# Capstone project 1: Loan Analysis Milestone Report

## Problem statement

Small and medium enterprises (SMEs) are businesses that have revenues, assets and a number of employees below a certain threshold. Although the definition of SMEs is different from country to country, their important role in the economy is undeniable. They are one of the strongest drivers of economic development, innovation and employment. The access to finance is recognized as the greatest obstacle to the growth of SMEs. The financing gap for SMEs is huge. Millions of SMEs have unmet financing needs all over the world. Lenders who can address this matter have great growing potential. The challenge for these lenders is how to manage loan risk efficiently and make accurate decisions. There are two basic risks here: one is a business loss results from not lending the good candidates, and the other is the financial loss for lending the candidates at bad risk. Therefore, the ability to recognize good candidates who are able to make payments is crucial.

This project focuses on using classification models in supervised machine learning (Logistic Regression, SVM, kNN, Random Forest, Decision Tree, Naive Bayes, Stochastic Gradient Descent) to predict whether a company is going to miss payments if money is lent. Based on this information along with the interest rate, requested disbursement amounts the lender can make a decision of lending or not.

## Data Source

The dataset is taken from a South African digital lender. The join data set consists of 1346 rows, representing 1346 companies who are already their customers, and 49 columns, representing the features. This dataset has continuous and categorical data along with missing values. The variable of interest is “bad” containing the payment history of these companies, either 1 meaning company missed payments or 0 meaning company did not miss payments.

# Data Wrangling

The raw data comes from two different tables “NEW_DATA.csv” and “More_data.csv”. The columns “AdvanceId” from the 1st table and “advance_id” from the second table are matching. We first load data from these two tables into Pandas DataFrames using **pd.read_csv** and then join them using **pd.merge.**

# Data Preprocessing

- ## Handling missing data:

  Many features from the dataset contain missing values. We replace a missing value by the most frequent value in the same feature using SimpleImputer.

- ## Handling Errors and Typos:

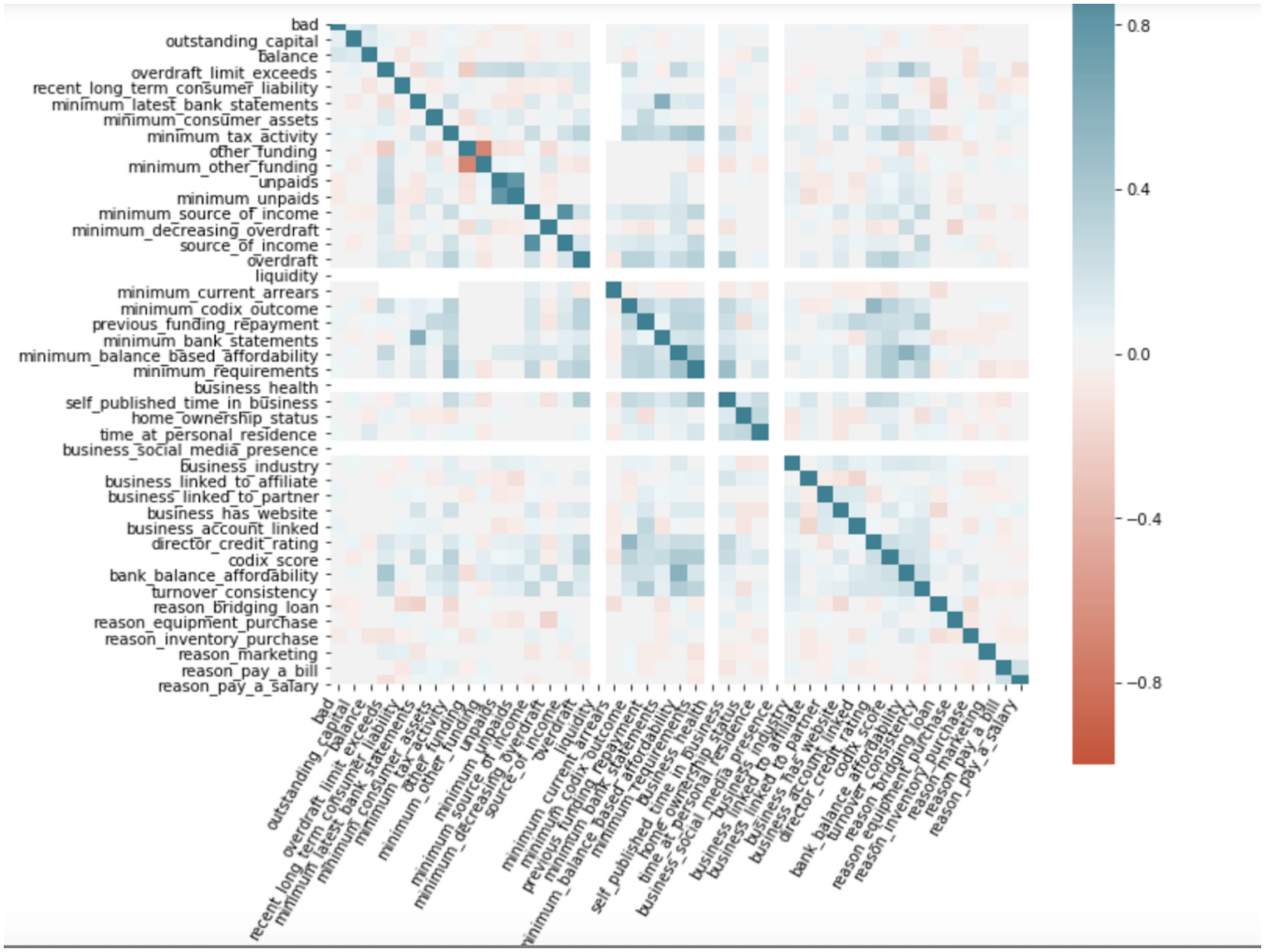
  The two features **reason_bridging_loan** and **reason_inventory_purchase** are supposed to be binary features. They can either take the value 0 or 1. However, when we look at the dataset, there is a fair amount of data in these two columns that are neither 0 nor 1. Our plan is to replace all positive values in these two columns by 1.

- ## Creating Dummy variable for categorical variable “gender”:

  The type of feature **gender** is object. We would like to have numerical features in order to apply machine learning algorithms. We use the function **get_dummies** with drop_first = True to create dummy variables for **'gender'**. The new variable is binary variable **'gender_Male'** suggesting whether the director gender is male (=1) or female(=0).

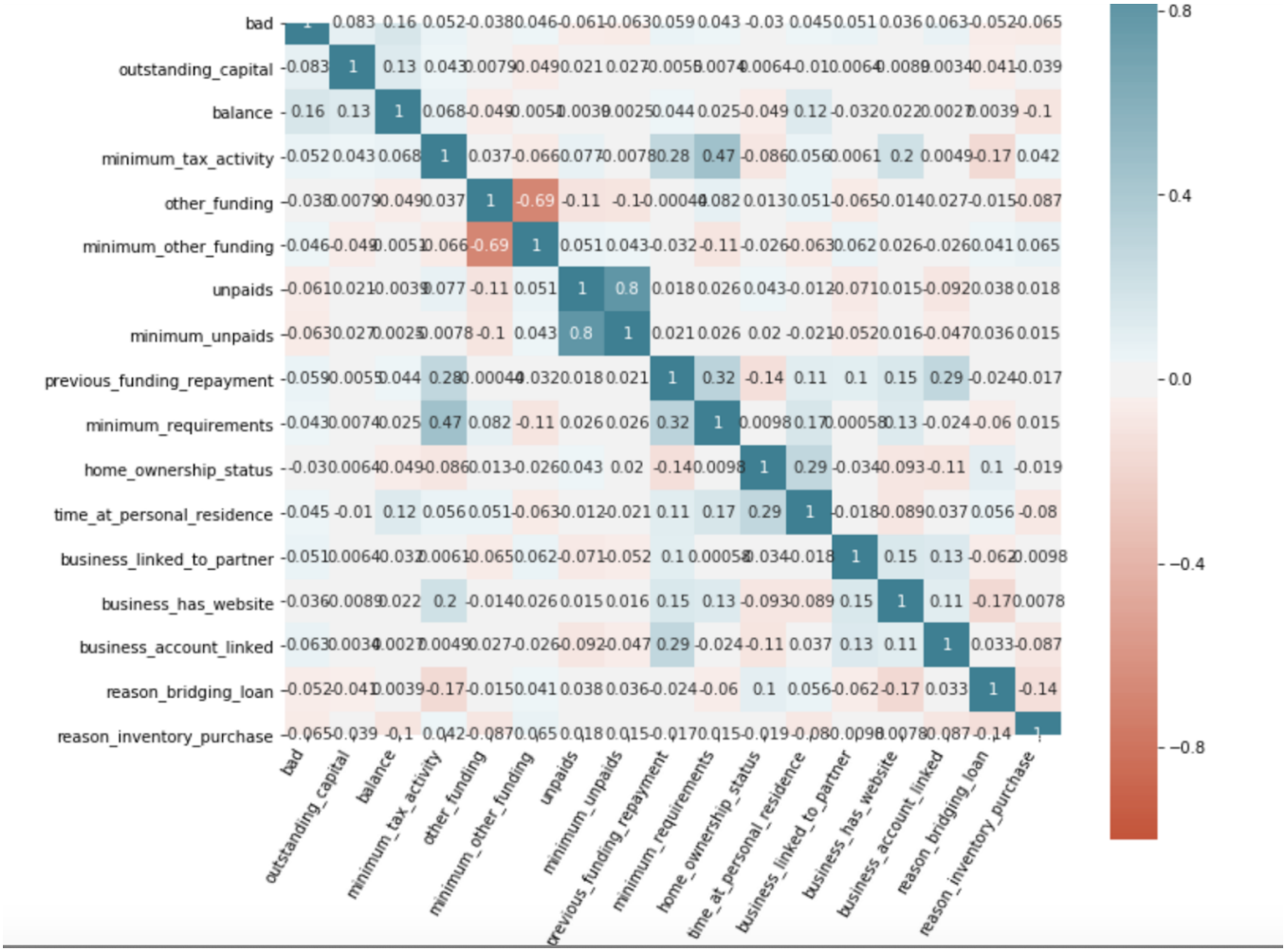
# Variables Selection

Out of 45 columns in **df**, **'bad'** is the target variable and 44 others are independent variables. Not all variables(features) are going to have an impact on the target **'bad'**. If we add these irrelevant variables in the model, it will just make the model worse. We want to know which variables potentially affect the target **'bad'**. In order to do that, we have to look at the correlation between these variables.




We observe that the relation with the target **'bad'** is strong in some variables and weak in others. We are only interested in variables which have some kind of relation with the target variable 'bad'. There are 16 variables (features) whose correlation with the target is more than 0.03. They are: **outstanding_capital , balance , minimum_tax_activity , other_funding , minimum_other_funding , unpaids , minimum_unpaids , previous_funding_repayment , minimum_requirements , home_ownership_status, time_at_personal_residence, business_linked_to_partner , business_has_website, business_account_linked, reason_bridging_loan, reason_inventory_purchase.**

We then look at the correlations between these variables to find the one highly correlated. If variables are highly correlated with each other, then we need to keep only one and drop the rest.




From the above heat map, it is seen that three pairs of features have high correlation. They are :

- **'minimum_unpaids'** and **'unpaids'**
- **'minimum_other_funding'** and **'other_funding'**
- **'minimum_requirements'** and **'minimum_tax_activity'**

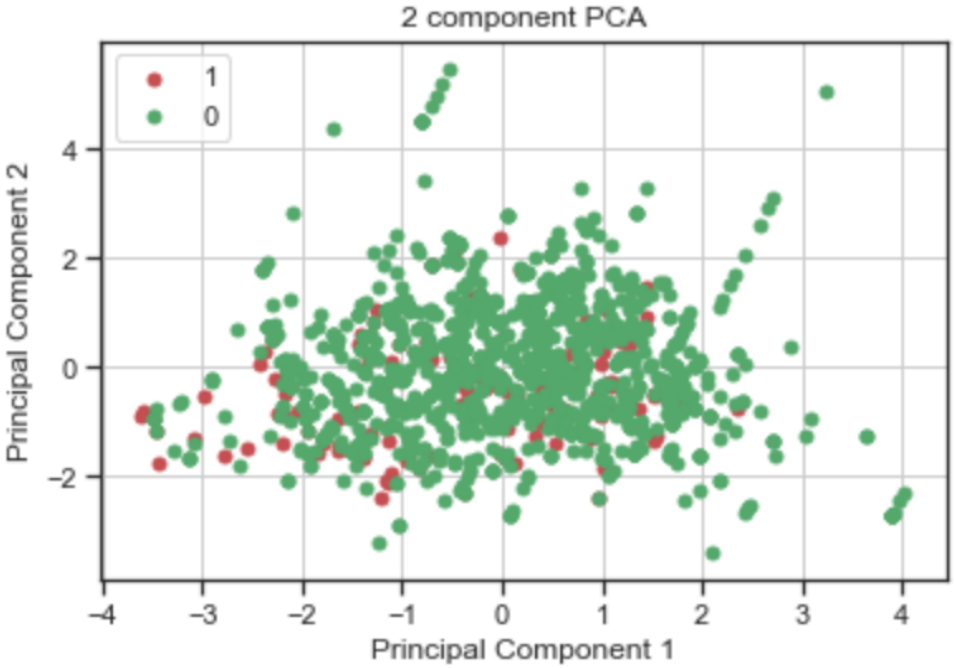
Hence, we would only keep one from each pair. We will keep **'minimum_unpaid','minimum_other_funding'and 'minimum_tax_activity'** since their correlation with **'bad'** are higher than that of **'unpaids','other_funding'** and **'minimum_requirements'**.

# Dimensionality Reduction PCA

The idea of PCA is to reduce the dimensionality of a dataset, while retaining as much variance in the data as possible.

All the variables should be on the same scale before applying PCA, otherwise a feature with large values will dominate the result. We first use **StandardScaler** in scikit-learn to standardize the data set's features onto unit scale (mean = 0 and variance = 1). Then we use **PCA** from **sklearn.processing** to fit our data. There are 13 features in the original data. So PCA will provide the same number of principal components. Use **pca.explained_variance*ratio,*** we get a list which reveals the amount of variance explained by principal components. The amount of variance explained by PC1, PC2, …, PC13 are respectively [0.1270347 , 0.10983186, 0.09469, 0.0879411 , 0.08444432, 0.07915834, 0.07561585, 0.06976647, 0.06185712, 0.05993513, 0.05675432, 0.04821427, 0.04475653]

The first principal component explains 12.7% of the variances, the second principal component explains 11% of the variances. The following graph describes how our data is classified using only the first and the second principal components.




To explain about 95% of the variance of the data, we need to keep all 13 features.

# Analyze by Pivoting Features

We can quickly analyze our feature correlations by pivoting features against each other. We can only do so at this stage for features which do not have any empty values. It also makes sense doing so only for features which are categorical or discrete type.

| | minimum_tax_activity | bad |
|---|---|---|
| 1 | 1.0 | 0.120773 |
| 0 | 0.0 | 0.088670 |

The group minimum_tax_activity = 1 has 12.08% missing payments while the group minimum_tax_activity = 0 has 8.87% missing payments. Group minimum_tax_activity = 1 tend to miss payments more.

| | minimum_other_funding | bad |
|---|---|---|
| 1 | 1.0 | 0.117816 |
| 0 | 0.0 | 0.078431 |

The group minimum_other_funding = 1 has 11.78% missing payments while the group minimum_other_funding = 0 has 7.84% missing payments. Group minimum_other_funding = 1 tend to miss payments more.

| | minimum_unpaids | bad |
|---|---|---|
| 0 | 0.0 | 0.161677 |
| 1 | 1.0 | 0.103608 |

The group minimum_unpaids = 0 has 16.17% missing payments while the group minimum_unpaids = 0 has 10.36% missing payments. Group minimum_unpaids = 0 tend to miss payments more.

| | reason_bridging_loan | bad |
|---|---|---|
| 0 | 0 | 0.126136 |
| 1 | 1 | 0.087983 |

The group reason_bridging_loan = 0 has 12.61% missing payments while the group reason_bridging_loan = 0 has 8.80% missing payments. The group getting the loan for bridging_loan purpose is less likely to miss the payments.

| | home_ownership_status | bad |
|---|---|---|
| 1 | 0.25 | 0.131034 |
| 2 | 0.50 | 0.121053 |
| 0 | 0.00 | 0.114600 |
| 3 | 1.00 | 0.076503 |

The group home_ownership_status = 1 has the lowest rate of missing payments. It means the directors who own 100% of their house are less likely to miss payments.

| | time_at_personal_residence | bad |
|---|---|---|
| 4 | 1.00 | 0.128866 |
| 3 | 0.75 | 0.126437 |
| 2 | 0.50 | 0.106061 |
| 0 | 0.00 | 0.095238 |
| 1 | 0.25 | 0.083333 |

The longer the director lives in his home, the higher the rate of missing payments.

| | business_linked_to_partner | bad |
|---|---|---|
| 1 | 1.0 | 0.177215 |
| 0 | 0.0 | 0.109005 |

The group business_linked_to_partner = 1 has 17.72% missing payments while the group business_linked_to_partner = 0 has 10.90% missing payments. Group minimum_tax_activity = 1 is more likely to miss payments.

| | business_has_website | bad |
|---|---|---|
| 1 | 1.0 | 0.123306 |
| 0 | 0.0 | 0.100494 |

The group business_has_website = 1 has 12.33% missing payments while the group business_has_website = 0 has 10.04% missing payments. Group business_has_website = 1 tend to miss payments more but the difference is very slight.

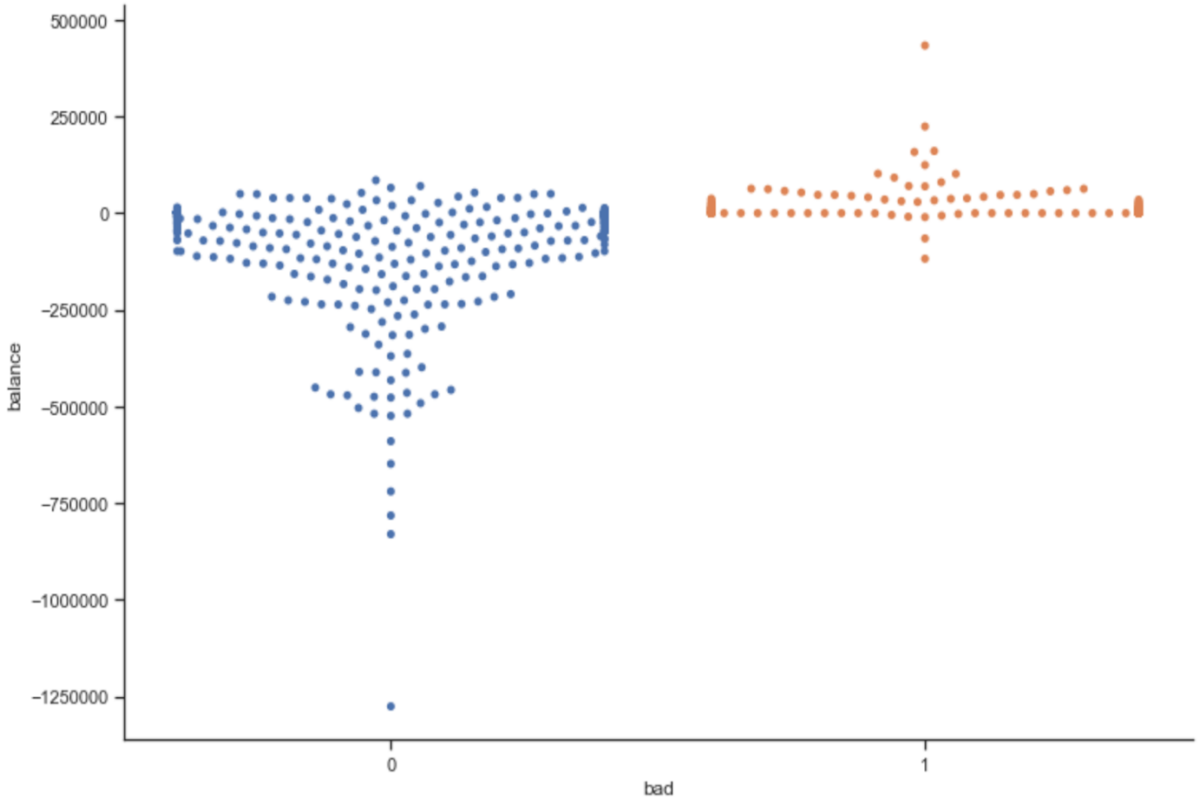
| | business_account_linked | bad |
|---|---|---|
| 1 | 1.0 | 0.131653 |
| 0 | 0.0 | 0.091918 |

The group business_account_linked = 1 has 13.16% missing payments while the group business_account_linked = 0 has 9.19% missing payments. The group business_account_linked = 1 tends to miss payments more.

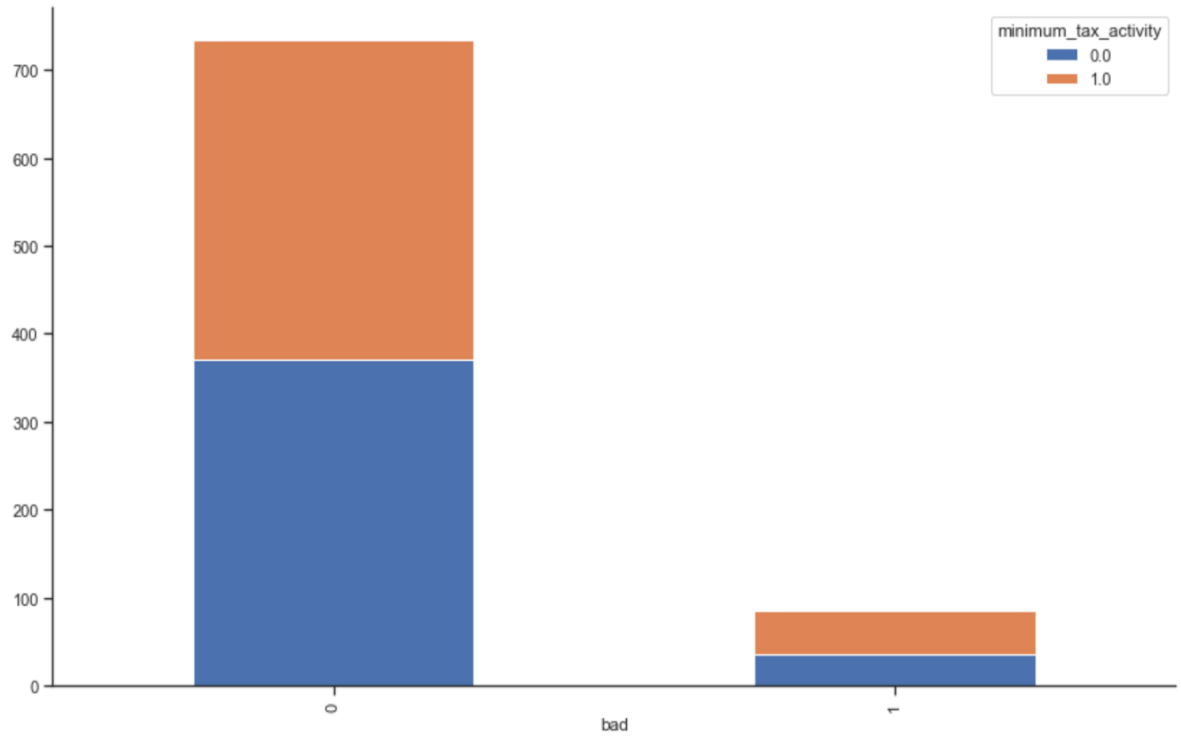
| | reason_inventory_purchase | bad |
|---|---|---|
| 0 | 0 | 0.124844 |
| 1 | 1 | 0.095413 |

The group reason_inventory_purchase = 0 has 12.48% missing payments while the group reason_inventory_purchase = 0 has 9.54% missing payments. The group getting the loan for inventory purpose is less likely to miss the payments.

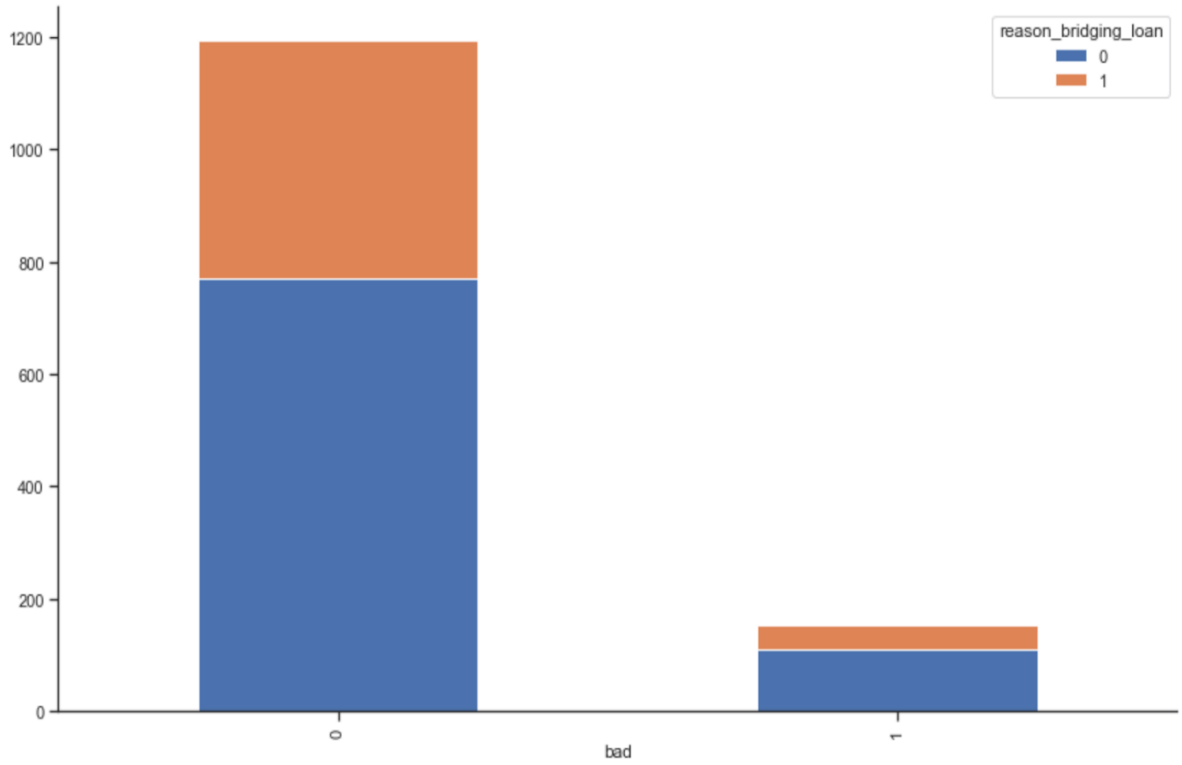
# Data Visualization



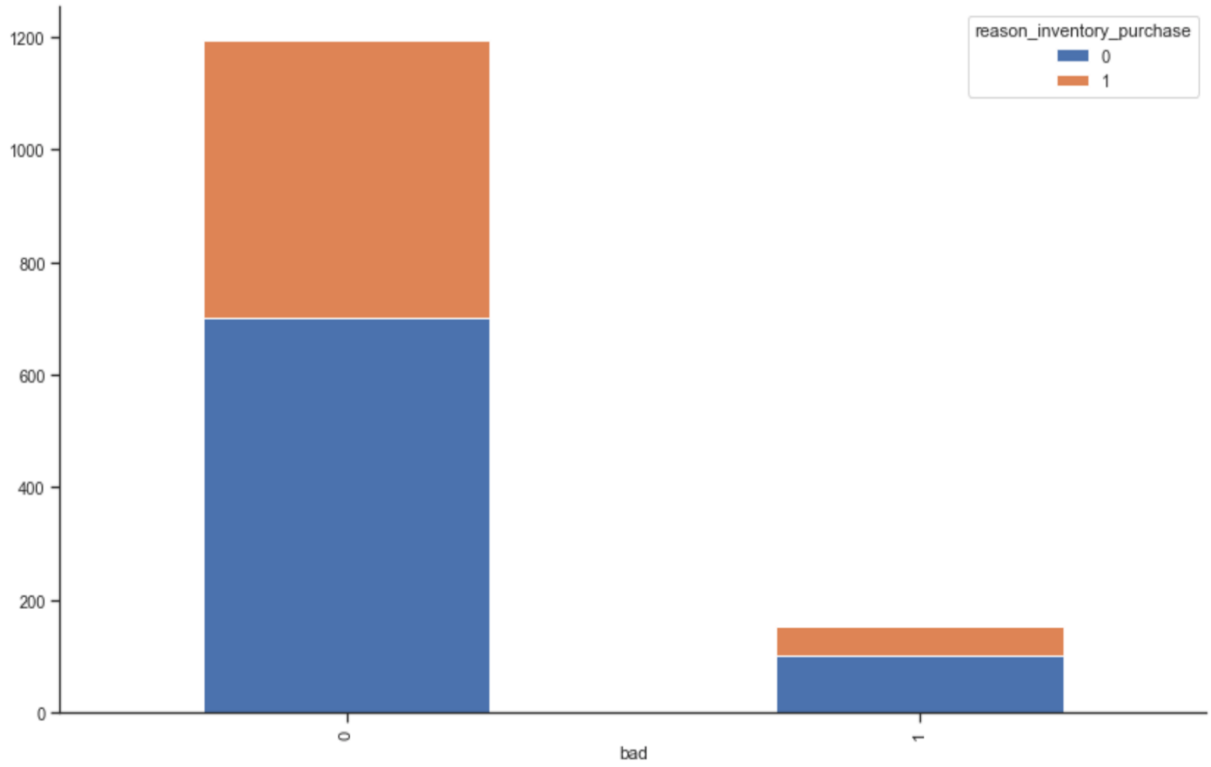

The above swarmplot shows that observations in the not missing payments group are more likely to have negative 'balance' and observations in the missing payments group are more likely to have positive 'balance'. Which makes sense here since companies with arrears tend to miss payments more.




In the group of not missing payments, 49.6% submitted tax returns('minimum_tax_activity' = 1). While in the group of missing payments, 58% submitted tax returns. The group of missing payments tend to submit tax more.



In the group of not missing payments, 35.59 % get the loan for bridging reasons. In the group of missing payments, 26.97 % get the loan for bridging reasons




In the group of not missing payments,41.29 % get the loan for inventory purchase reasons. In the group of missing payments, 34.21% get the loan for inventory purchase reasons

# Machine Learning Algorithms.

Now we are ready to train a model and predict the required solution. Here,

y = value of target feature 'bad'

and

X= all the other features (or independent variables, predictors or explanatory variables)

which we will use to fit a machine learning model and predict the value of y.

There are so many predictive modelling algorithms out there to choose from. We must understand the type of problem and solution requirement to narrow down to a select few models which we can evaluate. Our problem is a classification problem. We want to identify the relationship between the target feature (**bad**) with other variables or features. We are also performing a category of machine learning which is called supervised learning as we are training our model with a labeled dataset.

With these two criteria - Supervised Learning plus Classification, we can narrow down our choice of models to a few. These include:

- K_Nearest_Neighbor
- Logistic Regression
- Support Vector Machine
- Stochastic Gradient Descent
- Random Forest
- Decision Tree
- Naive Bayes

The metrics we use to compare the models are: accuracy, precision, recall, F1-score and area under ROC curve.

| | **Algorithm** | **Data** | **Accuracy** | **Precision** | **Recall** | **f1-score** | **AUC** |
|---|---|---|---|---|---|---|---|
| **0** | K-Nearest Neighbor | Original (n-neighbors=5) | 0.93 | 0.93 | 0.93 | 0.91 | 0.7503 |
| **1** | K-Nearest Neighbor | Oversampling (n-neighbors=2) | 0.84 | 0.86 | 0.84 | 0.85 | 0.6912 |
| **2** | K-Nearest Neighbor | Undersampling (n-neighbors=6) | 0.78 | 0.86 | 0.78 | 0.81 | 0.7234 |
| **3** | Logistic Regression | Original | 0.91 | 0.91 | 0.91 | 0.88 | 0.7325 |
| **4** | Logistic Regression | Oversampling | 0.92 | 0.91 | 0.92 | 0.91 | 0.7402 |
| **5** | Logistic Regression | Undersampling | 0.92 | 0.91 | 0.92 | 0.91 | 0.7601 |
| **6** | Support Vector Machine | Original | 0.93 | 0.93 | 0.93 | 0.91 | 0.7698 |
| **7** | Support Vector Machine | Oversampling | 0.87 | 0.88 | 0.87 | 0.87 | 0.7448 |
| **8** | Support Vector Machine | Undersampling | 0.92 | 0.91 | 0.92 | 0.91 | 0.7422 |
| **9** | Stochastic Gradient Descent | Original | 0.93 | 0.93 | 0.93 | 0.92 | 7.009 |
| **10** | Stochastic Gradient Descent | Oversampling | 0.86 | 0.87 | 0.86 | 0.87 | 0.6997 |

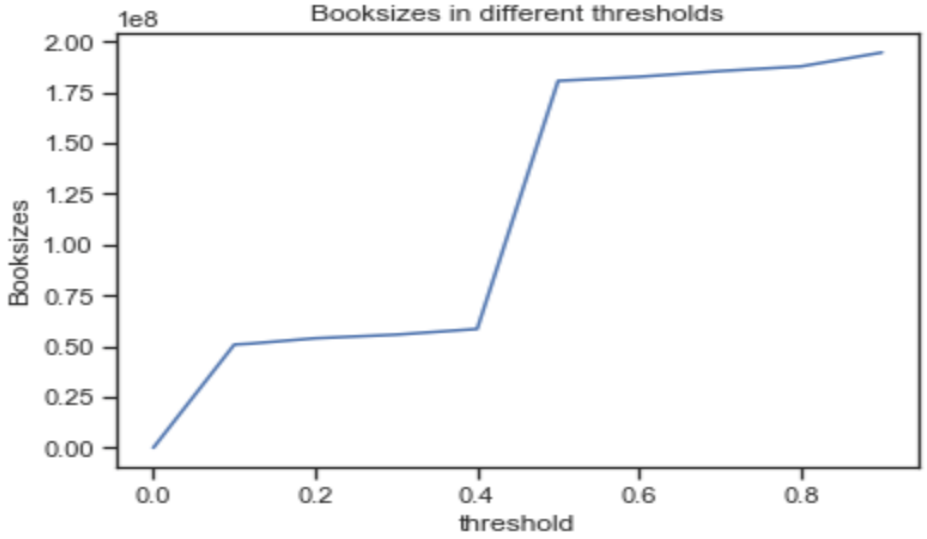
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **11** | Stochastic Gradient Descent | Undersampling | 0.55 | 0.85 | 0.55 | 0.63 | 0.6137 |
| **12** | Naïve Bayes | Original | 0.88 | 0.79 | 0.88 | 0.83 | 0.7414 |
| **13** | Naïve Bayes | Oversampling | 0.26 | 0.91 | 0.26 | 0.29 | 0.737 |
| **14** | Naïve Bayes | Undersampling | 0.91 | 0.89 | 0.91 | 0.89 | 0.7257 |
| **15** | Decision Tree | Original | 0.94 | 0.94 | 0.94 | 0.92 | 0.7277 |
| **16** | Decision Tree | Oversampling | 0.79 | 0.87 | 0.79 | 0.82 | 0.7567 |
| **17** | Decision Tree | Undersampling | 0.74 | 0.85 | 0.74 | 0.78 | 0.6804 |
| **18** | Random Forest | Original | 0.91 | 0.9 | 0.91 | 0.9 | 0.731 |
| **19** | Random Forest | Oversampling | 0.87 | 0.88 | 0.87 | 0.87 | 0.754 |
| **20** | Random Forest | Undersampling | 0.92 | 0.91 | 0.92 | 0.91 | 0.6725 |

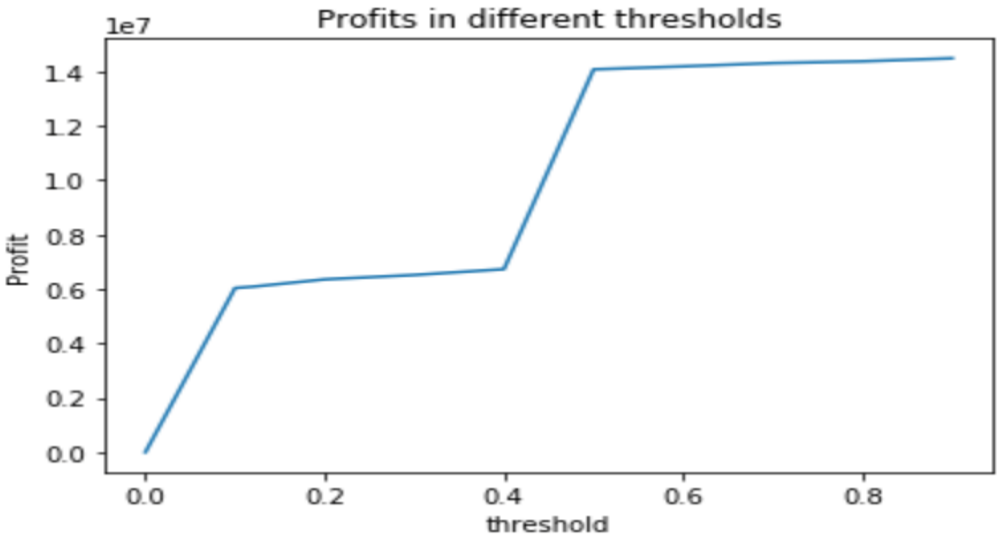
Based on the summary table, Logistic Regression on Undersampling train set is the best model.

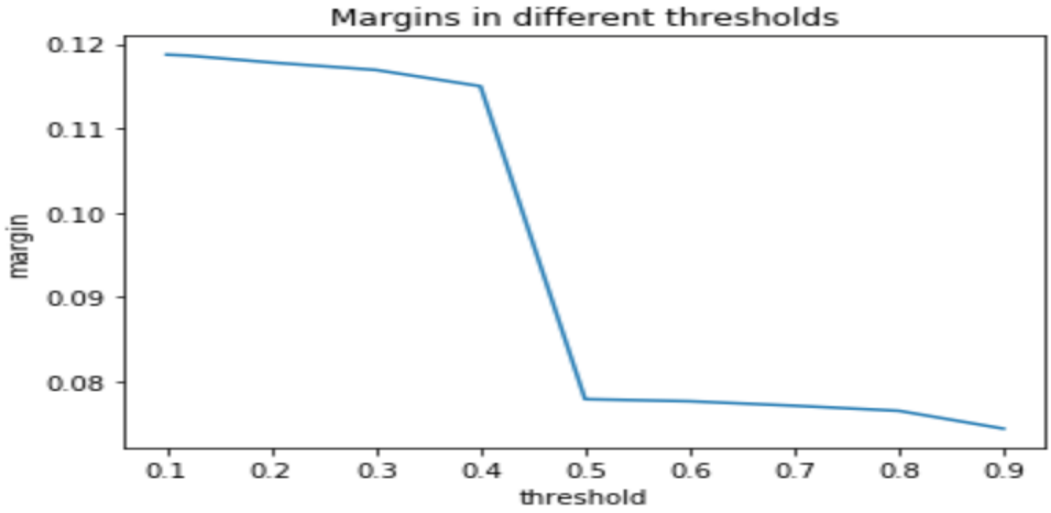
# Disbursement & Threshold

The disbursement table is provided which contains the disbursement amount requested.

By using **predict_proba** we are going to figure out the probability that a company is going to miss payments. By the default, the cut off is 0.5. If a company has more than 50% chance to miss payments, it is label 1('bad' = 1). Is 0.5 the best threshold? We are going to calculate the booksize, profit and margin at different thresholds and pick out the one that works the best for the lender.






As the threshold increases, the profit increases but the booksize also increases. Therefore we have to look at the margins to see the proportion of the profit over the booksize. In theory, we should choose the threshold corresponding to the highest margin. But in reality, we have to take the profit into consideration. When the threshold is 0.1, we get the highest margin (0.1187), the booksize is 50858820 and the profit is 6039308. When the threshold is 0.4, the margin is 0.115, the booksize is 58529835 and the profit is 6731253. Although the margin at threshold 0.4 is lower, it is the best threshold if we put the profit in consideration.