label-and-one-hot-encoding

1 Label and Onehot Encoding

[1]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

[2]:
```
# Step 1: Load the dataset
df = pd.read_csv("encoding.csv")
```

[3]:
```
# Step 2: Apply Label Encoding (Converting categorical columns into numeric␣
↪labels)
label_enc = LabelEncoder()
df['Color_LabelEncoded'] = label_enc.fit_transform(df['Color'])
df['Size_LabelEncoded'] = label_enc.fit_transform(df['Size'])
```

[4]:
```
df
```

[4]:

	ID	Color	Size	Price	Color_LabelEncoded	Size_LabelEncoded
0	1	Red	Small	10.5	2	2
1	2	Blue	Medium	15.0	0	1
2	3	Green	Large	12.3	1	0
3	4	Blue	Small	14.2	0	2
4	5	Red	Large	13.1	2	0
5	6	Green	Medium	16.8	1	1
6	7	Red	Small	11.5	2	2
7	8	Blue	Large	17.3	0	0
8	9	Green	Medium	14.9	1	1
9	10	Red	Large	15.7	2	0

[8]:
```
# Step 3: Apply One-Hot Encoding (Creating binary columns)
df_onehot = pd.get_dummies(df, columns=['Color', 'Size'], prefix=['Color',␣
↪'Size'])
```

[10]:
```
df_onehot
```

[10]:

	ID	Price	Color_LabelEncoded	Size_LabelEncoded	Color_Blue	Color_Green	\
0	1	10.5	2	2	False	False	
1	2	15.0	0	1	True	False	

```
2   3   12.3                 1                  0       False         True
3   4   14.2                 0                  2        True        False
4   5   13.1                 2                  0       False        False
5   6   16.8                 1                  1       False         True
6   7   11.5                 2                  2       False        False
7   8   17.3                 0                  0        True        False
8   9   14.9                 1                  1       False         True
9  10   15.7                 2                  0       False        False

   Color_Red  Size_Large  Size_Medium  Size_Small
0       True       False        False        True
1      False       False         True       False
2      False        True        False       False
3      False       False        False        True
4       True        True        False       False
5      False       False         True       False
6       True       False        False        True
7      False        True        False       False
8      False       False         True       False
9       True        True        False       False
```

[11]:
```
# Convert boolean values to integers
df_onehot = df_onehot.astype(int)
```

[12]:
```
df_onehot
```

[12]:
```
   ID  Price  Color_LabelEncoded  Size_LabelEncoded  Color_Blue  Color_Green  \
0   1     10                   2                  2           0            0
1   2     15                   0                  1           1            0
2   3     12                   1                  0           0            1
3   4     14                   0                  2           1            0
4   5     13                   2                  0           0            0
5   6     16                   1                  1           0            1
6   7     11                   2                  2           0            0
7   8     17                   0                  0           1            0
8   9     14                   1                  1           0            1
9  10     15                   2                  0           0            0

   Color_Red  Size_Large  Size_Medium  Size_Small
0          1           0            0           1
1          0           0            1           0
2          0           1            0           0
3          0           0            0           1
4          1           1            0           0
5          0           0            1           0
6          1           0            0           1
7          0           1            0           0
```

```
8          0           0            1           0
9          1           1            0           0
```

[16]:
```
print(df_onehot)
```

```
   ID  Price  Color_LabelEncoded  Size_LabelEncoded  Color_Blue  Color_Green  \
0   1     10                   2                  2           0            0
1   2     15                   0                  1           1            0
2   3     12                   1                  0           0            1
3   4     14                   0                  2           1            0
4   5     13                   2                  0           0            0
5   6     16                   1                  1           0            1
6   7     11                   2                  2           0            0
7   8     17                   0                  0           1            0
8   9     14                   1                  1           0            1
9  10     15                   2                  0           0            0

   Color_Red  Size_Large  Size_Medium  Size_Small
0          1           0            0           1
1          0           0            1           0
2          0           1            0           0
3          0           0            0           1
4          1           1            0           0
5          0           0            1           0
6          1           0            0           1
7          0           1            0           0
8          0           0            1           0
9          1           1            0           0
```

[14]:
```
from IPython.display import display
display(df_onehot)
```

```
   ID  Price  Color_LabelEncoded  Size_LabelEncoded  Color_Blue  Color_Green  \
0   1     10                   2                  2           0            0
1   2     15                   0                  1           1            0
2   3     12                   1                  0           0            1
3   4     14                   0                  2           1            0
4   5     13                   2                  0           0            0
5   6     16                   1                  1           0            1
6   7     11                   2                  2           0            0
7   8     17                   0                  0           1            0
8   9     14                   1                  1           0            1
9  10     15                   2                  0           0            0

   Color_Red  Size_Large  Size_Medium  Size_Small
0          1           0            0           1
1          0           0            1           0
2          0           1            0           0
```

3	0	0	0	1
4	1	1	0	0
5	0	0	1	0
6	1	0	0	1
7	0	1	0	0
8	0	0	1	0
9	1	1	0	0

k-means-clustering

[22]:
```
import pandas as pd
import numpy as np
import seaborn
import matplotlib.pyplot as plt
```

[12]:
```
df= pd.read_csv('Mall_Customers.csv')
```

[13]:
```
df
```

[13]:

	CustomerID	Gender	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]

[15]:
```
df.rename (columns = {'Gender': 'gender', 'Age': 'age', 'Annual Income (k$)' :␣
↪'income', 'Spending Score (1-100)':'score' },inplace = True)
```

[16]:
```
df
```

[16]:

	CustomerID	gender	age	income	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
..	…	…	…	…	…
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]
```

[9]:
```
df.shape
```

[9]:
```
(200, 5)
```

[11]:
```
df.isnull().values.any()
```

[11]:
```
False
```

[12]:
```
df.describe()
```

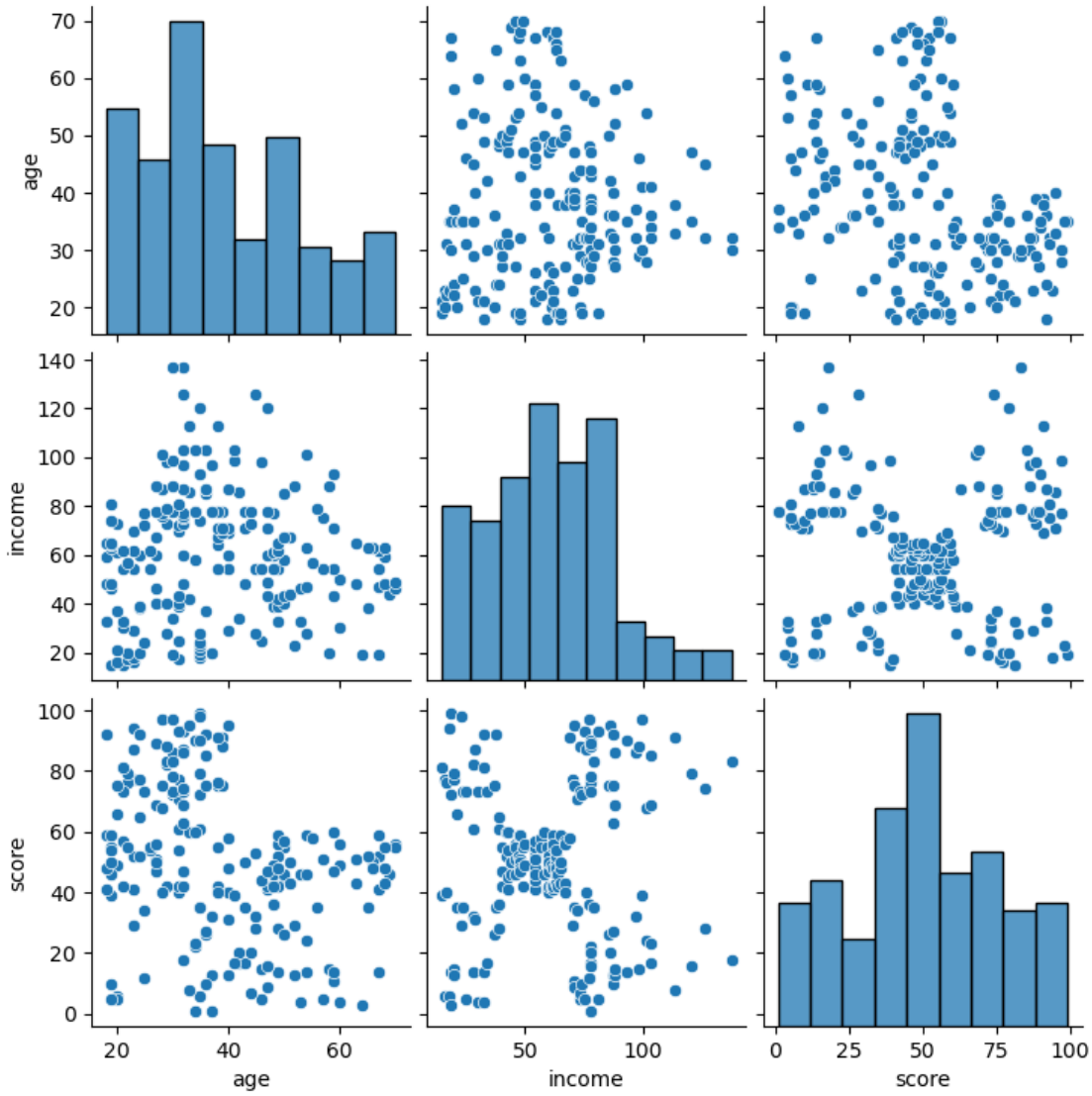
[12]:

	CustomerID	age	income	score
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

[15]:
```
seaborn.pairplot (df[['age', 'income','score']])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

[15]:
```
<seaborn.axisgrid.PairGrid at 0x1f152677b50>
```



[16]:
```
import sklearn.cluster as cluster
```

kmeans = cluster.KMeans(n_clusters = 5)

[24]:
```
kmeans = kmeans.fit(df[['score','income']])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
```

[25]:
```
#Finding out the centroids

kmeans.cluster_centers_
```

[25]:
```
array([[82.12820513, 86.53846154],
       [20.91304348, 26.30434783],
       [49.51851852, 55.2962963 ],
       [17.11428571, 88.2       ],
       [79.36363636, 25.72727273]])
```

[26]:
```
kmeans = kmeans.fit(df[['income','score']])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
```

[27]:
```
kmeans.cluster_centers_
```

[27]:
```
array([[25.72727273, 79.36363636],
       [55.2962963 , 49.51851852],
       [86.53846154, 82.12820513],
       [88.2       , 17.11428571],
       [26.30434783, 20.91304348]])
```

[28]:
```
df['income_clusters'] = kmeans.labels_
```

[29]:
```
df
```

[29]:

	CustomerID	gender	age	income	score	income_clusters
0	1	Male	19	15	39	4
1	2	Male	21	15	81	0
2	3	Female	20	16	6	4
3	4	Female	23	16	77	0
4	5	Female	31	17	40	4
..	…	…	…	…	…	…
195	196	Female	35	120	79	2
196	197	Female	45	126	28	3
197	198	Male	32	126	74	2
198	199	Male	32	137	18	3

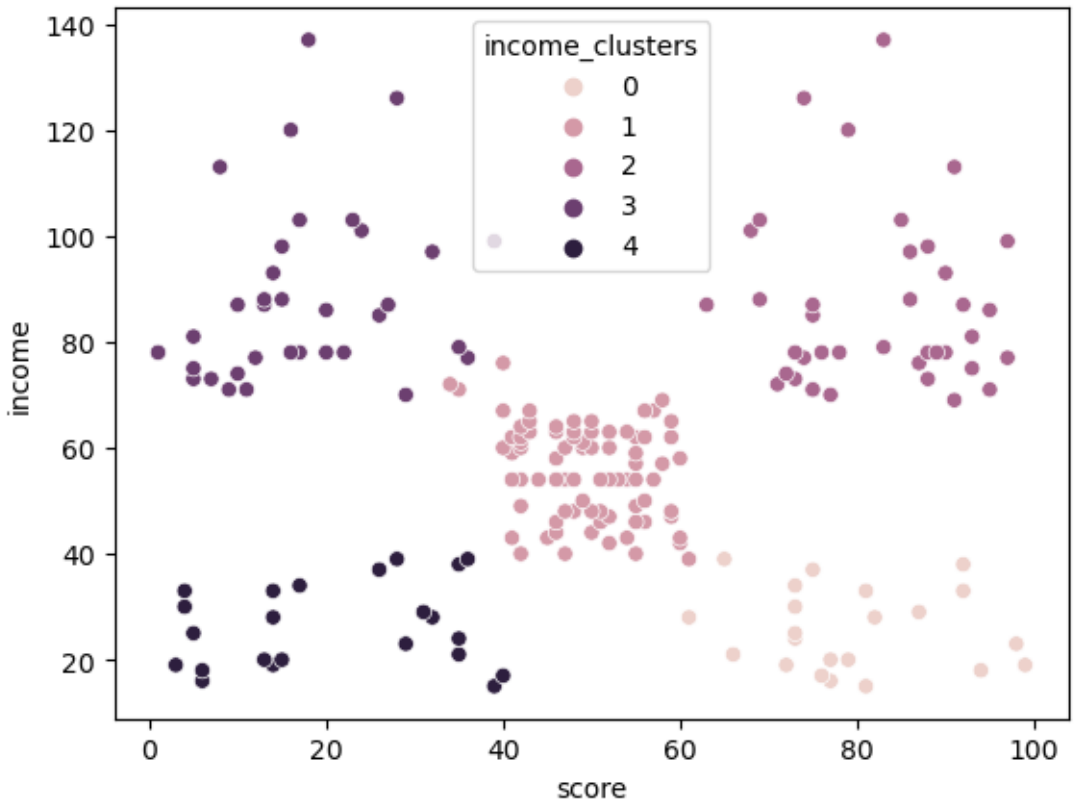
```
199         200    Male   30     137     83                2

[200 rows x 6 columns]
```

[30]:
```
# counting in which cluster how many values
df['income_clusters'].value_counts()
```

[30]:
```
income_clusters
1    81
2    39
3    35
4    23
0    22
Name: count, dtype: int64
```

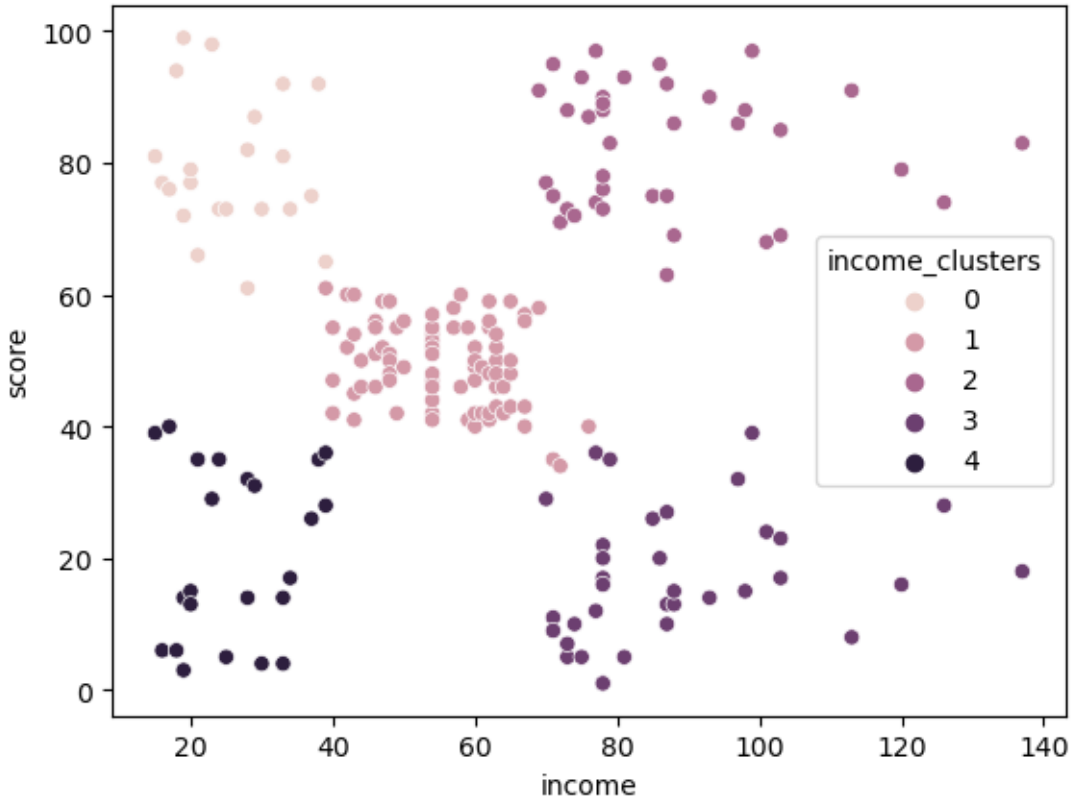
[31]:
```
seaborn.scatterplot(x= 'score', y = 'income', hue= 'income_clusters', data= df)
```

[31]:
```
<Axes: xlabel='score', ylabel='income'>
```



[32]:
```
seaborn.scatterplot(x= 'income', y = 'score', hue= 'income_clusters', data= df)
```

[32]: <Axes: xlabel='income', ylabel='score'>



[33]:
```
#trying with 2 clusters
kmeans = cluster.KMeans(n_clusters = 2)
```

[34]:
```
kmeans = kmeans.fit(df[['age','score']])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
```

[35]:
```
kmeans.cluster_centers_
```

[35]:
```
array([[28.95294118, 73.62352941],
       [46.16521739, 32.88695652]])
```

[36]:
```
df['age_clusters'] = kmeans.labels_
```

[37]:
```
df
```

[37]:

	CustomerID	gender	age	income	score	income_clusters	age_clusters
0	1	Male	19	15	39	4	1
1	2	Male	21	15	81	0	0
2	3	Female	20	16	6	4	1
3	4	Female	23	16	77	0	0
4	5	Female	31	17	40	4	1
..	…	…	…	…	…	…	…
195	196	Female	35	120	79	2	0
196	197	Female	45	126	28	3	1
197	198	Male	32	126	74	2	0
198	199	Male	32	137	18	3	1
199	200	Male	30	137	83	2	0

[200 rows x 7 columns]

[38]:
```
df['age_clusters'].value_counts()
```

[38]:
```
age_clusters
1    115
0     85
Name: count, dtype: int64
```

[39]:
```
df
```

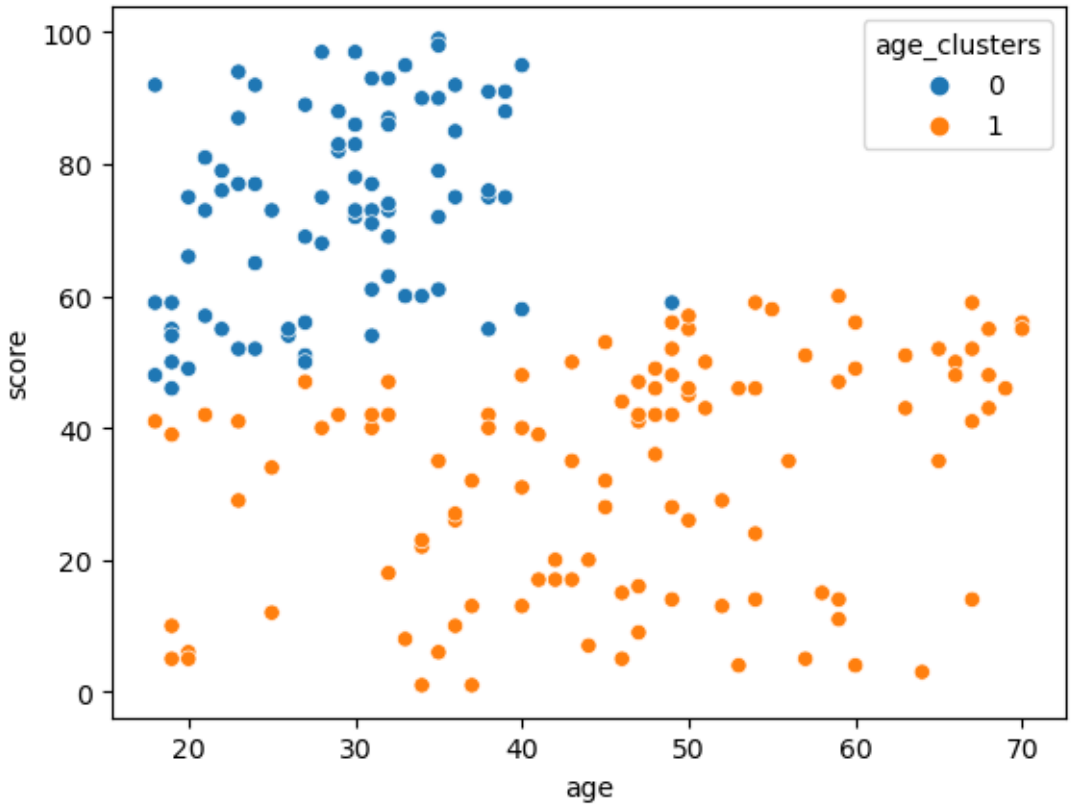
[39]:

	CustomerID	gender	age	income	score	income_clusters	age_clusters
0	1	Male	19	15	39	4	1
1	2	Male	21	15	81	0	0
2	3	Female	20	16	6	4	1
3	4	Female	23	16	77	0	0
4	5	Female	31	17	40	4	1
..	…	…	…	…	…	…	…
195	196	Female	35	120	79	2	0
196	197	Female	45	126	28	3	1
197	198	Male	32	126	74	2	0
198	199	Male	32	137	18	3	1
199	200	Male	30	137	83	2	0

[200 rows x 7 columns]

[40]:
```
seaborn.scatterplot(x= 'age', y = 'score', hue= 'age_clusters', data= df)
```

[40]:
```
<Axes: xlabel='age', ylabel='score'>
```



1 Applying Elbow methods

[3]:
```
#from sklearn.cluster import kMeans
from sklearn.cluster import KMeans
```

[6]:
```
# to calculate 12 clusters

k_range = range(1,12)
wcss = []
```

[20]:
```
# implementing for loop for 12 clusters to find out wcss value for each cluster
# adding wcss values of clusters in wcss array

for k in k_range:
    km= KMeans(n_clusters =k)
    km.fit(df[['income','score']])
    wcss.append (km.inertia_)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
```

```
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
```

```
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436:
UserWarning: 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(
```

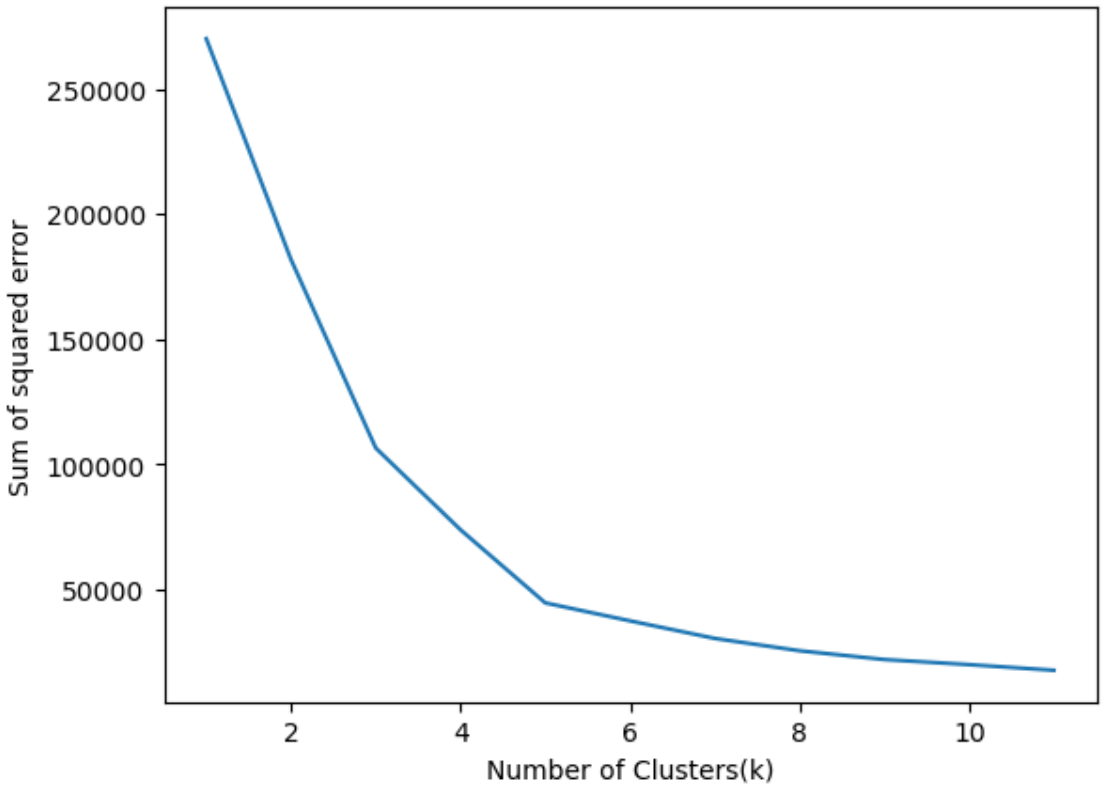
2 showing the values of wcss

[21]:
```
wcss
```

[21]:
```
[269981.28,
 181665.82312925172,
 106348.37306211122,
 73679.78903948836,
 44448.4554479337,
 37265.86520484346,
 30241.34361793658,
 25315.541822712163,
 21809.92756261518,
 19770.06950274374,
 17511.7418692661]
```

[23]:
```
plt.xlabel ('Number of Clusters(k)')
plt.ylabel ('Sum of squared error')
plt.plot (k_range , wcss)
```

[23]:
```
[<matplotlib.lines.Line2D at 0x2098852b0d0>]
```



3 Optimal number of k is 5