Credit Risk Analysis

By: Sama Amr Gouda

Table of Contents

01
INTRODUCTION

02
DATA DESCRIPTION

03
DATA PREPROCESSING

04
MODELLING

05
TEST DATA

06
GUI

07
FUTURE WORK

01
Introduction

What's Credit Default

- Clients with payment difficulties
  - Having late payments with more than X days on at least one of the first installments of the loan in our sample

- This has been increasing with the dramatic fluctuations in the worldwide economic state.

- There has been a drastic level of workload in order to meet this increased demand, which has slowed the process of issuing loans.

Problem Statement

Use Statistical and Machine learning techniques
to predict if a client will default or not based on
their historical behaviour and their current state

02

Data Description

Data Description

The data consists of 8 files each containing different sets of data:

Application_train

Training data with application info of each client with target variable

Application_test

Testing data with application info of each client without target variable

Bureau

All client's previous credits provided by other financial institutions

Bureau Balance

Monthly balances of previous credits in Credit Bureau.

Data Description

The data consists of 8 files each containing different sets of data:

Previous Application

All previous applications for Home Credit loans of clients who have loans in our sample.

Installments Payments

Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.

POS Cash Balance

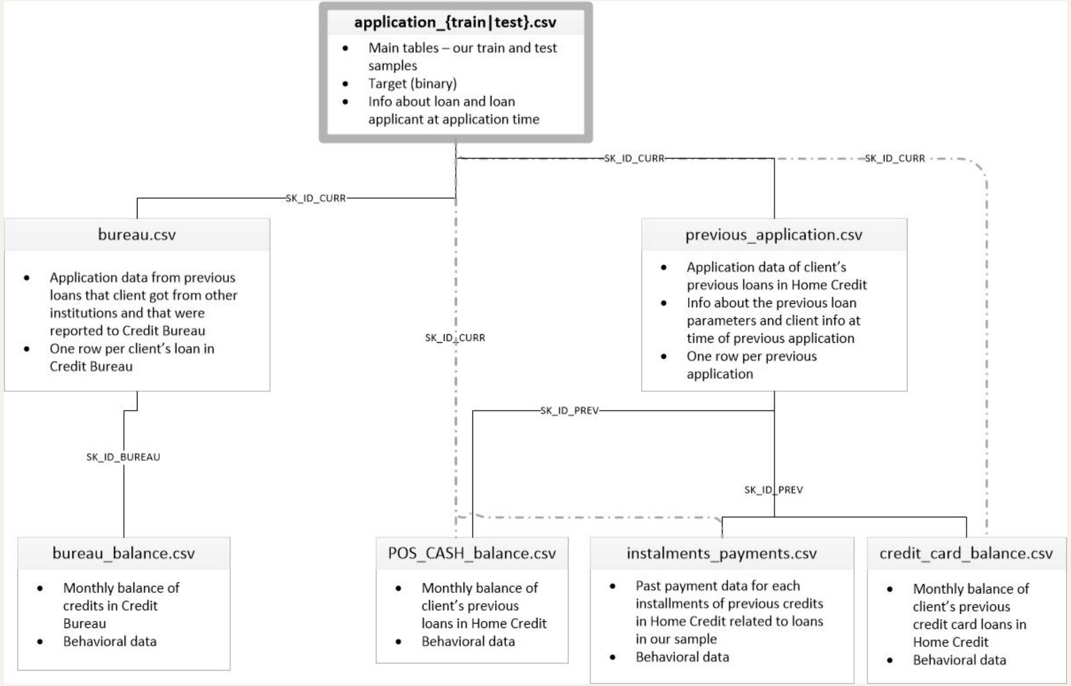
Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.

Credit Card Balance

Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.

Data Description

Here’s how the datasets are connected



application_{train|test}.csv

- Main tables – our train and test samples
- Target (binary)
- Info about loan and loan applicant at application time

bureau.csv

- Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
- One row per client’s loan in Credit Bureau

previous_application.csv

- Application data of client’s previous loans in Home Credit
- Info about the previous loan parameters and client info at time of previous application
- One row per previous application

bureau_balance.csv

- Monthly balance of credits in Credit Bureau
- Behavioral data

POS_CASH_balance.csv

- Monthly balance of client’s previous loans in Home Credit
- Behavioral data

instalments_payments.csv

- Past payment data for each installments of previous credits in Home Credit related to loans in our sample
- Behavioral data

credit_card_balance.csv

- Monthly balance of client’s previous credit card loans in Home Credit
- Behavioral data

03

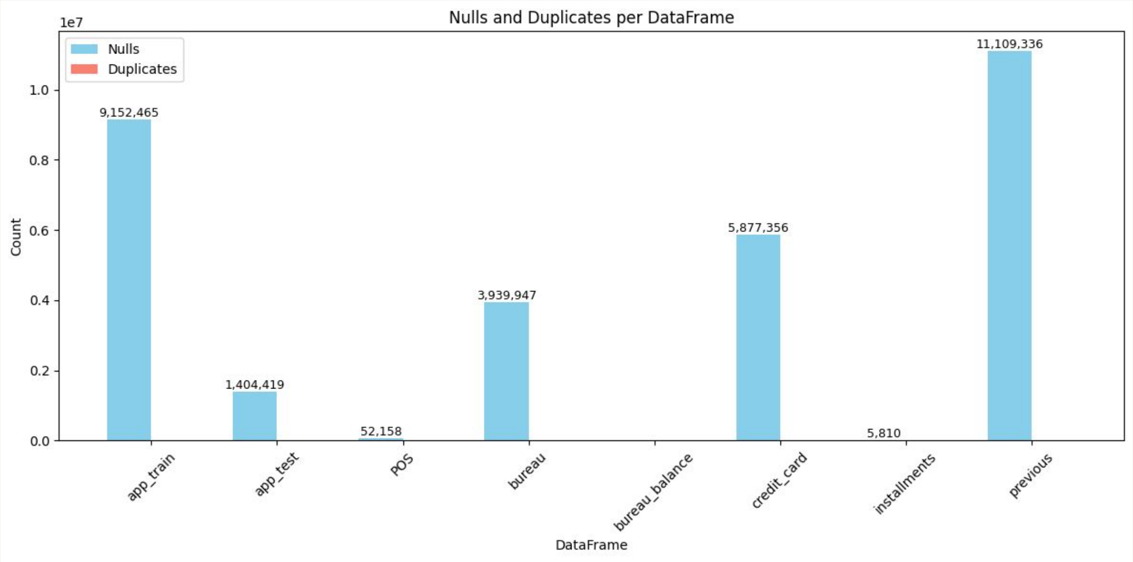
Data Preprocessing

Checking for Nulls and Duplicates

11 M **Previous**

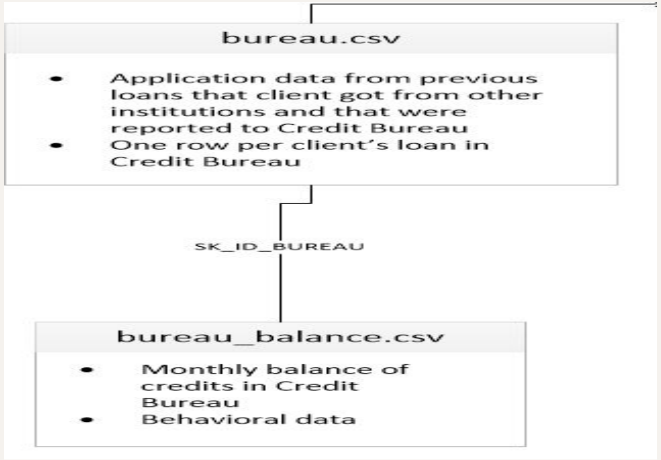
Highest number of Nulls

0 **Duplicates**



Merging Datasets

- In order to use all data at once I had to merge it:
  a. Aggregate Bureau and Bureau Balance using the mean , merged by SK_ID_BUREAU → Because there are as many rows as number of credits the client had in Credit Bureau before the application date.



bureau.csv

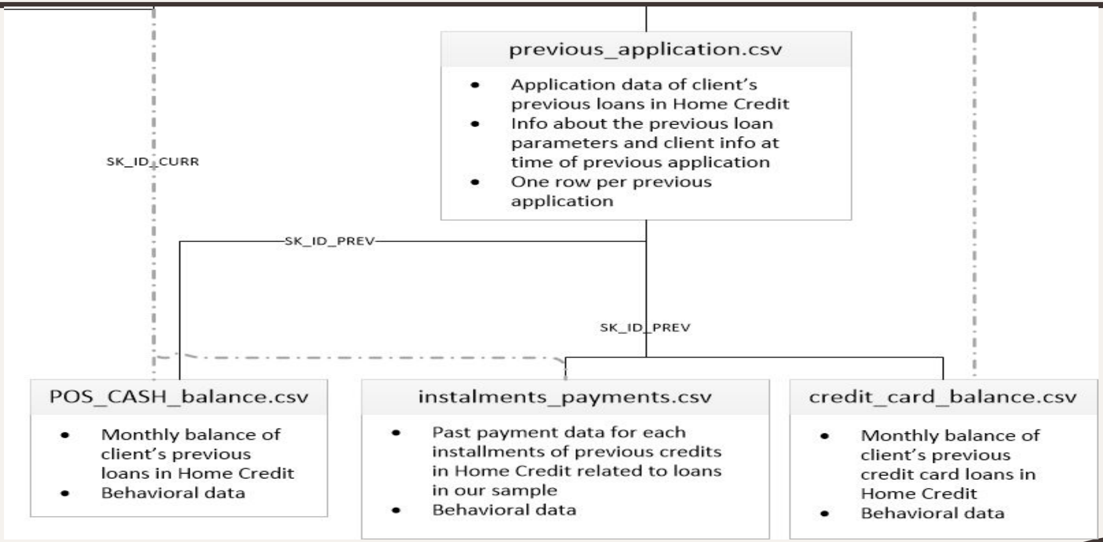
- Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
- One row per client's loan in Credit Bureau

SK_ID_BUREAU

bureau_balance.csv

- Monthly balance of credits in Credit Bureau
- Behavioral data

Merging Datasets



previous_application.csv

- Application data of client’s previous loans in Home Credit
- Info about the previous loan parameters and client info at time of previous application
- One row per previous application

POS_CASH_balance.csv

- Monthly balance of client’s previous loans in Home Credit
- Behavioral data

instalments_payments.csv

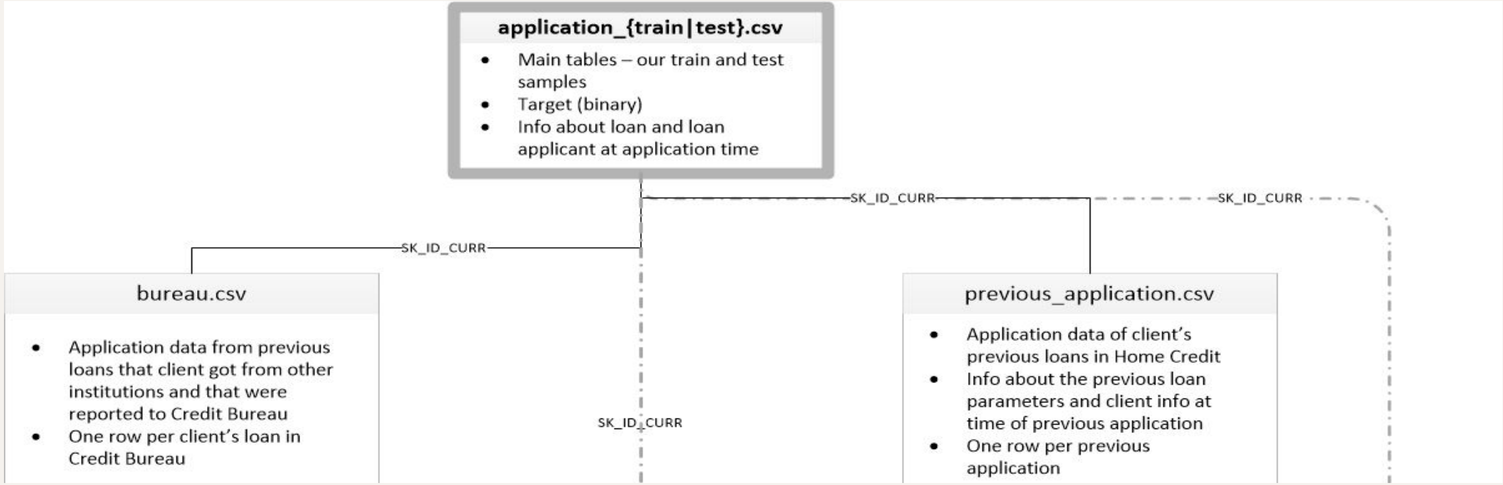
- Past payment data for each installments of previous credits in Home Credit related to loans in our sample
- Behavioral data

credit_card_balance.csv

- Monthly balance of client’s previous credit card loans in Home Credit
- Behavioral data

b. Merge the POS cash balance, installment payments, and Credit Card Balance with Previous application using SK_ID_PREV

Merging Datasets



application_{train|test}.csv

- Main tables – our train and test samples
- Target (binary)
- Info about loan and loan applicant at application time

bureau.csv

- Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
- One row per client's loan in Credit Bureau

previous_application.csv

- Application data of client's previous loans in Home Credit
- Info about the previous loan parameters and client info at time of previous application
- One row per previous application

c. Use a 'left' Join to add these onto the application train/test datasets

Merging Datasets

- In order to use all data at once I had to merge it:
    a. Aggregate Bureau and Bureau Balance using the mean , merged by SK_ID_BUREAU → Because there are as many rows as number of credits the client had in Credit Bureau before the application date.

    b. Merge the POS cash balance, installment payments, and Credit Card Balance with Previous application using SK_ID_PREV

    c. Use a 'left' Join to add these onto the application train/test datasets

Left join = returns all rows from the left table
and the matching rows from the right table

Type transformation

- Check if there are categorical variables that’ve been identified incorrectly as integers or floats
    - Check if the unique values within the variables are less than 20
    - If so: Cast ‘category’ type onto there variables

- Change variables identified as objects into ‘category’ type as well

- These Categorical variables are then Encoded using numbers from 1 to the length of present categories, in order for the models to better interpret them

Outliers

- Check if the data contains any outliers

72,217 = 23%

Max. number of outliers

In Days Employed

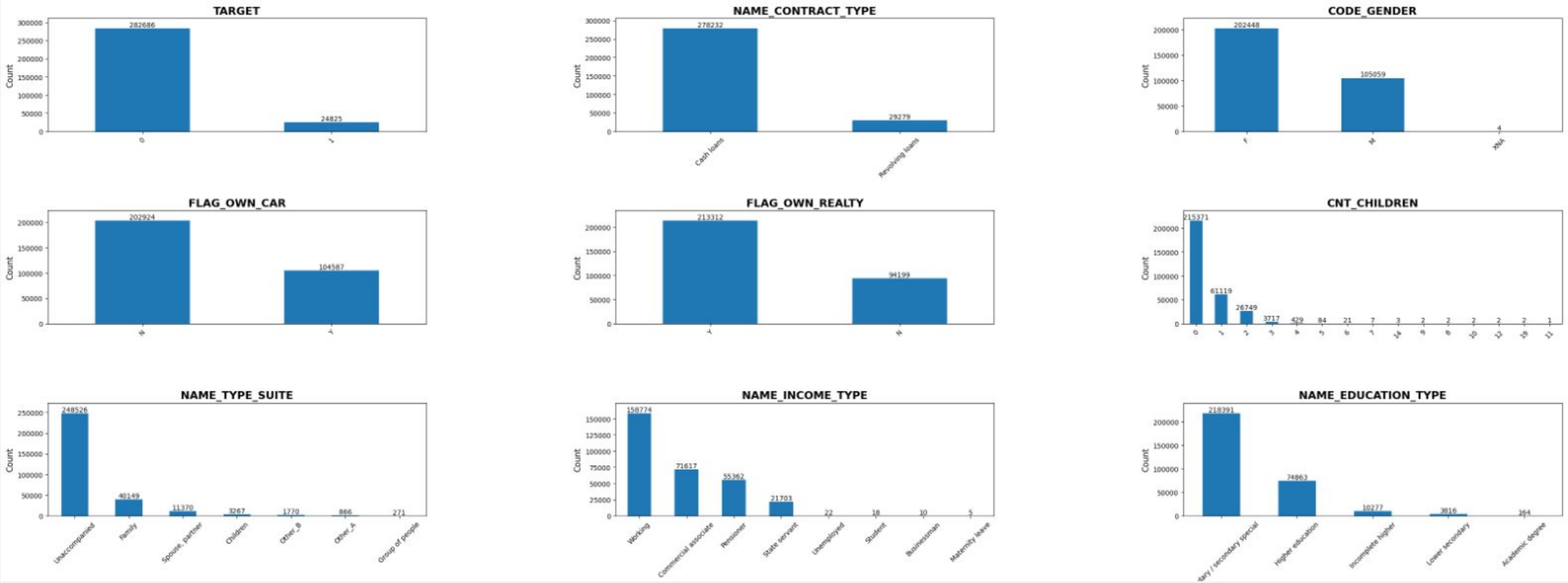
Replace by

→

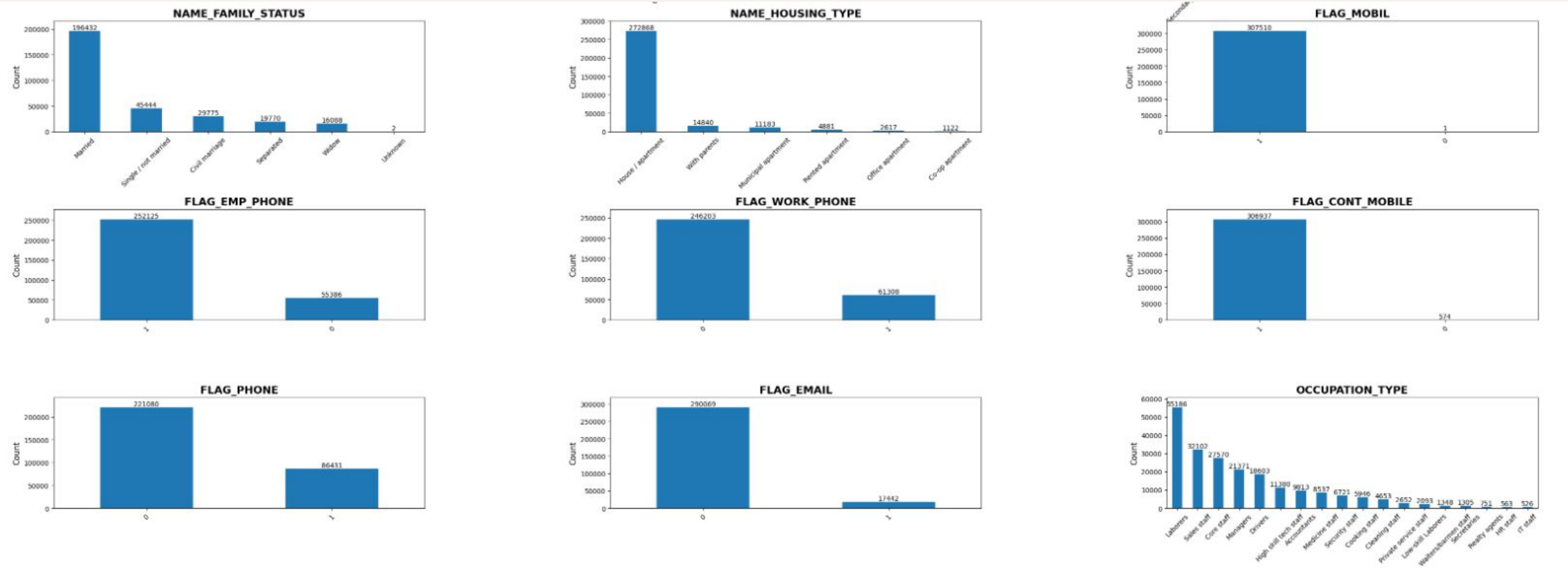
Median

More robust to outliers

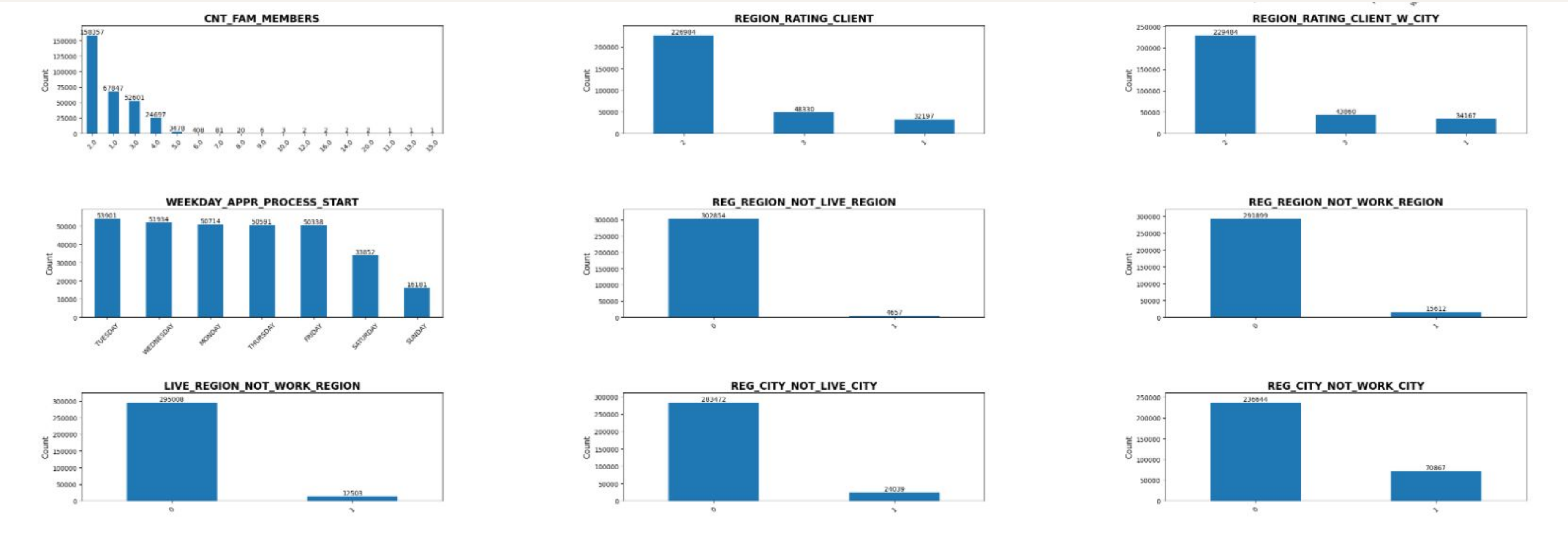
Distribution of some of Categorical Data



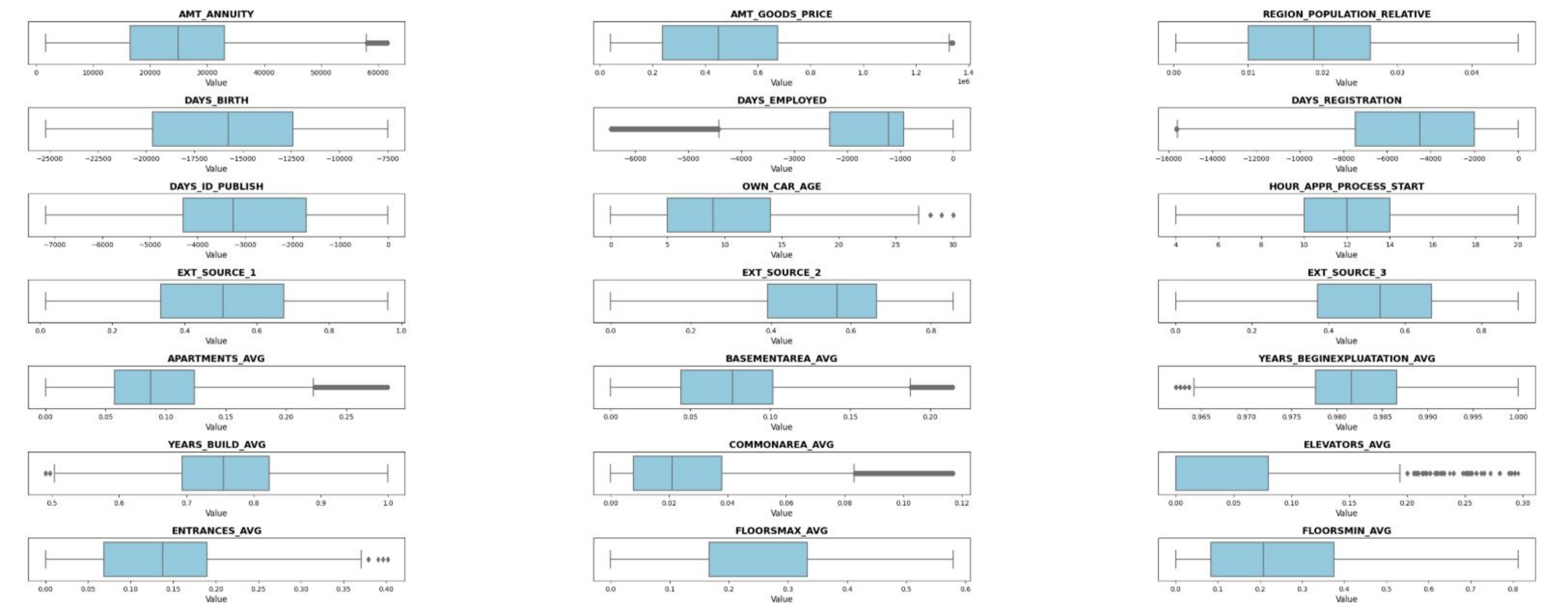
Distribution of some of Categorical Data



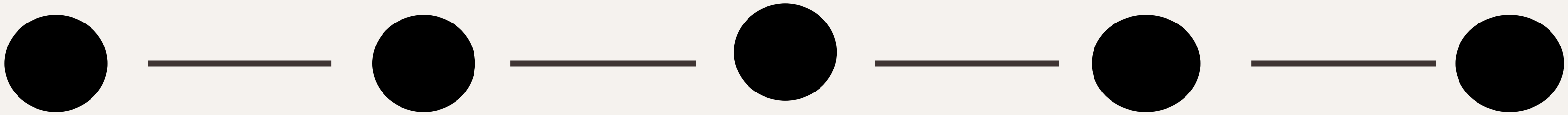
Distribution of some of Categorical Data



Distribution of some of Numeric Data



Handling data



Standardization

Using z-score

Train-Test Split

SMOTE

Synthetic Minority Over-sampling Technique

PCA

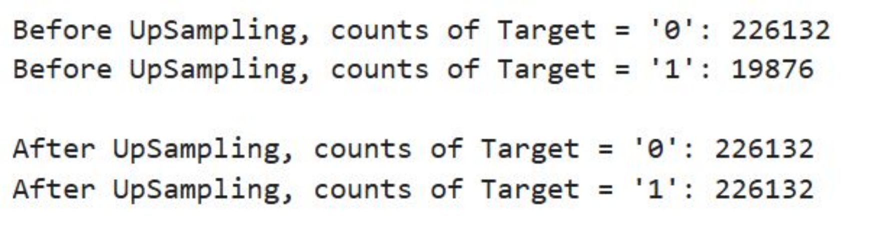
Principal Component analysis
52 components → for 95% of variance explained

Encodings

Encoding the categorical variables

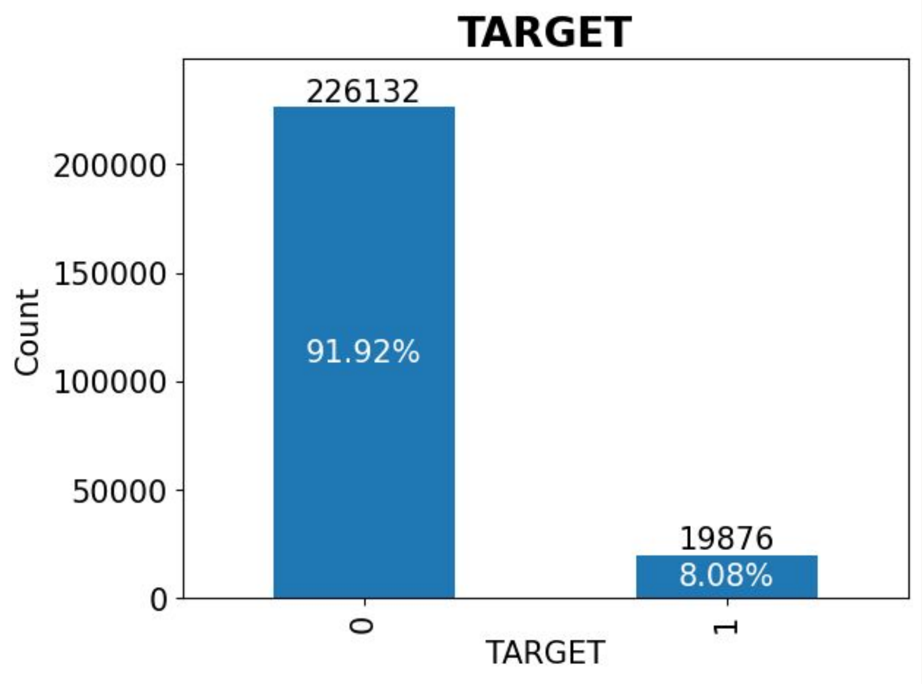
SMOTE

- Data Appears to be hugely imbalanced
- Solution is using SMOTE
- It over samples the minority class by generating new points
- Here’s how the data would look like



```
Before UpSampling, counts of Target = '0': 226132
Before UpSampling, counts of Target = '1': 19876

After UpSampling, counts of Target = '0': 226132
After UpSampling, counts of Target = '1': 226132
```



04

Modelling

Models

This is a binary classification problem

Random Forest

- Tree Based Model
- Trees Divided into Batches
- Handels non-linear data

XG-Boost

- Extreme Gradient Boosting
- Robust

Logistic Regression

- Simple
- Limited to linear relationship

LGBM

- Light Gradient Boosting
- Fast

Approach

- Make sure to remove the ‘SK_ID_CURR’, since it acts as an ID and it’s presence could lead to data leakage

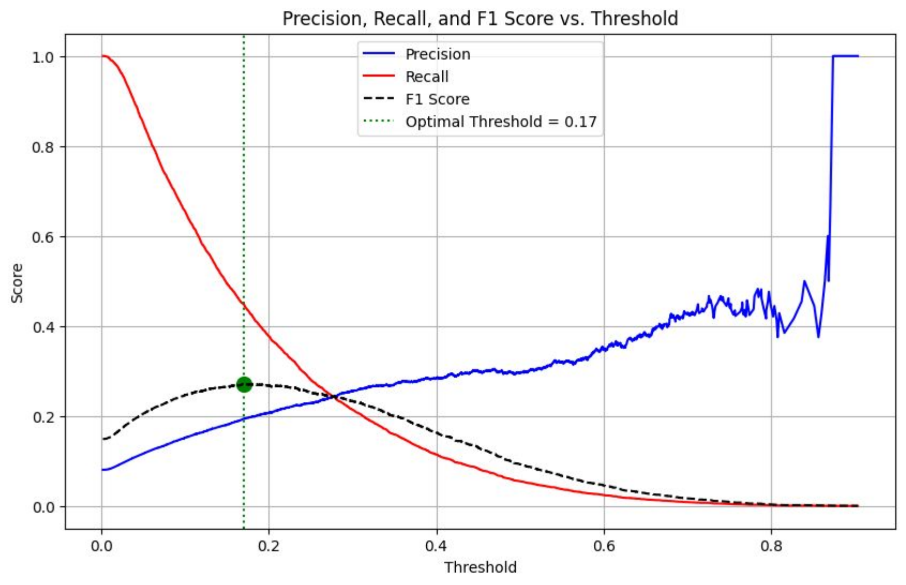
- The previous models where fitted 2 times:
  - Train data
  - Train data with PCA applied to it

- For Gradient Boosting Models
  - Fit once more after identifying the optimal threshold to divide the classes

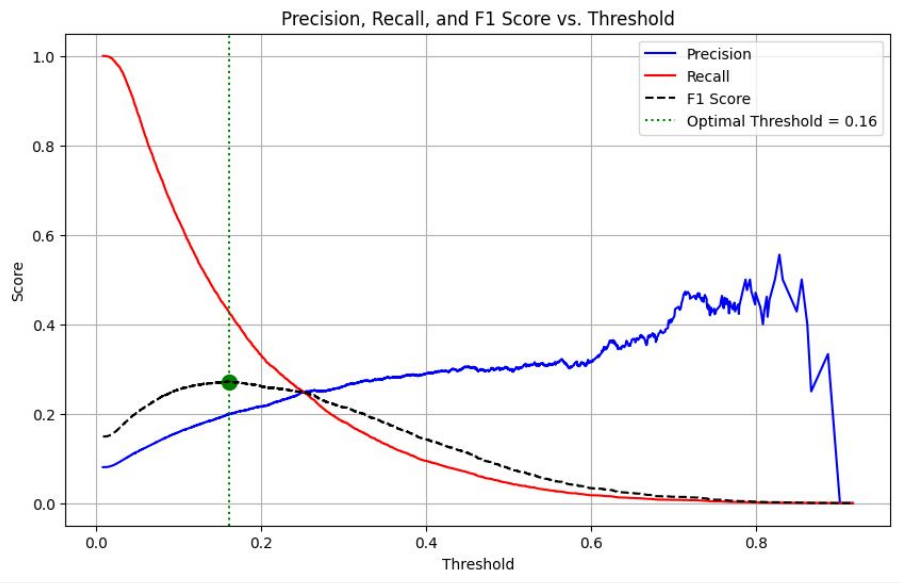
Finding Optimal Threshold

Optimal threshold is assumed to be the point that maximizes the F1-Score

XG- Boost

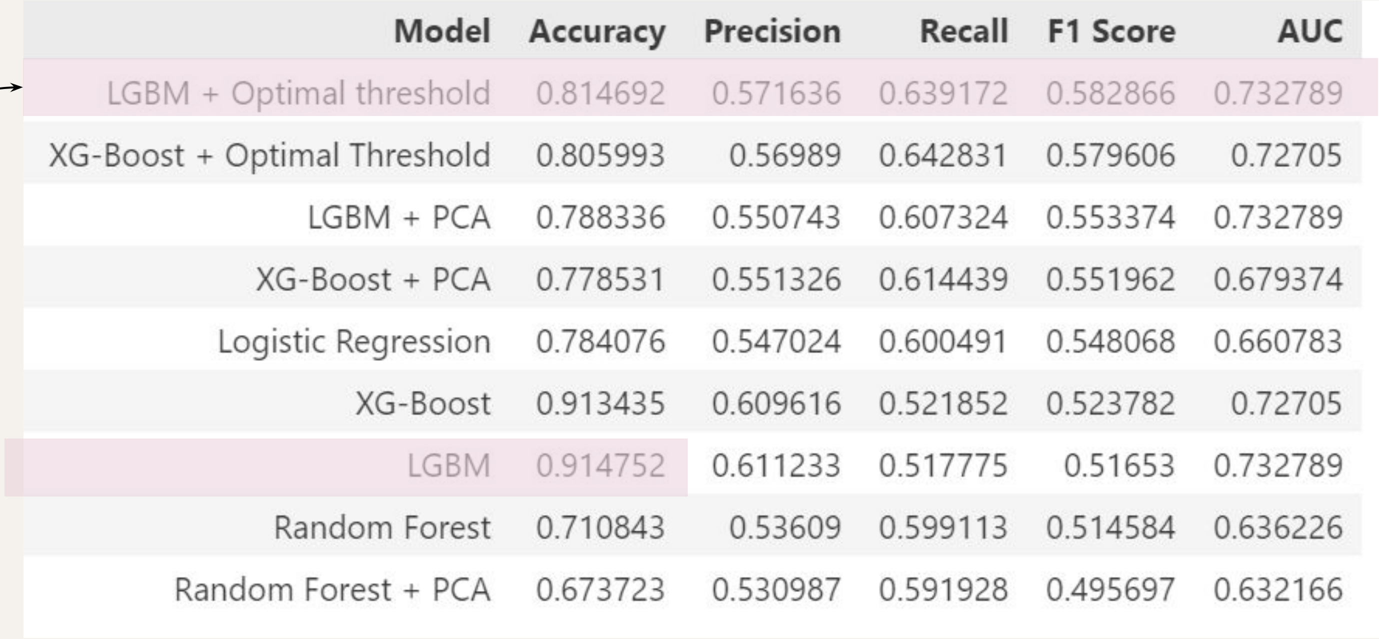


LGBM



Results

- Sorted by F1-Score since it accounts for both the effect of recall and precision



Model	Accuracy	Precision	Recall	F1 Score	AUC
LGBM + Optimal threshold	0.814692	0.571636	0.639172	0.582866	0.732789
XG-Boost + Optimal Threshold	0.805993	0.56989	0.642831	0.579606	0.72705
LGBM + PCA	0.788336	0.550743	0.607324	0.553374	0.732789
XG-Boost + PCA	0.778531	0.551326	0.614439	0.551962	0.679374
Logistic Regression	0.784076	0.547024	0.600491	0.548068	0.660783
XG-Boost	0.913435	0.609616	0.521852	0.523782	0.72705
LGBM	0.914752	0.611233	0.517775	0.51653	0.732789
Random Forest	0.710843	0.53609	0.599113	0.514584	0.636226
Random Forest + PCA	0.673723	0.530987	0.591928	0.495697	0.632166

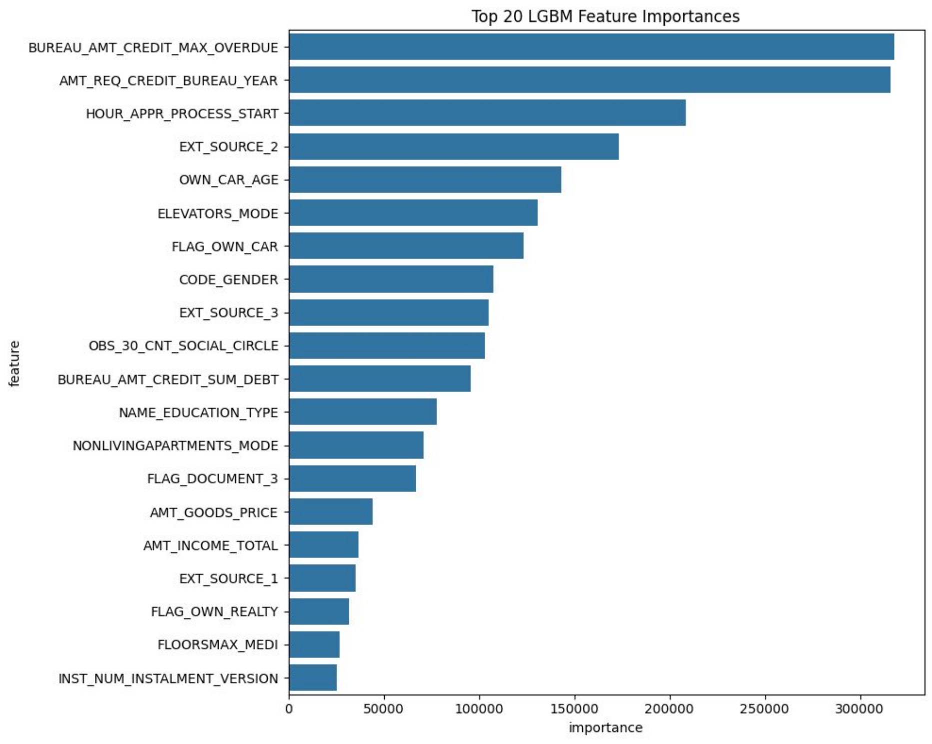
Highest performance by F1 score

Feature Importance

- Used the Best Model(LGBM + Optimal Threshold) to find the feature importance

- The graph displays only the top 20 features

- Turns out that the BUREAU_AMT_MAX_OVERDUE which reflects Maximal amount overdue on the Bureau credit so far (at application date of loan in our sample) has the highest importance

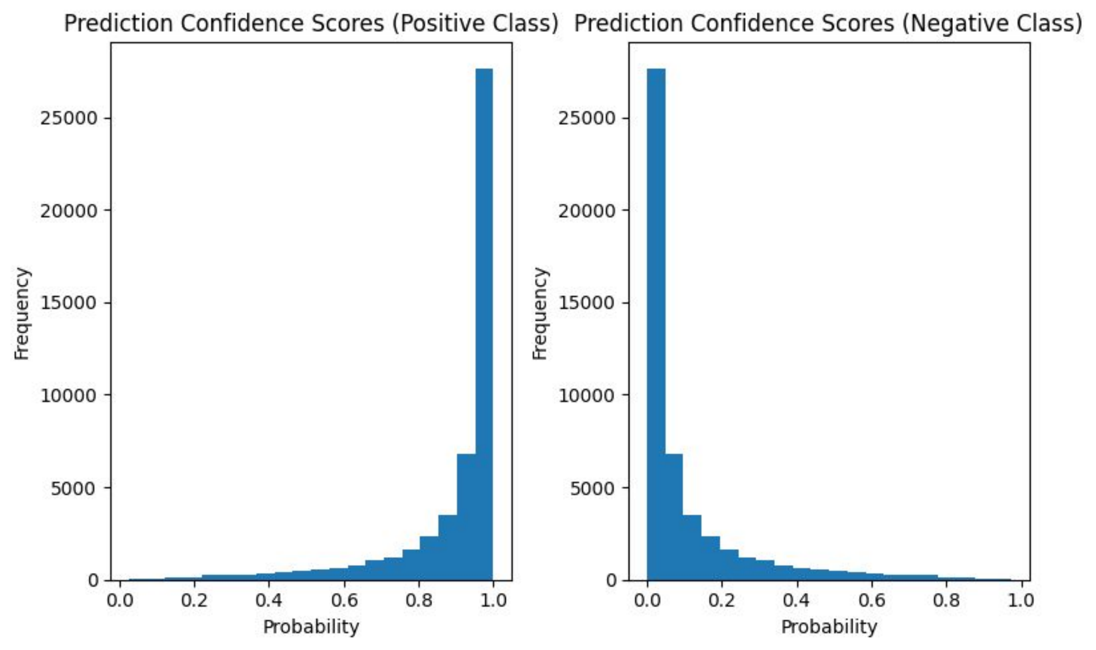


05
Test Data

Test Data

- Next Step was to use test data that hasn’t been observed yet by the model and test how the model would perform using best model (LGBM + Optimal Threshold)

- This graph shows that the model is highly confident in predicting both classes where it peaks near 0 or 1 in each of the cases.



06

Graphical User Interface

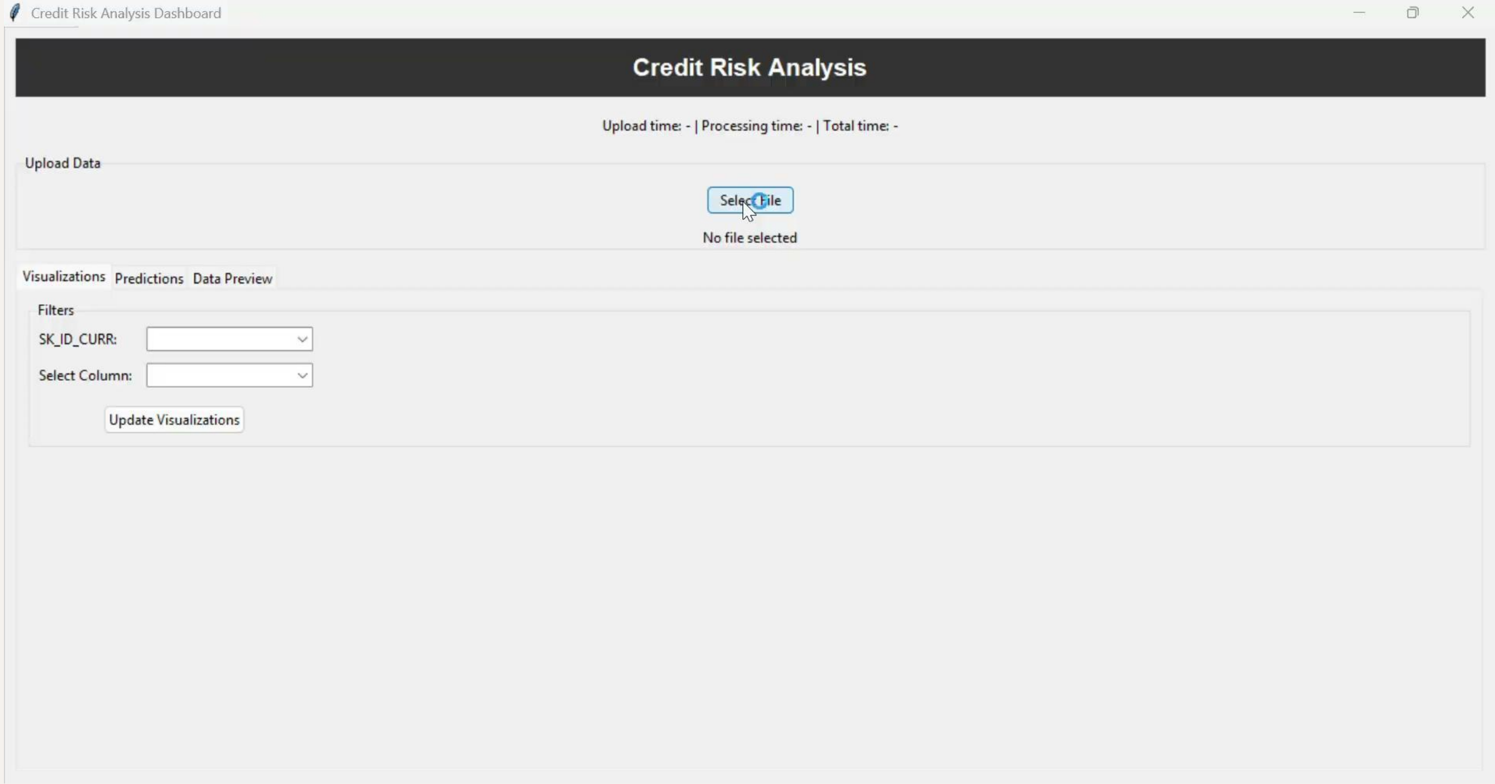
GUI

- The main reason for developing such a GUI is to automate the process and make it much easier to non-technicals, where the data is a click away. This would in turn decrease the prediction time; thus, reaching a decision much faster

- I composed a dashboard within a GUI using 2 different libraries ‘tkinter’ and ‘dash’; that’s mainly because tkinter is much faster, yet dash is much more organized and opens easily on a local host; however, it’s much slower in processing the data.

- Next slide shows a demonstration video on how the GUI works

GUI



Credit Risk Analysis Dashboard

Credit Risk Analysis

Upload time: - | Processing time: - | Total time: -

Upload Data

No file selected

Filters

SK_ID_CURR:

Select Column:

07

Future Work

Future Work

- Work on Hyper-Tuning the Best Model (LGBM + Optimal Threshold), Code is already there but wasn’t able to finish the execution completely due to time constraints

- Work on models that only have the features with the highest importance, might have higher performance metrics, and take advantage of the small sized dataset

- Adjust the GUI to add more functionalities

- Maybe try other classification models such as KNN, Decision trees, Naive Bayes … etc.

- Make the project scalable

08

Conclusion

THANK YOU!!

For more information: samaamr@aucegypt.edu