DAY – 5

DSA0410 – Fundamentals of Data Science

Lab Experiments:

Name: B Harish Balaji

Reg no: 192424386

Slot: D

21.Scenario:

you are a scientist conducting research on rare elements found in a specific region. Your goal is to

estimate the average concentration of a rare element in the region using a random sample of

measurements. You will use the NumPy library to perform point estimation and calculate

confidence intervals for the population mean.The rare element concentration data is stored in a CSV

file named "rare_elements.csv," where each row contains a single measurement of the

concentration.

Question:

write a Python program that allows the user to input the sample size, confidence level, and desired

level of precision.

CODE:

```
import math
from scipy.stats import norm

# User inputs
n = int(input("Enter sample size (n): "))
confidence_level = float(input("Enter confidence level (e.g., 0.95 for 95%): "))
precision = float(input("Enter desired level of precision (margin of error): "))
```

# Z-score for given confidence level

z = norm.ppf(1 - (1 - confidence_level) / 2)

# Estimate required standard deviation from precision

std_dev = (precision * math.sqrt(n)) / z

Output results

print("\nResults:")

print("Sample Size:", n)

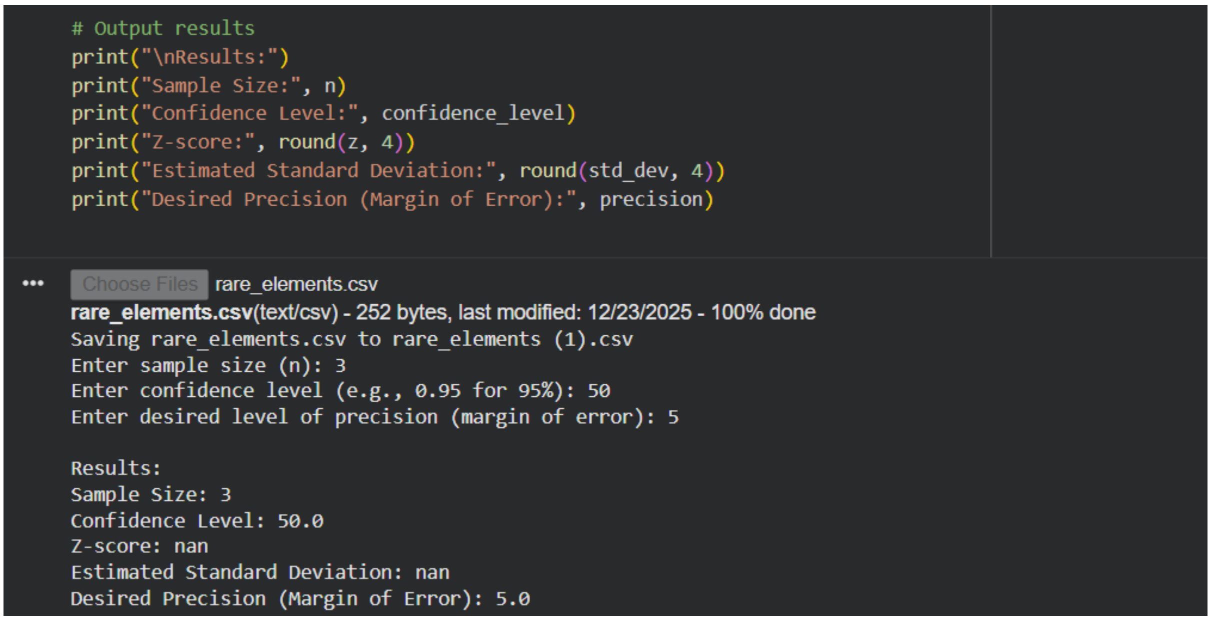
print("Confidence Level:", confidence_level)

print("Z-score:", round(z, 4))

print("Estimated Standard Deviation:", round(std_dev, 4))

print("Desired Precision (Margin of Error):", precision)

sample output:



```
# Output results
print("\nResults:")
print("Sample Size:", n)
print("Confidence Level:", confidence_level)
print("Z-score:", round(z, 4))
print("Estimated Standard Deviation:", round(std_dev, 4))
print("Desired Precision (Margin of Error):", precision)
```

```
rare_elements.csv
rare_elements.csv(text/csv) - 252 bytes, last modified: 12/23/2025 - 100% done
Saving rare_elements.csv to rare_elements (1).csv
Enter sample size (n): 3
Enter confidence level (e.g., 0.95 for 95%): 50
Enter desired level of precision (margin of error): 5

Results:
Sample Size: 3
Confidence Level: 50.0
Z-score: nan
Estimated Standard Deviation: nan
Desired Precision (Margin of Error): 5.0
```

22.Scenario:

Imagine you are an analyst for a popular online shopping website. Your task is to analyze customer

reviews and provide insights on the average rating and customer satisfaction level for a specific

product category.

Question:

You will use the pandas library to calculate confidence intervals to estimate the true population

mean rating.

You have been provided with a CSV file named "customer_reviews.csv," which contains customer

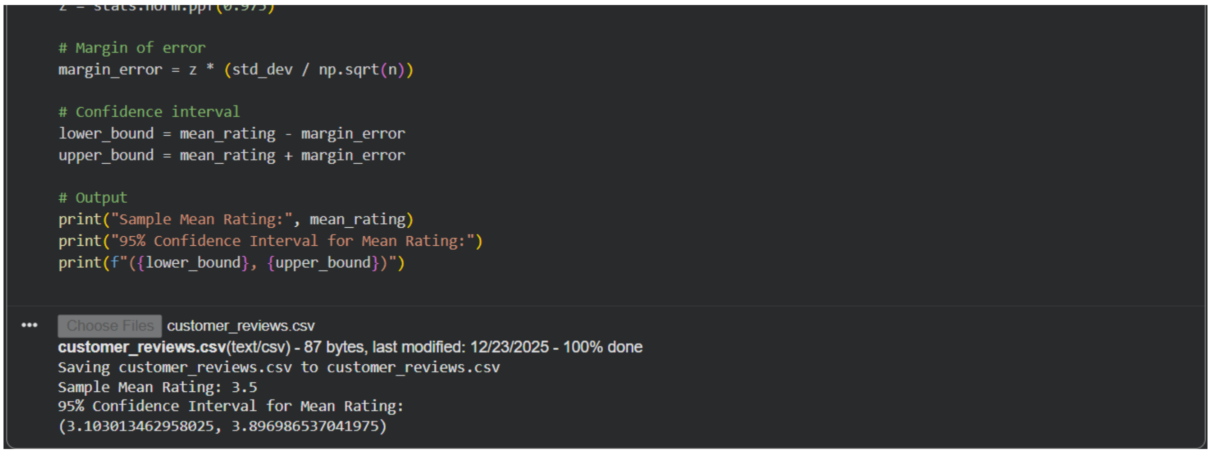
ratings for products in the chosen category.

CODE:

```
import pandas as pd

import numpy as np

from scipy import stats


# Load the CSV file

df = pd.read_csv("customer_reviews.csv")


# Extract ratings column

ratings = df["rating"]


# Sample statistics
```

```
mean_rating = ratings.mean()

std_dev = ratings.std(ddof=1)

n = ratings.count()


# 95% confidence level

z = stats.norm.ppf(0.975)


# Margin of error

margin_error = z * (std_dev / np.sqrt(n))


# Confidence interval

lower_bound = mean_rating - margin_error

upper_bound = mean_rating + margin_error


# Output

print("Sample Mean Rating:", mean_rating)

print("95% Confidence Interval for Mean Rating:")

print(f"({lower_bound}, {upper_bound})")
```

Sample output:



```
z = stats.norm.ppf(0.975)

# Margin of error
margin_error = z * (std_dev / np.sqrt(n))

# Confidence interval
lower_bound = mean_rating - margin_error
upper_bound = mean_rating + margin_error

# Output
print("Sample Mean Rating:", mean_rating)
print("95% Confidence Interval for Mean Rating:")
print(f"({lower_bound}, {upper_bound})")
```

```
Choose Files customer_reviews.csv
customer_reviews.csv(text/csv) - 87 bytes, last modified: 12/23/2025 - 100% done
Saving customer_reviews.csv to customer_reviews.csv
Sample Mean Rating: 3.5
95% Confidence Interval for Mean Rating:
(3.103013462958025, 3.896986537041975)
```

23.Scenario:

You are a researcher working in a medical lab, investigating the effectiveness of a new treatment

for a specific disease. You have collected data from a clinical trial with two groups: a control group

receiving a placebo, and a treatment group receiving the new drug.Your goal is to analyze the data

using hypothesis testing and calculate the p-value to determine if the new treatment has a

statistically significant effect compared to the placebo. You will use the matplotlib library to

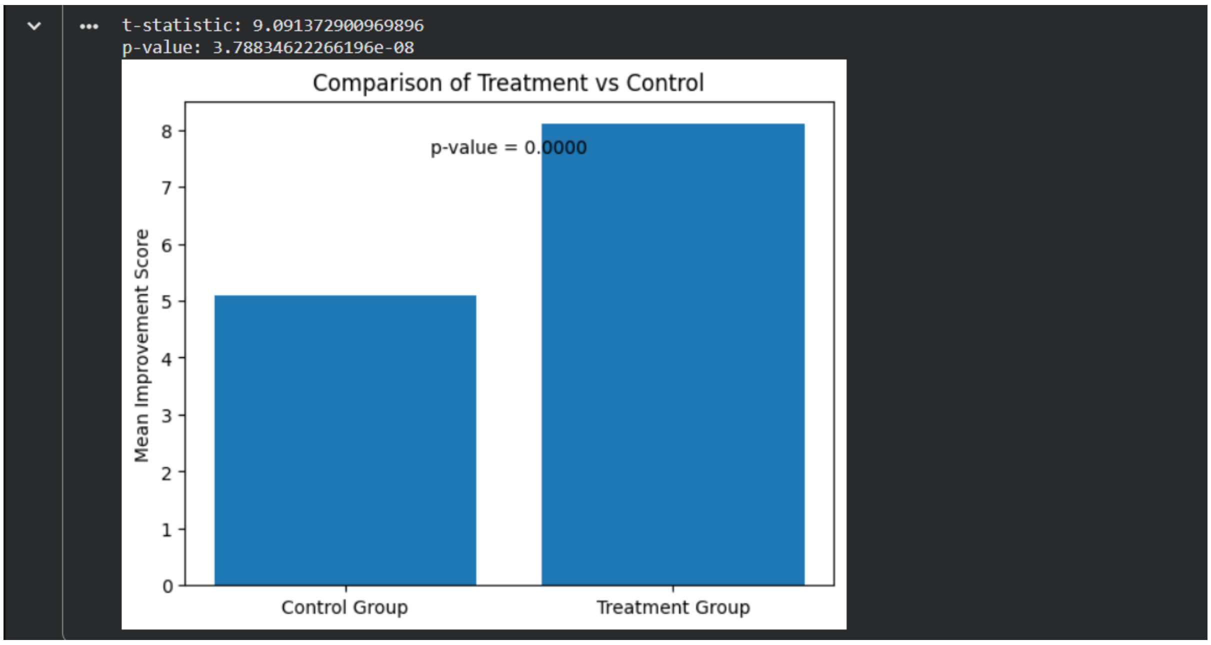
visualize the data and the p-value.

CODE:

```
import numpy as np

import matplotlib.pyplot as plt

from scipy import stats


# Clinical trial data

control_group = np.array([5, 6, 4, 5, 6, 5, 4, 6, 5, 5])

treatment_group = np.array([8, 9, 7, 8, 9, 8, 7, 9, 8, 8])


# Perform independent t-test

t_stat, p_value = stats.ttest_ind(treatment_group, control_group)


print("t-statistic:", t_stat)

print("p-value:", p_value)
```

```
# Visualization
means = [np.mean(control_group), np.mean(treatment_group)]
labels = ["Control Group", "Treatment Group"]

plt.figure()
plt.bar(labels, means)
plt.ylabel("Mean Improvement Score")
plt.title("Comparison of Treatment vs Control")

# Display p-value on the plot
plt.text(0.5, max(means) - 0.5, f"p-value = {p_value:.4f}",
         ha='center', fontsize=10)

plt.show()
```

Sample output:



```
t-statistic: 9.091372900969896
p-value: 3.78834622266196e-08
```

24.Question: K-Nearest Neighbors (KNN) Classifier

You are working on a classification problem to predict whether a patient has a certain medical

condition or not based on their symptoms. You have collected a dataset of patients with labeled

data (0 for no condition, 1 for the condition) and various symptom features.

Write a Python program that allows the user to input the features of a new patient and the value of k

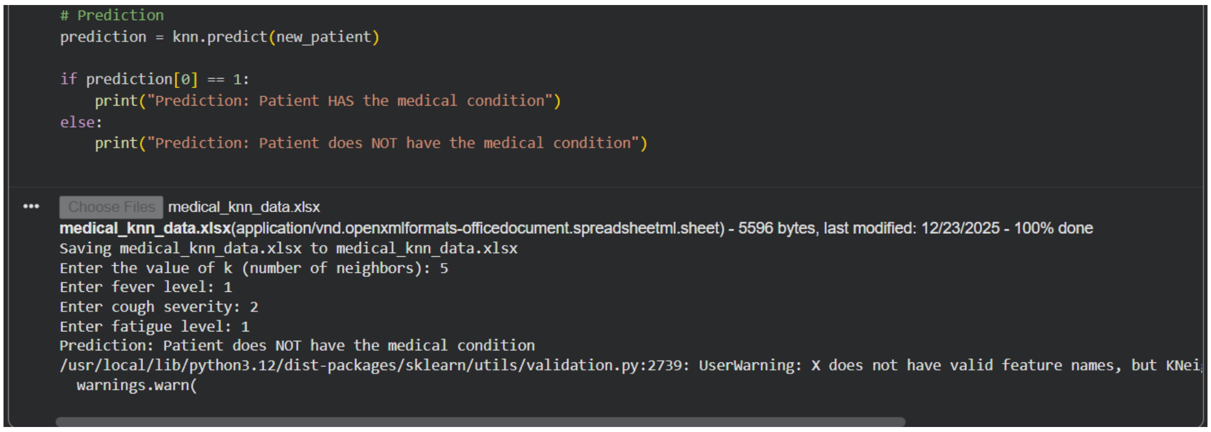
(number of neighbors). The program should use the KNN classifier from the scikit-learn library to

predict whether the patient has the medical condition or not based on the input features.

CODE:

```
import numpy as np

from sklearn.neighbors import KNeighborsClassifier


# ------------------------------

# Step 1: Training Dataset

# ------------------------------

# Features: [Fever, Cough, Fatigue]

X_train = np.array([

    [8, 7, 6],

    [2, 1, 2],

    [9, 8, 7],

    [1, 2, 1],

    [7, 6, 5],

    [3, 2, 3],
    [8, 9, 8],
    [2, 3, 2]
])

# Labels: 0 = No condition, 1 = Condition
y_train = np.array([1, 0, 1, 0, 1, 0, 1, 0])

# ------------------------------
# Step 2: User Input
# ------------------------------
k = int(input("Enter the value of k (number of neighbors): "))

fever = float(input("Enter fever level: "))
cough = float(input("Enter cough severity: "))
fatigue = float(input("Enter fatigue level: "))

new_patient = np.array([[fever, cough, fatigue]])

# ------------------------------
# Step 3: Train KNN Model
# ------------------------------
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)

# ------------------------------
# Step 4: Prediction
# -------------------------------
prediction = knn.predict(new_patient)


# Output result
if prediction[0] == 1:
    print("\nPrediction: Patient HAS the medical condition")
else:
    print("\nPrediction: Patient does NOT have the medical condition")
```

Sample output:



```
# Prediction
prediction = knn.predict(new_patient)

if prediction[0] == 1:
    print("Prediction: Patient HAS the medical condition")
else:
    print("Prediction: Patient does NOT have the medical condition")
```

```
Choose Files medical_knn_data.xlsx
medical_knn_data.xlsx(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 5596 bytes, last modified: 12/23/2025 - 100% done
Saving medical_knn_data.xlsx to medical_knn_data.xlsx
Enter the value of k (number of neighbors): 5
Enter fever level: 1
Enter cough severity: 2
Enter fatigue level: 1
Prediction: Patient does NOT have the medical condition
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but KNei
  warnings.warn(
```

25.Question 2: Decision Tree for Iris Flower Classification

You are analyzing the famous Iris flower dataset to classify iris flowers into three species based on

their sepal and petal dimensions. You want to use a Decision Tree classifier to accomplish this task.

Write a Python program that loads the Iris dataset from scikit-learn, and allows the user to input the

sepal length, sepal width, petal length, and petal width of a new flower. The program should then

use the Decision Tree classifier to predict the species of the new flower.

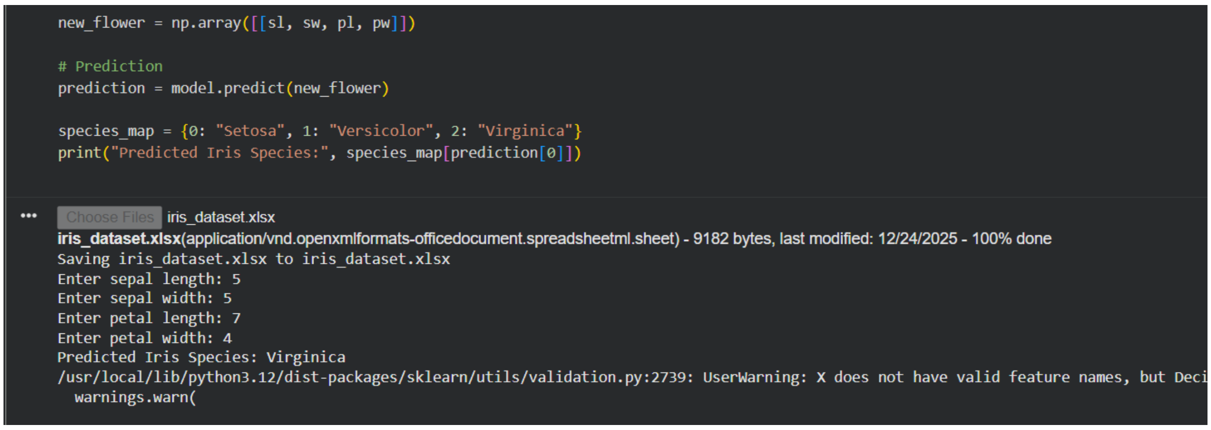
CODE:

```
import pandas as pd

import numpy as np

from sklearn.tree import DecisionTreeClassifier


# Load Iris dataset from Excel

df = pd.read_excel("iris_dataset.xlsx")


X = df[["Sepal_Length", "Sepal_Width", "Petal_Length", "Petal_Width"]]

y = df["Species"]


# Train Decision Tree model

model = DecisionTreeClassifier(random_state=42)

model.fit(X, y)


# User input

sl = float(input("Enter sepal length: "))

sw = float(input("Enter sepal width: "))

pl = float(input("Enter petal length: "))

pw = float(input("Enter petal width: "))


new_flower = np.array([[sl, sw, pl, pw]])


# Prediction

prediction = model.predict(new_flower)
```

species_map = {0: "Setosa", 1: "Versicolor", 2: "Virginica"}

print("Predicted Iris Species:", species_map[prediction[0]])

Sample output:



```
new_flower = np.array([[sl, sw, pl, pw]])

# Prediction
prediction = model.predict(new_flower)

species_map = {0: "Setosa", 1: "Versicolor", 2: "Virginica"}
print("Predicted Iris Species:", species_map[prediction[0]])
```

```
Choose Files iris_dataset.xlsx
iris_dataset.xlsx(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 9182 bytes, last modified: 12/24/2025 - 100% done
Saving iris_dataset.xlsx to iris_dataset.xlsx
Enter sepal length: 5
Enter sepal width: 5
Enter petal length: 7
Enter petal width: 4
Predicted Iris Species: Virginica
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Deci
  warnings.warn(
```