Importing necessary liabraries

In [11]:

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
from sklearn.preprocessing import StandardScaler
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
import scipy.cluster.hierarchy as sch
from sklearn.cluster import DBSCAN
from sklearn.cluster import AgglomerativeClustering
import seaborn as sns
from sklearn.preprocessing import normalize
```

Importing Dataset

In [3]:

```
crime_data=pd.read_csv("crime_data.csv")
crime_data.head(20)
```

Out[3]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6
5	Colorado	7.9	204	78	38.7
6	Connecticut	3.3	110	77	11.1
7	Delaware	5.9	238	72	15.8
8	Florida	15.4	335	80	31.9
9	Georgia	17.4	211	60	25.8
10	Hawaii	5.3	46	83	20.2
11	Idaho	2.6	120	54	14.2
12	Illinois	10.4	249	83	24.0
13	Indiana	7.2	113	65	21.0
14	Iowa	2.2	56	57	11.3
15	Kansas	6.0	115	66	18.0
16	Kentucky	9.7	109	52	16.3
17	Louisiana	15.4	249	66	22.2
18	Maine	2.1	83	51	7.8
19	Maryland	11.3	300	67	27.8

Initial investigation

In [4]:

```
del crime_data['Unnamed: 0']
```

In [6]:

```
crime_data.head()
```

Out[6]:

	Murder	Assault	UrbanPop	Rape
0	13.2	236	58	21.2
1	10.0	263	48	44.5
2	8.1	294	80	31.0
3	8.8	190	50	19.5
4	9.0	276	91	40.6

In [7]:

```
crime_data.shape
```

Out[7]:

```
(50, 4)
```

In [8]:

```
crime_data.isna().sum()
```

Out[8]:

```
Murder      0
Assault     0
UrbanPop    0
Rape        0
dtype: int64
```

In [9]:

```
crime_data.dtypes
```

Out[9]:

```
Murder      float64
Assault       int64
UrbanPop      int64
Rape        float64
dtype: object
```

In [12]:

```
# Normalization function
crime_norm=pd.DataFrame(normalize(crime_data),columns=crime_data.columns)
crime_norm
```

Out[12]:

	Murder	Assault	UrbanPop	Rape
0	0.054031	0.966016	0.237411	0.086778
1	0.036872	0.969739	0.176987	0.164081
2	0.026439	0.959624	0.261122	0.101185
3	0.044528	0.961392	0.252998	0.098669
4	0.030657	0.940134	0.309972	0.138295
5	0.035594	0.919142	0.351437	0.174367
6	0.024486	0.816202	0.571341	0.082362
7	0.023674	0.954965	0.288897	0.063397
8	0.044478	0.967547	0.231056	0.092134
9	0.078534	0.952332	0.270805	0.116446
10	0.054546	0.473419	0.854213	0.207893
11	0.019640	0.906483	0.407917	0.107267
12	0.039428	0.944007	0.314669	0.090989
13	0.054447	0.854521	0.491539	0.158805
14	0.027251	0.693660	0.706047	0.139971
15	0.044795	0.858568	0.492743	0.134385
16	0.079346	0.891624	0.425362	0.133335
17	0.059457	0.961347	0.254815	0.085710
18	0.021483	0.849097	0.521734	0.079795
19	0.036587	0.971339	0.216932	0.090011
20	0.025527	0.864425	0.493128	0.094565
21	0.045132	0.951126	0.276013	0.130920
22	0.027317	0.728452	0.667747	0.150749
23	0.061041	0.981958	0.166819	0.064832
24	0.046500	0.919676	0.361670	0.145701
25	0.048998	0.890131	0.432816	0.133928
26	0.035662	0.845935	0.514196	0.136842
27	0.045363	0.937005	0.301180	0.171041
28	0.026088	0.708107	0.695684	0.118018
29	0.040364	0.867284	0.485461	0.102547
30	0.038586	0.964660	0.236934	0.108651
31	0.041163	0.941927	0.318920	0.096789
32	0.038166	0.989371	0.132112	0.047267

	Murder	Assault	UrbanPop	Rape
33	0.012626	0.710188	0.694406	0.115208
34	0.050940	0.837376	0.523360	0.149332
35	0.039535	0.904523	0.407335	0.119804
36	0.027987	0.908164	0.382685	0.167353
37	0.048778	0.820702	0.557458	0.115363
38	0.017459	0.893478	0.446739	0.042620
39	0.050641	0.981163	0.168802	0.079126
40	0.038785	0.877767	0.459297	0.130644
41	0.066230	0.943274	0.296027	0.134968
42	0.058203	0.921161	0.366631	0.116864
43	0.021908	0.821558	0.547706	0.156781
44	0.037410	0.816227	0.544152	0.190453
45	0.050082	0.919147	0.371194	0.121964
46	0.024318	0.881521	0.443800	0.159282
47	0.062942	0.894442	0.430657	0.102695
48	0.030455	0.620812	0.773086	0.126505
49	0.039384	0.932482	0.347509	0.090352

In [13]:

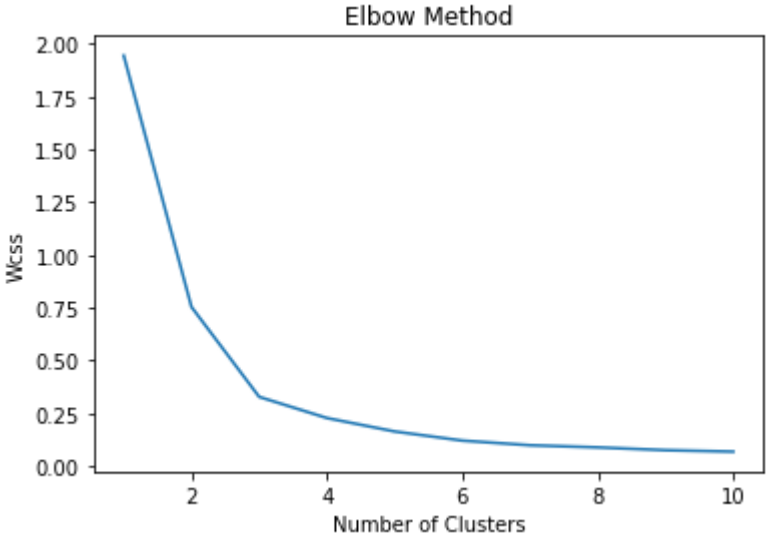
```
# How to find optimum number of  cluster
#The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluste
```

In [14]:

```
wcss=[]
for i in range(1,11):
    kmeans=KMeans( n_clusters=i,random_state=0)
    kmeans.fit(crime_norm)
    wcss.append(kmeans.inertia_)
    print(wcss)

plt.plot(range(1,11),wcss)
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Wcss')
plt.show()
```

```
[1.9452076233849003]
[1.9452076233849003, 0.7540963759591797]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727, 0.12059896172575112]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727, 0.12059896172575112, 0.09861877942595956]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727, 0.12059896172575112, 0.09861877942595956, 0.08846
696716061667]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727, 0.12059896172575112, 0.09861877942595956, 0.08846
696716061667, 0.07526863777639681]
[1.9452076233849003, 0.7540963759591797, 0.3278478050693132, 0.2276024576517
4872, 0.16395152491640727, 0.12059896172575112, 0.09861877942595956, 0.08846
696716061667, 0.07526863777639681, 0.06833415035436594]
```



BuildingK=3

In [15]:

```
#Build Cluster algorithm
# Cluster algorithm using K=3
clusters3=KMeans(3,random_state=19).fit(crime_norm)
clusters3
```

Out[15]:

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=19, tol=0.0001, verbose=0)
```

In [16]:

```
clusters3.labels_
```

Out[16]:

```
array([0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 1, 2, 0, 2, 1, 2, 2, 0, 2, 0, 2, 0,
       1, 0, 0, 2, 2, 0, 1, 2, 0, 0, 0, 1, 2, 2, 2, 2, 2, 0, 2, 0, 0, 2,
       2, 0, 2, 2, 1, 0])
```

In [20]:

```
# Assign clusters to the data set
crime_data3=crime_data.copy()
crime_data3['Clusters3']=clusters3.labels_
crime_data3
```

Out[20]:

	Murder	Assault	UrbanPop	Rape	Clusters3
0	13.2	236	58	21.2	0
1	10.0	263	48	44.5	0
2	8.1	294	80	31.0	0
3	8.8	190	50	19.5	0
4	9.0	276	91	40.6	0
5	7.9	204	78	38.7	0
6	3.3	110	77	11.1	2
7	5.9	238	72	15.8	0
8	15.4	335	80	31.9	0
9	17.4	211	60	25.8	0
10	5.3	46	83	20.2	1
11	2.6	120	54	14.2	2
12	10.4	249	83	24.0	0
13	7.2	113	65	21.0	2
14	2.2	56	57	11.3	1
15	6.0	115	66	18.0	2
16	9.7	109	52	16.3	2
17	15.4	249	66	22.2	0
18	2.1	83	51	7.8	2
19	11.3	300	67	27.8	0
20	4.4	149	85	16.3	2
21	12.1	255	74	35.1	0
22	2.7	72	66	14.9	1
23	16.1	259	44	17.1	0
24	9.0	178	70	28.2	0
25	6.0	109	53	16.4	2
26	4.3	102	62	16.5	2
27	12.2	252	81	46.0	0
28	2.1	57	56	9.5	1
29	7.4	159	89	18.8	2
30	11.4	285	70	32.1	0
31	11.1	254	86	26.1	0

	Murder	Assault	UrbanPop	Rape	Clusters3
32	13.0	337	45	16.1	0
33	0.8	45	44	7.3	1
34	7.3	120	75	21.4	2
35	6.6	151	68	20.0	2
36	4.9	159	67	29.3	2
37	6.3	106	72	14.9	2
38	3.4	174	87	8.3	2
39	14.4	279	48	22.5	0
40	3.8	86	45	12.8	2
41	13.2	188	59	26.9	0
42	12.7	201	80	25.5	0
43	3.2	120	80	22.9	2
44	2.2	48	32	11.2	2
45	8.5	156	63	20.7	0
46	4.0	145	73	26.2	2
47	5.7	81	39	9.3	2
48	2.6	53	66	10.8	1
49	6.8	161	60	15.6	0

In [21]:

```
#these are standardized values.
clusters3.cluster_centers_
```

Out[21]:

```
array([[0.04544868, 0.95210555, 0.27128802, 0.10877231],
       [0.02971377, 0.65577288, 0.73186384, 0.14305726],
       [0.03824062, 0.86500142, 0.47897132, 0.12490787]])
```

In [22]:

```
crime_data3.groupby('Clusters3').agg(['mean']).reset_index()
```

Out[22]:

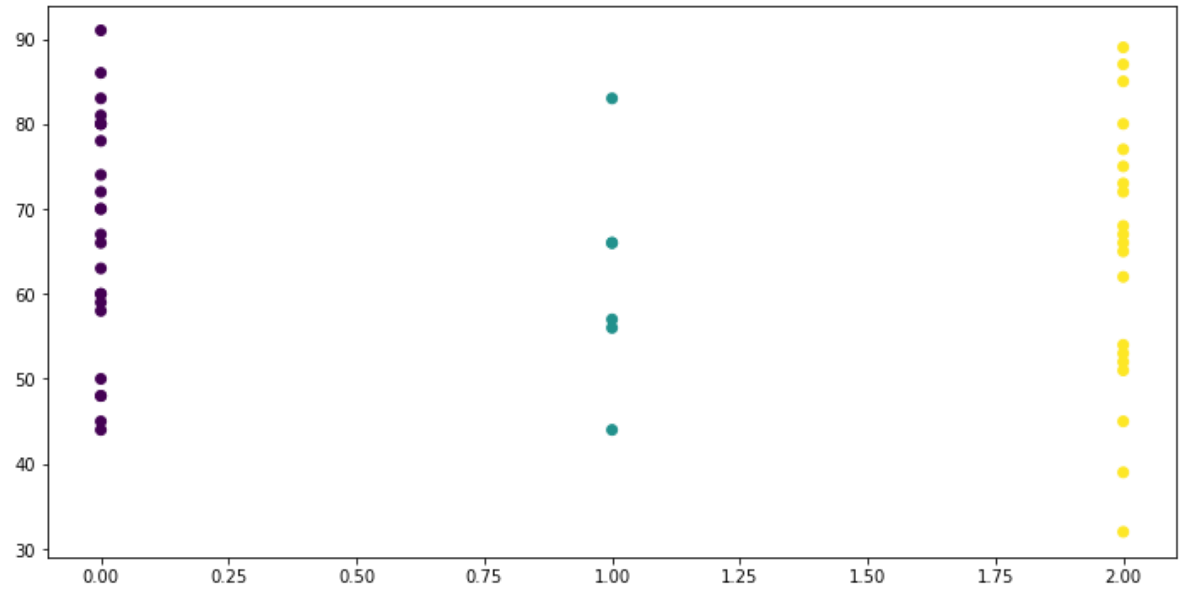
	Clusters3	Murder	Assault	UrbanPop	Rape
		mean	mean	mean	mean
0	0	11.387500	243.750000	67.208333	27.287500
1	1	2.616667	54.833333	62.000000	12.333333
2	2	5.020000	117.950000	64.600000	16.635000

In [26]:

```
# Plot Clusters
plt.figure(figsize=(12, 6))
plt.scatter(crime_data3['Clusters3'],crime_data3['UrbanPop'], c=clusters3.labels_)
```

Out[26]:

```
<matplotlib.collections.PathCollection at 0x23821e115c8>
```



Building K=5

In [27]:

```
# Cluster algorithm using K=5
clusters5=KMeans(5,random_state=21).fit(crime_norm)
clusters5
```

Out[27]:

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=21, tol=0.0001, verbose=0)
```

In [28]:

```
clusters5.labels_
```

Out[28]:

```
array([0, 0, 0, 0, 0, 4, 1, 0, 0, 0, 2, 4, 0, 1, 3, 1, 4, 0, 1, 0, 1, 0,
       3, 0, 4, 4, 1, 0, 3, 1, 0, 0, 0, 3, 1, 4, 4, 1, 4, 0, 4, 0, 4, 1,
       1, 4, 4, 4, 3, 4])
```

In [29]:

```
# Assign clusters to the data set
crime_data5=crime_data.copy()
crime_data5['clusters5']=clusters5.labels_
crime_data5
```

Out[29]:

	Murder	Assault	UrbanPop	Rape	clusters5
0	13.2	236	58	21.2	0
1	10.0	263	48	44.5	0
2	8.1	294	80	31.0	0
3	8.8	190	50	19.5	0
4	9.0	276	91	40.6	0
5	7.9	204	78	38.7	4
6	3.3	110	77	11.1	1
7	5.9	238	72	15.8	0
8	15.4	335	80	31.9	0
9	17.4	211	60	25.8	0
10	5.3	46	83	20.2	2
11	2.6	120	54	14.2	4
12	10.4	249	83	24.0	0
13	7.2	113	65	21.0	1
14	2.2	56	57	11.3	3
15	6.0	115	66	18.0	1
16	9.7	109	52	16.3	4
17	15.4	249	66	22.2	0
18	2.1	83	51	7.8	1
19	11.3	300	67	27.8	0
20	4.4	149	85	16.3	1
21	12.1	255	74	35.1	0
22	2.7	72	66	14.9	3
23	16.1	259	44	17.1	0
24	9.0	178	70	28.2	4
25	6.0	109	53	16.4	4
26	4.3	102	62	16.5	1
27	12.2	252	81	46.0	0
28	2.1	57	56	9.5	3
29	7.4	159	89	18.8	1
30	11.4	285	70	32.1	0

	Murder	Assault	UrbanPop	Rape	clusters5
31	11.1	254	86	26.1	0
32	13.0	337	45	16.1	0
33	0.8	45	44	7.3	3
34	7.3	120	75	21.4	1
35	6.6	151	68	20.0	4
36	4.9	159	67	29.3	4
37	6.3	106	72	14.9	1
38	3.4	174	87	8.3	4
39	14.4	279	48	22.5	0
40	3.8	86	45	12.8	4
41	13.2	188	59	26.9	0
42	12.7	201	80	25.5	4
43	3.2	120	80	22.9	1
44	2.2	48	32	11.2	1
45	8.5	156	63	20.7	4
46	4.0	145	73	26.2	4
47	5.7	81	39	9.3	4
48	2.6	53	66	10.8	3
49	6.8	161	60	15.6	4

In [30]:

```
# Compute the centroids for K=5 clusters with 4 variables
clusters5.cluster_centers_
```

Out[30]:

```
array([[0.04531605, 0.95994345, 0.24802482, 0.10322567],
       [0.03689098, 0.84108145, 0.52207436, 0.12738443],
       [0.05454612, 0.47341917, 0.85421286, 0.20789277],
       [0.0247473 , 0.69224362, 0.70739404, 0.13009016],
       [0.04205536, 0.90426714, 0.40250352, 0.12472689]])
```

In [31]:

```
# Group data by Clusters (K=5)
crime_data5.groupby('clusters5').agg(['mean']).reset_index()
```

Out[31]:

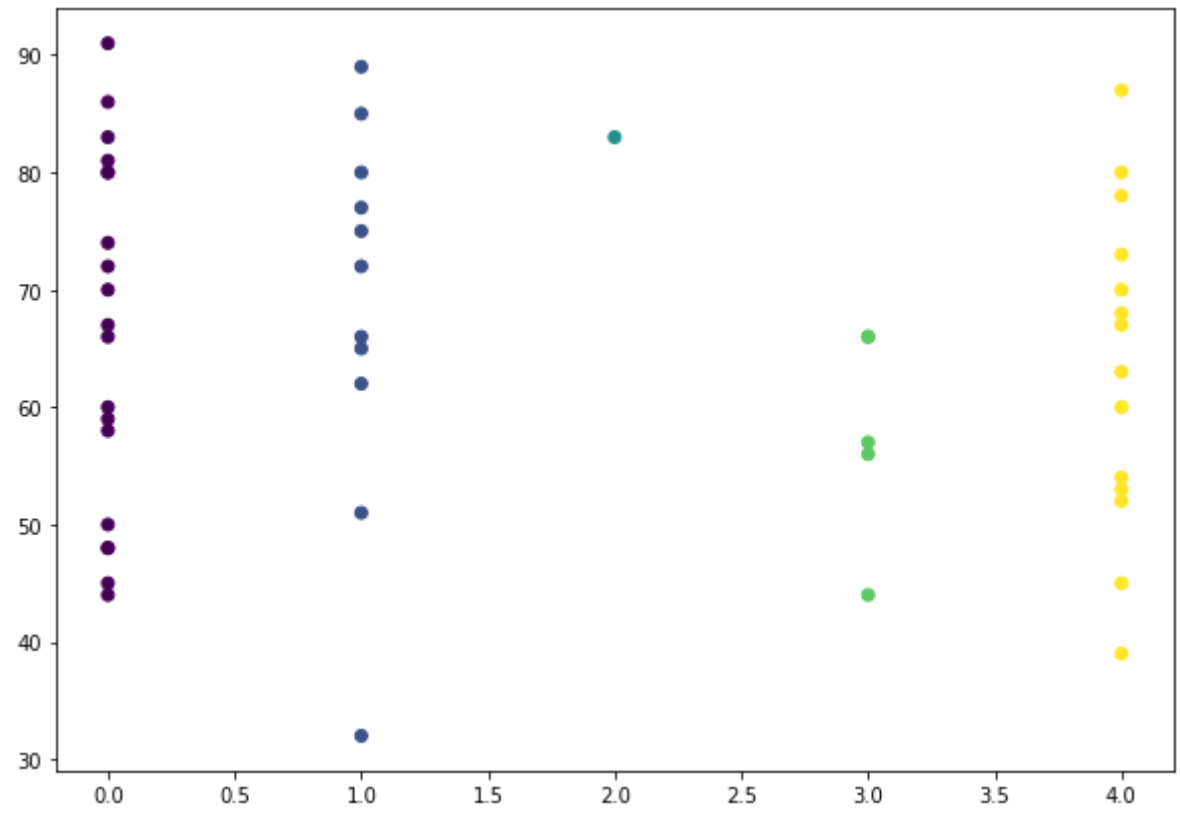
	clusters5	Murder	Assault	UrbanPop	Rape
		mean	mean	mean	mean
0	0	12.021053	260.526316	66.421053	27.694737
1	1	4.881818	111.363636	68.545455	16.354545
2	2	5.300000	46.000000	83.000000	20.200000
3	3	2.080000	56.600000	57.800000	10.760000
4	4	6.542857	145.285714	63.500000	20.107143

In [33]:

```
# Plot Clusters
plt.figure(figsize=(10, 7))
plt.scatter(crime_data5['clusters5'],crime_data5['UrbanPop'], c=clusters5.labels_)
```

Out[33]:

```
<matplotlib.collections.PathCollection at 0x23821dd6448>
```



Hierarchical clustering

In [34]:

```
crime_data=pd.read_csv('crime_data.csv')
crime_data
```

Out[34]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6
5	Colorado	7.9	204	78	38.7
6	Connecticut	3.3	110	77	11.1
7	Delaware	5.9	238	72	15.8
8	Florida	15.4	335	80	31.9
9	Georgia	17.4	211	60	25.8
10	Hawaii	5.3	46	83	20.2
11	Idaho	2.6	120	54	14.2
12	Illinois	10.4	249	83	24.0
13	Indiana	7.2	113	65	21.0
14	Iowa	2.2	56	57	11.3
15	Kansas	6.0	115	66	18.0
16	Kentucky	9.7	109	52	16.3
17	Louisiana	15.4	249	66	22.2
18	Maine	2.1	83	51	7.8
19	Maryland	11.3	300	67	27.8
20	Massachusetts	4.4	149	85	16.3
21	Michigan	12.1	255	74	35.1
22	Minnesota	2.7	72	66	14.9
23	Mississippi	16.1	259	44	17.1
24	Missouri	9.0	178	70	28.2
25	Montana	6.0	109	53	16.4
26	Nebraska	4.3	102	62	16.5
27	Nevada	12.2	252	81	46.0
28	New Hampshire	2.1	57	56	9.5
29	New Jersey	7.4	159	89	18.8
30	New Mexico	11.4	285	70	32.1
31	New York	11.1	254	86	26.1
32	North Carolina	13.0	337	45	16.1

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
33	North Dakota	0.8	45	44	7.3
34	Ohio	7.3	120	75	21.4
35	Oklahoma	6.6	151	68	20.0
36	Oregon	4.9	159	67	29.3
37	Pennsylvania	6.3	106	72	14.9
38	Rhode Island	3.4	174	87	8.3
39	South Carolina	14.4	279	48	22.5
40	South Dakota	3.8	86	45	12.8
41	Tennessee	13.2	188	59	26.9
42	Texas	12.7	201	80	25.5
43	Utah	3.2	120	80	22.9
44	Vermont	2.2	48	32	11.2
45	Virginia	8.5	156	63	20.7
46	Washington	4.0	145	73	26.2
47	West Virginia	5.7	81	39	9.3
48	Wisconsin	2.6	53	66	10.8
49	Wyoming	6.8	161	60	15.6

In [35]:

```
del crime_data['Unnamed: 0']
```

In [36]:

```
# Normalization function
def norm_func(i):
    x = (i-i.min())/(i.max()-i.min())
    return (x)
```

In [37]:

```
# Normalized data frame (considering the numerical part of data)
crime_norm = norm_func(crime_data)
crime_norm
```

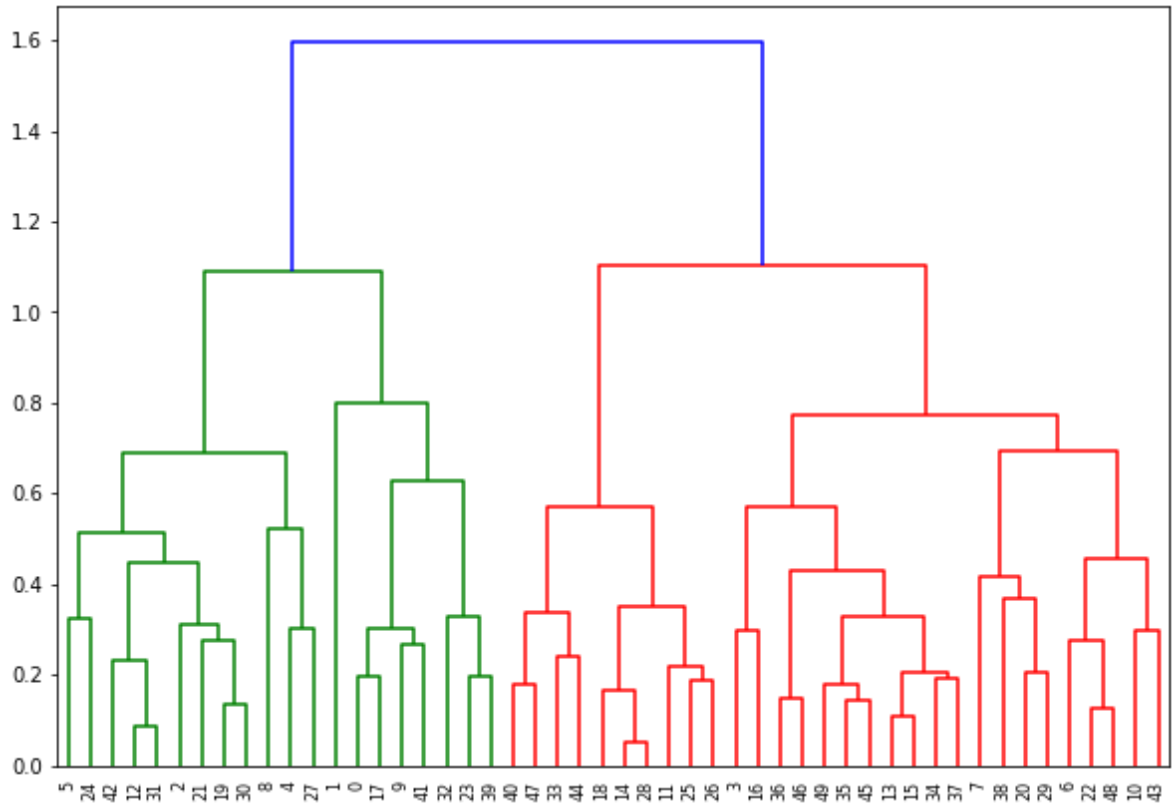
Out[37]:

	Murder	Assault	UrbanPop	Rape
0	0.746988	0.654110	0.440678	0.359173
1	0.554217	0.746575	0.271186	0.961240
2	0.439759	0.852740	0.813559	0.612403
3	0.481928	0.496575	0.305085	0.315245
4	0.493976	0.791096	1.000000	0.860465
5	0.427711	0.544521	0.779661	0.811370
6	0.150602	0.222603	0.762712	0.098191
7	0.307229	0.660959	0.677966	0.219638
8	0.879518	0.993151	0.813559	0.635659
9	1.000000	0.568493	0.474576	0.478036
10	0.271084	0.003425	0.864407	0.333333
11	0.108434	0.256849	0.372881	0.178295
12	0.578313	0.698630	0.864407	0.431525
13	0.385542	0.232877	0.559322	0.354005
14	0.084337	0.037671	0.423729	0.103359
15	0.313253	0.239726	0.576271	0.276486
16	0.536145	0.219178	0.338983	0.232558
17	0.879518	0.698630	0.576271	0.385013
18	0.078313	0.130137	0.322034	0.012920
19	0.632530	0.873288	0.593220	0.529716
20	0.216867	0.356164	0.898305	0.232558
21	0.680723	0.719178	0.711864	0.718346
22	0.114458	0.092466	0.576271	0.196382
23	0.921687	0.732877	0.203390	0.253230
24	0.493976	0.455479	0.644068	0.540052
25	0.313253	0.219178	0.355932	0.235142
26	0.210843	0.195205	0.508475	0.237726
27	0.686747	0.708904	0.830508	1.000000
28	0.078313	0.041096	0.406780	0.056848
29	0.397590	0.390411	0.966102	0.297158
30	0.638554	0.821918	0.644068	0.640827
31	0.620482	0.715753	0.915254	0.485788
32	0.734940	1.000000	0.220339	0.227390

	Murder	Assault	UrbanPop	Rape
33	0.000000	0.000000	0.203390	0.000000
34	0.391566	0.256849	0.728814	0.364341
35	0.349398	0.363014	0.610169	0.328165
36	0.246988	0.390411	0.593220	0.568475
37	0.331325	0.208904	0.677966	0.196382
38	0.156627	0.441781	0.932203	0.025840
39	0.819277	0.801370	0.271186	0.392765
40	0.180723	0.140411	0.220339	0.142119
41	0.746988	0.489726	0.457627	0.506460
42	0.716867	0.534247	0.813559	0.470284
43	0.144578	0.256849	0.813559	0.403101
44	0.084337	0.010274	0.000000	0.100775
45	0.463855	0.380137	0.525424	0.346253
46	0.192771	0.342466	0.694915	0.488372
47	0.295181	0.123288	0.118644	0.051680
48	0.108434	0.027397	0.576271	0.090439
49	0.361446	0.397260	0.474576	0.214470

In [38]:

```
# Create Dendrograms
plt.figure(figsize=(10, 7))
dendograms=sch.dendrogram(sch.linkage(crime_norm,'complete'))
```



In [39]:

```
# Create Clusters (y)
hclusters=AgglomerativeClustering(n_clusters=5,affinity='euclidean',linkage='ward')
hclusters
```

Out[39]:

```
AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                        connectivity=None, distance_threshold=None,
                        linkage='ward', memory=None, n_clusters=5,
                        pooling_func='deprecated')
```

In [40]:

```
y=pd.DataFrame(hclusters.fit_predict(crime_norm),columns=['clusters'])
y['clusters'].value_counts()
```

Out[40]:

```
0    13
2    12
4     9
1     9
3     7
Name: clusters, dtype: int64
```

In [41]:

```
# Adding clusters to dataset
crime_data['clusters']=hclusters.labels_
crime_data
```

Out[41]:

	Murder	Assault	UrbanPop	Rape	clusters
0	13.2	236	58	21.2	3
1	10.0	263	48	44.5	0
2	8.1	294	80	31.0	0
3	8.8	190	50	19.5	4
4	9.0	276	91	40.6	0
5	7.9	204	78	38.7	0
6	3.3	110	77	11.1	1
7	5.9	238	72	15.8	1
8	15.4	335	80	31.9	0
9	17.4	211	60	25.8	3
10	5.3	46	83	20.2	1
11	2.6	120	54	14.2	2
12	10.4	249	83	24.0	0
13	7.2	113	65	21.0	4
14	2.2	56	57	11.3	2
15	6.0	115	66	18.0	4
16	9.7	109	52	16.3	4
17	15.4	249	66	22.2	3
18	2.1	83	51	7.8	2
19	11.3	300	67	27.8	0
20	4.4	149	85	16.3	1
21	12.1	255	74	35.1	0
22	2.7	72	66	14.9	2
23	16.1	259	44	17.1	3
24	9.0	178	70	28.2	0
25	6.0	109	53	16.4	2
26	4.3	102	62	16.5	2
27	12.2	252	81	46.0	0
28	2.1	57	56	9.5	2
29	7.4	159	89	18.8	1
30	11.4	285	70	32.1	0
31	11.1	254	86	26.1	0
32	13.0	337	45	16.1	3

	Murder	Assault	UrbanPop	Rape	clusters
33	0.8	45	44	7.3	2
34	7.3	120	75	21.4	4
35	6.6	151	68	20.0	4
36	4.9	159	67	29.3	1
37	6.3	106	72	14.9	4
38	3.4	174	87	8.3	1
39	14.4	279	48	22.5	3
40	3.8	86	45	12.8	2
41	13.2	188	59	26.9	3
42	12.7	201	80	25.5	0
43	3.2	120	80	22.9	1
44	2.2	48	32	11.2	2
45	8.5	156	63	20.7	4
46	4.0	145	73	26.2	1
47	5.7	81	39	9.3	2
48	2.6	53	66	10.8	2
49	6.8	161	60	15.6	4

In [42]:

```
crime_data.groupby('clusters').agg(['mean']).reset_index()
```

Out[42]:

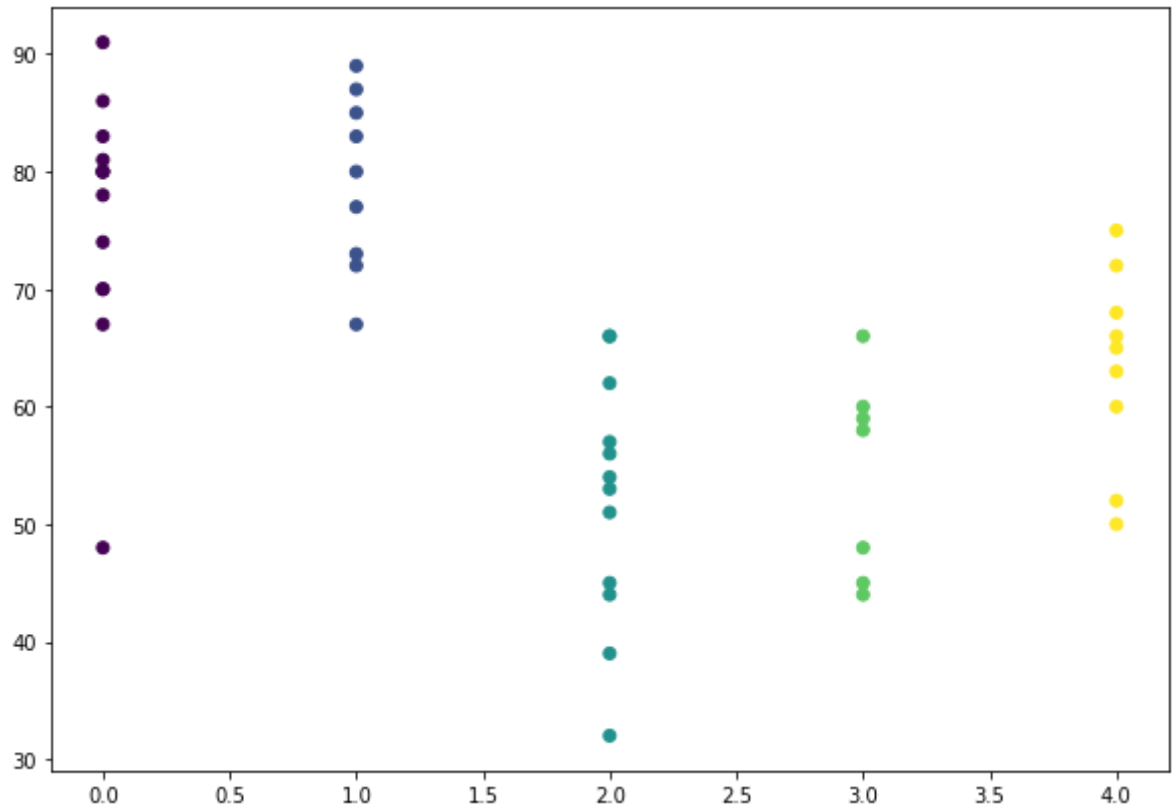
	clusters	Murder	Assault	UrbanPop	Rape
		mean	mean	mean	mean
0	0	10.815385	257.384615	76.000000	33.192308
1	1	4.644444	144.444444	79.222222	18.766667
2	2	3.091667	76.000000	52.083333	11.833333
3	3	14.671429	251.285714	54.285714	21.685714
4	4	7.466667	135.666667	63.444444	18.600000

In [43]:

```
#Plot Clusters
plt.figure(figsize=(10, 7))
plt.scatter(crime_data['clusters'],crime_data['UrbanPop'], c=hclusters.labels_)
```

Out[43]:

```
<matplotlib.collections.PathCollection at 0x23822bfb808>
```



DBSCAN clustering

In [46]:

```
crime_data=pd.read_csv('crime_data.csv')
crime_data.head(10)
```

Out[46]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6
5	Colorado	7.9	204	78	38.7
6	Connecticut	3.3	110	77	11.1
7	Delaware	5.9	238	72	15.8
8	Florida	15.4	335	80	31.9
9	Georgia	17.4	211	60	25.8

In [47]:

```
del crime_data['Unnamed: 0']
```

In [48]:

```
# Normalize heterogenous numerical data using standard scalar fit transform to dataset
crime_norm=StandardScaler().fit_transform(crime_data)
crime_norm
```

Out[48]:

```
array([[ 1.25517927,  0.79078716, -0.52619514, -0.00345116],
       [ 0.51301858,  1.11805959, -1.22406668,  2.50942392],
       [ 0.07236067,  1.49381682,  1.00912225,  1.05346626],
       [ 0.23470832,  0.23321191, -1.08449238, -0.18679398],
       [ 0.28109336,  1.2756352 ,  1.77678094,  2.08881393],
       [ 0.02597562,  0.40290872,  0.86954794,  1.88390137],
       [-1.04088037, -0.73648418,  0.79976079, -1.09272319],
       [-0.43787481,  0.81502956,  0.45082502, -0.58583422],
       [ 1.76541475,  1.99078607,  1.00912225,  1.1505301 ],
       [ 2.22926518,  0.48775713, -0.38662083,  0.49265293],
       [-0.57702994, -1.51224105,  1.21848371, -0.11129987],
       [-1.20322802, -0.61527217, -0.80534376, -0.75839217],
       [ 0.60578867,  0.94836277,  1.21848371,  0.29852525],
       [-0.13637203, -0.70012057, -0.03768506, -0.0250209 ],
       [-1.29599811, -1.39102904, -0.5959823 , -1.07115345],
       [-0.41468229, -0.67587817,  0.03210209, -0.34856705],
       [ 0.44344101, -0.74860538, -0.94491807, -0.53190987],
       [ 1.76541475,  0.94836277,  0.03210209,  0.10439756],
       [-1.31919063, -1.06375661, -1.01470522, -1.44862395],
       [ 0.81452136,  1.56654403,  0.10188925,  0.70835037],
       [-0.78576263, -0.26375734,  1.35805802, -0.53190987],
       [ 1.00006153,  1.02108998,  0.59039932,  1.49564599],
       [-1.1800355 , -1.19708982,  0.03210209, -0.68289807],
       [ 1.9277624 ,  1.06957478, -1.5032153 , -0.44563089],
       [ 0.28109336,  0.0877575 ,  0.31125071,  0.75148985],
       [-0.41468229, -0.74860538, -0.87513091, -0.521125  ],
       [-0.80895515, -0.83345379, -0.24704653, -0.51034012],
       [ 1.02325405,  0.98472638,  1.0789094 ,  2.671197  ],
       [-1.31919063, -1.37890783, -0.66576945, -1.26528114],
       [-0.08998698, -0.14254532,  1.63720664, -0.26228808],
       [ 0.83771388,  1.38472601,  0.31125071,  1.17209984],
       [ 0.76813632,  1.00896878,  1.42784517,  0.52500755],
       [ 1.20879423,  2.01502847, -1.43342815, -0.55347961],
       [-1.62069341, -1.52436225, -1.5032153 , -1.50254831],
       [-0.11317951, -0.61527217,  0.66018648,  0.01811858],
       [-0.27552716, -0.23951493,  0.1716764 , -0.13286962],
       [-0.66980002, -0.14254532,  0.10188925,  0.87012344],
       [-0.34510472, -0.78496898,  0.45082502, -0.68289807],
       [-1.01768785,  0.03927269,  1.49763233, -1.39469959],
       [ 1.53348953,  1.3119988 , -1.22406668,  0.13675217],
       [-0.92491776, -1.027393  , -1.43342815, -0.90938037],
       [ 1.25517927,  0.20896951, -0.45640799,  0.61128652],
       [ 1.13921666,  0.36654512,  1.00912225,  0.46029832],
       [-1.06407289, -0.61527217,  1.00912225,  0.17989166],
       [-1.29599811, -1.48799864, -2.34066115, -1.08193832],
       [ 0.16513075, -0.17890893, -0.17725937, -0.05737552],
       [-0.87853272, -0.31224214,  0.52061217,  0.53579242],
       [-0.48425985, -1.08799901, -1.85215107, -1.28685088],
       [-1.20322802, -1.42739264,  0.03210209, -1.1250778 ],
       [-0.22914211, -0.11830292, -0.38662083, -0.60740397]])
```

In [50]:

```
# DBSCAN Clustering
dbscan=DBSCAN(eps=1,min_samples=4)
dbscan.fit(crime_norm)
```

Out[50]:

```
DBSCAN(algorithm='auto', eps=1, leaf_size=30, metric='euclidean',
       metric_params=None, min_samples=4, n_jobs=None, p=None)
```

In [51]:

```
#Noisy samples are given the label -1.
dbscan.labels_
```

Out[51]:

```
array([ 0, -1, -1, -1, -1, -1,  1, -1, -1, -1, -1,  1, -1,  1,  1,  1,  1,
        0,  1, -1,  1, -1,  1, -1,  1,  1,  1, -1,  1,  1, -1, -1, -1,  1,
        1,  1,  1,  1,  1,  0,  1,  0, -1,  1,  1,  1,  1,  1,  1,  1],
      dtype=int64)
```

In [53]:

```
crime_data.groupby('clusters').agg(['mean']).reset_index()
```

Out[53]:

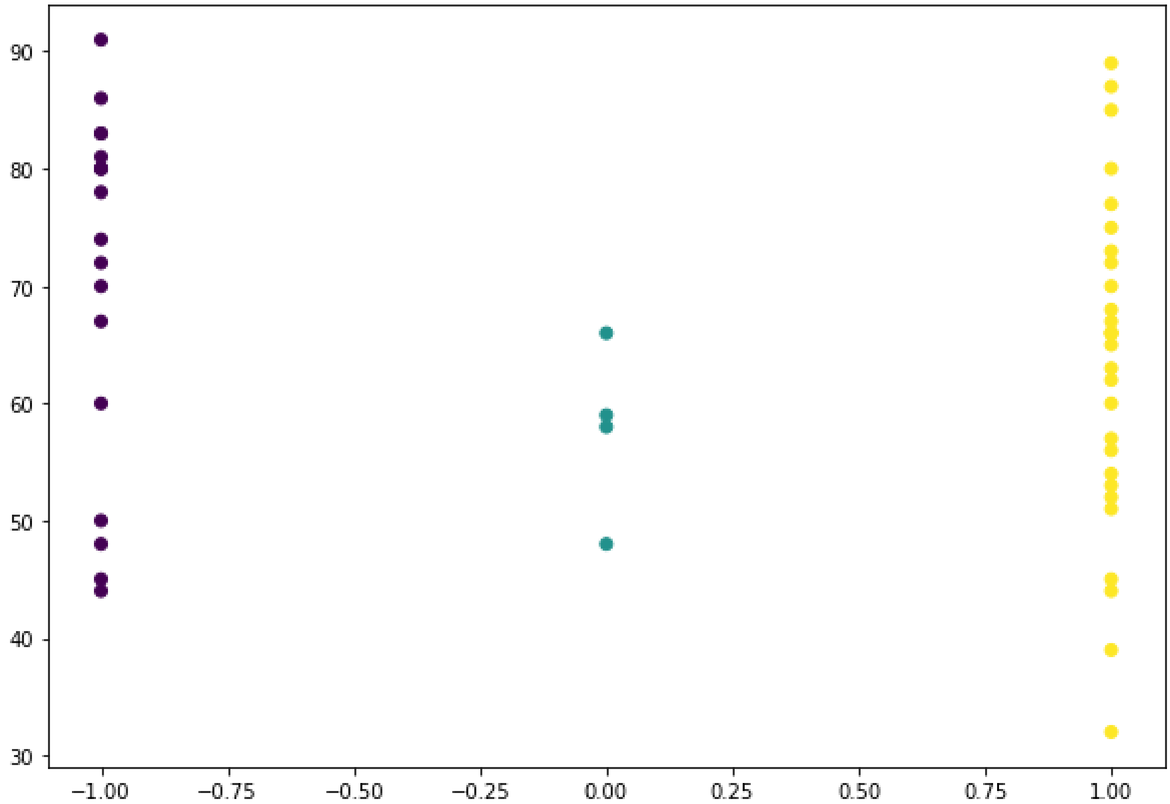
	clusters	Murder	Assault	UrbanPop	Rape
		mean	mean	mean	mean
0	-1	11.005556	247.166667	70.666667	28.766667
1	0	14.050000	238.000000	57.750000	23.200000
2	1	4.825000	112.035714	63.357143	16.107143

In [54]:

```
# Plot Clusters
plt.figure(figsize=(10, 7))
plt.scatter(crime_data['clusters'],crime_data['UrbanPop'], c=dbscan.labels_)
```

Out[54]:

```
<matplotlib.collections.PathCollection at 0x23824fe6588>
```



>>>The End!! >>