```
# Import Libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # data visualization
import seaborn as sns # data visualization

# read our dataset

df = pd.read_csv("IRIS.csv")
df
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
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
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

```
df1 = pd.read_csv("C:\\Users\\DELL\\Desktop\\AAIC-Assignments\\
IRIS.csv")
df1
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
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
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

```
# see the head

df.head()
```

```
   sepal_length  sepal_width  petal_length  petal_width species
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
```

```
df.tail()
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica
```

```
# value_counts() --> how many categories do we have
```

```
# see the columns in dataframe
```

```
df.columns
```

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

```
df["species"].value_counts() # giving counts of categorical data
```

```
versicolor    50
setosa        50
virginica     50
Name: species, dtype: int64
```

```
df.shape
```

```
(150, 5)
```

```
# info --> summary of your dataset
```

```
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
```

```
# dataset have null value --> isnull

df["sepal_length"].isnull().sum()
```

```
0
```

```
# description of iris dataset

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
```

```
df["sepal_length"].describe()
```

```
count    150.000000
mean       5.843333
std        0.828066
min        4.300000
25%        5.100000
50%        5.800000
75%        6.400000
max        7.900000
Name: sepal_length, dtype: float64
```

```
# mean --> average

ls = [10,20,30,40,50,60]
print("before outlier")
print(np.mean(ls))
print(np.median(ls))
```

```
before outlier
35.0
35.0
```

```
# mean --> average

ls = [10,20,30,40,50,60,300]
print("after outlier")
print(np.mean(ls))
print(np.median(ls))
```

```
# mean is highly affected by the outliers whereas median is not...
# fill - null value --> mean, median --> numerical --> median
```

```
after outlier
72.85714285714286
40.0
```

```
ls1 = [19,34,1,56,23,38]
ls2 = [1, 19, 23, 34, 38, 56, 678, nan] # odd numbers - middle --> 23
print(np.median(ls2))
print(np.mean(ls2))
```

```
34.0
121.28571428571429
```

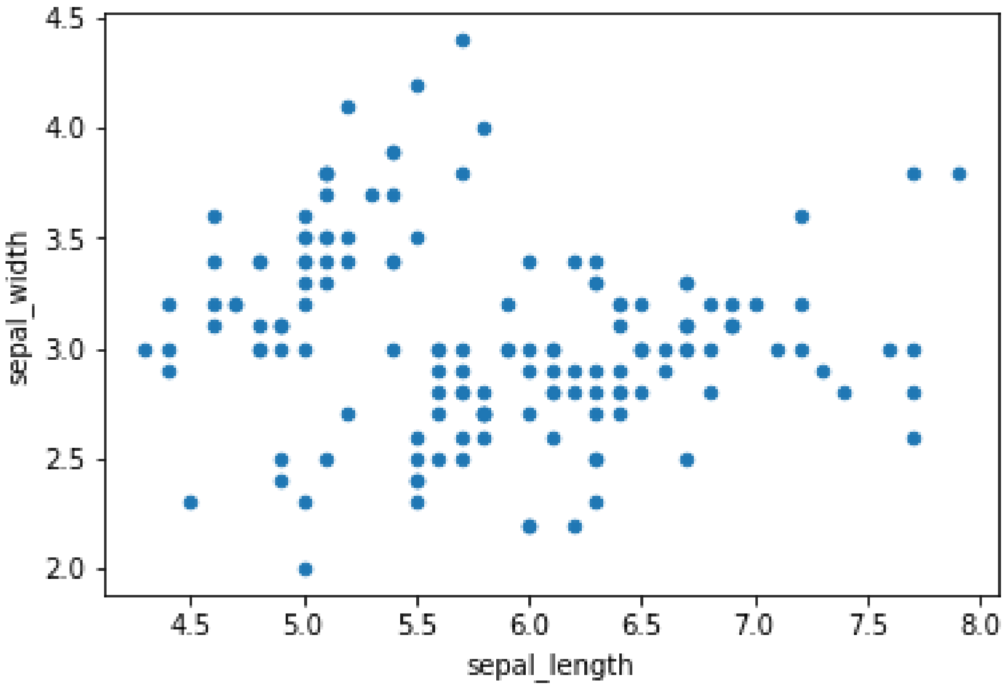
```
# df["col"].fillna(np.median(col))
```

```
# dataset - 50000 - outliers - 10 - 5000 - 1% - 500 (remove the
outliers)
# EDA - Univariant Analysis / Bi variant Analysis / Multi variant
Analaysis
```

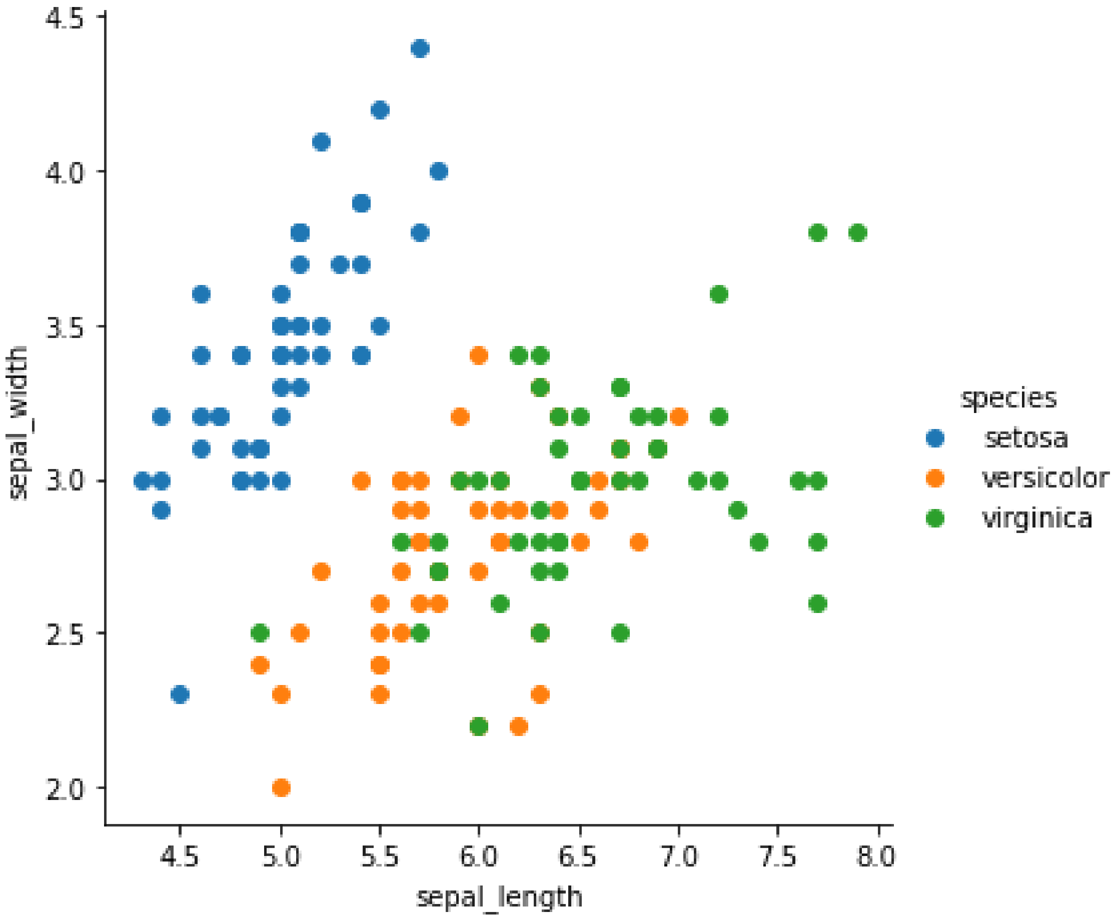
```
df.sepal_length.describe()
```

```
count    150.000000
mean       5.843333
std        0.828066
min        4.300000
25%        5.100000
50%        5.800000
75%        6.400000
max        7.900000
Name: sepal_length, dtype: float64
```

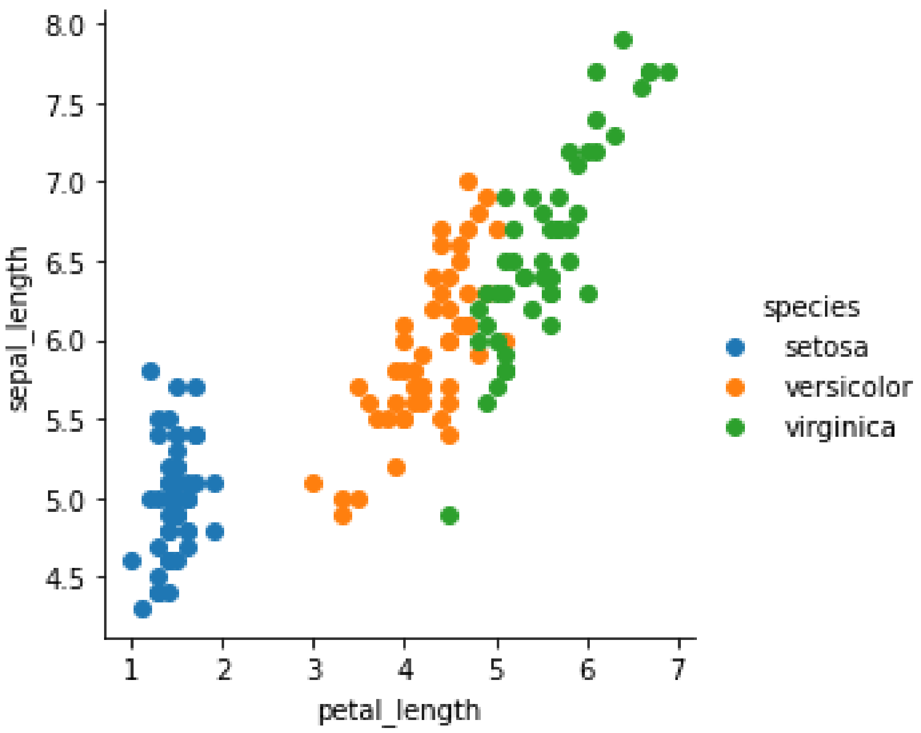
```
# Bi variant analysis -  we are analysing 2 parameters/columns
# x-axis --> sepal_length
# y-axis --> sepal_width
# scatter plot


df.plot(kind = "scatter", x = "sepal_length", y = "sepal_width")
plt.show()
```



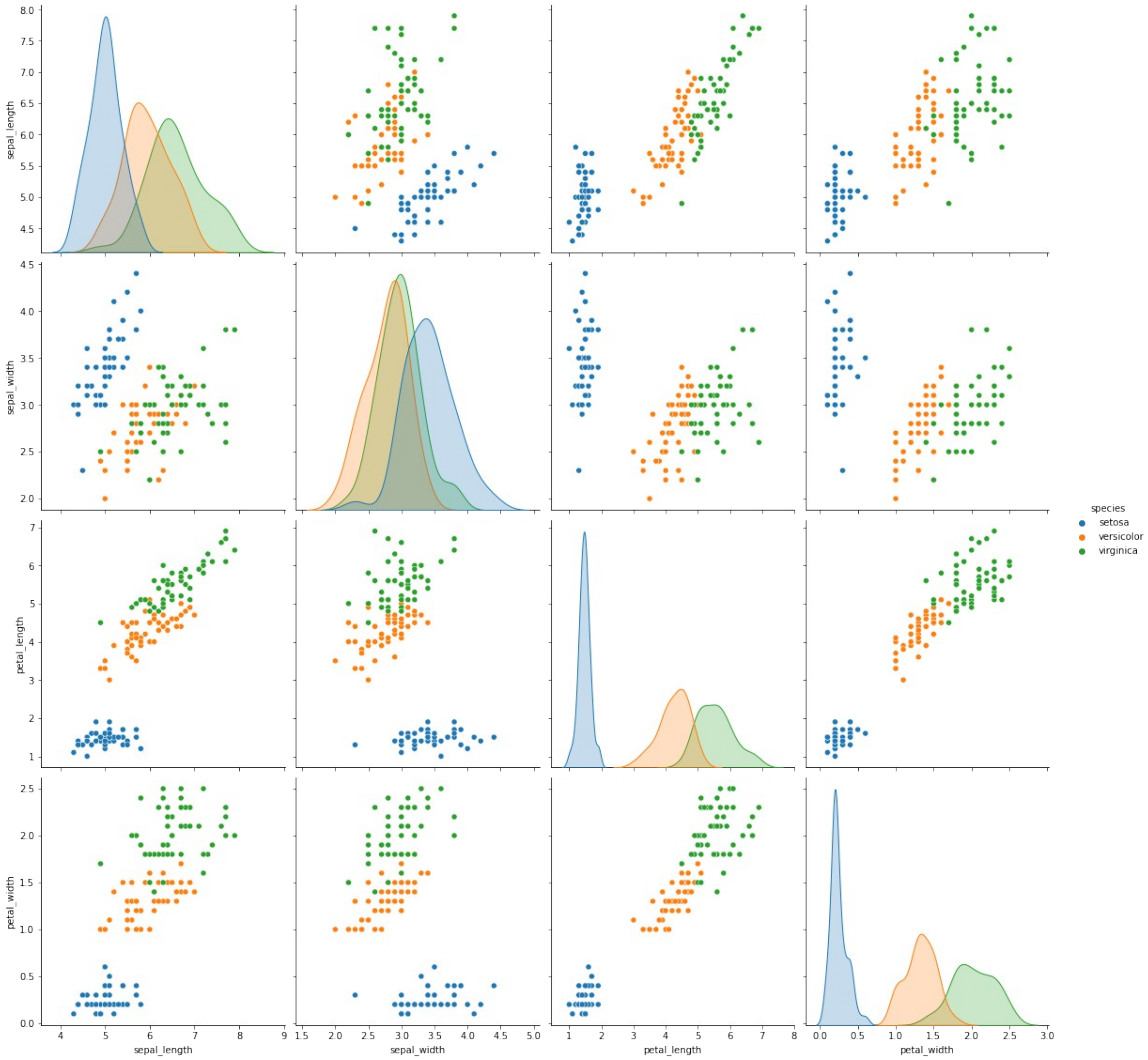
```
sns.FacetGrid(df, hue = "species", height = 5).map(plt.scatter,
"sepal_length", "sepal_width").add_legend()
plt.show()
```



```
sns.FacetGrid(df, hue = "species", height = 4).map(plt.scatter,
"petal_length", "sepal_length").add_legend()
plt.show()
```

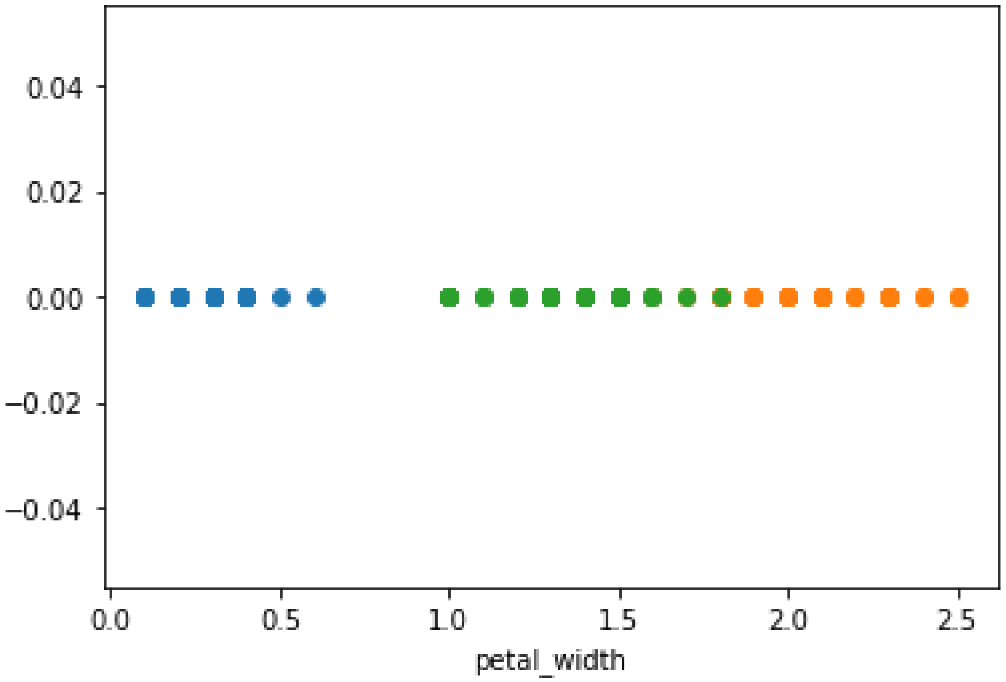


```
# Analysis
# if(petal_length>0 and petal_length<2.5):
#     print("Setosa")
# elif(petal_length>2.5 and petal_length<4.8):
#     print("versicolor")
# else:
#     print("virginica")

# 4 independent features --> sl, sw, pl, pw --> creating scatter plot
--> 2 features
# 4C2 = 12 graphs

[pl,pw,sl,sw] --> [pl,pw] -> [pl,sl] -> [pl,sw]
[pw, sl] --> [pl, sw]

# Pair - Plot - multi variant analysis

sns.pairplot(df, hue = "species", height = 4)
plt.show()
```



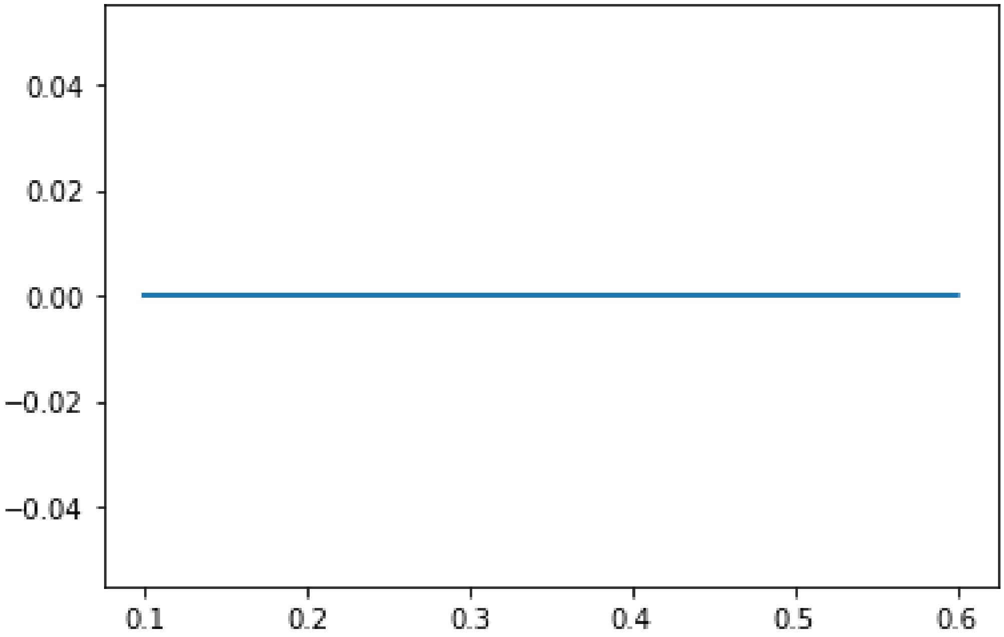
```
# petal_length --> petal_width
# petal_length --> sepal_width
# petal_length --> sepal_length
# petal_width --> sepal_length
# petal_width --> sepal_width


# Univariant Analysis --> histogram / PDF / CDF

# loc / iloc

iris_setosa = df.loc[df["species"] == "setosa"]

iris_virginica = df.loc[df["species"] == "virginica"]

iris_versicolor = df.loc[df["species"] == "versicolor"]
```

```
plt.plot(iris_setosa["petal_width"],
np.zeros_like(iris_setosa["petal_width"]),"o")
plt.plot(iris_virginica["petal_width"],
np.zeros_like(iris_virginica["petal_width"]),"o")
plt.plot(iris_versicolor["petal_width"],
np.zeros_like(iris_versicolor["petal_width"]),"o")

plt.xlabel("petal_width")

plt.show()
```

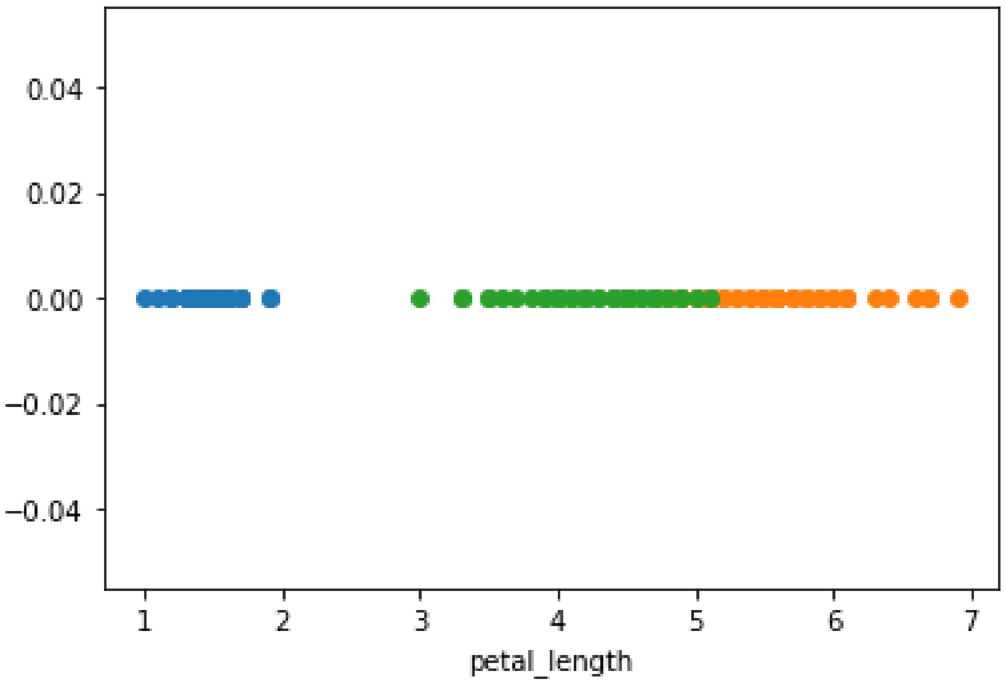


```
print(len(iris_setosa))

50

x = iris_setosa["petal_width"]
print(list(x))
y = np.zeros_like(iris_setosa["petal_width"])
print(y)

plt.plot(x,y)
plt.show()

[0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1,
0.2, 0.4, 0.4, 0.3, 0.3, 0.3, 0.2, 0.4, 0.2, 0.5, 0.2, 0.2, 0.4, 0.2,
0.2, 0.2, 0.2, 0.4, 0.1, 0.2, 0.1, 0.2, 0.2, 0.1, 0.2, 0.2, 0.3, 0.3,
0.2, 0.6, 0.4, 0.3, 0.2, 0.2, 0.2, 0.2]
```

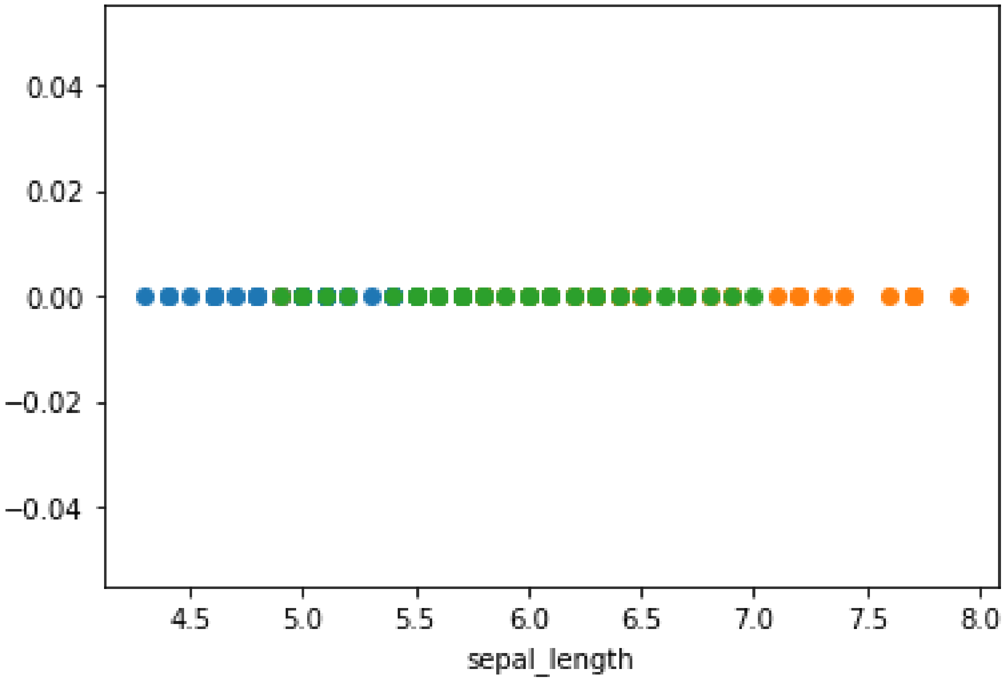
```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0.
 0. 0.]
```



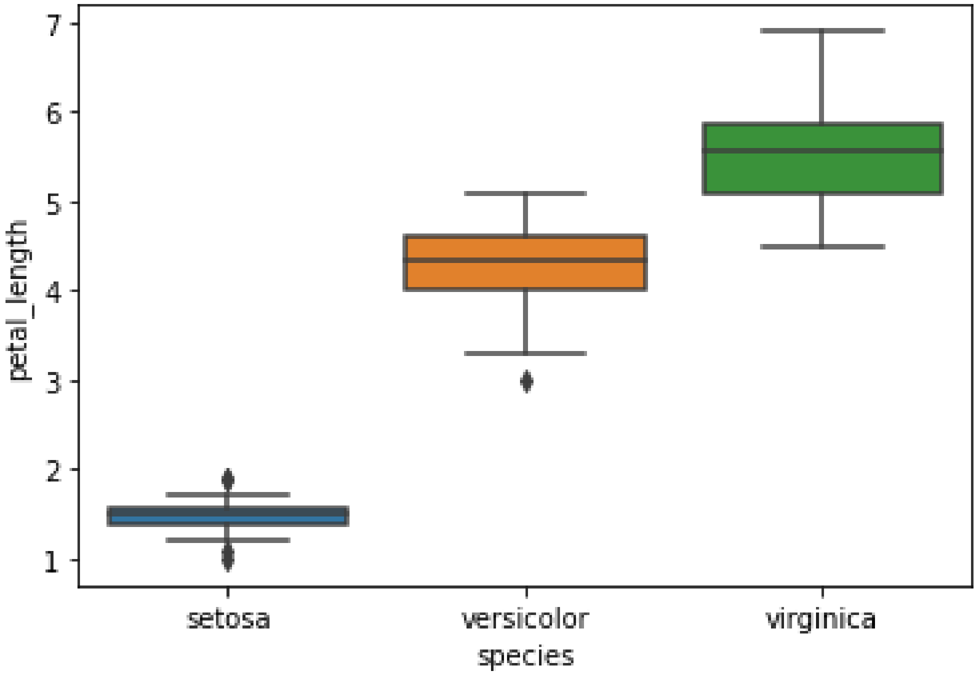
```
# loc / iloc

iris_setosa = df.loc[df["species"] == "setosa"]

iris_virginica = df.loc[df["species"] == "virginica"]

iris_versicolor = df.loc[df["species"] == "versicolor"]

plt.plot(iris_setosa["petal_length"],
np.zeros_like(iris_setosa["petal_length"]),"o")
plt.plot(iris_virginica["petal_length"],
np.zeros_like(iris_virginica["petal_length"]),"o")
plt.plot(iris_versicolor["petal_length"],
np.zeros_like(iris_versicolor["petal_length"]),"o")

plt.xlabel("petal_length")

plt.show()
```



```
# loc / iloc

iris_setosa = df.loc[df["species"] == "setosa"]

iris_virginica = df.loc[df["species"] == "virginica"]

iris_versicolor = df.loc[df["species"] == "versicolor"]

plt.plot(iris_setosa["sepal_length"],
np.zeros_like(iris_setosa["sepal_length"]),"o")
plt.plot(iris_virginica["sepal_length"],
np.zeros_like(iris_virginica["sepal_length"]),"o")
plt.plot(iris_versicolor["sepal_length"],
np.zeros_like(iris_versicolor["sepal_length"]),"o")

plt.xlabel("sepal_length")

plt.show()
```

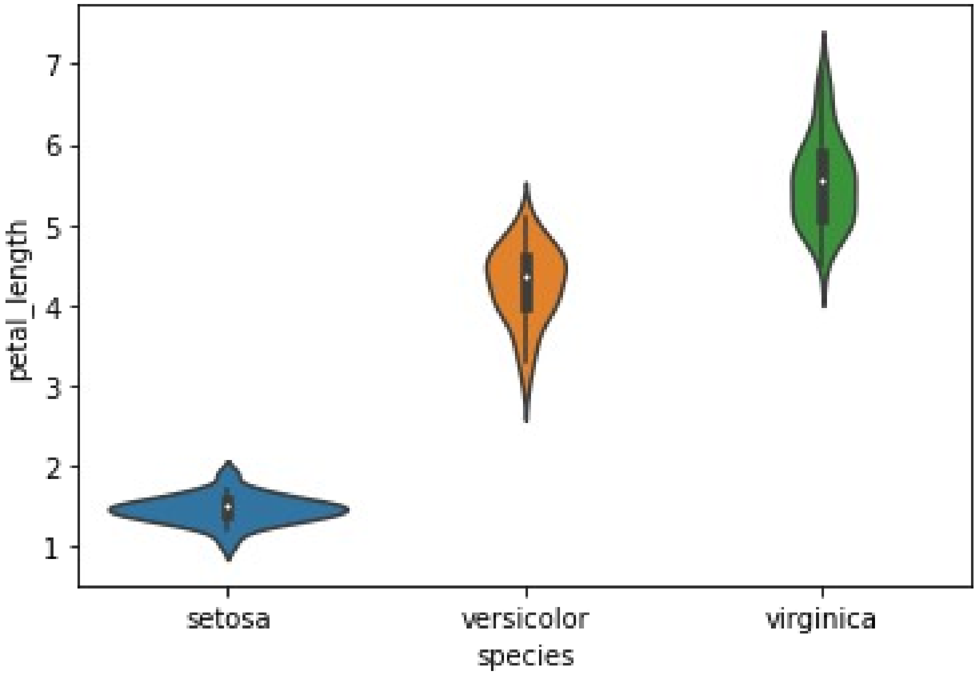


```
# Univariant analysis - Box Plot - Outliers

sns.boxplot(x = "species", y = "petal_length", data = df)
plt.show()
```



```
df[df["species"] == "versicolor"].describe()
```

```
       sepal_length  sepal_width  petal_length  petal_width
count     50.000000    50.000000     50.000000    50.000000
mean       5.936000     2.770000      4.260000     1.326000
std        0.516171     0.313798      0.469911     0.197753
min        4.900000     2.000000      3.000000     1.000000
25%        5.600000     2.525000      4.000000     1.200000
50%        5.900000     2.800000      4.350000     1.300000
75%        6.300000     3.000000      4.600000     1.500000
max        7.000000     3.400000      5.100000     1.800000

sns.violinplot(x = "species", y = "petal_length", data = df)
plt.show()
```



```
# Univariant --> histogram, boxplot, violinplot, scatter plot --> y as
zero
# Bivariant --> 2 variables --> Scatter plot
# Multi variant --> Pairplot

# Groupby

data = {"Name" : ["Akshay","Avinash","Rajat",
"Akshay1","Avinash1","Rajat1", "Akshay2","Avinash2","Rajat2"],
       "Department" : ["CSE","Mech","Civil", "CSE","Mech","Civil",
"CSE","Mech","Civil"],
       "Score" : [89,98,99, 89,98,99, 89,98,99]}

dataf = pd.DataFrame(data)
dataf
```

```
       Name Department  Score
0    Akshay        CSE     89
1   Avinash       Mech     98
2     Rajat      Civil     99
3   Akshay1        CSE     89
4  Avinash1       Mech     98
5    Rajat1      Civil     99
6   Akshay2        CSE     89
7  Avinash2       Mech     98
8    Rajat2      Civil     99
```

```
grouped_data = dataf.groupby("Department")
```

```
grouped_data.first()
```

```
               Name  Score
Department
CSE          Akshay     89
Civil         Rajat     99
Mech        Avinash     98
```

```
grouped_data.last()
```

```
                Name  Score
Department
CSE          Akshay2     89
Civil         Rajat2     99
Mech        Avinash2     98
```

```
grouped_data.sum() # if I want to get the sum of/ total number scores
as per department
```

```
            Score
Department
CSE           267
Civil         297
Mech          294
```

```
# average score in different departments
```

```
grouped_data.mean()
```

```
            Score
Department
CSE            89
Civil          99
Mech           98
```

```
grouped_data.count()
```

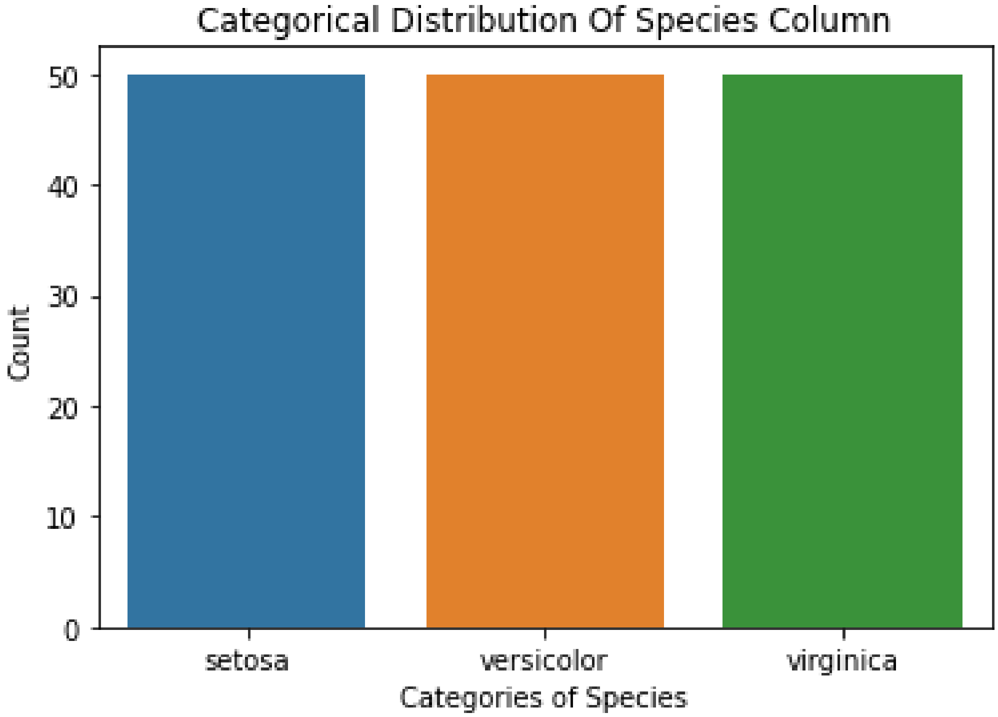
```
            Name  Score
Department
```

```
CSE              3      3
Civil            3      3
Mech             3      3
```

```
df.species.value_counts()
```

```
versicolor    50
setosa        50
virginica     50
Name: species, dtype: int64
```

```
sns.countplot(data  = df[["species"]],  x = "species")
plt.title("Categorical Distribution Of Species Column")
plt.xlabel("Categories of Species")
plt.ylabel("Count")
plt.show()
```



```
df[["species"]]
```

```
       species
0       setosa
1       setosa
2       setosa
3       setosa
4       setosa
..         ...
145  virginica
```

```
146  virginica
147  virginica
148  virginica
149  virginica

[150 rows x 1 columns]
```

```
grouped_data.groups # dictionary --> key --> categorical data // value
--> indexes of it
```

```
{'CSE': [0, 3, 6], 'Civil': [2, 5, 8], 'Mech': [1, 4, 7]}
```

```
# dataset - Families -- Expenditure --> Children

# f1 - 2C - 50k - savings - f3
# f2 - 3C - 67k
# f3 - 1C - 90k
```