Final Project Report Data Mining Method I

By

Swati Lathwal

For

Dr. Morgan Wang University of Central Florida 17th April 2020

1. Contents

2. ExecutiveSummary

In this project four models were used in order to predict the target variable: Logistic Regression, Logistic Regression with interactions, Random Forest and Gradient Boosting. After building the following models we concluded that Gradient Boosting is the best model. Accuracy score of the model is 82.96% and precision came out to be 73.24%, Moreover, it requires less computational time.

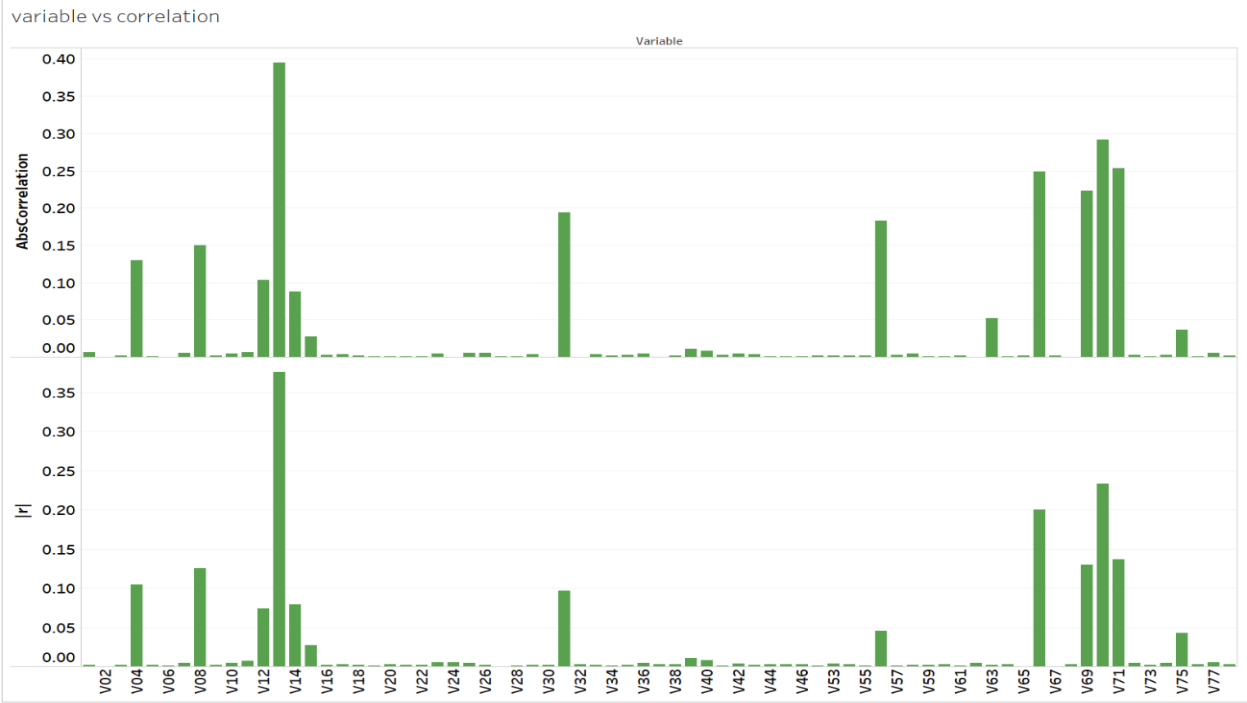
For feature selection, one of the wrapper method was used i.e. Backward Elimination. At first we feed all the features into the model. Simultaneously, we test the model's output then iteratively delete the worst output features until the model's overall performance falls in reasonable range. The performance metric used to measure feature performance is p-value. If the p-value is above 0.05 then we will delete the functionality, otherwise we will retain it. Seventeen features were selected and used to build the model. Set of important features obtained after using feature selection is ['feat4', 'feat8', 'feat12', 'feat13', 'feat14', 'feat15', 'feat20', 'feat31', 'feat40', 'feat42', 'feat56', 'feat63', 'feat66', 'feat69', 'feat70', 'feat71', 'feat75'].

In order to see the impact of these individual variables on the target variable Pearson coefficient was calculated. Below is the feature-target correlation:

{'feat4_target': 0.1298520511787094, 'feat8_target': 0.14964452160512282, 'feat12_target': 0.10361024915320233, 'feat13_target': 0.3953287852358873, 'feat14_target':0.08804329851562159, 'feat15_target': 0.027401238023139336, 'feat20_target': 0.31847911293627545, 'feat31_target':-0.1935955325995681, 'feat40_target': -0.008134550485959069, 'feat42_target': 0.004515636149982063, 'feat56_target': -0.18301225569208934, 'feat63_target':-0.05199757746849357,

'feat66_target': 0.24850812122367416, 'feat69_target': 0.22301965452813405, 'feat70_target': 0.29161618354594065, 'feat71_target': 0.25305347682777857, 'feat75_target': -0.03645484026901819}

Here, we can clearly see which features are highly co-related and is likely to have higher impact on target variable.



Coefficient between 0.5 to 1.0 and -0.5 to -1.0 indicates that there is a strong association between feature and the target variable. We are considering the above set of features as their coefficient is comparatively more with respect to other features i.e. they have a greater impact on the target variable.

3. Introduction

This project is about building predictive models using tools like “Logistic Regression” (with and without interactions), “Gradient Boosting” and “Random Forest” and comparing the performance of each with one another. The dataset PHY_TRAIN is used from the project that came from 2004 KDD CUP competition which is the perfect for building predictive models. The project includes additional excel file “Variables” to understand the data variables. The software jupyter notebook in python has been used to perform the operations like:

- Understanding the data: The procedure of exploratory data analysis was performed to understand the dimensions, columns, identifying missing values and data types in the data file.

- Data cleaning: The identified missing values were smoothed and normalized in order to make the data ready to be fitted in the predictive modelling.

- Building Predictive Models: For model building sklearn library was used. The predictive models like logistic regression, gradient boosting and random forest

were used to understand the correlation of “target” variable with the predictors set. We split the data into training set and validation set with proportion of 70% and 30% respectively. For logistic regression without interactions and gradient boosting we used the library to fit the model. In case of logistic regression with interactions we fitted the training dataset with five interaction columns and for random forest there was no need for us to do a lot of data processing as it is not sensitive to missing data.

- • Model comparison: The performance of fitted models were compared with the use of metrics such as accuracy, precision, confusion matrix, AUC and plotted ROC curve.

Objectives of the study:

- • Explore the data
- • Fit the data into predictive modelling
- • Identify which model suits the best for this type of dataset.

4. Methodology

1. Datapreparation:

   In order to prepare the data to fit in the model, the data was analyzed thoroughly, data type, number of columns and dimensions of dataset were identified. Missing columns were pointed out. For this dataset following are the columns with missing values :'feat20', 'feat21', 'feat22', 'feat29', 'feat44', 'feat45', 'feat46' and 'feat55'. The missing data was identified with the use of missing indicator. In the later part, the resulting data was transformed with the use of mean in place of missing values. Recognizing the missing values and fixing them has been a very integral part of the procedure, as only the normalized data is ready for fit in the predictive modeling.

2. Logistic regression formula without interaction terms:
   $Y = \beta 0 + \beta 1$ Feat4 + $\beta 2$ Feat 8 + $\beta 3$Feat12 + $\beta 4$ Feat13 + $\beta 5$ Feat14 +

   $\beta 6$ Feat15+ $\beta 7$ Feat20 + $\beta 8$ Feat31 + $\beta 9$ Feat40 + $\beta 10$ Feat 42 + $\beta 11$ Feat56 + $\beta 12$ Feat63 + $\beta 13$ Feat 66 + $\beta 14$ Feat 69+ $\beta 15$ Feat 70+ $\beta 16$ Feat 71+ $\beta 17$ Feat 75

3. Logistic regression formula with interaction terms:
   $Y = \beta 0 + \beta 1$ Feat4 + $\beta 2$ Feat 8 + $\beta 3$Feat12 + $\beta 4$ Feat13 + $\beta 5$ Feat14 +

β6 Feat15+ β7 Feat20 + β8 Feat31 + β9 Feat40 + β10 Feat 42 + β11 Feat56 + β12 Feat63 + β13 Feat 66 + β14 Feat 69+ β15 Feat 70+

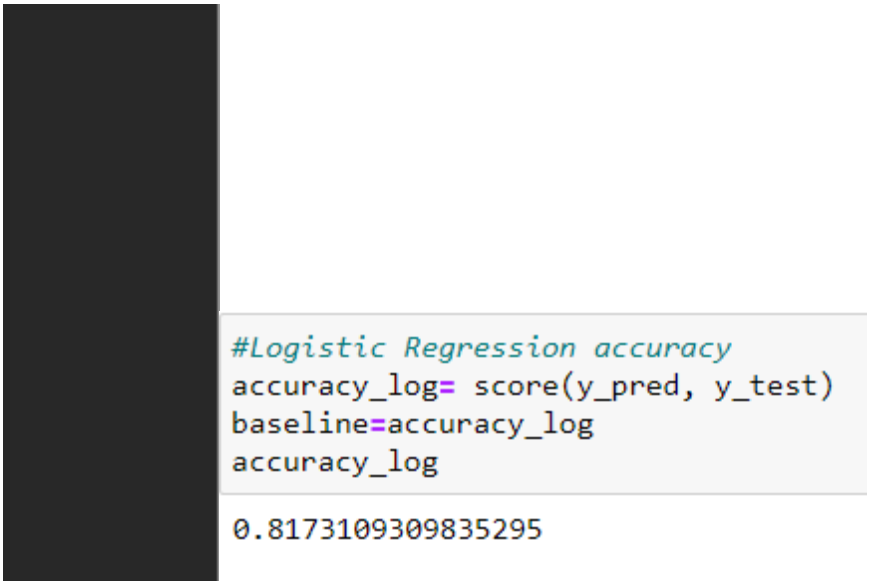
7

β16 Feat 71+ β17 Feat 75+ β18 (Feat71 *
Feat31) + β17 (Feat8 * Feat71) + β18 (Feat75*Feat15) + β19 (Feat69*Feat56) + β20 (Feat71* Feat66)

8

5. Modelling Performance Measures

a. Logistic Regression without Interaction Terms Accuracy Rate:



```
#Logistic Regression accuracy
accuracy_log= score(y_pred, y_test)
baseline=accuracy_log
accuracy_log
```

```
0.8173109309835295
```

Precision:

Confusion Matrix

True Positive Rate: TP/ (TP+FN) = 0.712 False Positive Rate: FP/ (FP+TN) = 0.292

b. Logistic Regression with Interaction Terms Accuracy Rate:

Precision

```
precision
```

```
0.7078757031877846
```

```
con_matrix_lg
```

```
array([[5362, 2181],
       [2172, 5285]], dtype=int64)
```

```
#Logistic regression with Interaction accuracy
inter_acc= score(inter_pred, y_test)
inter_acc
```

```
0.8172589053940887
```

```
#Logistic regression with Interaction precision
precision_inter
```

```
0.7078636607622115
```

Confusion Matrix

True Positive Rate: TP/ (TP+FN) = 0.711 False Positive Rate: FP/ (FP+TN) = 0.292

c. Random Forest Model Accuracy Rate:

Precision:

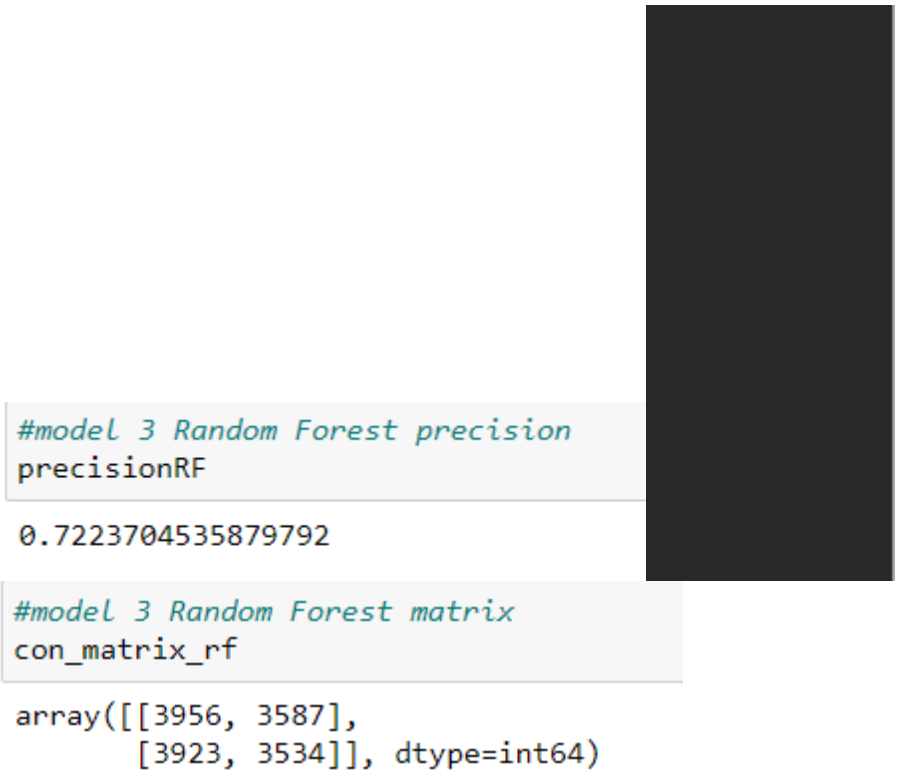
Confusion Matrix:

```
con_matrix_in
```

```
array([[5366, 2177],
       [2182, 5275]], dtype=int64)
```

```
#model 3 Random Forest accuracy
score(pred,y_testT)
```

```
0.8215077993956539
```



```
#model 3 Random Forest precision
precisionRF
```

```
0.7223704535879792
```

```
#model 3 Random Forest matrix
con_matrix_rf
```

```
array([[3956, 3587],
       [3923, 3534]], dtype=int64)
```

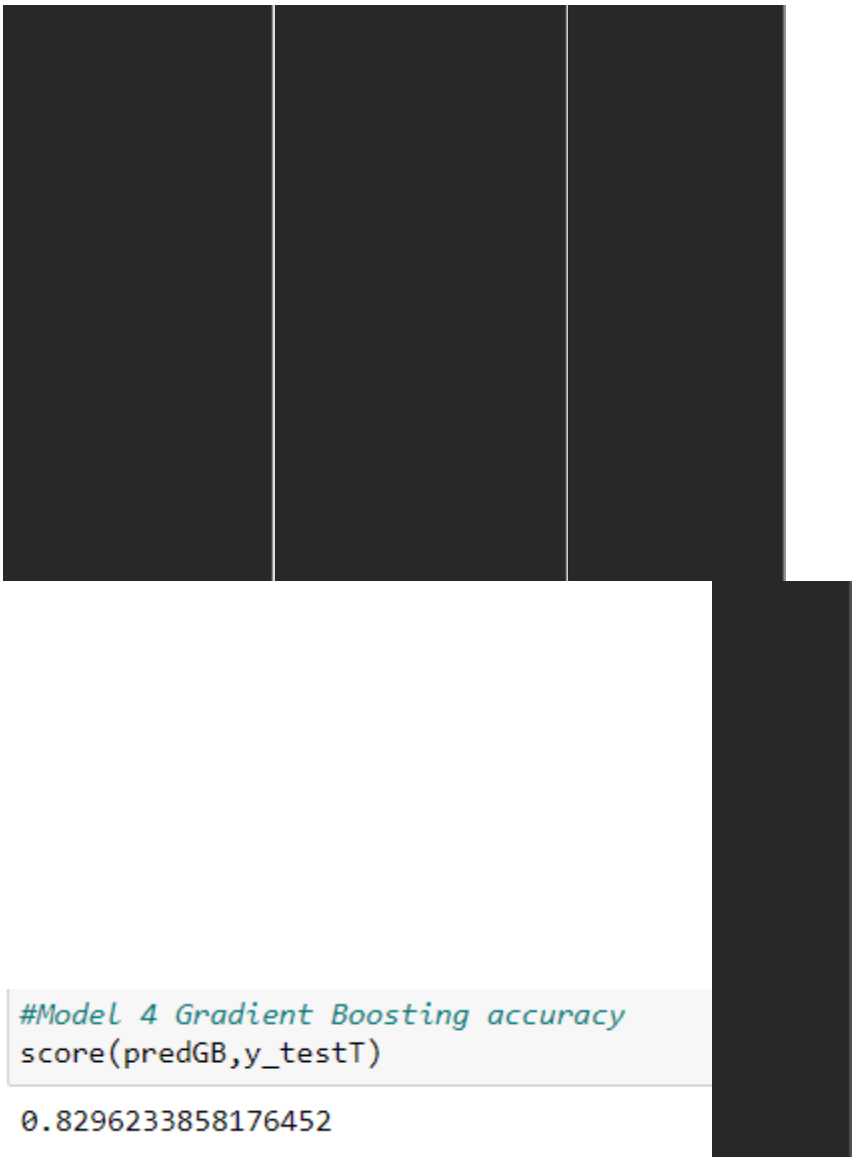

True Positive Rate: TP/ (TP+FN) = 0.502 False Positive Rate: FP/ (FP+TN) = 0.504

d. GradientBoostingModel Accuracy Rate:

Precision:

Confusion Matrix:

True Positive Rate: TP/ (TP+FN) = 0.501 False Positive Rate: FP/ (FP+TN) = 0.506



```
#Model 4 Gradient Boosting accuracy
score(predGB,y_testT)
```

```
0.8296233858176452
```

```
#Model 4 Gradient Boosting precision
precisionGB
```

```
0.732420220276758
```

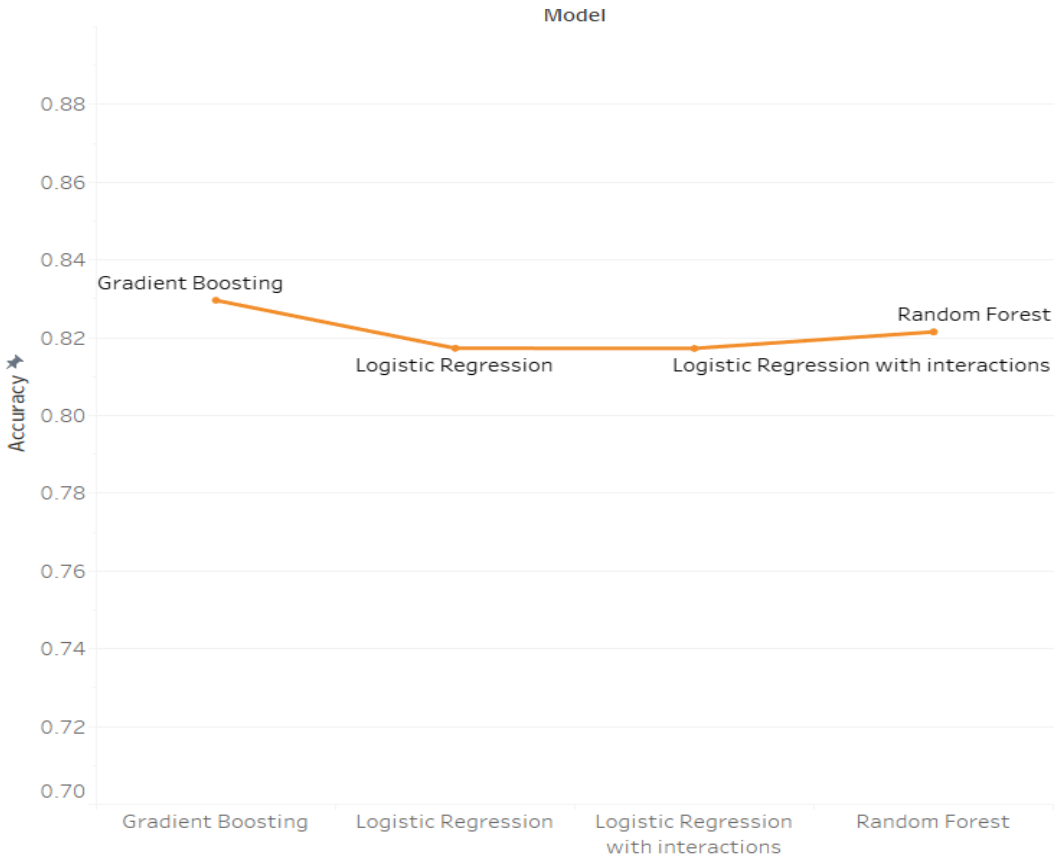
```
#Model 4 Gradient Boosting matrix
con_matrix_gb
```

```
array([[3961, 3582],
       [3957, 3500]], dtype=int64)
```


6. The model selected and the performance measure(s) used

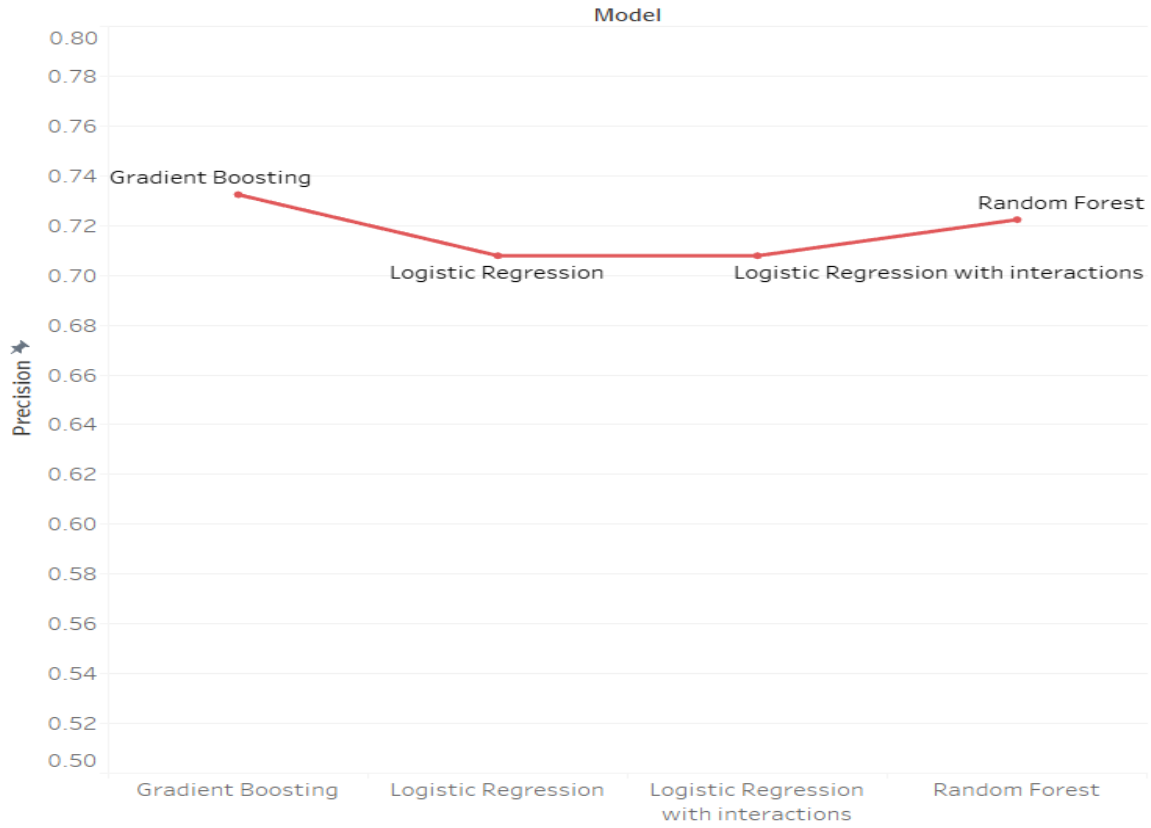
In order to decide which model is the best out of four, accuracy and precision were used as performance measure. Accuracy is the most intuitive indicator of accuracy, which is essentially a ratio of correctly predicted observation to total observations. Whereas, precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision means low false positive rate.

a. Accuracyvs.Model



The above plot compares the accuracy of different models and it can be observed that the gradient boosting has the greater value i.e. 0.83.

b. Precisionvs.Model



In this plot, we can observe that the precision of gradient boosting is more with respect to others.

In both the cases, Gradient Boosting stands out to be the best, thus we choose this model.

7. Conclusions

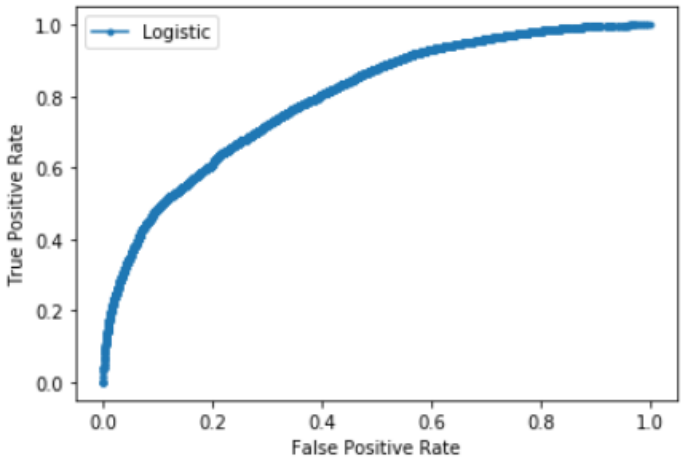
At the end, we can say that the after building all four models and comparing their performances, Gradient Boosting is the best suitable model for predicting target variable. With the accuracy of 82.96%, it is the best model for classification problem. Logistic regression model gave the accuracy of 81.73% whereas, logistic regression model with interactions gave the accuracy of 81.72%. Accuracy of both the models were approximately same. Then, random forest model was built and it gave the accuracy of 82.15%, which is comparatively more than the above two model.

Gradient boosting is said to reduce the error mainly by reducing bias and requires less computation, the model was built to see whether it works well with the particular problem. Accuracy of 82.96% was observed, the most accurate model by far.

Thus, we concluded that gradient boosting is the best model in this case.

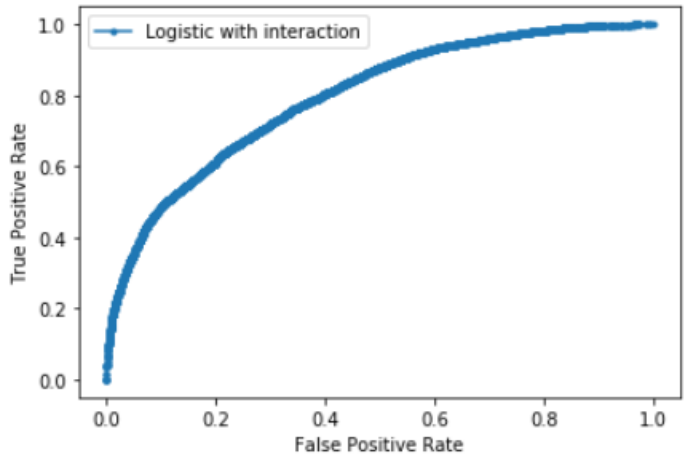
1. ROC Curve for Logistic Regression:
2. ROC Curve for Logistic Regression with interactions:

8. Appendix




```
print('Logistic: ROC Area Under the Curve=%.3f' % (lr_auc))
```

```
Logistic: ROC Area Under the Curve=0.797
```

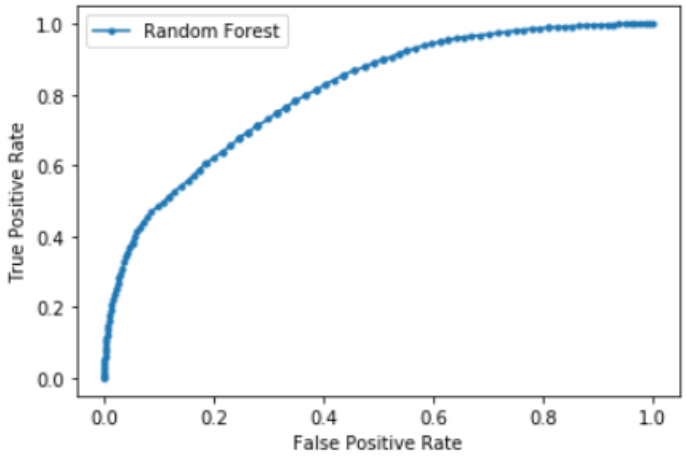



```
in_auc = roc_auc_score(y_test, y_pred)

print('Logistic: ROC Area Under the Curve=%.3f' % (in_auc))
```

```
Logistic: ROC Area Under the Curve=0.711
```

c. ROC Curve for Random Forest:

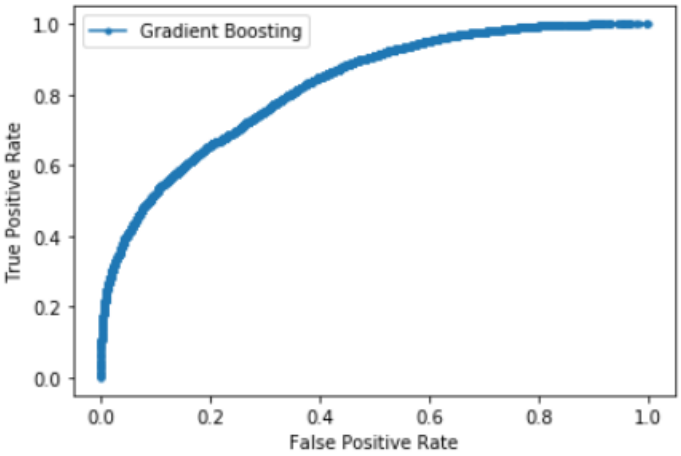



```
RF_auc = roc_auc_score(y_testT,pred)
```

```
print('Random Forest: ROC Area Under the Curve=%.3f' % (RF_auc))
```

```
Random Forest: ROC Area Under the Curve=0.808
```

d. ROC Curve for Gradient Boosting:




```
GB_auc = roc_auc_score(y_testT,predGB)
print('GB: ROC Area Under the Curve=%.3f' % (GB_auc))
```

```
GB: ROC Area Under the Curve=0.819
```


e. SummaryTable:

Features	exampleid	target	feat1	feat2	feat3	feat4	feat5	feat6	feat7	feat8	feat9	feat10	feat11	feat12	feat13	feat14	feat15	feat16	feat17	feat18	feat19	feat20
Count	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	1579
Mean	25000.5	0.49722	0.155606	0.084876	-0.05035	-6.00E-05	0.126569	0.049887	-0.03834	0.00286	0.848353	0.673485	-0.28392	-0.01092	0.00726	0.000856	-0.00082	0.85558	0.16806	0.076981	1.121156	0.00111
SD	14433.9011	0.499997	0.414875	0.295335	0.253748	0.392916	0.400694	0.223713	0.214168	0.322077	0.453585	0.511087	0.591971	0.993549	0.649573	0.303457	0.119658	1.068581	0.410681	0.180022	0.440878	1.05116
MIN	1	0	0	0	0	-1	-1	0	0	-1	-1	0	0	-1	-1	-1	-1	-0.99987	0	0	0	0	-2.4537
25%	12500.75	0	0	0	0	0	0	0	0	0	0	0.522596	0.250301	-0.80789	-1	-0.28127	-0.00056	-3.03E-05	0	0	0	0.803207	-0.8149
50%	25000.5	0	0	0	0	0	0	0	0	0	0	0.787572	0.599672	-0.45178	0	1.59E-06	0	0	1	0	0	1.068862	0.50421
75%	37500.25	1	0	0	0	0	0	0	0	0	0	1.105687	1.018602	0.171791	1	0.355395	0.000583	2.95E-05	1	0	0	1.387689	0.81191
MAX	50000	1	2.63902	3.42959	0.999954	1	2.719006	3.054644	0.999274	1	6.699783	5.283748	0.999906	1	0.999999	0.999992	0.999796	9	4	0.504665	6.077315	4.50716
Skewness	-2.41E-17	0.011121	2.826244	4.424666	-1.56247	-0.00053	3.245044	5.761064	-2.22607	0.058966	0.86224	0.658286	0.605376	0.021841	-0.01009	0.00912	-0.31962	1.498027	2.464306	1.911028	0.676742	-0.014
Missing %	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	68.40

Features	feat21	feat22	feat23	feat24	feat25	feat26	feat27	feat28	feat29	feat30	feat31	feat32	feat33	feat34	feat35	feat36	feat37	feat38	feat39	feat40	feat41	feat42
Count	15798	15798	50000	50000	50000	50000	50000	50000	19938	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000
Mean	-0.06716	-0.6199	0.928406	0.826006	0.523078	0.404894	0.091401	0.014918	0	0.40062	0.00168	361.6766	4.370206	0.110897	0.02354	0.02306	0.002711	0.004575	-0.00482	0.003195	0.003666	0.008785
SD	0.845419	0.281082	0.532563	0.53015	0.431308	0.391197	0.084919	0.118304	0	0.490029	0.775403	617.7553	5.668861	0.108687	0.151612	0.150096	0.280587	0.289169	0.421314	0.031784	0.037866	0.112061
MIN	-1	-0.99997	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	-6.47017	-10.7719	-17.3215	0	0	0
25%	-0.90264	-0.86633	0.537355	0.467676	0.252724	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0
50%	-0.51484	-0.67705	0.819246	0.711095	0.456446	0.353613	0.094655	3.89E-07	0	0	0	0	3.808594	0.109303	0	0	0	0	0	0	0	0
75%	0.881655	-0.40164	1.195838	1.052268	0.689711	0.81221	0.164785	8.92E-07	0	1	1	625	6.103516	0.189062	0	0	0	0	0	0	0	0
MAX	1	-2.83E-05	4.508337	4.790007	6.4519	0.999999	0.249992	0.999998	0	1	1	10000	75	0.526621	1	1	7.56551	9.437345	14.76335	1.938482	3.548287	6.192523
Skewness	0.162028	0.509192	1.235282	1.407299	1.874689	0.224139	0.219656	7.864358	0	0.405625	-0.0029	5.995289	2.626815	0.545894	6.285488	6.355404	0.268543	1.064332	-1.77095	18.28859	27.95949	24.13063
Missing %	68.404	68.404	0	0	0	0	0	0	60.124	0	0	0	0	0	0	0	0	0	0	0	0	0

Features	feat43	feat44	feat45	feat46	feat47	feat48	feat49	feat50	feat51	feat52	feat53	feat54	feat55	feat56	feat57	feat58	feat59	feat60	feat61	feat62	feat63	feat64
Count	50000	35531	35531	35531	50000	50000	50000	50000	50000	50000	50000	50000	31398	50000	50000	50000	50000	50000	50000	50000	50000	50000
Mean	0.60124	0.42765	0.369191	-0.26815	0	0	0	0	0	3.7204	0.05907	0.012835	0	0.00256	218.7778	2.658876	0.072116	0.353685	0.312237	-0.21596	0.00234	0.010682
SD	0.489648	0.557655	0.431658	0.358799	0	0	0	0	0	4.833538	0.083323	0.10962	0	0.609952	508.2985	4.89722	0.106258	0.522011	0.418076	0.332459	4.107353	0.01228
MIN	0	0	0	-0.99999	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	-0.99995	-22	0
25%	0	0	0	-0.59395	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.43463	0	0.002568
50%	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.007293
75%	1	0.779908	0.875205	0	0	0	0	0	0	10	0.129985	4.77E-07	0	0	463.8672	4.394531	0.144993	0.690444	0.81857	0	0	0.014545
MAX	1	2.87883	0.999997	0	0	0	0	0	0	10	0.249992	0.999998	0	1	10000	75	0.540209	3.043963	0.999997	0	22	0.099991
Skewness	-0.41354	1.099309	0.399998	-0.86273	0	0	0	0	0	0.529488	0.947863	8.485259	0	-0.00131	7.084518	3.333389	1.236501	1.34516	0.672916	-1.18341	-0.04403	2.750394
Missing %	0	28.938	28.938	28.938	0	0	0	0	0	0	0	0	37.204	0	0	0	0	0	0	0	0	0

Features	feat57	feat58	feat59	feat60	feat61	feat62	feat63	feat64	feat65	feat66	feat67	feat68	feat69	feat70	feat71	feat72	feat73	feat74	feat75	feat76	feat77	feat78
Count	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000	50000
Mean	218.7778	2.658876	0.072116	0.353685	0.312237	-0.21596	0.00234	0.010682	0.965762	-0.00364	0.784289	0.159905	0.00812	0.000478	0.003198	0.052807	0.066944	-0.0141	-0.00146	0.09448	0.002843	0.066545
SD	508.2985	4.89722	0.106258	0.522011	0.418076	0.332459	4.107353	0.01228	0.504865	0.95138	0.289596	0.478684	0.769302	0.446978	0.381329	0.18071	0.283114	0.176896	0.295939	0.315841	0.019081	0.223091
MIN	0	0	0	0	0	-0.99995	-22	0	0	-1	0	-0.99999	-1	-1	-0.908	0	0	-1	-1	0	0	0
25%	0	0	0	0	0	-0.43463	0	0.002568	0.6663	-1	0.748446	0	-1	-1.27E-05	-0.00158	0	0	0	0	0	0	0
50%	0	0	0	0	0	0	0	0.007293	0.922424	0	0.904182	0	0	0	0	0	0	0	0	0	0	0
75%	463.8672	4.394531	0.144993	0.690444	0.81857	0	0	0.014545	1.275267	1	0.968595	0.566938	1	1.50E-05	0.003002	0	0	0	0	0	0	0
MAX	10000	75	0.540209	3.043963	0.999997	0	22	0.099991	5.626167	1	1	0.999997	1	0.999999	0.907744	0.999953	3.42959	0.999869	1	3	0.385513	1
Skewness	7.084518	3.333389	1.236501	1.34516	0.672916	-1.18341	-0.04403	2.750394	0.16159	0.007251	-1.8651	-0.09628	-0.01383	0.012134	-0.01065	3.475199	5.025059	-1.41922	-0.04153	3.464927	12.11273	3.223882
Missing %	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0