# ✯ K-Means Clustering Algorithm

- K-means is an unsupervised learning algorithm that partitions data into k distinct clusters by iteratively minimizing the distance between points and their cluster centers.

- Core idea :- K-means answers 'given k groups, how do i assign each point to a group such that points within a group are as similar as possible?

- Mathematical objective :-

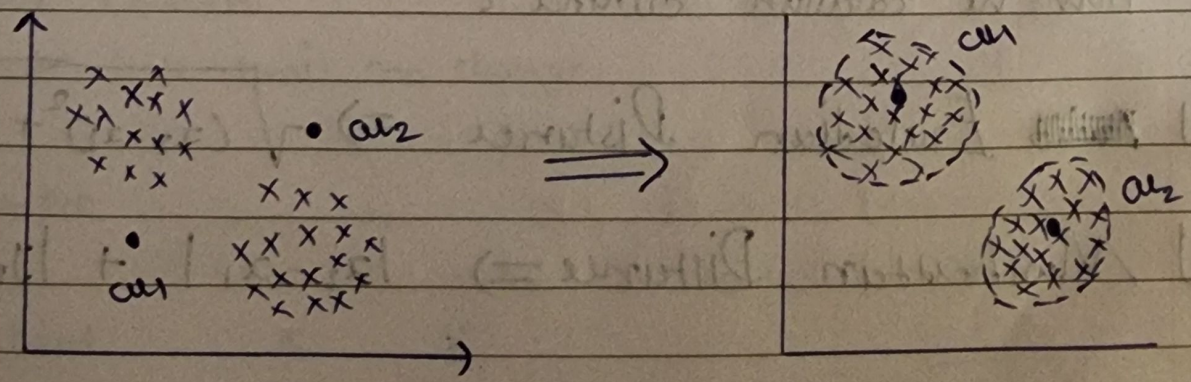
We minimize the within-cluster sum of squares (WCSS)

$$J = \sum_{k=1}^{K} \sum_{x_i \in c_k} \|x_i - c_k\|^2$$

↳ Squared euclidean distance

$c_k$ = centroid

$x_i$ = points

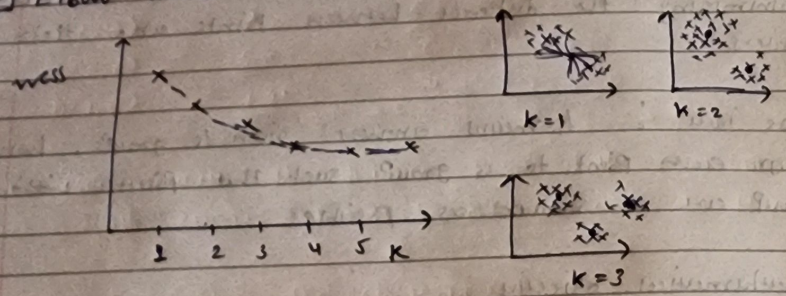



If k = 2 centroids

- Now the Question is how to find best k value?

- To find the optimal K value we have multiple methods

1) Elbow method (WCSS Analysis)




- When wcss decrease sharply slow down we take that k value.

- When we only have k=1 the wcss is too high as single centroid try to fit all points, k=2, now 2 centroid try to fit it's own spece and due to similarity the wcss decrease.

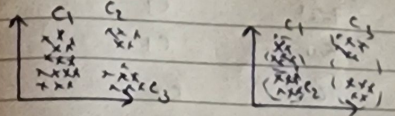
* How we calculate distance ?

1) Euclidian Distance $\Rightarrow \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$

2) Manhattan Distance $\Rightarrow |x_2-x_1| + |y_2-y_1|$

* Their is one Problem with random initialization of centroids which we call random initialization Trap.

1) Duplicate centroids
2) Poor Convergence
3) Inconsistent results
4) suboptimal wcss




Is random is good




In worst case It cluster like this

- To address this issue we can use

1) K-means ++ :- use maximum distance between centroids
2) multiple random starts :- not best

* Algorithm steps :-

1) Choose K (number of centroid)
2) Initialize K centroid randomly
3) Assign each point to nearest centroid
4) update centroid.
5) Repeate 3-4 until no change.

- example :-

| Point | X | Y |
|---|---|---|
| A | 1 | 2 |
| B | 2 | 2 |
| C | 6 | 5 |
| D | 7 | 5 |

$c_1 = (1,1)$

$c_2 = (7,5)$

Step 1 :-

Step 2 :-

| Point | $c_1$ | $c_2$ | cluster |
|---|---|---|---|
| A(1,1) | 0 | 7.21 | $C_1$ |
| B(2,1) | 2 | 6.4 | $C_1$ |
| C(6,5) | 6.4 | 2 | $C_2$ |
| D(7,5) | 7.21 | 0 | $C_3$ |

- We calculate the distance from both centroid and assign the cluster to minimum distance to that centre

Step 3 : Update the centroid

$$c_{u_1} = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1.5, 1)$$

$$c_{u_2} = \left(\frac{6+7}{2}, \frac{5+5}{2}\right) = (6.5, 5)$$

Step 4 : Reassign and do some until the below of any condition true

1) no assignment changes
2) centroid stop moving
3) WCSS stop decreasing

Result : $C_1 : \{A, B\}, u_1(1.5, 1)$
$C_2 : \{C, D\}, u_2(6.5, 5)$

WCSS : $C_1$: $(1-1.5)^2 + (1-1)^2 + (2-1.5)^2 + (1-1)^2 = 0.5$

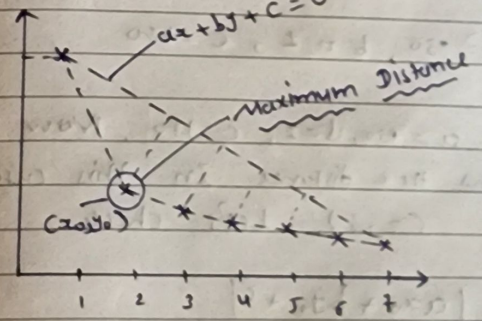
$C_2$: $(6-6.5)^2 + (5-5)^2 + (7-6.5)^2 + (5-5)^2 = 0.5$

* To validate the K value we have 2 methods

1) KneeLocator
2) Silhouette

1) Knee Locator :-

- After plotting the graph between the WCSS and K value we try to find the maximum distance from the K=1 to K=n line




- Now the question is how to find the maximum distance? Math help us.

- The distance $d = \frac{ax_0 + by_0 + c}{\sqrt{a^2+b^2}}$, The perpendicular distance from $(x_0, y_0)$ to line $ax+by+c=0$

- In above case we have 2 points (1, 150) and (1, 0) and first we want line.

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0-150}{6-1} = -30$$

$y = mx + c$

$$m = \frac{y - y_1}{x - x_1}$$

If I know any one point and slop I get bias and line equation

$$m(x - x_1) = y - y_1$$

$$y - 0 = -30(x - 1)$$

$$y = -30x + 180$$ → Rearrange → $30x + y - 180 = 0$

Standard form : $ax + by + c = 0$

$$a = 30,\ b = 1,\ c = -180$$

- now we know $a = +30,\ b = 1,\ c = -180$, Now for any point we can find the distance. In this case the closest point is (2,3). Let's check

$$d = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

$$= \frac{|30(2) + 1(3) + (-180)|}{\sqrt{30^2 + 1^2}}$$

$$= 117/\sqrt{901} = 117/30.02 = 3.90$$

which is maximum.

2) silhouette

- Silhouette answers : Is each point closer to its own cluster than to other clusters?

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

$a(i)$ = cluster a, i point avg distance from (within cluster) all other point in same cluster a.

$b(i)$ = cluster b, i point avg distance from (outer clusters) all other point in b cluster.

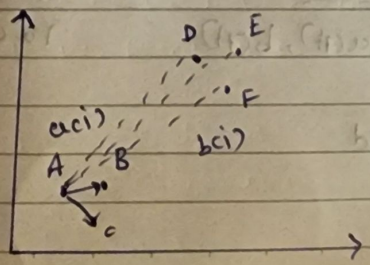
$a(i) >> b(i)$ = Closure to other cluster → wrong cluster

$a(i) << b(i)$ = closure to own cluster → good

$a(i) == b(i)$ = equally close to own and other

Denominator : $\max(a(i), b(i))$ use to normalize the score to range $[-1, +1]$.

$s(i) = +1$ good
$s(i) = 0$ between
$s(i) = -1$ misclassified




$a(A)$ = avg distance within a cluster points

$b(A)$ = avg distance other cluster point

- example :-

- we have 6 points , K=2

cluster 1 : A(1,1), B(2,1), C(2,2)
cluster 2 : D(8,8), E(9,8), F(9,9)

step 1 : a(A) :

distance (A,B) = $\sqrt{(1-2)^2 + (1-1)^2} = 1$

d(A,C) = 1.41

$a(A) = \frac{1 + 1.41}{2} = 2.21$

step 2 : b(A) :

d(A,D) = 9.90
d(A,E) = 10.63
d(A,F) = 11.31

b(A) = 10.61

step 3 : S(A)

$S(A) = \frac{b(A) - a(A)}{\max(a(A), b(A))} = \frac{9.40}{10.61} = 0.89$

A is well clustered

- calculate for all point

| Point | a(i) | b(i) | s(i) |
|---|---|---|---|
| A(1,1) | 1.21 | 10.61 | 0.89 |
| B(2,1) | 1.21 | 9.90 | 0.88 |
| C(2,2) | 1.21 | 9.23 | 0.87 |
| D(8,8) | 1.21 | 9.23 | 0.87 |
| ⋮ | ⋮ | ⋮ | ⋮ |

avg s(i) = 0.88 - good k value else change and retry this