MS4610

Introduction to Data Analytics

Term Project

Loan Default Prediction

Saarthak Sandip Marathe - ME17B162

Burhanuddin Sabuwala - BE17B011

G Prashant - BS17B011

Raj Jain - CH17B066

Sriram Ragunathan - CH17B072

Shashank H S - BE18B006

Indian Institute of Technology Madras

Jul-Nov 2020

January 6, 2021

Contents

List of Figures

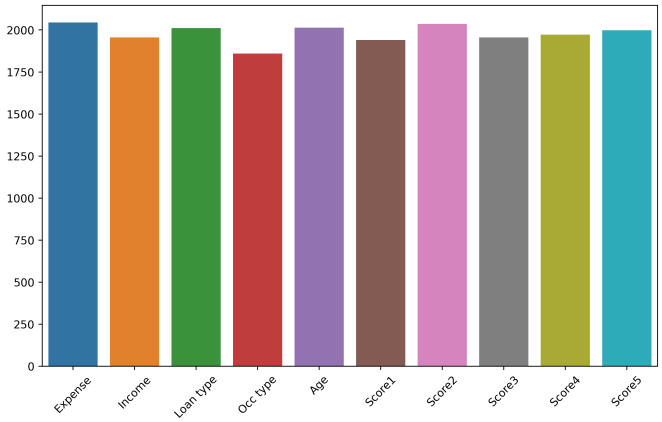
1 Introduction

We are given a dataset of various attributes of loans taken by people in the past, alongside the information whether the loan was defaulted or not. We wish to predict whether a loan will be default in the future given the same attributes. The problem at hand is of binary classification. We are given the following attributes/features for each loan:

ID	A unique identifier for every financial loan that is being considered
Loan Type	Type of loan taken (Two types, 'A' or 'B')
Occupation Type	Occupation of the customer (Three occupation types, 'X', 'Y', 'Z')
Income	A continuous variable that is indicative of the annual income of the customer
Expense	A continuous variable that is indicative of the annual expense of the customer
Age	Age of customer – Value of '0' is considered as below 50, and value of '1' is considered as above 50
Score 1	Represents five different metrics calculated by the organization, about the customer and the loan that is being considered.
Score 2	
Score 3	
Score 4	
Score 5	

2 Imputation and Encoding

As seen above, the dataset consists of numerical, categorical and ordinal features. A significant part of the dataset consisted of missing values in all the features. The bar plot below shows the number of missing values (y axis) in each feature across all data points.



2.1 Imputation of Categorical and Ordinal Features

'Loan Type' and 'Occupation Type' are categorical features, while 'Age' is an ordinal feature. We performed imputation of missing values in these three features by substituting with the most frequent entry corresponding to which class the data point belongs. The table below shows the class-wise most frequent entry for each of the three features.

Firstly, we used two encoding methods to convert the categorical features to numerical data, namely, One Hot Encoding and Label Encoding. After trying both the methods, we finally chose One Hot Encoding as it gave better predictions. One reason for this could be because Label Encoding tends to denote an unsaid hierarchy while tree-based models are being implemented, whereas, One Hot Encoding doesn't do the same.

Feature / Class	**y=0**	**y=1**
Age	0	1
Loan Type	A	B
Occupation Type	Y	Y

2.2 Imputation of Numerical Features

Iterative imputation was performed to impute the missing values in numerical features such as Expense, Income, Score1, Score2, Score3, Score4 and Score5, with maximum ten iterations and appropriate early stopping conditions. In iterative imputation, each feature is iteratively modelled as a function of the other features using regression. Then, the regressor is used to predict the missing values of that feature.

3 Exploratory Data Analysis

3.1 Label Counts

Class	y=0	y=1	Missing
Count	71064	5033	3903

Here, we see that there is a heavy class imbalance in the available dataset, as the data points corresponding to y=1 (defaults), are much lesser in comparison to the ones with y=0 (non-defaults). Intuitively, it makes sense in the real world to have such data as the number of defaults should be much lower than the non-default loans for the bank to keep operating profitably.

3.2 Describing the data

First, we look at some central tendencies, minimum and maximum bounds and standard deviation of the continuous features:

	Expense	**Income**	**Age**	**Score1**	**Score2**	**Score3**	**Score4**	**Score5**
mean	1733.99	15641.11	0.44	0.19	192.07	9.37	600.40	3417.74
std	133.24	1065.62	0.50	0.12	28.56	8.76	3.83	64.39
min	1126.81	11171.70	0.00	-0.56	40.57	-28.89	581.81	3124.41
25%	1644.26	14925.66	0.00	0.11	173.42	3.51	597.89	3374.41
50%	1736.28	15624.26	0.00	0.19	191.06	8.88	600.10	3418.79
75%	1824.38	16346.08	1.00	0.27	209.73	14.75	602.60	3461.38
max	2309.13	20728.92	1.00	0.71	338.07	50.69	619.62	3692.73

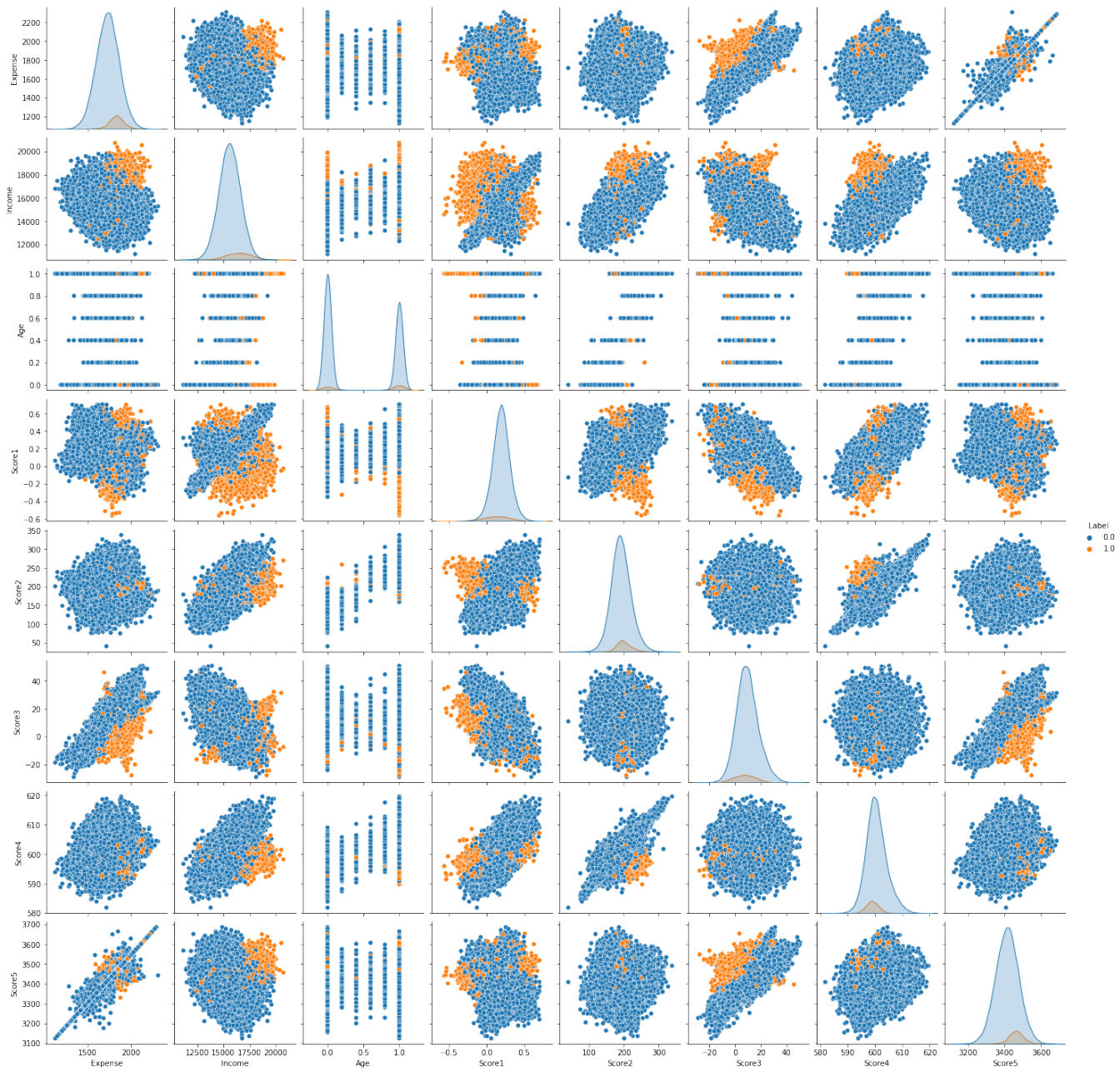
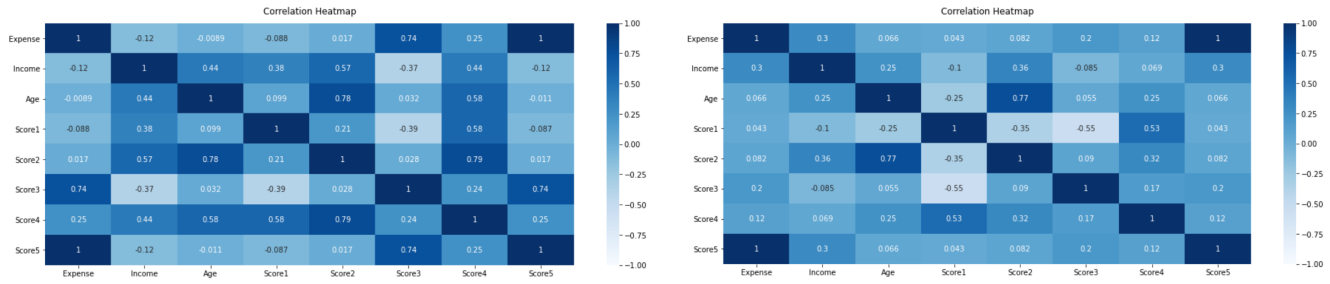


Figure 1: Pairplot of numeric variables

From this we see that for simultaneously high values of both expense and income indicators, we see a lot of loans with y=1. This tempted us to add a feature corresponding to the product of Expense and Income indicator variables.

Now we look at the correlation between the numerical features:



(a) Training data

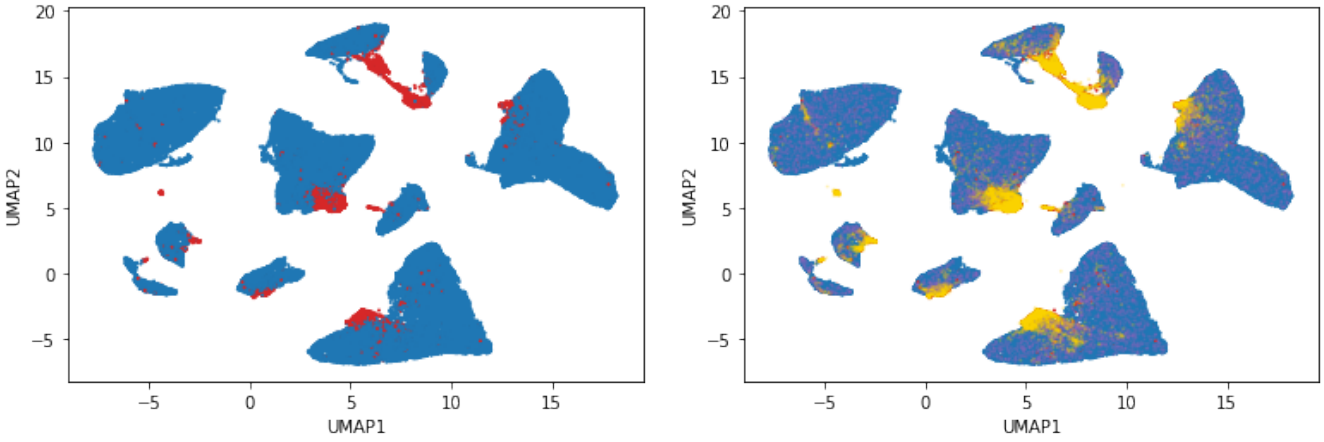
(b) Test data

Figure 2: Correlation between the numerical features in the training and test data

Here, we observe that the features score5 and Expense have very high correlation. We also see high correlation in the training data between the following pairs:

- Score2-Age
- Score5-Score3
- Score4-Score2

However, the same is not observed in the testing data, hence we do not use feature reduction for these. We also see some loose correlation between Income and Age, and intuitively it makes sense for people with higher age to have higher income on average.



(a) UMAP of Training data. Label 1 is shown in red and label 0 is shown in blue

(b) UMAP of Test data superimposed on training data. Predicted label 1 is shown in yellow and predicted label 0 is shown in purple

Figure 3: Correlation between the numerical features in the training and test data

4 Models Tried

Since most of the attributes of the dataset are decision-based features (for, e.g., an entry with the high expense is more likely to be a loan defaulter), and given the prevalence of class imbalance, we decided to prefer tree-based ensemble methods as well as boosting algorithms. The following

techniques were tried in order to choose the best machine learning algorithm suitable for this problem:

- K-Nearest Neighbors
- Logistic Regression
- Random Forests
- AdaBoost
- Gradient Boosting Classifier
- XGBoost
- Light Gradient Boosting Machine (LGBM)

4.1 Model Selection

The table below gives the accuracy, F1 score and ROC AUC of each of the classifiers that were implemented.

Models	Accuracy	F1 Score	ROC AUC
K-Nearest Neighbors	93.38%	0.65	0.5
Logistic Regression with L1-regularization	96.89%	0.7056	0.912
Random Forests	98.37%	0.8603	0.9713
AdaBoost (with RF base estimator)	98.47%	0.8682	0.9747
Gradient Boosting Classifier	98.35%	0.8614	0.9701
XGBoost	98.54%	0.8776	0.9728
Light Gradient Boosting Machine (LGBM)	98.62%	0.8838	0.9758

From the above preliminary analyses, it can be clearly noticed that the Light Gradient Boosting Machine (LightGBM or LGBM) classifier performs the best in terms of accuracy, F1 Score and ROC AUC. Hence, this algorithm was chosen for further analysis.

4.2 Model Training using LGBM Classifier

LightGBM (or LGBM) is a gradient boosting framework that uses tree-based learning algorithms. The advantages of using this are that it has higher training speed, takes care of the complex relations between variables (even non-linear) and tends to have higher efficiency than other models tried before.

Principal Component Analysis (PCA) was performed and it was noted that models trained with 10 Principal Components gave a better performance on the validation set. After hyperparameter tuning done with 10-fold cross-validation, the following model was selected:

Hyperparameter	Value
n_estimators: Number of boosted trees	3300
max_depth: Maximum tree depth for base learners	9
learning_rate: Boosting Learning Rate	0.12
subsample: Subsample ratio of the training data	0.7
colsample_bytree: Subsample ratio of columns when constructing each tree	1.0
Probability threshold: Threshold probability for class 1	0.20

The objective function was binary crossentropy. The summary of the results obtained through 10-fold cross-validation is given below:

Metric	Average Value (with 95.5% interval)
Training Accuracy	$(100.0 \pm 0.0)\%$
Validation Accuracy	$(98.642 \pm 0.00085)\%$
F1 Score	0.89264 ± 0.00738
Precision	0.93409 ± 0.0061
Recall	0.85508 ± 0.01394
ROC AUC	0.9780 ± 0.0020

It can therefore be noticed with appropriate dimensionality reduction and hyperparameter tuning, the performance increases in terms of all the given metrics. The comparatively lower value of recall suggests the prevalence of false positives.

4.3 Evaluation on Test Dataset

After selecting the model with the best set of hyperparameters, we again trained on all the data after performing PCA with ten components. The test data was preprocessed in the same way as the training data, where categorical features were One-Hot Encoded, and using the PCA model; it was transformed into ten dimensions. Following this, using a threshold cutoff of 0.2 for Class 1, the labels of the test data were predicted. 7134 test data points were predicted a class of 0 (i.e., not a defaulter), which accounts to 35.67% of the data, while 12866 test data points were predicted as Class 1 (i.e., defaulter), which accounts to 64.33% of data. The proportion of predicted positives in the test dataset is much greater than the number of positive labels in the training dataset. This indicates that the test data is not a random sample of the entire dataset.

5 Feature Importance

5.1 Using Logistic Regression Classifier with L1 norm

We tried Logistic Regression with L1 norm on the given dataset in order to understand the importance of individual features. It is seen that Age, Occupation type and Loan type seem to have very low predictive value. To further confirm the claim, we tried running Logistic Regression only using the Age, Occupation and Loan type as features. We found that the classifier would be highly biased, and it would classify all the elements into 0 class. Additionally, a decision tree fitted on

the same set of features would give similar results on the validation set. This observation is also consistent with the LightGB Model, as shown in Figure 5.

	coeff	std err	z	P>|z|	[0.025	0.975]
const	-2.7846	1.66e+06	-1.68e-06	1.000	-3.25e+06	3.25e+06
Expense**	1.6013	0.131	12.238	0.000	1.345	1.858
Age	-0.1669	0.061	-2.747	0.006	-0.286	-0.048
Loan type A	-0.6074	0.497	-1.221	0.222	-1.582	0.368
Loan type B	-0.3207	0.498	-0.644	0.519	-1.297	0.655
Occupation type X	-1.1396	1.66e+06	-6.87e-07	1.000	-3.25e+06	3.25e+06
Occupation type Y	-0.8414	1.66e+06	-5.07e-07	1.000	-3.25e+06	3.25e+06
Occupation type Z	-0.8035	1.66e+06	-4.84e-07	1.000	-3.25e+06	3.25e+06
Income**	0.8742	0.029	30.562	0.000	0.818	0.930
Score1**	-0.7543	0.070	-10.743	0.000	-0.892	-0.617
Score2**	0.9824	0.076	12.919	0.000	0.833	1.131
Score3**	-1.9376	0.081	-23.946	0.000	-2.096	-1.779
Score4**	-1.2010	0.108	-11.160	0.000	-1.412	-0.990
Score5**	1.0299	0.129	7.962	0.000	0.776	1.283

** - p-value $>$ 0.005

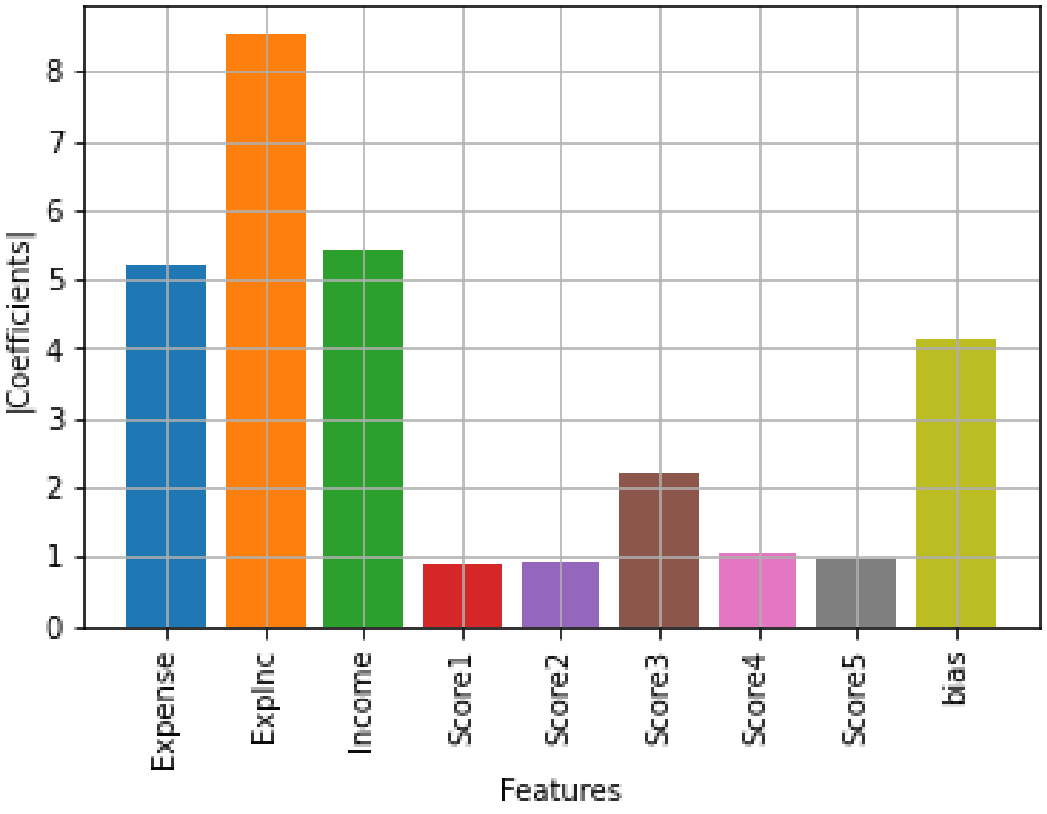


Figure 4: Feature coefficients given by Logistic model after excluding Age, Occupation type and Loan type. (ExpInc = Expense · Income)

5.2 Using LGBM Classifier Model

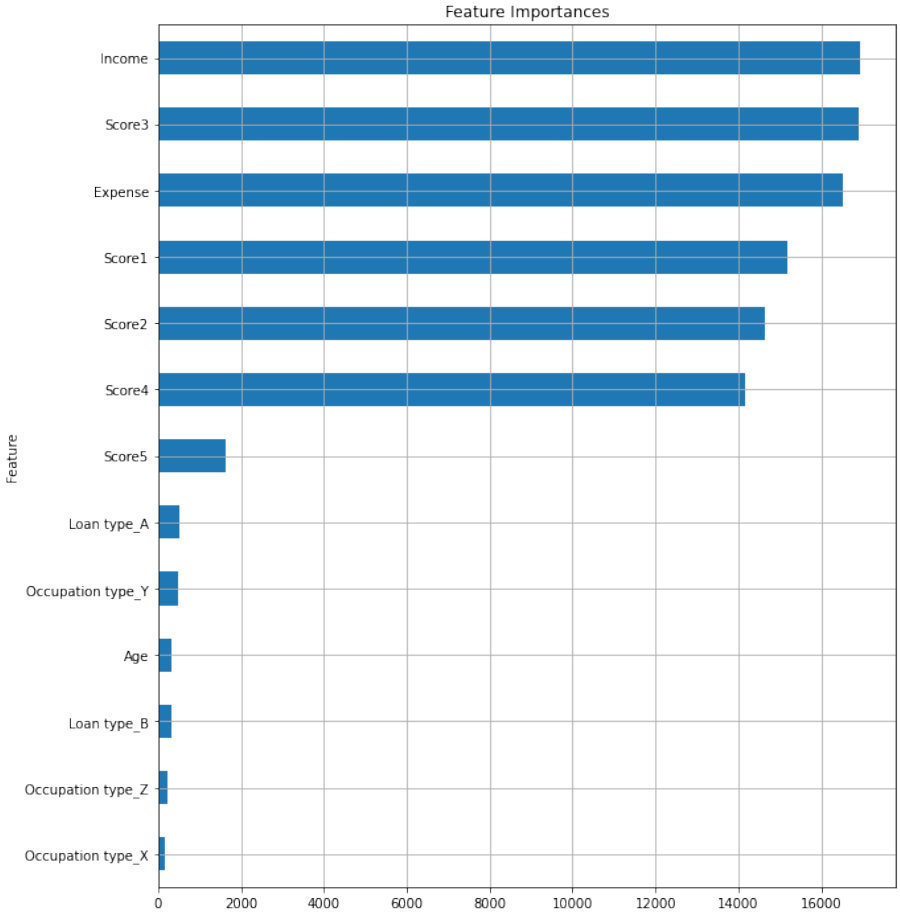
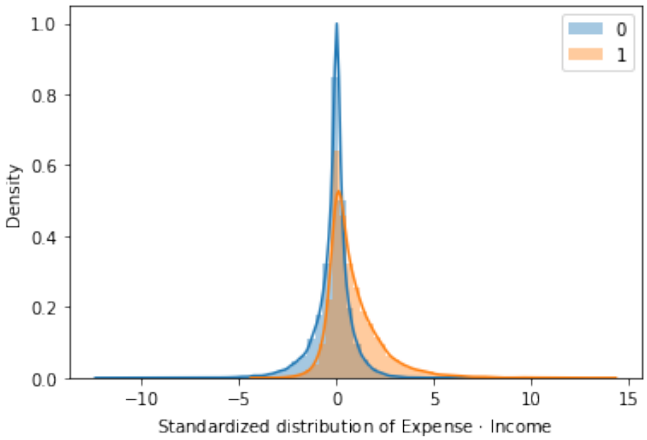


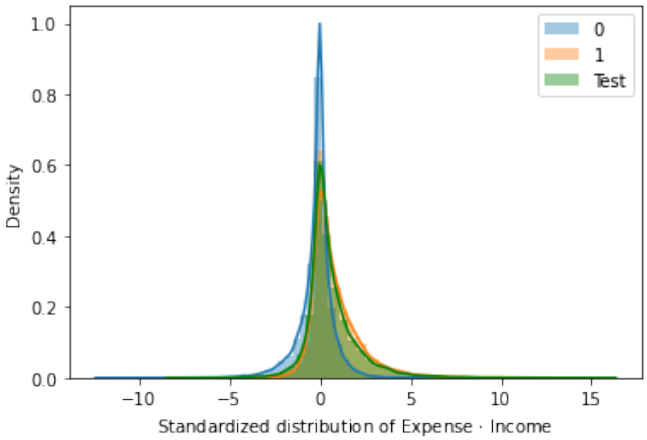
Figure 5: Feature Importance given by LGBM model

6 Insights

- The product of Expense and Income turns out to be an informative feature. It is seen from fig.1 that the data points with label 1 shifts slightly towards the right, i.e. the income and expense of defaulters are higher on an average. To get a better understanding of this, we used the product of these features. The distribution is shown in the histogram figure of Expense · Income drawn below.
- We created two models for using LGBM. One which included the non-linear relations related features (model1) and one with only the given input variables (model2). The said model1 had the features related to squared of the score, exp ·inc and the input variables. This increased the complexity of the model and tipped it to become over-fitting and have higher variance. Whereas model2 was simpler and gave a better fit. This was verified using 10-fold cross-validation.



(a) Standardized distribution of Expense · Income in the training dataset



(b) Standardized distribution of Expense · Income after including the test dataset

7 Conclusion

- We get better results for tree-based than other models and of these tree-based, the best one is LGBM model.
- For tree-based models, One Hot Encoding works better than Label Encoding as Label Encoding tends to define a certain hierarchy to the features which are not ideal for the current problem.
- Iterative imputations for the NaN values work the best among all the methods tried.
- LGBM model takes care of the non-linear relations between the significant features.
- Logistic regression model works best with non-linear relations established between significant features as suggested from the correlations and putting these as separate features. These non-linear relations are dropped while using LGBM to avoid over-fitting.
- Bias-variance dichotomy was verified, and the identified over-fitting LGBM model was overruled. Another reason to overrule this model was that it showed higher false negatives than the simpler model.

8 Code Availability

The codes for rest of the models is available at:
https://github.com/burhan1118/IDA-Course-Project---Team-9.

—— **Fin.** ——