CREDIT CARD CUSTOMER ATTRITION (CHURN) PREDICTION

CAPSTONE TWO: PROJECT REPORT

DATA SCIENCE CAREER TRACK

SPRINGBOARD

SUBMITTED BY: SHAHJAHAN AHMED

DATE: OCTOBER 27, 2021

Introduction

Building a predictive churn model helps us make proactive changes to the retention efforts that drive down churn rates. Understanding how churn impacts the current revenue goals and making predictions about how to manage those issues in the future also helps us stem the flow of churned customers. The ability to predict that a particular customer is at a high risk of churning, while there is still time to do something about it, represents a huge additional potential revenue source for every online business. To gain revenue and retain customers flow, I have selected my capstone project topics as “Credit Card Customer Attrition (Churn) Prediction”.

1. Problem Identification

1.1 Problem Statement Formation:

Predict the credit card customer attrition rate and find the potential churning customers by analyzing data for a specific period (last twelve month) of time to minimize churn rate by providing better service for company’s growth and financial stability.

1.2 Context:

Customer churn is a tendency of customers to abandon a brand and stop being a paying client of a particular business. The percentage of customers that discontinue using a company’s products or services during a particular period is known as customer churn or attrition rate. A Churn rate is a health indicator for businesses whose customers are subscribers and paying for services on a recurring basis and a churn rate higher than a certain threshold can have both tangible and intangible effects on a company's business success. In order to get profit, keep the company's growth and financial stability, the company needs to find the possible churning customer to tackle the churning rate.

1.3 Criteria for Success:

The company needs to find the potential churning customer and the attributes affecting them to churn to provide them better services and turn customers' decisions in the opposite direction. By finding potential churners and making proactive efforts will help keep the company’s growth and financial stability.

1.4 Scope of Solution Space:

The main goal of this project is to predict the potential churning customer by analyzing data for a specific period of time. To do this accurately, I will use the binary classification algorithm to find the possible churning customer.

1.5 Constraints:

By analyzing the data, I have found that only 16.07% of customers have churned and there is a correlation between two attributes also. Thus, it will be a bit difficult to have high accuracy of getting churning customers.

1.6 Stakeholders:

Bank CEO, and the Bank Manager.

1.7 Data Sources:

Customer's credit card account data from a bank. The dataset link is provided below: https://www.kaggle.com/sakshigoyal7/credit-card-customers

2. Data Wrangling

2.1 Description of Attributes:

By importing necessary libraries and loading the downloaded dataset 'BankChurners.csv', I have observed that there are total of 21 attributes. The description of attributes is given below:

CLIENTNUM: Client number. Unique identifier for the customer holding the account

Attrition_Flag: Internal event (customer activity) variable - if the account is closed then 1 else 0

Customer_Age: Demographic variable - Customer's Age in Years

Gender: Demographic variable - M=Male, F=Female

Dependent_count: Demographic variable - Number of dependents

Education_Level: Demographic variable - Educational Qualification of the account holder (example: high school, college graduate, etc.)

Marital_Status: Demographic variable - Married, Single, Divorced, Unknown

Income_Category: Demographic variable - Annual Income Category of the account holder (< 40K,40K,40K - 60K, 60K−60K−80K, 80K−80K−120K, >

Card_Category: Product Variable - Type of Card (Blue, Silver, Gold, Platinum)

Months_on_book: Period of relationship with bank

Total_Relationship_Count: Total no. of products held by the customer

Months_Inactive_12_mon: No. of months inactive in the last 12 months

Contacts_Count_12_mon: No. of Contacts in the last 12 months

Credit_Limit; Credit Limit on the Credit Card

Total_Revolving_Bal: Total Revolving Balance on the Credit Card

Avg_Open_To_Buy: Open to Buy Credit Line (Average of last 12 months)

Total_Amt_Chng_Q4_Q1: Change in Transaction Amount (Q4 over Q1)

Total_Trans_Amt: Total Transaction Amount (Last 12 months)

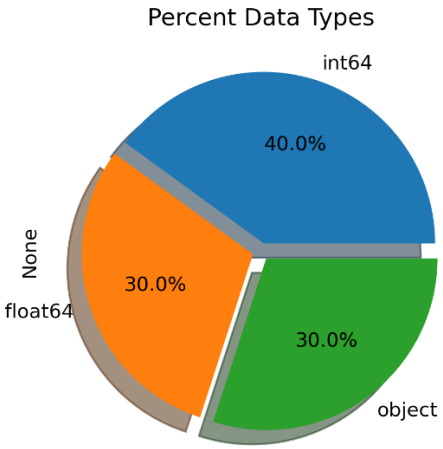
Total_Trans_Ct: Total Transaction Count (Last 12 months)

Total_Ct_Chng_Q4_Q1: Change in Transaction Count (Q4 over Q1)

Avg_Utilization_Ratio: Average Card Utilization Ratio

2.2 Dataset Summary:

The dataset had initially 10135 observations and 21 attributes. I have deleted the unnecessary 'CLIETNUM' columns and renamed the necessary attributes for simplicity. The index of the dataset is range index and columns are mixed with numerical and categorical values. Among the rest 20 attributes, 7 columns contain categorical values, and 14 columns contain numerical values. Broadly speaking 40% of the columns data types are 'int64', 30% of the columns data type is 'float64', and the rest 30% of the columns data types are 'object'.



Figures: Distribution of Attributes Based on Data Types

2.3 Initial Data Wrangling:

The steps I followed to do initial cleaning is described below:

- By observing the summary statistics, I have found that the dataset was not too messy. The 'Dependent_Count' column only contained few missing values. I have imputed those missing values by using the median value of that column.
- Among the 10135 observations, 6 of them contained duplicated values. I have removed those duplicated entries from the dataset.
- By inspecting the categorial columns, I have found some irrelevant values in the Gender columns labeled by 'U' instead of 'M' or 'F'. I have removed those entries from the dataset.
- There are few observations in the 'Credit_Limit' column contained 0 value. I have removed those observations as they are irrelevant or should not be included in the dataset.
- The shape of the dataset after initial cleaning is (10120, 20).
- The numerical columns in the dataset have some high range value in some columns compared with some other low range value in other columns. A significant number of observations are below numerical value 1 and some are above 35000. To handle those issues, it will require to do Standardization in the preprocessing step.
- Also, I have done some initial visualizations to find some more irregularities in the dataset and relationship among columns.

2.4 Identifying Dependent Variable and Independent Variables:

The ‘Attrition_Flag’ column is our dependent variable or response variable, and the rest columns are predictor variable or independent variable.

3. Exploratory data Analysis

3.1 Dataset Summary:

The dataframes info() method clearly shows us that there is no missing values in the dataframe, and it shows the distribution of columns among numerical and categorical types.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10120 entries, 0 to 10126
Data columns (total 20 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Attrition_Flag            10120 non-null  object
 1   Age                       10120 non-null  int64
 2   Gender                    10120 non-null  object
 3   Dependent_Count           10120 non-null  int64
 4   Education                 10120 non-null  object
 5   Marital_Status            10120 non-null  object
 6   Income                    10120 non-null  object
 7   Card_Category             10120 non-null  object
 8   Months_On_Book            10120 non-null  int64
 9   Total_Relationship_Count  10120 non-null  int64
 10  Months_Inactive           10120 non-null  int64
 11  Contacts_Count            10120 non-null  int64
 12  Credit_Limit              10120 non-null  float64
 13  Total_Revolving_Bal       10120 non-null  int64
 14  Avg_Open_To_Buy           10120 non-null  float64
 15  Total_Amt_Chng_Q4_Q1      10120 non-null  float64
 16  Total_Trans_Amt           10120 non-null  int64
 17  Total_Trans_Ct            10120 non-null  int64
 18  Total_Ct_Chng_Q4_Q1       10120 non-null  float64
 19  Avg_Utilization_Ratio     10120 non-null  float64
dtypes: float64(5), int64(9), object(6)
memory usage: 1.6+ MB
```

Figure: Brief info of the dataset

Also, the dataframes describe() method shows the statistical summary of numerical columns. From that we can get clear picture of the distribution of each numerical attribute.

	count	mean	std	min	25%	50%	75%	max
Age	9307.0	46.354249	7.960148	26.00	41.000	46.000	52.0000	70.000
Dependent_Count	9307.0	2.352530	1.296491	0.00	1.000	2.000	3.0000	5.000
Months_On_Book	9307.0	35.962394	7.927289	13.00	32.000	36.000	40.0000	56.000
Total_Relationship_Count	9307.0	3.855055	1.549022	1.00	3.000	4.000	5.0000	6.000
Months_Inactive	9307.0	2.299774	0.931666	0.00	2.000	2.000	3.0000	5.000
Contacts_Count	9307.0	2.440529	1.079767	0.00	2.000	2.000	3.0000	5.000
Credit_Limit	9307.0	8408.561835	8967.290840	1438.30	2496.500	4377.000	10605.0000	34516.000
Total_Revolving_Bal	9307.0	1150.677340	815.999191	0.00	192.000	1263.000	1772.5000	2517.000
Avg_Open_To_Buy	9307.0	7257.884496	8973.526223	3.00	1252.500	3312.000	9445.0000	34516.000
Total_Amt_Chng_Q4_Q1	9307.0	0.743677	0.183304	0.12	0.625	0.731	0.8510	1.412
Total_Trans_Amt	9307.0	4007.236489	2618.980107	510.00	2156.000	3872.000	4657.0000	14576.000
Total_Trans_Ct	9307.0	63.630601	21.547059	10.00	45.000	67.000	79.0000	132.000
Total_Ct_Chng_Q4_Q1	9307.0	0.695644	0.199214	0.00	0.574	0.694	0.8120	1.421
Avg_Utilization_Ratio	9307.0	0.278764	0.278354	0.00	0.012	0.180	0.5135	0.999

Figure: Statistical Summary of numerical attributes

3.2 Visualizing the Distribution of Numerical Attributes using Histogram and ECDF Plot

The histogram of any numerical column will depict the visual distribution of values of that column. The figure below shows the distribution of numerical columns by using histogram. By observing histogram its clear that some of the numerical columns are continuous and the rest are discrete. The columns ‘Dependent_Count’, ‘Total_Relationship_Count’, ‘Months_Inactive’, ‘Contacts_Count’ columns are discrete. Among those discrete columns the values of ‘Dependent_Count’ and ‘Contacts_Count’ columns are normally distributed and the other two columns have no trend or skewness. The rest numerical columns are continuous and among them the 'Total_Amt_Chng_Q4_Q1', 'Total_Ct_Chng_Q4_Q1', and the ‘Age’ columns are also normally distributed. The values of ‘Total_Trans_Count’ column shows bimodal distribution. The values of the ‘Credit_Limit’, ‘Avg_Utilization_Ratio’, and ‘Average_Open_To_Buy’ columns are skewed to the right. The peak count value of those three columns at ‘0’ clearly shows us that a significant amount customer is not utilizing their credits at all. The ‘Total_Revolving_Bal’ column shows that also a huge number of customers utilizes all their credit limit and does not have any balance to spend. By observing the ‘Total_Trans_Amt’ column it is clear that most of the customers spending is in between 0 and 5000 USD for 12-month duration period.

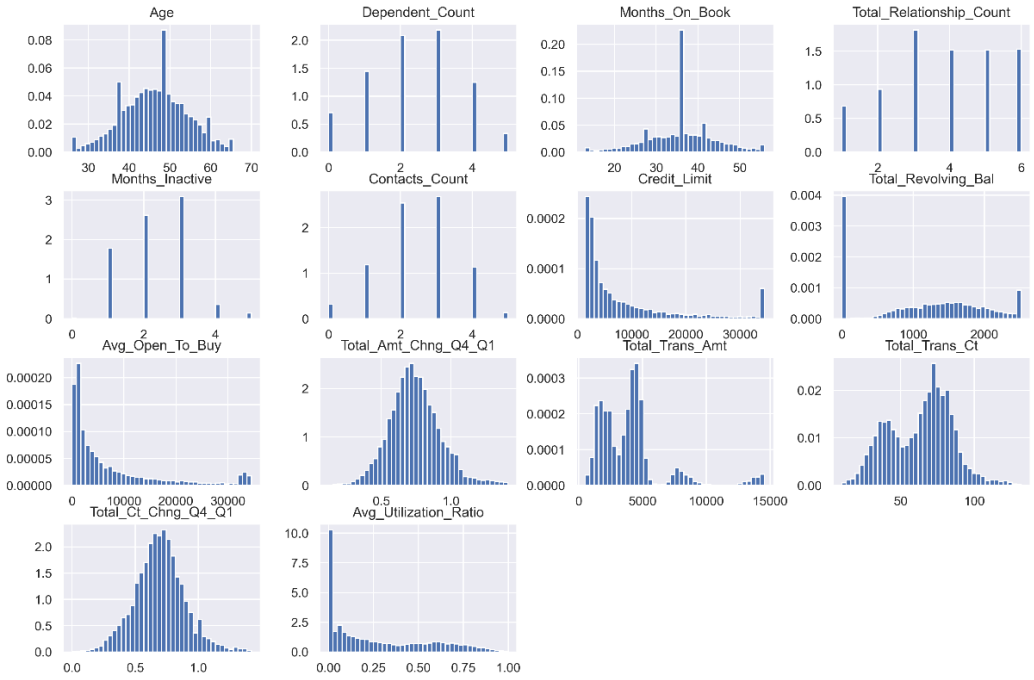
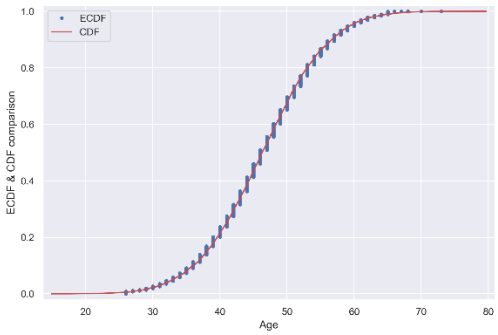
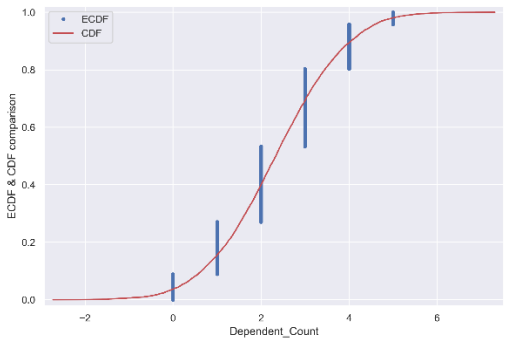


Figure: Histogram of numerical columns

The figure below shows the comparison between the theoretical cdf with empirical cdf. I have used this visualization to more precisely make a conclusion about the distribution of numerical columns whether it is normally distributed or not. It is clearly visible that the continuous numerical columns 'Age', ‘'Total_Amt_Chng_Q4_Q1', 'Total_Ct_Chng_Q4_Q1', and in the dataframe are approximately normally distributed and the rest are not normally distributed. Also, the discrete numerical columns 'Dependent_Count', 'Total_Relationship_Count', 'Months_Inactive', 'Contacts_Count' are also approximately normally distributed.





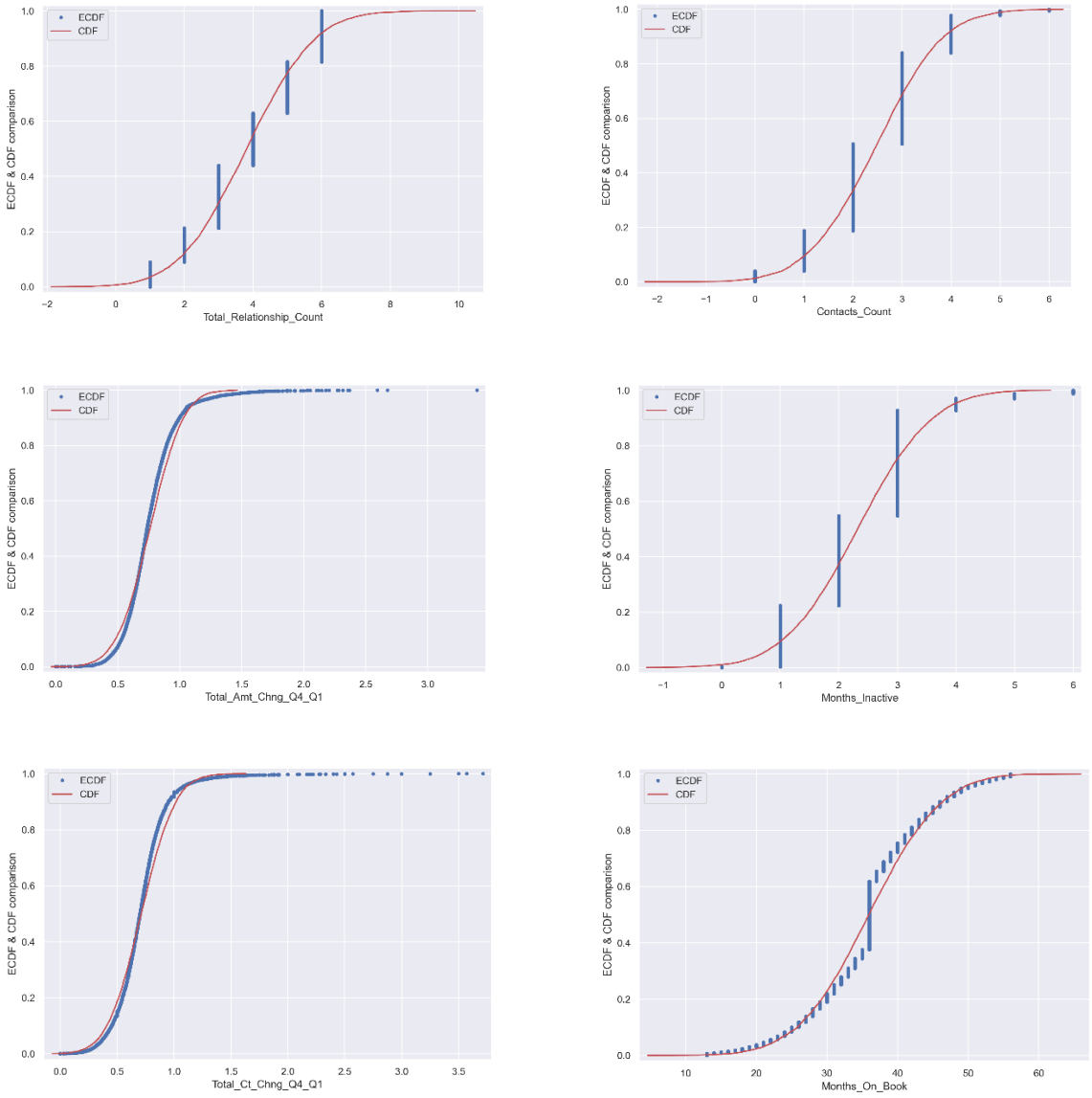


Figure: Theoretical cdf and ecdf comparison of numerical columns

3.3 Detecting and Removing Outliers

Boxplot is an essential visualization type for detecting outlier of a numerical column. Figure below shows the boxplot of few numerical columns and the visualization of standard deviation-based outlier detection. To accurately detect and correct the columns that contain outlier value I have used standard deviation-based outlier detection. If the distribution of values in a particular column is above or below the $Mean \pm 3 * Standard\_Deviation$ the column contain outlier and it must be removed to get better classification accuracy. By observing the histogram and vertical lines (Mean, $Mean \pm 3 * Standard\_Deviation$) its visible that the numerical columns 'Months_Inactive', 'Contact_Count', 'Total_Amt_Chng_Q4_Q1', 'Total_Trans_Amt',

'Total_Ct_Chng_Q4_Q1' contains significant outliers. Finally, I have removed those outlier values from the dataset without the discrete 'Months_Inactive', and 'Contact_Count' columns. The result of outlier value removal creates few ‘NaN’ entries in the dataset and those null values have further removed from the dataset.

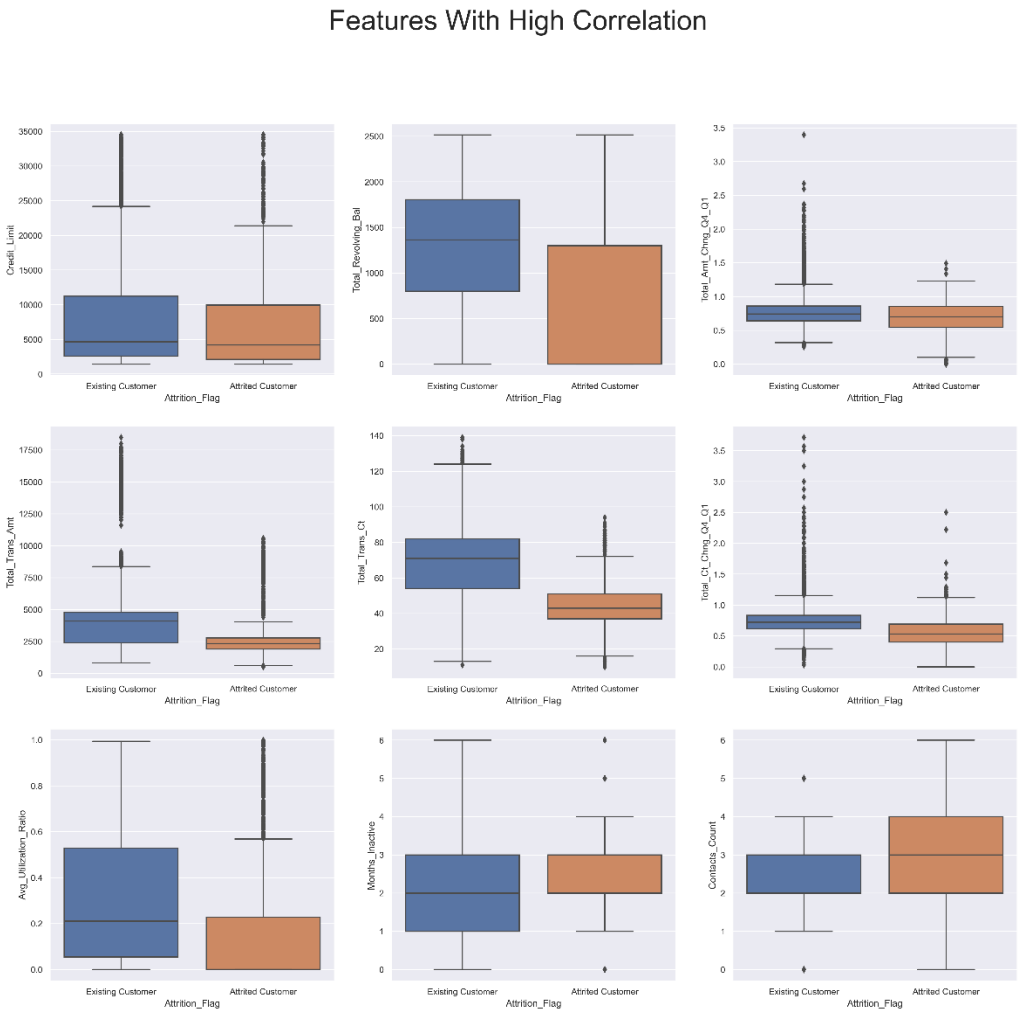


Figure: Visualization of outliers using boxplot

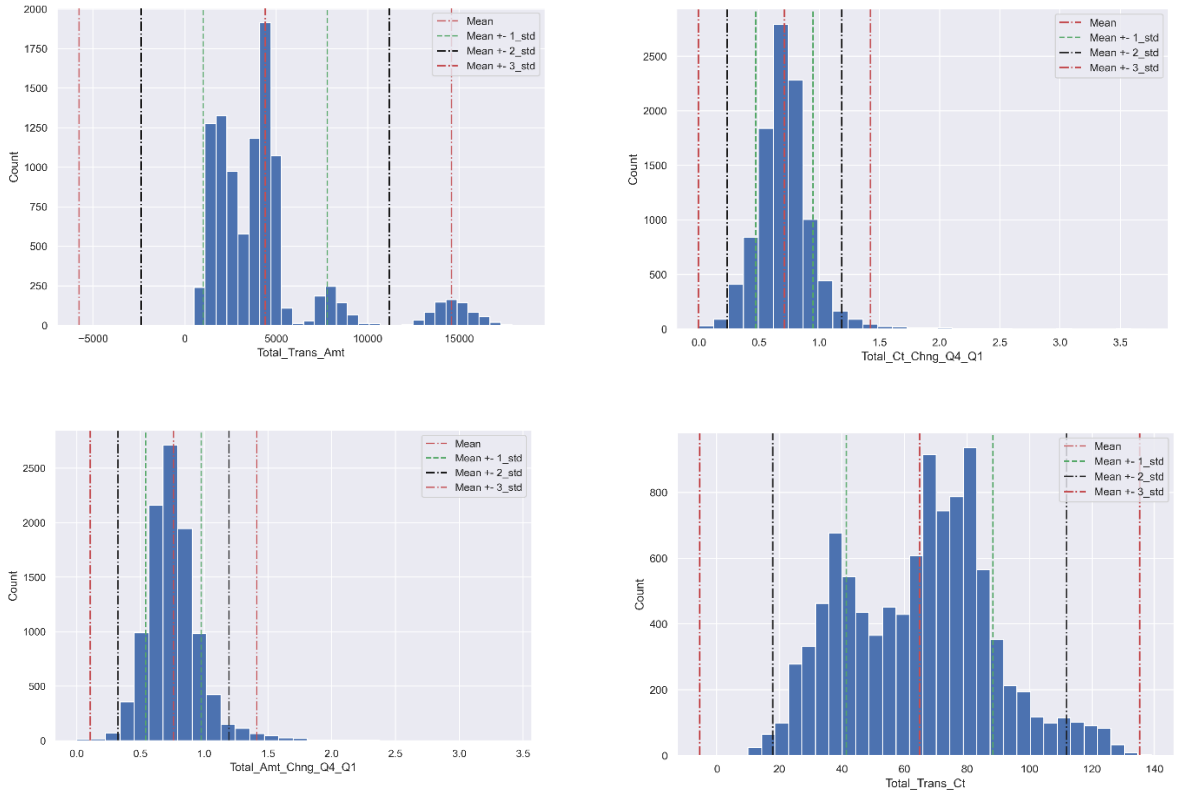


Figure: $Mean \pm 3 * Standard\_Deviation$ based outlier detection.

3.4 Detecting and Removing Correlations

Heatmaps are used to show relationships between two variables, one plotted on each axis. By observing how cell colors change across each axis, we can observe if there are any patterns in value for one or both variables. By observing the output of heatmap it is clearly visible that there is a significantly high positive correlations between the 'Credit_Limit' and 'Avg_Open_To_Buy' columns and the Pearson correlation coefficient value is 0.9958. Also, there is a positive and negative correlation between few other columns also.

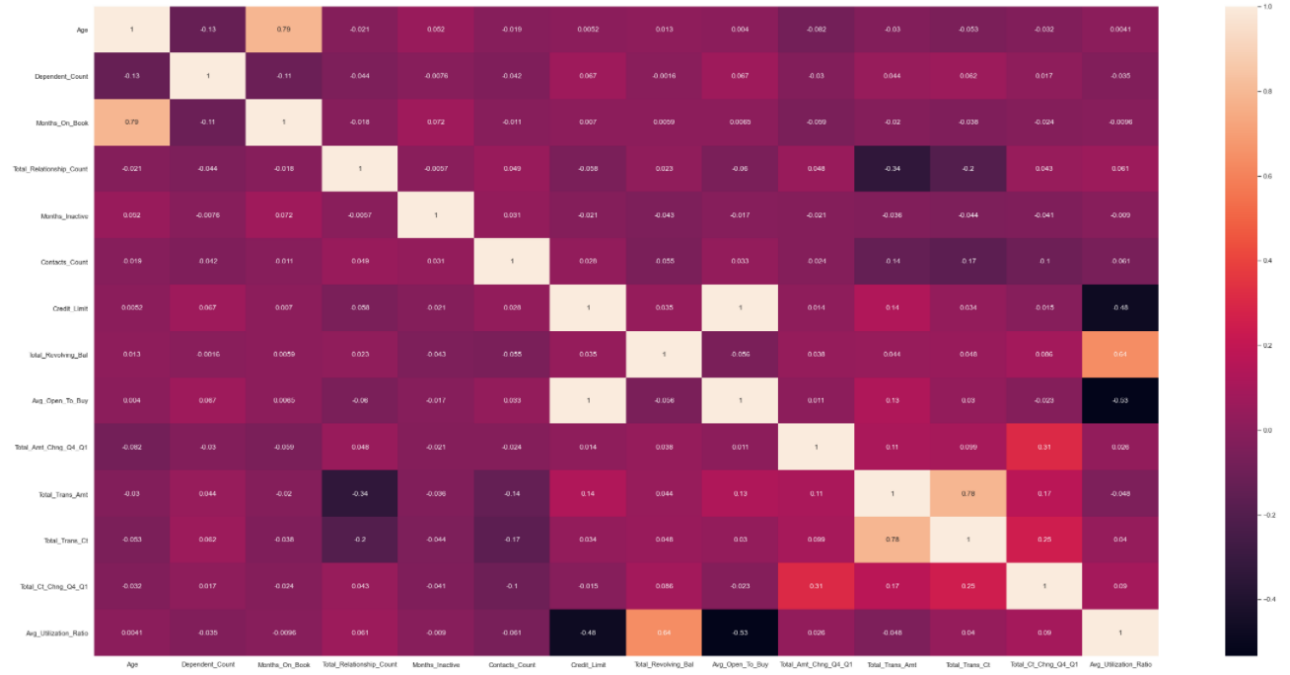
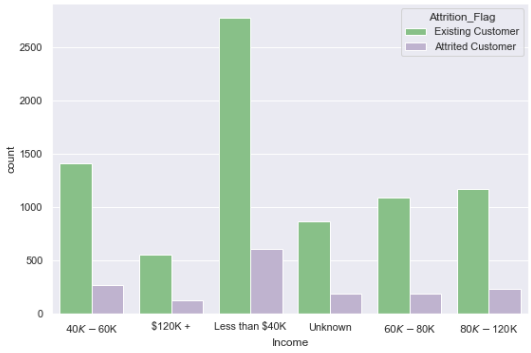


Figure: Heatmap of numerical attributes

3.5 Visualizing the Distribution of Categorical Attributes using Count Plot

The figures below show the count plot which visualizes the distribution of categorical columns. The 'Card_Category' column contains class imbalance data.



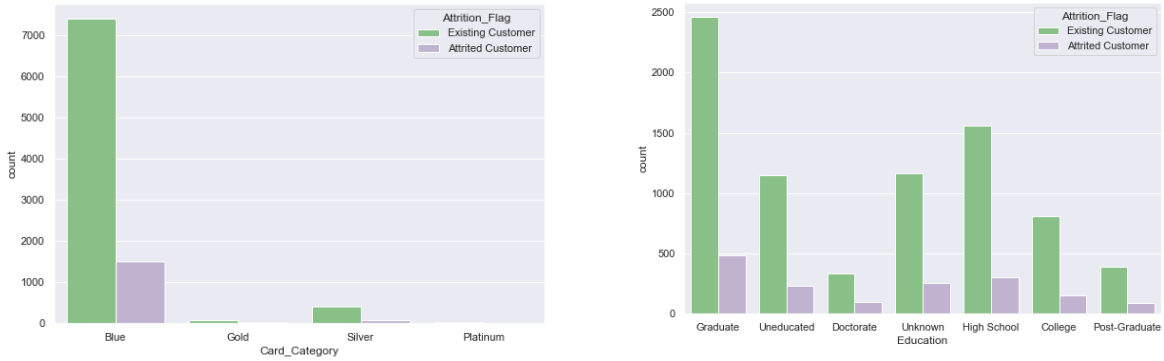


Figure: The distribution of Categorical Attributes

4. Preprocessing and Training Data Development

Data preprocessing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important to preprocess the data before feeding it into our model. The steps I followed for preprocess the data is given below:

- Encoded the target attribute ‘Attrition_Flag’ to binary 1 and 0. I have encoded the binary ‘1’ for 'Attrited Customer' and binary ‘0’ for 'Existing Customer'. The main goal of the classification algorithms is to find the possible 'Attrited Customer' or is to predict ‘1’.

- Separated the target/dependent variable y, and the predictor/Independent variables X from the dataset.

- Split the data (X and y) into train and test with default test size of 0.25 for performing training and evaluation for each classifier. After splitting the shape of the X_train, X_test, y_train, and y_test respectively (7108, 19), (2370, 19), (7108,), and (2370,)

- I have used StandardScaler as scaling function for training and test set separately because StandardScaler removes the mean and scales each feature/variable to unit variance. This operation is performed feature-wise in an independent way. I have used scaling function because the numerical columns of the dataset still contained outliers and the range of each individual columns are different. A significant number of observations are below numerical value 1 and some are above 35000.

- Finally, I have used the pandas get_dummies() function for one hot encoding of the categorical attributes. It converts each of the categorical data into dummy or indicator variables.

5. Modeling

To predict the class or category we need to use the classification algorithm. The Classification algorithm's function is to identify the category of new observations based on training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into classes or groups. Such as, Yes or No, 0 or 1. In classification algorithm, a discrete output function(y) is mapped to input variable(x).

5.1 Evaluation Metrics for Model Evaluation:

Accuracy:

Accuracy in classification problems is the number of correct predictions made by the model over all kinds of predictions made.

Precision:

Precision in binary classification (Yes/No) refers to a model's ability to correctly interpret positive observations. In other words, how often does a positive value forecast turn out to be correct? We may manipulate this metric by only returning positive for the single observation in which we have the most confidence.

Recall:

The recall is also known as sensitivity. In binary classification (Yes/No) recall is used to measure how "sensitive" the classifier is to detecting positive cases. To put it another way, how many real findings did we "catch" in our sample? We may manipulate this metric by classifying both results as positive. In our case recall is a measure that tells us what proportion of customer in the dataset that going to churn will diagnosed by the algorithm as possible churner.

AUC-ROC curve:

ROC curve stands for Receiver Operating Characteristics Curve and AUC stands for Area Under the Curve. It is a graph that shows the performance of the classification model at different thresholds. To visualize the performance of the multi-class classification model, we use the AUC-ROC Curve. The ROC curve is plotted with TPR and FPR, where TPR (True Positive Rate) on Y-axis and FPR(False Positive Rate) on X-axis.

F1 Score:

The F1 score can be thought of as a weighted average of precision and recall, with the best value being 1 and the worst being 0. Precision and recall also make an equal contribution to the F1 ranking.

5.2 Classification Algorithms for Modeling

To find the best classifier algorithm to predict the possible churning customer I have used different classification algorithms including KNN, Logistic Regression, Decision Tree. Support Vector Machines, Random Forest, Gradient Boosting, and XGBoost Classifiers. For each of

those classification algorithms, I have used hyperparameter tuning by using the randomized search cv to get best performance from the model. The result of each classification model and the corresponding roc curve shown below:

5.2.1 Customer Churn Prediction using KNN Classifier:

Model: KNeighborsClassifier(leaf_size=5, n_neighbors=3, p=1)		
Precision Score: 0.791	Recall Score: 0.603	Training Accuracy: 0.951
Test Accuracy: 0.907	F1 Score: 0.902	Area Under ROC: 0.895

Figure: KNN evaluation metrics score

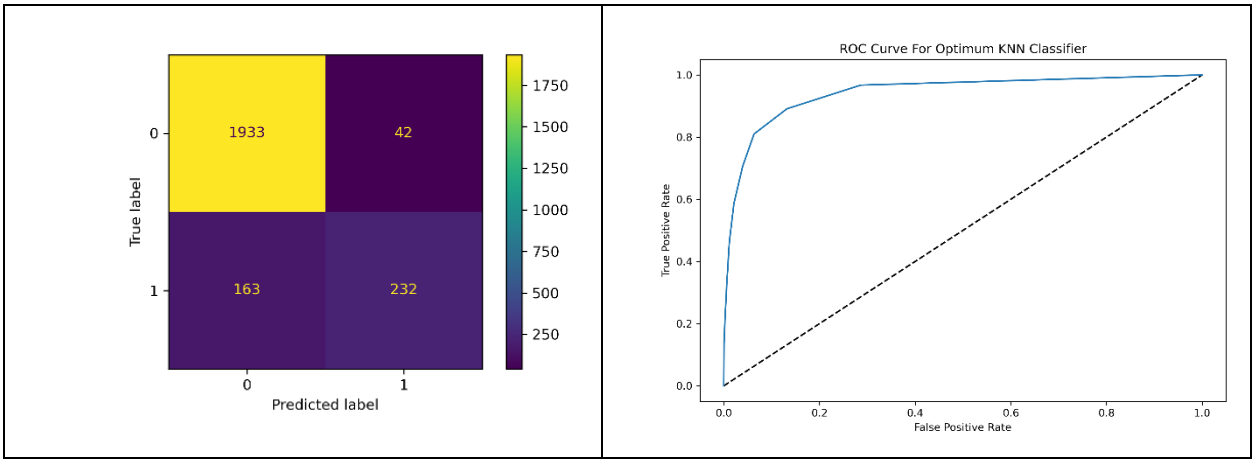


Figure: Confusion matrix and ROC curve for KNN Classifier

5.2.2 Customer Churn Prediction using Logistic Regression Classifier::

Model: LogisticRegression(C=0.1)		
Precision Score: 0.768	Recall Score: 0.580	Training Accuracy: 0.900
Test Accuracy: 0.901	F1 Score: 0.895	Area Under ROC: 0.934

Figure: Logistic Regression evaluation metrics score

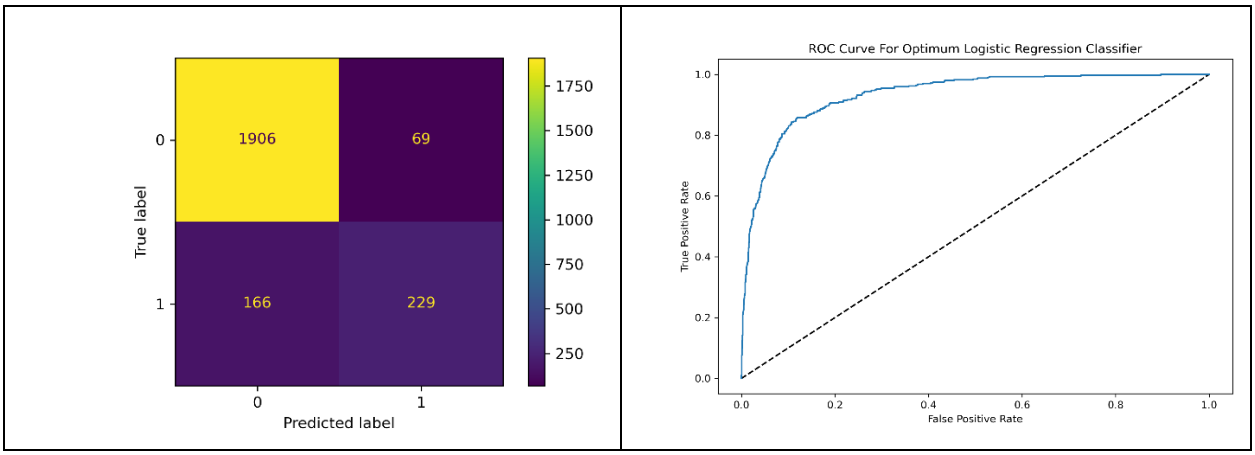


Figure: Confusion matrix and ROC curve for Logistic Regression Classifier

5.2.3 Customer Churn Prediction using Decision Tree Classifier:

Model: DecisionTreeClassifier(max_depth=9)		
Precision Score: 0.853	Recall Score: 0.762	Training Accuracy: 0.982
Test Accuracy: 0.938	F1 Score: 0.937	Area Under ROC: 0.900

Figure: Decision Tree evaluation metrics score

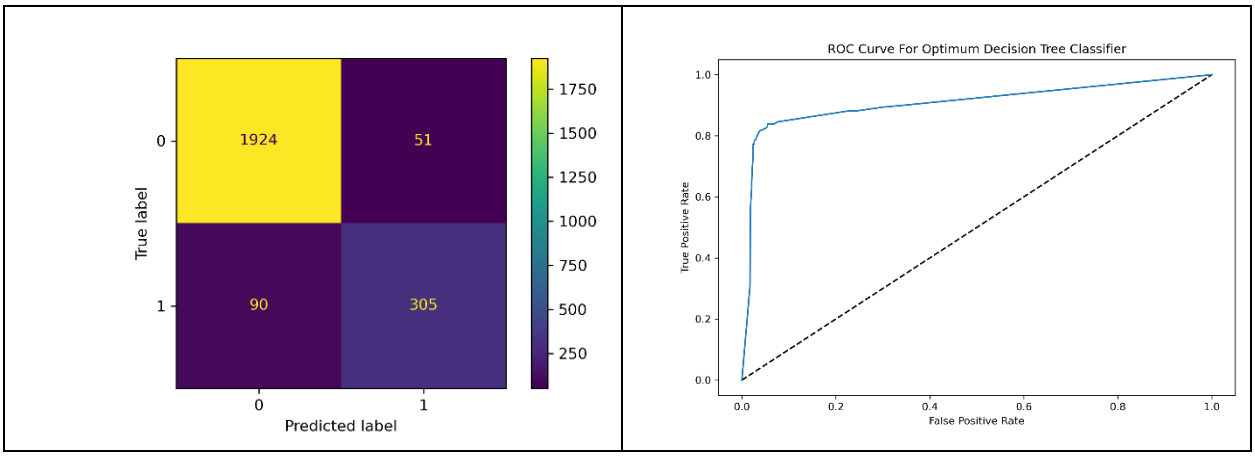


Figure: Confusion matrix and ROC curve for Decision Tree Classifier

5.2.4 Customer Churn Prediction using SVM Classifier:

Model: SVC(C=1, gamma=0.1, probability=True)		
Precision Score: 0.885	Recall Score: 0.719	Training Accuracy: 0.966
Test Accuracy: 0.938	F1 Score: 0.935	Area Under ROC: 0.970

Figure: SVM evaluation metrics score

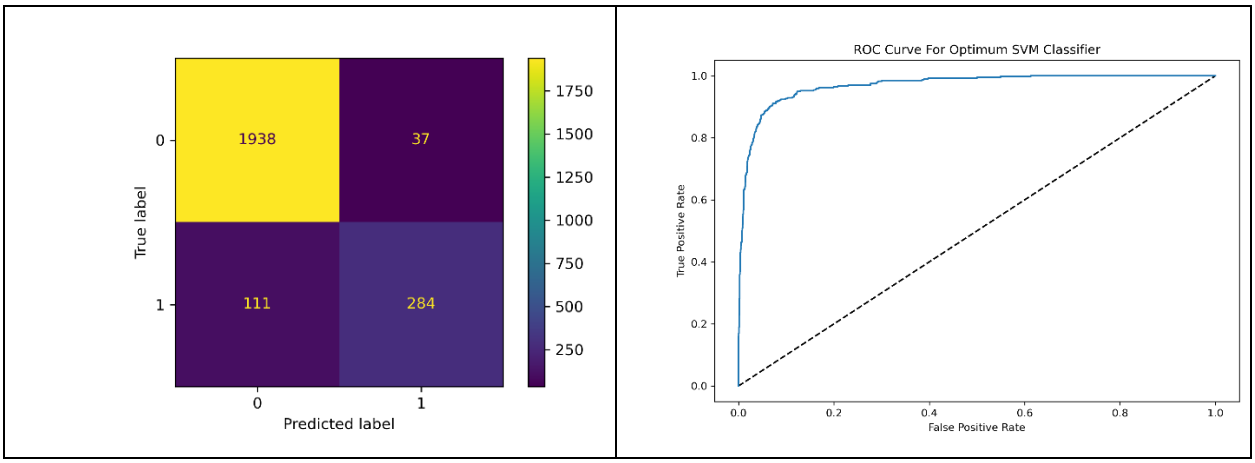


Figure: Confusion matrix and ROC curve for SVM Classifier

5.2.5 Customer Churn Prediction using Random Forest Classifier:

Model: RandomForestClassifier(n_estimators=800, random_state=42)		
Precision Score: 0.909	Recall Score: 0.830	Training Accuracy: 1.000
Test Accuracy: 0.958	F1 Score: 0.957	Area Under ROC: 0.987

Figure: Random Forest evaluation metrics score

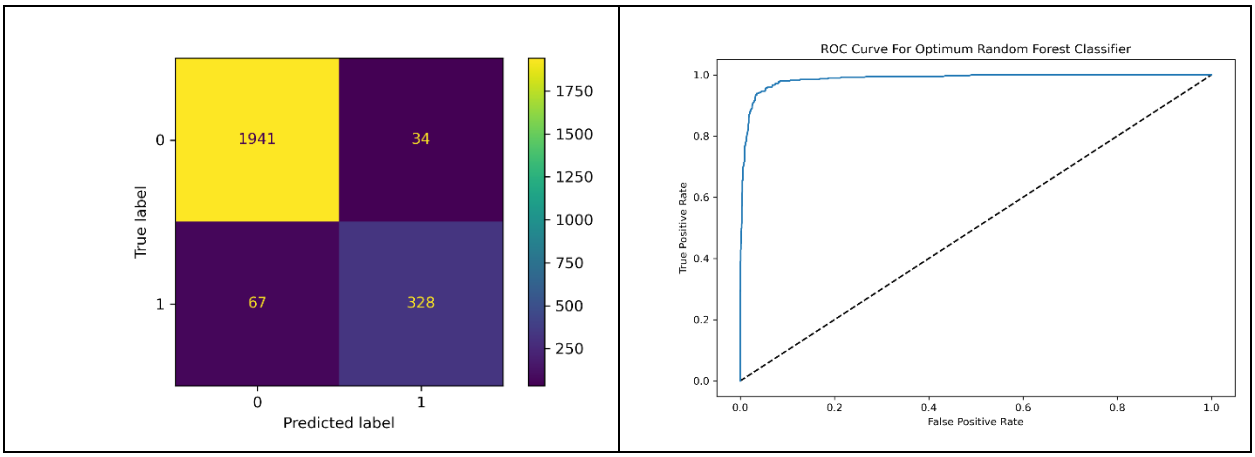


Figure: Confusion matrix and ROC curve for Random Forest Classifier

5.2.6 Customer Churn Prediction using Gradient Boosting Classifier:

Model: GradientBoostingClassifier(n_estimators=500)		
Precision Score: 0.934	Recall Score: 0.901	Training Accuracy: 0.997
Test Accuracy: 0.973	F1 Score: 0.973	Area Under ROC: 0.994

Figure: Gradient Boosting evaluation metrics score

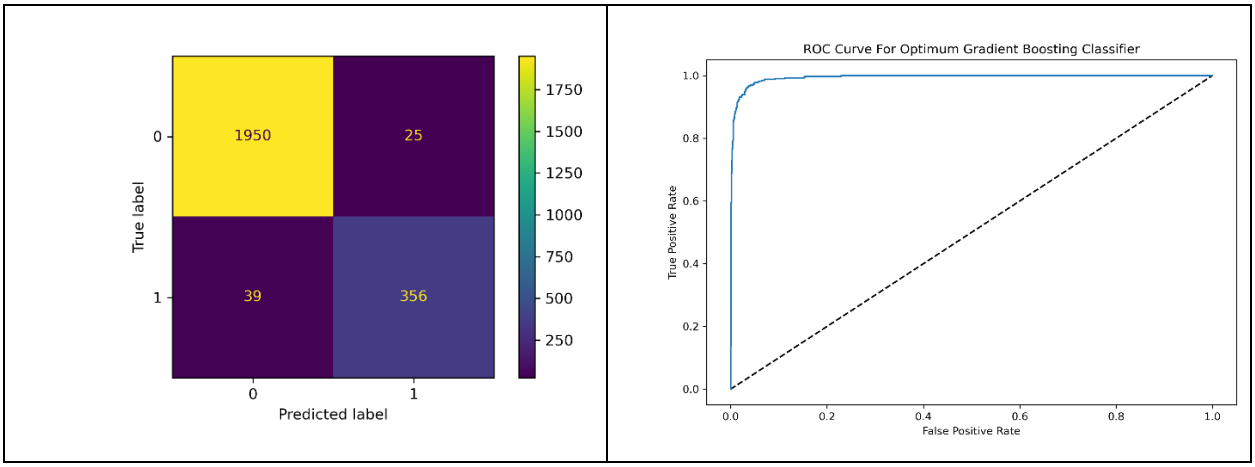


Figure: Confusion matrix and ROC curve for Gradient Boosting Classifier

5.2.7 Customer Churn Prediction using XGBoost Classifier:

Model: XGBClassifier(metric='logloss', learning_rate=0.3, max_depth=3, n_estimators=500)		
Precision Score: 0.935	Recall Score: 0.909	Training Accuracy: 0.999
Test Accuracy: 0.974	F1 Score: 0.974	Area Under ROC: 0.994

Figure: XGBoost evaluation metrics score

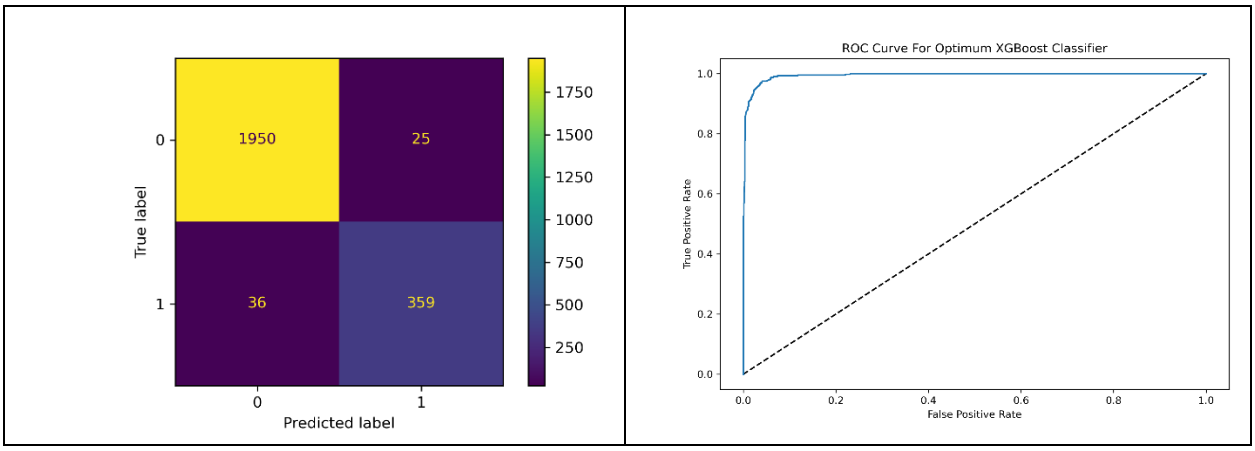


Figure: Confusion matrix and ROC curve for XGBoost Classifier

6. Best Model for Predicting Customer Churn and Findings:

Among all the algorithms mentioned above, the XGBoost Classifier performed well based on the several model evaluation metrics I have used. The resultant summary of XGBoost classifier is Precision Score: 0.935, Recall Score: 0.909, Training Accuracy: 0.999, Test Accuracy: 0.974, F1 Score: 0.974, Area Under ROC: 0.994. To predict all possible churning customer accurately or to get a model that will have low false negatives, I have used the metric 'Recall' as my main evaluation metric. A good classification model should have a high recall value to classify all possible churning customers. Despite the target attribute of my model contained class imbalance data, the 'Recall' value of 0.909 or 91% indicates that my model worked well on predicting possible churning customers. Also, the 'F1 Score' of 0.974 and the Area under roc curve of 0.994 indicates that my model performed well.

Conclusion and future work:

The target attributes have class imbalance data, and approximately 16% of the customers are labeled with 'Attrited Customer' and the remaining 84% customers are labeled with 'Current Customer'. Because of the class imbalance data, few correlations, and outliers in the dataset, my model did not work well as I expected. Spending more time on preprocessing to tackle those issues will be helpful to get better accuracy on each of the evaluation metrics I have used.