Course Work-2 Report

Cross Selling Insurance Products

in Partial Fulfillment of the Requirements

for the Degree of

Bachelor of Technology

In

Computer Science and Engineering Department

By

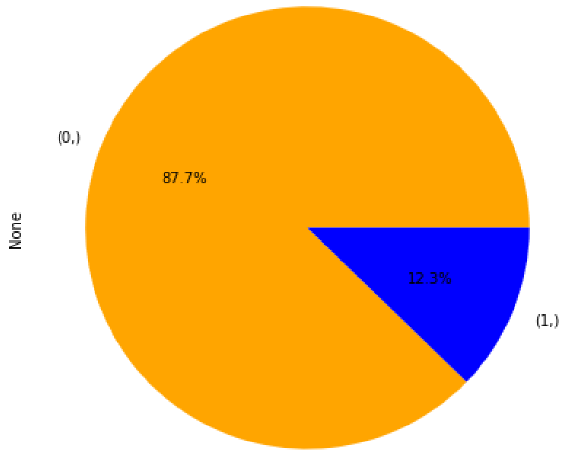
Indradhar Paka
1800315C203
CSE-III
3rd year

SCHOOL OF ENGINEERING AND TECHNOLOGY

BML MUNJAL UNIVERSITY GURGAON

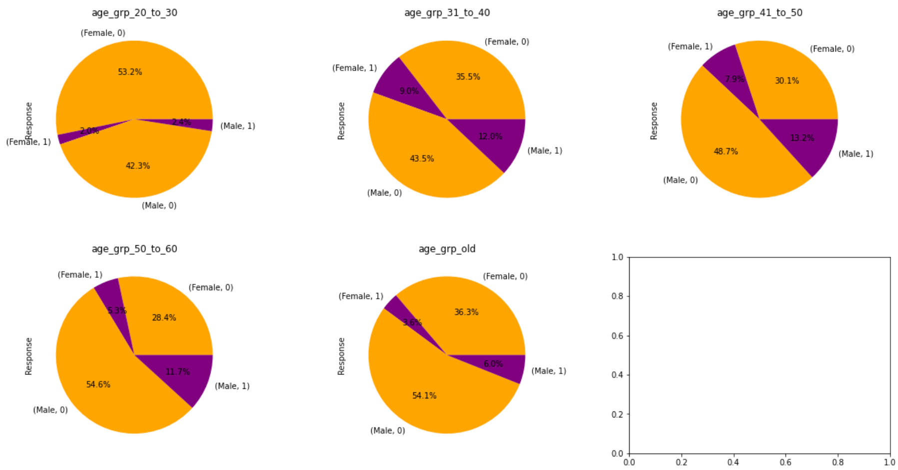
November,2020

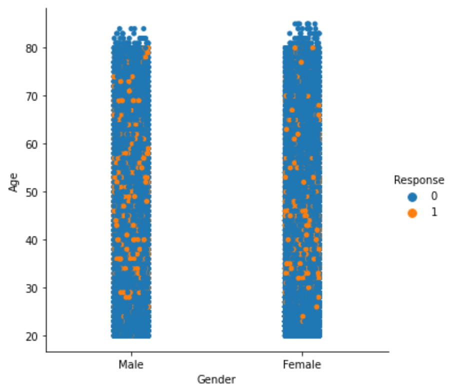
1. **Undertake Exploratory Data Analysis to identify patterns in the data to discover insights that could help you build better models**



Data is Imbalanced. Only 12.3% of customers are likely to buy insurance

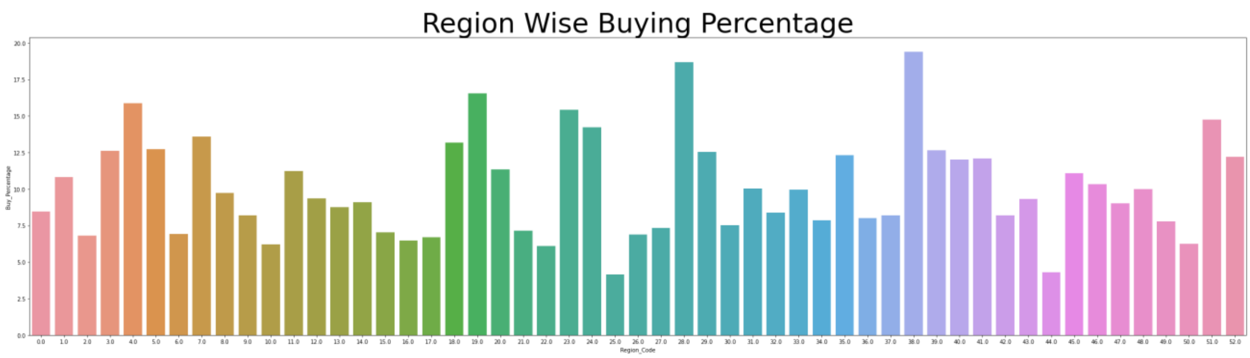
Response Percentage of Different Age Groups with Genders



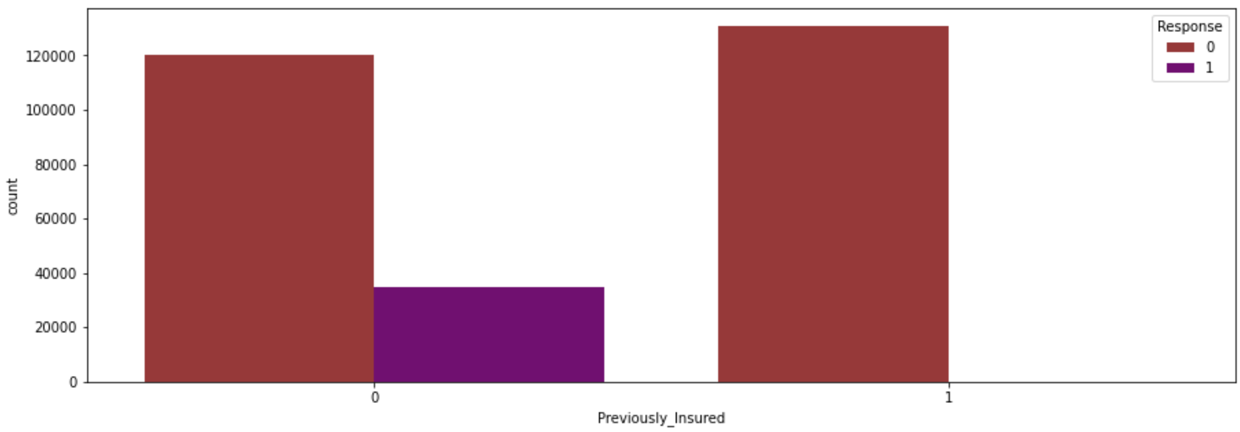


- Customers of age between 30 to 60 are more likely to buy insurance.
- Customes of age between 20 to 30 are less likely to buy insurance.
- In almost every age group, 'Male's are more likely to buy insurance.
- Females under age 30 are very less likely to buy insurance

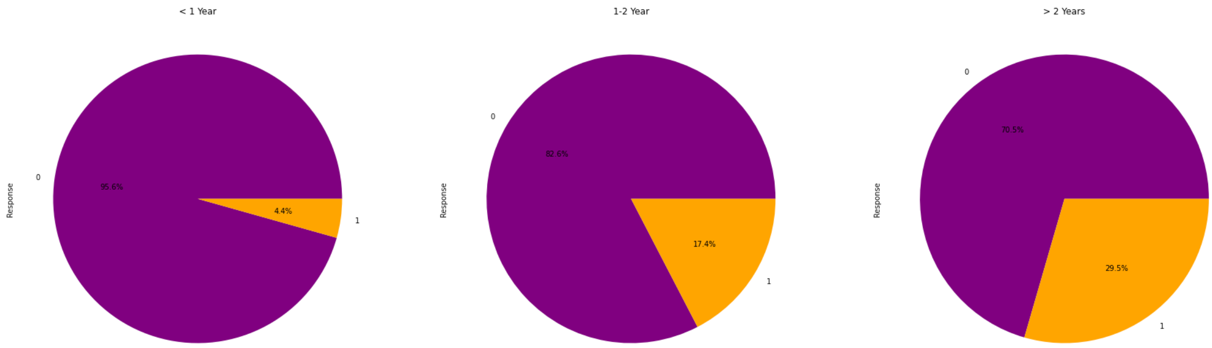
- Very few customers don't have Driving License.
- Customers with Driving License have higher chance of buying Insurance



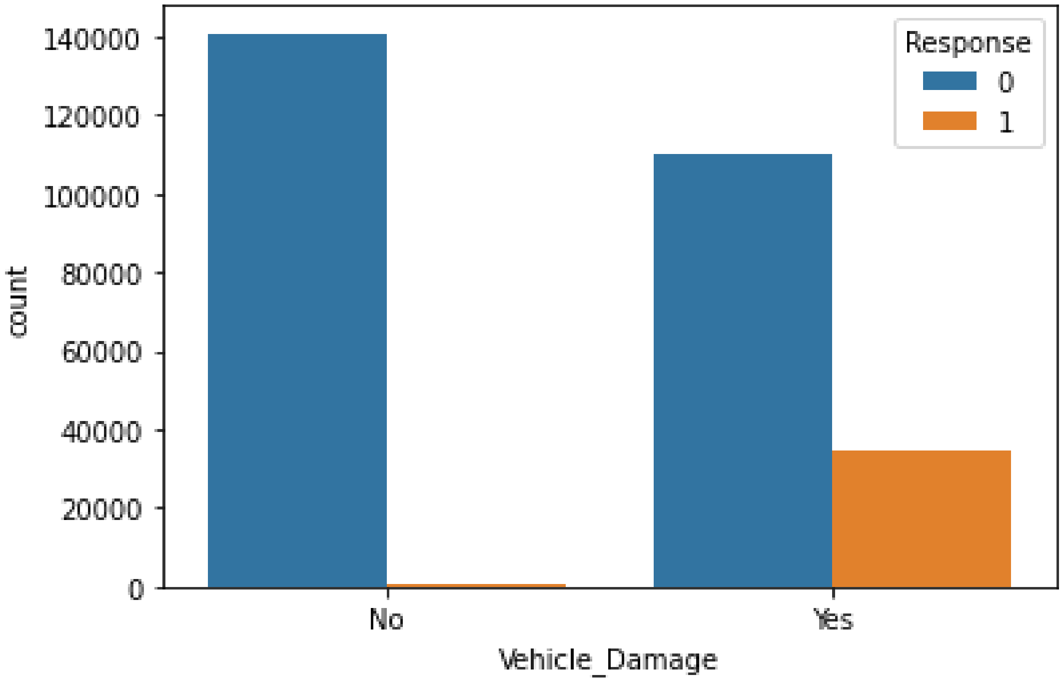
- We have most of the customers from Region_Code : 28.
- Region_Codes: [4,19,23,24,,28,38,51] have higher percentage of buying insurance.
- Region_Codes: 25 and 44 have lower percentage of buying insurance.



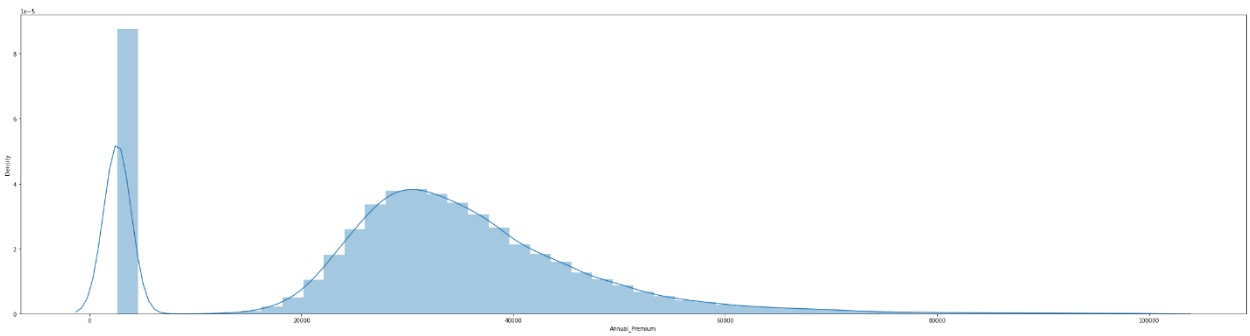
- Customers who Previously_Insured are very likely to buy Insurnce now.
- Customers who didn't Previously_Insured have good chance of buying Insurnce.



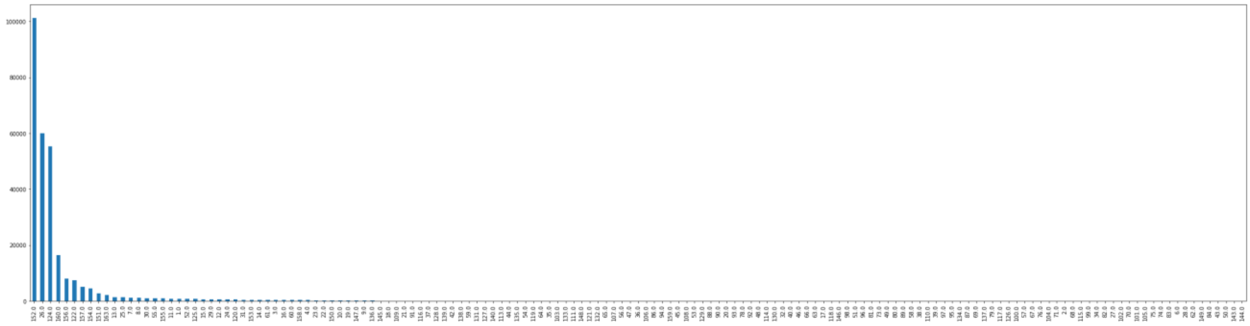
- We have half of our customers with Vehicle_Age 1-2 years.
- We have very few customers (4.2%) with Vehicle_Age `>2 years.
- Customers with Vehicle_Age >2years have better chance (29.4%) of buying Insurance.
- Customers with with Vehicle_Age <1 years have very less chance of buying Insurance.



- We have almost same number of customes with damaged and non_damaged vehicle.
- Customers with Vehicle_Damage are likely to buy insurance.
- Customers with non damaged vehicle have least chance (less than 1%) of buying insurance.



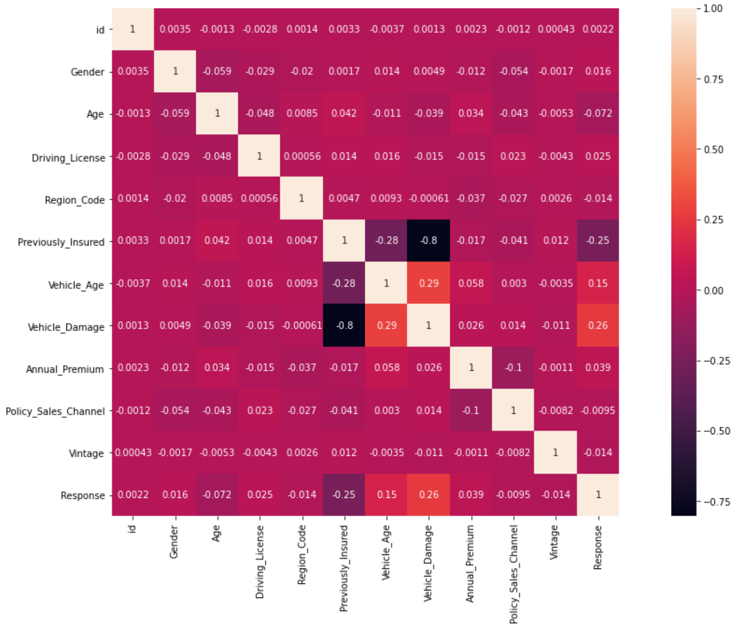
- Annual Premium' data is highlt left skewed.
- Most of the customers have "Annual_Premium' in range (0, 10000) and (20000 to 50000)
- In every 'Annual Premium' range, the insurance buy percentage is almost same.



- Policy_Sales_Channel no. 152 have higest number of customers.
- Policy_Sales_Channel no. [152,26,124,160,156,122,157,154,151,163] have most of the customers.

Understandings:

- Customers of age between 30 to 60 are more likely to buy insurance.
- Customes of age between 20 to 30 are less likely to buy insurance.
- In almost every age group, 'Male's are more likely to buy insurance.
- Females under age 30 are very less likely ho buy insurance.
- Very few customers don't have Driving License.
- Customers with Driving License have higher chance of buying Insurance.
- We have most of the customers from Region_Code : 28.
- Region_Codes: [4,19,23,24,,28,38,51] have higher percentage of buying insurance.
- Region_Codes: 25 and 44 have lower percentage of buying insurance.
- Customers who Previously_Insured are very likely to buy Insurnce now.
- Customers who didn't Previously_Insured have good chance of buying Insurnce.
- We have half of our customers with Vehicle_Age 1-2 years.
- We have very few customers (4.2%) with Vehicle_Age >2 years.
- Customers with Vehicle_Age >2years have better chance (29.4%) of buying Insurance.
- Customers with with Vehicle_Age <1 years have very less chance of buying Insurance.
- We have almost same number of customes with damaged and non_damaged vehicle.
- Customers with Vehicle_Damage are likely to buy insurance.
- Customers with non damaged vehicle have least chance (less than 1%) of buying insurance.
- 'Annual Premium' data is highlt left skewed.
- Most of the customers have "Annual_Premium' in range (0, 10000) and (20000 to 50000)
- In every 'Annual Premium' range, the insurance buy percentage is almost same.
- Policy_Sales_Channel no. 152 have higest number of customers.
- Policy_Sales_Channel no. [152,26,124,160,156,122,157,154,151,163] have most of the customers.
- Every 'Vintage' value have almost same number of customers.



- 'Previously_Insured' and 'Vehicle_Damage' are highly positively corelated.
- 'Age' and 'Policy_Sales_Channel' are negatively corelated.
- 'Age' and 'Vehicle_Age' are negatively corelated.

2. Build models using the standard classification algorithms that you have studied during the course i.e. logistic regression, k-nearest neighbour, naive Bayes, decision trees, support vector machines, random forest and gradient boosted decision trees

Logistic Regression Model

	precision	recall	f1-score	support
0	0.88	1.00	0.94	62835
1	0.00	0.00	0.00	8623
accuracy			0.88	71458
macro avg	0.44	0.50	0.47	71458
weighted avg	0.77	0.88	0.82	71458

Logistic Regression Base Accuracy: 0.879327716980604
Logistic Regression Base ROC_AUC_SCORE: 0.5932222113546538

K-nearest neighbour Model

	precision	recall	f1-score	support
0	0.88	0.97	0.92	62835
1	0.21	0.06	0.09	8623
accuracy			0.86	71458

	precision	recall	f1-score	support
macro avg	0.55	0.51	0.51	71458
weighted avg	0.80	0.86	0.82	71458

KNN Base Accuracy: 0.8606314198550198
KNN Base ROC_AUC_SCORE: 0.5973507473674144

Naïve bayes Model

	precision	recall	f1-score	support
0	0.91	0.88	0.90	62835
1	0.30	0.36	0.33	8623
accuracy			0.82	71458
macro avg	0.60	0.62	0.61	71458
weighted avg	0.84	0.82	0.83	71458

Naive Bayes Base Accuracy : 0.8185507570880797
Naive Bayes Base ROC_AUC_SCORE: 0.8163922368797205

Decision Trees model:

	precision	recall	f1-score	support
0	0.91	0.90	0.90	62835
1	0.30	0.32	0.31	8623
accuracy			0.83	71458
macro avg	0.60	0.61	0.60	71458
weighted avg	0.83	0.83	0.83	71458

Decision Trees Base Accuracy : 0.8259117243695597
Decision Trees Base ROC_AUC_SCORE: 0.6073689699079801

Support Vector Machines

	precision	recall	f1-score	support
0	0.88	1.00	0.94	62835
1	0.00	0.00	0.00	8623
accuracy			0.88	71458
macro avg	0.44	0.50	0.47	71458
weighted avg	0.77	0.88	0.82	71458

Random Forest Model

	precision	recall	f1-score	support
0	0.89	0.97	0.93	62835
1	0.36	0.12	0.17	8623

accuracy			0.87	71458
macro avg	0.62	0.54	0.55	71458
weighted avg	0.82	0.87	0.84	71458

Random Forest Base Accuracy: 0.8679923871364997
Random Forest Base ROC_AUC_SCORE: 0.8332263331560348

Gradient boosted Decision Trees

	precision	recall	f1-score	support
0	0.88	1.00	0.94	62835
1	0.00	0.00	0.00	8623
accuracy			0.88	71458
macro avg	0.44	0.50	0.47	71458
weighted avg	0.77	0.88	0.82	71458

Gradient boosted Base Accuracy : 0.879327716980604
Gradient boosted Base ROC_AUC_SCORE: 0.8316874873189273

3. Noting the skew in the distribution, study methods for addressing the skew using over or under-sampling and SMOTE and apply them to the problem.

Now we are going to check how the models work with the same parameters as baseline will predict using upsampled data with new features.

```
Train set target class count with over-sampling:
0    250798
1     90905
Name: Response, dtype: int64
Validation set target class count:
0    50070
1     7097
```

Logistic Regression

	precision	recall	f1-score	support
0	0.91	0.89	0.90	50070
1	0.31	0.36	0.34	7097
accuracy			0.82	57167
macro avg	0.61	0.63	0.62	57167
weighted avg	0.83	0.82	0.83	57167

Logistic Regression After tuning Accuracy : 0.822222610946875
Logistic Regression After tuning ROC_AUC_SCORE: 0.8087978675704374

k-nearest neighbor

	precision	recall	f1-score	support
0	0.89	0.81	0.85	50070
1	0.18	0.28	0.22	7097
accuracy			0.75	57167
macro avg	0.53	0.55	0.53	57167
weighted avg	0.80	0.75	0.77	57167

KNN After tuning Accuracy : 0.7484387846135008
KNN After tuning ROC_AUC_SCORE: 0.7624193143830004

Naive Bayes Model

	precision	recall	f1-score	support
0	0.90	0.90	0.90	50070
1	0.27	0.26	0.26	7097
accuracy			0.82	57167
macro avg	0.58	0.58	0.58	57167
weighted avg	0.82	0.82	0.82	57167

Naive Bayes After tuning Accuracy : 0.8236919901341683
Naive Bayes After tuning ROC_AUC_SCORE: 0.7058074142164055

Decision Tree Model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	50070
1	1.00	1.00	1.00	7097
accuracy			1.00	57167
macro avg	1.00	1.00	1.00	57167
weighted avg	1.00	1.00	1.00	57167

Decision Trees After Tuning Accuracy : 1.0
Decision Trees After Tuning ROC_AUC_SCORE: 1.0

Support vector Machines

	precision	recall	f1-score	support
0	0.88	1.00	0.93	50070
1	0.00	0.00	0.00	7097
accuracy			0.88	57167
macro avg	0.44	0.50	0.47	57167
weighted avg	0.77	0.88	0.82	57167

Random Forest Model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	50070
1	1.00	1.00	1.00	7097
accuracy			1.00	57167
macro avg	1.00	1.00	1.00	57167
weighted avg	1.00	1.00	1.00	57167

Random Forest After Tuning Accuracy : 1.0
Random Forest After Tuning ROC_AUC_SCORE: 1.0

Gradient Boosted Decision Trees

	precision	recall	f1-score	support
0	0.88	1.00	0.93	50070
1	0.00	0.00	0.00	7097
accuracy			0.88	57167
macro avg	0.44	0.50	0.47	57167
weighted avg	0.77	0.88	0.82	57167

Gradient boosted After Tuning Accuracy : 0.8758549512830829
Gradient boosted Base ROC_AUC_SCORE: 0.8206411657749884

4. Use methods discussed in class for hyperparameter tuning of the models

Tuning Hyper Parameters:

Logistic Regression Model:
Best Parameters: C=100, penalty='l2', random_state=None, solver='newton-cg', tol=0.0001.
Best Estimator: LogisticRegression(C=100, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, l1_ratio=None, max_iter=1000,
multi_class='auto', n_jobs=None, penalty='l2',
random_state=None, solver='liblinear', tol=0.0001, verbose=0,
warm_start=False)
Accuracy Score: 0.879327716980604

Decision Tree Classifier:
max_depth=450, max_features='auto', min_samples_leaf=1, min_samples_split=5

Random Forest:
criterion='gini',max_depth=10, max_features='sqrt', max_leaf_nodes=None,
min_samples_leaf=4, min_samples_split=2,min_weight_fraction_leaf=0.0, n_estimators=10.
Best Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 4,
'min_samples_split': 2}
Best Estimator: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=10, max_features='sqrt', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=4, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
oob_score=False, random_state=None, verbose=0,
warm_start=False)
Accuracy Score: 0.8779967288534369

GaussianNb:
'var_smoothing': 1e-10

Knn:
algorithm='auto', leaf_size=1, metric='minkowski',metric_params=None, n_jobs=None,
n_neighbors=1, p=1,weights='uniform'

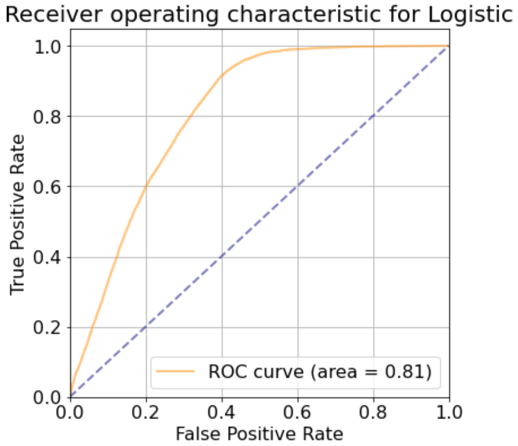
GradientBoosted:
'criterion': 'friedman_mse', 'max_depth': 32, 'max_features': 'auto', 'min_samples_leaf': 6,
'min_samples_split': 5

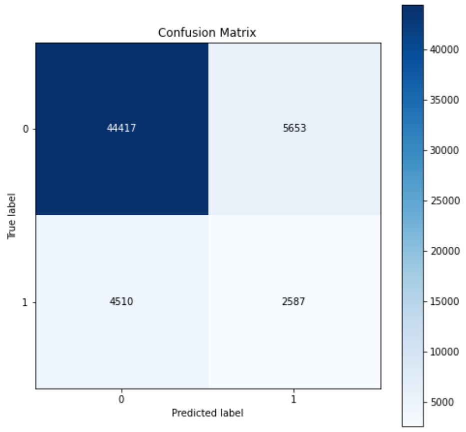
SVC:
'gamma': 0.01, 'kernel': 'linear', 'max_iter': 50, 'probability': True

5. Using area under ROC curve to select the best performing model

Logistic Regression:

ROC AUC score for Logistic model with over-sampling: 0.8088
F1 score: 0.3374



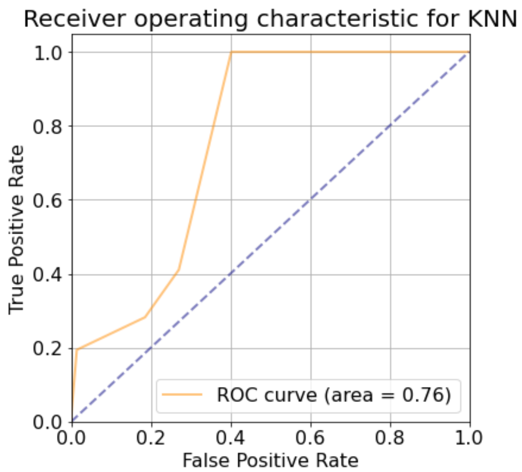


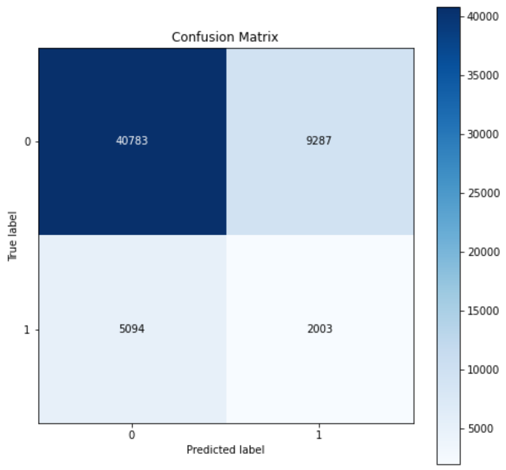
Now we got much better True Positives, and quite acceptable AUC and f1 scores

k- Nearest Neighbour:

ROC AUC score for KNN with over-sampling: 0.7624

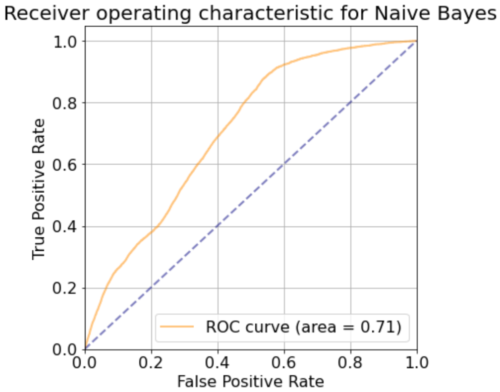
F1 score: 0.2179

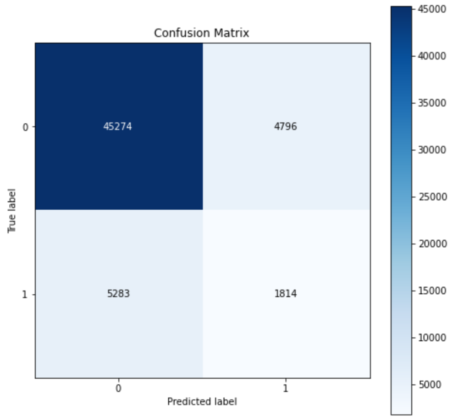




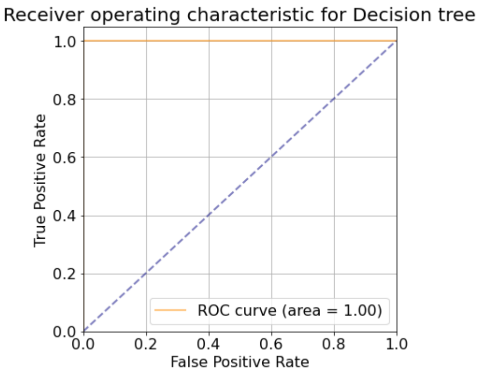
Naive Baye's

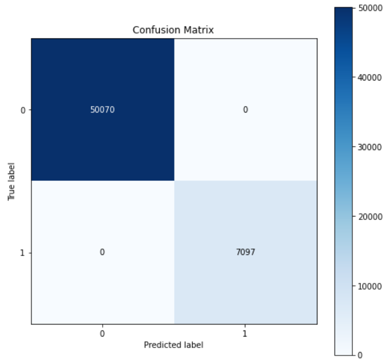
ROC AUC score for Naive Bayes with over-sampling: 0.7058
F1 score: 0.2647





Decision Tree Model:

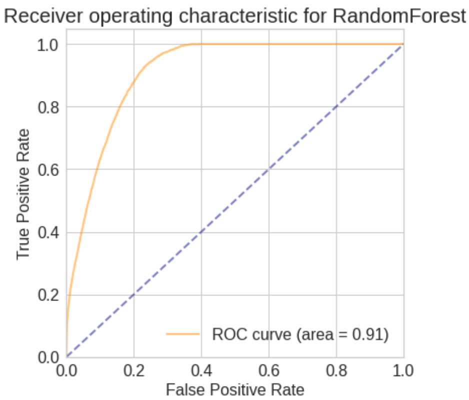


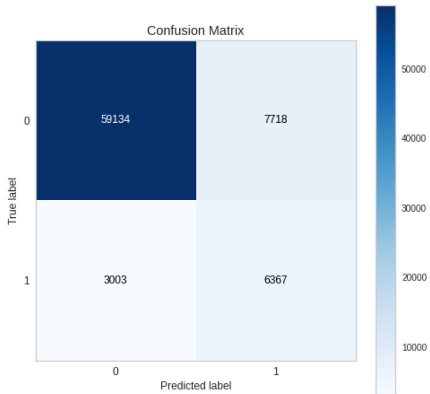


ROC AUC score for Decision Tree with over-sampling: 1.0000
F1 score: 1.0000

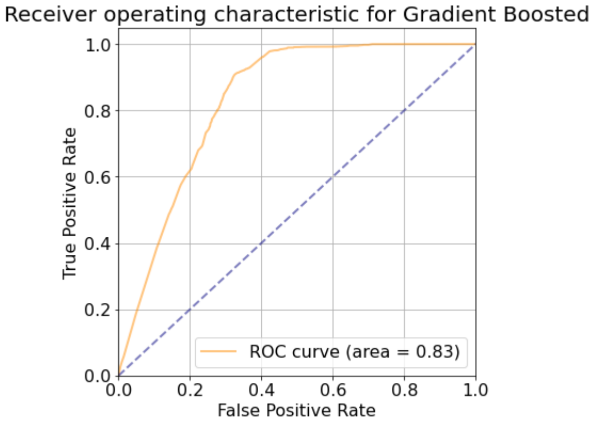
Random Forest

ROC AUC score for RandomForest model with over-sampling: 0.9111
Optimized RF f1-score 0.5429119590705607





Gradient boosted decision trees



6. Based on the ROC AUC score we got Random Forest outperformed all the other models. So, we will deploy Random forest model as FLASK API