Bankruptcy prediction of Polish companies using machine learning methods

Nikola Miszalska
Faculty of Mathematics and Information Science
Warsaw University of Technology
Warsaw, Poland
nikola.miszalska.stud@pw.edu.pl

Grzegorz Zakrzewski
Faculty of Mathematics and Information Science
Warsaw University of Technology
Warsaw, Poland
grzegorz.zakrzewski.stud@pw.edu.pl

June 9, 2022

Abstract

This article presents the research results relating to Polish companies' bankruptcy prediction. That covers data analysis, data preprocessing, model creation, and explanation. We conducted exploratory data analysis to obtain the most crucial information and essential characteristics. We dealt with problems discovered during preprocessing: unbalanced classes, missing values, and outliers. Then, we examined and compared four machine learning models: Logistic Regression, Support Vector Machine, Random Forest and XGBoost. We executed hyper-parameter tuning on the best-performing XGBoost model and attempted to explain it with SHAP and DALEX packages.

Keywords bankruptcy prediction · xgboost · unbalanced classes · Polish companies

1 Introduction

Our study was conducted to create a machine learning model for the task of bankruptcy prediction. We were working with Polish companies bankruptcy data set[Zięba et al., 2016].

2 Data analysis and preprocessing

The main purpose of data mining was to find a model that, based on the financial indicators of Polish companies, would be able to predict bankruptcy (1) or nonbankruptcy (0) of the company. For creating this model, we used a specific data mining task - classification. The individual classification models were initially generated on the training set and subsequently evaluated on the testing set.

2.1 Data set description

The data set is about bankruptcy prediction of Polish companies. The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013. Overall, there are 43405 observations. Each row describes one particular company.

The data is divided into five sets based on the bankruptcy prediction period. Each set contains a different number of records (companies), the same number and meaning of attributes, and different values of each attribute.

The data set consists of 64 various econometric indicators. Each indicator combines the econometric measures using arithmetic operations (mostly division). See the table for a full description 1. Also, there is one binary target variable named `class`. Value `1` indicates that company has bankrupted, and `0` - that has not.

Table 1: The set of features considered in classification process

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

2.2 Exploratory data analysis

During this step, we gather and summarise information about our data set.

Target variable were strongly unbalanced. Companies usually do not go bankrupt, and out of total 43405 rows there were only 2091 corresponding to bankrupted companies. In other words, by classifying each company as `0` (not bankrupt), you would get more than 95% of accuracy.

By the analysis we found, that data had many missing values. Every single column contained some missing values. In most cases (56 columns), missing data represented less than 1% of total number. Accordingly, half of the rows had missing values. Usually, that was one or two N/As, but in all cases.

Moreover, every single feature had a strongly skew distribution and/or a lot of outliers from both sides - look at figure 2.2. We didn't even know if some features can take negative values. A domain-specific knowledge is needed to understand the meaning of financial indicators.

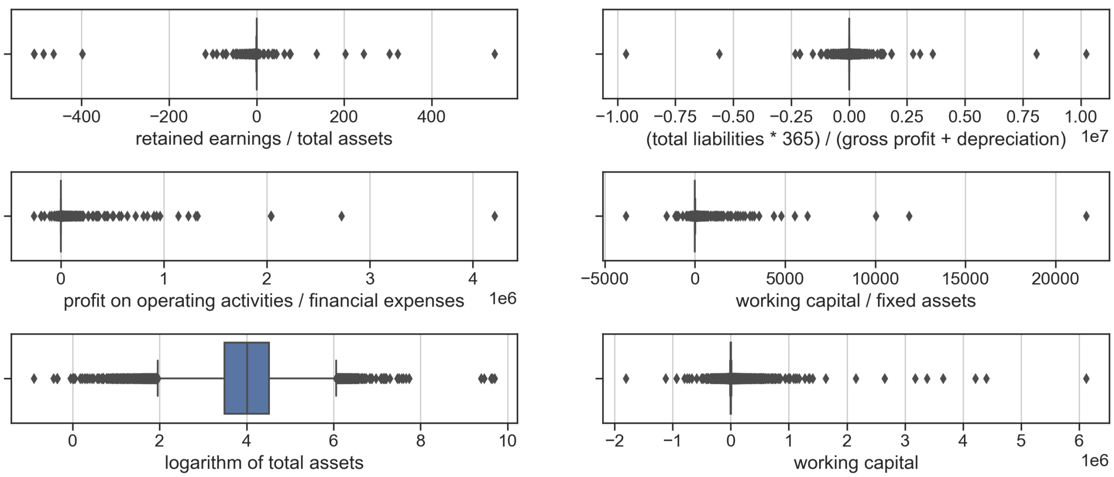


Figure 1: Box plots of six features

Last but not least, in our data set existed strong correlations between features. This was understandable because a lot of our features - financial indicators have the same numerator or denominator (see 1. For example, phrase *total assets* occurs more than twenty times.

2.3 Data preparation

We decided to drop columns and rows with the highest numbers of missing values. These were three columns with more than 5% of missing values and about 243 rows with more than seven missing values. In attributes with a lower number of missing values, we replaced them with column medians.

As we mentioned before, there were many outliers, so we cut them to quantiles: 0.025 from left and 0.975 from right. We also used a standard scaler to standardize the range of functionality of our data set.

None of the columns were correlated with the target variable, but there were groups of strongly correlated attributes. We tried generating all of them and keep only one from each group to see what would happen, but it turned out that it didn't have any positive effect on the models.

Last but not least, in our data set existed strong correlations between features. This was understandable because a lot of our features - financial indicators have the same numerator or denominator (see 1. For example, phrase *total assets* occurs more than twenty times.

3 Models

3.1 Overview

Our goal was to maximize *f1-score*. We decided to use this metric because we did't have any business-connected goal to minimize false-positive or true-negative rate, that is *precision* or *recall* score.

In the first modeling phase, we applied Logistic Regression, Support Vector Machine, Random Forest and XGBoost, but the first two approaches gave us very poor results on both train and test set. It seemed that these ones were too weak for our problem. On the other hand, Random Forest model fited too closely with training data. It was a good sign - we could predict bankruptcy based on our data. But the best score was achieved with the XGBoost model. The results of this phase are placed in the table 2. We decided to tune hyper-parameters of our two last models.

Table 2: F1-score of four models after first modelling phase

	Logistic Regression	Support Vector Machine	Random Forest	XGBoost
Train Set	0.2200	0.3436	0.9935	0.3241
Test Set	0.2103	0.2383	0.0976	0.2774

3.2 Random Forest

Random Forest is a model that has fitted too much to the training data. We had a hope that with proper hyper-parameters it would perform much better. Our goal was to prune trees. Using randomized search approach we've found out best-scoring values of `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_features` and `max_leaf_nodes` hyper-parameters. Although the final score was much better, it was not satisfactory.

Table 3: F1-score of Random Forest model after hyper-parameters tuning

	Train Set	Test Set
F1 score	0.4930	0.3538

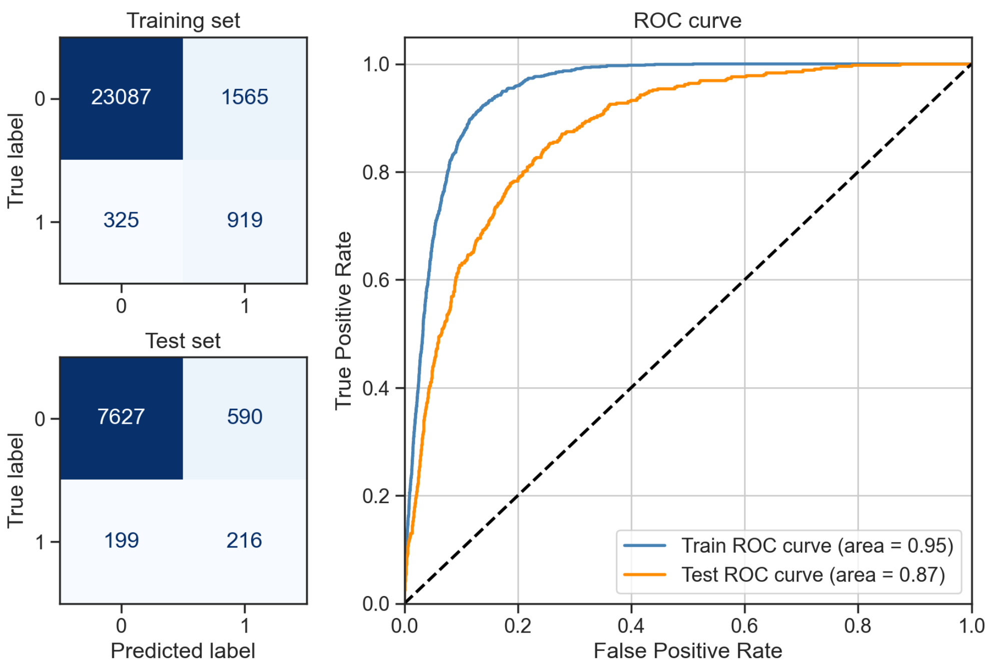


Figure 2: Confusion matrix and ROC curve of Random Forest predictions

3.3 XGBoost

XGBoost turned out to be our best model. The highest F1 score was achieved due to this approach. We searched for the best hyper-parameters using randomized and grid search methods. The best ones are presented in the table 4. Worthy mentioning is the fact that we were using `scale_pos_weight` parameter, which is set as a ratio: number of non-bankrupted companies to bankrupted ones. Besides, it may occur to somebody that our model is slightly over-fitted. We tried to prevent this by pruning trees, but it has negative impact on the overall score.

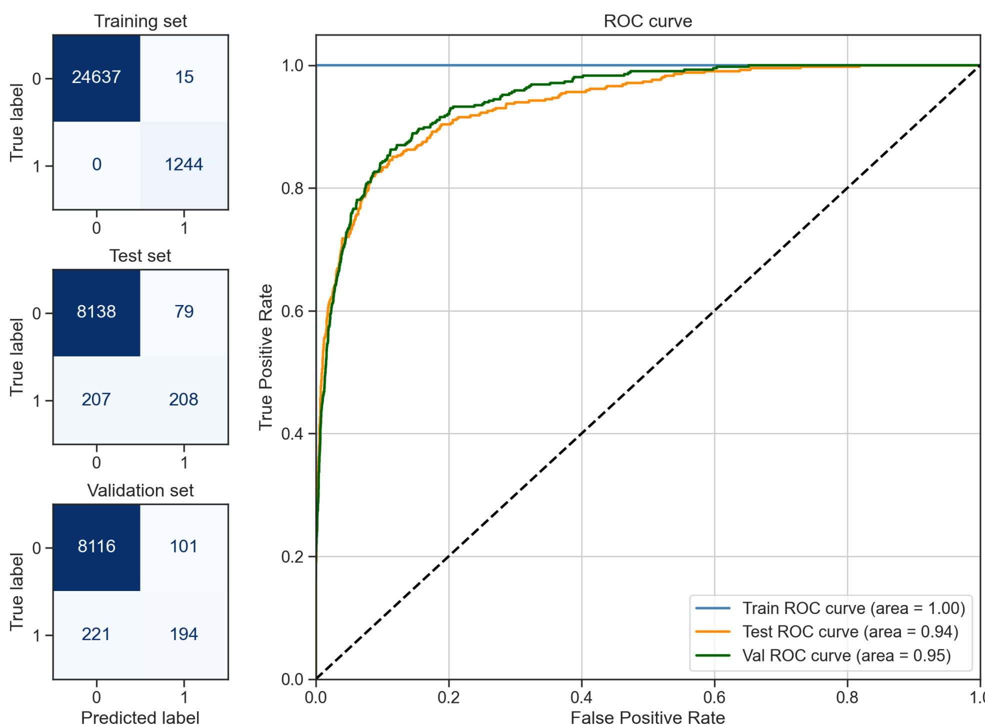
Table 4: XGBoost hyper-parameters

n_estimators	1000
max_depth	6
min_child_weight	4
learning_rate	0.1
gamma	0.5
scale_pos_weight	9.x
lambda	1.2
objective	binary:logistic
eval_metric	logloss
random_state	0

Table 5: Scores of XGBoost model after hyper-parameters tuning

	Training	Test	Validation
precision	0.9881	0.7247	0.6576
recall	1.000	0.5012	0.4675
f1	0.9940	0.5926	0.5465

Figure 3: Confusion matrix and ROC curve of XGBoost predictions



4 XAI

We were looking for the most important variables and their impact on model predictions. For this purpose we used SHAP [Lundberg et al., 2020] and DALEX [Biecek, 2018] packages.

We took a look at the SHAP summary plot and also created DALEX variable importance plot. Fortunately, both packages agreed on the selection of features (see 4). Finally, we tried partial dependence plots (4).

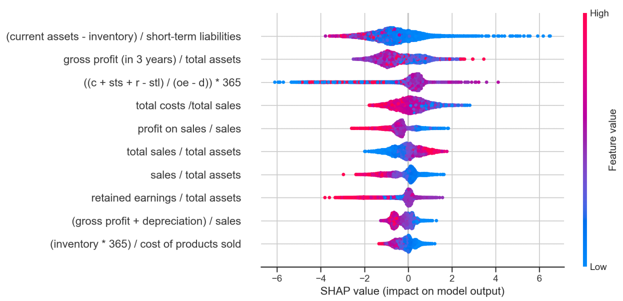
Conclusions:

- single feature has rather small impact on overall prediction;

- features have rather *flat* partial dependence profile;
- most important features are:
  - `current assets - inventory / short-term liabilities` - lower values = bankruptcy
  - `sales / total assets` - lower values = bankruptcy;
  - `gross profit (in 3 years) / total assets` - higher values = bankruptcy;
- it seems that when feature has keyword *assets* in its name, it is important.

We can also see that prediction level on variable importance and partial dependence plots is shifted to zero. It happens because of unbalanced classes.

SHAP summary plot



Variable importance plot

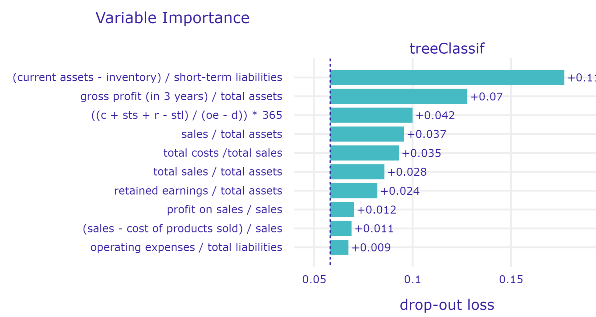
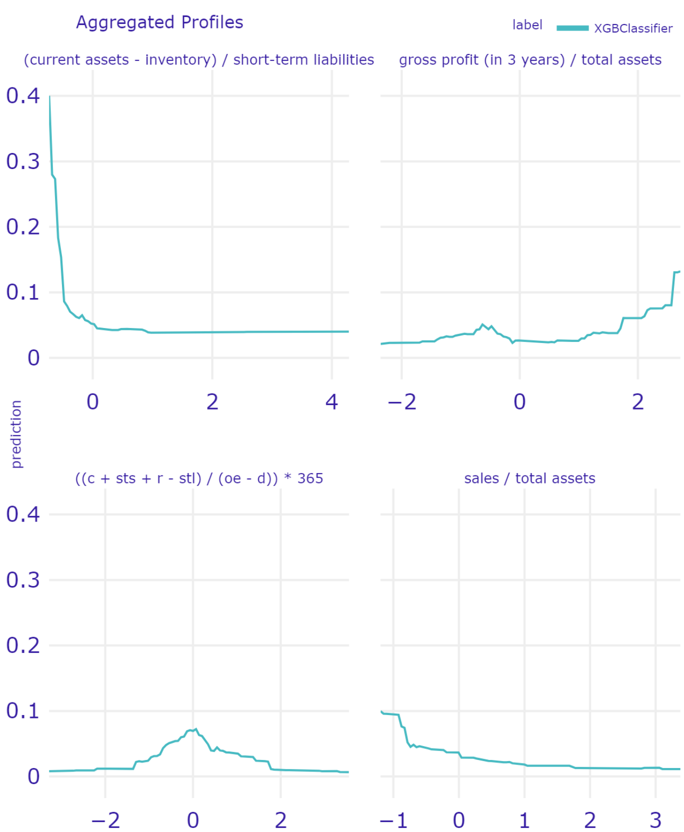


Figure 4: Partial dependence plot



5 Conclusions

In this section, we will shortly summarise our study. We encountered some difficulties with our data set. The main problem was unbalanced classes, which required some cautiousness in interpreting popular model performance measures. There were also missing values and outliers, and we had to take necessary steps during preprocessing phase. Also, domain-specific language made understanding data impossible. We wanted to maximize equally precision and recall, so we decided to check *f1-score*. It was achieved with the XGBoost model. At the end, we found out several important features like `current assets - inventory / short-term liabilities` or `gross profit (in 3 years) / total assets`.

References

Maciej Zięba, Sebastian K Tomczak, and Jakub M Tomczak. Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction. *Expert Systems with Applications*, 2016.

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, 2(1):2522–5839, 2020.

Przemyslaw Biecek. Dalex: Explainers for complex predictive models in r. *Journal of Machine Learning Research*, 19 (84):1–5, 2018. URL `https://jmlr.org/papers/v19/18-416.html`.