Experiment-1

Student Name: YANA SRIVASTAVA
UID: 20BCS2279
Branch: CSE
Section/Group: 906/B
Semester: 5th
Date of Performance: 18/08/2022
Subject Name: ML Lab
Subject Code: 20CSP-317

1. Aim/Overview of the practical: Implement Exploratory Data Analysis on any data set.

2. Task to be done:

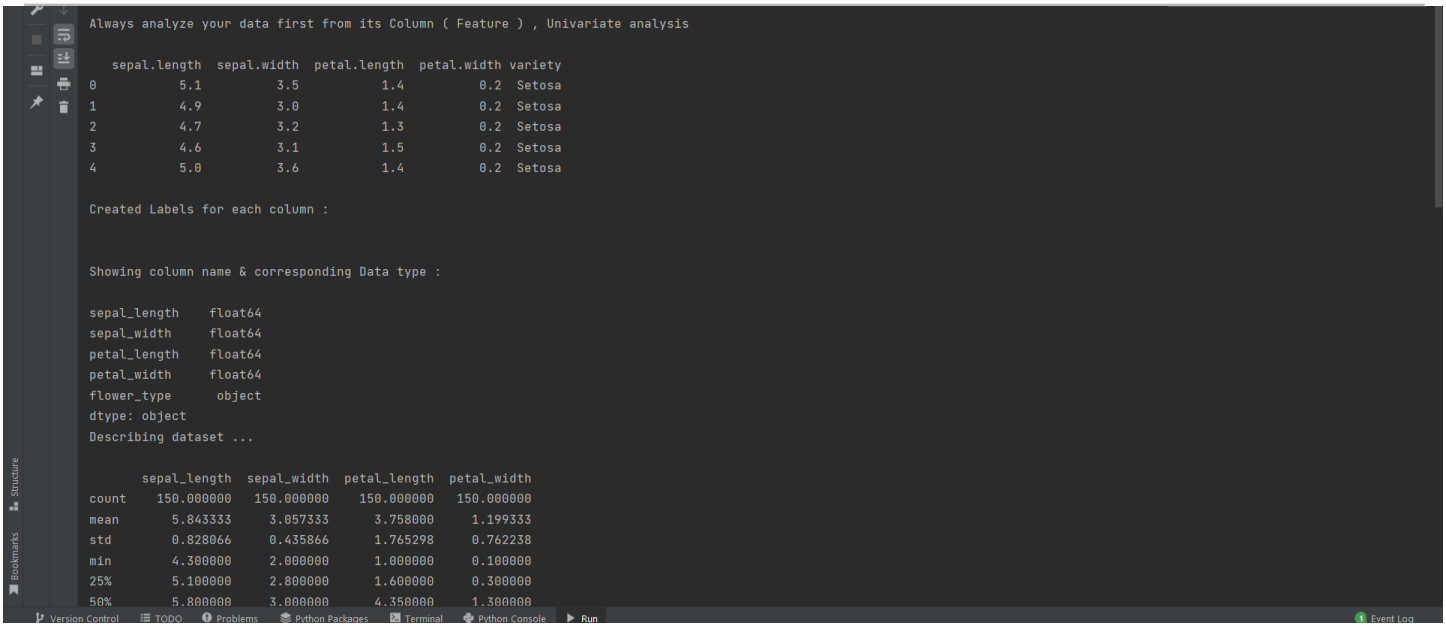
- Load dataset.
- Create a deep copy of dataset so that your original dataset remain same.
- Analyse column of dataset & if necessary then rename it .
- Check datatype of each column & check for NAN also.
- If there is NAN value then check the count of such number & fill or drop it accordingly.
- To get more insights about dataset create more columns i.e. Feature Scaling.
- Show changes that you made so far.

3. CODE:

```
#CHIRAG BITHER
#20BCS1838
#Pandas :
import pandas as pd
import numpy as np
# Dataset is loaded into an object i.e : iris

iris = pd.read_csv("C:\\Users\\satya\\Downloads\\iris.csv")

# By doing this you will get another object of iris with the same value but different reference
# Deep copy :
dataFrame = iris.copy()
print("\nAlways analyze your data first from its Column ( Feature ) , Univariate analysis \n")
print(dataFrame.head())
iris.columns = ["sepal_length","sepal_width","petal_length","petal_width","flower_type"]
print("\nCreated Labels for each column :\n")
dataFrame.columns = ["sepal_length","sepal_width","petal_length","petal_width","flower_type"]
# It will show column name & corresponding Data type :
print("\nShowing column name & corresponding Data type :\n")
print(dataFrame.dtypes)
# Gives us statistical frequently used information about our data ONLY in Number columns :
print("Describing dataset ...\n ")
print(dataFrame.describe())
# To access particular column you can write df.column
# To check is there any null in column :
print("\nChecking for null in each column : \n")
print(dataFrame.isnull().sum())

# Handling NAN :
# Creating NAN so that we can learn how to handle this problem :
# row & column specifically with iloc : row as slicing & column as slicing
dataFrame.iloc[2:4,1:3] = np.nan
print(dataFrame.describe())
```

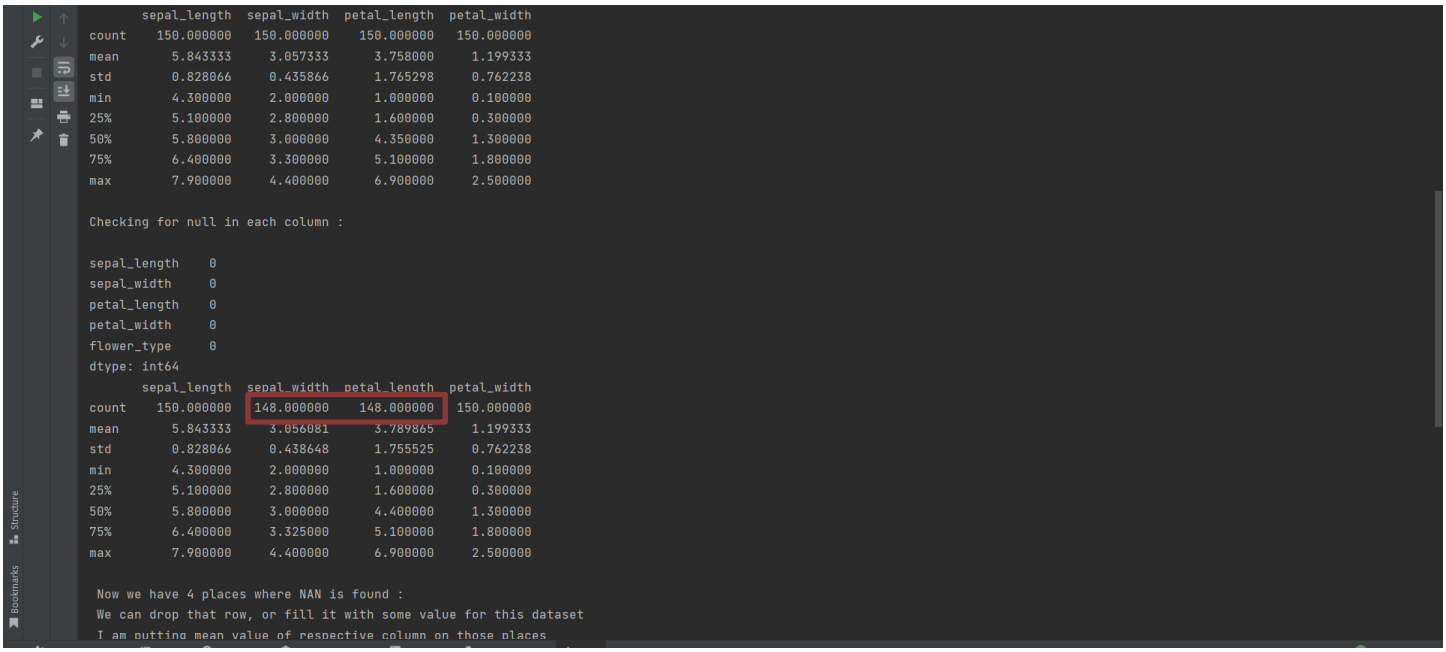
```
print("\n Now we have 4 places where NAN is found :")
print(" We can drop that row, or fill it with some value for this dataset ")
print(" I am putting mean value of respective column on those places\n ")

# Putting mean value inplace of NAN :
dataFrame.sepal_width.fillna(iris.sepal_width.mean(),inplace = True)
dataFrame.petal_length.fillna(iris.sepal_width.mean(),inplace = True)

print(dataFrame.describe())

# Adding new feature : ( Feature scaling )
print("\n Feature scaling : Creating new column for difference of petal length & width : \n")

dataFrame["Difference_Of_Petal_Length_Width"] = dataFrame["petal_length"] - dataFrame["petal_width"]

print(dataFrame.describe())
```

4. OUTPUT:

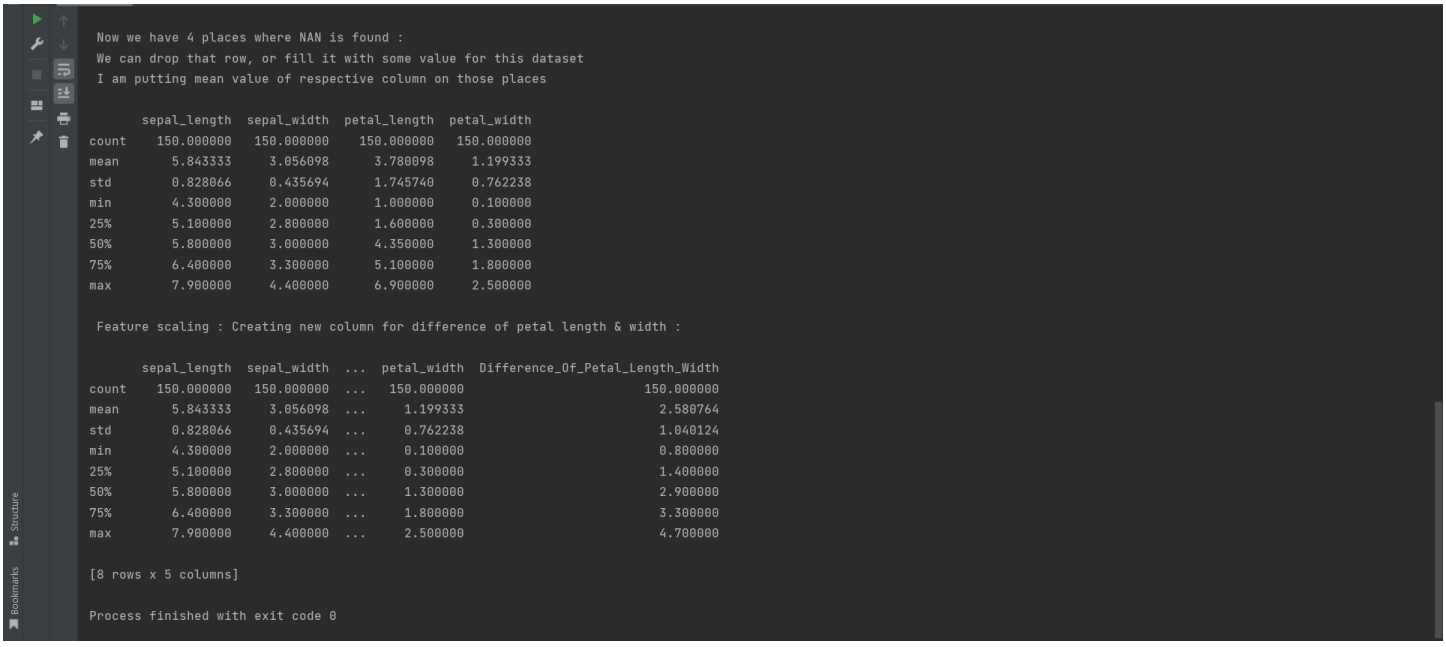


```
Always analyze your data first from its Column ( Feature ) , Univariate analysis

   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa

Created Labels for each column :


Showing column name & corresponding Data type :

sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
flower_type      object
dtype: object
Describing dataset ...

       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.057333      3.758000     1.199333
std        0.828066     0.435866      1.765298     0.762238
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
```

Here we can see that total 4 count is less that we made so that we can fill it with mean value.



```
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.057333      3.758000     1.199333
std        0.828066     0.435866      1.765298     0.762238
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

Checking for null in each column :

sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
flower_type     0
dtype: int64
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   148.000000    148.000000   150.000000
mean       5.843333     3.056081      3.789865     1.199333
std        0.828066     0.438648      1.755525     0.762238
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.400000     1.300000
75%        6.400000     3.325000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000

 Now we have 4 places where NAN is found :
 We can drop that row, or fill it with some value for this dataset
 I am putting mean value of respective column on those places
```



```
 Now we have 4 places where NAN is found :
 We can drop that row, or fill it with some value for this dataset
 I am putting mean value of respective column on those places

       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.056098      3.780098     1.199333
std        0.828066     0.435694      1.745740     0.762238
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

 Feature scaling : Creating new column for difference of petal length & width :

       sepal_length  sepal_width  ...  petal_width  Difference_Of_Petal_Length_Width
count    150.000000   150.000000  ...   150.000000                        150.000000
mean       5.843333     3.056098  ...     1.199333                          2.580764
std        0.828066     0.435694  ...     0.762238                          1.040124
min        4.300000     2.000000  ...     0.100000                          0.800000
25%        5.100000     2.800000  ...     0.300000                          1.400000
50%        5.800000     3.000000  ...     1.300000                          2.900000
75%        6.400000     3.300000  ...     1.800000                          3.300000
max        7.900000     4.400000  ...     2.500000                          4.700000

[8 rows x 5 columns]

Process finished with exit code 0
```

5. Learning outcomes (What I have learnt):

- **Application of numpy library in loading dataset as well as creating & handling NaN values.**
- **Application of Pandas in the form of Data Frames with various functions.**
- **Learned how data scientist creates inferences out of dataset with feature scaling prediction.**
- **Learned how to deal with un-pre-processed dataset.**

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			