Name:- Rohini Janardan Devkar
Roll no:- 23272
PRN no:- 72030818G
Class:- TE2
DSBDA pr-10.

Practical No: 10.

Data Visualization - III

Aim:-

Download the Iris flower dataset or any other dataset into a dataframe. (e.g. https://archieve.ics.uci.edu/ml/datasets/Iris). Scan the dataset and give the inference as:-

1. How many features are there and what are their types (e.g. numeric, nominal)?
2. Create a histogram for each feature in the dataset to illustrate the feature distributions.
3. Create a boxplot for each feature in the dataset.
4. Compare distributions and identify outliers.

Theory:-

* Data Visualization:-

Data Visualization is a field in data analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data.

Python offers several plotting ~~p~~ libraries, namely Matplotlib, Seaborn and many other data visualization packages with different features for creating Informative, customized and appealing plots to present data in the most simple and effective way.

* Benefits of data visualization:-

1) It promotes improved absorption of business information.
2) With the help of data visualization, decision-makers can easily understand how the data is being interpreted to determine business variations.
3) A large amount of data is handled and is visualized to establish patterns in the data. Many meaningful insights and the evidence behind the data can be used to establish a business goal.
4) Visualizing the data helps managers to achieve growth and use the new pattern trends found in business strategies.

* Python libraries:-

1) Seaborn:-

When you read the official documentation on Seaborn, it is defined as the data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphies. Putting it simply, seaborn is an extension of Matplotlib with advanced features.

2) Matplotlib:-

This is undoubtedly my favourite and a quitessential python library. You can create stories with the data visualized with Matplotlib. Another library from the SciPy stack, Matplotlib plots 2D figures.

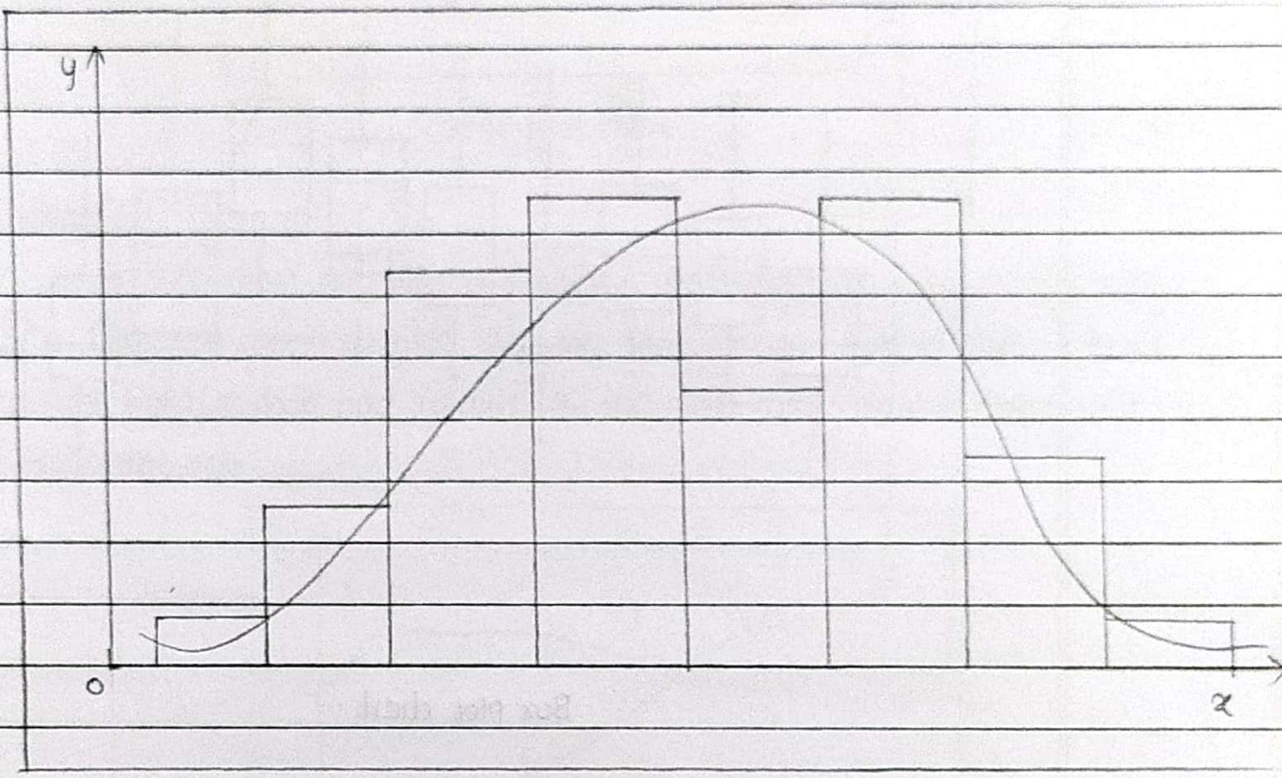
* ~~Benefits of Data Vi~~

Page No.
Date

* Types of graphs :-

1) Histogram :-

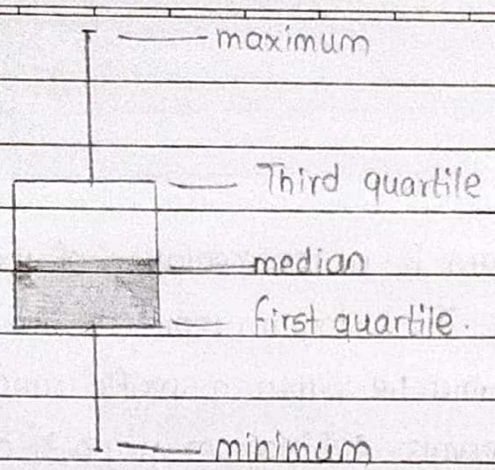
The histogram is a representation of the numerical data, not accurate but an estimate. The histogram represents the frequency of occurence of specific phenomena which lie within a specific range of values and arranged in consecutive and fixed Intervals. A histogram graph is a popular graphing tool that provides a visual representation of data distribution.

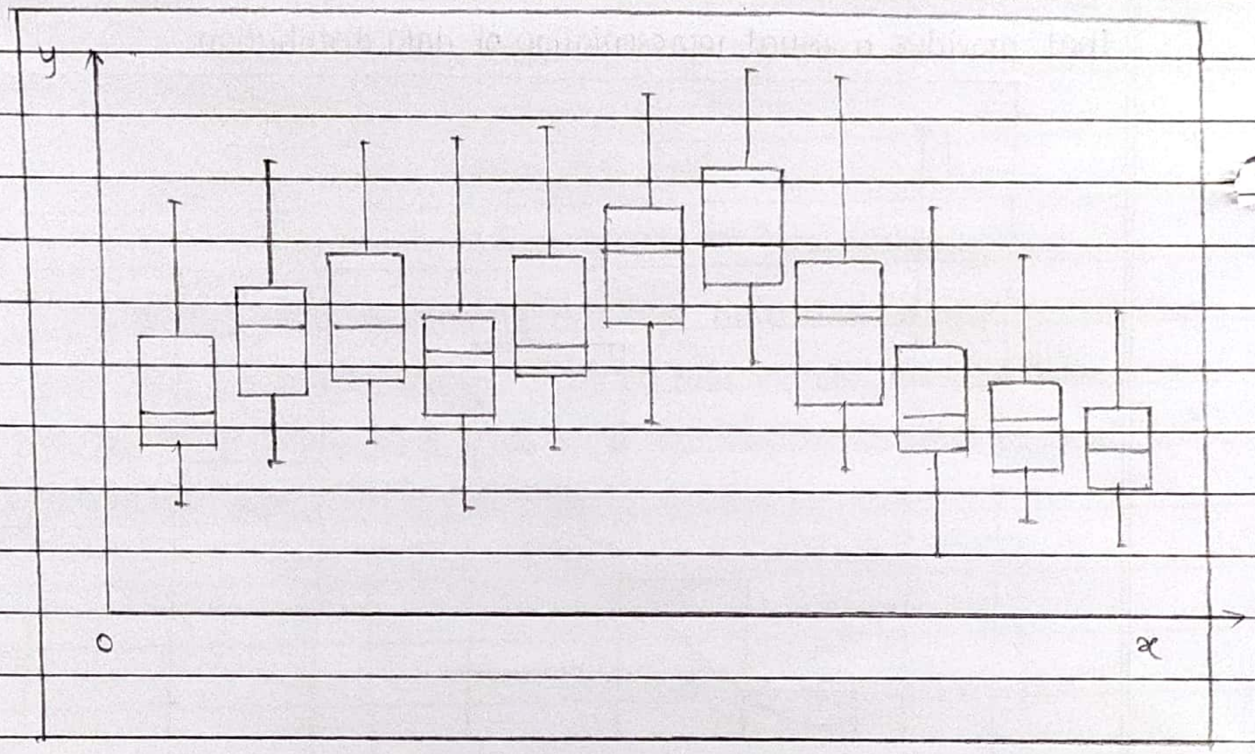


Histogram.

2) Box plot chart :-

A box plot chart is a graphical representation of statistical data based of the minimum, first quartile, median, third quartile and maximum. The term "box plot" comes from the fact that the graph looks like a rectangle with lines extending from the top and bottom. Because of the extending lines, this type of graph is sometimes a box-and-whisker plot.

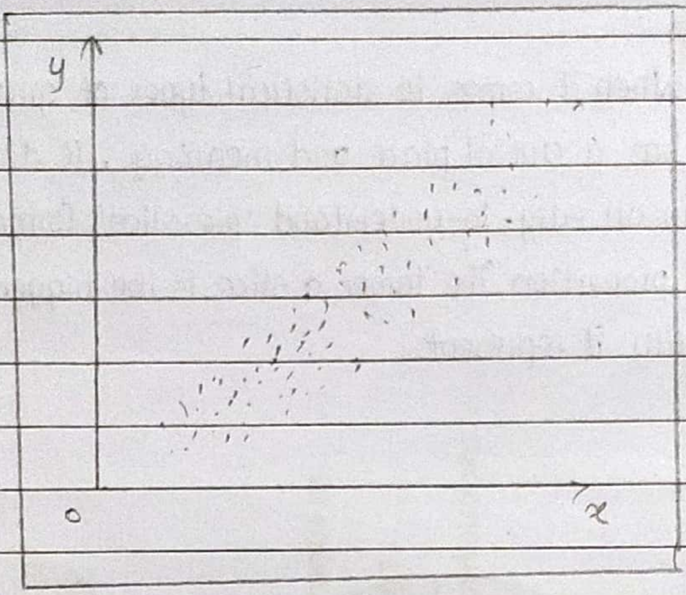




Box plot chart.

3) Scatter plot:-

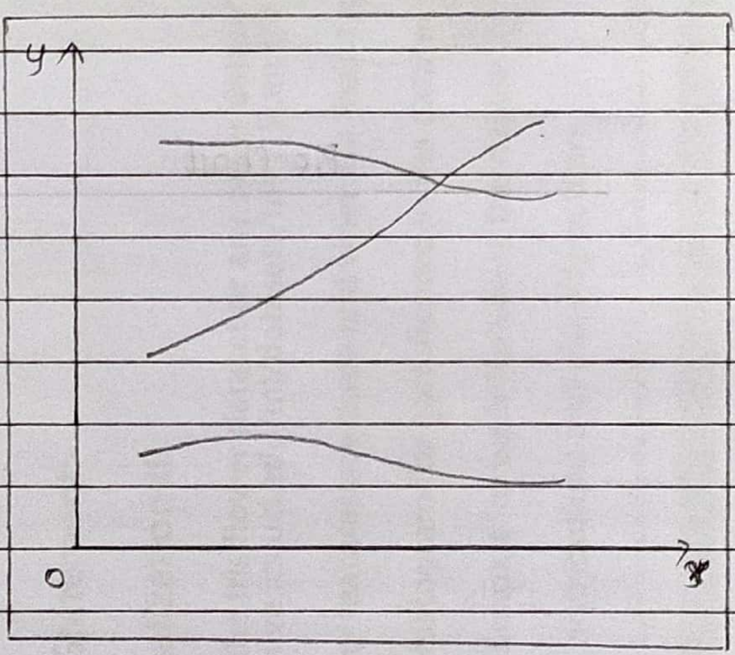
A scatter plot shows the relationship two different variables and it can reveal the distribution trends. It should be used when there are many different data points. The data are displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of other variable determining the position on the vertical axis.



Scatter plot chart.

4) Line graph:-

A line chart graphically displays data that changes continuously over time. Each line graph consists of points that connect data to show a trend. Line graphs have x-axis and y-axis. In the most cases, time is distributed on the horizontal axis.

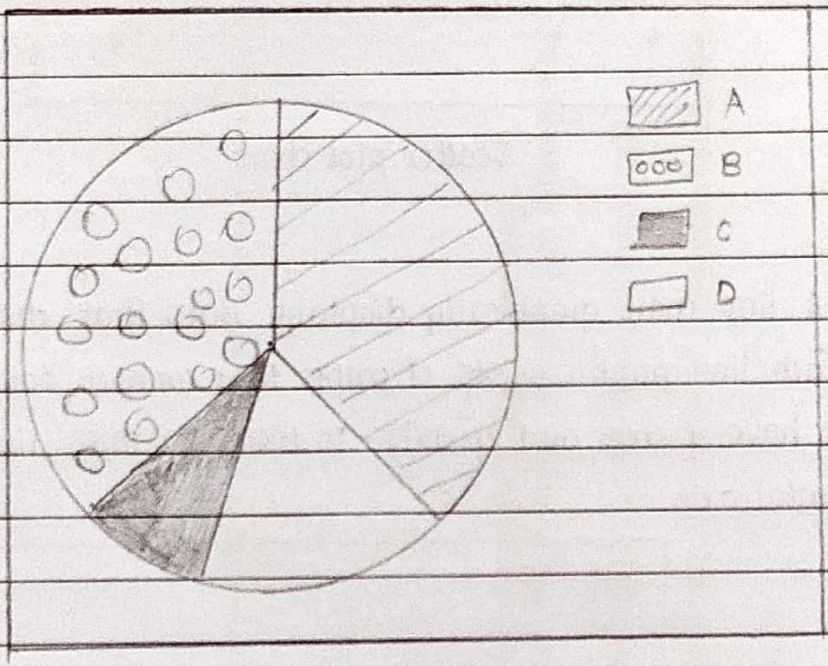


line charts track several variables at once

Line graph.

5) Pie charts:-

When it comes to statistical types of graphs and charts, the pie-chart has a crucial place and meaning. It displays ~~an~~data and statistics in an easy-to-understand 'pie-slice' format and illustrates numerical proportion. The larger a slice is the bigger portion of the total quantity it represents.



Pie Chart.

Data Science And Big Data Analytics Practical - 10

==

Name:- Rohini Devkar

Roll no:- 23272

Prn no:- 72030818G

Class :- TE-2 (COMPUTER)

==

Problem Statement:-

Data Visualization III

Download the Iris flower dataset or any other dataset into a DataFrame. (eg https://archive.ics.uci.edu/ml/datasets/Iris). Scan the dataset and give the inference as:

1. How many features are there and what are their types (e.g., numeric, nominal)?

2. Create a histogram for each feature in the dataset to illustrate the feature distributions.

3. Create a boxplot for each feature in the dataset.

4.Compare distributions and identify outliers.

==

In [23]:

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

In [24]:

```
path = "iris.csv"
```

```
df = pd.read_csv(path, header=None)

headers = ["Sepal-length", "Sepal-width", "Petal-length", "Petal-width", "Species"]
df.columns = headers
```

In [25]:

```
print(df.head())
```

```
   Sepal-length  Sepal-width  Petal-length  Petal-width      Species
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa
```

In [26]:

```
print(df.tail())
```

```
     Sepal-length  Sepal-width  Petal-length  Petal-width         Species
145           6.7          3.0           5.2          2.3  Iris-virginica
146           6.3          2.5           5.0          1.9  Iris-virginica
147           6.5          3.0           5.2          2.0  Iris-virginica
148           6.2          3.4           5.4          2.3  Iris-virginica
149           5.9          3.0           5.1          1.8  Iris-virginica
```

In [27]:

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Sepal-length  150 non-null    float64
 1   Sepal-width   150 non-null    float64
 2   Petal-length  150 non-null    float64
 3   Petal-width   150 non-null    float64
 4   Species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None
```

In [28]:

```
print(df.shape)
```

```
(150, 5)
```

In [29]:

```
print(df.dtypes)
```

```
Sepal-length    float64
Sepal-width     float64
Petal-length    float64
Petal-width     float64
Species          object
dtype: object
```

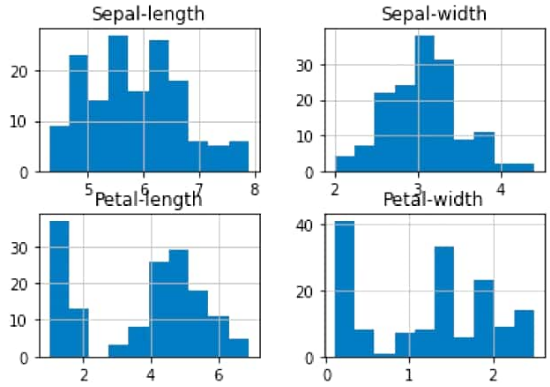
In [30]:

```
print(df.describe())
```

```
       Sepal-length  Sepal-width  Petal-length  Petal-width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

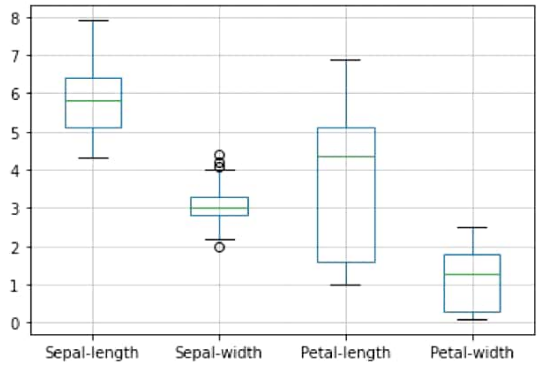
In [31]:

```
df.hist()
plt.show()
```



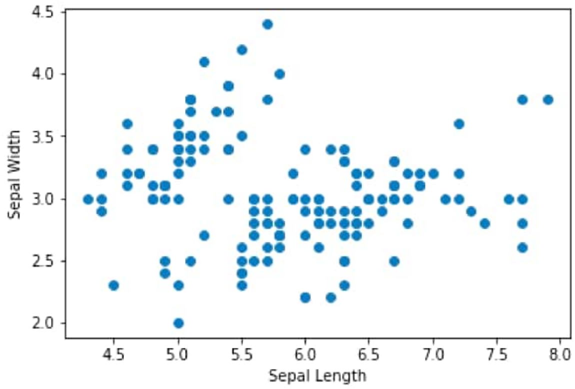
In [32]:

```
df.boxplot()
plt.show()
```



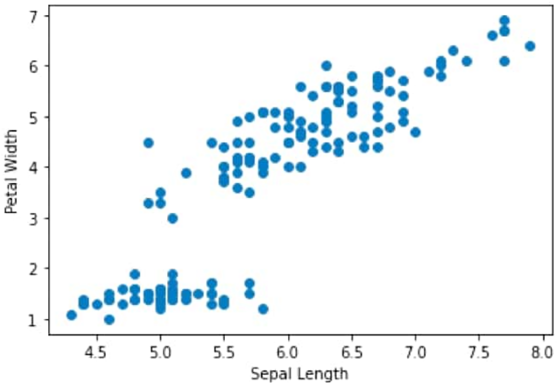
In [33]:

```
plt.scatter(df["Sepal-length"], df["Sepal-width"])
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.show()
```



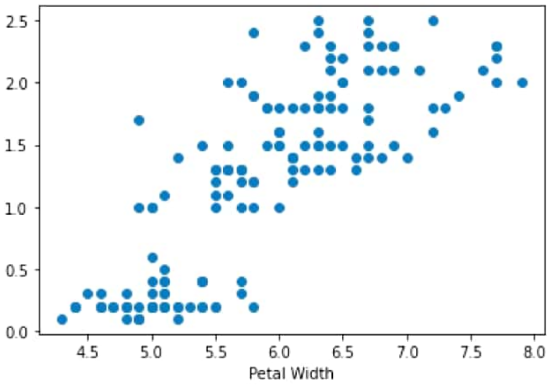
In [34]:

```
plt.scatter(df["Sepal-length"], df["Petal-length"])
plt.xlabel('Sepal Length')
plt.ylabel('Petal Width')
plt.show()
```



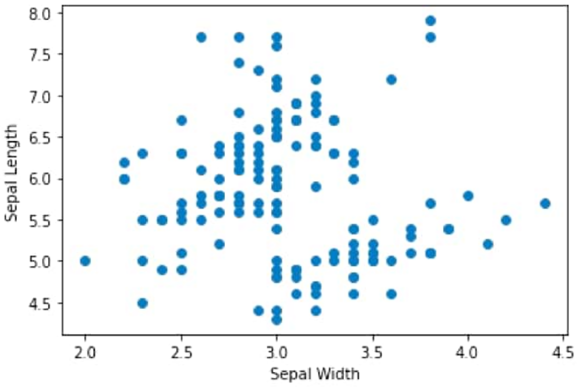
In [35]:

```
plt.scatter(df["Sepal-length"], df["Petal-width"])
plt.xlabel('Sepal Length')
plt.xlabel('Petal Width')
plt.show()
```



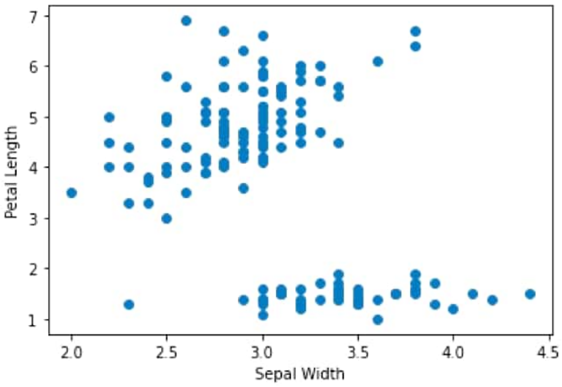
In [36]:

```
plt.scatter(df["Sepal-width"], df["Sepal-length"])
plt.xlabel('Sepal Width')
plt.ylabel('Sepal Length')
plt.show()
```

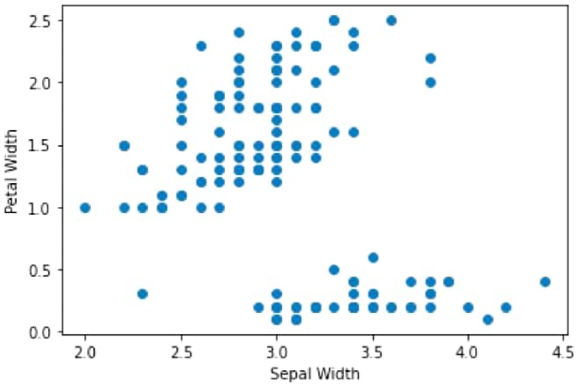


In [37]:

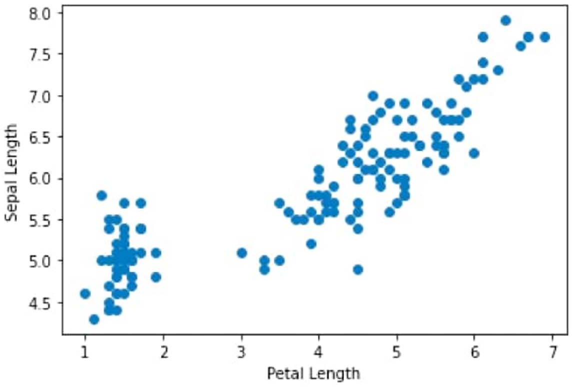
```
plt.scatter(df["Sepal-width"], df["Petal-length"])
plt.xlabel('Sepal Width')
plt.ylabel('Petal Length')
plt.show()
```



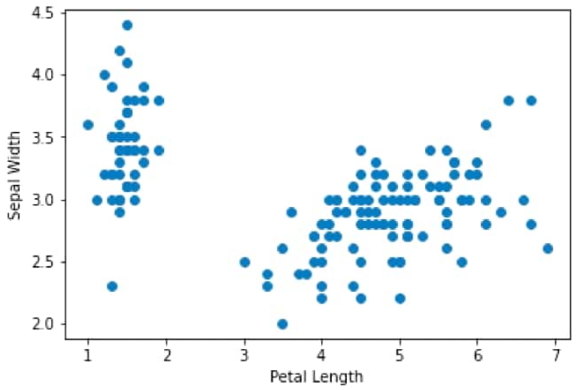
```
In [38]: plt.scatter(df["Sepal-width"], df["Petal-width"])
         plt.xlabel('Sepal Width')
         plt.ylabel('Petal Width')
         plt.show()
```



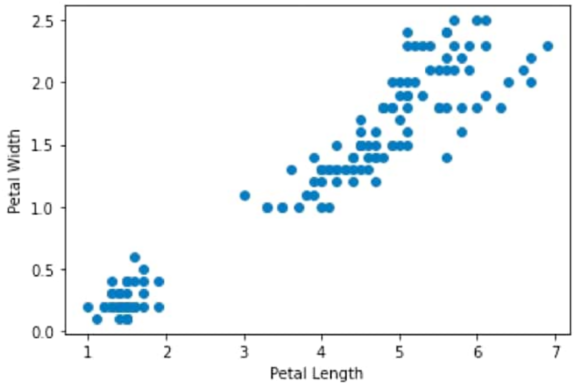
```
In [39]: plt.scatter(df["Petal-length"], df["Sepal-length"])
         plt.xlabel('Petal Length')
         plt.ylabel('Sepal Length')
         plt.show()
```



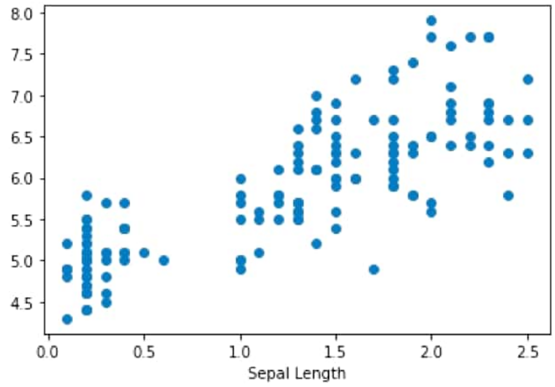
```
In [40]: plt.scatter(df["Petal-length"], df["Sepal-width"])
         plt.xlabel('Petal Length')
         plt.ylabel('Sepal Width')
         plt.show()
```



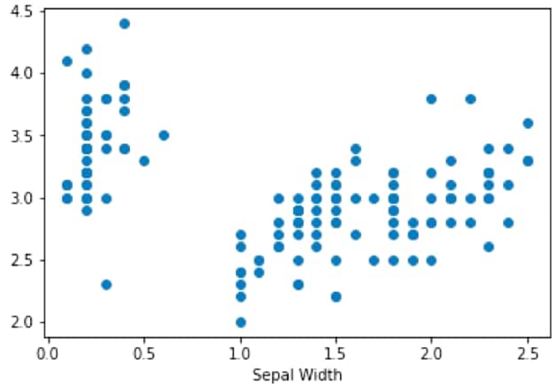
```
In [41]: plt.scatter(df["Petal-length"], df["Petal-width"])
         plt.xlabel('Petal Length')
         plt.ylabel('Petal Width')
         plt.show()
```

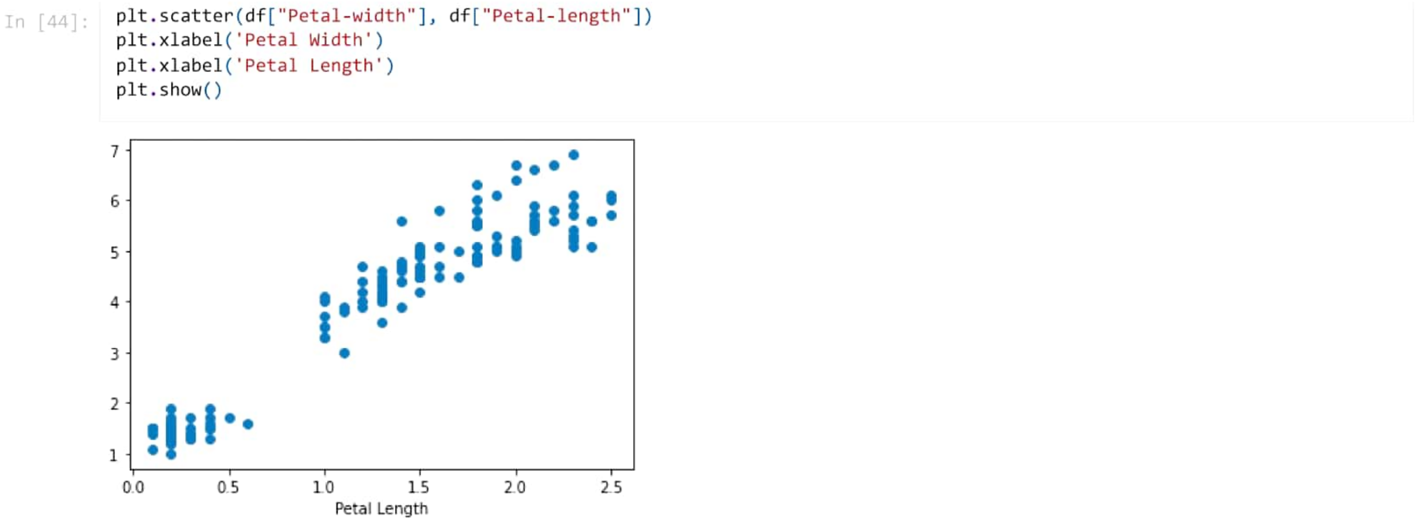


```
In [42]: plt.scatter(df["Petal-width"], df["Sepal-length"])
         plt.xlabel('Petal Width')
         plt.xlabel('Sepal Length')
         plt.show()
```



```
In [43]: plt.scatter(df["Petal-width"], df["Sepal-width"])
         plt.xlabel('Petal Width')
         plt.xlabel('Sepal Width')
         plt.show()
```





In [44]:

```
plt.scatter(df["Petal-width"], df["Petal-length"])
plt.xlabel('Petal Width')
plt.xlabel('Petal Length')
plt.show()
```

Page No.
Date

* Conclusion:-

In this practical, we use matplotlib library. We create a histogram foreach feature in the dataset and we also create boxplot for each feature in the dataset.