1. **Scenario** : You are a data scientist working for a company that sells products online. You have been tasked with creating a simple plot to show the sales of a product over time.

**Question**: 1. Write code to create a simple line plot in Python using Matplotlib to predict sales happened in a month?

1. Write code to create a scatter plot in Python using Matplotlib to predict sales happened in a month?
2. Develop a Python program to create a bar plot of the monthly sales data.

**Program:**

import matplotlib.pyplot as plt

months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun"]

sales = [1000, 1200, 1100, 1300, 1400, 1600]

plt.figure(figsize=(10, 6))

plt.plot(months, sales, marker='o', linestyle='-', color='b', label='Monthly Sales')

plt.title('Monthly Sales Over Time')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.legend()

plt.grid(True)

plt.show()

plt.figure(figsize=(10, 6))

plt.scatter(months, sales, color='g', label='Monthly Sales', marker='o')

plt.title('Monthly Sales Scatter Plot')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.legend()

plt.grid(True)

plt.show()

plt.figure(figsize=(10, 6))

plt.bar(months, sales, color='r', label='Monthly Sales')

plt.title('Monthly Sales Bar Plot')

plt.xlabel('Month')

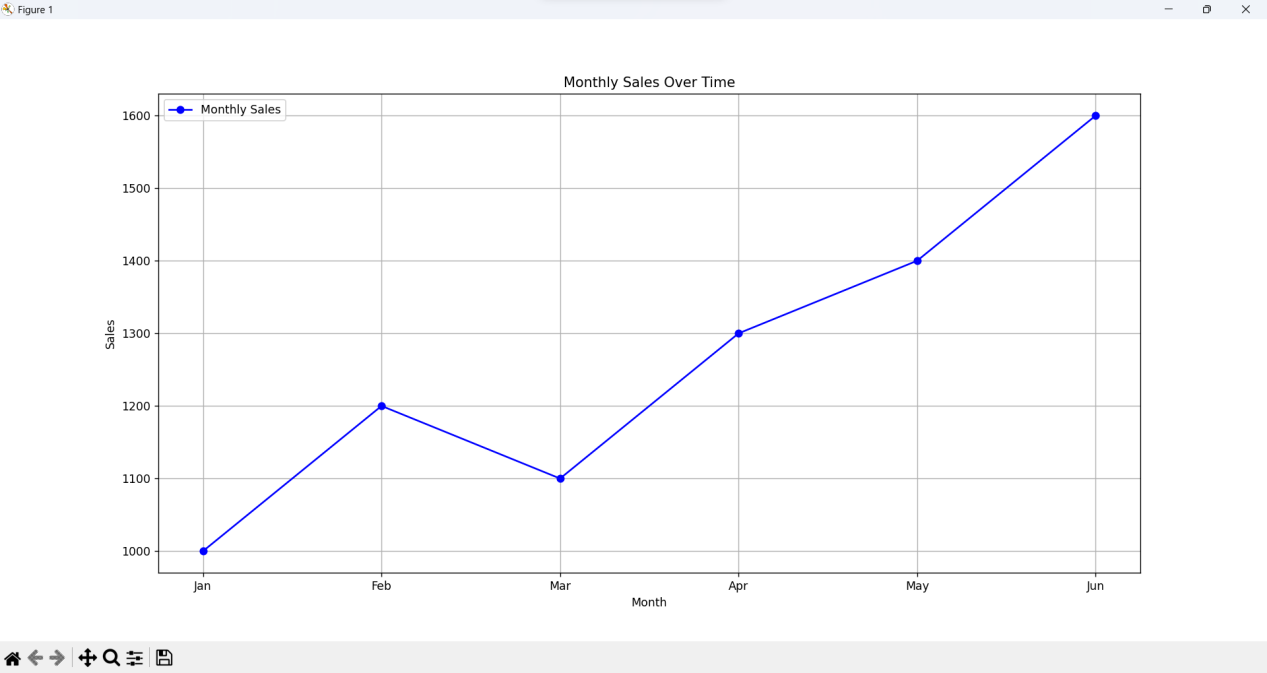
plt.ylabel('Sales')

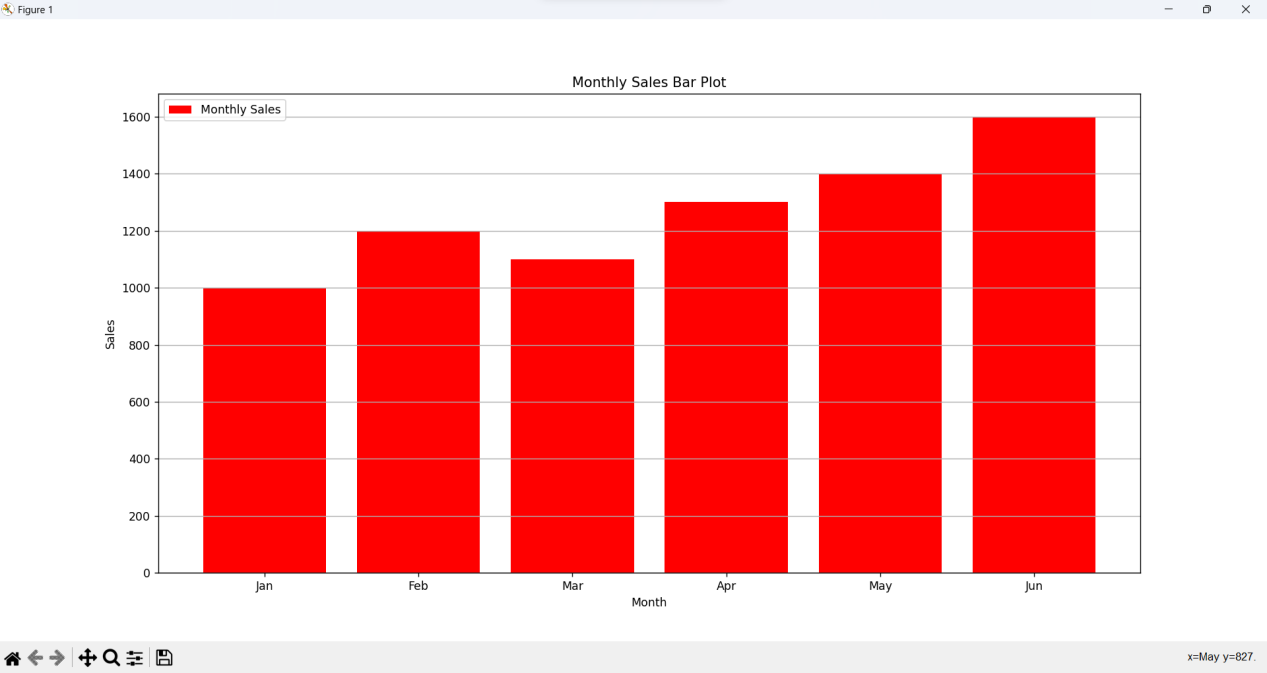
plt.legend()

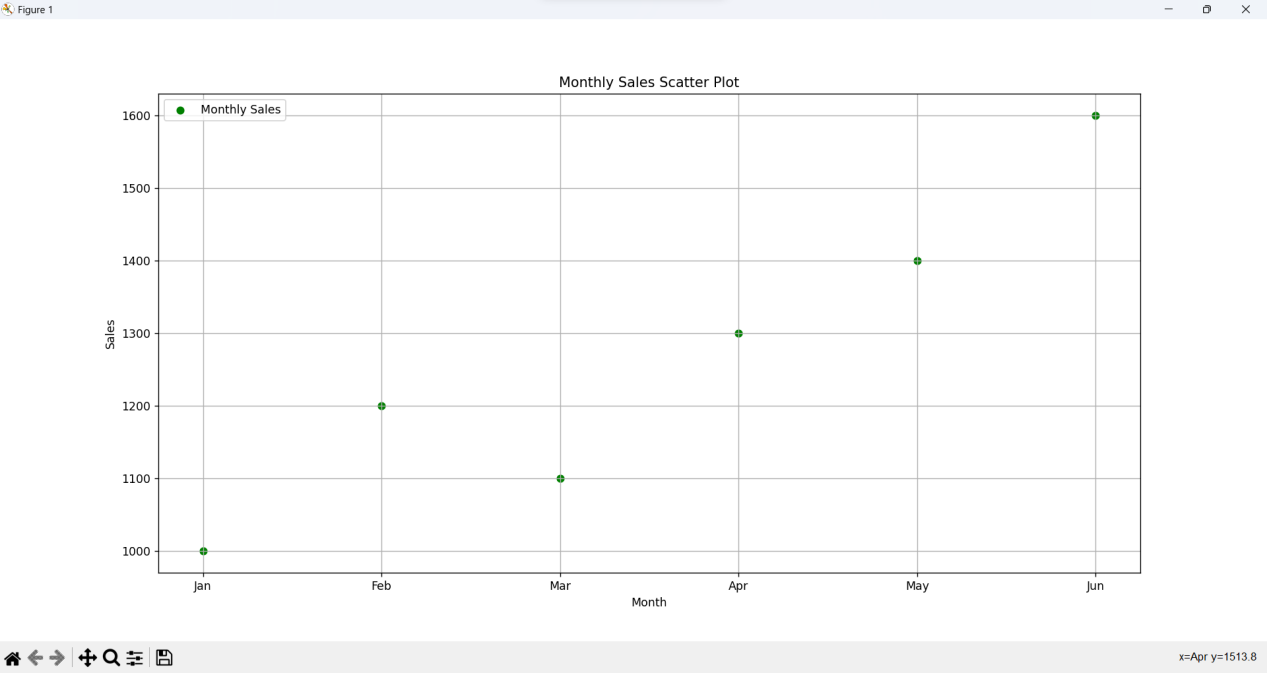
plt.grid(axis='y')

plt.show()

**Output:**







1. **Scenario**: You are working on a data analysis project that involves analyzing the monthly temperature and rainfall data for a city. You have a dataset containing the monthly temperature and rainfall values for each month of a year. Your task is to develop a Python program that generates line plots and scatter plots to visualize the temperature and rainfall data.

**Question:** 1. Develop a Python program to create a line plot of the monthly temperature data.

2: Develop a Python program to create a scatter plot of the monthly rainfall data.

**Program:**

import matplotlib.pyplot as plt

months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]

rainfall = [50, 45, 60, 80, 100, 120, 130, 110, 90, 80, 70, 60]

temperature = [10, 12, 15, 18, 22, 26, 29, 28, 25, 20, 15, 12]

plt.figure(figsize=(10, 6))

plt.plot(months, temperature, marker='o', linestyle='-', color='b', label='Temperature (°C)')

plt.title('Monthly Temperature Data')

plt.xlabel('Month')

plt.ylabel('Temperature (°C)')

plt.legend()

plt.grid(True)

plt.show()

plt.figure(figsize=(10, 6))

plt.scatter(months, rainfall, color='g', label='Rainfall (mm)', marker='o')

plt.title('Monthly Rainfall Data')

plt.xlabel('Month')

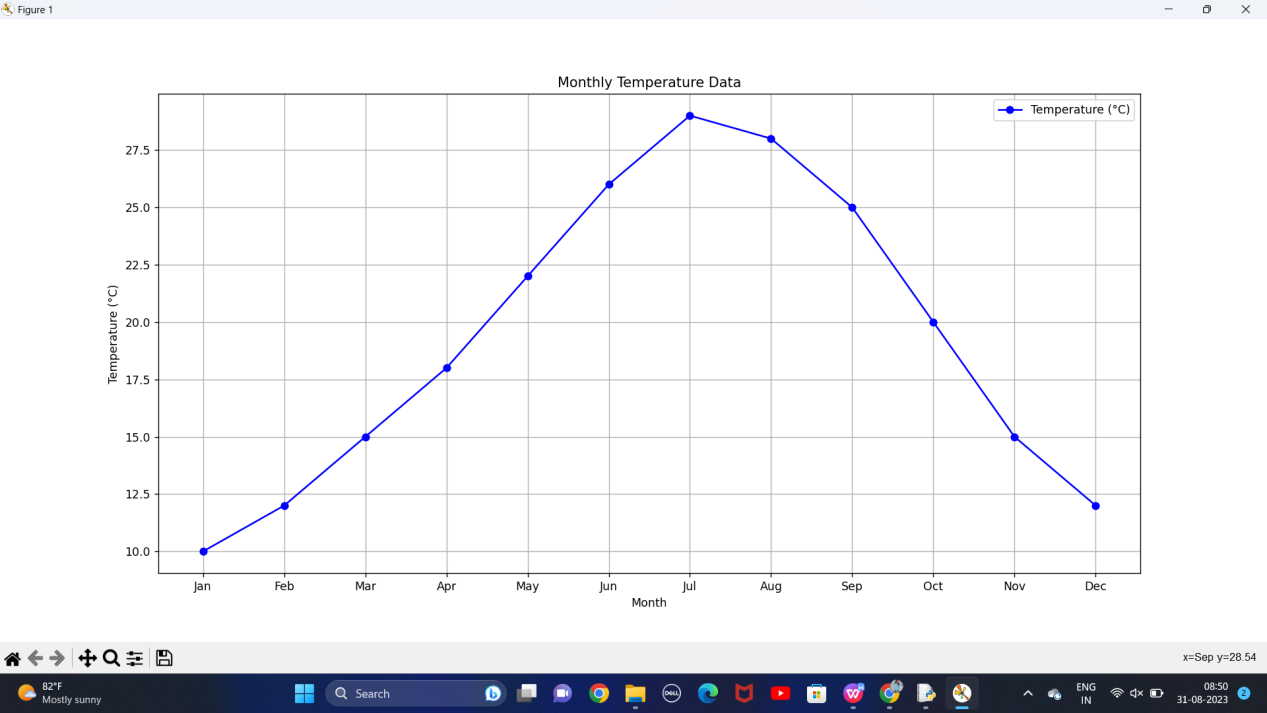
plt.ylabel('Rainfall (mm)')

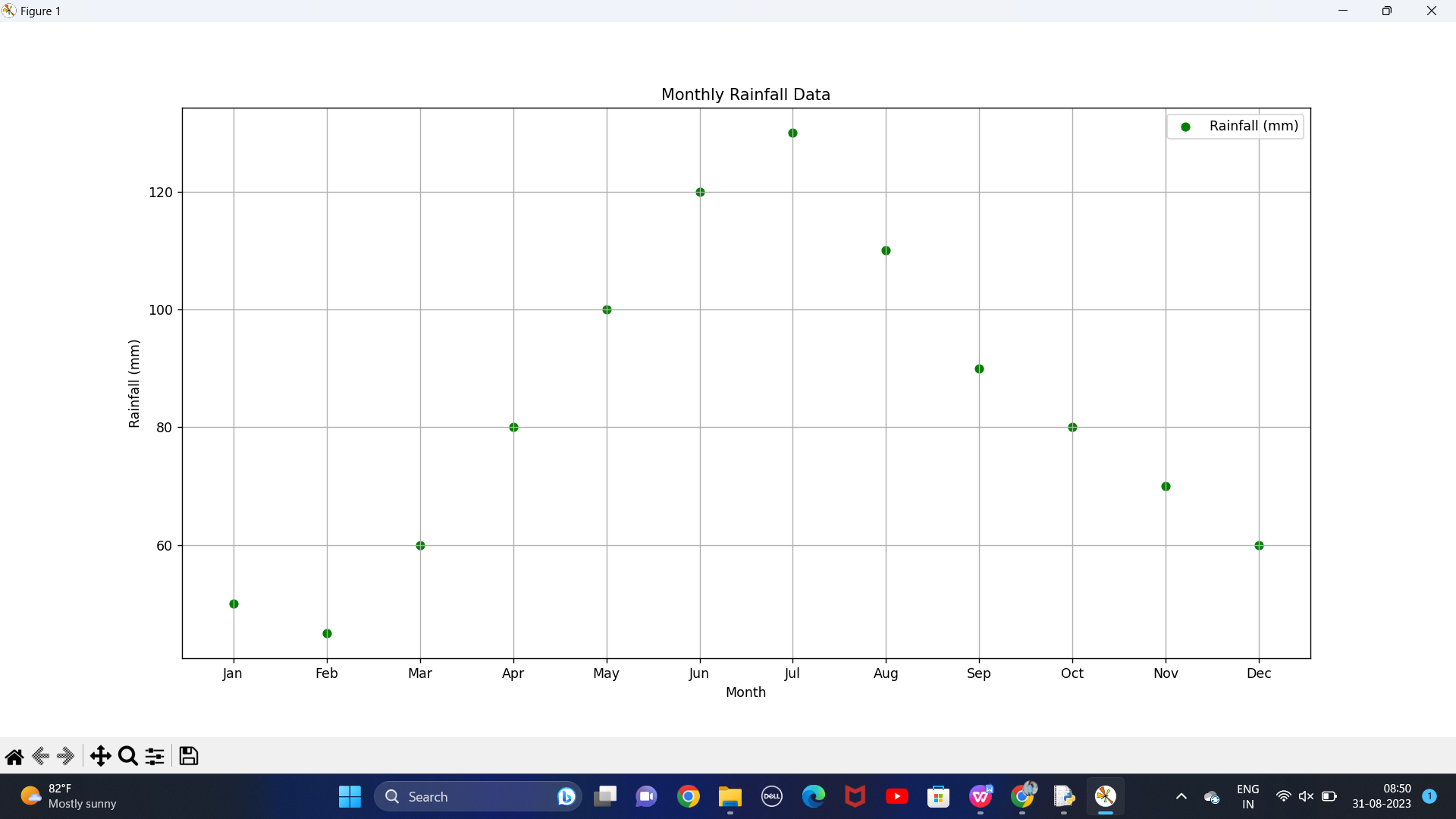
plt.legend()

plt.grid(True)

plt.show()

**Output:**





1. **Scenario**: You are working on a text analysis project and need to determine the frequency distribution of words in a given text document. You have a text document named "sample\_text.txt" containing a paragraph of text. Your task is to develop a Python program that reads the text document, processes the text, and generates a frequency distribution of the words.

**Question**: How would you develop a Python program to calculate the frequency distribution of words in a text document?

**Program:**

import re

from collections import defaultdict

with open("sample.txt", "r") as file:

text = file.read()

words = text.split()

word\_freq = defaultdict(int)

for word in words:

word = re.sub(r'[^\w\s]', '', word).lower()

if word:

word\_freq[word] += 1

sorted\_word\_freq = dict(sorted(word\_freq.items(), key=lambda x: x[1], reverse=True))

for word, freq in sorted\_word\_freq.items():

print(f"{word}: {freq}")

**Output:**

please: 1

enjoy: 1

these: 1

great: 1

stories: 1

fairytales: 1

fables: 1

and: 1

nursery: 1

rhymes: 1

for: 1

children: 1

1. **Scenario**: You are a data analyst working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a Pandas data frame.

**Question:** Develop a code in python to find the frequency distribution of the ages of the customers who have made a purchase in the past month.

**Program:**

import pandas as pd

data = {

'customer\_age': [25, 30, 35, 25, 40, 30, 35, 30, 45, 25, 35, 40]

}

sales\_data = pd.DataFrame(data)

age\_freq = sales\_data['customer\_age'].value\_counts().reset\_index()

age\_freq.columns = ['Age', 'Frequency']

age\_freq = age\_freq.sort\_values(by='Age')

print(age\_freq)

**Output:**

Age Frequency

0 25 3

1 30 3

2 35 3

3 40 2

4 45 1

1. **Scenario:** You are a data analyst working for a social media platform. As part of your analysis, you have a dataset containing user interaction data, including the number of likes received by each post. Your task is to develop a Python program that calculates the frequency distribution of likes among the posts.

**Question:** Develop a Python program to calculate the frequency distribution of likes among the posts?

**Program:**

import pandas as pd

data = {

'post\_id': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

'likes': [50, 120, 80, 35, 35, 90, 150, 150, 110, 110]

}

df = pd.DataFrame(data)

likes\_freq = df['likes'].value\_counts().reset\_index()

likes\_freq.columns = ['Likes', 'Frequency']

likes\_freq = likes\_freq.sort\_values(by='Likes')

print(likes\_freq)

**Output:**

Likes Frequency

0 35 2

3 50 1

5 80 1

6 90 1

2 110 2

4 120 1

1 150 2

**16.Scenario:** You are working on a project that involves analyzing customer reviews for a product. You have a dataset containing customer reviews, and your task is to develop a Python program that calculates the frequency distribution of words in the reviews.

**Question:** Develop a Python program to calculate the frequency distribution of words in the customer reviews dataset?

**Program:**

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

from collections import Counter

nltk.download('punkt')

nltk.download('stopwords')

def preprocess\_text(text):

words = word\_tokenize(text.lower())

stop\_words = set(stopwords.words('english'))

words = [word for word in words if word.isalnum() and word not in stop\_words]

return words

def calculate\_word\_frequency(reviews):

word\_freq = Counter()

for review in reviews:

words = preprocess\_text(review)

word\_freq.update(words)

return word\_freq

if \_\_name\_\_ == "\_\_main\_\_":

reviews = [

"This product is amazing. I love it!",

" It is a waste of money. Not Worth for anyone",

"Great value for the price.I love Quality. ",

]

word\_freq = calculate\_word\_frequency(reviews)

print("Top 10 Most Common Words:")

for word, freq in word\_freq.most\_common(10):

print(f"{word}: {freq}")

**Output:**

Top 10 Most Common Words:

love: 2

product: 1

amazing: 1

waste: 1

money: 1

worth: 1

anyone: 1

great: 1

value: 1

quality: 1

1. **Scenario:** You are a data analyst working for a marketing research company. Your team has collected a large dataset containing customer feedback from various social media platforms. The dataset consists of thousands of text entries, and your task is to develop a Python program to analyze the frequency distribution of words in this dataset. Your program should be able to perform the following tasks: Load the dataset from a CSV file (data.csv) containing a single column named "feedback" with each row representing a customer comment. Preprocess the text data by removing punctuation, converting all text to lowercase, and eliminating any stop words (common words like "the," "and," "is," etc. that don't carry significant meaning).Calculate the frequency distribution of words in the preprocessed dataset.Display the top N most frequent words and their corresponding frequencies, where N is provided as user input. Plot a bar graph to visualize the top N most frequent words and their frequencies.

**Question:** Create a Python program that fulfills these requirements and helps your team gain insights from the customer feedback data.

**Program:**

import pandas as pd

import matplotlib.pyplot as plt

from wordcloud import WordCloud

from nltk.corpus import stopwords

from collections import Counter

stop\_words = set(stopwords.words('english'))

def load\_dataset(file\_path):

try:

dataset = pd.read\_csv('C:/Users/krish/OneDrive/Documents/ds/feedback.csv')

return dataset['feedback']

except FileNotFoundError:

print("File not found. Please make sure 'feedback.csv' exists in the specified path.")

exit(1)

def preprocess\_text(text):

text = text.lower()

words = text.split()

words = [word.strip('.,!?()[]{}"\'') for word in words if word not in stop\_words]

return words

def calculate\_word\_frequencies(feedback\_data):

word\_freq = Counter()

for feedback in feedback\_data:

words = preprocess\_text(feedback)

word\_freq.update(words)

return word\_freq

def display\_top\_words(word\_freq, top\_n):

top\_words = word\_freq.most\_common(top\_n)

wordcloud = WordCloud(width=800, height=400, background\_color='white').generate\_from\_frequencies(dict(top\_words))

print(f"Top {top\_n} Most Frequent Words:")

for word, freq in top\_words:

print(f"{word}: {freq}")

plt.figure(figsize=(12, 6))

plt.bar(\*zip(\*top\_words))

plt.xlabel('Words')

plt.ylabel('Frequency')

plt.title(f'Top {top\_n} Most Frequent Words')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.figure(figsize=(10, 5))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis("off")

plt.title(f'Word Cloud of Top {top\_n} Words')

plt.show()

def main():

dataset = load\_dataset('C:/Users/DELL/OneDrive/Documents/feedback.csv')

top\_n = int(input("Enter the number of top words to display: "))

word\_freq = calculate\_word\_frequencies(dataset)

display\_top\_words(word\_freq, top\_n)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output:**

Enter the number of top words to display: 30

Top 30 Most Frequent Words:

product: 4

service: 2

quality: 2

company: 2

worth: 2

amazing: 1

love: 1

it: 1

great: 1

customer: 1

fast: 1

delivery: 1

poor: 1

terrible: 1

experience: 1

highly: 1

recommended: 1

excellent: 1

never: 1

buy: 1

again: 1

awesome: 1

i'm: 1

satisfied: 1

average: 1

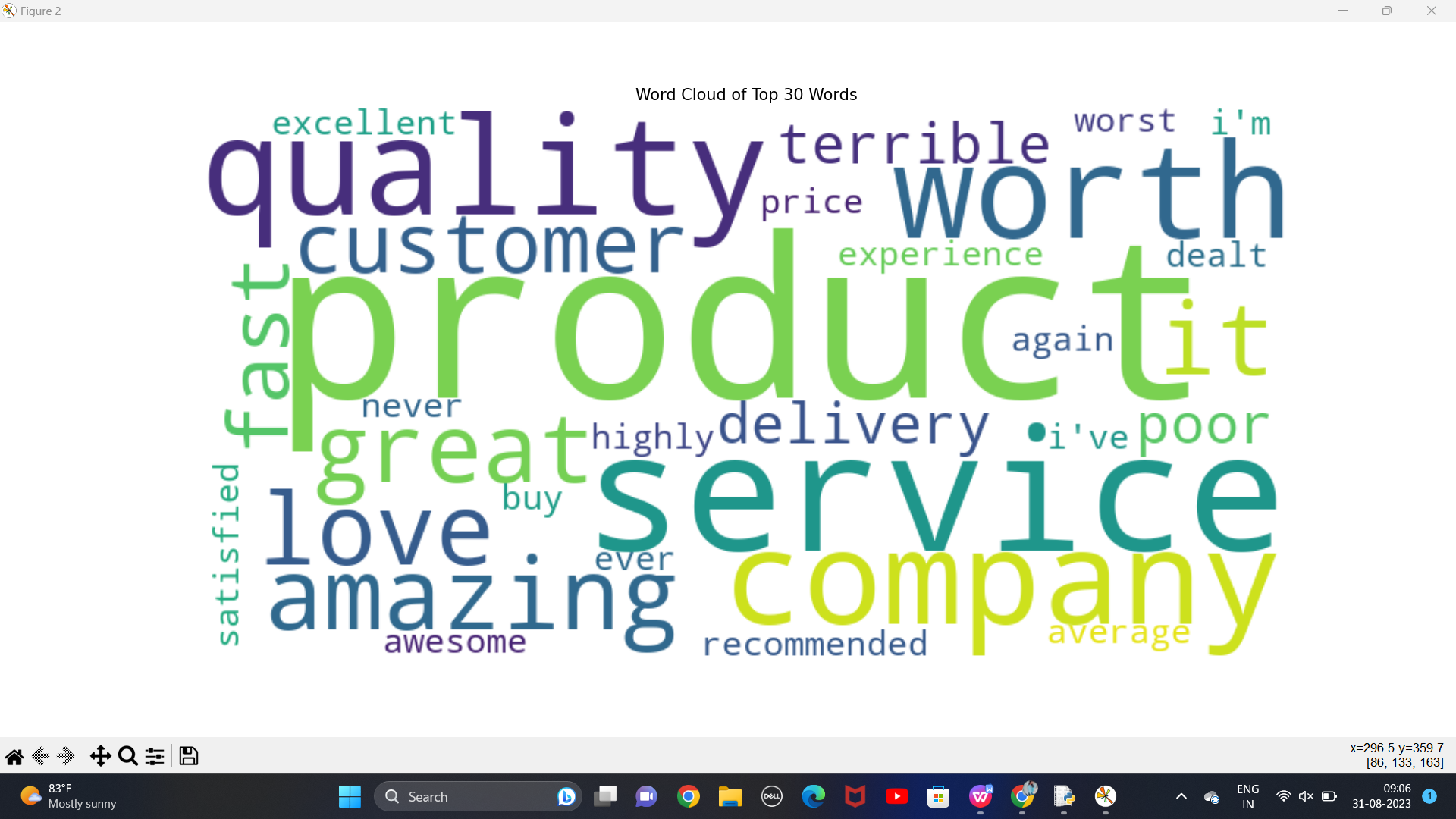
price: 1

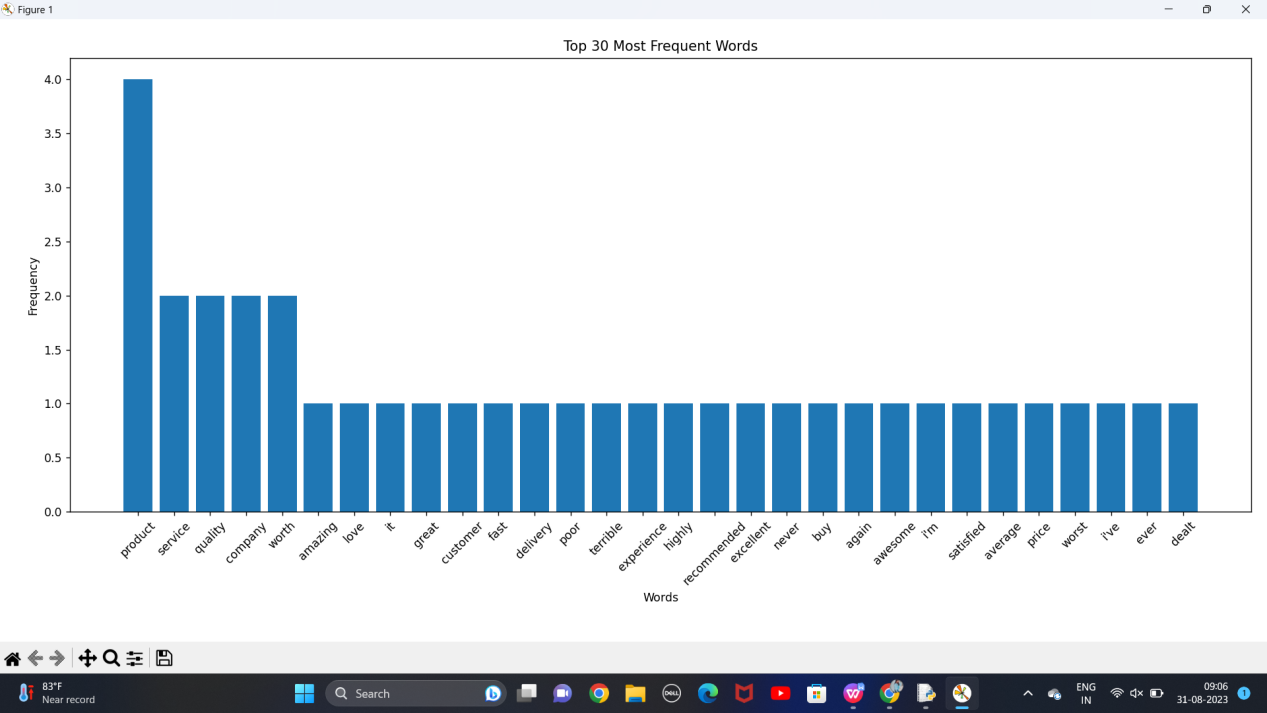
worst: 1

i've: 1

ever: 1

dealt: 1





1. Suppose a hospital tested the age and body fat data for 18 randomly selected adults with the following result.

**Question:** Calculate the mean, median and standard deviation of age and %fat using Pandas.  Draw the boxplots for age and %fat. Draw a scatter plot and a q-q plot based on these two variables.

**Program:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import scipy.stats as stats

data = pd.read\_csv('C:/Users/krish/OneDrive/documents/ds/fat.csv')

mean\_age = data['Age'].mean()

median\_age = data['Age'].median()

std\_dev\_age = data['Age'].std()

mean\_fat = data['%Fat'].mean()

median\_fat = data['%Fat'].median()

std\_dev\_fat = data['%Fat'].std()

print("Age statistics:")

print(f"Mean: {mean\_age}")

print(f"Median: {median\_age}")

print(f"Standard Deviation: {std\_dev\_age}")

print("\n%Fat statistics:")

print(f"Mean: {mean\_fat}")

print(f"Median: {median\_fat}")

print(f"Standard Deviation: {std\_dev\_fat}")

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)

sns.boxplot(y=data['Age'])

plt.title("Box Plot of Age")

plt.subplot(1, 2, 2)

sns.boxplot(y=data['%Fat'])

plt.title("Box Plot of %Fat")

plt.tight\_layout()

plt.show()

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

plt.scatter(data['Age'], data['%Fat'])

plt.xlabel("Age")

plt.ylabel("%Fat")

plt.title("Scatter Plot of Age vs. %Fat")

plt.grid(True)

plt.show()

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

stats.probplot(data['Age'], dist="norm", plot=plt)

plt.title("Q-Q Plot of Age")

plt.show()

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

stats.probplot(data['%Fat'], dist="norm", plot=plt)

plt.title("Q-Q Plot of %Fat")

plt.show()

**Output:**

Age statistics:

Mean: 46.44444444444444

Median: 51.0

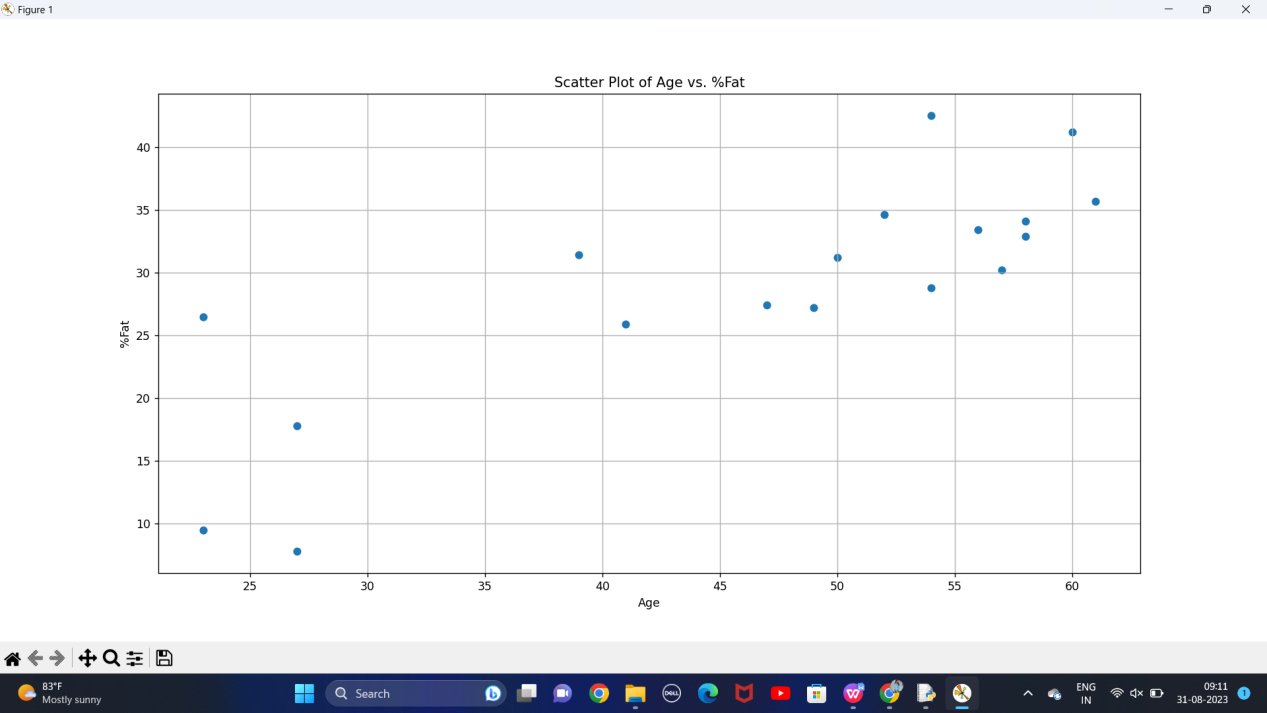
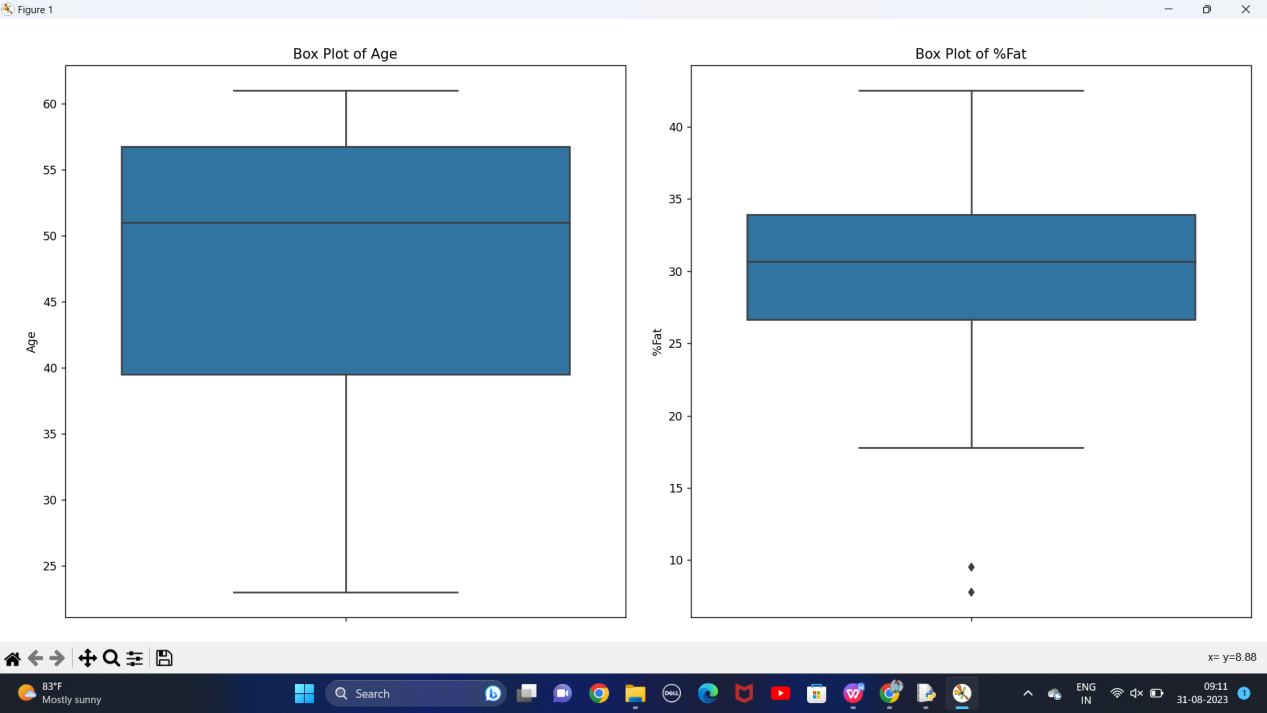
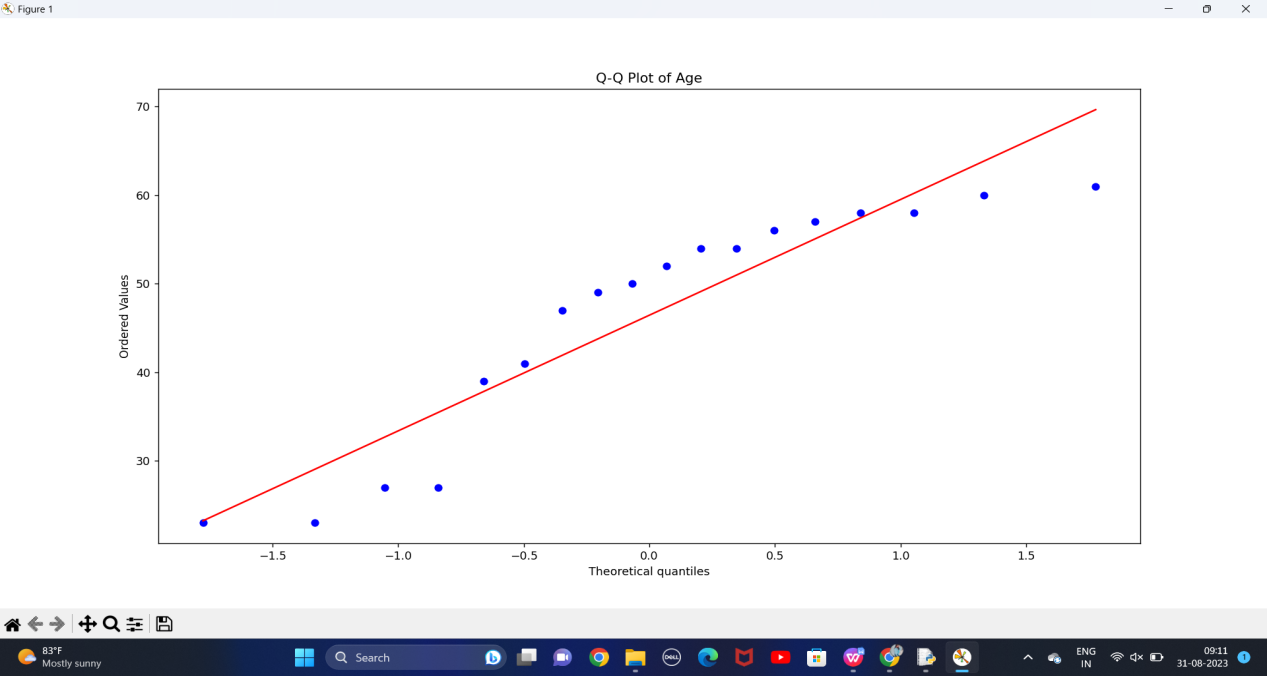
Standard Deviation: 13.218624215242329

%Fat statistics:

Mean: 28.783333333333335

Median: 30.7

Standard Deviation: 9.2543948224296

****

**19.Scenario:** You are a medical researcher investigating the effectiveness of a new drug in reducing blood pressure. You conduct a clinical trial with a sample of 50 patients who were randomly assigned to receive either the new drug or a placebo. After measuring their blood pressure levels at the end of the trial, you obtain the data for both groups. Now, you want to determine the confidence intervals for the mean reduction in blood pressure for both the drug and placebo groups.

**Question:** What is the 95% confidence interval for the mean reduction in blood pressure for patients who received the new drug? Also, what is the 95% confidence interval for the mean reduction in blood pressure for patients who received the placebo?

**Program:**

import numpy as np

from scipy import stats

drug\_group = np.array([10, 12, 9, 8, 11, 10, 9, 9, 11, 10, 9, 8, 11, 12, 10, 11, 9, 10, 11, 10,

9, 10, 11, 10, 9, 8, 11, 10, 9, 10, 12, 11, 10, 9, 10, 11, 10, 9, 8, 11,

10, 12, 11, 10, 9, 10, 11, 10])

placebo\_group = np.array([5, 6, 4, 7, 5, 6, 4, 5, 7, 5, 6, 4, 5, 7, 5, 6, 4, 7, 5, 6,

4, 5, 7, 5, 6, 4, 5, 7, 5, 6, 4, 5, 7, 5, 6, 4, 7, 5, 6,

4, 5, 7, 5, 6, 4, 5, 7])

confidence\_level = 0.95

def calculate\_confidence\_interval(data):

mean = np.mean(data)

std\_dev = np.std(data, ddof=1)

n = len(data)

z\_score = stats.norm.ppf((1 + confidence\_level) / 2)

margin\_of\_error = z\_score \* (std\_dev / np.sqrt(n))

lower\_bound = mean - margin\_of\_error

upper\_bound = mean + margin\_of\_error

return lower\_bound, upper\_bound

lower\_bound\_drug, upper\_bound\_drug = calculate\_confidence\_interval(drug\_group)

lower\_bound\_placebo, upper\_bound\_placebo = calculate\_confidence\_interval(placebo\_group)

print(f"95% Confidence Interval for Drug Group: ({lower\_bound\_drug:.4f}, {upper\_bound\_drug:.4f})")

print(f"95% Confidence Interval for Placebo Group: ({lower\_bound\_placebo:.4f}, {upper\_bound\_placebo:.4f})")

**Output:**

95% Confidence Interval for Drug Group: (9.7149, 10.3268)

95% Confidence Interval for Placebo Group: (5.1231, 5.7280)

**20.Scenario**: You are a data scientist working for an e-commerce company. The marketing team has conducted an A/B test to evaluate the effectiveness of two different website designs (A and B) in terms of conversion rate. They randomly divided the website visitors into two groups, with one group experiencing design A and the other experiencing design B. After a week of data collection, you now have the conversion rate data for both groups. You want to determine whether there is a statistically significant difference in the mean conversion rates between the two website designs.

**Question:** "Based on the data collected from the A/B test, is there a statistically significant difference in the mean conversion rates between website design A and website design B

**Program:**

import numpy as np

import scipy.stats as stats

design\_A = [0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0]

design\_B = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

t\_stat, p\_value = stats.ttest\_ind(design\_A, design\_B)

alpha = 0.05

if p\_value < alpha:

result = "reject the null hypothesis"

else:

result = "fail to reject the null hypothesis"

print(f"t-statistic: {t\_stat:.4f}")

print(f"p-value: {p\_value:.4f}")

print(f"Conclusion: Based on a significance level of {alpha}, we {result}.\n")

if p\_value < alpha:

print("There is a statistically significant difference in the mean conversion rates between website design A and B.")

else:

print("There is no statistically significant difference in the mean conversion rates between website design A and B.")

**Output:**

t-statistic: 0.0000

p-value: 1.0000

Conclusion: Based on a significance level of 0.05, we fail to reject the null hypothesis.

There is no statistically significant difference in the mean conversion rates between website design A and B.