AUTHOR: DEEKSHITHA C H

DATA SCIENCE AND BUSINESS ANALYTICS INTERN @ THE SPARKS FOUNDATION

TASK 2: PREDICTION USING UNSUPERVISED ML

DATASET: IrisDataset(https://bit.ly/3kXTdox)

IMPORTING NECESSARY LIBRARIES

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd
        from sklearn import datasets

        %matplotlib inline
        print("Imported Sucessfully.")
```

```
Imported Sucessfully.
```

LOADING THE IRIS DATASET

```
In [2]: data = pd.read_csv("C:/Users/HP/Downloads/Iris.csv")
        print("Iris Data Loaded Successfully.")
```

```
Iris Data Loaded Successfully.
```

DATA EXPLORATION AND ANALYSIS

```
In [3]: data.head()
```

Out[3]:

	ID	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	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

```
In [4]: data.tail()
```

Out[4]:

	ID	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

UNDERSTANDING THE DATA

```
In [5]: #CHECKING HOW MUCH DATA POINTS DO EACH SPECIES HAVE
        data['Species'].value_counts()
```

```
Out[5]: Iris-setosa        50
        Iris-versicolor    50
        Iris-virginica     50
        Name: Species, dtype: int64
```

```
In [6]: #CHECKING FOR NULL VALUES
        data.isnull().sum()
```

```
Out[6]: ID              0
        Sepal_Length    0
        Sepal_Width     0
        Petal_Length    0
        Petal_Width     0
        Species         0
        dtype: int64
```

```
In [7]: #DISPLAY THE SUMMARY
        data.describe()
```

Out[7]:

	ID	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

PREPARING THE DATA

```
In [8]: #DESCRIBE THE SHAPE - NUMBER OF ROWS AND COLUMNS
        data.shape
```

```
Out[8]: (150, 6)
```

```
In [9]: #REMOVING A COLUMN FROM THE DATASET
        del data['ID']
```

```
In [10]: #DESCRIBE THE SHAPE - NUMBER OF ROWS AND COLUMNS AFTER REMOVING A COLUMN
         data.shape
```

```
Out[10]: (150, 5)
```

```
In [11]: #INFORMATION REGARDING THE DATASETS
         data.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
```

FINDING THE OPTIMUM NUMBER OF CLUSTERS BY USING K-MEANS CLUSTERING.

THIS IS A PARTITIONING CLUSTERING WHICH REQUIRES THE USER TO SPECIFY THE NUMBER OF CLUSTERS TO BE GENERATED.

THE ELBOW METHOD TO DETERMINE THE OPTIMAL NUMBER OF CLUSTERS:

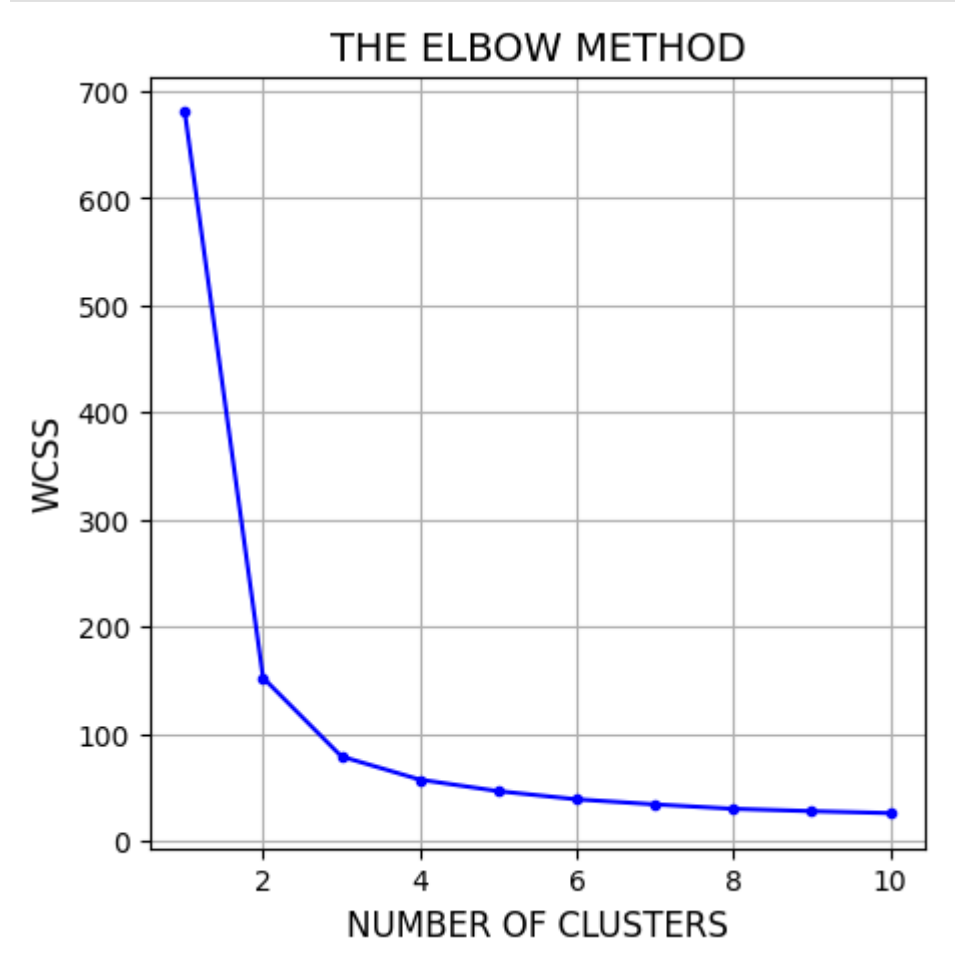
1. **FOR EACH CLUSTERS K, WE CALCULATE THE TOTAL WCSS.**
2. **PLOTTING THE CURVE ACCORDING TO THE NUMBER OF CLUSTERS K.**
3. **THE POINT WHERE THE LINE BEND IN THE PLOT IS CONSIDERED AS APPROPRIATE NUMBER OF CLUSTERS.**

```
In [12]: x = data.iloc[:,:4].values
         from sklearn.cluster import KMeans

         wcss = []
         for i in range(1, 11):
             kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
             kmeans.fit(x)
             wcss.append(kmeans.inertia_)

         pd.DataFrame({"NUMBER OF CLUSTERS": range(1,11), "WCSS": wcss})
```

Out[12]:

	NUMBER OF CLUSTERS	WCSS
0	1	680.824400
1	2	152.368706
2	3	78.940841
3	4	57.317873
4	5	46.561630
5	6	38.930963
6	7	34.190688
7	8	30.063874
8	9	27.879288
9	10	26.048202

```
In [13]: #PLOTTING NUMBER OF CLUSTERS VS WCSS
         plt.figure(figsize=[5,5])
         plt.plot(range(1,11), wcss, 'b.-')
         plt.title("THE ELBOW METHOD", fontsize=14)
         plt.xlabel("NUMBER OF CLUSTERS",fontsize=12)
         plt.ylabel("WCSS", fontsize=12)
         plt.grid()
         plt.show()
```



FROM THE ABOVE ELBOW METHOD WE CAN SEE THAT THE LINE BEND AT THE POINT 3. THERFORE, WE CONSIDER 3 AS NUMBER OF CLUSTERS.

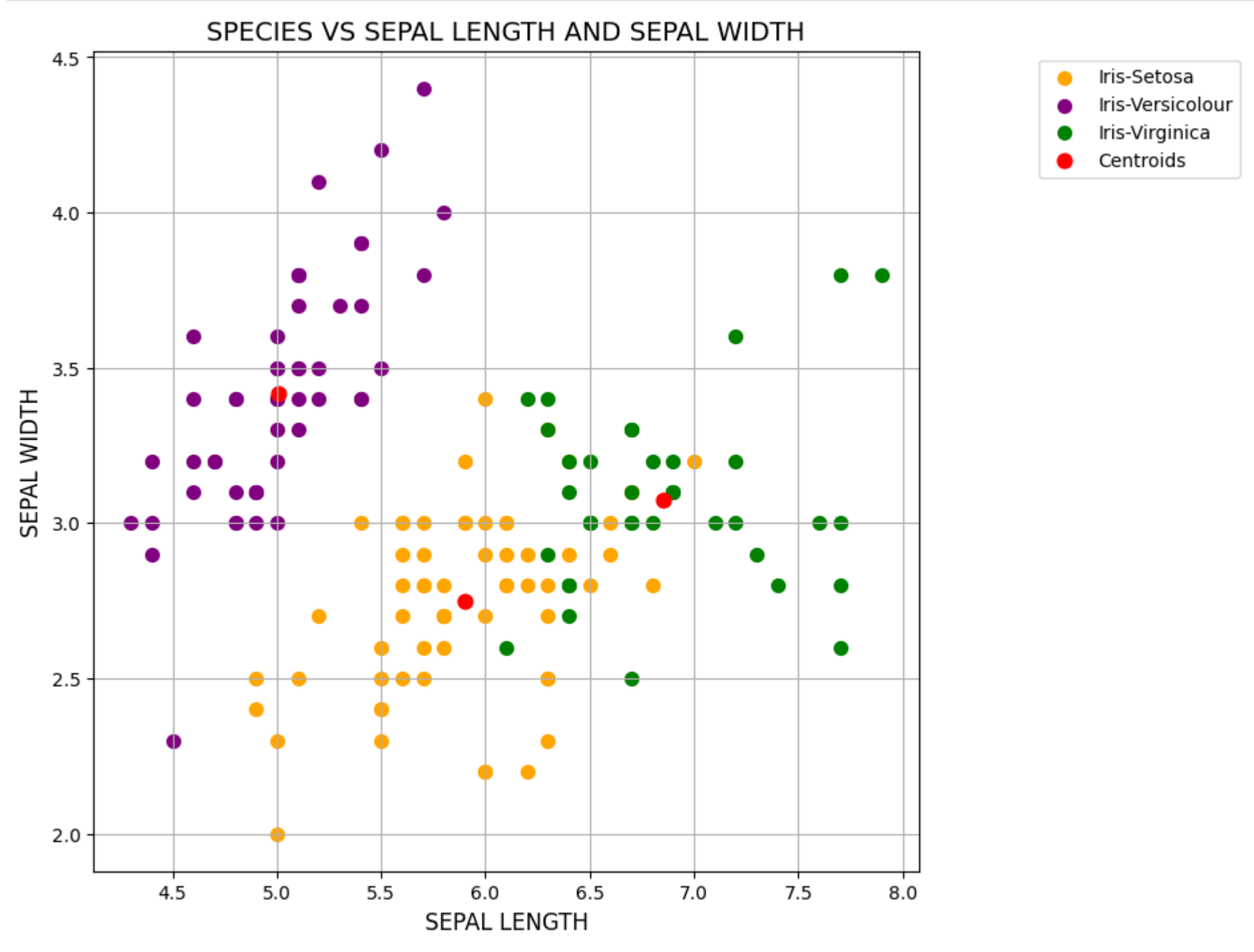
APPLYING K-MEANS TO THE DATASET WITH NUMBER OF CLUSTERS AS K=3

```
In [14]: kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
         y_kmeans = kmeans.fit_predict(x)
         y_kmeans
```

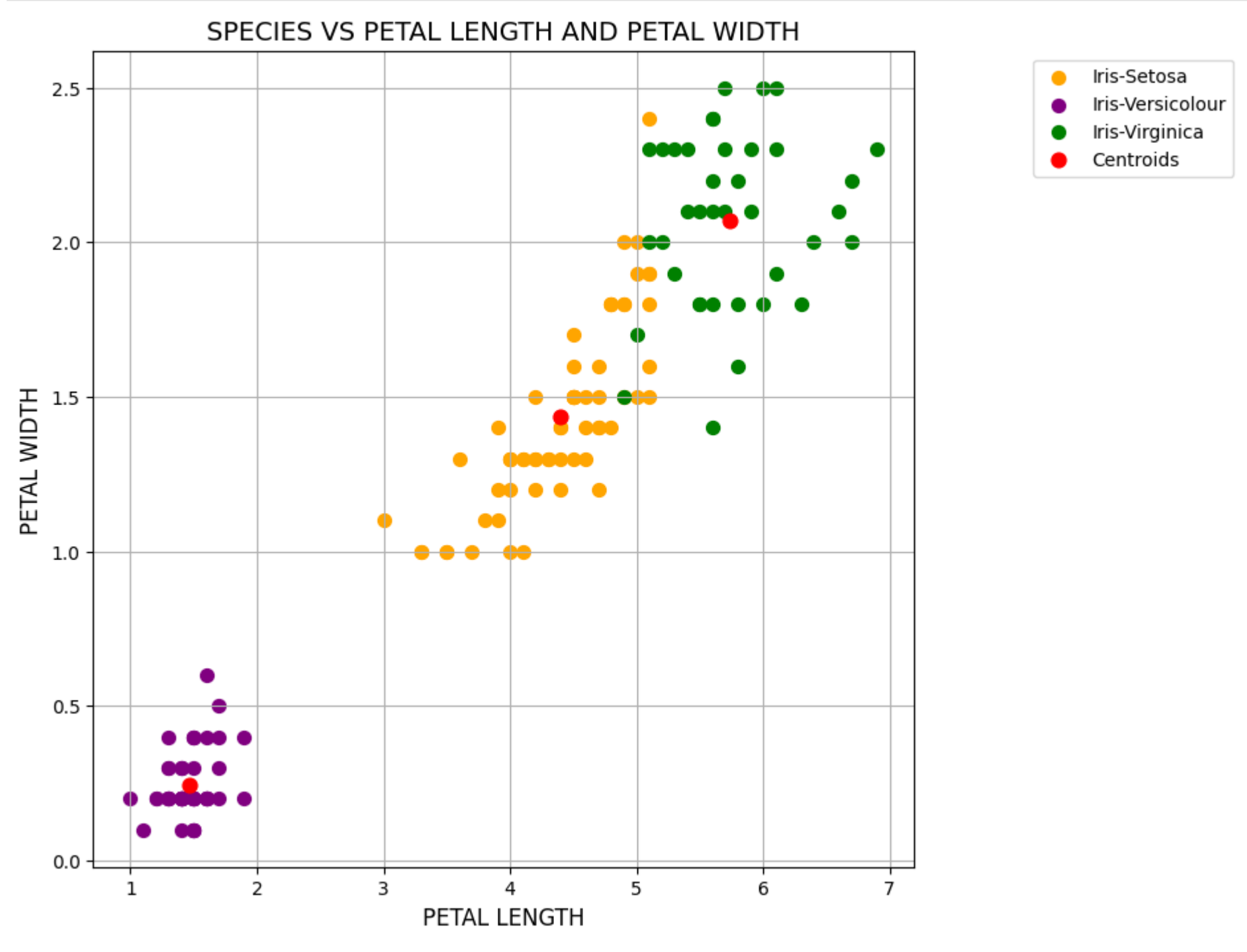
```
Out[14]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 2, 2, 2, 0, 2, 2, 2,
                2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
                2, 0, 2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0])
```

VISUALISING THE CLUSTERS - SCATTER PLOT

```
In [15]: #VISUALISING THE THREE CLUSTERS OF FIRST TWO COLUMNS (SEPAL LENGTH AND SEPAL WIDTH) WITH RESPECT TO SPECIES
         plt.figure(figsize=(8,8))
         plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 50, c = 'Orange', label = 'Iris-Setosa')
         plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 50, c = 'Purple', label = 'Iris-Versicolour')
         plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 50, c = 'Green', label = 'Iris-Virginica')

         #PLOTTING THE CENTROIDS OF THE CLUSTERS
         plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,1], s = 60, c = 'Red', label = 'Centroids')

         plt.legend(loc=1, bbox_to_anchor=(1.4, 1))
         plt.title("SPECIES VS SEPAL LENGTH AND SEPAL WIDTH", fontsize=14)
         plt.xlabel("SEPAL LENGTH", fontsize=12)
         plt.ylabel("SEPAL WIDTH", fontsize=12)
         plt.grid()
         plt.show()
```



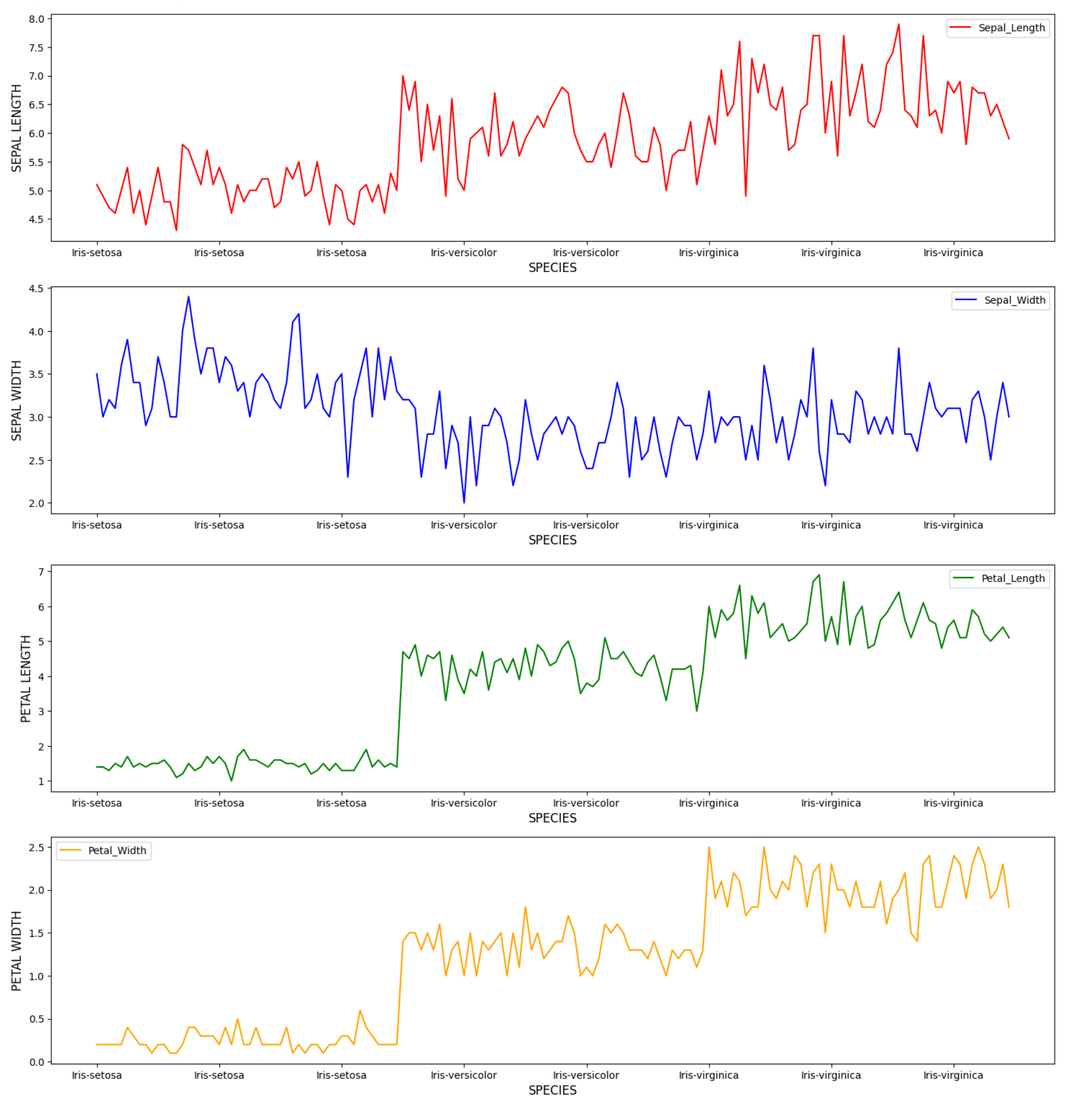

```
In [16]: #VISUALISING THE THREE CLUSTERS OF THIRD AND FOURTH COLUMNS (PETAL LENGTH AND PETAL WIDTH) WITH RESPECT TO SPECIES
         plt.figure(figsize=(8,8))
         plt.scatter(x[y_kmeans == 0, 2], x[y_kmeans == 0, 3], s = 50, c = 'Orange', label = 'Iris-Setosa')
         plt.scatter(x[y_kmeans == 1, 2], x[y_kmeans == 1, 3], s = 50, c = 'Purple', label = 'Iris-Versicolour')
         plt.scatter(x[y_kmeans == 2, 2], x[y_kmeans == 2, 3], s = 50, c = 'Green', label = 'Iris-Virginica')

         #PLOTTING THE CENTROIDS OF THE CLUSTERS
         plt.scatter(kmeans.cluster_centers_[:, 2], kmeans.cluster_centers_[:,3], s = 60, c = 'Red', label = 'Centroids')

         plt.legend(loc=1, bbox_to_anchor=(1.4, 1))
         plt.title("SPECIES VS PETAL LENGTH AND PETAL WIDTH", fontsize=14)
         plt.xlabel("PETAL LENGTH", fontsize=12)
         plt.ylabel("PETAL WIDTH", fontsize=12)
         plt.grid()
         plt.show()
```



VISUALISING THE CLUSTERS - LINE PLOT

```
In [17]: #VISUALISING EACH SPECIES WITH LINE PLOT
         fig, (ax1,ax2) = plt.subplots(2)
         fig, (ax3,ax4) = plt.subplots(2)

         #SUBPLOT 1:
         data.plot(kind='line', color='Red',x='Species', y='Sepal_Length', figsize=(18,9), ax=ax1)
         ax1.set_xlabel('SPECIES', fontsize=12)
         ax1.set_ylabel('SEPAL LENGTH', fontsize=12)

         #SUBPLOT 2:
         data.plot(kind='line', color='Blue',x='Species', y='Sepal_Width', figsize=(18,9), ax=ax2)
         ax2.set_xlabel('SPECIES', fontsize=12)
         ax2.set_ylabel('SEPAL WIDTH', fontsize=12)

         #SUBPLOT 3:
         data.plot(kind='line', color='Green',x='Species', y='Petal_Length', figsize=(18,9), ax=ax3)
         ax3.set_xlabel('SPECIES', fontsize=12)
         ax3.set_ylabel('PETAL LENGTH', fontsize=12)

         #SUBPLOT 4:
         data.plot(kind='line', color='Orange',x='Species', y='Petal_Width', figsize=(18,9), ax=ax4)
         ax4.set_xlabel('SPECIES', fontsize=12)
         ax4.set_ylabel('PETAL WIDTH', fontsize=12)
```

```
Out[17]: Text(0, 0.5, 'PETAL WIDTH')
```



OBSERVATION AND CONCLSION:

1. **AFTER VISUALISING THE IRIS DATASET, WE CAME TO OBSERVE THAT THE SPECIES SETOSA IS VERY DIFFERENT FROM VIRGINICA AND VERSICOLOR.**
2. **SETOSA HAS SMALL SEPAL LENGTH, PETAL LENGTH AND PETAL WIDTH BUT LARGE VALUE FOR SEPAL WIDTH.**
3. **WHEREAS, VIRGINICA AND VERSICOLOR ARE QUITE SIMILAR TO EACH OTHER, BUT NOT EXACTLY. VIRGINICA AS HIGH VALUE FOR PETAL LENGTH, PETAL WIDTH, SEPAL LENGTH AND VERSICOLOR AS MUCH LOWER VALUE THAN VIRGINICA.**

THANK YOU!!