DEPARTMENT OF CSE, TIEIT (TRUBA), BHOPAL

LAB MANUAL of CS – 605 [Data Analytics]

PRACTICAL 1

Objective: Study of Data Analytics.

Introduction: Data Analysis is the process of Inspecting, Cleansing, Transforming and Modeling Data with the goal of discovering useful Information, informing Conclusion and support for Decision Making. In today‟s world, Data Analytics plays an important role for improving the growth of Business as it enable Business organizations to take Decisions that lead them to better growth in near future. It incorporates the approaches like Artificial Intelligence, Machine Learning, Soft Computing, Deep Learning, Association Rules, Clustering, Classification, etc. for carrying out its operations.

Data Analytics incorporates the domains like Information Technology, Business and Statistics with a goal to enable Organizations and Businesses grow rapidly. The primary goal of Data Analyst is to increase Efficiency and improve Performance by discovering useful patterns in Data. Statistical Analysis is the heart of Data Analytics. Both Statistics and Machine Learning are used to analyze Big Data. Big Data is used to create statistical models that reveal trends in Data. These models can then be applied to New Data to make Predictions and inform Decision Making.

Data Analytics refers to the set of Qualitative and Quantitative set of approaches for deriving valuable insights from Data.

Types of Data Analytics: There are 4 types of Data Analytics, as:

*** Descriptive Analytics:** It involves the analysis and description about the Features of Data. It deals with the summarization of Information. When coupled with Visual Analysis, it provides a comprehensive structure of Data.

Here, we deal with the past Data to draw Conclusions and present our Data in the form of Dashboards. In Businesses, it is used for determining the Key Performance Indicators (KPI) to evaluate the Performance of a Business.

*** Predictive Analysis:** It determines the Future outcomes on the basis of the Historical Data. It makes use of Descriptive Analysis to generate Predictions about the Future using Machine Learning and other advance approaches.

It is a complex Domain that requires a large amount of Data, skilled implementation of Predictive Models and its tuning to obtain accurate Predictions. It requires skilled Workforce that can develop effective Machine Learning Models.

Prepared by Prof. Puneet Gurbani

*** Diagnostic Analysis:** Sometimes, Businesses are required to think critically about the nature of Data and understand the Descriptive Analysis in depth. In order to find issues in Data, we need to find anomalous patterns that might contribute towards the poor performance of the Model.

With Diagnostic Analysis, we are able to diagnose various problems that are exhibited through our Data. Businesses use this technique to reduce their Losses and optimize their Performance.

*** Prescriptive Analysis:** It combines the insights from all of the above specified types of Analytical techniques. It allows Businesses to take Decisions based on them. It makes heavy use of Artificial Intelligence in order to felicitate Businesses in making careful Business Decisions.

Process of Data Analytics: The process of carrying out the Analysis of Data is as follows:

*** Business Understanding:** In case of a New Requirement, first of all, we need to determine the Business Objective, assess the situation, determine Goals and then produce the Project Plan as per the Requirements.

Business Objectives are defined in this Phase.

*** Data Exploration:** The next task is to gather initial Data, describe and explore that Data and lastly verify Data Quality to ensure that it contains the Data we require. Data collected from the various sources is described in terms of its application and the need for the Project in this Phase.

This is necessary to verify the Quality of Data collected.

*** Data Preparation:** From the Data collected in the previous Phase, we need to select Data as per the need, Clean it, Construct it to get useful Information and then Integrate it all. Finally, we need to format the Data to get the appropriate Insights.

Data is selected, Cleaned and Integrated into the Format finalized for the Analysis in this Phase.

*** Data Modeling:** After gathering the Data, we perform Data Modeling on it. For this, we need to select the Modeling Technique, generate Test Design, build a Model and assess the built Model. The Data Model is built to analyze Relationships between various selected objects in the Data.

Test Cases are built for assessing the Model and Model is tested and implemented on the Data in this Phase.

*** Data Evaluation:** Here, we evaluate the Results from the previous Phase, review the scope of Error and determine the next tasks to perform.

We evaluate the Results of the Test Cases and Review the scope of Errors in this Phase.

*** Deployment:** We need to plan the Deployment, Monitoring and Maintenance and produce a Final Report and Review the Process being carried out.

In this Phase, we deploy the Results of the Analysis. This is also known as Reviewing the Project.

Data Analytics Framework: It has 6 basic steps of operation, as:

S. No.	Task	Operations
1	**Collection**	* Streaming Data (Event Data, Time Series Data, etc.) * Historical Data
2	**Cleaning**	* Identify / Remove Quality Issues * Label / Structure * Add Context
3	**Integration**	* Align Data (Existing Data Sets and Common Vocabulary)
4	**Analysis**	* Descriptive * Predictive and Perspective (Machine Learning, Natural Language Processing, Image Processing, Computer Vision, etc.)
5	**Visualization**	* Histogram and Bar Charts * Scatter Plots * Heat Maps * Network Analysis
6	**Alerting**	* Custom / Dashboard Alerts (Spike in Traffic, Goal Completion / Miss, etc.) * E Mail Notification

Tools for Data Analytics: The various Tools that can be used for Data Analytics include APACHE SPARK, PYTHON, HADOOP, SQL, R, TABLEAU, SPLUNK, etc.

Result: The study of Data Analytics has been done successfully.

PRACTICAL 2

Objective: Write a Program in PYTHON for Data Visualization using Bar Chart and Pie Chart.

Program:

(1) BAR CHART

```
import matplotlib.pyplot as plt

n = int(input("Specify the Number of Elements : "))
lstlang = []
lstnoprog = []

for i in range(n):
    templang = input("Name of Language : ")
    tempnoprog = int(input("Total Number of Programmers : "))

    lstlang.append(templang)
    lstnoprog.append(tempnoprog)

print("Languages : ")
print(lstlang)
print("Total Programmers in each Language : ")
print(lstnoprog)

plt.bar(lstlang, lstnoprog)
plt.show()
```

(2) PIE CHART

```
import matplotlib.pyplot as plt
n = int(input("Specify the Number of Elements : "))

lstlang = []
lstnoprog = []

for i in range(n):
    templang = input("Name of Language : ")
    tempnoprog = int(input("Total Number of Programmers : "))
```

```
    lstlang.append(templang)
    lstnoprog.append(tempnoprog)

print("Languages : ")
print(lstlang)
print("Total Programmers in each Language : ")
print(lstnoprog)

plt.pie(lstnoprog, labels = lstlang, autopct = '%1.2f%%')
plt.show()
```

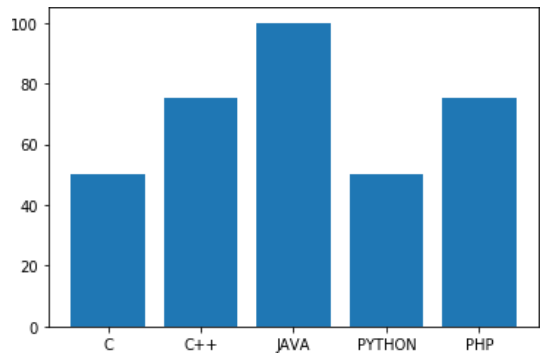
Output:

(1) BAR CHART

```
Specify the Number of Elements : 5

Name of Language : C
Total Number of Programmers : 50
Name of Language : C++
Total Number of Programmers : 75
Name of Language : JAVA
Total Number of Programmers : 100
Name of Language : PYTHON
Total Number of Programmers : 50
Name of Language : PHP
Total Number of Programmers : 75

Languages :
['C', 'C++', 'JAVA', 'PYTHON', 'PHP']
Total Programmers in each Language :
[50, 75, 100, 50, 75]
```

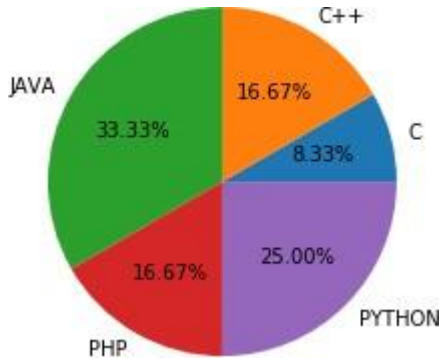


(2) PIE CHART

Specify the Number of Elements : 5

Name of Language : C
Total Number of Programmers : 25
Name of Language : C++
Total Number of Programmers : 50
Name of Language : JAVA
Total Number of Programmers : 100
Name of Language : PHP
Total Number of Programmers : 50
Name of Language : PYTHON
Total Number of Programmers : 75

Languages :
['C', 'C++', 'JAVA', 'PHP', 'PYTHON']
Total Programmers in each Language :
[25, 50, 100, 50, 75]



Result: The program in PYTHON for Data Visualization using Bar Chart and Pie Chart has been implemented successfully.

PRACTICAL 3

Objective: Write a Program in PYTHON for Data Visualization using Box Plot.

Program:

```
import matplotlib.pyplot as plt

lstfinal = []
n = int(input("Specify the Number of Elements : "))

for i in range(n):
    print("Specify the Values for Element " + str(i + 1) + " : ")
    m = int(input())

    lsttemp = []

    for j in range(m):
        tempval = int(input("Enter Item : "))
        lsttemp.append(tempval)

    lstfinal.append(lsttemp)

print(lstfinal)

plt.boxplot(lstfinal)
plt.show()
```

Output:

```
Specify the Number of Elements : 3

Specify the Values for Element 1 : 5

Enter Item : 2
Enter Item : 8
Enter Item : 3
Enter Item : 7
Enter Item : 5
```

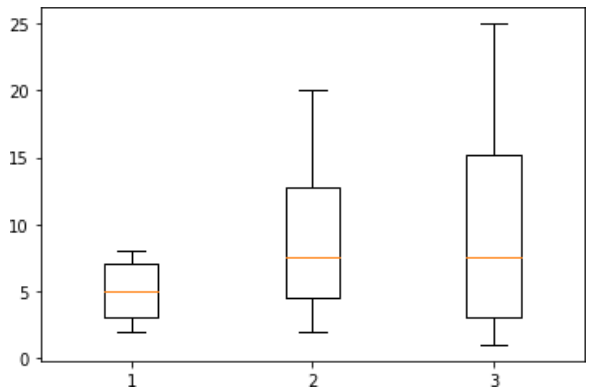
Specify the Values for Element 2 : 8

Enter Item : 12
Enter Item : 20
Enter Item : 2
Enter Item : 8
Enter Item : 7
Enter Item : 5
Enter Item : 3
Enter Item : 15

Specify the Values for Element 3 : 12

Enter Item : 1
Enter Item : 3
Enter Item : 5
Enter Item : 8
Enter Item : 7
Enter Item : 3
Enter Item : 2
Enter Item : 20
Enter Item : 15
Enter Item : 16
Enter Item : 12
Enter Item : 25

[[2, 8, 3, 7, 5], [12, 20, 2, 8, 7, 5, 3, 15], [1, 3, 5, 8, 7, 3, 2, 20, 15, 16, 12, 25]]



Result: The program in PYTHON for Data Visualization using Box Plot has been implemented successfully.

PRACTICAL 4

Objective: Write a Program in PYTHON to import Data Set (EXCEL / CSV) and display its Content.

Program:

```
import pandas as pd

data = pd.read_excel("DataSet_PR4.xlsx")
print("##### Printing the Data Set : #####")
print(data)

print("\n \n \n ##### Printing the Top 5 Rows of Data Set : #####")
print(data.head())
print("\n \n \n ##### Columns as List : #####")
print(data.columns.ravel())

spcol = pd.read_excel("DataSet_PR4.xlsx", usecols=['Enrol_No', 'College'])
print("\n \n \n ##### Reading Data from Specific Columns of EXCEL File and Printing it : #####")
print(spcol)

brdataall = (data['Branch'].tolist())
print("\n \n \n ##### Data of Branch Column : #####")
print(brdataall)
```

DEPARTMENT OF CSE, TIEIT (TRUBA), BHOPAL

LAB MANUAL of CS – 605 [Data Analytics]

EXCEL DATA SET:

S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
1	CS_1	CSETRUBA1	CSE	7	TIEIT	1
2	CS_2	CSETRUBA2	CSE	7	TIEIT	1
3	CS_3	CSETRUBA3	CSE	7	TIEIT	1
4	CS_4	CSETRUBA4	CSE	8	TIEIT	0
5	CS_5	CSETRUBA5	CSE	8	TCST	1
6	CS_6	CSETRUBA6	CSE	7	TIEIT	1
7	CS_7	CSETRUBA7	CSE	7	TIEIT	1
8	CS_8	CSETRUBA8	CSE	8	TIEIT	0
9	CS_9	CSETRUBA9	CSE	8	TIEIT	0
10	CS_10	CSETRUBA10	CSE	8	TCST	1
11	CS_11	CSETRUBA11	CSE	8	TIEIT	0
12	CS_12	CSETRUBA12	CSE	7	TIEIT	1
13	EE_1	EETRUBA1	EE	7	TIEIT	1
14	EE_2	EETRUBA2	EE	7	TIEIT	1
15	EE_3	EETRUBA3	EE	8	TIEIT	0
16	EE_4	EETRUBA4	EE	8	TIEIT	0
17	EE_5	EETRUBA5	EE	8	TIEIT	0
18	EE_6	EETRUBA6	EE	7	TIEIT	1
19	EE_7	EETRUBA7	EE	7	TIEIT	1
20	EE_8	EETRUBA8	EE	8	TIEIT	0
21	ME_1	METRUBA1	ME	8	TIEIT	0
22	ME_2	METRUBA2	ME	7	TIEIT	1
23	ME_3	METRUBA3	ME	7	TIEIT	1
24	ME_4	METRUBA4	ME	8	TIEIT	0
25	ME_5	METRUBA5	ME	8	TCST	1

Output:

Printing the Data Set :

	S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
0	1	CS_1	CSETRUBA1	CSE	7	TIEIT	1
1	2	CS_2	CSETRUBA2	CSE	7	TIEIT	1
2	3	CS_3	CSETRUBA3	CSE	7	TIEIT	1
3	4	CS_4	CSETRUBA4	CSE	8	TIEIT	0
4	5	CS_5	CSETRUBA5	CSE	8	TCST	1
5	6	CS_6	CSETRUBA6	CSE	7	TIEIT	1
6	7	CS_7	CSETRUBA7	CSE	7	TIEIT	1
7	8	CS_8	CSETRUBA8	CSE	8	TIEIT	0
8	9	CS_9	CSETRUBA9	CSE	8	TIEIT	0
9	10	CS_10	CSETRUBA10	CSE	8	TCST	1
10	11	CS_11	CSETRUBA11	CSE	8	TIEIT	0
11	12	CS_12	CSETRUBA12	CSE	7	TIEIT	1
12	13	EE_1	EETRUBA1	EE	7	TIEIT	1
13	14	EE_2	EETRUBA2	EE	7	TIEIT	1
14	15	EE_3	EETRUBA3	EE	8	TIEIT	0
15	16	EE_4	EETRUBA4	EE	8	TIEIT	0
16	17	EE_5	EETRUBA5	EE	8	TIEIT	0
17	18	EE_6	EETRUBA6	EE	7	TIEIT	1
18	19	EE_7	EETRUBA7	EE	7	TIEIT	1
19	20	EE_8	EETRUBA8	EE	8	TIEIT	0
20	21	ME_1	METRUBA1	ME	8	TIEIT	0
21	22	ME_2	METRUBA2	ME	7	TIEIT	1
22	23	ME_3	METRUBA3	ME	7	TIEIT	1
23	24	ME_4	METRUBA4	ME	8	TIEIT	0
24	25	ME_5	METRUBA5	ME	8	TCST	1

Printing the Top 5 Rows of Data Set :

	S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
0	1	CS_1	CSETRUBA1	CSE	7	TIEIT	1
1	2	CS_2	CSETRUBA2	CSE	7	TIEIT	1
2	3	CS_3	CSETRUBA3	CSE	7	TIEIT	1
3	4	CS_4	CSETRUBA4	CSE	8	TIEIT	0
4	5	CS_5	CSETRUBA5	CSE	8	TCST	1

Columns as List :

```
['S_No' 'Enrol_No' 'Name_of_Student' 'Branch' 'Semester' 'College'
 'Bool_Val']
```

Reading Data from Specific Columns of EXCEL File and Printing it :

	Enrol_No	College
0	CS_1	TIEIT
1	CS_2	TIEIT
2	CS_3	TIEIT
3	CS_4	TIEIT
4	CS_5	TCST
5	CS_6	TIEIT
6	CS_7	TIEIT
7	CS_8	TIEIT
8	CS_9	TIEIT
9	CS_10	TCST
10	CS_11	TIEIT
11	CS_12	TIEIT
12	EE_1	TIEIT
13	EE_2	TIEIT
14	EE_3	TIEIT
15	EE_4	TIEIT
16	EE_5	TIEIT
17	EE_6	TIEIT
18	EE_7	TIEIT
19	EE_8	TIEIT
20	ME_1	TIEIT
21	ME_2	TIEIT
22	ME_3	TIEIT
23	ME_4	TIEIT
24	ME_5	TCST

Data of Branch Column :

['CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'EE', 'EE', 'EE', 'EE', 'EE', 'EE', 'EE', 'EE', 'ME', 'ME', 'ME', 'ME', 'ME']

Result: The program in PYTHON to READ Data from an EXCEL File and display it has been implemented successfully.

PRACTICAL 5

Objective: Write a Program in PYTHON to identify the Missing Values and Data Types of Attributes / Columns from a Data Set.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_excel("DataSet_PR5.xlsx")
print("##### Total Missing Values in Data Set : #####")
print(data.isnull().sum())

print("\n \n \n \n ##### Data Type of Attributes (Columns) in Data Set : #####")
dtyps = data.dtypes
print(dtyps)

cols = data.columns.ravel()
print(cols)

lstcolnull = []
nullcols = []

for i in cols:
    if data[i].isnull().any():
        temp = data[i].isnull().sum()
        lstcolnull.append(temp)
        nullcols.append(i)

print("\n \n \n ##### Columns with NULL Values : #####")
print(cols)
print(lstcolnull)

print("\n \n \n VISUALIZATION of NULL Values using BAR CHART")
plt.pie(lstcolnull, labels = nullcols, autopct = '%1.2f%%')
plt.show()

print("\n \n \n VISUALIZATION of NULL Values using PIE CHART")
plt.bar(nullcols, lstcolnull)
plt.show()
```

EXCEL DATA:

S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
1	Enrol1	StudentName1	CSE	7	TIEIT	1
2	Enrol2	StudentName2	CSE	7		1
3	Enrol3	StudentName3		7	TIEIT	1
4	Enrol4	StudentName4		8	TIEIT	0
5	Enrol5	StudentName5	CSE	8	TCST	1
6	Enrol6	StudentName6		7		1
7	Enrol7	StudentName7		7	TIEIT	1
8	Enrol8	StudentName8	CSE	8	TIEIT	0
9	Enrol9	StudentName9	CSE	8	TIEIT	0
10	Enrol10	StudentName10		8		1
11	Enrol11	StudentName11	CSE	8	TIEIT	0
12	Enrol12	StudentName12	CSE	7	TIEIT	1
13	Enrol13	StudentName13	EE	7		1
14	Enrol14	StudentName14	EE	7	TIEIT	1
15	Enrol15	StudentName15		8	TIEIT	0
16	Enrol16	StudentName16	EE	8	TIEIT	0
17	Enrol17	StudentName17		8	TIEIT	0
18	Enrol18	StudentName18	EE	7		1
19	Enrol19	StudentName19		7	TIEIT	1
20	Enrol20	StudentName20	EE	8	TIEIT	0
21	Enrol21	StudentName21	ME	8		0
22	Enrol22	StudentName22		7	TIEIT	1
23	Enrol23	StudentName23	ME	7	TIEIT	1
24	Enrol24	StudentName24		8	TIEIT	0
25	Enrol25	StudentName25	ME	8		1

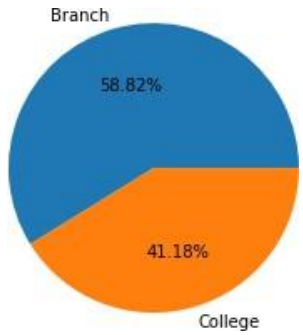
Output:

```
##### Total Missing Values in Data Set : #####
S_No            0
Enrol_No        0
Name_of_Student 0
Branch          10
Semester        0
College         7
Bool_Val        0
dtype: int64
```

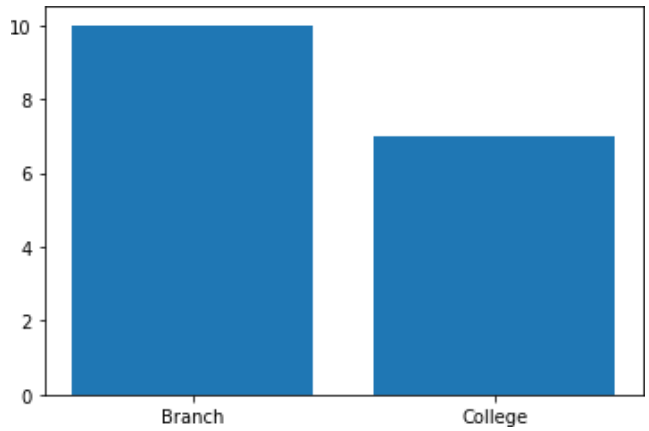
```
##### Data Type of Attributes (Columns) in Data Set : #####
S_No              int64
Enrol_No          object
Name_of_Student   object
Branch            object
Semester          int64
College           object
Bool_Val          int64
dtype: object
['S_No' 'Enrol_No' 'Name_of_Student' 'Branch' 'Semester' 'College'  'Bool_Val']

##### Columns with NULL Values : #####
['S_No' 'Enrol_No' 'Name_of_Student' 'Branch' 'Semester' 'College' 'Bool_Val']
[10, 7]
```

VISUALIZATION of NULL Values using PIE CHART



VISUALIZATION of NULL Values using PIE CHART



Result: The program in PYTHON to identify Missing Values and Data Types of Columns (Attributes) from a Data Set has been implemented successfully.

PRACTICAL 6

Objective: Write a Program in PYTHON to retrieve Data from Data Set based on specified criteria.

Program:

```
import pandas as pd

data = pd.read_excel("DataSet_PR4.xlsx")
print(data.head())

booltruedata = data[data['Bool_Val'] == 1]
print("##### Data with TRUE Value in BOOL_VAL Column #####")
print(booltruedata)

cntr = len(booltruedata.axes[0])
print("\n \n \n Rows with TRUE Value in BOOL_VAL Column : ", cntr)

br = input("Specify Branch to select Data (CSE / EE / ME) : ")
clg = input("Specify Branch to select Data (TIEIT / TCST) : ")

fltr1 = data['Branch'] == br
fltr2 = data['College'] == clg

newdata = data.where(fltr1 & fltr2)
tempdata = newdata.dropna()

print("\n \n \n ##### NEW FILTERED DATA #####")
print(tempdata)
```

EXCEL DATA:

S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
1	CS_1	CSETRUBA1	CSE	7	TIEIT	1
2	CS_2	CSETRUBA2	CSE	7	TIEIT	1
3	CS_3	CSETRUBA3	CSE	7	TIEIT	1
4	CS_4	CSETRUBA4	CSE	8	TIEIT	0
5	CS_5	CSETRUBA5	CSE	8	TCST	1
6	CS_6	CSETRUBA6	CSE	7	TIEIT	1
7	CS_7	CSETRUBA7	CSE	7	TIEIT	1
8	CS_8	CSETRUBA8	CSE	8	TIEIT	0
9	CS_9	CSETRUBA9	CSE	8	TIEIT	0
10	CS_10	CSETRUBA10	CSE	8	TCST	1
11	CS_11	CSETRUBA11	CSE	8	TIEIT	0
12	CS_12	CSETRUBA12	CSE	7	TIEIT	1
13	EE_1	EETRUBA1	EE	7	TIEIT	1
14	EE_2	EETRUBA2	EE	7	TIEIT	1
15	EE_3	EETRUBA3	EE	8	TIEIT	0
16	EE_4	EETRUBA4	EE	8	TIEIT	0
17	EE_5	EETRUBA5	EE	8	TIEIT	0
18	EE_6	EETRUBA6	EE	7	TIEIT	1
19	EE_7	EETRUBA7	EE	7	TIEIT	1
20	EE_8	EETRUBA8	EE	8	TIEIT	0
21	ME_1	METRUBA1	ME	8	TIEIT	0
22	ME_2	METRUBA2	ME	7	TIEIT	1
23	ME_3	METRUBA3	ME	7	TIEIT	1
24	ME_4	METRUBA4	ME	8	TIEIT	0
25	ME_5	METRUBA5	ME	8	TCST	1

Output:

```
##### Reading Data from EXCEL #####
   S_No Enrol_No Name_of_Student Branch Semester College  Bool_Val
0     1    CS_1      CSETRUBA1    CSE        7   TIEIT         1
1     2    CS_2      CSETRUBA2    CSE        7   TIEIT         1
2     3    CS_3      CSETRUBA3    CSE        7   TIEIT         1
3     4    CS_4      CSETRUBA4    CSE        8   TIEIT         0
4     5    CS_5      CSETRUBA5    CSE        8    TCST         1
```

Data with TRUE Value in BOOL_VAL Column

	S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
0	1	CS_1	CSETRUBA1	CSE	7	TIEIT	1
1	2	CS_2	CSETRUBA2	CSE	7	TIEIT	1
2	3	CS_3	CSETRUBA3	CSE	7	TIEIT	1
4	5	CS_5	CSETRUBA5	CSE	8	TCST	1
5	6	CS_6	CSETRUBA6	CSE	7	TIEIT	1
6	7	CS_7	CSETRUBA7	CSE	7	TIEIT	1
9	10	CS_10	CSETRUBA10	CSE	8	TCST	1
11	12	CS_12	CSETRUBA12	CSE	7	TIEIT	1
12	13	EE_1	EETRUBA1	EE	7	TIEIT	1
13	14	EE_2	EETRUBA2	EE	7	TIEIT	1
17	18	EE_6	EETRUBA6	EE	7	TIEIT	1
18	19	EE_7	EETRUBA7	EE	7	TIEIT	1
21	22	ME_2	METRUBA2	ME	7	TIEIT	1
22	23	ME_3	METRUBA3	ME	7	TIEIT	1
24	25	ME_5	METRUBA5	ME	8	TCST	1

Rows with TRUE Value in BOOL_VAL Column : 15

Specify Branch to select Data (CSE / EE / ME) : CSE

Specify Branch to select Data (TIEIT / TCST) : TIEIT

NEW FILTERED DATA

	S_No	Enrol_No	Name_of_Student	Branch	Semester	College	Bool_Val
0	1.0	CS_1	CSETRUBA1	CSE	7.0	TIEIT	1.0
1	2.0	CS_2	CSETRUBA2	CSE	7.0	TIEIT	1.0
2	3.0	CS_3	CSETRUBA3	CSE	7.0	TIEIT	1.0
3	4.0	CS_4	CSETRUBA4	CSE	8.0	TIEIT	0.0
5	6.0	CS_6	CSETRUBA6	CSE	7.0	TIEIT	1.0
6	7.0	CS_7	CSETRUBA7	CSE	7.0	TIEIT	1.0
7	8.0	CS_8	CSETRUBA8	CSE	8.0	TIEIT	0.0
8	9.0	CS_9	CSETRUBA9	CSE	8.0	TIEIT	0.0
10	11.0	CS_11	CSETRUBA11	CSE	8.0	TIEIT	0.0
11	12.0	CS_12	CSETRUBA12	CSE	7.0	TIEIT	1.0

Result: The program in PYTHON to retrieve Data from Data Set based on specific criteria has been implemented successfully.

PRACTICAL 7

Objective: Study of Probabilistic Reasoning and BAYE‟s Theorem.

Introduction: Most Facts in Real World are of Uncertain nature, i. e., we can‟t say exactly that the Fact is TRUE or it is FALSE. For example, the Fact "Sun rises in the West" is always FALSE and the Fact "15th August is the Independence Day of India" is always TRUE. But all Real World situations can‟t have this certain nature. For example, in Weather Forecasting, it has been predicted that there will be heavy Rainfall on tomorrow. Sometimes, it comes out to be TRUE but in some cases, it comes out as FALSE. This is because it is a Real World Scenario and is affected by Uncertainty.

Uncertainty in Scenarios occurs because of the various sources like Information occurred from Unreliable Sources, Equipment Fault, Experimental Errors, Temperature Variation, Climate Change, etc. For these Scenarios, we need Uncertain Reasoning or the Reasoning that can incorporate Uncertainty in Data and then make Decisions. One such approach is Probabilistic Reasoning.

Probabilistic Reasoning uses the concept of Probability to indicate the Uncertainty in Knowledge. It combines Probability Theory with Logic to handle Uncertainty. Probabilistic Reasoning is used when there are Unpredictable Outcomes, when specifications or possibilities of Predicates become too large to handle and when an unknown Error occurs during an Experiment.

Probability: It specifies the change that an uncertain Event may occur. It is a numerical measure of the likelihood of an Event that may occur that ranges between 0 to 1, i. e., if „E" is an Event and „P(E)‟ is the Probability of that Event to occur, then **0 <= P(E) <= 1**.

Probability of Event = Number of Desired Outcomes / Total Number of Outcomes

Let, „P(~E)" be the Probability of not happening an Event, then, **P(E) + P(~E) = 1**.

Conditional Probability: It is the Probability of an Event to occur when another Event has already happened. Suppose, we want to calculate the Probability of „Event A‟ to occur when „Event B" has already occurred, then "Probability of Event A under Conditions of Event B" is defined as:

$$P(A \mid B) = P(A \wedge B) / P(B)$$

Where, P(A ^ B) is the Joint Probability (Intersection) of A and B, P(B) is the Probability (Marginal Probability) of Event B to occur and P(A | B) is the Conditional Probability that specifies the occurrences of „Event A" when „Event B" has already occurred.

BAYE's Theorem: It is also known as BAYE‟s Rule or BAYE‟s Law or Bayesian Reasoning, which determines the Probability of an Event with Uncertain Knowledge. In Probability Theory, it relates the Conditional Probability and Marginal Probability of two Random Events. It was named after the British Mathematician „Thomas Bayes‟. The Bayesian Inference is an application of BAYE‟s Theorem, which is fundamental to Bayesian Statistics.

It computes P(B | A), when P(A | B) is already known. Let, A and B be two independent Events, then Conditional Probability of Event A with Event B already known is:

$$P(A \mid B) = P(A \wedge B) / P(B)$$

It can also be represented, as:

$$P(A \wedge B) = P(A \mid B) * P(B) \quad \text{..... (1)}$$

Similarly, Conditional Probability of Event B with Event A already known is:

$$P(B \mid A) = P(A \wedge B) / P(A)$$

It can also be represented, as:

$$P(A \wedge B) = P(B \mid A) * P(A) \quad \text{..... (2)}$$

Equating (1) and (2), we get,

$$P(A \mid B) = [P(B \mid A) * P(A)] / P(B)$$

The above equation is called BAYE‟s Rule or BAYE‟s Theorem. It shows the Relationship between Joint Probability and Conditional Probability.

Example: Suppose that P1, P2 and P3 are 3 Artists in an Art Competition. Out of all the Paintings exhibited in Competition, 50%, 17% and 33% are designed by P1, P2 and P3 respectively. 4%, 6% and 3% of Paintings designed by P1, P2 and P3 respectively have won the First Prize. Compute the Probabilities for P1, P2 and P3 to win the First Prize.

Solution: Let, P(P1), P(P2) and P(P3) be the Probabilities that the Paintings are designed by Painters P1, P2 and P3 respectively. So,

P(P1) = 50% = 0.5
P(P2) = 17% = 0.17
P(P3) = 33% = 0.33

Let, P(First | P1), P(First | P2) and P(First | P3) be the Probabilities that Paintings of P1, P2 and P3 have won the First Prize. So,

P(First | P1) = 4% = 0.04
P(First | P2) = 6% = 0.06
P(First | P3) = 3% = 0.03

Now, we need to calculate the Probability of each Painter to won First Prize. Let, P(P1 | First), P(P2 | First) and P(P3 | First) be the respective Probabilities that Painter P1, P2 and P3 will win the First Prize.

So, the Probability that Painter P1 will win the First Prize is:

P(P1 | First) = P(P1) * P(First | P1) /
[P(P1) * P(First | P1) + P(P2) * P(First | P2) + P(P3) * P(First | P3)]

=====> P(P1 | First) = 0.5 * 0.04 / [0.5 * 0.04 + 0.17 * 0.06 + 0.33 * 0.03]
=====> P(P1 | First) = 200 / [200 + 102 + 99]
=====> P(P1 | First) = 200 / 401
=====> **P(P1 | First) = 0.4988**

=====> **P(P1 | First) = 49.88 %**

Now, the Probability that Painter P2 will win the First Prize is:

P(P2 | First) = P(P2) * P(First | P2) /
[P(P1) * P(First | P1) + P(P2) * P(First | P2) + P(P3) * P(First | P3)]

=====> P(P2 | First) = 0.17 * 0.06 / [0.5 * 0.04 + 0.17 * 0.06 + 0.33 * 0.03]
=====> P(P2 | First) = 102 / [200 + 102 + 99]
=====> P(P2 | First) = 102 / 401
=====> **P(P2 | First) = 0.2544**

=====> **P(P2 | First) = 25.44 %**

Now, the Probability that Painter P3 will win the First Prize is:

P(P3 | First) = 1 – [P(P1 | First) + P(P2 | First)]

This is because Sum of Probabilities of all Events in a given Scenario can‟t exceed 1 or 100 %.

=====> P(P3 | First) = 1 – [0.4988 + 0.2544]
=====> P(P3 | First) = 1 – 0.7532
=====> **P(P3 | First) = 0.2468**

=====> **P(P3 | First) = 24.68 %**

Result: The study of Probabilistic Reasoning and BAYE‟s Theorem has been done successfully.

PRACTICAL 8

Objective: Study of Correlation and Regression.

Introduction: Correlation and Regression are the two analyses based on Multivariate Distributions. A Multivariate Distribution is described as a Distribution of multiple Variables.

Correlation: It is described as the Analysis which let us know the Association or the absence of the Relationship between two Variables „x" and „y". It is a measure of Linear Association between two Variables. It quantifies the Association between two Continuous Variables (An Independent Variable and a Dependent Variable or between two Independent Variables).

The value of Correlation Coefficient lies between „-1 to +1" with „+l" indicates that two Variables are perfectly related in a Positive Linear Sense, „-1" indicates that two Variables are perfectly related in a Negative Linear Sense and „0" indicates that there is no Linear Relationship between two Variables.

Regression: It predicts the Value of a Dependent Variable based on the Known Value of one or more Independent Variables. It is a related technique to assess the Relationship between an Outcome Variable and one or more Risk Factors or Confounding Variables. The Outcome Variable is also called the Response Variable or Dependent Variable (Y) and the Risk Factors or Confounders are called the Predictors or Explanatory or Independent Variables (X). It is a widely used technique for evaluating multiple Independent Variables.

Simple Linear Regression: When there is a single Continuous Dependent Variable and a single Independent Variable, the analysis is called Simple Linear Regression Analysis. It assumes that there is a Linear Association between two Variables. The Simple Linear Regression Equation is given as:

$$Y = C1 + C2 * X$$

Where, Y is the Predicted or Expected Value of the Outcome, X is the Predictor, C1 and C2 are Constants known as Y Intercept and Estimated Slope respectively. C1 and C2 are estimated from Sample Data in order to minimize the Sum of Squared Differences between the Observed and the Predicted Values of the Outcome.

If Z is the Observed Value, then it reduces

$$SUM((Z - Y)^2)$$

Let, MY and MX be the Sampled Means of Y and X, SX and SY be the Standard Deviations of Independent Variable X and Dependent Variable Y, R is the Sample Correlation Coefficient, then,

$$C1 = R * (SY / SX)$$

And

$$C2 = MY – C1 * MX$$

Multiple Linear Regression: It is an extension of Simple Linear Regression Analysis, used to assess the Association between two or more Independent Variables and a single Continuous Dependent Variable. The Multiple Linear Regression Equation is given, as:

$$Y = C0 + C1 * X1 + C2 * X2 + C3 * X3 + ------+ CP * XP$$

Logistic Regression: It is a popular and widely used analysis, similar to Linear Regression except that the Outcome is Dichotomous (Success / Failure or Yes / No or True / False). Simple Logistic Regression Analysis refers to the Regression application with one Dichotomous Outcome and one Independent Variable. Multiple Logistic Regression Analysis is applicable when there is a single Dichotomous Outcome and more than one Independent Variable.

The Outcome of Logistic Regression Analysis is often coded as 0 or 1, where „1" indicates that the Outcome of interest is Present and „0" indicates that the Outcome of interest is Absent. IF we define „P" as the Probability that the Outcome is „1", then the multiple Logistic Regression Model can be given, as:

$$Z = EXP(C0 + C1 * X1 + C2 * X2 + ------ + CP * XP) / [1 + EXP(C0 + C1 * X1 + C2 * X2 + ------+ CP * XP)]$$

Where, Z is the Expected Probability that the Outcome is Present, X1, X2, ..., XP are Distinct Independent Variables and C0, C1, ..., CP are Regression Coefficients.

In Multiple Regression Model, sometimes the Outcome is the Expected LOG of the Odds that the Outcome is Present.

$$LN (Z / (1 – Z))$$

$$LN[Z / (1 – Z)] = C0 + C1 * X1 + C2 * X2 + C3 * X3 + ----- + CP * XP$$

Where, LN represents "log" in Base "e" or Natural LOG.

Difference between Correlation and Regression:

S. No.	Parameter	Correlation	Regression
1	**Meaning**	It is a Statistical Measure which determines Co – Relationship or Association of two Variables.	It describes how an Independent Variable is numerically related to the Dependent Variable.
2	**Usage**	To represent Numerical Relationship between two Variables	To fit a Best Line and estimate one Variable on the basis of another Variable
3	**Indicates**	Correlation Coefficient indicates the extent to which two Variables move together.	It indicates the impact of a Unit Change in the Known Variable (X) on the Estimated Variable (Y).
4	**Objective**	To find a Numerical Value expressing the Relationship between Variables	To estimate Values of Random Variable on the basis of Values of Fixed Variable

Regression and Correlation are examples of Multivariable Methods. They are computationally Complex and generally requires the use of a Statistical Computing Package, like PYTHON, R, etc.

Result: The study of Correlation and Regression has been done successfully.

PRACTICAL 9

Objective: Write a Program in PYTHON to implement Linear Regression Model.

Program:

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics

dset = pd.read_csv("DataSet_PR9.csv")
print(dset.head())

xind = pd.DataFrame(dset['ind_var'])
ydep = pd.DataFrame(dset['dep_var'])

indtrain, indtest, deptrain, deptest = train_test_split(xind, ydep, test_size = 0.2, random_state = 1)

print("Training Data for Independent Variable : \n", indtrain)
print("Testing Data for Independent Variable : \n", indtest)
print("Training Data for Dependent Variable : \n", deptrain)
print("Testing Data for Dependent Variable : \n", deptest)

print("Shape of Training Data (Independent) : ", indtrain.shape)
print("Shape of Testing Data (Independent) : ", indtest.shape)
print("Shape of Training Data (Dependent) : ", deptrain.shape)
print("Shape of Testing Data (Dependent) : ", deptest.shape)

regmodel = LinearRegression()
regmodel.fit(indtrain, deptrain)

print("Value of Regression Intercept is : ", regmodel.intercept_)
print("Value of Regression Coefficient is : ", regmodel.coef_)

print("Observed Testing Data : \n", deptest)
deppred = regmodel.predict(indtest)
print("Predicted Testing Data : \n", deppred)
```

```
print("Mean Absolute Error : ", metrics.mean_absolute_error(deptest, deppred))
print("Mean Squared Error : ", metrics.mean_squared_error(deptest, deppred))

print("ROOT MEAN SQUARED ERROR : ", np.sqrt(metrics.mean_squared_error(deptest, deppred)))
```

CSV DATA:

ind_var	dep_var
0	1
1	3
2	2
3	5
4	7
5	8
6	8
7	9
8	10
9	12
10	12
11	11
12	16
13	15
14	14
15	20
16	17
17	19
18	16
19	25
20	24
21	23
22	18
23	21
24	16

Output:

```
   ind_var dep_var
0        0        1
1        1        3
2        2        2
3        3        5
4        4        7
```

Training Data for Independent Variable :

```
    ind_var
10       10
18       18
19       19
4         4
2         2
20       20
6         6
7         7
22       22
1         1
16       16
0         0
15       15
24       24
23       23
9         9
8         8
12       12
11       11
5         5
```

Testing Data for Independent Variable :

```
    ind_var
14       14
13       13
17       17
3         3
21       21
```

Training Data for Dependent Variable :

	dep_var
10	12
18	16
19	25
4	7
2	2
20	24
6	8
7	9
22	18
1	3
16	17
0	1
15	20
24	16
23	21
9	12
8	10
12	16
11	11
5	8

Testing Data for Dependent Variable :

	dep_var
14	14
13	15
17	19
3	5
21	23

Shape of Training Data (Independent) : (20, 1)
Shape of Testing Data (Independent) : (5, 1)
Shape of Training Data (Dependent) : (20, 1)
Shape of Testing Data (Dependent) : (5, 1)

Value of Regression Intercept is : [3.18863143]
Value of Regression Coefficient is : [[0.82856626]]

Observed Testing Data :

	dep_var
14	14
13	15
17	19
3	5
21	23

Predicted Testing Data :

```
[[14.78855902]
 [13.95999276]
 [17.27425778]
 [ 5.6743302 ]
 [20.58852281]]
```

Mean Absolute Error : 1.3280231716147717
Mean Squared Error : 2.1903140060015893

ROOT MEAN SQUARED ERROR : 1.4799709476883622

Result: The program in PYTHON for Linear Regression Model is implemented successfully.

PRACTICAL 10

Objective: Write a Program in PYTHON to implement Logistic Regression Model.

Program:

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import classification_report

input_data_set = pd.read_excel("DataSet_PR10.xlsx")
print("Top 5 Rows of Data Set : \n", input_data_set.head())

independent_features = ['Ind1', 'Ind2', 'Ind3', 'Ind4', 'Ind5']
print("Independent Features : \n", independent_features)

dependent_feature_vals = input_data_set.Dep
independent_feature_vals = input_data_set[independent_features]

print("Independent Variables Data Set : \n", independent_feature_vals)
print("Dependent Variable Data Set : \n", dependent_feature_vals)

x_tr, x_tst, y_tr, y_tst = train_test_split(independent_feature_vals, dependent_feature_vals,
test_size = 0.25, random_state = 2)
print("Independent Training Data Set : \n", x_tr)
print("Independent Testing Data Set : \n", x_tst)
print("Dependent Training Data Set : \n", y_tr)
print("Dependent Testing Data Set : \n", y_tst)

logregmodtst = LogisticRegression()
logregmodtst.fit(x_tr, y_tr)

y_pred = logregmodtst.predict(x_tst)
print(y_pred)

conf_mat = metrics.confusion_matrix(y_tst, y_pred)
print(conf_mat)
```

```
print("ACCURACY : ", metrics.accuracy_score(y_tst, y_pred))
print("PRECISION : ", metrics.precision_score(y_tst, y_pred))
print("RECALL : ", metrics.recall_score(y_tst, y_pred))

print("CLASSIFICATION REPORT : \n", classification_report(y_tst, y_pred))
```

EXCEL DATA:

S_No	Ind1	Ind2	Ind3	Ind4	Ind5	Dep
1	1	11	1	7	1	1
2	2	12	1	7	1	0
3	3	13	21	7	2	1
4	4	14	1	8	1	0
5	5	15	1	8	2	1
6	6	16	21	7	1	1
7	7	17	1	7	1	0
8	8	18	1	8	2	0
9	9	19	21	8	1	0
10	10	110	1	8	2	1
11	11	111	1	8	1	0
12	12	112	1	7	1	1
13	13	21	2	7	1	0
14	14	22	2	7	2	1
15	15	23	2	8	1	0
16	16	24	22	8	2	0
17	17	25	2	8	1	1
18	18	26	22	7	1	1
19	19	27	2	7	2	1
20	20	28	2	8	1	0
21	21	31	23	8	1	1
22	22	32	3	7	2	1
23	23	33	3	7	1	1
24	24	34	23	8	1	0
25	25	35	3	8	2	1

Output:

Top 5 Rows of Data Set :

	S_No	Ind1	Ind2	Ind3	Ind4	Ind5	Dep
0	1	1	11	1	7	1	1
1	2	2	12	1	7	1	0
2	3	3	13	21	7	2	1
3	4	4	14	1	8	1	0
4	5	5	15	1	8	2	1

Independent Features :
['Ind1', 'Ind2', 'Ind3', 'Ind4', 'Ind5']

Independent Variables Data Set :

	Ind1	Ind2	Ind3	Ind4	Ind5
0	1	11	1	7	1
1	2	12	1	7	1
2	3	13	21	7	2
3	4	14	1	8	1
4	5	15	1	8	2
5	6	16	21	7	1
6	7	17	1	7	1
7	8	18	1	8	2
8	9	19	21	8	1
9	10	110	1	8	2
10	11	111	1	8	1
11	12	112	1	7	1
12	13	21	2	7	1
13	14	22	2	7	2
14	15	23	2	8	1
15	16	24	22	8	2
16	17	25	2	8	1
17	18	26	22	7	1
18	19	27	2	7	2
19	20	28	2	8	1
20	21	31	23	8	1
21	22	32	3	7	2
22	23	33	3	7	1
23	24	34	23	8	1
24	25	35	3	8	2

Dependent Variable Data Set :

0	1
1	0
2	1
3	0
4	1
5	1
6	0
7	0
8	0
9	1
10	0
11	1
12	0
13	1
14	0
15	0
16	1
17	1
18	1
19	0
20	1
21	1
22	1
23	0
24	1

Name: Dep, dtype: int64

Independent Training Data Set :

Ind1	Ind2	Ind3	Ind4	Ind5
16	17	25	2	8	1
12	13	21	2	7	1
22	23	33	3	7	1
4	5	15	1	8	2
10	11	111	1	8	1
5	6	16	21	7	1
19	20	28	2	8	1
1	2	12	1	7	1
2	3	13	21	7	2
7	8	18	1	8	2
21	22	32	3	7	2
20	21	31	23	8	1
18	19	27	2	7	2
11	12	112	1	7	1
24	25	35	3	8	2
13	14	22	2	7	2
15	16	24	22	8	2
8	9	19	21	8	1

Independent Testing Data Set :

Ind1	Ind2	Ind3	Ind4	Ind5
14	15	23	2	8	1
0	1	11	1	7	1
17	18	26	22	7	1
6	7	17	1	7	1
23	24	34	23	8	1
9	10	110	1	8	2
3	4	14	1	8	1

Dependent Training Data Set :
16 1
12 0
22 1
4 1
10 0
5 1
19 0
1 0
2 1
7 0
21 1
20 1
18 1
11 1
24 1
13 1
15 0
8 0

Name: Dep, dtype: int64

Dependent Testing Data Set :
14 0
0 1
17 1
6 0
23 0
9 1
3 0

Name: Dep, dtype: int64

Predicted Outcomes :
[0 0 1 0 1 1 0]

Confusion Matrix :
[[3 1]
[1 2]]

ACCURACY : 0.7142857142857143
PRECISION : 0.6666666666666666
RECALL : 0.6666666666666666

CLASSIFICATION REPORT :

	precision	recall	f1-score	support
0	0.75	0.75	0.75	4
1	0.67	0.67	0.67	3
micro avg	0.71	0.71	0.71	7
macro avg	0.71	0.71	0.71	7
weighted avg	0.71	0.71	0.71	7

Result: The program in PYTHON for Logistic Regression Model is implemented successfully.

PRACTICAL 11

Objective: Write a Program in PYTHON to implement KNN Classifier.

Program:

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn import metrics

in_dt_set = pd.read_excel("DataSet_Pr11.xlsx")
print("First 5 Rows of Data Set : \n", in_dt_set.head())
print("\n Shape of Data Set : \n", in_dt_set.shape)

ind_features = ['Ind1', 'Ind2', 'Ind3', 'Ind4', 'Ind5']
print("\n List of Independent Features : ", ind_features)

dep_feature_vals = in_dt_set.Dep
ind_feature_vals = in_dt_set[ind_features]

print("\n Independent Data Set : \n", ind_feature_vals)
print("\n Dependent Data Set : \n", dep_feature_vals)

x_tr, x_tst, y_tr, y_tst = train_test_split(ind_feature_vals, dep_feature_vals, test_size = 0.2,
random_state = 2, stratify = dep_feature_vals)

knncltest = KNeighborsClassifier(n_neighbors = 3)
knncltest.fit(x_tr, y_tr)

y_pred = knncltest.predict(x_tst)
print("\n Observed Tested Values of Dependent Data Set : \n", y_tst)
print("\n Predicted Tested Values of Dependent Data Set : \n", y_pred)

print("\n Test Data Set Score : \n", knncltest.score(x_tst, y_tst))

conf_mat = confusion_matrix(y_tst, y_pred)
print("\n Confusion Matrix : \n", conf_mat)
```

```
print("\n Accuracy : \n", metrics.accuracy_score(y_tst, y_pred))
print("\n Precision : \n", metrics.precision_score(y_tst, y_pred))
print("\n Recall : \n", metrics.recall_score(y_tst, y_pred))
```

EXCEL DATA SET:

S_No	Ind1	Ind2	Ind3	Ind4	Ind5	Dep
1	1	11	1	7	1	1
2	2	12	1	7	1	0
3	3	13	21	7	2	1
4	4	14	1	8	1	0
5	5	15	1	8	2	1
6	6	16	21	7	1	1
7	7	17	1	7	1	0
8	8	18	1	8	2	0
9	9	19	21	8	1	0
10	10	110	1	8	2	1
11	11	111	1	8	1	0
12	12	112	1	7	1	1
13	13	21	2	7	1	0
14	14	22	2	7	2	1
15	15	23	2	8	1	0
16	16	24	22	8	2	0
17	17	25	2	8	1	1
18	18	26	22	7	1	1
19	19	27	2	7	2	1
20	20	28	2	8	1	0
21	21	31	23	8	1	1
22	22	32	3	7	2	1
23	23	33	3	7	1	1
24	24	34	23	8	1	0
25	25	35	3	8	2	1

Output:

```
First 5 Rows of Data Set :
   S_No Ind1 Ind2  Ind3  Ind4  Ind5  Dep
0    1    1   11     1     7     1    1
1    2    2   12     1     7     1    0
2    3    3   13    21     7     2    1
3    4    4   14     1     8     1    0
4    5    5   15     1     8     2    1
```

Shape of Data Set :
(25, 7)

List of Independent Features : ['Ind1', 'Ind2', 'Ind3', 'Ind4', 'Ind5']

Independent Data Set :

	Ind1	Ind2	Ind3	Ind4	Ind5
0	1	11	1	7	1
1	2	12	1	7	1
2	3	13	21	7	2
3	4	14	1	8	1
4	5	15	1	8	2
5	6	16	21	7	1
6	7	17	1	7	1
7	8	18	1	8	2
8	9	19	21	8	1
9	10	110	1	8	2
10	11	111	1	8	1
11	12	112	1	7	1
12	13	21	2	7	1
13	14	22	2	7	2
14	15	23	2	8	1
15	16	24	22	8	2
16	17	25	2	8	1
17	18	26	22	7	1
18	19	27	2	7	2
19	20	28	2	8	1
20	21	31	23	8	1
21	22	32	3	7	2
22	23	33	3	7	1
23	24	34	23	8	1
24	25	35	3	8	2

Dependent Data Set :

0	1
1	0
2	1
3	0
4	1
5	1
6	0
7	0
8	0
9	1
10	0
11	1
12	0
13	1
14	0
15	0
16	1
17	1
18	1
19	0
20	1
21	1
22	1
23	0
24	1

Name: Dep, dtype: int64

Observed Tested Values of Dependent Data Set :

20	1
11	1
23	0
21	1
15	0

Name: Dep, dtype: int64

Predicted Tested Values of Dependent Data Set :
[1 1 1 1 1]

Test Data Set Score :
0.6

Confusion Matrix :
[[0 2]
[0 3]]

Accuracy :
0.6

Precision :
0.6

Recall :
1.0

Result: The program in PYTHON for KNN Classifier has been implemented successfully.

PRACTICAL 12

Objective: Write a Program in R for Data Visualization using Histogram.

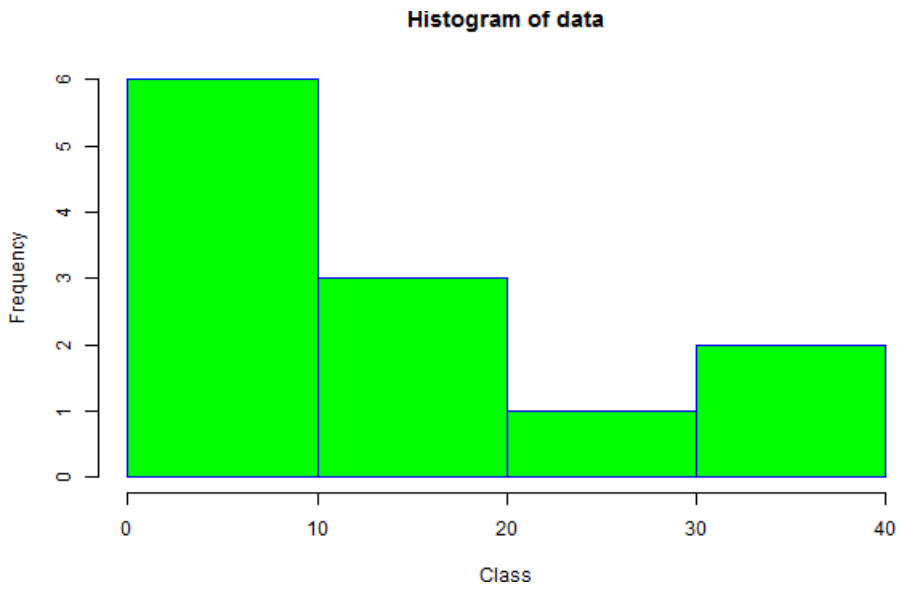
Program:

```
data <- c(2, 3, 5, 7, 8, 1, 20, 15, 12, 25, 40, 32)

png(filename = "histdemo.png", width = 600, height = 400, units = "px")
hist(data, xlab = "Class", col = "green", border = "blue", breaks = 5)

dev.off()
```

Output:



Result: The program in R for Data Visualization using Histogram has been implemented successfully.

PRACTICAL 13

Objective: Write a Program in R to implement Linear Regression Model.

Program:

```
indvardata <- c(2, 5, 8, 1, 17, 3, 9, 6, 4, 16, 10, 11)
depvardata <- c(1, 8, 4, 2, 26, 5, 14, 3, 2, 8, 5, 17)

model <- lm(depvardata ~ indvardata)
print(model)

print("Summary of Relation \n")
print(summary(model))

a <- data.frame(indvardata = 12)
result <- predict(model, a)

print("PREDICTION")
print(result)

a <- data.frame(indvardata = 7)
result <- predict(model, a)

print("PREDICTION")
print(result)
```

Output:

```
Coefficients:
(Intercept)  indvardata
   -0.3961       1.0843

"Summary of Relation \n"

Residuals:
    Min      1Q  Median      3Q     Max
-8.9522 -3.4017 0.2697 3.3904 7.9635


Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.3961     2.7676 -0.143 0.88905
indvardata    1.0843     0.3029 3.580 0.00501 **
---
Signif. codes:  0 „***" 0.001 „**" 0.01 „*" 0.05 „." 0.1 „ " 1

Residual standard error: 5.217 on 10 degrees of freedom
Multiple R-squared:  0.5617,            Adjusted R-squared:  0.5179
F-statistic: 12.82 on 1 and 10 DF, p-value: 0.005012


"PREDICTION"

12.61517

"PREDICTION"

7.19382
```

Result: The program in R for Linear Regression has been implemented successfully.

PRACTICAL 14

Objective: Study of MATLAB as Data Analytics Tool.

Introduction: MATLAB (Matrix Laboratory) is a Multi Programming Numerical Computing Environment and Proprietary Programming Language, developed by MATHWORKS. It allows Matrix Multiplications, Plotting of Functions and Data, Implementation of Algorithms, Creation of User Interfaces and Interfacing with Programs written in other Languages.

MATLAB is used to develop Data Analytics Application to access and analyze Data from a variety of Sources and can be scaled to Clusters, Cloud and Big Data Platforms like HADOOP or SPARK. It enables Engineers and Domain Experts to develop their own Data Analytics Applications.

MATLAB makes Data Science easy with Tools to access and preprocess Data, build Machine Learning and Predictive Models and deploy Models to Enterprise IT Systems.

Benefits:

- Access Data stored in Flat Files, Data Bases, Data Historians and Cloud Storage or connect to Live Sources, such as Data Acquisition Hardware and Financial Data Feeds.
- Manage and Clean Data using Data Types and Preprocessing Capabilities for Programmatic and Interactive Data Preparation, including APPs for Ground – Truth Labeling.
- Document Data Analysis with MATLAB Graphics and the Live Editor Note Book Environment.
- Apply Domain Specific Feature Engineering Techniques for Sensor, Text, Image, Video and other types of Data.
- Explore a wide variety of Modeling approaches using Machine Learning and Deep Learning APPs.
- Fine Tune Machine Learning and Deep Learning Models with Automated Feature Selection and Hyper Parameter Tuning Algorithms.
- Deploy Machine Learning Models to Production IT Systems,.
- Automatically convert Machine Learning Models to Stand Alone C / C++ Code.

MATLAB for Data Analytics: MATLAB supports Exploratory Data Analysis (Reduces Time for Preprocessing of Data), Applied Machine Learning (Fit the Best Machine Learning Model for Classification and Regression) and Multi Platform Deployment (Deploy Machine Learning Models on Web, Enterprise IT Systems, Cloud or on Stand Alone Systems).

MATLAB provides Services like Regression Learner APP, Classification Learner APP, Text Analytics, Bayesian Optimization, Feature Selection, Big Data Handling using Map / Reduce, Data Preprocessing, Cloud Storage, Restful API and JSON.

Result: Study of MATLAB as a Tool for Data Analytics has been done successfully.

PRACTICAL 15

Objective: Write a Program in R to implement Multiple Linear Regression Model.

Program:

```
dset = read.csv("E:/EXAMPLES - R/Data Analytics LAB/MultiLinearRegDataSet.csv", header
= TRUE)
print("First 5 Rows of Imported Data Set : \n")
print(head(dset))

inputdataset <- dset[, c("Ind1", "Ind2", "Ind3", "Dep")]
print("First 5 Rows of Selected Attributes from Imported Data Set : \n")
print(head(inputdataset))

model <- lm(Dep ~ Ind1 + Ind2 + Ind3, data = inputdataset)
print("Multiple Linear Regression Model : \n")
print(model)

a <- coef(model)[1]
print("Value of Intercept : \n")
print(a)

cfind1 <- coef(model)[2]
cfind2 <- coef(model)[3]
cfind3 <- coef(model)[4]

print("Coefficient of Independent Variable 1 is : \n")
print(cfind1)

print("Coefficient of Independent Variable 2 is : \n")
print(cfind2)

print("Coefficient of Independent Variable 3 is : \n")
print(cfind3)

print("Multiple Linear Regression Equation is : \n")
print(paste("DEP = ", a, " + ", cfind1, " * Ind1 + ", cfind2, " * Ind2 + ", cfind3, " * Ind3"))
```

```
chkx1 = as.integer(readline(prompt = "Enter Value for IND 1 to Check the Model : "))
chkx2 = as.integer(readline(prompt = "Enter Value for IND 2 to Check the Model : "))
chkx3 = as.integer(readline(prompt = "Enter Value for IND 3 to Check the Model : "))

chkdep = a + cfind1 * chkx1 + cfind2 * chkx2 + cfind3 * chkx3
print(paste("Value of Dependent Variable for Check Inputs is : ", chkdep))
```

CSV DATA SET:

S_No	Ind1	Ind2	Ind3	Dep
1	2	3	5	185
2	2	3	7	225
3	2	3	8	245
4	2	5	7	255
5	2	5	8	275
6	2	7	8	305
7	3	5	7	267
8	3	5	8	287
9	3	7	8	317
10	5	7	8	341
11	3	5	2	167
12	3	7	2	197
13	3	8	2	212
14	5	7	2	221
15	5	8	2	236
16	1	2	3	118
17	1	2	5	158
18	1	2	7	198
19	1	2	8	218
20	1	3	5	173
21	1	3	7	213
22	1	3	8	233
23	1	5	7	243
24	1	5	8	263
25	1	7	8	293

Output:

"First 5 Rows of Imported Data Set : \n"

	S_No	Ind1	Ind2	Ind3	Dep
1	1	2	3	5	185
2	2	2	3	7	225
3	3	2	3	8	245
4	4	2	5	7	255
5	5	2	5	8	275
6	6	2	7	8	305

"First 5 Rows of Selected Attributes from Imported Data Set : \n"

	Ind1	Ind2	Ind3	Dep
1	2	3	5	185
2	2	3	7	225
3	2	3	8	245
4	2	5	7	255
5	2	5	8	275
6	2	7	8	305

"Multiple Linear Regression Model : \n"

Coefficients:

(Intercept)	Ind1	Ind2	Ind3
16	12	15	20

"Value of Intercept : \n"
16

"Coefficient of Independent Variable 1 is : \n"
12

"Coefficient of Independent Variable 2 is : \n"
15

"Coefficient of Independent Variable 3 is : \n"
20

"Multiple Linear Regression Equation is : \n"

"DEP = 15.9999999999998 + 12 * Ind1 + 15 * Ind2 + 20 * Ind3"

Enter Value for IND 1 to Check the Model : 8

Enter Value for IND 2 to Check the Model : 3

Enter Value for IND 3 to Check the Model : 2

"Value of Dependent Variable for Check Inputs is : 197"

Result: The program in R for Multiple Linear Regression Model has been implemented successfully.