Predicting Customer Churn for a Subscription Service

Project Description: This project focuses on building a machine learning model to predict customer churn for a subscription-based service. Customer churn refers to when customers stop using a service or cancel their subscription. Predicting churn is crucial for businesses as retaining existing customers is often more cost-effective than acquiring new ones. By accurately identifying customers who are likely to churn, businesses can implement targeted retention strategies to improve customer loyalty and reduce revenue loss.

Project Title: Predicting Customer Churn for a Subscription-Based Service

Project Overview:

This project is about creating a machine learning model to predict when customers will stop using a subscription service (called "churn"). It is important for companies to know if customers might leave because keeping current customers is usually cheaper than getting new ones. By knowing which customers are likely to stop their subscription, businesses can take action to keep them.

Objective:

The main goal of this project is to build a model that can predict which customers are at risk of leaving the service. This will help the company create special offers or strategies to keep these customers loyal.

Steps Involved:

1. **Data Collection**: I worked with customer data that included information like customer age, subscription dates, frequency of purchases, support tickets, and whether they left (churned).
2. **Data Cleaning and Preparation**: I cleaned and organized the data to make sure it was ready for building the machine learning model. This included handling missing data, formatting dates, and creating useful features for the model.
3. **Exploratory Data Analysis (EDA)**: I analyzed the data to understand customer behavior and which factors might lead to churn. I visualized things like customer age, purchase frequency, and support ticket resolution times to find patterns.
4. **Building the Model**: I used machine learning algorithms, such as Logistic Regression, Decision Trees, or Random Forest, to train a model that could predict whether a customer would churn or not.
5. **Evaluation**: I tested the model's accuracy using evaluation metrics like accuracy, precision, recall, and the F1 score to see how well it predicted churn.
6. **Results**: The model successfully identified customers likely to churn, helping the business to create strategies to retain them.

Importing important libraries

```
[147]: """Predicting Customer Churn for a Subscription Service
Project Description :This project focuses on building a machine learning model to
predict customer churn for a subscription-based service. Customer churn refers to
when customers stop using a service or cancel their subscription. Predicting churn is
crucial for businesses as retaining existing customers is often more cost-effective
than acquiring new ones. By accurately identifying customers who are likely to churn,
businesses can implement targeted retention strategies to improve customer loyalty and reduce revenue loss.
"""
```

```
[147]: 'Predicting Customer Churn for a Subscription Service\nProject Description :This project focuses on building a machine learning model to \npr
edict customer churn for a subscription-based service. Customer churn refers to \nwhen customers stop using a service or cancel their subscri
ption. Predicting churn is \ncrucial for businesses as retaining existing customers is often more cost-effective \nthan acquiring new ones. B
y accurately identifying customers who are likely to churn, \nbusinesses can implement targeted retention strategies to improve customer loya
lty and reduce revenue loss. \n'
```

```
[1]: # Import necessary libraries
import pandas as pd  # For data manipulation
from sklearn.model_selection import train_test_split  # For splitting data into train and test sets
from sklearn.preprocessing import LabelEncoder  # For encoding categorical variables
from sklearn.ensemble import RandomForestClassifier  # For building the Random Forest model
from sklearn.metrics import classification_report, accuracy_score  # For model evaluation
import matplotlib.pyplot as plt  # For plotting graphs
import seaborn as sns  # For visualization
```

Importing data from excel sheet using pandas

```
[2]: # Load the dataset using the provided file path
data = pd.read_excel(r"E:\folder\Machine Learning\yhillsproject\CHURN ML\churn_data.xlsx")

# Display the first few rows to understand the structure of the data
data
```

[2]:

	Customer_ID	Subscription_Start_Date	Subscription_End_Date	Age	Gender	Location	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution
0	1	2023-01-01	2023-02-01	25	Male	Delhi	10	20	500	2	
1	2	2023-01-08	2023-02-08	34	Female	Mumbai	25	5	1200	5	
2	3	2023-01-15	2023-02-15	28	Male	Bangalore	15	18	800	3	
3	4	2023-01-22	2023-02-22	42	Female	Hyderabad	30	2	1500	2	
4	5	2023-01-29	2023-03-01	40	Male	Delhi	29	17	1910	1	
...	...	...	...	...	...	...	...	...	...	...	
95	96	2024-10-27	2024-11-27	25	Female	Delhi	10	20	500	2	
96	97	2024-11-03	2024-12-04	34	Male	Mumbai	25	5	1200	5	
97	98	2024-11-10	2024-12-11	28	Female	Bangalore	15	18	800	3	
98	99	2024-11-17	2024-12-18	42	Male	Hyderabad	30	2	1500	2	
99	100	2024-11-24	2024-12-25	40	Female	Delhi	29	17	1910	1	

100 rows × 12 columns

```
[3]: data.head()
```

[3]:

	Customer_ID	Subscription_Start_Date	Subscription_End_Date	Age	Gender	Location	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution_
0	1	2023-01-01	2023-02-01	25	Male	Delhi	10	20	500	2	
1	2	2023-01-08	2023-02-08	34	Female	Mumbai	25	5	1200	5	
2	3	2023-01-15	2023-02-15	28	Male	Bangalore	15	18	800	3	
3	4	2023-01-22	2023-02-22	42	Female	Hyderabad	30	2	1500	2	
4	5	2023-01-29	2023-03-01	40	Male	Delhi	29	17	1910	1	

Isnull().sum() checks the null values

```
[4]: data.tail()
```

	Customer_ID	Subscription_Start_Date	Subscription_End_Date	Age	Gender	Location	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution
95	96	2024-10-27	2024-11-27	25	Female	Delhi	10	20	500	2	
96	97	2024-11-03	2024-12-04	34	Male	Mumbai	25	5	1200	5	
97	98	2024-11-10	2024-12-11	28	Female	Bangalore	15	18	800	3	
98	99	2024-11-17	2024-12-18	42	Male	Hyderabad	30	2	1500	2	
99	100	2024-11-24	2024-12-25	40	Female	Delhi	29	17	1910	1	

```
[5]: data.describe()
```

	Customer_ID	Age	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution_Time_Days	Churn
count	100.000000	100.000000	100.000000	100.000000	100.00000	100.0000	100.000000	100.000000
mean	50.500000	33.800000	21.800000	12.400000	1182.00000	2.6000	2.800000	0.500000
std	29.011492	6.618004	7.974707	7.429126	501.00505	1.3633	0.752101	0.502519
min	1.000000	25.000000	10.000000	2.000000	500.00000	1.0000	2.000000	0.000000
25%	25.750000	28.000000	15.000000	5.000000	800.00000	2.0000	2.000000	0.000000
50%	50.500000	34.000000	25.000000	17.000000	1200.00000	2.0000	3.000000	0.500000
75%	75.250000	40.000000	29.000000	18.000000	1500.00000	3.0000	3.000000	1.000000
max	100.000000	42.000000	30.000000	20.000000	1910.00000	5.0000	4.000000	1.000000

```
[6]: # Check for missing values
print(data.isnull().sum())
```

```
Customer_ID                0
Subscription_Start_Date    0
Subscription_End_Date      0
Age                        0
Gender                     0
Location                   0
Purchase_Frequency         0
Recency                    0
Monetary_Value             0
Support_Tickets            0
Resolution_Time_Days       0
Churn                      0
dtype: int64
```

In this data we don't have any null values so we will be continuing with the code and check the churn counts.

By – data['Churn'].values_counts()

As we can see we have 50-1 & 50-0, which is equal yes and no.

```
[7]: #here there is no null values in our data
```

```
[8]: data['Churn'].value_counts()
```

```
[8]: Churn
1    50
0    50
Name: count, dtype: int64
```

```
[9]: # Convert the subscription start and end dates to datetime format
data['Subscription_Start_Date'] = pd.to_datetime(data['Subscription_Start_Date'])
data['Subscription_End_Date'] = pd.to_datetime(data['Subscription_End_Date'])
```

```
[10]: # Calculate the duration of the subscription in days
data['Subscription_Duration'] = (data['Subscription_End_Date'] - data['Subscription_Start_Date']).dt.days
```

```
[11]: print(data['Subscription_Duration']) #shows the duration from Subscription_Start_Date and Subscription_End_Date
```

```
0     31
1     31
2     31
3     31
4     31
      ..
95    31
96    31
97    31
98    31
99    31
Name: Subscription_Duration, Length: 100, dtype: int64
```

```
[12]: # Check data types
data.dtypes
```

```
[12]: Customer_ID                          int64
Subscription_Start_Date     datetime64[ns]
Subscription_End_Date       datetime64[ns]
Age                                  int64
Gender                              object
Location                            object
Purchase_Frequency                   int64
Recency                              int64
Monetary_Value                       int64
Support_Tickets                      int64
Resolution_Time_Days                 int64
Churn                                int64
Subscription_Duration                int64
dtype: object
```

- **Convert the data types of Subscription dates to date and time(datetime).**
- **Count the duration of start_date and end_date.**
- **Counting the duration creates the new column in the table as Subscription_Duration.**

Convert the GENDER and Location to numerical values

{categorical data to numerical data}

[13]:
```python
# Initialize LabelEncoder to convert categorical data into numerical form
le = LabelEncoder()

# Encode Gender (e.g., Male -> 1, Female -> 0)
data['Gender'] = le.fit_transform(data['Gender'])

# Encode Location (e.g., different locations -> different integers)
data['Location'] = le.fit_transform(data['Location'])
```

[14]:
```python
# Check data types
data.dtypes #the datatype for gender and location changes from object to int
```

[14]:
```
Customer_ID                           int64
Subscription_Start_Date      datetime64[ns]
Subscription_End_Date        datetime64[ns]
Age                                   int64
Gender                                int32
Location                              int32
Purchase_Frequency                    int64
Recency                               int64
Monetary_Value                        int64
Support_Tickets                       int64
Resolution_Time_Days                  int64
Churn                                 int64
Subscription_Duration                 int64
dtype: object
```

[15]:
```python
data.head(6)
```

[15]:

	Customer_ID	Subscription_Start_Date	Subscription_End_Date	Age	Gender	Location	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution_Ti
0	1	2023-01-01	2023-02-01	25	1	1	10	20	500	2	
1	2	2023-01-08	2023-02-08	34	0	3	25	5	1200	5	
2	3	2023-01-15	2023-02-15	28	1	0	15	18	800	3	
3	4	2023-01-22	2023-02-22	42	0	2	30	2	1500	2	
4	5	2023-01-29	2023-03-01	40	1	1	29	17	1910	1	
5	6	2023-02-05	2023-03-08	25	0	1	10	20	500	2	

Drop the unnecessary Columns :

[Customer_ID, Subscription_Start_Date, Subscription_End_Date]

[16]:
```python
# Drop Customer_ID and the original date columns since they are no longer needed
data.drop(['Customer_ID', 'Subscription_Start_Date', 'Subscription_End_Date'], axis=1, inplace=True)
```

[17]:
```python
data.head(6)
```

[17]:

	Age	Gender	Location	Purchase_Frequency	Recency	Monetary_Value	Support_Tickets	Resolution_Time_Days	Churn	Subscription_Duration
0	25	1	1	10	20	500	2	3	1	31
1	34	0	3	25	5	1200	5	2	0	31
2	28	1	0	15	18	800	3	4	1	31
3	42	0	2	30	2	1500	2	2	0	31
4	40	1	1	29	17	1910	1	3	1	31
5	25	0	1	10	20	500	2	3	0	31

Work on the model for prediction

[58]:
```
# Define the feature variables (X) and target variable (y)
X = data.drop('Churn', axis=1)  # Features are all columns except 'Churn'
y = data['Churn']  # Target is the 'Churn' column

# Split the dataset into 70% training and 30% testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

[59]:
```
# Initialize the Random Forest classifier with 100 trees
rf = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model on the training data
rf.fit(X_train, y_train)

# Predict churn on the test data
y_pred = rf.predict(X_test)
```

[60]:
```
# Calculate and print the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

# Print the classification report for detailed performance metrics (precision, recall, F1-score)
print('Classification Report:')
print(classification_report(y_test, y_pred))
```

```
Accuracy: 93.33%
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        12
           1       0.94      0.94      0.94        18

    accuracy                           0.93        30
   macro avg       0.93      0.93      0.93        30
weighted avg       0.93      0.93      0.93        30
```

Conclusion: By using machine learning to predict customer churn, the company can concentrate on keeping valuable customers instead of spending more money to get new ones. This project shows how predictive models can give useful insights, helping businesses act quickly and effectively to lower churn rates and keep their customers loyal. With an accuracy of **93.33%**, the model is very good at identifying customers who might leave. This project can be used to improve customer relationships, boost satisfaction, and increase profits.