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)# creating a dictionary of models
```

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

Model	Accuracy	TN	FP	FN	TP
AdaBoost	0.915351	1589	11	22	0
Bagging Tree	0.940313	1634	4	22	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: Decision Tree Classifier

Dataset: 1year
Accuracy: 0.9362356550219445
Precision: [0.96142349 0.]
Recall: [0.97481217 0.]

Dataset: 2year
Accuracy: 0.932557191141262
Precision: [0.96066863 0.]
Recall: [0.97188856 0.]

Dataset: 3year
Accuracy: 0.9098219895287958
Precision: [0.95285714 0.]
Recall: [0.95696485 0.]

Dataset: 4year
Accuracy: 0.9028685081974135
Precision: [0.9473953 0.]
Recall: [0.95547321 0.]

Dataset: 5year
Accuracy: 0.9003384094754654
Precision: [0.93062606 0.]
Recall: [0.96971235 0.]

Model: AdaBoost Classifier

Dataset: 1year
Accuracy: 0.9342429749472266
Precision: [0.96142349 0.]
Recall: [0.97281949 0.]

Dataset: 2year
Accuracy: 0.9337365040019907
Precision: [0.96066863 0.]
Recall: [0.97306787 0.]

Dataset: 3year
Accuracy: 0.9062984066544276
Precision: [0.95285714 0.]
Recall: [0.95344126 0.]

Dataset: 4year
Accuracy: 0.9044006838868928
Precision: [0.9473953 0.]
Recall: [0.95700538 0.]

Dataset: 5year
Accuracy: 0.9015228426395939
Precision: [0.93062606 0.]
Recall: [0.97089679 0.]

Model: Bagging Tree Classifier

Dataset: 1year
Accuracy: 0.9574388361015072
Precision: [0.96142349 0.]
Recall: [0.99601535 0.]

Dataset: 2year
Accuracy: 0.9530013360101857
Precision: [0.96066863 0.]
Recall: [0.9923327 0.]

Dataset: 3year
Accuracy: 0.9395280705333061
Precision: [0.95285714 0.]
Recall: [0.98667093 0.]

Dataset: 4year
Accuracy: 0.9314630726627214
Precision: [0.9473953 0.]
Recall: [0.98406777 0.]

Dataset: 5year
Accuracy: 0.9201353637901862
Precision: [0.93062606 0.]
Recall: [0.98950931 0.]

Model: Neural Network Classifier

Dataset: 1year
Accuracy: 0.9306851672800353
Precision: [0.96142349 0.]
Recall: [0.96926168 0.]

Dataset: 2year
Accuracy: 0.8943236720227871
Precision: [0.96066863 0.]
Recall: [0.93365504 0.]

Dataset: 3year
Accuracy: 0.923248793091725
Precision: [0.95285714 0.]
Recall: [0.97039165 0.]

Dataset: 4year
Accuracy: 0.8907253966789043
Precision: [0.9473953 0.]
Recall: [0.9433301 0.]

Dataset: 5year
Accuracy: 0.8456852791878171
Precision: [0.93062606 0.]
Recall: [0.91505922 0.]

Averaged Precision
AdaBoost = 94%
Bagging = 94.8%
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:

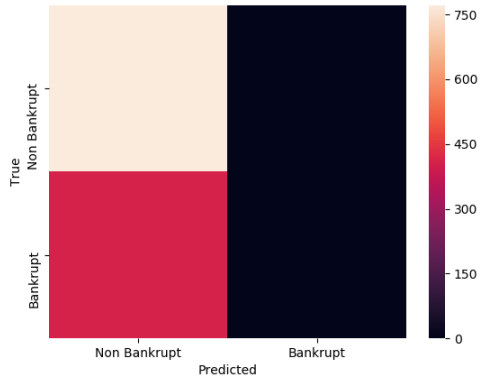

Fig 1. Heat Map

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]