16. Word Frequency Distribution in Customer Reviews

Program:-

```
import string
reviews = [
    "The product is good and useful",
    "I like the product very much",
    "Good quality and good value",
    "The product is not bad"
]
text = " ".join(reviews).lower()
text = text.translate(str.maketrans("", "", string.punctuation))
words = text.split()
freq = {}
for w in words:
    freq[w] = freq.get(w, 0) + 1
print("Word Frequency Distribution:")
for word, count in freq.items():
    print(word, ":", count)
```

Output:-

```
Word Frequency Distribution:
the : 3
product : 3
is : 2
good : 3
and : 2
useful : 1
i : 1
like : 1
very : 1
```

much : 1

quality : 1

value : 1

not : 1

bad : 1

17. Word Frequency Analysis on Large Social Media Feedback Dataset

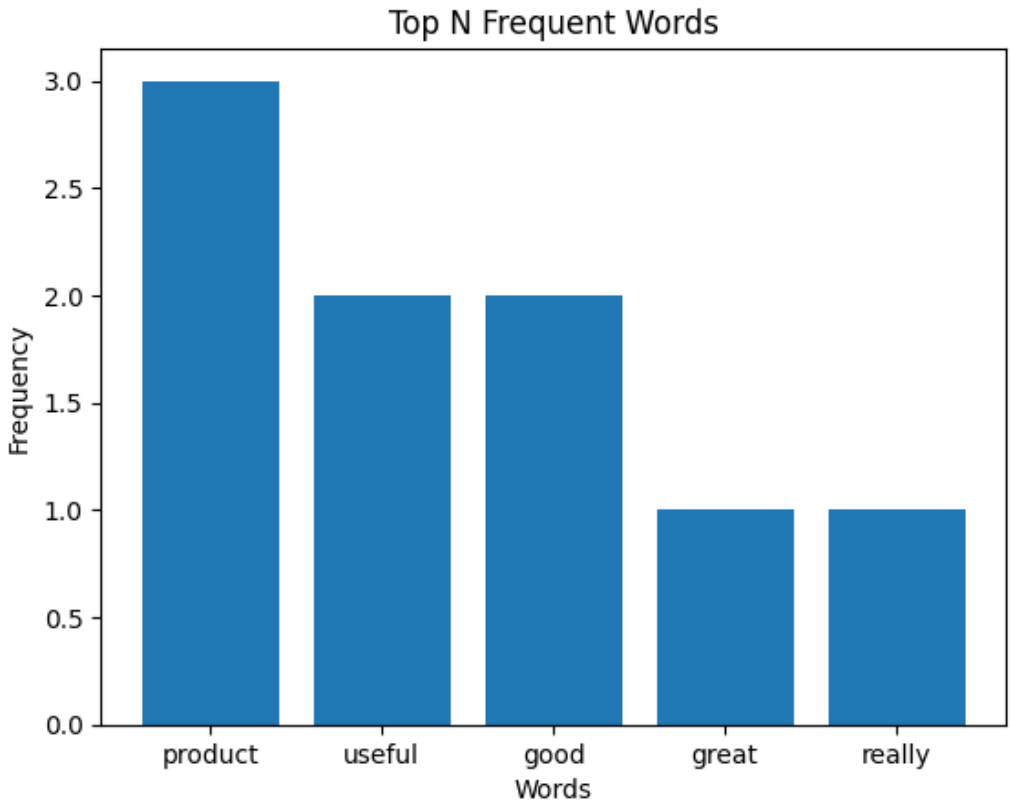
Program:-

```
import pandas as pd
import string
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
import csv
# Create a dummy data.csv file for demonstration
with open('data.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['feedback'])
    writer.writerow(['This product is great and very useful.'])
    writer.writerow(['I really like this product, it is good quality.'])
    writer.writerow(['Not a bad product, but could be better.'])
    writer.writerow(['Very useful and good value for money.'])
df = pd.read_csv("data.csv")
# Ensure NLTK stopwords are downloaded if not already present
try:
    stopwords.words('english')
except LookupError:
    import nltk
    nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
text = " ".join(df["feedback"].astype(str)).lower()
```

```
text = text.translate(str.maketrans('', '', string.punctuation))
words = [w for w in text.split() if w not in stop_words
freq = {}
for w in words:
    freq[w] = freq.get(w, 0) + 1
N = int(input("Enter N: "))
sorted_freq = sorted(freq.items(), key=lambda x: x[1], reverse=True)[:N]
print("Top", N, "words:")
for word, count in sorted_freq:
    print(word, ":", count)
words_list = [w[0] for w in sorted_freq]
count_list = [w[1] for w in sorted_freq]
plt.bar(words_list, count_list)
plt.title("Top N Frequent Words")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.show()
```

Output:-

```
Enter N: 5
Top 5 words:
product : 3
useful : 2
good : 2
great : 1
really : 1
```



18. Age and Body Fat Analysis (Mean, Median, SD, Boxplot, Scatter, Q–Q Plot)

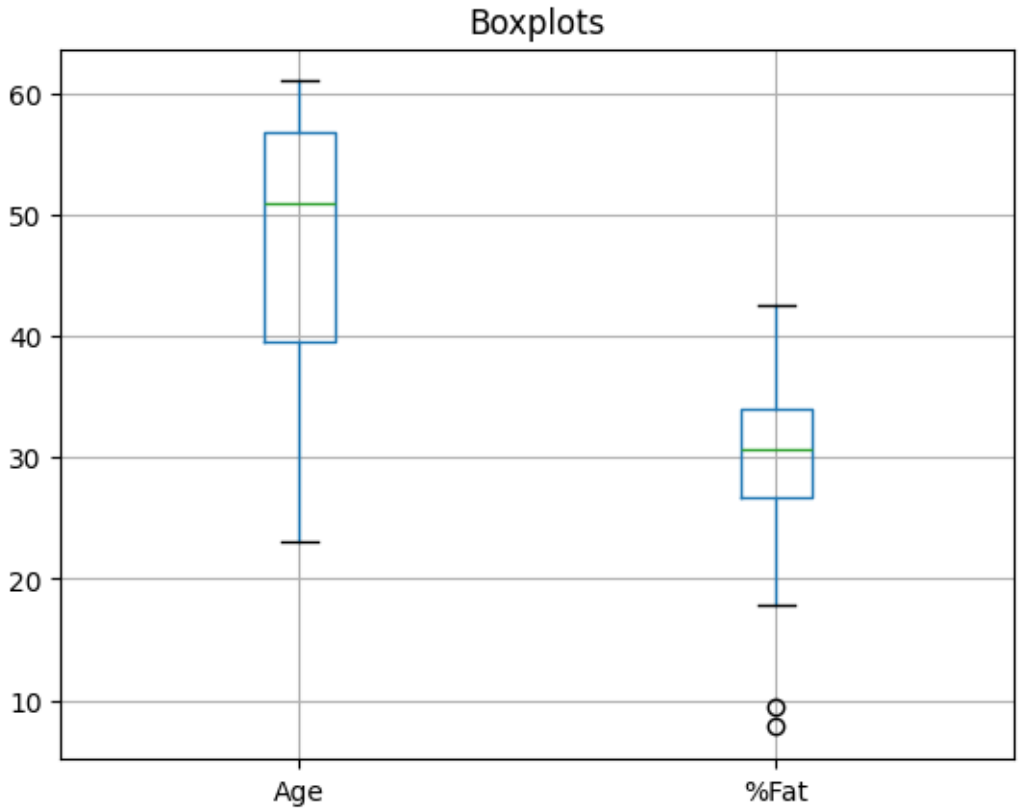
Program:-

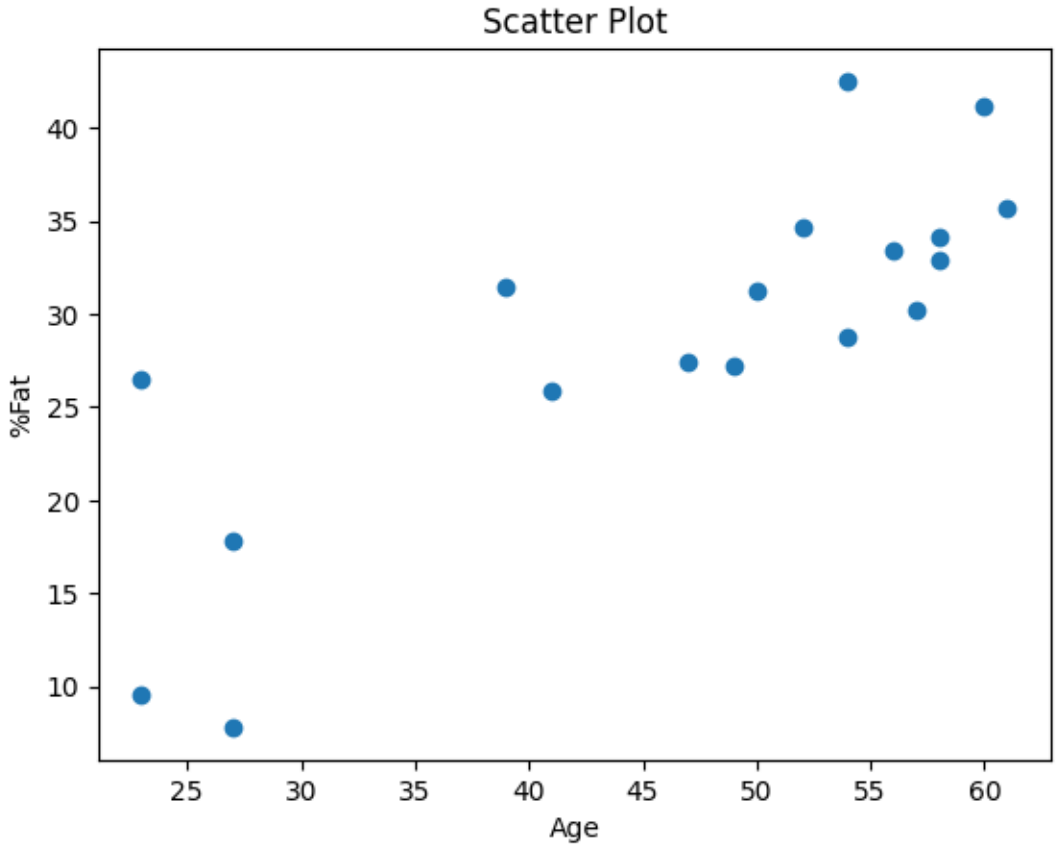
```
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as stats
age = [23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61]
fat = [9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34.6,42.5,28.8,33.4,30.2,34.1,32.9,41.2,35.7]
df = pd.DataFrame({"Age": age, "%Fat": fat})
print("Mean:\n", df.mean())
print("Median:\n", df.median())
print("Standard Deviation:\n", df.std())
df.boxplot(column=["Age", "%Fat"])
plt.title("Boxplots")
```

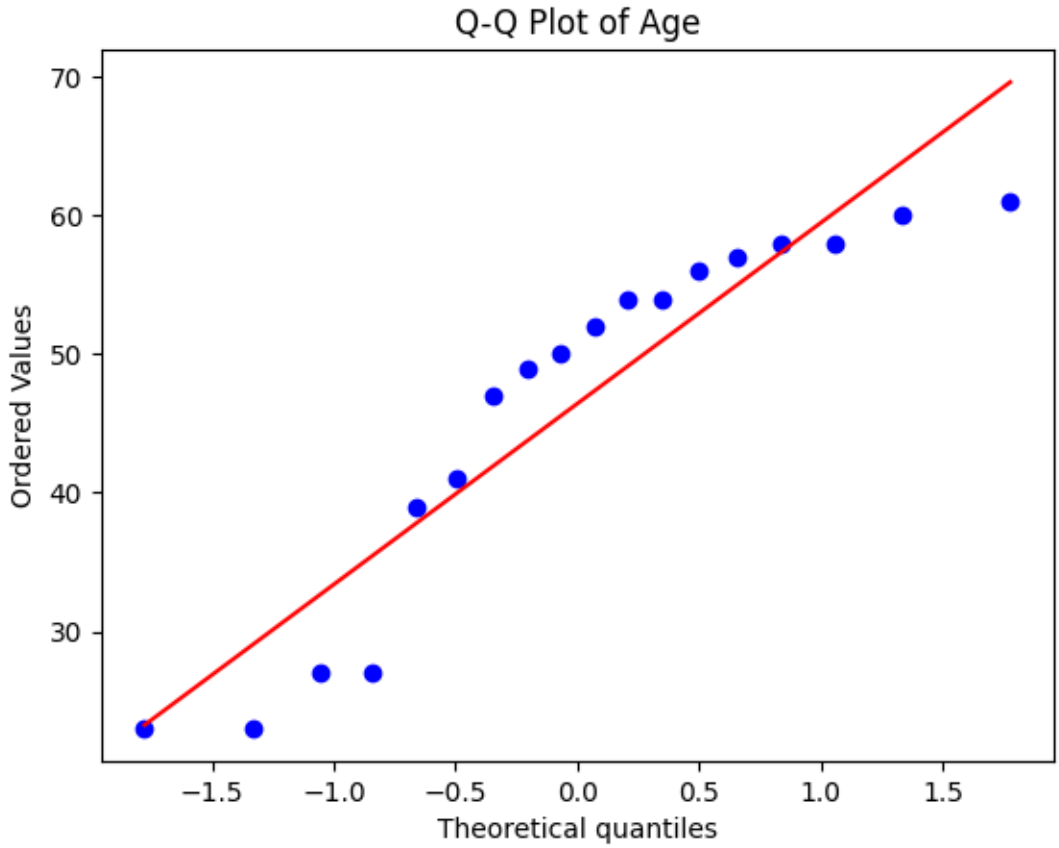
```
plt.show()
plt.scatter(df["Age"], df["%Fat"])
plt.xlabel("Age")
plt.ylabel("%Fat")
plt.title("Scatter Plot")
plt.show()
stats.probplot(df["Age"], dist="norm", plot=plt)
plt.title("Q-Q Plot of Age")
plt.show()
```

Output:-

```
Mean:
 Age     46.444444
%Fat    28.783333
dtype: float64
Median:
 Age     51.0
%Fat    30.7
dtype: float64
Standard Deviation:
 Age     13.218624
%Fat     9.254395
dtype: float64
```


Boxplots
60
50
40
30
20
10
Age
%Fat


Scatter Plot
%Fat
40
35
30
25
20
15
10
25
30
35
40
45
50
55
60
Age



19. Confidence Interval for Drug Trial

Program:-

```
import numpy as np

import scipy.stats as stats

drug = np.array([12,15,14,13,16,18,10,11,17,14])

placebo = np.array([5,7,6,4,5,6,7,8,5,6])

drug_mean = np.mean(drug)

placebo_mean = np.mean(placebo)

drug_ci = stats.t.interval(0.95, len(drug)-1, loc=drug_mean, scale=stats.sem(drug))

placebo_ci = stats.t.interval(0.95, len(placebo)-1, loc=placebo_mean,
scale=stats.sem(placebo))

print("95% CI for Drug Group:", drug_ci)

print("95% CI for Placebo Group:", placebo_ci)
```

Output:-

95% CI for Drug Group: (np.float64(12.152956411008462), np.float64(15.847043588991538))

95% CI for Placebo Group: (np.float64(5.043561120557066), np.float64(6.756438879442935))

20.A/B Test – Difference in Conversion Rates

Program:-

```
import numpy as np
from scipy.stats import ttest_ind
designA = np.array([0.12,0.15,0.11,0.14,0.16,0.13])
designB = np.array([0.18,0.20,0.17,0.19,0.21,0.20])
t_stat, p_value = ttest_ind(designA, designB)
print("T-statistic:", t_stat)
print("P-value:", p_value)
if p_value < 0.05:
    print("Result: Significant difference between designs A and B")
else:
    print("Result: No significant difference")
```

Output:-

T-statistic: -5.830951894845298

P-value: 0.0001658354765713581 7

Result: Significant difference between designs A and B