In [1]:
```
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('crime_data.csv')
df.head()
```

Out[2]:

	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

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  50 non-null     object 
 1   Murder      50 non-null     float64
 2   Assault     50 non-null     int64  
 3   UrbanPop    50 non-null     int64  
 4   Rape        50 non-null     float64
dtypes: float64(2), int64(2), object(1)
memory usage: 2.1+ KB
```

In [4]:
```
def normfunc(i):
    x = (i-i.min())/(i.max()-i.min())
    return x
```

In [5]:
```
df_norm = normfunc(df.iloc[:,1:])
df_norm
```

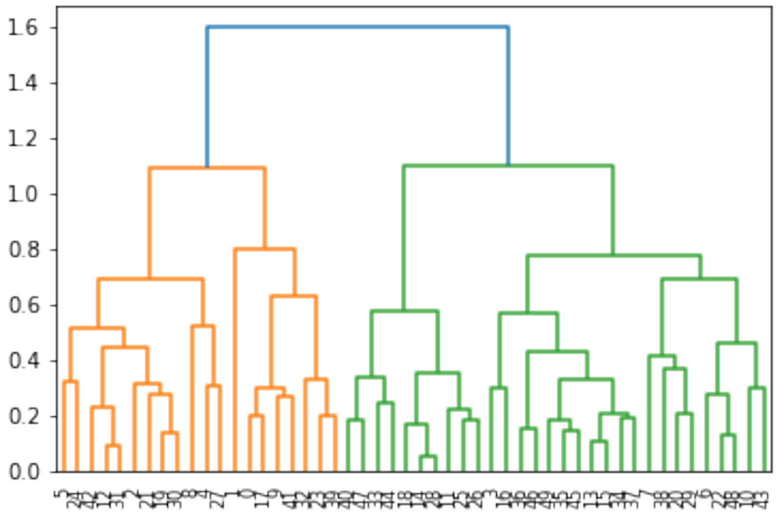
Out[5]:

	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
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 [6]:
```
dendrogram = sch.dendrogram(sch.linkage(df_norm, method = "complete
```



In [7]:
```
hc = AgglomerativeClustering(n_clusters=4, affinity='euclidean', li
```

In [8]:
```
df_pred = hc.fit_predict(df_norm)
df_pred
```

Out[8]:
```
array([0, 0, 3, 1, 3, 3, 1, 1, 3, 0, 1, 2, 3, 1, 2, 1, 1, 0, 2, 3,
1, 3,
       1, 0, 3, 2, 2, 3, 2, 1, 3, 3, 0, 2, 1, 1, 1, 1, 1, 0, 2, 0,
3, 1,
       2, 1, 1, 2, 1, 1])
```

In [10]:
```
hc_data = df.copy()
hc_data.head()
```

Out[10]:

	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

In [12]:
```
hc_data["Clusters"] = df_pred
hc_data
```

Out[12]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape	Clusters
0	Alabama	13.2	236	58	21.2	0
1	Alaska	10.0	263	48	44.5	0
2	Arizona	8.1	294	80	31.0	3
3	Arkansas	8.8	190	50	19.5	1
4	California	9.0	276	91	40.6	3
5	Colorado	7.9	204	78	38.7	3
6	Connecticut	3.3	110	77	11.1	1
7	Delaware	5.9	238	72	15.8	1
8	Florida	15.4	335	80	31.9	3
9	Georgia	17.4	211	60	25.8	0
10	Hawaii	5.3	46	83	20.2	1
11	Idaho	2.6	120	54	14.2	2
12	Illinois	10.4	249	83	24.0	3
13	Indiana	7.2	113	65	21.0	1
14	Iowa	2.2	56	57	11.3	2
15	Kansas	6.0	115	66	18.0	1
16	Kentucky	9.7	109	52	16.3	1
17	Louisiana	15.4	249	66	22.2	0
18	Maine	2.1	83	51	7.8	2
19	Maryland	11.3	300	67	27.8	3
20	Massachusetts	4.4	149	85	16.3	1

21	Michigan	12.1	255	74	35.1	3
22	Minnesota	2.7	72	66	14.9	1
23	Mississippi	16.1	259	44	17.1	0
24	Missouri	9.0	178	70	28.2	3
25	Montana	6.0	109	53	16.4	2
26	Nebraska	4.3	102	62	16.5	2
27	Nevada	12.2	252	81	46.0	3
28	New Hampshire	2.1	57	56	9.5	2
29	New Jersey	7.4	159	89	18.8	1
30	New Mexico	11.4	285	70	32.1	3
31	New York	11.1	254	86	26.1	3
32	North Carolina	13.0	337	45	16.1	0
33	North Dakota	0.8	45	44	7.3	2
34	Ohio	7.3	120	75	21.4	1
35	Oklahoma	6.6	151	68	20.0	1
36	Oregon	4.9	159	67	29.3	1
37	Pennsylvania	6.3	106	72	14.9	1
38	Rhode Island	3.4	174	87	8.3	1
39	South Carolina	14.4	279	48	22.5	0
40	South Dakota	3.8	86	45	12.8	2
41	Tennessee	13.2	188	59	26.9	0
42	Texas	12.7	201	80	25.5	3
43	Utah	3.2	120	80	22.9	1
44	Vermont	2.2	48	32	11.2	2
45	Virginia	8.5	156	63	20.7	1
46	Washington	4.0	145	73	26.2	1
47	West Virginia	5.7	81	39	9.3	2
48	Wisconsin	2.6	53	66	10.8	1
49	Wyoming	6.8	161	60	15.6	1

Method 2 - KMeans

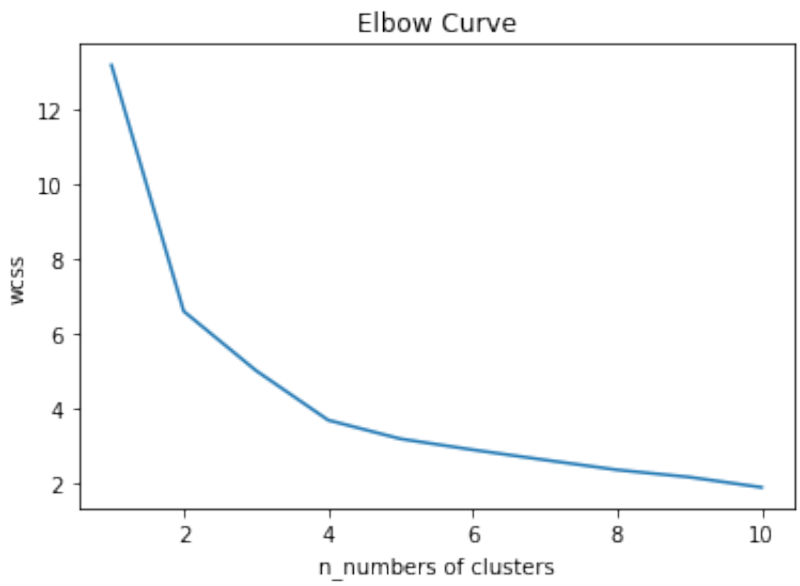
In [13]:
```
from sklearn.cluster import KMeans
df.head()
```

Out[13]:

	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

In [14]:
```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i)
    kmeans.fit(df_norm)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title("Elbow Curve")
plt.xlabel("n_numbers of clusters")
plt.ylabel("wcss")
```

Out[14]:
```
Text(0, 0.5, 'wcss')
```



In [15]:
```
model = KMeans(n_clusters=4)
model.fit(df_norm)
model.labels_
```

Out[15]:
```
array([3, 0, 0, 3, 0, 0, 1, 1, 0, 3, 1, 2, 0, 1, 2, 1, 2, 3, 2, 0,
1, 0,
       2, 3, 0, 2, 2, 0, 2, 1, 0, 0, 3, 2, 1, 1, 1, 1, 1, 3, 2, 3,
0, 1,
       2, 1, 1, 2, 2, 1], dtype=int32)
```

In [17]:
```
km_data = df.copy()
km_data.head()
km_data["km_Cluster"] = model.labels_
km_data.iloc[:,0:5].groupby(km_data["km_Cluster"]).mean()
```

Out[17]:

km_Cluster	Murder	Assault	UrbanPop	Rape
0	10.815385	257.384615	76.000000	33.192308
1	5.656250	138.875000	73.875000	18.781250
2	3.600000	78.538462	52.076923	12.176923
3	13.937500	243.625000	53.750000	21.412500

DBSCAN

In [19]:
```
from sklearn.cluster import DBSCAN
from sklearn import metrics
from sklearn.preprocessing import StandardScaler
df.head()
```

Out[19]:

	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

In [21]:
```
db = df.iloc[:,1:5].values
sc = StandardScaler().fit(db)
x = sc.transform(db)
```

In [22]:
```
dbscan = DBSCAN(eps=2,min_samples=6)
dbscan.fit(x)
dbscan.labels_
```

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

In [23]:
```
cl = pd.DataFrame(dbscan.labels_,columns=["clust"])
cl.head()
```

Out[23]:

	clust
0	0
1	-1
2	0
3	0
4	0

In [25]:
```
df["clust"] = cl
df.head()
```

Out[25]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape	clust
0	Alabama	13.2	236	58	21.2	0
1	Alaska	10.0	263	48	44.5	-1
2	Arizona	8.1	294	80	31.0	0
3	Arkansas	8.8	190	50	19.5	0
4	California	9.0	276	91	40.6	0

BY dbscan we can idetify which recourds should eligibale to perform a clustering after that we perform clustering to datasets by usng of KMeans or Heirarchy

In [ ]: