Predicting customers’ defaults

ING Lion’s Den credit risk modeling exercise
Monika Kaczan

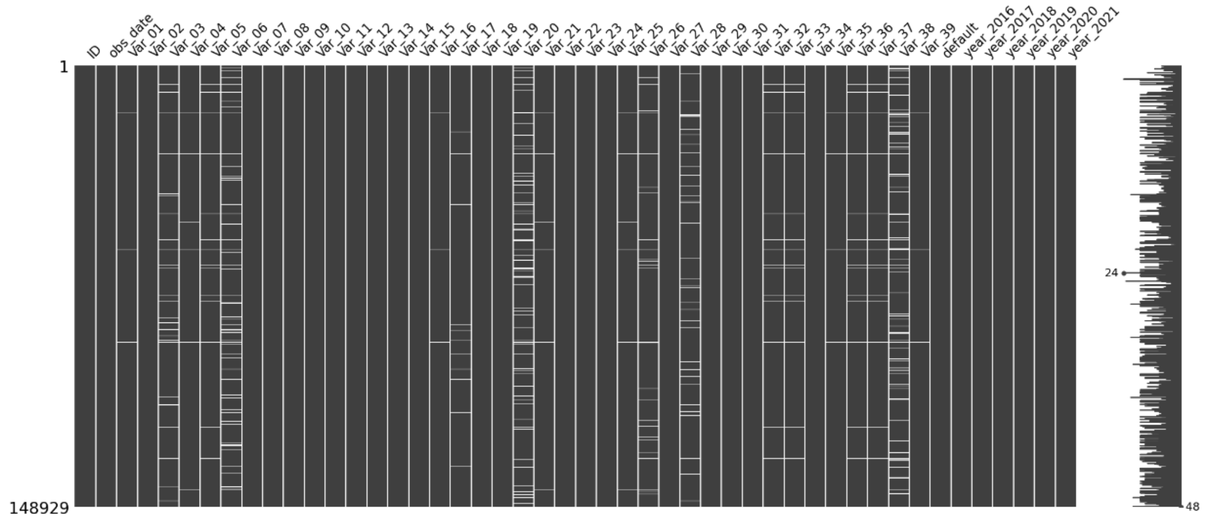
Introduction

The goal of this project is to predict defaults of Big Lion Bank’s customers. I received annual data on borrowing companies including financial information and information whether a particular company defaulted within the next 12 months. Using this data, I build logit, ridge, KNN and XGBoost models. I compared their performance based on F1 score and other metrics and chose the best one.

Data preprocessing

The raw data contained 148929 observations of unique customers observed in the years 2015-2021. It included 39 numerical variables describing their financial characteristics, time variable, and one dummy variable indicating performing company defaulting within the next 12 months.

<u>Handling missing values</u>


Visualization of missing values in the dataset and particular variables

Over 40,000 of observations (around 27%) contained at least one missing value. Therefore, I find simply deleting them as not reasonable. I decided to pursue following approach:

- deleting variables which had very high correlations with other variables AND had more than 1500 (~1% of entire dataset) NaNs: Var_03, Var_17, Var_20, Var_26, Var_38
- predicting missing values using linear regression for variables which had more than 1500 NaNs but only moderate correlations with other variables: Var_06 and Var_28

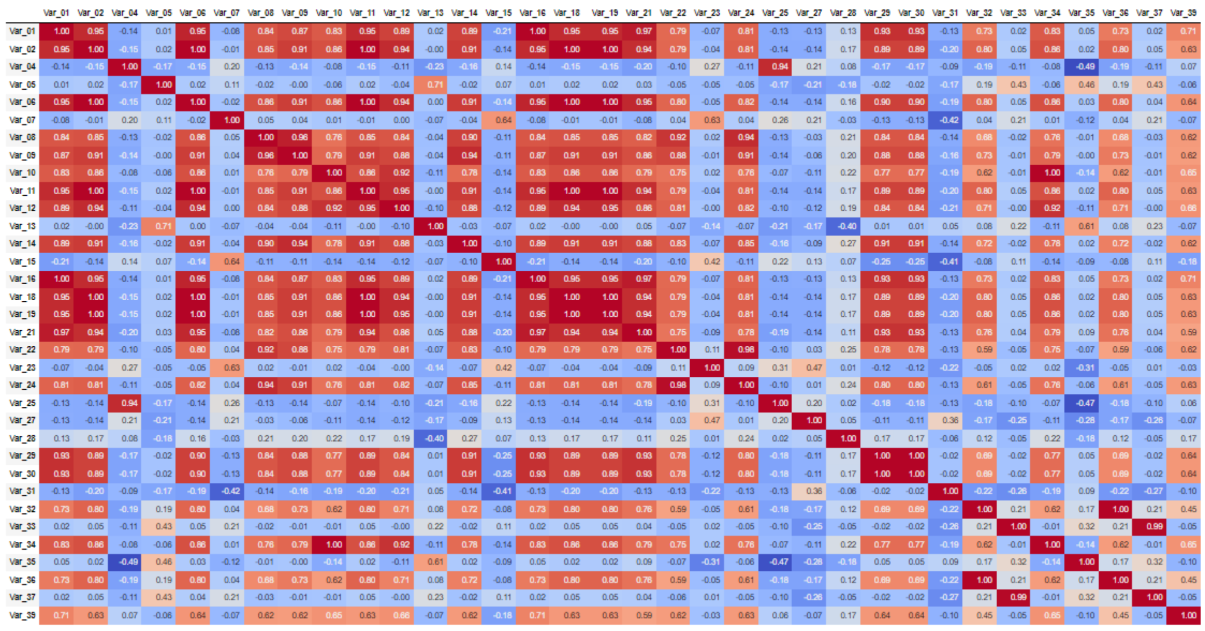
- filling NaNs with median values (as the distributions were skewed) for rest of variables which contained NaNs

<u>Outliers detection and handling</u>
Investigating variables' distributions showed that they contain observations which could be classified as outliers. This could be due to human errors or just the characteristics of such financial data. Nevertheless, outliers can significantly affect performance of many types of models. This is why I decided to cap and floor each column with 99th and 1st percentile respectively.

<u>Splitting data into training and testing samples</u>
I split the data into training (70% of entire dataset) and testing (30% of entire dataset) samples. I made sure that distribution of the target variable remained similar in both samples.

<u>Resampling and standardization</u>
The data received was highly unbalanced - companies which defaulted accounted for only around 5,87% of the entire dataset. I used SMOTE to create new synthetic observations in a training sample based on existing ones and thus balancing the dataset.
I also standarized the data so that it can be correctly used by different models.

<u>Variables selection</u>



Correlation plot for X_train after resampling, outliers handling and standardization

Based on the Pearson correlation plot, I see that some of the variables are highly correlated. Due to time and resources constraints, I decided to drop selected variables manually (instead of, i.e. performing PCA) based on their correlation with other variables and economic interpretation. All variables deleted below had very high (coefficient of almost 1) correlation with kept variables and indicated similar positions in the balance sheet such as IFRS Assets Total vs. Assets Total or Tangible Net Worth and Eff Tang Net Worth Actual.

Deleted variables:

- Var_06, Var_11, Var_18, Var_19 - keeping Var_02
- Var_32 - keeping Var_36
- Var_10 - keeping Var_34
- Var_16 - keeping Var_01

Additionally, I added 6 dummy variables indicating the year (2016 to 2021 and 2015 as default) in which each observation was observed.

Finally, the training dataset consisted of 196252 observations with 40 variables including 34 numeric variables on financial data and 6 dummy year variables. The dataset was balanced in terms of target variable thanks to SMOTE.

Modelling

For all models, I used F1 score as the primary evaluation metric for tuning the parameters. F1 is the harmonic mean of precision and recall and I used it because it is more suitable for unbalanced data than simple accuracy or ROC AUC score (I rebalanced training data, but will still eventually evaluate performance on unbalanced test data).
To properly evaluate models' results and avoid overfitting, each time I used five fold cross-validation (repeated three times in selected algorithms).

Logistic regression
Logistic regression (logit) is one of the most basic models for classification and was used here as a benchmark for more sophisticated models. I achieved an F1 score of 0.7963 which is okay, but not very good.

Ridge regression
Ridge regression is a method of regularization which imposes restrictions on less important parameters that bring them close to 0. I tuned in penalization parameter alpha and the best F1 score of 0.7936 was obtained for alpha = 1. Therefore I were not able to improve results compared to logit.

K nearest neighbors (KNN)
KNN is a non-parametric model which uses proximity to make classifications about the grouping of an individual data point. I tuned in the number of neighbors parameter (how many neighbors of an observation will be checked to determine the classification). I obtained the best result of F1 score of 0.9698 for 11 neighbors. It is a significant improvement compared to logit and ridge and overall a very good result. However, I should be careful and check it later on testing data as it might be due to overfitting.

XGBoost
XGboost is a type of gradient boosting i.e. a procedure, where final prediction is created by combining results of many decision trees which are dependent on the previous ones. I tuned the learning rate, number of estimators, subsample and maximum depth parameters. I obtained the best F1 score of 0.9677 for learning_rate = 0.1, max_depth = 9, n_estimators = 500 and subsample = 0.8. Thus, the results were very similar to KNN.

Comparison and conclusions

Let's recap the performance of models on training data and check and compare their performance on testing data.

Sample & performance measure	Logit	Ridge	KNN	**XGBoost**
Training F1	0.7963	0.7936	0.9698	**0.9677**
Testing F1	0.2392	0.2365	0.2524	**0.4483**
Testing ROC AUC	0.6730	0.6594	0.7205	**0.7168**

On training data, the best models in terms of F1 score were KNN and XBoost with F1 score of around 0.97 for both. Logit and Ridge performed much worse and both had an F1 score of around 0.8.
On testing data, unfortunately the results in terms of F1 were very bad, most probably as a result of some error - maybe in the prediction procedure itself (which is rather not probable but what I hope for), but also possibly for example in resampling or optimization of parameters as the ROC AUC score were poor, but not terrible which could indicate that the negative class was predicted okay.
If I must choose a model for prediction, I would pick XGBoost, as it has significantly higher performance on the testing sample than other methods but I acknowledge that it is still poor and should be investigated further with checking classification matrix etc.

Further analysis
I acknowledge that due to lack of time and processing power the analysis contains some shortcomings. Given more time and resources, the author would implement the following:
- First and foremost, investigating the origin of such terrible results on testing data.
- Better handling of missing values and outliers, for example investigating the cases for each variable separately
- Performing PCA on all or selected highly correlated variables and treating them as predictors to reduce dimension of the dataset. I did not do that as this would require a significant amount of additional time put into ensuring economic interpretation of obtained principal components.
- More robust validation, such as increasing the number of folds in cross-validation and optimizing parameters based on multiple evaluation metrics, not just F1.
- Implementing support vector machine (SVM) and neural networks models which require more time spent on computation and parameters tuning.