MACHINE LEARNING

Submitted by

- Name: JAGANNATH V V
- Register number: 21122024
- Class : 2MScDS

Lab overview

Illustrate KMeans and Agglomerative Hierarchical Clustering on Iris Dataset, considering only two features - Sepal Length and Petal Width.

Use Elbow Method as a way to find optimum number of clusters

Problem Definition

illustrating kmeans and agglomerative heirarchial clustering

IMPORTING THE NECESSARY LIBRARIES

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
df=pd.read_csv(r"C:\Users\jagan\OneDrive\Desktop\Iris.csv")
df.head()
```

Out[2]:

	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

In [3]:
```
df.tail()
```

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	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

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [5]:
```
df.shape
```

Out[5]:
```
(150, 6)
```

In [6]:
```
df.describe()
```

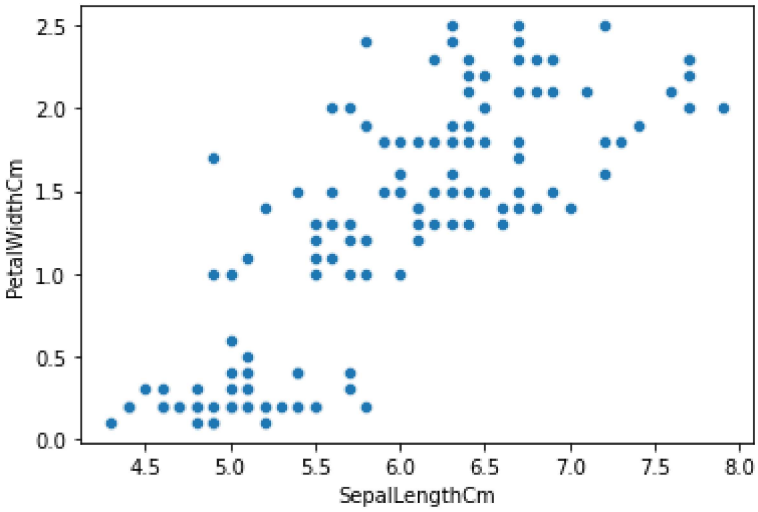
Out[6]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
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

visualization

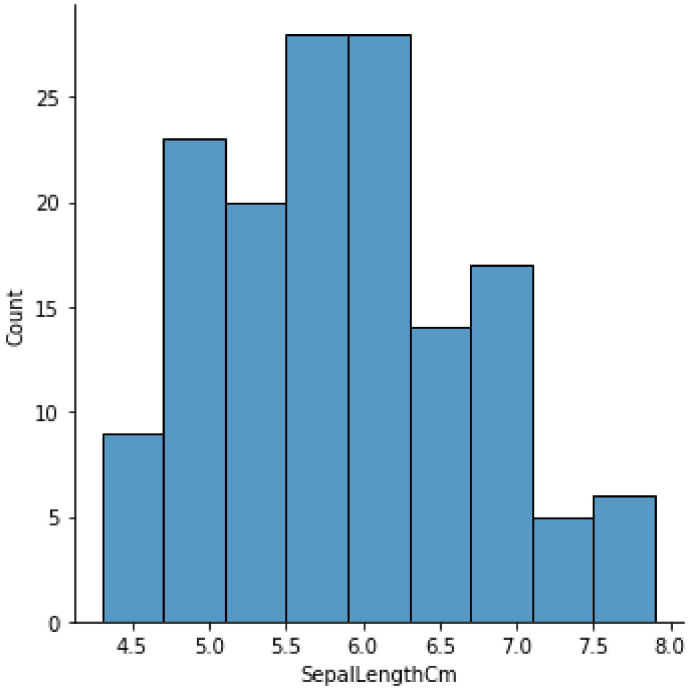
In [38]:
```
sns.scatterplot(data=df, x="SepalLengthCm", y="PetalWidthCm")
```

Out[38]:
```
<AxesSubplot:xlabel='SepalLengthCm', ylabel='PetalWidthCm'>
```

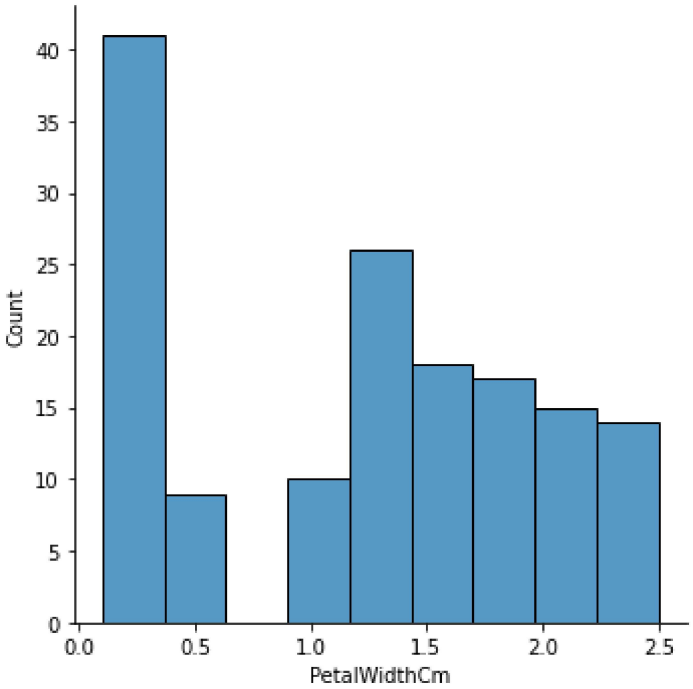


In [ ]:

In [7]:
```
sns_plot1=sns.displot(df["SepalLengthCm"])
```

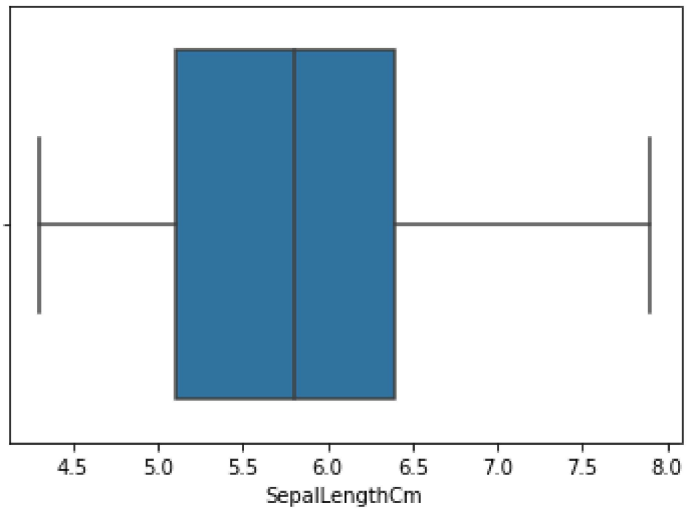


In [8]:
```
sns_plot2=sns.displot(df["PetalWidthCm"])
```



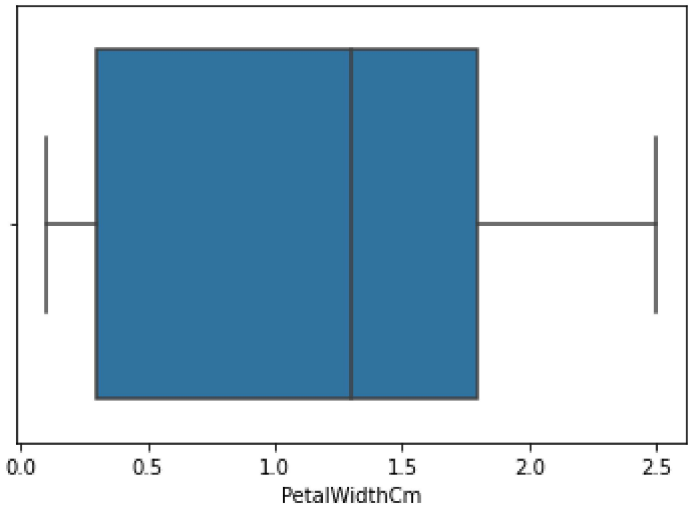
In [9]:
```
sns_plot3=sns.boxplot(df["SepalLengthCm"])
```

```
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid position
al argument will be `data`, and passing other arguments without an explicit keyword will
result in an error or misinterpretation.
  warnings.warn(
```



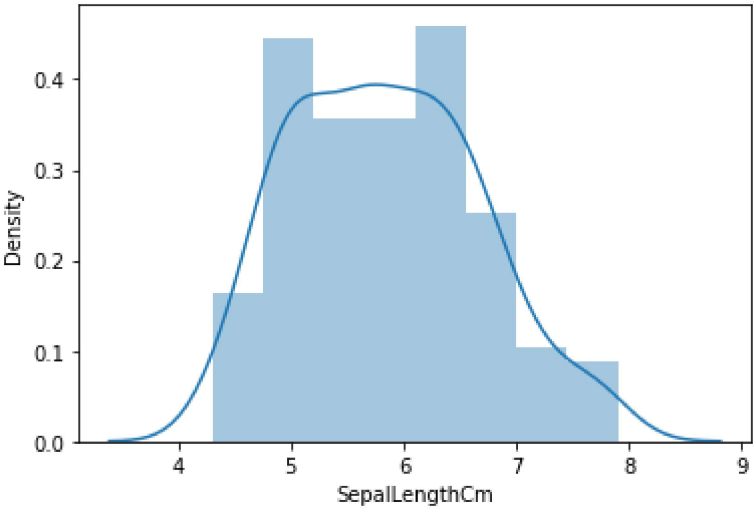
In [10]:
```
sns_plot4=sns.boxplot(df["PetalWidthCm"])
```

```
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid position
al argument will be `data`, and passing other arguments without an explicit keyword will
result in an error or misinterpretation.
  warnings.warn(
```



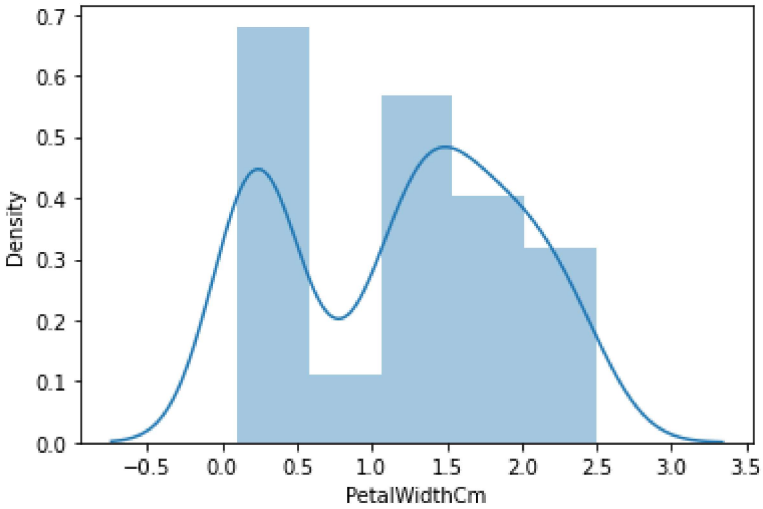
In [11]:
```
sns_plot5=sns.distplot(df["SepalLengthCm"])
```

```
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

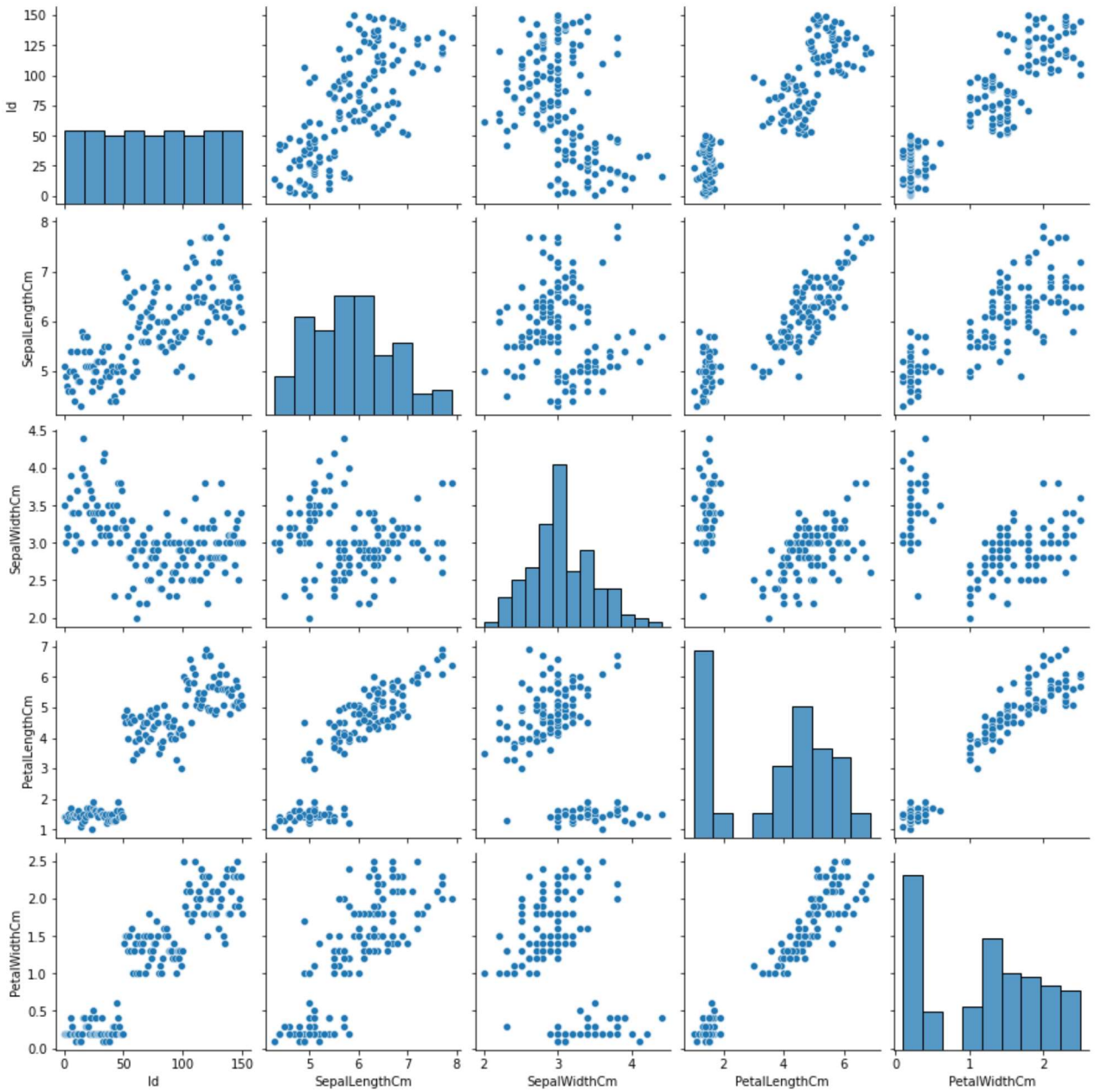


In [12]:
```
sns_plot5=sns.distplot(df["PetalWidthCm"])
```

```
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [13]:
```
sns_plot6=sns.pairplot(df)
```



kmeans clustering

In [14]:
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import MinMaxScaler
```

In [15]:
```
x=df[["SepalLengthCm","PetalWidthCm"]]
x.head()
```

Out[15]:

	SepalLengthCm	PetalWidthCm
0	5.1	0.2
1	4.9	0.2
2	4.7	0.2
3	4.6	0.2
4	5.0	0.2

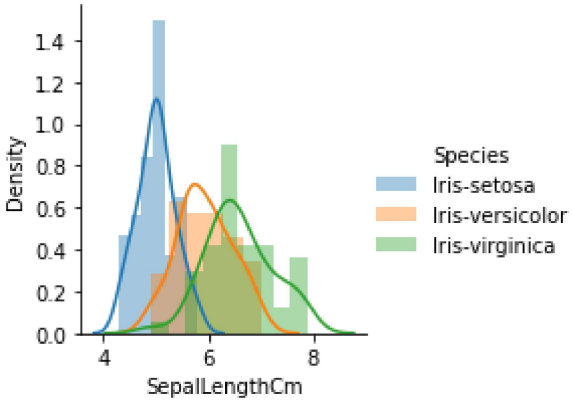
In [16]:
```
x.info()
```

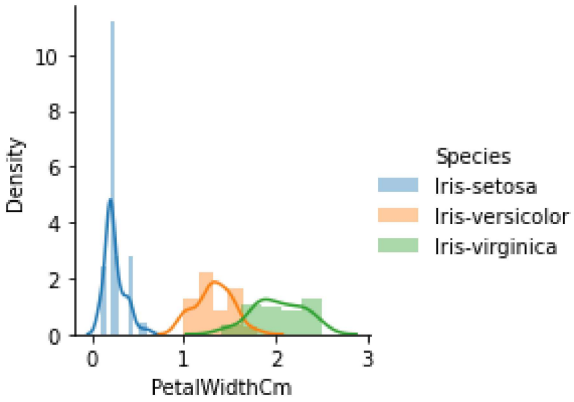
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   PetalWidthCm   150 non-null    float64
dtypes: float64(2)
memory usage: 2.5 KB
```

In [17]:
```
sns.FacetGrid(df,hue="Species",height=3).map(sns.distplot,"SepalLengthCm").add_legend()
sns.FacetGrid(df,hue="Species",height=3).map(sns.distplot,"PetalWidthCm").add_legend()
plt.show()
```

```
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
```

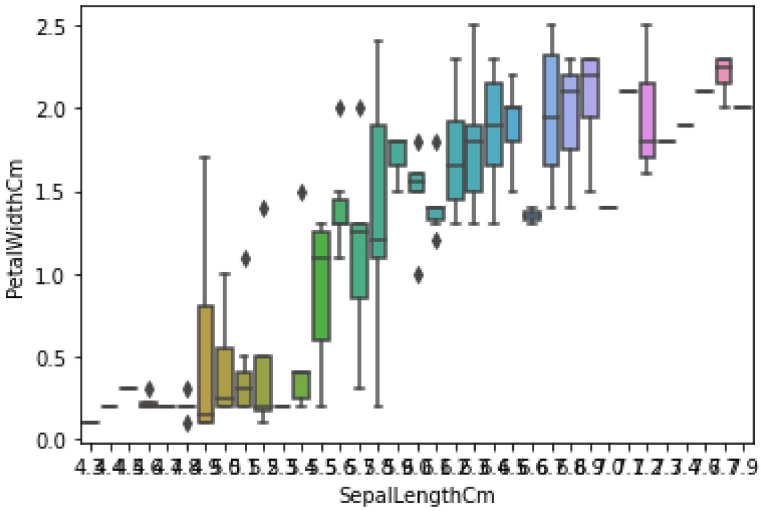
```
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\jagan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





In [18]:

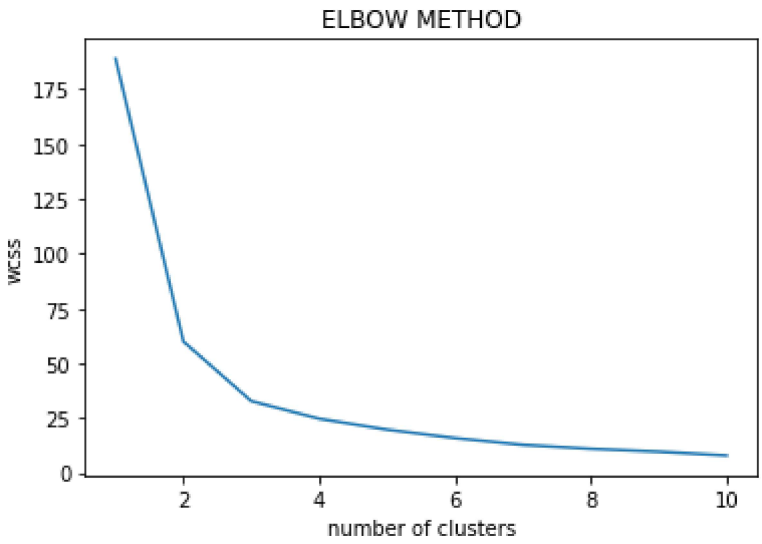
```
sns.boxplot(x="SepalLengthCm",y="PetalWidthCm",data=df)
plt.show()
```



In [19]:
```
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, ra
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

```
C:\Users\jagan\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWarning:
KMeans is known to have a memory leak on Windows with MKL, when there are less chunks th
an available threads. You can avoid it by setting the environment variable OMP_NUM_THREA
DS=1.
  warnings.warn(
```

In [20]:
```
plt.plot(range(1,11),wcss)
plt.title("ELBOW METHOD")
plt.xlabel("number of clusters")
plt.ylabel("wcss")
plt.show()
```



IMPLENTING K MEANS CLUSTERING

In [21]:
```
kmeans=KMeans(n_clusters=3,init='k-means++',max_iter=300,n_init=10,random_state=0)
y_kmeans=kmeans.fit_predict(x)
```

In [22]:
```
y_kmeans
```

Out[22]:
```
array([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, 1, 2, 1, 2, 1, 2, 2, 0, 1, 2, 0, 2, 2, 2, 2, 1,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2,
       2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 0, 2, 1, 2, 1, 2, 1, 1, 2, 1, 1, 1,
       1, 1, 1, 2, 2, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 1, 2, 2, 1, 1, 1, 1,
       1, 2, 2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2])
```

In [23]:
```
df['class']=y_kmeans
df
```

Out[23]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	class
0	1	5.1	3.5	1.4	0.2	Iris-setosa	0
1	2	4.9	3.0	1.4	0.2	Iris-setosa	0
2	3	4.7	3.2	1.3	0.2	Iris-setosa	0
3	4	4.6	3.1	1.5	0.2	Iris-setosa	0
4	5	5.0	3.6	1.4	0.2	Iris-setosa	0
...	...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica	1
146	147	6.3	2.5	5.0	1.9	Iris-virginica	1
147	148	6.5	3.0	5.2	2.0	Iris-virginica	1
148	149	6.2	3.4	5.4	2.3	Iris-virginica	1
149	150	5.9	3.0	5.1	1.8	Iris-virginica	2

150 rows × 7 columns

In [24]:
```
df['class'].value_counts()
```

Out[24]:
```
0    54
2    52
1    44
Name: class, dtype: int64
```

df['class'].value_counts()

In [35]:
```
x2=kmeans.cluster_centers_
x2
```

Out[35]:
```
array([[5.00555556, 0.30185185],
       [6.84318182, 1.98409091],
       [5.86730769, 1.46538462]])
```

In [37]:
```
#Visualising the clusters
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'purple', label = 'I
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'orange', label = 'I
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Ir

#Plotting the centroids of the clusters
plt.scatter(kmeans.cluster_centers_[:, 0], x2[:,1], s = 100, c = 'red', label = 'Centro

plt.legend()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_15788/33352755.py in <module>
      1 #Visualising the clusters
----> 2 plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'purple', la
bel = 'Iris-setosa')
```

```
      3 plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'orange', la
bel = 'Iris-versicolour')
      4 plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', lab
el = 'Iris-virginica')
      5
```

```
~\anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   3456             if self.columns.nlevels > 1:
   3457                 return self._getitem_multilevel(key)
-> 3458             indexer = self.columns.get_loc(key)
   3459             if is_integer(indexer):
   3460                 indexer = [indexer]
```

```
~\anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method,
 tolerance)
   3359             casted_key = self._maybe_cast_indexer(key)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:
   3363                 raise KeyError(key) from err
```

```
~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.g
et_loc()
```

```
~\anaconda3\lib\site-packages\pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.g
et_loc()
```

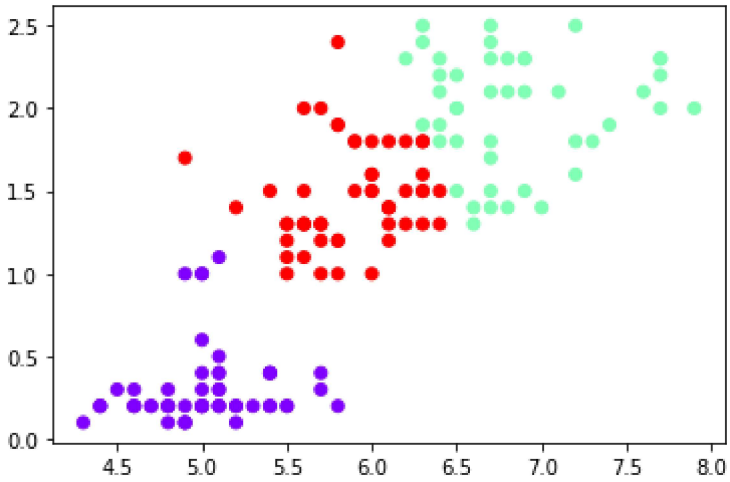
```
TypeError: '(array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True, False, False, False, False,
       False, False, False,  True, False, False,  True, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False,  True, False, False, False, False,  True,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False]), 0)' is an invalid key
```

In [49]:

```
df_1=x.copy()
df_1['Clusters']=y_kmeans
plt.scatter(df_1['SepalLengthCm'],df_1["PetalWidthCm"],c=df_1['Clusters'],cmap='rainbow
```

Out[49]:

```
<matplotlib.collections.PathCollection at 0x1f95d345670>
```



Agdlomerative clustering

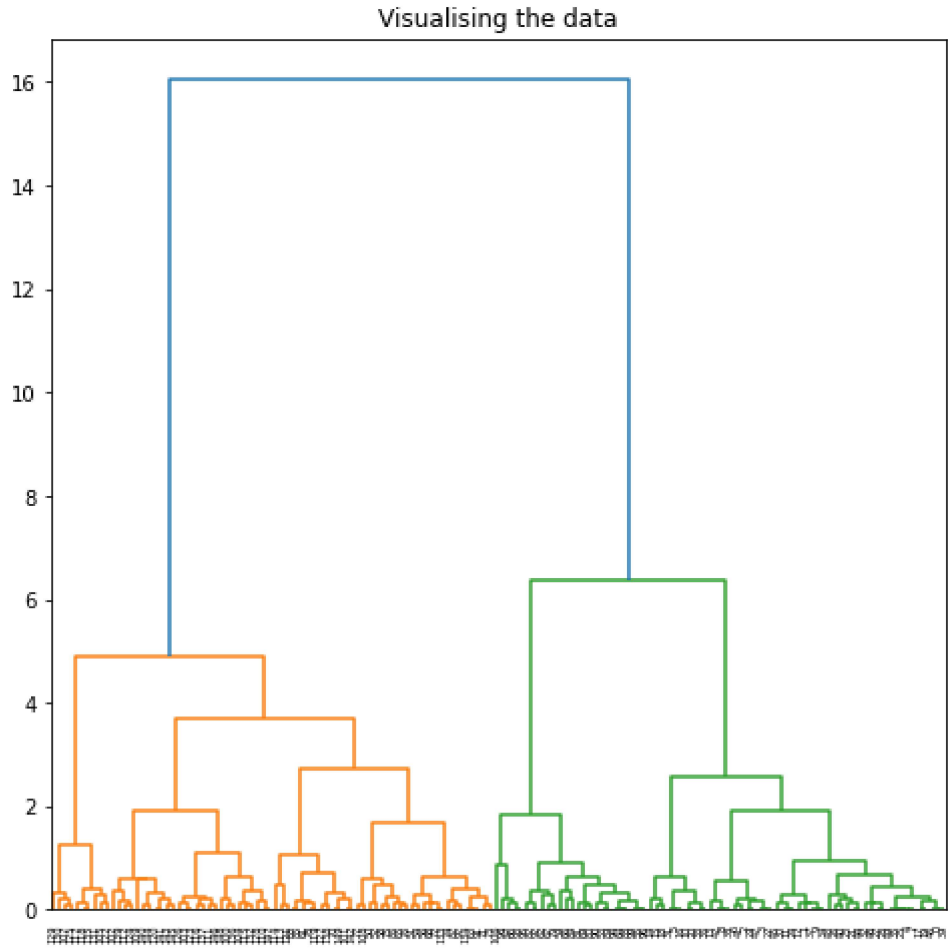
In [50]:
```
from sklearn.cluster import AgglomerativeClustering
```

In [51]:
```
df_scaled=x.apply(zscore)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_15788/3015838970.py in <module>
----> 1 df_scaled=x.apply(zscore)

NameError: name 'zscore' is not defined
```

In [52]:
```
import scipy.cluster.hierarchy as shc
```

In [53]:
```
plt.figure(figsize =(8, 8))
plt.title('Visualising the data')
Dendrogram = shc.dendrogram((shc.linkage(x, method ='ward')))
```

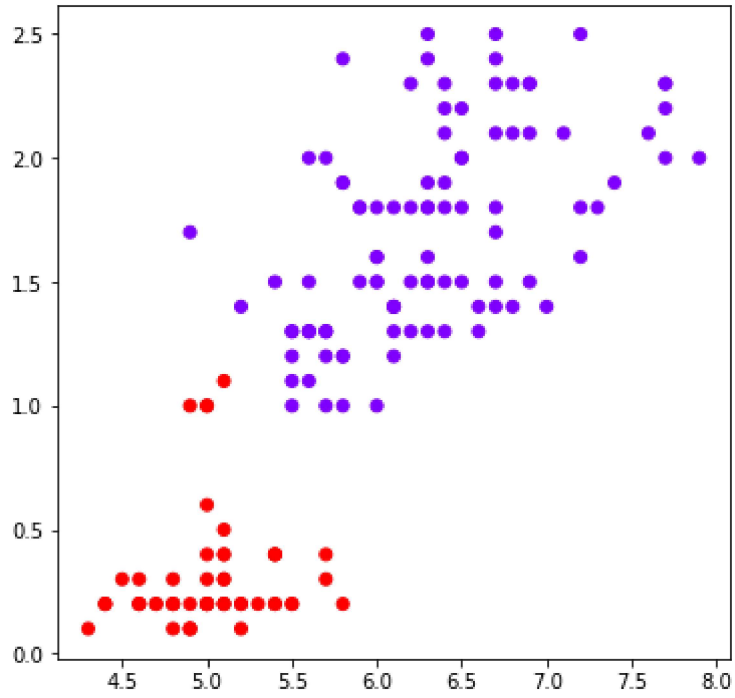


Dendograms are used to divide a given cluster into many different clusters*

k=2

In [58]:

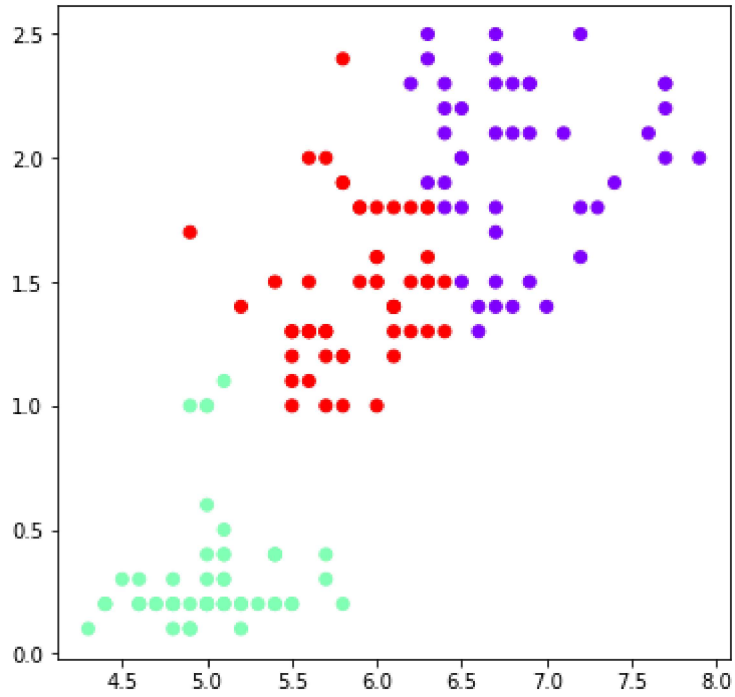
```
agc2 = AgglomerativeClustering(n_clusters = 2)

# Visualizing the clustering
plt.figure(figsize =(6, 6))
plt.scatter(df_1['SepalLengthCm'], df_1['PetalWidthCm'],
            c = agc2.fit_predict(df_1), cmap ='rainbow')
plt.show()
```



k=4

In [59]:

```
agc3= AgglomerativeClustering(n_clusters = 3)
plt.figure(figsize =(6, 6))
plt.scatter(df_1['SepalLengthCm'], df_1['PetalWidthCm'],
           c = agc3.fit_predict(df_1), cmap ='rainbow')
plt.show()
```

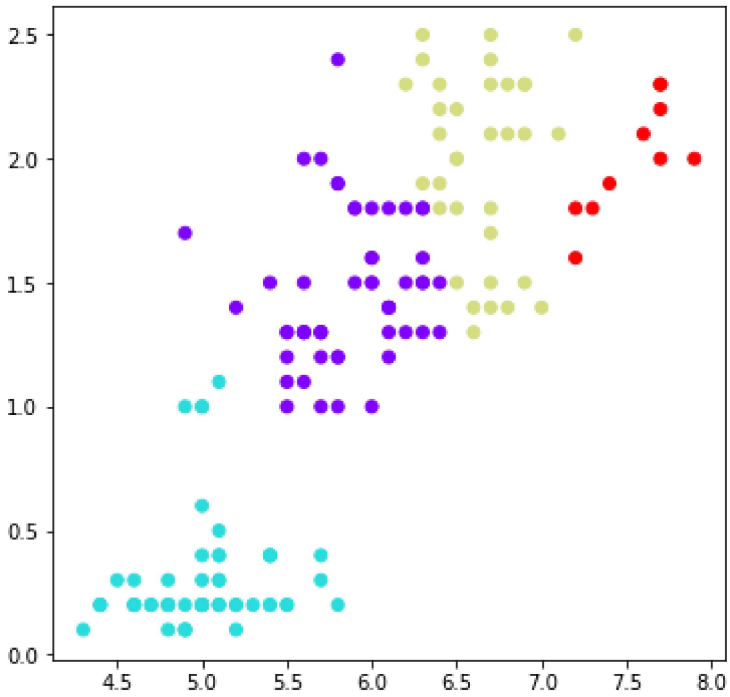


k=4

In [60]:

```
agc4= AgglomerativeClustering(n_clusters = 4)

# Visualizing the clustering
```

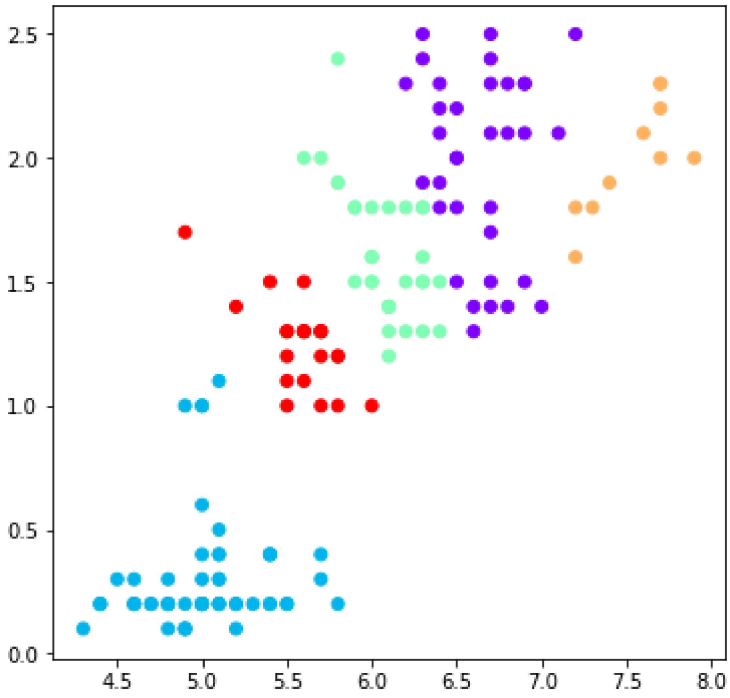
```
plt.figure(figsize =(6, 6))
plt.scatter(df_1['SepalLengthCm'], df_1['PetalWidthCm'],
            c = agc4.fit_predict(df_1), cmap ='rainbow')
plt.show()
```



k=5

In [61]:

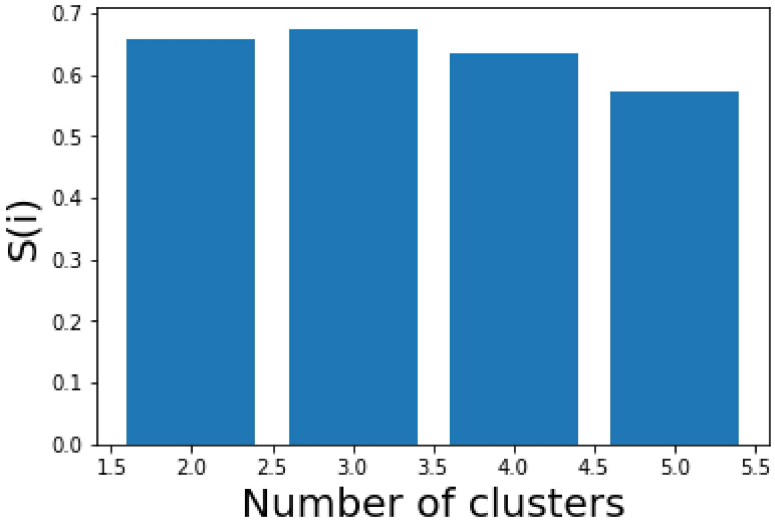
```
agc5= AgglomerativeClustering(n_clusters = 5)

# Visualizing the clustering
plt.figure(figsize =(6, 6))
plt.scatter(df_1['SepalLengthCm'], df_1['PetalWidthCm'],
            c = agc5.fit_predict(df_1), cmap ='rainbow')
plt.show()
```



Evaluating the different models and Visualizing the results.

In [63]:
```
k = [2, 3, 4, 5]

# Appending the silhouette scores of the different models to the list
silhouette_scores = []
silhouette_scores.append(
        silhouette_score(df_1, agc2.fit_predict(df_1)))
silhouette_scores.append(
        silhouette_score(df_1, agc3.fit_predict(df_1)))
silhouette_scores.append(
        silhouette_score(df_1, agc4.fit_predict(df_1)))
silhouette_scores.append(
        silhouette_score(df_1, agc5.fit_predict(df_1)))
```

In [64]:
```
# Plotting a bar graph to compare the results
plt.bar(k, silhouette_scores)
plt.xlabel('Number of clusters', fontsize = 20)
plt.ylabel('S(i)', fontsize = 20)
plt.show()
```



CONCLUSION

here i tried to explore more about the clusterig mainly the kmean clustering and Aggloomerative clustering and also with the help of the silhouette scores, it can concluded that the optimal number of clusters for the given data and clustering technique is 3.

REFERENCES

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans

https://www.analyticsvidhya.com/blog/2021/04/k-means-clustering-simplified-in-python/

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html#sklearn.cluster.Agglo

https://www.geeksforgeeks.org/implementing-agglomerative-clustering-using-sklearn/

https://stackabuse.com/hierarchical-clustering-with-python-and-scikit-learn/

In [ ]: