Introduction to Data Analytics
MS4610

To predict whether a loan will go default or not, given the dataset which consists of the details on loans taken by different customers previously (Supervised Learning)

Model to Predict Loan Default

MM17B003 Pranav Pawar
MM17B105 A Ramprasad
NA17B114 Nimeesh Pitta
MM17B026 R Arun Prakash
MM17B035 Utkarsh Attela
MM17B009 Alexander D
MM17B029 Ritik Bhilware

Model to Predict Loan Default

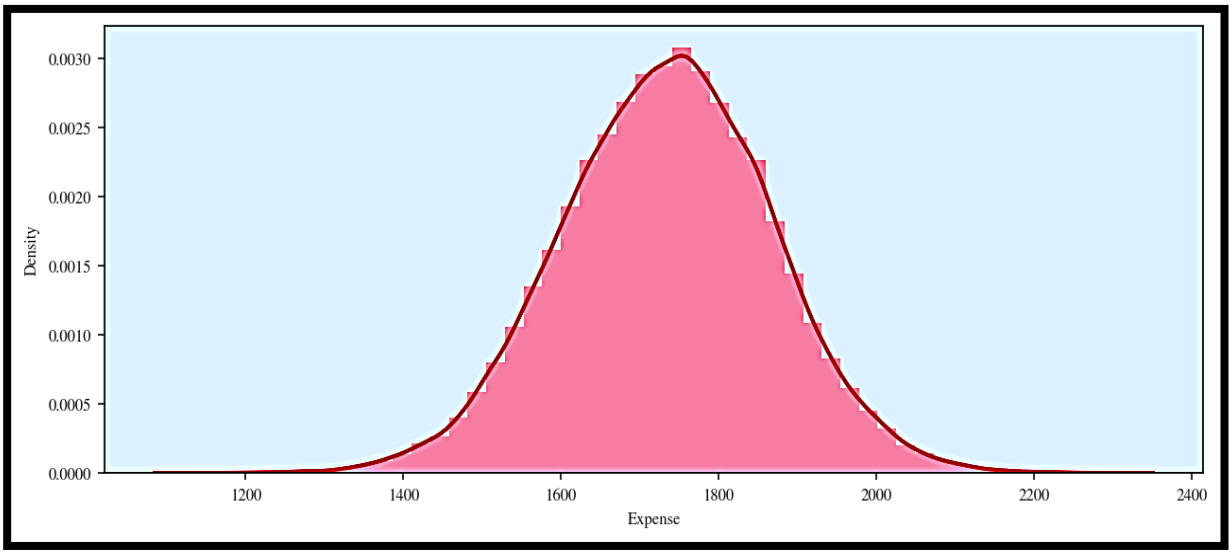
I. Data Preprocessing:

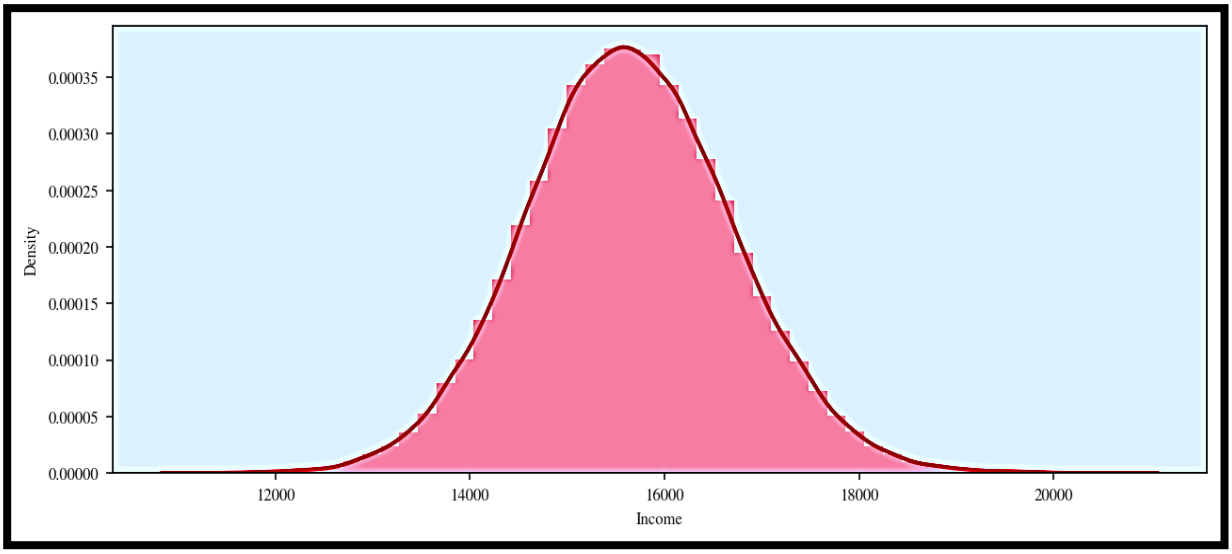
- ❖ Missing values:
  - Drop rows with no labels
  - Missing data points in numeric data imputed with -999 to represent them as outliers.
  - NaN values in categorical data are kept as it is as LightGBM and XGBM are capable of handling them.

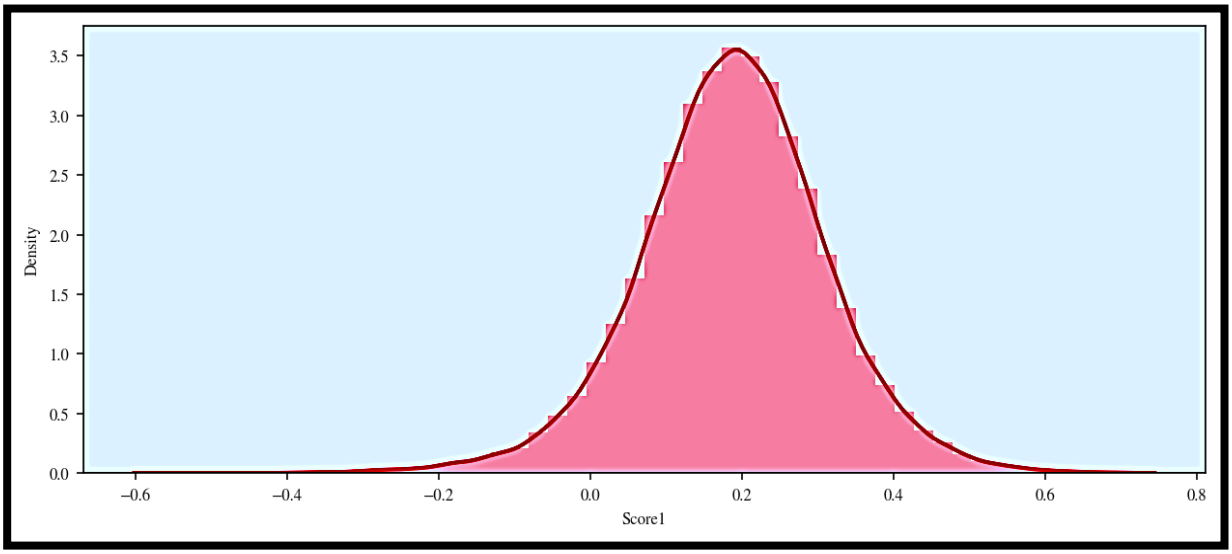
- ❖ Encoding categorical data:
  - The variables 'Occupation type' and 'Loan type' were in the form of categorical data.
  - The training and test datasets were concatenated and the variables were encoded using sklearn LabelEncoder.

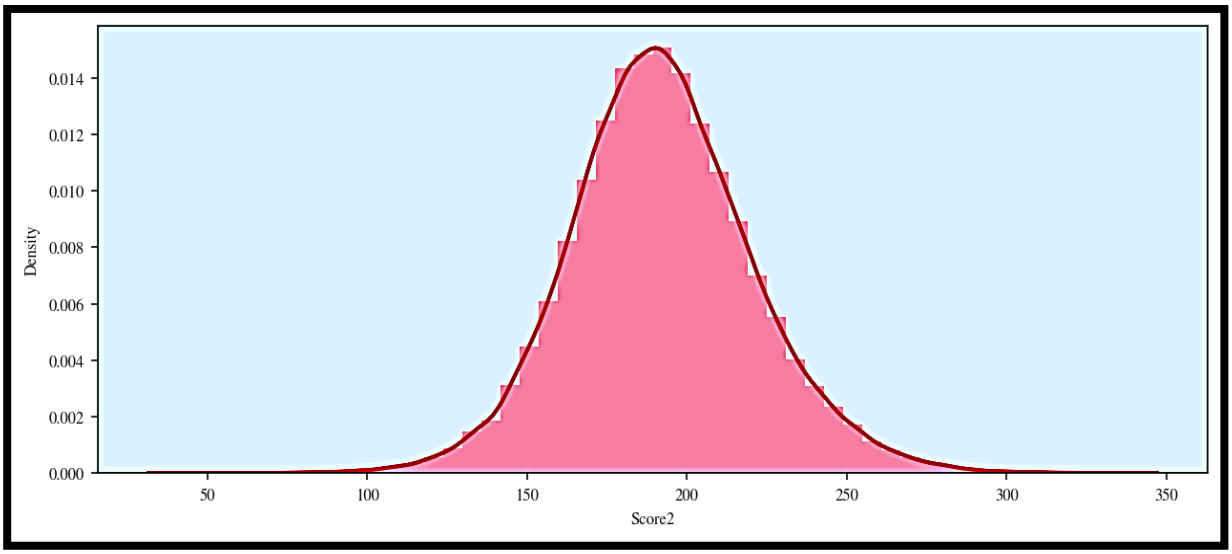
II. Data Visualization:

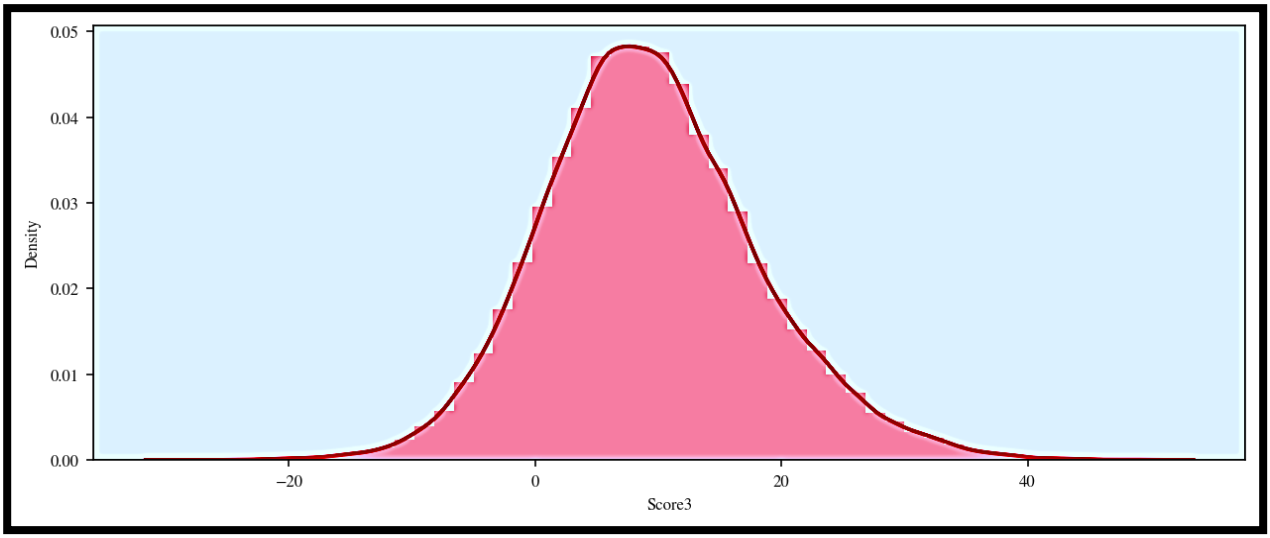
- ❖ Numeric Features:
  - Dist plot of each numeric feature is plotted to know the distribution of data

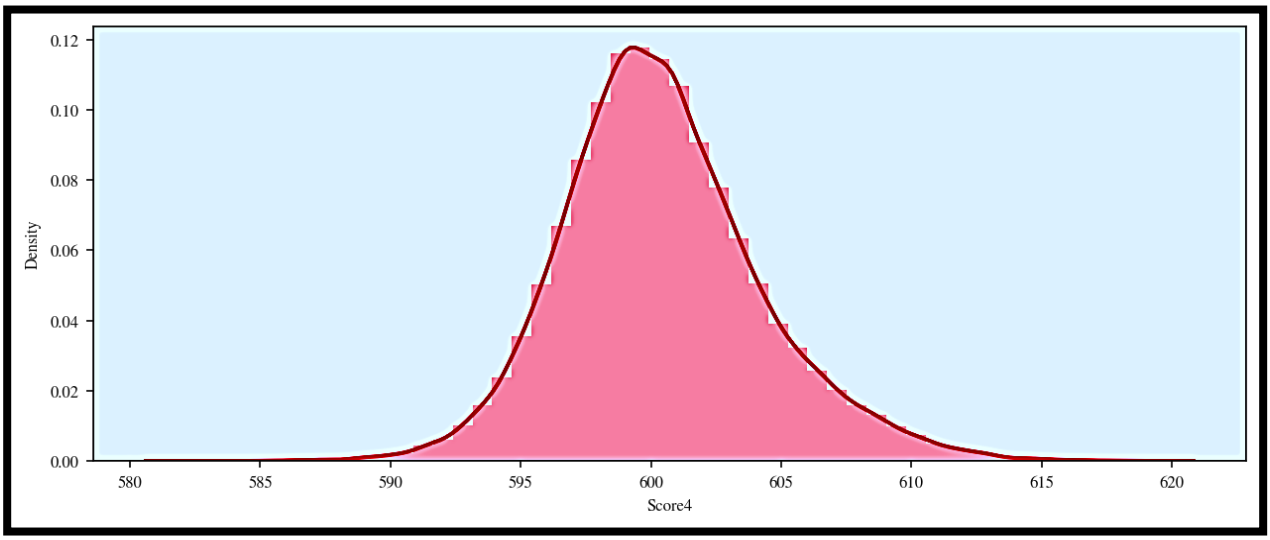


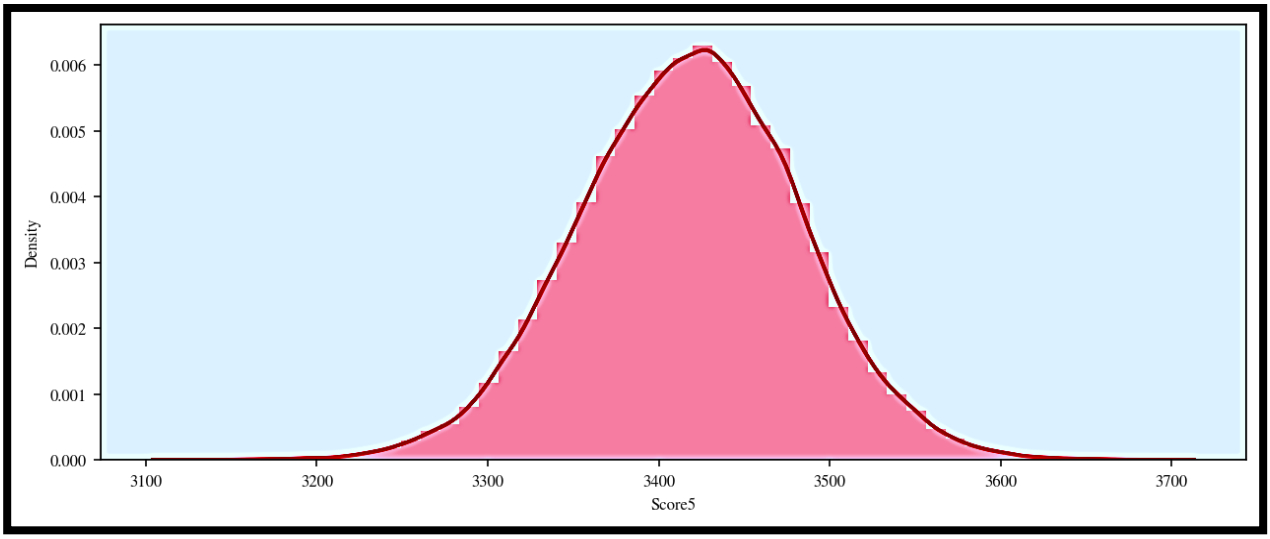

Density
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
12000
14000
16000
18000
20000
Income


Density
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
Score1


Density
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
50
100
150
200
250
300
350
Score2



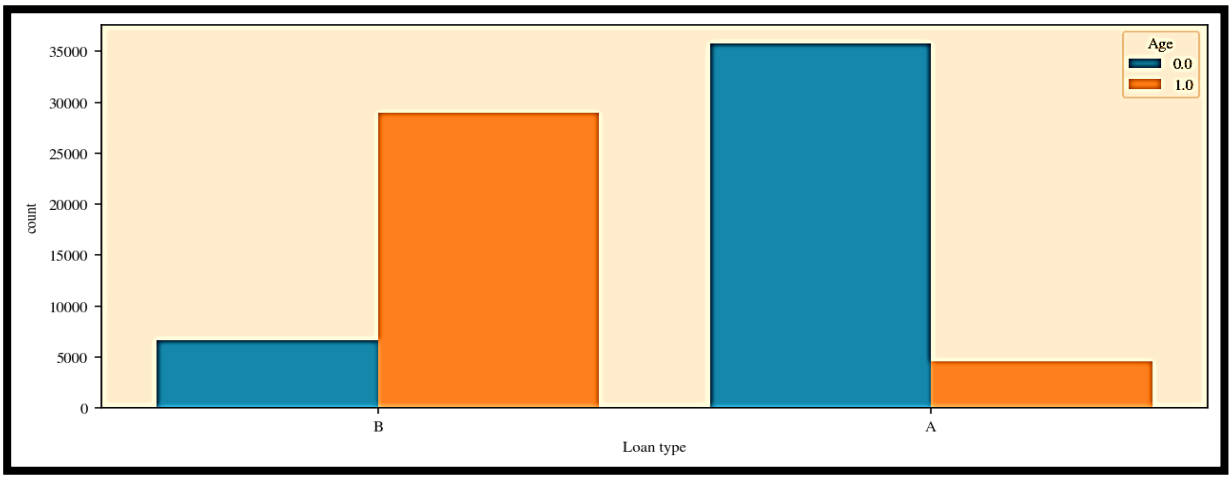


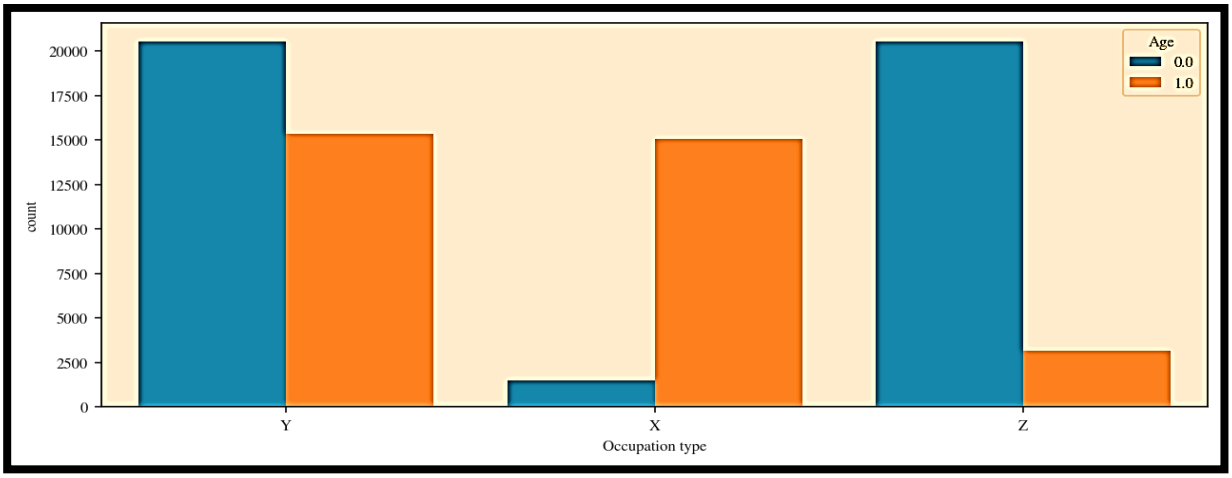


- All the numeric features are normally distributed, which is a good sign. No much processing of data is required.

❖ Categorical Features:

- Age group '0.0' are preferring **Loan A** more
- Whereas Age group '1.0' are preferring **Loan B** more





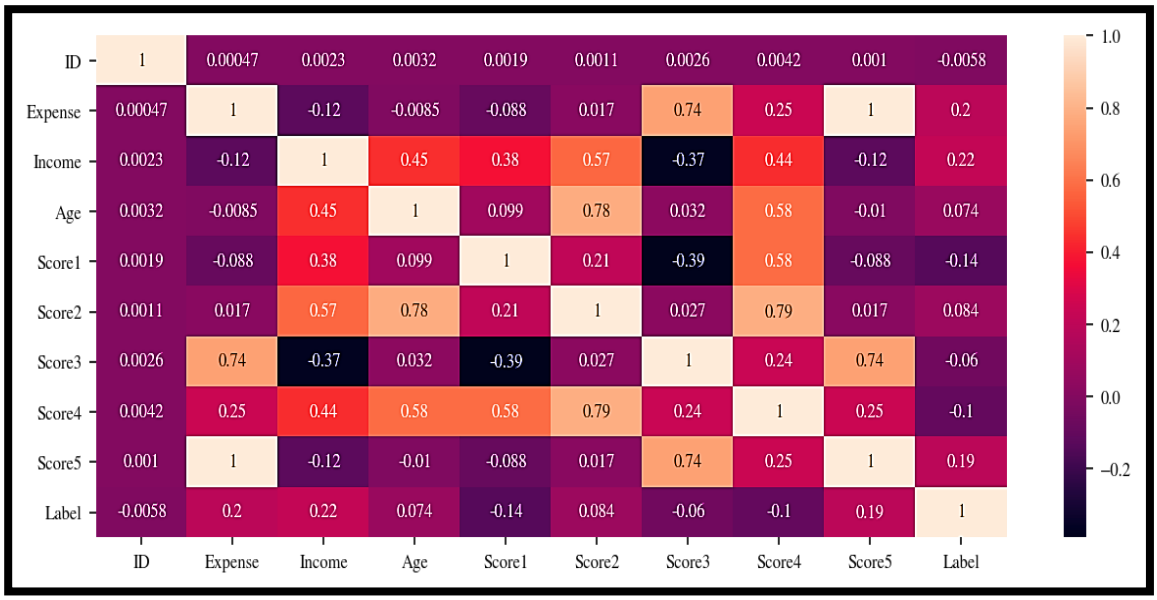
III. Feature Engineering:

Different Feature Engineering approaches are carried out, for example

- Creating domain-based features like Expense/Income ratio and possible algebraic combinations of the SCORE_ variables
- Grouping the numerical features wrt some Age group (or Age itself), then saving the numeric feature means for every such possible group, and then for each data point calculating the difference between its actual feature value and mean value of its group for given numeric feature. (Doing this for all possible numeric features)

These methods didn't turn out to be useful, hence we moved on to the next step:

- Plotting the correlation between features to identify the correlated features.



- From this, we can infer that **Score4** is correlated to almost all features. Hence we dropped Score4.
- Also, **Score5** is highly correlated to Expense (value 1.0), But dropping **Score5** or **Expense** didn't improve the model accuracy.

Conclusion: **Score4** is removed from feature matrix

IV. Model Building (XGBoost):

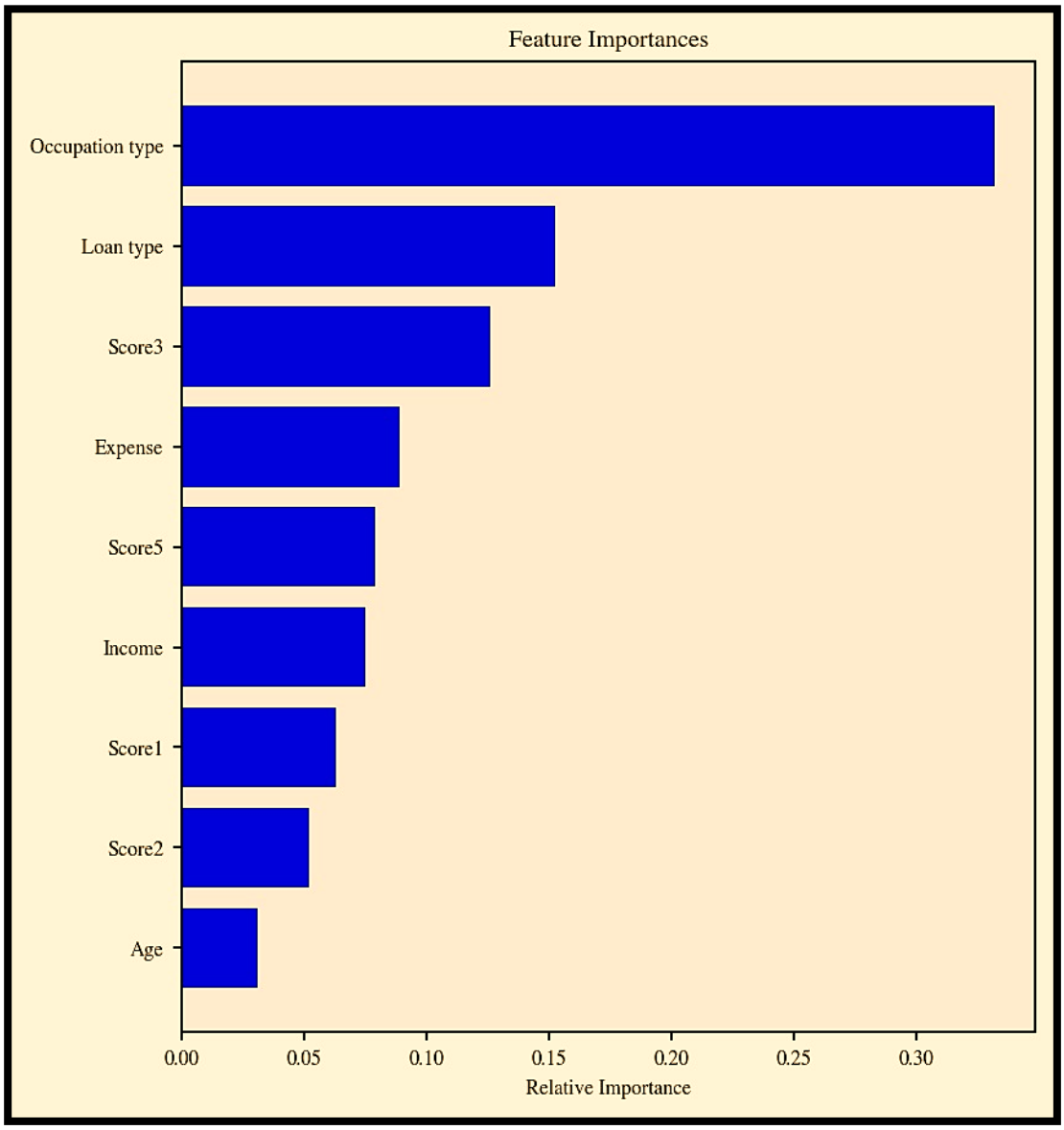
- Initially, we tried different classification Algorithms like Logistic Regression, KNN, SVM's, Random Forest Classifier etc, but the validation score was not great.
- Then we thought of boosting techniques using LightGBM, Xgboost; which eventually turned out to be the best performing model on validation set.

XGBoost: We tested this algorithm on the feature engineered data with random parameters and **K fold validated** to get an accuracy of 0.980 on validation data.

Parameter Tuning: Then we tuned the parameters using GridSearchCV to obtain valid parameters which later on applying to XGBoost gave an accuracy of 0.984.

V. Feature Importance:

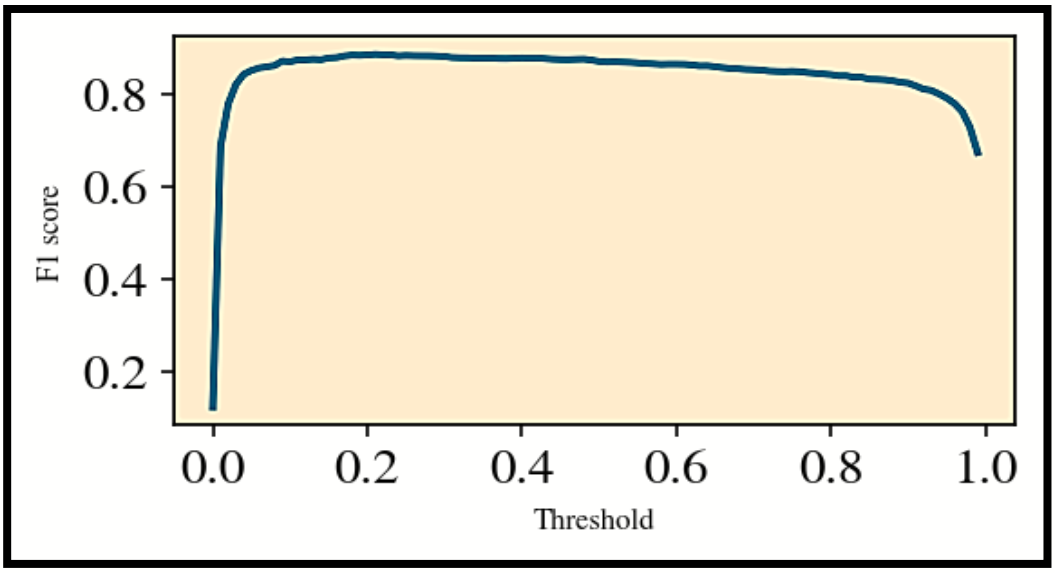
- The following plot shows the relative feature importance of different features in the dataset obtained from the learnt model using feature_importances_ method



- As can be seen from the plot, **Age** has the lowest feature importance
- **Occupation type** and **Loan type** have the highest importance as such they contribute the most in predicting the loan type of a given instance
- **Score3, Expense** and **Score5** are the other features in top 5 with highest feature importance

VI. Model Evaluation (F1-score):

- The key objective of solving this problem is to detect defaulting loans, hence true positives should be our focus
- As we know, this problem suffers from great class imbalance, so keeping 0.5 as a decision threshold isn't plausible here. We need to lower the threshold, in order to obtain more true positives.
- We can build a custom function which tunes the threshold values ranging from 0 to 1, and finds the optimal one, where we have maximum F1-Score
- But to make sure the quality of our probability predictions, we can use **roc_auc_score**, this allows us to compare multiple models wrt their probability prediction quality, once we are sure of the highest roc_auc_score, we can tune predictions of that model wrt multiple thresholds



- By doing so we obtained a threshold of **0.22,** which increased the F-1 Score and Accuracy of the model when compared to the default threshold 0.5
- By thresholding to 0.22, we obtained an accuracy of 0.985 on the validation set.

VII. Prediction:

This trained model is used for predicting instances of **test_x** file. After obtaining the prediction probabilities the threshold is set to 0.22 for predictions. That is a probability **>0.22 is 1** and probability **<0.22 is 0.** The predicted file is attached in the submission.