Exploratory Data Analysis

In statistics, exploratory data analysis (EDA) is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling and thereby contrasts traditional hypothesis testing. Exploratory data analysis has been promoted by John Tukey since 1970 to encourage statisticians to explore the data, and possibly formulate hypotheses that could lead to new data collection and experiments. EDA is different from initial data analysis (IDA), which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed. EDA encompasses IDA.

Importing the required dependencies for the process.

In [43]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
#import the required dependencies for EDA
from collections import Counter
import pandas_profiling as pp
```

Creating a dataframe using pandas,here im using the IRIS dataset from UCI repository,.

In [44]:
```
df=pd.read_csv('Data/Iris.csv')
```

Checking the top 5 values to see the data contents.

In [45]:
```
df.head()
```

Out[45]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Checking the shape to get more info about the data.

In [46]:
```
df.shape
```

Out[46]:
```
(150, 6)
```

We found out that there are 150 rows with 6 features,which includes the target Y(Species) and an Id part which should be dropped.

In [47]:
```
df=df.drop(['Id'], axis=1)
```

In [48]:
```
df.describe()
```

Out[48]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
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 [49]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

Now its 5 features with 150 rows.

In [50]:
```
#EDA
pp.ProfileReport(df)
```

```
Summarize dataset:                                30/30 [00:02<00:00, 10.58it/s,
100%                                              Completed]

Generate report structure:                              1/1 [00:01<00:00,
100%                                                    1.28s/it]

Render HTML:                                           1/1 [00:00<00:00,
100%                                                   2.65it/s]
```

Lets see how to do a univariate analysis.

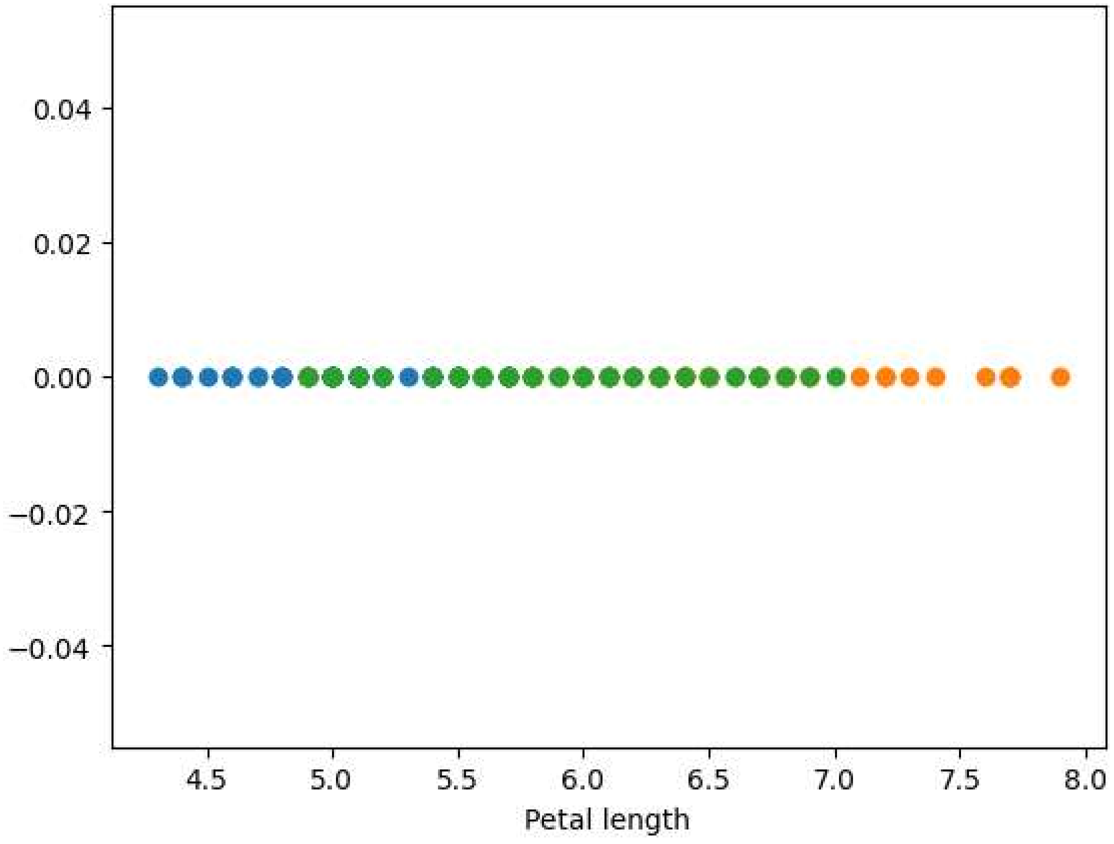
Univariate Analysis

Univariate analysis is the simplest form of analyzing data which consists of observations on only a single characteristic or attribute.It looks at the range of values, as well well as the central tendency of the values and describes the pattern of response to each variable on its own. Descriptive statistics describe and summarize univariate descriptive statistics which describe individual variables.

From the analysis we found that there are three types of targets,Setosa ,Virginica and Versicolour.

In [51]:
```
df_setosa=df.loc[df['Species']=='Iris-setosa']
df_virginica=df.loc[df['Species']=='Iris-virginica']
df_versicolor=df.loc[df['Species']=='Iris-versicolor']
```

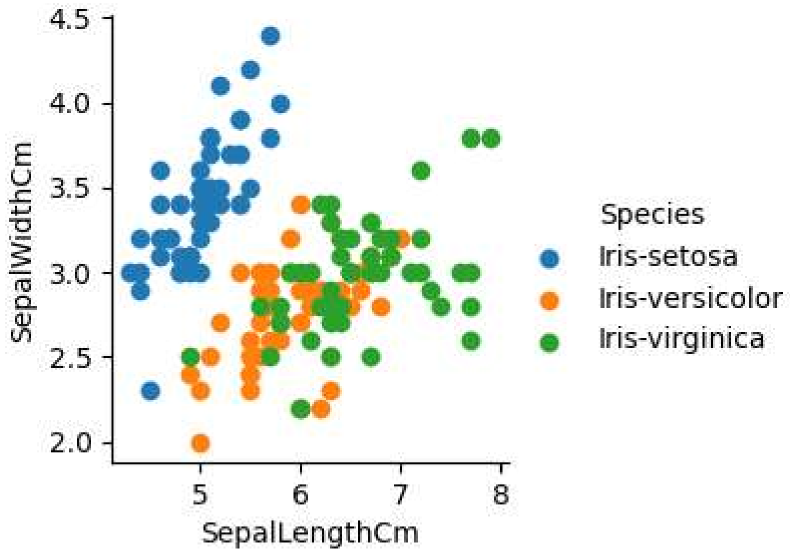
In [52]:
```
plt.plot(df_setosa['SepalLengthCm'],np.zeros_like(df_setosa['SepalLengthCm']),
plt.plot(df_virginica['SepalLengthCm'],np.zeros_like(df_virginica['SepalLength
plt.plot(df_versicolor['SepalLengthCm'],np.zeros_like(df_versicolor['SepalLeng
plt.xlabel('Petal length')
plt.show()
```



Bi-Variate Analysis

Bivariate analysis is a quantitative (statistical) technique used to determine the empirical relationship between two sets of values.It usually involves the variables X and Y and is conducted to determine whether a statistical association exists between them, as well as the degree of association if one does exist.

```
In [53]: sns.FacetGrid(df,hue="Species").map(plt.scatter,"SepalLengthCm","SepalWidthCm"
         plt.show()
```



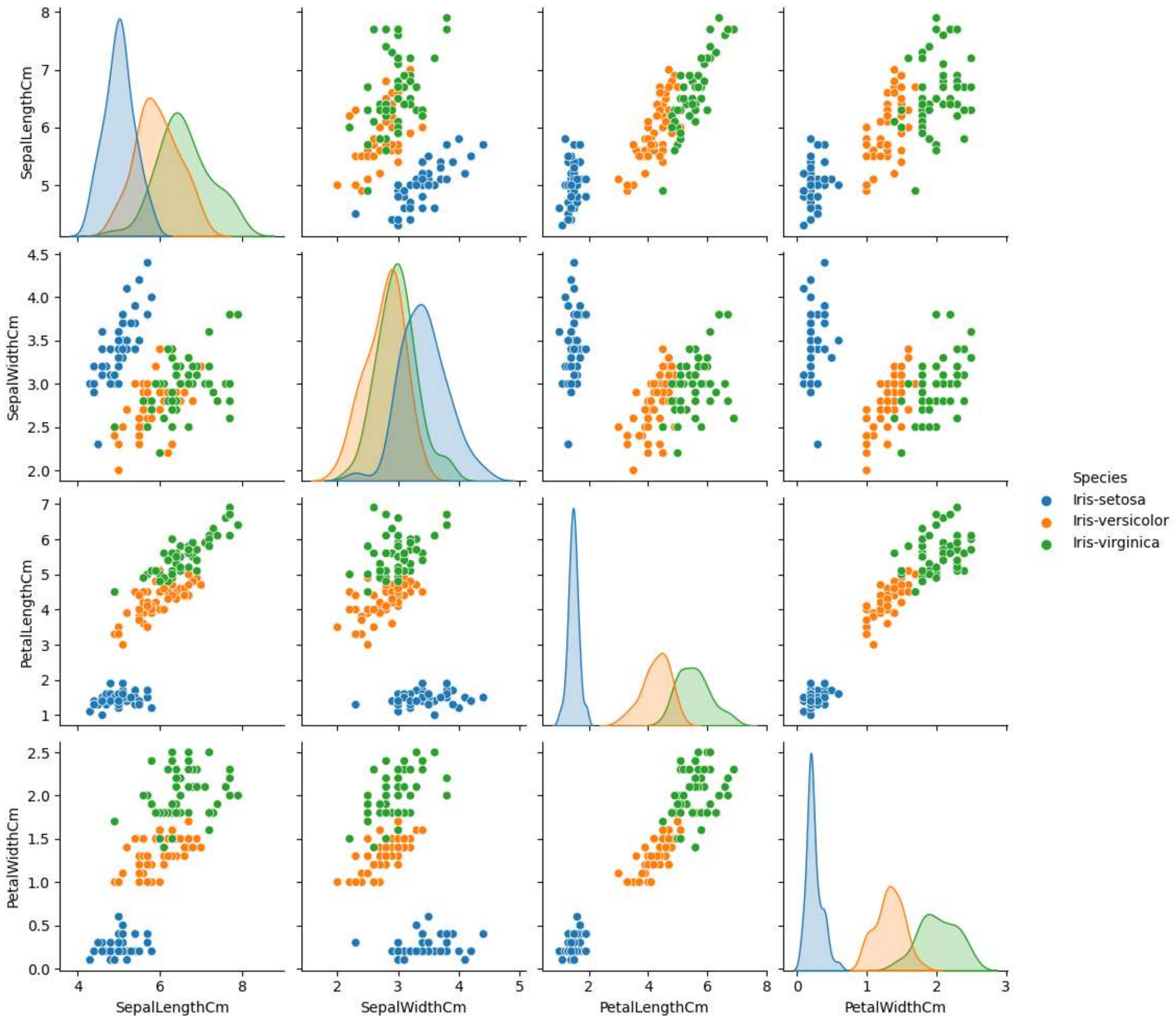
Multi-Variate Analysis

Multivariate analysis is the statistical study of experiments in which multiple measurements are made on each experimental unit and for which the relationship among multivariate measurements and their structure are important to the experiment’s understanding.

as a 3D being we cant comprehend a 4th dimension hence we need to split them for visualization,hence we use pairplot from seaborn

```
In [54]: sns.pairplot(df,hue="Species")
```

```
Out[54]: <seaborn.axisgrid.PairGrid at 0x20d2e8e6770>
```



Using these types of analysis methods we can choose what algorithm we should proceed with and what steps should be taken

```
In [ ]:
```