#===============================================

Predicting Customer Churn Using Machine Learning to Uncover Hidden patterns

#===============================================

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.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_auc\_score

import shap

# Sample dataset

data = {

    'customerID': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

    'gender': ['Male', 'Female'] \* 5,

    'SeniorCitizen': ['No', 'Yes'] \* 5,

    'Partner': ['Yes', 'No'] \* 5,

    'Dependents': ['No', 'Yes'] \* 5,

    'tenure': [12, 24, 6, 36, 18, 12, 24, 6, 36, 18],

    'PhoneService': ['Yes', 'No'] \* 5,

    'MultipleLines': ['Yes', 'No'] \* 5,

    'InternetService': ['DSL', 'Fiber optic'] \* 5,

    'OnlineSecurity': ['Yes', 'No'] \* 5,

    'OnlineBackup': ['Yes', 'No'] \* 5,

    'DeviceProtection': ['Yes', 'No'] \* 5,

    'TechSupport': ['Yes', 'No'] \* 5,

    'StreamingTV': ['Yes', 'No'] \* 5,

    'StreamingMovies': ['Yes', 'No'] \* 5,

    'Contract': ['Month-to-month', 'One year'] \* 5,

    'PaperlessBilling': ['Yes', 'No'] \* 5,

    'PaymentMethod': ['Credit card', 'Bank transfer'] \* 5,

    'MonthlyCharges': [50, 60, 40, 70, 55, 65, 45, 75, 50, 60],

    'TotalCharges': [600, 720, 480, 840, 660, 780, 540, 900, 600, 720],

    'Churn': ['No', 'Yes'] \* 5

}

df = pd.DataFrame(data)

# Drop customerID

df.drop("customerID", axis=1, inplace=True)

# Encode categorical variables

for column in df.select\_dtypes(include=['object']).columns:

    if column != 'Churn':

        if df[column].nunique() == 2:

            df[column] = LabelEncoder().fit\_transform(df[column])

        else:

            df = pd.get\_dummies(df, columns=[column])

# Encode target

df['Churn'] = LabelEncoder().fit\_transform(df['Churn'])

# Features and target

X = df.drop("Churn", axis=1)

y = df["Churn"]

# Split

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

# Scale

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Train model

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

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

# Evaluate

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

print(confusion\_matrix(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

print("ROC AUC Score:", roc\_auc\_score(y\_test, model.predict\_proba(X\_test\_scaled)[:, 1]))