

ml-week-11

June 13, 2024

```
[2]: # k means clustering
```

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

```
[3]: data = pd.read_excel(r"C:\Users\lenovo\Downloads\Clustering_ex.xlsx.xlsx")
```

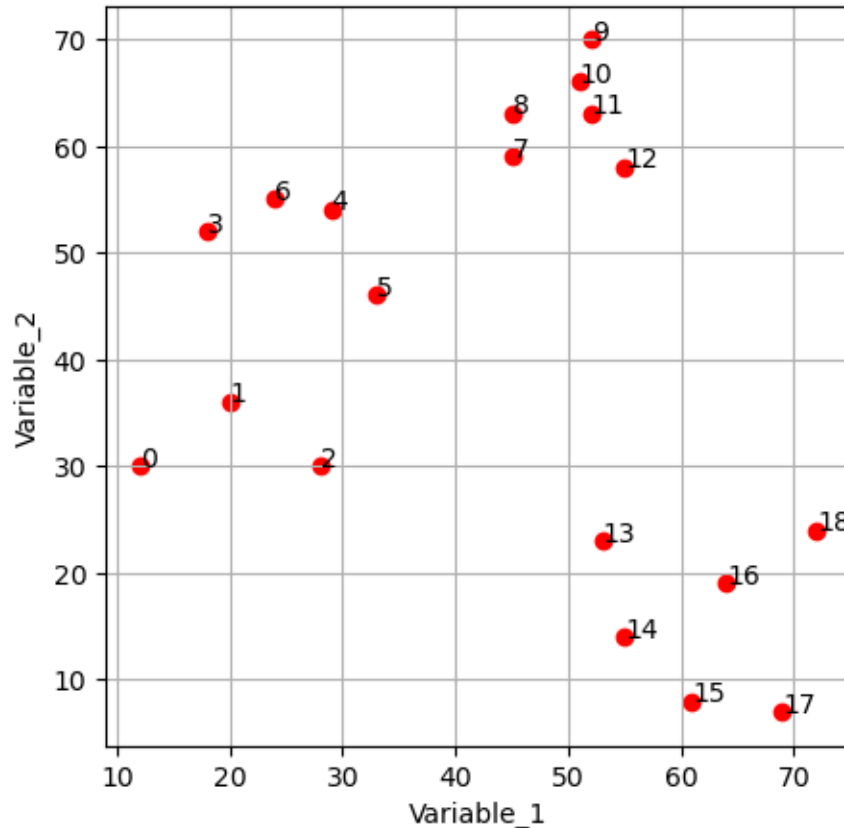
```
[4]: data
```

```
[4]:
```

	Variable_1	Variable_2
0	12	30
1	20	36
2	28	30
3	18	52
4	29	54
5	33	46
6	24	55
7	45	59
8	45	63
9	52	70
10	51	66
11	52	63
12	55	58
13	53	23
14	55	14
15	61	8
16	64	19
17	69	7
18	72	24

```
[5]: fig = plt.figure(figsize = (5,5))
x = data["Variable_1"]
y = data["Variable_2"]
n = range(0,19)
plt.grid()
plt.scatter(x, y, marker = 'o', c = 'red' )
```

```
plt.xlabel('Variable_1')
plt.ylabel('Variable_2')
for i, txt in enumerate(n):
    plt.annotate(txt, (x[i], y[i]))
```



```
[51]: from sklearn.cluster import KMeans
individual_clustering_score = []
for i in range(1,5):
    kmeans = KMeans(n_clusters = i)
    kmeans.fit(data)
    individual_clustering_score.append(kmeans.inertia_)
```

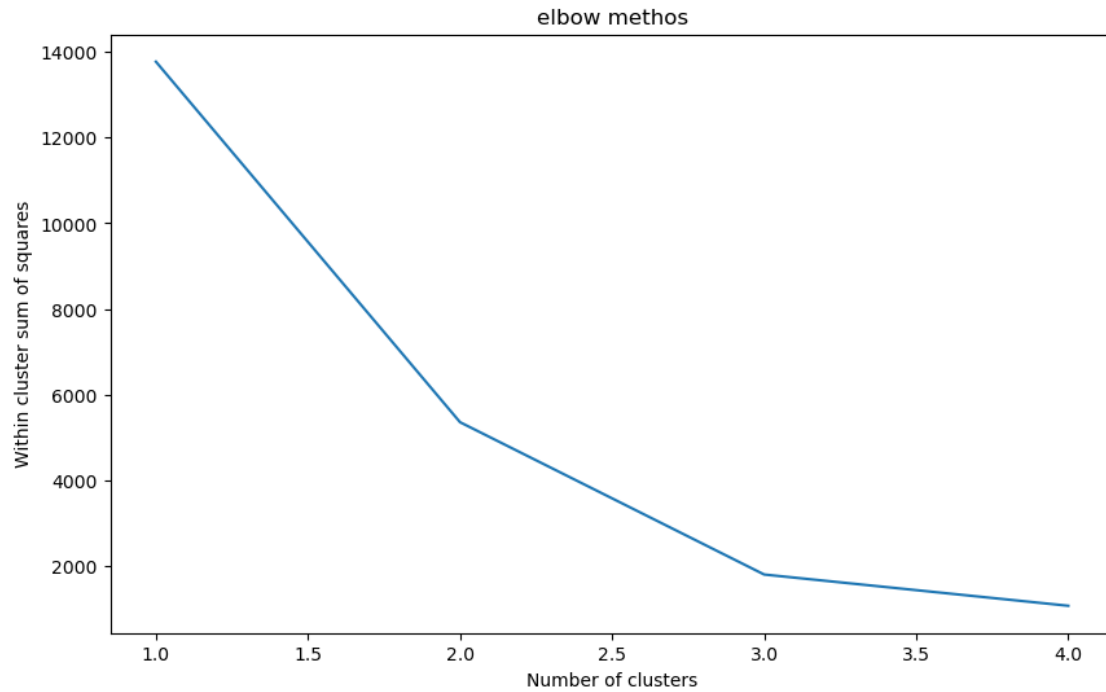
C:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
warnings.warn(
C:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster_kmeans.py:1382:
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:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:870:
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
warnings.warn(
C:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1382:
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:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:870:
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
warnings.warn(
C:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1382:
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:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:870:
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
warnings.warn(
C:\Users\lenovo\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1382:
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(
```

```
[61]: individual_clustering_score
```

```
[61]: [13773.57894736842, 5352.166666666667, 1794.142857142857, 1063.75]
```

```
[62]: plt.figure(figsize=(10,6))
plt.plot(range(1,5), individual_clustering_score)
plt.title("elbow methos")
plt.xlabel("Number of clusters")
plt.ylabel("Within cluster sum of squares")
plt.show()
```



```
[63]: labels = kmeans.predict(data)
```

```
[64]: labels
```

```
[64]: array([3, 3, 3, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1])
```

```
[65]: kmeans.labels_
```

```
[65]: array([3, 3, 3, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1])
```

```
[66]: centroids = kmeans.cluster_centers_
```

```
[60]: # cluster centers
centroids
```

```
[60]: array([[26.          , 51.75          ],
            [62.33333333, 15.83333333],
            [50.          , 63.16666667],
            [20.          , 32.          ]])
```

```
[84]: fig = plt.figure(figsize = (5,5))
# dictionary- map numbers to colors
colmap = {1:'m', 2:'b', 3:'g', 4:'k'}
# map will assign colors to labels
```

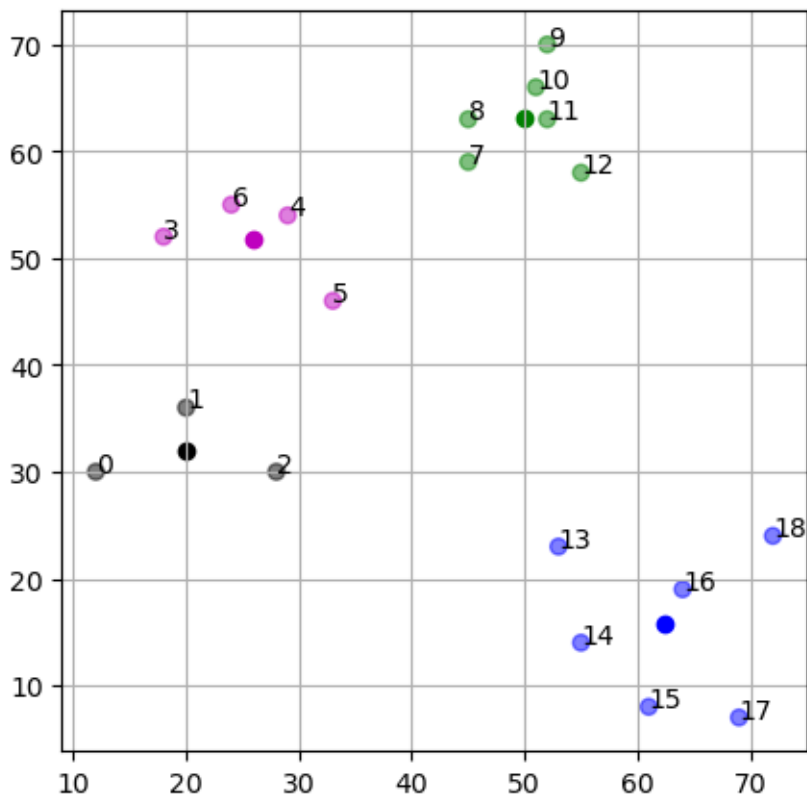
```

colors = map(lambda x: colmap[x+1], labels)

colors1=list(colors)
plt.scatter(x, y, color= colors1, alpha = 0.5 )
# plotting the centroids wrt color
for idx, centroid in enumerate(centroids):
    plt.scatter(*centroid, color = colmap[idx+1])
# labeling the points as 0,1,2,...18
for i, txt in enumerate(n):
    plt.annotate(txt, (x[i], y[i]))
plt.grid()

```

<map object at 0x0000025EB57B2080>



```

[16]: from sklearn.metrics import silhouette_score
      silhouette_score(data, labels)

```

[16]: 0.6179376814567372

```

[86]: print(colors1)

```

['k', 'k', 'k', 'm', 'm', 'm', 'm', 'g', 'g', 'g', 'g', 'g', 'g', 'b', 'b', 'b',

'b', 'b', 'b']

[]: