Clustering

K-Means Clustering

posted Jan 30, 2018, 9:11 AM by Atul Rana **[updated 43 minutes ago]**

Introduction

K-means clustering partitions a dataset into a small number of clusters by minimizing the distance between each data point and the centre of the cluster it belongs to. Since the distance is Euclidean, the model assumes the form of the cluster is spherical and all clusters have a similar scatter.

```
import numpy as np
import pandas as pd
```

Get the Data

```
dataset = pd.read_csv('Mall_Customers.csv')
X = dataset.iloc[:, [3, 4]].values
# y = dataset.iloc[:, 3].values
```

As K-means clustering algorithm is unsupervised learning algorithm, means we don't have the labels of actual prediction result we just find groups in given data.

```
dataset.head()
```

	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

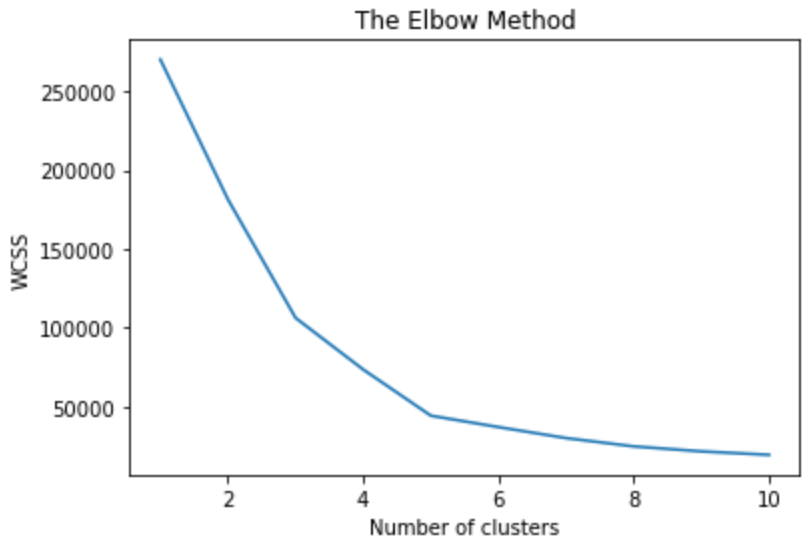
```
X
```

```
array([[ 15,  39],
       [ 15,  81],
       [ 16,   6],
       [ 16,  77],
       [ 17,  40],
       [ 17,  76],
       [ 18,   6],
       [ 18,  94],....])
```

Using the elbow method to find the optimal number of clusters.

K-means clustering is based on minimizing data to some cluster based on Euclidean distance.

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

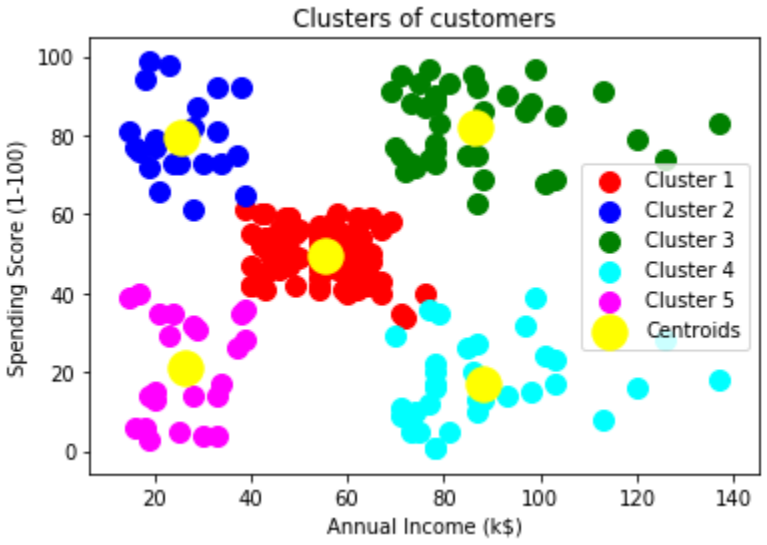


Of course cluster distance to cluster data point minimizes as number of clusters increases but we are interested in minimum no. of cluster.(elbow)

```
kmeans = KMeans(n_clusters = 5, init = 'k-means++')
y_kmeans = kmeans.fit_predict(X)
```

```
kmeans.inertia_
```

44448.455447933709




```
kmeans.labels_
```

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

```
kmeans.cluster_centers_
```

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

1-1 of 1