4b—5b—8b—9b

Write a program of Naive Bayesian classification using Python/R programming language.

```
from sklearn.naive_bayes import GaussianNB

X = [    [180, 80],      [170, 70],      [160, 60],    [158, 54],     [166, 65]   ]

y = [1, 1, 0, 0, 1]

model = GaussianNB()

model.fit(X, y)

test = [[165, 62]]

predicted = model.predict(test)

print("Test Data:", test)

print("Predicted Class (0=Female, 1=Male):", predicted[0])
```

OUTPUT

```
Test Data: [[165, 62]]

Predicted Class (0=Female, 1=Male): 1
```

3b – 6b – 7b

Write a program of cluster analysis using simple k-means algorithm in any programming language.

```
from sklearn.cluster import KMeans

X = [[1,2],[1,4],[1,0],[10,2],[10,4],[10,0]]  # sample data

kmeans = KMeans(n_clusters=2, n_init=10)

kmeans.fit(X)

print("Cluster centers:", kmeans.cluster_centers_)

print("Labels:", kmeans.labels_)
```

OUTPUT

```
Cluster centers: [[10.  2.] [ 1.  2.]]                         Labels: [1 1 1 0 0 0]
```

10b – 12b

Write a program to calculate chi-square value using Python/R. Report your observation

```
from scipy.stats import chi2_contingency

data = [[10, 20, 30]  [6,  9,  17]]

chi2, p, dof, expected = chi2_contingency(data)

print("Chi-square value:", chi2)

print("p-value:", p)

print("Degrees of freedom:", dof)

print("Expected frequencies:\n", expected)
```

OUTPUT

```
Chi-square value: 0.2727272727272727

p-value: 0.872509960787963

Degrees of freedom: 2

Expected frequencies:

 [[10.28571429 18.0 30.71428571]

 [ 5.71428571 11.0 16.28571429]]
```

Observation:
Since the **p-value (0.87) > 0.05**, we **fail to reject the null hypothesis** — meaning there is **no significant association** between the two categorical variables.

11b – 13b

Write a Python program to generate frequent item sets / association rules using Apriori Algorithm

```
from mlxtend.frequent_patterns import apriori, association_rules

from mlxtend.preprocessing import TransactionEncoder

import pandas as pd
```

```
data=[['milk','bread'],['milk','butter'],['bread','butter']]

df=pd.DataFrame(TransactionEncoder().fit(data).transform(data),columns=['bread','butter','milk']
)

f=apriori(df,min_support=0.5,use_colnames=True)

print(f)
```

OUTPUT

```
   support     itemsets
0      0.67     (bread)
1      0.67    (butter)
2      0.67      (milk)
3      0.67 (bread, butter)
```

14b – 15b

Write a java program to prepare a simulated data set with unique instances.

```
import java.util.*;

public class SimulatedData {

    public static void main(String[] args) {

        Set<String> data = new HashSet<>();

        data.add("ID1, 25, M, Engineer");

        data.add("ID2, 30, F, Doctor");

        data.add("ID3, 22, M, Student");

        data.add("ID4, 28, F, Teacher");

        System.out.println("Simulated Dataset:");

       for (String record : data)

            System.out.println(record);     }          }
```

OUTPUT

Simulated Dataset:

ID1, 25, M, Engineer

ID2, 30, F, Doctor

ID3, 22, M, Student

ID4, 28, F, Teacher

1b – 2b

Visualize the datasets using matplotlib in python/R.(Histogram, Box plot, Bar chart, Pie chart etc.,)

```
import matplotlib.pyplot as plt
# Sample dataset
ages = [22, 25, 30, 35, 40, 45, 50, 55, 60]
genders = ['M', 'F', 'M', 'M', 'F', 'F', 'M', 'F', 'M']
# Histogram
plt.hist(ages, bins=5, color='skyblue')
plt.title("Age Distribution")
plt.xlabel("Age"); plt.ylabel("Frequency")
plt.show()
# Box plot
plt.boxplot(ages)
plt.title("Box Plot of Ages")
plt.show()
# Bar chart
plt.bar(['M', 'F'], [genders.count('M'), genders.count('F')], color=['blue','pink'])
plt.title("Gender Count")
plt.show()
# Pie chart
plt.pie([genders.count('M'), genders.count('F')], labels=['Male','Female'], autopct='%1.1f%%')
plt.title("Gender Distribution")
plt.show()
```

OUTPUT

1 Histogram — *Age Distribution*

- X-axis → Age ranges (20–30, 30–40, 40–50, etc.)
- Y-axis → Number of people in each range
  Visual result:

A light-blue histogram with 5 bars showing that ages are spread fairly evenly from 20 to 60.

---

2 Box Plot — *Box Plot of Ages*

- A single vertical box with:
    - **Median** ≈ 40
    - **Lower whisker** ≈ 22
    - **Upper whisker** ≈ 60

      Visual result:
      A simple box showing data spread — no outliers.

---

3 Bar Chart — *Gender Count*

- Two bars:
    - **Male (M):** height = 5
    - **Female (F):** height = 4

      Visual result:
      A blue bar for males slightly taller than the pink bar for females.

---

4 Pie Chart — *Gender Distribution*

- Circle divided into two sections:
    - **Male:** 55.6 %
    - **Female:** 44.4 %

      Visual result:
      A neat pie chart labeled “Male” and “Female” with percentage values shown.

Each of these charts appears **in a separate window** sequentially when you run the code.