**Institute of Information Technology (IIT)**

Jahangirnagar University



**Lab Report: 08**

Submitted by:

Name: Md. Shakil Hossain  
Roll No: 2023

Lab Date: 28/08/2023  
Submission Date: 03/09/2023

Import libraries

In [3]:

**import** numpy **as** np *# linear algebra*

**import** pandas **as** pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

**import** matplotlib.pyplot **as** plt *# for data visualization*

**import** seaborn **as** sns *# for statistical data visualization*

**%**matplotlib inline

In [6]:

df **=** pd.read\_csv('Mall\_Customers.csv')

Exploratory data analysis

In [7]:

df.shape

Out[7]:

(200, 5)

In [8]:

df.head()

Out[8]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 |

<class 'pandas.core.frame.DataFrame'> RangeIndex: 200 entries, 0 to 199

Data columns (total 5 columns):

# Column Non-Null Count Dtype

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  | CustomerID |  | 200 | non-null |  | int64 |
| 1 |  | Genre |  | 200 | non-null |  | object |
| 2 |  | Age |  | 200 | non-null |  | int64 |
| 3 |  | Annual Income | (k$) | 200 | non-null |  | int64 |

4 Spending Score (1-100) 200 non-null int64 dtypes: int64(4), object(1)

memory usage: 7.9+ KB

In [10]:

df.isnull().sum()

Out[10]:

CustomerID 0

Genre 0

Age 0

Annual Income (k$) 0

Spending Score (1-100) 0

dtype: int64

In [11]:

df.drop(['Genre'], axis**=**1, inplace**=True**)

In [12]:

df.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 200 entries, 0 to 199

Data columns (total 4 columns):

# Column Non-Null Count Dtype

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  | CustomerID |  | 200 | non-null |  | int64 |
| 1 |  | Age |  | 200 | non-null |  | int64 |
| 2 |  | Annual Income | (k$) | 200 | non-null |  | int64 |

3 Spending Score (1-100) 200 non-null int64 dtypes: int64(4)

memory usage: 6.4 KB

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

In [14]:

df['CustomerID'].unique()

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[14]: |  | | | | | | | | | | | |
| array([ 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, |
| 14, | 15, | 16, | 17, | 18, | 19, | 20, | 21, | 22, | 23, | 24, | 25, | 26, |
| 27, | 28, | 29, | 30, | 31, | 32, | 33, | 34, | 35, | 36, | 37, | 38, | 39, |
| 40, | 41, | 42, | 43, | 44, | 45, | 46, | 47, | 48, | 49, | 50, | 51, | 52, |
| 53, | 54, | 55, | 56, | 57, | 58, | 59, | 60, | 61, | 62, | 63, | 64, | 65, |
| 66, | 67, | 68, | 69, | 70, | 71, | 72, | 73, | 74, | 75, | 76, | 77, | 78, |
| 79, | 80, | 81, | 82, | 83, | 84, | 85, | 86, | 87, | 88, | 89, | 90, | 91, |
| 92, | 93, | 94, | 95, | 96, | 97, | 98, | 99, | 100, | 101, | 102, | 103, | 104, |
| 105, | 106, | 107, | 108, | 109, | 110, | 111, | 112, | 113, | 114, | 115, | 116, | 117, |
| 118, | 119, | 120, | 121, | 122, | 123, | 124, | 125, | 126, | 127, | 128, | 129, | 130, |
| 131, | 132, | 133, | 134, | 135, | 136, | 137, | 138, | 139, | 140, | 141, | 142, | 143, |
| 144, | 145, | 146, | 147, | 148, | 149, | 150, | 151, | 152, | 153, | 154, | 155, | 156, |
| 157, | 158, | 159, | 160, | 161, | 162, | 163, | 164, | 165, | 166, | 167, | 168, | 169, |
| 170, | 171, | 172, | 173, | 174, | 175, | 176, | 177, | 178, | 179, | 180, | 181, | 182, |
| 183, | 184, | 185, | 186, | 187, | 188, | 189, | 190, | 191, | 192, | 193, | 194, | 195, |
| 196, | 197, | 198, | 199, | 200], dtype=int64) | | | | | | | | |

In [15]:

len(df['CustomerID'].unique())

Out[15]:

200

array([19, 4,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 21, | 20, | 23, | 31, | 22, | 35, | 64, | 30, | 67, | 58, | 24, | 37, | 52, | 25, | 46, | 5 |
| 45, | 40, | 60, | 53, | 18, | 49, | 42, | 36, | 65, | 48, | 50, | 27, | 33, | 59, | 47, | 5 |
| 70, | 63, | 43, | 68, | 32, | 26, | 57, | 38, | 55, | 34, | 66, | 39, | 44, | 28, | 56, | 4 |

1,

1],

29,

69,

dtype=int64)

In [17]:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| len(df['Age'].unique()) |  | | | | | | | |
| Out[17]: |  |  |  |  |  |  |  |  |
| 51 |  |  |  |  |  |  |  |  |
| In [18]: |  |  |  |  |  |  |  |  |
| df['Annual Income (k$)'].unique() |  |  |  |  |  |  |  |  |
| Out[18]: |  |  |  |  |  |  |  |  |
| array([ 15, 16, 17, 18, 19, | 20, | 21, | 23, | 24, | 25, | 28, | 29, | 30, |
| 33, 34, 37, 38, 39, | 40, | 42, | 43, | 44, | 46, | 47, | 48, | 49, |
| 50, 54, 57, 58, 59, | 60, | 61, | 62, | 63, | 64, | 65, | 67, | 69, |
| 70, 71, 72, 73, 74, | 75, | 76, | 77, | 78, | 79, | 81, | 85, | 86, |
| 87, 88, 93, 97, 98, | 99, | 101, | 103, | 113, | 120, | 126, | 137], |  |

dtype=int64)

In [19]:

len(df['Annual Income (k$)'].unique())

Out[19]:

64

In [20]:

df.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 200 entries, 0 to 199

Data columns (total 4 columns):

# Column Non-Null Count Dtype

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  | CustomerID |  | 200 | non-null |  | int64 |
| 1 |  | Age |  | 200 | non-null |  | int64 |
| 2 |  | Annual Income | (k$) | 200 | non-null |  | int64 |

3 Spending Score (1-100) 200 non-null int64 dtypes: int64(4)

memory usage: 6.4 KB

df.head()

Out[21]:

|  |  |  |  |
| --- | --- | --- | --- |
| **CustomerID** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| **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 |

In [25]:

plt.figure(figsize**=**(10,6))

plt.scatter(df['Age'],df['Annual Income (k$)']) plt.xlabel('Number of Age')

plt.ylabel('number of Annual Income (k$)') plt.title('LIVE Income')

Out[25]:

Text(0.5, 1.0, 'LIVE Income')



df.head(2)

Out[26]:

|  |  |  |  |
| --- | --- | --- | --- |
| **CustomerID** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| **0** 1 | 19 | 15 | 39 |
| **1** 2 | 21 | 15 | 81 |

In [29]:

**from** sklearn.preprocessing **import** LabelEncoder le **=** LabelEncoder()

df['CustomerID'] **=** le.fit\_transform(df['CustomerID'])

In [30]:

y**=**df

cols **=** y.columns

**from** sklearn.preprocessing **import** MinMaxScaler ms **=** MinMaxScaler()

y **=** ms.fit\_transform(y)

y **=** pd.DataFrame(y, columns**=**[cols])

In [31]:

X **=** y.values

X[:5] *# Show first 5 records only*

Out[31]:

array([[0. , 0.01923077, 0. , 0.3877551 ],

[0.00502513, 0.05769231, 0. , 0.81632653],

|  |  |  |  |
| --- | --- | --- | --- |
| [0.01005025, | 0.03846154, | 0.00819672, | 0.05102041], |
| [0.01507538, | 0.09615385, | 0.00819672, | 0.7755102 ], |
| [0.0201005 , | 0.25 , | 0.01639344, | 0.39795918]]) |

**from** sklearn.cluster **import** KMeans clustering\_score **=** []

**for** i **in** range(1, 11):

kmeans **=** KMeans(n\_clusters **=** i, init **=** 'random', random\_state **=** 42) kmeans.fit(X)

clustering\_score.append(kmeans.inertia\_)

plt.figure(figsize**=**(10,6))

plt.plot(range(1, 11), clustering\_score)

plt.scatter(4,clustering\_score[3], s **=** 200, c **=** 'red', marker**=**'\*') plt.title('The Elbow Method')

plt.xlabel('No. of Clusters') plt.ylabel('Clustering Score') plt.show()

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by

setting the environment variable OMP\_NUM\_THREADS=1. warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

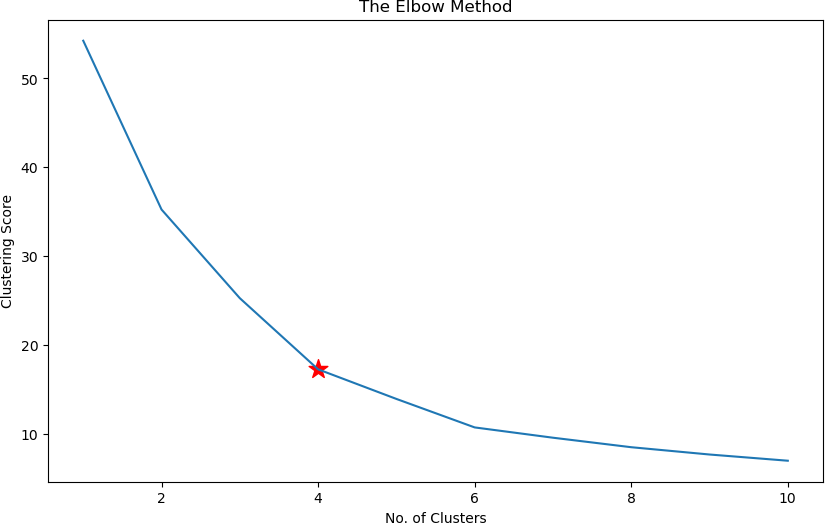
warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(



kmeans**=** KMeans(n\_clusters **=** 5, random\_state **=** 42)

*# Compute k-means clustering*

kmeans.fit(X)

*# Compute cluster centers and predict cluster index for each sample.*

pred **=** kmeans.predict(X) pred

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'aut o' in 1.4. Set the value of `n\_init` explicitly to suppress the warning

warnings.warn(

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:138 2: UserWarning: KMeans is known to have a memory leak on Windows with MK L, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

warnings.warn(

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Out[33]: |  | | | | | | | | | | | | | | | | | | | | |
| array([2, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, |
| 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 0, | 2, | 3, | 2, |
| 3, | 2, | 0, | 2, | 2, | 2, | 0, | 2, | 2, | 0, | 0, | 0, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 0, | 2, |
| 0, | 0, | 2, | 2, | 0, | 0, | 0, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, |
| 2, | 0, | 0, | 2, | 0, | 0, | 2, | 2, | 0, | 2, | 0, | 2, | 2, | 0, | 0, | 2, | 0, | 2, | 0, | 0, | 0, | 0, |
| 0, | 2, | 4, | 2, | 2, | 2, | 0, | 0, | 0, | 0, | 1, | 4, | 1, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1]) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

df['Cluster'] **=** pd.DataFrame(pred, columns**=**['cluster'] )

print('Number of data points in each cluster= \n', df['Cluster'].value\_counts()) df

Number of data points in each cluster=

|  |  |  |
| --- | --- | --- |
|  | 2 | 52 |
| 0 |  | 46 |
| 1 |  | 41 |
| 4 |  | 40 |
| 3 |  | 21 |

Name: Cluster, dtype: int64 Out[34]:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **CustomerID** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** | **Cluster** |
| **0** | 0 | 19 | 15 | 39 | 2 |
| **1** | 1 | 21 | 15 | 81 | 2 |
| **2** | 2 | 20 | 16 | 6 | 3 |
| **3** | 3 | 23 | 16 | 77 | 2 |
| **4** | 4 | 31 | 17 | 40 | 3 |
| **...** | ... | ... | ... | ... | ... |
| **195** | 195 | 35 | 120 | 79 | 1 |
| **196** | 196 | 45 | 126 | 28 | 4 |
| **197** | 197 | 32 | 126 | 74 | 1 |
| **198** | 198 | 32 | 137 | 18 | 4 |
| **199** | 199 | 30 | 137 | 83 | 1 |

200 rows × 5 columns

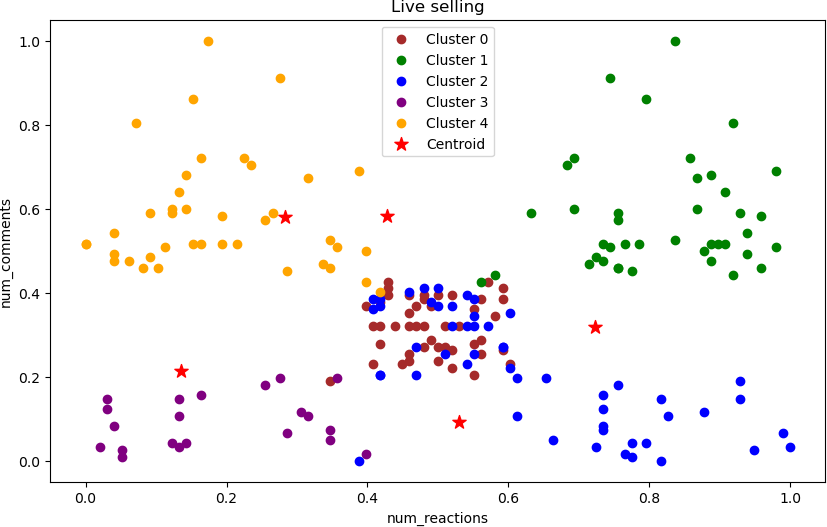
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| plt.scatter(kmeans.cluster\_centers\_[:,1], | kmeans.cluster\_centers\_[:, | 2],s | **=**100, | c | **=** | 'red |
| plt.xlabel('num\_reactions') |  |  |  |  |  |  |
| plt.ylabel('num\_comments') |  |  |  |  |  |  |
| plt.legend() |  |  |  |  |  |  |
| plt.title('Live selling') |  |  |  |  |  |  |

Out[35]:

plt.figure(figsize**=**(10,6))

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| plt.scatter(X[pred | **==** | 0, | 3], | X[pred | **==** | 0, | 2], | c | **=** | 'brown', label **=** 'Cluster 0') |
| plt.scatter(X[pred | **==** | 1, | 3], | X[pred | **==** | 1, | 2], | c | **=** | 'green', label **=** 'Cluster 1') |
| plt.scatter(X[pred | **==** | 2, | 3], | X[pred | **==** | 2, | 2], | c | **=** | 'blue', label **=** 'Cluster 2') |
| plt.scatter(X[pred | **==** | 3, | 3], | X[pred | **==** | 3, | 2], | c | **=** | 'purple', label **=** 'Cluster 3') |
| plt.scatter(X[pred | **==** | 4, | 3], | X[pred | **==** | 4, | 2], | c | **=** | 'orange', label **=** 'Cluster 4') |

Text(0.5, 1.0, 'Live selling')



In [36]:

labels1 **=** kmeans.labels\_

centroids1 **=** kmeans.cluster\_centers\_ labels1

Out[36]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| array([2, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, |
| 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 3, | 2, | 0, | 2, | 3, | 2, |
| 3, | 2, | 0, | 2, | 2, | 2, | 0, | 2, | 2, | 0, | 0, | 0, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 0, | 2, |
| 0, | 0, | 2, | 2, | 0, | 0, | 0, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, | 0, | 0, | 2, |
| 2, | 0, | 0, | 2, | 0, | 0, | 2, | 2, | 0, | 2, | 0, | 2, | 2, | 0, | 0, | 2, | 0, | 2, | 0, | 0, | 0, | 0, |
| 0, | 2, | 4, | 2, | 2, | 2, | 0, | 0, | 0, | 0, | 1, | 4, | 1, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, | 4, | 1, |
| 4, | 1]) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

kmeans.inertia\_

Out[37]:

13.967176243333345

In [ ]: