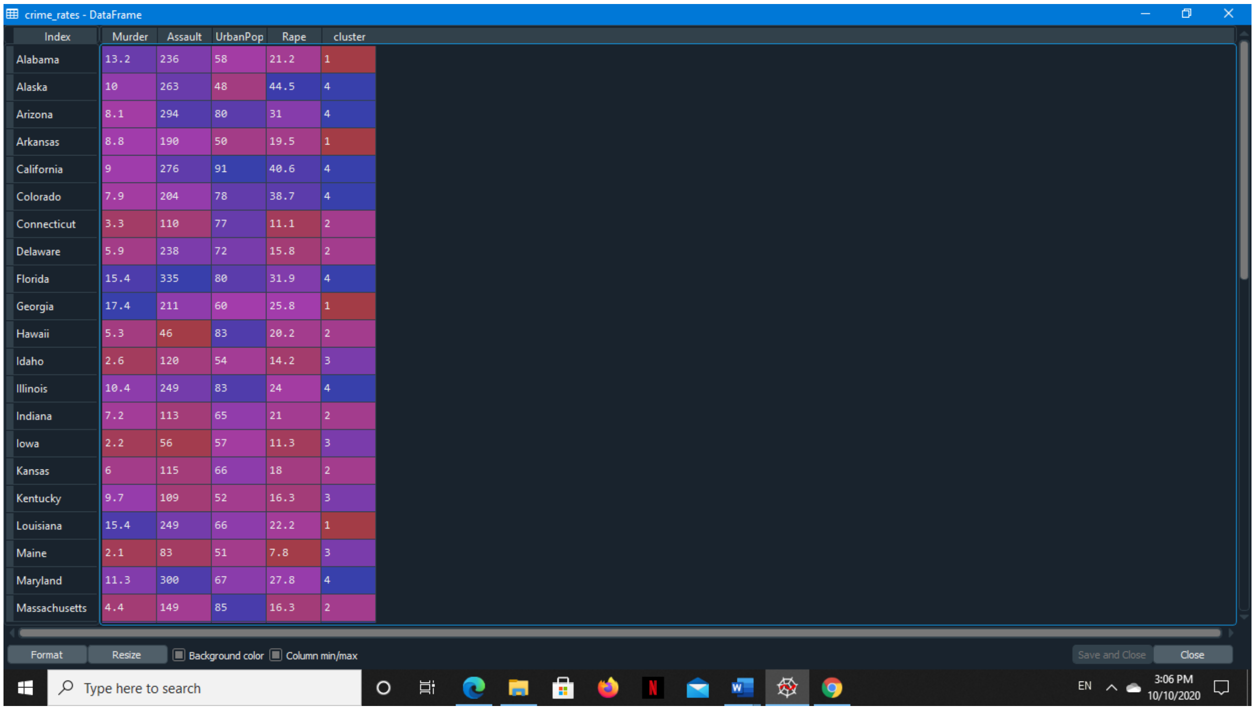
2.) Perform K-Means Clustering for the crime data and identify the number of clusters formed and draw inferences.

In this Assignment first we have to load the data set using some library and the data set is in the form of excel and load that dataset using Common Separated values (CSV).

```
import numpy as np # linear algebra

import pandas as pd
```

```
crime_rates=pd.read_csv("USArrests.csv", index_col=0)
```



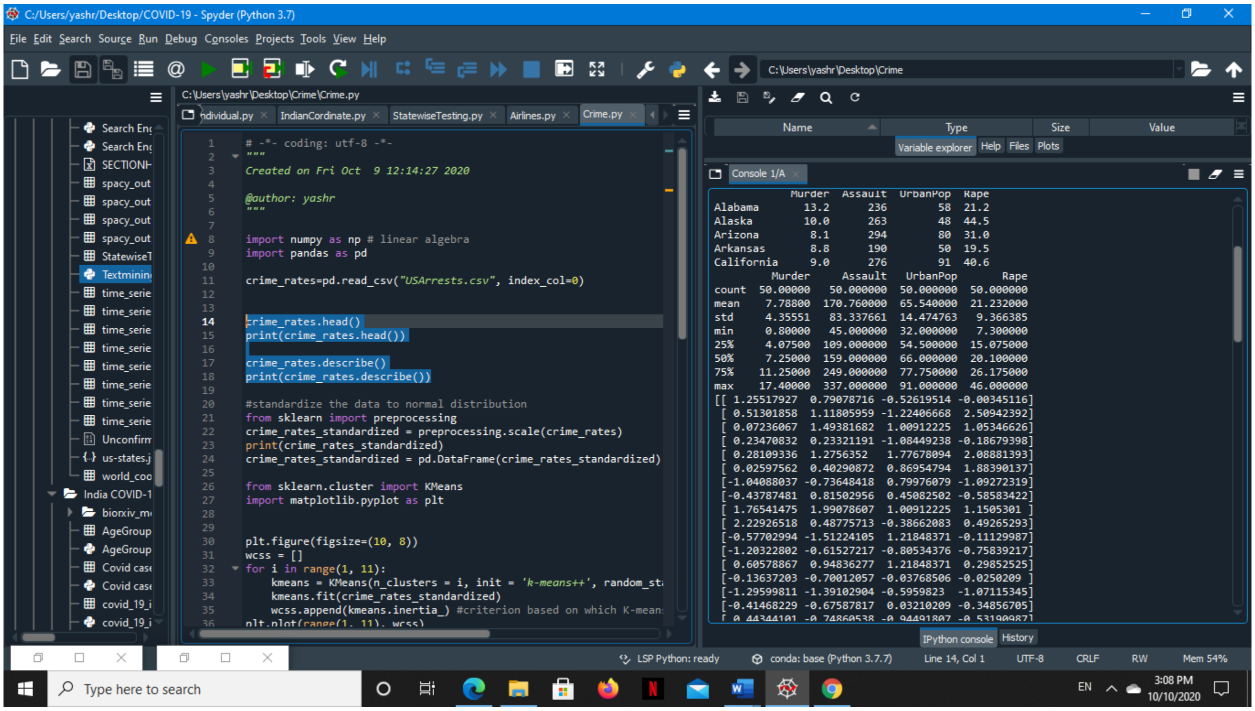
Index	Murder	Assault	UrbanPop	Rape	cluster
Alabama	13.2	236	58	21.2	1
Alaska	10	263	48	44.5	4
Arizona	8.1	294	80	31	4
Arkansas	8.8	190	50	19.5	1
California	9	276	91	40.6	4
Colorado	7.9	204	78	38.7	4
Connecticut	3.3	110	77	11.1	2
Delaware	5.9	238	72	15.8	2
Florida	15.4	335	80	31.9	4
Georgia	17.4	211	60	25.8	1
Hawaii	5.3	46	83	20.2	2
Idaho	2.6	120	54	14.2	3
Illinois	10.4	249	83	24	4
Indiana	7.2	113	65	21	2
Iowa	2.2	56	57	11.3	3
Kansas	6	115	66	18	2
Kentucky	9.7	109	52	16.3	3
Louisiana	15.4	249	66	22.2	1
Maine	2.1	83	51	7.8	3
Maryland	11.3	300	67	27.8	4
Massachusetts	4.4	149	85	16.3	2

Now we have to perform some data preprocessing technique to access and clean the data set.

```
crime_rates.head()

print(crime_rates.head())
```

crime_rates.describe()

print(crime_rates.describe())



```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct  9 12:14:27 2020

@author: yashr
"""

import numpy as np # linear algebra
import pandas as pd

crime_rates=pd.read_csv("USArrests.csv", index_col=0)


crime_rates.head()
print(crime_rates.head())

crime_rates.describe()
print(crime_rates.describe())

#standardize the data to normal distribution
from sklearn import preprocessing
crime_rates_standardized = preprocessing.scale(crime_rates)
print(crime_rates_standardized)
crime_rates_standardized = pd.DataFrame(crime_rates_standardized)

from sklearn.cluster import KMeans
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 8))
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_sta
    kmeans.fit(crime_rates_standardized)
    wcss.append(kmeans.inertia_) #criterion based on which K-mean
plt.plot(range(1, 11), wcss)
```

```
            Murder  Assault  UrbanPop  Rape
Alabama       13.2      236        58  21.2
Alaska        10.0      263        48  44.5
Arizona        8.1      294        80  31.0
Arkansas       8.8      190        50  19.5
California     9.0      276        91  40.6
         Murder     Assault   UrbanPop       Rape
count  50.00000   50.000000  50.000000  50.000000
mean    7.78800  170.760000  65.540000  21.232000
std     4.35551   83.337661  14.474763   9.366385
min     0.80000   45.000000  32.000000   7.300000
25%     4.07500  109.000000  54.500000  15.075000
50%     7.25000  159.000000  66.000000  20.100000
75%    11.25000  249.000000  77.750000  26.175000
max    17.40000  337.000000  91.000000  46.000000
[[ 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]
```

#standardize the data to normal distribution

from sklearn import preprocessing

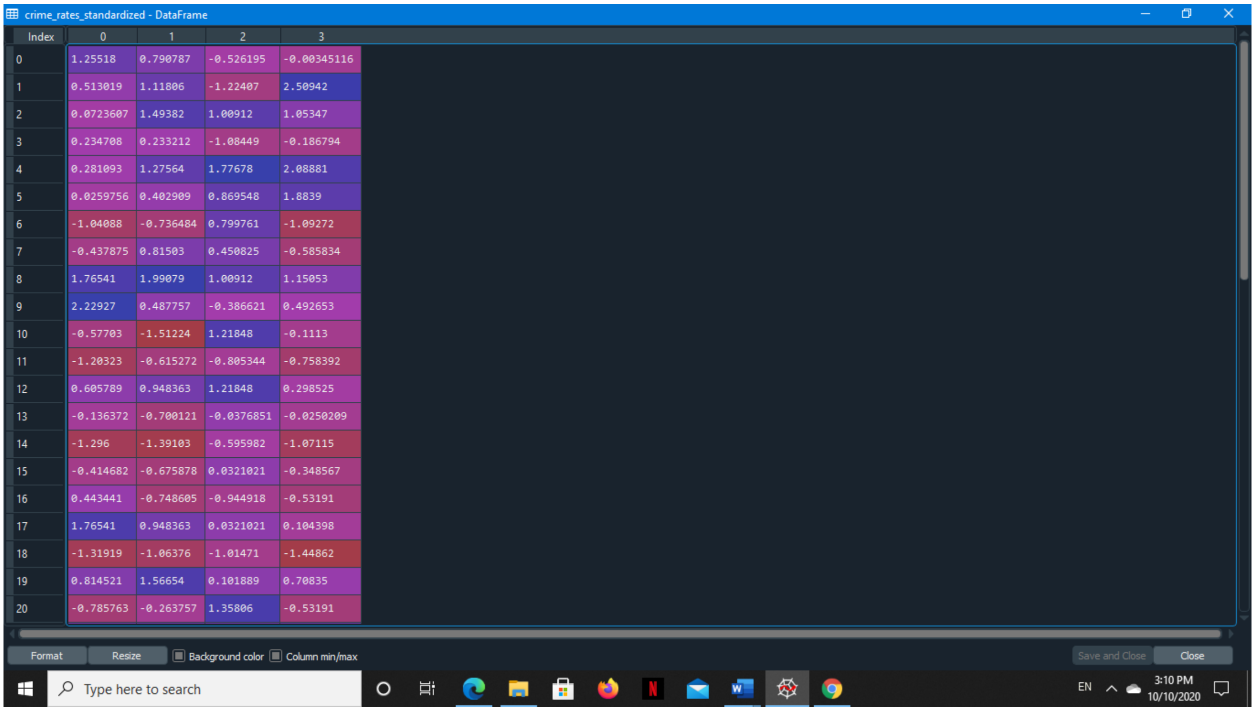
crime_rates_standardized = preprocessing.scale(crime_rates)

print(crime_rates_standardized)

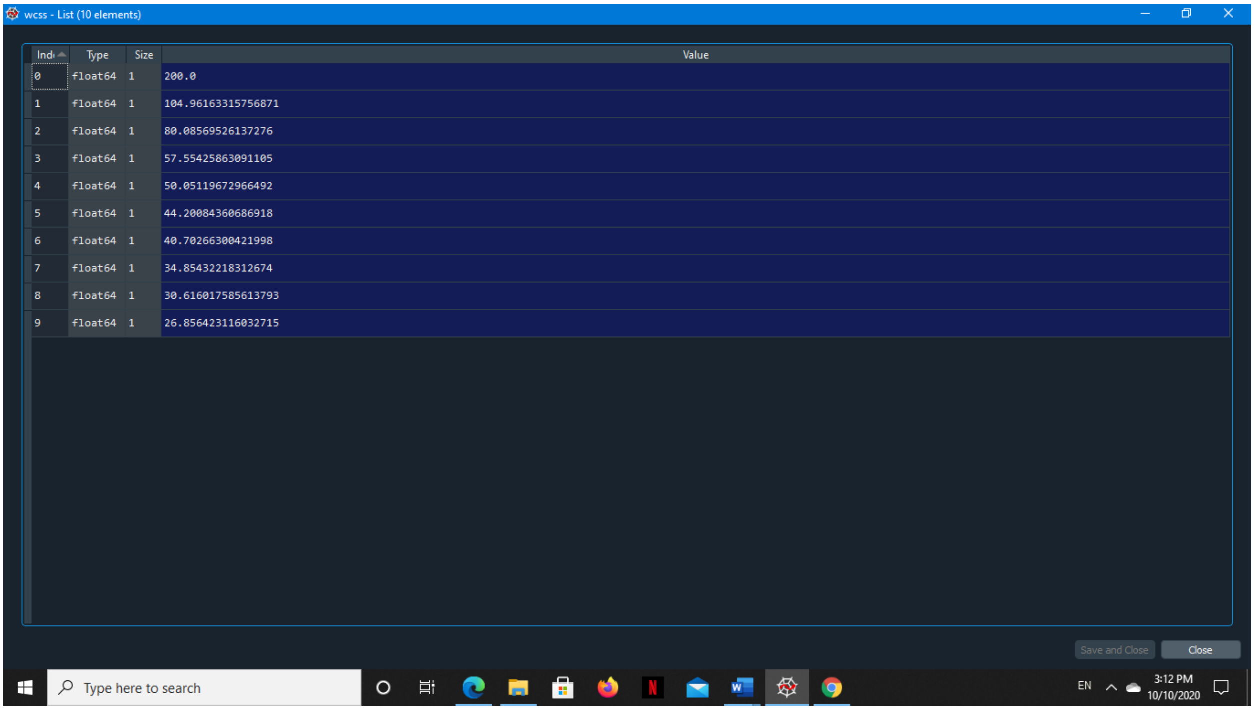
crime_rates_standardized = pd.DataFrame(crime_rates_standardized)

Now we are going to perform Clustering

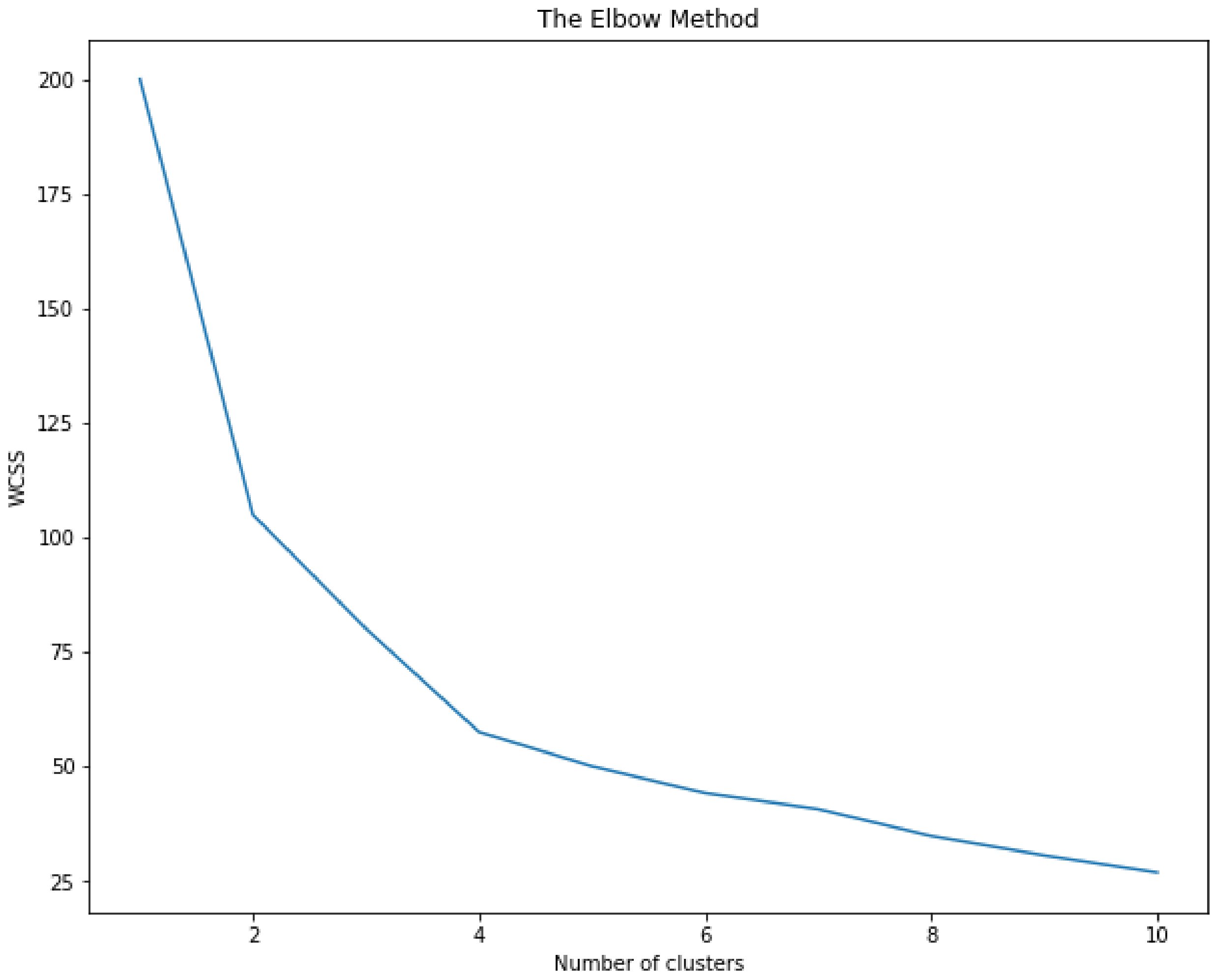
```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 8))
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(crime_rates_standardized)
    wcss.append(kmeans.inertia_) #criterion based on which K-means clustering works
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



Index	0	1	2	3
0	1.25518	0.790787	-0.526195	-0.00345116
1	0.513019	1.11806	-1.22407	2.50942
2	0.0723607	1.49382	1.00912	1.05347
3	0.234708	0.233212	-1.08449	-0.186794
4	0.281093	1.27564	1.77678	2.08881
5	0.0259756	0.402909	0.869548	1.8839
6	-1.04088	-0.736484	0.799761	-1.09272
7	-0.437875	0.81503	0.450825	-0.585834
8	1.76541	1.99079	1.00912	1.15053
9	2.22927	0.487757	-0.386621	0.492653
10	-0.57703	-1.51224	1.21848	-0.1113
11	-1.20323	-0.615272	-0.805344	-0.758392
12	0.605789	0.948363	1.21848	0.298525
13	-0.136372	-0.700121	-0.0376851	-0.0250209
14	-1.296	-1.39103	-0.595982	-1.07115
15	-0.414682	-0.675878	0.0321021	-0.348567
16	0.443441	-0.748605	-0.944918	-0.53191
17	1.76541	0.948363	0.0321021	0.104398
18	-1.31919	-1.06376	-1.01471	-1.44862
19	0.814521	1.56654	0.101889	0.70835
20	-0.785763	-0.263757	1.35806	-0.53191

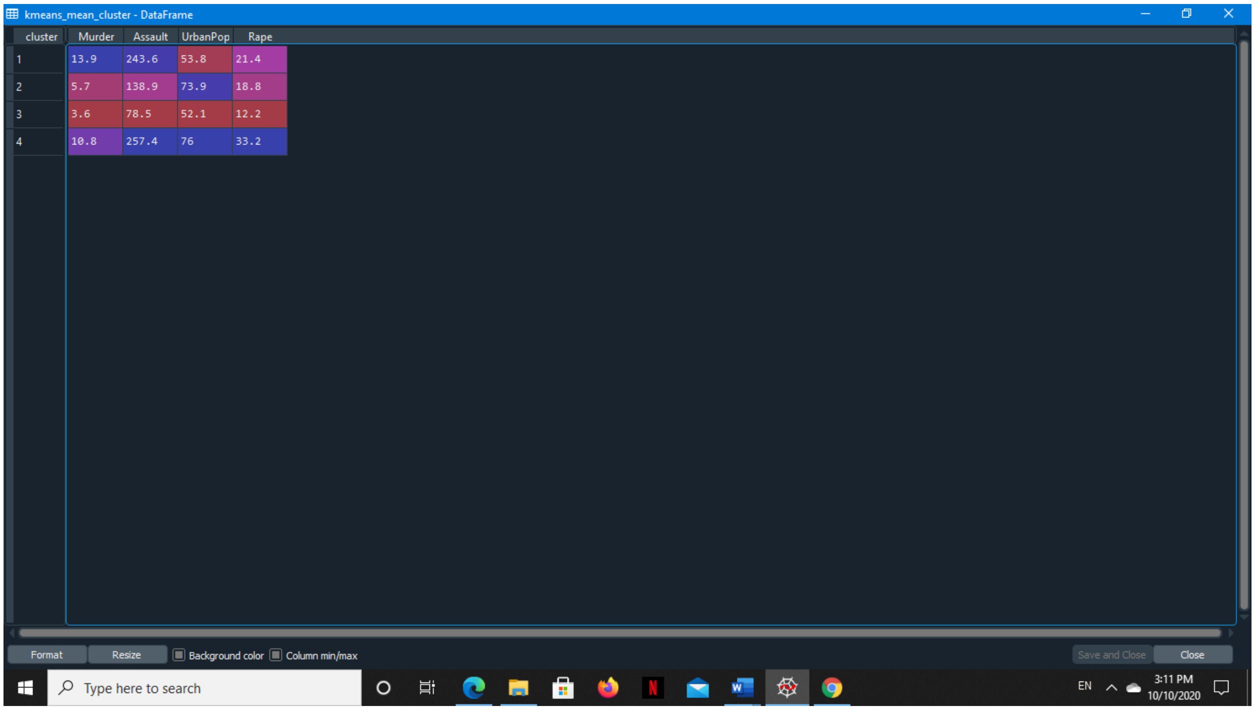


Index	Type	Size	Value
0	float64	1	200.0
1	float64	1	104.96163315756871
2	float64	1	80.08569526137276
3	float64	1	57.55425863091105
4	float64	1	50.05119672966492
5	float64	1	44.20084360686918
6	float64	1	40.70266300421998
7	float64	1	34.85432218312674
8	float64	1	30.616017585613793
9	float64	1	26.856423116032715

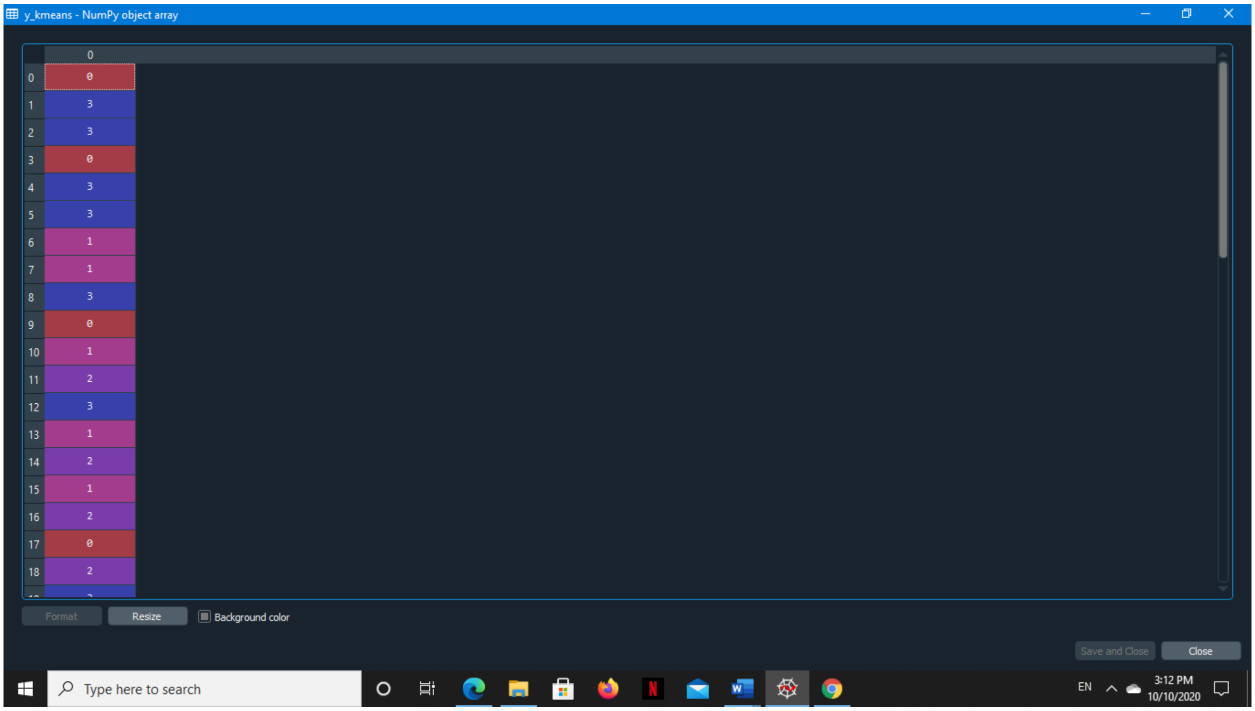


```
# Fitting K-Means to the dataset

kmeans = KMeans(n_clusters = 4, init = 'k-means++', random_state = 42)

y_kmeans = kmeans.fit_predict(crime_rates_standardized)


y_kmeans

print(y_kmeans)


#beginning of  the cluster numbering with 1 instead of 0

y_kmeans1=y_kmeans+1
```

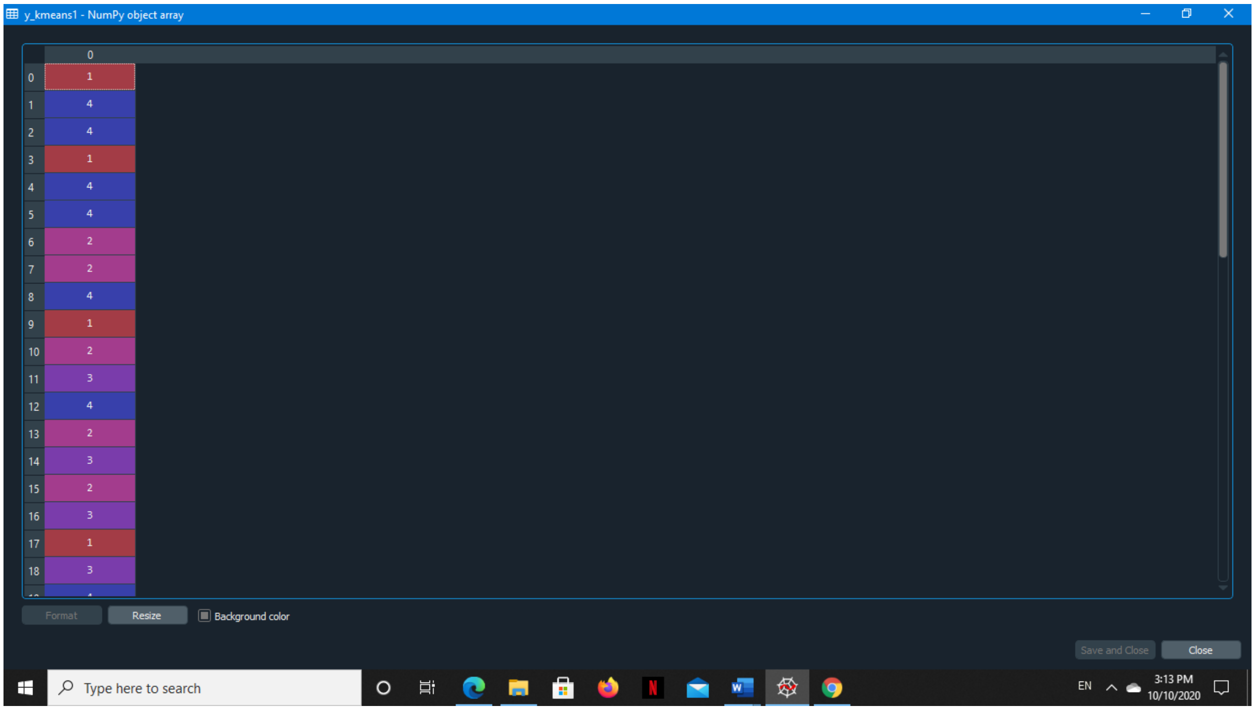
```
# New list called cluster

cluster = list(y_kmeans1)

# Adding cluster to our data set

crime_rates['cluster'] = cluster

#Mean of clusters 1 to 4

kmeans_mean_cluster = pd.DataFrame(round(crime_rates.groupby('cluster').mean(),1))

kmeans_mean_cluster

print(kmeans_mean_cluster)
```



cluster	Murder	Assault	UrbanPop	Rape
1	13.9	243.6	53.8	21.4
2	5.7	138.9	73.9	18.8
3	3.6	78.5	52.1	12.2
4	10.8	257.4	76	33.2



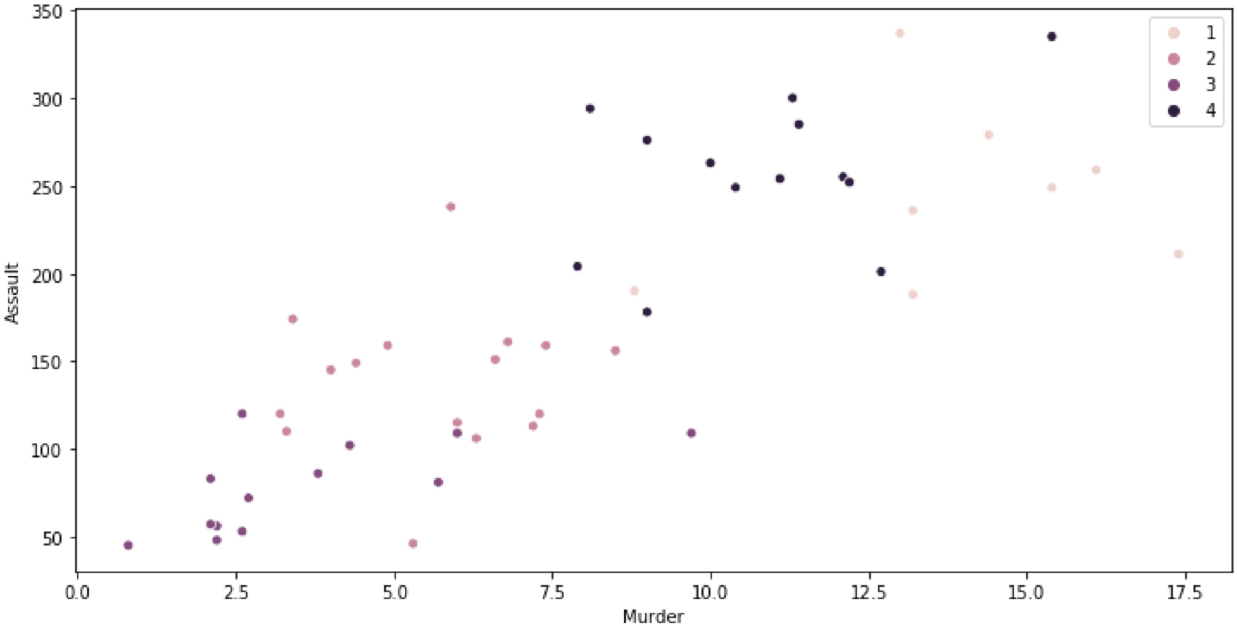
	0
0	0
1	3
2	3
3	0
4	3
5	3
6	1
7	1
8	3
9	0
10	1
11	2
12	3
13	1
14	2
15	1
16	2
17	0
18	2



	0
0	1
1	4
2	4
3	1
4	4
5	4
6	2
7	2
8	4
9	1
10	2
11	3
12	4
13	2
14	3
15	2
16	3
17	1
18	3

```
import seaborn as sns
```

```
plt.figure(figsize=(12,6))

sns.scatterplot(x=crime_rates['Murder'], y = crime_rates['Assault'],hue=y_kmeans1)


crime_rates[crime_rates['cluster']==1]

print(crime_rates[crime_rates['cluster']==1])
```


Assault
Murder