

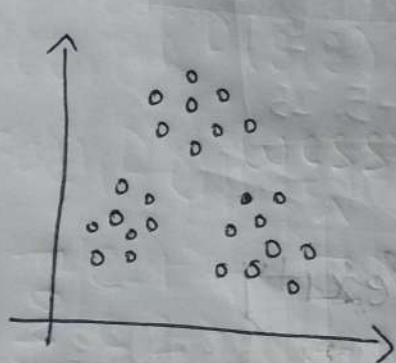
## K-means clustering

K-means clustering is popular unsupervised ML. Algo. used to partition data into groups or clusters.

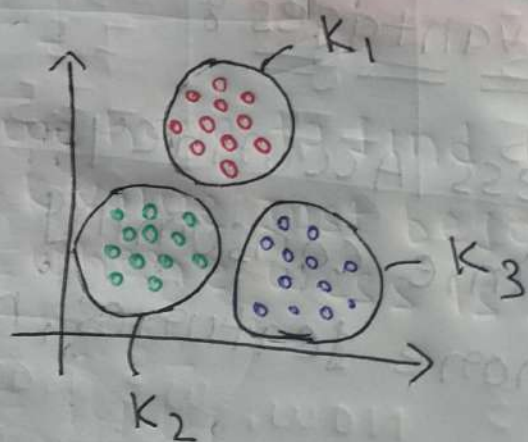
'K' in K-means represent No. of clusters you want to create. Goal is to partition data into 'K' clusters.

centroid: center of mass or AVG. of All data points in cluster.

$$\text{Euclidean Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



K=3  
K-means  
⇒



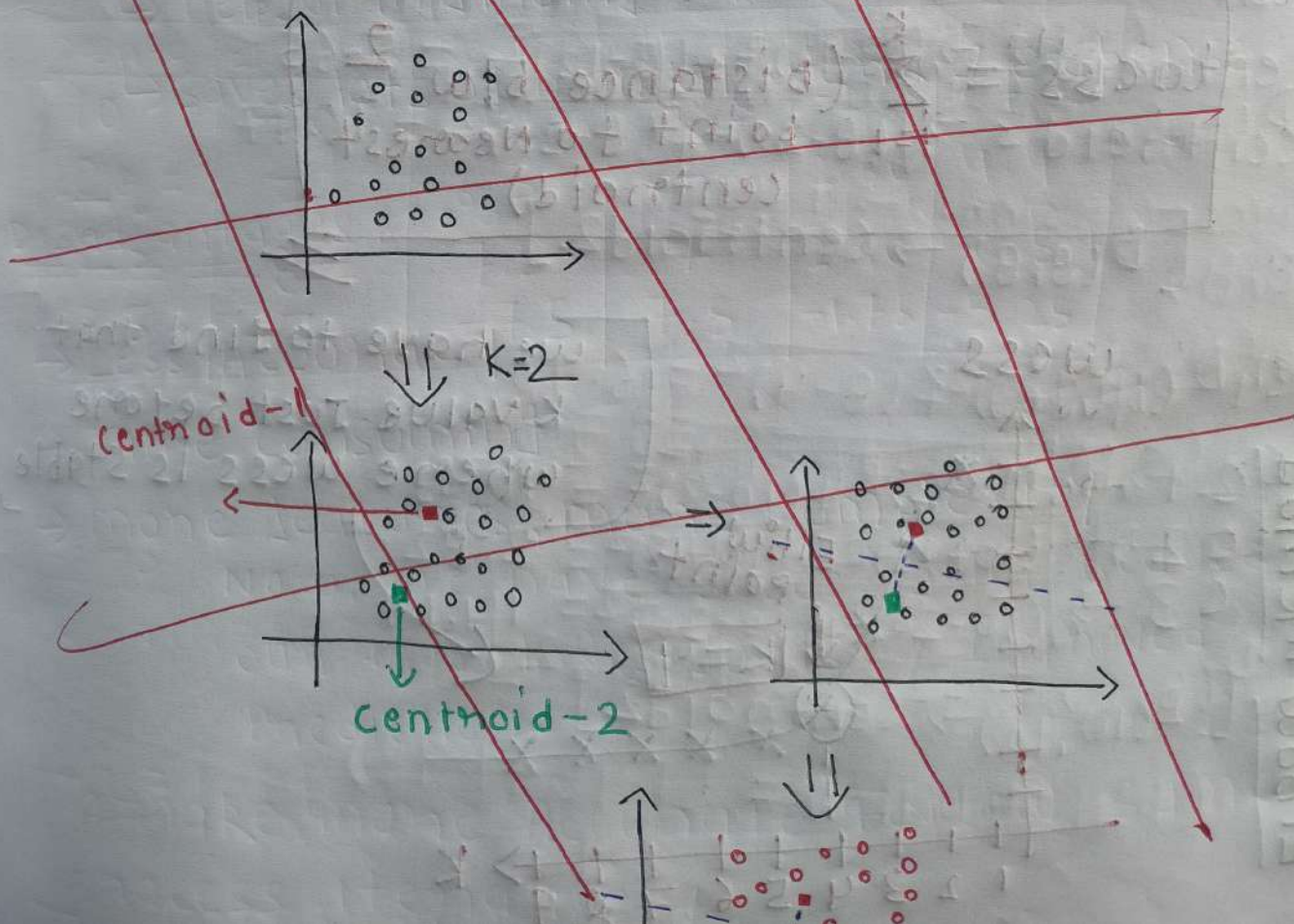
↳ steps



working:

- ↳ select  $K$ .
- ↳ select centroids.
- ↳ Assign each data point to their closest centroid.
- ↳ calculate variance & place new centroid of each cluster.
- ↳ Repeat 3<sup>rd</sup> step, reassign each datapoint to new closest centroid of each cluster.
- ↳ If any reassignment occurs, then go to step-4 else go to Finish.
- ↳ model is Ready.

Graphical intuition:





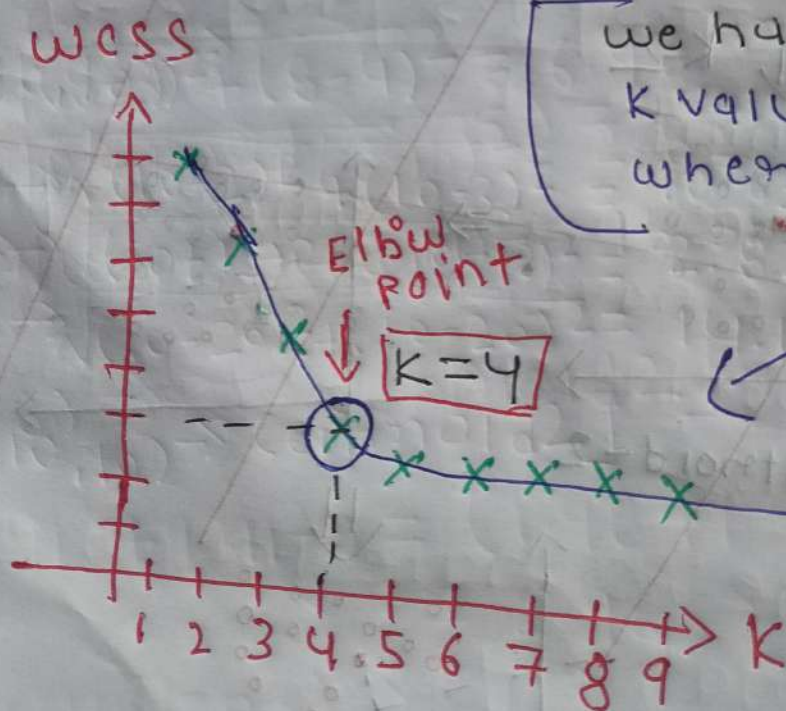
\* How to choose "K No. of clusters"?

→ Elbow Method?

Elbow method is most optimal way to find No. of clusters. This method use concept of wcss value. (within cluster sum of sq.)

$$WCSS = \sum_{i=1}^K (\text{distance b/w point to Nearest centroid})^2$$

Elbow Method



we have to find that K value just before where wcss is stable



# \* K-Means Clustering Implementation \*

<u>X</u>	<u>Y</u>
2	3
3	4
5	6
8	8
10	10
12	12
<del>14</del>	<del>14</del>
<del>16</del>	<del>16</del>
18	18
20	20

Random centroid:

centroid 1  $\rightarrow (2, 3)$

centroid 2  $\rightarrow (12, 12)$

centroid 3  $\rightarrow (20, 20)$

If K=3,

$\hookrightarrow$  clusters = 3

$\hookrightarrow$  No. of centroid = 3

For  $D(2, 3) \rightarrow$

Centroid  
to  
Datapoint

Euclidean  
Distance

$$= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Distance to centroid 1 =  $\sqrt{(2-2)^2 + (3-3)^2} = 0$  ✓

" " centroid 2 =  $\sqrt{(2-12)^2 + (3-12)^2} = 13.45$

" " centroid 3 =  $\sqrt{(2-20)^2 + (3-20)^2} = 24.75$

$[D(2, 3) \rightarrow \text{centroid 1}]$



For  $D(3,4) \rightarrow$

$$\text{centroid 1} \rightarrow \sqrt{(3-2)^2 + (4-3)^2} = 1.41$$

$$\text{centroid 2} \rightarrow \sqrt{(3-12)^2 + (4-12)^2} = 12.04$$

$$\text{centroid 3} \rightarrow \sqrt{(3-20)^2 + (4-20)^2} = 23.34$$

$[D(3,4) \rightarrow \text{centroid 1}]$

For  $D(5,6) \rightarrow$

$$\text{centroid 1} \rightarrow \sqrt{(5-2)^2 + (6-3)^2} = 4.24$$

$$\text{centroid 2} \rightarrow \sqrt{(5-12)^2 + (6-12)^2} = 9.21$$

$$\text{centroid 3} \rightarrow \sqrt{(5-20)^2 + (6-20)^2} = 20.51$$

$[D(5,6) \rightarrow \text{centroid 1}]$

For  $D(8,8) \rightarrow$

$$\text{centroid 1} \rightarrow \sqrt{(8-2)^2 + (8-3)^2} = 7.81$$

$$\text{centroid 2} \rightarrow \sqrt{(8-12)^2 + (8-12)^2} = 5.65$$

$$\text{centroid 3} \rightarrow \sqrt{(8-20)^2 + (8-20)^2} = 16.97$$

$[D(8,8) \rightarrow \text{centroid 2}]$

For  $D(10,10) \rightarrow$

$$C_1 \rightarrow \sqrt{(10-2)^2 + (10-3)^2} = 10.63$$

$$C_2 \rightarrow \sqrt{(10-12)^2 + (10-12)^2} = 2.82$$

$$C_3 \rightarrow \sqrt{(10-20)^2 + (10-20)^2} = 14.14$$

$[D(10,10) \rightarrow \text{centroid 2}]$



$$[D(12,12) \rightarrow \text{centroid 2}]$$

$$\text{For } D(18,18) \rightarrow$$

$$C1 \rightarrow \sqrt{(18-2)^2 + (18-3)^2} = 21.93$$

$$C2 \rightarrow \sqrt{(18-12)^2 + (18-12)^2} = 8.48$$

$$C3 \rightarrow \sqrt{(18-20)^2 + (18-20)^2} = 2.82$$

$$[D(18,18) \rightarrow \text{centroid 3}]$$

$$[\text{For } D(20,20) \rightarrow \text{centroid 3}]$$

Update centroid

X	Y	centroid
2	3	1
3	4	1
5	6	1
8	8	2
10	10	2
12	12	2
18	18	3
20	20	3

$$\begin{cases} \text{New centroid 1} = (3, 4) \\ \text{New centroid 2} = (10, 10) \\ \text{New centroid 3} = (19, 19) \end{cases}$$

Now calculate new centroids →

$$\text{New centroid 1} = X: (2+3+5)/3 = 3.33 \rightarrow 3$$

$$Y: (3+4+6)/3 = 4.33 \rightarrow 4$$

$$\text{New centroid 2} = X: (8+10+12)/3 = 10$$

$$Y: (8+10+12)/3 = 10$$

$$\text{New centroid 3} = X: (18+20)/2 = 19$$

$$Y: (18+20)/2 = 19$$



make clusters with new centroid:

$D(2,3) \rightarrow$

$$C1 \rightarrow \sqrt{(2-3)^2 + (3-4)^2} = 1.41$$

$$C2 \rightarrow \sqrt{(2-10)^2 + (3-10)^2} = 10.63$$

$$C3 \rightarrow \sqrt{(2-19)^2 + (3-19)^2} = 23.34$$

$[D(2,3) \rightarrow \text{centroid 1}]$

$[D(3,4) \rightarrow \text{centroid 1}]$

$D(5,6) \rightarrow$

$$C1 \rightarrow \sqrt{(5-3)^2 + (6-4)^2} = 2.82$$

$$C2 \rightarrow \sqrt{(5-10)^2 + (6-10)^2} = 6.40$$

$$C3 \rightarrow \sqrt{(5-19)^2 + (6-19)^2} = 19.10$$

$[D(5,6) \rightarrow \text{centroid 1}]$

$D(8,8) \rightarrow$

$$C1 \rightarrow \sqrt{(8-3)^2 + (8-4)^2} = 6.40$$

$$C2 \rightarrow \sqrt{(8-10)^2 + (8-10)^2} = 2.82$$

$$C3 \rightarrow \sqrt{(8-19)^2 + (8-19)^2} = 15.55$$

$[D(8,8) \rightarrow \text{centroid 2}]$

centroid	Y	X
1	3	4
2	10	10
3	19	19



$$D(10,10) \rightarrow$$

$$C_1 \rightarrow \sqrt{(10-3)^2 + (10-4)^2}$$

$$C_2 \rightarrow \sqrt{(10-10)^2 + (10-10)^2} = 0 \checkmark$$

$$C_3 \rightarrow \sqrt{(10-19)^2 + (10-19)^2}$$

$$[D(10,10) \rightarrow \text{centroid 2}]$$

$$D(12,12) \rightarrow$$

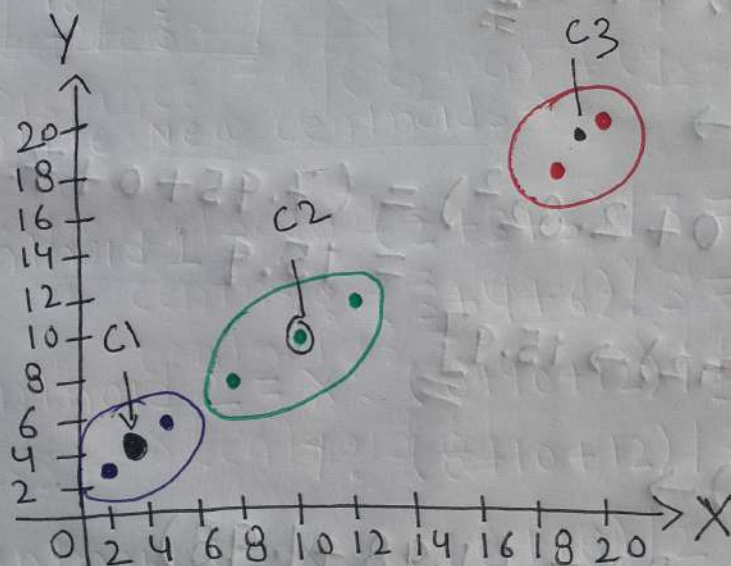
$$C_1 \rightarrow \sqrt{(12-3)^2 + (12-4)^2} = 12.04$$

$$C_2 \rightarrow \sqrt{(12-10)^2 + (12-10)^2} = 2.82 \checkmark$$

$$C_3 \rightarrow \sqrt{(12-19)^2 + (12-19)^2} = 9.89$$

same for  $D(18,18)$  &  $D(20,20)$

we get same clusters :-





$$WCSS = \sum_{i=1}^n \sum_{j=1}^k d(c_j, x_i)^2$$

$WCSS \rightarrow$  within clusters sum of square

$n \rightarrow$  Total No. of data points

$k \rightarrow$  Total No. of clusters

$c_j \rightarrow$  centroid of  $j$ th clusters

$x_i \rightarrow$   $i$ th data point

$d(c_j, x_i) \rightarrow$  Euclidean Distance

$$WCSS(k=3) = WCSS(\text{cluster1}) + WCSS(\text{cluster2}) + WCSS(\text{cluster3})$$

$WCSS(\text{cluster1}) \rightarrow$

$$(1.41^2 + 0^2 + 2.82^2) = (1.98 + 0 + 7.95) = 9.93$$

$$[WCSS(\text{cluster1}) \rightarrow 9.93]$$

$WCSS(\text{cluster2}) \rightarrow$

$$(2.82^2 + 0 + 2.82^2) = (7.95 + 0 + 7.95) = 15.9$$

$$[WCSS(\text{cluster2}) \rightarrow 15.9]$$

$WCSS(\text{cluster3}) \rightarrow$

$$(1.41^2 + 1.41^2) = (1.98 + 1.98) = 3.96$$

$$[WCSS(\text{cluster3}) \rightarrow 3.96]$$



$$\text{WCSS}(K=3) = 9.93 + 15.9 + 3.96 \\ = 29.79$$

Like this,

$$\text{WCSS}(K=1) = 582.37$$

$$\text{WCSS}(K=2) = 144.16$$

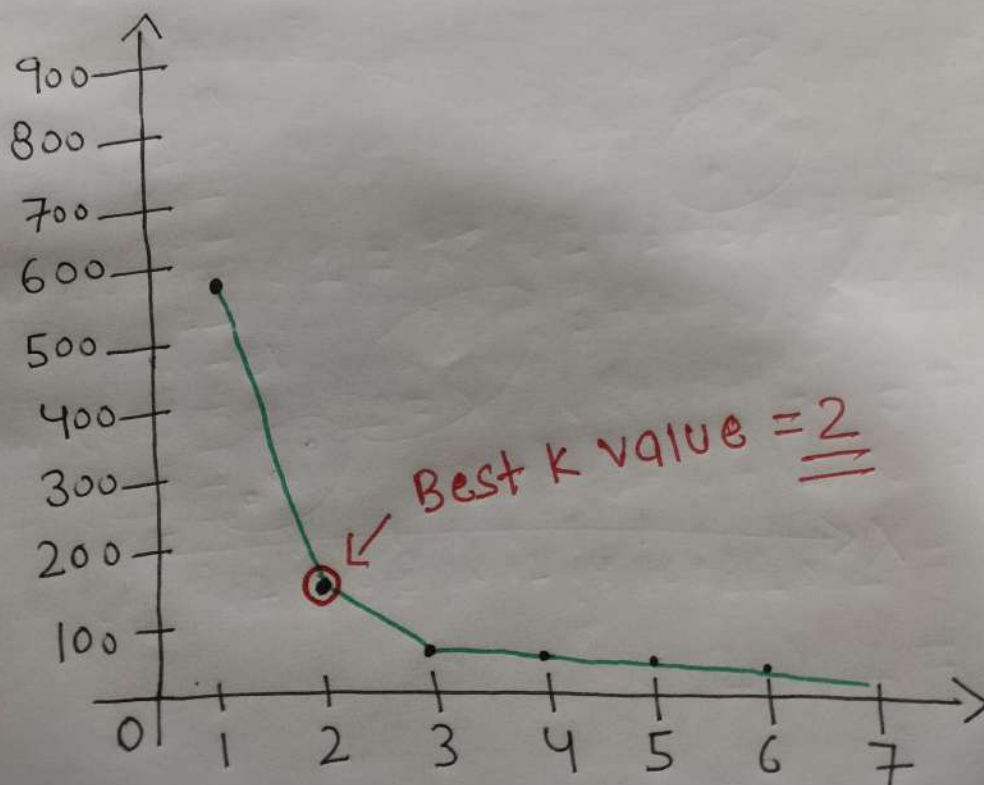
$$\text{WCSS}(K=3) = 29.79$$

$$\text{WCSS}(K=4) = 15.50$$

$$\text{WCSS}(K=5) = 9.0$$

$$\text{WCSS}(K=6) = 5.0$$

Elbow method :





```
In [1]: import pyforest
        from sklearn.cluster import KMeans
        import warnings
        warnings.filterwarnings('ignore')
        %matplotlib inline
```

```
In [2]: # Create a dictionary with the data
        data = {
            'X': [2, 3, 5, 8, 10, 12, 18, 20],
            'Y': [3, 4, 6, 8, 10, 12, 18, 20]
        }

        # Create a DataFrame from the dictionary
        df = pd.DataFrame(data)

        # Display the DataFrame
        df
```

```
Out[2]:
```

	X	Y
0	2	3
1	3	4
2	5	6
3	8	8
4	10	10
5	12	12
6	18	18
7	20	20

## Find K Value (No. Of Clusters):



$$WCSS = \sum_{C_k}^{C_n} \left( \sum_{d_i \in C_i}^{d_m} distance(d_i, C_k)^2 \right)$$

Where,

$C$  is the cluster centroids and  $d$  is the data point in each Cluster.

```
In [ ]: wcss= []
        for k in range(1,7):
            kmeans= KMeans(n_clusters=k,init='k-means++')
            kmeans.fit(df)
            wcss.append(kmeans.inertia_)
```

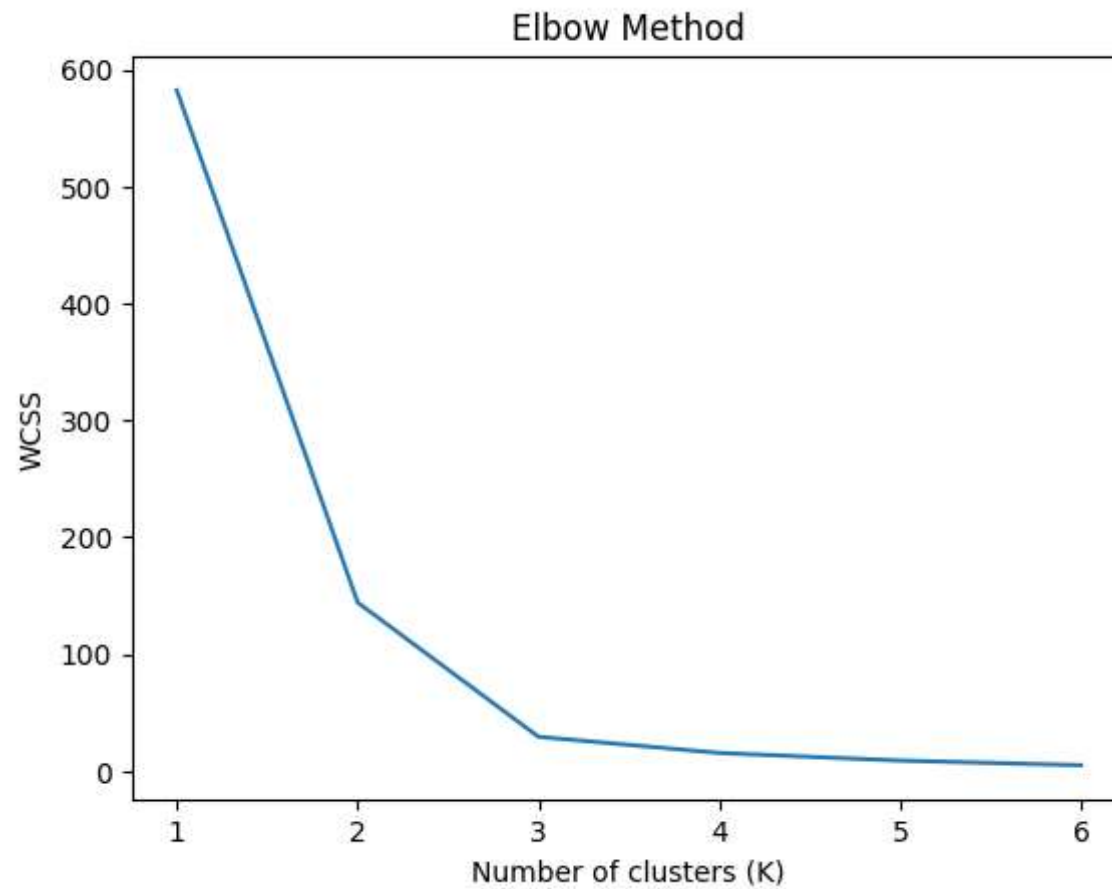
```
In [4]: WCSS_Values=pd.DataFrame({'K Value': [1,2,3,4,5,6], 'WCSS':wcss})
        WCSS_Values
```

```
Out[4]:
```

	K Value	WCSS
0	1	582.375000
1	2	144.166667
2	3	29.333333
3	4	15.500000
4	5	9.000000
5	6	5.000000

```
In [5]: plt.plot(range(1,7), wcss)
        plt.title('Elbow Method')
        plt.xlabel('Number of clusters (K)')
        plt.ylabel('WCSS')
        plt.show()
```





**Calculate Centroid and Eucliden Distance for Clustering:**

$$d(x, y) = \sqrt{\sum_{i=1}^n (y_i - x_i)^2}$$

In [6]: df

Out[6]:

	X	Y
0	2	3
1	3	4
2	5	6
3	8	8
4	10	10
5	12	12
6	18	18
7	20	20

In [7]: *# Random Centroids For Initialize: df index 0 and 7*  
c1= (3,4)  
c2= (10,10)



```
In [8]: #Euclidean Distance Each Data Point to Each(2) centroids:

# c1 and c2 to all data points euclidean distance:
df['Distance_to_c1']= np.sqrt((df['X']-c1[0])**2+(df['Y']-c1[1])**2)
df['Distance_to_c2'] = np.sqrt((df['X'] - c2[0])**2 + (df['Y'] - c2[1])**2)
```

```
In [9]: df
```

```
Out[9]:
```

	X	Y	Distance_to_c1	Distance_to_c2
0	2	3	1.414214	10.630146
1	3	4	0.000000	9.219544
2	5	6	2.828427	6.403124
3	8	8	6.403124	2.828427
4	10	10	9.219544	0.000000
5	12	12	12.041595	2.828427
6	18	18	20.518285	11.313708
7	20	20	23.345235	14.142136

```
In [10]: # Assign each data point to the cluster with the minimum distance
df['Cluster'] = np.where(df['Distance_to_c1'] < df['Distance_to_c2'], 'Cluster 1', 'Cluster 2')
```

```
In [11]: df
```

```
Out[11]:
```

	X	Y	Distance_to_c1	Distance_to_c2	Cluster
0	2	3	1.414214	10.630146	Cluster 1
1	3	4	0.000000	9.219544	Cluster 1
2	5	6	2.828427	6.403124	Cluster 1
3	8	8	6.403124	2.828427	Cluster 2
4	10	10	9.219544	0.000000	Cluster 2
5	12	12	12.041595	2.828427	Cluster 2
6	18	18	20.518285	11.313708	Cluster 2
7	20	20	23.345235	14.142136	Cluster 2

## Update Centroids by Calculating Average:

```
In [12]: new1_c1 = (df[df['Cluster'] == 'Cluster 1']['X'].mean(), df[df['Cluster'] == 'Cluster 1']['Y'].mean())
new1_c2 = (df[df['Cluster'] == 'Cluster 2']['X'].mean(), df[df['Cluster'] == 'Cluster 2']['Y'].mean())
```

```
In [13]: new1_c1, new1_c2
```

```
Out[13]: ((3.3333333333333335, 4.333333333333333), (13.6, 13.6))
```

```
In [14]: #Euclidean Distance Each Data Point to Each(2) new_centroids:

# new_c1 and new_c2 to all data points euclidean distance:
df['Distance_to_new1_c1'] = np.sqrt((df['X'] - new1_c1[0])**2 + (df['Y'] - new1_c1[1])**2)
df['Distance_to_new1_c2'] = np.sqrt((df['X'] - new1_c2[0])**2 + (df['Y'] - new1_c2[1])**2)
```

```
In [15]: df
```



Out[15]:

	X	Y	Distance_to_c1	Distance_to_c2	Cluster	Distance_to_new1_c1	Distance_to_new1_c2
0	2	3	1.414214	10.630146	Cluster 1	1.885618	15.713688
1	3	4	0.000000	9.219544	Cluster 1	0.471405	14.301049
2	5	6	2.828427	6.403124	Cluster 1	2.357023	11.476933
3	8	8	6.403124	2.828427	Cluster 2	5.934831	7.919596
4	10	10	9.219544	0.000000	Cluster 2	8.749603	5.091169
5	12	12	12.041595	2.828427	Cluster 2	11.571037	2.262742
6	18	18	20.518285	11.313708	Cluster 2	20.047167	6.222540
7	20	20	23.345235	14.142136	Cluster 2	22.874051	9.050967

In [16]:

```
# Assign each data point to the cluster with the minimum distance
df['new_Cluster'] = np.where(df['Distance_to_new1_c1'] < df['Distance_to_new1_c2'], 'Cluster 1', 'Cluster 2')
```

In [17]:

```
df
```

Out[17]:

	X	Y	Distance_to_c1	Distance_to_c2	Cluster	Distance_to_new1_c1	Distance_to_new1_c2	new_Cluster
0	2	3	1.414214	10.630146	Cluster 1	1.885618	15.713688	Cluster 1
1	3	4	0.000000	9.219544	Cluster 1	0.471405	14.301049	Cluster 1
2	5	6	2.828427	6.403124	Cluster 1	2.357023	11.476933	Cluster 1
3	8	8	6.403124	2.828427	Cluster 2	5.934831	7.919596	Cluster 1
4	10	10	9.219544	0.000000	Cluster 2	8.749603	5.091169	Cluster 2
5	12	12	12.041595	2.828427	Cluster 2	11.571037	2.262742	Cluster 2
6	18	18	20.518285	11.313708	Cluster 2	20.047167	6.222540	Cluster 2
7	20	20	23.345235	14.142136	Cluster 2	22.874051	9.050967	Cluster 2

```
In [18]: # Create a scatter plot for the data points in each cluster
plt.scatter(df[df['new_Cluster'] == 'Cluster 1']['X'], df[df['new_Cluster'] == 'Cluster 1']['Y'], label='Cluster 1',
plt.scatter(df[df['new_Cluster'] == 'Cluster 2']['X'], df[df['new_Cluster'] == 'Cluster 2']['Y'], label='Cluster 2',

# Plot the centroids as well
plt.scatter(new1_c1[0], new1_c1[1], label='Centroid 1', marker='x', c='black', s=100)
plt.scatter(new1_c2[0], new1_c2[1], label='Centroid 2', marker='x', c='green', s=100)

# Add Labels and a Legend
plt.xlabel('X')
plt.ylabel('Y')
plt.legend()

# Show the scatter plot
plt.show()
```

