Santander Customer Transaction Prediction

Sanket Mote

30 January 2020

Table of Contents

Chapter 1

Introduction

1.1 Problem Statement

At Santander, mission is to help people and businesses prosper. We have anonymous dataset wherein we have a ID codes, target variable and various anonymous variables. Base on the dependent variables we need to generate the target variable for our test dataset. In general, based on various variables we need to predict if the customer would transact or not based on the historical data provided.

1.2 Data

Our task is to build a model which will predict the number of transactions customer would do based anonymous variables provided. Given below is a sample data set which we will be using for the model development and prediction of number target variable in the test dataset:

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...	var_190	var_191	var_192	var_193	var_194	var_195	var_196	var
0	train_0	0	8.9255	-6.7863	11.9081	5.0930	11.4607	-9.2834	5.1187	18.6266	...	4.4354	3.9642	3.1364	1.6910	18.5227	-2.3978	7.8784	8
1	train_1	0	11.5006	-4.1473	13.8588	5.3890	12.3622	7.0433	5.6208	16.5338	...	7.6421	7.7214	2.5837	10.9516	15.4305	2.0339	8.1267	8
2	train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	14.6155	...	2.9057	9.7905	1.6704	1.6858	21.6042	3.1417	-6.5213	8
3	train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	14.9250	...	4.4666	4.7433	0.7178	1.4214	23.0347	-1.2706	-2.9275	10
4	train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	19.2514	...	-1.4905	9.5214	-0.1508	9.1942	13.2876	-1.5121	3.9267	9

Table 1.1: Training Data (Observations: 1-5)

As you can see in the data, we have 1 categorical variables (target), 200 continuous variables (var_0 to var_199) and around 2 lakh observations in the training data.

	ID_code	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	var_8	...	var_190	var_191	var_192	var_193	var_194	var_195	var_196	'
0	test_0	11.0656	7.7798	12.9536	9.4292	11.4327	-2.3805	5.8493	18.2675	2.1337	...	-2.1556	11.8495	-1.4300	2.4508	13.7112	2.4669	4.3654	
1	test_1	8.5304	1.2543	11.3047	5.1858	9.1974	-4.0117	6.0196	18.6316	-4.4131	...	10.6165	8.8349	0.9403	10.1282	15.5765	0.4773	-1.4852	
2	test_2	5.4827	-10.3581	10.1407	7.0479	10.2628	9.8052	4.8950	20.2537	1.5233	...	-0.7484	10.9935	1.9803	2.1800	12.9813	2.1281	-7.1086	
3	test_3	8.5374	-1.3222	12.0220	6.5749	8.8458	3.1744	4.9397	20.5660	3.3755	...	9.5702	9.0766	1.6580	3.5813	15.1874	3.1656	3.9567	
4	test_4	11.7058	-0.1327	14.1295	7.7506	9.1035	-8.5848	6.8595	10.6048	2.9890	...	4.2259	9.1723	1.2835	3.3778	19.5542	-0.2860	-5.1612	

Table 1.2: Test Data (Observations: 1-5)

As you can see in the data, we have 200 continuous variables (var_0 to var_199) and around 2 lakh observations in the training data. Based on the continuous variables our task is to predict target variable using specific machine learning models.

Chapter 2

Methodology

2.1 Pre Processing

Before starting predictive modelling, we need to check the data and understand various data definitions. Also, we need to check whether data is in proper format before developing model and feeding the data, if data is not in proper format, we need to clean the data in order to increase the efficiency of the model. To start the process, we have basic steps which needs to be completed such as data cleaning, visualization using graph or plots, standardizing if data is not uniformly distributed these steps are a part of Exploratory Data Analysis.

In fig. 2.1 we have plotted a bar graph wherein we analyze distribution of our target variable, which will help understand if data is uniformly distributed or not.

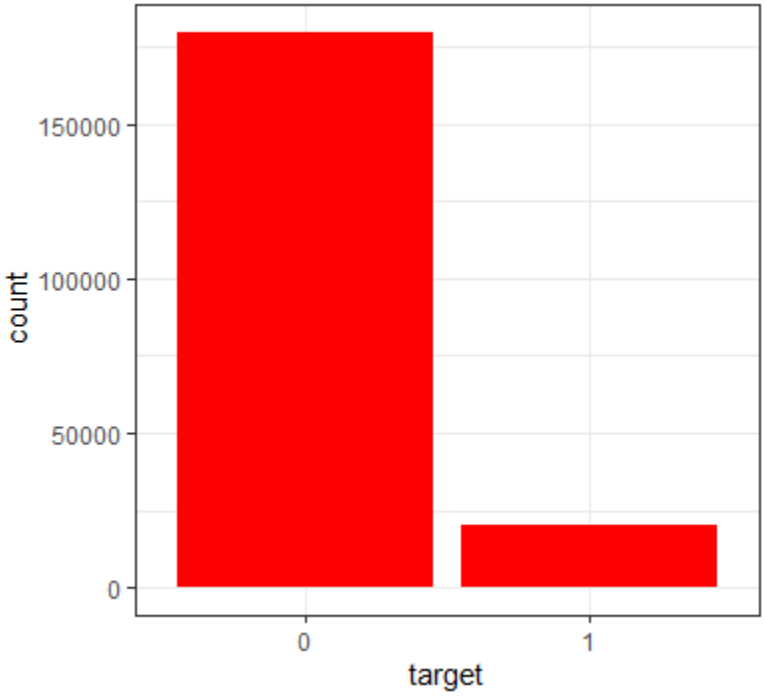


Fig. 2.1 Data Distribution of Dependent Variable

2.1.1 Missing Value Analysis

We check whether dataset contains any missing value. If it contains missing value, we need to impute missing observations in a variable if it has at least 70% of data. We impute the missing data with the help of mean, median in case of continuous variable and mode in case of categorical variable. We can also use K Nearest Neighbor (KNN) in order to impute missing values. In our dataset we do not have any missing value therefore we can proceed to the next step.

2.1.2 Outlier Analysis

Since data is anonymous, we will not check for outliers since data is in converted form and not in original format. We will proceed and start with the model development.

2.2 Modelling

Once we clean the data in the data exploration phase, we can proceed with the model development. After data exploration initial phase, we came to know few points about our dataset such as categorical variables and continuous variables, dependent and independent variables. We also know our target or dependent variable is continuous since we need to predict the number of daily bikes rented. Therefore, our problem is a regression problem. We will be using Decision tree regression, Random forest and Linear Regression algorithm.

2.2.1 Logistic Regression Algorithm

Logistic regression algorithm can only be used when the target or dependent variable is categorical and can be only used in classification model. Inputs can be continuous as well as categorical. Possible outcomes can be both class, probabilities based on the inputs provided. Classes can be binomial, ordinal or multinomial. There are few assumptions which needs to be taken care before considering logistic regression model such as ratio of cases to variable using discrete variables requires that there are enough responses in every given category, multicollinearity is not present amongst variables, no outliers present and Independence of errors. Below is the code snippet of implementing Logistic Regression Model:

Logistic Regression

```
#Building Logistic Regression after seecting features whose p vlaue is less than 0.05
logit = sm.Logit(training_set['target'], training_set.iloc[:,1:201]).fit()
```

```
Optimization terminated successfully.
         Current function value: 0.230069
         Iterations 8
```

Fig. 2.2.1 Logistic Regression Model Code

Post our model is trained we can proceed for predictions on test data and after that we can use various metrics such as precision, recall and area under curve (AUC) to access the model predicted values and to know how accurate results predicted by our model are. Below is the summary of our logistic regression model wherein for every variable we can see the correlation, standard deviation, z value and probability (p) value. Base on this we can select variables which are contributing more to predict target variable. We will consider all variables whose p value is greater than 0.05 and re generate or train our model.

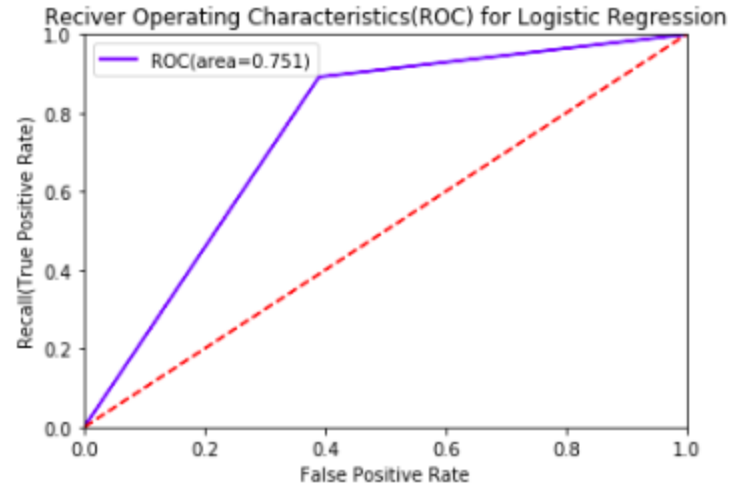
```
print(logit.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                 target   No. Observations:               159942
Model:                          Logit   Df Residuals:                   159742
Method:                           MLE   Df Model:                          199
Date:                Sun, 26 Jan 2020   Pseudo R-squ.:                 0.2932
Time:                        18:41:20   Log-Likelihood:                -36798.
converged:                       True   LL-Null:                       -52062.
Covariance Type:            nonrobust   LLR p-value:                    0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
var_0          0.0568      0.003     18.144      0.000       0.051       0.063
var_1          0.0403      0.002     16.853      0.000       0.036       0.045
var_2          0.0674      0.004     18.698      0.000       0.060       0.074
var_3          0.0189      0.005      3.977      0.000       0.010       0.028
var_4          0.0246      0.006      4.125      0.000       0.013       0.036
var_5          0.0136      0.001     11.033      0.000       0.011       0.016
var_6          0.2655      0.011     23.959      0.000       0.244       0.287
var_7         -0.0020      0.003     -0.697      0.486      -0.008       0.004
```

Fig. 2.2.2 Summary of Logistic Model

Once we predict the target values using our model, we can use measurement metrics to measure the accuracy of our model. In our case we will be using Precision, Recall and Area Under Curve (AUC) since our target variable is not equally distributed. If we use general measures such as accuracy there are high chances that we get incorrect results since 90% of our dependent variable data is negative and 10% data is positive which might result in predictions to be biased towards positive results. Below are the scores on various measurements of our Logistic Model:

Precision Score: 0.21
Recall Score: 0.89
AUC: 0.75
Accuracy: 64.37




```
AUC: 0.7509796491926033
```

Fig. 2.2.3 AUC of Logistic Model

2.2.2 Random Forest Algorithm

Random Forest Algorithm uses the CART decision tree algorithm to generate the random forest. It is an ensemble which consists of many decision trees. Random forest uses the mean for regression problems, and it can be used for both classification and regression problems. Below is the Random forest model implementation python code. We can use the RandomForestClassifier function defined in the sklearn.ensemble library. In the function n_estimators is the number of trees to be used in the forest. A sub optimal greedy algorithm is repeated several times using random selections of features and samples. The random_state parameter allows controlling these random choices. Followed with these we need to provide the independent and dependent variable in the fit function to train the model.

Random Forest Classifier Algorithm

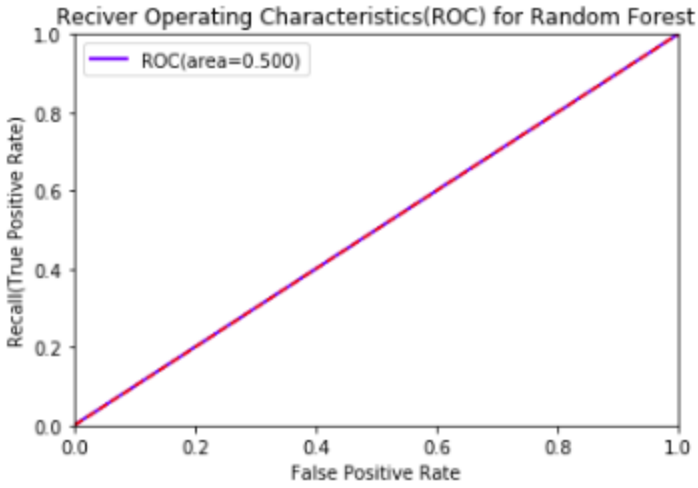
```
# Building Random Forest Model
RF_model = RandomForestClassifier(n_estimators = 200).fit(training_set_v2.iloc[:,1:173], training_set_v2['target'])
```

```
RF_Predictions = RF_model.predict(test_set_v2.iloc[:,1:173])
```

Fig. 2.2.2 Random Forest Model Code

Once the model is trained on train data we check the summary of the model using summary() function. We can test the model by predict() and passing independent continuous variables. Below are the scores on various measurements of our Logistic Model:

Precision Score: 1
Recall Score: 0
AUC: 0.50
Accuracy: 89.94




```
AUC_RF: 0.5002466699555994
```

Fig. 2.2.4 AUC of Random Forest Model

From the above results we conclude that Random forest algorithm does not works on our dataset since our data has 90% of negative data our model is biased and predicting all values as negative which is incorrect prediction.

2.2.3 Naive Bayes Algorithm

Naïve Bayes is one of the practical learning methods and works on probabilistic classification. It works on the baye's theorem of probability to predict the class of unknown dataset. It can accept continuous as well as categorical variables as input. Below is the implementation code of Naïve Bayes model on our dataset.

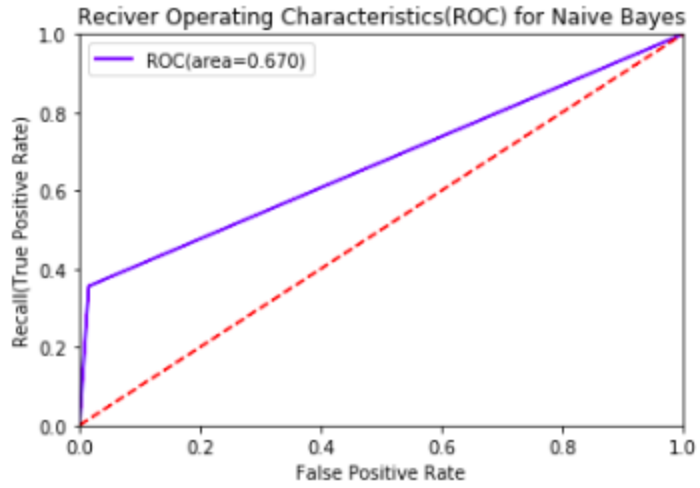
Naive Bayes Algorithm ¶

```
#Running Naive Bayes Algorithm
NB_model = GaussianNB().fit(training_set_v2.iloc[:,1:173],training_set_v2['target'])
```

Fig. 2.2.2 Naïve Bayes Model Code

After training the model we can proceed with the prediction of the data and later we can measure the accuracy of the predictions of our model. Below are the scores on various measurements and AUC of our Naive Bayes Model:

Precision Score: 0.72
Recall Score: 0.36
AUC: 0.67
Accuracy: 92.11



AUC: 0.670208129729154

Fig. 2.2.5 AUC of Naïve Bayes Model

From above results we can say that Naïve Bayes model is performing better as compared to other models on our dataset.

Chapter 3

Conclusion

3.1 Model Evaluation

After generating predictions from few models as above we need to finalize and select the best performing and efficient model. Mostly models are compared based on below criteria:

- Predictive Performance
- Interpretability
- Computational Efficiency

In our case of Santander transaction prediction data since we have predicted the target variable, we will judge the models based on their predictive performance. Predictive performance of a classification model can be measured using Confusion Matrix, Precision, Recall, Receiver operating characteristics (ROC), Area under curve (AUC).

3.1.1 Confusion Matrix

Confusion matrix is majorly used to see the results predicted as compared to actual results in a matrix form. In our case we had 2 classes in our dataset so we will be getting a 2*2 matrix wherein the actual classes will be vertical and predicted classes will be horizontal. We can use crosstab() which is defined in pandas library function in python and table() function in R. We just have to pass the actual target values and predicted values in crosstab or table function respectively to get the confusion matrix.

3.1.2 Precision

Precision is used when the data in dependent variable is not having equal distribution. Precision means the percentage of results which are relevant. We can calculate precision as

Precision = True Positive / Actual Results or

Precision = True Positive / True Positive + False Positive

3.1.3 Recall

Recall refers to the percentage of total relevant results correctly classified by our algorithm.

Recall = True Positive / Predicted results or

Recall = True Positive / True Positive + False Negative

3.1.4 Receiver Operating curve (ROC), Area Under Curve (AUC)

AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represent degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.

The ROC curve is plotted with TPR against the FPR where TPR is on y-axis and FPR is on the x-axis.

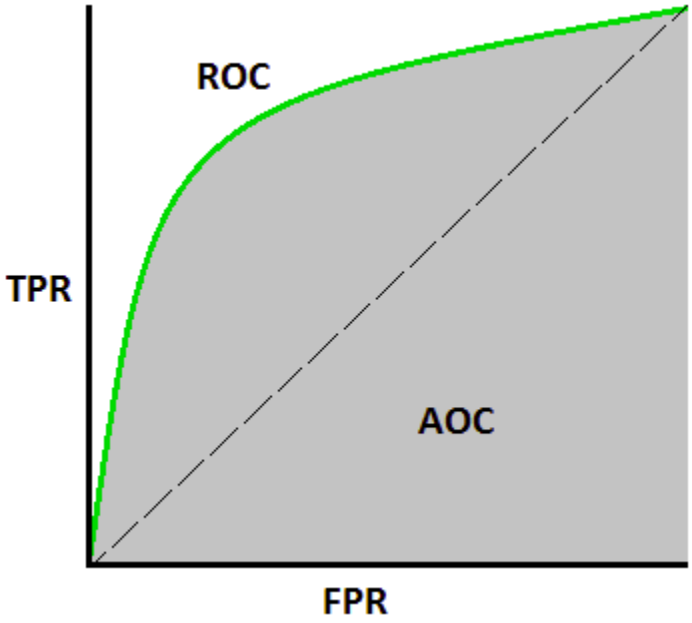


Fig. 3.1.1. ROC – AUC Curve (Image Courtesy: My Photoshopped Collection)

3.1 Model Selection

After running various model, we evaluated above performance metrics on our data set. As we can see in the code implementation, below is the table of machine learning algorithm implemented with their accuracy, precision, recall and AUC values. Percentages will be different in both Python and R language because we made a data split into train and test data randomly.

	Python					R				
	Accuracy	Precision	Recall	F1 Score	AUC	Accuracy	Precision	Recall	F1 Score	AUC
Naïve Bayes	92.11	0.72	0.36	0.48	0.67	92.18	0.04	0.36	0.070	0.67
Random Forest	89.94	1	0	0	0.5	90.03	0.003	0.03	0.005	0.51
Logistic Regression	64.37	0.21	0.89	0.33	0.75	63.76	0.13	0.88	0.226	0.75

Fig 3.3 Output Values Comparison

It is clearly visible that **Naïve Bayes Model** is the best fit to predict the target variable in our data set since it has high precision rate and accuracy.

References

Edwisor study materials

Machine Learning (in Python & R) dummies by John Paul Mueller & Luca Massaron

R and Python official documents

https://towardsdatascience.com/precision-vs-recall-386cf9f89488https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5