import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix, ConfusionMatrixDisplay, roc\_curve, auc

import streamlit as st

# Load dataset (replace with actual path if needed)

df = pd.read\_csv('Telco-Customer-Churn.csv')

# Data Preprocessing

df.dropna(inplace=True)

df['TotalCharges'] = pd.to\_numeric(df['TotalCharges'], errors='coerce')

df.dropna(subset=['TotalCharges'], inplace=True)

df['Churn'] = df['Churn'].map({'Yes': 1, 'No': 0})

df.drop(['customerID'], axis=1, inplace=True)

# Encode categorical variables

df\_encoded = pd.get\_dummies(df)

# Split data

X = df\_encoded.drop('Churn', axis=1)

y = df\_encoded['Churn']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train model

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

# Evaluate model

y\_pred = model.predict(X\_test)

print("Classification Report:\n", classification\_report(y\_test, y\_pred))

cm = confusion\_matrix(y\_test, y\_pred)

ConfusionMatrixDisplay(confusion\_matrix=cm).plot()

plt.title("Confusion Matrix")

plt.show()

# ROC Curve

y\_prob = model.predict\_proba(X\_test)[:, 1]

fpr, tpr, \_ = roc\_curve(y\_test, y\_prob)

roc\_auc = auc(fpr, tpr)

plt.plot(fpr, tpr, label=f'ROC Curve (AUC = {roc\_auc:.2f})')

plt.plot([0, 1], [0, 1], linestyle='--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve')

plt.legend()

plt.show()

# Streamlit Web App

# To run: streamlit run script.py

st.title("Customer Churn Prediction")

# Collect user input

gender = st.selectbox("Gender", ['Male', 'Female'])

tenure = st.slider("Tenure", 0, 72, 12)

monthly\_charges = st.number\_input("Monthly Charges", min\_value=0.0)

contract = st.selectbox("Contract Type", ['Month-to-month', 'One year', 'Two year'])

# Construct input vector

input\_data = pd.DataFrame({

'gender': [gender],

'tenure': [tenure],

'MonthlyCharges': [monthly\_charges],

'Contract\_Month-to-month': [1 if contract == 'Month-to-month' else 0],

'Contract\_One year': [1 if contract == 'One year' else 0],

'Contract\_Two year': [1 if contract == 'Two year' else 0]

})

# Align input\_data columns with training data columns

input\_data = input\_data.reindex(columns=X.columns, fill\_value=0)

# Predict and display result

if st.button("Predict"):

pred = model.predict(input\_data)

st.write("Prediction:", "Churn" if pred[0] == 1 else "Not Churn")