PREDICTIVE ANALYSIS DOCUMENT

1 Import Libraries & Load Data in Python

```
#Import Libraries & Load Data
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import LabelEncoder
import joblib
import openpyxl
import matplotlib.pyplot as plt
import seaborn as sns

#Define the path to the excel file
file_path = r"D:\POWERBI\POWERBI PROJECT\Customer Churn - Data Analysis\Predictive Data.xlsx"

#Define sheet name to read data from
sheet_name = 'new_churndata'

#Read the data from the specified sheet into pandas Dataframe
data = pd.read_excel(file_path, sheet_name=sheet_name)

#Display the first few rows of the fetched data
print(data.head())
```

2 Data Processing

```
# Data Processing
# Drop columns that won't be used for prediction

data = data.drop(['CUSTOMER_ID','Churn_Category','Churn_Reason'], axis = 1)

#list of Columns to be Label encoded.
columns_to_encode = ['Gender', 'Married','State','Value_Deal','Phone_Service','Multiple_Lines'
    ,'Internet_Service','Internet_Type','Online_Security','Online_Backup','
    Device_Protection_Plan','Premium_Support', 'Streaming_TV' , 'Streaming_Movies','
    Streaming_Music','Unlimited_Data', 'Contract','Paperless_Billing','Payment_Method']

label_encoders = {}

for column in columns_to_encode:

    label_encoders[column] = LabelEncoder()

    data[column] = label_encoders[column].fit_transform(data[column])

 # In this encoding, all categorical data is arranged in alphabetical order :
 # e.g gender will have two values -male and female, so Female =0 , male =1 is labeled
```

```
#Manually encode the target variable 'Customer_Status'
data['Customer_Status'] = data['Customer_Status'].map({'Stayed':0, 'Churned':1})

#Split data into features and target
X = data.drop ('Customer_Status', axis = 1)
# In X we are storing all column except customer_status, that why we drop it,
y=  data['Customer_Status']
# In y, we are storing Customer_status

#Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2, random_state = 42)

#In training any model, we need to have train and test dataset, we are considering 80% as
    train dataset and 20% as test datasets
# we are using train_test_split to separate dataset as train and test datasets.
```

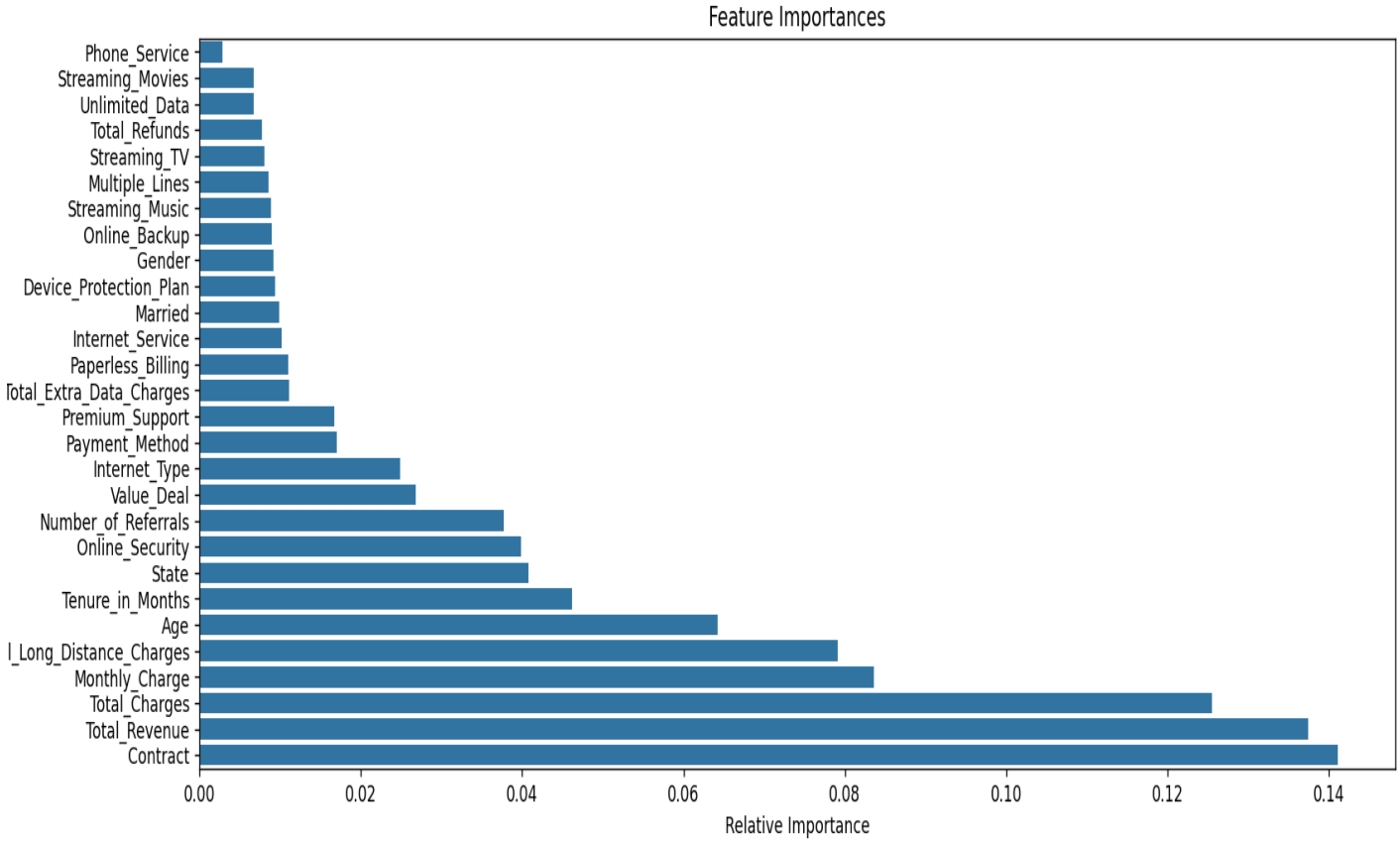
3 Train the Model

```
#Train Random Forest Model
#Initialise the Random Forest Classifier
rf_model = RandomForestClassifier(n_estimators = 100, random_state = 42)
#Estimators = to train a model 100 times
# random_state = randomness in operations like data splitting or sampling is controlled. To
    reproduce output


#Train the model
rf_model.fit(X_train,y_train)
```

4 Evaulating Model

```
#Evaluate model
#Make Predictions
y_pred = rf_model.predict(X_test)

#Evaluate the model
print("Confusion Matrix")
print(confusion_matrix(y_test,y_pred))
print("\nClassifaction Report:")
print(classification_report(y_test,y_pred))


#Feature Selection using Feature Importance
importances = rf_model.feature_importances_
indices = np.argsort(importances)[::1]
#Sorting the importances in highest to lowest

#Plot the features importances
plt.figure(figsize = (15,6))
sns.barplot(x=importances[indices], y=X.columns[indices])
plt.title('Feature Importances')
plt.xlabel('Relative Importance')
plt.ylabel('Feature Names')
plt.show()
```

```
Confusion Matrix
[[778  69]
 [129 226]]

Classifaction Report:
              precision    recall  f1-score   support

           0       0.86      0.92      0.89       847
           1       0.77      0.64      0.70       355

    accuracy                           0.84      1202
   macro avg       0.81      0.78      0.79      1202
weighted avg       0.83      0.84      0.83      1202
```



5 Use Model for Prediction on New Data

```
#Predict on new Data
#Define the path to the Joiner Data excel file
file_path = r"D:\POWERBI\POWERBI PROJECT\Customer Churn - Data Analysis\Predictive Data.xlsx"

#Define sheet name to read data from
sheet_name = 'new_Joineddata'
```

```
#Read the data from the specified sheet into the pandas DataFrame
new_data  = pd.read_excel(file_path, sheet_name=sheet_name)

#Display the orginal DataFrame to preserve unencoded columns
orginal_data = new_data.copy()

#Retain the Customer_ID column
customer_ids = new_data['CUSTOMER_ID']

#Drop column that won't be used for prediction in the encoded Dataframe
new_data = new_data.drop(['CUSTOMER_ID', 'Customer_Status','Churn_Category','Churn_Reason'],
    axis = 1)
#axis =1 means we are considering columns
#axis =0 means we are considering rows

#Encode categorical variables using the saved label encoders.
for column in new_data.select_dtypes(include = ['object']).columns:
    new_data[column] = label_encoders[column].transform(new_data[column])

#Make Predictions
new_predictions = rf_model.predict(new_data)

#Add Predictions to the original DataFrame
orginal_data['Customer_Status_Predicted'] = new_predictions

#Filter the DataFrame to include only records predicted as 'Churned'
orginal_data =  orginal_data[orginal_data['Customer_Status_Predicted'] == 1]

#Save the results
orginal_data.to_csv(r"D:\POWERBI\POWERBI␣PROJECT\Customer␣Churn␣-Data␣Analysis\Predictive␣
    Analysis.csv", index= False)
```