American Express Campus Competition

Team: Third Degree Burn

Course: MS4610 - Introduction to Data Analytics

## Aryan Pandey - NA19B030

## Drishti Darshan - AE19B102

## Rinkle Sebastian - NA19B056

## Pranav Srichinta - CH19B077



Jul - Nov 2022

# Contents

1. Introduction
   1. Problem Statement
2. Data
   1. Data Exploration
   2. Data Preparation
3. Model Building
   1. CatBoost Classifier
   2. Model Parameters
4. Additional Experiments
   1. Random Forest Classifier
   2. Random Forest Classifier - With Adversarial Validation
   3. XGBoost Classifier
   4. LightGBM Classifier
   5. Multi-Layer Perceptron
   6. Stacking Classifier
5. Results
   1. Feature Importance
   2. Conclusions

# Introduction

A bank can offer two types of cards:

1. Charge card: The balance is required to be paid in full each month
2. Lending card: Lending cards allow the customer to pay the balance over a period of time subject to interest being charged

An individual can apply for any one of the two types of card on offer. In order to extend the card to the individuals, banks must first underwrite the applicant. Underwriting is the process by which the lender decides whether an applicant is creditworthy and should receive a credit line. Along with the data present in application forms, banks also have access to the consumer bureau. Bureau is an agency that aggregates consumer borrowing and payment information for the purpose of assessing credit-worthiness of an individual and setting a limit on the cumulative credit that can be extended to an individual by lenders.

## 

## 1.1 Problem Statement

In this competition, the dataset has the customer application and bureau data with the default tagging i.e., if a customer has missed a cumulative of 3 payments across all open trades, his default indicator is 1 else 0. Data consists of independent variables at the time T0 and the actual performance of the individual (Default/ Non Default) after 12 months i.e., at time T12.

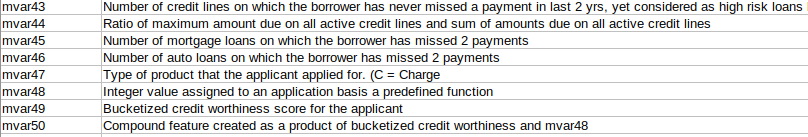
We need to predict if an applicant will default in the next 12 months from a new credit card application.

# Data

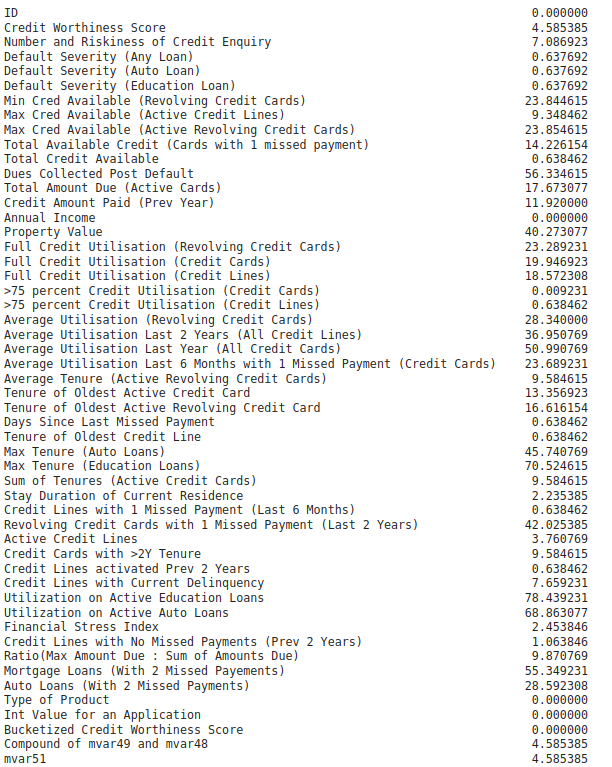
## 2.1 Data Exploration

The dataset has 51 variables along with an ID number for each applicant. The description of the features is given below:

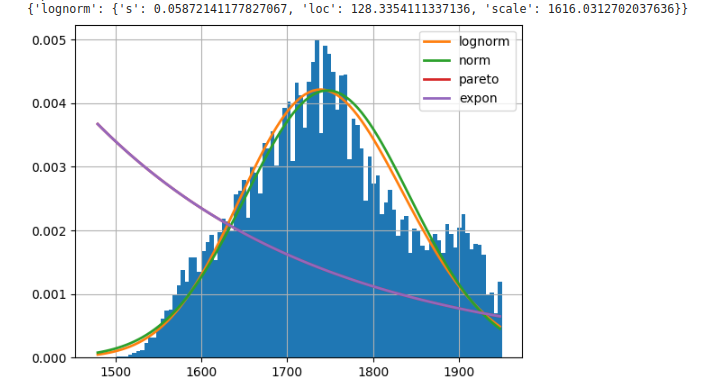




The dataset is filled with Null Values in a lot of columns, the picture given below shows the percentage of null values in each column when we combine the train and test set

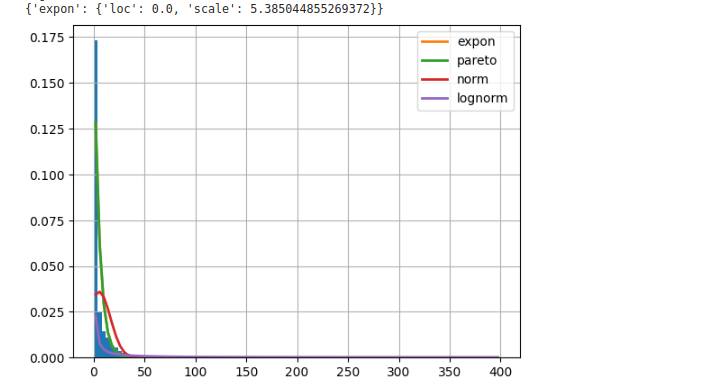


The Credit Worthiness Score Feature has a lognormal distribution as shown in the following image:

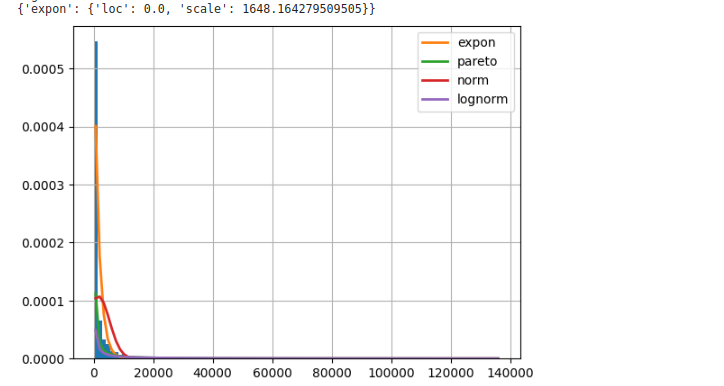


All of the other columns have an exponential distribution. This shows that there are a large number of outliers in the dataset in every feature

Following is the distribution of the Default Severity:



Following is the distribution for the minimum credit available:



## 2.2 Data Preparation

In order to get the data ready for the model, we imputed all the missing values with -1 when working with Tree-Based Models and with the median column value when dealing with models like Multi Layered Perceptrons. Apart from this for the Multi Layered Perceptrons, we also Log Scale the features and scale them all to be in the same range. The exact details of the model architecture and their performance will be described in the later sections. We also oversample the minority class using SMOTE when we use a LightGBM.

# Model Building

## 3.1 CatBoost Classifier

After a good bit of experimentation, A CatBoost Classifier was the final model that we finalized on. This model performed better than other tree based models like XGBoost in which the strategy to build trees was depthwise. CatBoost builds the trees breadthwise which allows the Tree to be symmetric while maintaining the depth of the tree. This is especially important in this kind of a dataset which has a large number of features as well as a large number of observations.

We use a 5 Fold Stratified Cross Validation to train this model and evaluate its performance based on the Accuracy and F1-Score of the model, with the F1-Score being given preference in the evaluation due to the unbalanced nature of the dataset. We also use Early Stopping to improve our validation strategy and avoid overfitting. We achieved an F1-Score of 0.59616 and an Accuracy of 71.38% on the Validation Split with this model which translated to 61.07% on the Leaderboard.

Given Below is the set of parameters that were used to train the model in the most optimal way along with a small description of what they do

## 3.2 Model Parameters

* N\_estimators : Value - 10,000

This Parameter sets the maximum number of trees that the model would build on the train set if it were allowed to do so without early stopping

* Early Stopping Rounds: Value - 1000

This Parameter sets the number of rounds of training that the model will undergo to see if there is any further decrease in the loss value or not. If there is no decrease in the minimum loss for these many rounds the training get terminated early

* Max\_Depth: Value - 8

This Parameter controls the maximum depth which the tree will get built till. This tree is built in a symmetric manner so the maximum it will go up to is a depth of 8

* Eval\_Metric: Value - “Cross Entropy”

If an early stopping is provided, this Parameter specifies the evaluation metric upon which the training will be terminated early

* Scale\_Pos\_Weight: Value - 59145/23855

This parameter is used in an unbalanced classification scenario such as this one and is the ratio of number of negative class (0 in our case) to the positive class (1 in our case)

* Bootstrap\_type: Value - “Bernoulli”

This parameter affects the Regularization and Speed while choosing a split for a tree when building the tree structure. With the Value of Bernoulli, the model follows a Stochastic Gradient Boosting Method. All the sampled examples have an equal weight and the sample for choosing the split is obtained with a probability defined by the subsample parameter which in our case is 1.

* Learning Rate: Value - 0.01

This parameter controls the learning rate of the model, too high a learning rate could lead to divergence and too small a learning rate could take a very long time for the model to reach convergence

* Random\_State : Value - 0

This parameter controls the Random Seed Value of the Model.

# Additional Experiments

## 4.1 Random Forest Classifier

* In order to have a benchmark for how effective our performance with other models is, we decided to train a Random Forest Classifier on the entire dataset with the base parameters.
* This model was able to achieve a performance of an F1 Score of 0.51 on the Validation Set which translated to 54.73% on the Leaderboard
* We changed the parameters a bit in order to make the fit a little better and were able to optimize its performance till an F1 Score of 0.56 which translated to 58.55% on the Leaderboard.

We decided to do some level of data exploration and observed that the base statistics of the train and test set were a little different. In order to see if there was truly any difference in their distributions we tried out an Adversarial Validation Model wherein we fit a model with the task of differentiating between the train and the test set.

This model achieved an AUC of 0.63 which meant that there truly was some disparity in the train and test set. We then plot the Feature Importance and removed those features from our train set which this model thought were most important. This is because these features are the ones which are the ones which are separating the train and test distributions

## 4.2 Random Forest Classifier - With Adversarial Validation

* We then retrained the Random Forest Classifier with the features we dropped through Adversarial Validation with the same parameters of our most optimal Random Forest with the Raw Dataset.
* This model saw a small spike in performance where the model gave an F1 Score of 0.57 which translated to 59.15% on the leaderboard

## 4.3 XGBoost Classifier

* This model also used early stopping to detect overfitting and was trained in a 5 Fold Stratified Split manner
* The model did not perform well at all possibly because this model builds its trees depthwise instead of a symmetric fashion and that did not work out well for this dataset.
* This model achieved an F1 Score of 0.51 which translated to 54.75% on the Leaderboard

## 4.4 LightGBM Classifier

* This model came the closest to our current best model which is the catboost classifier.
* We trained this model by mentioning the class imbalance through the scale\_pos\_weight parameter, the functionality of which has been described in Section 3.2.
* A possible explanation for the good performance of this model again is the fact that it builds the trees breadthwise instead of depthwise
* This model achieved an F1-Score of 0.59 which translated to 60.48% on the leaderboard

## 4.5 Multi Layer Perceptron

* This model did not perform well at all even after we made sure to scale all the feature values to the same scale and log transform all the features which had an exponential distribution in order to give them a Gaussian appearance.
* A possible explanation could be the lack of training data to train a Neural Network with some amount of reliability. 30,000 entries doesn’t quite cut it.
* This model achieved an F1-Score of 0.527 which translated to 56.3% on the Leaderboard.

## 4.6 Stacking Classifier

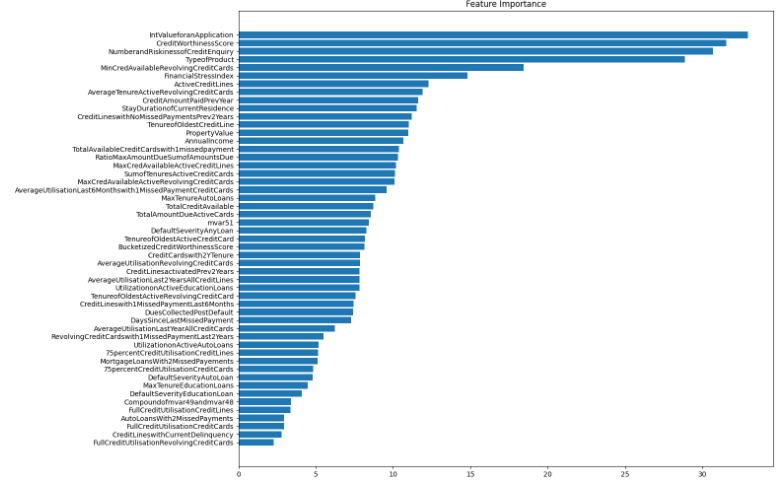
* This model used all the above mentioned models in addition to the best performing catboost model as the base estimators
* The final estimator was also a CatBoost Classifier which was trained both on the predictions of the base estimators as well as the original dataset
* Although it gave a validation F1 score of 0.68, there’s a high possibility of this being a biased estimate since some of the models in the 5 fold scenario would have been trained on some of the data points. Given some more time, we would like to try making a holdout set and training this classifier to obtain its true performance.
* This Classifier Achieved 60.47% on the Leaderboard.

# Results

| **Model Name** | **Leaderboard Score** |
| --- | --- |
| Random Forest Classifier | 58.55% |
| Random Forest Classifier- With Adversarial Validation | 59.15% |
| XGBoost Classifier | 54.75% |
| LightGBM Classifier | 60.48% |
| Multi Layer Perceptron | 56.3% |
| Stacking Classifier | 60.47 |
| **CatBoost Classifier** | **61.07%** |

## 5.1 Feature Importance

Given below is the Feature Importance observed from our best CatBoost model



As you can see, the most important features turn out to be “Integer Value for an Application”, “Credit Worthiness Score” and “Number and Riskiness of Credit Enquiry” which even logically should hold the most weight when deciding this kind of a prediction mechanism.

## 5.2 Conclusion

In this competition, we used Various models and Validation Methods to analyze the credit application data given to us. The best results were obtained using a CatBoost with the parameters as mentioned in Section 3.2