DATA MINING

FINAL PROJECT REPORT

“TELECOM COSTUMER CHRUN PREDICTION”

Anwar Hussain Sofi

Researcher | AI/ML/DL Enthusiast | Neural Networks | Image Segmentation Specialist

1. INTRODUCTION

1.1. Background

Customer churn is the occurrence of clients ending their association with a business. Due to fierce competition and the availability of several service providers, client turnover in the telecom sector is a big worry. Accurate customer churn prediction helps lower customer attrition and enhance customer retention efforts for telecom companies.

Since keeping existing customer's costs less than recruiting new ones, customer churn is a major problem for telecom firms. We wanted to create a reliable predictive model for telecom customer attrition with this project. We developed a model that can detect customers who are likely to churn, enabling the business to take preventative efforts to retain them. This model was created by evaluating previous customer data and utilizing advanced machine learning techniques. The approach, outcomes, and major conclusions of our telecom customer churn prediction experiment are presented in this report.

In this project, we utilized the "WA_Fn-UseC_-Telco-Customer-Churn.csv" dataset from Kaggle to develop a churn prediction model.

1.2. Dataset

The "Telecom Customer Churn" dataset includes details on telecom customers' characteristics. Every customer is represented by a row, and every attribute or quality of each customer is represented by a column. The dataset has 21 columns that reflect distinct attributes and 7043 rows that represent various clients. This dataset's primary goal is to forecast client attrition, as seen by the "Churn" column. Customers who have stopped doing business with the telecom firm are referred to as churning. The target variable for predicting customer attrition is the "Churn" column.

The remaining columns in the dataset provide additional information about the customers. These columns contain customer attributes described in the column metadata. The specific attributes can include demographic information, services subscribed, billing details, customer account information, customer service interactions, and more. The dataset provides a comprehensive view of customers and their characteristics, allowing for analysis and prediction of churn behaviors.

The data in this dataset can be used by telecom businesses to create churn prediction models using machine learning and predictive modelling methods. These models can assist in identifying consumers who are most likely to leave, allowing businesses to take proactive steps to keep them. Telecom firms can develop targeted retention tactics and raise customer satisfaction by comprehending the patterns and causes of client turnover.

It is important to note that the precise information and descriptions of each attribute cannot be provided without access to the particular column metadata. Therefore, in order to fully comprehend the customer attributes and their possible impact on churn prediction, more analysis, exploration, and knowledge of the dataset's column metadata would be required.

1.3. Implementation Method

In this project used to implemented and evaluated several machine learning models for predictions using the K-Nearest Neighbors (KNN), Random Forest, Decision Tree Classifier, and AdaBoost Classifier algorithms.

K-Nearest Neighbors (KNN):

KNN is a non-parametric algorithm that classifies a new data point based on the majority class of its nearest neighbors. To implement the KNN model, we calculated the distance between the new data point and existing data points in the training set. The class label of the new data point was determined based on the majority vote of its nearest neighbors.

Random Forest:

Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. Each decision tree is trained on a random subset of the training data, and the final prediction is made by aggregating the predictions of all the individual trees. We implemented the Random Forest model by constructing a forest of decision trees and averaging the predictions to obtain the final result.

Decision Tree Classifier:

The Decision Tree Classifier builds a binary tree based on feature values to make predictions. Each internal node of the tree represents a feature, and the edges represent the possible outcomes of the feature. To implement the Decision Tree Classifier, we recursively split the data based on the features that best separate the classes until a stopping criterion is reached. The class label of a new data point is determined by traversing the tree from the root node to a leaf node.

AdaBoost Classifier:

AdaBoost, short for Adaptive Boosting, is a boosting algorithm that combines weak classifiers into a strong classifier. In each iteration, the AdaBoost Classifier assigns higher weights to misclassified samples, allowing subsequent weak classifiers to focus more on these samples. The final prediction is obtained by aggregating the predictions of all the weak classifiers, weighted by their performance.

To evaluate the performance of these models, we employed various evaluation metrics such as accuracy, precision, recall, and F1-score. We split the dataset into training and testing sets, using the training set to train the models and the testing set to evaluate their performance. Additionally, techniques like cross-validation and hyper parameter tuning were used to optimize the models and improve their generalization capabilities.

By comparing the evaluation metrics and analyzing the results, we were able to assess the effectiveness of each model for making predictions. The model with the highest performance on the chosen evaluation metrics would be considered the most suitable for the specific prediction task at hand.

2. EXPERIMENT SETUP

2.1. Important Library

Importing the libraries needed to create the software comes before designing the system. pandas (import pandas as pd):

Pandas is a powerful library for data manipulation and analysis. It provides data structures such as Data Frames that allow for easy handling and manipulation of structured data. The library offers various functions and methods to read, write, filter, aggregate, and transform data. By importing pandas as pd, we can refer to its functions using the pd prefix.

NumPy (import NumPy as np):

NumPy is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. By importing NumPy as np, we can access its functions using the np prefix.

matplotlib.pyplot (import matplotlib.pyplot as plt):

Matplotlib is a widely used plotting library in Python. The pyplot module of Matplotlib provides a MATLAB-like interface for creating various types of plots, such as line plots, scatter plots, bar plots, histograms, etc. By importing matplotlib.pyplot as plt, we can use its functions to create and customize plots. seaborn (import seaborn as sns):

Seaborn is a data visualization library built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn simplifies the process of creating complex visualizations, such as heatmaps, pair plots, categorical plots, and more. By importing seaborn as sns, we can use its functions to enhance our data visualizations. Plotly express (import plotly.express as px):

Plotly Express is a high-level API for creating interactive visualizations. It offers a concise syntax to generate a wide range of charts, including scatter plots, line plots, bar plots, box plots, and more. Plotly Express allows for easy customization and interactivity in visualizations. plotly.graph_objects (import plotly.graph_objects as go):

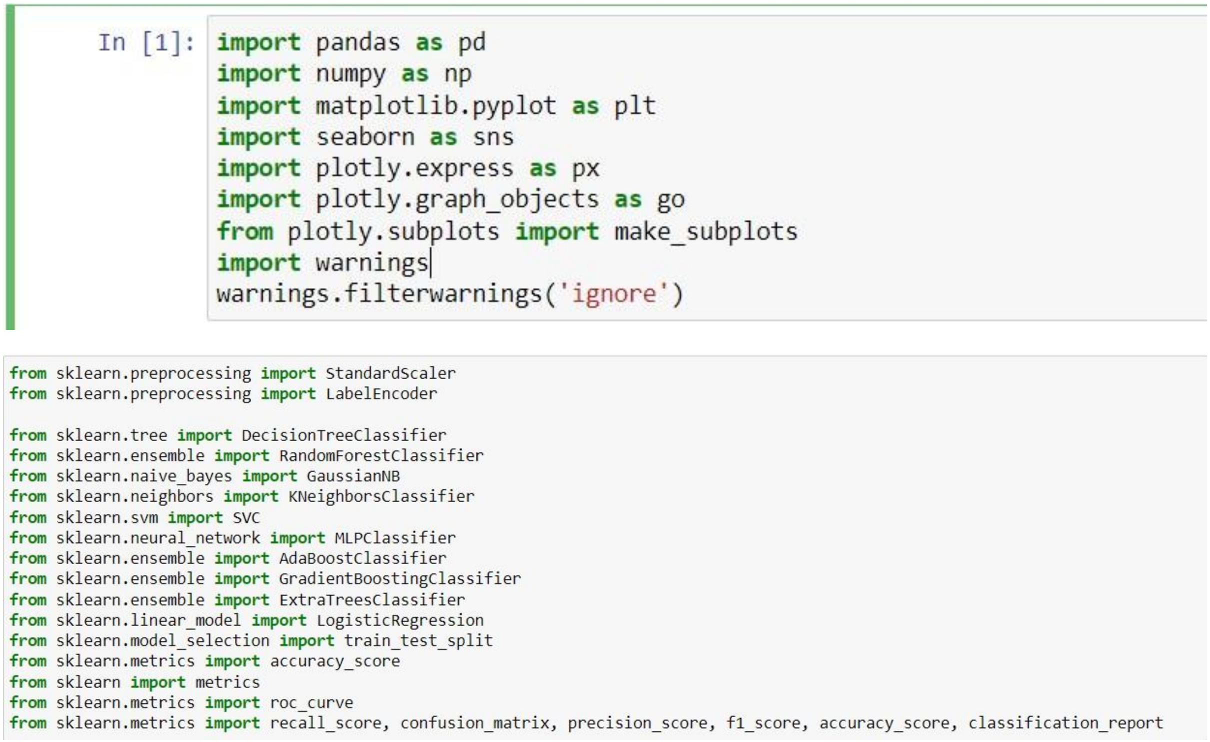
Plotly Graph Objects is a lower-level API in the Plotly ecosystem that provides more control and flexibility in creating visualizations. It allows you to build complex and custom plots using individual components like traces, layouts, and figures. By importing plotly.graph_objects as go, we can leverage its capabilities to create highly customizable visualizations.

plotly.subplots (from plotly.subplots import make_subplots):

Plotly Subplots is a module that enables the creation of multiple subplots within a single figure. It provides a convenient way to organize and compare multiple plots within a grid or other layout structures. warnings (import warnings):

The warnings module provides a mechanism to handle warning messages in Python. In the given code, warnings. filter warnings('ignore') suppresses warning messages, allowing the code to run without displaying warnings. This can be useful in scenarios where certain warnings are expected or not critical to the execution of the code.

These libraries are commonly used in data analysis and visualization tasks, offering a wide range of functions and capabilities to work with data and create meaningful visual representations.



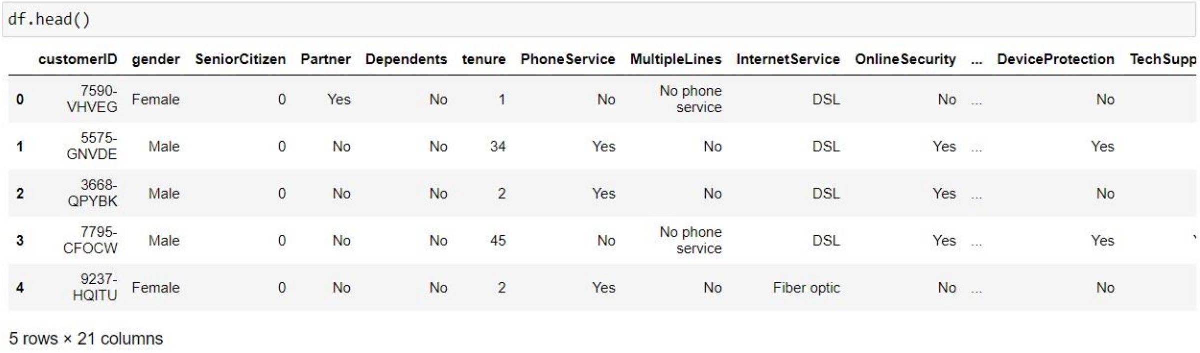
```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        import plotly.express as px
        import plotly.graph_objects as go
        from plotly.subplots import make_subplots
        import warnings
        warnings.filterwarnings('ignore')
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder

from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn import metrics
from sklearn.metrics import roc_curve
from sklearn.metrics import recall_score, confusion_matrix, precision_score, f1_score, accuracy_score, classification_report
```

Figure 2.1. Imported Libraries

2.2. Understanding about the data

Each column lists the characteristics of each customer represented by a row in the table.



```
df.head()
```

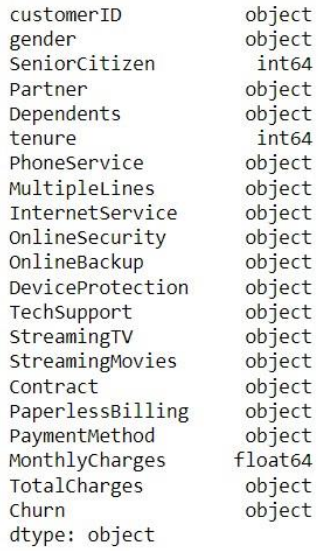
	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupp
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

5 rows × 21 columns

Figure 2.2. Data list

The data set includes information about:

- Customers who left within the last month – the column is called Churn.
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies.
- Customer account information - how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges.
- Demographic info about customers – gender, age range, and if they have partners and dependents.



```
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges         object
Churn                object
dtype: object
```

Figure 2.3. Data Type

The target is to predict the **Churn.**

2.3. Data Visualization

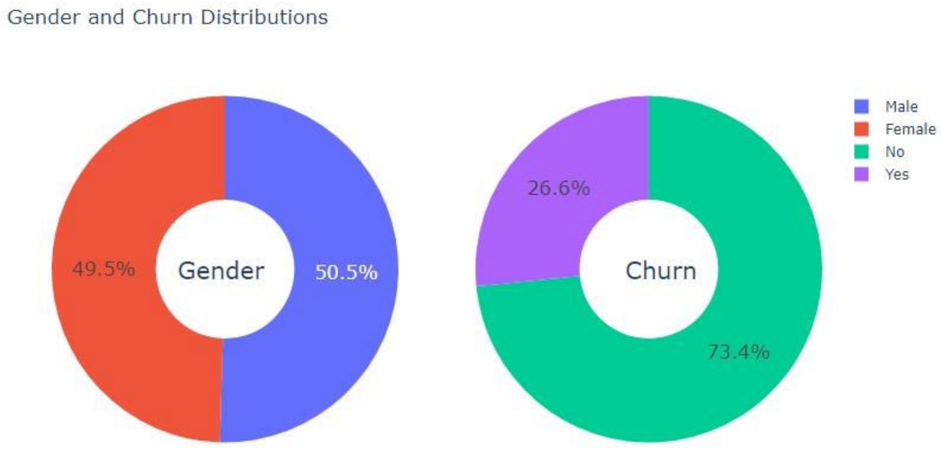


Figure 2.4. Gender and Churn Distribution

From the data as we can see costumers are mostly male with 50.5% and 49.5% female, and the costumers who decide to churn 26.6% switched to another firm.

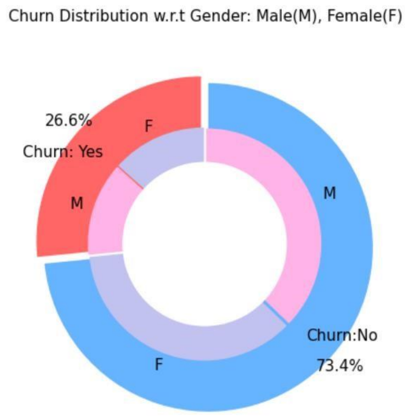


Figure 2.5. Churn Distribution

The percentage or number of customers that switched service providers differs just little. When it came to switching to a different service provider or organization, all genders acted similarly.

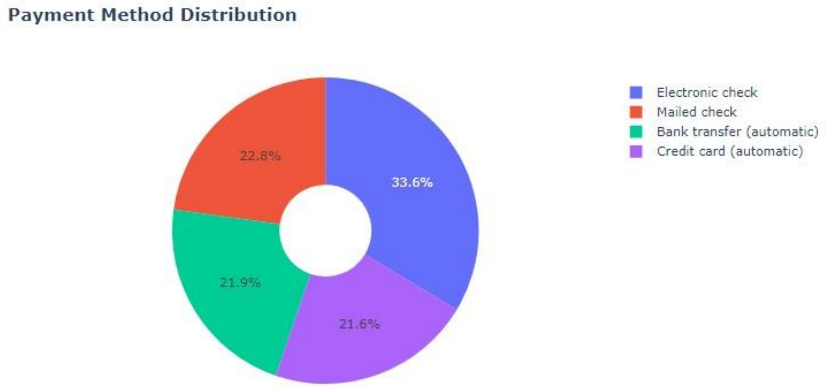


Figure 2.6. Payment Method Distribution

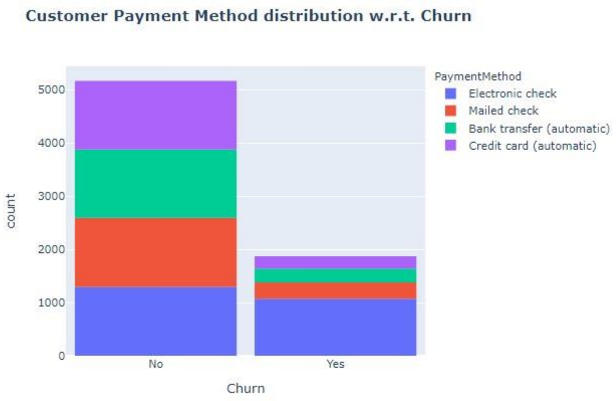


Figure 2.7. Costumer Payment Method Distribution with Churn

Major clients who departed used electronic checks as their preferred method of payment. Customers who selected mailed checks, credit-card automatic transfers, or bank transfers as their payment method were less likely to leave.

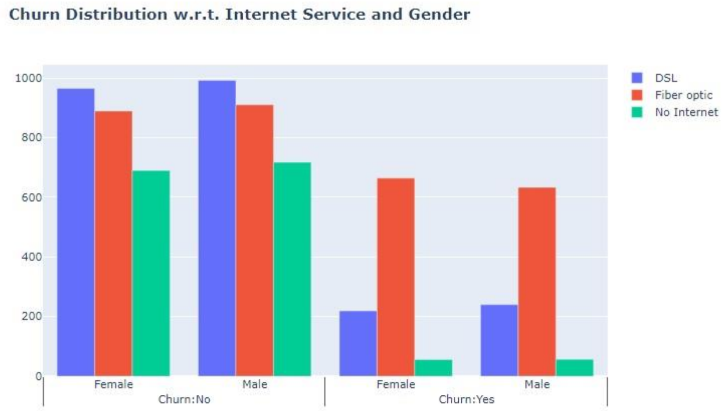


Figure 2.8. Churn Distribution with Internet Service and Gender

Numerous people pick the Fiber optic service, and it is also clear that these customers have a high turnover rate, which may indicate that they are not happy with this kind of internet service. The majority of customers have DSL service, which has a lower turnover rate than fiber optic service.

2.4.Data Pre-Processing

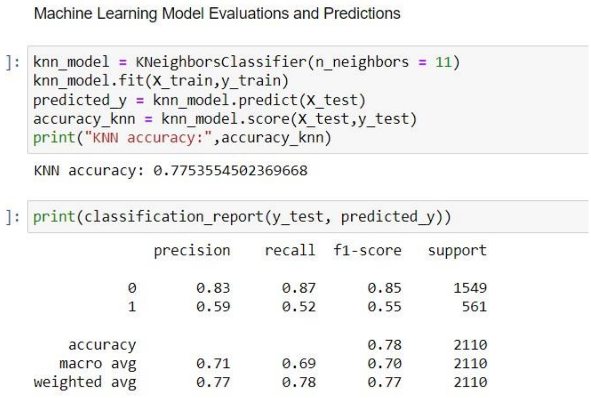
Data preparation is an essential stage in projects involving data analysis and machine learning. It entails converting unprocessed data into a format appropriate for analysis and modeling. Enhancing data quality, handling missing values, dealing with outliers, and making sure the data is compatible with the being used algorithms are the key objectives of data preparation. From this project used to the data into train and test and also standardize numeric attributes.

For the data pre-processing in this project we used to delete the rows with missing values since some the variable does not effect on the main data and in other hand to solve the problem of missing values in Total Charges column, we decided to fill it with the mean of Total Charges values.

CHAPTER 3 RESULT ANALYSIS AND DISCUSSION

3.1. Result

KNN Model

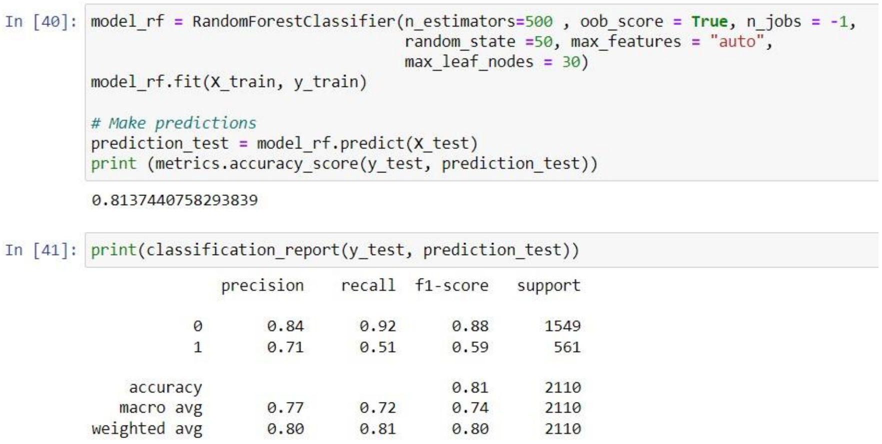


Machine Learning Model Evaluations and Predictions

```
]: knn_model = KNeighborsClassifier(n_neighbors = 11)
   knn_model.fit(X_train,y_train)
   predicted_y = knn_model.predict(X_test)
   accuracy_knn = knn_model.score(X_test,y_test)
   print("KNN accuracy:",accuracy_knn)

   KNN accuracy: 0.7753554502369668

]: print(classification_report(y_test, predicted_y))

                 precision    recall  f1-score   support

              0       0.83      0.87      0.85      1549
              1       0.59      0.52      0.55       561

       accuracy                           0.78      2110
      macro avg       0.71      0.69      0.70      2110
   weighted avg       0.77      0.78      0.77      2110
```

Figure 2.4. Result KNN Model

Random Forest Classifier Model



```
In [40]: model_rf = RandomForestClassifier(n_estimators=500 , oob_score = True, n_jobs = -1,
                                           random_state =50, max_features = "auto",
                                           max_leaf_nodes = 30)
         model_rf.fit(X_train, y_train)

         # Make predictions
         prediction_test = model_rf.predict(X_test)
         print (metrics.accuracy_score(y_test, prediction_test))

         0.8137440758293839

In [41]: print(classification_report(y_test, prediction_test))

                       precision    recall  f1-score   support

                    0       0.84      0.92      0.88      1549
                    1       0.71      0.51      0.59       561

             accuracy                           0.81      2110
            macro avg       0.77      0.72      0.74      2110
         weighted avg       0.80      0.81      0.80      2110
```

Figure 2.5. Result Random Forest Classifier Model

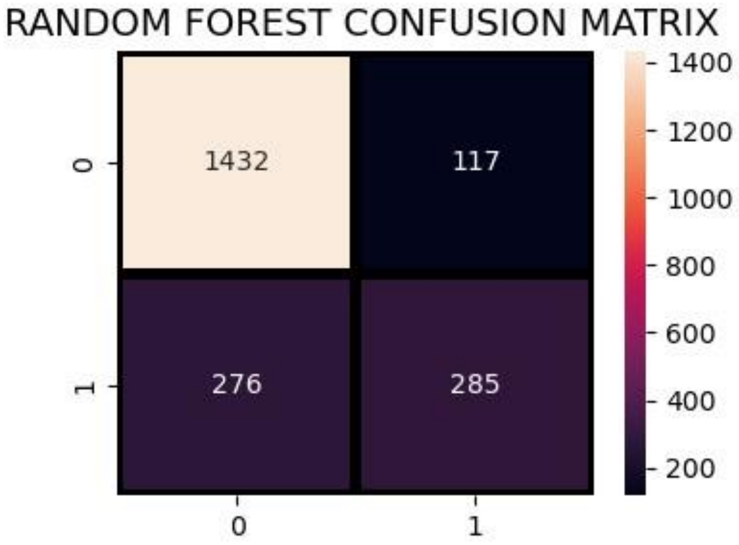


Figure 2.6. Result Random Forest Classifier Confusion Matrix

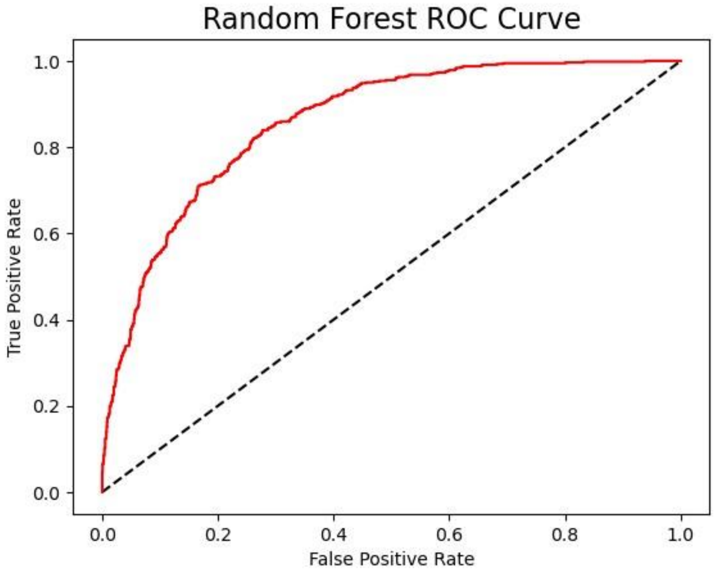
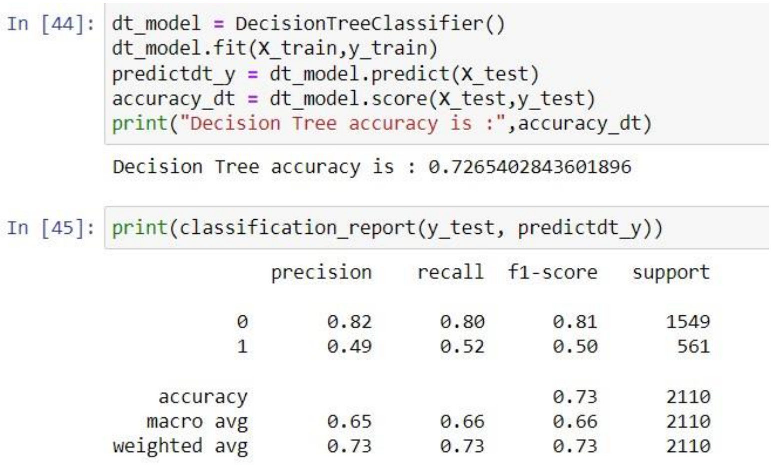


Figure 2.7. Result Random Forest Classifier ROC Curve

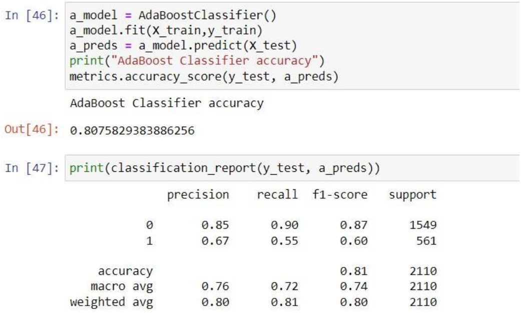
Decision Tree Model



```
In [44]: dt_model = DecisionTreeClassifier()
         dt_model.fit(X_train,y_train)
         predictdt_y = dt_model.predict(X_test)
         accuracy_dt = dt_model.score(X_test,y_test)
         print("Decision Tree accuracy is :",accuracy_dt)

         Decision Tree accuracy is : 0.7265402843601896

In [45]: print(classification_report(y_test, predictdt_y))

                       precision    recall  f1-score   support

                    0       0.82      0.80      0.81      1549
                    1       0.49      0.52      0.50       561

             accuracy                           0.73      2110
            macro avg       0.65      0.66      0.66      2110
         weighted avg       0.73      0.73      0.73      2110
```

Figure 2.8. Decision Tree Model

Ada Boost Classifier Model



```
In [46]: a_model = AdaBoostClassifier()
         a_model.fit(X_train,y_train)
         a_preds = a_model.predict(X_test)
         print("AdaBoost Classifier accuracy")
         metrics.accuracy_score(y_test, a_preds)

         AdaBoost Classifier accuracy

Out[46]: 0.8075829383886256

In [47]: print(classification_report(y_test, a_preds))

                       precision    recall  f1-score   support

                    0       0.85      0.90      0.87      1549
                    1       0.67      0.55      0.60       561

             accuracy                           0.81      2110
            macro avg       0.76      0.72      0.74      2110
         weighted avg       0.80      0.81      0.80      2110
```

Figure 2.9. Result Ada Boost Model

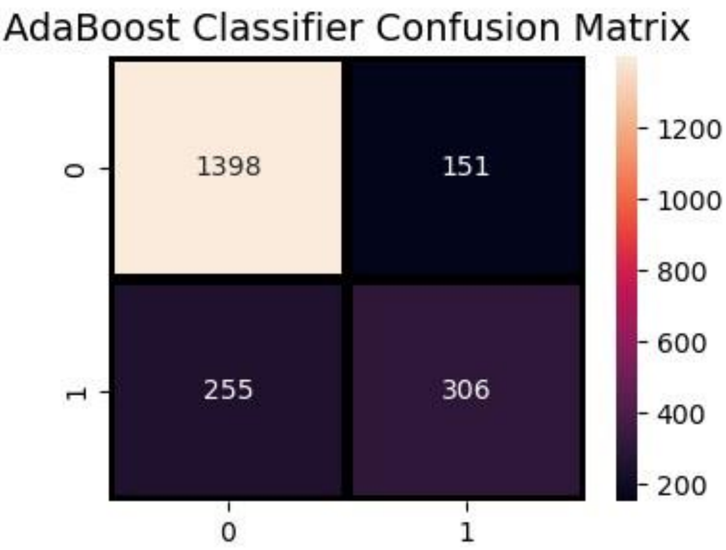


Figure 2.5. AdaBoost Classifier Confusion Matrix

3.2. Discussion

To assess the performance of churn prediction models, appropriate evaluation metrics are necessary. In this section, we discuss common evaluation metrics such as accuracy, precision, recall, F1-score, and area under the Receiver Operating Characteristic (ROC) curve. We also highlight the importance of cross-validation and other techniques to ensure reliable model performance evaluation.

Based on our evaluation metrics, the Random Forest Classifier Model algorithm outperformed other models in terms of accuracy and F1-score. The selected model achieved an accuracy of 0.8137440758293839 and an F1-score of 0.81 on the test dataset. The key features influencing customer churn were [insert feature names], indicating the significant factors that contribute to customer attrition.

CHAPTER 4 CLOSING

4.1. Future Improvements

We made use of the current set of features for prediction in this project. However, investigating additional feature engineering methods might help the models perform better. To extract more useful information, this may entail developing new features, altering already-existing features, or utilizing domain-specific knowledge.

Although we tested various individual models, including KNN, Random Forest, Decision Tree Classifier, and AdaBoost Classifier, more research into ensemble techniques may be useful. The predictability and robustness of the system may be improved by methods like stacking, which incorporate predictions from numerous models.

The predicted accuracy of the models could be increased by incorporating external data sources, such as customer sentiment analysis, social media data, or extra consumer activity data. By providing a more complete picture of the consumers and their interactions through this extended data, it is possible to make better predictions and develop retention tactics.

Customer preferences and behavior are subject to change throughout time. Establishing a procedure for routine model retraining utilizing up-to-date data is crucial for ensuring the model's accuracy and applicability. The model can adapt to shifting patterns and keep its predictive power by being frequently retrained.

4.2. Conclusion

Customer churn has a negative impact on a company's profitability. There are numerous tactics that can be used to reduce client churn. Knowing a company's customers well is the best strategy to prevent customer churn. This entails identifying clients who run the danger of leaving and trying to increase their contentment. Naturally, the primary priority for solving this problem is to improve customer service. Another tactic to lower customer churn is to increase customer loyalty through meaningful experiences and personalized service. To take proactive measures to prevent future customer churn, several companies conduct surveys of existing customers to learn why they left.

REFERENCES

Telecom Churn Dataset 2022, retrieved from Kaggle: https://www.kaggle.com/datasets/blastchar/telco-customer-churn?resource=download

B. Huang, M. T. Kechadi, and B. Buckley, “Customer churn prediction in telecommunications,” *Expert Syst. Appl.*, vol. 39, no. 1, pp. 1414–1425, 2012.

K. Dahiya and S. Bhatia, “Customer churn analysis in telecom industry,” in *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions)*, IEEE, 2015, pp. 1–6.

K. Ahmad, A. Jafar, and K. Aljoumaa, “Customer churn prediction in telecom using machine learning in big data platform,” *J. Big Data*, vol. 6, no. 1, pp. 1–24, 2019.

N. Hashmi, N. A. Butt, and M. Iqbal, “Customer churn prediction in telecommunication a decade review and classification,” *Int. J. Comput. Sci. Issues*, vol. 10, no. 5, p. 271, 2013.