter, by 'relevant year' is intended the year of default for defaulting companies and the year of default of their Propensity Scores Matched defaulting firms for non-defaulting companies (remark that each defaulting firm has been matched with five non-defaulting)

Whenever a ratio is not available in a specific year, perhaps because the four years average dates back to 2008 values which have not been found available in AIDA database, the average is simply computed on the remaining, available years. This is shown in Figure 3.2 when, considering 2014 as relevant year and three years as chosen distance, the average only includes 2009, 2010 and 2011 ratios (yellow segment).

Figure 3.2 Three chosen distances, the four years average and the unavailable 2008 ratio




As Figure 3.2 illustrates, to look for the robustness of prediction models in both sets of results -testing sample internal to the Veneto region and external to it- over time, three distances from the relevant year are chosen and observed: one, two and three years. These, in the example depicted above, corresponds to the brown line (averaging from 2010 to 2013), the green one (from 2009 to 2012) and to the yellow segment (from 2009 to 2011, excluding 2008).

## 3.1 TESTING SAMPLE RESULTS OF VENETO FIRMS

In this section it will be deepened the results reached by the six prediction models on average ratios computed with all three chosen distances from relevant years. In particular, two main assessments will be carried out for each of the three cases: first, an analysis of the accuracy levels reached by every prediction model on all six priority lists created over several correlation thresholds[35] will be accomplished; second, three target correlation thresholds, namely 0,3, 0,6 and 0,9, are compared examining accuracy, recall, precision and ROC AUC for each prediction model and priority list. As to conclude, along with their individual analysis, the three scenarios (i.e. brown, green and yellow segments in Figure 3.2) will be submitted to a concise comparison.

[35] As pointed out in the previous chapter, the creation of a ready-to-use priority list entails first the ranking of all 54 financial ratios on the basis of a common parameter (e.g. the ratio individual accuracy in the univariate Logistic regression); secondly, the setting of a correlation threshold (e.g. 0,7); third, the isolation of those pairs of indices whose absolute level of correlation exceeds the thresholds (where pairs are identified on the basis of the average correlation matrix, still described in chapter 2); finally, the selection of the best ranking indices among over correlating pairs and rejection of the low ranking ones. The ready-to-use priority list is thus the sum of best ranking indices and those ratios not over correlating with any other. The ready-to-use priority list determines which ratios will be applied to the prediction model.

### 3.1.1 One-year distance from the relevant year

The first of the three cases entails the construction of the matrix of ratio averages considering one year as the distance between the four years average and the relevant year. Logically, since this scenario is the closest to the relevant year, one should expect it to outperform the two others in terms of overall accuracy in prediction. On the other side, however, its practical employment seems to be limited with respect to the other scenarios. Indeed, being able to predict failure/non-failure one year before it should actually happen, is certainly less useful than forecasting it with two or even three years in advance, *ceteris paribus*.

That said, the first analysis relates to the examination of prediction models' accuracy charts built to compare all six priority lists over multiple correlation level. Specifically, the correlation thresholds taken into account belongs to the 0,3 to 0,9 range with inner interval of 0,01 (i.e. 0,30, 0,31, …, 0,89, 0,90). Moreover, the six priority lists are, again, structured on the basis of: ratio individual accuracy, precision and recall deriving from the univariate Logistic regression; complexity[36] and Information Value retrieved from the Binning and Weight of Evidence procedures; and finally, from the literature most frequent financial ratios[37]. All Python code written for these results is available in appendix 8.

At each correlation level, a final ready-to-use list of ratios is created for every priority list through the correlation funnel procedure detailed in chapter 2. Then, all six ready-to-use lists are applied in each prediction model, namely Multivariate Logistic regression, Support Vector Machines, K-Nearest Neighbour, AdaBoost, Decision Tree and XGBoost. To run the prediction models, the sample data is split into a 75% randomly selected training set and 25% test set. The random composition of both sets should guarantee unbiasedness of outcomes.

Following, six charts showing the accuracy levels reached by each model one year before failure, are reported. On the y-axis can be found the accuracy in prediction while the x-axis constitutes the correlation threshold applied. In the lower left, the legend indicates which curve belong to which priority list. In this case, by priority list is, of course, intended as the ready-to-use list of ratios determined at each correlation level.

[36] Complexity is proxied by the slope of the line of best fit in the binning curve described in chapter 2.
[37] The Literature priority list is based on the review published by Bellovary et al. (2008)

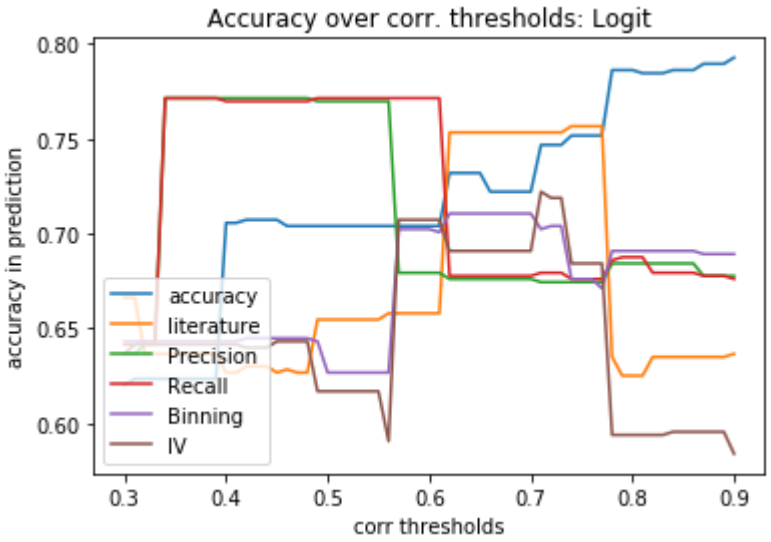



Figure 3.3 Accuracy levels achieved by the Logistic regression model on the six ready-to-use lists of financial ratios

Figure 3.3 describes the performance in terms of overall accuracy gained by the Logit model. The highest level is reached around a correlation threshold of 0.8 by the priority list based on the individual accuracy of each ratio. Moreover, precision and recall lists appear to behave especially well for relatively lower levels of correlation thresholds. Overall, there does not seem to be a best performer among priority lists while IV almost always displays as least performer. On average Logistic regression stands between 60% and 80% of accuracy at one year before default, in line with the relevant literature.

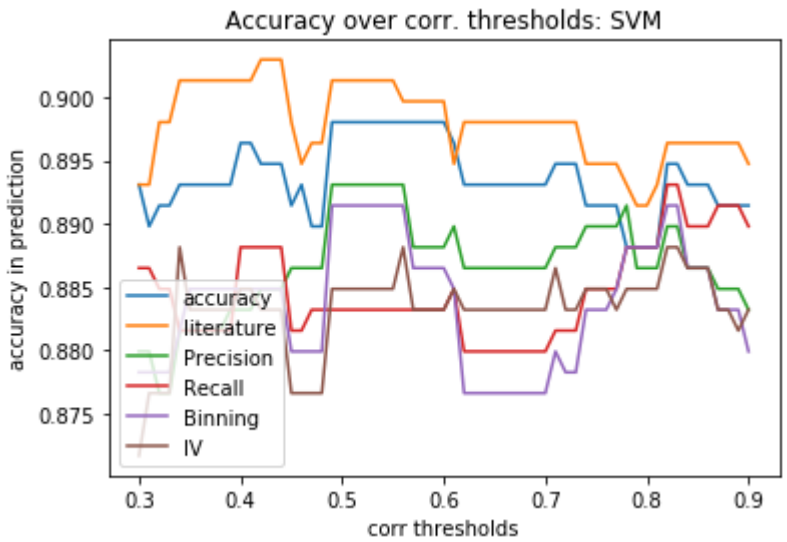



Figure 3.4 Accuracy levels achieved by the SVM model on the six ready-to-use lists of financial ratios

Further, Figure 3.4 describes the accuracy levels achieved by the SVM model. Three facts are immediately clear from it: first, the average accuracy level is significantly higher than the Logistic regression model; second, the interval in accuracy (y-axis) is substantially narrower than Logit's; third, unlike previously, there seems to be a relative constant difference among priority lists in general, with the literature list steadily outperforming all others. One conclusion can already be drawn from chart 3.4: SVM performs significantly better than Logistic regression at one year before default on Veneto region firms. Also, the highest results, over

90% accuracy, are achieved by applying the literature most frequent financial ratios and filtering them up at relatively low correlation thresholds (i.e. admitting low degrees of correlation among indices). Again, lists retrieved from binning complexity proxy and IV exhibit lower than average accuracy.

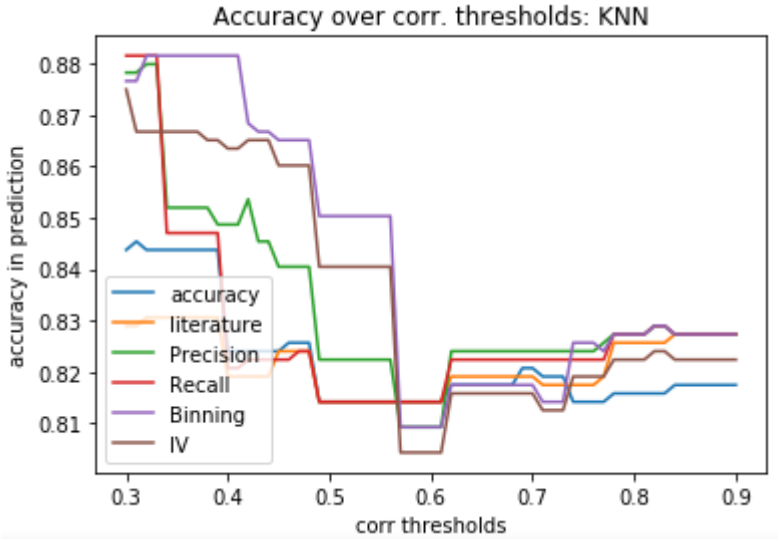



Figure 3.5 Accuracy levels achieved by the KNN model on the six ready-to-use lists of financial ratios

Figure 3.5 illustrates the accuracy levels from the KNN classification model over multiple correlation thresholds. Interestingly, the first immediate feature that can be noticed is the difference between accuracy gained before and after 0,6 as threshold. In fact, there appears to be a sharp decrease in performance at said correlation boundary and, furthermore, it is clear the strong negative relationship between accuracy and threshold applied: the higher the correlation allowed among ratios, the lower the accuracy. The reason for this peculiar behaviour should be searched in the KNN execution algorithm. In general, KNN builds a classification on the basis of the distance between each datapoint and $k^{th}$ neighbours: the closest neighbours belong to the same group. When a relatively high level of correlation is allowed, datapoints distance between each other is affected by the trends introduced by correlating ratios, resulting in higher numbers of overlapping neighbours. This, in turn, increases the frequency of misclassifications. To conclude the argument, it needs to be added that, though this behaviour is clearly occurring, its effect has only slight implication since the loss in prediction entails only few percentage points overall (5% to 6% points at most).

Other than that, Figure 3.5 indicates that KNN performs better than Logistic regression but worse than SVM on average. Finally, Binning and IV curves display, contrarily to the previous models, highest average performances against the other lists, especially within low correlation thresholds.

The previous conclusion can now be updated: SVM model outperforms both Logit and KNN in terms of overall accuracy, with KNN achieving significantly higher performances than Logit in the testing sample.

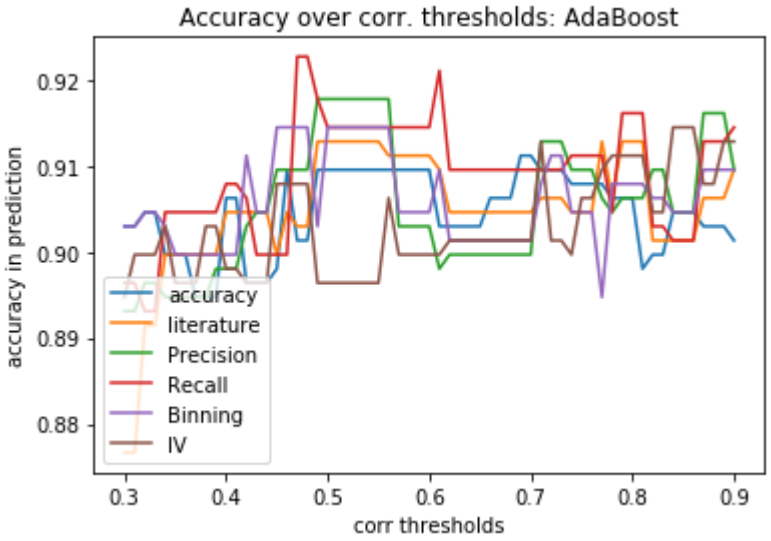



Figure 3.6 Accuracy levels achieved by the AdaBoost model on the six ready-to-use lists of financial ratios

Figure 3.6 depicts AdaBoost accuracy performance over all correlation thresholds identified. The chart brings about three main considerations: first, the accuracy levels obtained are contained in a relatively narrow interval (from 90% to 92% accuracy if the marginal 0,3 to 0,4 correlation thresholds are not taken into account); also, AdaBoost appears to be the best average performer with slightly higher values than SVM; as last, priority lists are constantly overlapping without a clear best performer.

An interesting observation can be suggested from the first consideration: so far there seems to be a quite clear relationship between best performing models and length of the accuracy interval. In other words, the most accurate models also exhibit narrow intervals of prediction accuracy over correlation thresholds, which may indicate that the best models are also less sensitive to changes in correlation boundaries.

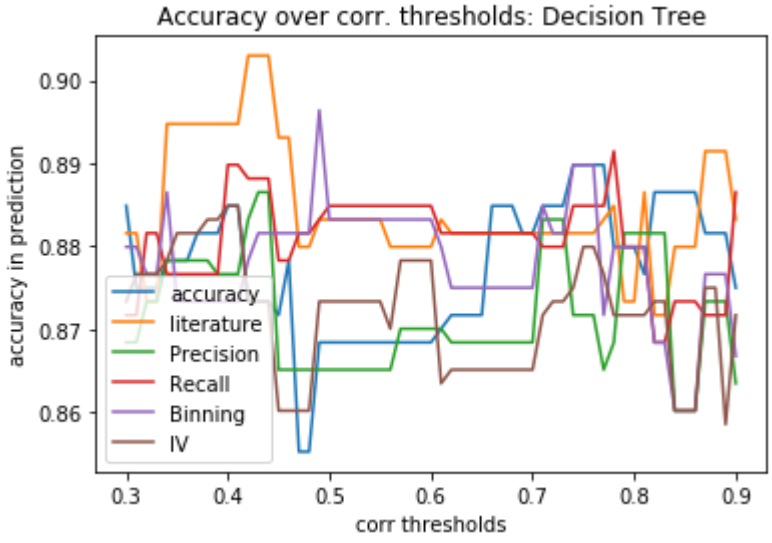



Figure 3.7 Accuracy levels achieved by the Decision Tree model on the six ready-to-use lists of financial ratios

Figure 3.7 represent the one year to bankruptcy accuracy levels in prediction of Decision Tree model. The average accuracy, among all lists, seems to perform just better than KNN and worse than SVM. Moreover, literature ready-to-use list of indices appears to be the best overall performer peaking above 90% of accuracy between 0,4 and 0,5 as correlation thresholds. Again, the suggestion previously made about the relationship between the average level of accuracy and the length of the accuracy interval appears to be here confirmed. Indeed, Decision Tree shows narrower accuracy range than both Logit and KNN, which are, on average, less optimal, but wider than SVM and AdaBoost. Finally, given the steady mean trend of the six curves, it does seem that Decision Tree is only weakly affected, if none at all affected, by the correlation level allowed in the sample.

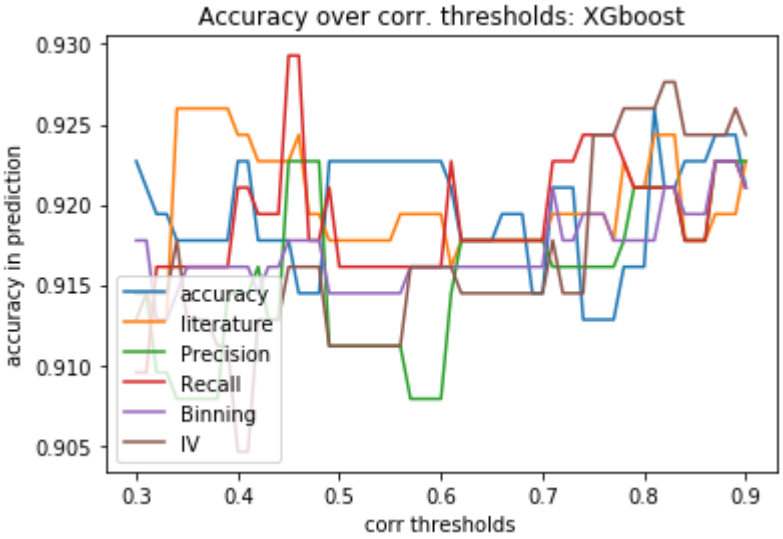



Figure 3.8 Accuracy levels achieved by the XGBoost model on the six ready-to-use lists of financial ratios

The final chart, Figure 3.8, presents the accuracy trend over correlation thresholds achieved by XGBoost. This results to be the overall best performers with average accuracy slightly but significantly higher than AdaBoost. Again, the model does not seem to particularly 'prefer' any of the six priority lists even though Recall does exhibit a steadily higher than average accuracy. More importantly, XGBoost further corroborates the suggestion over the relationship between narrow interval and high average accuracy (indeed this ensemble method shows the narrowest interval of all). Finally, XGBoost does not appear to be affected by the level of correlation allowed: the only observation at this regard indicates, very weakly, that the higher the correlation threshold, the narrower the accuracy interval and thus more precise the final prediction.

To summarize, Table 3.1 summarizes the rank on the basis of the average accuracy, of which is reported an indicative range.

Table 3.1 Ranking of the best predicting models (left to right) with accuracy intervals (in %)

| MODEL | XGBoost | AdaBoost | SVM | Decision T | KNN | Logistic |
|---|---|---|---|---|---|---|
| RANGE | 90,5 - 93 | 88 - 92 | 87 - 90 | 85,5 - 90 | 81 - 88 | 60 - 80 |

Table 3.1 confirms once more the relationship between higher accuracy and narrower interval in results. Indeed, with the sole exception of AdaBoost, whose interval features wider range of accuracy levels compared with SVM which follows in the ranking[38], all other ranges are perfectly in line with the average accuracy ranking. This bring to the conclusion that better performing models are also those less affected by different levels of correlation allowed inside the sample.

A second analysis that comes at help for the interpretation of results is the examination of the Area Under the Receiving Operating Characteristic Curve (ROC AUC) per each prediction model. ROC AUC measures the entire two-dimensional area underneath the entire ROC curve and provides an aggregate measure of performance across all possible prediction frameworks. It is plotted through a True Positive Rate (y-axis) and False Positive Rate (x-axis) chart, where the $TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$ , is the ratio between the correctly predicted defaulting firms and all defaulting, and $FPR = \frac{FP}{N} = \frac{FP}{FP+TN}$ , is the ratio between the incorrectly predicted non-failing companies and all non-defaulting. The AUC is built so to fall in the range between 0, poorest results, and 1, maximum achievable.

To report ROC AUCs and compare their values under different parameters, three correlation levels have been identified: 0,3, 0,6 and 0,9. These three have been chosen for they allow to clarify the behaviour of prediction models in a correlation range that do not discards too many financial indices from the initial group of 54[39], which is crucial to study the combined ratios behaviour in multivariate methodologies, while also showing outcomes with relatively high risk of multicollinearity issues. Further, to deepen the research on the predictions elicited, each ROC AUC chart will be joint by a table showing three features: the accuracy in prediction, the

[38] It should also be noted that AdaBoost regains its second position also accounting for the length of its interval when the first correlation levels from 0,3 to 0,4, marginal to the overall picture, are disregarded from the examination.

[39] Indeed, through the correlation funnel described in chapter 2, ratios figuring into over correlating pairs are always discarded or kept on the basis of the ranking defined by the order inside the priority list at use.

precision in prediction and the recall measure for every model. Finally, to complete the picture, the list of financial ratios effectively implemented in the models is detailed.

For sake of clarity, only metrics concerning the Accuracy priority list, at every correlation level, have been reported, whilst all other priority lists charts can be found in appendix 9.

Figure 3.9 graphs models' ROC curves at 0,3 correlation boundary against the diagonal, dotted red, indicating a hypothetical model with a performance, measured by the AUC, of 1/2.

```
Priority list: accuracy; Correlation threshold: 0.3
```

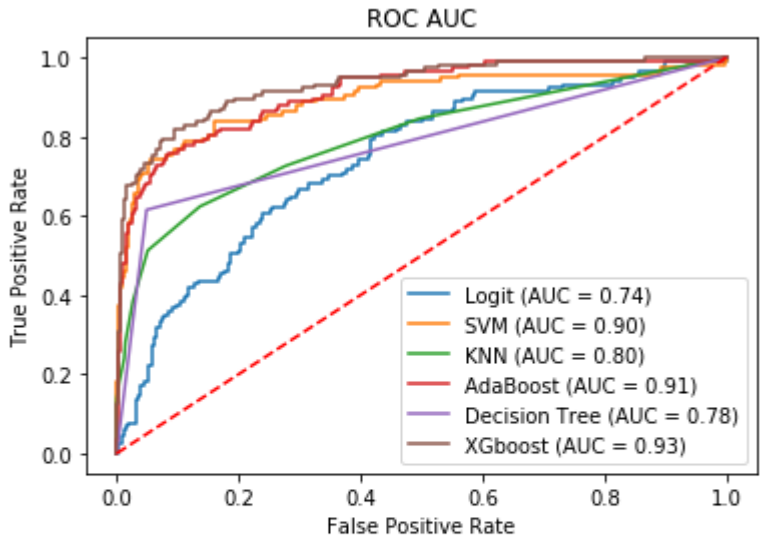



Figure 3.9 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,3 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.620066 | 0.618047 | 0.689897 |
| SVM | 0.893092 | 0.893464 | 0.745008 |
| KNN | 0.84375 | 0.834239 | 0.613548 |
| AdaBoost | 0.902961 | 0.864372 | 0.806456 |
| Decision Tree | 0.886513 | 0.831055 | 0.783252 |
| XGboost | 0.922697 | 0.912325 | 0.828442 |

Confirming the previous analysis, Figure 3.9 suggests XGBoost as best performing predictor, with an assessment as high as 0.93, followed by AdaBoost, 0.91, SVM, 0.90, KNN, 0.80, Decision Tree, 0.78, and eventually Logistic regression, 0.74. It is moreover straightforward that the models can basically be split into two main groups of performance: XGBoost, AdaBoost and SVM belongs to the first, best performing, while the other three to the least performing. Inside the two groups, the differences are significant but fairly small.

Attached to the chart, a table showing accuracy, precision and recall is reported. The table draws a different picture with respect to what can be elicited from the chart. Here, the difference among models toward Logistic regression is exacerbated in essentially all three dimensions, with the only exception of the recall measure for KNN. Interestingly, recall is higher than

accuracy and precision only for Logistic regression, with KNN and SVM being the most exposed to weaknesses under it. This marks that the percentage of correctly predicted failing firms on the total sample of defaulting entities is relatively lower for the higher accuracy models than the overall weaker Logistic regression. This, in turn, indicates a certain exposition of models to type I error, the misprediction of defaulting companies, the error bearing higher risks for lenders. Though Logit has a relatively higher recall performance, its absolute percentage remains lower than all other prediction models with the sole exception, again, of KNN.

At 0,3 threshold and under the ranking established by the accuracy priority list, models apply only 9 of the 54 ratios: Net Income/Total Assets, Net sales/Cash from sales, Cash Flow from Operations, Inventory/Sales, Operating expenses/Operating income, Sales/NAR, Cash/Total Assets, Other Revenues/Total Produced Value and log(Total Assets).

```
Priority list: accuracy; Correlation threshold: 0.6
```

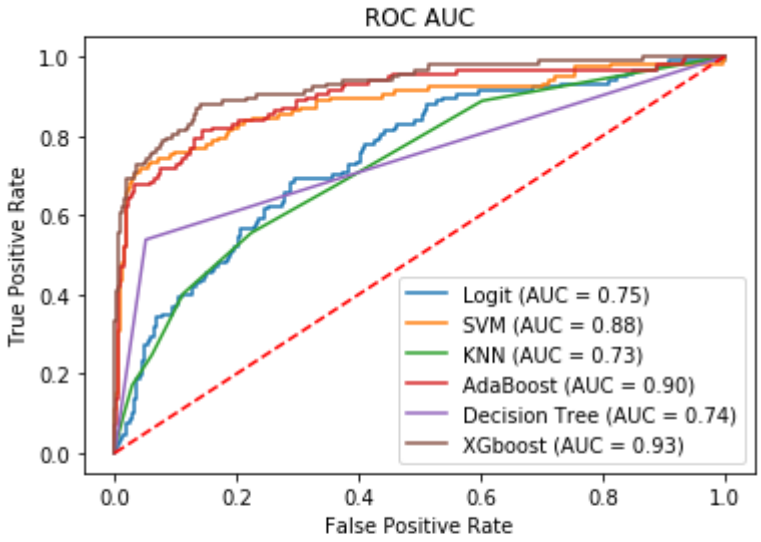



Figure 3.10 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.703947 | 0.633003 | 0.699514 |
| SVM | 0.898026 | 0.888823 | 0.764339 |
| KNN | 0.814145 | 0.715725 | 0.53988 |
| AdaBoost | 0.909539 | 0.879719 | 0.813785 |
| Decision Tree | 0.870066 | 0.806031 | 0.743773 |
| XGboost | 0.922697 | 0.912325 | 0.828442 |

An intermediate set of results is then represented in Figure 3.10 which report the same metrics setting the correlation threshold at 0,6. The graph shows how the overall picture remains similar to the previous set of results. XGBoost still appears to be the best performing model with AdaBoost and SVM respectively, although totalling lower AUCs, chasing right after. Interestingly though, KNN drastically lowers its AUC, -0,07, ranking last after Decision Tree, also lowering its overall performance and Logistic regression which is slightly improved.

Modifications also occur on the three dimensions in the attached table. Again, KNN reveals lower accuracy while Logit materially increases its own. KNN behaviour should not be surprising since, as illustrated by Figure 3.5, all priority lists accuracy levels decrease substantially at the 0.6 correlation threshold. Further, recall records confirm the intuition on the general weakness in prediction of defaulting firms with even lower recall outcomes.

At 0,6 correlation boundary and under the ranking established by the accuracy priority list, models apply 20 of the 54 initial ratios. Other than those already applied for 0,3 correlation, other 11 are added: Total Debt/Total Assets, Working capital/Sales, Retained earnings/Total assets, EBIT/Interest, CFO/EBIT, Tax Expenses/EBIT, Current Assets/Total Assets, Current Assets/Sales, Long-term debt/Total Assets, Turnover Payables, Turnover Inventory.

As last assessment, Figure 3.11 illustrates the ROC AUC and the accuracy, precision and recall table under 0,9 correlation threshold and accuracy priority list.

```
Priority list: accuracy; Correlation threshold: 0.9
```

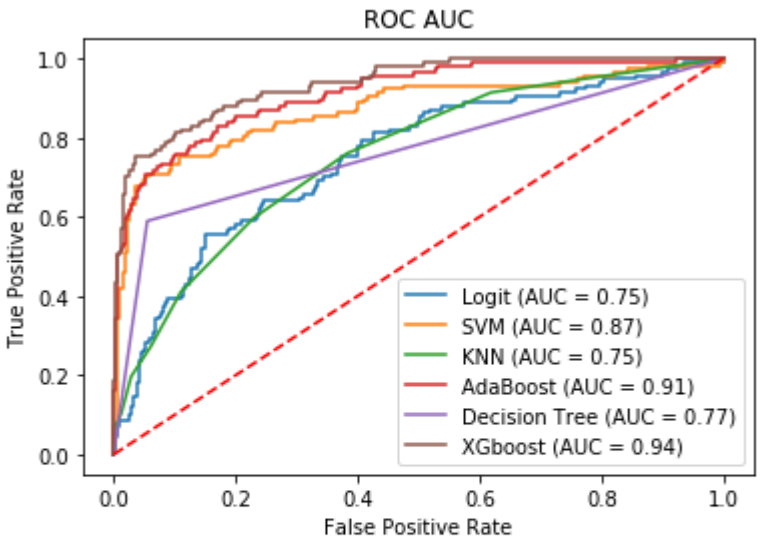



Figure 3.11 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,9 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.792763 | 0.678376 | 0.702421 |
| SVM | 0.891447 | 0.881907 | 0.747245 |
| KNN | 0.817434 | 0.74435 | 0.545172 |
| AdaBoost | 0.901316 | 0.860039 | 0.805438 |
| Decision Tree | 0.876645 | 0.8125 | 0.767377 |
| XGboost | 0.921053 | 0.910848 | 0.824168 |

Here, a much similar evaluation to the 0,6 correlation can be carried out. The first observation concerns with the fact that both ensemble methods, XGBoost and AdaBoost benefit from the higher level of interconnection allowed among ratios: both models increase their AUC by 0.01. On the contrary, SVM seems to be slightly disadvantaged with a loss of 0.01. Finally, while

Decision Tree improves its score, KNN and Logit end the ranking at par, 0.75 AUC each. What is more, the table attached does not appear to exhibit drastic changes from the previous version.

At 0,9 threshold 35 of the 54 ratios have been applied. From the 0,6 correlation list, other 15 are added: Operating Income/Total Assets, Fixed Charges EBIT Coverage, Net Income/Sales, Net Income/Net Worth, Total liabilities/net worth, Working Capital/Total Assets, Current Liabilities/Total Assets, Acid Ratio, quick assets/Sales, no-credit interval (Current Assets/Daily OPEX), Asset Turnover, CFO/Financial Debt, quick assets/Total assets, EBITDA/EBIT and Working Capital/Net worth.

### 3.1.2 Two- and three-year distances from the relevant year

In this section the same metrics of outcomes considered in the precedent paragraph are examined. This time, however, results relate to four years average ratio values taken both two and three years before the relevant year of default. In other terms, this section studies prediction model performances when asked to predict with a forecasting timespan of two and three years. Here, two- and three-years distances from the relevant year results are condensed for sake of brevity.

Similarly to above, first, accuracy levels over multiple correlation thresholds will be displayed while ROC AUC and attached table of features will be left as conclusion.

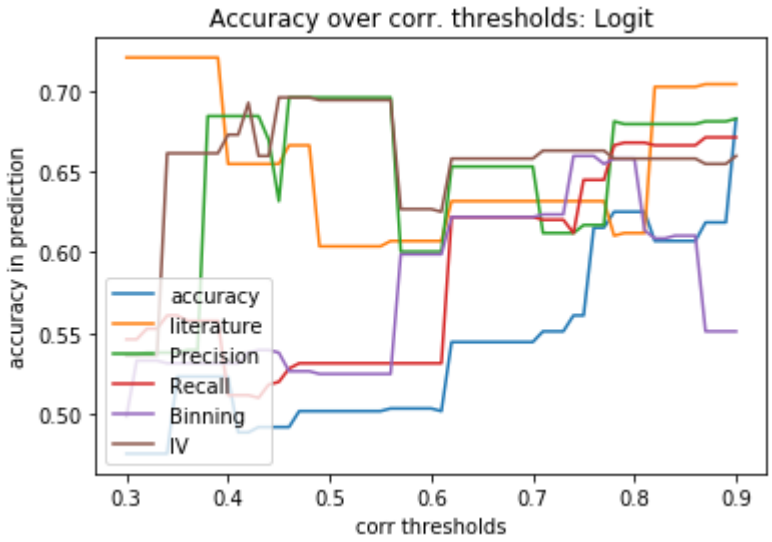



Figure 3.12 Accuracy levels achieved by the Logistic regression model on the six ready-to-use lists of financial ratios

Starting from two-years distance, Figure 3.12 plots the accuracy levels per each priority list reached by Logistic regression on the test set over different correlation thresholds. Three main arguments are immediately clear from it: first, as for the one-year distance, accuracies span over a relatively wide range of levels which, in this case, is significantly lower than the one-year distance outcome; also, the accuracy shaped priority list appears to be the least performing

except for the very last thresholds while there is no clear best performer, though literature ranking display an above average behaviour overall; higher correlation thresholds seem to benefit worse performers without materially impacting high accuracy gainers on average. Thus in general, Figure 3.12 describes a scenario with essentially the same features chart 3.3 holds but at an all lower range of accuracy.

Proceeding on, Figure 3.13 accomplishes the same task as Figure 3.12 but for Support Vector Machines. Here again the first main observation relates with the lower range of accuracy achieved. Interestingly, the width of the accuracy range almost perfectly resembles the parallel range in Figure 3.4 at the one-year distance. Moreover, a similar trend as for Logit is exhibited: while accuracy priority list performs rather poorly compared with other lists, except for the last thresholds, literature outperforms others in much of the chart. Finally, there seems to be a slight tendency in higher accuracies the higher the correlation allowed among financial indices.

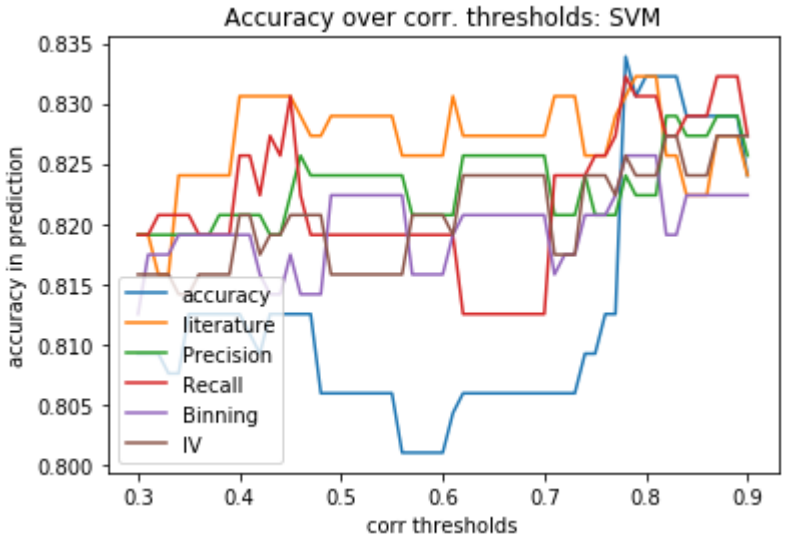



Figure 3.13 Accuracy levels achieved by the SVM model on the six ready-to-use lists of financial ratios

Chart 3.14, then, illustrates the trends followed under the KNN framework. Its behaviour quite surprisingly leads the accuracy range at being significantly narrower than in Figure 3.5 and at the exact same level of the accuracies elicited from 0,6 to 0,9 range of correlation thresholds. Further, there is no sign of the negative relationship between correlation allowed and level of accuracy which has been discussed for one-year distance outcomes. Here in fact, KNN do not show any meaningful and visible dependence to the level of correlation allowed. The only traceable observation in this sense links to the fact that higher thresholds bring to much narrow accuracy range. Again, as for Logit and SVM before, the accuracy ready-to-use priority list performs below average except for the very final thresholds. On the other hand, there does not seem to be an individual best performer with basically all other priority lists gravitating around the same mean. Finally, a comparison among the three first model presented brings interesting

conclusions. If on one side, Logistic regression is clearly the least performer overall, on the other, there is no clear-cut way to establish which model shows superiority in results. What should be underlined, in any case, is that KNN shows so far, the highest robustness in terms of distance from relevant year of default.

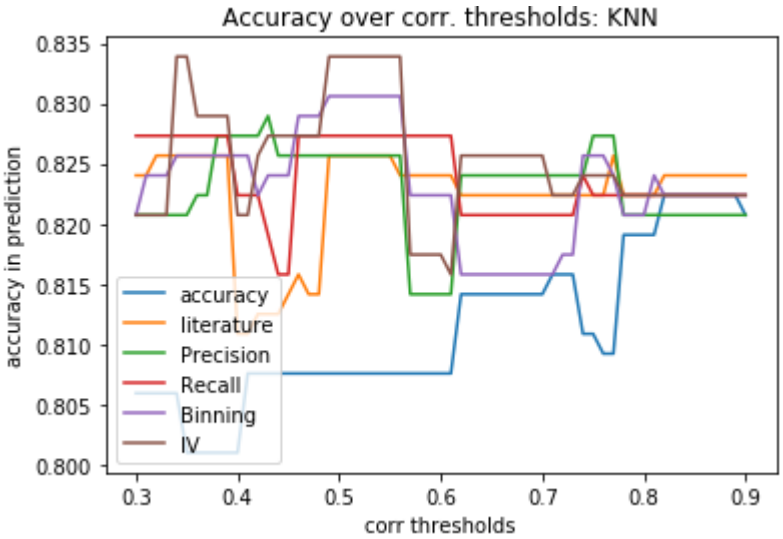



Figure 3.14 Accuracy levels achieved by the KNN model on the six ready-to-use lists of financial ratios

Afterward, Figure 3.15, depicts the evolution of the six priority lists under the AdaBoost ensemble model. AdaBoost shows a 10% average accuracy loss from the previous scenario of one-year distance. Excluding Logistic regression for its wider than average accuracy ranges, such loss is the deepest so far encountered and signals AdaBoost relative weakness as the prediction time increases. Moreover, in contrast with the first three models presented, accuracy priority list does not underperform the others. Indeed, here there does not seem to appear either a clear 'winner' or 'loser'.

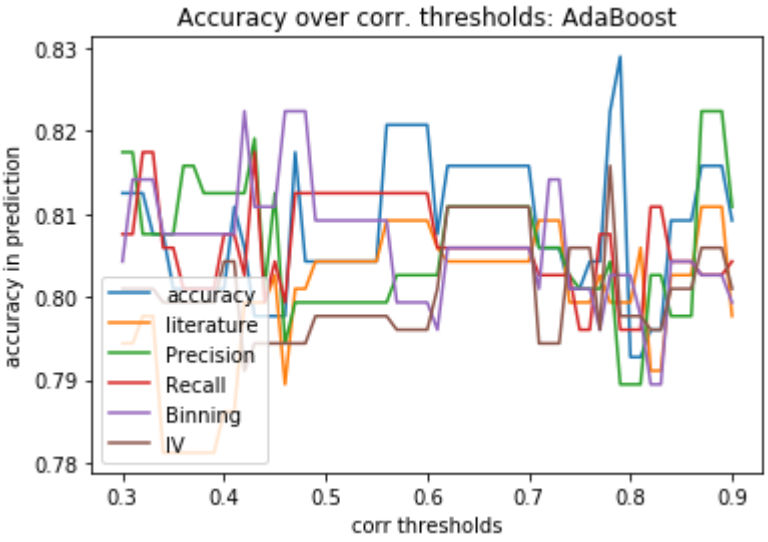



Figure 3.15 Accuracy levels achieved by the AdaBoost model on the six ready-to-use lists of financial ratios

Moving over, Figure 3.16, instantiates Decision Tree model reached accuracy levels over multiple correlation boundaries. Again, the chart suggests an overall loss of accuracy with the

greater time distance, though not as deep as AdaBoost shows. Also, the accuracy priority list ranks again at the bottom, on average, in terms of performance with a trough between 0.4 and 0.5 thresholds. Finally, different levels of covariance allowed among ratios do not appear to affect the mean accuracy unless marginally.

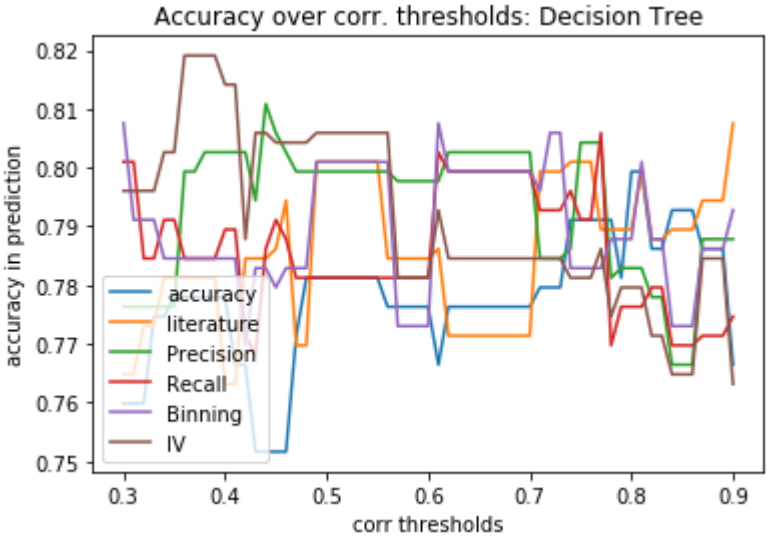



Figure 3.16 Accuracy levels achieved by the Decision Tree model on the six ready-to-use lists of financial ratios

The last graph, picture 3.17, reports XGBoost model accuracy outcomes. Interestingly, as with the only other ensemble model, also XGBoost presents a loss of about 10% in its average accuracy. However, it is hard to conclude that ensemble methods suffer more than others a greater timespan of prediction. Indeed, both AdaBoost and XGBoost, also share the highest prediction performances in the one-year distance scenario. To shed more light on this it will be helpful looking at the behaviour at a three-years distance from the relevant defaulting year.

A second observation for Figure 3.17, is represented by the fact that it looks to be a weak but significant negative relationship between the correlation allowed and the overall level of accuracy reached. This contrast with the very weak or absent relationship in Figure 3.8.

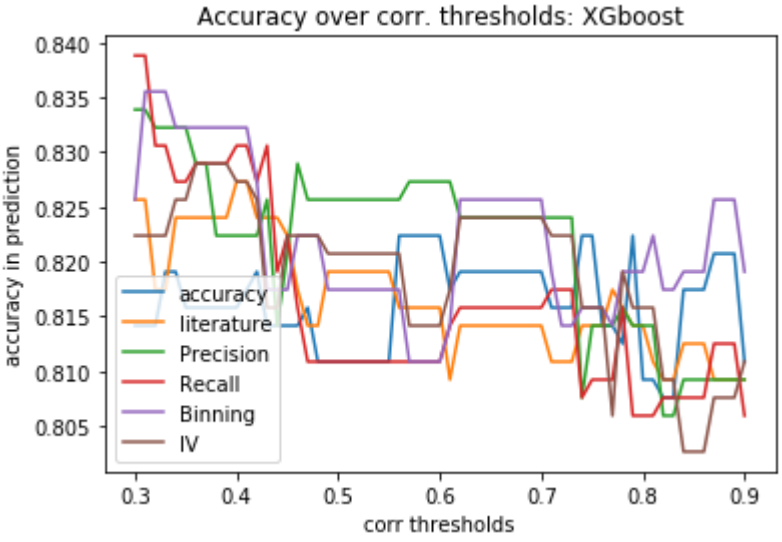



Figure 3.17 Accuracy levels achieved by the XGBoost model on the six ready-to-use lists of financial ratios

To conclude, table 3.2 ranks prediction models (left to right) and reports their ranges.

Table 3.2 Ranking of the best predicting models (left to right) with accuracy intervals (in %)

| MODEL | XGBoost | SVM | KNN | AdaBoost | Decision T | Logistic |
|---|---|---|---|---|---|---|
| RANGE | 80 - 84 | 80 - 84 | 80 - 84 | 78 - 83 | 75 - 82 | 50 - 70 |

From Table 3.2 an interesting conclusion can be drawn: adding one year to the final prediction period, levels prediction models accuracies to the point that there is not anymore a best performer. This is of course true excluding Logistic regression which confirms a certain poorer than average ability to predict companies' status.

To spot even more clearly the differences of performance due to increased prediction time, the curves entailing three-year distance accuracy levels over multiple correlation thresholds are plotted.

To this end, Figure 3.18 charts Logistic regression results. Remarkably, the model does not worsen its outcomes in the new scenario, the bottom boundary is still limited to 50% accuracy, but rather exhibit a slight enhancement as the upper bound rises to 75% with the accuracy priority list. Though, the arguably more surprising fact relates with the increased performance of the accuracy priority list which was found as poorest performer in the two-years case. Finally, literature list results appear to be almost perfectly in line with the precedent chart, Figure 3.13.

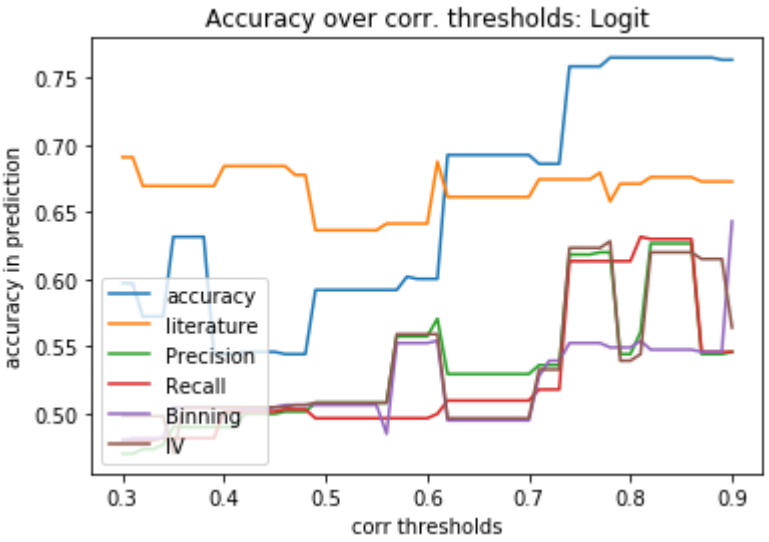



Figure 3.18 Accuracy levels achieved by the Logistic regression model on the six ready-to-use lists of financial ratios

Further, Figure 3.19 also describes an intriguing occurrence. Contrary to the logical reasoning through which the loss generated from two- to three-years prediction timespan should resemble the previous loss generated from one- to two- years distance, SVM seems to be suffering less than the one- to two-years change. Indeed, the average accuracy deviation stands at just 3,5% points for the upper bound and near to nothing for the lower bound. In other words, the average

loss in accuracy does not translate into a general shift of the range over the y-axis, but rather in only a decrease of the upper bound level. This may already suggest, at a practical stage, that it is more valuable following SVM predictions at a three years horizon rather than two since the cost in terms of efficiency is so little that the benefit coming from one more year of 'knowledge' might be greater.

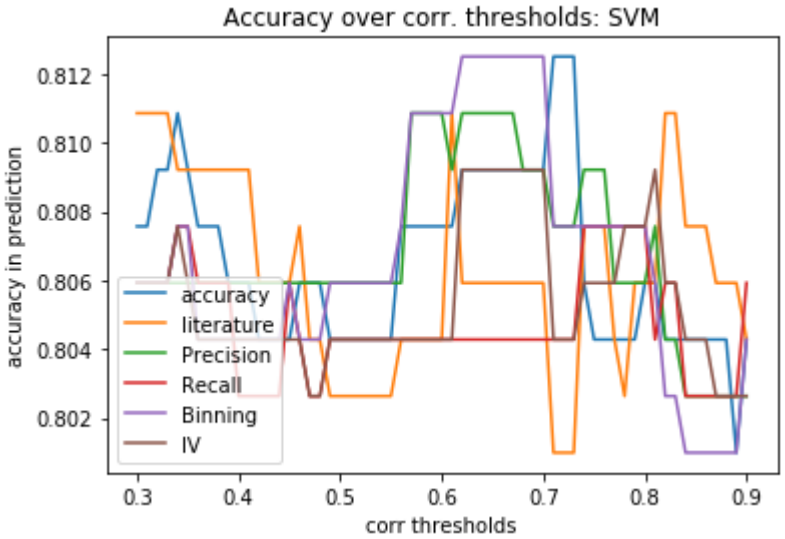



Figure 3.19 Accuracy levels achieved by the SVM model on the six ready-to-use lists of financial ratios

Going forward, Figure 3.20 plots KNN outcomes. Here the same dynamic as with SVM is detected: there is only a loss in average accuracy due to the thinning of the accuracy range. Furthermore, as for the two-years distance from the relevant year, there appears to be no strict relationship between the correlation threshold and the average level of accuracy.

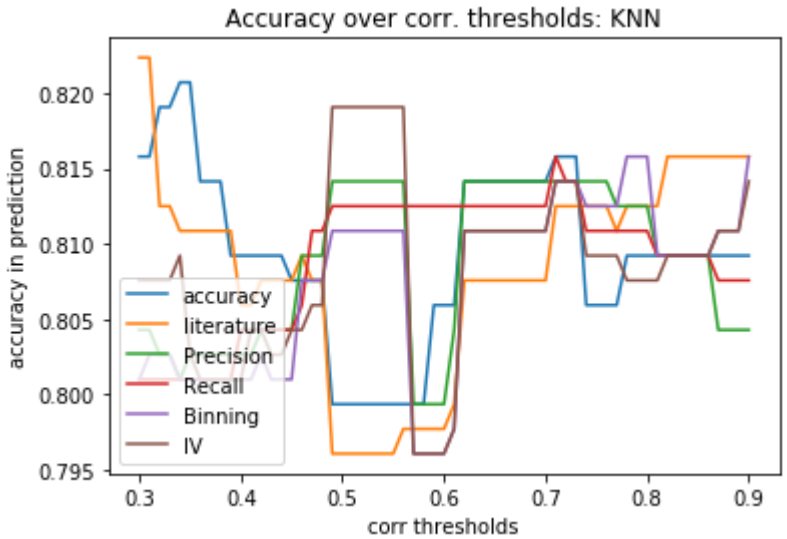



Figure 3.20 Accuracy levels achieved by the KNN model on the six ready-to-use lists of financial ratios

Unlike the previous cases, Figure 3.21, showing AdaBoost results over multiple correlations, illustrate that the model lowers both upper and lower bounds. Indeed, the chart exhibit a small but relevant shift in the y-axis, of about 1,5% points. The shift, however, confirms once more that the loss from increasing the prediction time between one and two years and from two and

three years has differing magnitude. Further, as manifested by the one- and two-years distance charts, there seems to be no clear interrelation between the set correlation threshold and the average level of accuracy in the model.

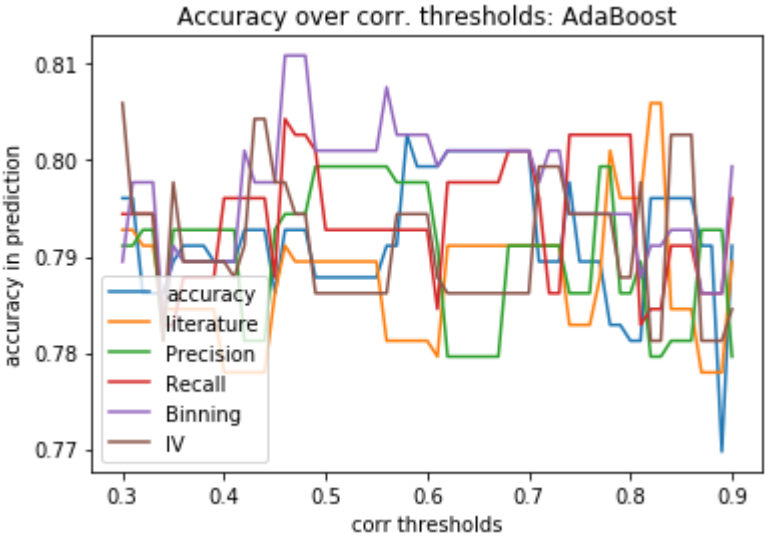



Figure 3.21 Accuracy levels achieved by the AdaBoost model on the six ready-to-use lists of financial ratios

Finally, Figure 3.22 and 3.23, reports Decision Tree and XGBoost accuracy achievements under a three-years distance timespan scenario respectively. Both plots follow AdaBoost findings in that their accuracy intervals shift of few percentage points on the y-axis, 1% for Decision Tree and just less than one for XGBoost. In both, moreover, there does not appears to be a best performer nor a worst one. The only observation in this sense is linked to the accuracy priority list trend which overperforms its peers in up to 0,6 correlation threshold and only under XGBoost prediction frame.

What is more, both graphs do not exhibit any definitive relationship between accuracy levels and correlation thresholds. The only, rather weak, connection can be evidenced for XGBoost which, similarly to the one-year distance time, shows a narrower range of accuracy records at higher correlation thresholds.

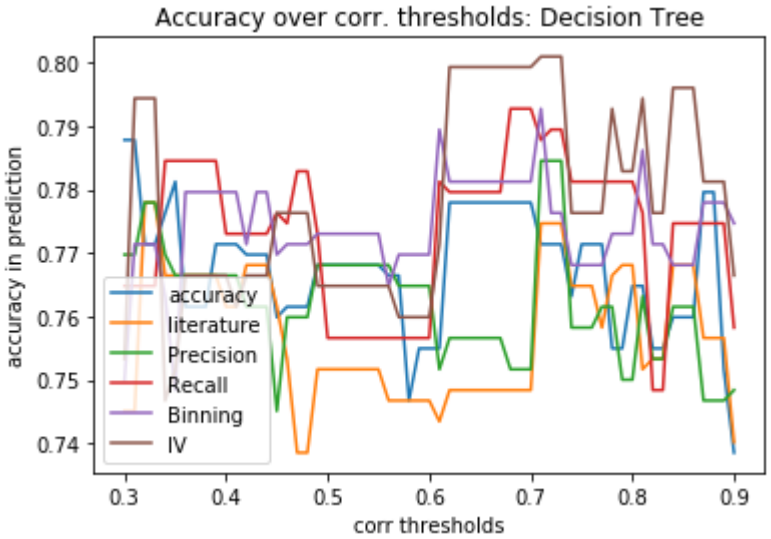



Figure 3.22 Accuracy levels achieved by the Decision Tree model on the six ready-to-use lists of financial ratios

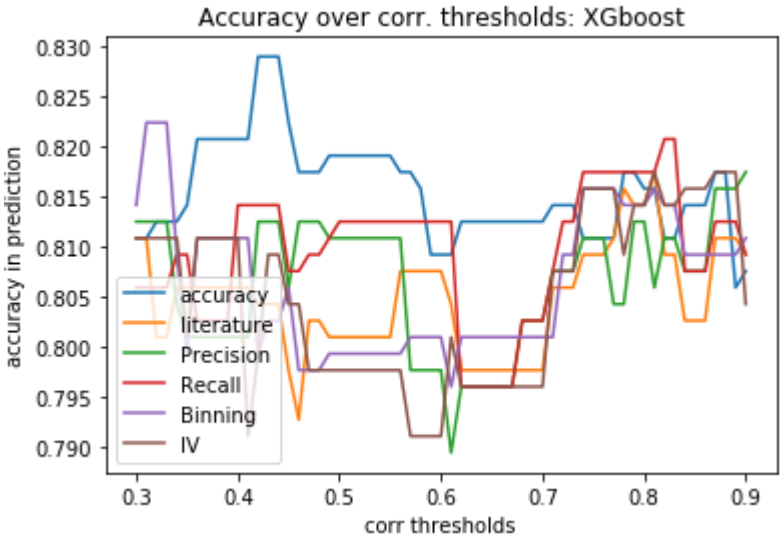



Figure 3.23 Accuracy levels achieved by the XGBoost model on the six ready-to-use lists of financial ratios

To conclude, table 3.3 ranks prediction models (left to right) and reports their ranges for the three-years distance from the relevant defaulting time.

Table 3.3 Ranking of the best predicting models (left to right) with accuracy intervals (in %)

| MODEL | XGBoost | SVM | KNN | AdaBoost | Decision T | Logistic |
|---|---|---|---|---|---|---|
| RANGE | 79 - 83 | 80 - 81 | 79 - 82 | 77 - 81 | 74 - 80 | 50 - 75 |

The second step of the analysis, entailing the examination of ROC AUC for specific correlation boundaries is here limited to the observation of the 0,6 threshold scenario and only for the individual accuracy based priority list. This will pursue two objectives: first, it allows to conduct a relevant comparison between the three years distance cases on the most critical, for practical applications, correlation threshold; second, it will keep the analysis straightforward since the differences from 0,3, 0,6 and 0,9 do not appear substantial, as verified in the previous section dedicated to the one-year distance case. Nonetheless, results related to 0,3 and 0,9 thresholds for both two- and three-years distances can be consulted in appendix 10.

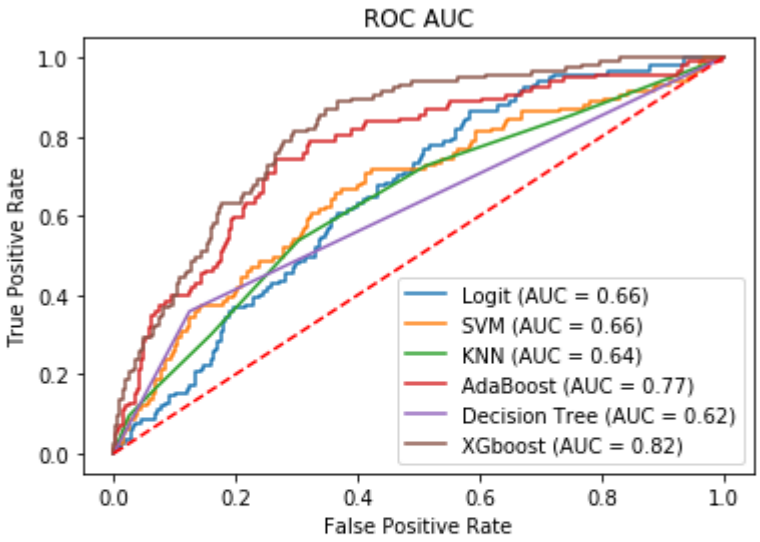
Figure 3.24 represents the ROC AUC for all six prediction models under the 0,6 correlation threshold and two-years distance scenario. Although the final ranking it suggests very much resembles Figure 3.10's, it depicts an interesting situation. The primary observation that can be spotted, relates to the closedness in AUC results. Indeed, contrary to the one-year case all AUC measures, except for AdaBoost and XGBoost, appear to be close to each other. This may be explained by the average loss of predictive ability that the increase prediction time has brought to the picture. Also, from AUC AdaBoost is solidly performing as second best predicting model which is in neat contrast to the ranking based on accuracy only. Further, XGBoost performance

reveals a relative high robustness to the increased prediction time since its AUC mark is the least damaged among the top three in the one-year case (SVM, AdaBoost and XGBoost). Overall then, AUC prizes the ensemble models significantly more than the accuracy measure.

The least performing model is Decision Tree which, except for Logistic Regression, is in line with the accuracy findings. Finally, at a general level, SVM appears to suffer the most from the change in prediction timespan.

Attached to the chart a table presents accuracy, precision and recall measures at the 0,6 correlation threshold. It, as already pointed out, present a different situation than the one elicited from AUC data. Although it is evident a loss in performance affecting all six models, the accuracy dimension clearly identifies Logit as the worst predictor. Moreover, the gap in performance between AdaBoost and XGBoost seems here replenished: they appear to achieve the same level. Also, in line with the observation for the one-year distance case, the recall dimension shows poor results, when compared with precision and accuracy for all models except Logistic regression.

```
Priority list: accuracy; Correlation threshold: 0.6
```




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.503289 | 0.594292 | 0.640382 |
| SVM | 0.800987 | 0.571868 | 0.508947 |
| KNN | 0.807566 | 0.654801 | 0.50651 |
| AdaBoost | 0.820724 | 0.706264 | 0.631843 |
| Decision Tree | 0.776316 | 0.629626 | 0.617369 |
| XGboost | 0.822368 | 0.716021 | 0.60682 |

Figure 3.24 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

Figure 3.25 plots ROCs and reports AUC values for 0,6 correlation thresholds and three-years distance to the relevant defaulting time. The first impactful information the chart conveys

relates to AUC outcomes for Logistic regression, SVM and KNN. These are in fact labelled as performing better on a three years timespan than on, the more reasonable, two years. This fact, given that the difference is by no means substantial, indicates again that the loss in performance, on average, suffered between two years and three years cases is only marginal compared to the loss experienced in moving from the one-year to the two-years distance. The other three prediction models, namely AdaBoost, Decision Tree and XGBoost, instead follow an expected behaviour and decrease their overall valuation. Interestingly, the three years scenario features the lowest total gap among models. The actual ranking, however, appears to remain unchanged from the previous cases.

The attached table confirms the findings with several models that slightly improve their outcomes instead of decreasing them. Interesting is in particular the accuracy recorded for the Logit analysis since it drastically improves, by 10% points, against Figure 3.24 attached table. This behaviour is reflected in the already seen unexpected trend of the accuracy priority list followed in chart 3.18 (blue line). Finally, recall measures display a behaviour essentially in line with the preceding findings.

```
Priority list: accuracy; Correlation threshold: 0.6
```

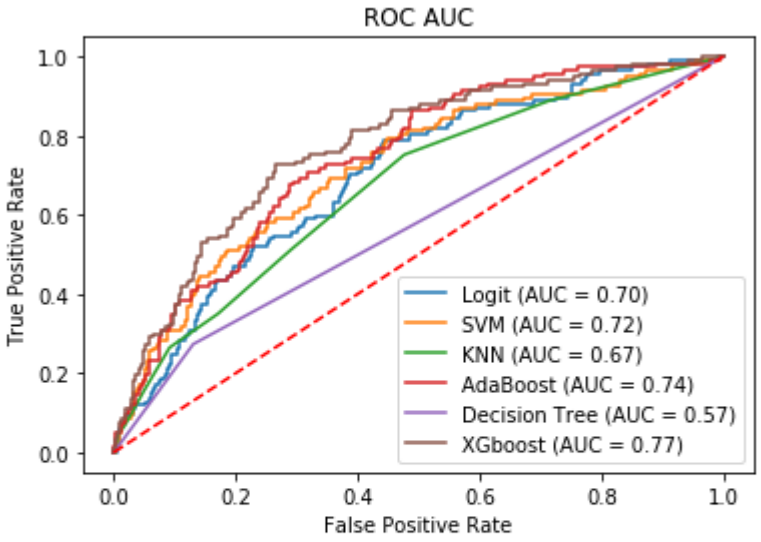



Figure 3.25 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.600329 | 0.606441 | 0.671166 |
| SVM | 0.807566 | 0.656879 | 0.519531 |
| KNN | 0.805921 | 0.643078 | 0.525023 |
| AdaBoost | 0.799342 | 0.646846 | 0.582798 |
| Decision Tree | 0.754934 | 0.583659 | 0.571579 |
| XGboost | 0.809211 | 0.670538 | 0.549846 |

The list of ratios applied at 0,6 correlation threshold is the same adopted for the one-year case and is thus not reported.

## 3.2 TESTING SAMPLE RESULTS OF EXTERNAL FIRMS

This section aims at examining the six prediction models' performance on a test set composed of 3482 companies headquartered in Italy. The analysis compares Veneto only models' accuracy measures and ROC AUCs with the new test sample to look for model ability to generalize predictions. All companies' data are taken from the AIDA database as already described for Veneto only firms. Moreover, entities only belong to a selected list of ATECO subclasses as they are object of analysis in a parallel study looking to identify the best prediction models for Italian firms with specific features.

All 3482 are employed as test for the prediction models trained on the previous described training set. That is to say that the training set is only composed by Veneto companies while this test set includes also, and in majority, external companies. Such train and test sets composition bring about an appreciable drawback: results on the test set might be distorted due to specific characteristics affecting companies outside Veneto region. In other terms, training the models on data related to a restricted area, Veneto in this case, may result in biased performance metrics from external firms since the peculiarities, the features, through which models determine their key parameters and thresholds might be insufficient to truly represent companies headquartered outside Veneto. The analysis here proposed is nonetheless relevant and reliable. This because of two reasons: first, the main characteristics connoting firms in Veneto can be extended to all Italian companies as proxy of their activity with limited degree of misrepresentation; secondly, since all ATECO classes have been included in the training set, models are to be considered able to handle them also for larger samples[40]. It is also true, however, that the issue aforementioned should be bore in mind when interpreting results.

### 3.2.1 One-year distance results

Table 3.4 Veneto test set accuracy results (already introduced) per model and priority list

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.703947 | 0.898026 | 0.814145 | 0.909539 | 0.868421 | 0.922697 |
| 1 | 0.657895 | 0.899671 | 0.814145 | 0.911184 | 0.879934 | 0.919408 |
| 2 | 0.679276 | 0.888158 | 0.809211 | 0.902961 | 0.870066 | 0.907895 |
| 3 | 0.771382 | 0.883224 | 0.814145 | 0.914474 | 0.884868 | 0.916118 |
| 4 | 0.702303 | 0.886513 | 0.809211 | 0.904605 | 0.883224 | 0.916118 |
| 5 | 0.707237 | 0.883224 | 0.804276 | 0.899671 | 0.883224 | 0.916118 |

[40] As explained in chapter 2, ATECO first figure 9 is not included in the sample for lack of data. This however do not hinder the analysis since no external company belongs to the 9th class.

Table 3.5 External test set accuracy results per model and priority list

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.709362 | 0.820505 | 0.820218 | 0.819070 | 0.815049 | 0.402642 |
| 1 | 0.738656 | 0.821080 | 0.820793 | 0.831132 | 0.442275 | 0.708501 |
| 2 | 0.735210 | 0.812464 | 0.823952 | 0.818782 | 0.650488 | 0.283171 |
| 3 | 0.729179 | 0.822803 | 0.819644 | 0.812177 | 0.663125 | 0.368754 |
| 4 | 0.637852 | 0.819644 | 0.823952 | 0.821367 | 0.309879 | 0.339460 |
| 5 | 0.680356 | 0.817921 | 0.821367 | 0.590465 | 0.806720 | 0.258185 |

The first set of results assessed relates to the one-year distance from the relevant defaulting year. looking at it, only the 0,6 correlation threshold and accuracy based priority list are taken into account. This allow a thorough comparison between the external test set and the Veneto one, while keeping only informative metrics. All results concerning to 0,6 threshold and other priority lists, not illustrated here, are reported in appendix 11.

Table 3.4 and 3.5 present the accuracies achieved by all six prediction models on the various priority lists in the Veneto test set and the external test set respectively. Row numbers represent a specific priority list, namely: 0 for accuracy, 1 for Literature, 2 for precision, 3 for recall, 4 for binning and 5 for IV based list of ratios.

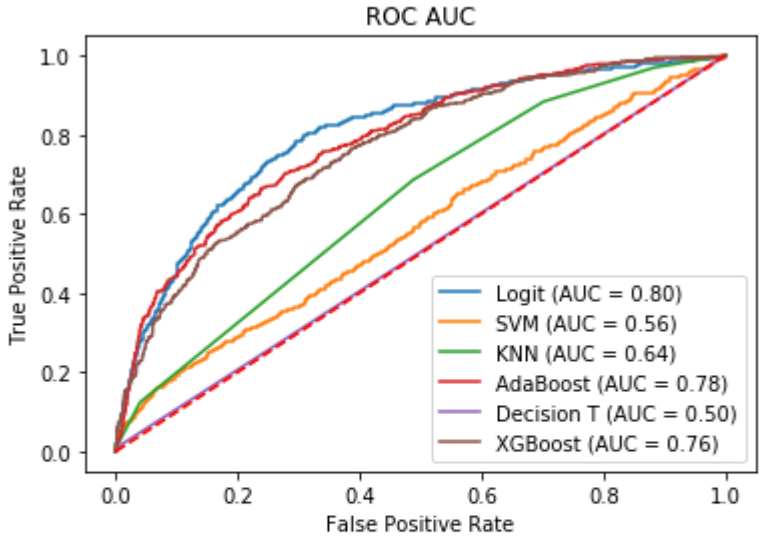
The comparison between the two table elicits interesting results. As first, looking the three least performers for Veneto only test set, logistic regression and KNN appears to be behave equally well under the two test sets. On the contrary, Decision Tree suffers from the change operated. Specifically, Decision Tree exhibit low results for the binning based priority list where it reaches only 31% overall accuracy. Since this is an isolated result, in the sense that 31% represents an outlier with respect to the other accuracies achieved, this may well be attributed to the changing features of the external test set. In other words, it can be assumed that given 31% is the lowest recorded performance, far from the other recorded measures, the thresholds set by Decision Tree under binning priority list during the training process can be assumed as different from those that would have been set if the training set included a significant number of external firm data. A comparable argument can be applied to Decision Tree results on the literature list, reaching 44% accuracy.

A second observation considers the top three performers in Veneto only test set: SVM, AdaBoost and XGBoost. While the two former show in-line performances with an expected small decrease in accuracies, XGBoost reports fairly poor results. Indeed, if SVM and AdaBoost only loose few percentage points, with the only exception represented by the latter

applied to IV based priority list, XGBoost always looses more than 50% accuracy with the sole exception represented by the literature priority list. Its poor results might be symptom of overfitting. Overfitting occurs whenever a model does not show a sufficient ability to generalize the performances obtained on a limited test set on o broader number of tests. Here in fact, XGBoost shows poor ability in extending to the new, broader test set, the achievements reached under the Veneto only test set. Even discounting for the presence of the external factors, that brings in any case to essentially comparable results in the other models, it cannot be explained the lower than average results related with XGBoost. For this reasoning it can be concluded that XGBoost might suffer from overfitting.

As a further step in the analysis, the ROC AUC curve plot is examined.

```
Priority list: accuracy; Correlation threshold: 0.6
```




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.709362 | 0.65075 | 0.744461 |
| SVM | 0.820505 | 0.674371 | 0.52226 |
| KNN | 0.820218 | 0.910057 | 0.501592 |
| AdaBoost | 0.81907 | 0.534862 | 0.500271 |
| Decision T | 0.815049 | 0.555161 | 0.504028 |
| XGBoost | 0.402642 | 0.594271 | 0.616349 |

Figure 3.26 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

Figure 3.26, reporting the ROC AUC values for the six prediction models under 0,6 correlation threshold and for the accuracy based priority list, depicts quite a different scenario from what just seen from the accuracy metrics. The chart indeed, presents Logistic regression as the best absolute model, confirming the generalization ability of the model. Then, AdaBoost and XGBoost follow. If the former could be expected from Table 3.5 to be ranked as second best performing, the same cannot be stated for the latter. To understand it, it is sufficient to look at the table attached in Figure 3.26. From it can be elicited that whilst XGBoost accuracy, as

previously seen, is lower than average, its recall ability stands as second best, 61% recall. Through this fact it can be assumed that is accuracy drops significantly, other measures of performance show different outcomes so to determine an AUC value seemingly reversed in results. To conclude, KNN, AUC of 0,64, is chased by SVM, 0,56, and Decision Tree, 0,50.

### 3.2.1 Two-years distance results

Moving on, the two-years distance from relevant defaulting year scenario is analysed. Table 3.6 and 3.7 report the accuracy over all models and for all priority lists for Veneto only and external test sets respectively. In the first table it is possible to recognize the loss in accuracy risen from the increased prediction timespan discuss in chapter 3.1 and visible with respect to Table 3.4. From the comparison between the two tables is then possible to see the change in performance brought from the new test set. The first, main consideration that can be adduced from such comparison regards the difference between the relative modifications in the increase prediction period with the different test sets. Indeed, when focusing on the passage from Table 3.4 to Table 3.6 and from Table 3.5 to Table 3.7 the deltas that can be spotted appear to differ in magnitude from model to model.

Table 3.6 Veneto test set accuracy results (already introduced) per model and priority list

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.503289 | 0.800987 | 0.807566 | 0.820724 | 0.776316 | 0.822368 |
| 1 | 0.606908 | 0.825658 | 0.824013 | 0.809211 | 0.784539 | 0.815789 |
| 2 | 0.600329 | 0.820724 | 0.814145 | 0.802632 | 0.797697 | 0.827303 |
| 3 | 0.531250 | 0.819079 | 0.827303 | 0.812500 | 0.781250 | 0.810855 |
| 4 | 0.598684 | 0.815789 | 0.822368 | 0.799342 | 0.766447 | 0.810855 |
| 5 | 0.626645 | 0.820724 | 0.817434 | 0.796053 | 0.781250 | 0.814145 |

Table 3.7 External test set accuracy results per model and priority list

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.515796 | 0.326824 | 0.819070 | 0.818782 | 0.494543 | 0.819070 |
| 1 | 0.700460 | 0.215681 | 0.799540 | 0.580414 | 0.532165 | 0.458644 |
| 2 | 0.632970 | 0.576967 | 0.822516 | 0.813613 | 0.558013 | 0.481333 |
| 3 | 0.582998 | 0.483343 | 0.820505 | 0.818495 | 0.473866 | 0.770534 |
| 4 | 0.549397 | 0.310167 | 0.733774 | 0.819070 | 0.663125 | 0.813900 |
| 5 | 0.623779 | 0.802412 | 0.751580 | 0.784032 | 0.674612 | 0.641011 |

That is to say, all prediction models exhibit a higher decrease in overall accuracy for the new test set rather than Veneto only sample. The only exception being XGBoost, which seems to regain one of the first position in the accuracy ranking, and, slightly, Decision Tree, which does not decrease its average accuracy more than the decrease experienced under the smaller test set case. The worst achiever results to be SVM which, in line with the precedent reasoning on XGBoost might reveal signs of overfitting in this phase. Further, Logistic regression again show certain degree of generalization ability with accuracies essentially comparable among scenarios. Finally, also AdaBoost and KNN remarks their compatibility with the new test set with values equal, or approximately equal, in the two tables.

As for the second assessment step, Figure 3.27 plots the ROC AUC of the six prediction models undergoing the same conditions of correlation threshold and priority list. The chart describes a situation basically dominated by two groups of results. On the one hand the group of models totalling more than 0,5 in AUC and those below the red dotted line with lower than 0,5 AUC. Confirming the findings from Figure 3.26, Logistic regression leads in pair with AdaBoost and strictly followed by XGBoost, 0,69 and 0,68 respectively. Then, still higher than 0,5, KNN reaches 0,61 while SVM and Decision Tree confirms to be the least performers with just 0,46 and 0,45.

```
Priority list: accuracy; Correlation threshold: 0.6
```

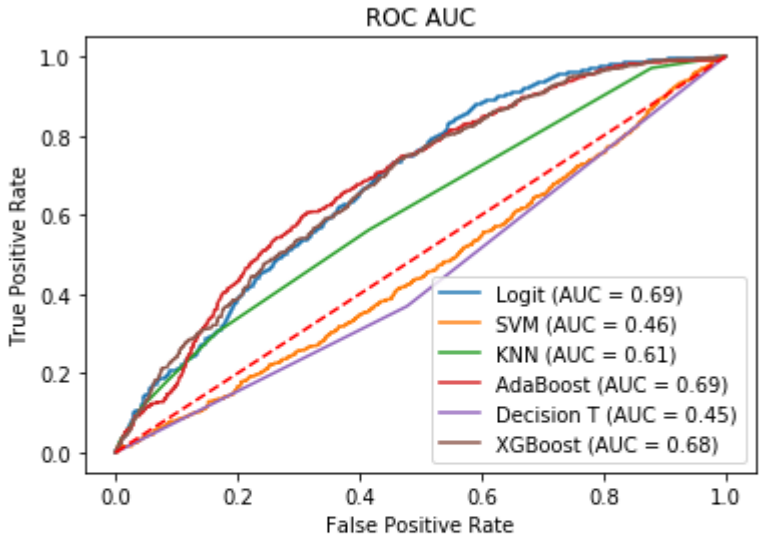



Figure 3.27 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.515796 | 0.589626 | 0.644389 |
| SVM | 0.326824 | 0.485582 | 0.481918 |
| KNN | 0.81907 | 0.649034 | 0.511448 |
| AdaBoost | 0.818782 | 0.591931 | 0.501958 |
| Decision T | 0.494543 | 0.467698 | 0.445751 |
| XGBoost | 0.81907 | 0.534862 | 0.500271 |

Finally, from the attached table it can be elicited both that precision and recall measures reach quite low outcomes for all models and once more recall for logistic regression corroborates the model generalization ability.

### 3.2.1 Three-years distance results

The third and last scenario considered refers to the three-years distance from the relevant defaulting year under the 0,6 correlation threshold and accuracy based priority list conditions. To complete it, Table 3.8 and 3.9 report accuracy levels for all six prediction models over all six priority lists for Veneto and external test sets.

A first consideration relates with the level of SVM which appears to achieve higher results than under the two-years case. This fact might indicate that the probable overfitting condition is limited to the two-year status. All other models reach results essentially in line to what expected from the previous analysed tables. Here KNN and AdaBoost rank as best performers, followed by XGBoost, SVM and logistic regression in average values. The fact that XGBoost both in two- and three-years scenarios achieves over average accuracy outcomes and that it exhibits AUC values that almost always stand in the top positions, strengthen the possibility that Table 3.5 records truly reflect an overfitting condition. The final, least performing framework is represented by Decision Tree with 24% as lowest point.

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.600329 | 0.807566 | 0.805921 | 0.799342 | 0.759868 | 0.809211 |
| 1 | 0.641447 | 0.804276 | 0.797697 | 0.781250 | 0.746711 | 0.807566 |
| 2 | 0.557566 | 0.810855 | 0.799342 | 0.797697 | 0.764803 | 0.797697 |
| 3 | 0.496711 | 0.804276 | 0.812500 | 0.792763 | 0.756579 | 0.812500 |
| 4 | 0.552632 | 0.810855 | 0.796053 | 0.802632 | 0.774671 | 0.800987 |
| 5 | 0.559211 | 0.804276 | 0.796053 | 0.794408 | 0.769737 | 0.791118 |

Table 3.8 Veneto test set accuracy results (already introduced) per model and priority list

| | Logit | SVM | KNN | AdaBoost | Decision T | XGBoost |
|---|---|---|---|---|---|---|
| 0 | 0.635267 | 0.211086 | 0.813326 | 0.818208 | 0.511775 | 0.721999 |
| 1 | 0.645893 | 0.265939 | 0.811028 | 0.817346 | 0.747846 | 0.823952 |
| 2 | 0.577829 | 0.806720 | 0.819070 | 0.697300 | 0.252728 | 0.547387 |
| 3 | 0.531017 | 0.818208 | 0.819931 | 0.810741 | 0.556577 | 0.820793 |
| 4 | 0.559736 | 0.797530 | 0.819357 | 0.612292 | 0.235497 | 0.702757 |
| 5 | 0.550258 | 0.797530 | 0.819644 | 0.540781 | 0.543653 | 0.822229 |

Table 3.9 External test set accuracy results per model and priority list

```
Priority list: accuracy; Correlation threshold: 0.6
```

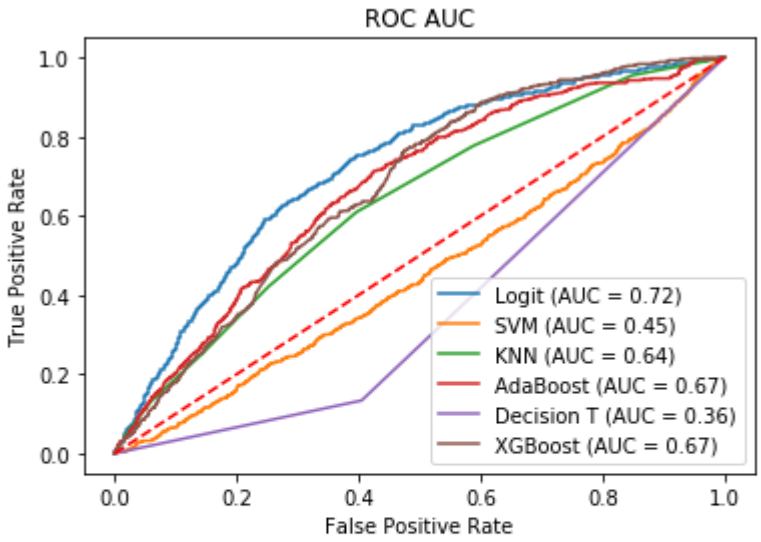



Figure 3.27 ROC AUC for all six prediction models and related to the results obtained with the accuracy list of ratios at a 0,6 maximum correlation threshold

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.635267 | 0.606272 | 0.678148 |
| SVM | 0.211086 | 0.474523 | 0.490801 |
| KNN | 0.813326 | 0.593214 | 0.514154 |
| AdaBoost | 0.818208 | 0.625368 | 0.508439 |
| Decision T | 0.511775 | 0.412551 | 0.364356 |
| XGBoost | 0.721999 | 0.560105 | 0.568979 |

From Figure 3.27 moreover, an interesting result can be observed: not only logistic regression is confirmed as the overall best prediction model but it also increases, though only slightly, its AUC value. Other than Logit, also KNN slightly increases its AUC performance while all the others lower their outcomes for few points. The exception to this is objectified by Decision Tree which drastically worsen its condition, reaching only 0,36. The fact that all prediction models, except Decision Tree, display similar outcomes to the two-years distance scenario confirms the tendency first spotted in the previous analysis, of steady results between the two- and three-years distances. In other words, it is once more clear that the gap in performances created when moving from one- to two-years of prediction timespan finds almost no comparison with the subsequent time change generated gap.

To conclude, the attached table presents a situation very much alike to the one depicted by the AUC measures. The least performers are SVM and Decision Tree which reaches the lowest recall at 36%. Also, logistic regression outperforms all others in the recall statistics while it is joint and slightly overcome by AdaBoost under the precision parameter.

## 4. CONCLUSIONS

The following section recaps the main steps undertaken throughout the research, outlines the conclusions reached, suggests new direction of research and ends with the author personal comment.

To begin with a brief recap, this study aims at developing and applying sound bankruptcy prediction frameworks on datasets composed by companies headquartered inside Veneto region in Italy. To do so, six prediction models ranging from more traditional statistical concepts to newer machine learning based algorithms, have been employed, namely: Logistic regression, Support Vector Machines, K - Nearest Neighbour, AdaBoost, Decision Tree and XGBoost.

Deepening, the whole process can be split into 6 steps:

-At first, an analysis of the data available have been carried out. Data, ten years of financial statements drawn from Bureau Van Dijk's AIDA database (a Moody's analytics company), refers to 424 companies defaulted between 2013 and 2019 and 29711 non-defaulting firms. 'Default' entails here the Italian legal discipline of 'Concordato preventivo' and 'Procedura concorsuale liquidatoria'. Firms are for the majority small and medium enterprises mainly based in Vicenza and Padova provinces with high defaulting occurrences between 2014 and 2017.

-Secondly, all non-defaulting entities have been filtered through the Propensity Score Maching procedure. This has been applied to both reduce the imbalance in datapoints availability between the two groups and to optimally match sound companies to failing ones. Five non-defaulting firms have been associated with each defaulting on the basis of Sales, to account for the size of the business run, and Equity to total Assets, to include features pertaining to the balance sheet solidity (e.g. leverage) to which firms are exposed to. A total of 2430 firms' data, 405 failing and 2025 non-failing, results from the procedure.

-Further, 54 financial indices have been computed for each enterprise. Indices refers to the most used ratios in the relevant literature and are mostly chosen from Bellovary et al. (2007) review.

-As fourth step, an individual analysis of each financial index has been executed. It comprised undertaking an univariate logistic regression, from which measures of ROC AUC, accuracy, precision and recall have been elicited, and the binning procedure. Binning is performed by charting the number of defaults per decile inside the specific ratio distribution to look for the average ability in prediction as well as generally determining the complexity associated with it;

calculating Weight of Evidence values per every decile; and finally compute the overall Information Value granted by the ratio.

-Forward, from results obtained in the fourth step, six rankings of ratios have been established. These, called 'priority lists', defines best and worst financial ratios on the basis of their individual assessment and are based upon: accuracy, precision, recall measures from the univariate logistic regression analysis, the slope of the best fitting line from the binning charts (which accounts for either inability or complexity in interpretation of the single ratio), the individual Information Value scored and the frequency of adoption in relevant past researches on bankruptcy prediction.

-Moreover, correlations among all financial indices have been computed. These, contained in the average correlation matrix, refers to the average correlation found over ratios in each company dataset. From the correlation matrix it has been then possible to select a maximum correlation threshold to determine the maximum degree of interrelation accepted. After setting it, those ratios found to be over correlating have been judged on the premise of priority lists: identified the over correlating pair, the 'loosing' ratio is discarded.

-Finally, all surviving datapoints, after a pre-processing phase to take care of outliers and missing data, have undergone the six multivariate prediction models. Three main scenarios have been tested: one, two and three years of prediction time (i.e. the timespan between the prediction and the predicted moments). Models have been proved on both hold-out Veneto firms and Italian companies test sets.

### 4.1 Resulting remarks

A first relevant conclusion concerns Net Income to total Assets and Working Capital to Equity (or Net Worth) ratios.

Indeed, looking at appendix 6 reporting all six individual assessment-based priority lists, Net Income to total Assets always appears in the top position. This already tells that in line with the relevant literature, as implicitly stated by the literature based list, such financial ratio exhibits the highest individual level of performance for prediction purposes. In other words, in order to conduct a basilar and simple evaluation of companies in which just one parameter is considered, the most useful indicators is Net Income to total Assets for firm headquartered in Veneto region. Further, confirming the main theory on bankruptcy prediction, this conclusion can be extended to point out that whenever only a single parameter has to be involved in the assessment, then applying Net Income to total Assets should maximise results reliability.

On the contrary, at bottom positions of the majority of priority lists figures Working Capital to Equity (or Net Worth). This fact indicates that, under analysis conducted through the univariate logistic regression, it should not be preferred to any of the other 53 financial ratio considered.

It is however needless to set out that already these first observation should be taken as only applicable in contexts similar to the one set up in this study. For instance, it is possible that a revision might occur if a different individual assessment model was to be selected. Moreover, following the same reasoning it is hard to establish a conclusive and objective ranking of ratios individual predictive ability. This is due to the fact that weights can be assigned to priority lists to reflect the variety of needs that come in practice, making essentially subjective the final score given to each ratio. Here, however, Net Income to total Assets and Working Capital to Equity (or Net Worth) are the only two clearly positioned, leaving small or no room for subjective modifications.

A second straight conclusion comes from the comparison of the three prediction time scenarios considered. In fact, results suggest that the drop in performance observable from one year to two years dwarfs the parallel drop occurring when moving from two years to three years prediction period. This, in turn, implies that at a practical level choosing two years as prediction time to predict bankruptcy for Veneto companies with the model fine-tuned in this analysis, is essentially useless. In other words, since the performances elicited from two years and three years scenarios are almost comparable, it should always be more productive selecting the latter to have longer time predictions. The critical trade-off is thus between the single year scenario and the three years one whose performances differ significantly in favour of the one year setting.

What is more, looking at chapter 3 presenting the results, the metrics of performance achieved are in line with the literature except for the mentioned fairly limited drop in performance operated from two to three years. This outcome is in fact quite rare in other studies.

A third deduction relates to the fact that there do not seem to be an optimal level of correlation overall. This is particularly true looking at all prediction models' charts showing accuracy levels over multiple correlation thresholds for all six priority lists under all three prediction times scenarios. From them all is indeed clear that the level of maximum correlation allowed among financial ratios only slightly affect accuracy outcomes. In addition, ROC AUC charts for the three selected thresholds (i.e. 0,3; 0,6; 0,9) confirm the intuition. That is to say that inside the examined 0,3 to 0,9 range of thresholds, no one appears to display substantially better outcomes than the others. The sole exception is represented by Figure 3.5 illustrating the accuracy levels for KNN model over multiple correlation thresholds for one year prediction time and Veneto

only test set. In this case the level of interrelations does affect the average level of accuracy. Nonetheless, the same behaviour disappears under any other scenarios (e.g. increased prediction time span or different test set) and the actual change in average accuracy only accounts to six percentage points.

Furthermore, a fourth conclusion can be gained examining results from the external test set, which comprises companies headquartered in all Italy. From them, Logistic regression and KNN show the highest generalization ability. In other terms, they demonstrate the ability to reach accuracy and ROC AUC outcomes comparable to those achieved under the Veneto only test set. At an intermediate level of reliability figures then AdaBoost which seems to suffer only under particular combination of ratios and in the minority of results. The other three models on the contrary, show a rather significant inability in generalization. Specifically, while Decision Tree appears to score low but stable metrics on all three prediction times scenarios, XGBoost and SVM exhibit drastic decrease in accuracy and ROC AUC for only one specific time period: one-year and two-years prediction period respectively. These latter may be considered to overfit training datapoints for the two mentioned scenarios. However, the overfitting argument, as deepened in the previous chapter, cannot be conclusive since the inability in generalization shown could be due to radical differences in the characteristics connoting the external test set with respect to the internal, Veneto only, sample test. Nonetheless, it appears as an evidence the different behaviour in generalization between ensemble, tree-based prediction models (i.e. XGBoost, AdaBoost and Decision Tree) with SVM and Logistic regression with KNN.

A final, perhaps most interesting in practice, conclusion has to do with the definition of a general ranking of the six prediction models in the context analysed (i.e. for companies headquartered in Veneto). Carrying out this task is however quite a challenging operation due to mainly two reasons: first, looking to Veneto only results lowers the impact of the generalization argument which has not been proved except for a relatively small test set, set at 25% of the overall initial sample; further, the absence of an optimal correlation threshold implies a ranking based on intervals of results rather than straight and clean outcomes thresholds with the issue represented by overlapping section among intervals.

That being cleared, metrics of performance from the Veneto only test set, univocally indicate XGBoost as the most accurate and reliable prediction model. This is constantly confirmed by both metrics such as accuracy (e.g. it reaches a maximum accuracy of 93%), precision and recall and ROC AUC values at multiple correlation thresholds, under all three prediction times scenarios and for every priority list built. Following AdaBoost shows slightly higher degree of reliability with respect to SVM. Both models achieve just below 90% accuracy in their best

conditions and scenarios. Although the two models appear to suffer particularly the change from one- to two- and three-years distance from the relevant defaulting year, they always exhibit performances only limitedly poorer than XGBoost. Following, KNN stands as intermediate framework. Its metrics of performance, along with its shown generalization ability make it preferable to Logistic regression and Decision Tree. An important but secondary comment on KNN concerns its easy to interpret outcomes. Indeed, for its simplicity in execution, KNN represents a solid methodology with relevant results: 88% maximum accuracy under one-year prediction time scenario. To conclude, even though their performance metrics are many times overlapping each other, it can be said that Logistic regression is a more reliable prediction model than Decision Tree. The former indeed shows higher metrics of performance in both two and three years of prediction time scenarios to which it should be added that these are constant over prediction time while Decision Tree's outcomes drop drastically moving from one scenario to another, making them less predictable. Moreover, Logistic regression appears to embed a fairly higher generalization ability, to which Decision Tree demonstrate poor capability.

Again, it should be noted that the ranking so far detailed is only worth inside a context featuring equal setting and characteristics as those developed all over the analysis.

### 4.2 Further research directions

Along with the conclusions so far delineated, the author of this thesis is convinced that significant importance should be put on the suggestion of future research paths to pursue. Indeed, though conclusive and employable for practical use, the results listed above only represent a narrow and first attempt toward a general application of the bankruptcy prediction knowledge in Veneto and, more ambitiously in Italy. In other words, lots can be done to expand and deepen the efficacy of the six prediction models here included, let alone introducing altogether new prediction models or approaches.

Specifically, four main directions have been identified as valuable future streams of research: financial indices' structure and dynamic time composition, integration of parameters from additional fields, model integration with human intuition and prediction models combination.

The first suggestion refers to both the exploration of new financial indices structures and their time relevant composition. On the one hand, by 'exploration of new financial indices structures' is intended the possibility of considering new form of indices previously unseen. In fact, so far, the literature has been concerned with the exploitation of financial indices introduced from either practitioner intuition or the necessity some firm faced in its internal accounting processes.

In other terms, up to now the source of indices structure has historically been identified in figures (firms, academics, practitioners, etc.) aiming to solve some kind of measurement issue. On the contrary, all those non-practical, and perhaps more subtle, indices have not been adequately explored. The suggestion thus relates to a more rigorous exploration of all, or many more, kinds of financial ratios to look for less human-friendly parameters achieving high individual assessments. The reasoning underpinning this proposal starts exactly from recognizing that there may be some machine-friendly financial ratios, some hidden knowledge, that could be exploited in team with the traditional, more understandable indices to achieve higher prediction performances and reliability.

On the other hand, for 'time relevant composition' is considered the computation of the average financial ratio then applied to prediction models. In this research four years mean has been used as conclusive parameters. It would be undoubtedly useful looking for other combination of year ratios as a two, three, five or six average values to look for the most efficient and efficacious overall. Moreover, and more importantly, new, less trivial form of averages could be considered. As an example, a more dynamic average could result from a moving average with the advantage of incorporating some degree of knowledge from parameters past levels. In such a case the 'amount of knowledge' from the past could be fine-tuned assigning weights during the average computation.

Another path of research that could be sought concerns the inclusion of parameters measuring different set of dimensions related to firms and their failure. As mention at the end of chapter 1, macroeconomic and corporate governance indicators could be beneficial for predictions due to the measurement of relevant factors, able to significantly affect the performances of a company, that financial ratios do not capture. This would lead to a clearer background of the specific area being scrutinized (e.g. Veneto region in this study) and consequently to more reliable and accurate outcomes. Another quite known example in this sense entails the use of market based ratios (e.g. prices, trends, etc.). Though undoubtedly useful and valid when applicable, market based parameters carry the critical problem of being essentially useless for any non-quoted enterprise. This should evidently taken into consideration since the vast majority of companies is usually not quoted to exchanges and thus models developed in such a way could find little adoption for small firms.

As last suggestion, prediction frameworks should integrate the formal financial statement analysis, as described in this study, with an overall judgement reached by expert analysts. As a matter of facts this suggestion could be considered as an extension of the previous point. Indeed, it could be well said that a significant portion of soft information that can easily be identified

as crucial to predict bankruptcy remains hidden behind the rigorousness of financial statements. By soft information it is here referred to all knowledge describing non-quantifiable, or hardly and costly so, dimensions that can find examples in type of relationships of intercurrent among employees, or the responsibility and intrinsic ability of the management to find proportionate solution to their companies hurdles, etc. These and similar dimensions are in fact hard to measure but nonetheless crucial and determinant for company success. In this context, the suggestion relates to seeking methodologies to fruitfully integrate and exploit machine and human prediction abilities[41]. This would have the advantage on one side, of filling the gap of soft information parameters with the human ability to recognize at least general behavioural patterns inside companies, while, on the other, to limit human biasedness in assessments, plentifully examined and proved by, as first, Kahneman and Tversky famous *Prospect theory* (1979).

### 4.3 Conclusive comments

In this last section a conclusive comment by the author of the study is proposed. The need for it comes from Ohlson (1980) nipping question on the reasons that should underpin the search for more and more performing prediction models. In part, the reasons adduced in the introduction of this document, already answer said need of reasons. To recap, the main *whys* driving researches on bankruptcy prediction relates to the need to decrease lending institutions operating risk while increasing their overall profits, define sound provisions to guarantee the stability of the modern credit-based financial systems, look for enhanced ways to allocate resources more wisely to productive activities with higher likelihood to succeed, etc.

Cleared the perspective chased by bankruptcy prediction researches, the matter that is here introduced questions whether the approach so far presented and followed is the right one: the one capable to reach the objectives just listed. The approach under assessment, the one applied also in this study, starts from the assumption that with a substantial amount of data, financial data, there could be drawn a picture precise enough to even propose forecasts on newly considered entities. This is carried out in practice by observing large amount of financial statements records, training prediction models and applying those trained models to the new entity being scrutinized. In other words, can this procedure, for how complete and sophisticated a future prediction model might become, find answers to at least one of the goals listed above?

[41] Human prediction abilities are tested and claimed by Zimmer (1980) who, as explained in chapter 1, looked at prediction accuracies achieved by loan officers in executing the task of making annual predictions of corporate failure based on a time series of ratios

I am convinced that this approach can, and will, only partially represent those answers.

The reasons for it is to be sought in the fact that by its own nature statistical models guarantee a level of quality and reliability proportional to the quality of data being deployed. Machine learning practitioners slang it as *GIGO*, which stands for 'garbage in, garbage out'. In the context of bankruptcy prediction, data employed does not resemble 'garbage', though is not always clear whether some financial statement aggregate results from real world quantities or rather the company own interest. Nonetheless, a question arises as to what amount of data, both in terms of absolute number of records available and viewpoints covered[42], should be considered sufficient to ascertain the reliability of results. In the case of bankruptcy prediction, the amount of data needed for models to achieve higher and higher performances needs to be set in accordance with every aspect of a firm life. Thus, in this sense, it is already clear the limitation of the statistical approach so far applied: an important amount of critical data cannot be collected and employed.

It can be pointed out that it is in any case useful have a prediction with the data that can be retrieved, so to reach at least a viewpoint. I disagree with statements of this sort. The reason for it can be found in the brilliant description that professor Taleb depicts in his bestseller *The Black Swan* where he introduces two fictitious countries: Mediocristan and Extremistan. Mediocristan is to be considered as the place where everything averages out, a 'boring' place where outliers do not appear. On the contrary, Extremistan is the country of outliers, where all distributions depart from the Gaussian description of the world. Then, in our picture, prediction models are dealing with some kind of Extremistan. In other terms, the determination of the binary 'failure' 'non-failure' is so dependent to events whose occurrence is inherently not predictable that a prediction based on the average patterns followed by a company resembling, in terms of data points, the one under assessment might even be cause of distortion of more naïve forecasts. These events are related to the business of the company, the macroeconomic and socio-political context surrounding it, the intricate set of human interconnections and abilities, etc. that dominate the life of a firm.

To summarize, the very same presence of *black swans* affecting companies' life makes the bankruptcy prediction research so far conducted only partially effective to materially achieve a decrease in lending risk, more stable financial markets and an effective allocation of resources.

[42] By viewpoints are here intended the fields, relevant for the model to meaningfully operate, the data belongs to. As an example, financial data only covers the aspect of financial results that firms are subjected to, leaving vacant any information on the management team, employees loyalty, etc. which are all relevant factors to predict the firm future.

**Bibliography**

Altman, Edward I. (1968) "Financial ratios, discriminant analysis and the prediction of corporate bankruptcy" *The Journal of Finance* Volume 23 (Issue 4), September 1968: Pages 589-609. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1968.tb00843.x

Altman, Edward I. (1984) "The success of business failure prediction models" *Journal of Banking and Finance,* Volume 8: Pages 171-198.

Barboza, F; Kimura, Herbert; Altman, Edward (2017) "Machine learning models and bankruptcy prediction" *Expert Systems with Applications* Volume 83, 15 October 2017, Pages 405-417

Beaver William H. (1966) "Financial Ratios As Predictors of Failure" *Journal of Accounting Research* Volume 4 Empirical Research in Accounting: Selected Studies 1966 (1966), Pages 71-111. Available at: https://www.jstor.org/stable/2490171?seq=1

Begley, Joy; Ming, Jin; Watts, Susan G. (1996) "Bankruptcy Classification Errors in the 1980s: An Empirical Analysis of Altman's and Ohlson's Models" *Review of Accounting Studies,* Volume 1 (Issue 4), December 1996: Pages 267-284.

Bell, T., G. Ribar and J. Verchio. (1990) "Neural nets versus logistic regression: A comparison of each model's ability to predict commercial bank failures) *Proceedings of the 1990 D&T, University of Kansas Symposium on Auditing Problems*

Bellovary, Jodi L., Giacomino, Don E., and Akers Michael D. (2007) "A Review of Bankruptcy Prediction Studies: 1930 to Present" *Journal of Financial Education* Volume 33 (Issue 1), Winter 2007: Pages 1-42. Available at: https://www.jstor.org/stable/41948574?seq=1

Breiman L, Friedman JH, Olshen RA, Stone CJ. (1984) "Classification and Regression Trees" New York: Chapman and Hall.

Breiman Leo (1996) "Bagging Predictors" *Machine Learning*, Volume 24: Pages 123-140

Breiman, Leo (2001) "Random Forests" *Machine Learning*, Volume 45: Pages 5-32

Bryant, S. (1996) "A case-based reasoning approach to bankruptcy prediction modelling" Ph.D. dissertation, Louisiana State University.

Cadden, D. (1991) "Neural networks and the mathematics of chaos - an investigation of these methodologies as accurate predictions of corporate bankruptcy" *The First International Conference on Artificial Intelligence Applications of Wall Street*. New York: IEEE Computer Society Press.

Casey, C. 1980 "The usefulness of accounting ratios for subjects' predictions of corporate failure: Replication and extensions" *Journal of Accounting Research,* Volume 18(Issue 2): Pages 603-613.

Chen, K.H. and Shimerda, T.A. (1981) "An Empirical Analysis of Useful Financial Ratios" *Financial Management*, Volume 10, Pages 51-60.

Chen, Tianqi; Guestrin Carlos. (2016) "XGBoost: A Scalable Tree Boosting System" *the 22nd ACM SIGKDD International Conference*, Pages 785–794.

Chudson, W. (1945). “The Pattern of Corporate Financial Structure”. New York: National Bureau of Economic Research.

Cunningham Pàdraig and Delany Sarah J. (2007) “k-Nearest Neighbour Classifiers” Technical Report UCD-CSI-2007-4, March 27

Dimitras, A., R. Slowinksi, R. Susmaga and C. Zopounidis. (1999) “Business failure prediction using rough sets” *European Journal of Operational Research*, Volume 114 (Issue 2): Pages 263-280.

Du Jardin, Philippe. (2016) “A two-stage classification technique for bankruptcy prediction” *European Journal of Operational Research* Volume 254 (Issue 1), October 2016: Pages 236-251. Available at: https://www.sciencedirect.com/science/article/pii/S0377221716301369?via%3Dihub

FitzPatrick, P. J. (1932). “A Comparison of the Ratios of Successful Industrial Enterprises with Those of Failed Companies” *The Certified Public Accountant* October 1932: Pages 598-605.

Frydman, H., E. Altman and D. Kao. (1985) “Introducing recursive partitioning for financial classification: The case of financial distress” *The Journal of Finance*, Volume 40(Issue 1): Pages 269-291.

Gepp, Adrian and Kuldeep Kumar (2010). “The Role of Survival Analysis in Financial Distress Prediction” *International Research Journal of Finance and Economics,* Volume 16 (Issue 16).

Hillegeist Stephen A., Keating Elizabeth K., Cram Donald P. and Lundstedt Kyle G. (2003) “Assessing the Probability of Bankruptcy” *Review of Accounting Studies*, Volume 9 : Pages 5–34.

Jackendoff, N. (1962)”A Study of Published Industry Financial and Operating Ratios” Philadelphia: Temple University, Bureau of Economic and Business Research.

Kahneman, D. (2011) “Thinking, fast and slow” New York: Farrar, Straus and Giroux.

Laswad Fauzi; Oyelere Peter B.; Kuruppu Nirosh (2003) “The Efficacy of Liquidation and Bankruptcy Prediction Models for Assessing Going Concern” *Managerial Auditing Journal* Volume 18 (issue 6), August 2003

Liang, Deron, Chia - Chi Lu, Chih - Fong Tsai, and Guan - An Shih. (2016) “Financial Ratios and Corporate Governance Indicators in Bankruptcy Prediction: A Comprehensive Study.” *European Journal of Operational Research* Volume 252 (Issue 2), July 2016: Pages 561–72. Available at: https://www.sciencedirect.com/science/article/pii/S0377221716000412?casa_token=l5vLFvoSZ-AAAAAA:anAYNCG11aev0u5RyKkjveO8S9_lInj68UxriVPMqtk8e8K8yJgKDQ2HZ0jQE5Hq-vu-B1tNEg.

Libby, R. (1975). “Accounting ratios and the prediction of failure: Some behavioural evidence” *Journal of Accounting Research,* Volume 13(Issue 1): Pages 150-16l.

Luoma, M. and E. Laitinen. (1991) “Survival analysis as a tool for company failure Prediction” *Omega*, Volume 19 (Issue 6): Pages 673-678.

Martin, D. 1977. “Early warning of bank failures: A logit regression approach” *Journal of Banking and Finance,* Volume 1: Pages 249-276.

Merwin, C. (1942). “Financing small corporations in five manufacturing industries” New York: National Bureau of Economic Research.

Meyer, P. and H. Pifer. (1970) “Prediction of bank failures” *The Journal of Finance* Volume 25 (Issue 4): Pages 853-868.

Min Jae H., Lee Young-Chan. (2005) “Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters” *Expert Systems with* Applications Volume 28, Issue 4, May 2005, Pages 603-614.

Ohlson, James A. (1980) “Financial Ratios and the Probabilistic Prediction of Bankruptcy” *Journal of Accounting Research,* Volume 18 (Issue 1), Spring 1980: Pages 109-131. Available at: https://www.jstor.org/stable/2490395?seq=1

Peng, C., So, T., Stage, F., & St. John, E. (2002) “The Use and Interpretation of Logistic Regression” *Higher Education Journals*, Volume 43 (Issue 3): Pages 259-293. Available at: http://www.jstor.org/stable/40196455

Pinches, George E., Eubank, Arthur A., Mingo, Kent A., Caruthers, J. Kent, (1975) "The hierarchical classification of financial ratios," *Journal of Business Research, Elsevier*, vol. 3(4), October, pages 295-310.

Ptak-Chmielewska Aneta (2019) “Predicting Micro-Enterprise Failures Using Data Mining Techniques” *Journal of Risk and Financial Management*, Volume 12 (Issue 1): Page 30.

Rokach L. and Maimon O., "Top-down induction of decision trees classifiers - a survey," *IEEE Transactions on Systems, Man, and Cybernetics*, Part C (Applications and Reviews), Volume 35 (Issue 4): Pages 476-487.

Smith, R. and A. Winakor. (1935). “Changes in Financial Structure of Unsuccessful Industrial Corporations” *Bureau of Business Research*, Bulletin No. 51. Urbana: University of Illinois Press.

Taleb, Nassim Nicholas, (2007). “The Black Swan: the Impact of the Highly Improbable” New York :Random House.

Tam, K. and Kiang, M. (1992) “Managerial applications of neural networks - the case of bank failure predictions” *Management Science*, Volume 38 (Issue 7): Pages 926-947.

Vapnick, V. N. (1998). “Statistical learning theory” *John Wiley and Sons Inc*

Wilcox, J. (1973) “A prediction of business failure using accounting data” *Journal of Accounting Research*, Volume 11: Pages 163-179.

Zadeh Lotfi A. (1994) “Soft Computing and Fuzzy Logic” *IEEE Software*, Volume 11 (Issue 6), November 1994: Pages 48-56.

Zhou L., K. K. Lai and J. Yen, (2010) "Bankruptcy Prediction Incorporating Macroeconomic Variables Using Neural Network," *International Conference on Technologies and Applications of Artificial Intelligence*, Hsinchu, pp. 80-85.

**APPENDIX 1**. Python code for Propensity Score Matching procedure

Loading and arrangement section

```
import warnings
warnings.filterwarnings('ignore')
from pymatch.Matcher import Matcher
import pandas as pd
import numpy as np

%matplotlib inline
```

```
# relevant columns names might change from file to file
name_treated = 'treatd Veneto only'
relevant_cols = ['Ragione sociale', 'Partita IVA', 'ATECO 2007 codice', 'Default']
relavant_changing = ['Ricavi vendite e prestazioni migl EUR ', 'TOTALE ATTIVO migl EUR ',
                     'TOTALE PATRIMONIO NETTO migl EUR ']
years_relev = [2009,2010,2011,2012,2013,2014,2015]
for x in years_relev:
    for t in relavant_changing:
        relevant_cols.append(t+'{}'.format(x))
relevant_cols
```

```
#Loading treated file with its column - must have treted in excel file in same folder of this .ipynb document
xl_file_treated = pd.read_excel(name_treated + ".xlsx")[relevant_cols]

#renaming
xl_file_treated = xl_file_treated.rename(columns={'Ragione sociale': 'Rag_soc',
                                                  'Partita IVA': 'P_iva', 'ATECO 2007 codice': 'ATECO_2007'})
for d in years_relev:
    xl_file_treated = xl_file_treated.rename(columns={
        'Ricavi vendite e prestazioni migl EUR {}'.format(d):'ricavi_{}'.format(d),
        'TOTALE ATTIVO migl EUR {}'.format(d): 'Tot_a_{}'.format(d),
        'TOTALE PATRIMONIO NETTO migl EUR {}'.format(d):'Tot_pn_{}'.format(d)})

xl_file_treated.drop_duplicates(subset='P_iva', keep='first', inplace=True)
```

```
#adjusting for relevant default years
xl_file_treated_adjusted = pd.DataFrame()
for a in years_relev:
    meta_reader = xl_file_treated[xl_file_treated.Default == a+4]
    meta_reader2 = meta_reader[['Rag_soc', 'P_iva', 'ATECO_2007', 'Default',
                                'ricavi_{}'.format(a), 'Tot_a_{}'.format(a), 'Tot_pn_{}'.format(a)]]
    xl_file_treated_adjusted = pd.concat([xl_file_treated_adjusted, meta_reader2])
#xl_file_treated_adjusted
```

```
lista_voci_utili = []
for l in years_relev:
    lista_voci_utili.append('Ricavi vendite e prestazioni\nmigl EUR\n{}'.format(l))
    lista_voci_utili.append('TOTALE PATRIMONIO NETTO\nmigl EUR\n{}'.format(l))
    lista_voci_utili.append('TOTALE ATTIVO\nmigl EUR\n{}'.format(l))

questa = ['Ragione sociale','Partita IVA','ATECO 2007\ncodice'] + lista_voci_utili
questa
```

```
#Loading CONTROL file with lits column
reng = list(range(27,71))+list(range(72,101))
xl_file_control= pd.DataFrame()

for v in reng:
    meta_reader = pd.read_excel('control group total/Aida_Export_{}'.format(v) + ".xlsx")[questa]
    xl_file_control = pd.concat([xl_file_control,meta_reader])
    print(v)

#renaming
xl_file_control = xl_file_control.rename(columns={'Ragione sociale': 'Rag_soc',
                                                  'Partita IVA': 'P_iva', 'ATECO 2007\ncodice': 'ATECO_2007'})
for d in years_relev:
    xl_file_control = xl_file_control.rename(columns={
        'Ricavi vendite e prestazioni\nmigl EUR\n{}'.format(d):'ricavi_{}'.format(d),
        'TOTALE ATTIVO\nmigl EUR\n{}'.format(d): 'Tot_a_{}'.format(d),
        'TOTALE PATRIMONIO NETTO\nmigl EUR\n{}'.format(d):'Tot_pn_{}'.format(d)})
xl_file_control.drop_duplicates(subset='P_iva', keep='first', inplace=True)
#creating the Default (=status) column for control group
xl_file_control['Default'] = 0
```

```
#deleting duplicates in the 2 groups
for t in xl_file_treated_adjusted['P_iva']:
    xl_file_control = xl_file_control [xl_file_control['P_iva'] != t]
```

```
#concatening treated_adjusted and control groups
aggregato = pd.concat([xl_file_treated_adjusted,xl_file_control], ignore_index=True)
#aggregato
```

```
aggregati_anni = {}
#dropping n.d. and NaN and creating dictionary
for d in years_relev:
    aggregati_anni['aggregato_{}'.format(d)] = aggregato[['Rag_soc', 'P_iva', 'ATECO_2007',
                                                          'Default', 'ricavi_{}'.format(d),
                                                          'Tot_pn_{}'.format(d), 'Tot_a_{}'.format(d)]]

for s in years_relev:
    aggregati_anni['aggregato_{}'.format(s)].dropna(inplace=True)
    aggregati_anni['aggregato_{}'.format(s)] = aggregati_anni['aggregato_{}'.format(s)]
    [aggregati_anni['aggregato_{}'.format(s)]['ricavi_{}'.format(s)] != 'n.d.']
    aggregati_anni['aggregato_{}'.format(s)] = aggregati_anni['aggregato_{}'.format(s)]
    [aggregati_anni['aggregato_{}'.format(s)]['Tot_pn_{}'.format(s)] != 'n.d.']
    aggregati_anni['aggregato_{}'.format(s)] = aggregati_anni['aggregato_{}'.format(s)]
    [aggregati_anni['aggregato_{}'.format(s)]['Tot_a_{}'.format(s)] != 'n.d.']

#computing e_tot_a

aggregami = aggregati_anni
for s in years_relev:
    aggregami['aggregato_{}'.format(s)]['e_tot_a_{}'.format(s)] = aggregami['aggregato_{}'.format(s)]
    ['Tot_pn_{}'.format(s)]/aggregami['aggregato_{}'.format(s)]['Tot_a_{}'.format(s)]
    aggregami['aggregato_{}'.format(s)]['meta_{}'.format(s)] = aggregami['aggregato_{}'.format(s)]
    ['ATECO_2007'].astype(str).str[0] #for ATECO 1st figure splitting
    aggregami['aggregato_{}'.format(s)] = aggregami['aggregato_{}'.format(s)]
    [['Rag_soc','P_iva', 'ATECO_2007', 'Default', 'ricavi_{}'.format(s), 'e_tot_a_{}'.format(s), 'meta_{}'.format(s)]]
    aggregami['aggregato_{}'.format(s)].reset_index(drop= True, inplace=True)


#aggregami['aggregato_2009']
```

```
#further splitting based on Ateco 1st figure
prime_cifre = [1,2,3,4,5,6,7,8,9]
agg_ateco_year = {}

for z in prime_cifre:
    for d in years_relev:
        agg_ateco_year['agg_{}_{}'.format(z, d)] = aggregami['aggregato_{}'.format(d)]
        [aggregami['aggregato_{}'.format(d)]['meta_{}'.format(d)] == str(z)]
        agg_ateco_year['agg_{}_{}'.format(z, d)].drop(columns=['meta_{}'.format(d)], inplace=True)
        agg_ateco_year['agg_{}_{}'.format(z, d)].reset_index(drop = True, inplace = True)

agg_ateco_year['agg_1_2009']
```

```
#computing proportions defaulted vs controlled
#getting the np.float64 on relevantt data
for g in agg_ateco_year:
    agg_ateco_year[g]['ricavi_{}'.format(int(g[6:]))] = agg_ateco_year[g]
    ['ricavi_{}'.format(int(g[6:]))].astype(np.float64)
    agg_ateco_year[g]['e_tot_a_{}'.format(int(g[6:]))] = agg_ateco_year[g]
    ['e_tot_a_{}'.format(int(g[6:]))].astype(np.float64)
    print(g+':',len(agg_ateco_year[g][agg_ateco_year[g]['Default'] != 0].index), 'on a total of',
          len(agg_ateco_year[g].index) )
```

PyMatch application section

```
name_to_be = 'agg_7&8_2015'
test = agg_ateco_year[name_to_be][agg_ateco_year[name_to_be]['Default'] != 0]
control = agg_ateco_year[name_to_be][agg_ateco_year[name_to_be]['Default'] == 0]
test.Default = 1
control.Default = 0


#checking if it is meaningful to run the psm
if test.shape[0] == 0:
    print('test is empty!')

#applying matcher
m = Matcher(test, control, yvar = "Default", exclude = ["Rag_soc", "P_iva", "ATECO_2007"])

# for reproducibility
np.random.seed(20170926)

m.fit_scores(balance=True, nmodels=2)

m.predict_scores()
m.plot_scores()
```

```
m.tune_threshold(method='random')
```

```
thrs = float(input('0.000'))
m.match(method="min", nmatches=5, threshold=thrs)
```

```
m.record_frequency()
```

```
tobeexported = m.matched_data.sort_values("match_id")
tobeexported.to_excel('PSM_results/'+ name_to_be + '.xlsx', index = False)
tobeexported
```

```
# pvalues from both the KS-test and the grouped permutation of the Chi-Square distance after matching should be > 0.05
cc = m.compare_continuous(return_table=True)
cc
```

Manual Matching section

```
year_used = 14  #last 2 figures
man_name = 'agg_3_2014'
manuale = agg_ateco_year[man_name][agg_ateco_year[man_name]['ATECO_2007'] > 1]
#condition to avoid modifications to original
manuale_treated = manuale [manuale['Default'] != 0]
manuale_control = manuale [manuale['Default'] == 0]

manuale_finale = pd.DataFrame()
#print(manuale_treated.Rag_soc)

for name in manuale_treated.Rag_soc:
    man_treat_only = manuale_treated [manuale_treated['Rag_soc'] == name]
    man_treat_only.reset_index(drop = True, inplace = True)
    scaling_factor = man_treat_only['ricavi_20{}'.format(year_used)][0]/man_treat_only['e_tot_a_20{}'.format(year_used)][0]
    #print(scaling_factor)
    manuale_control ['scores'] = abs(manuale_control
                                     ['ricavi_20{}'.format(year_used)]- man_treat_only
                                     ['ricavi_20{}'.format(year_used)][0])  + (abs(manuale_control
                                    ['e_tot_a_20{}'.format(year_used)] - man_treat_only
                                    ['e_tot_a_20{}'.format(year_used)][0]) * scaling_factor)
    manuale_control.sort_values('scores',inplace=True)
    manuale_fin = pd.concat([man_treat_only, manuale_control])

    manuale_fin = manuale_fin.head(6)
    manuale_finale = pd.concat ([manuale_finale, manuale_fin])

manuale_finale.to_excel('PSM_results/'+'manual_{}'.format(man_name)+'.xlsx', index = False)
manuale_finale
```

## **APPENDIX 2**. Python code for retrieving financial indices

```
import warnings
warnings.filterwarnings('ignore')
from pymatch.Matcher import Matcher
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets

%matplotlib inline
```

```
#ratios are computed on PS Matched items
#loading data ready for ratios.xslx file
ready_for_indeces = pd.read_excel("ready for ratios.xlsx")
```

### Computing components for ratios

```
#adding columns with semi-aggregates (e.g. current assets and liabilities)
relevant_years = [2009,2010,2011,2012,2013,2014,2015,2016,2017,2018]

#current assets
curret_assets_list = ['TOT. DISPON. LIQUIDE migl EUR {}', 'Crediti a breve migl EUR {}',
                      'CREDITI FIN. A BREVE migl EUR {}', 'TOTALE RIMANENZE migl EUR {}']
for year in relevant_years:
    ready_for_indeces['Current Assets {}'.format(year)] = 0
    for d in curret_assets_list:
        ready_for_indeces['Current Assets {}'.format(year)] += ready_for_indeces[d.format(year)].replace('n.d.',0)

#current liabilities
for year in relevant_years:
    ready_for_indeces['Current Liabilities {}'.format(year)] =
        ready_for_indeces['DEBITI A BREVE migl EUR {}'.format(year)].replace('n.d.',0) +
        ready_for_indeces['Obblig.ni entro migl EUR {}'.format(year)].replace('n.d.',0) +
        ready_for_indeces['Obblig.ni convert. entro migl EUR {}'.format(year)].replace('n.d.',0)

#quick assets
for year in relevant_years:
    ready_for_indeces['quick assets {}'.format(year)] =
        ready_for_indeces['Current Assets {}'.format(year)].replace('n.d.',0) -
        ready_for_indeces['TOTALE RIMANENZE migl EUR {}'.format(year)].replace('n.d.',0)

#EBITDA
for year in relevant_years:
    ready_for_indeces['EBITDA {}'.format(year)] =
        ready_for_indeces['RISULTATO OPERATIVO migl EUR {}'.format(year)].replace('n.d.',0) +
        ready_for_indeces['TOT Ammortamenti e svalut. migl EUR {}'.format(year)].replace('n.d.',0)
```

```
#Net Working Capital
for year in relevant_years:
    ready_for_indeces['Net Working Capital {}'.format(year)] =
        ready_for_indeces['Current Assets {}'.format(year)].replace('n.d.',0) -
        ready_for_indeces['Current Liabilities {}'.format(year)].replace('n.d.',0)

relevant_years_delta = [2010,2011,2012,2013,2014,2015,2016,2017,2018]
#Delta NWC
for yeard in relevant_years_delta:
        ready_for_indeces['Delta NWC {}'.format(yeard)] =
        ready_for_indeces['Net Working Capital {}'.format(yeard)].replace('n.d.',0) -
        ready_for_indeces['Net Working Capital {}'.format(yeard-1)].replace('n.d.',0)

#Cash flow from operation
for years in relevant_years_delta:
    ready_for_indeces['Cash Flow Operat {}'.format(years)] =
        ready_for_indeces['EBITDA {}'.format(years)].replace('n.d.',0) -
        ready_for_indeces['Delta NWC {}'.format(years)].replace('n.d.',0)

#Total Liabilities (Passivo-patrimonio netto)
for year in relevant_years:
    ready_for_indeces['Total Liabilities {}'.format(year)] =
        ready_for_indeces['TOTALE PASSIVO migl EUR {}'.format(year)].replace('n.d.',0) -
        ready_for_indeces['TOTALE PATRIMONIO NETTO migl EUR {}'.format(year)].replace('n.d.',0)

ready_for_indeces.reset_index(drop=True, inplace=True)
#ready_for_indeces
```

Computing all 54 financial ratios: The majority of them is initialised in the next three lists: name_ratio, containing the final name given to ratios; First_comp, the list containing the numerator components; Second_comp, the list including all denominators.

```
#Computing Indeces
final_indeces = pd.DataFrame()

indeces = pd.DataFrame()

#List with names of indeces
name_ratio = ['Net Income/Total Assets ', 'Total Debt/Total Assets ', 'Net Income/Net Worth ',
              'Total Liabilities/Total Assets ', 'Inventory/Sales ',
             'Operating Income/Total Assets ', 'Net Income/Sales ', 'Long-term debt/Total Assets ',
              'Total liabilities/net worth ', 'Operating expenses/Operating income ',
              'Current Ratio ', 'Working Capital/Total Assets ', 'Retained earnings/Total assets ',
              'Current Assets/Total Assets ',
             'Current Liabilities/Total Assets ', 'Current Assets/Sales ', 'Working Capital/Net worth ',
              'quick ratio (quick ass/current liab) ', 'Sales/Total assets ',
             'quick assets/Total assets ', 'quick assets/Sales ', 'EBIT/Total assets ', 'EBIT/Interest ',
              'Working capital/Sales ',
             'CFO/Total assets ', 'CFO/Total debt ', 'CFO/Sales ', 'CFO/Current Liabilities ', 'CFO/Total liabilities ',
             'Cash/Total Assets ', 'Net Worth/Total Assets ', 'Total Debt/Net Worth ', 'Cash/Current Liabilities ',
              'Net Worth/Total liabilities ', 'no-credit interval (Curr Ass/Daily OPEX) ',
             'Asset Turnover ', 'Return on Total Asset ', 'Ebitda/EBIT ', 'CFO/EBIT ', 'Tax Expenses/EBIT ',
             'Other Revenues/Total Produced Value ', 'Cash Flow ratio ', 'Interest Coverage ']
```

```
#first component of the ratio
first_comp = ['UTILE/PERDITA DI ESERCIZIO migl EUR ', 'TOTALE DEBITI migl EUR ', 'Utile/perdita di esercizio migl EUR ',
              'Total Liabilities ', 'Var. rimanenze prodotti migl EUR ',
             'RISULTATO OPERATIVO migl EUR ', 'Utile/perdita di esercizio migl EUR ',
              "Total debiti oltre l'esercizio migl EUR ", 'TOTALE PASSIVO migl EUR ',
              'COSTI DELLA PRODUZIONE migl EUR ',
              'Current Assets ', 'Net Working Capital ', 'Utile/perdita a nuovo migl EUR ',
              'Current Assets ',
             'Current Liabilities ', 'Current Assets ', 'Net Working Capital ', 'quick assets ',
              'Ricavi vendite e prestazioni migl EUR ',
             'quick assets ', 'quick assets ', 'RISULTATO OPERATIVO migl EUR ', 'RISULTATO OPERATIVO migl EUR ',
              'Net Working Capital ',
             'Cash Flow Operat ', 'Cash Flow Operat ', 'Cash Flow Operat ', 'Cash Flow Operat ', 'Cash Flow Operat ',
             'TOT. DISPON. LIQUIDE migl EUR ', 'TOTALE PATRIMONIO NETTO migl EUR ',
              'TOTALE DEBITI migl EUR ', 'TOT. DISPON. LIQUIDE migl EUR ', 'TOTALE PATRIMONIO NETTO migl EUR ',
              'Current Assets ',
             'Ricavi vendite e prestazioni migl EUR ', 'RISULTATO OPERATIVO migl EUR ', 'EBITDA ', 'Cash Flow Operat ',
              'Totale Imposte sul reddito correnti, differite e anticipate migl EUR ',
             'Altri ricavi migl EUR ', 'Cash Flow Operat ', 'EBITDA ']
```

```
#2nd component of the ratio
second_comp = ['TOTALE ATTIVO migl EUR ', 'TOTALE ATTIVO migl EUR ', 'TOTALE PATRIMONIO NETTO migl EUR ',
               'TOTALE ATTIVO migl EUR ', 'Ricavi vendite e prestazioni migl EUR ',
              'TOTALE ATTIVO migl EUR ', 'Ricavi vendite e prestazioni migl EUR ', 'TOTALE ATTIVO migl EUR ',
               'TOTALE PATRIMONIO NETTO migl EUR ', 'RISULTATO OPERATIVO migl EUR ',
               'Current Liabilities ', 'TOTALE ATTIVO migl EUR ', 'TOTALE ATTIVO migl EUR ', 'TOTALE ATTIVO migl EUR ',
              'TOTALE ATTIVO migl EUR ', 'Ricavi vendite e prestazioni migl EUR ', 'TOTALE PATRIMONIO NETTO migl EUR ',
               'Current Liabilities ', 'TOTALE ATTIVO migl EUR ',
              'TOTALE ATTIVO migl EUR ', 'Ricavi vendite e prestazioni migl EUR ', 'TOTALE ATTIVO migl EUR ',
               'TOTALE PROVENTI E ONERI FINANZIARI migl EUR ', 'Ricavi vendite e prestazioni migl EUR ',
              'TOTALE ATTIVO migl EUR ', 'TOTALE DEBITI migl EUR ', 'Ricavi vendite e prestazioni migl EUR ',
               'Current Liabilities ', 'Total Liabilities ',
              'TOTALE ATTIVO migl EUR ', 'TOTALE ATTIVO migl EUR ', 'TOTALE PATRIMONIO NETTO migl EUR ',
               'Current Liabilities ', 'Total Liabilities ', 'COSTI DELLA PRODUZIONE migl EUR ',
              'TOTALE ATTIVO migl EUR ', 'TOTALE ATTIVO migl EUR ', 'RISULTATO OPERATIVO migl EUR ',
               'RISULTATO OPERATIVO migl EUR ', 'RISULTATO OPERATIVO migl EUR ',
              'TOT. VAL. DELLA PRODUZIONE migl EUR ', 'Current Liabilities ',
               'TOTALE PROVENTI E ONERI FINANZIARI migl EUR ']
```

The computation structure and the other, more complex indices (from 'inserting "single" ratios')

```
#check on grops lenght
print(len(name_ratio)==len(first_comp), len(first_comp)==len(second_comp))

for x in list(range (0,len(name_ratio))):
    if name_ratio[x] not in ['CFO/Total assets ', 'CFO/Total debt ', 'CFO/Sales ', 'CFO/Current Liabilities ',
                             'CFO/Total liabilities ', 'CFO/EBIT ', 'Cash Flow ratio ']:
        for year in relevant_years:
            indeces[name_ratio[x]+'{}'.format(year)] =
                ready_for_indeces[first_comp[x]+'{}'.format(year)].replace('n.d.',0)/
                ready_for_indeces[second_comp[x]+'{}'.format(year)].replace('n.d.',0)
            final_indeces = pd.concat([final_indeces,indeces[name_ratio[x]+'{}'.format(year)]], axis=1)
    else:
        for year in relevant_years_delta:
            indeces[name_ratio[x]+'{}'.format(year)] =
                ready_for_indeces[first_comp[x]+'{}'.format(year)].replace('n.d.',0)/
                ready_for_indeces[second_comp[x]+'{}'.format(year)].replace('n.d.',0)
            final_indeces = pd.concat([final_indeces,indeces[name_ratio[x]+'{}'.format(year)]], axis=1)

#inserting 'single' ratios
#1.CFO
for years in relevant_years_delta:
    final_indeces['Cash Flow Operat {}'.format(years)] = ready_for_indeces['Cash Flow Operat {}'.format(years)]

#2.log total assets
for year in relevant_years:
    final_indeces['log(Total Assets) {}'.format(year)] =
        np.log(ready_for_indeces['TOTALE ATTIVO migl EUR {}'.format(year)].replace('n.d.',0))
```

```
#COGS (not relevant index)
for year in relevant_years:
    ready_for_indeces['COGS {}'.format(year)] =
        ready_for_indeces['Materie prime e consumo migl EUR {}'.format(year)].replace('n.d.',0) +
        ready_for_indeces['Servizi migl EUR {}'.format(year)].replace('n.d.',0) +
        ready_for_indeces['Godimento beni di terzi migl EUR {}'.format(year)].replace('n.d.',0)

#3.Turnover Payables
for year in relevant_years:
    final_indeces['Turnover Payables {}'.format(year)] =
        ready_for_indeces['COGS {}'.format(year)].replace('n.d.',0) * 1.22 /
        (ready_for_indeces['Fornitori entro migl EUR {}'.format(year)].replace('n.d.',0) +
         ready_for_indeces['Fornitori oltre migl EUR {}'.format(year)].replace('n.d.',0))

#4.Turnover Receivables
for year in relevant_years:
    final_indeces['Turnover Receivables {}'.format(year)] =
        ready_for_indeces['Ricavi vendite e prestazioni migl EUR {}'.format(year)].replace('n.d.',0) * 1.22 /
        (ready_for_indeces['Cred. vs Clienti entro migl EUR {}'.format(year)].replace('n.d.',0) +
         ready_for_indeces['Cred. vs Clienti oltre migl EUR {}'.format(year)].replace('n.d.',0))

#5.Turnover Inventory
for year in relevant_years:
    final_indeces['Turnover Inventory {}'.format(year)] =
    (ready_for_indeces['COSTI DELLA PRODUZIONE migl EUR {}'.format(year)].replace('n.d.',0) -
     ready_for_indeces['Incrementi di immob. migl EUR {}'.format(year)].replace('n.d.',0)) /
    ready_for_indeces['TOTALE RIMANENZE migl EUR {}'.format(year)].replace('n.d.',0)
```

```
#6.Acid Ratio
for year in relevant_years:
    final_indeces['Acid Ratio {}'.format(year)] =
    (ready_for_indeces['ATTIVO CIRCOLANTE migl EUR {}'.format(year)].replace('n.d.',0) -
     ready_for_indeces['TOTALE RIMANENZE migl EUR {}'.format(year)].replace('n.d.',0)) /
    ready_for_indeces['Current Liabilities {}'.format(year)].replace('n.d.',0)

#Total Customer Receivables (not relevant index)
for year in relevant_years:
    ready_for_indeces['Total Customer Receivables {}'.format(year)] =
    ready_for_indeces['Cred. vs Clienti entro migl EUR {}'.format(year)].replace('n.d.',0) +
    ready_for_indeces['Cred. vs Clienti oltre migl EUR {}'.format(year)].replace('n.d.',0)

#Delta Customer Receivables (not relevant index)
for years in relevant_years_delta:
    ready_for_indeces['Delta Customer Receivables {}'.format(years)] =
    ready_for_indeces['Total Customer Receivables {}'.format(years)].replace('n.d.',0) -
    ready_for_indeces['Total Customer Receivables {}'.format(years-1)].replace('n.d.',0)
```

```
#cash from sales (not relevant index)
for years in relevant_years_delta:
    ready_for_indeces['cash from sales {}'.format(years)] =
    ready_for_indeces['Ricavi vendite e prestazioni migl EUR {}'.format(years)].replace('n.d.',0) -
    ready_for_indeces['Delta Customer Receivables {}'.format(years)].replace('n.d.',0)

#7.Net sales/Cash from sales
for years in relevant_years_delta:
    final_indeces['Net sales/Cash from sales {}'.format(years)] =
    ready_for_indeces['Ricavi vendite e prestazioni migl EUR {}'.format(years)].replace('n.d.',0) /
    ready_for_indeces['cash from sales {}'.format(years)].replace('n.d.',0)

#8.Net Sales/NAR
for year in relevant_years:
    final_indeces['Sales/NAR {}'.format(year)] =
    ready_for_indeces['Ricavi vendite e prestazioni migl EUR {}'.format(year)].replace('n.d.',0) /
    (ready_for_indeces['Total Customer Receivables {}'.format(year)].replace('n.d.',0) -
     ready_for_indeces['Svalut. crediti migl EUR {}'.format(year)].replace('n.d.',0))

#9.CFO/Financial Debt
for year in relevant_years_delta:
    final_indeces['CFO/Financial Debt {}'.format(year)] =
    ready_for_indeces['Cash Flow Operat {}'.format(year)].replace('n.d.',0) /
    (ready_for_indeces['Banche entro migl EUR {}'.format(year)].replace('n.d.',0) +
     ready_for_indeces['Banche a lungo migl EUR {}'.format(year)].replace('n.d.',0) +
     ready_for_indeces['Altri finanziatori entro migl EUR {}'.format(year)].replace('n.d.',0) +
     ready_for_indeces['Altri finanziatori oltre migl EUR {}'.format(year)].replace('n.d.',0))
```

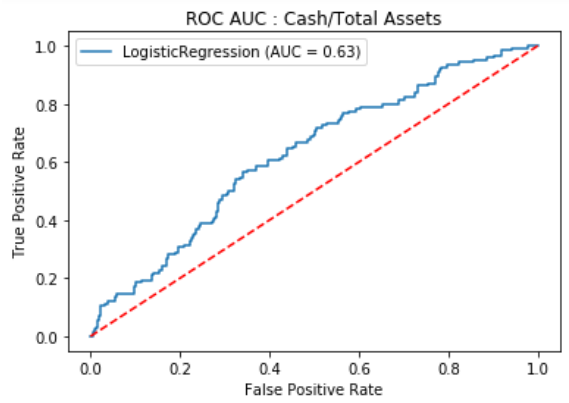
```
#Principal (not relevant index)
principal_set = ['Obblig.ni entro migl EUR {}', 'Obblig.ni oltre migl EUR {}', 'Soci per Finanziamenti entro migl EUR {}',
                 'Soci per Finanziamenti oltre migl EUR {}', 'Banche entro migl EUR {}', 'Banche a lungo migl EUR {}',
                  'Altri finanziatori entro migl EUR {}',
                 'Altri finanziatori oltre migl EUR {}', 'Titoli di credito entro migl EUR {}',
                  'Titoli di credito oltre migl EUR {}']
for year in relevant_years:
    for d in principal_set:
        ready_for_indeces['Principal {}'.format(year)] = 0
        ready_for_indeces['Principal {}'.format(year)] +=  ready_for_indeces[d.format(year)].replace('n.d.',0)

#Delta Principal (not relevant index)
for year in relevant_years_delta:
    ready_for_indeces['Delta Principal {}'.format(year)] =
    ready_for_indeces['Principal {}'.format(year)].replace('n.d.',0) -
    ready_for_indeces['Principal {}'.format(year-1)].replace('n.d.',0)

#10.Fixed Charges Cash Coverage
for year in relevant_years_delta:
    final_indeces['Fixed Charges Cash Coverage {}'.format(year)] =
    (ready_for_indeces['Delta Principal {}'.format(year)].replace('n.d.',0) +
     ready_for_indeces['Totale Oneri finanziari migl EUR {}'.format(year)].replace('n.d.',0) +
     final_indeces['Cash Flow Operat {}'.format(year)].replace('n.d.',0)) /
    ready_for_indeces['Current Liabilities {}'.format(year)].replace('n.d.',0)

#11.Fixed Charges EBIT Coverage
for year in relevant_years_delta:
    final_indeces['Fixed Charges EBIT Coverage {}'.format(year)] =
    (ready_for_indeces['Delta Principal {}'.format(year)].replace('n.d.',0) +
     ready_for_indeces['Totale Oneri finanziari migl EUR {}'.format(year)].replace('n.d.',0) + r
     eady_for_indeces['RISULTATO OPERATIVO migl EUR {}'.format(year)].replace('n.d.',0)) /
    ready_for_indeces['Current Liabilities {}'.format(year)].replace('n.d.',0)
```

```
#taking defaulted out for correlation purposes
final_indeces['Default'] = ready_for_indeces['Default']
final_indeces['Partita IVA'] = ready_for_indeces['Partita IVA']

#creating finall_all for future usage
final_all = final_indeces[list(final_indeces)]

final_indeces = final_indeces[final_indeces['Default'] == 0] #considering only non-defaulting
final_indeces.drop_duplicates(subset=['Partita IVA']) #deleting duplicates
final_indeces.drop('Partita IVA', axis=1, inplace=True)
final_indeces.drop('Default', axis=1, inplace=True)
final_indeces.reset_index(drop=True, inplace=True)
#final_indeces
```

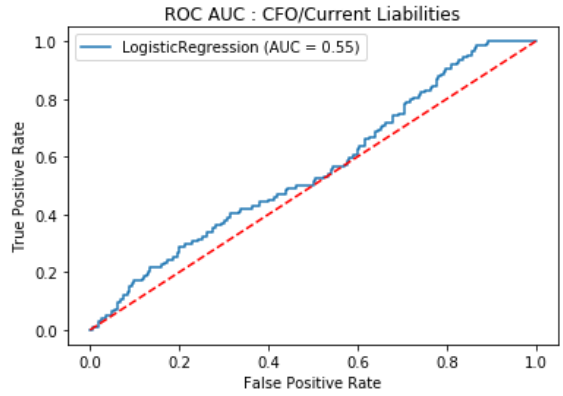
```
aggreg = 0
for x in list(final_indeces):
    #if final_indeces[x].isna().sum()!= 0:
        #print (x,final_indeces[x].isna().sum())
    aggreg += final_indeces[x].isna().sum()
#aggreg
```

**APPENDIX 3.** Univariate logistic regression results

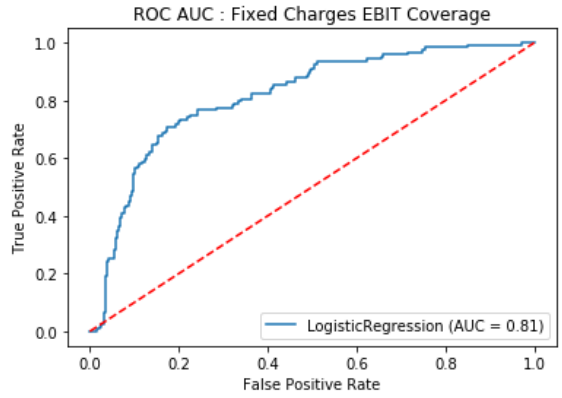
First are reported ROC AUC and confusion matrices; then, Recall, Precision and Accuracy; Finally, python code is transcribed.



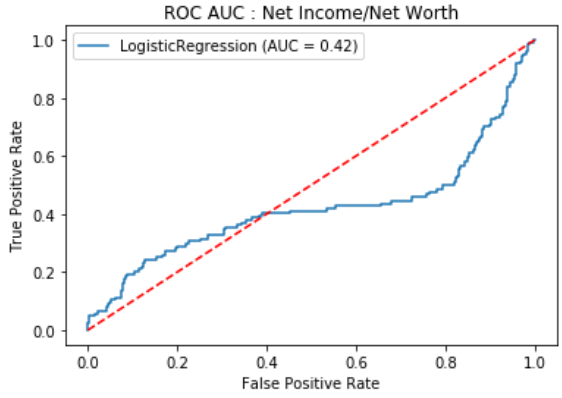

```
[[368 189]
 [ 54  70]]
```



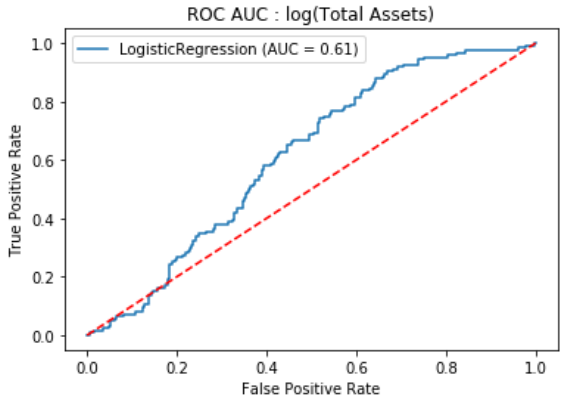

```
[[ 75 482]
 [  2 122]]
```



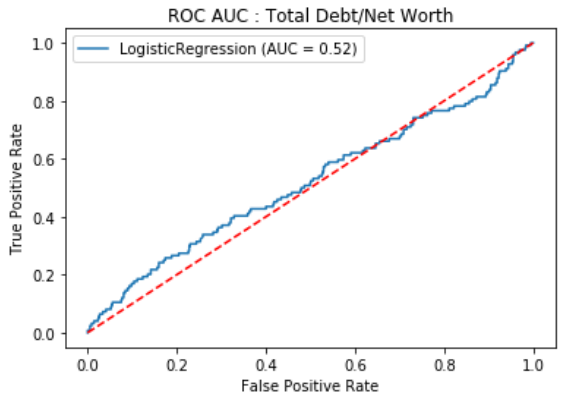

```
[[460  97]
 [ 36  88]]
```



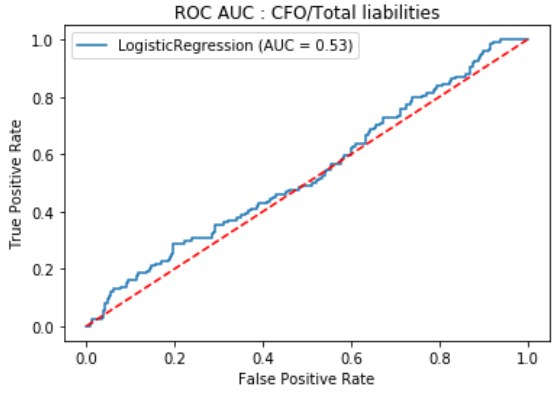

```
[[486  71]
 [ 94  30]]
```



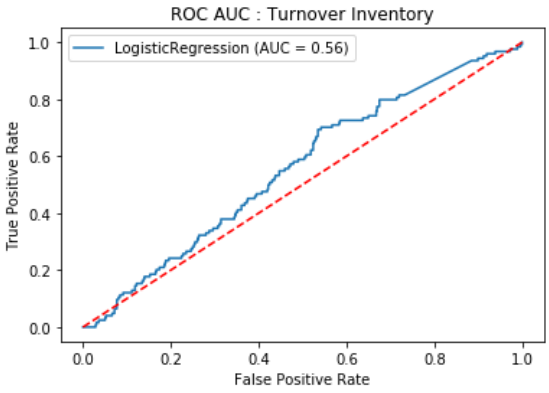

```
[[199 358]
 [ 15 109]]
```



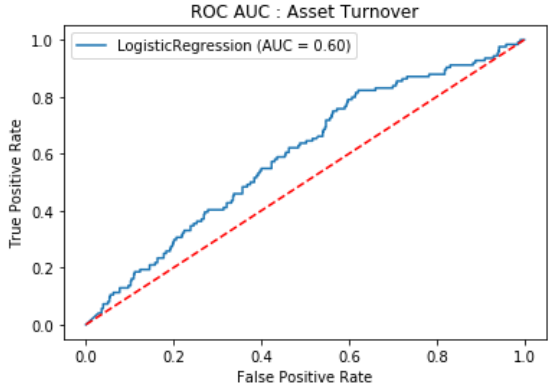

```
[[460  97]
 [ 92  32]]
```



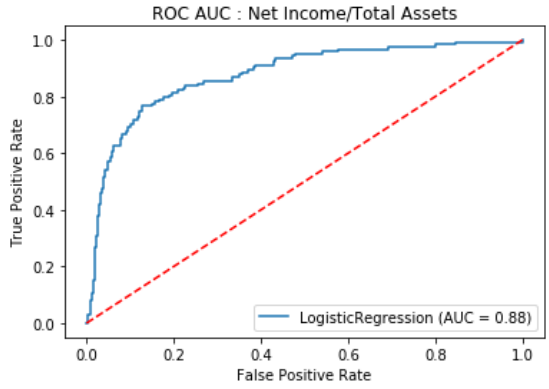

```
[[447 110]
 [ 88  36]]
```



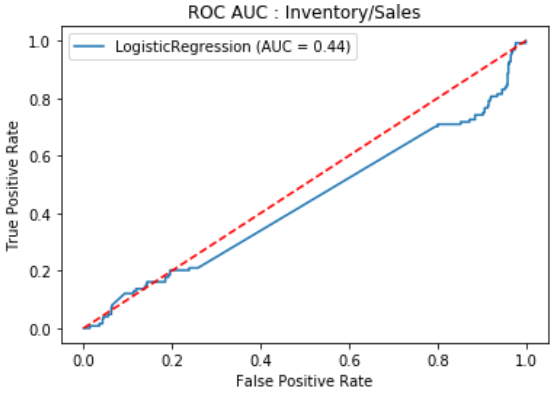

```
[[255 302]
 [ 37  87]]
```



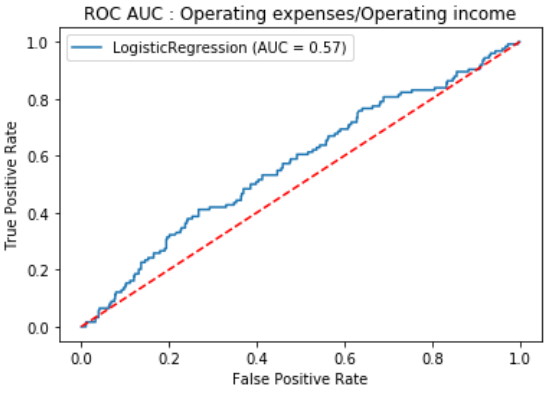

```
[[211 346]
 [ 22 102]]
```



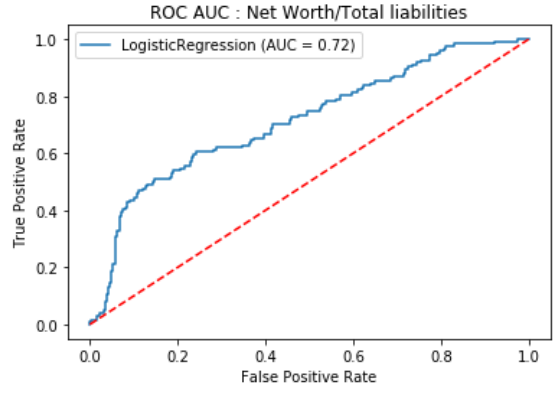

```
[[487  70]
 [ 29  95]]
```



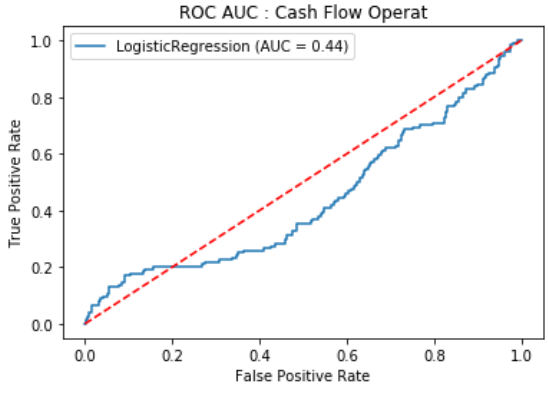

```
[[505  52]
 [109  15]]
```



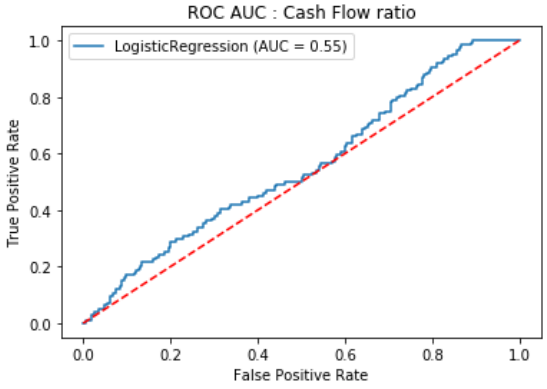

```
[[408 149]
 [ 73  51]]
```



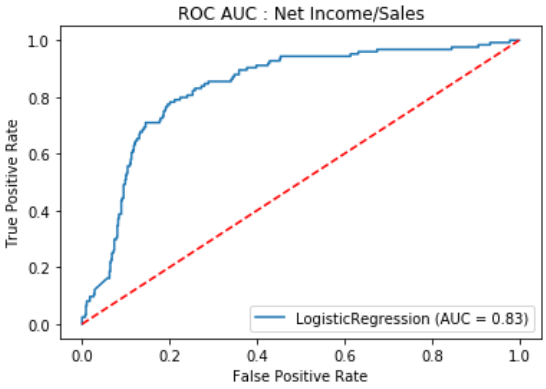

```
[[423 134]
 [ 49  75]]
```



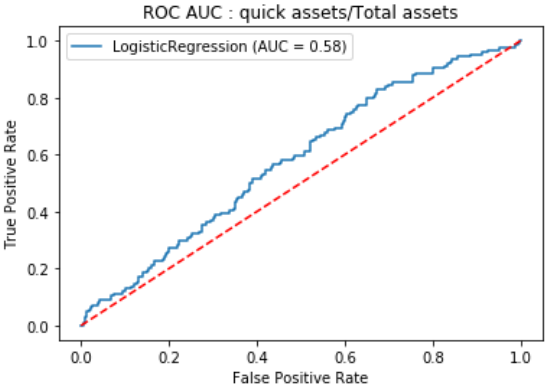

```
[[507  50]
 [103  21]]
```



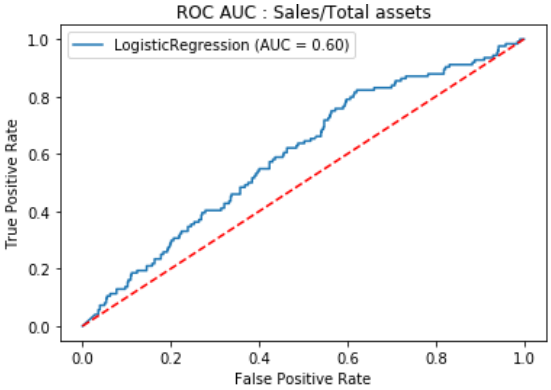

```
[[ 75 482]
 [  2 122]]
```



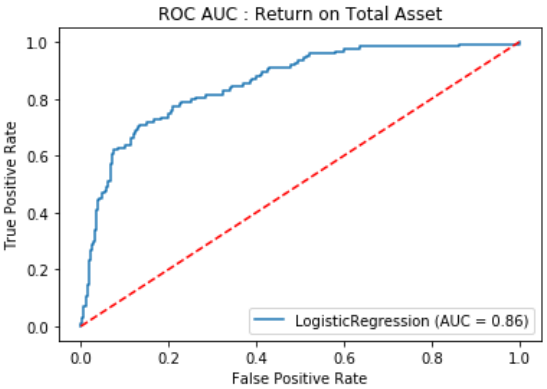

```
[[445 112]
 [ 27  97]]
```

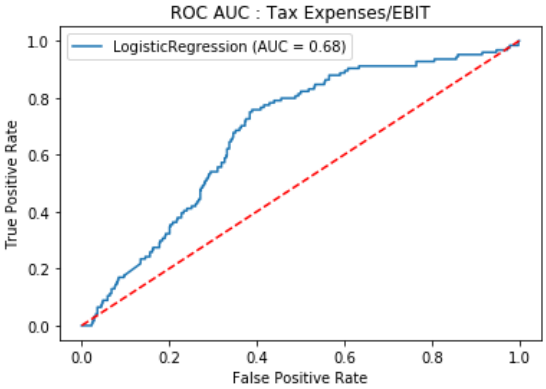


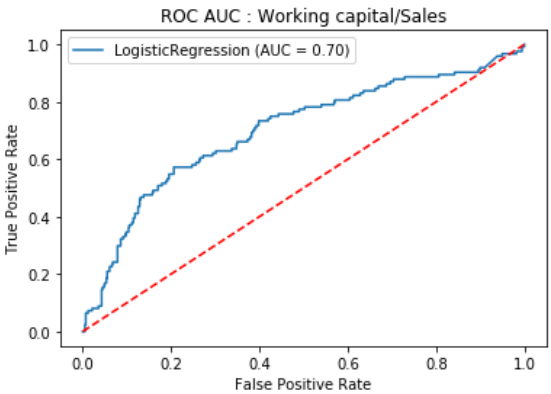

```
[[182 375]
 [ 21 103]]
```



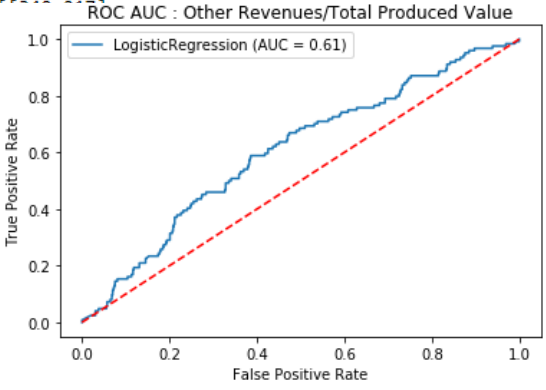

```
[[211 346]
 [ 22 102]]
```



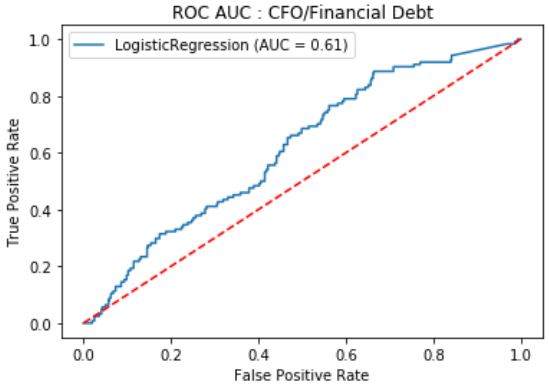



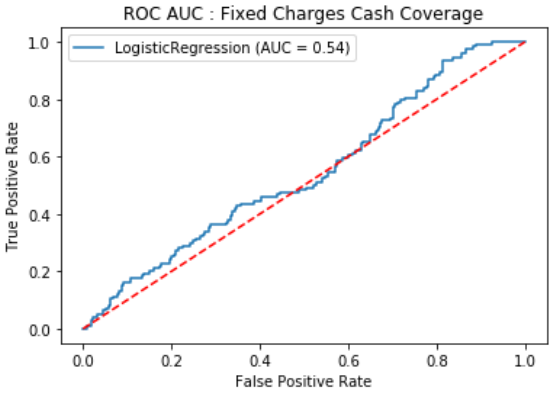

```
[[442 115]
 [ 53  71]]
```



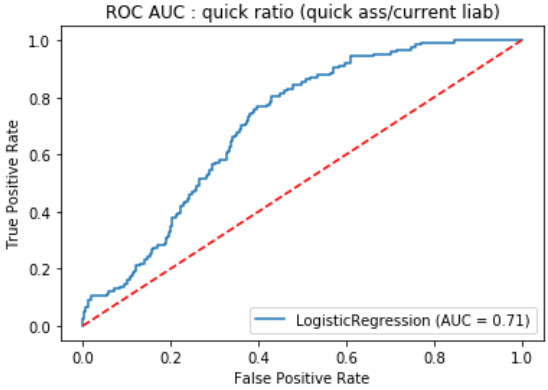

```
[[342 215]
 [ 51  73]]
```



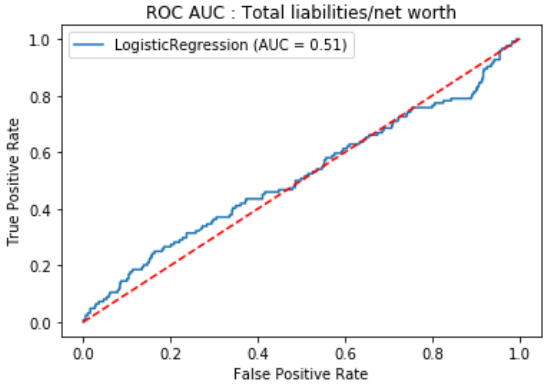

```
[[187 370]
 [ 14 110]]
```



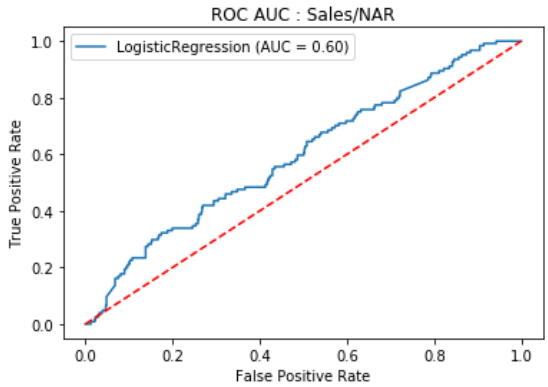

```
[[103 454]
 [  8 116]]
```



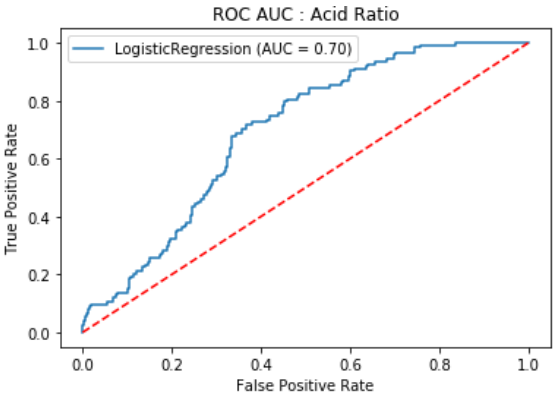

```
[[318 239]
 [ 24 100]]
```



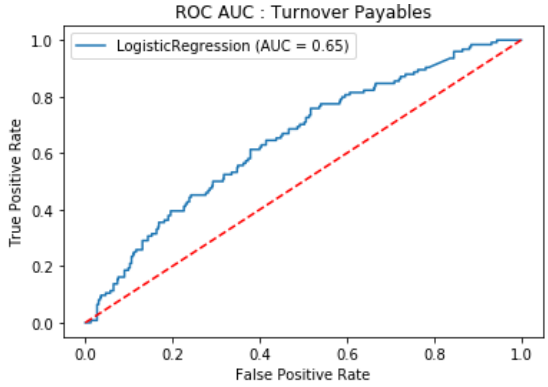

```
[[466  91]
 [ 93  31]]
```



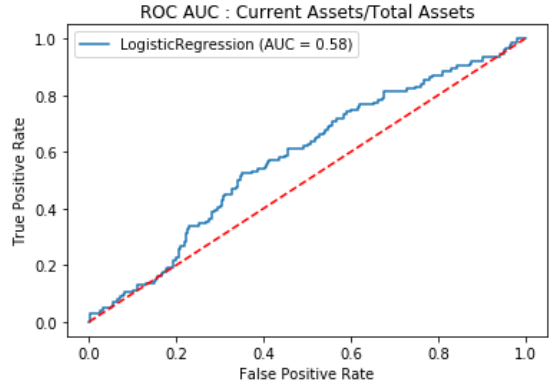

```
[[407 150]
 [ 72  52]]
```



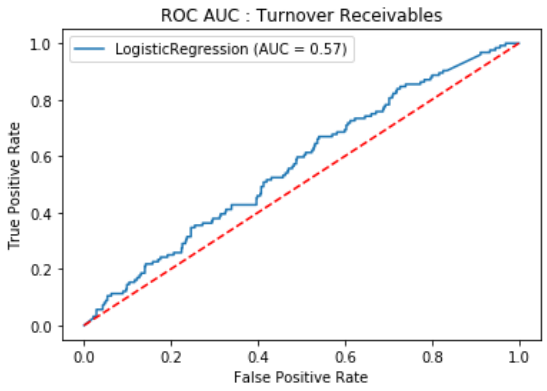

```
[[353 204]
 [ 35  89]]
```



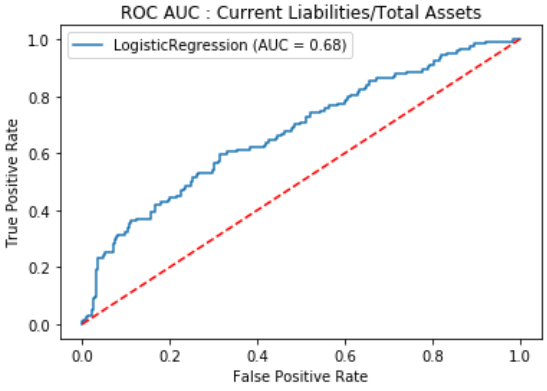

```
[[269 288]
 [ 30  94]]
```



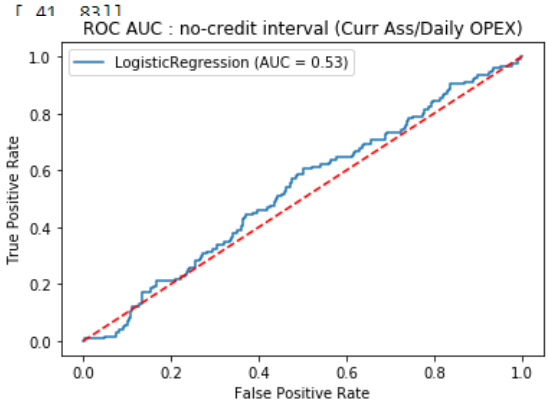

```
[[361 196]
 [ 59  65]]
```



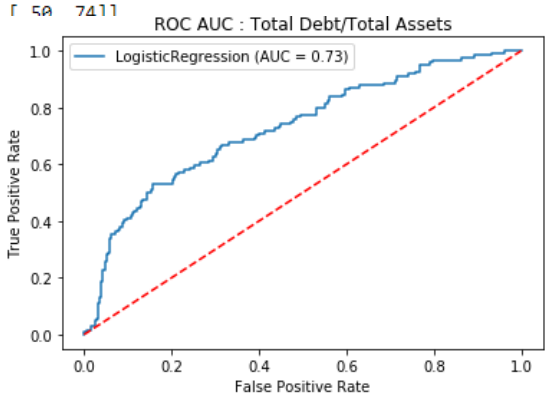

[[256 301]
 [ 41 83]]



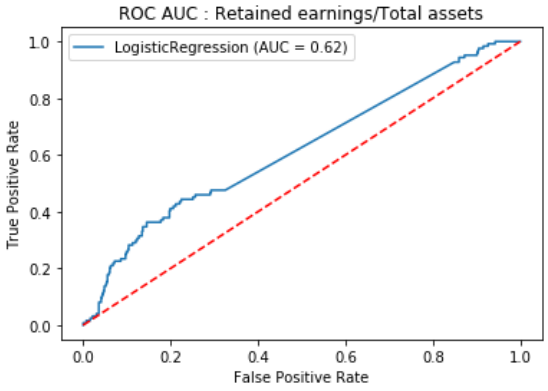

[[382 175]
 [ 50 74]]



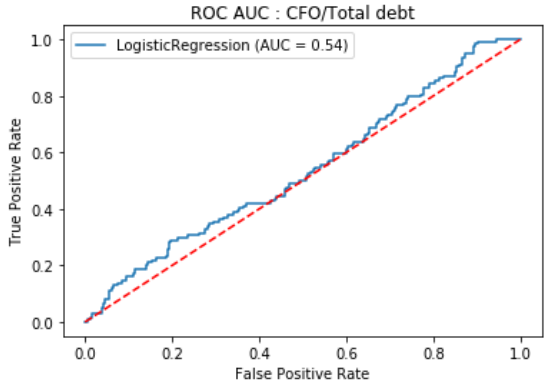

[[277 280]
 [ 49 75]]



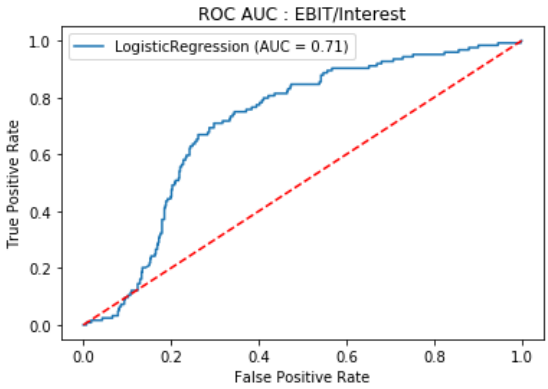

[[469 88]
 [ 58 66]]



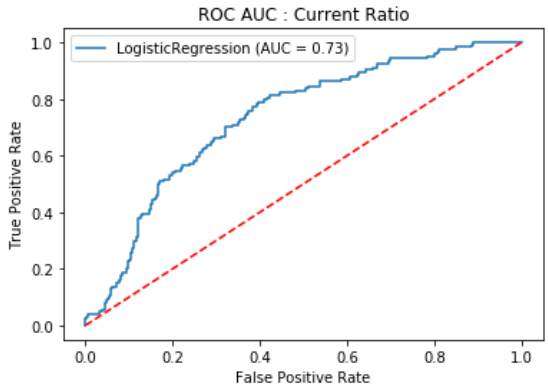

[[432 125]
 [ 69 55]]



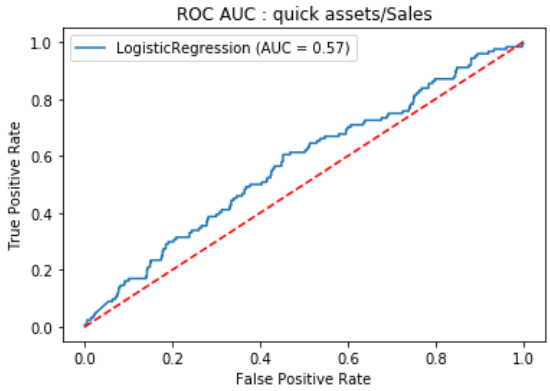

[[448 109]
 [ 88 36]]



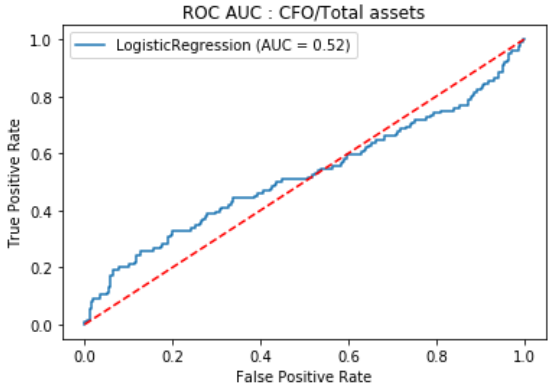

[[391 166]
 [ 36 88]]



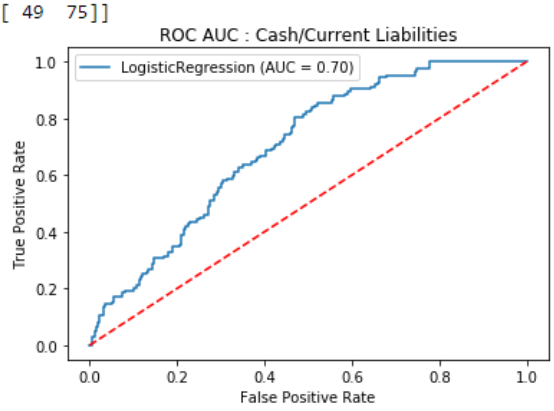

[[337 220]
 [ 26 98]]



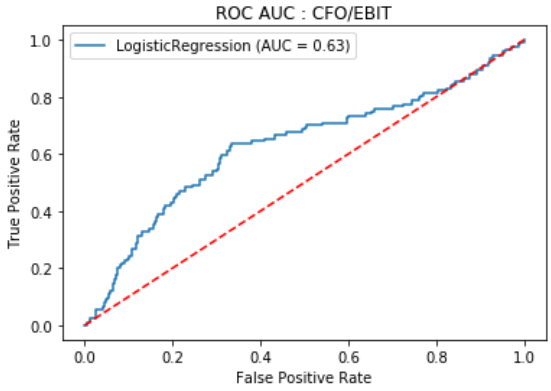

```
[[305 252]
 [ 49  75]]
```



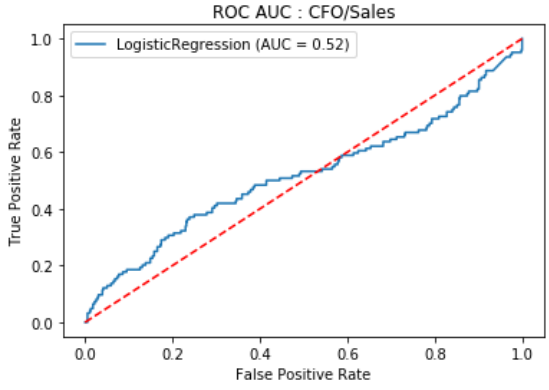

```
[[487  70]
 [ 92  32]]
```



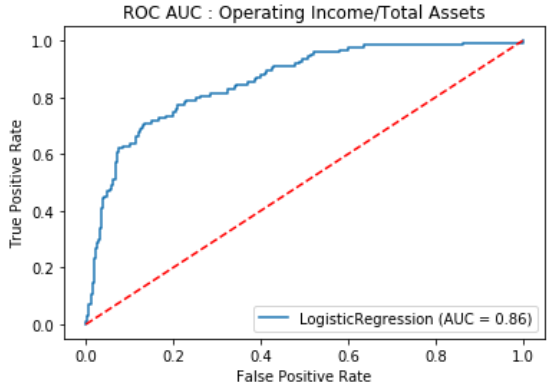

```
[[295 262]
 [ 24 100]]
```



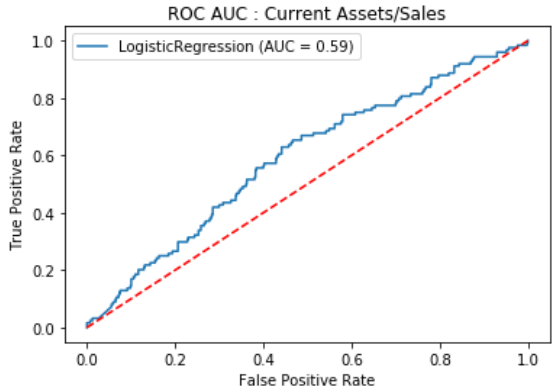

```
[[370 187]
 [ 45  79]]
```



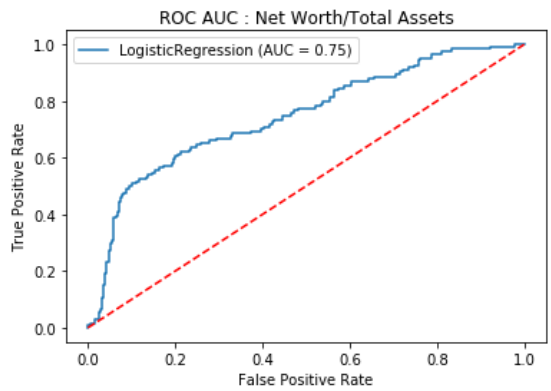

```
[[424 133]
 [ 78  46]]
```



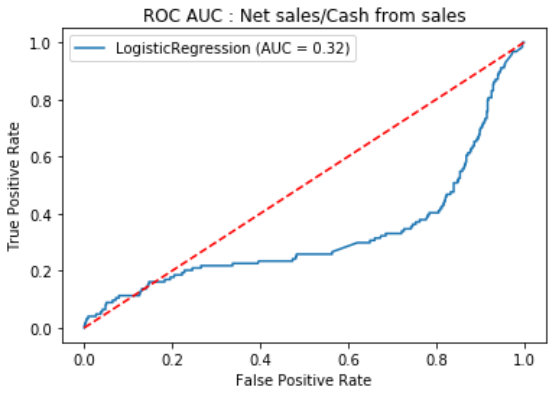

```
[[483  74]
 [ 36  88]]
```



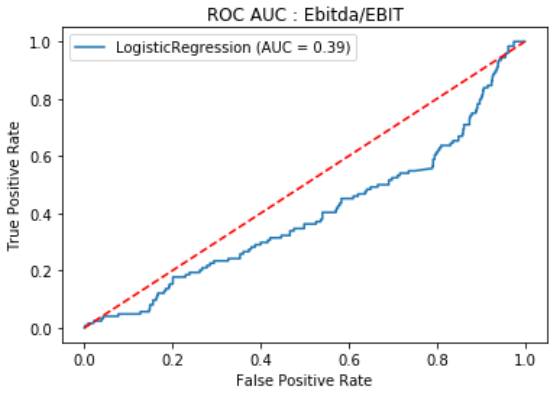

```
[[311 246]
 [ 46  78]]
```



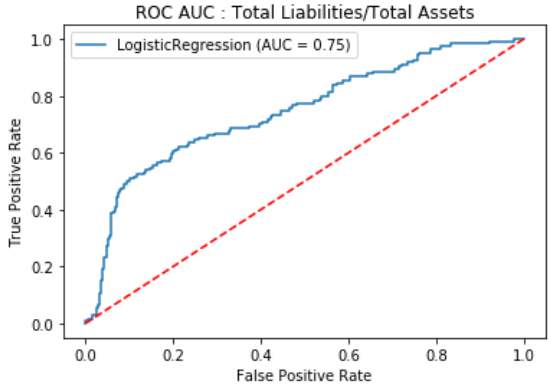

```
[[439 118]
 [ 47  77]]
```



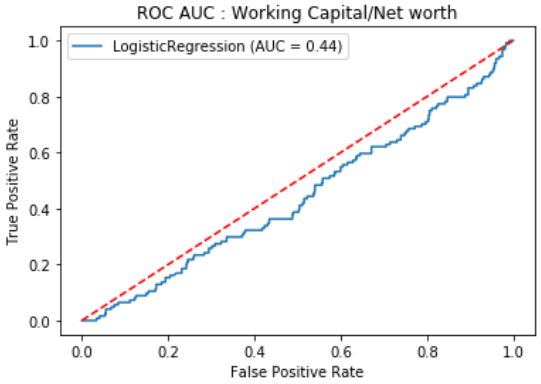

```
[[529  28]
 [113  11]]
```



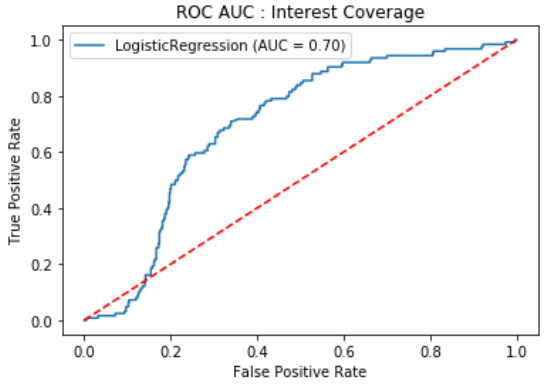

```
[[ 14 543]
 [  0 124]]
```



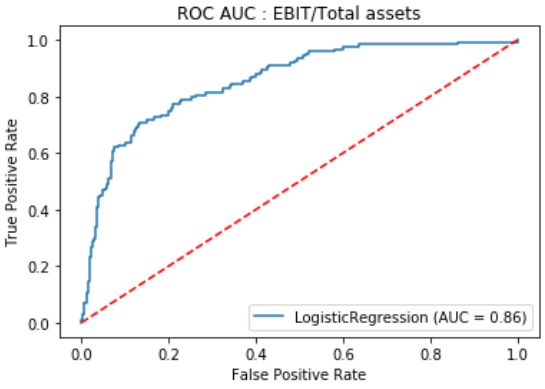

```
[[439 118]
 [ 47  77]]
```



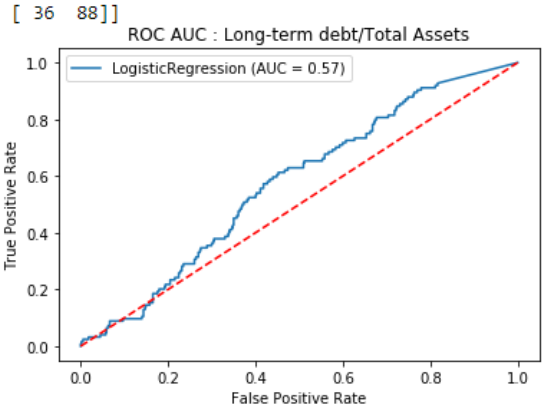

```
[[  5 552]
 [  0 124]]
```



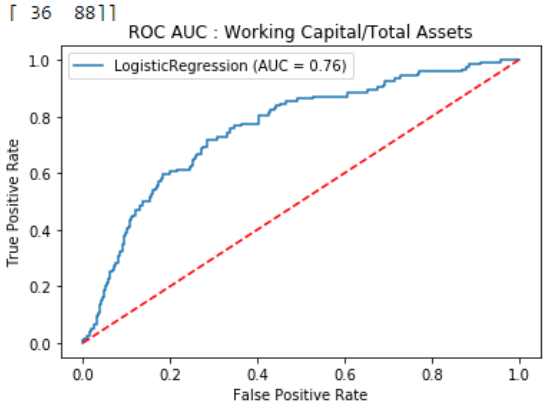

```
[[368 189]
 [ 36  88]]
```




```
[[483  74]
 [ 36  88]]
```




```
[[305 252]
 [ 48  76]]
```




```
[[399 158]
 [ 35  89]]
```

| | Macro average Precision | Macro average recall | Accuracy |
|---|---|---|---|
| Cash/Total Assets | 0.571154 | 0.612599 | 0.643172 |
| CFO/Current Liabilities | 0.588006 | 0.559260 | 0.289280 |
| Fixed Charges EBIT Coverage | 0.701548 | 0.767765 | 0.804699 |
| Net Income/Net Worth | 0.567480 | 0.557233 | 0.757709 |
| log(Total Assets) | 0.581656 | 0.618152 | 0.452276 |
| Total Debt/Net Worth | 0.540698 | 0.541959 | 0.722467 |
| CFO/Total liabilities | 0.541045 | 0.546418 | 0.709251 |
| Turnover Inventory | 0.548469 | 0.579711 | 0.502203 |
| Asset Turnover | 0.566629 | 0.600698 | 0.459618 |
| Net Income/Total Assets | 0.759778 | 0.820228 | 0.854626 |
| Inventory/Sales | 0.523178 | 0.513805 | 0.763583 |
| Operating expenses/Operating income | 0.551616 | 0.571893 | 0.674009 |
| Net Worth/Total liabilities | 0.627519 | 0.682132 | 0.731278 |
| Cash Flow Operat | 0.563461 | 0.539794 | 0.775330 |
| Cash Flow ratio | 0.588006 | 0.559260 | 0.289280 |
| Net Income/Sales | 0.703456 | 0.790590 | 0.795888 |
| quick assets/Total assets | 0.556016 | 0.578698 | 0.418502 |
| Sales/Total assets | 0.566629 | 0.600698 | 0.459618 |
| Return on Total Asset | 0.736923 | 0.788411 | 0.838473 |
| Tax Expenses/EBIT | 0.610585 | 0.684239 | 0.637298 |
| Working capital/Sales | 0.637325 | 0.683059 | 0.753304 |
| Other Revenues/Total Produced Value | 0.561851 | 0.601357 | 0.609398 |
| CFO/Financial Debt | 0.579757 | 0.611412 | 0.436123 |
| Fixed Charges Cash Coverage | 0.565718 | 0.560202 | 0.321586 |
| quick ratio (quick ass/current liab) | 0.612405 | 0.688684 | 0.613803 |
| Total liabilities/net worth | 0.543865 | 0.543312 | 0.729809 |
| Sales/NAR | 0.553556 | 0.575028 | 0.674009 |
| Acid Ratio | 0.606774 | 0.675747 | 0.649046 |
| Turnover Receivables | 0.539049 | 0.564480 | 0.497797 |
| Current Liabilities/Total Assets | 0.590724 | 0.641296 | 0.669604 |
| no-credit interval (Curr Ass/Daily OPEX) | 0.530480 | 0.551073 | 0.516887 |
| Total Debt/Total Assets | 0.659257 | 0.687134 | 0.785609 |
| Retained earnings/Total assets | 0.583916 | 0.609566 | 0.715125 |
| CFO/Total debt | 0.542048 | 0.547316 | 0.710720 |
| EBIT/Interest | 0.631074 | 0.705826 | 0.703377 |

| | Macro average Precision | Macro average recall | Accuracy |
|---|---|---|---|
| Current Ratio | 0.618275 | 0.697675 | 0.638767 |
| quick assets/Sales | 0.545470 | 0.576208 | 0.558003 |
| CFO/Total assets | 0.577415 | 0.566196 | 0.762115 |
| Cash/Current Liabilities | 0.600504 | 0.668037 | 0.580029 |
| CFO/EBIT | 0.594279 | 0.650685 | 0.659325 |
| CFO/Sales | 0.550802 | 0.566094 | 0.690162 |
| Operating Income/Total Assets | 0.736923 | 0.788411 | 0.838473 |
| Current Assets/Sales | 0.555945 | 0.593690 | 0.571219 |
| Net Worth/Total Assets | 0.649082 | 0.704559 | 0.757709 |
| Net sales/Cash from sales | 0.553019 | 0.519220 | 0.792952 |
| Ebitda/EBIT | 0.592954 | 0.512567 | 0.202643 |
| Total Liabilities/Total Assets | 0.649082 | 0.704559 | 0.757709 |
| Working Capital/Net worth | 0.591716 | 0.504488 | 0.189427 |
| Interest Coverage | 0.614290 | 0.685180 | 0.669604 |
| EBIT/Total assets | 0.736923 | 0.788411 | 0.838473 |
| Long-term debt/Total Assets | 0.547865 | 0.580240 | 0.559471 |
| Working Capital/Total Assets | 0.639839 | 0.717040 | 0.716593 |
| Turnover Payables | 0.572869 | 0.620504 | 0.533040 |
| Current Assets/Total Assets | 0.554283 | 0.586154 | 0.625551 |

Data Pre-processing and univariate logistic regression code

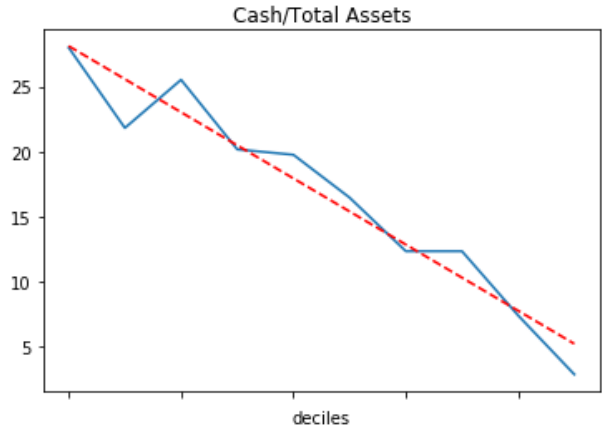
```
for_logit = medie_4y
#preprocessing outliars
mean_ratio = np.mean(for_logit)
std_ratio = np.std(for_logit)
z_score = (for_logit-mean_ratio)/std_ratio
z_score[abs(z_score) >= 3] = np.nan
print(np.sum(z_score[abs(z_score) >= 3].count()))
z_score = z_score*std_ratio+mean_ratio #getting back to normal values
for_logit = z_score
#filling nan
for_logit=for_logit.replace([np.inf, -np.inf], np.nan) #should not be necessary
for_logit.fillna(for_logit.mean(), inplace=True)
#adjusting default
for_logit['Default'] = medie_4y['Default']

#standardization already carried out for outliars detection

#df to store loop results
acc_n_ = pd.DataFrame()
```

```
#standardization already carried out for outliars detection

#df to store loop results
acc_n_ = pd.DataFrame()

#Applying Logit (univariate)
#looping over all ratios
for ratio in list(for_logit):
    x= np.array(for_logit[ratio]).reshape(-1,1)
    y= for_logit['Default']

    # train e test set
    x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.28, random_state=0)
    # fit a model
    model = LogisticRegression(solver='lbfgs')
    model.fit(x_train, y_train)
    # predict probabilities
    y_pred = model.predict_proba(x_test)
    # keep probabilities for the positive outcome only
    y_pred = y_pred[:, 1] #taking only 'positive' probability
    # calculate roc curves
    fpr, tpr, thresholds = roc_curve(y_test, y_pred)
    #roc_auc = auc(fpr, tpr)
    #plt.plot(fpr,tpr, label = f"AUC: {round(roc_auc,2)}")
    metrics.plot_roc_curve(model, x_test, y_test)
    plt.plot([0, 1], [0, 1],'r--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC AUC : {ratio}')
    plt.legend()
    plt.show()

    # get the best threshold
    J = tpr - fpr
    ix = np.argmax(J)
    best_thresh = thresholds[ix]
    #print('Best Threshold=%f' % (best_thresh))
    y_pred = pd.DataFrame(y_pred)
    y_pred['new_thrsld'] = np.where(y_pred >= best_thresh, 1, 0) #new threshold set (sklearn logit has default of 0.5)
    #print(y_pred['new_thrsld'])
    #Creating the Confusion matrix
    cm= confusion_matrix(y_test, y_pred['new_thrsld'])
    print (cm)
    #check evaluation metrics for the confusion table
    classif = classification_report(y_test, y_pred['new_thrsld'], output_dict=True)
    metalist=[]
    metalist.append(classif['macro avg']['precision'])
    metalist.append(classif['macro avg']['recall'])
    metalist.append(classif['accuracy'])
    #metalist.append(sklearn.metrics.r2_score(y_test, y_pred['new_thrsld']))

    acc_n_[ratio] = metalist
    #fpr, tpr, thresholds = roc_curve(y_test,y_pred['new_thrsld'])

acc_n_.drop('Default', inplace=True, axis=1)
acc_n_ = acc_n_.T.rename(columns={0:'Macro average Precision', 1: 'Macro average recall', 2:'Accuracy', 3:'R2'})
acc_n_
```

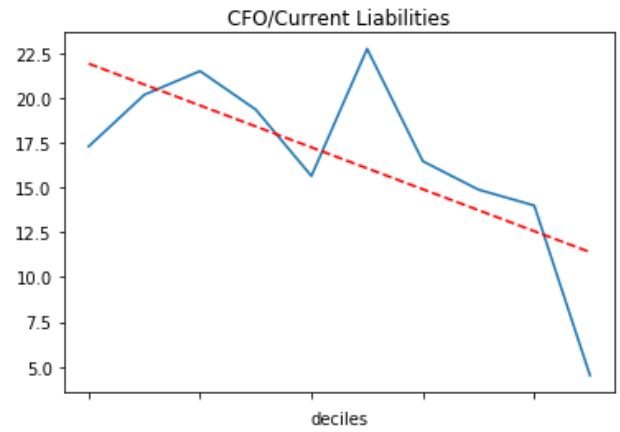
**APPENDIX 4.** Binning and Information Values results

All financial indices charts for binning categorization are reported along with the final IV computed. For sake of brevity Weight of Evidence results are computed through the code but not showed. Each chart is anticipated by the IV of the ratio it belongs to.

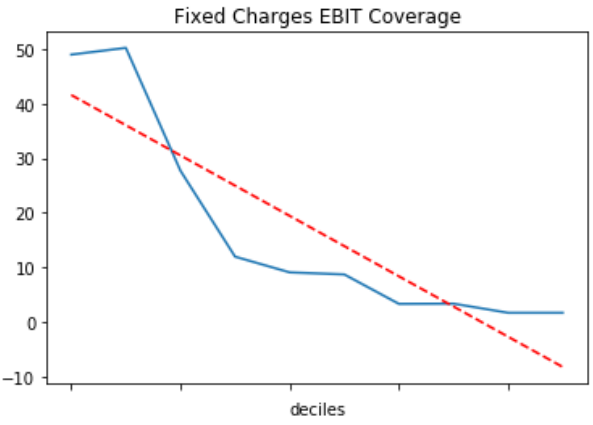
```
0.37401615097452423
```



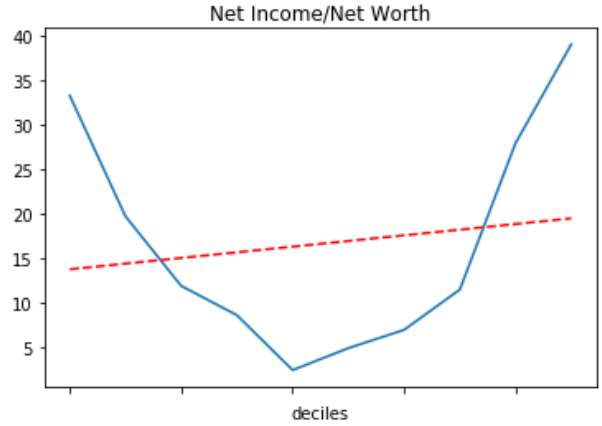

```
0.16559615312798315
```



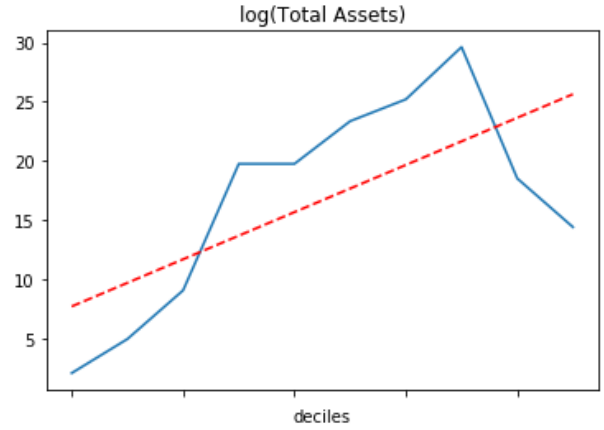

```
1.7379135206919916
```



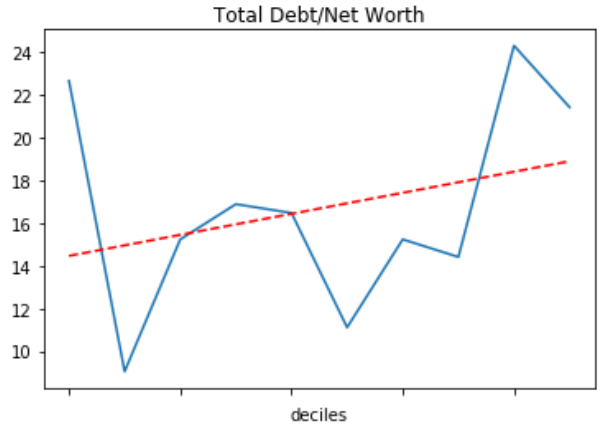

```
0.8050665499623704
```



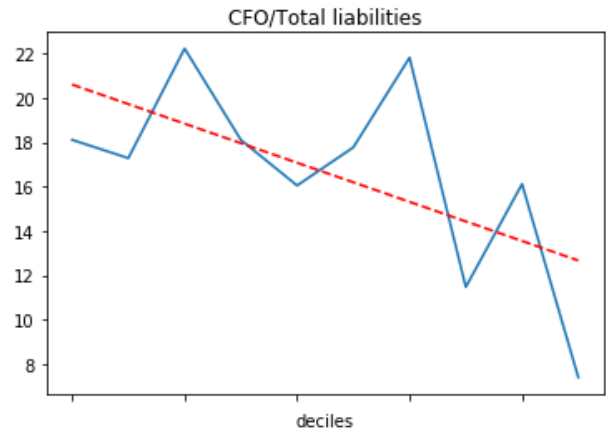

```
0.5094043409664 84
```



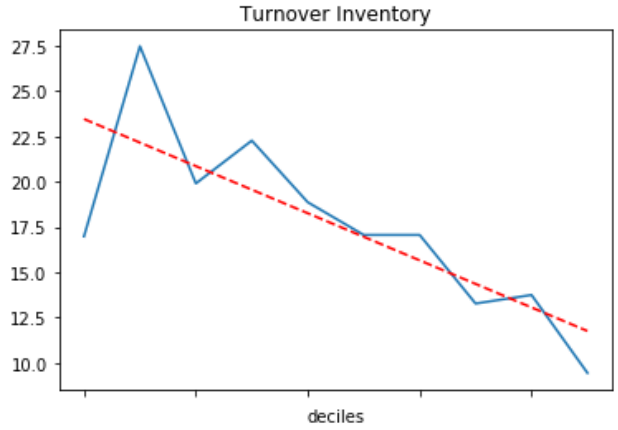

```
0.11312760599463333
```



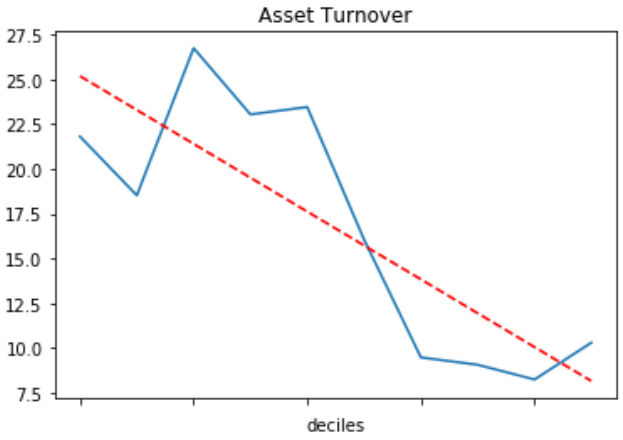
0.10545330479911268



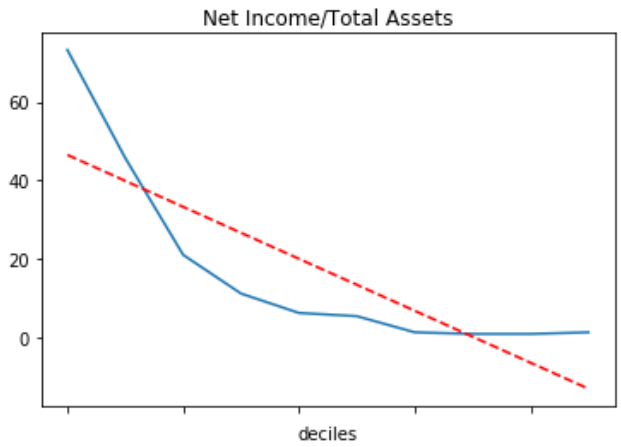

0.10873950094329488



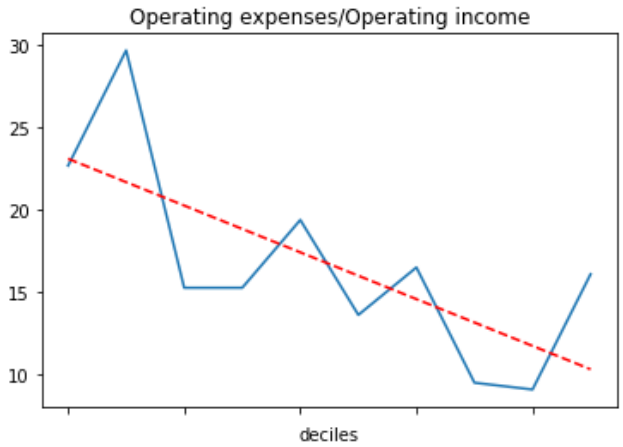

0.23953149897517395



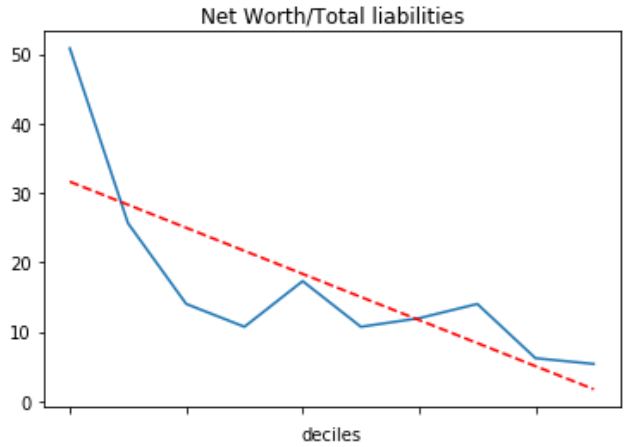

2.8196336977392633



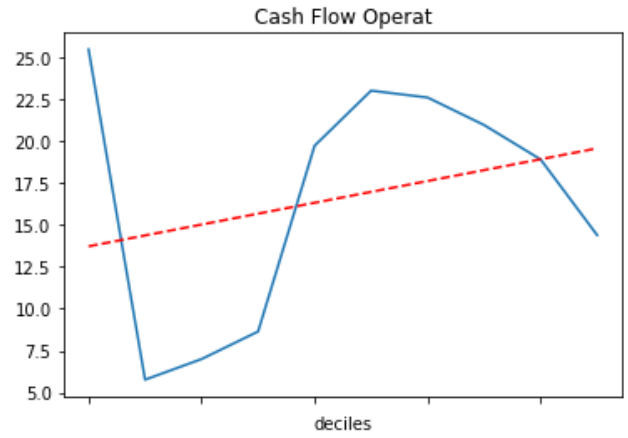

0.1665537837072422



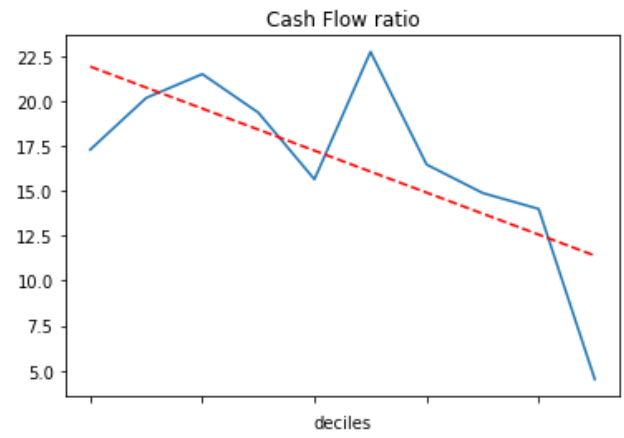

0.683166919652213



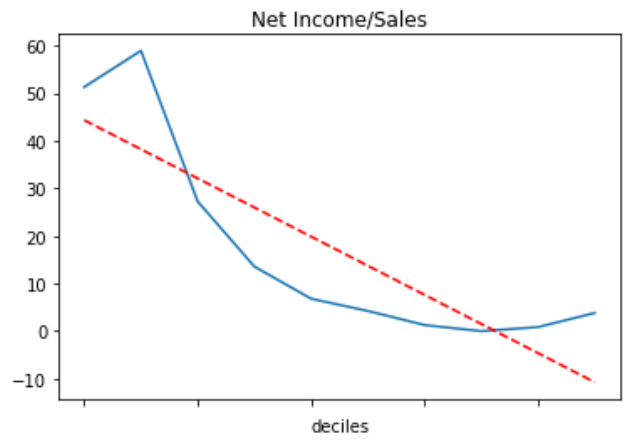
0.28671827839883174



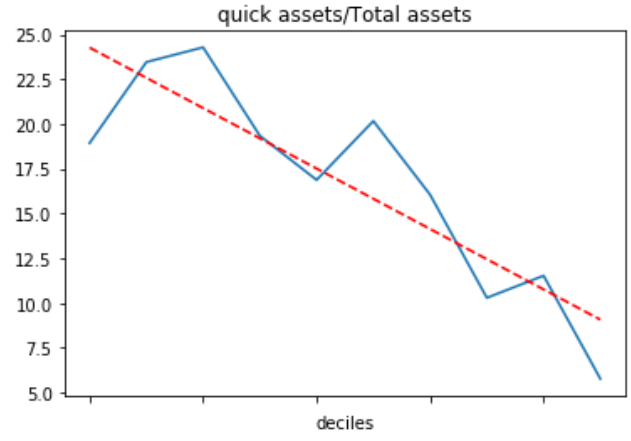

0.16559615312798315




2.558340689502477



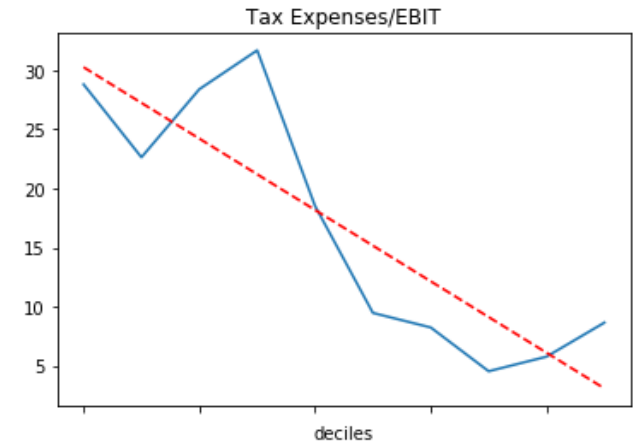

0.18969763999872158



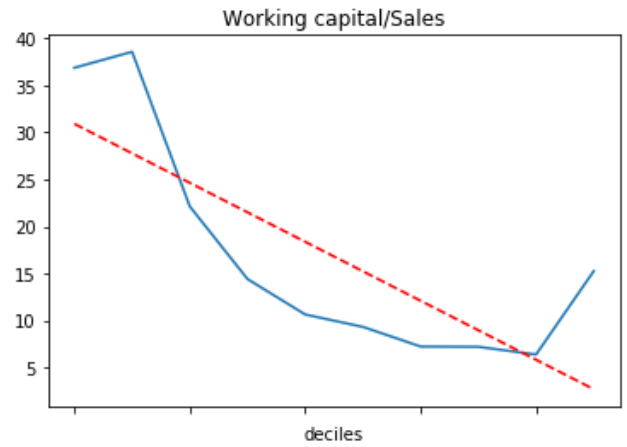

0.23953149897517395


2.4159614599535675

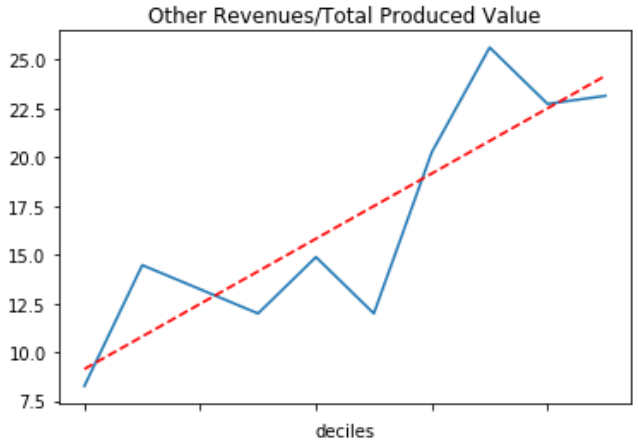
0.5622149932259825



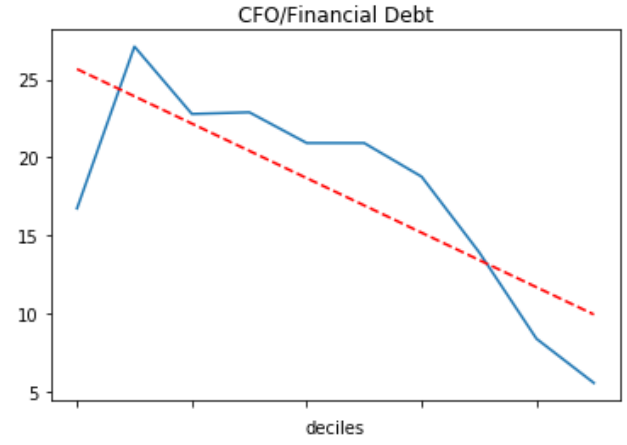

0.6072961789724135



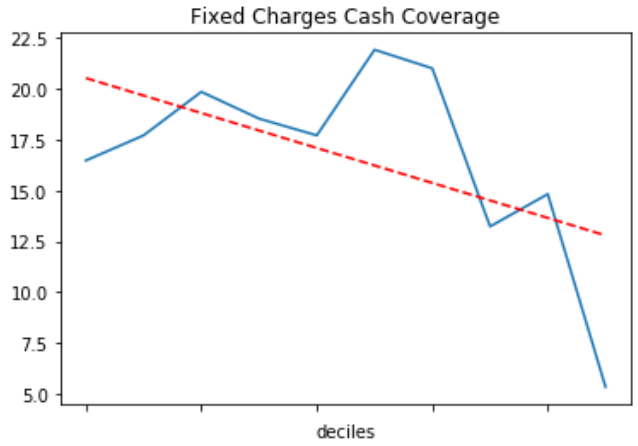

0.15999271218329236



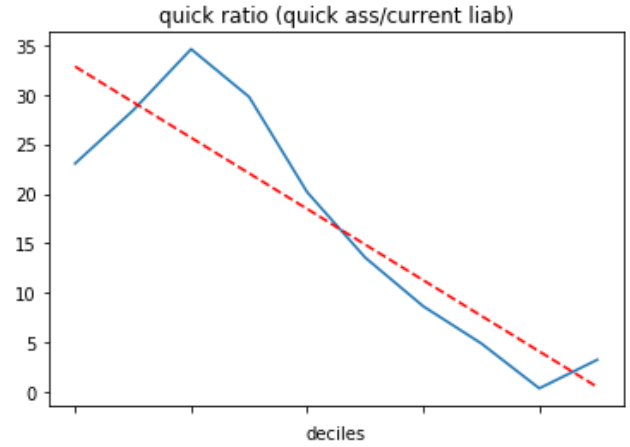

0.2312397421603016



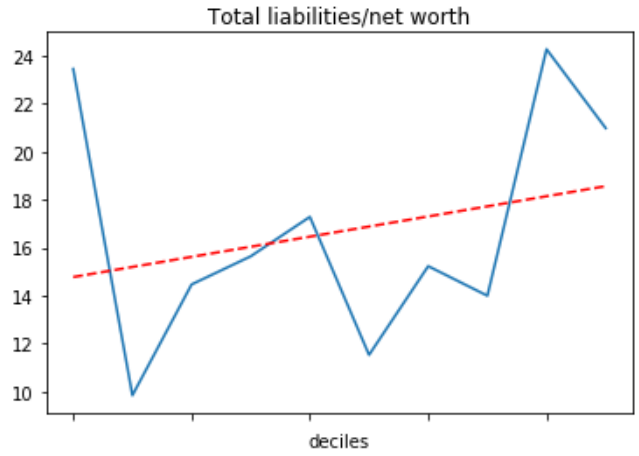

0.13788296693371246



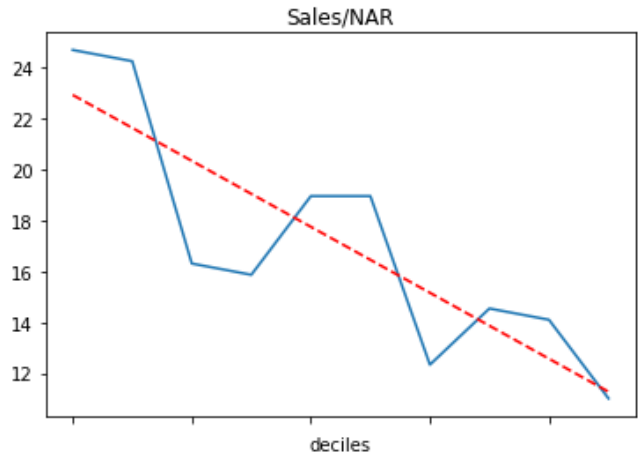

1.0037831668225239



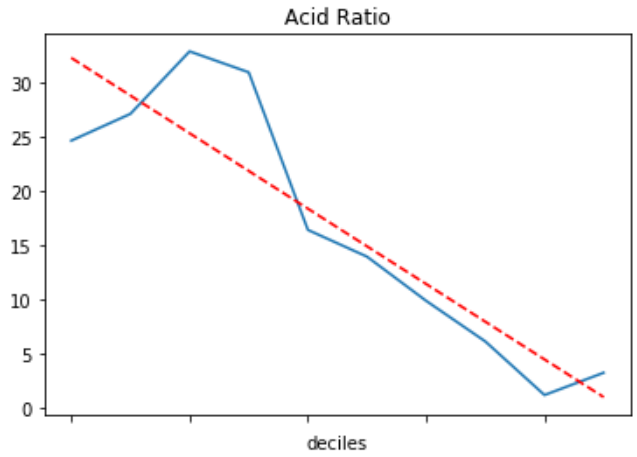
0.10832082330309412



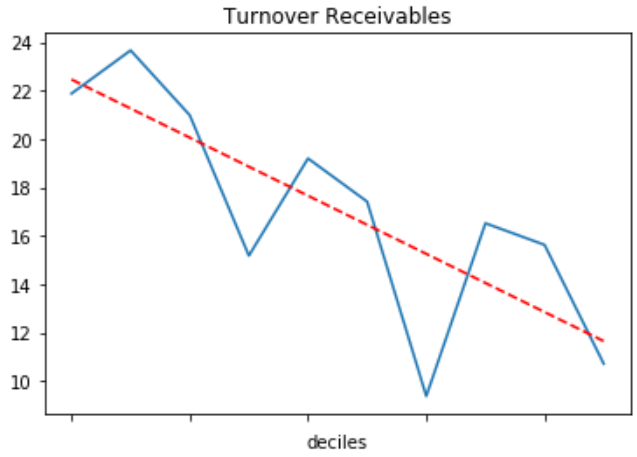

0.09288473259187786



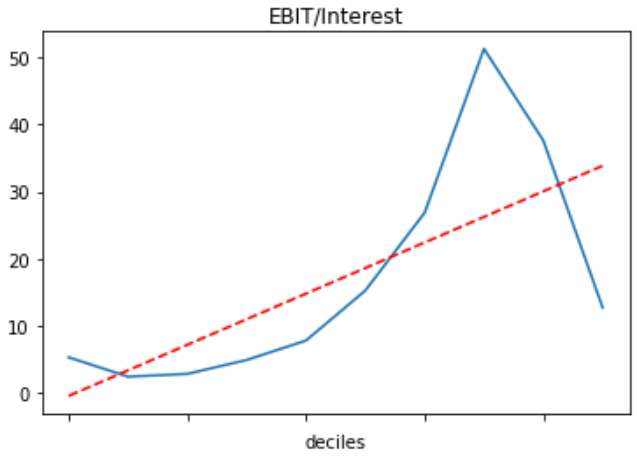

0.8312912032238695



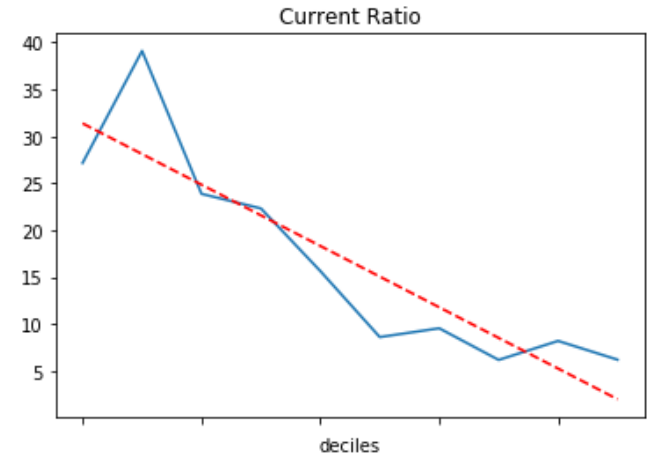

0.10181008900863507



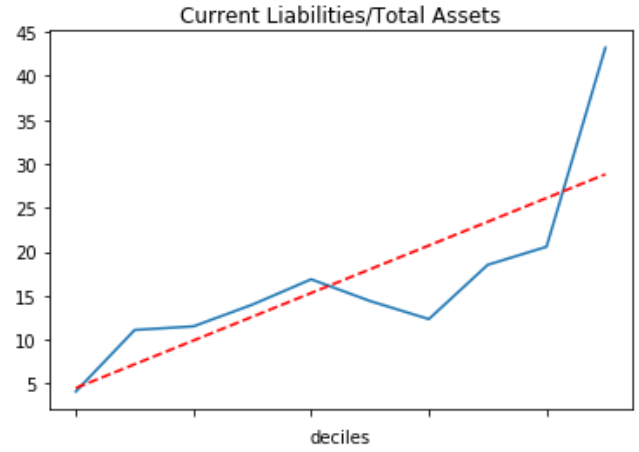

1.2740537218705743



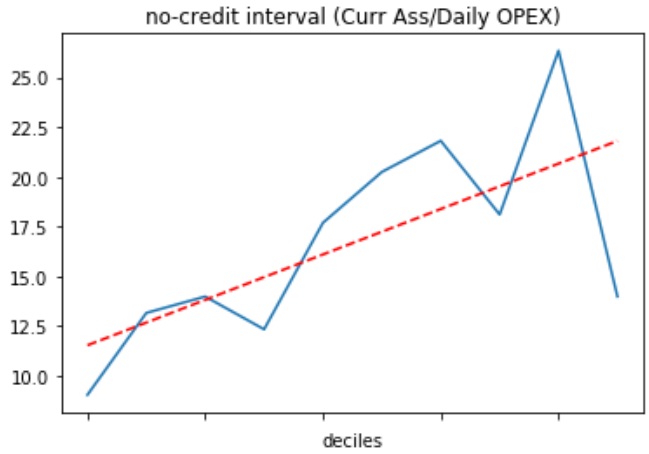

0.5587564936528632



0.4484592919300364

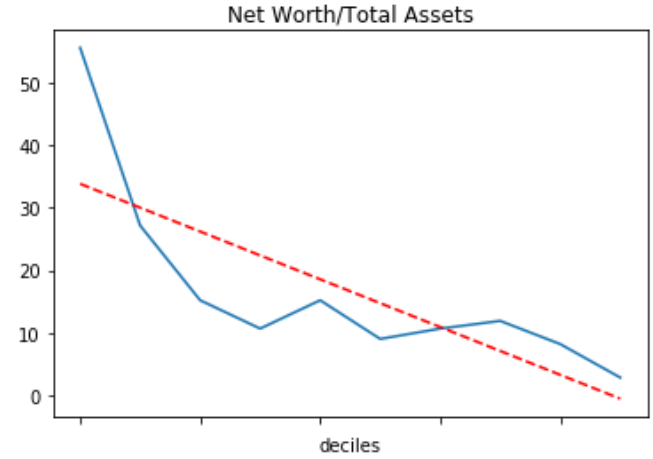



0.12276235108498305

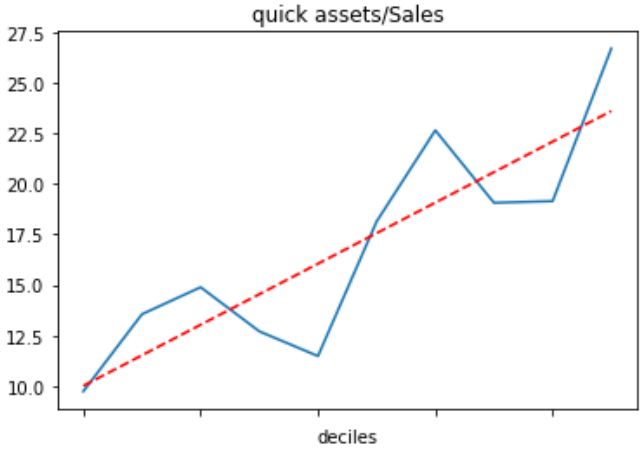



0.6987591760070249

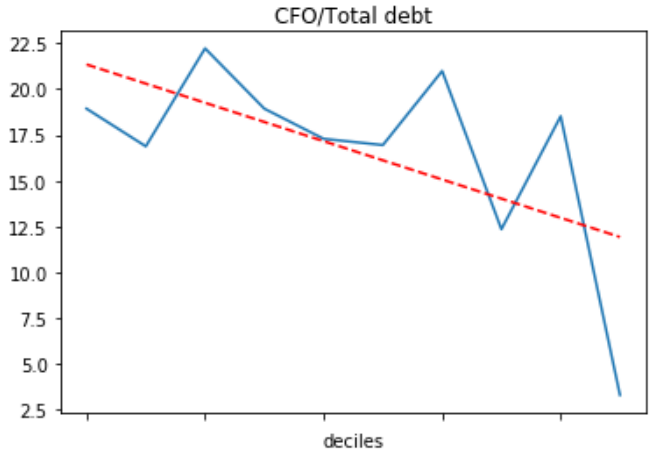

0.8818362356083904



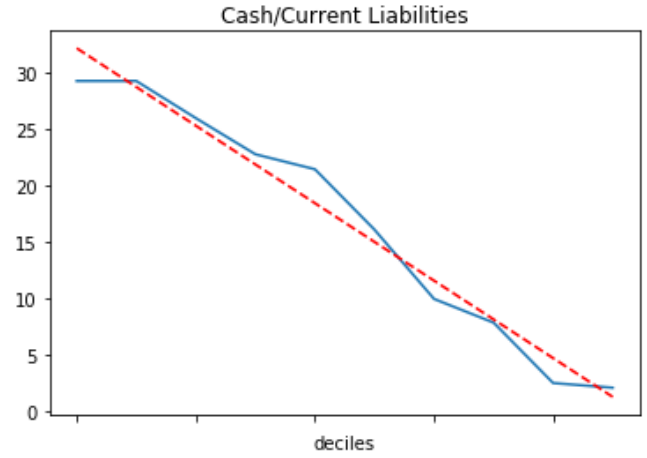

0.12702887392582768



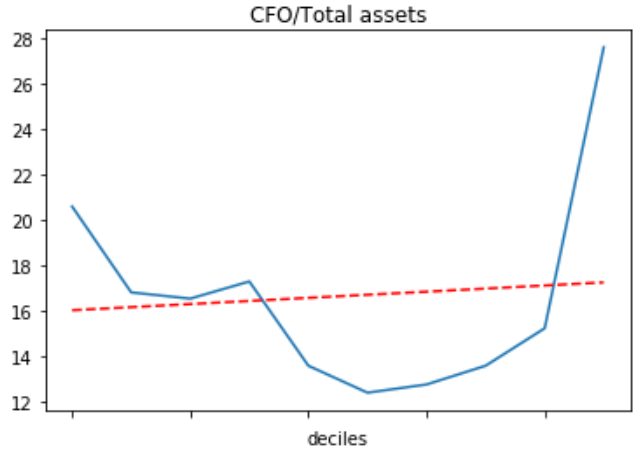

0.20564483509349085



0.7060478750461419

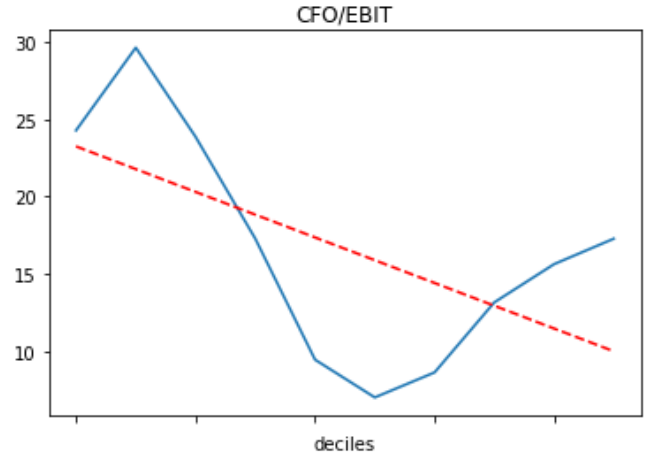



0.08826721207075075

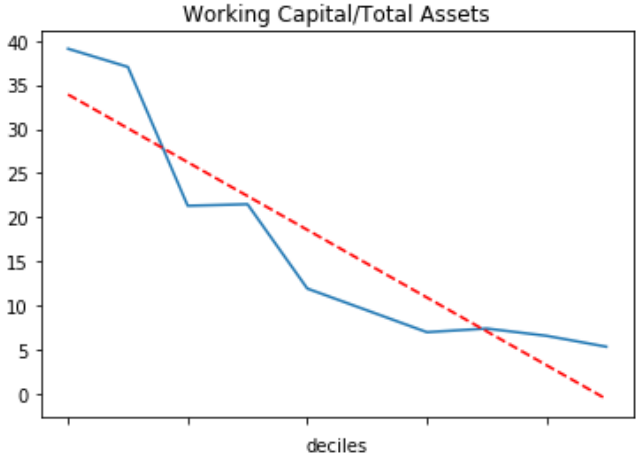



0.25273533612569105

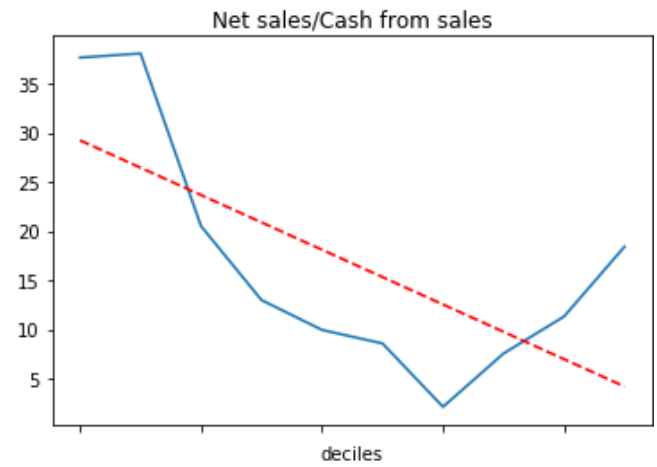

0.2666862962638735



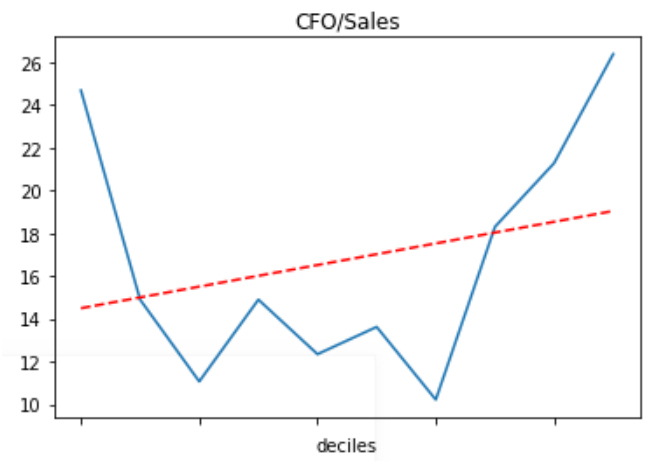

0.7099922415971118



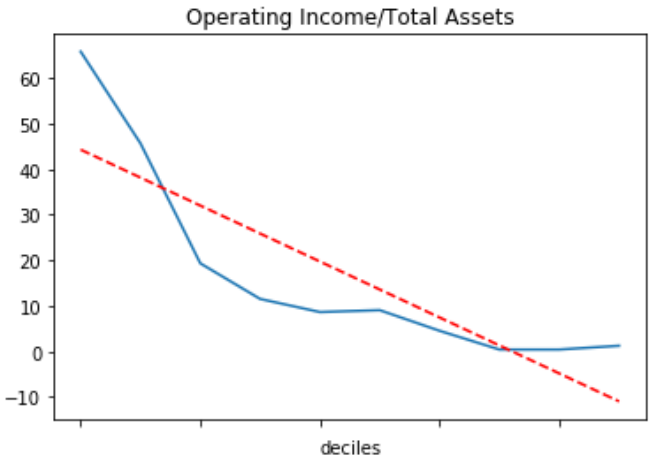

0.7183691399783614



0.14307387670686703

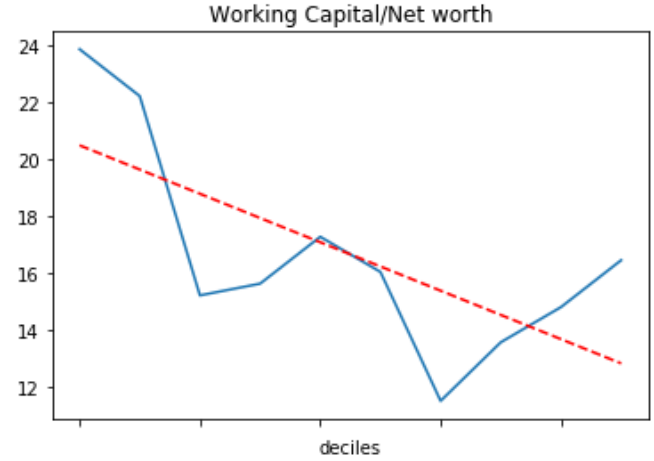



2.4159614599535675

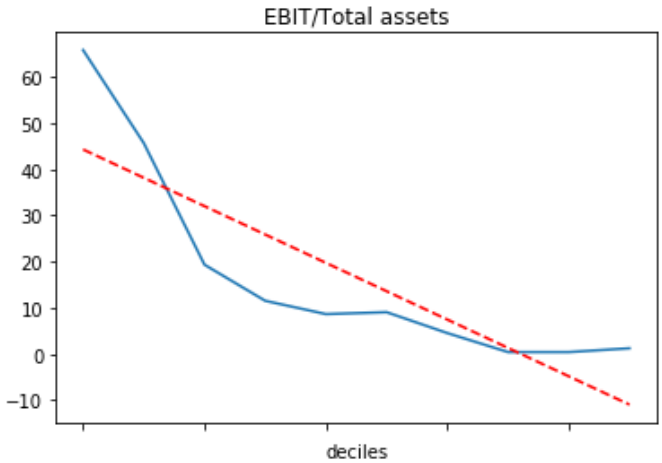



0.8818362356083906

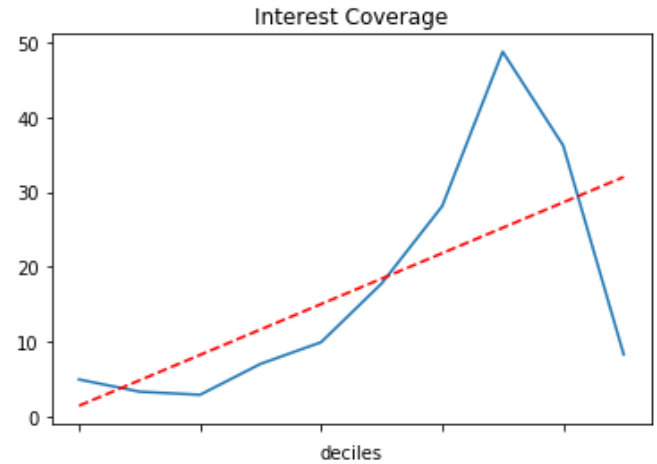

0.06197178539875638



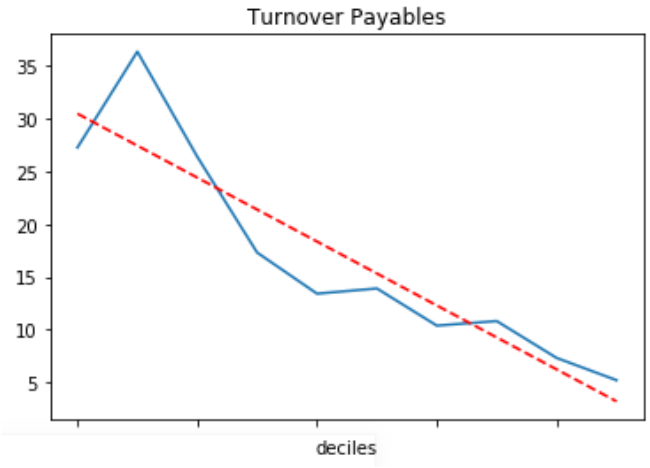

2.4159614599535675



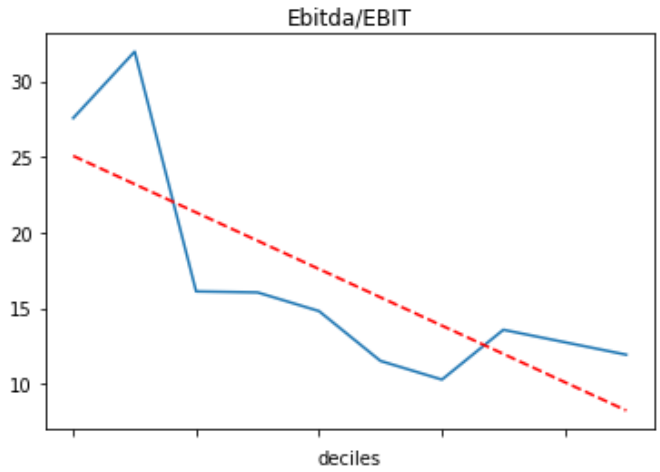

1.1555879149583228



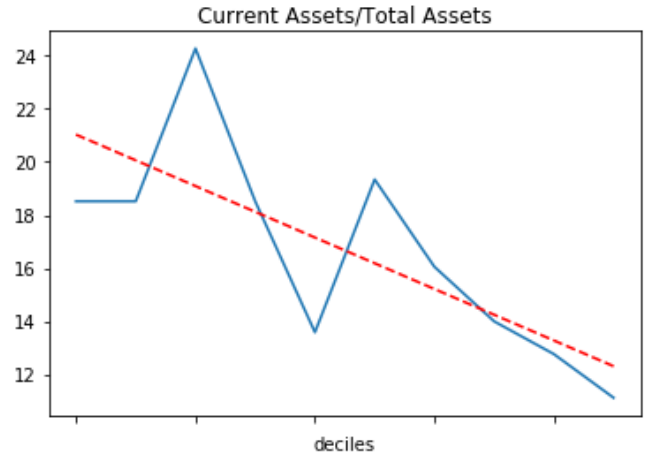
```
0.4526597606454751
```




```
0.21331800910919235
```




```
0.0706533557660128
```




Long-term debt/Total Assets, Retained Earnings/Total Assets and Inventory/Sales could not being displayed since their distribution reported a too high occurrences of 0s. This has prevented the code from finding the right end of each decile. For this reason, the following error message is printed on Python Console, for each of them:

```
Inventory/Sales distribution has too many equal values (0.0).
Binning cannot be carried out: python does not know how to create decil
es
```

Python code for Binning, WoE and IVs

```
#computing deciles to ratios distributions
rel_dec = [0,1,2,3,4,5,6,7,8,9]
binning = medie_4y[:]
n_indeces_check=0
prepriority=pd.DataFrame()
prepriority_iv=pd.DataFrame()
```

```
for input_tobe in list(binning.drop('Default', axis=1)):
    n_indeces_check+=1
    result_list=[]
    #deciles_list=[]
    stats=pd.DataFrame()
    non_def = []  #WOE and IV
    default = []

    try:
        binning['deciles']= pd.qcut(binning[input_tobe],10,labels= rel_dec, retbins=False, precision=3, duplicates='raise')
    except ValueError:
        prepriority['{}'.format(input_tobe)] = [-1]
        prepriority_iv['{}'.format(input_tobe)] = [-1000]
        print('{} distribution has too many equal values (0.0). \nBinning cannot be carried out:',
              'python does not know how to create deciles'.format(input_tobe))
        continue
```

```
    beta={}
    alpha = binning[['deciles', 'Default']]
    for x in rel_dec:
        beta['decile {}'.format(x)] = alpha[alpha['deciles']==x]
        numb_defaulted = beta['decile {}'.format(x)]['Default'].replace(np.inf,np.nan).sum()
        non_def.append(len(beta['decile {}'.format(x)]['Default'].replace(np.inf,np.nan).dropna()) - numb_defaulted)
        default.append(numb_defaulted)
        result_list.append(beta['decile {}'.format(x)]['Default'].replace(np.inf,np.nan).mean()*100)
        #deciles_list.append(beta['decile {}'.format(x)]['deciles'][0])
    stats['freq_default'] = result_list
    #print(default,non_def)
    woe=[]
    iv =0
    for x in default:
        perc_non = ((non_def[default.index(x)]+0.5)/(sum(non_def)+0.5))
        perc_def = ((x+0.5)/(sum(default)+0.5))
        woe_decile = np.log( perc_non/perc_def )
        woe.append(woe_decile) #WOE of the decile
        iv += (perc_non - perc_def) * woe_decile
    #print(sum(non_def), sum(default))
    #print(woe)
    if input_tobe == 'EBIT/Total assets':
        print(non_def, default, woe)
    meta_iv = []
    meta_iv.append(iv)
    print(iv)

    stats['deciles'] = ['q1', 'q2', 'q3', 'q4', 'q5', 'q6', 'q7', 'q8', 'q9', 'q10']
    stats = stats.set_index('deciles')
```

```
    lines1 = stats['freq_default'].plot.line()

    #tendency line
    z = np.polyfit(rel_dec, list(stats['freq_default']), 1)
    p = np.poly1d(z)
    plt.plot(rel_dec,p(rel_dec),"r--")
    slope_abs = abs(p(rel_dec)[1]-p(rel_dec)[0]) #we only care about abs slope
    meta=[]
    meta.append(slope_abs)
    prepriority['{}'.format(input_tobe)] = meta
    prepriority_iv['{}'.format(input_tobe)] = meta_iv

    plt.title(input_tobe)
    plt.show()
print(n_indeces_check)

prepriority = prepriority.T.sort_values(by=[0], axis=0, ascending= False)
prepriority_iv= prepriority_iv.T.sort_values(by=[0], axis=0, ascending= False)
print(list(prepriority_iv[0]))
priority_binning = list(prepriority.T)
priority_IV = list(prepriority_iv.T)
#priority_IV,priority_binning
```

## **APPENDIX 5.** Code for retrieving correlations

```
#check with the ending check
print(len(list(final_indeces)))
#dropping every '2009' column --> dropping for sake of simplicity
for x in list(final_indeces):
    if x[-4:] == '2009':
        final_indeces.drop([x], inplace=True, axis=1)
#check
print(len(list(final_indeces)))
```

```
#creating matrices for correlation (TRANSPOSING ALL ELEMENTS PREVIOUSLY COMPUTED)
final_matrix = {}
#creating list of ratios name
index_name_list = []
for u in list(final_indeces):
    if u[-4:]=='2010':
        index_name_list.append(u[:-5])
#print(len(index_name_list))
#print(index_name_list)

for a in list(range(0,final_indeces.shape[0])):
    columns_index = pd.DataFrame()
    #if a%500==0:
        #print(a)
    for name_ind in index_name_list:
        columns_index[name_ind] =final_indeces.iloc[a]
        [name_ind + ' 2010':name_ind + ' 2018'].T.rename(index={
            name_ind + ' 2010': 2010, name_ind + ' 2011': 2011,
            name_ind + ' 2012': 2012, name_ind + ' 2013': 2013,
            name_ind + ' 2014': 2014, name_ind + ' 2015': 2015,
            name_ind + ' 2016': 2016, name_ind + ' 2017': 2017,
                                      name_ind + ' 2018': 2018})
        #print(columns_index[name])
    final_matrix ['azienda n {}'.format(a)] =  columns_index

#final_matrix['azienda n 1']
```

```
#computing correlation coefficients
corr_dict = {}

for item in final_matrix:

    corr_dict[item] = final_matrix[item].replace([np.inf, -np.inf], np.nan).corr()

#corr_dict['azienda n 2']
```

```
aggregat = 0
for xi in list(corr_dict):
    for yi in list(corr_dict[xi]):
        aggregat += corr_dict[xi][yi].isna().sum()
#aggregat
```

```
#average correlation
average_correls = pd.DataFrame(0, index=index_name_list, columns=index_name_list)
division_matrix = pd.DataFrame(0, index=index_name_list, columns=index_name_list)

for azienda in corr_dict:
    meta_matr = pd.DataFrame(1, index=index_name_list, columns=index_name_list)
    meta_matr = meta_matr - corr_dict [azienda].isna().astype(int)
    division_matrix += meta_matr

#division_matrix

for t in corr_dict:
    average_correls += corr_dict[t].replace(np.nan, 0)

#dividing for the division_matrix
average_correls = average_correls / division_matrix

average_correls
```

```
%store average_correls
%store index_name_list
%store final_all
```

**APPENDIX 6.** Priority lists and the code to build them

| | accuracy | literature | Precision | Recall | Binning | IV |
|---|---|---|---|---|---|---|
| 0 | Net Income/Total Assets | Net Income/Total Assets | Net Income/Total Assets | Net Income/Total Assets | Net Income/Total Assets | Net Income/Total Assets |
| 1 | EBIT/Total assets | Current Ratio | Operating Income/Total Assets | Net Income/Sales | Operating Income/Total Assets | Net Income/Sales |
| 2 | Operating Income/Total Assets | Working Capital/Total Assets | Return on Total Asset | Operating Income/Total Assets | Return on Total Asset | Operating Income/Total Assets |
| 3 | Return on Total Asset | Retained earnings/Total assets | EBIT/Total assets | EBIT/Total assets | EBIT/Total assets | EBIT/Total assets |
| 4 | Fixed Charges EBIT Coverage | EBIT/Total assets | Net Income/Sales | Return on Total Asset | Net Income/Sales | Return on Total Asset |
| 5 | Net Income/Sales | Sales/Total assets | Fixed Charges EBIT Coverage | Fixed Charges EBIT Coverage | Fixed Charges EBIT Coverage | Fixed Charges EBIT Coverage |
| 6 | Net sales/Cash from sales | quick ratio (quick ass/current liab) | Total Debt/Total Assets | Working Capital/Total Assets | Working Capital/Total Assets | EBIT/Interest |
| 7 | Total Debt/Total Assets | Total Debt/Total Assets | Total Liabilities/Total Assets | EBIT/Interest | Total Liabilities/Total Assets | Interest Coverage |
| 8 | Cash Flow Operat | Current Assets/Total Assets | Net Worth/Total Assets | Net Worth/Total Assets | Net Worth/Total Assets | quick ratio (quick ass/current liab) |
| 9 | Inventory/Sales | Net Income/Net Worth | Working Capital/Total Assets | Total Liabilities/Total Assets | EBIT/Interest | Total Liabilities/Total Assets |
| 10 | CFO/Total assets | Total Liabilities/Total Assets | Working capital/Sales | Current Ratio | Total Debt/Total Assets | Net Worth/Total Assets |
| 11 | Net Worth/Total Assets | Cash/Total Assets | EBIT/Interest | quick ratio (quick ass/current liab) | quick ratio (quick ass/current liab) | Acid Ratio |
| 12 | Net Income/Net Worth | CFO/Total assets | Net Worth/Total liabilities | Total Debt/Total Assets | Acid Ratio | Net Income/Net Worth |
| 13 | Total Liabilities/Total Assets | CFO/Total liabilities | Current Ratio | Interest Coverage | Cash/Current Liabilities | Net sales/Cash from sales |
| 14 | Working capital/Sales | Current Liabilities/Total Assets | Interest Coverage | Tax Expenses/EBIT | Interest Coverage | Working Capital/Total Assets |
| 15 | Net Worth/Total liabilities | CFO/Total debt | quick ratio (quick ass/current liab) | Working capital/Sales | Net Worth/Total liabilities | Cash/Current Liabilities |
| 16 | Total liabilities/net worth | quick assets/Total assets | Tax Expenses/EBIT | Net Worth/Total liabilities | Current Ratio | Total Debt/Total Assets |
| 17 | Total Debt/Net Worth | Current Assets/Sales | Acid Ratio | Acid Ratio | Working capital/Sales | Net Worth/Total liabilities |
| 18 | Working Capital/Total Assets | EBIT/Interest | Cash/Current Liabilities | Cash/Current Liabilities | Turnover Payables | Working capital/Sales |
| 19 | Retained earnings/Total assets | Inventory/Sales | CFO/EBIT | CFO/EBIT | Tax Expenses/EBIT | Tax Expenses/EBIT |
| 20 | CFO/Total debt | Operating Income/Total Assets | Ebitda/EBIT | Current Liabilities/Total Assets | Net sales/Cash from sales | Current Ratio |
| 21 | CFO/Total liabilities | CFO/Sales | Working Capital/Net worth | Turnover Payables | Current Liabilities/Total Assets | log(Total Assets) |
| 22 | EBIT/Interest | Net Income/Sales | Current Liabilities/Total Assets | log(Total Assets) | Cash/Total Assets | Turnover Payables |
| 23 | CFO/Sales | Long-term debt/Total Assets | Cash Flow ratio | Cash/Total Assets | Current Assets/Sales | Current Liabilities/Total Assets |
| 24 | Operating expenses/Operating income | Net Worth/Total Assets | CFO/Current Liabilities | CFO/Financial Debt | log(Total Assets) | Cash/Total Assets |
| 25 | Sales/NAR | Total Debt/Net Worth | Retained earnings/Total assets | Retained earnings/Total assets | Asset Turnover | Cash Flow Operat |
| 26 | Current Liabilities/Total Assets | Total liabilities/net worth | log(Total Assets) | Other Revenues/Total Produced Value | Sales/Total assets | CFO/EBIT |
| 27 | Interest Coverage | Cash/Current Liabilities | CFO/Financial Debt | Sales/Total assets | Ebitda/EBIT | Current Assets/Sales |
| 28 | CFO/EBIT | CFO/Current Liabilities | CFO/Total assets | Asset Turnover | CFO/Financial Debt | Sales/Total assets |
| 29 | Acid Ratio | Working capital/Sales | Turnover Payables | Current Assets/Sales | quick assets/Total assets | Asset Turnover |
| 30 | Cash/Total Assets | Net Worth/Total liabilities | Cash/Total Assets | Current Assets/Total Assets | Other Revenues/Total Produced Value | CFO/Financial Debt |
| 31 | Current Ratio | no-credit interval (Curr Ass/Daily OPEX) | Net Income/Net Worth | Long-term debt/Total Assets | quick assets/Sales | Ebitda/EBIT |
| 32 | Tax Expenses/EBIT | Cash Flow Operat | Asset Turnover | Turnover Inventory | CFO/EBIT | CFO/Total debt |
| 33 | Current Assets/Total Assets | Operating expenses/Operating income | Sales/Total assets | quick assets/Total assets | Operating expenses/Operating income | quick assets/Total assets |
| 34 | quick ratio (quick ass/current liab) | quick assets/Sales | Fixed Charges Cash Coverage | quick assets/Sales | Turnover Inventory | Operating expenses/Operating income |
| 35 | Other Revenues/Total Produced Value | Working Capital/Net worth | Cash Flow Operat | Sales/NAR | Sales/NAR | Cash Flow ratio |

| | accuracy | literature | Precision | Recall | Binning | IV |
|---|---|---|---|---|---|---|
| 36 | Cash/Current Liabilities | Asset Turnover | Other Revenues/Total Produced Value | Operating expenses/Operating income | Turnover Receivables | CFO/Current Liabilities |
| 37 | Current Assets/Sales | Return on Total Asset | quick assets/Total assets | CFO/Total assets | CFO/Current Liabilities | Other Revenues/Total Produced Value |
| 38 | Long-term debt/Total Assets | Ebitda/EBIT | Current Assets/Sales | CFO/Sales | Cash Flow ratio | CFO/Sales |
| 39 | quick assets/Sales | CFO/EBIT | Current Assets/Total Assets | Turnover Receivables | no-credit interval (Curr Ass/Daily OPEX) | Fixed Charges Cash Coverage |
| 40 | Turnover Payables | Tax Expenses/EBIT | Sales/NAR | Fixed Charges Cash Coverage | CFO/Total debt | quick assets/Sales |
| 41 | no-credit interval (Curr Ass/Daily OPEX) | Other Revenues/Total Produced Value | Net sales/Cash from sales | Cash Flow ratio | Current Assets/Total Assets | no-credit interval (Curr Ass/Daily OPEX) |
| 42 | Turnover Inventory | Cash Flow ratio | Operating expenses/Operating income | CFO/Current Liabilities | CFO/Total liabilities | Total Debt/Net Worth |
| 43 | Turnover Receivables | Interest Coverage | CFO/Sales | Net Income/Net Worth | Fixed Charges Cash Coverage | Turnover Inventory |
| 44 | Sales/Total assets | log(Total Assets) | Turnover Inventory | no-credit interval (Curr Ass/Daily OPEX) | Working Capital/Net worth | Total liabilities/net worth |
| 45 | Asset Turnover | Turnover Payables | Long-term debt/Total Assets | CFO/Total debt | Cash Flow Operat | CFO/Total liabilities |
| 46 | log(Total Assets) | Turnover Receivables | quick assets/Sales | CFO/Total liabilities | Net Income/Net Worth | Turnover Receivables |
| 47 | CFO/Financial Debt | Turnover Inventory | Total liabilities/net worth | Total liabilities/net worth | CFO/Sales | Sales/NAR |
| 48 | quick assets/Total assets | Acid Ratio | CFO/Total debt | Total Debt/Net Worth | Total Debt/Net Worth | CFO/Total assets |
| 49 | Fixed Charges Cash Coverage | Net sales/Cash from sales | CFO/Total liabilities | Cash Flow Operat | Total liabilities/net worth | Current Assets/Total Assets |
| 50 | Cash Flow ratio | Sales/NAR | Total Debt/Net Worth | Net sales/Cash from sales | CFO/Total assets | Working Capital/Net worth |
| 51 | CFO/Current Liabilities | CFO/Financial Debt | Turnover Receivables | Inventory/Sales | Inventory/Sales | Inventory/Sales |
| 52 | Ebitda/EBIT | Fixed Charges Cash Coverage | no-credit interval (Curr Ass/Daily OPEX) | Ebitda/EBIT | Retained earnings/Total assets | Retained earnings/Total assets |
| 53 | Working Capital/Net worth | Fixed Charges EBIT Coverage | Inventory/Sales | Working Capital/Net worth | Long-term debt/Total Assets | Long-term debt/Total Assets |

Code for Binning and IV priority lists

```
prepriority = prepriority.T.sort_values(by=[0], axis=0, ascending= False)
prepriority_iv= prepriority_iv.T.sort_values(by=[0], axis=0, ascending= False)
print(list(prepriority_iv[0]))
priority_binning = list(prepriority.T)
priority_IV = list(prepriority_iv.T)
#priority_IV,priority_binning
```

Code for Accuracy, Precision and Recall priority lists

```
#rtriving priority_accuracy
sorted_ = acc_n_.sort_values(['Accuracy'])
priority_accuracy = list(sorted_.T)[::-1]
priority_accuracy
#rtriving priority_Precision
sorted_ = acc_n_.sort_values(['Macro average Precision'])
priority_Precision = list(sorted_.T)[::-1]
priority_accuracy
#rtriving priority_Recall
sorted_ = acc_n_.sort_values(['Macro average recall'])
priority_Recall = list(sorted_.T)[::-1]
priority_Recall
```

The priority list built on the relevant Literature has been created manually.

**APPENDIX 7.** Data pre-processing and correlation funnel for prediction models

Prioritizing function

```
def prioritizing (threshold):

    final = pd.DataFrame()
    final = df_out[abs(df_out[0]) > threshold]
    #print(len(final))

    #print(ratio_A,ratio_B)
    #deciding which one is better to keep
    to_be_used = {}

    for prior_name in list(prior_dict):


        winners = []
        losers = []

        left_out = []


        index_name_list = prior_dict[prior_name][:]
        for t in [1,2, 3]:
            check = 0
            for pair in list(final.T):
                ratio_A = pair [:list(pair).index('_')]
                ratio_B = pair [list(pair).index('_')+1:]
                #print(ratio_A,'-',ratio_B)
```

```
                if ratio_A in losers and ratio_B in losers:
                    check+=1
                    continue

                elif ratio_A in winners and ratio_B in winners:
                    check+=1
                    if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                        winners.remove(ratio_B)
                        losers.append(ratio_B)
                    else:
                        winners.remove(ratio_A)
                        losers.append(ratio_A)

                elif ratio_A in winners and ratio_B in losers:
                    check+=1
                    if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                        continue
                    else:
                        winners.remove(ratio_A)
                        losers.append(ratio_A)

                elif ratio_B in winners and ratio_A in losers:
                    check+=1
                    if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                        winners.remove(ratio_B)
                        losers.append(ratio_B)
                    else:
                        continue
```

```
                elif ratio_A in winners:
                    check+=1
                    if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                        losers.append(ratio_B)
                    else:
                        winners.remove(ratio_A)
                        losers.append(ratio_A)
                        winners.append(ratio_B)

                elif ratio_A in losers:
                    check+=1
                    if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                        losers.append(ratio_B)
                    else:
                        winners.append(ratio_B) #problem! 2 times rolling to solve it
```

```
            elif ratio_B in winners:
                check+=1
                if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                    winners.remove(ratio_B)
                    losers.append(ratio_B)
                    winners.append(ratio_A)
                else:
                    losers.append(ratio_A)
```

```
    #preparing the final tabel
    lista_assegna = []
    for uy in index_name_list:
        if uy in winners:
            lista_assegna.append('winner')
        elif uy in losers:
            lista_assegna.append('loser')
        elif uy in left_out:
            lista_assegna.append('to be used (no corr)')
        else:
            print('ERROR!!! probabile errore: NON TORNANO I N DI INDICI NELLE DUE LISTE')
    #print(lista_assegna)
    #print('Per {}\nDa utilizzare: '.format(prior_name),(len(winners)+len(left_out)),'\nDa scartare:
    #', len(losers),'\nNON CORRELANTI: ',len(left_out))
    final_tabel = pd.DataFrame()
    final_tabel['INDICI'] = index_name_list
    final_tabel['status'] = lista_assegna
    da_usare = list(final_tabel[final_tabel['status'] != 'loser']['INDICI'])
    #final_tabel
    to_be_used[prior_name] = da_usare
return to_be_used
```

```
            elif ratio_B in losers:
                check+=1
                if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                    winners.append(ratio_A) #problem! 2 times rolling to solve it
                else:
                    losers.append(ratio_A)

            else:
                check+=1
                if index_name_list.index(ratio_A) < index_name_list.index(ratio_B):
                    winners.append(ratio_A)
                    losers.append(ratio_B)
                else:
                    winners.append(ratio_B)
                    losers.append(ratio_A)

    #taking care of left out values
    metalistx = winners+losers
    #print(metalistx)
    for c in index_name_list:
        if c not in metalistx:
            left_out.append(c)
    #check
    #print(len(left_out+metalistx)==len(index_name_list))
    #union = (left_out+metalistx)
    #checker = 0
    #for cv in index_name_list:
        #if cv not in union:
            #checker+=1
```

Pre-processing section

```
#try to put everything inside
corr_range=0.01
ROC_AUC_corr = 0.7   #it needs to be included in the range at the beginning of the loop
#PREPROCESSING
multi_ = medie_4y
#preprocessing outliars
mean_ratio = np.mean(multi_)
std_ratio = np.std(multi_)
z_score = (multi_-mean_ratio)/std_ratio
z_score[abs(z_score) >= 3] = np.nan
print(np.sum(z_score[abs(z_score) >= 3].count())==0)
z_score = z_score*std_ratio+mean_ratio #getting back to normal values
multi_ = z_score
#filling nan
multi_=multi_.replace([np.inf, -np.inf], np.nan) #should not be necessary
multi_.fillna(multi_.mean(), inplace=True)

#adjusting default
multi_['Default'] = medie_4y['Default']
```

Embedding *Prioritizing* into Correlation funnel

```
acc_logit ={}
acc_SVM ={}
acc_KNN ={}
acc_AdaBoost={}
acc_DecisionTree={}
acc_XGboost={}

test_logit =[]
test_SVM =[]
test_KNN =[]
test_AdaBoost=[]
test_DecisionTree=[]
test_XGboost=[]   #it varies depending on ROC_AUC_corr

ROC_AUC = {}
to_be_u = prioritizing(ROC_AUC_corr)
for screamed in list(to_be_u):
    ROC_AUC[screamed]={}

for corr_level in list(np.around(list(np.arange(0.3,0.9,corr_range)),4)):
    to_be_used = prioritizing(corr_level)
    for screamed in list(to_be_used):
```

**APPENDIX 8.** Multivariate prediction models complete code (with testing section)

```
#try to put everything inside
corr_range=0.01
ROC_AUC_corr = 0.7   #it needs to be included in the range at the beginning of the loop
#PREPROCESSING
multi_ = medie_4y
#preprocessing outliars
mean_ratio = np.mean(multi_)
std_ratio = np.std(multi_)
z_score = (multi_-mean_ratio)/std_ratio
z_score[abs(z_score) >= 3] = np.nan
print(np.sum(z_score[abs(z_score) >= 3].count())==0)
z_score = z_score*std_ratio+mean_ratio #getting back to normal values
multi_ = z_score
#filling nan
multi_=multi_.replace([np.inf, -np.inf], np.nan) #should not be necessary
multi_.fillna(multi_.mean(), inplace=True)

#adjusting default
multi_['Default'] = medie_4y['Default']

acc_logit ={}
acc_SVM ={}
acc_KNN ={}
acc_AdaBoost={}
acc_DecisionTree={}
acc_XGboost={}
```

```
test_logit =[]
test_SVM =[]
test_KNN =[]
test_AdaBoost=[]
test_DecisionTree=[]
test_XGboost=[]   #it varies depending on ROC_AUC_corr

ROC_AUC = {}
to_be_u = prioritizing(ROC_AUC_corr)
for screamed in list(to_be_u):
    ROC_AUC[screamed]={}

for corr_level in list(np.around(list(np.arange(0.3,0.9,corr_range)),4)):
    to_be_used = prioritizing(corr_level)
    for screamed in list(to_be_used):

        if corr_level == 0.3: #only creating once
            acc_logit[screamed] =[]
            acc_SVM[screamed] =[]
            acc_KNN[screamed] =[]
            acc_AdaBoost[screamed] = []
            acc_DecisionTree[screamed] = []
            acc_XGboost[screamed] = []
```

```
        X= np.array(multi_[to_be_used[screamed]])
        #feature scaling
        min_max_scaler = preprocessing.MinMaxScaler()
        X_scaled = min_max_scaler.fit_transform(X)
        X = pd.DataFrame(X_scaled)
        y= multi_['Default']
        dial_t = 0.25

        #Logistic regression
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        model = LogisticRegression(solver='lbfgs')
        model.fit(x_train, y_train)
        y_pred = model.predict_proba(x_test)
        # keep probabilities for the positive outcome only
        y_pred = y_pred[:, 1]
        fpr, tpr, thresholds = roc_curve(y_test, y_pred)
        # get the best threshold
        J = tpr - fpr
        ix = np.argmax(J)
        best_thresh = thresholds[ix]
        y_pred = pd.DataFrame(y_pred)
        y_pred['new_thrsld'] = np.where(y_pred >= best_thresh, 1, 0) #new threshold set (sklearn logit has default of 0.5)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['Logit'] = [x_test, y_test, model]
            test_logit.append(model)
        acc_logit[screamed].append(metrics.accuracy_score(y_test, y_pred['new_thrsld']))
```

```
        # SVM
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        rbf_svc = svm.SVC(C=150,kernel='poly')
        rbf_svc.fit(x_train, y_train)
        y_pred = rbf_svc.predict(x_test)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['SVM'] = [x_test,y_test,rbf_svc]
            test_SVM.append(rbf_svc)
        acc_SVM[screamed].append(metrics.accuracy_score(y_test, y_pred))

        # KNN
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        knn = KNeighborsClassifier(n_neighbors=12)
        knn.fit(x_train, y_train)
        y_pred = knn.predict(x_test)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['KNN'] = [x_test,y_test, knn]
            test_KNN.append(knn)
        acc_KNN[screamed].append(metrics.accuracy_score(y_test, y_pred))

        #AdaBoost
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        ada = AdaBoostClassifier(n_estimators=150, random_state=0)
        ada.fit(x_train, y_train)
        y_pred = ada.predict(x_test)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['AdaBoost'] = [x_test,y_test,ada]
            test_AdaBoost.append(ada)
        acc_AdaBoost[screamed].append(metrics.accuracy_score(y_test, y_pred))
```

```
priorityz=prioritizing(0.7)
testing_models={'logit':test_logit, 'svm':test_SVM, 'knn': test_KNN, 'ada':test_AdaBoost, 'tree':test_DecisionTree,
                'xgb':test_XGboost}
%store testing_models
%store priorityz

print('loading plots...')

#one chart per method
methods = [acc_logit, acc_SVM, acc_KNN, acc_AdaBoost, acc_DecisionTree, acc_XGboost]
names=['Logit', 'SVM', 'KNN', 'AdaBoost', 'Decision Tree', 'XGboost']
for method in methods:
    x=[]
    y=list(np.arange(0.3,0.9,corr_range))
    for screamed in list(method):
        x.append(method[screamed])

    for t in range(len(x)):
        plt.plot(y,x[t], label = f"{list(method)[t]}")

    # Show/save figure as desired.
    plt.xlabel('corr thresholds')
    plt.ylabel('accuracy in prediction')
    plt.title(f'accuracy vs corr thresholds: {names[methods.index(method)]}')
    plt.legend()
    plt.show()
```

```
        #Decision Tree
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        tree = DecisionTreeClassifier(random_state=0)
        tree.fit(x_train, y_train)
        y_pred = tree.predict(x_test)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['Decision Tree'] = [x_test,y_test,tree]
            test_DecisionTree.append(tree)
        acc_DecisionTree[screamed].append(metrics.accuracy_score(y_test, y_pred))

        #XGboost
        x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= dial_t, random_state=0)
        xgbo = XGBClassifier(n_estimators=100, objective="binary:logistic", random_state=42)
        xgbo.fit(x_train, y_train)
        y_pred = xgbo.predict(x_test)
        if corr_level==ROC_AUC_corr:
            ROC_AUC[screamed]['XGboost'] = [x_test,y_test,xgbo]
            test_XGboost.append(xgbo)
        acc_XGboost[screamed].append(metrics.accuracy_score(y_test, y_pred))
        #print(metrics.accuracy_score(y_test, y_pred))
```

```
#ROC AUC
for prior_mod in list(ROC_AUC):

    for pred_model in list(ROC_AUC[prior_mod]):
        classifier = ROC_AUC[prior_mod][pred_model][2]
        y_test = ROC_AUC[prior_mod][pred_model][1]
        x_test = ROC_AUC[prior_mod][pred_model][0]
        #fpr, tpr, thresholds = roc_curve(y_test,y_pred)
        #roc_auc = auc(fpr, tpr)
        #plt.plot(fpr,tpr, label = f"{pred_model}: {round(roc_auc,2)}")
        metrics.plot_roc_curve(classifier, x_test, y_test)
        plt.plot([0, 1], [0, 1],'r--')
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title(f'ROC AUC : {prior_mod} : correlation threshold {ROC_AUC_corr}')
        plt.legend()
```

Code to test for hold-out sample

```
%store -r testing_models
%store -r priorityz
```

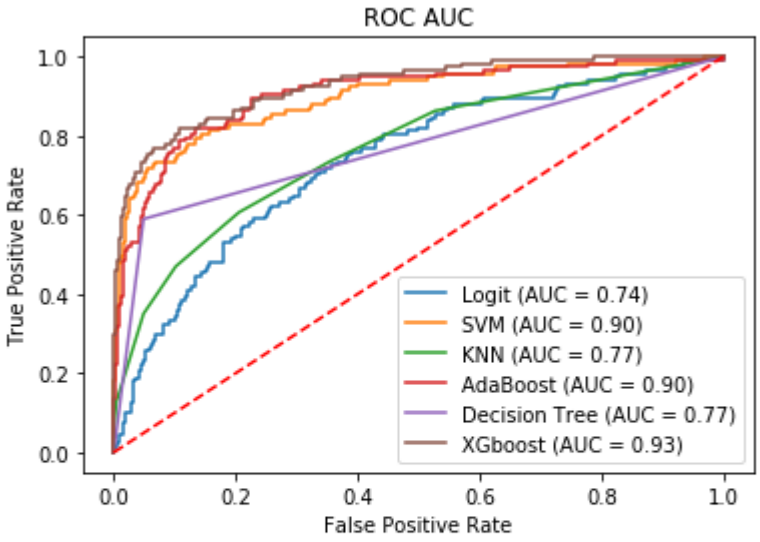
```
#preprocessing means
multi_ = medie_4y
#preprocessing outliars
mean_ratio = np.mean(multi_)
std_ratio = np.std(multi_)
z_score = (multi_-mean_ratio)/std_ratio
z_score[abs(z_score) >= 3] = np.nan
print(np.sum(z_score[abs(z_score) >= 3].count())==0)
z_score = z_score*std_ratio+mean_ratio #getting back to normal values
multi_ = z_score
#filling nan
multi_=multi_.replace([np.inf, -np.inf], np.nan) #should not be necessary
multi_.fillna(multi_.mean(), inplace=True)

fin_test_acc={}
fin_test_acc['logit']=[]
fin_test_acc['svm']=[]
fin_test_acc['knn']=[]
fin_test_acc['ada']=[]
fin_test_acc['tree']=[]
fin_test_acc['xgb']=[]
```

```
count=0
for priory in list(priorityz):
    X= np.array(multi_[priorityz[priory]])
    #feature scaling
    min_max_scaler = preprocessing.MinMaxScaler()
    X_scaled = min_max_scaler.fit_transform(X)
    X = pd.DataFrame(X_scaled)
    y= multi_['Default']

    #Logistic regression
    y_pred = testing_models['logit'][count].predict_proba(X)
    y_pred = y_pred[:, 1]
    fpr, tpr, thresholds = roc_curve(y, y_pred)
    # get the best threshold
    J = tpr - fpr
    ix = np.argmax(J)
    best_thresh = thresholds[ix]
    y_pred = pd.DataFrame(y_pred)
    y_pred['new_thrsld'] = np.where(y_pred >= best_thresh, 1, 0) #new threshold set (sklearn logit has default of 0.5)
    fin_test_acc['logit'].append(metrics.accuracy_score(y, y_pred['new_thrsld']))

    # SVM
    y_pred = testing_models['svm'][count].predict(X)
    fin_test_acc['svm'].append(metrics.accuracy_score(y, y_pred))

    # KNN
    y_pred = testing_models['knn'][count].predict(X)
    fin_test_acc['knn'].append(metrics.accuracy_score(y, y_pred))
```

```
    # ADABOOST
    y_pred = testing_models['ada'][count].predict(X)
    fin_test_acc['ada'].append(metrics.accuracy_score(y, y_pred))

    # decision tree
    y_pred = testing_models['tree'][count].predict(X)
    fin_test_acc['tree'].append(metrics.accuracy_score(y, y_pred))

    # XGBoost
    y_pred = testing_models['xgb'][count].predict(X)
    fin_test_acc['xgb'].append(metrics.accuracy_score(y, y_pred))

    count+=1


fin_test_acc=pd.DataFrame(fin_test_acc)
fin_test_acc
```

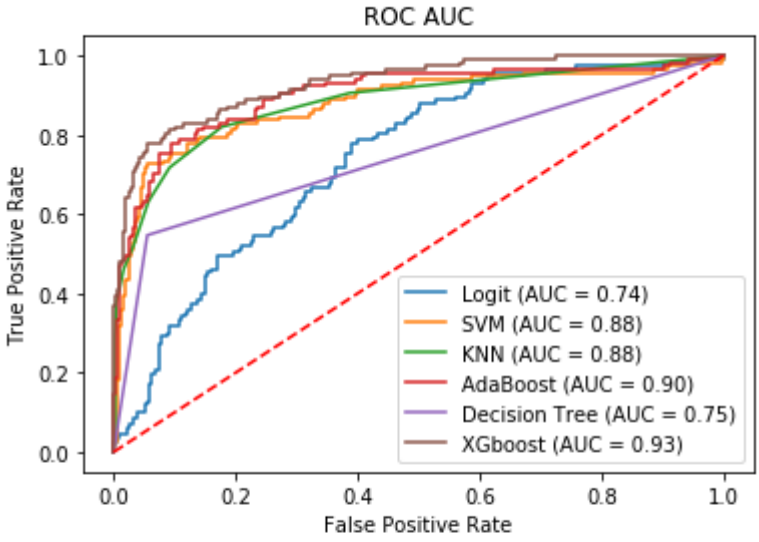
**APPENDIX 9** ROC AUCs retrieved from prediction models for one-year distance

Priority list: literature; Correlation threshold: 0.3



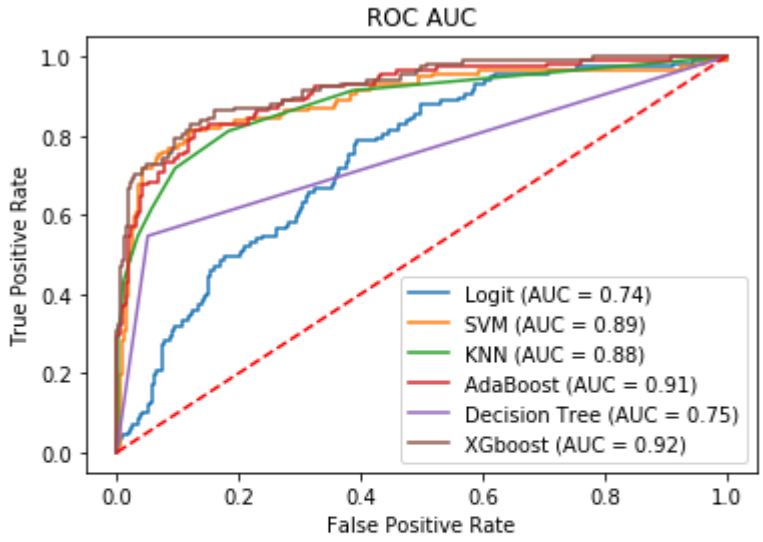

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.666118 | 0.618744 | 0.685858 |
| SVM | 0.893092 | 0.893464 | 0.745008 |
| KNN | 0.828947 | 0.854882 | 0.562066 |
| AdaBoost | 0.876645 | 0.825767 | 0.744591 |
| Decision Tree | 0.881579 | 0.824366 | 0.770432 |
| XGboost | 0.914474 | 0.897434 | 0.813585 |

Priority list: Precision; Correlation threshold: 0.3



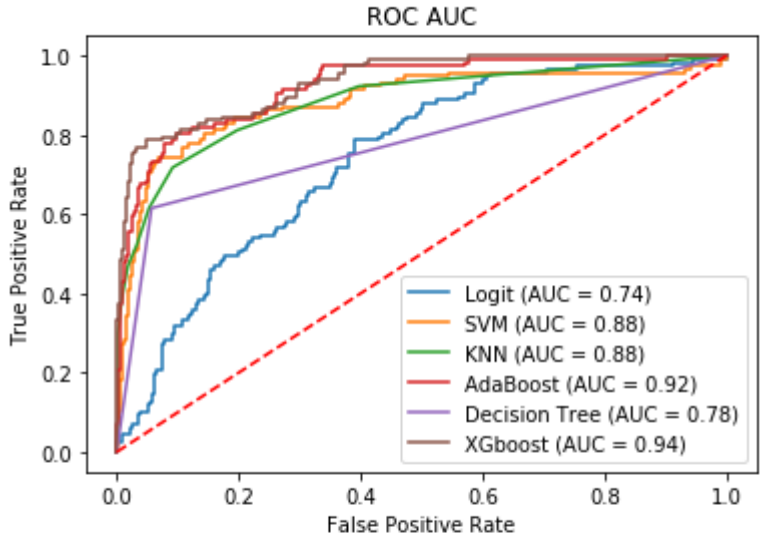

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.636513 | 0.62066 | 0.69357 |
| SVM | 0.879934 | 0.855283 | 0.727096 |
| KNN | 0.878289 | 0.897948 | 0.696781 |
| AdaBoost | 0.893092 | 0.8477 | 0.787326 |
| Decision Tree | 0.868421 | 0.800391 | 0.746009 |
| XGboost | 0.914474 | 0.894049 | 0.81684 |

Priority list: Recall; Correlation threshold: 0.3



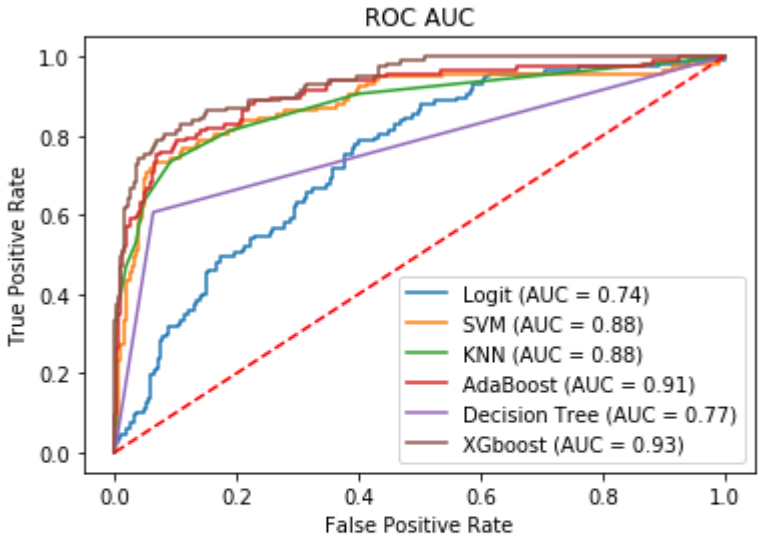

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.641447 | 0.622704 | 0.696625 |
| SVM | 0.886513 | 0.867525 | 0.740935 |
| KNN | 0.881579 | 0.893967 | 0.708584 |
| AdaBoost | 0.896382 | 0.854103 | 0.792618 |
| Decision Tree | 0.871711 | 0.808491 | 0.748046 |
| XGboost | 0.909539 | 0.896249 | 0.797509 |

Priority list: Binning; Correlation threshold: 0.3



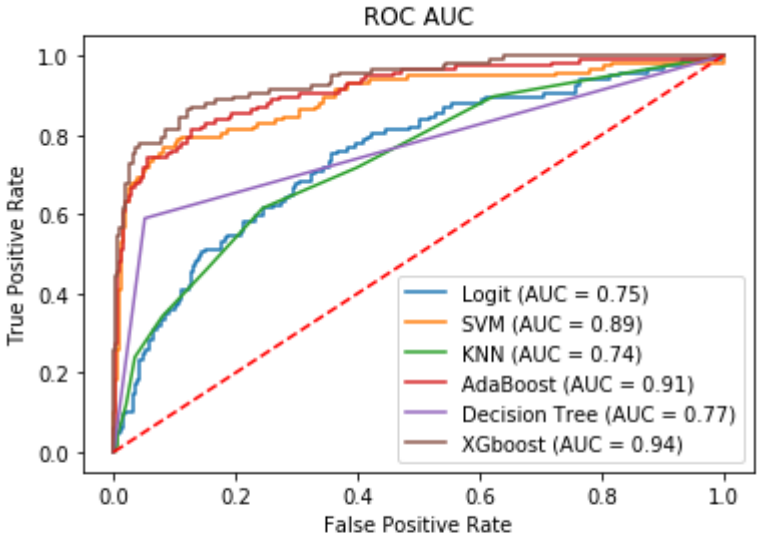

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.643092 | 0.623392 | 0.697643 |
| SVM | 0.878289 | 0.849211 | 0.726078 |
| KNN | 0.876645 | 0.888973 | 0.695763 |
| AdaBoost | 0.902961 | 0.864372 | 0.806456 |
| Decision Tree | 0.879934 | 0.815709 | 0.779179 |
| XGboost | 0.917763 | 0.907862 | 0.815621 |

Priority list: IV; Correlation threshold: 0.3



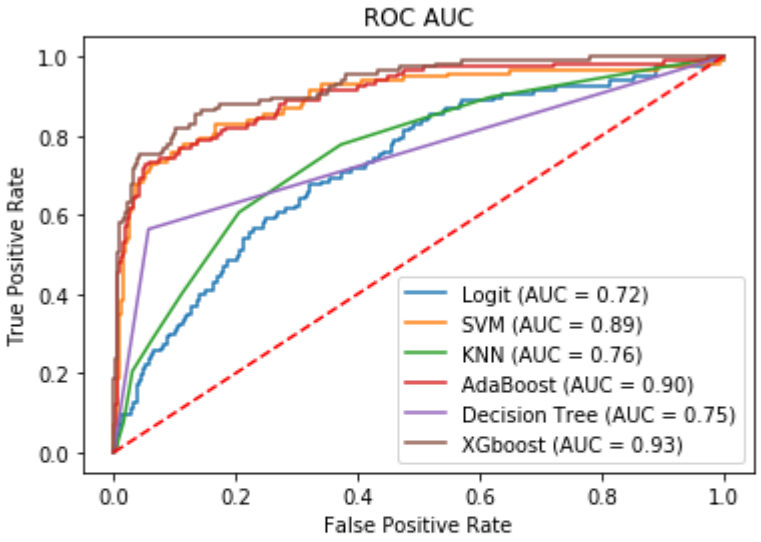

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.636513 | 0.62066 | 0.69357 |
| SVM | 0.871711 | 0.840147 | 0.708984 |
| KNN | 0.875 | 0.894783 | 0.688234 |
| AdaBoost | 0.894737 | 0.84174 | 0.80462 |
| Decision Tree | 0.873355 | 0.802585 | 0.771851 |
| XGboost | 0.912829 | 0.886122 | 0.819077 |

Priority list: literature; Correlation threshold: 0.6



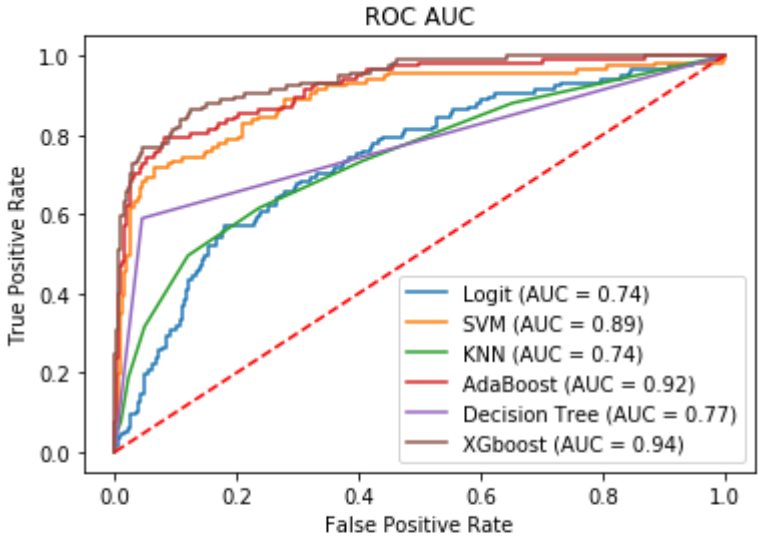

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.657895 | 0.62413 | 0.697042 |
| SVM | 0.899671 | 0.904956 | 0.758847 |
| KNN | 0.814145 | 0.703482 | 0.549646 |
| AdaBoost | 0.911184 | 0.887467 | 0.811548 |
| Decision Tree | 0.879934 | 0.820329 | 0.769414 |
| XGboost | 0.919408 | 0.90936 | 0.819895 |

Priority list: Precision; Correlation threshold: 0.6



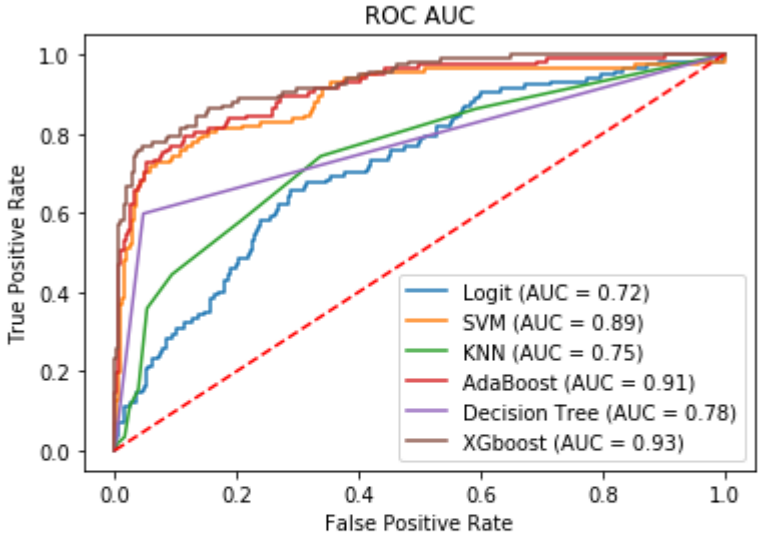

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.679276 | 0.616298 | 0.677729 |
| SVM | 0.888158 | 0.873733 | 0.741953 |
| KNN | 0.809211 | 0.673335 | 0.530315 |
| AdaBoost | 0.902961 | 0.866911 | 0.803201 |
| Decision Tree | 0.870066 | 0.801453 | 0.753538 |
| XGboost | 0.907895 | 0.884021 | 0.803001 |

Priority list: Recall; Correlation threshold: 0.6



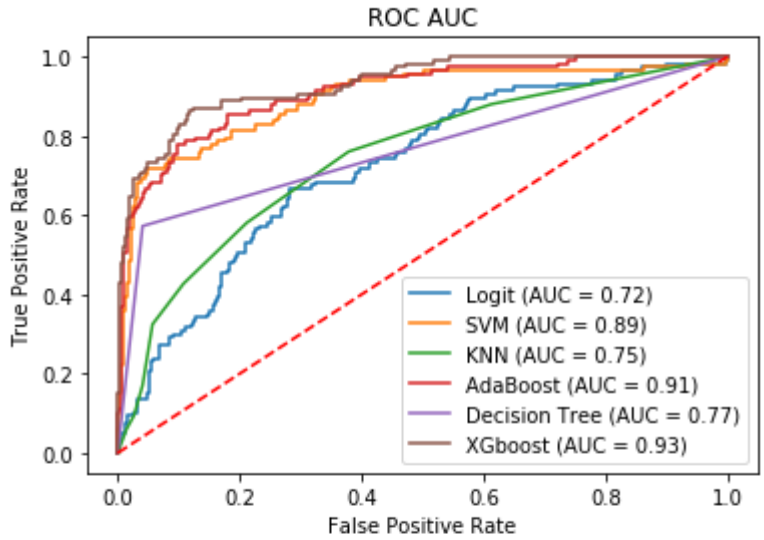

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.771382 | 0.659434 | 0.695693 |
| SVM | 0.883224 | 0.872702 | 0.725878 |
| KNN | 0.814145 | 0.730519 | 0.53337 |
| AdaBoost | 0.914474 | 0.882301 | 0.829861 |
| Decision Tree | 0.884868 | 0.832699 | 0.772469 |
| XGboost | 0.916118 | 0.899038 | 0.817858 |

Priority list: Binning; Correlation threshold: 0.6



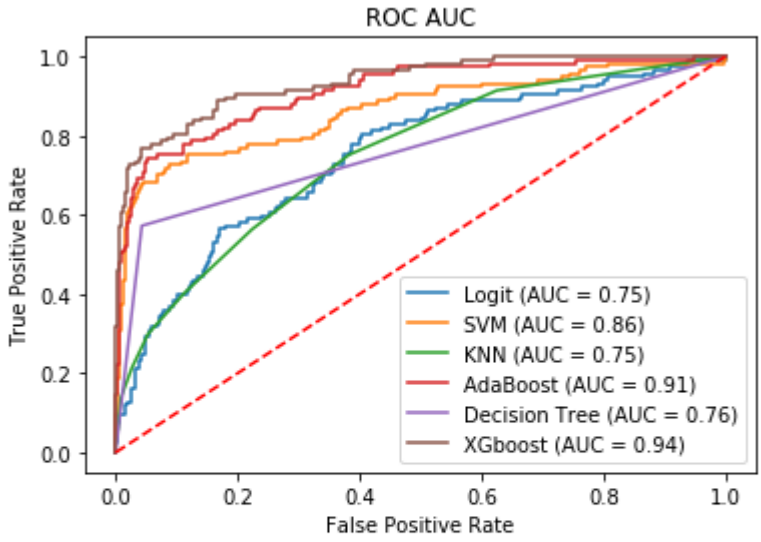

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.702303 | 0.625323 | 0.685475 |
| SVM | 0.886513 | 0.871841 | 0.73768 |
| KNN | 0.809211 | 0.670806 | 0.543336 |
| AdaBoost | 0.904605 | 0.874305 | 0.800964 |
| Decision Tree | 0.884868 | 0.830713 | 0.775724 |
| XGboost | 0.916118 | 0.902588 | 0.814603 |

Priority list: IV; Correlation threshold: 0.6



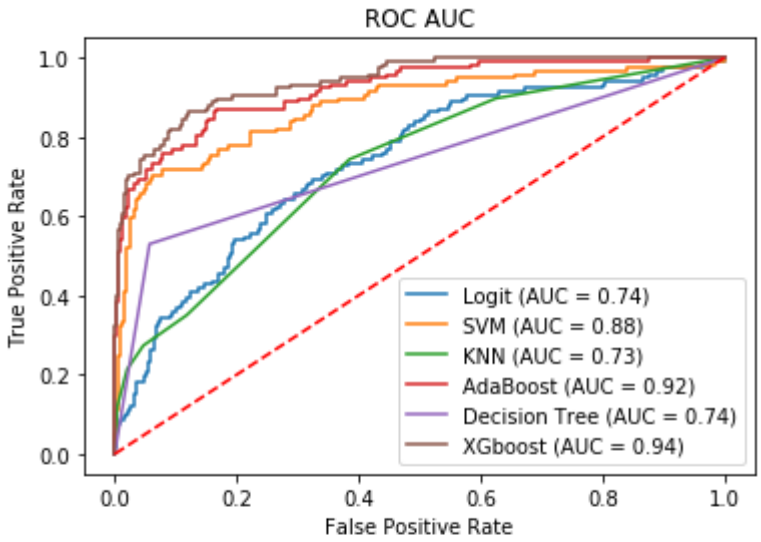

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.707237 | 0.629851 | 0.691785 |
| SVM | 0.883224 | 0.867969 | 0.729133 |
| KNN | 0.804276 | 0.638413 | 0.53377 |
| AdaBoost | 0.899671 | 0.85808 | 0.801165 |
| Decision Tree | 0.884868 | 0.837073 | 0.765958 |
| XGboost | 0.916118 | 0.895689 | 0.821113 |

Priority list: literature; Correlation threshold: 0.9



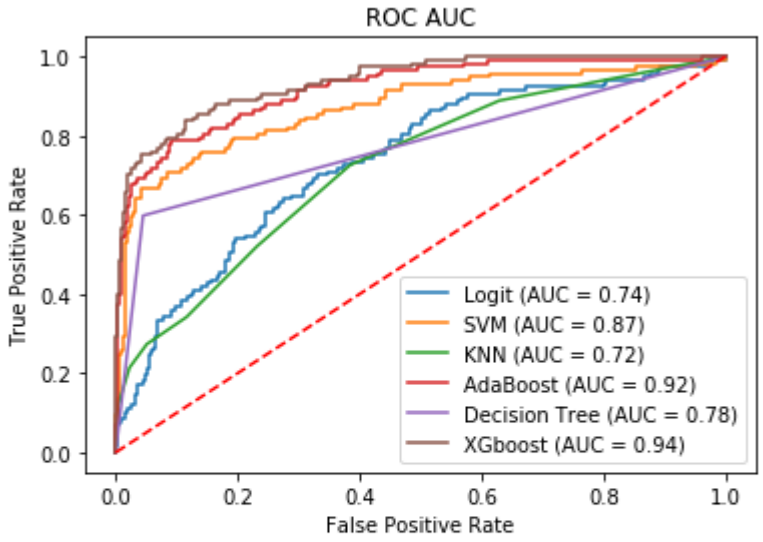

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.636513 | 0.624566 | 0.70008 |
| SVM | 0.894737 | 0.890081 | 0.752537 |
| KNN | 0.827303 | 0.814116 | 0.564303 |
| AdaBoost | 0.909539 | 0.889045 | 0.804019 |
| Decision Tree | 0.883224 | 0.832605 | 0.76494 |
| XGboost | 0.922697 | 0.912325 | 0.828442 |

Priority list: Precision; Correlation threshold: 0.9



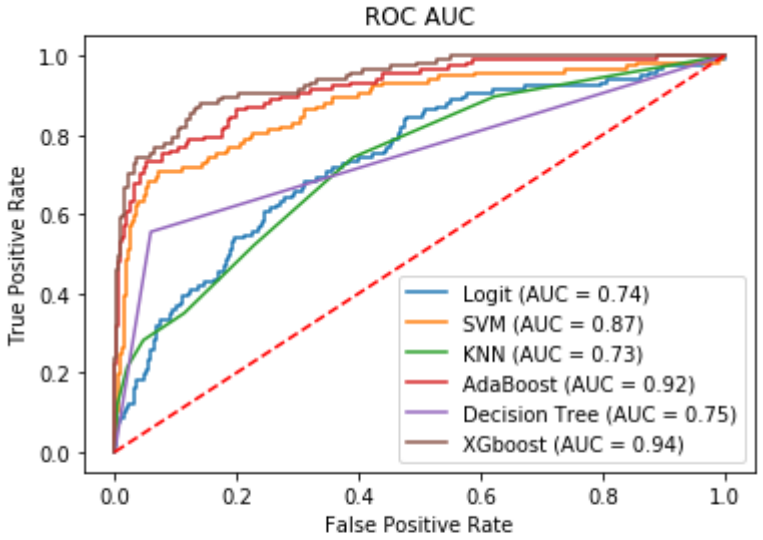

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.677632 | 0.619003 | 0.683221 |
| SVM | 0.883224 | 0.872702 | 0.725878 |
| KNN | 0.827303 | 0.850507 | 0.557792 |
| AdaBoost | 0.909539 | 0.876956 | 0.81704 |
| Decision Tree | 0.863487 | 0.791356 | 0.736444 |
| XGboost | 0.922697 | 0.90873 | 0.831697 |

Priority list: Recall; Correlation threshold: 0.9




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.675987 | 0.619956 | 0.685458 |
| SVM | 0.889803 | 0.884966 | 0.739717 |
| KNN | 0.827303 | 0.850507 | 0.557792 |
| AdaBoost | 0.914474 | 0.887838 | 0.82335 |
| Decision Tree | 0.886513 | 0.834892 | 0.776742 |
| XGboost | 0.921053 | 0.907211 | 0.827424 |

Priority list: Binning; Correlation threshold: 0.9

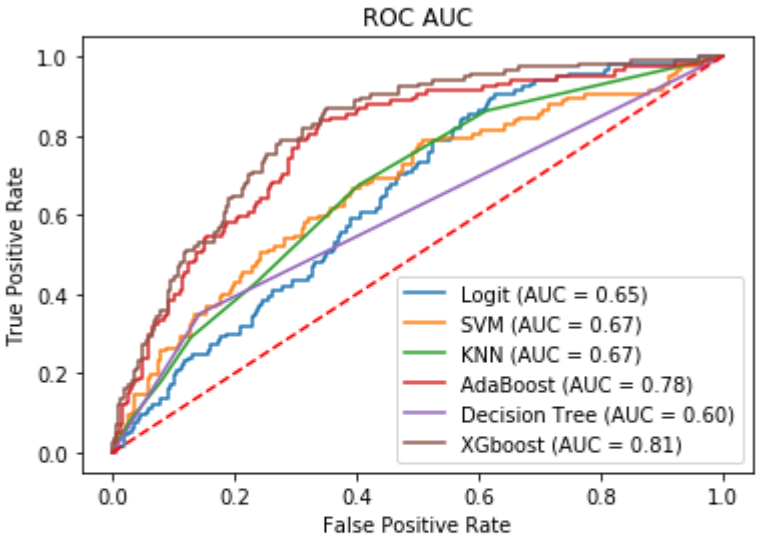



| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.689145 | 0.623212 | 0.687094 |
| SVM | 0.879934 | 0.863971 | 0.720586 |
| KNN | 0.827303 | 0.850507 | 0.557792 |
| AdaBoost | 0.909539 | 0.879719 | 0.813785 |
| Decision Tree | 0.866776 | 0.795161 | 0.748246 |
| XGboost | 0.921053 | 0.910848 | 0.824168 |

**APPENDIX 10** ROC AUCs retrieved from prediction models for two- and three-years cases

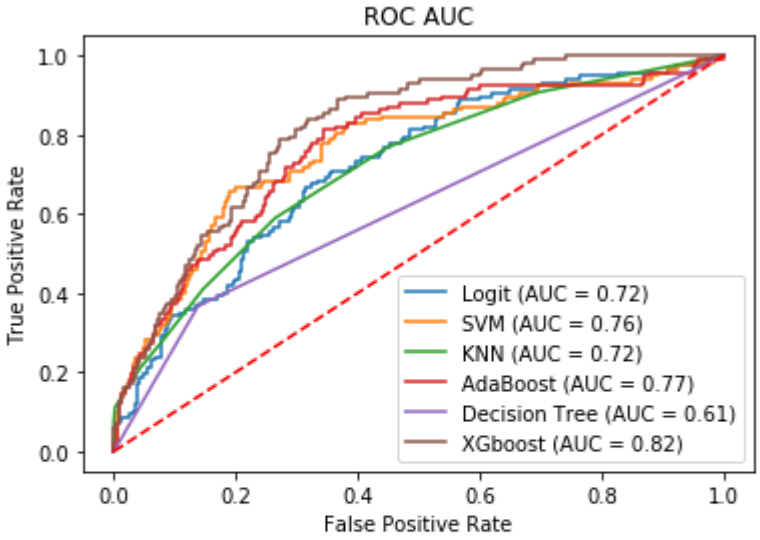
```
Priority list: accuracy; Correlation threshold: 0.3
```

Two-years distance from the relevant defaulting year




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.475329 | 0.599669 | 0.639346 |
| SVM | 0.809211 | 0.738292 | 0.507529 |
| KNN | 0.805921 | 0.627898 | 0.512002 |
| AdaBoost | 0.8125 | 0.686442 | 0.626752 |
| Decision Tree | 0.759868 | 0.608226 | 0.603931 |
| XGboost | 0.814145 | 0.689355 | 0.591963 |

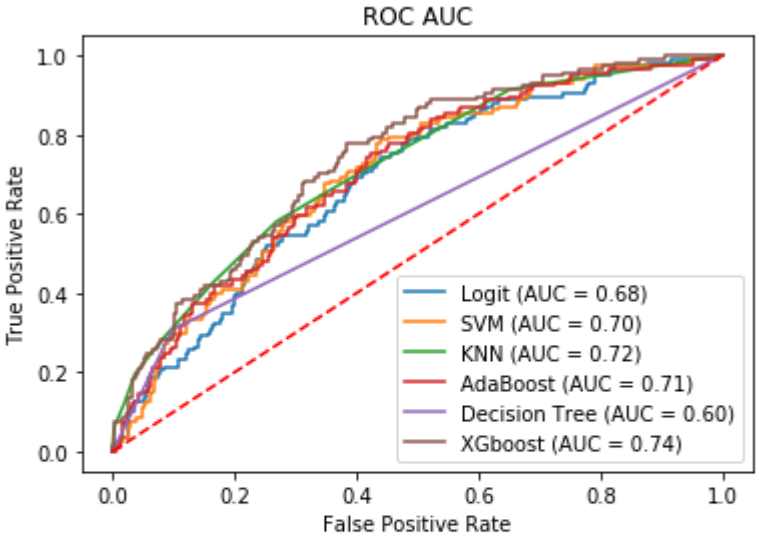
```
Priority list: accuracy; Correlation threshold: 0.9
```




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.682566 | 0.616242 | 0.676511 |
| SVM | 0.824013 | 0.727031 | 0.598073 |
| KNN | 0.820724 | 0.731892 | 0.569995 |
| AdaBoost | 0.809211 | 0.6798 | 0.62797 |
| Decision Tree | 0.766447 | 0.619247 | 0.614514 |
| XGboost | 0.810855 | 0.67953 | 0.599692 |

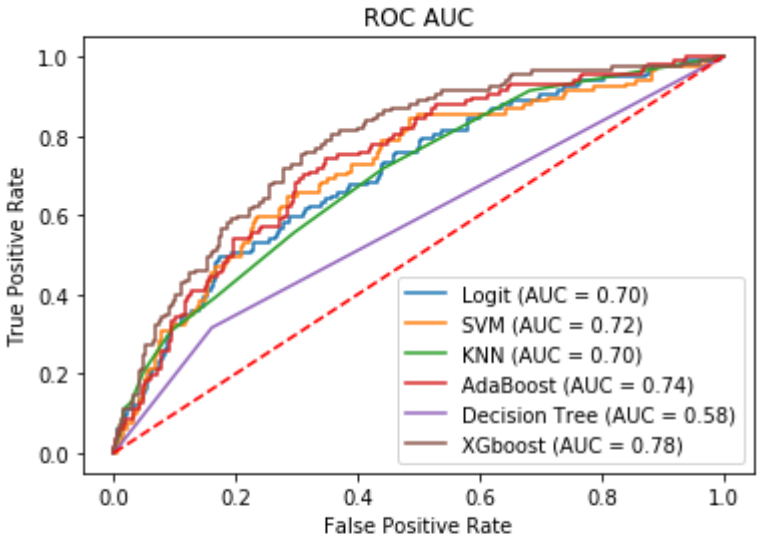
Priority list: accuracy; Correlation threshold: 0.3

Three-years distance from the relevant defaulting year




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.597039 | 0.59502 | 0.652854 |
| SVM | 0.807566 | 0.654801 | 0.50651 |
| KNN | 0.815789 | 0.716314 | 0.550664 |
| AdaBoost | 0.796053 | 0.632296 | 0.567741 |
| Decision Tree | 0.787829 | 0.636996 | 0.604966 |
| XGboost | 0.810855 | 0.680743 | 0.547609 |

Priority list: accuracy; Correlation threshold: 0.9

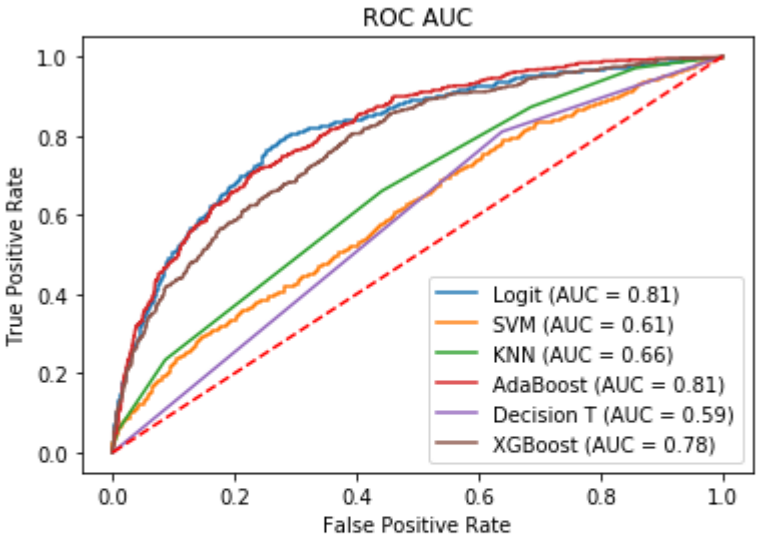



| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.763158 | 0.639356 | 0.661305 |
| SVM | 0.804276 | 0.633163 | 0.52726 |
| KNN | 0.809211 | 0.670538 | 0.549846 |
| AdaBoost | 0.791118 | 0.624392 | 0.571196 |
| Decision Tree | 0.738487 | 0.578182 | 0.577672 |
| XGboost | 0.807566 | 0.664613 | 0.565103 |

APPENDIX 11 ROC AUCs retrieved from testing, external sample

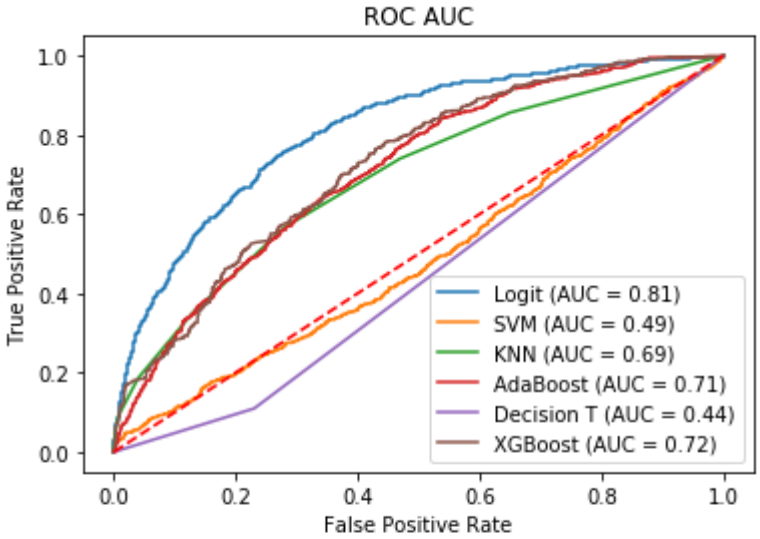
```
Priority list: literature; Correlation threshold: 0.6
```

One-year distance from the relevant defaulting year



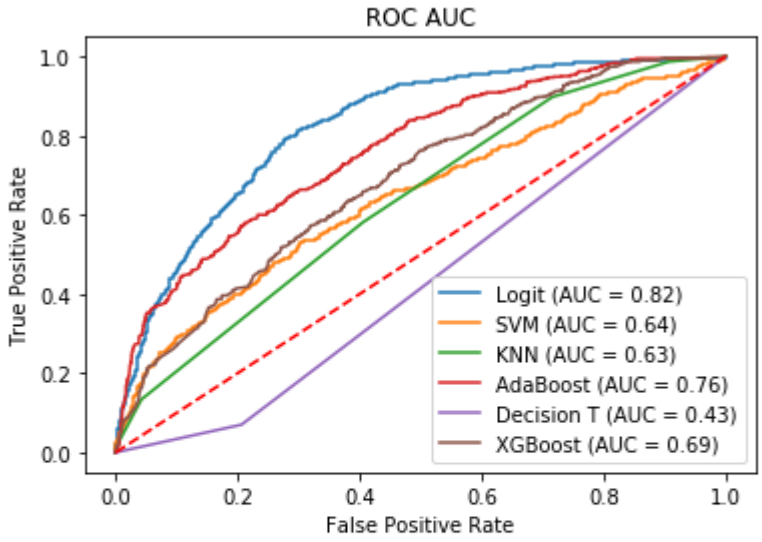

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.738656 | 0.663627 | 0.756121 |
| SVM | 0.82108 | 0.910411 | 0.503981 |
| KNN | 0.820793 | 0.910293 | 0.503185 |
| AdaBoost | 0.831132 | 0.753146 | 0.556066 |
| Decision T | 0.442275 | 0.557395 | 0.585878 |
| XGBoost | 0.708501 | 0.626884 | 0.69674 |

```
Priority list: Precision; Correlation threshold: 0.6
```



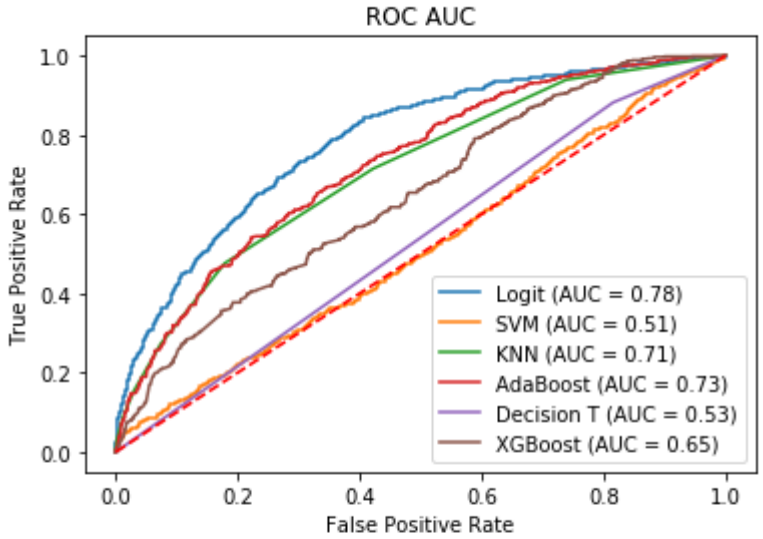

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.73521 | 0.655008 | 0.739736 |
| SVM | 0.812464 | 0.582582 | 0.512387 |
| KNN | 0.823952 | 0.882277 | 0.512564 |
| AdaBoost | 0.818782 | 0.409744 | 0.499474 |
| Decision T | 0.650488 | 0.446003 | 0.43966 |
| XGBoost | 0.283171 | 0.593239 | 0.559614 |

Priority list: Recall; Correlation threshold: 0.6



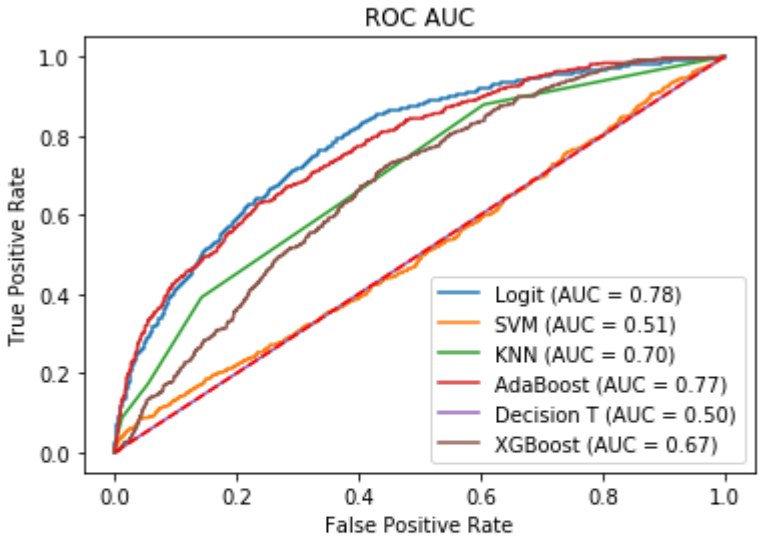

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.729179 | 0.661464 | 0.75717 |
| SVM | 0.822803 | 0.872752 | 0.509379 |
| KNN | 0.819644 | 0.659914 | 0.500621 |
| AdaBoost | 0.812177 | 0.677979 | 0.666216 |
| Decision T | 0.663125 | 0.432263 | 0.431843 |
| XGBoost | 0.368754 | 0.579891 | 0.590087 |

Priority list: Binning; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.637852 | 0.628778 | 0.717604 |
| SVM | 0.819644 | 0.661492 | 0.511178 |
| KNN | 0.823952 | 0.812062 | 0.515048 |
| AdaBoost | 0.821367 | 0.690555 | 0.519681 |
| Decision T | 0.309879 | 0.534311 | 0.533059 |
| XGBoost | 0.33946 | 0.574887 | 0.574701 |

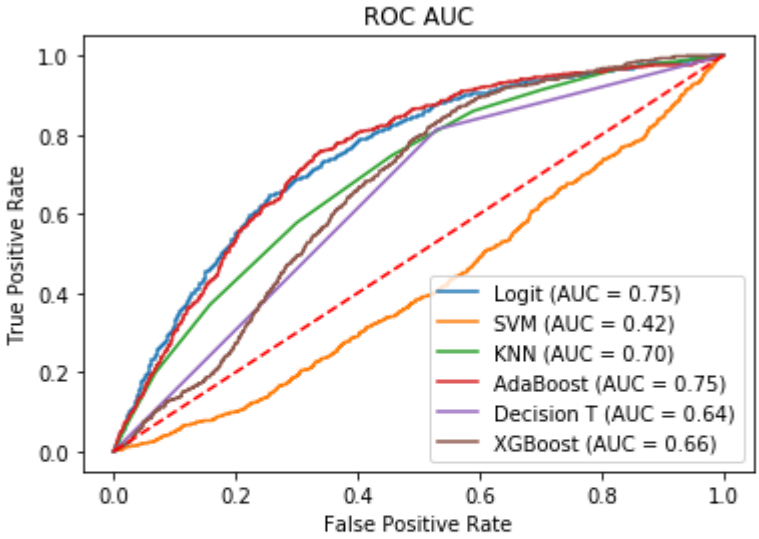
Priority list: IV; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.680356 | 0.630088 | 0.713726 |
| SVM | 0.817921 | 0.632961 | 0.513232 |
| KNN | 0.821367 | 0.848122 | 0.505398 |
| AdaBoost | 0.590465 | 0.605945 | 0.678762 |
| Decision T | 0.80672 | 0.5014 | 0.500189 |
| XGBoost | 0.258185 | 0.59024 | 0.544993 |

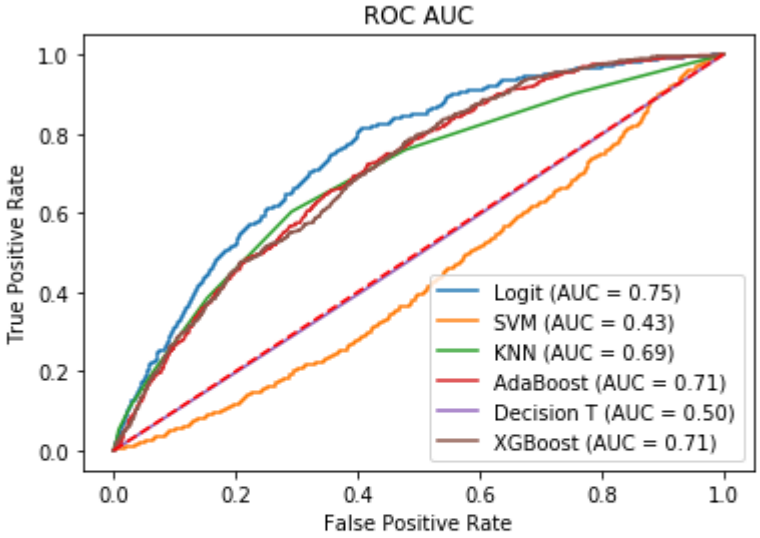
Priority list: literature; Correlation threshold: 0.6

Two-years distance from the relevant defaulting year



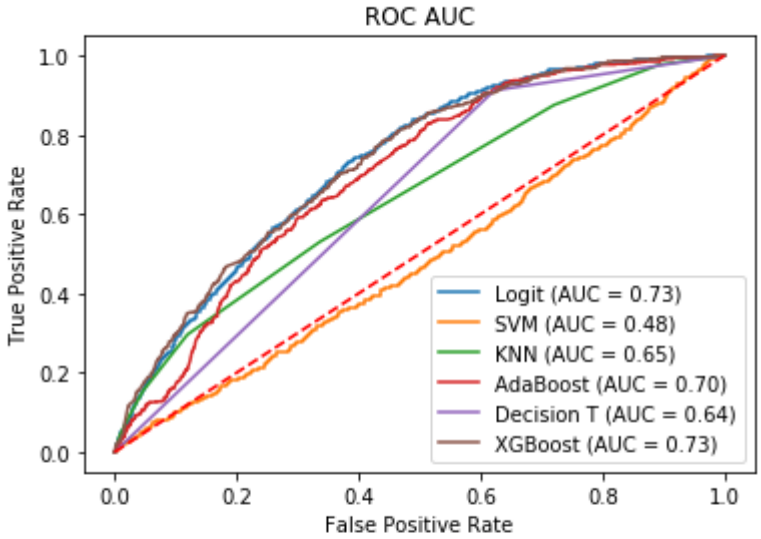

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.70046 | 0.624084 | 0.69494 |
| SVM | 0.215681 | 0.463091 | 0.483669 |
| KNN | 0.79954 | 0.615774 | 0.56536 |
| AdaBoost | 0.580414 | 0.611119 | 0.686292 |
| Decision T | 0.532165 | 0.586202 | 0.641956 |
| XGBoost | 0.458644 | 0.596382 | 0.638712 |

Priority list: Precision; Correlation threshold: 0.6



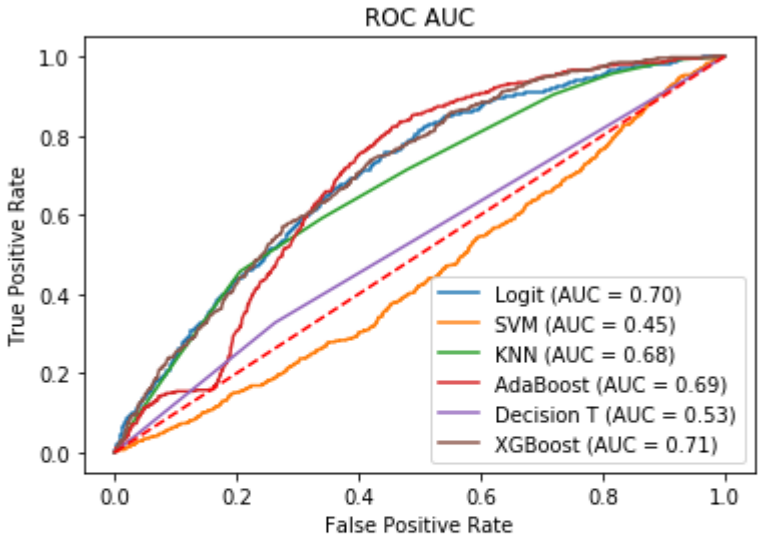

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.63297 | 0.620134 | 0.702827 |
| SVM | 0.576967 | 0.460526 | 0.441384 |
| KNN | 0.822516 | 0.707807 | 0.522244 |
| AdaBoost | 0.813613 | 0.598927 | 0.515572 |
| Decision T | 0.558013 | 0.498093 | 0.496888 |
| XGBoost | 0.481333 | 0.591678 | 0.639512 |

Priority list: Recall; Correlation threshold: 0.6



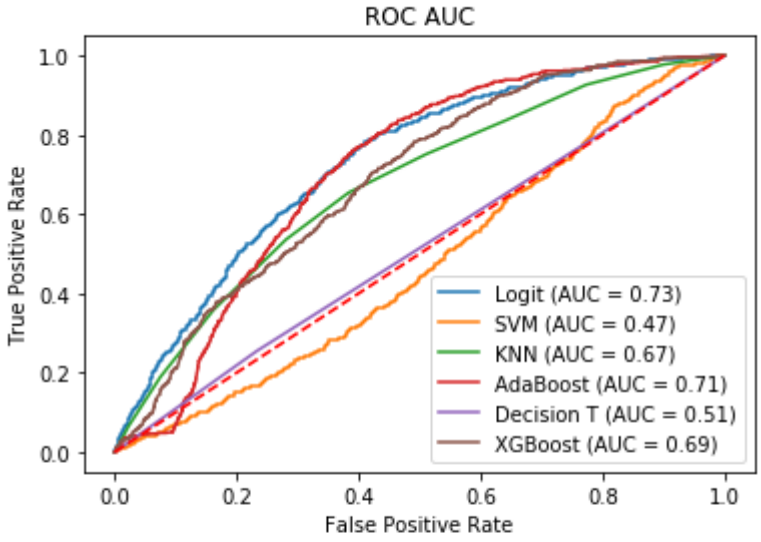

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.582998 | 0.604184 | 0.675449 |
| SVM | 0.483343 | 0.489214 | 0.481766 |
| KNN | 0.820505 | 0.678185 | 0.51605 |
| AdaBoost | 0.818495 | 0.559994 | 0.501162 |
| Decision T | 0.473866 | 0.597622 | 0.644893 |
| XGBoost | 0.770534 | 0.61481 | 0.617216 |

Priority list: Binning; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.549397 | 0.594462 | 0.656814 |
| SVM | 0.310167 | 0.482221 | 0.479208 |
| KNN | 0.733774 | 0.599025 | 0.62584 |
| AdaBoost | 0.81907 | 0.624609 | 0.503376 |
| Decision T | 0.663125 | 0.524064 | 0.532443 |
| XGBoost | 0.8139 | 0.585689 | 0.510779 |

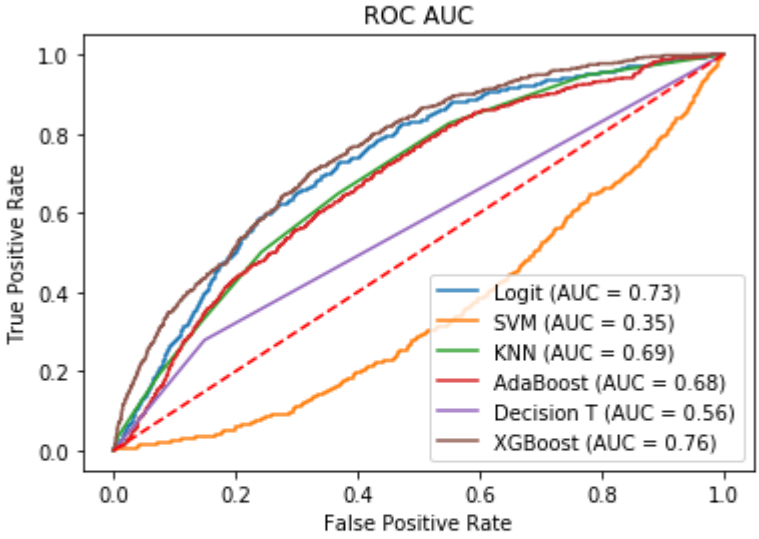
Priority list: IV; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.623779 | 0.610214 | 0.686044 |
| SVM | 0.802412 | 0.471671 | 0.495698 |
| KNN | 0.75158 | 0.591662 | 0.598201 |
| AdaBoost | 0.784032 | 0.488913 | 0.496285 |
| Decision T | 0.674612 | 0.5089 | 0.510885 |
| XGBoost | 0.641011 | 0.576018 | 0.622658 |

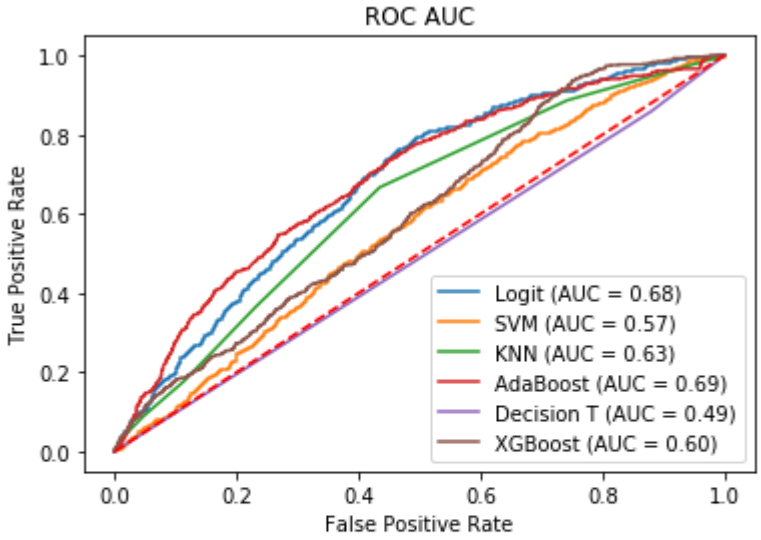
Priority list: literature; Correlation threshold: 0.6

Three-years distance from the relevant defaulting year



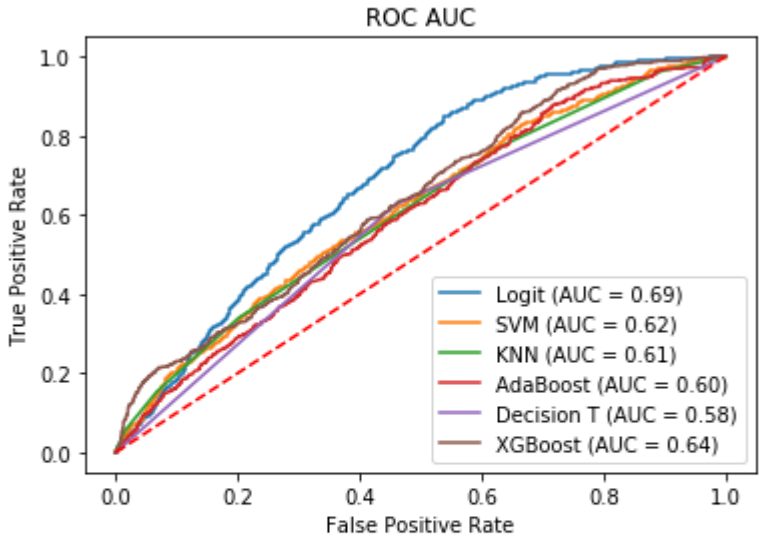

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.645893 | 0.606531 | 0.677178 |
| SVM | 0.265939 | 0.43147 | 0.424905 |
| KNN | 0.811028 | 0.612386 | 0.53014 |
| AdaBoost | 0.817346 | 0.548964 | 0.501703 |
| Decision T | 0.747846 | 0.567242 | 0.564874 |
| XGBoost | 0.823952 | 0.709043 | 0.533677 |

Priority list: Precision; Correlation threshold: 0.6



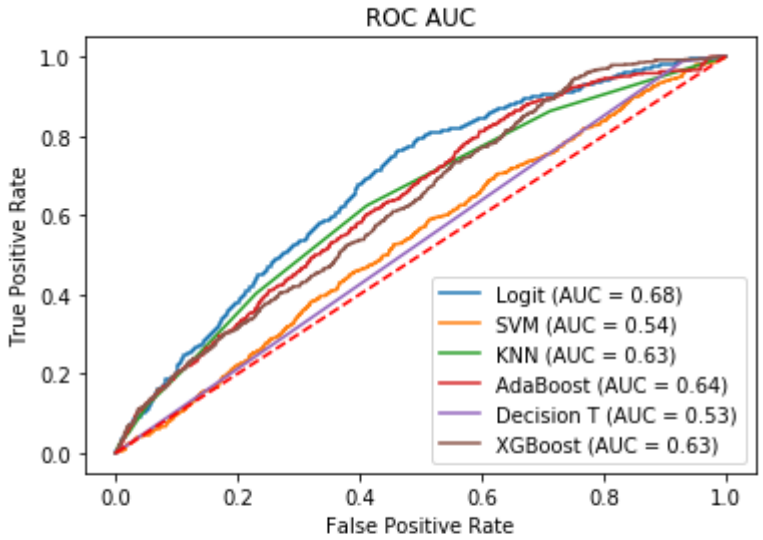

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.577829 | 0.588015 | 0.648698 |
| SVM | 0.80672 | 0.480905 | 0.497705 |
| KNN | 0.81907 | 0.624609 | 0.503376 |
| AdaBoost | 0.6973 | 0.594432 | 0.638987 |
| Decision T | 0.252728 | 0.485605 | 0.489502 |
| XGBoost | 0.547387 | 0.528576 | 0.548157 |

Priority list: Recall; Correlation threshold: 0.6



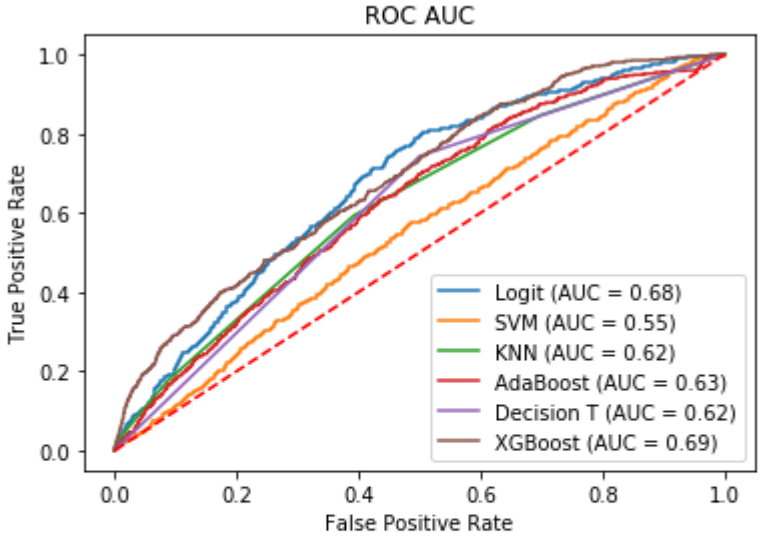

| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.531017 | 0.594594 | 0.654296 |
| SVM | 0.818208 | 0.630875 | 0.510302 |
| KNN | 0.819931 | 0.688086 | 0.50328 |
| AdaBoost | 0.810741 | 0.557732 | 0.508231 |
| Decision T | 0.556577 | 0.549441 | 0.583571 |
| XGBoost | 0.820793 | 0.678354 | 0.522435 |

Priority list: Binning; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.559736 | 0.589092 | 0.649459 |
| SVM | 0.79753 | 0.506039 | 0.501414 |
| KNN | 0.819357 | 0.651739 | 0.507898 |
| AdaBoost | 0.612292 | 0.553453 | 0.58713 |
| Decision T | 0.235497 | 0.57779 | 0.52929 |
| XGBoost | 0.702757 | 0.548671 | 0.559725 |

Priority list: IV; Correlation threshold: 0.6




| Model | Accuracy | Precision | Recall |
|---|---|---|---|
| Logit | 0.550258 | 0.589926 | 0.649888 |
| SVM | 0.79753 | 0.511005 | 0.502656 |
| KNN | 0.819644 | 0.661304 | 0.509936 |
| AdaBoost | 0.540781 | 0.557459 | 0.596912 |
| Decision T | 0.543653 | 0.572494 | 0.62164 |
| XGBoost | 0.822229 | 0.681561 | 0.563675 |