Predicting Bankruptcy for Polish Companies

Sonam Gupta
Sgupta@my.harrisburgu.edu

Jiaojie Bai
jbai@my.harrisburgu.edu

Muhammad Huzaifa
mhuzaifa@my.harrisburgu.edu

Abstract

Financial analysis of any firm or company is a crucial perspective of company's operations and how the company projects its future financial health is core input for all of its stakeholders. This paper entails of calculating the financial projections of the Polish companies (data set retrieved from UCI repository) in terms of going bankrupt or staying in business. Multiple machine learning tools were used and compared for this task including neural networks, boosting and bagging. Bagging came out to be the most accurate in predicting the bankruptcy level of the companies under consideration.

Keywords: Neural Network, Financial Ratios, Bankruptcy, Machine Learning

1. Introduction

The field of economics, creditors, investors are always interested in understanding what makes a company go bankrupt. There can be several attributes like net profit, total assets, equity, sales, depreciation and such others that if these numbers are not ideal for the company, it may file for bankruptcy. There have been great improvements in machine learning as well deep learning models for the task of classification. Using some of those classifier models, we believe that we will be able to achieve our goal of identifying the factors that contribute towards any company to go bankrupt and build predictive models. Therefore, our goal is to use the data for Polish companies, freely available on the internet and identify which of these companies are bankrupt or non-bankrupt using the several attributes like net profit, total liabilities, sales and many others. There has been previous research over this problem, so the goal is to build classifiers and see which one of those give us better predicted values. We believe analytics can help us understand if the values of several features can help us categorize the future of Polish companies in terms of bankruptcy.

2. Related work

Corporate Finance is a big part of an economy and the overall entire society in this world. Given the political changes around the globe, we wanted to understand what are the major factors that contribute in a company to go bankrupt. In order to do so, we found a historical data for Polish companies beginning from 2000 to 2013. The detailed description for the dataset is mentioned in later section of this paper. We were motivated to choose this dataset as it is well-founded with large records of data, clear research target, and available for analysis using deep learning methods.

There has been a lot of research done over bankruptcy predictions for companies, but the classification models did not always perform the best [1]. Zhang [1] mentioned several classification algorithms like random forest, neural networks, k-nearest neighbors and gained better predictions from k-nearest and random forest classifiers. Another research group [2] which was a bachelor's thesis, the authors wanted to estimate the risk factors of corporate bankruptcies for investors and credit institutions. To do so, they chose the path of using machine learning and analytics to predict bankruptcy for companies. They want to understand how machine learning could harness from Economics [2]. The results from this study were similar to other previous research findings in terms of predictions and classifiers.

The other interesting paper [3] aimed to compare the deep learning and improved machine learning methods. Their findings suggest the new and improved classifier models like support vector machines (SVM), neural networks predict bankruptcies, with higher accuracies and control over-fitting issues. These models do have drawbacks such as SVM need to use k-fold cross validation which gets expensive in terms of classifiers.

3. Dataset Description

The dataset used in this project is Polish companies bankruptcy data [4] that involves 5 files and 64 attributes along with labeled categories: Bankrupt or non-bankrupt. The following table is taken from [4]:

Data Set Information:

The dataset is about bankruptcy prediction of Polish companies. The data was collected from Emerging Markets Information Service (EMIS, [Web Link]), which is a database containing information on emerging markets around the world. The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013.
Basing on the collected data five classification cases were distinguished, that depends on the forecasting period:
- 1stYear â€" the data contains financial rates from 1st year of the forecasting period and corresponding class label that indicates bankruptcy status after 5 years. The data contains 7027 instances (financial statements), 271 represents bankrupted companies, 6756 firms that did not bankrupt in the forecasting period.
- 2ndYear â€" the data contains financial rates from 2nd year of the forecasting period and corresponding class label that indicates bankruptcy status after 4 years. The data contains 10173 instances (financial statements), 400 represents bankrupted companies, 9773 firms that did not bankrupt in the forecasting period.
- 3rdYear â€" the data contains financial rates from 3rd year of the forecasting period and corresponding class label that indicates bankruptcy status after 3 years. The data contains 10503 instances (financial statements), 495 represents bankrupted companies, 10008 firms that did not bankrupt in the forecasting period.
- 4thYear â€" the data contains financial rates from 4th year of the forecasting period and corresponding class label that indicates bankruptcy status after 2 years. The data contains 9792 instances (financial statements), 515 represents bankrupted companies, 9277 firms that did not bankrupt in the forecasting period.
- 5thYear â€" the data contains financial rates from 5th year of the forecasting period and corresponding class label that indicates bankruptcy status after 1 year. The data contains 5910 instances (financial statements), 410 represents bankrupted companies, 5500 firms that did not bankrupt in the forecasting period.

Exploratory Data Analysis

We use all the files in order to have sufficient data for our analysis. As a part of exploratory data analysis, we converted the name of attribute headers into x1 to x64 for easier understanding and assigned 0 for non-bankrupt class and 1 to bankrupt class.

4. Handling missing values

Handling missing values is itself a big task to solve. There are several methods to handle or impute missing data from huge datasets. In [5] the authors compare six different methods to handle the missing values, such as, “Mean, K-nearest neighbors (KNN), fuzzy K-means (FKM), singular value decomposition (SVD), Bayesian principal component analysis (BPCA) and multiple imputation by chained equations (MICE)”. From their analysis [5], they concluded that mean imputation was the most powerful for handling missing values from the dataset they used.

Normally, missing values can be either substituted by other values or the records that have missing values can be removed entirely. In our case, neither is an option since there is a risk of missing out on a lot of important data resulting into bad prediction accuracies. Therefore, we used mean imputation as a technique to handle the missing values from all the five years of forecasting files. After renaming the column names, we checked the summary of the data to see if there are any missing values and/or NAs. Through this analysis we saw that each file has following number of missing data

1 Year: Total # instances: 7027 Missing Data= 3833
2 Year: Total # instances: 10173 Missing Data= 6085
3 Year: Total # instances: 10503 Missing Data= 5618
4 Year: Total # instances: 9792 Missing Data= 5023
5 Year: Total # instances: 5910 Missing Data= 2879

We substituted these missing values with mean imputation technique. There are 65 independent variables (x), and 1 dependent variable (y). Out of the 43405 records, 41,314 are 0 that means non-bankruptcy, 2091 are 1 that means bankruptcy.

5. Data Imputation

As we think the missing rows are a moderate portion of the total records, we decided to refill the gap by imputing the missing data using “mean imputation method”.

6. Technical Approach

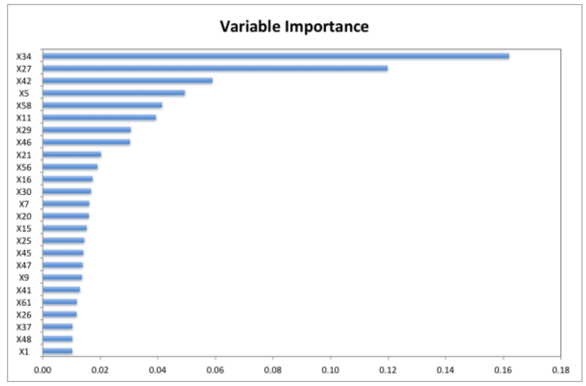
The machine learning classifier models that we implemented are bagging and Adaboost classifiers. For these classifiers, decision trees is the input. Using the scikit-learn libraries, we were able to use the Decision Tree Classifier. These classifiers are explained in more detail in the following paragraph. Neural networks, a deep learning algorithm is another classifier model implemented for this problem.

A. Boosting

Boosting ensemble algorithms creates a sequence of models that attempt to correct the mistakes of the models before them in the sequence [8]. Once created, the models make predictions which may be weighed by their demonstrated accuracy and the results are combined to create a final output prediction.

Ada Boosting weights instances in the dataset by how easy or difficult they are to classify, allowing the algorithm to pay or or less attention to them in the construction of subsequent models.

ab_classifier = AdaBoostClassifier (n_estimators=5, base_estimator = DecisionTreeClassifier (random_state = seed))



B. Bagging

Bootstrap Aggregation or bagging involves taking multiple samples from your training dataset (with replacement) and training a model for each sample. The final output prediction is averaged across the predictions of all of the sub-models [8].

Here we use a BaggingClassifier with the Classification and Regression Trees algorithm.

bt_classifier = BaggingClassifier (base_estimator = DecisionTreeClassifier (random_state=seed), n_estimators = 5, random_state=seed)

Result shows the AdaBoosting method has less accuracy than the Bagging Tree.

Model	Accuracy	Precision	Recall	TN	FP	FN	TP
AdaBoost	91.61%	95.06%	96.55%	1590	11	21	0
Bagging Tree	94.03%	95.06%	98.97%	1634	4	21	0

Table 1.1

C. Neural Networks

Neural networks have been an old research area in deep learning. The basic principle of neural networks is that it learns everything from training examples and has been widely applied in complex tasks like handwriting recognition, image recognition, classification tasks, fingerprint recognition and many more. In neural networks, the input acts as neurons to the model like the neurons in our brain.

In our case, we implemented scikit-learn’s [6] multilayer perceptron classifier algorithm.

```
mlp = MLPClassifier(hidden_layer_sizes=(12,12,12),max_iter=500)
```

A multilayer perceptron algorithm is popular to work well when the data you want to classify is a binary classification. This algorithm can be applied to a supervised learning problem like ours as we have the data already classified into two categories. The classifier model (MLPClassifier) learns from the training input-output data and derives the correlation. In order to optimize the error, backpropagation is used in this classifier to find a good balance for weights and biases.
In between all of this computation, we have layers set to be input *(12)*, hidden*(12)*, and output*(12)* for simplicity [7] with the default activation function being 'relu'. There are several ways to choose the number of neurons for the layers, but we chose equal number of neurons for all the three layers.

7. Test and Evaluation

A. K-fold Cross validation

K-fold cross validation is a very popular technique to verify the predictive model. The idea behind the technique is to randomize the dataset, split it and run it several times. In our case we initialized k = 5 which means the dataset is ran 5 times for all the 5 years of files. All the data gets a chance to act as training and training set to avoid bias. With the cross validation, we then calculate accuracy, precision, recall for all the classifier models.

8. Results

The results are formatted per model, per year and has values for accuracy, precision and recall.

```
-------------------------------------------------------------------------------------------------------------------------
Model: AdaBoost Classifier
                    Dataset: 1year
('Accuracy:', 0.9360938124863954)
('Precision:', array([0.96142349, 0.        ]))
('Recall:', array([0.97467032, 0.        ]))
('TN:', 1315.6)
('FP:', 35.6)
('FN:', 54.2)
('TP:', 0.0)
                    Dataset: 2year
('Accuracy:', 0.9330482050836034)
('Precision:', array([0.96066863, 0.        ]))
('Recall:', array([0.97237957, 0.        ]))
('TN:', 1898.4)
('FP:', 56.2)
('FN:', 80.0)
('TP:', 0.0)
                    Dataset: 3year
('Accuracy:', 0.9082029419097483)
('Precision:', array([0.95285714, 0.        ]))
('Recall:', array([0.9553458, 0.        ]))
('TN:', 1907.8)
('FP:', 93.8)
('FN:', 99.0)
('TP:', 0.0)
                    Dataset: 4year
('Accuracy:', 0.9023572094119438)
('Precision:', array([0.9473953, 0.        ]))
('Recall:', array([0.95496191, 0.        ]))
('TN:', 1767.2)
('FP:', 88.2)
('FN:', 103.0)
('TP:', 0.0)
                    Dataset: 5year
('Accuracy:', 0.9025380710659897)
('Precision:', array([0.93062606, 0.        ]))
('Recall:', array([0.97191201, 0.        ]))
('TN:', 1066.8)
('FP:', 33.2)
('FN:', 82.0)
('TP:', 0.0)
-------------------------------------------------------------------------------------------------------------------------
Model: Bagging Tree Classifier
                    Dataset: 1year
('Accuracy:', 0.9574388361015072)
('Precision:', array([0.96142349, 0.        ]))
('Recall:', array([0.99601535, 0.        ]))
('TN:', 1345.6)
('FP:', 5.6)
('FN:', 54.2)
('TP:', 0.0)
                    Dataset: 2year
('Accuracy:', 0.9530013360101857)
('Precision:', array([0.96066863, 0.        ]))
('Recall:', array([0.9923327, 0.        ]))
('TN:', 1939.0)
('FP:', 15.6)
('FN:', 80.0)
('TP:', 0.0)
                    Dataset: 3year
('Accuracy:', 0.9395280705333061)
('Precision:', array([0.95285714, 0.        ]))
('Recall:', array([0.98667093, 0.        ]))
('TN:', 1973.6)
('FP:', 28.0)
('FN:', 99.0)
('TP:', 0.0)
                    Dataset: 4year
('Accuracy:', 0.9314630726627214)
('Precision:', array([0.9473953, 0.        ]))
('Recall:', array([0.98406777, 0.        ]))
('TN:', 1824.2)
('FP:', 31.2)
('FN:', 103.0)
('TP:', 0.0)
                    Dataset: 5year
('Accuracy:', 0.9201353637901862)
('Precision:', array([0.93062606, 0.        ]))
('Recall:', array([0.98950931, 0.        ]))
('TN:', 1087.6)
('FP:', 12.4)
('FN:', 82.0)
('TP:', 0.0)
```


Model: Neural Network Classifier
Dataset: 1year
Accuracy: 0.9306851672800353
Precision: [0.96142349 0.   ]
Recall: [0.96926168 0.   ]
('TN:', 1061.2)
('FP:', 0)
('FN:', 43.2)
('TP:', 0)

Dataset: 2year
Accuracy: 0.8943236720227871
Precision: [0.96066863 0.   ]
Recall: [0.93365504 0.   ]
('TN:', 1819.6)
('FP:', 80)
('FN:',135)
('TP:', 0)

Dataset: 3year
Accuracy: 0.923248793091725
Precision: [0.95285714 0.   ]
Recall: [0.97039165 0.   ]
('TN:', 1939.4)
('FP:', 99)
('FN:', 62.2)
('TP:', 0)

Dataset: 4year
Accuracy: 0.8907253966789043
Precision: [0.9473953 0.   ]
Recall: [0.9433301 0.   ]
('TN:',1744.4)
('FP:', 103)
('FN:', 111)
('TP:', 0)

Dataset: 5year
Accuracy: 0.8456852791878171
Precision: [0.93062606 0.   ]
Recall: [0.91505922 0.   ]
('TN:', 999.6)
('FP:', 82)
('FN:', 100.4)
('TP:', 0)

Boosting Variable Importance

Var	Variable Name	Variable Importance
X34	operating expenses / total liabilities	0.16201
X27	profit on operating activities / financial expenses	0.11980
X42	profit on operating activities / sales	0.05895
X5	[(cash + short-term securities + receivables - short-term liabilities) / (operating expenses - depreciation)] * 365	0.04935
X58	total costs /total sales	0.04146
X11	(gross profit + extraordinary items + financial expenses) / total assets	0.03935
X29	logarithm of total assets	0.03057
X46	(current assets - inventory) / short-term liabilities	0.03030
X21	sales (n) / sales (n-1)	0.02023
X56	(sales - cost of products sold) / sales	0.01903
X16	(gross profit + depreciation) / total liabilities	0.01732
X30	(total liabilities - cash) / sales	0.01677
X7	EBIT / total assets	0.01613
X20	(inventory * 365) / sales	0.01602
X15	(total liabilities * 365) / (gross profit + depreciation)	0.01527
X25	(equity - share capital) / total assets	0.01441
X45	net profit / inventory	0.01410
X47	(inventory * 365) / cost of products sold	0.01388
X9	sales / total assets	0.01365
X41	total liabilities / ((profit on operating activities + depreciation) * (12/365))	0.01290
X61	sales / receivables	0.01184
X26	(net profit + depreciation) / total liabilities	0.01177
X37	(current assets - inventories) / long-term liabilities	0.01032
X48	EBITDA (profit on operating activities - depreciation) / total assets	0.01028
X1	net profit / total assets	0.01025
X39	profit on sales / sales	0.01003
X6	retained earnings / total assets	0.00958
X44	(receivables * 365) / sales	0.00954
X40	(current assets - inventory - receivables) / short-term liabilities	0.00937
X55	working capital	0.00937
X36	total sales / total assets	0.00888
X33	operating expenses / short-term liabilities	0.00886
X32	(current liabilities * 365) / cost of products sold	0.00860

X3	working capital / total assets	0.00836
X63	sales / short-term liabilities	0.00820
X51	short-term liabilities / total assets	0.00763
X43	rotation receivables + inventory turnover in days	0.00729
X57	(current assets - inventory - short-term liabilities) / (sales - gross profit - depreciation)	0.00707
X19	gross profit / sales	0.00701
X54	constant capital / fixed assets	0.00639
X10	equity / total assets	0.00630
X53	equity / fixed assets	0.00627
X22	profit on operating activities / total assets	0.00598
X59	long-term liabilities / equity	0.00595
X24	gross profit (in 3 years) / total assets	0.00576
X52	(short-term liabilities * 365) / cost of products sold)	0.00564
X64	sales / fixed assets	0.00559
X38	constant capital / total assets	0.00452
X2	total liabilities / total assets	0.00422
X8	book value of equity / total liabilities	0.00420
X18	gross profit / total assets	0.00412
X50	current assets / total liabilities	0.00403
X4	current assets / short-term liabilities	0.00402
X12	gross profit / short-term liabilities	0.00397
X60	sales / inventory	0.00353
X13	(gross profit + depreciation) / sales	0.00342
X23	net profit / sales	0.00341
X17	total assets / total liabilities	0.00333
X49	EBITDA (profit on operating activities - depreciation) / sales	0.00328
X35	profit on sales / total assets	0.00276
X28	working capital / fixed assets	0.00232
X62	(short-term liabilities *365) / sales	0.00207
X31	(gross profit + interest) / sales	0.00170
X14	(gross profit + interest) / total assets	0.00150

Averaged Precision
AdaBoost = 91.64%
Bagging = 94.03%
Neural Networks = 95%

From the above averages over the 5 years dataset, we see that neural network had the highest precision score which means that false positives were higher for this model, followed by bagging and adaboost classifiers.
Bagging classifier has the highest accuracy around 95% on average for all the 5 files among all the classifiers which means the model predicted the classes 95% accurately.

The heatmap for confusion matrix for true positives, false positives, true negatives and false negatives is as follows:

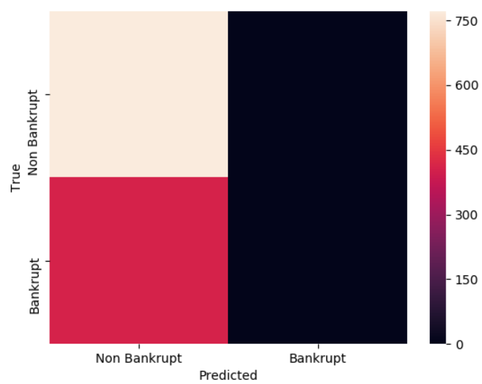


Fig 1. Heat Map

9. Discussion

From the related research, we think that neural networks worked pretty similar to the results we achieved from this project. The approach we chose, gave us quite satisfactory results. We could have implemented different other classifiers like naive bayes, logistic regression and such.

When model ranks were compared, Bagging Classifier outperformed the other classifiers performance.

10. Future Work

Reducing the dimensionality of features can be helpful. Since we just used mean imputation for handling missing values, the other methods may have been proven useful too. Having such a big amount of missing values in data like for Polish companies, feature extraction and importance can be biased.

References

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

[2] *Corporate Bankruptcy Prediction using Machine Learning Techniques* BJÖRN MATTSSON OLOF STEINERT. Bachelor's Thesis in Economics, 2017.

[3] Wang, N. (2017) "*Bankruptcy Prediction Using Machine Learning*". *Journal of Mathematical Finance*, **7**, 908-918. doi: 10.4236/jmf.2017.74049.

[4] Zieba, M., Tomczak, S. K., & Tomczak, J. M. (2016). "*Ensemble Boosted Trees with Synthetic Features Generation in Application to Bankruptcy Prediction. Expert Systems with Applications"*

[5] Schmitt P, Mandel J, Guedj M (2015) "*A Comparison of Six Methods for Missing Data Imputation"*. J Biom Biostat 6:224. doi: 10.4172/2155-6180.1000224

[6] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011

[7] J. Portella, "*A beginner's guide to Neural Network in Python and SciKit Learn o.18*", March 21, 2017. Available: https://www.springboard.com/blog/beginners-guide-neural-network-in-python-scikit-learn-0-18/ [Accessed on October 5, 2018]

[8] J. Browniee, "*Ensemble Machine Learning Algorithm in Python with scikit-learn*", Jun. 3, 2016. Available: https://machinelearningmastery.com/ensemble-machine-learning-algorithms-python-scikit-learn/ [Accessed on September 25, 2018]