Project Evaluation and Interpretation:

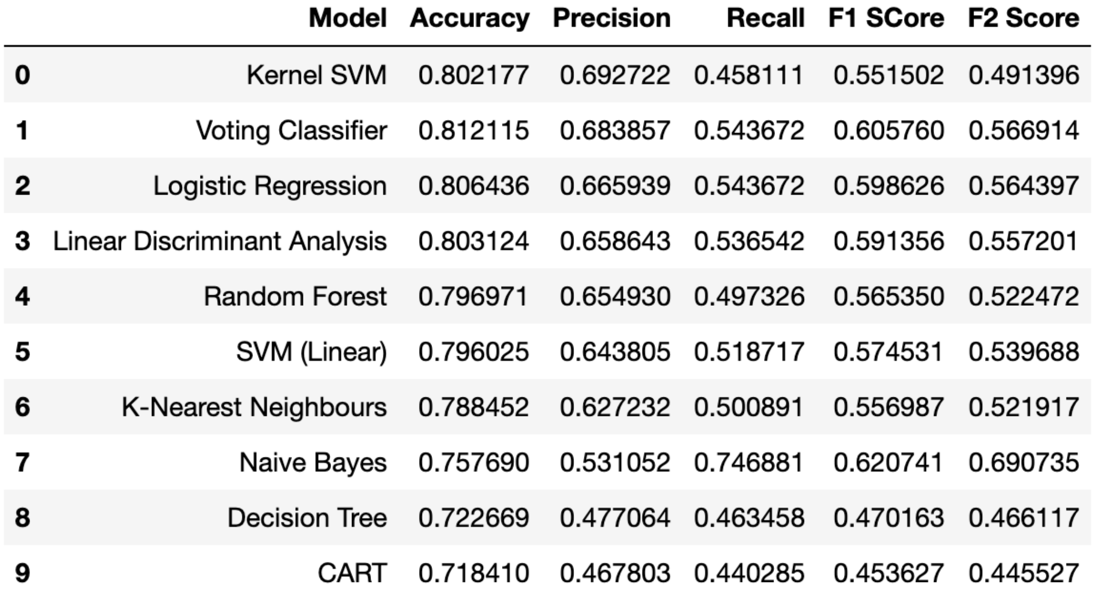
Objectives of this project:

- Finding the % of churn customers and customers that are loyal to the active services.
- Selecting the most optimal model for correct classification of churn and non-churn customers.

Model chosen to be implemented:

1. Logistic Regression
2. Linear Discriminant Analysis

We had a few reasons for choosing these models for our problem. One of the reasons was its accuracy values along with the other performance metrics, namely precision, recall and F1-score.

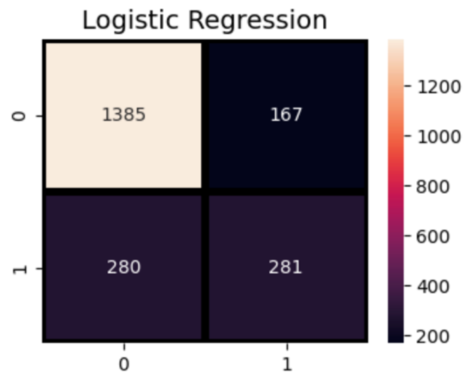


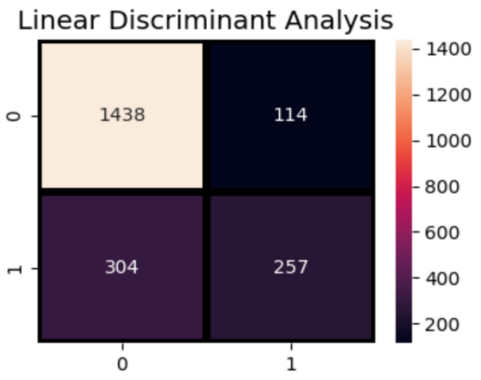
	Model	Accuracy	Precision	Recall	F1 SCore	F2 Score
0	Kernel SVM	0.802177	0.692722	0.458111	0.551502	0.491396
1	Voting Classifier	0.812115	0.683857	0.543672	0.605760	0.566914
2	Logistic Regression	0.806436	0.665939	0.543672	0.598626	0.564397
3	Linear Discriminant Analysis	0.803124	0.658643	0.536542	0.591356	0.557201
4	Random Forest	0.796971	0.654930	0.497326	0.565350	0.522472
5	SVM (Linear)	0.796025	0.643805	0.518717	0.574531	0.539688
6	K-Nearest Neighbours	0.788452	0.627232	0.500891	0.556987	0.521917
7	Naive Bayes	0.757690	0.531052	0.746881	0.620741	0.690735
8	Decision Tree	0.722669	0.477064	0.463458	0.470163	0.466117
9	CART	0.718410	0.467803	0.440285	0.453627	0.445527

As you can see from this image, we had obtained high accuracy levels for both Logistic Regression and Linear Discriminant Analysis with the difference in value between the two models to be almost negligible. On further investigation we notice difference of performance in the other metrics. But since the differences were too minimal, we had collectively decided to implement both models.

Since our focus was on classification, we decided to choose Linear discriminant analysis and Logistic regression for our problem since they are both binary classification models that can handle categorical target variables with two classes (“yes” or “no”).

This is the confusion matrix we had obtained for each model. A heat map of the confusion matrix is made so has to facilitate ease in comparison between various models.


Logistic Regression


Linear Discriminant Analysis

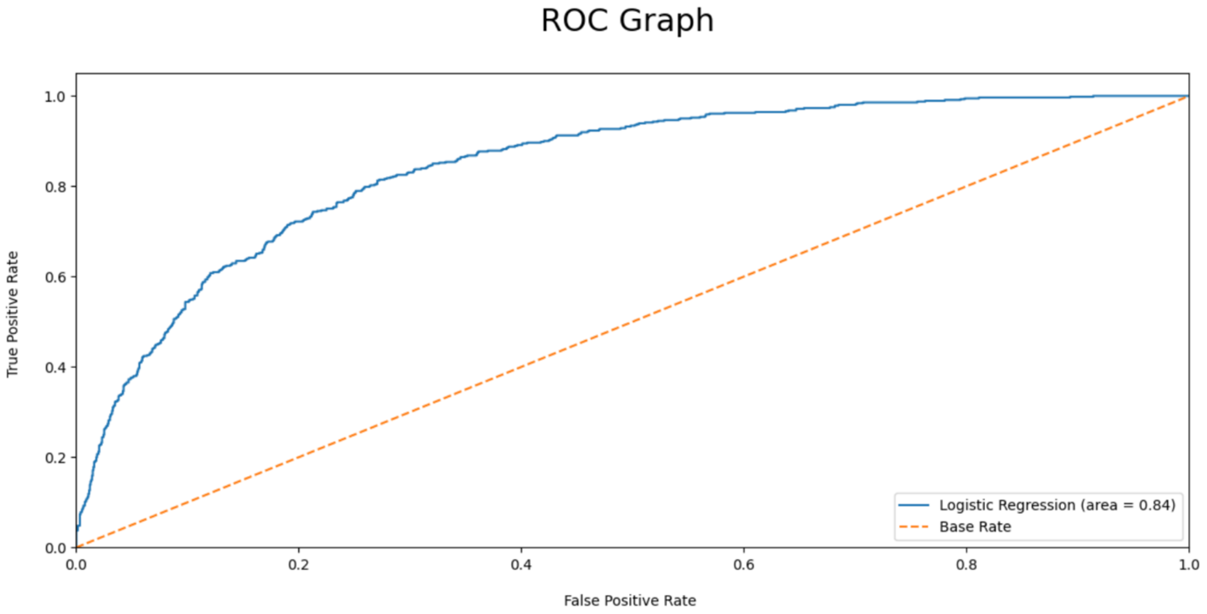
Logistic Regression:

Error rate = (FP + FN) / (TP + FP + TN + FN) = (280 + 167) / (1385 + 280 + 281 + 167) = 0.234
Sensitivity = TP / (TP + FN) = 1385 / (1385 + 167) = 0.892
Specificity = TN / (TN + FP) = 281 / (281 + 280) = 0.501
Therefore, the error rate is 0.234 or 23%, the sensitivity is 0.892 or 89.2%, and the specificity is 0. 501.or 50%.

Linear Regression:

Error rate = (304 + 114) / 2113 = 0.21 or 21%
Sensitivity = 1438 / (1438 + 114) = 0.93 or 93%
Specificity = 257 / (257 + 304) = 0.46 or 46%
Therefore, the error rate is 21%, the sensitivity is 93%, and the specificity is 46%.

ROC Curve:


ROC Graph

Drawbacks and Limitations of the models,

Logistic Regression:

- This model can overfit the data if there are too many features. This makes the model more complex, and it might lead to poor performance.
- This model is sensitive to outliers that can affect the result and also hinder performance.

Solutions:

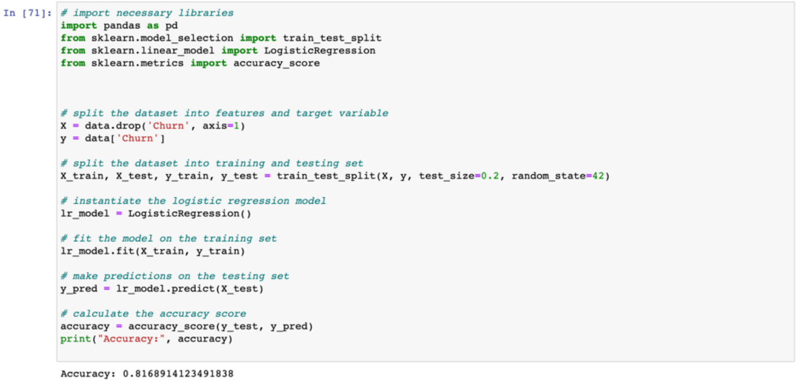
- Regularization methods such as L1 and L2 can help reduce overfitting.
- Detecting and removing the outliers to improve performance.

Linear Discriminant Analysis:

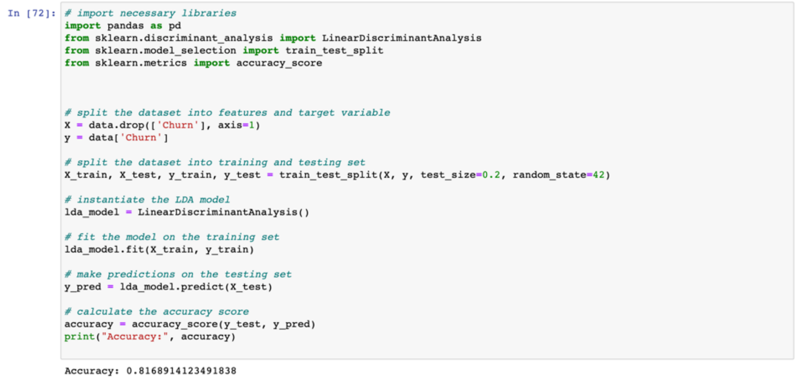
- This model assumes that the variance is the same for all classes which may not be the case for all datasets.
- This model does not perform well when the number of samples is smaller than the number of features.

Solutions:

- Quadratic Discriminant Analysis is an effective way to tackle the variance problem.
- Dimensionality reduction techniques such as Principal Component Analysis(PCA) can help reduce the number of features and improve the model's overall performance.



```
In [71]: # import necessary libraries
         import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LogisticRegression
         from sklearn.metrics import accuracy_score



         # split the dataset into features and target variable
         X = data.drop('Churn', axis=1)
         y = data['Churn']

         # split the dataset into training and testing set
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # instantiate the logistic regression model
         lr_model = LogisticRegression()

         # fit the model on the training set
         lr_model.fit(X_train, y_train)

         # make predictions on the testing set
         y_pred = lr_model.predict(X_test)

         # calculate the accuracy score
         accuracy = accuracy_score(y_test, y_pred)
         print("Accuracy:", accuracy)

         Accuracy: 0.8168914123491838
```



```
In [72]: # import necessary libraries
         import pandas as pd
         from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score



         # split the dataset into features and target variable
         X = data.drop(['Churn'], axis=1)
         y = data['Churn']

         # split the dataset into training and testing set
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # instantiate the LDA model
         lda_model = LinearDiscriminantAnalysis()

         # fit the model on the training set
         lda_model.fit(X_train, y_train)

         # make predictions on the testing set
         y_pred = lda_model.predict(X_test)

         # calculate the accuracy score
         accuracy = accuracy_score(y_test, y_pred)
         print("Accuracy:", accuracy)

         Accuracy: 0.8168914123491838
```

Final Accuracy values for both models are in the above code snippet showing almost a negligible difference in score.