CLUSTERING TASK

Clustering is an unsupervised machine learning method of identifying and grouping similar data points in larger datasets without concern for the specific outcome. Clustering (sometimes called cluster analysis) is usually used to classify data into structures that are more easily understood and manipulated.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

import dataset

```
df = pd.read_csv(r'C:\Users\hp\Dropbox\My PC (LAPTOP-7K4M1D0J)\
Downloads\2.K-MEANS CLUSTERING\2.K-MEANS CLUSTERING\
Mall_Customers.csv')
```

```
df
```

```
     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]
```

```
x=df.iloc[:,[3,4]].values
```

```
x
```

```
array([[ 15,  39],
       [ 15,  81],
       [ 16,   6],
       [ 16,  77],
       [ 17,  40],
       [ 17,  76],
       [ 18,   6],
       [ 18,  94],
       [ 19,   3],
       [ 19,  72],
       [ 19,  14],
       [ 19,  99],
       [ 20,  15],
       [ 20,  77],
       [ 20,  13],
       [ 20,  79],
       [ 21,  35],
       [ 21,  66],
       [ 23,  29],
       [ 23,  98],
       [ 24,  35],
       [ 24,  73],
       [ 25,   5],
       [ 25,  73],
       [ 28,  14],
       [ 28,  82],
       [ 28,  32],
       [ 28,  61],
       [ 29,  31],
       [ 29,  87],
       [ 30,   4],
       [ 30,  73],
       [ 33,   4],
       [ 33,  92],
       [ 33,  14],
       [ 33,  81],
       [ 34,  17],
       [ 34,  73],
       [ 37,  26],
       [ 37,  75],
       [ 38,  35],
       [ 38,  92],
       [ 39,  36],
       [ 39,  61],
       [ 39,  28],
       [ 39,  65],
       [ 40,  55],
       [ 40,  47],
```

```
[ 40,  42],
[ 40,  42],
[ 42,  52],
[ 42,  60],
[ 43,  54],
[ 43,  60],
[ 43,  45],
[ 43,  41],
[ 44,  50],
[ 44,  46],
[ 46,  51],
[ 46,  46],
[ 46,  56],
[ 46,  55],
[ 47,  52],
[ 47,  59],
[ 48,  51],
[ 48,  59],
[ 48,  50],
[ 48,  48],
[ 48,  59],
[ 48,  47],
[ 49,  55],
[ 49,  42],
[ 50,  49],
[ 50,  56],
[ 54,  47],
[ 54,  54],
[ 54,  53],
[ 54,  48],
[ 54,  52],
[ 54,  42],
[ 54,  51],
[ 54,  55],
[ 54,  41],
[ 54,  44],
[ 54,  57],
[ 54,  46],
[ 57,  58],
[ 57,  55],
[ 58,  60],
[ 58,  46],
[ 59,  55],
[ 59,  41],
[ 60,  49],
[ 60,  40],
[ 60,  42],
[ 60,  52],
[ 60,  47],
[ 60,  50],
```

```
[ 61,  42],
[ 61,  49],
[ 62,  41],
[ 62,  48],
[ 62,  59],
[ 62,  55],
[ 62,  56],
[ 62,  42],
[ 63,  50],
[ 63,  46],
[ 63,  43],
[ 63,  48],
[ 63,  52],
[ 63,  54],
[ 64,  42],
[ 64,  46],
[ 65,  48],
[ 65,  50],
[ 65,  43],
[ 65,  59],
[ 67,  43],
[ 67,  57],
[ 67,  56],
[ 67,  40],
[ 69,  58],
[ 69,  91],
[ 70,  29],
[ 70,  77],
[ 71,  35],
[ 71,  95],
[ 71,  11],
[ 71,  75],
[ 71,   9],
[ 71,  75],
[ 72,  34],
[ 72,  71],
[ 73,   5],
[ 73,  88],
[ 73,   7],
[ 73,  73],
[ 74,  10],
[ 74,  72],
[ 75,   5],
[ 75,  93],
[ 76,  40],
[ 76,  87],
[ 77,  12],
[ 77,  97],
[ 77,  36],
[ 77,  74],
```

```
[ 78,  22],
[ 78,  90],
[ 78,  17],
[ 78,  88],
[ 78,  20],
[ 78,  76],
[ 78,  16],
[ 78,  89],
[ 78,   1],
[ 78,  78],
[ 78,   1],
[ 78,  73],
[ 79,  35],
[ 79,  83],
[ 81,   5],
[ 81,  93],
[ 85,  26],
[ 85,  75],
[ 86,  20],
[ 86,  95],
[ 87,  27],
[ 87,  63],
[ 87,  13],
[ 87,  75],
[ 87,  10],
[ 87,  92],
[ 88,  13],
[ 88,  86],
[ 88,  15],
[ 88,  69],
[ 93,  14],
[ 93,  90],
[ 97,  32],
[ 97,  86],
[ 98,  15],
[ 98,  88],
[ 99,  39],
[ 99,  97],
[101,  24],
[101,  68],
[103,  17],
[103,  85],
[103,  23],
[103,  69],
[113,   8],
[113,  91],
[120,  16],
[120,  79],
[126,  28],
[126,  74],
```

```
       [137,  18],
       [137,  83]], dtype=int64)
```

using the elbow method to find the number of clusters

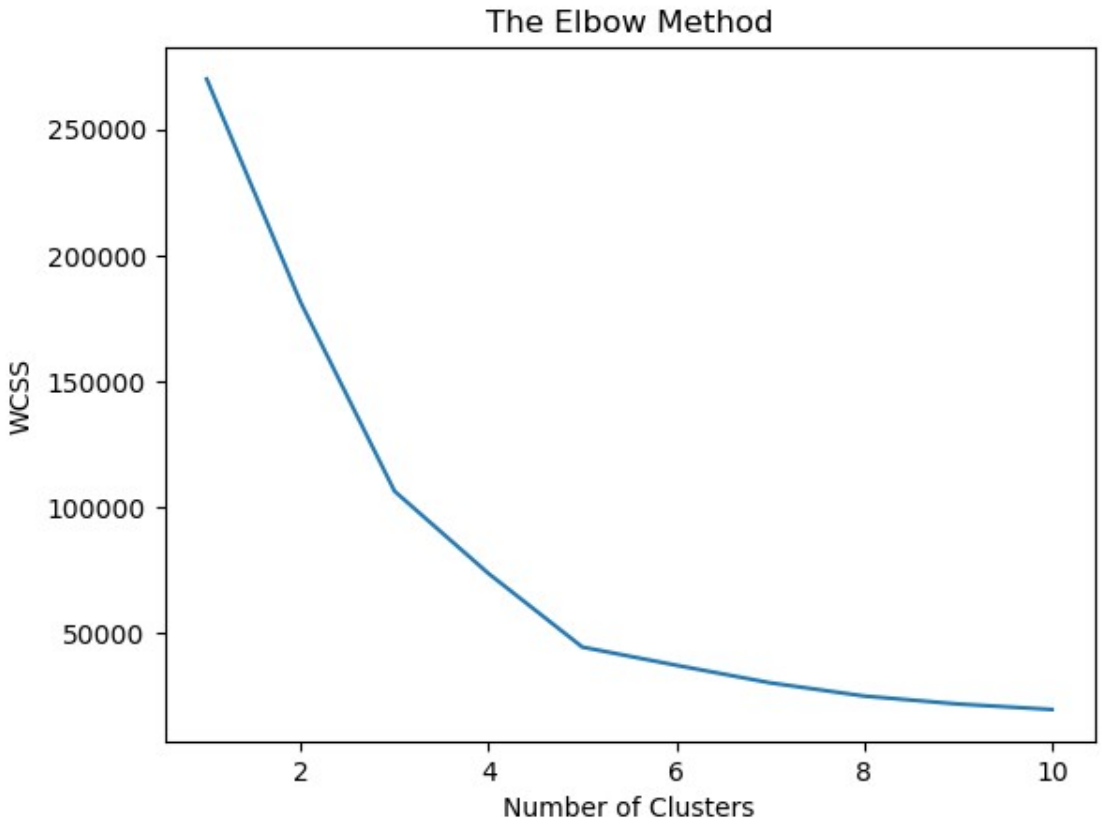
```
from sklearn.cluster import KMeans

wcss=[]

for i in range(1,11):
    kmeans = KMeans (n_clusters=i,init='k-means++',random_state=(42))
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1036: 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(
```

```
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```



Training the K-Means model on the dataset

```
kmeans=KMeans(n_clusters=5,init='k-means++',random_state=(42))

y_kmeans=kmeans.fit_predict(x)
```

y_kmeans=pd.dataframe(y_kmeans)

```
df['Cluster']=y_kmeans
```

Visualising the clusters

```
df
```

```
     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
```

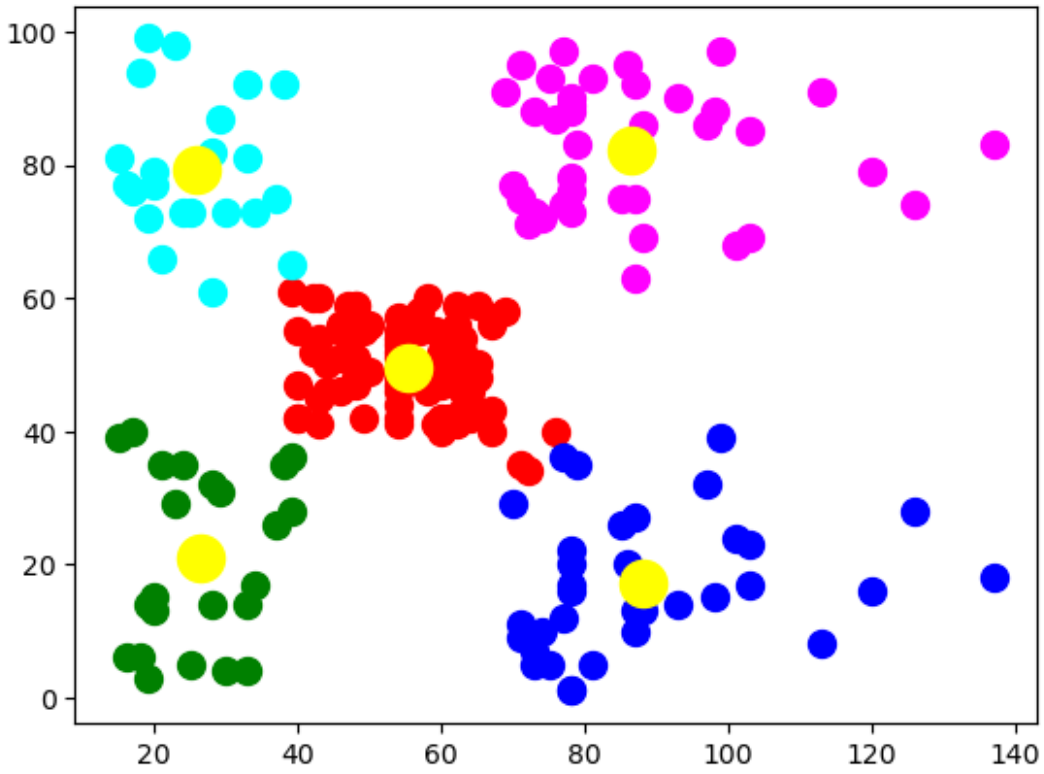
```
4             5  Female   31                 17
40
..          ...     ...  ...                ...                    .
..
195         196  Female   35                120
79
196         197  Female   45                126
28
197         198    Male   32                126
74
198         199    Male   32                137
18
199         200    Male   30                137
83

     Cluster
0          2
1          3
2          2
3          3
4          2
..       ...
195        4
196        1
197        4
198        1
199        4

[200 rows x 6 columns]
```

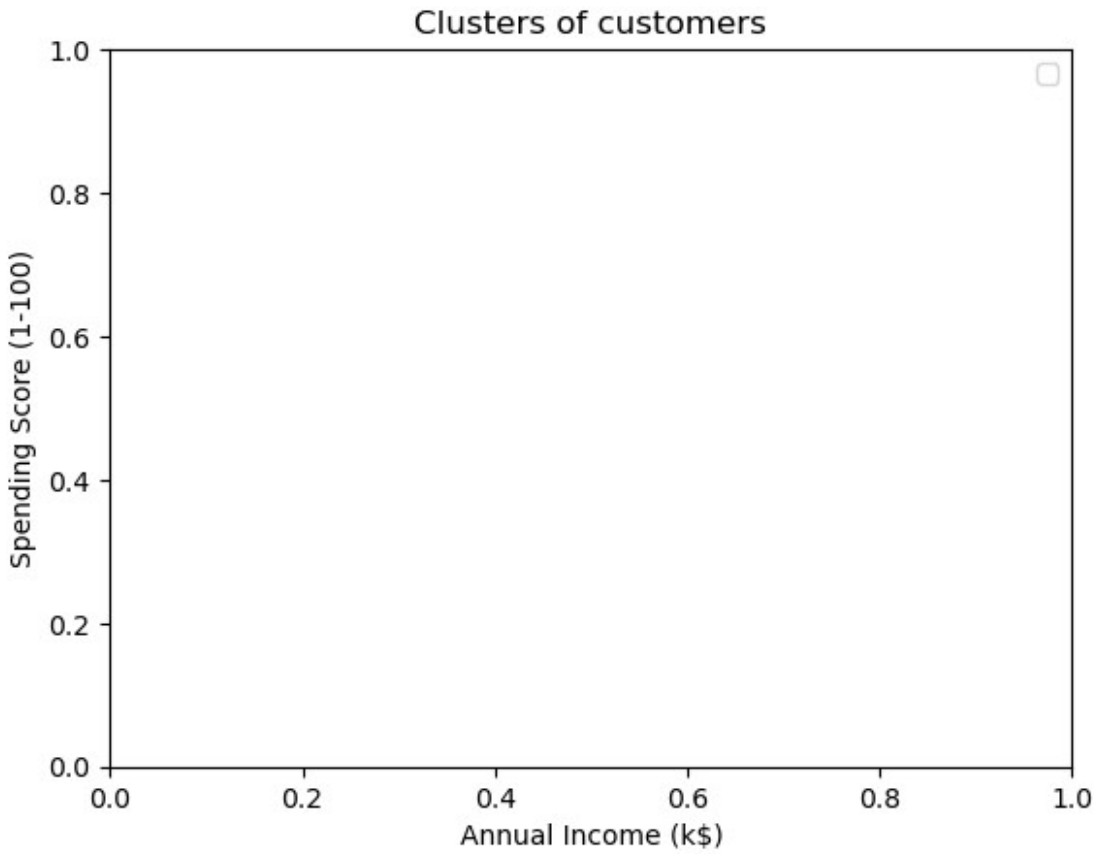
```
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c =
'red', label = 'Cluster 1')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c =
'blue', label = 'Cluster 2')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c =
'green', label = 'Cluster 3')
plt.scatter(x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c =
'cyan', label = 'Cluster 4')
plt.scatter(x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c =
'magenta', label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,
1], s = 300, c = 'yellow', label = 'Centroids')
```

```
<matplotlib.collections.PathCollection at 0x1d39db29ca0>
```

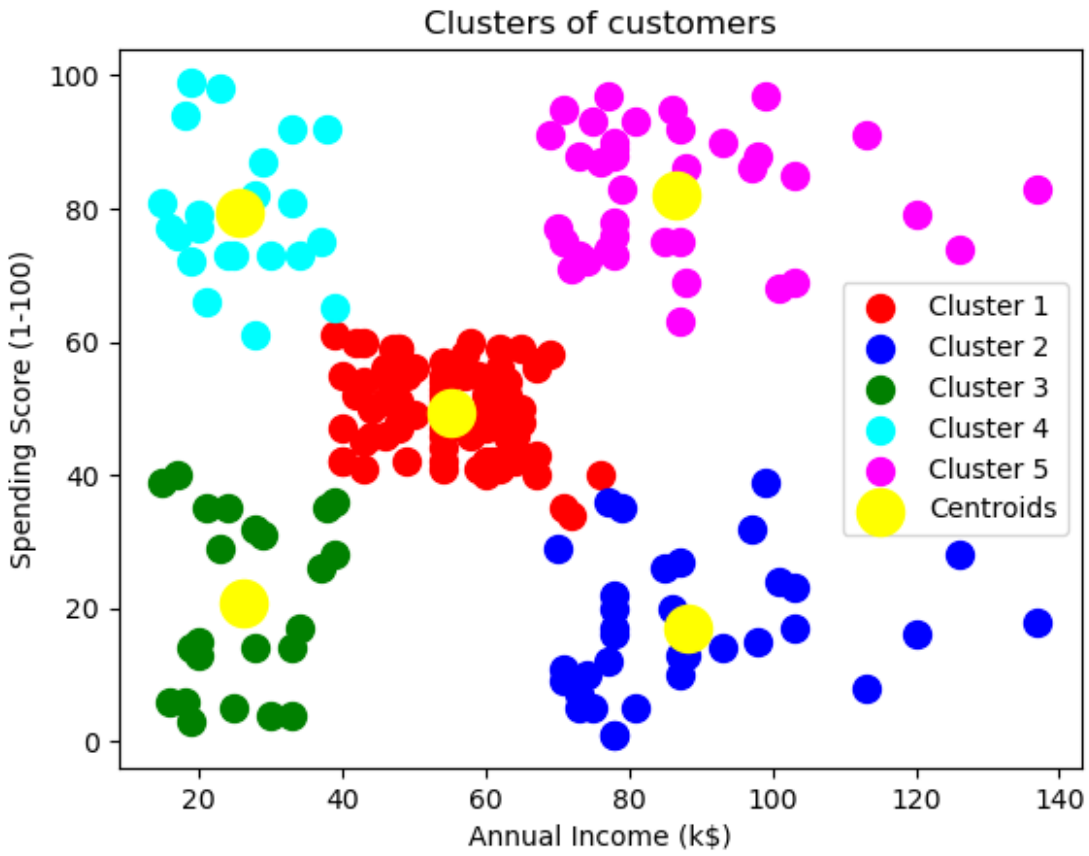



```
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is
called with no argument.
```




```
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c =
'red', label = 'Cluster 1')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c =
'blue', label = 'Cluster 2')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c =
'green', label = 'Cluster 3')
plt.scatter(x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c =
'cyan', label = 'Cluster 4')
plt.scatter(x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c =
'magenta', label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,
1], s = 300, c = 'yellow', label = 'Centroids')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```




```
df.to_csv('Mall_Customers.csv', index=False, mode='w', header=False)
y_kmeans
```

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

```
df
```

```
     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

     Cluster
0          2
1          3
2          2
3          3
4          2
..       ...
195        4
196        1
197        4
198        1
199        4

[200 rows x 6 columns]
```

```
x=df['Cluster'].value_counts()
```

```
x
```

```
0    81
4    39
1    35
2    23
3    22
Name: Cluster, dtype: int64
```

```
y=df[df['Cluster']==4]
```

```
len(y)
```

```
39
```

```
y
```

```
     CustomerID   Genre  Age  Annual Income (k$)  Spending Score (1-
100)  \
123         124    Male   39                  69
91
125         126  Female   31                  70
77
127         128    Male   40                  71
95
129         130    Male   38                  71
75
131         132    Male   39                  71
75
133         134  Female   31                  72
71
135         136  Female   29                  73
88
137         138    Male   32                  73
73
139         140  Female   35                  74
72
141         142    Male   32                  75
93
143         144  Female   32                  76
87
145         146    Male   28                  77
97
147         148  Female   32                  77
74
149         150    Male   34                  78
90
151         152    Male   39                  78
88
153         154  Female   38                  78
76
155         156  Female   27                  78
89
157         158  Female   30                  78
78
159         160  Female   30                  78
73
161         162  Female   29                  79
83
163         164  Female   31                  81
```

```
93
165         166  Female   36                  85
75
167         168  Female   33                  86
95
169         170    Male   32                  87
63
171         172    Male   28                  87
75
173         174    Male   36                  87
92
175         176  Female   30                  88
86
177         178    Male   27                  88
69
179         180    Male   35                  93
90
181         182  Female   32                  97
86
183         184  Female   29                  98
88
185         186    Male   30                  99
97
187         188    Male   28                 101
68
189         190  Female   36                 103
85
191         192  Female   32                 103
69
193         194  Female   38                 113
91
195         196  Female   35                 120
79
197         198    Male   32                 126
74
199         200    Male   30                 137
83

     Cluster
123        4
125        4
127        4
129        4
131        4
133        4
135        4
137        4
139        4
141        4
143        4
```

145	4
147	4
149	4
151	4
153	4
155	4
157	4
159	4
161	4
163	4
165	4
167	4
169	4
171	4
173	4
175	4
177	4
179	4
181	4
183	4
185	4
187	4
189	4
191	4
193	4
195	4
197	4
199	4