```python
#### import libraries
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import AgglomerativeClustering, KMeans
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import seaborn as sns

#### loading dataset
data = pd.read_csv('C:/Users/arifa/Downloads/Assignment/Assignment/5.
Clustering - Capstone Project 3/5. Clustering/crime_data.csv')

data.head()
```

	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

```python
##### renaming cloumn for unnamed.
data.rename(columns={'Unnamed: 0': 'City'}, inplace=True)

data.head()
```

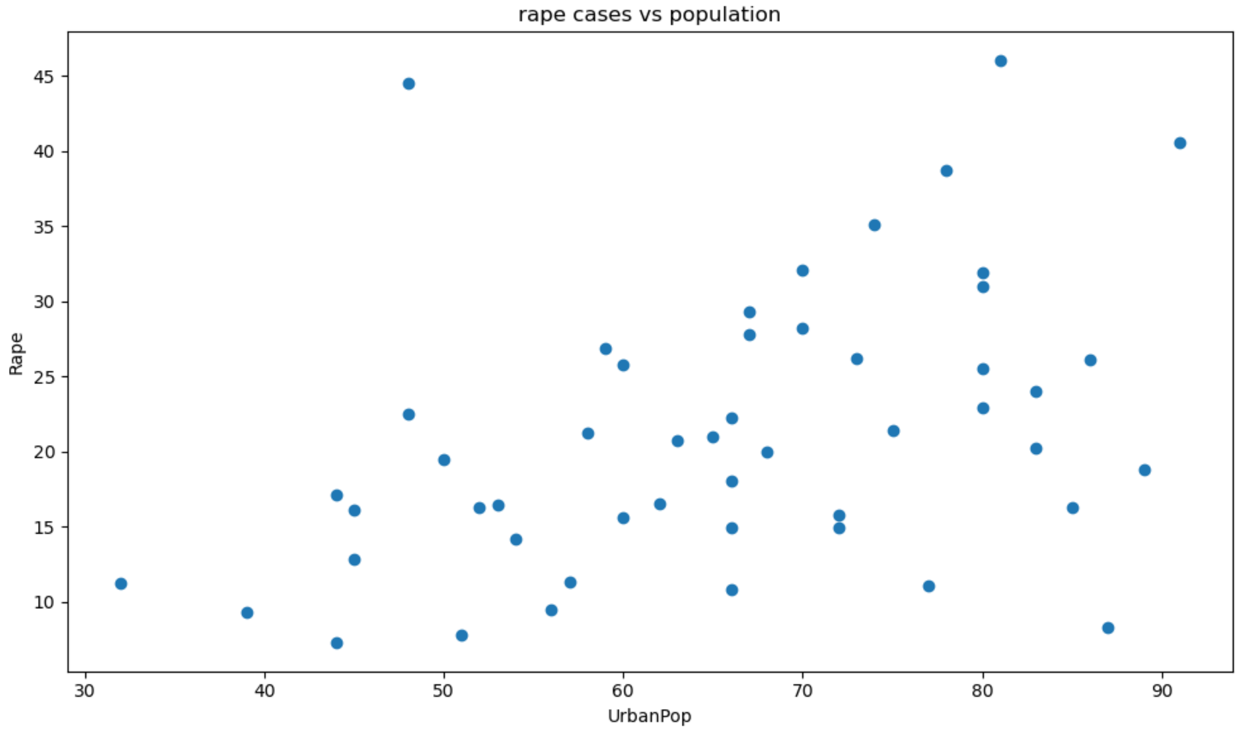
	City	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6

```python
#### check the null values.
data.isnull().sum()
```

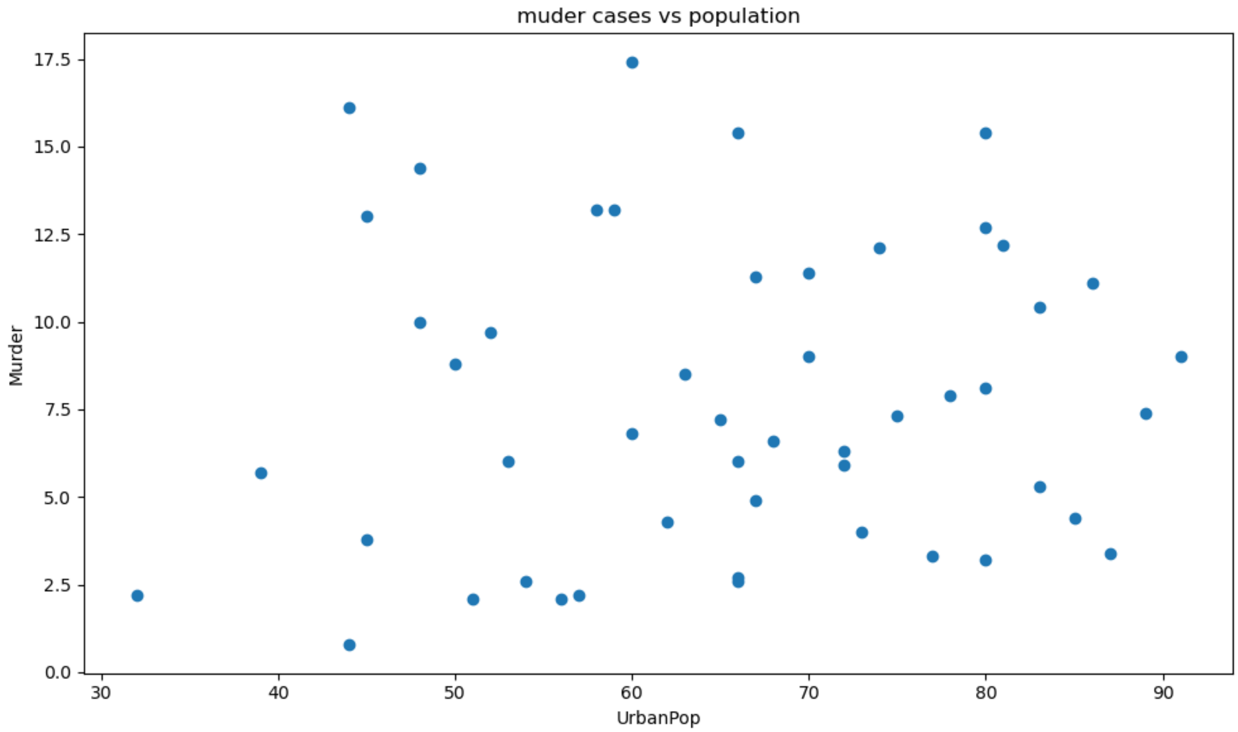
```
City        0
Murder      0
Assault     0
UrbanPop    0
Rape        0
dtype: int64
```

```python
#### exploratory data analysis-EDA.
#### Insights- cases are increasing with increase in population
plt.figure(figsize=(10, 6))
plt.scatter(data['UrbanPop'],data['Rape'])
plt.title('rape cases vs population')
plt.xlabel('UrbanPop')
plt.ylabel('Rape')
```

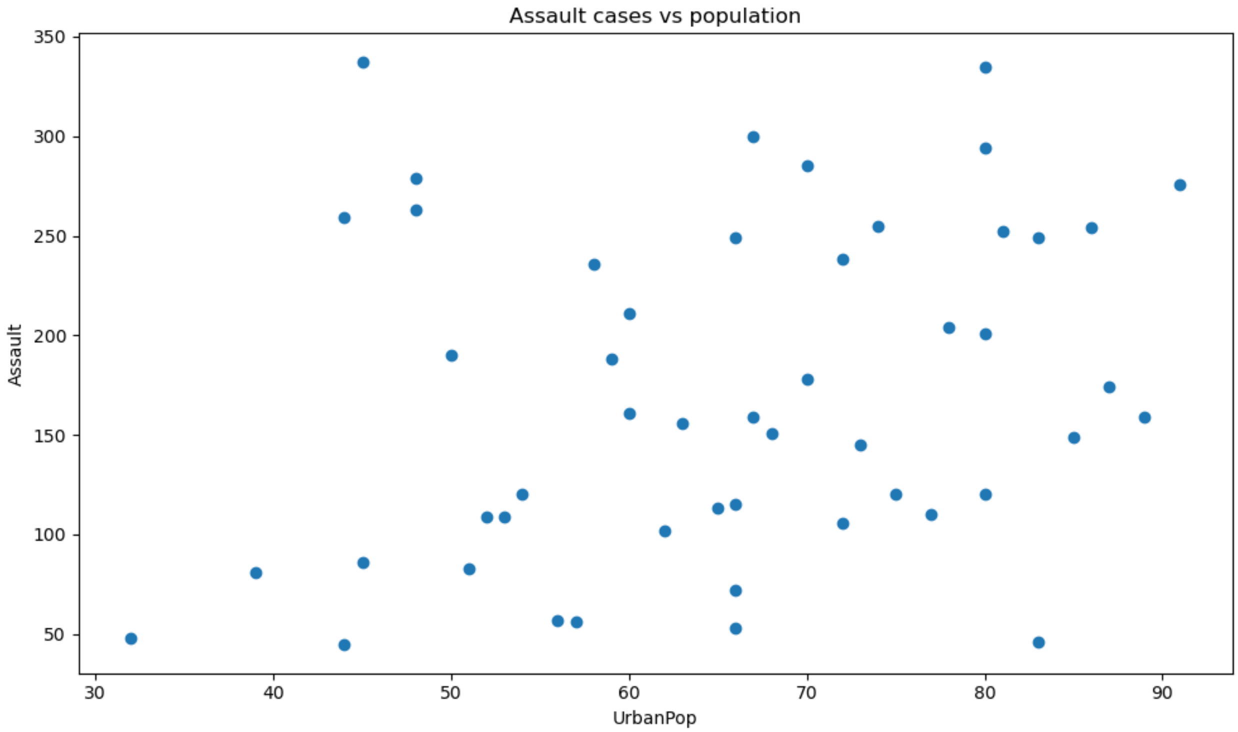
```
plt.tight_layout()
plt.show()
```



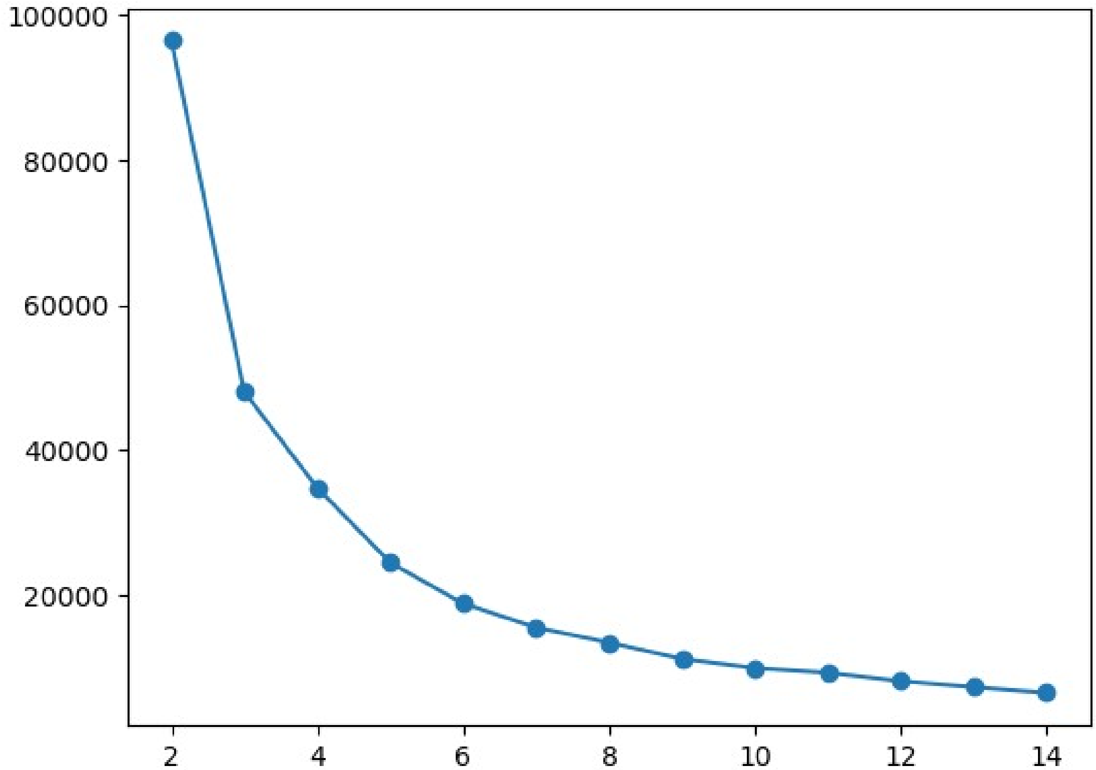

```
####  more cases in high population.
plt.figure(figsize=(10, 6))
plt.scatter(data['UrbanPop'],data['Murder'])
plt.title('muder cases vs population')
plt.xlabel('UrbanPop')
plt.ylabel('Murder')
plt.tight_layout()
plt.show()
```




```
#### more cases in high population.
plt.figure(figsize=(10, 6))
plt.scatter(data['UrbanPop'],data['Assault'])
plt.title('Assault cases vs population')
plt.xlabel('UrbanPop')
plt.ylabel('Assault')
plt.tight_layout()
plt.show()
```




```
#### clustering only works on numerical data
x = data[['Murder', 'Assault', 'UrbanPop', 'Rape']]

##### how to select the optimal clusters.
cluster_range = [2,3,4,5,6,7,8,9,10,11,12,13,14]
inertias = [] ###variances
for c in cluster_range:
    kmeans =  KMeans(n_clusters=c,random_state=0,).fit(x)
    inertias.append(kmeans.inertia_)

plt.figure()
plt.plot(cluster_range,inertias,marker = 'o')
plt.show()
```




```
#### data preprocessing scaling all the data in unique unit.
scaler = StandardScaler()
scaler.fit(x)
scaled_data = scaler.fit_transform(x)

scaled_data

array([[ 1.25517927,  0.79078716, -0.52619514, -0.00345116,  1.5466429
],
       [ 0.51301858,  1.11805959, -1.22406668,  2.50942392, -
0.41113292],
       [ 0.07236067,  1.49381682,  1.00912225,  1.05346626, -
0.41113292],
       [ 0.23470832,  0.23321191, -1.08449238, -0.18679398,  1.5466429
],
       [ 0.28109336,  1.2756352 ,  1.77678094,  2.08881393, -
0.41113292],
       [ 0.02597562,  0.40290872,  0.86954794,  1.88390137, -
0.41113292],
       [-1.04088037, -0.73648418,  0.79976079, -1.09272319,
0.56775499],
       [-0.43787481,  0.81502956,  0.45082502, -0.58583422,
0.56775499],
       [ 1.76541475,  1.99078607,  1.00912225,  1.1505301 , -
0.41113292],
```

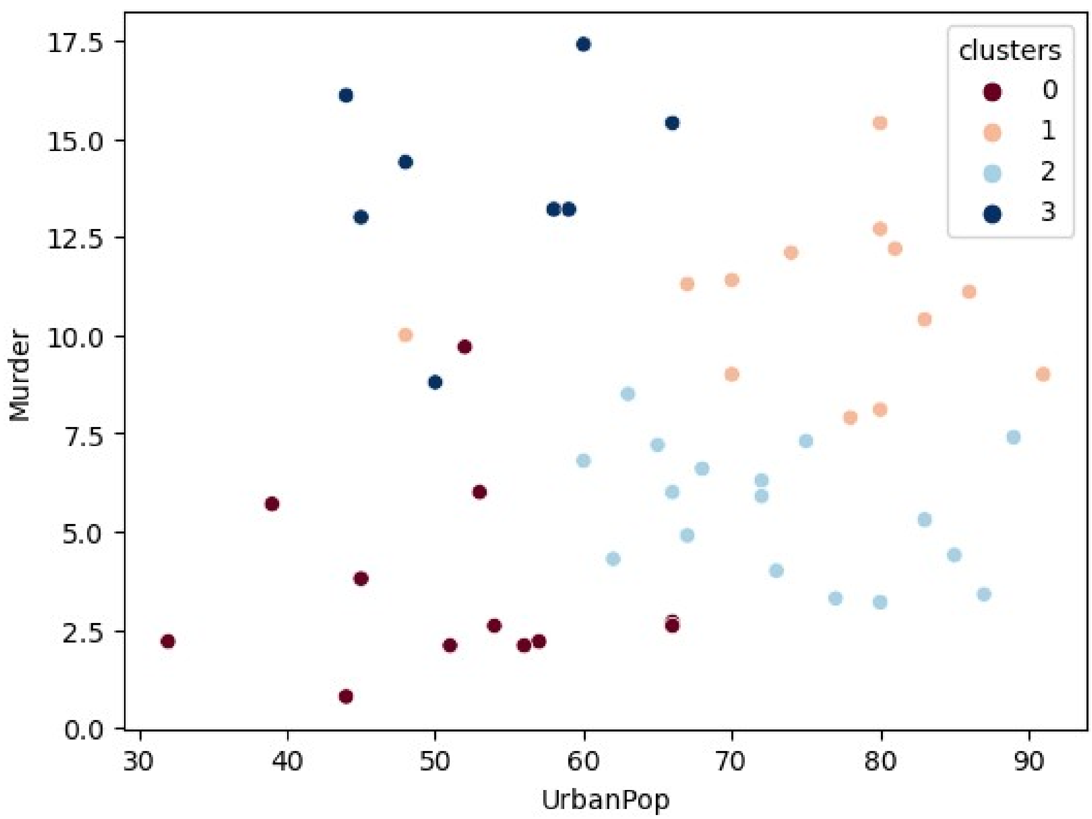
```
       [ 2.22926518,  0.48775713, -0.38662083,  0.49265293,  1.5466429
],
       [-0.57702994, -1.51224105,  1.21848371, -0.11129987,
0.56775499],
       [-1.20322802, -0.61527217, -0.80534376, -0.75839217, -
1.39002083],
       [ 0.60578867,  0.94836277,  1.21848371,  0.29852525, -
0.41113292],
       [-0.13637203, -0.70012057, -0.03768506, -0.0250209 ,
0.56775499],
       [-1.29599811, -1.39102904, -0.5959823 , -1.07115345, -
1.39002083],
       [-0.41468229, -0.67587817,  0.03210209, -0.34856705,
0.56775499],
       [ 0.44344101, -0.74860538, -0.94491807, -0.53190987, -
1.39002083],
       [ 1.76541475,  0.94836277,  0.03210209,  0.10439756,  1.5466429
],
       [-1.31919063, -1.06375661, -1.01470522, -1.44862395, -
1.39002083],
       [ 0.81452136,  1.56654403,  0.10188925,  0.70835037, -
0.41113292],
       [-0.78576263, -0.26375734,  1.35805802, -0.53190987,
0.56775499],
       [ 1.00006153,  1.02108998,  0.59039932,  1.49564599, -
0.41113292],
       [-1.1800355 , -1.19708982,  0.03210209, -0.68289807, -
1.39002083],
       [ 1.9277624 ,  1.06957478, -1.5032153 , -0.44563089,  1.5466429
],
       [ 0.28109336,  0.0877575 ,  0.31125071,  0.75148985, -
0.41113292],
       [-0.41468229, -0.74860538, -0.87513091, -0.521125  , -
1.39002083],
       [-0.80895515, -0.83345379, -0.24704653, -0.51034012,
0.56775499],
       [ 1.02325405,  0.98472638,  1.0789094 ,  2.671197  , -
0.41113292],
       [-1.31919063, -1.37890783, -0.66576945, -1.26528114, -
1.39002083],
       [-0.08998698, -0.14254532,  1.63720664, -0.26228808,
0.56775499],
       [ 0.83771388,  1.38472601,  0.31125071,  1.17209984, -
0.41113292],
       [ 0.76813632,  1.00896878,  1.42784517,  0.52500755, -
0.41113292],
       [ 1.20879423,  2.01502847, -1.43342815, -0.55347961,  1.5466429
],
       [-1.62069341, -1.52436225, -1.5032153 , -1.50254831, -
```

```
1.39002083],
       [-0.11317951, -0.61527217,  0.66018648,  0.01811858,
0.56775499],
       [-0.27552716, -0.23951493,  0.1716764 , -0.13286962,
0.56775499],
       [-0.66980002, -0.14254532,  0.10188925,  0.87012344,
0.56775499],
       [-0.34510472, -0.78496898,  0.45082502, -0.68289807,
0.56775499],
       [-1.01768785,  0.03927269,  1.49763233, -1.39469959,
0.56775499],
       [ 1.53348953,  1.3119988 , -1.22406668,  0.13675217,  1.5466429
],
       [-0.92491776, -1.027393  , -1.43342815, -0.90938037, -
1.39002083],
       [ 1.25517927,  0.20896951, -0.45640799,  0.61128652,  1.5466429
],
       [ 1.13921666,  0.36654512,  1.00912225,  0.46029832, -
0.41113292],
       [-1.06407289, -0.61527217,  1.00912225,  0.17989166,
0.56775499],
       [-1.29599811, -1.48799864, -2.34066115, -1.08193832, -
1.39002083],
       [ 0.16513075, -0.17890893, -0.17725937, -0.05737552,
0.56775499],
       [-0.87853272, -0.31224214,  0.52061217,  0.53579242,
0.56775499],
       [-0.48425985, -1.08799901, -1.85215107, -1.28685088, -
1.39002083],
       [-1.20322802, -1.42739264,  0.03210209, -1.1250778 , -
1.39002083],
       [-0.22914211, -0.11830292, -0.38662083, -0.60740397,
0.56775499]])

#### we got optimum clusters as 4 from elbow method.
km = KMeans(n_clusters=4)
km.fit(scaled_data)
predicted_labels = km.predict(scaled_data)

predicted_labels

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

#### we are putting clusters in the dataframe.
x['clusters']= predicted_labels
x.head(5)
```

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

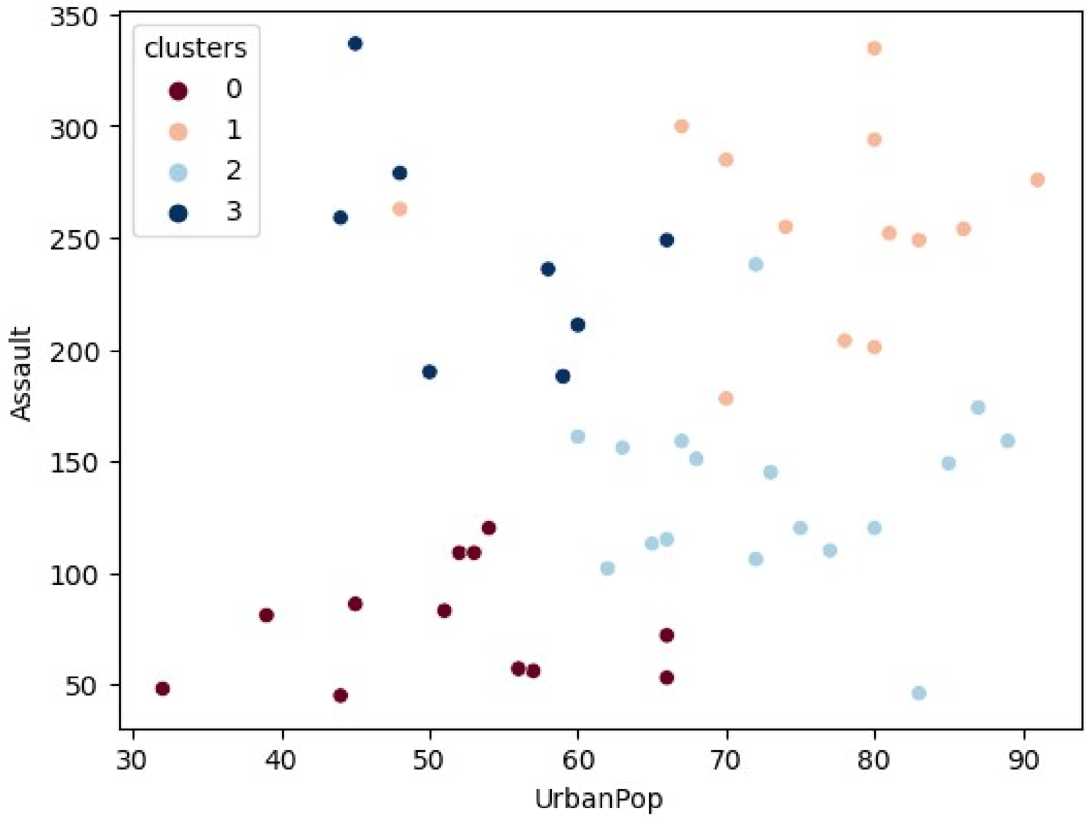
```
#### ploting  clusters using features to analyze.
sns.scatterplot(x = 'UrbanPop',y = 'Murder', hue =
'clusters',data=x,palette='RdBu')
```

```
<AxesSubplot:xlabel='UrbanPop', ylabel='Murder'>
```



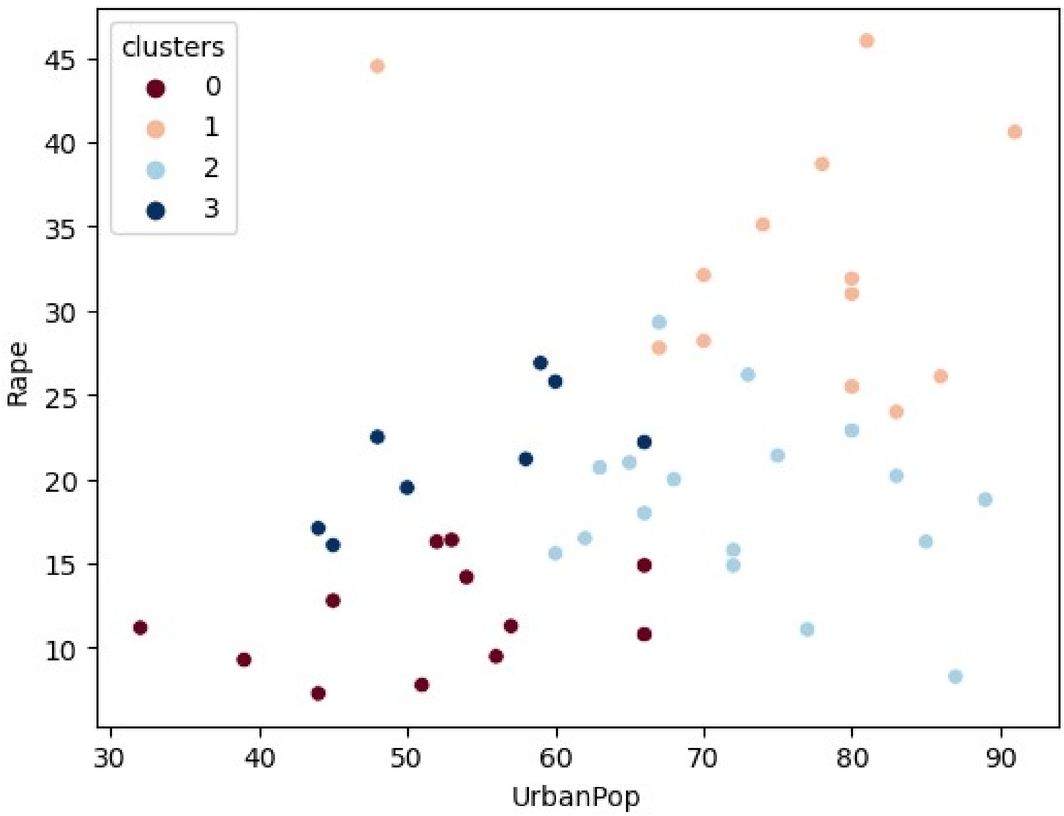

```
sns.scatterplot(x = 'UrbanPop',y = 'Assault', hue =
'clusters',data=x,palette='RdBu')
```

```
<AxesSubplot:xlabel='UrbanPop', ylabel='Assault'>
```




```
sns.scatterplot(x = 'UrbanPop',y = 'Rape', hue =
'clusters',data=x,palette='RdBu')
```

```
<AxesSubplot:xlabel='UrbanPop', ylabel='Rape'>
```




```
#### putting all numerical features in X. For Hierarchical clustering.
X=x[['Murder','Assault','UrbanPop','Rape']]
X.head(5)
```

	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

```
#### using AgglomerativeClustering  Hierarchical clustering.
from sklearn.cluster import AgglomerativeClustering
```

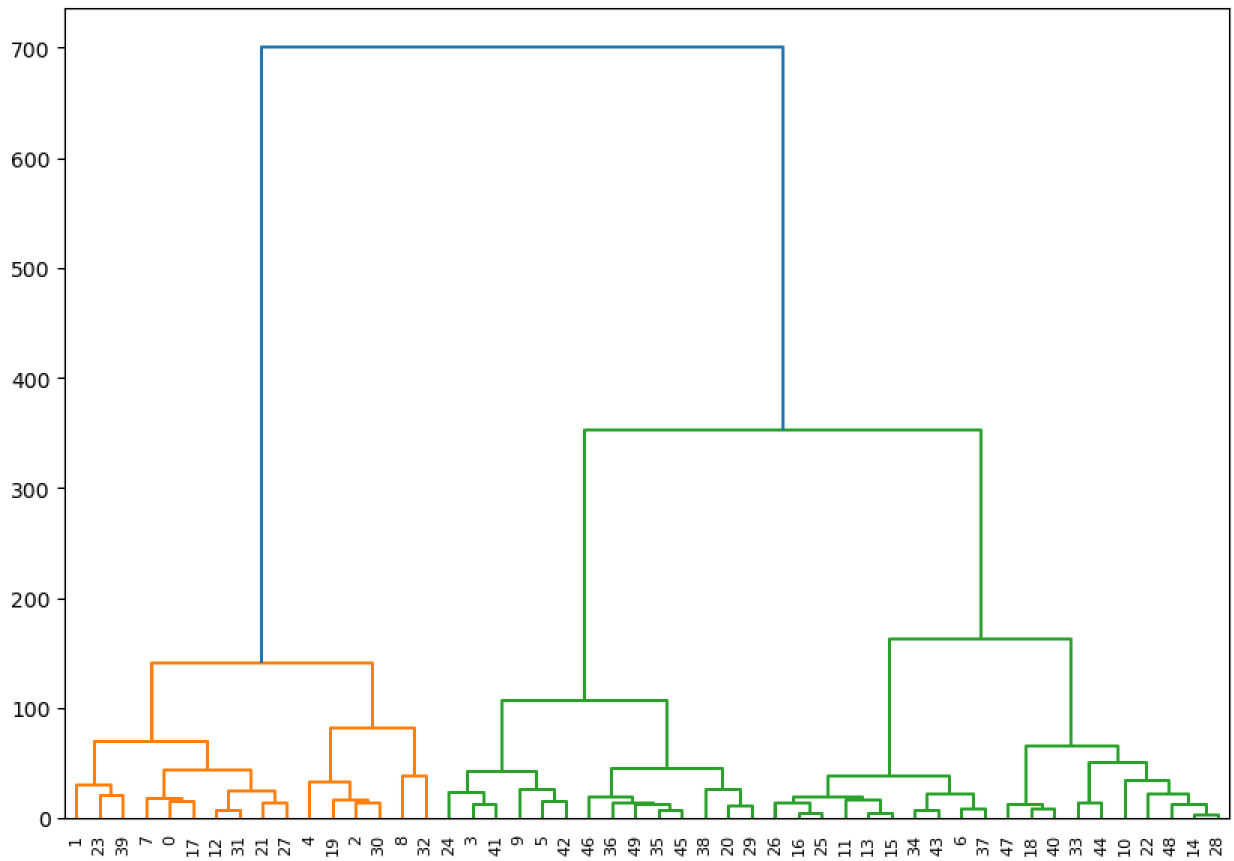
```
#### as we got optimum clusters as 4 . using single linkage.
agm =AgglomerativeClustering(n_clusters = 4,linkage = 'single')
agm.fit(X)
```

```
AgglomerativeClustering(linkage='single', n_clusters=4)
```

```
pred_agm = agm.fit_predict(X)
pred_agm
```

```
array([0, 3, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
```

```
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0], dtype=int64)

##### ploting dendogram to show the clustering process.
import scipy.cluster.hierarchy as sch
plt.figure(figsize=(10,7))
dend = sch.dendrogram(sch.linkage(X,method='ward'))
```

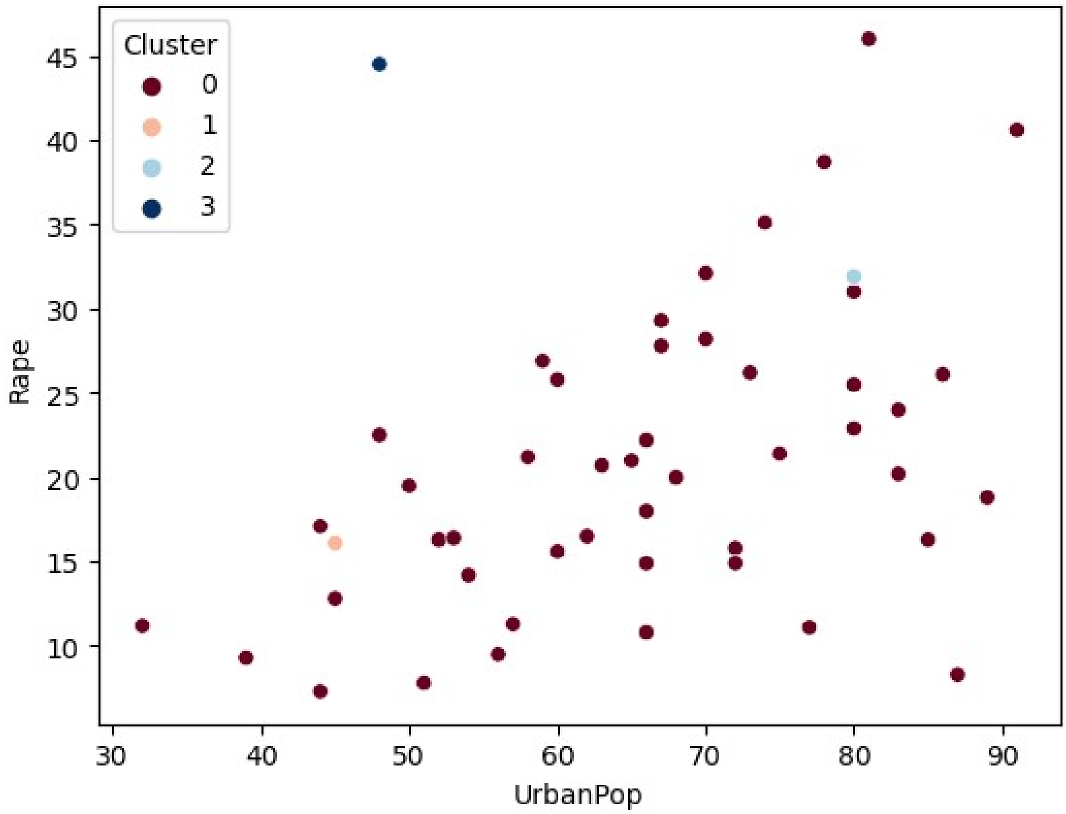



```
X['Cluster']=pred_agm
X.head(5)
```

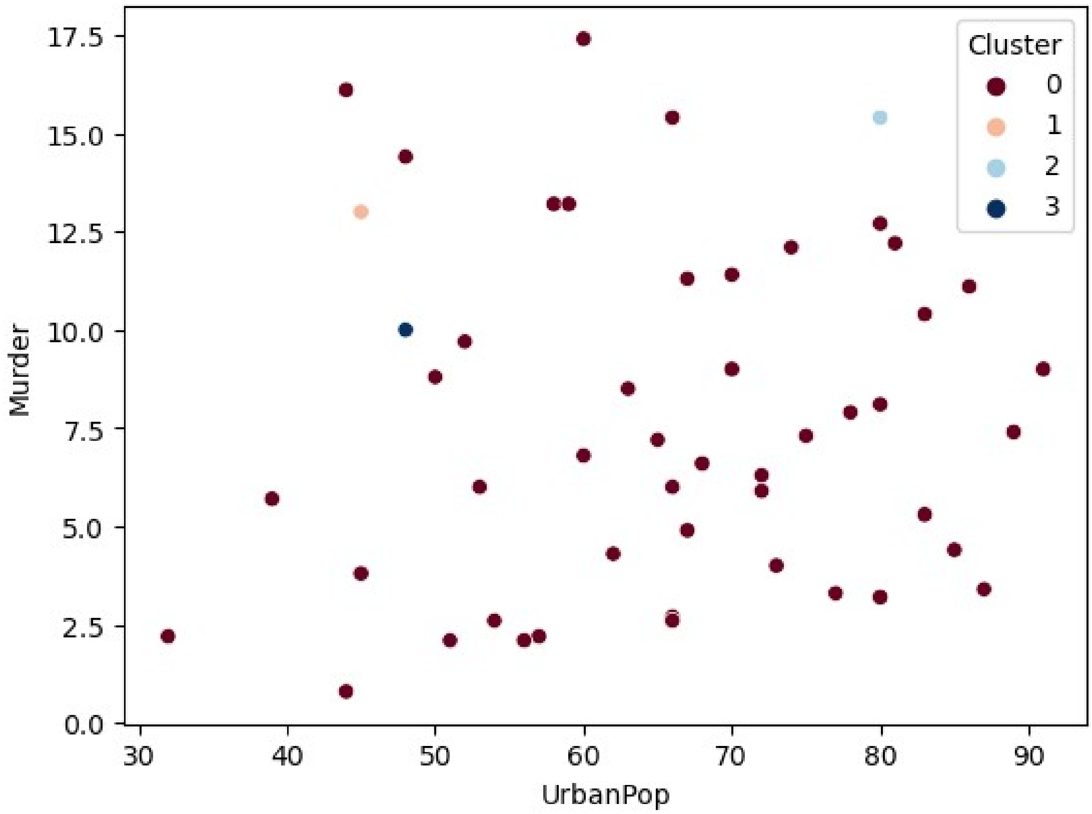
	Murder	Assault	UrbanPop	Rape	Cluster
0	13.2	236	58	21.2	0
1	10.0	263	48	44.5	3
2	8.1	294	80	31.0	0
3	8.8	190	50	19.5	0
4	9.0	276	91	40.6	0

```
#### ploting to gain insights using scatterplot.
sns.scatterplot(x = 'UrbanPop',y = 'Rape', hue =
'Cluster',data=X,palette='RdBu')
```

```
<AxesSubplot:xlabel='UrbanPop', ylabel='Rape'>
```

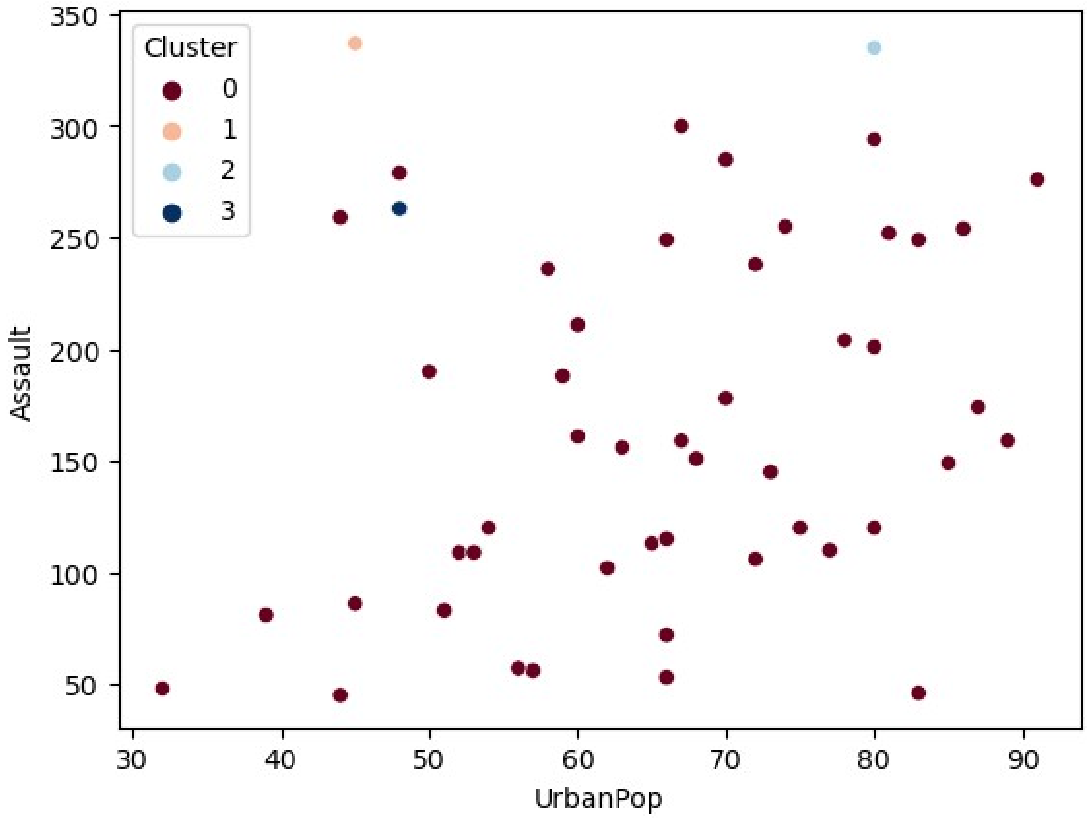



```
sns.scatterplot(x = 'UrbanPop',y = 'Murder', hue =
'Cluster',data=X,palette='RdBu')

<AxesSubplot:xlabel='UrbanPop', ylabel='Murder'>
```




```
sns.scatterplot(x = 'UrbanPop',y = 'Assault', hue =
'Cluster',data=X,palette='RdBu')
```

```
<AxesSubplot:xlabel='UrbanPop', ylabel='Assault'>
```




```
##### performed the Kmeans and Hierarchical clustering.
#### number of clusters are formed is "4" using elbow method.
#### From EDA gained the insughts that is "high crime rate is in more population cities.
#### outcome from the both the models is "we have to focus on the more populated cities to decrease the crime_rate.
```