Name : Arul kumar ARK

Roll No. : 22522103

Lab : 11

```
                                Shapping Mall Customer Segmentation Using Clustering
```

Step : 1

In [1]:

```
import pandas as pd
```

In [2]:

```
data = pd.read_csv('Mall_Customers.csv')
```

In [3]:

```
data.head()
```

Out[3]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
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 [4]:

```
data.shape
```

Out[4]:

```
(200, 5)
```

In [5]:

```
data.columns
```

Out[5]:

```
Index(['CustomerID', 'Genre', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

In [6]:

```
data.info
```

Out[6]:

```
<bound method DataFrame.info of      CustomerID   Genre  Age  Annual Income
(k$)  Spending Score (1-100)
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
..          ...     ...  ...                ...                     ...
195         196  Female   35                120                      79
196         197  Female   45                126                      28
197         198    Male   32                126                      74
198         199    Male   32                137                      18
199         200    Male   30                137                      83

[200 rows x 5 columns]>
```

In [7]:

```
data.dtypes
```

Out[7]:

```
CustomerID                 int64
Genre                     object
Age                        int64
Annual Income (k$)         int64
Spending Score (1-100)     int64
dtype: object
```

In [8]:

```
data.value_counts
```

Out[8]:

```
<bound method DataFrame.value_counts of      CustomerID   Genre  Age  Annual
Income (k$)  Spending Score (1-100)
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
..          ...     ...  ...                ...                     ...
195         196  Female   35                120                      79
196         197  Female   45                126                      28
197         198    Male   32                126                      74
198         199    Male   32                137                      18
199         200    Male   30                137                      83

[200 rows x 5 columns]>
```

Step : 2

In [9]:

```
from sklearn.preprocessing import LabelEncoder
```

In [10]:

```
label = LabelEncoder()
```

In [11]:

```
data['Genre'] = label.fit_transform(data['Genre'])
```

In [12]:

```
data['Genre']
```

Out[12]:

```
0      1
1      1
2      0
3      0
4      0
      ..
195    0
196    0
197    1
198    1
199    1
Name: Genre, Length: 200, dtype: int32
```

In [13]:

```
data.tail()
```

Out[13]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
195	196	0	35	120	79
196	197	0	45	126	28
197	198	1	32	126	74
198	199	1	32	137	18
199	200	1	30	137	83

Step : 3

In [14]:

```
data.describe()
```

Out[14]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	0.440000	38.850000	60.560000	50.200000
std	57.879185	0.497633	13.969007	26.264721	25.823522
min	1.000000	0.000000	18.000000	15.000000	1.000000
25%	50.750000	0.000000	28.750000	41.500000	34.750000
50%	100.500000	0.000000	36.000000	61.500000	50.000000
75%	150.250000	1.000000	49.000000	78.000000	73.000000
max	200.000000	1.000000	70.000000	137.000000	99.000000

In [15]:

```
data.describe().var()
```

Out[15]:

```
CustomerID                5151.857673
Genre                     4979.211557
Age                       3660.413623
Annual Income (k$)        3845.287651
Spending Score (1-100)    3773.026460
dtype: float64
```

Step : 4

In [16]:

```
data.describe().skew()
```

Out[16]:

```
CustomerID                0.069830
Genre                     2.828231
Age                       2.376316
Annual Income (k$)        1.291233
Spending Score (1-100)    1.649639
dtype: float64
```

Step : 5

In [17]:

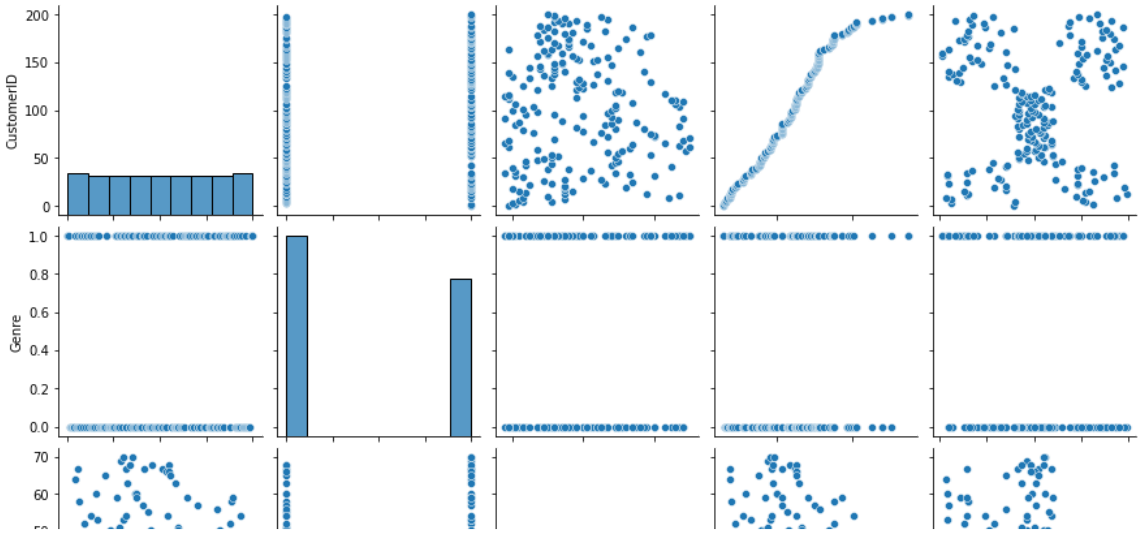
```
import numpy as np
import seaborn as sns
```

In [34]:

```
sns.pairplot(data)
```

Out[34]:

```
<seaborn.axisgrid.PairGrid at 0x21911eb9e50>
```

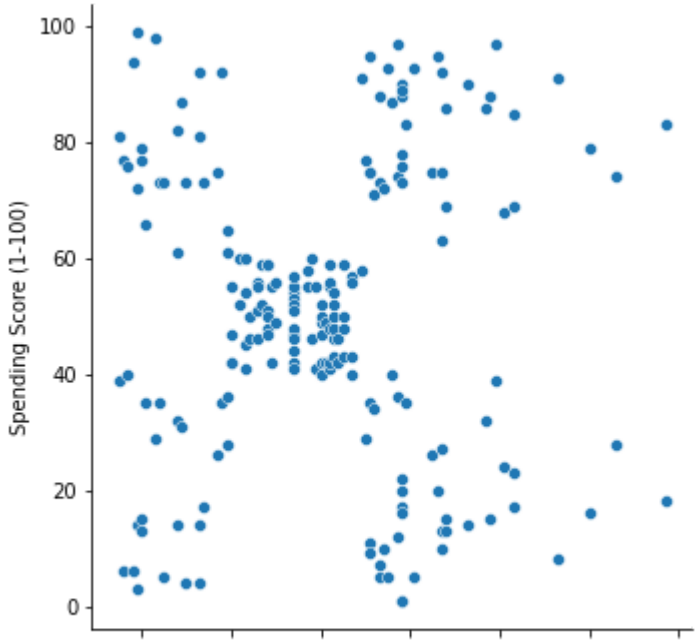


In [35]:

```
data.drop('CustomerID',axis = 1,inplace=True)
```

In [36]:

```
sns.pairplot(data, x_vars=['Annual Income (k$)'], y_vars=['Spending Score (1-100)'], height
```



step : 6

In [31]:

```
from sklearn.cluster import KMeans
```

In [37]:

```
kmeans = KMeans(n_clusters=5, random_state=42)
kmeans.fit(data)
labels = kmeans.labels_
centroids = kmeans.cluster_centers_
print("Cluster Labels:")
print(labels)
print("\nCluster Centroids:")
print(centroids)
```

```
Cluster Labels:
[0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0
 4 0 4 0 4 0 3 0 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 1 2 1 3 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1
 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1]

Cluster Centroids:
[[ 0.39130435 45.2173913  26.30434783 20.91304348]
 [ 0.46153846 32.69230769 86.53846154 82.12820513]
 [ 0.51351351 40.32432432 87.43243243 18.18918919]
 [ 0.41772152 43.12658228 54.82278481 49.83544304]
 [ 0.40909091 25.27272727 25.72727273 79.36363636]]
```

In [48]:

```
data['labels'] = labels
```

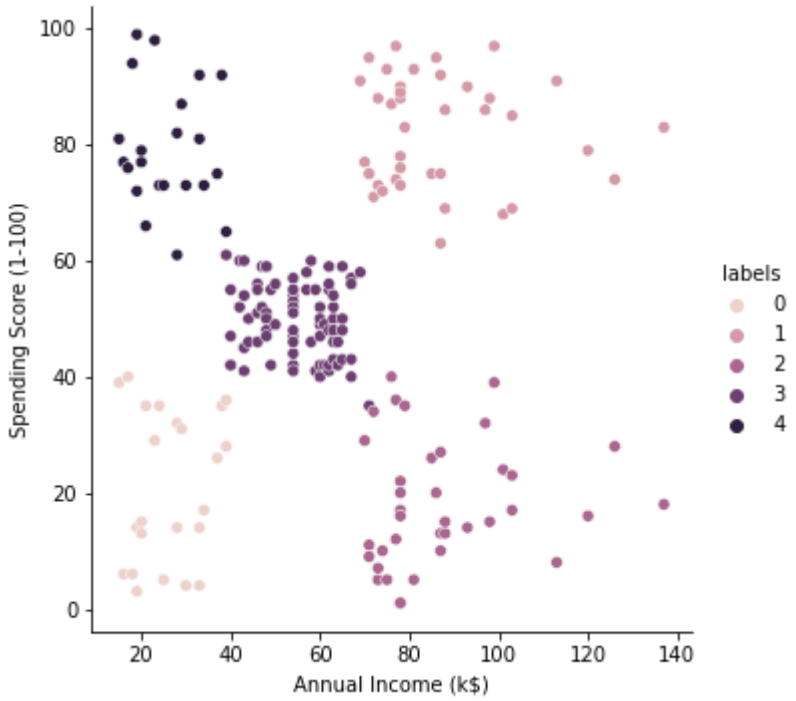
Step : 7

In [50]:

```
sns.pairplot(data, x_vars=['Annual Income (k$)'], y_vars=['Spending Score (1-100)'], hue='l
```

Out[50]:

```
<seaborn.axisgrid.PairGrid at 0x21914ba9a90>
```



Step : 8

In [53]:

```
kmeans2 = KMeans(n_clusters = 5, init='k-means++')
kmeans2.fit(data)
```

Out[53]:

```
KMeans(n_clusters=5)
```

In [54]:

```
pred = kmeans2.predict(data)
```

In [64]:

```
pred
```

Out[64]:

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

In [55]:

```
frame = pd.DataFrame(data)
frame['cluster'] = pred
frame.cluster.value_counts()
```

Out[55]:

```
4    77
2    39
0    36
3    25
1    23
Name: cluster, dtype: int64
```

In [60]:

```
frame.head()
```

Out[60]:

	Genre	Age	Annual Income (k$)	Spending Score (1-100)	labels	cluster
0	1	19	15	39	0	1
1	1	21	15	81	4	3
2	0	20	16	6	0	1
3	0	23	16	77	4	3
4	0	31	17	40	0	1

In [57]:

```
C0 = data[data['cluster'] == 0]
C1 = data[data['cluster'] == 1]
C2 = data[data['cluster'] == 2]
C3 = data[data['cluster'] == 3]
C4 = data[data['cluster'] == 4]
```

In [66]:

```
import statistics as ss
```

C0

In [67]:

```
print('Average Age : ',C0['Age'].mean())
print('Average Annual Income : ',C0['Annual Income (k$)'].mean())
print('Deviation of the mean for annual Income : ',ss.stdev(C0['Annual Income (k$)']))
print('No. of Customers ie shape :' ,C0.shape)
print('From those Customers We have',C0.Genre.value_counts()[1],'male and',C0.Genre.value_c
```

```
Average Age :  40.666666666666664
Average Annual Income :  87.75
Deviation of the mean for annual Income :  16.387059354433127
No. of Customers ie shape : (36, 6)
From those Customers We have 19 male and 17 female
```

C1

In [69]:

```
print('Average Age : ',C1['Age'].mean())
print('Average Annual Income : ',C1['Annual Income (k$)'].mean())
print('Deviation of the mean for annual Income : ',ss.stdev(C1['Annual Income (k$)']))
print('No. of Customers ie shape :' ,C1.shape)
print('From those Customers We have',C1.Genre.value_counts()[1],'male and',C1.Genre.value_c
```

```
Average Age :  45.21739130434783
Average Annual Income :  26.304347826086957
Deviation of the mean for annual Income :  7.893811054517766
No. of Customers ie shape : (23, 6)
From those Customers We have 9 male and 14 female
```

C2

In [70]:

```
print('Average Age : ',C2['Age'].mean())
print('Average Annual Income : ',C2['Annual Income (k$)'].mean())
print('Deviation of the mean for annual Income : ',ss.stdev(C2['Annual Income (k$)']))
print('No. of Customers ie shape :' ,C2.shape)
print('From those Customers We have',C2.Genre.value_counts()[1],'male and',C2.Genre.value_c
```

```
Average Age :  32.69230769230769
Average Annual Income :  86.53846153846153
Deviation of the mean for annual Income :  16.312484972924967
No. of Customers ie shape : (39, 6)
From those Customers We have 18 male and 21 female
```

C3

In [73]:

```
print('Average Age : ',C3['Age'].mean())
print('Average Annual Income : ',C3['Annual Income (k$)'].mean())
print('Deviation of the mean for annual Income : ',ss.stdev(C3['Annual Income (k$)']))
print('No. of Customers ie shape :' ,C3.shape)
print('From those Customers We have',C3.Genre.value_counts()[1],'male and',C3.Genre.value_c
```

```
Average Age :  24.96
Average Annual Income :  28.04
Deviation of the mean for annual Income :  9.654359982239457
No. of Customers ie shape : (25, 6)
From those Customers We have 11 male and 14 female
```

C4

In [72]:

```
print('Average Age : ',C4['Age'].mean())
print('Average Annual Income : ',C4['Annual Income (k$)'].mean())
print('Deviation of the mean for annual Income : ',ss.stdev(C4['Annual Income (k$)']))
print('No. of Customers ie shape :' ,C4.shape)
print('From those Customers We have',C4.Genre.value_counts()[1],'male and',C4.Genre.value_c
```

```
Average Age :  43.72727272727273
Average Annual Income :  55.48051948051948
Deviation of the mean for annual Income :  8.742832236527411
No. of Customers ie shape : (77, 6)
From those Customers We have 31 male and 46 female
```

Step : 9

In [75]:

```
SSE = []
for clust in range(1,20):
    KM = KMeans(n_clusters= clust, init='k-means++')
    KM = KM.fit(data)
    SSE.append(KM.inertia_)
```

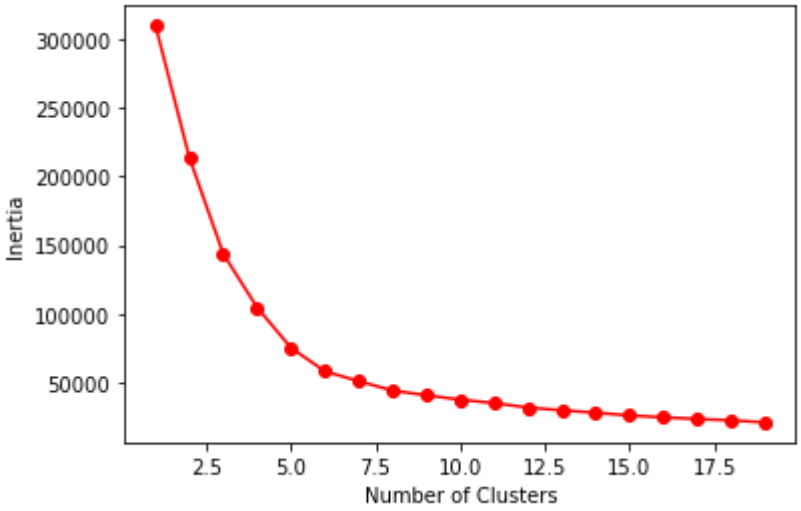
```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: U
serWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting th
e environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

In [77]:

```
import matplotlib.pyplot as plt
plt.plot(np.arange(1,20), SSE,'ro-')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
```

Out[77]:

```
Text(0, 0.5, 'Inertia')
```



Step : 10

In [78]:

```
from sklearn.decomposition import PCA
```

In [79]:

```
pca = PCA(n_components=2)
_PCA = pca.fit_transform(data)
```

In [80]:

```
PCA_Components = pd.DataFrame(_PCA)
PCA_Components
```

Out[80]:

	0	1
0	-31.456730	-33.319342
1	1.510195	-56.835415
2	-57.250266	-13.793627
3	-1.466048	-53.512608
4	-31.793833	-30.715663
...	...	...
195	57.936771	31.800193
196	19.022211	66.743017
197	58.009655	39.127062
198	19.924326	79.682895
199	71.889871	42.774752

200 rows × 2 columns

In [81]:

```
KM1 = KMeans(n_clusters=5)
KM1.fit(PCA_Components)
KM1.cluster_centers_
KM1.labels_
```

Out[81]:

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

Step : 11

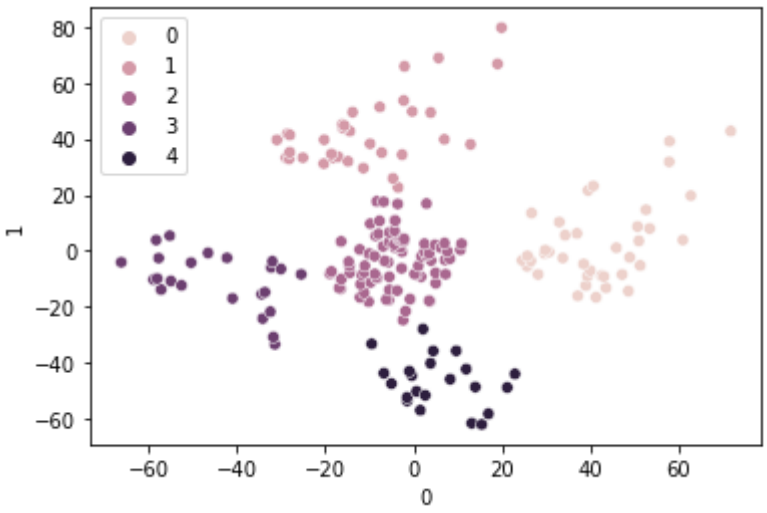
In [82]:

```
sns.scatterplot(PCA_Components[0], PCA_Components[1], hue=KM1.labels_)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[82]:

```
<AxesSubplot:xlabel='0', ylabel='1'>
```



Step : 12

In [83]:

```
from sklearn.cluster import MeanShift, AgglomerativeClustering
```

In [84]:

```
MS = MeanShift(bandwidth = 50)
MS.fit(PCA_Components)
MS.cluster_centers_
```

Out[84]:

```
array([[ 0.4060829 , -4.11099779]])
```

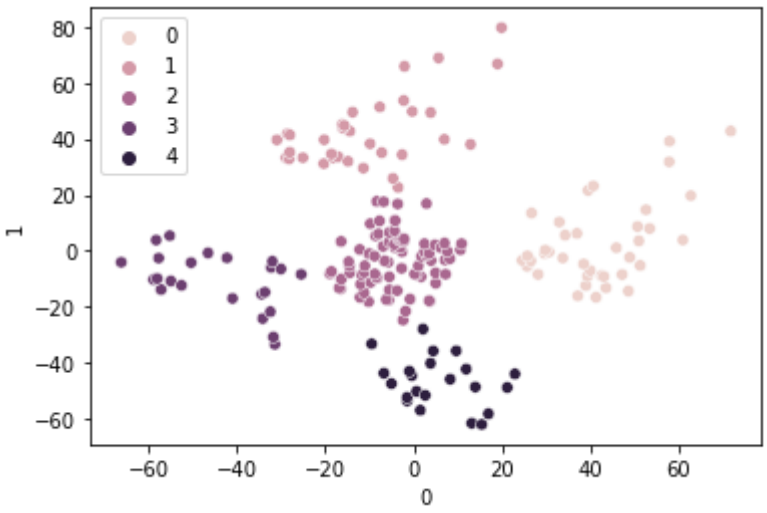
In [85]:

```
sns.scatterplot(PCA_Components[0], PCA_Components[1], hue=KM1.labels_)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[85]:

```
<AxesSubplot:xlabel='0', ylabel='1'>
```



Step : 13

In [86]:

```
AC = AgglomerativeClustering(n_clusters = 5, linkage='ward',compute_full_tree=True)
AC.fit(data)
```

Out[86]:

```
AgglomerativeClustering(compute_full_tree=True, n_clusters=5)
```

In [87]:

```
AC.labels_
```

Out[87]:

```
array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
       4, 3, 4, 3, 4, 0, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 0,
       4, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 0, 2, 1, 2, 1, 2,
       0, 2, 1, 2, 1, 2, 1, 2, 1, 2, 0, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2], dtype=int64)
```

In [89]:

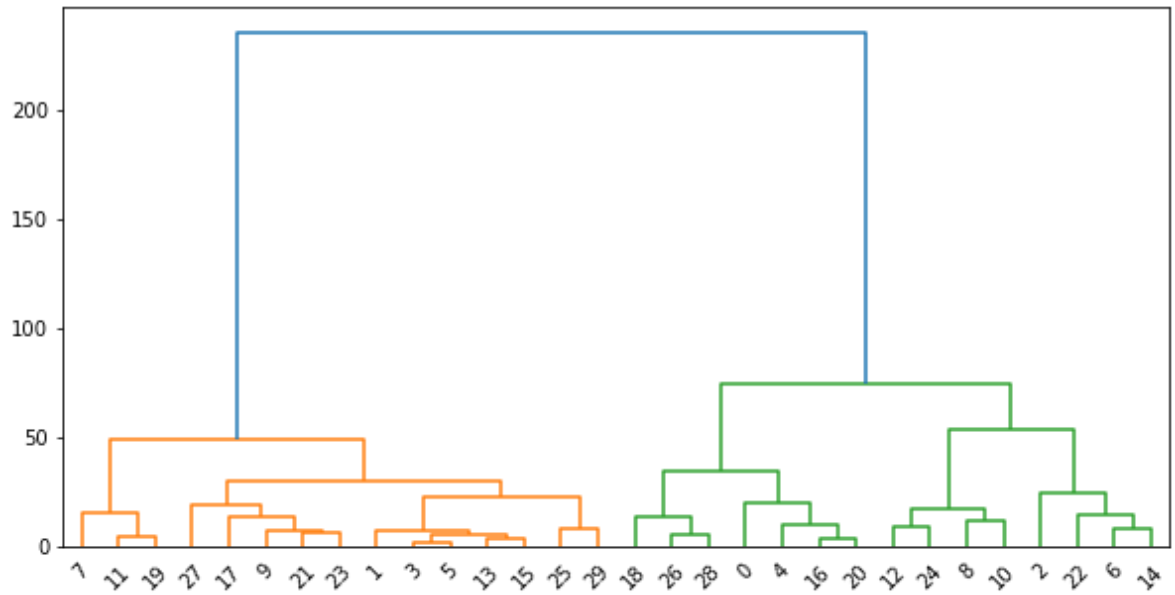
```
data['Cluster'] = AC.labels_
```

In [90]:

```
import scipy.cluster.hierarchy as sch
from scipy.cluster import hierarchy
```

In [91]:

```
Z= hierarchy.linkage(data[:30], 'ward')
plt.figure(figsize=(10,5))
dn = hierarchy.dendrogram(Z)
```



Step : 14

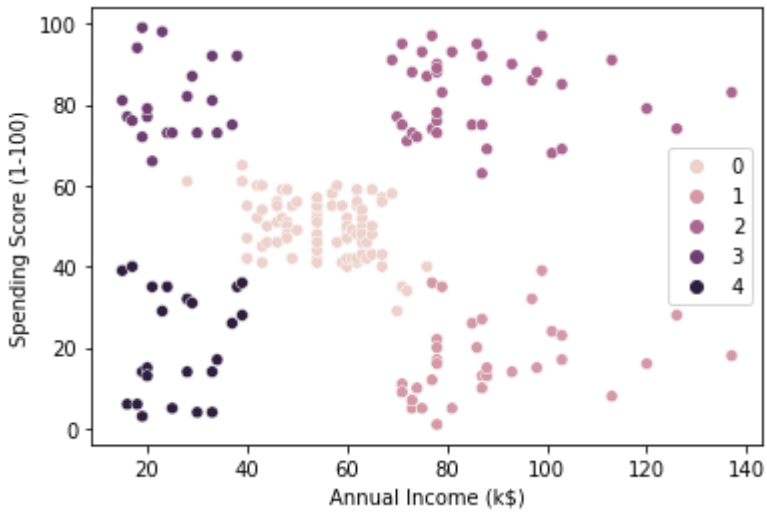
In [94]:

```
sns.scatterplot(data['Annual Income (k$)'], data['Spending Score (1-100)'], hue=AC.labels_)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[94]:

```
<AxesSubplot:xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



In [ ]: