Home Credit Default Risk Prediction

Group 10 - Kishore Tumarada

05/04/2021

Table of Contents

1 Introduction

Aim of our project is to assess the capability of each applicant in repaying a loan and predict Home credit default risk. The project is based on Kaggle competition hosted by Home credit organization, which is a service dedicated to providing loans to the unbanked population. Originally the data comprises of 7 datasets as shown in figure 1. In this project, we have used 3 datasets – Application data for each loan, Credit bureau report for each client, and Monthly balances of previous credits in Credit Bureau.

Essentially it is a binary classification problem with an objective to predict whether a client will default or not in future.

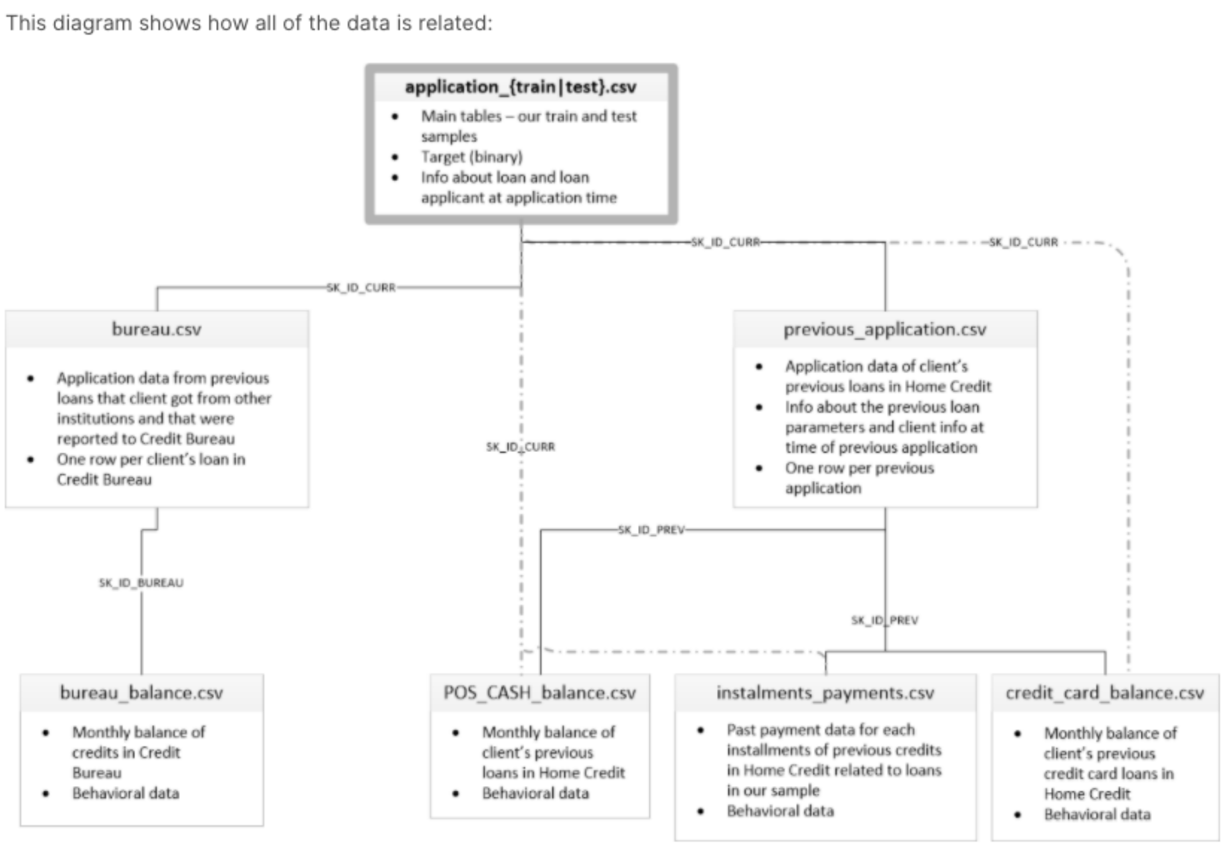

This diagram shows how all of the data is related:
application_{train|test}.csv
Main tables – our train and test samples
Target (binary)
Info about loan and loan applicant at application time
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
bureau.csv
Application data from previous loans that client got from 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
SK_ID_BUREAU
SK_ID_PREV
SK_ID_PREV
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

Figure 1 Flow chart showing relationship between different tables with Base table Application_train/test dataset

2 Exploratory data analysis

2.1 Application dataset

We will first explore Application dataset, which has 121 features with Target column, which is class label. Figure 2 shows the distribution of Target column, 1 refers to client with default risk and 0 refers to no risk client.

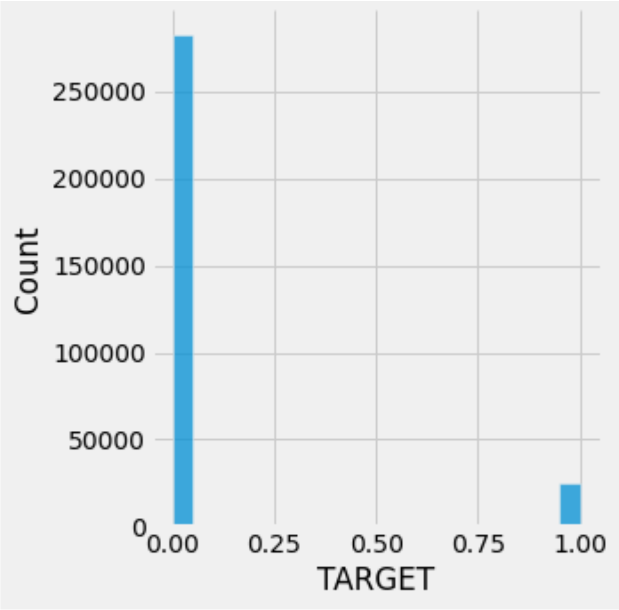
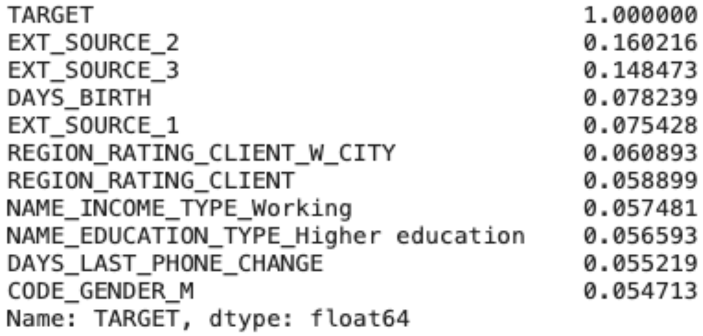


Figure 2 Histogram of Target column

Table 1 Top 10 highly correlated features with TARGET column



```
TARGET                                      1.000000
EXT_SOURCE_2                                0.160216
EXT_SOURCE_3                                0.148473
DAYS_BIRTH                                  0.078239
EXT_SOURCE_1                                0.075428
REGION_RATING_CLIENT_W_CITY                 0.060893
REGION_RATING_CLIENT                        0.058899
NAME_INCOME_TYPE_Working                    0.057481
NAME_EDUCATION_TYPE_Higher education        0.056593
DAYS_LAST_PHONE_CHANGE                      0.055219
CODE_GENDER_M                               0.054713
Name: TARGET, dtype: float64
```

Table 1 shows the top 10 highly correlated columns with TARGET column. We can see that EXT_SOURCE refers to external credit rating of the customers. Figure 3 shows the kernel density(histogram) of top 4 columns for two target classes. EXT_SOURCE_3 shows the greatest difference between the target classes. Similarly, younger age clients have higher rate of default compared to older age clients.

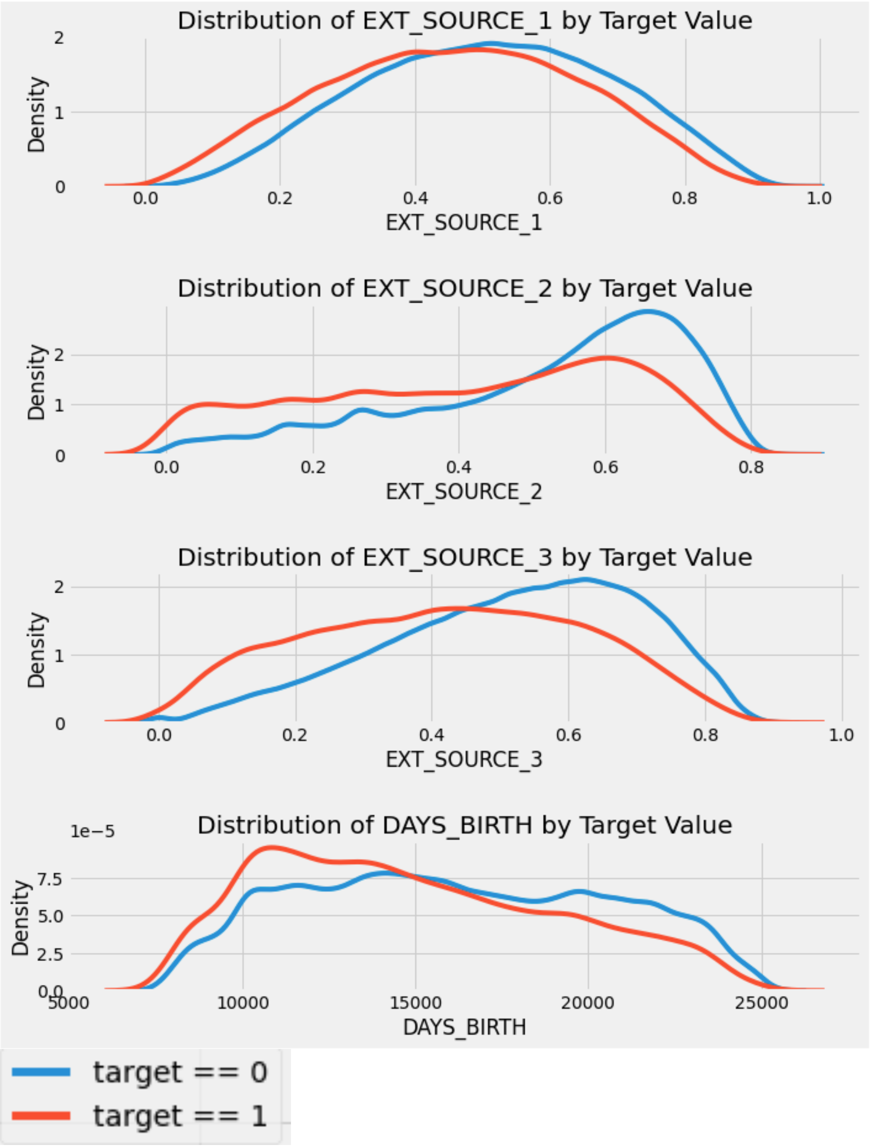
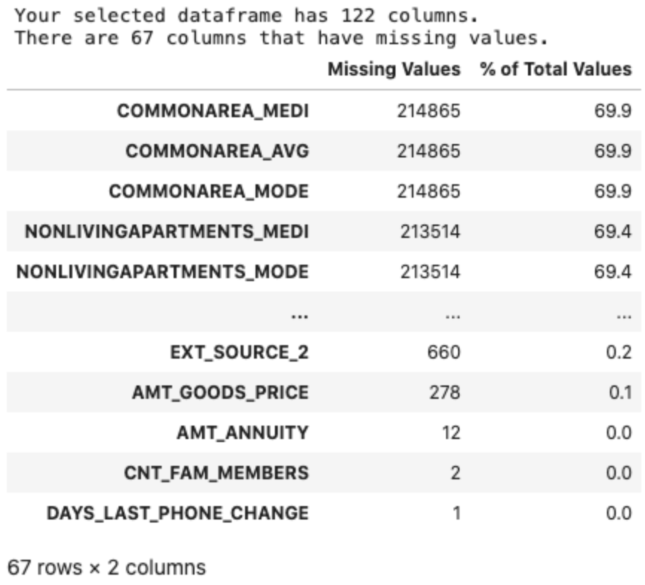


Figure 2 kde plots for highest correlation columns

Missing value analysis is shown in Figure 4



```
Your selected dataframe has 122 columns.
There are 67 columns that have missing values.
```

	Missing Values	% of Total Values
COMMONAREA_MEDI	214865	69.9
COMMONAREA_AVG	214865	69.9
COMMONAREA_MODE	214865	69.9
NONLIVINGAPARTMENTS_MEDI	213514	69.4
NONLIVINGAPARTMENTS_MODE	213514	69.4
...	...	...
EXT_SOURCE_2	660	0.2
AMT_GOODS_PRICE	278	0.1
AMT_ANNUITY	12	0.0
CNT_FAM_MEMBERS	2	0.0
DAYS_LAST_PHONE_CHANGE	1	0.0

67 rows × 2 columns

Figure 3 Missing value analysis

For numerical columns, linear interpolation is used to fill missing values. For discrete columns such as OCCUPATION_TYPE column, missing values are fille with mode.

2.2 Credit bureau report dataset

This dataset has 16 features indexed with multiple bureau reports for each loan application in previous table. We have aggregated numerical columns such as DAYS_CREDIT, CREDIT_DAY_OVERDUE and calculated count, mean, max, min and sum for each of columns and generated new features. Similarly, we have transformed discrete columns such as CREDIT_ACTIVE, CREDIT_TYPE to dummy numerical variables columns and aggregated them. Thus, we have generated 46 new features from 17 features.

2.3 Credit bureau balances dataset

This dataset has 3 features with credit bureau id. Like previous dataset, numeric features are aggregated, and discrete features are transformed to dummy variables and aggregated.

All the three datasets are now merged to a single dataset with 442 columns.

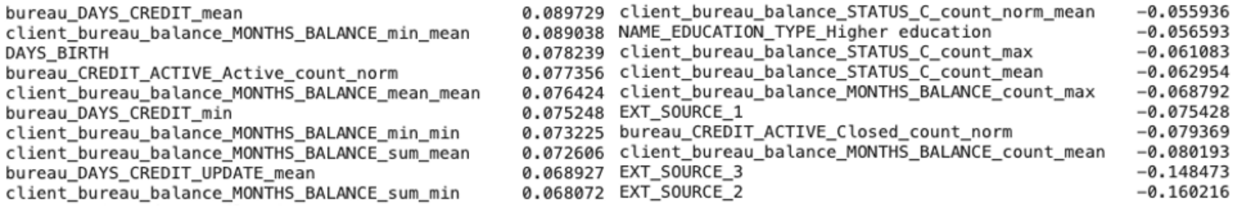
3 Feature engineering

Now we will generate new features from these 442 columns that have better discrimination capacity than existing features using Pearson correlations, multicollinearity analysis, variance analysis.

3.1 Correlation analysis

We have calculated Pearson correlations of each feature with TARGET column(class labels). Figure 5 shows the top positive and negative correlation features. In addition to our inferences in section 2, there is high correlation with features related to

a. CREDIT_ACTIVE – Active/closed Status of the Credit Bureau (CB) reported credits,
b. DAYS_CREDIT - number of days, before current application, did client apply for Credit with another bureau
c. MONTHS_BALANCE - Month of balance in bureau report relative to application date
d. STATUS C – closed past credits



```
bureau_DAYS_CREDIT_mean                          0.089729 client_bureau_balance_STATUS_C_count_norm_mean    -0.055936
client_bureau_balance_MONTHS_BALANCE_min_mean    0.089038 NAME_EDUCATION_TYPE_Higher education              -0.056593
DAYS_BIRTH                                       0.078239 client_bureau_balance_STATUS_C_count_max          -0.061083
bureau_CREDIT_ACTIVE_Active_count_norm           0.077356 client_bureau_balance_STATUS_C_count_mean         -0.062954
client_bureau_balance_MONTHS_BALANCE_mean_mean   0.076424 client_bureau_balance_MONTHS_BALANCE_count_max    -0.068792
bureau_DAYS_CREDIT_min                           0.075248 EXT_SOURCE_1                                      -0.075428
client_bureau_balance_MONTHS_BALANCE_min_min     0.073225 bureau_CREDIT_ACTIVE_Closed_count_norm            -0.079369
client_bureau_balance_MONTHS_BALANCE_sum_mean    0.072606 client_bureau_balance_MONTHS_BALANCE_count_mean   -0.080193
bureau_DAYS_CREDIT_UPDATE_mean                   0.068927 EXT_SOURCE_3                                      -0.148473
client_bureau_balance_MONTHS_BALANCE_sum_min     0.068072 EXT_SOURCE_2                                      -0.160216
```

Figure 4 Features with highest Correlations with TARGET class

3.2 Multicollinear features removal

Multicollinearity is a phenomenon in which one predictor variable in a multiple regression model can be linearly predicted from the others with a substantial degree of accuracy. So we can use one feature out of collinear features set.

In this stage, we have checked correlations among features and removed one of the features with correlation more than 0.8 threshold value. We found 135 such multicollinear features and removed them.

3.3 Low variance features

If the features have a very low variance (i.e., very close to 0), they are close to being constant and thus, do not add any value to any model at all. It would just be better to get rid of them and hence lower the complexity.

Here, we have taken a variance threshold of 0.1 and dropped features with variance less than that threshold. Finally, we have extracted 101 features from transformed dataset in section 2.

4 Model building and evaluation

As shown in section 2, our data has high imbalance in classes with 91:9 proportion of No default risk to default risk TARGET classes. In this context, we have used following techniques to handle this issue:

1. Higher test to train split of 30% to 70%
2. Rather than using accuracy performance metric, we have used confusion matrix, ROC (Receiver operating characteristic) and micro-F1 score.
3. We have created synthetic samples from Default risk class, using SMOTE (Synthetic minority over-sampling technique)

Table 2 shows the sizes of different classes in train and test datasets. We have maintained the proportion of both classes in both train and test datasets.

Table 2 Sizes and class proportions for Train and test dataset

	Class 1(Default risk)	**Class 0(No Default risk)**	**Total**	**Class 1/ Class 0**
Train data	17377	197880	215257	0.09
Test data	7448	84806	92254	0.09

4.1 Logistic regression

We have normalized the train data with mean and standard deviation of corresponding columns. We have used the train mean and standard deviation to standardize test dataset as well.

As a baseline model, we have trained logistic regression model with default parameters. In its basic form, it uses a logistic function to model a binary dependent variable. Following is the simple equation form for logistic regression:

$$\mathrm{y} = \frac{1}{1 + e^{-(\beta_0+\beta_1 * \mathrm{x})}}$$

After fitting with the train data, we have obtained ROC-AUC (Area under curve) values of 0.6945 for Train data and 0.6864 for test data.
Significance of ROC curve is:

1. the closer the curve follows left hand border and the top border of ROC space, the more accurate the model is.
2. The closer the curve comes to 45-degreee diagonal of ROC space, the less accurate the mode is
3. The area under ROC curve (AUC) represents the discriminating ability of model to correctly classify the cases.

Figure 6 shows the confusion matrix and ROC curve. It shows that the model has been able to predict No Default class with high accuracy, whereas it has very poor accuracy in detecting Default risk class. Particularly, there are many false positives when compared to false negatives.

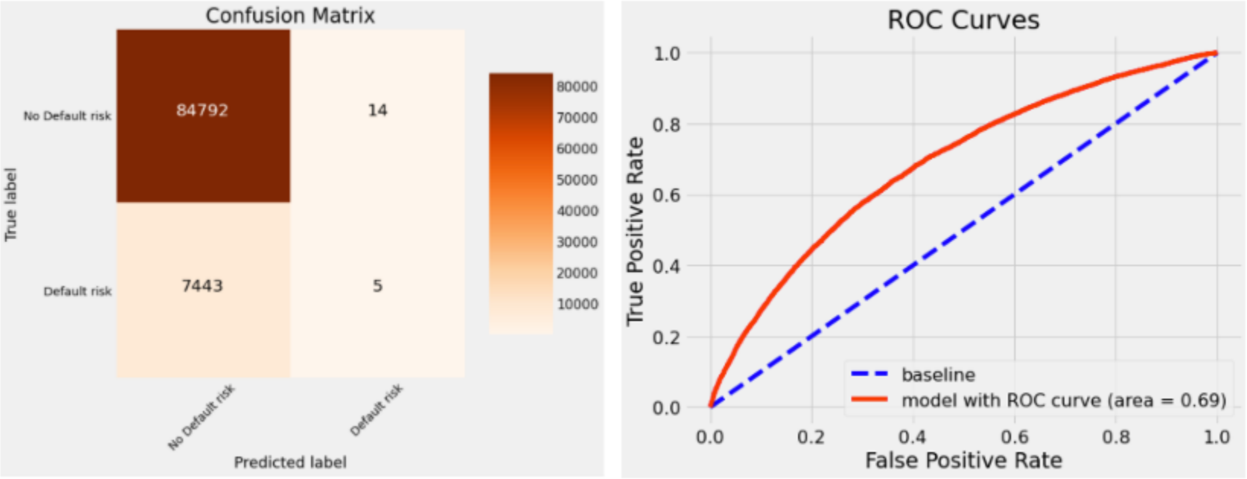


Figure 5 Confusion matrix and ROC curve for test data for logistic regression

$$\text{F1 score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Micro-F1 score calculates metrics globally by counting the total true positives, false negatives, and false positives. For this model, we got a score of 91.92.

4.2 Random forest

Random forest model is created by constructing a multitude of decision trees at training time and outputting the mode of the classes of individual trees for each case. It performs better than decision trees, where a tree can grow very deep tending to learn noise along with signal in the data thereby leading to overfitting, i.e., low bias but very high variance. Random forest uses bootstrap aggregation for individual decision trees, so that it selects a random sample with replacement of training set and fits trees to the bootstrapped samples. Thus, it decreases variance of the model without increasing the bias disproportionately.

Unlike logistic regression model, we have not normalized the data and used original dataset for training.

4.2.1 Default parameters model

We have used following default parameters for training:

1. Number of estimators/trees = 100
2. Max_features to consider when looking for best split = sqrt(# features)
3. Bootstrap = True
4. Gini impurity criterion
5. Max leaf nodes – unlimited
6. Min samples split required for an internal node – 2

Figure 7 shows the confusion matrix and ROC curve for test dataset. As per the results, the True negative detection rate is low, but False positive rate decreased marginally. However, ROC-AUC value is less than logistic regression results.

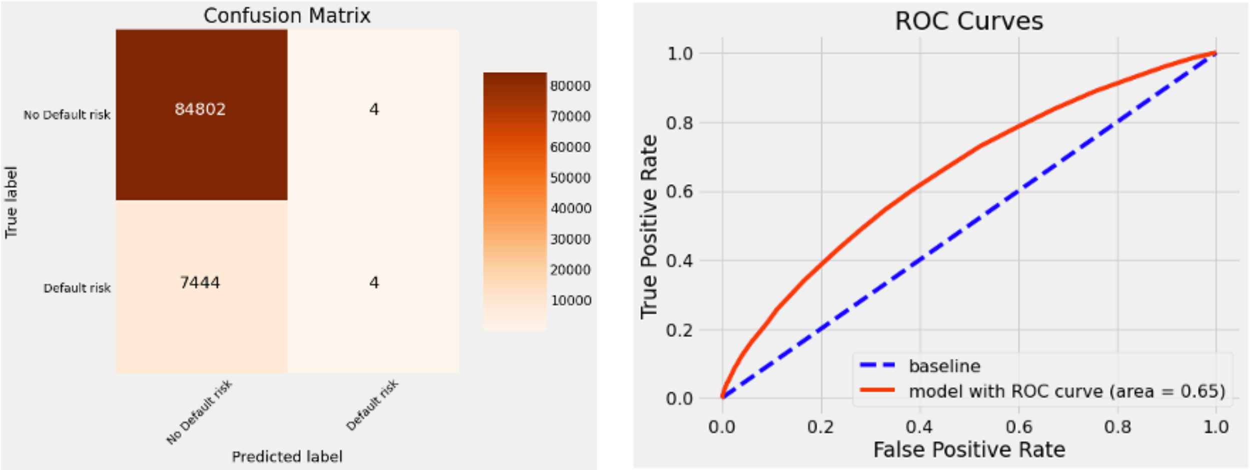


Figure 6 Confusion matrix and ROC curve for test data for default Random forest model

4.2.2 Hyper parameter optimization

We have done a randomized search along with cross validation in the parameter space shown in Table 3

Table 3 Hyper parameter space for randomized CV search for Random forest

S No	Feature	Parameter space
1	#estimators	[10, 200]
2	Max depth	[3,20] and unlimited
3	Max features	sqrt, all features; range (0.5,1, 0.1)
4	Max leaf nodes	Unlimited; [10,50]
5	Min samples split	2,5,10
6	Bootstrap	True and False

4.2.3 Best model

Based on ROC-AUC score metric, we found the following set of parameters gave highest score :

1. 'n_estimators': 134,
2. 'min_samples_split': 10,
3. 'max_leaf_nodes': 48,
4. 'max_features': sqrt',
5. 'max_depth': 18,
6. 'bootstrap': True

Figure 8 shows the confusion matrix and ROC curve for test dataset. As per the results, the True negative detection rate is zero, but False positive rate also zero. This is unusual since our model

cannot detect Default risk, which is important. This might be because of low Default risk class cases.
However, ROC-AUC value is marginally less than logistic regression results and more than default parameters. Micro-f1 score is 91.93, which is almost same as for logistic regression.

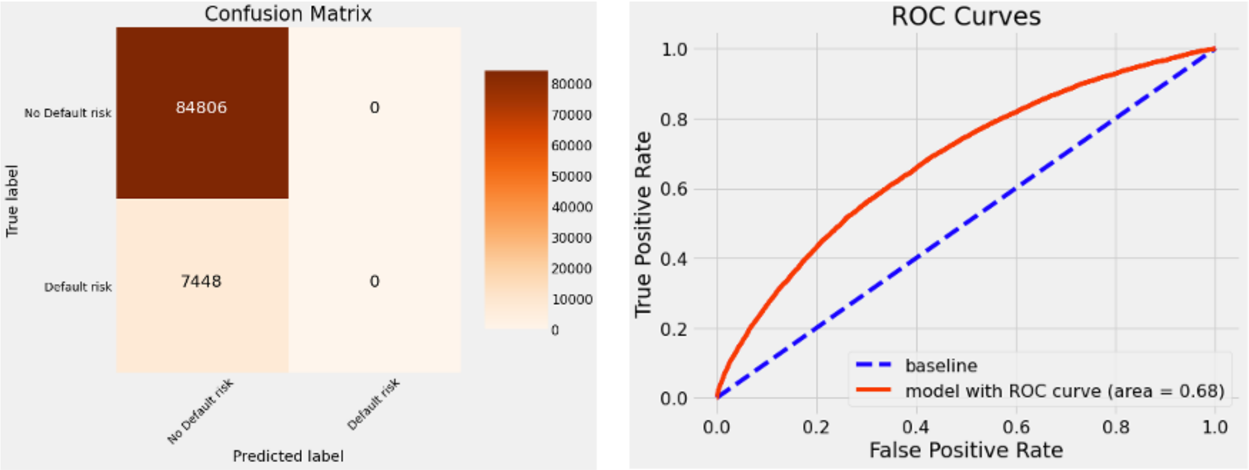


Figure 7 Confusion matrix and ROC curve for test data for best Random forest model

4.3 SMOTE and Random forest

4.3.1 SMOTE with kNN

SMOTE refers to Synthetic Minority Oversampling technique, which is a data augmentation technique for minority class. It selects examples that are close in feature space, drawing a line between examples in the feature space and drawing a new sample at a point along that line. Specifically, a random example from minority class is first chosen. Then k of the nearest neighbors for that example are found, typically n =5 is chosen. A randomly selected neighbor is chosen, and a synthetic example is created at a randomly selected point between the two examples in feature space.

This approach is effective because new synthetic examples from minority class are created that are plausible since they are relatively close in feature space to existing minority class examples.

4.3.2 Hyperparameter optimization

We have used the same set of hyperparameter space as in section 4.2.2.

4.3.3 Best Model

Based on ROC-AUC score metric, we found the following set of parameters gave highest score:

1. 'n_estimators': 134,
2. 'min_samples_split': 10,
3. 'max_leaf_nodes': 48,
4. 'max_features': sqrt',
5. 'max_depth': 18,

6. 'bootstrap': True

These are same parameter set as Random forest model.

Figure 9 shows the confusion matrix and ROC curve for test dataset. As per the results, True negative case detection rate has improved compared to regular Random forest model, though there is an increase in False negative rate compared to Random forest results in Figure 8. However, ROC-AUC score 0.63 and micro-F1 score are lesser than Random forest results.

Overall, we can say that this model has better ability to detect Default risk than other models even though overall ROC-AUC and micro-F1 scores are less.

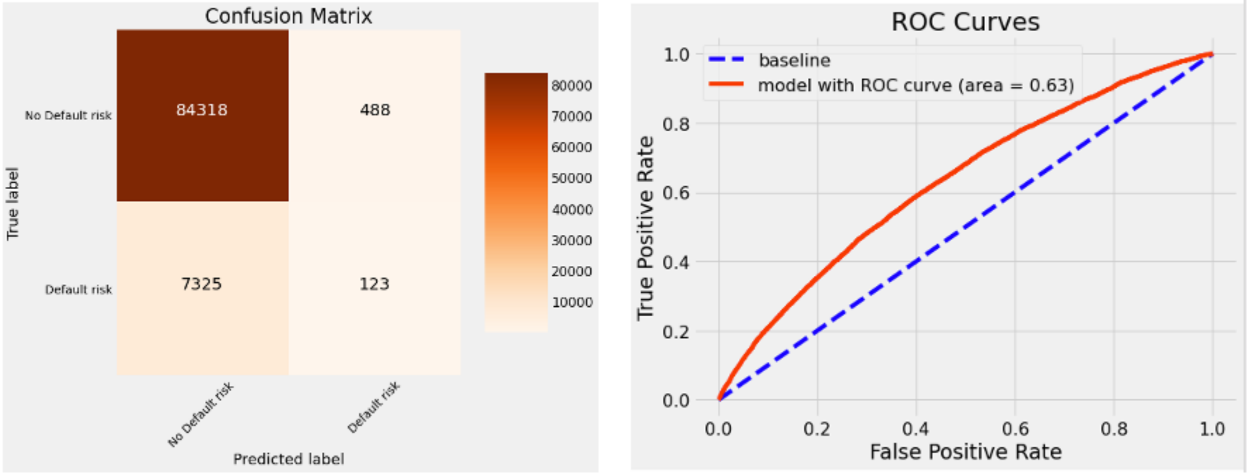


Figure 8 Confusion matrix and ROC curve for test data for best SMOTE- Random forest model

5 Model interpretation

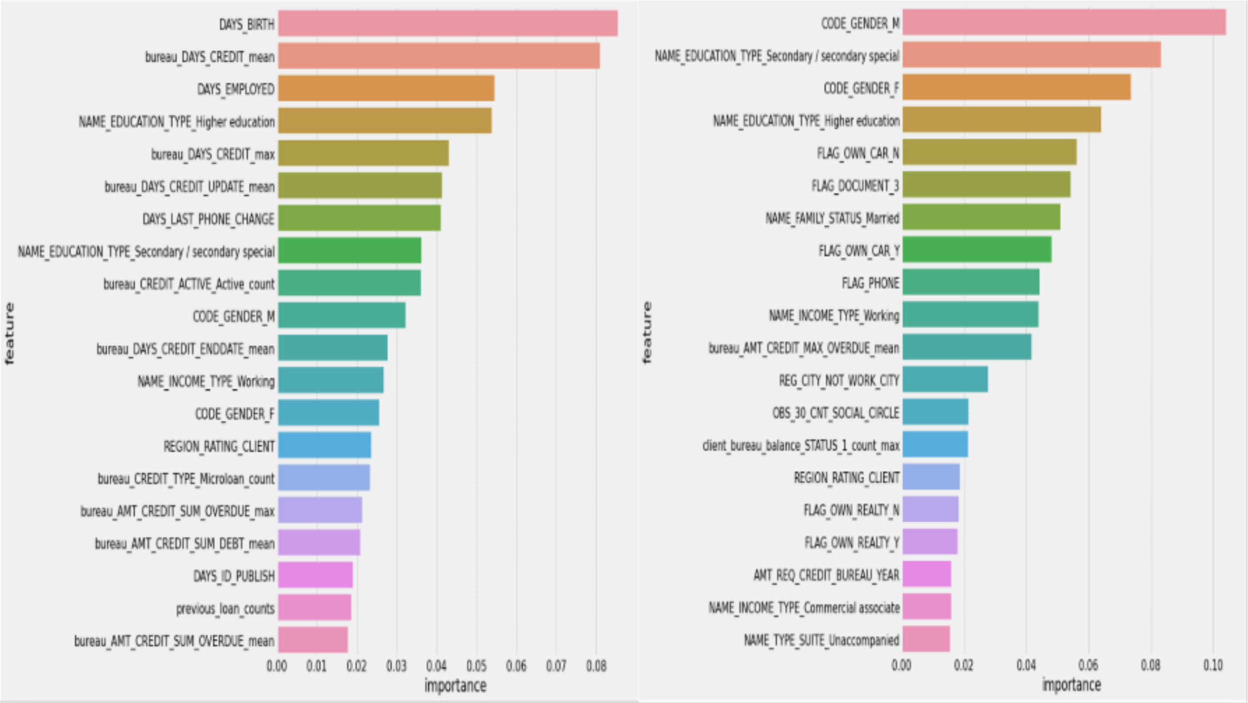


Figure 9 Top 20 features based on importance from Random forest best model(left) and SMOTE-Random forest best model(right)

Figure 10 shows the top 20 features based on their importance, extracted from best random forest model on left side and SMOTE-Random Forest model on right side. We can see that features such as Gender, education, car ownership, family status, property ownership are more important features in detecting credit default risk. This is more holistic set of features than the features seen in left side bar plot, which includes features mostly related to past credit history.

6 Conclusion

In this project, we have explored and generated features from application data and past credit history to predict home credit default risk of a customer. We have analyzed the data using several models such as logistic regression, random forest, SMOTE-random forest models. We have found that accounting for imbalance in dataset using SMOTE has improved the prediction of credit default risk, which is more important than prediction no default risk class. In future work, we would like to incorporate other datasets such as previous application data, past installment payment history, credit card history and try different techniques of filling missing values and generate more reliable features. We can also use gradient boosting machines, which can work better on missing data features.

7 Appendix

Code for this project is hosted at https://colab.research.google.com/drive/1KmtPLZMW-lG7565GkXX9R8wj1WCGCgDD?usp=sharing