ISOM 3360 Project Final Report

Group 12: CHIU, Hsuan, CHOR, Ka Ching Skye, LAU, Pak Hei, LAW, Yuen Ning

1. Introduction

(a) Background

Bankruptcy is one of the most common reasons causing firm closure. Over 20,000 of US firms face the bankruptcy issue annually. Prediction of bankruptcy becomes crucial to create early warnings for investors and policy makers. With the prediction, they can take proactive measures to minimize the negative impacts of bankruptcies and protect their financial assets.

(b) Methodology

Our group aims to leverage the existing dataset about previous firms' bankruptcy status to predict the likelihood of bankruptcy of existing companies. With the input of financial metrics and company information to the trained models, a classification about whether the company will go bankruptcy within a specified period (e.g., a year) will be predicted.

(c) Potential Impact

Successful classification models may benefit:

- Investors – to prevent entering long positions in companies with high likelihood of going bankrupt soon, which can help reduce potential drastic loss in portfolios.
- Policy makers – to allocate resources efficiently towards industry group(s) with significant portion of companies which have high likelihood of going bankrupt in the near future (given that such industry group(s) are still valuation to the economy or the society).

2. Data Understanding

(a) Brief description

Our team obtained the Polish Companies Bankruptcy Data Set from UCI Machine Learning Repository (Link: http://archive.ics.uci.edu/ml/datasets/Polish+companies+bankruptcy+data).

There are 5 different sets of data, with bankruptcy status of Polish companies throughout different timeframes. “year 1” dataset includes the bankruptcy status after 5 years; “year 2” dataset includes the bankruptcy status after 4 years, and so on.

(b) Summary of records and attributes (For details, please refer to Appendix 1)

Records	• Around 5,000 – 10,000 instances for each dataset of a particular timeframe • Each record represents a company, with attributes and bankruptcy status
Attributes	• Each dataset contains 64 features (Attr1 – Attr64) • All features are multivariate, representing a specific financial metrics, e.g., debt ratio, which provides insights into operational efficiency of companies

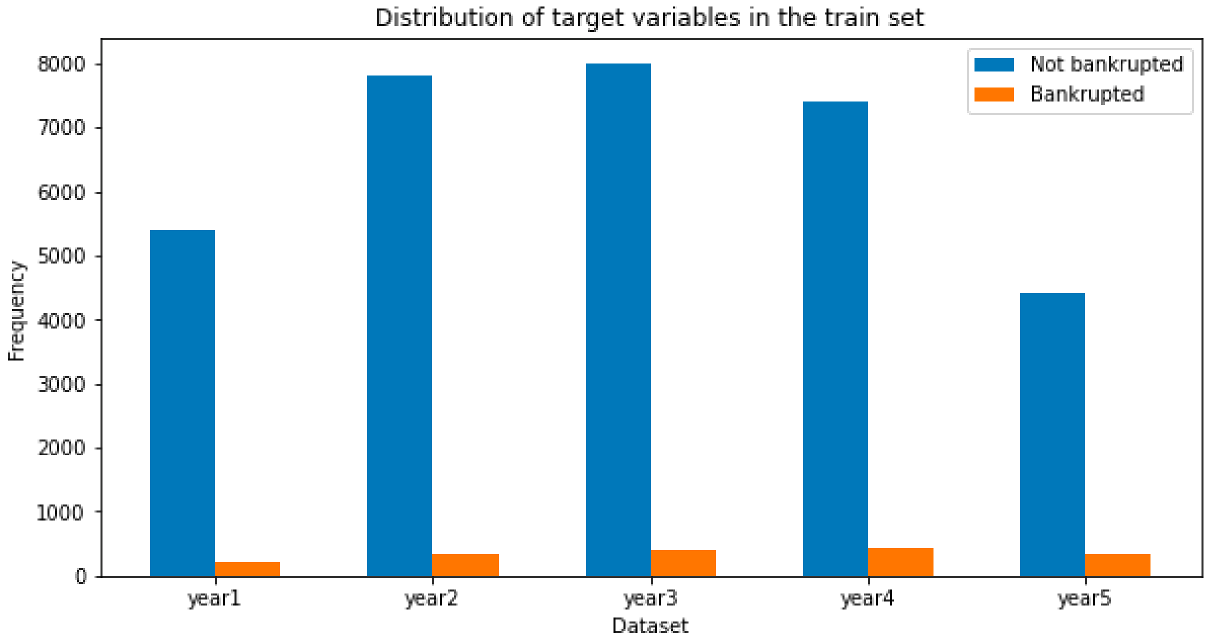
(c) Missing values

The issue of missing values is quite serious. Take Attr35 in “year 1” dataset as an illustration, among 5621 total entries, only 3435 have non-null values, while 2186 of them have NaN value.

All the missing values are filled with corresponding attribute mean in the training set.

(d) Class imbalance

It is expected that bankrupted companies will take a significantly lower portion among all the companies. The distribution of bankrupted / non-bankrupted in the train set is plotted below:



Obviously from the above plot, number of companies which are bankrupted take up less than 10% of all records for all datasets. Bankrupted class (orange) is the minority group when compared to the non-bankrupted class (blue).

Since our interest mainly on the bankrupted class (positive), evaluation metrics such as precision and recall should be emphasized over accuracy during the evaluation step.

3. Model Building

Our team performed five models for this classification task, summarized as follow:

Single learner	Ensemble model
• Decision Tree Model • Logistic Regression Model • Gaussian Naïve Bayes Model	• Random Forest Model • Adaboost Classifier

Our team tried to perform normalization on all the attributes' values and find out that there is no significant impacts on the models. For probabilistic classification tasks, normalization is rather optional when compared to regression tasks, which normalization may help faster convergence.

For all the hyper-parameters, our team has utilized the library GridSearchCV to find out the "best" hyper-parameters throughout the K-fold cross validation.

The 5 trained models are discussed as follows:

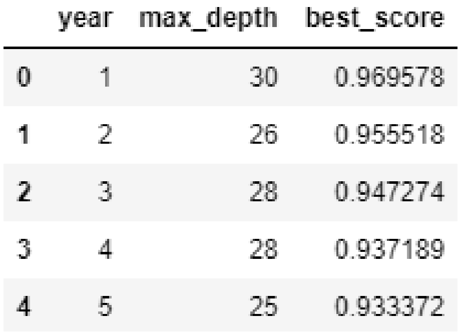
(a) Decision Tree Model

Summary

The following summarizes the training details:

Node split criterion	entropy
Hyper-parameter(s)	max_depth

The "max_depth" hyper-parameter is used to prevent the tree from growing into full depth, which may lead to serious over-fitting problem. (The "best" hyperparameter from the result of GridSearchCV is summarized on the right)

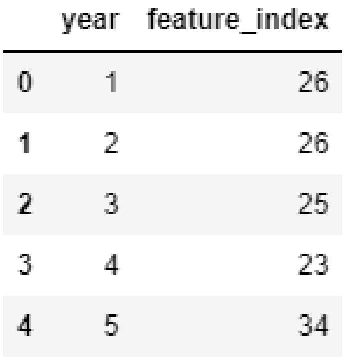


	year	max_depth	best_score
0	1	30	0.969578
1	2	26	0.955518
2	3	28	0.947274
3	4	28	0.937189
4	5	25	0.933372

Implication

Since the splitting criterion used is entropy, the root node in each trained tree can be considered as the most informative attribute for the classification task.

The information about the root node feature for the models of different timeframes is summarized by the table on the right.



	year	feature_index
0	1	26
1	2	26
2	3	25
3	4	23
4	5	34

Attr27 (since the features are 0-indexed) serves as the root node of multiple timeframes (year1 & year2), implying that Attr27 may be an important indicator of whether a company will go bankruptcy in the near future.

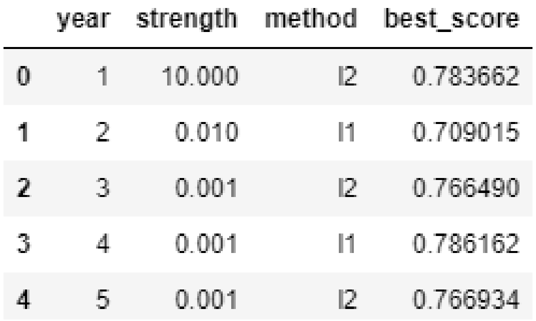
Attr27 corresponds to the financial metrics $\frac{profit\ on\ operating\ activites}{financial\ expenses}$, which is a similar profitability measure as the profit margin. According to finance experts, profitability is major key for companies to survive in the long run. The insights resonate with the implication obtained from our decision tree model.

(b) Logistic Regression Model

Summary

The following summaries the training details:

Solver	saga
Max_iter	1000
Hyper-parameter(s)	• C: regularization strength • penalty: regularization method



	year	strength	method	best_score
0	1	10.000	l2	0.783662
1	2	0.010	l1	0.709015
2	3	0.001	l2	0.766490
3	4	0.001	l1	0.786162
4	5	0.001	l2	0.766934

The "saga" solver is used for the optimization problem, since it runs much faster than "liblinear" solver, and it supports both L1 (Lasso) and L2 (Ridge) regularizations.

Max_iter is set to 1000, limiting the number of iterations taken for the regularization solvers to converge. The major use of this parameter is to prevent the process from running too long.

Two hyper-parameters are used:

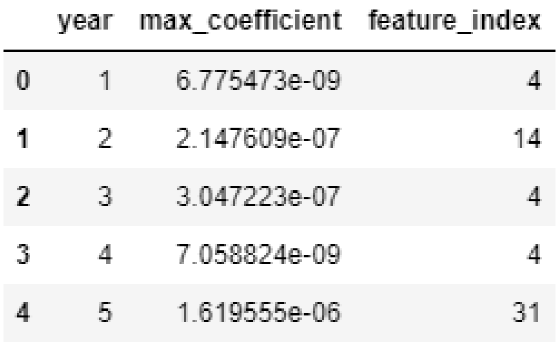
(i) C – regularization strength, smaller values of C specify strong regularization. Performing regularization helps picking up informative attributes and "dropping" irrelevant and uninformative features (since they will have zero coefficient).

(ii) Penalty – regularization method. "l1" stands for Lasso regularization, "l2" stands for Ridge regularization. GridSearchCV can help find out which method suits the particular dataset the most.

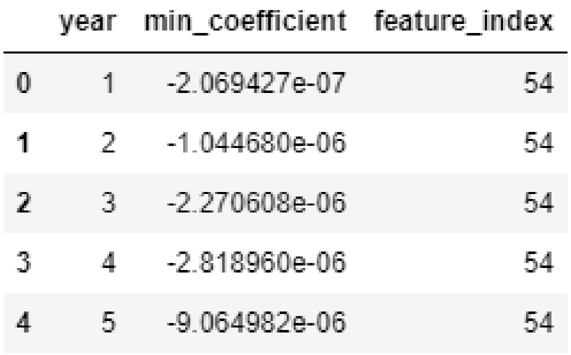
Implication

A positive coefficient means that a particular feature has a positive impact on the probability of the prediction outcome being 1; A negative coefficient means the opposite.

Our team focused on the most positive (largest) and the most negative (smallest) coefficient, as summarized by the tables below:



	year	max_coefficient	feature_index
0	1	6.775473e-09	4
1	2	2.147609e-07	14
2	3	3.047223e-07	4
3	4	7.058824e-09	4
4	5	1.619555e-06	31



	year	min_coefficient	feature_index
0	1	-2.069427e-07	54
1	2	-1.044680e-06	54
2	3	-2.270608e-06	54
3	4	-2.818960e-06	54
4	5	-9.064982e-06	54

Attr5 has the most positive coefficient in multiple timeframes (year1, year3 & year4), which corresponds to the financial metrics $\frac{cash+short\ term\ assets-short\ term\ liabilities}{operating\ expenses-depreciation}\times 365 \approx \frac{net\ working\ capital}{short\ term\ expense}\times 365$.

Unfortunately, the result does not coincide to normal heuristics that the higher the $\frac{net\ working\ capital}{short\ term\ expense}$ ratio, the lower the chance of going to bankruptcy, since higher net working capital implies a better short term financil health.

Attr55 has the most negative coefficient in all timeframes (from year1 to year5), which corresponds to the financial metrics working capital. The result does coincide with the heristics that the higher the working capital level, the lower change of going to bankruptcy, vice versa.

(c) Gaussian Naïve Bayes Model

Since all features in the datasets are multivariate, our team utilized the Gaussian Naïve Bayes Model instead of the Multinomial Naïve Bayes Model.

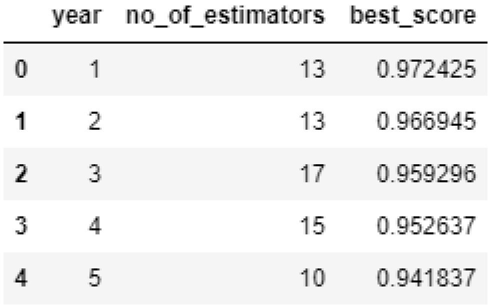
Generally there are no hyper-parameters for the Naïve Bayes Model

(d) Random Forest Model

Summary

The following summarizes the training details:

Node split criterion	entropy
Hyper-parameter(s)	n_estimators



	year	no_of_estimators	best_score
0	1	13	0.972425
1	2	13	0.966945
2	3	17	0.959296
3	4	15	0.952637
4	5	10	0.941837

The “n_estimators” parameter refers to the number of decision trees in the forest.

There is no need to prune each decision trees since bagging can deal with potential overfitting issue.

Implication

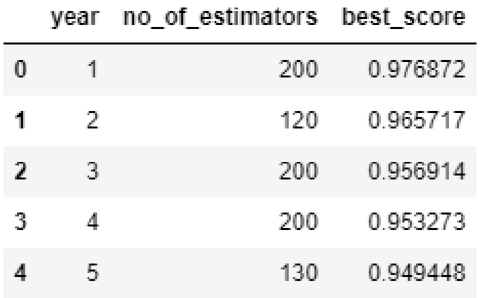
Since Random Forest Model is an ensemble model, it is considered as a “blackbox” method which is difficult to interpret the model result.

(e) Adaboost Classifier

Summary

The following summarizes the training details:

Weak learner used	Decision Tree Classifier
Max_depth of the tree	1 (i.e., 1 node)
Hyper-parameter(s)	n_estimators



	year	no_of_estimators	best_score
0	1	200	0.976872
1	2	120	0.965717
2	3	200	0.956914
3	4	200	0.953273
4	5	130	0.949448

The “n_estimators” refers to the number of rounds of boosting used in Adaboost Classifier.

Implication

Since AdaBoost Classifier is an ensemble model, it is considered as a “blackbox” method which is difficult to interpret the model result.

4. Performance Evaluation

As stated before, since the datasets are imbalanced, evaluation metrics such as precision and recall should be emphasized over the accuracy rate.

All the performance measures below are obtained by fitting the trained model upon separate test sets.

(a) Summary

The performance metrics accuracy, recall, precision, and AUC are summarized in tables below:

Accuracy

	Decision tree	Logistic regression	Naïve Bayes	Random Forest	Adaboost
Year 1	95.7%	79.2%	7.5%	96.5%	97.2%
Year 2	95.3%	68.6%	7.2%	96.8%	96.3%
Year 3	94.0%	78.4%	8.6%	95.6%	95.8%
Year 4	94.5%	80.9%	8.8%	95.6%	96.0%
Year 5	94.7%	75.2%	12.4%	94.5%	95.8%

From the accuracy metrics above, the performance of Naïve Bayes was extremely poor, with accuracy lower than 10%. It is meaningless to continue examining the performance of such a poor model, and thus our team does not include other metrics about the Naïve Bayes Model.

Recall

	Decision tree	Logistic regression	Naïve Bayes	Random Forest	Adaboost
Year 1	50.0%	22.6%		25.8%	53.2%
Year 2	50.6%	44.2%		18.2%	20.8%
Year 3	34.8%	33.7%		7.6%	22.8%
Year 4	42.9%	42.9%		13.3%	33.7%
Year 5	65.1%	60.5%		27.9%	57.0%

Precision

	Decision tree	Logistic regression	Naïve Bayes	Random Forest	Adaboost
Year 1	51.7%	5.4%		84.2%	76.7%
Year 2	41.1%	5.4%		87.5%	51.6%
Year 3	32.7%	7.3%		50.0%	55.3%
Year 4	45.2%	11.7%		92.9%	70.2%
Year 5	62.9%	16.7%		88.9%	80.3%

AUC

	Decision tree	Logistic regression	Naïve Bayes	Random Forest	Adaboost
Year 1	73.9%	58.7%		82.4%	87.2%
Year 2	73.9%	57.7%		80.8%	84.3%
Year 3	65.7%	57.2%		77.7%	85.7%
Year 4	70.1%	66.4%		87.8%	87.0%
Year 5	81.0%	76.8%		88.3%	94.4%

(b) Discussion

The following section will discuss the performance of each model one by one:

Decision Tree Model

- The model performance is satisfactory, can be considered “the best” among all models
- Accuracy higher than 90% for all 5 years, although this is not our main focus
- Precision higher than 50%, recall around 50% for all 5 years, which are significantly higher than the majority classifier (i.e., the Naïve Learner, with 0% precision and recall)

Logistic Regression Model

- The model performance is significantly worse than the Decision Tree Model.
- The underlying reason may lie in the linearity of the logistic regression model, since the decision boundary of logistic regression is linear, but a linear boundary may not be able to capture the majority pattern in the datasets.

Gaussian Naïve Bayes Model

- The model performance is extremely poor, no matter in terms of accuracy, precision, recall or AUC
- The imbalance property in the datasets may be one of the underlying reasons
- Another possible reason for such a poor performance is the violation of the “Naïve” assumption that all features are conditionally independent. The conditional independent assumption is obviously not true for different financial metrics, since financial metrics are usually correlated with one another

Radom Forest Model

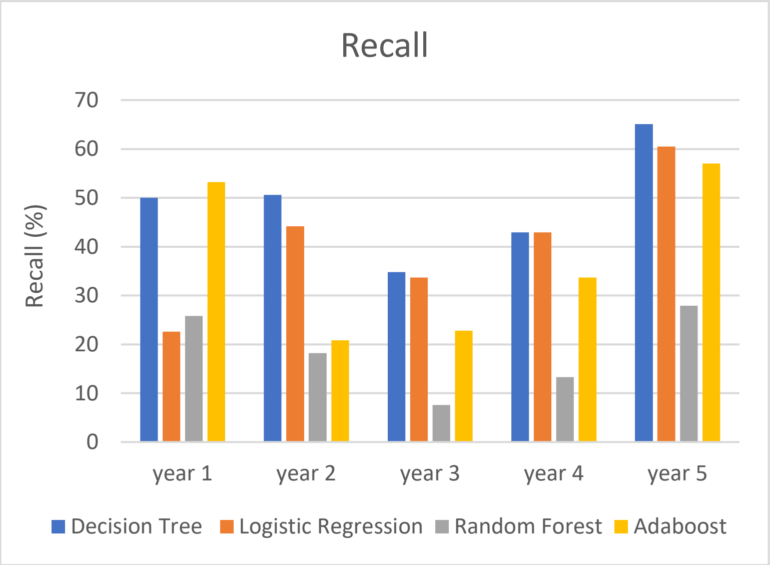
- The performance of Random Forest Model is fair
- Although it has a high accuracy, its recall is significantly lower than the single learner Decision Tree Model
- Such a surprising result may be due to the imbalance nature of the datasets, since random forest model is probably not suited to classification problems with a skewed distribution

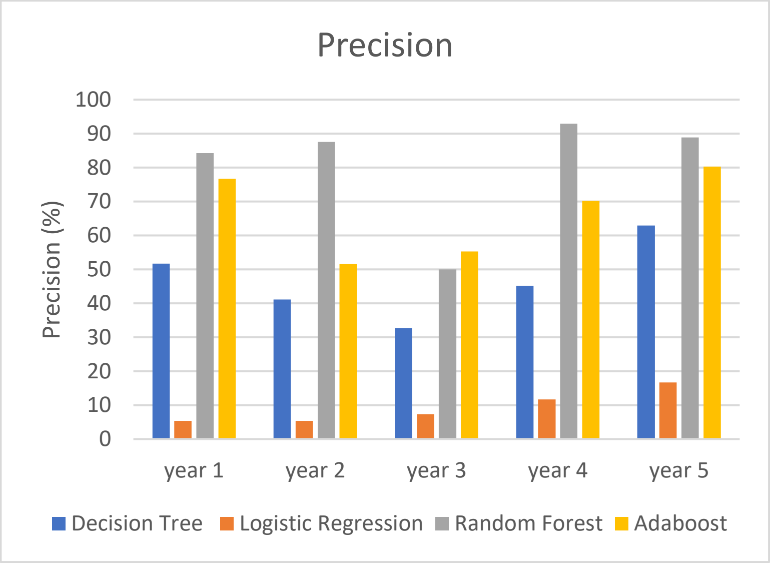
Adaboost Classifier

- The performance of Adaboost Classifier is satisfactory
- Accuracy higher than 95% for all years, precision of all 5 years higher than 50%, recall of 2 out of 5 years higher than 50%
- The satisfactory performance is probably a result of the power of ensemble learning, since the weak learns probably generate prediction errors that are uncorrelated, which are then smoothed out by combining all the weak learners.

(c) Comparison of models

The precision and recall metrics of different models are plotted below (neglecting the Naïve Bayes Model):





Our team performed comparison on the following 2 horizons:

Performance across models

As discussed in part (b) and from the plots above, it is observable that (i) Decision Tree Model (in blue) and (ii) Adaboost Classifier (in yellow) has a relative better precision and recall (when jointly consider the two metrics)

Performance across timeframes

Since Decision Tree Model (blue) and Adaboost Classifier (yellow) seem to be a better model, the discussion below will only focus on these two models.

By observing recall and precision plots for different timeframes, it is observable can observe that for both the models, the performance is better in year 1 dataset (class label indicates bankruptcy status after 5 years) & year 5 dataset (class label indicates bankruptcy status after 1 years)

The above observation may imply that the models:

- Perform better in predicting bankruptcy status in a relatively longer or relative shorter timeframes
- Perform worse in predicting bankruptcy status in a moderate timeframe

(d) Cost and benefit analysis

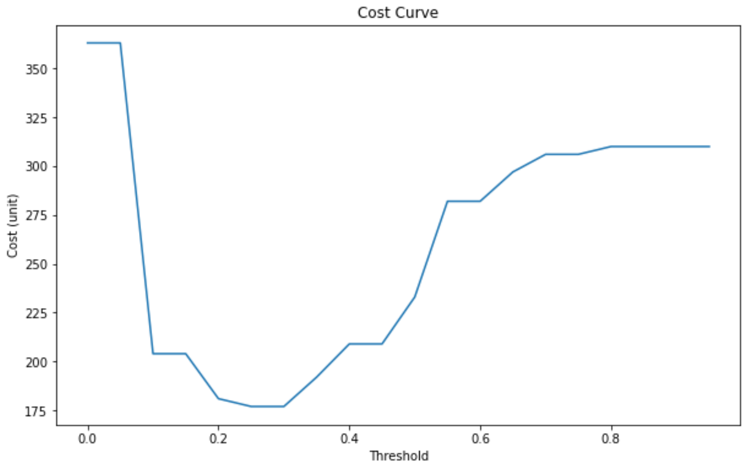
In bankruptcy classification problem, there are mainly 2 types of errors:

- False negative error: labelling a bankrupted company as normal
- False positive error: labelling a normal company as bankrupted

Although concrete figures are hard to find, it is reasonable to conclude that the cost of false negative errors is significantly higher than the cost of false positive errors, since:

- False negative mistake can be interpreted as “potentially suggesting investors to invest in companies which probably go bankruptcy in the near future”, which may lead to severe loss, since equity value and price of a stock will drop to zero upon bankruptcy, which will lead to severe loss on investors
- False positive mistake can be interpreted as “ruling out investors from investing in normal companies”. Although potentially investors may loss some investment opportunities, but choices are still readily available in different security markets

It is assumed that cost of false negative error to cost of false positive error to be 5 : 1. Using the Random Forest Model at year 1, our team obtained the following cost curves at different decision thresholds:



From the cost curve on the right, the optimal threshold associated with the lowest cost is at 0.25.

This suggests that potential users (e.g. investors) who are interested to similar classification models may better choose a lower decision threshold to maximize the utility of the models.

5. Conclusion

To summarize, companies' bankruptcy prediction is a serious challenge faced by stakeholders such as investors and policy makers. Throughout this project, our team has:

- Built several basic machine learning models which may help automate the bankruptcy prediction task.
- Obtained useful insights (e.g., which financial metrics are more indicative about future bankruptcy), which can be applied to bankruptcy prediction or even other aspects of application.

Nevertheless, the models trained throughout the project are definitely not perfect, and more follow up works can be done. Some of them are listed as follows:

Model building

- About model building, since the datasets are significantly imbalanced, some generative models which perform oversampling or under sampling may result in a better performance (e.g., extreme gradient boosting classifier, balance bagging classifier), which is one possible direction to try out.
- In addition, it is possible to synthesize more features considering all the attributes as base features, which may lead to an increase in performance.

Follow up work

- About follow up work, since the datasets in our project focus only on Polish companies, it is possible to conduct similar processes on companies in different geographics.

6. References

Financial metrics understanding

Profitability: https://www.extension.iastate.edu/agdm/wholefarm/html/c3-24.html

Working capital: https://www.investopedia.com/terms/w/workingcapital.asp

Model building (majority of them quoted in the Python codebook)

Class weights: https://www.analyticsvidhya.com/blog/2020/10/improve-class-imbalance-class-weights/

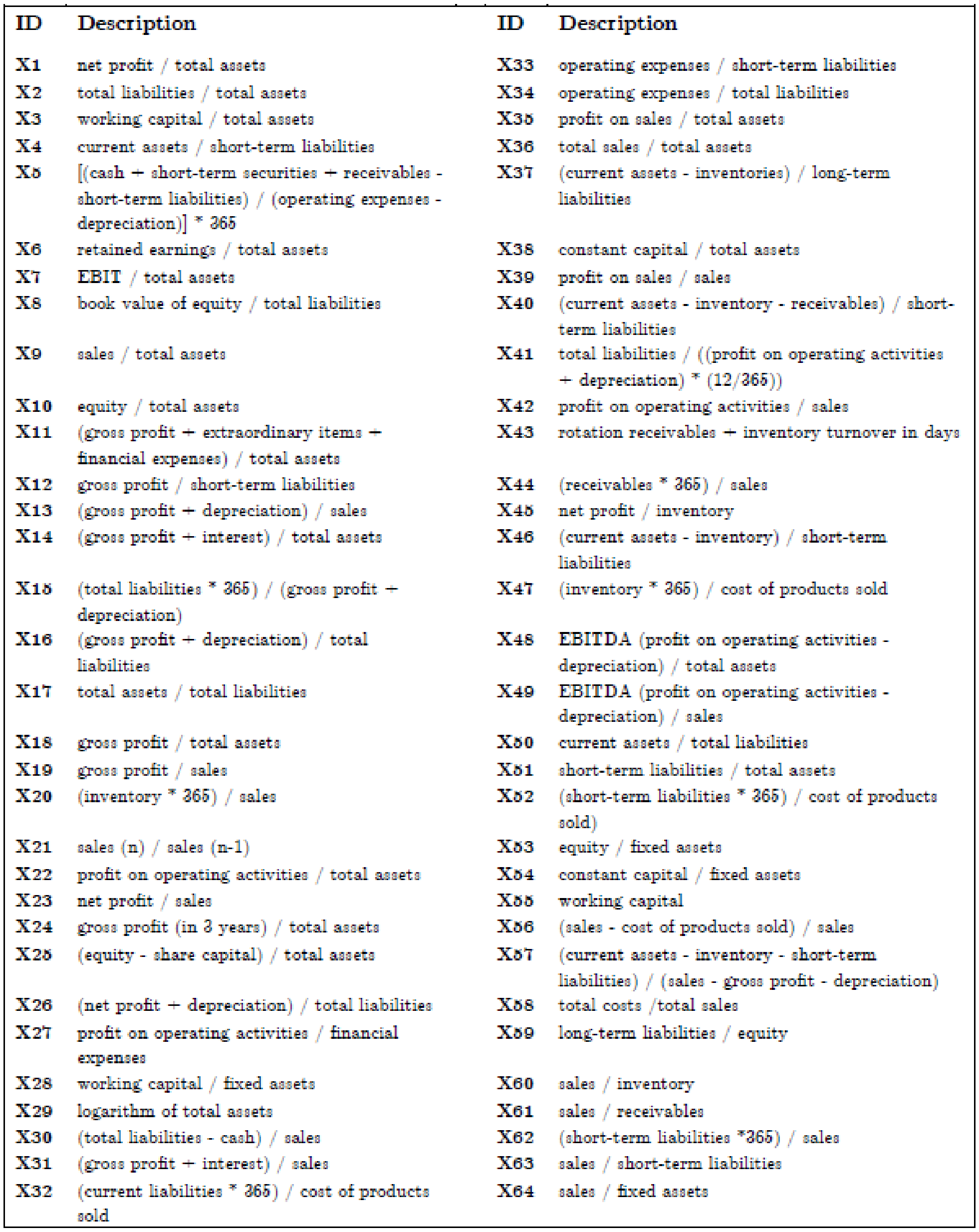
Class imbalance problem: https://www.analyticsvidhya.com/blog/2021/06/5-techniques-to-handle-imbalanced-data-for-a-classification-problem/

Appendix 1

Data count for each year

Data	Total instances	Bankrupted instances	Non-bankrupted instances
Year 1	7027	271	6756
Year 2	10173	400	9773
Year 3	10503	495	10008
Year 4	9792	515	9227
Year 5	5910	410	5500

Attribute list



ID	Description	ID	Description
X1	net profit / total assets	X33	operating expenses / short-term liabilities
X2	total liabilities / total assets	X34	operating expenses / total liabilities
X3	working capital / total assets	X35	profit on sales / total assets
X4	current assets / short-term liabilities	X36	total sales / total assets
X5	[(cash + short-term securities + receivables - short-term liabilities) / (operating expenses - depreciation)] * 365	X37	(current assets - inventories) / long-term liabilities
X6	retained earnings / total assets	X38	constant capital / total assets
X7	EBIT / total assets	X39	profit on sales / sales
X8	book value of equity / total liabilities	X40	(current assets - inventory - receivables) / short-term liabilities
X9	sales / total assets	X41	total liabilities / ((profit on operating activities + depreciation) * (12/365))
X10	equity / total assets	X42	profit on operating activities / sales
X11	(gross profit + extraordinary items + financial expenses) / total assets	X43	rotation receivables + inventory turnover in days
X12	gross profit / short-term liabilities	X44	(receivables * 365) / sales
X13	(gross profit + depreciation) / sales	X45	net profit / inventory
X14	(gross profit + interest) / total assets	X46	(current assets - inventory) / short-term liabilities
X15	(total liabilities * 365) / (gross profit + depreciation)	X47	(inventory * 365) / cost of products sold
X16	(gross profit + depreciation) / total liabilities	X48	EBITDA (profit on operating activities - depreciation) / total assets
X17	total assets / total liabilities	X49	EBITDA (profit on operating activities - depreciation) / sales
X18	gross profit / total assets	X50	current assets / total liabilities
X19	gross profit / sales	X51	short-term liabilities / total assets
X20	(inventory * 365) / sales	X52	(short-term liabilities * 365) / cost of products sold)
X21	sales (n) / sales (n-1)	X53	equity / fixed assets
X22	profit on operating activities / total assets	X54	constant capital / fixed assets
X23	net profit / sales	X55	working capital
X24	gross profit (in 3 years) / total assets	X56	(sales - cost of products sold) / sales
X25	(equity - share capital) / total assets	X57	(current assets - inventory - short-term liabilities) / (sales - gross profit - depreciation)
X26	(net profit + depreciation) / total liabilities	X58	total costs /total sales
X27	profit on operating activities / financial expenses	X59	long-term liabilities / equity
X28	working capital / fixed assets	X60	sales / inventory
X29	logarithm of total assets	X61	sales / receivables
X30	(total liabilities - cash) / sales	X62	(short-term liabilities *365) / sales
X31	(gross profit + interest) / sales	X63	sales / short-term liabilities
X32	(current liabilities * 365) / cost of products sold	X64	sales / fixed assets