In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```python
A=pd.read_csv("C:/Users/HP/Downloads/Mall_Customers.csv")
A.head()
```

Out[2]:

	CustomerID	Genre	Age	Annual_Income_(k$)	Spending_Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [3]:
```python
A.drop(['Genre'],axis=1,inplace=True)
A
```

Out[3]:

	CustomerID	Age	Annual_Income_(k$)	Spending_Score
0	1	19	15	39
1	2	21	15	81
2	3	20	16	6
3	4	23	16	77
4	5	31	17	40
...	...	...	...	...
195	196	35	120	79
196	197	45	126	28
197	198	32	126	74
198	199	32	137	18
199	200	30	137	83

200 rows × 4 columns

In [4]:
```
#define standard scaler
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
#transform data
scaled = scaler.fit_transform(A)
print(scaled)
```

```
[[-1.7234121  -1.42456879 -1.73899919 -0.43480148]
 [-1.70609137 -1.28103541 -1.73899919  1.19570407]
 [-1.68877065 -1.3528021  -1.70082976 -1.71591298]
 [-1.67144992 -1.13750203 -1.70082976  1.04041783]
 [-1.6541292  -0.56336851 -1.66266033 -0.39597992]
 [-1.63680847 -1.20926872 -1.66266033  1.00159627]
 [-1.61948775 -0.27630176 -1.62449091 -1.71591298]
 [-1.60216702 -1.13750203 -1.62449091  1.70038436]
 [-1.5848463   1.80493225 -1.58632148 -1.83237767]
 [-1.56752558 -0.6351352  -1.58632148  0.84631002]
 [-1.55020485  2.02023231 -1.58632148 -1.4053405 ]
 [-1.53288413 -0.27630176 -1.58632148  1.89449216]
 [-1.5155634   1.37433211 -1.54815205 -1.36651894]
 [-1.49824268 -1.06573534 -1.54815205  1.04041783]
 [-1.48092195 -0.13276838 -1.54815205 -1.44416206]
 [-1.46360123 -1.20926872 -1.54815205  1.11806095]
 [-1.4462805  -0.27630176 -1.50998262 -0.59008772]
 [-1.42895978 -1.3528021  -1.50998262  0.61338066]
 [-1.41163905  0.94373197 -1.43364376 -0.82301709]
```

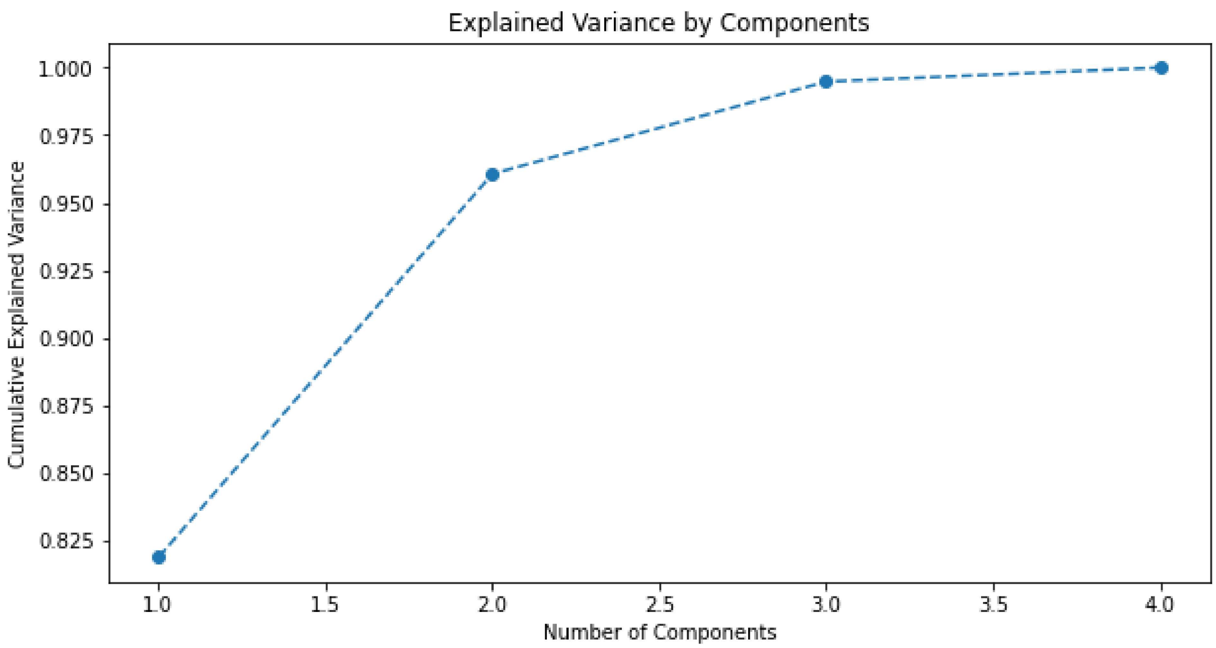
In [5]:
```
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
```

In [6]:
```
pc=PCA()
pc.fit(A)
pc.explained_variance_ratio_
pc.explained_variance_ratio_.cumsum()
```

Out[6]:
```
array([0.81899053, 0.96068389, 0.99481231, 1.        ])
```

In [7]:
```
plt.figure(figsize=(10,5))
plt.plot(range(1,5),pc.explained_variance_ratio_.cumsum(),marker='o',linestyle='-
plt.title('Explained Variance by Components')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
```

Out[7]:
```
Text(0, 0.5, 'Cumulative Explained Variance')
```



In [8]:
```
pc=PCA(n_components=2)
pc.fit(A)
pc.explained_variance_ratio_
pc.explained_variance_ratio_.cumsum()
```

Out[8]:
```
array([0.81899053, 0.96068389])
```

In [9]:
```
df=pc.transform(A)
print(df)
```

```
[[-1.09384203e+02  5.47772303e+00]
 [-1.08203115e+02 -3.49325311e+01]
 [-1.07376109e+02  3.78413764e+01]
 [-1.06007616e+02 -3.05622516e+01]
 [-1.04980136e+02  7.29722069e+00]
 [-1.03774532e+02 -2.98007906e+01]
 [-1.02996298e+02  4.13232718e+01]
 [-1.01426552e+02 -4.70708699e+01]
 [-1.00952459e+02  5.09230257e+01]
 [-9.93803318e+01 -2.40341569e+01]
 [-9.90709231e+01  4.09171508e+01]
 [-9.74038927e+01 -4.91519408e+01]
 [-9.67775993e+01  3.78943584e+01]
 [-9.52521231e+01 -3.02503511e+01]
 [-9.48419680e+01  3.50267217e+01]
 [-9.34012477e+01 -3.26459773e+01]
 [-9.24497381e+01  1.31746134e+01]
 [-9.12422731e+01 -2.04357200e+01]
 [-8.99474604e+01  2.29449767e+01]
```

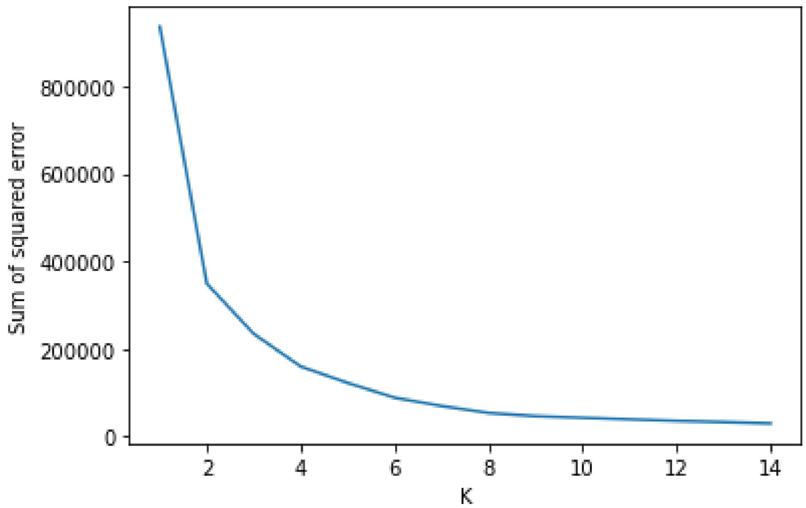
In [10]:
```
from sklearn.cluster import KMeans
sse=[]
Kmeans=range(1,15)
for k in  (Kmeans):
    km=KMeans(n_clusters=k)
    km.fit(df)
    sse.append(km.inertia_)
    print(sse)
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWar
ning: KMeans is known to have a memory leak on Windows with MKL, when there are
less chunks than available threads. You can avoid it by setting the environment
variable OMP_NUM_THREADS=1.
  warnings.warn(

[937111.3767673883]
[937111.3767673883, 349085.5582349874]
[937111.3767673883, 349085.5582349874, 234127.1293450116]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596, 42212.563183440136]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596, 42212.563183440136, 38534.38533089137]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596, 42212.563183440136, 38534.38533089137, 35049.71844414756]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596, 42212.563183440136, 38534.38533089137, 35049.71844414756, 3237
1.733303339537]
[937111.3767673883, 349085.5582349874, 234127.1293450116, 159578.12984974362, 1
21982.05019190397, 88087.06076189269, 69054.84353621714, 53145.22976830189, 458
78.096626255596, 42212.563183440136, 38534.38533089137, 35049.71844414756, 3237
1.733303339537, 29169.223009163215]
```

In [11]:
```
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(Kmeans,sse)
```

Out[11]:
```
[<matplotlib.lines.Line2D at 0x1f4ca910e50>]
```



In [12]:
```
km=KMeans(n_clusters=4)
y_predicted=km.fit_predict(df)
y_predicted
```

Out[12]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2, 0, 2, 0, 3, 0, 3, 0,
       3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
       3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
       3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0, 3, 0,
       3, 0])
```

In [13]:
```
km.cluster_centers_
```

Out[13]:
```
array([[ 66.99373894, -31.97145841],
       [-75.76706574,  -0.40743488],
       [ -7.36859761,   1.62813171],
       [ 68.90626218,  32.43166601]])
```

In [14]:
```
A2=np.transpose(df)
pca1=A2[0]
pca2=A2[1]
```

In [15]:
```
df_plot=pd.DataFrame()
df_plot['pca1']=np.transpose(pca1)
df_plot['pca2']=np.transpose(pca2)
df_plot['cluster']=y_predicted
df_plot
```

Out[15]:

	pca1	pca2	cluster
0	-109.384203	5.477723	1
1	-108.203115	-34.932531	1
2	-107.376109	37.841376	1
3	-106.007616	-30.562252	1
4	-104.980136	7.297221	1
...	...	...	...
195	111.652786	-27.964063	0
196	114.615565	24.018448	3
197	115.911268	-23.730584	0
198	120.939693	30.859322	3
199	122.297518	-32.853688	0

200 rows × 3 columns

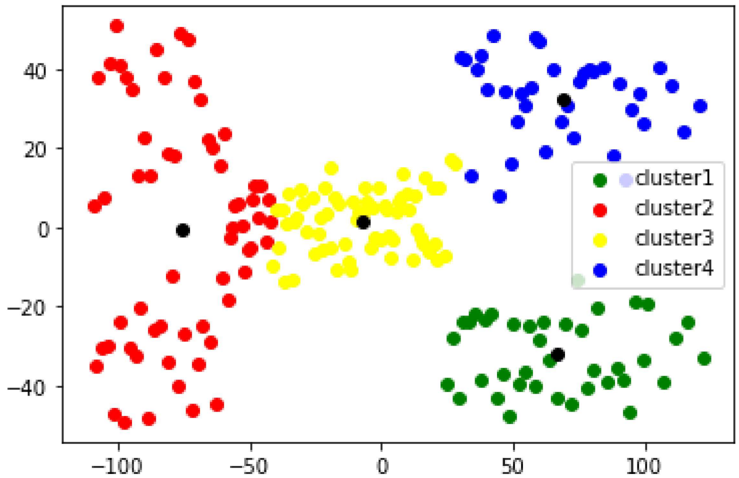
In [16]:
```
df_plot1=df_plot[df_plot.cluster==0]
df_plot2=df_plot[df_plot.cluster==1]
df_plot3=df_plot[df_plot.cluster==2]
df_plot4=df_plot[df_plot.cluster==3]
```

In [17]:
```
df_plot1['pca1']
```

Out[17]:
```
123     25.168475
125     27.355968
127     29.657038
129     31.361296
131     33.181251
133     35.435388
135     37.794296
137     39.502557
139     41.712185
141     44.103614
143     46.297512
145     48.621497
147     50.270578
149     52.599366
151     54.382536
153     56.134246
155     58.111205
157     59.846123
159     61.638620
161     63.945012
163     66.641879
165     69.950682
167     72.335447
169     74.361933
171     76.291181
173     78.183384
175     80.412465
177     82.142580
179     86.101769
181     89.550785
183     91.815590
185     94.103591
187     96.564000
189     99.272335
191    101.014978
193    107.032888
195    111.652786
197    115.911268
199    122.297518
Name: pca1, dtype: float64
```

In [18]:
```
plt.scatter(df_plot1['pca1'],df_plot1['pca2'],color='green',label='cluster1')
plt.scatter(df_plot2['pca1'],df_plot2['pca2'],color='red',label='cluster2')
plt.scatter(df_plot3['pca1'],df_plot3['pca2'],color='yellow',label='cluster3')
plt.scatter(df_plot4['pca1'],df_plot4['pca2'],color='blue',label='cluster4')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='black')
plt.legend()
```

Out[18]:
```
<matplotlib.legend.Legend at 0x1f4ca9e7af0>
```



In [ ]: