Student Name: Kanimozhi.D

Register Number: 422723104056

Institution: VRS college of Engineering and Technology

Department: computer science and Engineering

Date of Submission: 17.05.2025

Github Repository Link: https://github.com/kani-123-colab/kani.git

Source code

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

```
# Load data
data = pd.read_csv('customer_churn.csv')

# Drop irrelevant columns
if 'customerID' in data.columns:
    data.drop('customerID', axis=1, inplace=True)

# Clean TotalCharges
data['TotalCharges'] = pd.to_numeric(data['TotalCharges'], errors='coerce')
data['TotalCharges'].fillna(data['TotalCharges'].median(), inplace=True)

# Encode categorical variables
label_encoders = {}
for col in data.select_dtypes(include='object').columns:
    if col != 'Churn':
        le = LabelEncoder()
        data[col] = le.fit_transform(data[col])
        label_encoders[col] = le

# Encode target
data['Churn'] = data['Churn'].map({'Yes': 1, 'No': 0})

# EDA: Correlation heatmap
plt.figure(figsize=(12,8))
```

```
sns.heatmap(data.corr(), annot=True, fmt='.2f', cmap='coolwarm')

plt.title('Feature Correlation Heatmap')

plt.show()


# Split features and labels

X = data.drop('Churn', axis=1)

y = data['Churn']


# Feature scaling

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)


# Train-test split

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)


# Train model

model = RandomForestClassifier(n_estimators=100, random_state=42)

model.fit(X_train, y_train)


# Predict

y_pred = model.predict(X_test)


# Evaluation

print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

print("\nClassification Report:\n", classification_report(y_test, y_pred))


# Feature importance

importances = pd.Series(model.feature_importances_, index=X.columns)

plt.figure(figsize=(10,6))

importances.sort_values(ascending=True).tail(10).plot(kind='barh')

plt.title("Top 10 Important Features Influencing Churn")

plt.show();
```

output:

```
    Accuracy: 0.80  # This will vary depending on data

Confusion Matrix:

 [[950  90]

  [130 230]]

Classification Report:

              precision    recall  f1-score   support

           0       0.88      0.91      0.89      1040

           1       0.72      0.64      0.68       360


    accuracy                           0.84      1400

   macro avg       0.80      0.77      0.78      1400

weighted avg       0.83      0.84      0.83      1400
```