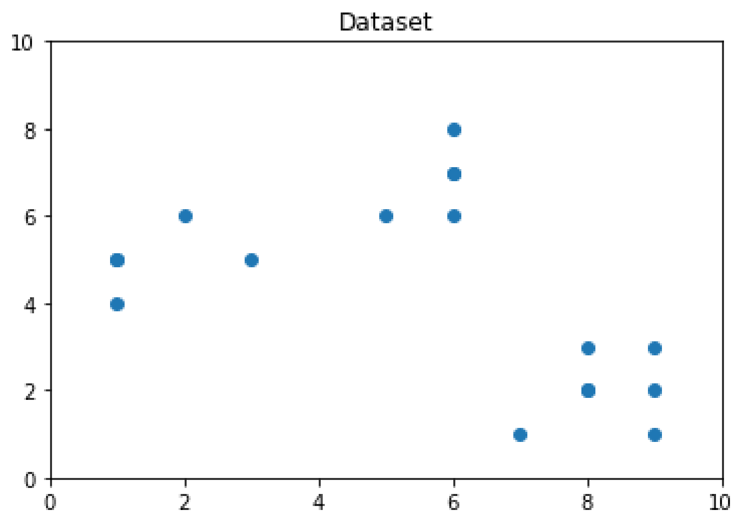
```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt
```

```
#Creating the data
x1 = np.array([3, 1, 1, 2, 1, 6, 6, 6, 5, 6, 7, 8, 9, 8, 9, 9, 8])
x2 = np.array([5, 4, 5, 6, 5, 8, 6, 7, 6, 7, 1, 2, 1, 2, 3, 2, 3])
x_new = np.array(list(zip(x1, x2))).reshape(len(x1), 2)
x_new
```

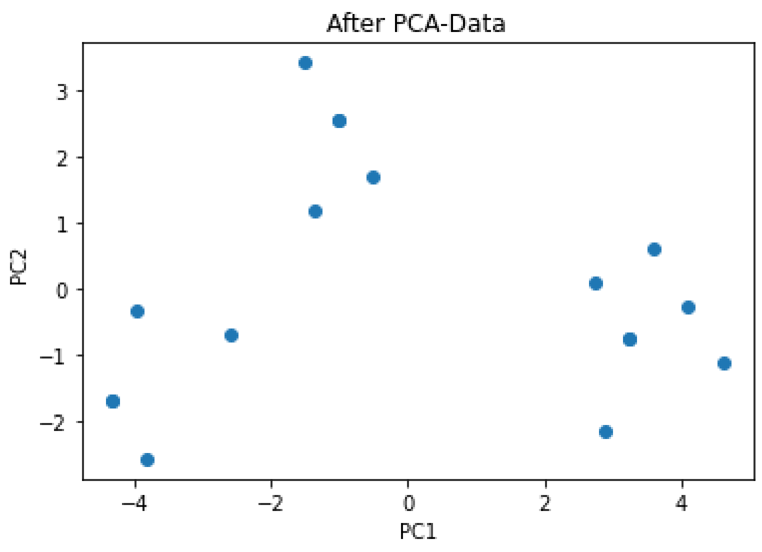
```
    array([[3, 5],
           [1, 4],
           [1, 5],
           [2, 6],
           [1, 5],
           [6, 8],
           [6, 6],
           [6, 7],
           [5, 6],
           [6, 7],
           [7, 1],
           [8, 2],
           [9, 1],
           [8, 2],
           [9, 3],
           [9, 2],
           [8, 3]])
```

```
#Visualizing the data
plt.plot()
plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title('Dataset')
plt.scatter(x1, x2)
plt.show()
```



Applying PCA to Data

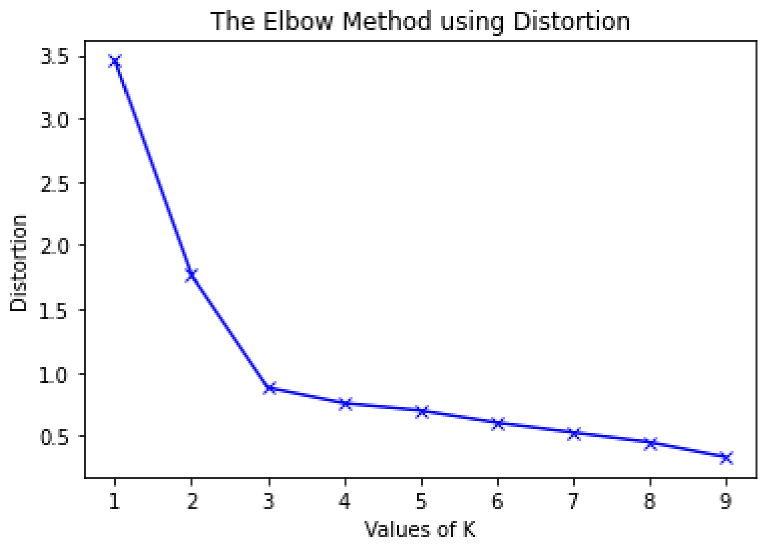
```
from sklearn.decomposition import PCA
pca = PCA()
X = pca.fit_transform(x_new)
plt.scatter(X[:,0], X[:,1])
plt.title('After PCA-Data')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.show()
```



Building the clustering model and calculating the values of the Distortion and inertia

```
distortions = []
inertias = []
mapping1 = {}
mapping2 = {}
K = range(1,10)
for k in K:
 #Building and fitting the model
 kmeanModel = KMeans(n_clusters=k)
 kmeanModel.fit(X)
 distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_,'euclidean'),axis=1))
 inertias.append(kmeanModel.inertia_)
 mapping1[k] = sum(np.min(cdist(X, kmeanModel.cluster_centers_, 'euclidean'),axis=1)) / X.
 mapping2[k] = kmeanModel.inertia_
# Tabulating and Visualizing the results
#a)Using the different values of Distortion
for key,val in mapping1.items():
 print(str(key)+' : '+str(val))
plt.plot(K, distortions, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Distortion')
plt.title('The Elbow Method using Distortion')
plt.show()
```

```
1 : 3.4577032384495703
2 : 1.7687413573405677
3 : 0.8819889697423958
4 : 0.7587138847606587
5 : 0.7004644881908969
6 : 0.6046508654312025
7 : 0.5274410771884641
8 : 0.4505481782959177
9 : 0.3333333333333334
```

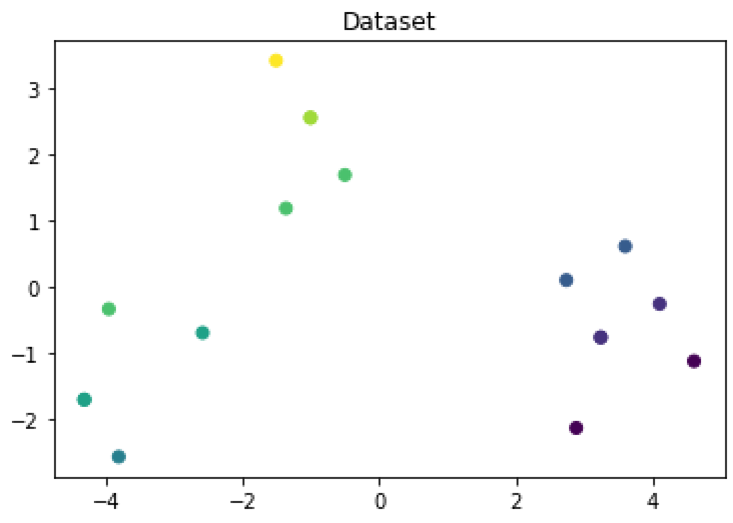


```
#b) Using the different values of Inertia
for key,val in mapping2.items():
 print(str(key)+' : '+str(val))
plt.plot(K, inertias, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Inertia')
plt.title('The Elbow Method using Inertia')
plt.show()
```

```
1 : 217.64705882352936
2 : 68.42857142857142
3 : 16.228571428571424
```

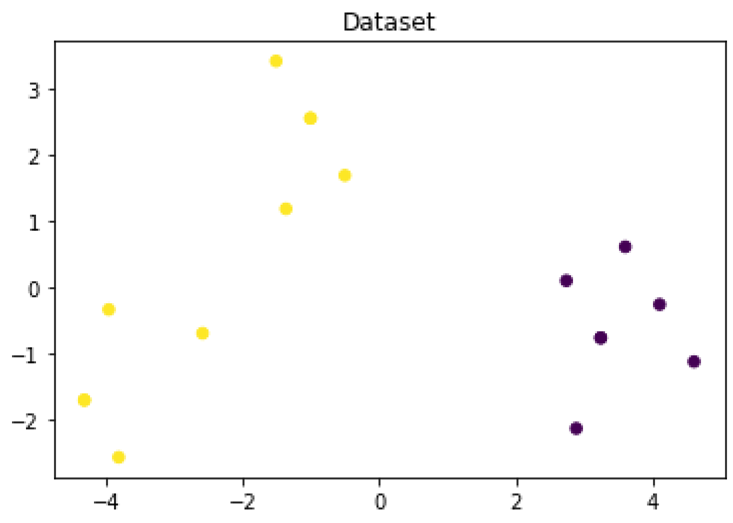
Visualizing the data for different K-value

```
plt.title('Dataset')
plt.scatter(X[:,0], X[:,1], c=x2)
plt.show()
```



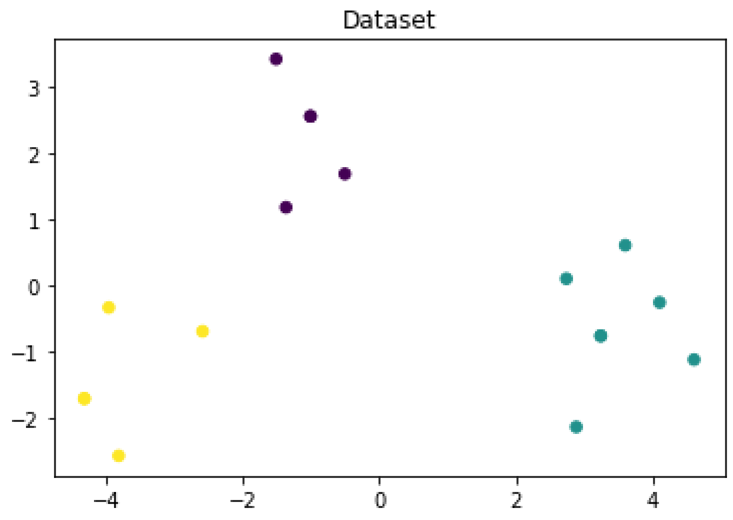
```
from sklearn.cluster import KMeans, MeanShift
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
plt.title('Dataset')
plt.scatter(X[:,0], X[:,1], s=30, label='lab', c=kmeans.predict(X))
#plt.scatter(x1, x2, s=30, c=kmeans.predict(X))
```

```
<matplotlib.collections.PathCollection at 0x7f3e7e2a4690>
```



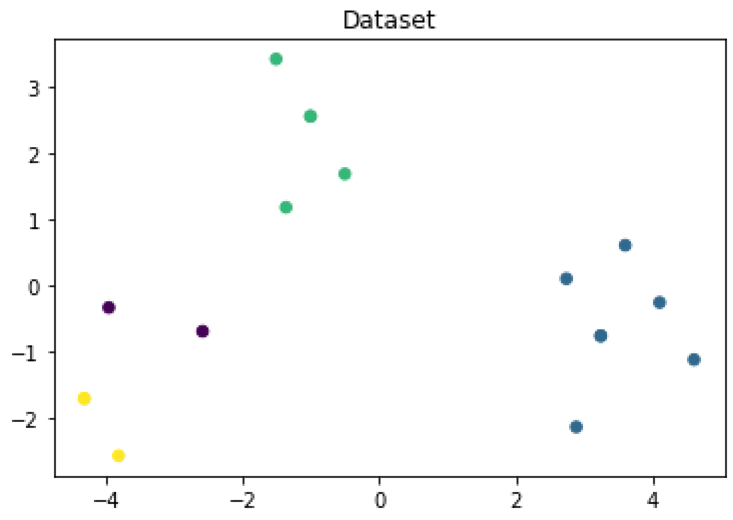
```
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
plt.title('Dataset')
plt.scatter(X[:,0], X[:,1], s=30, label='lab', c=kmeans.predict(X))
```

```
<matplotlib.collections.PathCollection at 0x7f3e7e20f7d0>
```



```
kmeans = KMeans(n_clusters=4)
kmeans.fit(X)
plt.title('Dataset')
plt.scatter(X[:,0], X[:,1], s=30, label='lab', c=kmeans.predict(X))
```

```
<matplotlib.collections.PathCollection at 0x7f3e7e306310>
```


Dataset