Bank Loan Default: Part 1

Presented By: Kho Guan Guo

Agenda

- Background and Problem Statement
- How I started and how I ended up with this project
- EDA
- Feature Engineering
- Modelling
- Conclusion
- Recommendations

X 3

Background

- Banks run into losses when a customer doesn't pay their loans on time that can run into the **MILLIONS** every year.
- The bank runs the risk of losing potential business if it rejects a loan application and the prediction of default is wrong.

Problem Statement

- Using the given dataset, this project aims to achieve 3 things as its Primary objectives.

1. To utilize the information given and quantify feature importance to accurately predict loan defaults.

2. To engineer features to help better predict loan defaults.

3. To act as a stepping stone to develop better models to be deployed, that can address this issue that banks have and reduce monetary loss.

Starting point

- Originally started out wanting to make a model to deploy with 2 goals:

1. For banks to use by keying in customer information and getting prediction which will be used in the assessment of the loan application.

2. For banks to let customers do their homework online before applying for loans in order to cut operation time. Customers would key in their information and know the probable outcome.

Metrics and Models

- I will be using Accuracy, F1 and Log Loss scores in order to evaluate the models in this project.

- Models used: Multinomial Naïve Bayes, Logistic Regression, Random Forest Classifier and XGBoost.

EDA

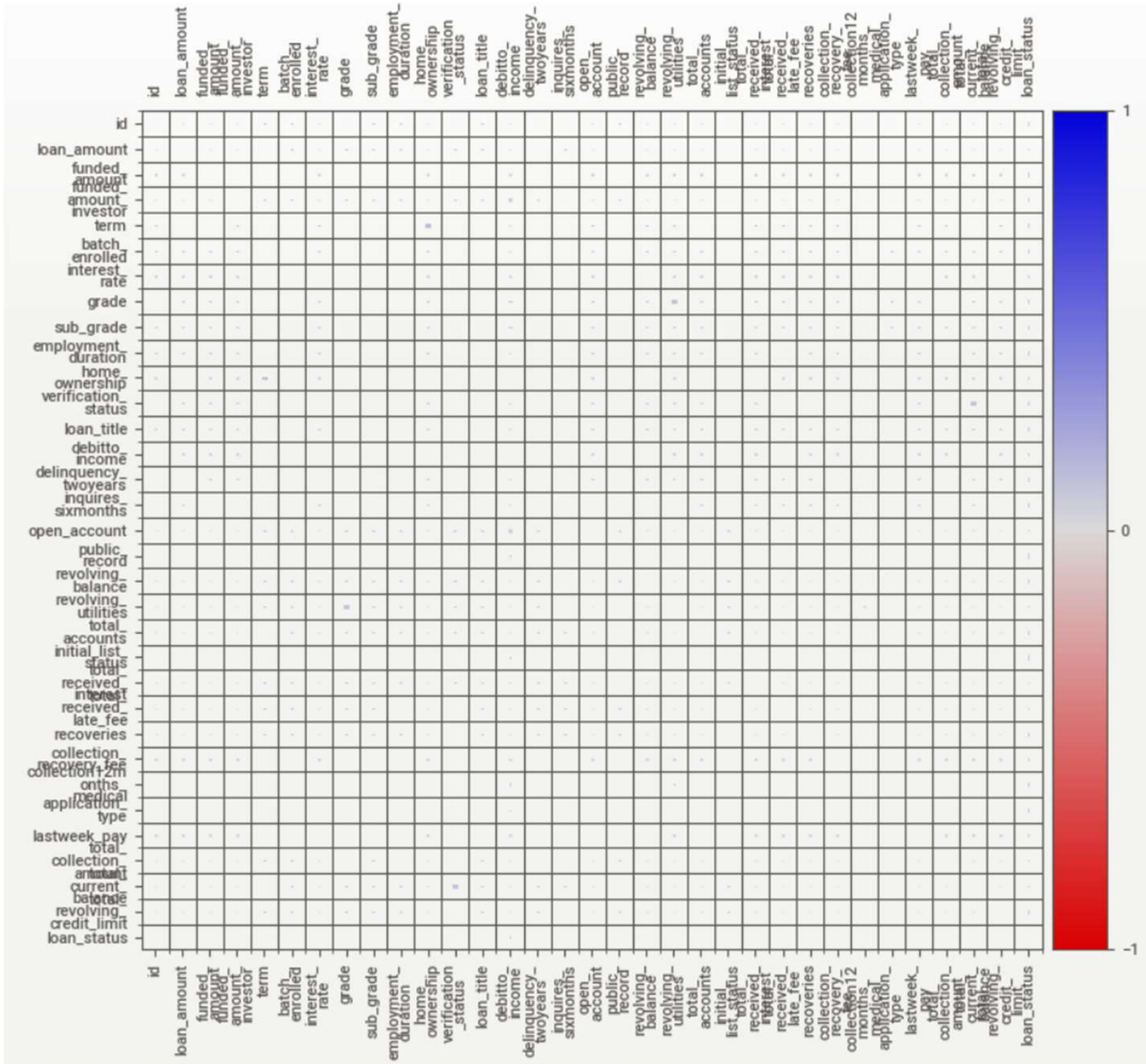
- 35 features including the target column
- 8 categorical
- 26 continuous
- Imbalanced dataset with a 1:10 ratio in target column

Associations

[Only including dataset "DataFrame"]

■ **Squares** are categorical associations (uncertainty coefficient & correlation ratio) from 0 to 1. The uncertainty coefficient is **assymmetrical**, (i.e. ROW LABEL values indicate how much they PROVIDE INFORMATION to each LABEL at the TOP).

• **Circles** are the symmetrical numerical correlations (Pearson's) from -1 to 1. The **trivial diagonal** is intentionally left blank for clarity.


id
loan_amount
funded_amount
funded_amount_investor
term
batch_enrolled
interest_rate
grade
sub_grade
employment_duration
home_ownership
verification_status
loan_title
debitto_income
delinquency_twoyears
inquires_sixmonths
open_account
public_record
revolving_balance
revolving_utilities
total_accounts
initial_list_status
total_received_interest
total_received_late_fee
recoveries
collection_recovery_fee
collection12months_medical
application_type
lastweek_pay
total_collection_amount
total_current_balance
total_revolving_credit_limit
loan_status
1
0
-1

Numerical Features

- Boxplots: Yielded no discernible patterns
- Histplots: Yielded no discernible patterns
- Scatterplots: Yielded no discernible patterns
- We’re going to have to engineer some features

Categorical Features

- Checking number of defaults as a percentage of the total number of those feature value observations.
- Yielded some more obvious patterns than the continuous features did by looking at the difference in percentages, although they were generally normally distributed across most unique values.

Feature Engineering

- loan_title: reduced number of unique values from 109 to 17 by broadly categorizing them
- batch_enrolled: 41 batches reduced to 3 groups
- grade: 7 grades to 3 groups
- sub_grade: 35 sub grades to 4 groups
- Added arithmetic features

Modelling

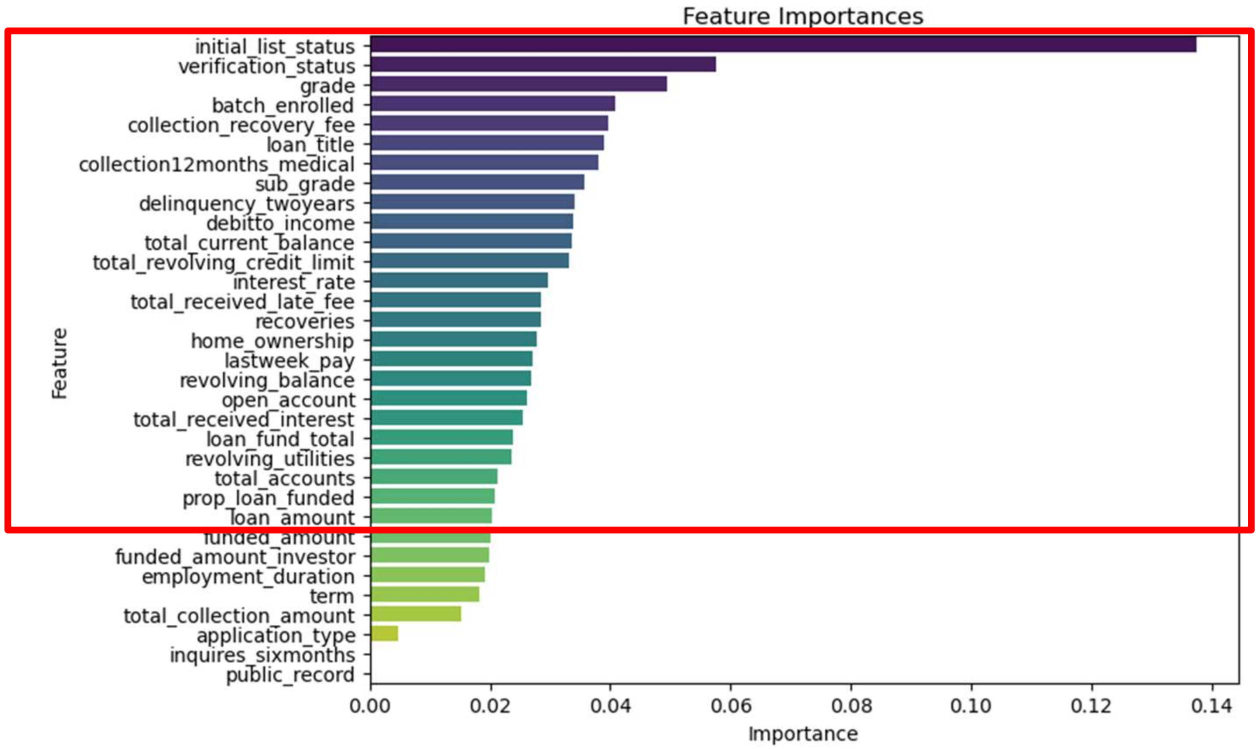
classifier	cv_train	roc_auc_train	roc_auc_val	accuracy_train	accuracy_val	f1_val	f1_train	log_loss_train	log_loss_val
MultinomialNB()	16.8180	0.5130	0.5142	0.5177	0.5204	0.1597	0.1628	16.8180	16.8464
LogisticRegression()	0.6232	0.4881	0.4827	0.6544	0.6489	0.1268	0.1358	0.6232	0.6231
RandomForestClassifier(random_state=42)	0.5537	0.7201	0.4971	0.7729	0.7348	0.1234	0.2296	0.5537	0.5578
XGBClassifier(base_score=None, booster=None, c...	0.5173	0.5928	0.5042	0.7776	0.7615	0.1274	0.1560	0.5173	0.5120

- Also ran a LightGBM model but it took 5 hours to run and the results were similar to XGBoost.

Feature Engineering

- Converted all continuous features into categorical ones by binning each feature into 4 ranges of values before modelling again.
- Used the top 25 features from the resulting to try modelling again for a better score.

Feature Engineering



Modelling

classifier	cv_train	roc_auc_train	roc_auc_val	accuracy_train	accuracy_val	f1_val	f1_train	log_loss_train	log_loss_val
MultinomialNB()	0.6759	0.5054	0.5110	0.5399	0.5412	0.1603	0.1583	0.6759	0.6782
LogisticRegression()	0.5796	0.4889	0.4875	0.6969	0.6981	0.1310	0.1325	0.5796	0.5836
RandomForestClassifier(random_state=42)	0.5402	0.7535	0.4960	0.7832	0.7470	0.1282	0.2458	0.5402	0.5479
XGBClassifier(base_score=None, booster=None, c...	0.5162	0.5635	0.4902	0.7727	0.7583	0.1160	0.1471	0.5162	0.5231
MultinomialNB()	0.6813	0.4984	0.5040	0.5315	0.5350	0.1569	0.1539	0.6813	0.6817
LogisticRegression()	0.5913	0.4933	0.4898	0.6842	0.6824	0.1341	0.1373	0.5913	0.5962
RandomForestClassifier(random_state=42)	0.5492	0.7486	0.4969	0.7708	0.7304	0.1322	0.2509	0.5492	0.5584
XGBClassifier(base_score=None, booster=None, c...	0.5371	0.5645	0.4913	0.7494	0.7358	0.1221	0.1535	0.5371	0.5484

Final scores on test data with best model

```
Log Loss Score: 0.52156
Accuracy Score: 0.7605
F1 Score: 0.1246
```

Conclusion

- Just as a benchmark, the Log Loss score for the top performer in this competition was between 0.34 to 0.35.

- The process of feature engineering when dealing with a dataset like this is important in getting better scores.

- Binning features and making them categorical seems to work especially well with Random Forest Classifier and XGBoost.

Recommendations

- The model is not yet the finished product to be deployed, but can be used as a base to continue improving.

- Feature engineering based on the feature importance from the best model will play a big role in the next steps in the next phase of the project which I will continue to work on over the next few weeks.

- Obtaining the right data from customers to be used as features in the model besides what was provided might prove more useful than tweaking existing data.

- In order to get the model good enough to deploy, a few things need to be accomplished.

1. Feature engineering and continued tuning to get better Accuracy and F1 scores
2. Reduction of features such that the deployment feature ranges are easily obtainable by bank clients and the bank.
3. Model needs to run fast in order to be deployed for customer use.

Questions?

To Be Continued…