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.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Load the dataset

df = pd.read\_csv('bank-additional-full.csv', sep=';')

# Preprocessing

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

df['y'] = np.where(df['deposit'] == 'yes', 1, 0)

df['age'] = df['age'].astype(int)

df['duration'] = df['duration'].astype(int)

df['campaign'] = df['campaign'].astype(int)

df['pdays'] = df['pdays'].astype(int)

df['previous'] = df['previous'].astype(int)

df['poutcome'] = df['poutcome'].astype('category').cat.codes

df['job'] = df['job'].astype('category').cat.codes

df['marital'] = df['marital'].astype('category').cat.codes

df['education'] = df['education'].astype('category').cat.codes

df['default'] = df['default'].astype('category').cat.codes

df['housing'] = df['housing'].astype('category').cat.codes

df['loan'] = df['loan'].astype('category').cat.codes

df['contact'] = df['contact'].astype('category').cat.codes

df['day\_of\_week'] = df['day\_of\_week'].astype('category').cat.codes

df['month'] = df['month'].astype('category').cat.codes

df['emp.var.rate'] = df['emp.var.rate'].astype(float)

df['cons.price.idx'] = df['cons.price.idx'].astype(float)

df['cons.conf.idx'] = df['cons.conf.idx'].astype(float)

df['euribor3m'] = df['euribor3m'].astype(float)

df['nr.employed'] = df['nr.employed'].astype(int)

# Data visualization

sns.countplot(x='y', data=df)

plt.xlabel('Subscribed to Term Deposit')

plt.ylabel('Count')

plt.title('Distribution of Subscription')

plt.show()

# Feature and target variables

X = df.drop('y', axis=1)

y = df['y']

# Splitting the 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)

# Building the decision tree classifier

clf = DecisionTreeClassifier(random\_state=42)

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

# Predicting the target variable

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

# Evaluating the model

print('Accuracy:', accuracy\_score(y\_test, y\_pred))

print('\nClassification Report:\n', classification\_report(y\_test, y\_pred))