Project Report On

Santander Customer Transaction Prediction

Submitted By

POONAM LAL

INDEX

CHAPTER 01

INTRODUCTION

1.1 Problem Statement

At Santander, mission is to help people and businesses prosper. We are always looking for ways to help our customers understand their financial health and identify which products and services might help them achieve their monetary goals.
Our data science team is continually challenging our machine learning algorithms, working with the global data science community to make sure we can more accurately identify new ways to solve our most common challenge, binary classification problems such as: is a customer satisfied? Will a customer buy this product? Can a customer pay this loan?
In this challenge, we need to identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

1.2 Data Understanding

Understanding of data is the very first and important step in the process of finding solution of any business problem. Here in our case our company has provided a data set with following features, we need to go through each and every variable of it to understand and for better functioning.

- In this project, our task is to build classification models which will be used to predict which customers will make a specific transaction in the future.
- Size of Train Dataset: - 200000 rows, 202 Columns (including dependent variable).
- Size of Test Dataset: - 200000 rows, 201 Columns.

CHAPTER 02

Methodology

1. Pre-Processing

When we required to build a predictive model, we require to look and manipulate the data before we start modelling which includes multiple pre-processing steps such as exploring the data, cleaning the data as well as visualizing the data through graph and plots, all these steps is combined under one shed which is **Exploratory Data Analysis**, which includes following steps:

- Data exploration and Cleaning
- Outlier Analysis
- Missing values treatment
- Feature Selection
- Feature Engineering
- Features Scaling
    - Skewness and Log transformation

2. Modelling

Once all the Pre-Processing steps has been done on our data set, we will now further move to our next step which is modelling. Modelling plays an important role to find out the good inferences from the data. Choice of models depends upon the problem statement and data set. As per our problem statement and dataset, we will try some models on our pre-processed data and post comparing the output results we will select the best suitable model for our problem. As our data is imbalanced we are going to use multiple approaches for dealing with imbalanced datasets.

- Oversample minority class.
- Under sample majority class.
- Change of performance matrix.
- SMOTE (Synthetic Minority Oversampling technique)
- ROSE
- Change of algorithm (Light GBM)

3. Model Selection

The final step of our methodology will be the selection of the model based on the different output and results shown by different models. We have multiple parameters which we will study further in our report to test whether the model is suitable for our problem statement or not.

CHAPTER 03

Pre-Processing

3.1 Exploratory Data Analysis (EDA)

Exploratory data analysis is one of the most important steps in data mining in order to know features of data. It involves the loading dataset, target classes count, data cleaning, typecasting of attributes, missing value analysis, Attributes distributions and trends. So, we must clean the data otherwise it will affect on performance of the model. Now we are going to explain one by one as follows. In this EDA I explained with seaborn visualizations.

3.1.(a) The data we have looks like this:

Table 1.1: Train dataset (Columns:1-202)

```
In [6]: #see top 5 observation
        train.head()
```

Out[6]:

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	var_8	var_9	var_10	var_11	var_12	var_13	var_14	var_15
0	train_0	0	8.9255	-6.7863	11.9081	5.0930	11.4607	-9.2834	5.1187	18.6266	-4.9200	5.7470	2.9252	3.1821	14.0137	0.5745	8.7989	14.5691
1	train_1	0	11.5006	-4.1473	13.8588	5.3890	12.3622	7.0433	5.6208	16.5338	3.1468	8.0851	-0.4032	8.0585	14.0239	8.4135	5.4345	13.7003
2	train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	14.6155	-4.9193	5.9525	-0.3249	-11.2648	14.1929	7.3124	7.5244	14.6472
3	train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	14.9250	-5.8609	8.2450	2.3061	2.8102	13.8463	11.9704	6.4569	14.8372
4	train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	19.2514	6.2654	7.6784	-9.4458	-12.1419	13.8481	7.8895	7.7894	15.0553

Table 1.2: Test Dataset (Columns: 1-201)

```
In [9]: #Importing the test dataset:-
        test=pd.read_csv("test.csv")

In [10]: test.head()
```

Out[10]:

	ID_code	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	var_8	var_9	var_10	var_11	var_12	var_13	var_14	var_15	var_1
0	test_0	11.0656	7.7798	12.9536	9.4292	11.4327	-2.3805	5.8493	18.2675	2.1337	8.8100	-2.0248	-4.3554	13.9696	0.3458	7.5408	14.5001	7.702
1	test_1	8.5304	1.2543	11.3047	5.1858	9.1974	-4.0117	6.0196	18.6316	-4.4131	5.9739	-1.3809	-0.3310	14.1129	2.5667	5.4988	14.1853	7.019
2	test_2	5.4827	-10.3581	10.1407	7.0479	10.2628	9.8052	4.8950	20.2537	1.5233	8.3442	-4.7057	-3.0422	13.6751	3.8183	10.8535	14.2126	9.883
3	test_3	8.5374	-1.3222	12.0220	6.5749	8.8458	3.1744	4.9397	20.5660	3.3755	7.4578	0.0095	-5.0659	14.0526	13.5010	8.7660	14.7352	10.038
4	test_4	11.7058	-0.1327	14.1295	7.7506	9.1035	-8.5848	6.8595	10.6048	2.9890	7.1437	5.1025	-3.2827	14.1013	8.9672	4.7276	14.5811	11.861

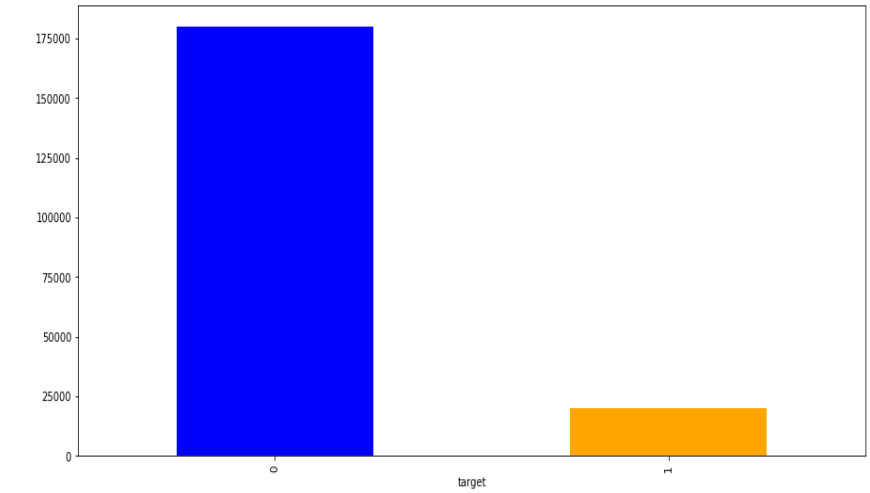
From the table below, we have the following 16 variables, using which we have to predict the bike rental count:

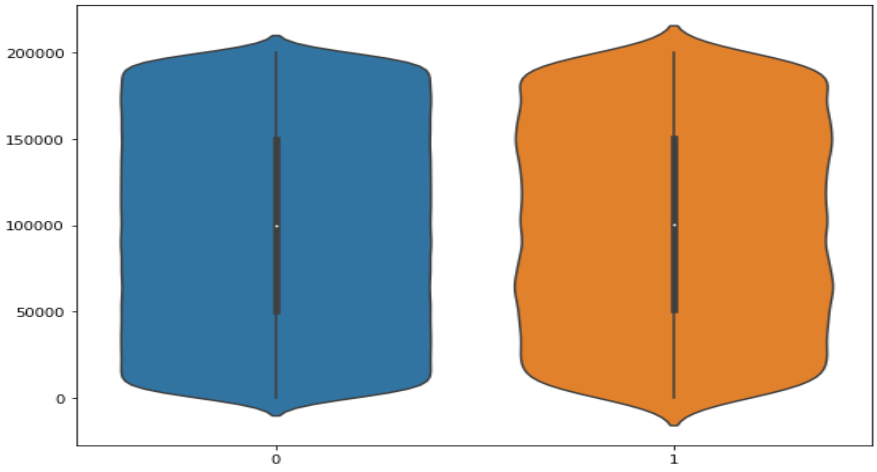
Table 1.3: Predictor Variables

SL.No.	Predictor
1	ID-code
2	var0
3	var1
4	var2
5	var3
6	var4
7	var5
….	.. …
…..	

…. …. …. …. ….	…… …… ……. ……. …….
…. 202	…….. var199

3.1(b) Target classes count





Observation: -

- The data is unbalanced with respect with target value.
- We are having a unbalanced data, where 90% of the data is no. of customers who will not make a transaction & 10 % of the data are those who will make a transaction.
- From the violin plots, it seems that there is no relationship between the target and index of the data frame, it is more dominated by zero compare to one's.

3.2 Outlier

In this project, we haven't performed outlier analysis due to the data is imbalanced and also not required for imbalanced data.

3.3 Missing value Analysis

In this, we have to find out any missing values are present in dataset. If it's present then either delete or impute the values using mean, median and KNN imputation method. We have not found any missing values in both train and test data.

Out[22]:

	index	0
0	ID_code	0
1	target	0
2	var_0	0
3	var_1	0
4	var_2	0

```
In [23]: train.isnull().values.any()
Out[23]: False

In [24]: test.isnull().values.any()
Out[24]: False
```

Observation: -

- There are no missing data in train and test datasets.

3.4 Attributes distributions and trends

- **Distribution of train attributes**

  - Let's show now the density plot of variables in train dataset.
  - We represent with different colors the distribution for values with target value 0 and 1.
  - Let us look distribution of train attributes from var_0 to var_99
  - Observation:

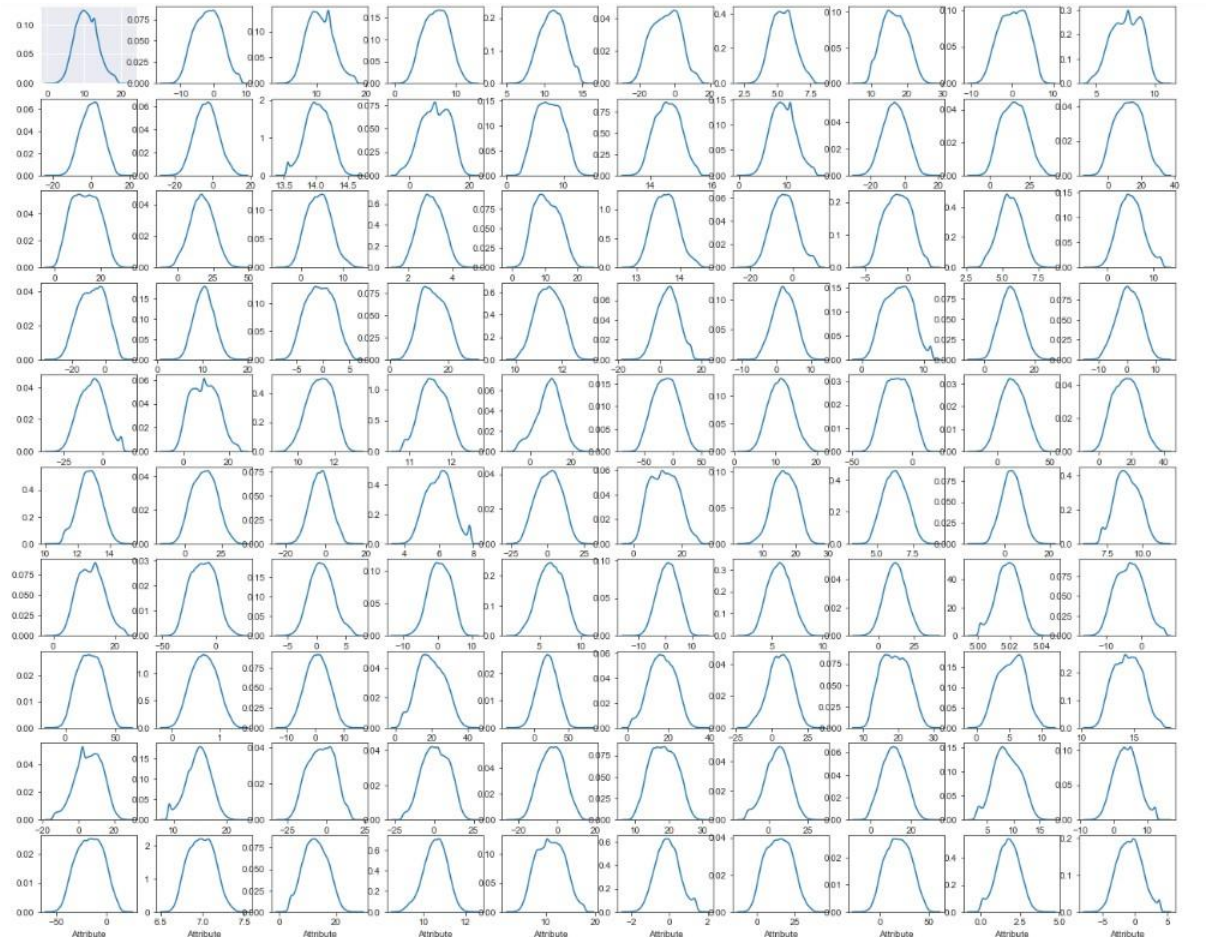
    1. We can observe that there is a considerable number of features which are significantly have different distributions for two target variables. For example, like var_0, var_1, var_9, var_19, var_18 etc.
    2. We can observe that there is a considerable number of features which are significantly have same distributions for two target variables. For example, like var_3, var_7, var_10, var_17, var_35 etc.

Table 3.1 Density plot of variable in train dataset

- **Distribution of test attributes**

Let us look distribution of test attributes from var_0 to var_99

Table 3.2- Density plot for variable in test dataset



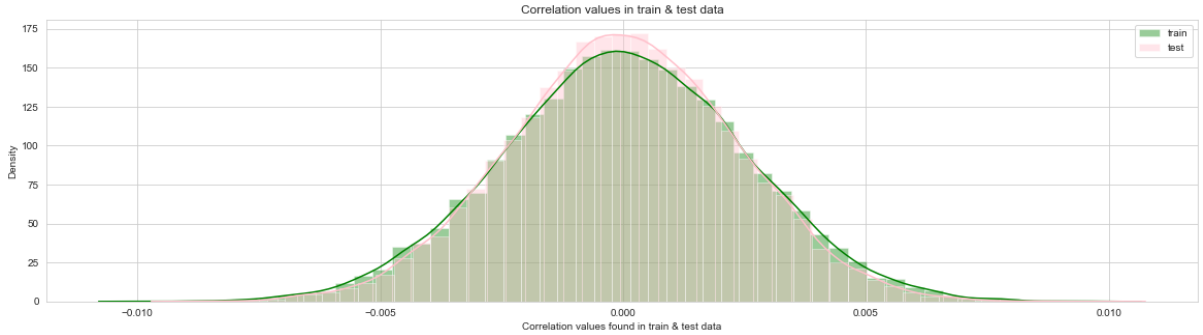
3.5 Feature Selection

Feature selection is very important for modelling the dataset. Every dataset has good and unwanted features. The unwanted features would affect on performance of model, so we have to delete those features. We have to select best features by using ANOVA, Chi-Square test and correlation matrix statistical techniques and so on. In this, we are selecting best features by using Correlation matrix.

Correlation matrix

Correlation matrix, it tells about linear relationship between attributes and help us to build better models.
From correlation distribution plot, we can observe that correlation between both train and test attributes are very small. It means that all both train and test attributes are independent to each other.

Table 3.3 Correlation values in train and test data



3.6 Feature engineering

We are performing feature engineering by using

Permutation importance: -

- Permutation feature importance is a model inspection technique that can be used for any fitted estimator when the data is rectangular. This is especially useful for non-linear or opaque estimators. The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled

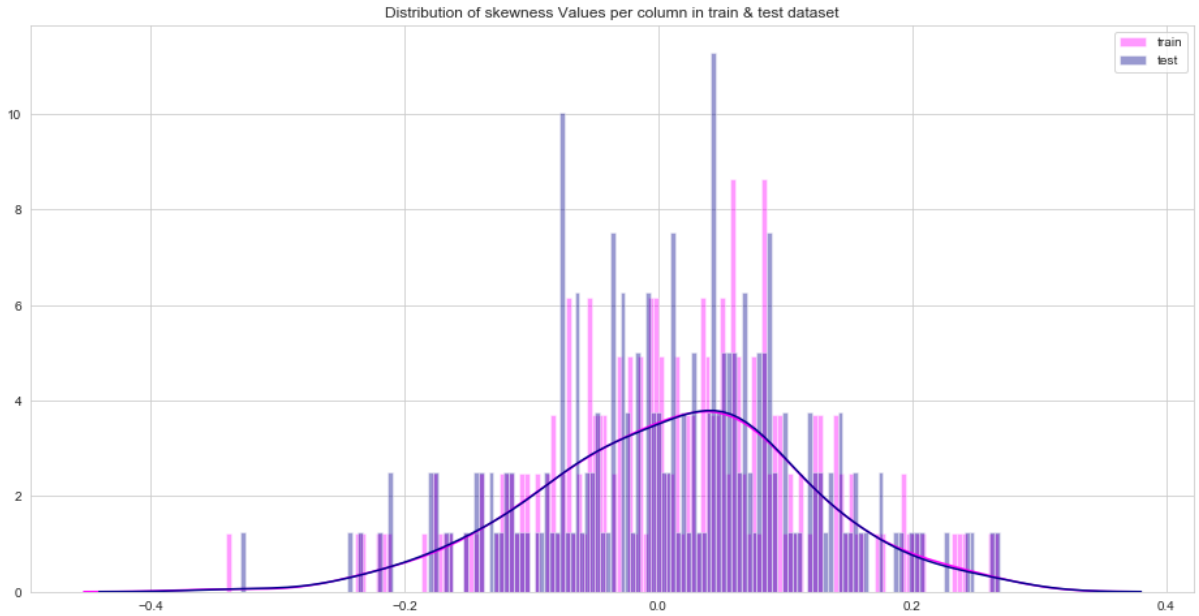
- Permutation variable importance measure in a random forest for classification and regression. The variables which are mostly contributed to predict the model.

Weight	Feature
0.0004 ± 0.0002	var_81
0.0003 ± 0.0002	var_146
0.0003 ± 0.0002	var_109
0.0003 ± 0.0002	var_12
0.0002 ± 0.0001	var_110
0.0002 ± 0.0000	var_173
0.0002 ± 0.0001	var_174
0.0002 ± 0.0002	var_0
0.0002 ± 0.0002	var_26
0.0001 ± 0.0001	var_166
0.0001 ± 0.0001	var_169
0.0001 ± 0.0001	var_22
0.0001 ± 0.0001	var_99
0.0001 ± 0.0001	var_53
0.0001 ± 0.0001	var_8

Observation: -

- Importance of features is decreasing as we move down the top of column.
- Features showing in green indicates they are having positive impact on our prediction.
- Features showing in white showing they have no impact on prediction.
- Most important feature is var_81.

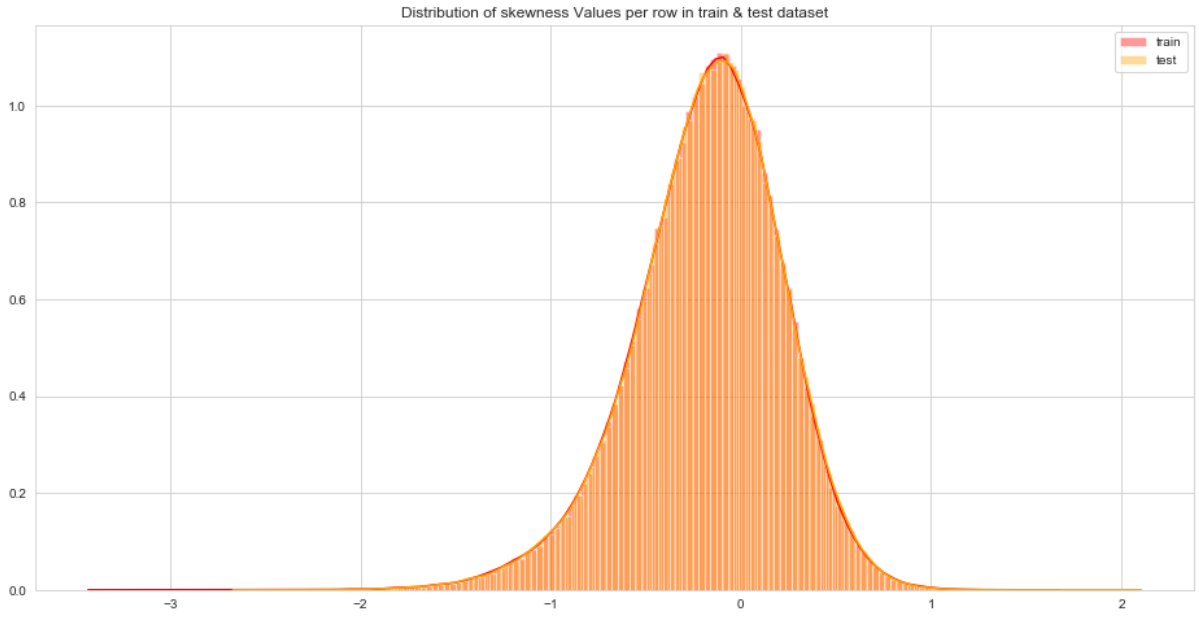
3.7 Feature scaling



Distribution of skewness values in train and test dataset

Let us look distribution of skewness values per column in train and test dataset: -

Let us look distribution of skewness values per column in train and test dataset:-



CHAPTER 04

Modeling

After all early stages of preprocessing, then we will do model selection. So, we have to select best model for this project with the help of some metrics.
The dependent variable can fall in either of the four categories:

1. Nominal
2. Ordinal
3. Interval
4. Ratio

If the dependent variable is Nominal the only predictive analysis that we can perform is **Classification**, and if the dependent variable is Interval or Ratio like this project, the normal method is to do a **Regression** analysis, or classification after binning.

4.1 Handling of imbalance data

Now we are going to explore 5 different approaches for dealing with imbalanced datasets.

1. Oversample minority class
2. Under sample majority class
3. Change the performance metric
4. Synthetic Minority Oversampling Technique (SMOTE) in Python
   Random Oversampling Examples (ROSE) in R
5. Change the algorithm (Light GBM)

Here we apply first three technique on logistic Regression model

4.2 Model Used

1. Logistic Regression
2. SMOTE/ROSE
3. Light GBM

We always start model building from the simplest to more complex.

Logistic Regression

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is binary. Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

In this project Accuracy of the model is not the best metric to use when evaluating the imbalanced datasets as it may be misleading. So, we are going to change the performance metric.

4.1(1) *Oversampling Minority Class*

- Adding more copies of minority class.
- It can be a good option we don't have that much large data to work.
- Drawback of this process is we are adding info. That can lead to overfitting or poor performance on test data.

4.1(2) ***Under sampling Majority Class***

- Removing some copies of majority class.
- It can be a good option if we have very large amount of data say in millions to work.
- Drawback of this process is we are removing some valuable info. that can leads to underfitting & poor performance on test data.

4.1(3) ***Change of Performance Metric***

- For classification problems, the **confusion matrix** used for evaluation.
- But, in our case the data is imbalanced. So, **roc_auc_score** is used for evaluation.

4.1(4) ***Synthetic Minority Oversampling Technique (SMOTE)***

- In order to balance imbalanced data, we are going to use SMOTE sampling method in Python.
- SMOTE uses a nearest neighbor's algorithm to generate new and synthetic data to use for training the model.
- As per the drawbacks of both the Oversampling and Under sampling model we will use SMOTE (Synthetic Minority Oversampling technique) that is comparatively best model.

4.1(5) ***Random Oversampling Examples (ROSE)***

- In order to balance imbalanced data, we are going to use ROSE sampling method in R.
- It creates a sample of synthetic data by enlarging the features space of minority and majority class examples.
- As per the drawbacks of both the Oversampling and Under sampling model we will use ROSE that is comparatively best model.

4.1(6) ***LightGBM***

LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Capable of handling large-scale data.

CHAPTER 05

Conclusion

5.1 Model Evaluation

Now, we have three models for predicting the target variable, but we need to decide which model better for this project. There are many metrics used for model evaluation. Classification accuracy may be misleading if we have an imbalanced dataset or if we have more than two classes in dataset.

For classification problems, the confusion matrix used for evaluation. But, in our case the data is imbalanced. So, roc_auc_score is used for evaluation.

In this project, we are using two metrics for model evaluation as following:

5.1 (a) *Confusion Matrix*: -

- It is a technique for summarizing the performance of a classification algorithm.
- The number of correct predictions and incorrect predictions are summarized with count values and broken down by each class.

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

- Accuracy: - The ratio of correct predictions to total predictions

$$\text{Accuracy} = \frac{TP+TN}{Total\ predcition}$$

- Misclassification error: - The ratio of incorrect predictions to total predictions

$$\text{Error rate} = \frac{FN+FP}{Total\ prediction}$$

- Accuracy=1-Error rate
- $\text{Precision} = \frac{TP}{TP+FP}$
- $\text{Recall} = \frac{TP}{TP+FN}$ = True Positive Rate (TPR)
- $\text{Specificity} = \frac{TN}{TN+FP}$ = True Negative Rate (TNR)

- False Positive Rate (FPR) = $\frac{FP}{FP+TN}$
- False Negative rate (FNR) =$FN/FN+TP$

- F1 score: - Harmonic mean of precision and recall, used to indicate balance between them

5.1(b)*Receiver operating characteristics (ROC)_Area under curve(AUC) Score*

roc_auc_score :- It is a metric that computes the area under the Roc curve and also used metric for imbalanced data.
Roc curve is plotted true positive rate or Recall on y axis against false positive rate or specificity on x axis. The larger the area under the roc curve better the performance of the model.

Logistic Regression

On applying LR model the model accuracy comes-0.9121

(a) Confusion Matrix

3.2 Confusion Matrix:

```
In [41]: #Confusion matrix:-
         cm=confusion_matrix(Y1_valid,cv_predict)
         cm=pd.crosstab(Y1_valid,cv_predict)
         cm
```

Out[41]:

col_0	0	1
target		
0	35484	496
1	3093	927

(b) ROC_AUC score

3.2(a) ROC_AUC Score

```
In [140]: #ROC_AUC SCORE:-
          roc_score=roc_auc_score(Y1_valid,cv_predict)
          print('ROC Score:',roc_score)

          ROC Score: 0.6084057892859217
```

(c) Classification Report

3.2(b) Classification report(Precision, Recall, F1 score)

```
]: #Classification report:-
classification_scores=classification_report(Y1_valid,cv_predict)
print(classification_scores)
```

```
              precision    recall  f1-score   support

           0       0.92      0.99      0.95     35980
           1       0.65      0.23      0.34      4020

    accuracy                           0.91     40000
   macro avg       0.79      0.61      0.65     40000
weighted avg       0.89      0.91      0.89     40000
```

Observation: -

- On comparing roc_auc_score and model accuracy, model is not performing well on imbalanced data.
- As we see that f1 score is high for the customers who will not make a transaction, compare to those who will make a transaction. So, we are going to change the algorithm.

SMOTE/ROSE

(a) Confusion matrix

4.3 Confusion Matrix

```
In [148]: #Confusion matrix:-
cm=confusion_matrix(y_smote_v,cv_pred)
cm=pd.crosstab(y_smote_v,cv_pred)
cm
```

Out[148]:

col_0	0	1
target		
0	28192	7788
1	6874	29106

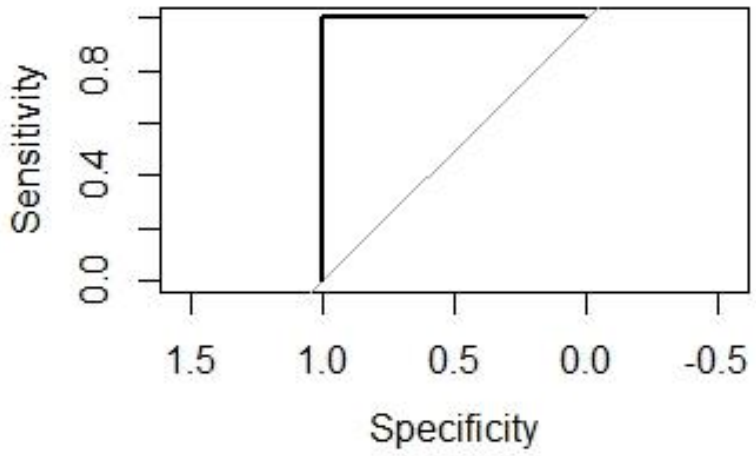
```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1 20012     0
         2     0 19988

               Accuracy : 1
                 95% CI : (0.9999, 1)
    No Information Rate : 0.5003
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1

 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0000
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 1.0000
             Prevalence : 0.5003
         Detection Rate : 0.5003
   Detection Prevalence : 0.5003
      Balanced Accuracy : 1.0000

       'Positive' Class : 1
```

(b) ROC_AUC score

4.3(a) ROC_AUC Score

```
In [149]: #ROC_AUC SCORE:-
          roc_score=roc_auc_score(y_smote_v,cv_pred)
          print('ROC score:',roc_score)
```

```
ROC score: 0.7962479155086158
```



(c) Classification Report

4.3(b) Classification report(Precision, Recall, F1 score)

```
:  #Classification Report:-
   scores=classification_report(y_smote_v,cv_pred)
   print(scores)
```

```
              precision    recall  f1-score   support

           0       0.80      0.78      0.79     35980
           1       0.79      0.81      0.80     35980

    accuracy                           0.80     71960
   macro avg       0.80      0.80      0.80     71960
weighted avg       0.80      0.80      0.80     71960
```

Observation –

- We can observe that the smote / rose model is performing well on imbalance data as compare to logistic regression.
- As we see that f1 score is high for the customers who will not make a transaction, as well as who will make a transaction.

LightGBM

We apply the LightGBM model on the train and test data.

LGBM model performance on test data:-

```
X1_test=test.drop(['ID_code'],axis=1)
#Predict the model:-

#probability predictions
lgbm_predict_prob=lgbm.predict(X_test,random_state=42,num_iteration=lgbm.best_iteration)

#Convert to binary output 1 or 0
lgbm_predict=np.where(lgbm_predict_prob>=0.5,1,0)
print(lgbm_predict_prob)
print(lgbm_predict)
```

```
[0.49802717 0.49891065 0.49846851 ... 0.49939738 0.50047844 0.49908944]
[0 0 0 ... 0 1 0]
```

Observation -

- Model is performing well on imbalanced data and predicting more accurate values as compared to the rest of other two models

5.2 Model Selection

When we compare scores of area under the ROC curve of all the models for an imbalanced data. We could conclude that below points as follow,

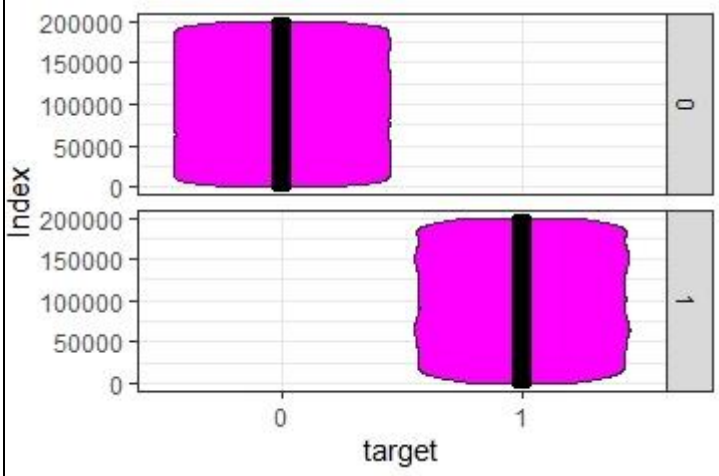
- Logistic regression model is not performed well on imbalanced data.
- We balance the imbalanced data using resampling techniques like SMOTE in python and ROSE in R.
- SMOTE/ROSE are performing well on imbalanced data and we got area under ROC curve is 1 which may not be possible.
- Baseline logistic regression model is performed well on balanced data.
- LightGBM model performed well on imbalanced data.

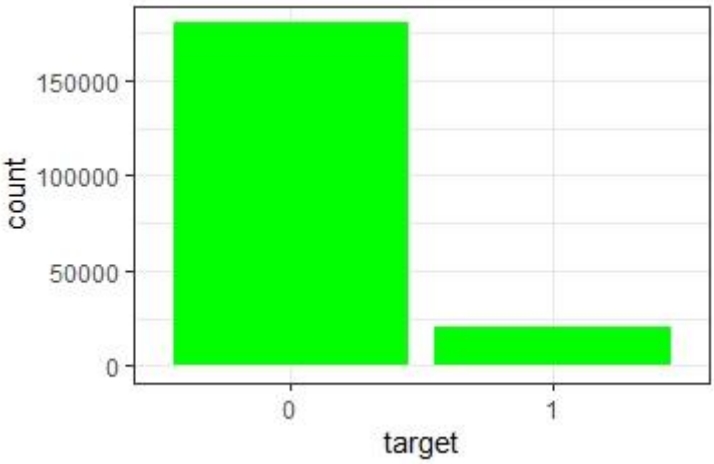
Finally, LightGBM is best choice for identifying which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

5.3 Visualization

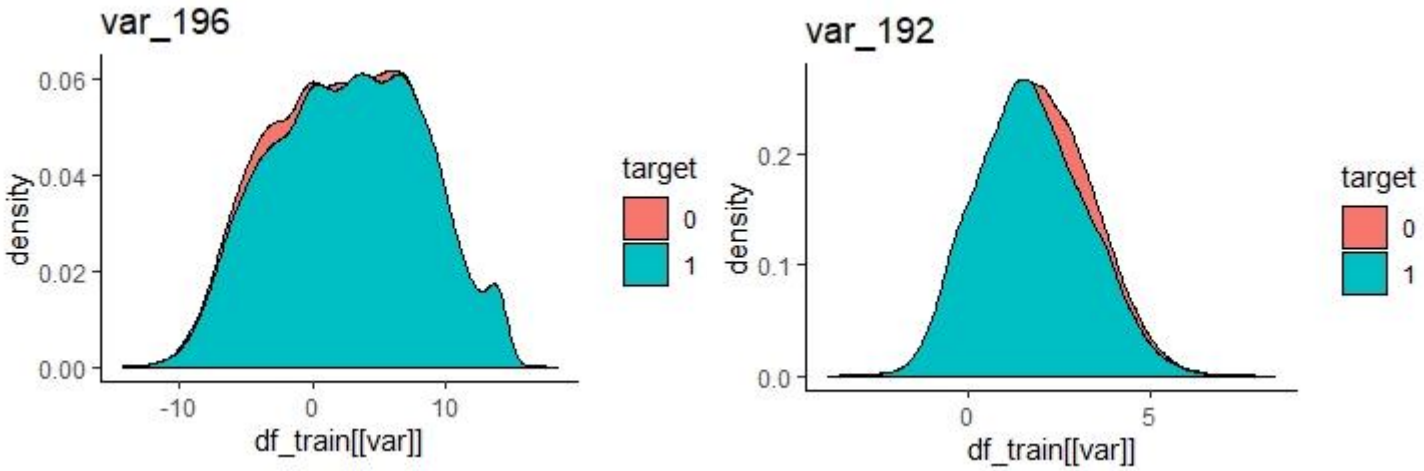
Some extra visualization figures in R

(a)Target class count and Violin plot

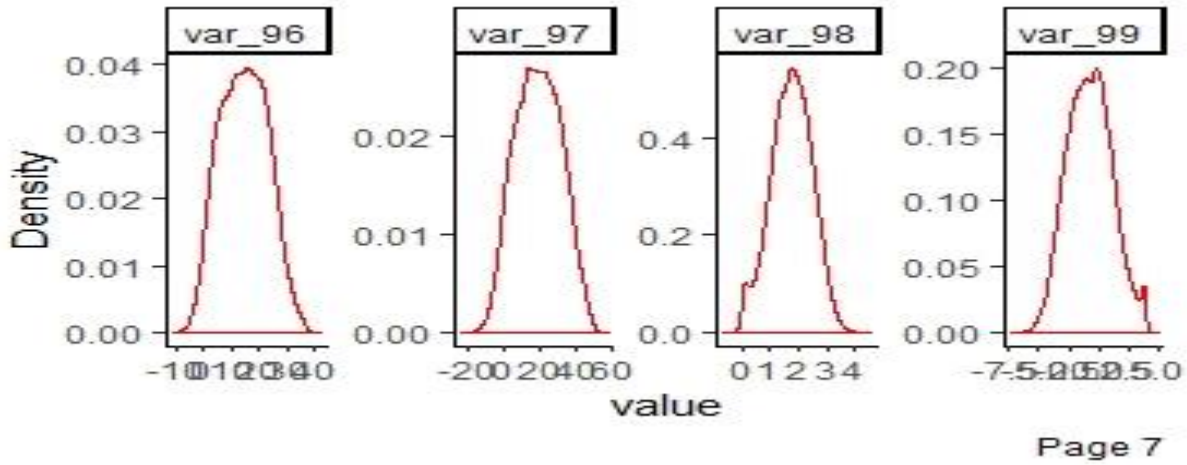




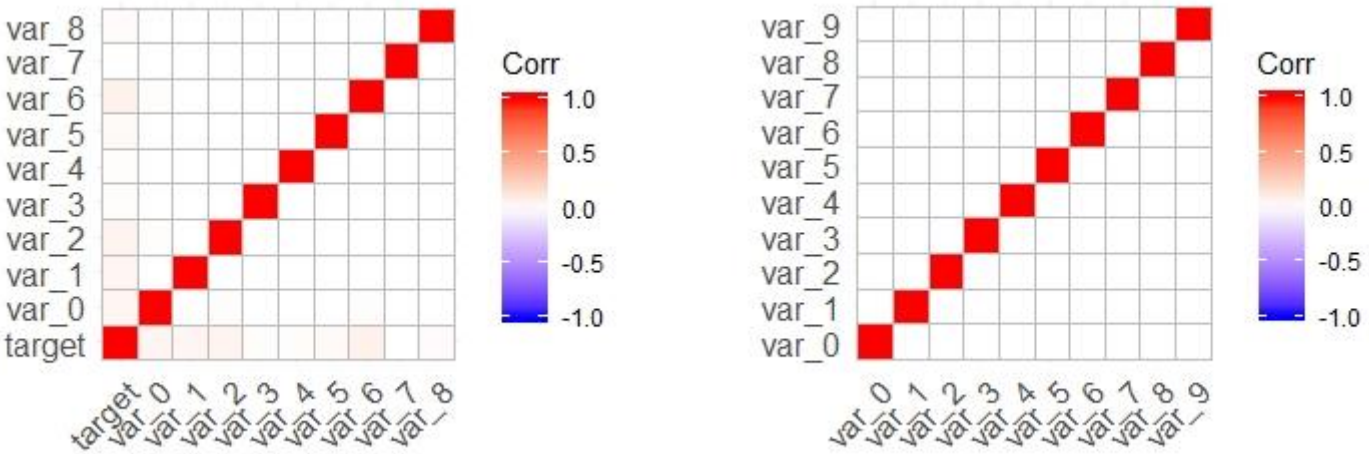
(b) Density Distribution of Target variable in Train dataset



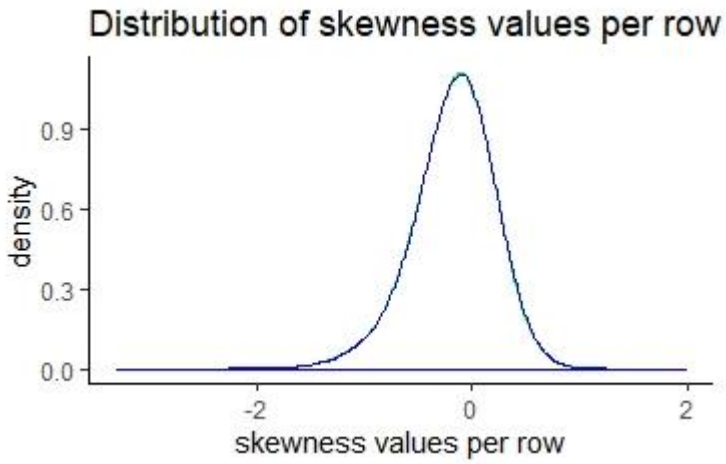
(c)Density Distribution plot of variables in Test dataset



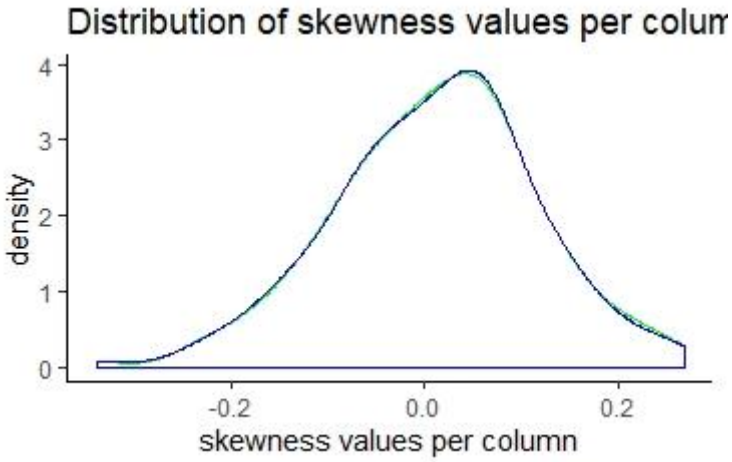
(d) Correlation plot of top 10 variables in Train and Test dataset respectively.



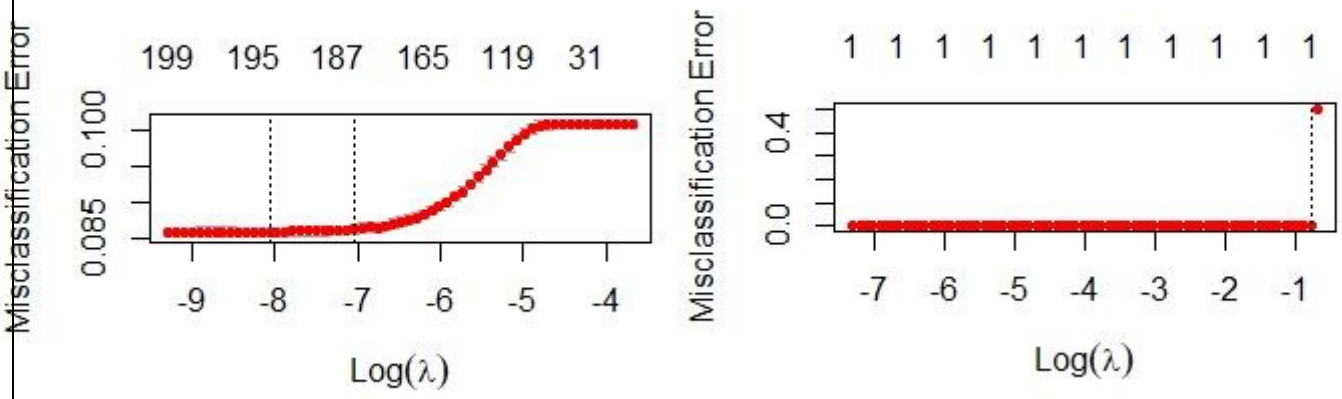
(e) Distribution of Skewness per row in Train and Test dataset



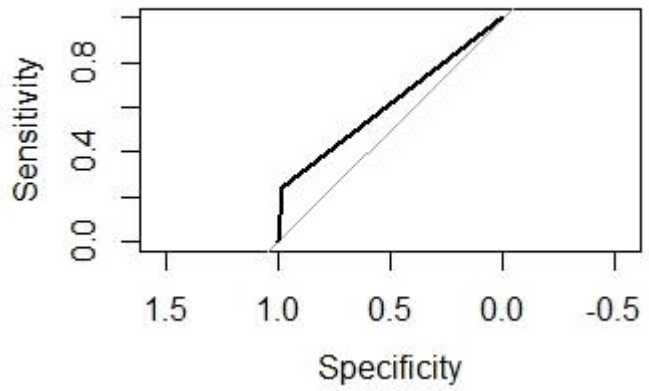
(f) Distribution of Skewness per column in Train and Test dataset

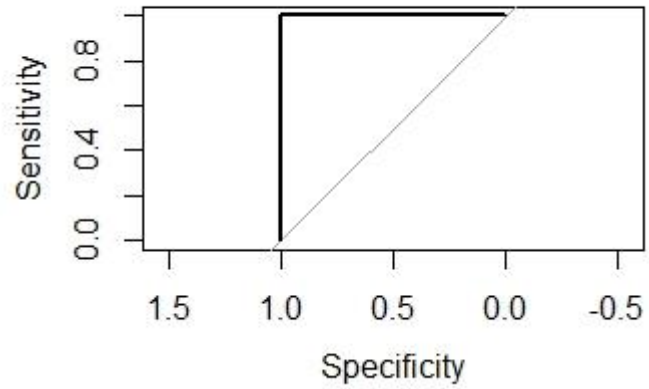


(g) Misclassification error Vs log(lambda) in Logistic Regression and ROSE respectively



(h) ROC_AUC curve in Logistic Regression and ROSE respectively





Reference

- https://edwisor.com/career-data-scientist /
- https://stackoverflow.com/
- http://www.sthda.com/