```
#import libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd


#dataset with elements
dataset = [[1,2,10],[2,2,5],[3,8,4],[4,5,8],[5,7,5],[6,6,4],[7,1,2],[8,4,9]]
print(dataset)
```

```
[[1, 2, 10], [2, 2, 5], [3, 8, 4], [4, 5, 8], [5, 7, 5], [6, 6, 4], [7, 1,
```

```
#DataFrame Creation
dataset = pd .DataFrame(dataset, columns=['Sr.NO','x1','x2'])
dataset.head()
```

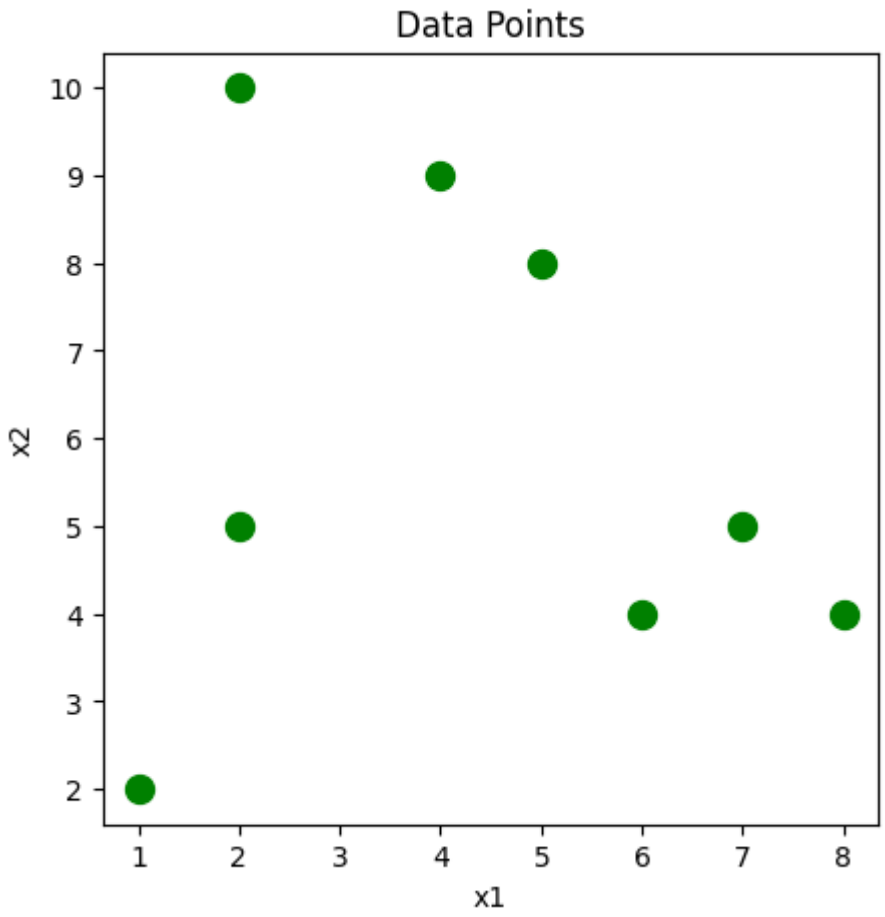
	Sr.NO	x1	x2
0	1	2	10
1	2	2	5
2	3	8	4
3	4	5	8
4	5	7	5

Next steps: Generate code with dataset | View recommended plots | New interactive sheet

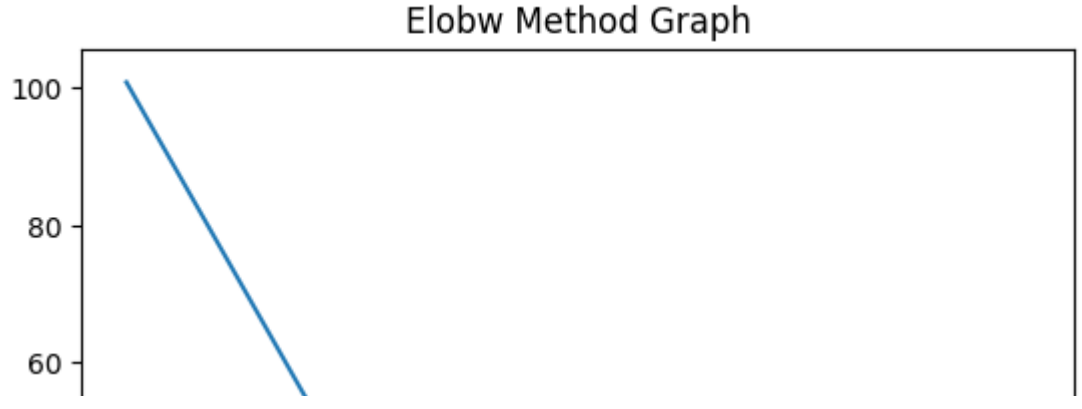
```
#Data Extraction
x = dataset.iloc[:, [1, 2]].values
#print values
x
```

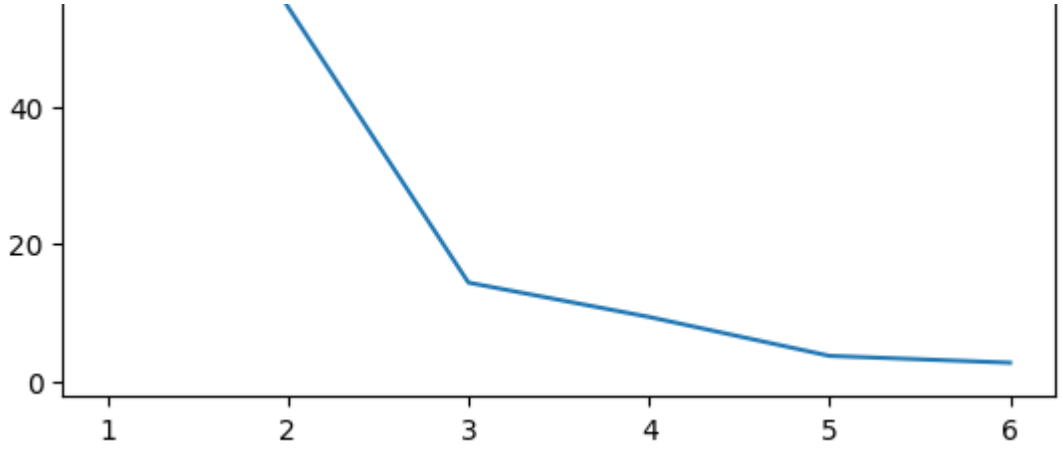
```
array([[ 2, 10],
       [ 2,  5],
       [ 8,  4],
       [ 5,  8],
       [ 7,  5],
       [ 6,  4],
       [ 1,  2],
       [ 4,  9]])
```

```
#data points plotting
plt.figure(figsize=(5,5))
plt.scatter(dataset.iloc[:,1], dataset.iloc[:,2], color="Green", s=100)

plt.xlabel("x1")
plt.ylabel("x2")
plt.title("Data Points")
plt.show()
```




```
#elbow method for optimal number of clusters in k means clustering

from sklearn.cluster import KMeans
wcss_list= [ ]


for i in range(1,7):
  kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
  kmeans.fit(x)
  wcss_list.append(kmeans.inertia_)

  #plotting
plt.plot(range(1, 7), wcss_list)
plt.title('Elobw Method Graph')
plt.show()
```






```
import numpy as np

initialclusters= np.array([[2,10], [5,8],[1,2]])
kmeans = KMeans(n_clusters=3, init=initialclusters, random_state= 42)
kmeans
```

```
KMeans(init=array([[ 2, 10],
       [ 5,  8],
       [ 1,  2]]), n_clusters=3,
       random_state=42)
```

```
#predictions
y_predict=kmeans.fit_predict(x)
```

```
#label to column named label
dataset["Assigned_Label"] = y_predict
print(dataset)
```

```
   Sr.NO  x1  x2  Assigned_Label
0      1   2  10               0
1      2   2   5               2
2      3   8   4               1
3      4   5   8               0
4      5   7   5               1
5      6   6   4               1
6      7   1   2               2
7      8   4   9               0
```

```
#labels and centroids
labels = kmeans.labels_
centroids = kmeans.cluster_centers_
print(" Labels :", labels)
print(" centroids :", centroids)
```

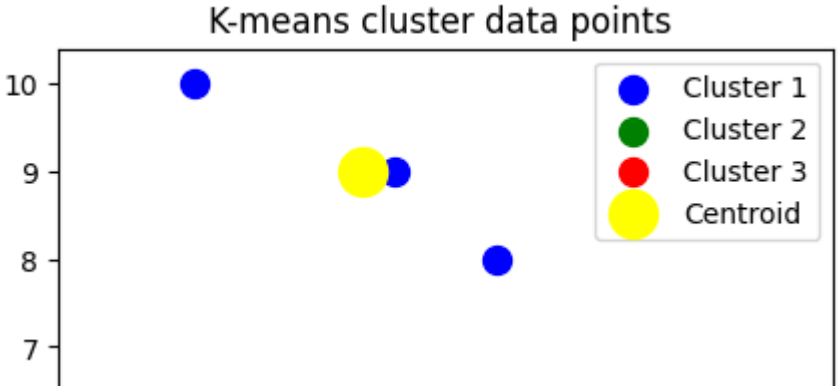
```
 Labels : [0 2 1 0 1 1 2 0]
 centroids : [[3.66666667 9.        ]
 [7.         4.33333333]
```

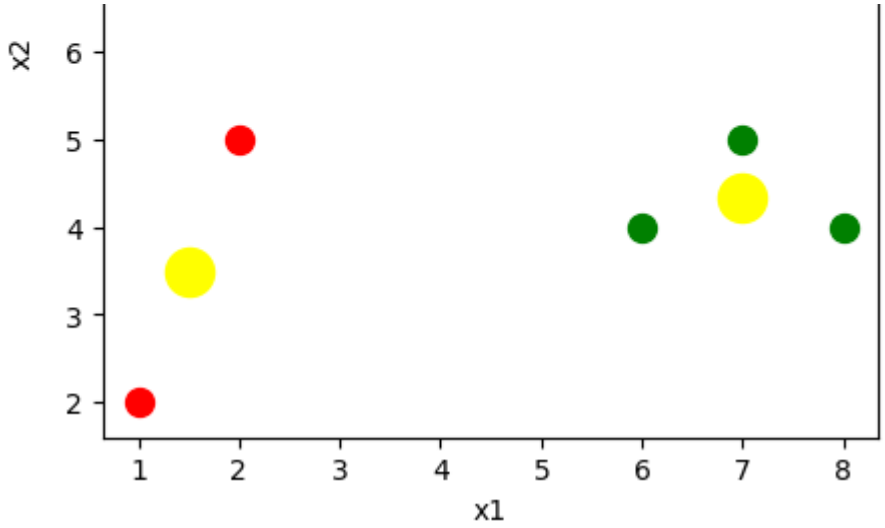
```
[1.5        3.5        ]]

#display y predict
y_predict

#new variable for dataset
df2=dataset
df2
```

	Sr.NO	x1	x2	Assigned_Label
0	1	2	10	0
1	2	2	5	2
2	3	8	4	1
3	4	5	8	0
4	5	7	5	1
5	6	6	4	1
6	7	1	2	2
7	8	4	9	0

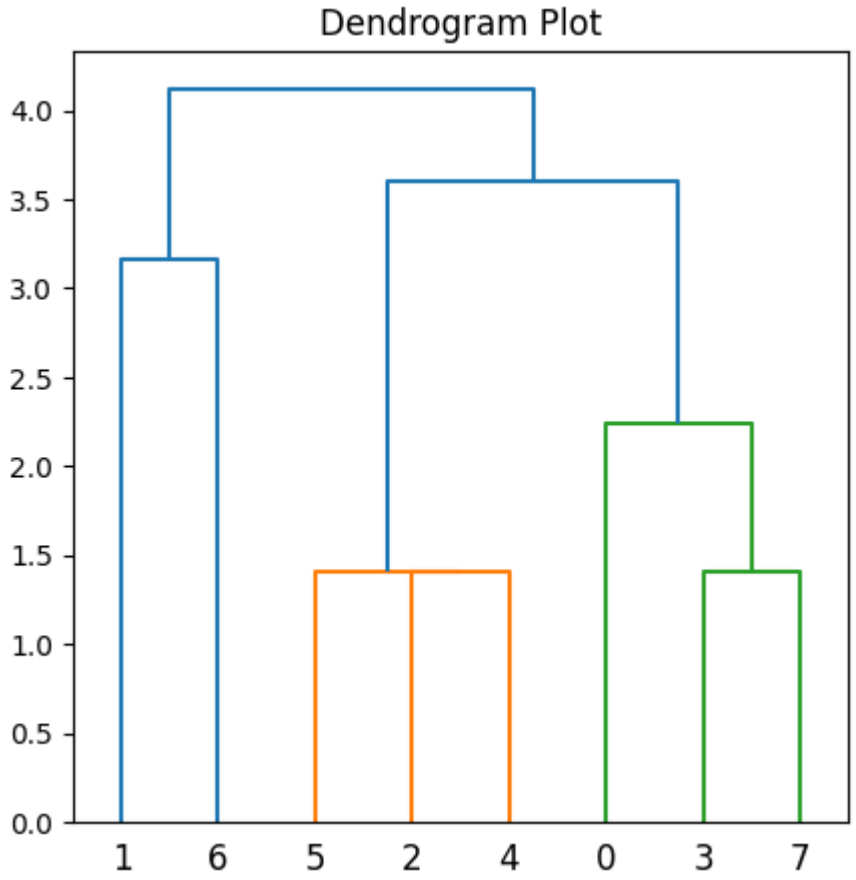
Next steps: Generate code with dataset View recommended plots New interactive sheet

```
import matplotlib.pyplot as plt

# cluster plotting
plt.figure(figsize=(5,5))
plt.scatter(df2.x1[df2.Assigned_Label == 0], df2["x2"][df2.Assigned_Label==0],s
plt.scatter(df2.x1[df2.Assigned_Label == 1], df2["x2"][df2.Assigned_Label==1],s
plt.scatter(df2.x1[df2.Assigned_Label == 2], df2["x2"][df2.Assigned_Label==2],s
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 3
plt.xlabel("x1")
plt.ylabel("x2")
plt.title("K-means cluster data points ")

plt.legend()
plt.show()
```



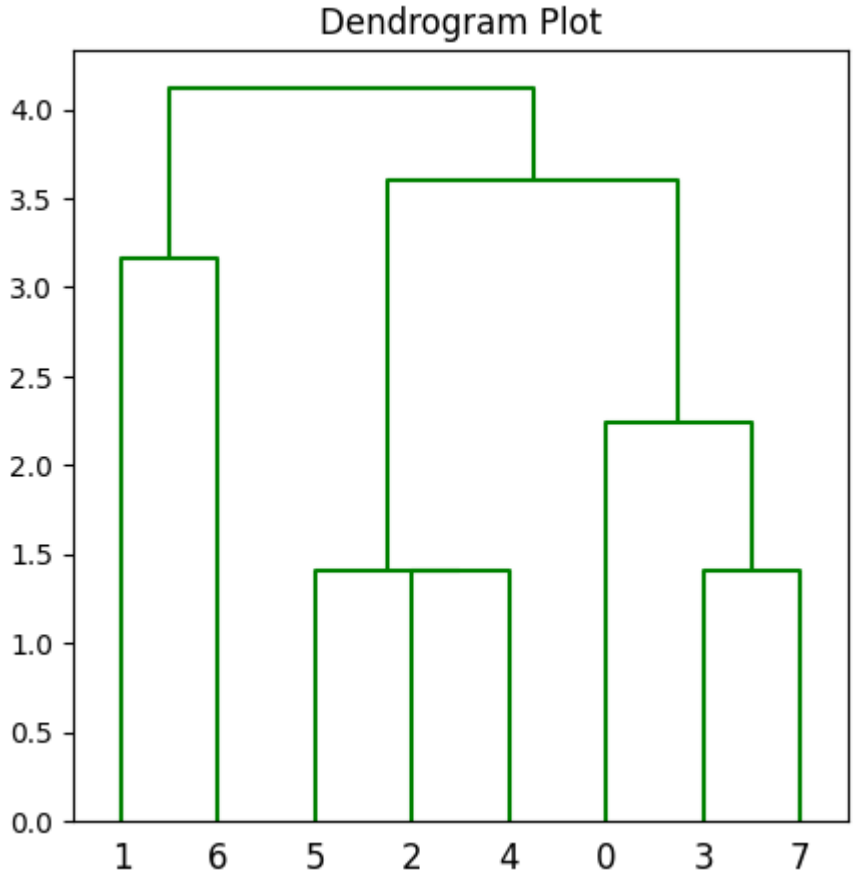



```
#hierarchical clustering
import scipy.cluster.hierarchy as shc
plt.figure(figsize =(5,5))
dendro = shc.dendrogram(shc.linkage(x, method="single"))
plt.title("Dendrogram Plot")
plt.show()
```

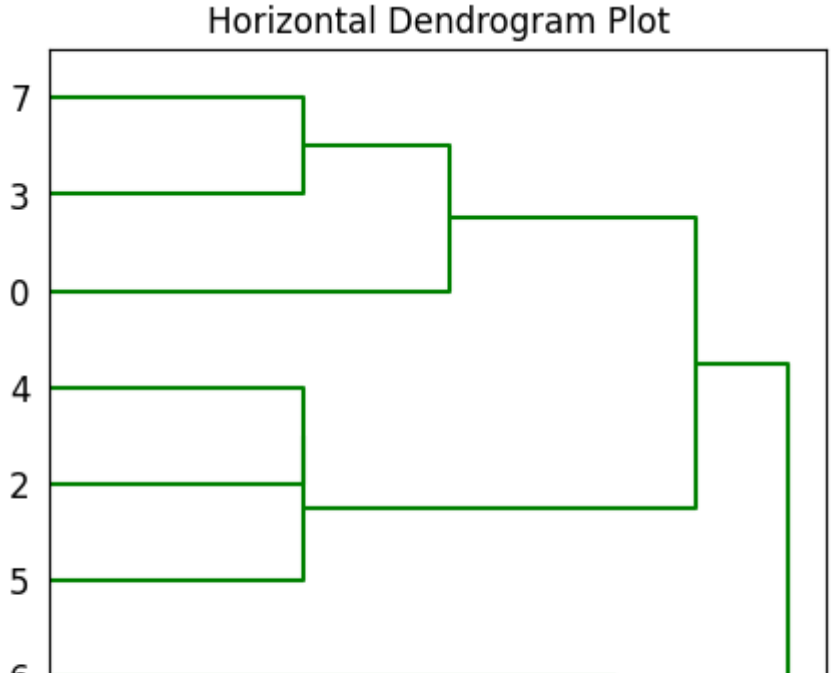


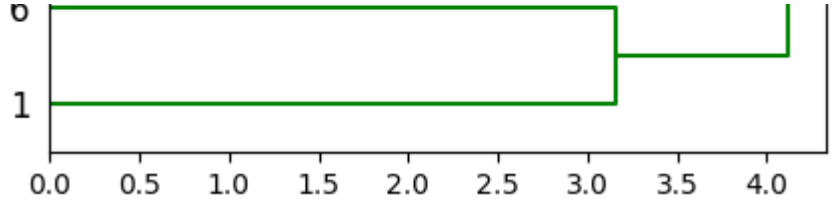

```
#second way of plotting
import scipy.cluster.hierarchy as shc
from scipy.cluster import hierarchy
temp= hierarchy.linkage(x,'single')
#plotting
plt.figure(figsize =(5,5))
```

```
plt.figure(figsize =(5,5))
dendro = hierarchy.dendrogram(temp, above_threshold_color="green", color_thresh
plt.title("Dendrogram Plot")
plt.show()
```




```
#dendrogram in horizonal orientation
plt.figure(figsize =(5,5))
dendro = hierarchy.dendrogram(temp, above_threshold_color="green", color_thresh
plt.title("Horizontal Dendrogram Plot")
plt.show()
```





Start coding or generate with AI.