K-Means clustering on socio-economic dataset

In [2]:
```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

df = pd.read_excel('Preprocessed_data_standardscaler.xlsx')

# Dropping the ID Column, as it is not required in clustering.

new_df = df.drop(columns=['ID'])
new_df
```

Out[2]:

	Normalised Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experience (10-20yr)	Experience (20-30yr)	Experience (>30yr)	Helme (Full mask
0	-0.927352	1	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	
1	0.387063	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0
2	-0.739578	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	
3	0.762611	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
4	-1.678446	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
476	-0.739578	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	
477	-1.302899	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	
478	2.077026	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	
479	-0.551804	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	
480	-0.551804	1	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	

481 rows × 19 columns

In [3]:
```
# Now the Silhouette Score method is used to determine the number of clusters to achieve the best possible clustering
# Here the scores are calculated for k = {2,3,4,5,6}

scores = {}

for i in range(2,7):
    kmeans = KMeans(n_clusters=i,random_state=42,n_init=10)
    kmeans.fit(new_df)
    score = silhouette_score(new_df,kmeans.labels_)
    scores[i] = score

df_scores = pd.DataFrame(list(scores.items()), columns=['no. of clusters', 'silhouette score'])
df_scores

# The value of K = 2 yields the highest silhouette score, as can be seen in the table below:
```

Out[3]:

	no. of clusters	silhouette score
0	2	0.209891
1	3	0.169886
2	4	0.184513
3	5	0.196913
4	6	0.203848

In [4]:
```
# Now applying the K-Means algorithm

k_means_1 = KMeans(n_clusters=2,random_state=42,n_init=10)
k_means_1.fit(new_df)

# The K-Means algorithm used here follows the Lloyd algorithm.

list1 = list(k_means_1.labels_)

# Made a copy of the new database and added a new column for the cluster labels.

clustered_data = new_df.copy()

clustered_data['Cluster'] = list1

clustered_data
```

Out[4]:

	Normalised Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experience (10-20yr)	Experience (20-30yr)	Experience (>30yr)	Helme (Full mask
0	-0.927352	1	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	
1	0.387063	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0
2	-0.739578	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	
3	0.762611	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
4	-1.678446	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
476	-0.739578	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	
477	-1.302899	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	
478	2.077026	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	
479	-0.551804	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	
480	-0.551804	1	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	

481 rows × 20 columns

In [5]:
```
# Now that the clustering is completed, the 'Normalised Age (Yrs)' column is dropped and replaced with the

final_df = clustered_data.copy()

old_df = pd.read_excel('Asansol socio-economic data 1.xlsx')
age = list(old_df['Age (Yrs)'])

final_df['Age (Yrs)'] = age

final_df = final_df.drop(columns = ['Normalised Age (Yrs)'])

list2 = list(final_df.columns)
new_order = [list2[-1]] + list2[0:-1]

final_df = final_df[new_order]
final_df
```

Out[5]:

	Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experience (10-20yr)	Experience (20-30yr)	Experience (>30yr)	Helmet (Full-mask)	Clus
0	26	1	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	
1	40	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
2	28	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	1	
3	44	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	
4	18	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
476	28	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	1	
477	22	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	
478	58	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	1	
479	30	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	1	
480	30	1	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	

481 rows × 20 columns

In [6]:
```
# now we seperate out the various clusters from the dataset

cluster_1 = final_df.loc[final_df['Cluster']==0]
cluster_1
cluster_1.describe()
```

Out[6]:

	Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experienc (10-20y
count	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000	315.000000
mean	29.568254	0.946032	0.193651	0.288889	0.473016	0.044444	0.285714	0.365079	0.266667	0.066667	0.015873	0.044444	0.114286	0.177778	0.434921	0.2222
std	5.362432	0.226315	0.395787	0.453967	0.500066	0.206408	0.452473	0.482218	0.442920	0.249841	0.125183	0.206408	0.318664	0.382934	0.496535	0.4164
min	16.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	26.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
50%	30.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
75%	34.000000	1.000000	0.000000	1.000000	1.000000	0.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.0000
max	41.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000

In [7]:
```
cluster_2 = final_df.loc[final_df['Cluster']==1]
cluster_2.describe()
```

Out[7]:

	Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experien (10-20
count	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.000000	166.0000
mean	47.849398	0.957831	0.180723	0.337349	0.439759	0.042169	0.156627	0.337349	0.295181	0.186747	0.024096	0.006024	0.018072	0.048193	0.120482	0.4578
std	7.470035	0.201582	0.385953	0.474236	0.497860	0.201582	0.364548	0.474236	0.457504	0.390887	0.153812	0.077615	0.133616	0.214821	0.326509	0.4997
min	36.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	42.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
50%	46.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
75%	52.000000	1.000000	0.000000	1.000000	1.000000	0.000000	0.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.0000
max	73.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000

In [8]:
```
# The following dataset contains the mean value for each parameter of each cluster.

overall_mean = final_df.groupby('Cluster').mean()

overall_mean
```

Out[8]:

	Age (Yrs)	Gender (Male)	Education (<Class 10)	Education (Class 10-12)	Graduate	Higher Edu	Income <20k	Income 20-40k	Income 40-60k	Income 60-80k	Income >80k	Experience (<1yr)	Experience (1-2yr)	Experience (2-5yr)	Experience (5-10yr)	Experience (10-20yr)	Experience (20-30yr)	Expe (
Cluster																		
0	29.568254	0.946032	0.193651	0.288889	0.473016	0.044444	0.285714	0.365079	0.266667	0.066667	0.015873	0.044444	0.114286	0.177778	0.434921	0.222222	0.006349	0.0
1	47.849398	0.957831	0.180723	0.337349	0.439759	0.042169	0.156627	0.337349	0.295181	0.186747	0.024096	0.006024	0.018072	0.048193	0.120482	0.457831	0.283133	0.0

In [9]:
```
# Exporting the datasets to Excel:

cluster_1.to_excel('Cluster_1_kmeans.xlsx')
cluster_2.to_excel('Cluster_2_kmeans.xlsx')

overall_mean.to_excel('Means_of_parameters_kmeans.xlsx')
final_df.to_excel('clustered_data_kmeans.xlsx')
```