Siddardha vemula

BL.EN.U4AIE23167

import numpy as np

import pandas as pd

import statistics

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing import LabelEncoder, OneHotEncoder, MinMaxScaler, StandardScaler

data\_file = "Lab Session Data.xlsx"

Question A1:

df\_purchase = pd.read\_excel(data\_file, sheet\_name="Purchase Data")

A = df\_purchase.iloc[:, :-1].values

C = df\_purchase.iloc[:, -1].values

vector\_space\_dim = A.shape[1]

num\_vectors = A.shape[0]

rank\_A = np.linalg.matrix\_rank(A)

pseudo\_inv\_A = np.linalg.pinv(A)

cost\_vector = np.dot(pseudo\_inv\_A, C)

Question A2:

X = np.dot(pseudo\_inv\_A, C)

Question A3:

df\_purchase["Customer\_Type"] = ["RICH" if x > 200 else "POOR" for x in C]

Question A4:

df\_stock = pd.read\_excel(data\_file, sheet\_name="IRCTC Stock Price")

price\_data = df\_stock["Price Column D"]

mean\_price = statistics.mean(price\_data)

variance\_price = statistics.variance(price\_data)

wednesday\_data = df\_stock[df\_stock["Day"] == "Wednesday"]["Price Column D"]

sample\_mean\_wednesday = statistics.mean(wednesday\_data)

df\_stock["Loss\_Prob"] = df\_stock["Chg% Column I"].apply(lambda x: x < 0)

prob\_loss = df\_stock["Loss\_Prob"].mean()

plt.figure(figsize=(8, 6))

sns.scatterplot(x=df\_stock["Day"], y=df\_stock["Chg% Column I"])

plt.xlabel("Day of the Week")

plt.ylabel("Chg%")

plt.title("Stock Change Percentage vs Day of the Week")

plt.show()

Question A5:

df\_thyroid = pd.read\_excel(data\_file, sheet\_name="thyroid0387\_UCI")

attribute\_types = df\_thyroid.dtypes

missing\_values = df\_thyroid.isnull().sum()

outlier\_detection = df\_thyroid.describe()

label\_enc = LabelEncoder()

one\_hot\_enc = OneHotEncoder()

for col in df\_thyroid.select\_dtypes(include=["object"]).columns:

df\_thyroid[col] = label\_enc.fit\_transform(df\_thyroid[col])

Question A6:

df\_thyroid.fillna(df\_thyroid.mean(), inplace=True)

Question A7:

scaler = MinMaxScaler()

df\_thyroid\_normalized = scaler.fit\_transform(df\_thyroid)

Question A8:

vec1, vec2 = df\_thyroid.iloc[0, :], df\_thyroid.iloc[1, :]

f11 = np.sum((vec1 == 1) & (vec2 == 1))

f00 = np.sum((vec1 == 0) & (vec2 == 0))

f01 = np.sum((vec1 == 0) & (vec2 == 1))

f10 = np.sum((vec1 == 1) & (vec2 == 0))

JC = f11 / (f01 + f10 + f11)

SMC = (f11 + f00) / (f00 + f01 + f10 + f11)

Question A9:

cos\_sim = np.dot(vec1, vec2) / (np.linalg.norm(vec1) \* np.linalg.norm(vec2))

Question A10:

similarity\_matrix = np.zeros((20, 20))

for i in range(20):

for j in range(20):

vec1, vec2 = df\_thyroid.iloc[i, :], df\_thyroid.iloc[j, :]

similarity\_matrix[i, j] = np.dot(vec1, vec2) / (np.linalg.norm(vec1) \* np.linalg.norm(vec2))

plt.figure(figsize=(10, 8))

sns.heatmap(similarity\_matrix, annot=True, cmap="coolwarm")

plt.title("Heatmap of Similarity Measures")

plt.show()