```
# For Google Colab: Upload file
from google.colab import files
uploaded = files.upload()

# Import the pandas library
import pandas as pd

# Read the uploaded CSV file (update filename as needed)
df = pd.read_csv("churn_dataset2 (2).csv")  # adjust filename if different
df.head()
```

```
# Step 1: Data Preprocessing

# Convert 'TotalCharges' to numeric, coerce errors to NaN
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')

# Check for missing values
missing_values = df.isnull().sum()

# Drop rows with missing values for simplicity
df_cleaned = df.dropna()

# Drop 'customerID' as it is not useful for prediction
df_cleaned = df_cleaned.drop('customerID', axis=1)

# Convert categorical variables to dummy/indicator variables
df_encoded = pd.get_dummies(df_cleaned, drop_first=True)

# Separate features and target variable
X = df_encoded.drop('Churn_Yes', axis=1)
y = df_encoded['Churn_Yes']

# Output the shape and check if preprocessing looks good
X.shape, y.shape, missing_values

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Split the dataset 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)

# Initialize the Random Forest Classifier
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
```

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

# Make predictions
y_pred = rf_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

accuracy, conf_matrix, report
```

```
from imblearn.over_sampling import SMOTE

# Apply SMOTE to balance the classes in the training set
smote = SMOTE(random_state=42)
X_train_balanced, y_train_balanced = smote.fit_resample(X_train, y_train)

# Check the new class distribution
y_train_balanced.value_counts()
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns # Import the seaborn library and alias it as 'sns'

# Get feature importances from the trained Random Forest model
importances = rf_model.feature_importances_
feature_names = X.columns

# Create a DataFrame for better visualization
feature_importance_df = pd.DataFrame({
    'Feature': feature_names,
    'Importance': importances
}).sort_values(by='Importance', ascending=False)

# Plot top 10 important features
plt.figure(figsize=(10, 6))
```

```
sns.barplot(x='Importance', y='Feature', data=feature_importance_df.head(10),
palette='viridis')
plt.title('Top 10 Important Features for Predicting Churn')
plt.xlabel('Importance Score')
plt.ylabel('Feature')
plt.tight_layout()
plt.show()
```