Jaypee Institute of Information Technology, Noida

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING AND INFORMATION TECHNOLOGY

Project Title: Customer Retention Analysis

Enroll. No.	**Name of Student**
9921103156	Aryan Patel

Course Name: Introduction to Big Data & Data Analytics
Course Code: 20B12CS333
Program: B. Tech. CS&E
3nd Year 5th Sem

2023 - 2024

1. Abstract:

In the dynamic landscape of the telecommunications industry, customer retention stands as a linchpin for sustained success. This project delves into the intricate world of customer churn analysis, employing state-of-the-art machine learning models to decipher patterns and unveil insights that can redefine how businesses approach customer relationships.

The datasets under scrutiny, "customer_churn_dataset-testing-master.csv" and "customer_churn_dataset-training-master.csv," sourced from Kaggle, serves as the bedrock for a comprehensive exploration. By investigating the interplay of twelve distinct features, ranging from the temporal tenure of customers to the nuances of their last interactions, our project not only seeks to predict churn but endeavors to illuminate the strategic pathways companies can traverse to fortify customer loyalty.

As a distinctive facet, this endeavor extends beyond prediction; it orchestrates a comparative analysis of multiple machine learning models. Linear regression, decision trees, random forests, naive Bayes, and XGBoost vie for prominence, and their results and accuracies stand as signposts in this journey. The fusion of predictive modeling, interpretability, and meticulous model comparison offers a roadmap for businesses to navigate the intricate terrain of customer relations, fostering a proactive approach to churn prevention.

What is Customer Churn?

The customer churn analysis can be defined as analytical work carried out on the possibility of a customer leaving a product or service. In its simplest definition, it means that customers are abandoned to choose the company because of competition [1]. The purpose is to identify this situation before leaving the customer's product or service, and then to carry out some preventive actions.

2. <u>Introduction</u>

Customer retention is the most important asset for any business as it is stated that “the cost of acquiring a new customer can be higher than that of retaining a customer by as much as 700%; increasing customer retention rates by a mere 5% could increase profits by 25% to 95%”. So one of the best solution to retain the customers is to reduce churn rate, where “churn” means moving the customer from service provider to another one, or stopping using specific services over specific periods for many reasons that can be detected previously if the company analyzes its data records and uses machine learning technology which enables the companies to predict the customers who are likely to churn. A lot of studies approved its efficiency to this situation so the company can respond quickly to the behavioral changes in the customer’s minds. Telco’s today is refining & optimizing the customer experience which is the key to sustaining a market differentiation and reducing churn, where retaining an existing customer costs much lower than acquiring a new one. This research studies the machine learning algorithms and recommended the best solutions for telecoms. In the competitive telecom sector, customers can easily switch from one provider to another, which leaves the telecom providers worried about their customers and how to retain them but they can predict the customers who will move to another provider previously by analyzing their behavior. They can retain them by providing offers and their preferred services according to their historical records so the aim of this study is to predict churn previously and detect the main factors that may let the user move to another provider in telecoms.

2.1 Problem Statement

In the dynamic landscape of the phone industry, customer retention is not merely a numerical challenge but a critical issue that significantly impacts a company's revenue and growth potential. Amidst a plethora of phone options and plans, ensuring customer satisfaction becomes an arduous task. The central challenge revolves around devising strategies to prevent customer attrition.

2.2 Motivation

Our motivation stems from viewing customer churn not only as a problem in need of resolution but as a substantial opportunity for companies to thrive. Given the constant evolution in phones and technology, it is imperative for companies to comprehend the reasons driving customer exits. By unraveling the intricate correlation between customer behavior and potential reasons for departure, companies can adopt a more intelligent approach to maintaining customer satisfaction.

2.3 Objective

Our primary objective goes beyond mere predictions of customer departure; it is about providing companies with astute insights to keep their customer base content. Leveraging the power of computational analysis, we aim to transform complex and chaotic data into a comprehensive guide that assists companies in navigating the turbulent waters of customer satisfaction. Also comparing various classification machine learning Algorithms like Naive Bayesian, Logistic Regression Decision Tree, Random Forest, Xtreme Grade Boosting.

2.4 Contribution

This project aspires to serve as a guidebook for companies, offering insights into the reasons behind customer churn. It transcends mere speculation, aiming to furnish companies with actionable knowledge. Our goal is to transform a problem into an avenue for growth, resilience, and delighted customers. Moreover, we are not confining our efforts to prediction alone; we are actively comparing various machine learning algorithms to determine their efficacy. This comparative analysis serves as a roadmap for companies, guiding them towards the most effective strategies to prevent customer attrition in the ever-evolving realm of phones and connections.

3. Detailed Description of the Project:

3.1 Proposed Work:

Our project embarks on a journey to decode the intricacies of customer churn in the telecommunications domain. Leveraging advanced data analysis and machine learning techniques, we aim to unearth patterns, relationships, and insights that empower businesses to proactively address customer churn.

Datasets Used:

For this ambitious undertaking, we turn to two key datasets: "customer_churn_dataset-testing-master.csv" and "customer_churn_dataset-training-master.csv," sourced from Kaggle. These datasets serve as our treasure troves of information, holding the keys to understanding customer behavior in the telecom world.

The datasets encompass 12 crucial feature columns, each providing a unique glimpse into the factors influencing customer churn:

- CustomerID **:** A unique identifier for each customer, allowing us to distinguish and track individual customer journeys.
- Age: The age of the customer, offering insights into the demographic distribution of the customer base.
- Gender: Gender information, helping us explore if gender plays a role in customer churn.
- Tenure: The duration in months for which a customer has been using the company's products or services, indicating customer loyalty.
- Usage Frequency: The number of times that the customer has used the company's services in the last month, showcasing engagement levels.
- Support Calls: The number of calls that the customer has made to customer support in the last month, reflecting customer satisfaction or issues.
- Payment Delay: The number of days that the customer has delayed their payment in the last month, a potential indicator of financial strain or dissatisfaction.
- Subscription Type: The type of subscription the customer has chosen, influencing service offerings and customer expectations.
- Contract Length: The duration of the contract that the customer has signed with the company, providing insights into commitment levels.

- Total Spend: The total amount of money the customer has spent on the company's products or services, a critical metric for revenue analysis.
- Last Interaction: The number of days since the last interaction that the customer had with the company, helping us understand customer engagement patterns.
- Churn: A binary label indicating whether a customer has churned (1) or not (0), our focal point for predictive modeling.

Data Visualization:

Complementing our rigorous data analysis is a comprehensive data visualization strategy. We recognize the power of visual representation in uncovering patterns and trends that might be less apparent in raw data. Utilizing tools such as matplotlib and seaborn, we have crafted a visual narrative to enhance our understanding and convey complex insights effectively.

- **Univariate Plots:**

  Histograms for age and tenure to understand the distribution of customer demographics and loyalty.
  Bar charts for categorical features like gender and subscription type, offering a visual grasp of their prevalence.

- **Multivariate Plots:**

  Scatter plots to explore relationships between variables such as total spend and usage frequency.
  Pair plots for a holistic view of feature interactions, aiding in the identification of potential correlations.

- **Continuous-Discrete Analysis:**

  Box plots to highlight payment delay patterns and their potential impact on churn.
  Line plots for visualizing trends in usage frequency and support calls over time.
  These visualizations serve not only as descriptive tools but also as interpretive aids, allowing stakeholders to grasp complex relationships intuitively.

Predictive Modeling:

With a solid foundation laid by data analysis and visualization, we transition to the predictive modeling phase. Here, we employ various machine learning models to predict customer churn based on the insights gleaned.

- **Logistic Regression:**

  Logistic Regression is a versatile algorithm used for binary classification tasks. It models the probability of an instance belonging to a particular class, making it ideal for scenarios where the outcome is dichotomous. Its simplicity and efficiency make it a go-to choice when analyzing relationships between independent variables and a binary outcome.

- **Naive Bayes:**

  Naive Bayes is a probabilistic algorithm based on Bayes' theorem. It assumes independence between features, simplifying computations and making it efficient for high-dimensional data. It is particularly useful for text classification and spam filtering, showcasing its adaptability to scenarios where feature independence is a reasonable assumption.

- **Decision Tree:**

  Decision Trees are intuitive models that recursively split data based on feature values. They are employed in various tasks such as classification and regression. Decision Trees are advantageous for their visual interpretability, making them suitable for scenarios where understanding the decision-making process is crucial.

- **Random Forest:**

  Random Forest is an ensemble learning method that builds multiple decision trees and combines their predictions. It excels in improving predictive performance and reducing overfitting, making it valuable for complex tasks. Random Forest is widely used in scenarios where capturing intricate relationships within the data is essential.

- **XGBoost (Extreme Gradient Boosting):**

  XGBoost is a powerful gradient boosting algorithm known for its high performance. It sequentially builds a series of weak learners, boosting overall predictive accuracy. Its adaptability to different data types and robustness against overfitting contribute to its popularity in various machine learning competitions and real-world applications.

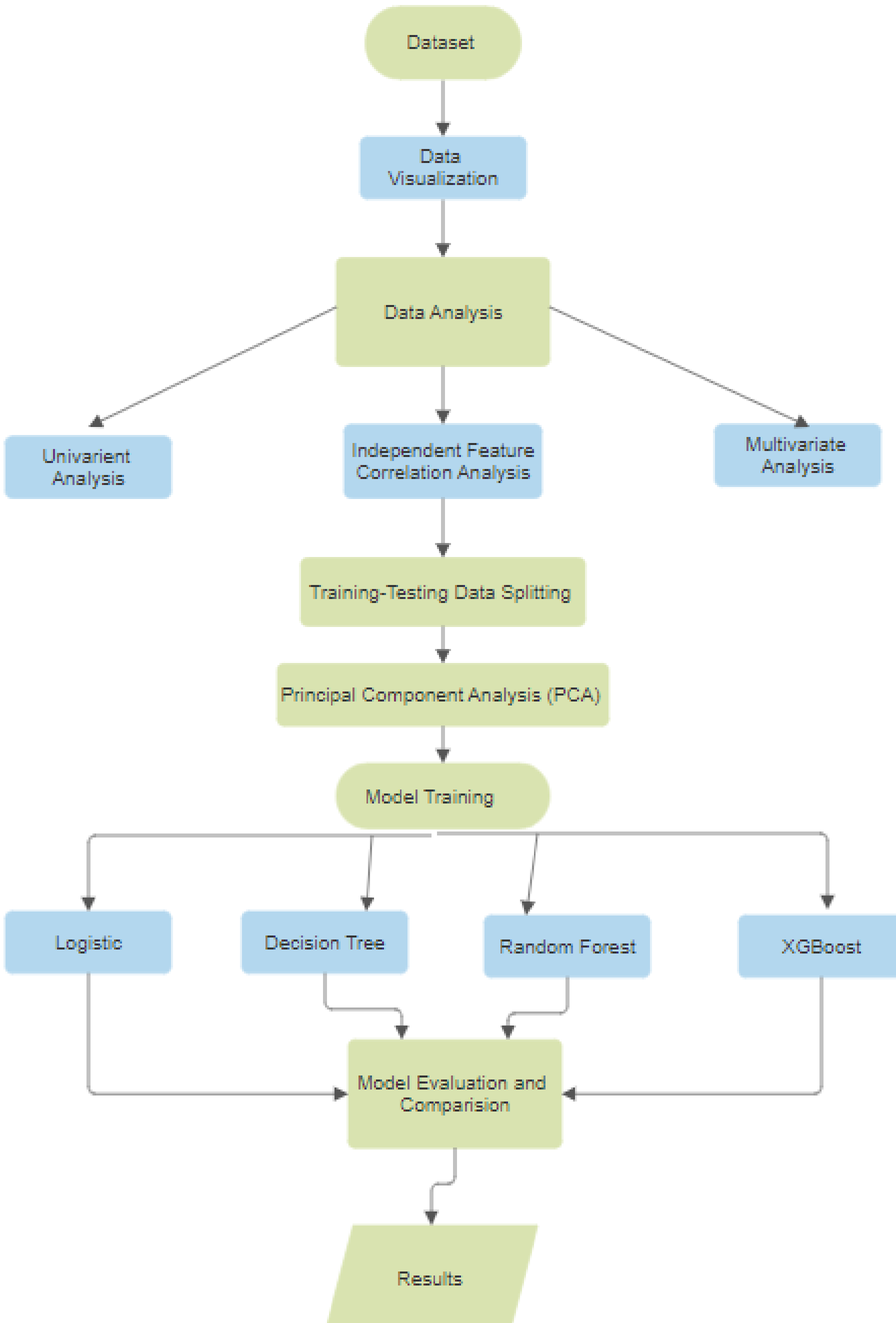
Model Training:

Each model is trained using the training dataset, allowing it to learn patterns and relationships within the data.

Model Comparison:

Comparative analysis is conducted to identify the model that exhibits superior predictive capabilities for customer churn.

3.2 WorkFlow Diagram



4. Implementation:

4.1. Program Code:

```
import numpy as np
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix,
classification_report

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import OneHotEncoder

import pickle

from sklearn.decomposition import PCA

from sklearn.linear_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

import xgboost as xgb
df = pd.concat(
    [
        pd.read_csv('./customer_churn_dataset-training-master.csv'),
        pd.read_csv('./customer_churn_dataset-testing-master.csv')
    ],
    axis=0)
df.reset_index(drop=True, inplace=True)
df.drop(columns='CustomerID', inplace=True)

df.columns = [col.lower().replace(' ', '_') for col in df.columns]
descrete_col = ['age', 'tenure', 'usage_frequency', 'support_calls', 'payment_delay',
'last_interaction', 'churn']
for col in descrete_col:
    df[col] = df[col].astype(int)
df
def make_histogram(df, target_feature, bins = 10, custom_ticks=None, unit='', additional=''):
    plt.figure(figsize=(10, 5))
```

```
    plt.hist(df[target_feature], bins=bins)
    if custom_ticks is not None:
        plt.xticks(custom_ticks)
    plt.ylabel('Count')
    plt.xlabel(target_feature)
    plt.title(f"Distribution of {target_feature.lower()}{additional}:\n")
    plt.grid()
    plt.show()
    print(f"Distribution of {target_feature.lower()}{additional}: {df[target_feature].mean():.2f} ± {df[target_feature].median():.2f} {unit}\nMedian: {df[target_feature].median():.2f} {unit}\nMinimum: {df[target_feature].min()} {unit}\nMaximum: {df[target_feature].max()} {unit}\n{df[target_feature].skew():.3f} Skewness\n")

def make_piechart(df, target_feature, additional=''):
    dict_of_val_counts = dict(df[target_feature].value_counts())
    data = list(dict_of_val_counts.values())
    keys = list(dict_of_val_counts.keys())

    palette_color = sns.color_palette('bright')
    plt.pie(data, labels=keys, colors=palette_color, autopct='%.0f%%')
    plt.title(f"Distribution of Cutomer's {target_feature}:")
    plt.show()
    print_str = f"Distribution of cutomer's {target_feature.lower()}{additional}:"
    for k, v in zip(keys, data):
        print_str += f"\n{v} {k}"
    print(print_str)

def make_barplot(df, target_feature, custom_ticks=None, unit='', additional=''):
    plt.figure(figsize=(10, 5))
    dict_of_val_counts = dict(df[target_feature].value_counts())
    data = list(dict_of_val_counts.values())
    keys = list(dict_of_val_counts.keys())
    plt.bar(keys, data)
    if custom_ticks is not None:
        plt.xticks(custom_ticks)
    plt.xlabel(f'{target_feature.capitalize()}{additional}')
    plt.ylabel('Frequency')
    plt.title(f"Distribution of cutomer's {target_feature.lower()}{additional}\n")
    plt.grid(axis='y')
    plt.show()
    print(f"Distribution of cutomer's {target_feature.lower()}{additional}: {df[target_feature].mean():.2f} ± {df[target_feature].median():.2f} {unit}\nMedian: {df[target_feature].median():.2f} {unit}\nMinimum: {df[target_feature].min()} {unit}\nMaximum: {df[target_feature].max()} {unit}\n\n{df[target_feature].skew():.3f} Skewness\n")
```

```
def make_boxplot(df, feature):
    plt.figure(figsize=(10,5))
    sns.boxplot(df, x=feature)
    plt.title(f"Boxplot of {feature}\n")
    plt.xlabel(feature)
    plt.ylabel("Values")
    plt.show()
make_piechart(df, 'gender')
make_piechart(df, 'subscription_type')
filtered = df.copy()
filtered['churn_category'] = ['Churn' if x == 1.0 else 'Not Churned' for x in df['churn']]
make_piechart(filtered, 'churn_category')
make_barplot(df, 'age', custom_ticks=np.arange(0, 66, 5), additional=' (years)', unit='years')
gender_churn = df.groupby(['gender', 'churn']).size().unstack()

X = list(gender_churn.index)
churn_0 = list(gender_churn.iloc[:, 0])
churn_1 = list(gender_churn.iloc[:, 1])

X_axis = np.arange(len(X))

plt.bar(X_axis - 0.2, churn_1, 0.4, label = 'Churn')
plt.bar(X_axis + 0.2, churn_0, 0.4, label = 'Not Churn')

plt.xticks(X_axis, X)
plt.xlabel('Gender')
plt.ylabel('Count')
plt.title("Gender wise churn rate")
plt.legend(loc='center right')
plt.grid(axis='y')
plt.show()
filtered = df.groupby(['payment_delay', 'churn']).size().unstack()

X = list(filtered.index)
churn_0 = list(filtered.iloc[:, 0])
churn_1 = list(filtered.iloc[:, 1])

X_axis = np.arange(len(X))

plt.bar(X_axis - 0.2, churn_1, 0.4, label = 'Churn')
plt.bar(X_axis + 0.2, churn_0, 0.4, label = 'Not Churn')

plt.xticks(X_axis, X, rotation=90)
plt.xlabel("Customer payment delays in days")
plt.ylabel('Count')
plt.title("Churn rate based on payment delays")
```

```
plt.legend(loc='center right')
plt.grid(axis='y')
plt.show()

y = df['churn']
X = df.drop(columns='churn')
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=23)

# Reset the index of the resulting DataFrames
X_train.reset_index(drop=True, inplace=True)
X_test.reset_index(drop=True, inplace=True)
y_train.reset_index(drop=True, inplace=True)
y_test.reset_index(drop=True, inplace=True)

def validate_test_data_categorical_columns(train_df, test_df):
  # Get the list of categorical columns for both train and test DataFrames
  train_df_categorical_columns = train_df.select_dtypes(include=['object', 'category']).columns.tolist()
  test_df_categorical_columns = test_df.select_dtypes(include=['object', 'category']).columns.tolist()

  # Check if the number of categorical columns is the same in both DataFrames
  if len(set(train_df_categorical_columns).intersection(set(test_df_categorical_columns))) == 0:
    print('Train and test dataframes have different categorical columns')
    return
  else:
    for cat_col in test_df_categorical_columns:
      # Create sets of unique values for the current categorical column in both DataFrames
      train_col = set(x for x in train_df[cat_col].unique().tolist() if not pd.isna(x))
      test_col = set(x for x in test_df[cat_col].unique().tolist() if not pd.isna(x))

      # Check if the sets are not equal, indicating different unique values
      if train_col != test_col:
        print(f'{cat_col} column has different unique values in train and test data:')
        print(f'Unique values in train data: {train_col}')
        print(f'Unique values in test data: {test_col}')
        return

    print('All categorical columns have consistent unique values in train and test data.')
    return

validate_test_data_categorical_columns(X_train, X_test)
encoder = OneHotEncoder(sparse_output=False)

encoder.fit(X_train[['gender', 'subscription_type', 'contract_length']])
```

```
feature_names = encoder.get_feature_names_out(['gender', 'subscription_type',
'contract_length'])
feature_names
train_categorical_one_encoded_data = encoder.transform(X_train[['gender', 'subscription_type',
'contract_length']])
train_OHE_df = pd.DataFrame(train_categorical_one_encoded_data, columns=feature_names)

test_categorical_one_encoded_data = encoder.transform(X_test[['gender', 'subscription_type',
'contract_length']])
test_OHE_df = pd.DataFrame(test_categorical_one_encoded_data, columns=feature_names)
X_train = X_train.drop(columns=['gender', 'subscription_type', 'contract_length'])
X_test = X_test.drop(columns=['gender', 'subscription_type', 'contract_length'])
X_train = pd.concat([X_train, train_OHE_df], axis=1)
X_test = pd.concat([X_test, test_OHE_df], axis=1)
with open('encoder.pkl', 'wb') as file:
    pickle.dump(encoder, file)
def print_evaluation_metrics(y_true, y_pred):
    accuracy = accuracy_score(y_true, y_pred)
    precision = precision_score(y_true, y_pred)
    recall = recall_score(y_true, y_pred)

    print(f"Accuracy: {accuracy:.2f}")
    print(f"Precision: {precision:.2f}")
    print(f"Recall: {recall:.2f}")
    print()

    conf_matrix = confusion_matrix(y_true, y_pred)
    print("Confusion Matrix:")
    print(conf_matrix)
    print()

    class_report = classification_report(y_true, y_pred)
    print("Classification Report:")
    print(class_report)

y_pred = naive_classifier.predict(X_test)

print_evaluation_metrics(y_test, y_pred)
y_pred = logistic_classifier.predict(X_test)

print_evaluation_metrics(y_test, y_pred)
# Testing decision trees

y_pred = decision_tree_classifier.predict(X_test)

print_evaluation_metrics(y_test, y_pred)
```

```
# Testing random forest

y_pred = random_forest_classifier.predict(X_test)

print_evaluation_metrics(y_test, y_pred)
# Testing xgboost

y_pred = xgb_classifier.predict(X_test)

print_evaluation_metrics(y_test, y_pred)
with open("random_forest_model.pkl", 'wb') as model_file:
   pickle.dump(random_forest_classifier, model_file)
with open("xgb_model.pkl", 'wb') as model_file:
   pickle.dump(xgb_classifier, model_file)
with open("decision_model.pkl", 'wb') as model_file:
   pickle.dump(decision_tree_classifier, model_file)
with open("logistic_model.pkl", 'wb') as model_file:
   pickle.dump(logistic_classifier, model_file)
class CustomerChurnClassifier:

   def __init__(self, model_path, encoder_path):

      with open(model_path, 'rb') as file:
         self.model = pickle.load(file)

      with open(encoder_path, 'rb') as file:
         self.encoder = pickle.load(file)

    def predict(self, age: int, tenure: int, usage_frequency: int, support_calls: int, payment_delay:
int, total_spend: float, last_interaction: int, gender: str, subscription_type: str, contract_length:
str):


      expected_data_types = [int, int, int, int, int, float, int, str, str, str]
              input_arguments = [age, tenure, usage_frequency, support_calls, payment_delay,
total_spend, last_interaction, gender, subscription_type, contract_length]
                  input_arguments_names = ['age', 'tenure', 'usage_frequency', 'support_calls',
'payment_delay', 'total_spend', 'last_interaction', 'gender', 'subscription_type', 'contract_length']

      for i in range(len(input_arguments)):
         current_arg_type = type(input_arguments[i])

         if current_arg_type != expected_data_types[i]:
                                  raise TypeError(f"Error: Given {input_arguments_names[i]}
({current_arg_type.__name__}) is not of the expected type
({expected_data_types[i].__name__}).")
```

```
        valid_genders = ['Female', 'Male']
        valid_subscription_types = ['Standard', 'Basic', 'Premium']
        valid_contract_lengths = ['Annual', 'Monthly', 'Quarterly']

        if gender not in valid_genders:
                    raise ValueError(f"Error: Invalid gender value '{gender}'. Expected one of
{valid_genders}.")

        if subscription_type not in valid_subscription_types:
            raise ValueError(f"Error: Invalid subscription_type value '{subscription_type}'. Expected
one of {valid_subscription_types}.")

        if contract_length not in valid_contract_lengths:
            raise ValueError(f"Error: Invalid contract_length value '{contract_length}'. Expected one
of {valid_contract_lengths}.")
        ohe_data = list(self.encoder.transform([[gender, subscription_type, contract_length]])[0])

                to_predict_array = [age, tenure, usage_frequency, support_calls, payment_delay,
total_spend, last_interaction] + ohe_data
        to_predict_array = np.array(to_predict_array).reshape((1, -1))

        prediction = self.model.predict(to_predict_array)[0]

        if prediction > 0.5:
            return 'Will Churn'
        else:
            return "Won't Churn"
customer_churn = CustomerChurnClassifier(
    model_path = 'random_forest_model.pkl',
    encoder_path = 'encoder.pkl'
)
customer_churn.predict(
    age = 22,
    tenure = 28,
    usage_frequency = 28,
    support_calls = 10,
    payment_delay = 13,
    total_spend = 584.0,
    last_interaction = 20,
    gender = 'Female',
    subscription_type = 'Standard',
    contract_length = 'Monthly'
)
```

4.2. Results Analysis:

4.2.1 Outputs:

Figure 1.

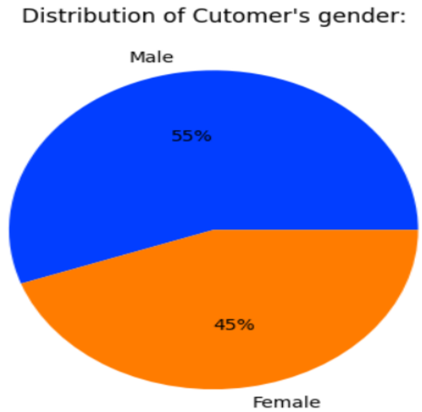


Figure 2.

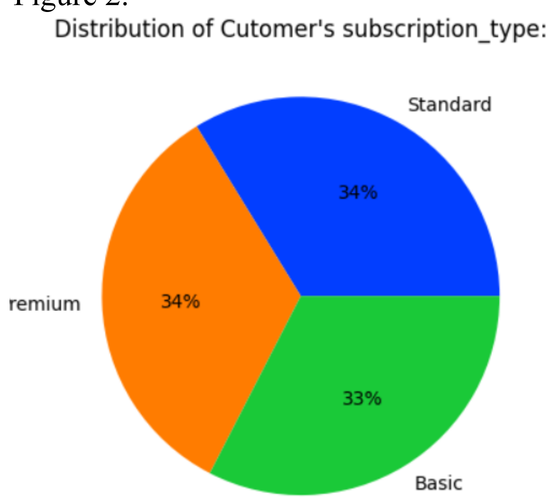


Figure 3.

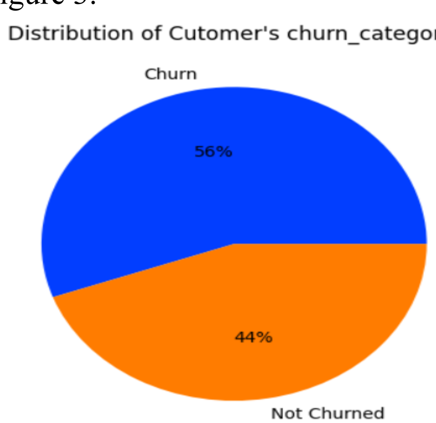


Figure 4.

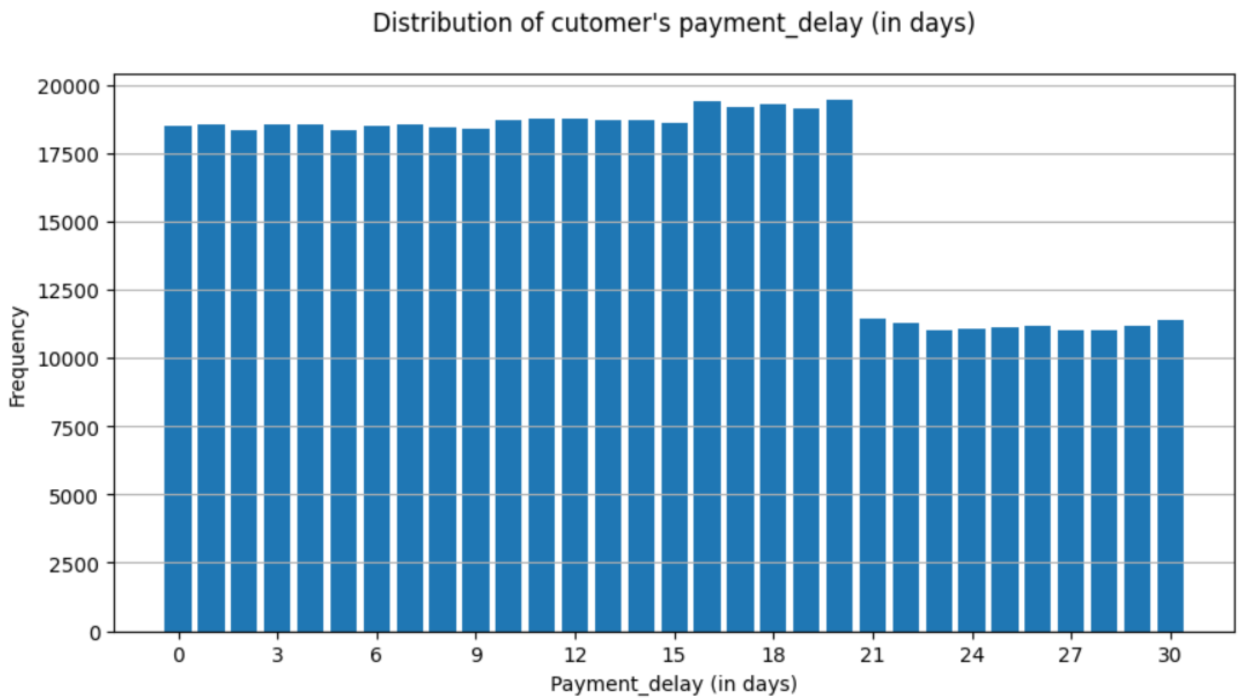


Figure 5.

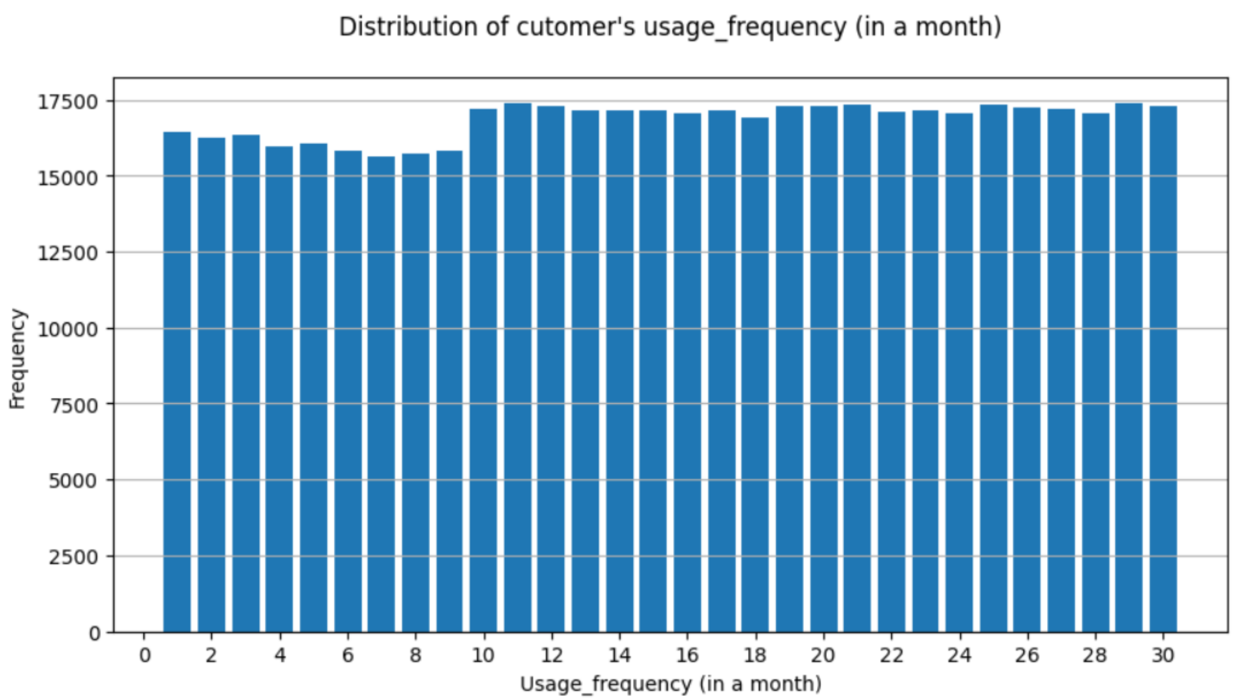


Figure 6.

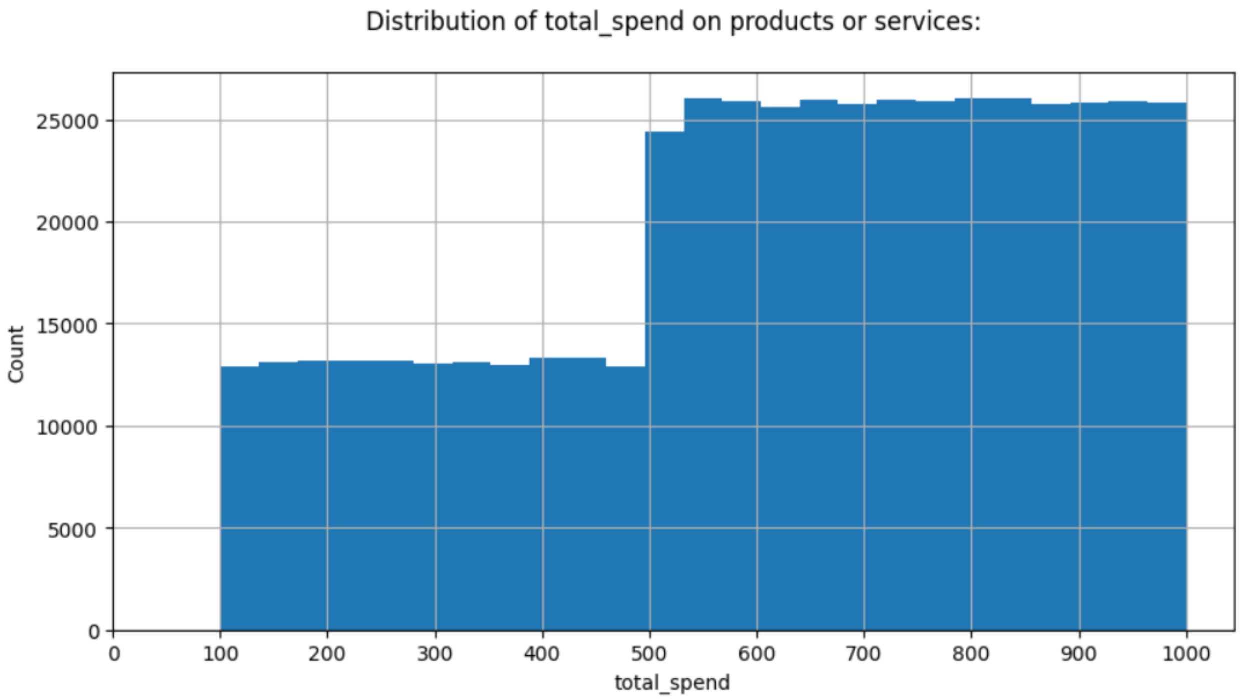


Figure 7.

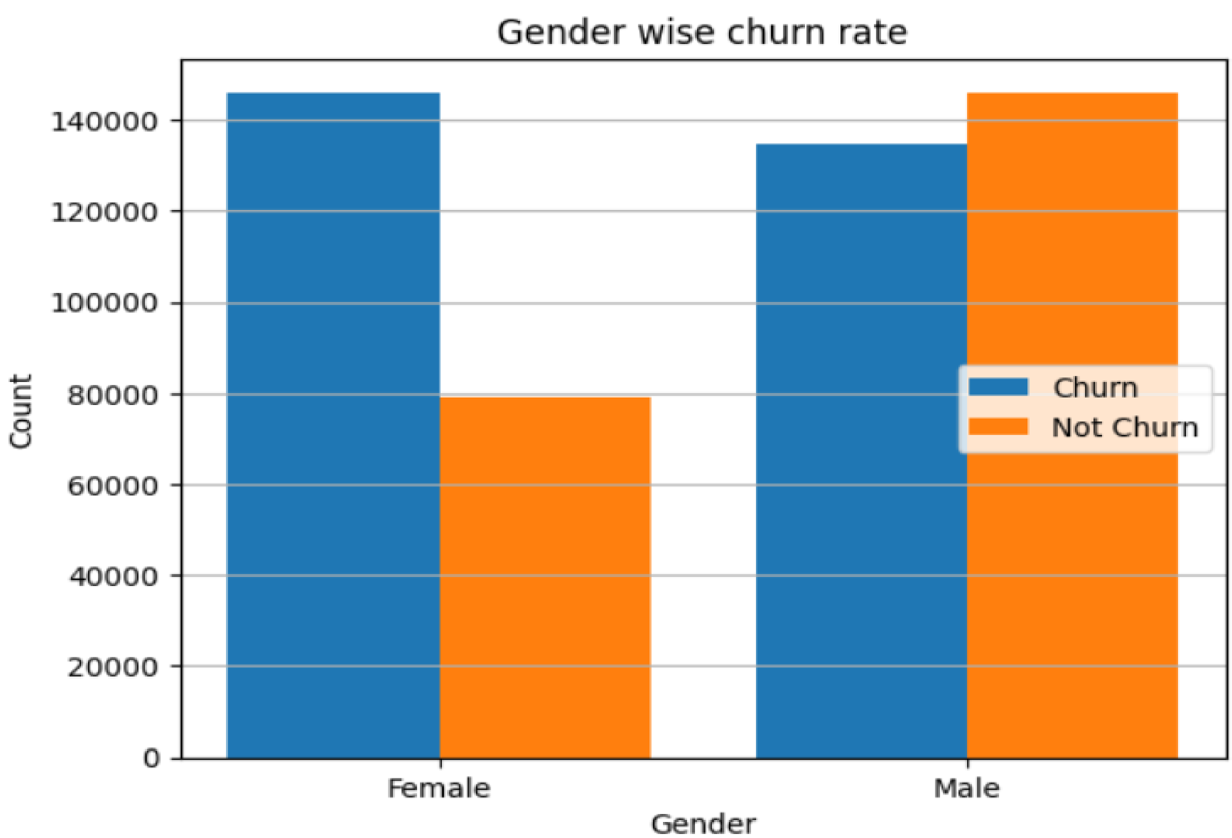


Figure 8.

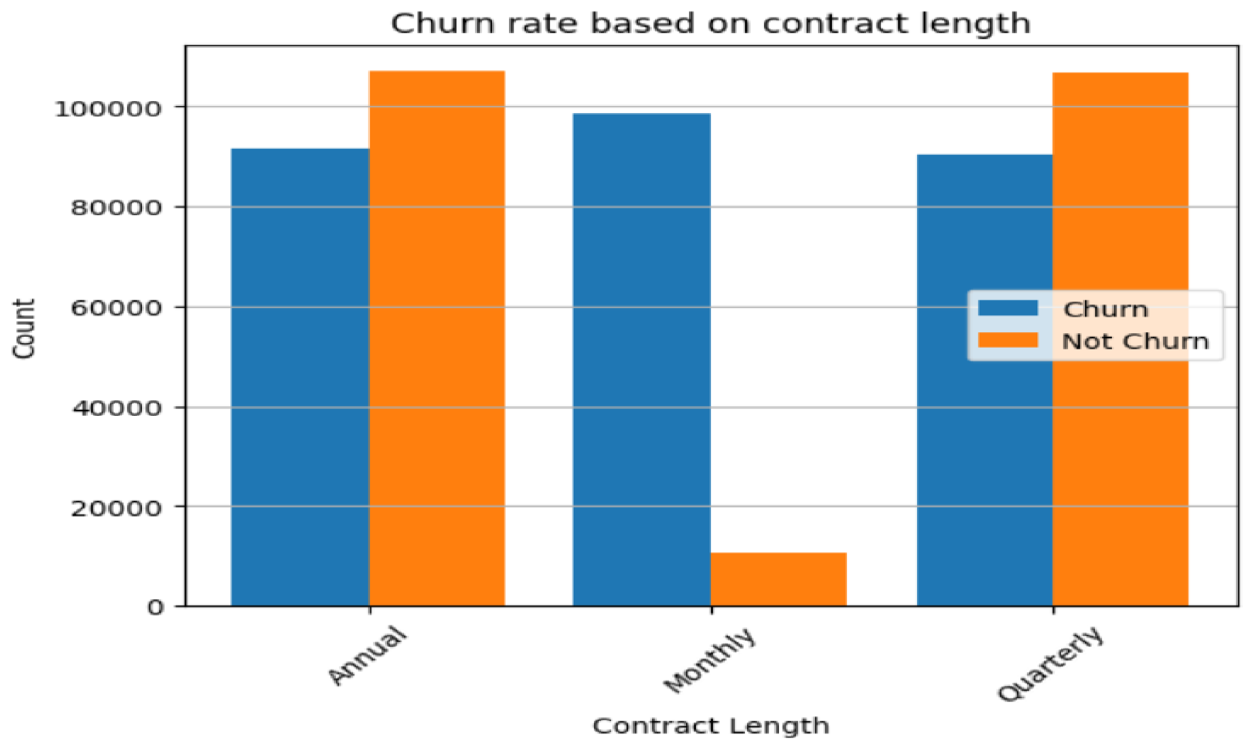


Figure 9.

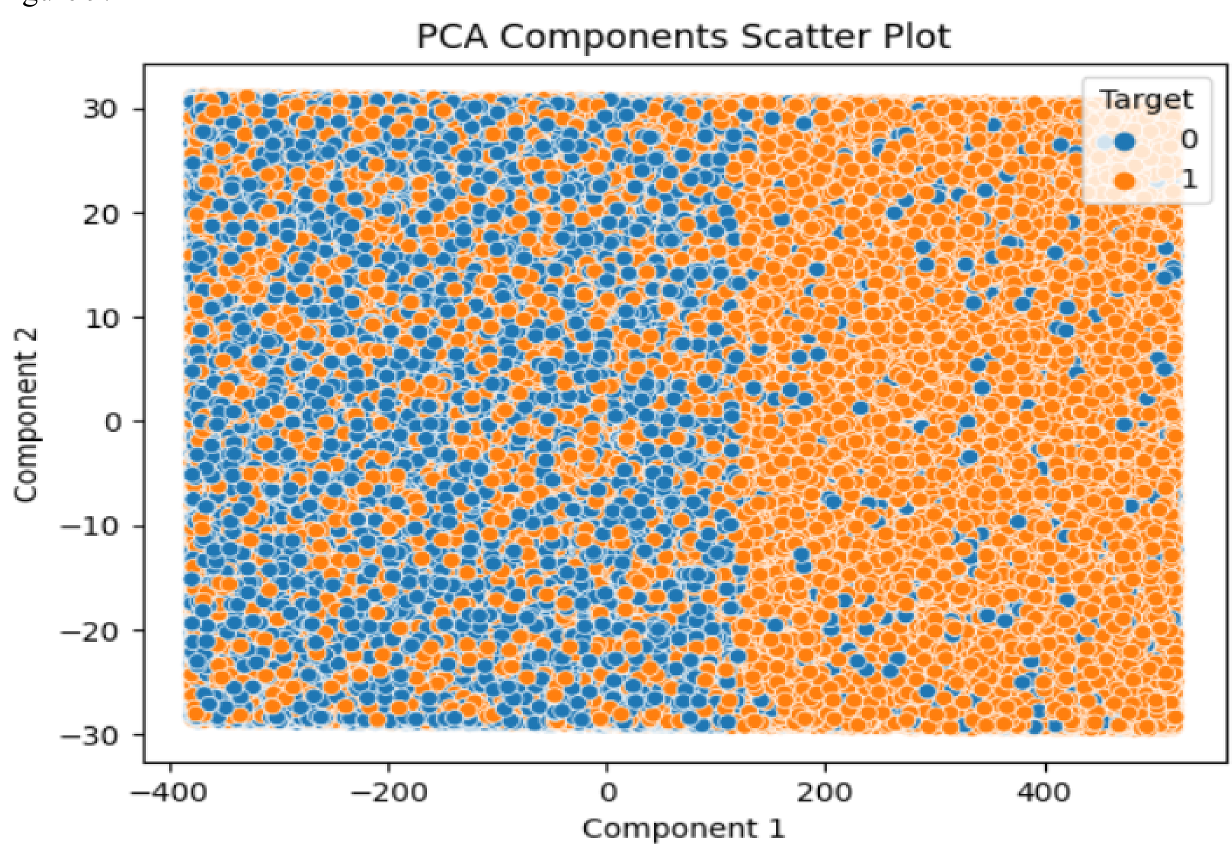


Figure 10. Logistic Classifier Confusion Matrix

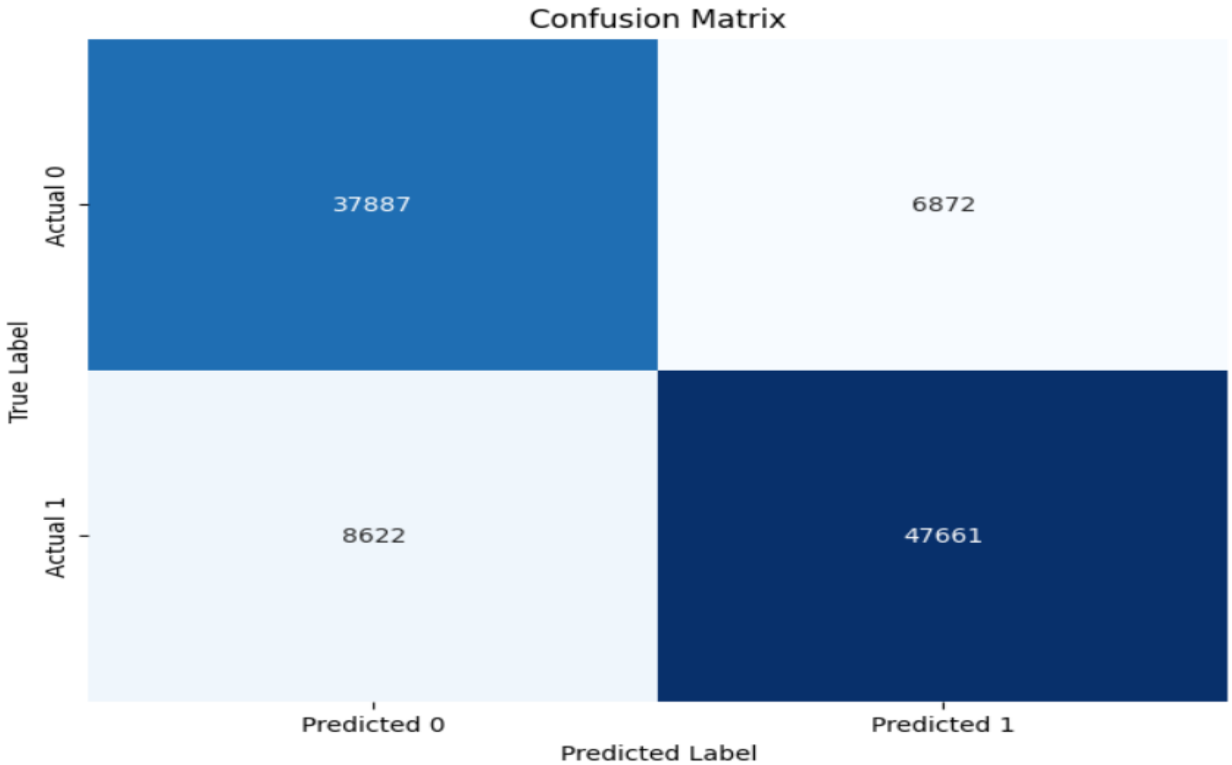


Figure 11. Decision TreeClassifier Confusion Matrix

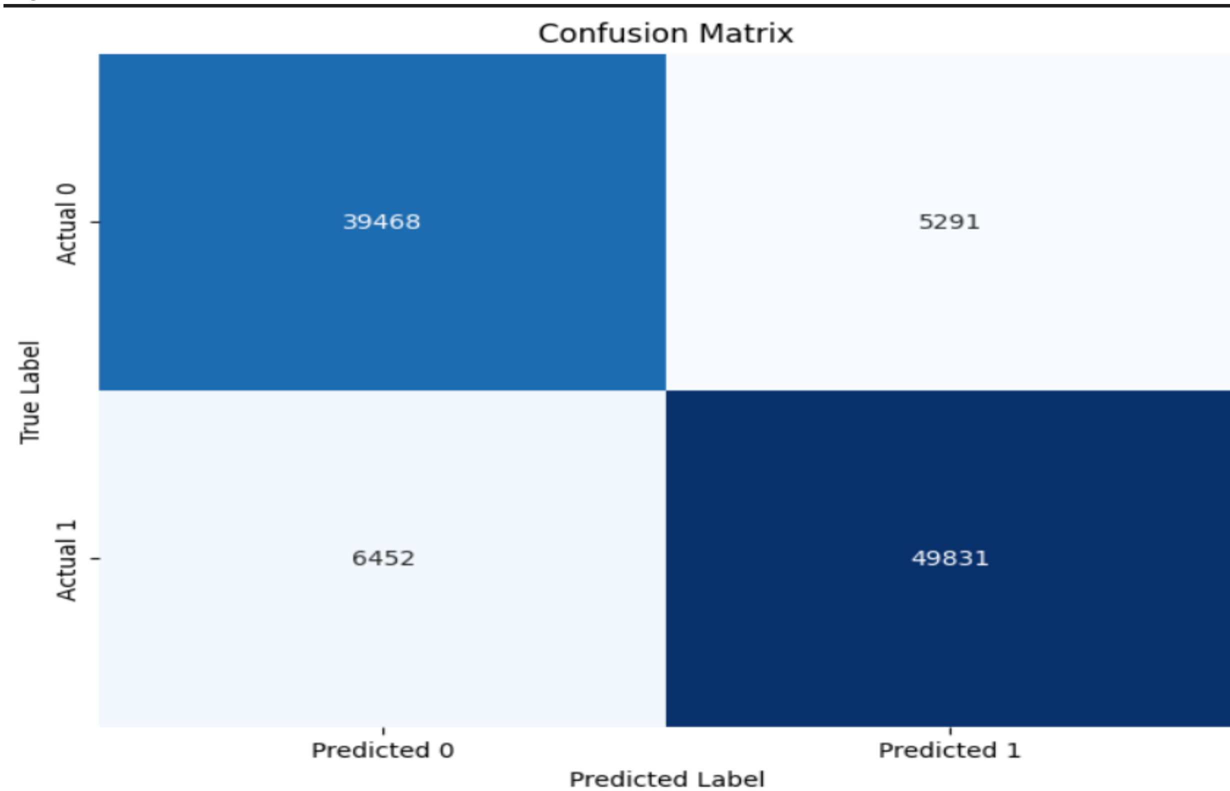


Figure 12. RandomForestClassifier Confusion Matrix

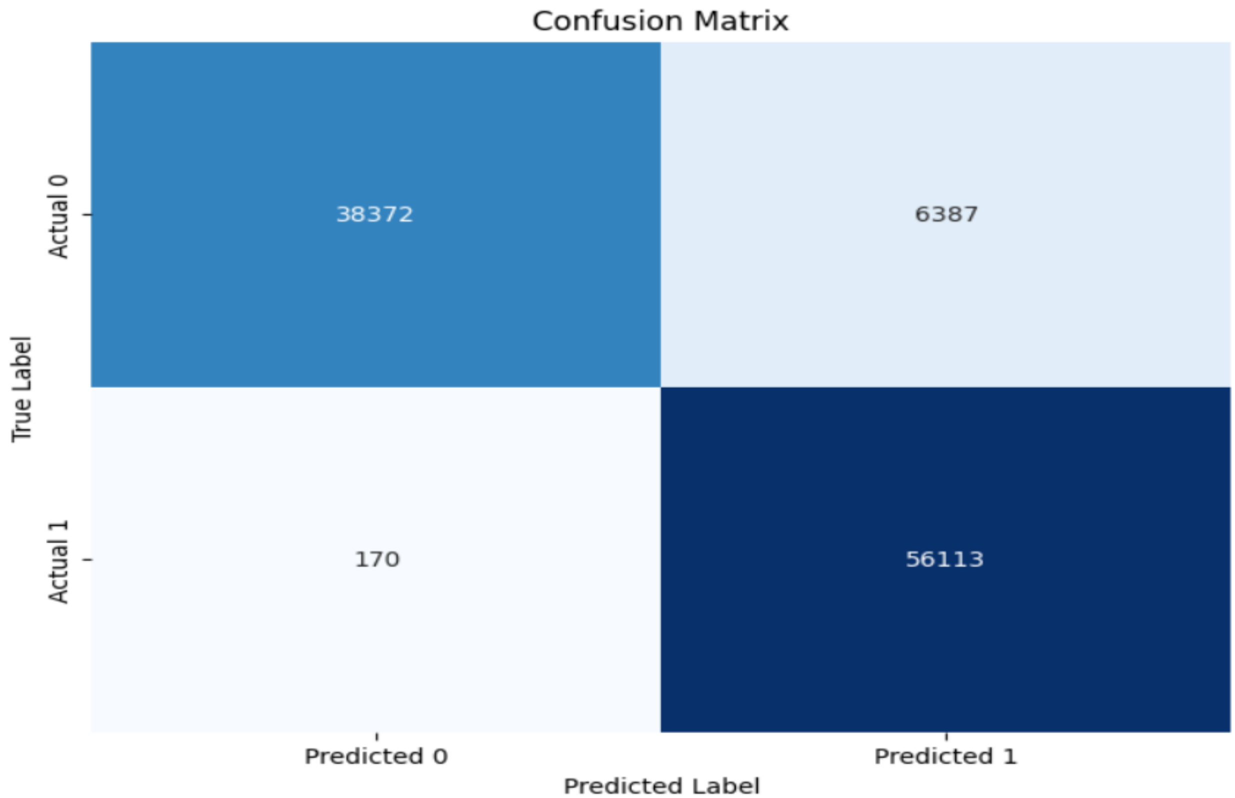


Figure 13. XGBoost Classifier Confusion Matrix

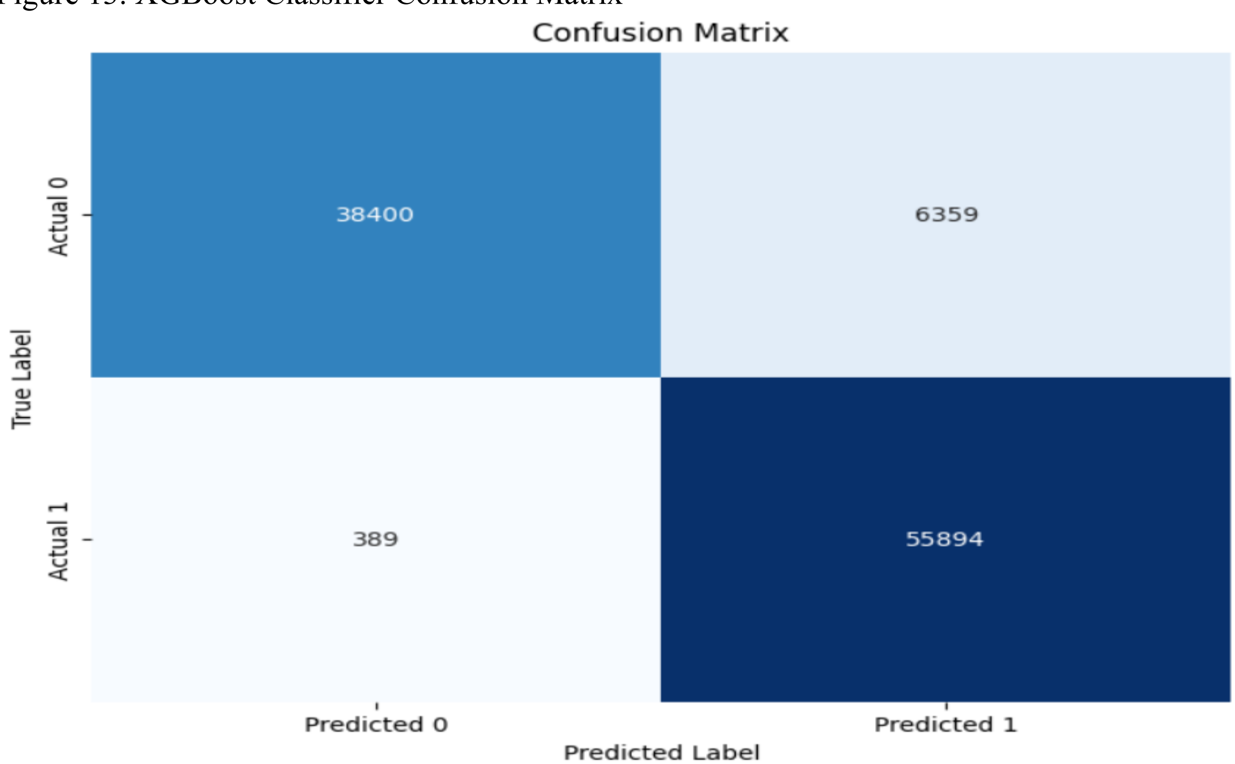


Figure 14. Prediction Using RandomForest Classifier Model

```
💡 Click here to ask Blackbox to help you code faster
customer_churn.predict(
    age = 22,
    tenure = 28,
    usage_frequency = 28,
    support_calls = 10,
    payment_delay = 13,
    total_spend = 584.0,
    last_interaction = 20,
    gender = 'Female',
    subscription_type = 'Standard',
    contract_length = 'Monthly'
)
[18] ✓ 0.0s                                                     Python

... c:\Users\91827\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but OneHot
      warnings.warn(

... 'Will Churn'
```

Table 1. Comparative Analysis

Model	**Accuracy**	**Precision**	**Recall**
Logistic Regression	**0.85**	**0.87**	**0.85**
Decision Tree	**0.88**	**0.90**	**0.89**
Random Forest	**0.94**	**0.90**	**1.00**
XGBoost	**0.93**	**0.90**	**0.99**

The above Table clearly shows that the Random Forest Classifier has the best Accuracy from others. Hence, it will be used for further Predictions

5. <u>Conclusion:</u>

In conclusion, this collegiate endeavor centered on customer churn analysis within the telecommunications sector has been a comprehensive exploration of data analytics and predictive modeling. The project's primary objective was to discern the factors influencing customer attrition within telecom services.

The datasets under examination, denoted as "customer_churn_dataset-testing-master.csv" and "customer_churn_dataset-training-master.csv" and sourced from Kaggle, constituted the foundation of our analytical pursuits. Comprising 12 distinct feature columns, ranging from demographic details to contractual specifics, these datasets facilitated an in-depth investigation into the intricacies of customer behavior.

Our approach commenced with a meticulous data analysis phase, wherein univariate, multivariate, and continuous-discrete analyses were employed to reveal patterns and relationships inherent in the data. Notably, gender analysis was undertaken to ascertain its potential impact on customer churn. Subsequently, our data exploration extended to a visualization strategy, utilizing plots such as histograms, scatter plots, and bar charts to enhance interpretability.

The predictive modeling phase marked the culmination of our efforts, encompassing the training and evaluation of machine learning models, including Naive Bayes, Decision Tree, Random Forest, and XGBoost. The selection and assessment of models were grounded in meticulous evaluation metrics such as accuracy, precision, recall, and F1 score.

In essence, this project transcends the mere prediction of churn; it serves as a testament to the integration of theoretical knowledge and practical application in the domain of data science. The synthesis of data analysis, visualization, and predictive modeling methodologies offers a robust framework for comprehending and addressing complex challenges in telecommunications.

As we draw the curtain on this project, it stands not only as a representation of academic inquiry but as an affirmation of the invaluable insights that data science can yield in deciphering the dynamics of customer behavior. The journey has been one of intellectual growth and technical acumen, laying the groundwork for future exploration and advancements in the realm of data analytics.

6. References:

[1] Mohammad Ridwan Ismail, Mohd Khalid Awang, M. Nordin A. Rahman,Mokhairi Makhtar, A multi-layer perceptron approach for customer churn prediction, International Journal of Multimedia and Ubiquitous Engineering 10 (7) (2015) 213–222.

[2] Farhad Shaikh, Brinda Pardeshi, Ajay Jachak, Akash Bendale,Nandakishor Sonune, Mangal Katkar, Customer churn prediction using nlp and machine learning: an overview, International Journal Of Advance Scientific Research And Engineering Trends 6 (2) (2021) 40–45.

[3] S. Babu, N.R. Ananthanarayanan, V. Ramesh, A study on efficiency of decision tree and multi layer perceptron to predict the customer churn in telecommunication using WEKA, Int. J. Comput. Appl. 140 (4) (2016) 26–30.

[4]Abhishek and Ratnesh ,"Predicting Customer Churn Prediction in Telecom Sector Using Various Machine Learning Techniques", In the proceedings of 2017 International Conference on Advanced Computation and Telecommunication, Bhopal, India, 2017.

[5].Abinash and Srinivasulu U ,"Machine Learning techniques applied to prepaid subscribers: case study on the telecom industry of Morocco", In the proceedings of 2017 International Conference on Inventive Computing and Informatics , Coimbatore, India, pp. 721-725, 2017.