

Customer Churn Prediction

Saloni Nimgaonkar || SunbaseData Assignment

Assignment Introduction

Dataset: customer_churn_large_dataset.xlsx

Task: To predict if the customer will churn service.

Data Preprocessing

1) Load the provided dataset and perform initial data exploration.

Load the data using pandas read_excel and preview 5 rows using head(5)

DATA PREPROCESSING

```
[ ] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
[ ] df = pd.read_excel("customer_churn_large_dataset.xlsx")
```

```
[ ] df.head(5)
```

	CustomerID	Name	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn	
0	1	Customer_1	63	Male	Los Angeles	17	73.36	236	0	
1	2	Customer_2	62	Female	New York	1	48.76	172	0	
2	3	Customer_3	24	Female	Los Angeles	5	85.47	460	0	
3	4	Customer_4	36	Female	Miami	3	97.94	297	1	
4	5	Customer_5	46	Female	Miami	19	58.14	266	0	

2) Handle missing data and outliers.

Check was conducted to identify to identify features containing missing data. No missing

data was found.

```
[70] # Check for missing values in the dataset
missing_values = df.isnull().sum()

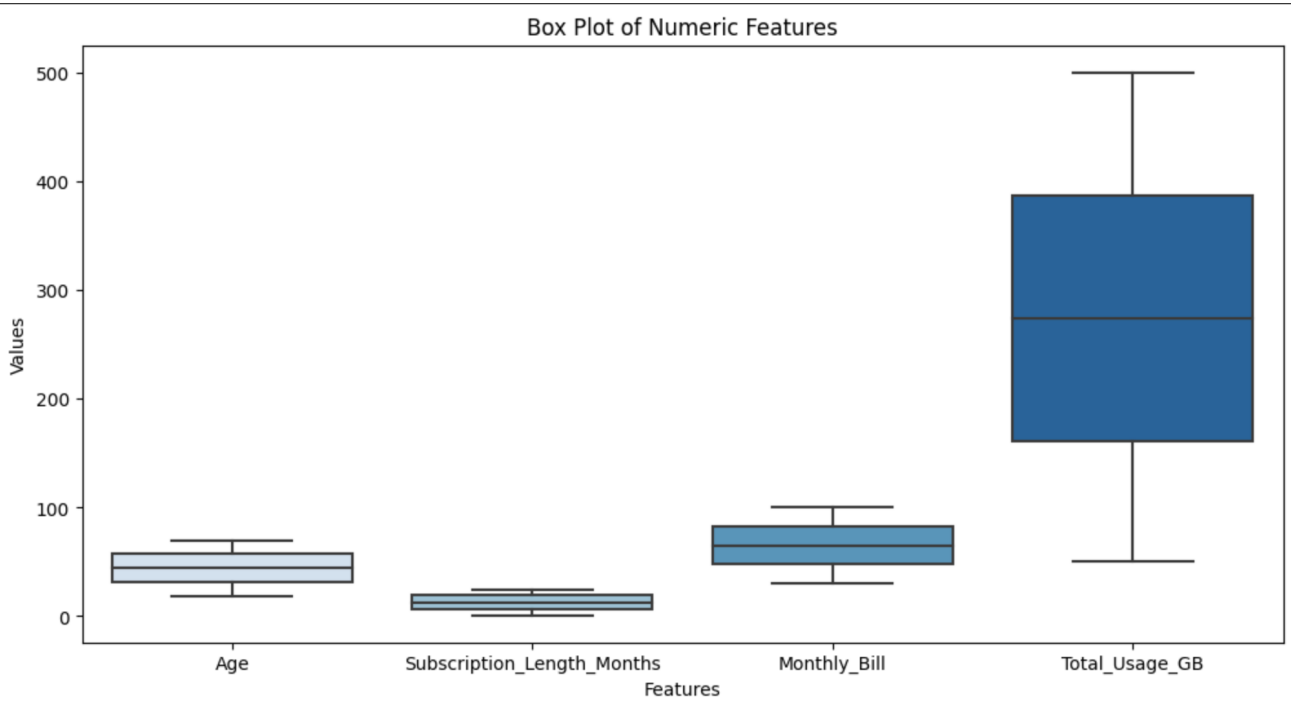
# count of missing values for each column
print("Missing Values in Each Column:")
print(missing_values)

# Total count of missing values
total_missing = missing_values.sum()
print("\nTotal Missing Values in the Dataset:", total_missing)

Missing Values in Each Column:
CustomerID      0
Name            0
Age            0
Gender          0
Location        0
Subscription_Length_Months  0
Monthly_Bill    0
Total_Usage_GB  0
Churn           0
dtype: int64

Total Missing Values in the Dataset: 0
```

Box-plot was used to identify outliers.



3) Prepare the data for machine learning by encoding categorical variables and splitting it into training and testing sets.

‘Gender’ and ‘Location’ Features contained categorical values. They were converted to numeric values using Label encoding (One hot encoding was also tested for the Location feature)

Then features like 'CustomerID' and 'Name' were dropped as they were not involved in making the predictions

```
[72] from sklearn.preprocessing import StandardScaler, LabelEncoder

# Encode categorical variables like 'Gender' using Label Encoding
label_encoder = LabelEncoder()
df['Gender'] = label_encoder.fit_transform(df['Gender'])
df['Location'] = label_encoder.fit_transform(df['Location'])

# # One-Hot Encode categorical variables like 'Location'
# df = pd.get_dummies(df, columns=['Location'], prefix=['Location'])

df = df.drop(columns=['CustomerID', 'Name'])
df.head(5)
```

	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn
0	63	1	2	17	73.36	236	0
1	62	0	4	1	48.76	172	0
2	24	0	2	5	85.47	460	0
3	36	0	3	3	97.94	297	1
4	46	0	3	19	58.14	266	0

80 - 20 split was performed on the data where churn was considered as the dependent variable while all the other features were the independent variables.

Feature Engineering

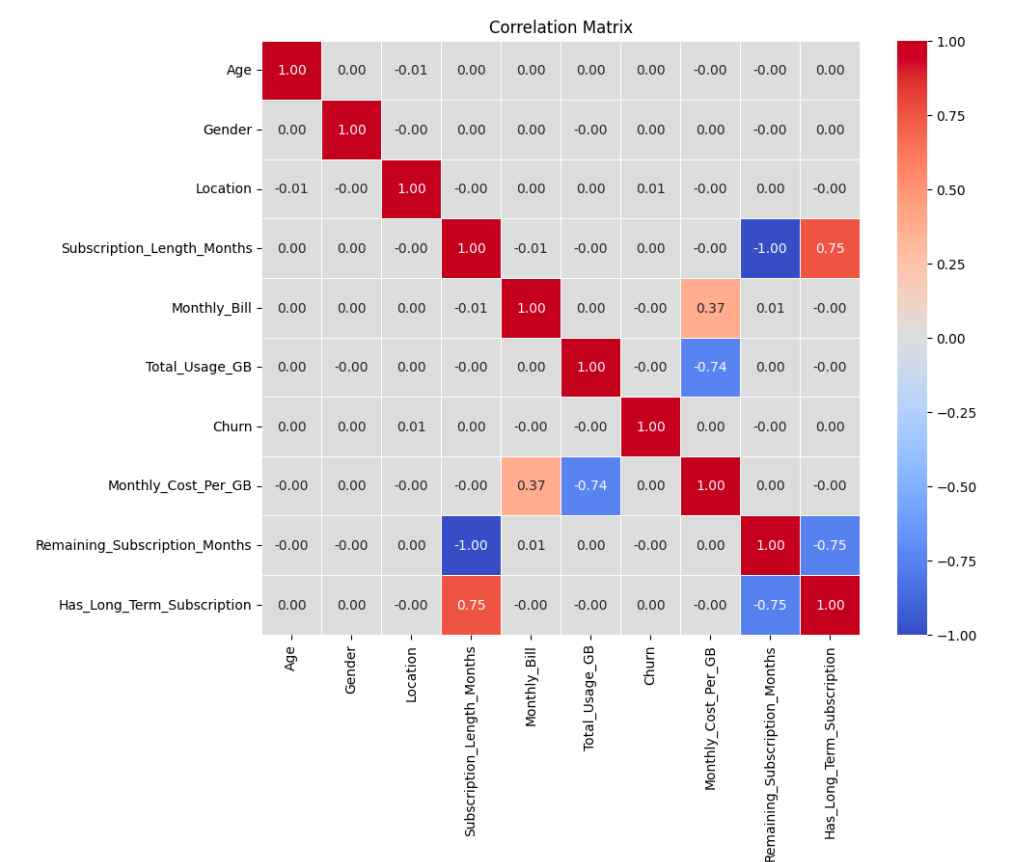
1) Generate relevant features from the dataset that can help improve the model's prediction accuracy.

As the data contained only 6 independent features, it was necessary to generate relevant features to improve the accuracy.

Features Created:

- (1) 'Monthly_Cost_Per_GB' by dividing 'Monthly_Bill' by 'Total_Usage_GB'
- (2) 'Remaining_Subscription_Months' by subtracting 'Subscription_Length_Months' from 12 (assuming annual subscriptions)
- (3) 'Has_Long_Term_Subscription' to indicate customers with subscriptions longer than 6 months

The correlation matrix can depict the negative and positive correlation between different features. And as this correlation is below 0.80 and -0.80, we can sure add the engineered additional features to the model.



2) Apply feature scaling or normalization if necessary.

Scaling (Normalization and Standardization) can improve the performance of some machine learning models.

ML models like Logistic Regression and K Nearest Neighbours perform better if the data is standardized (Range: -1 to 1)

```
[87] from sklearn.preprocessing import StandardScaler
# Scale the features using StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

But for other models like Decision Tree, it is not necessary.

Model Building

1) Choose appropriate machine learning algorithms (e.g., logistic regression, random forest, or neural networks).

6 ML models were tested which will work well with binary classification

- (1) Random Forest (combines multiple decision trees to make predictions)
- (2) Decision Trees (used for classification)
- (3) Naive Bayes (simple probabilistic model)
- (4) AdaBoost (combine multiple models to create a stronger predictive model)
- (5) KNN (calculates distances between data points and can be effective in capturing local patterns.)
- (6) Logistic Regression (interpretable model that works well for binary classification tasks)

2) Train and validate the selected model on the training dataset.

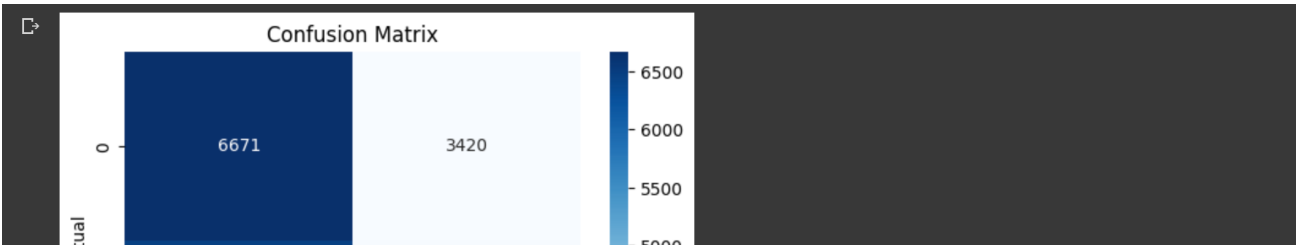
Used the scikit-learn to implement these models and trained the data.

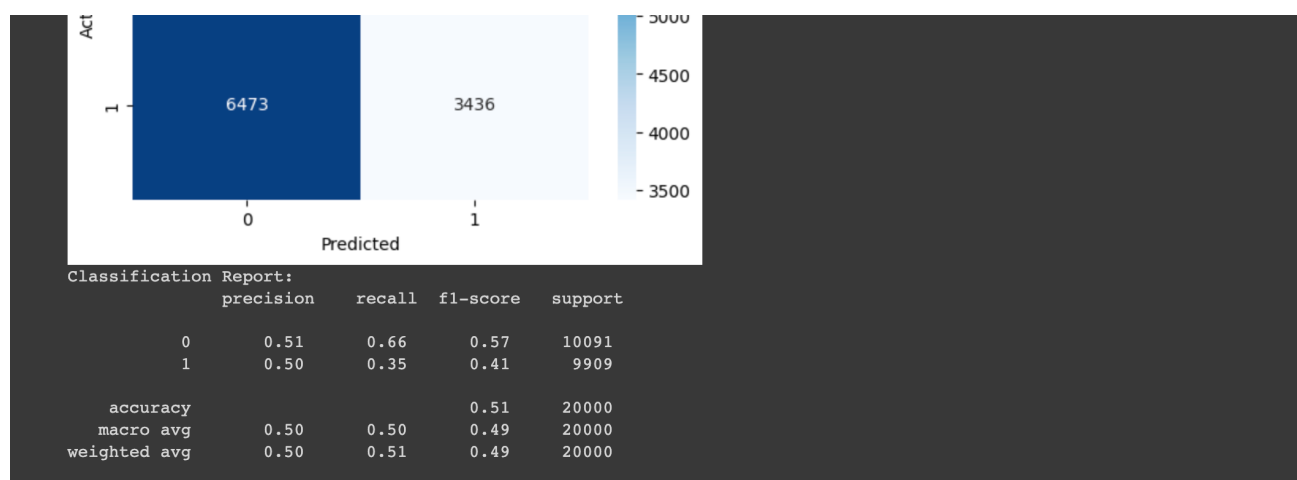
3) Evaluate the model's performance using appropriate metrics

Accuracy Report:

- (1) Random Forest - 0.5003
- (2) Decision Trees - 0.5011
- (3) Naive Bayes - 0.50215
- (4) AdaBoost - 0.4991
- (5) KNN - 0.4991
- (6) Logistic Regression - 0.50535

Logistic Regression had the highest accuracy of 50.53%





Model Optimization

1) Fine-tune the model parameters to improve its predictive performance. Explore techniques like cross-validation and hyperparameter tuning.

After testing the 6 models, I realised that independent feature like ‘gender’ is not fairly important in predicting the churn. Hence, feature engineering was done in order to optimise the performance of the model. 5 features were generated and the ones with correlation less than 0.80 were selected.

Techniques for hyper-parameter tuning like Grid-Search and Random-Search were also tested. But they were not helpful in improving the performance.

```
*** Random Forest Classifier ***
Accuracy: 0.5003

*** Decision Tree Classifier ***
Accuracy: 0.5011

*** Naive Bayes Classifier ***
Accuracy: 0.50215

*** AdaBoost Classifier ***
Accuracy: 0.4991
```

For improving the performance of Logistic and KNN model, standardization was done on the training dataset. This step was found significant in increasing the accuracy.

```
KNN Accuracy: 0.4991
Logistic Regression Accuracy: 0.50535
```

Model Deployment


1) Once satisfied with the model's performance, deploy it into a production-like environment (you can simulate this in a development environment). Ensure the model can take new customer data as input and provide churn predictions.

User Input and Prediction:

```
Please enter the following customer details:
Age: 63
Gender (0 for Female, 1 for Male): 1
Location (encoded as per your dataset): 2
Subscription Length (in months): 17
Monthly Bill: 73
Total Usage (in GB): 236

Based on the provided information, the customer is predicted to be: Not Churned
```

Similar Training Data:

	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn	
0	63	1	2	17	73.36	236	0	