

# K- Means Algorithm

```
In [2]: import warnings
warnings.filterwarnings('ignore')
```

```
In [3]: # importing libraries

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

```
In [4]: # importing dataset

dataset=pd.read_csv(r"C:\Users\Jan Saida\OneDrive\Documents\Desktop\Excel sheets\Mall_Customers.csv")
dataset
```

```
Out[4]:
```

|     | 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 × 5 columns

```
In [5]: x=dataset.iloc[:,[3,4]].values  
x
```

```
Out[5]: 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)
```

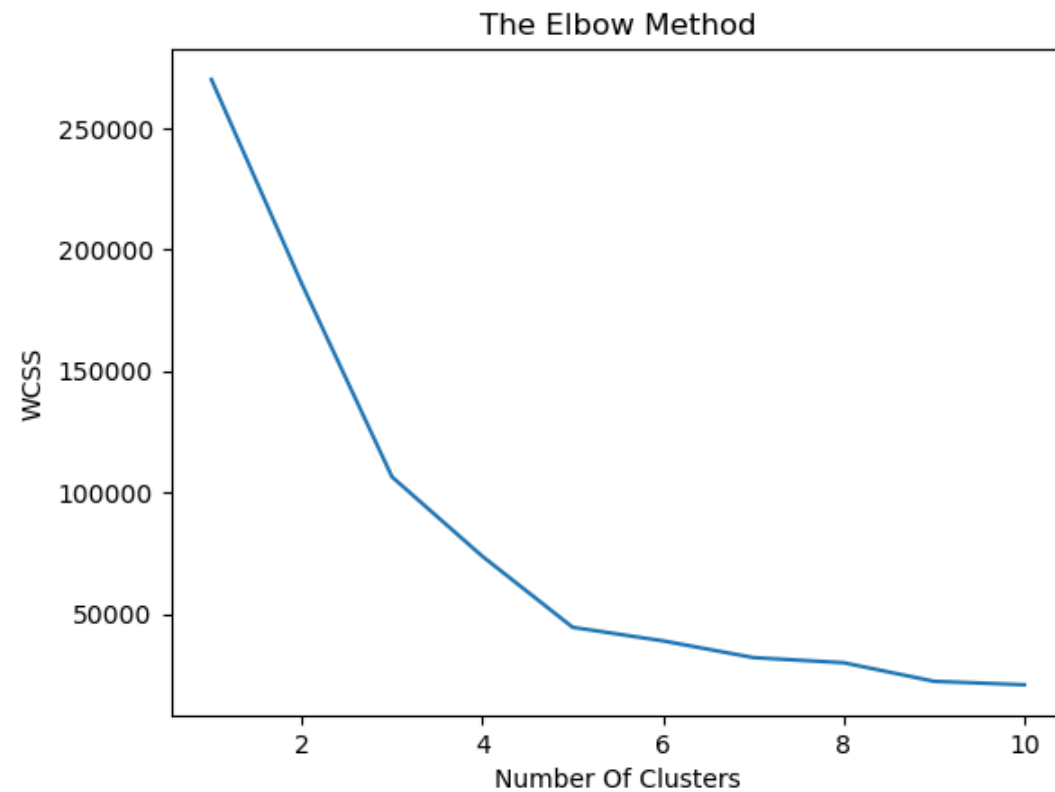
```
In [6]: # using elbow method to find the optimal number of clusters
```

```
from sklearn.cluster import KMeans
```

```
In [7]: wcss=[]
```

```
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init="k-means++",random_state=0)
    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()
```

```
# wcss we have very good parameter called inertia_ credit goes to sklearn , that computes the sum of square , formula it will compute
```



In [8]: *# training the K-Means model on the dataset*

```
kmeans=KMeans(n_clusters=5,init='k-means++',random_state=0)
kmeans
```

Out[8]:

▼ **KMeans** ⓘ ?  
KMeans(n\_clusters=5, random\_state=0)

In [9]: `y_kmeans=kmeans.fit_predict(x)`  
`y_kmeans`



```
Out[9]: array([3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4,
               3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 0,
               3, 4, 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, 1, 2, 1, 0, 1, 2, 1, 2, 1,
               0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
               2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
               2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
               2, 1])
```

```
In [10]: # visualising the clusters
```

```
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()
```



In [13]: *# pickling the file*

```
import pickle
filename='Mall_prediction.pkl'
with open (filename,'wb') as file:
    pickle.dump(kmeans,file)
print('Model has been saved and Pickled as Mall_prediction.pkl')
```

Model has been saved and Pickled as Mall\_prediction.pkl

In [14]: **import** os  
os.getcwd()

Out[14]: 'C:\\Users\\Jan Saida'