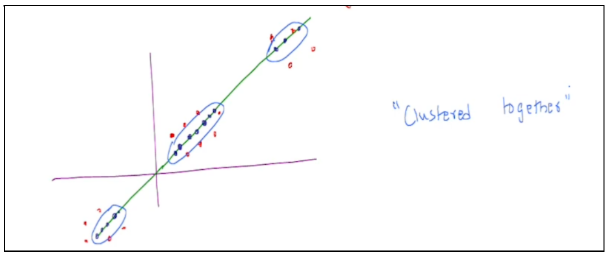
Clustering

Goal:

$\rightarrow \{x_1, x_2, \cdots, x_n\} \quad x_i \in \mathbb{R}^d$

$\rightarrow$ Partition the dataset into k different clusters.



Example:

$\rightarrow \{x_1, x_2, x_3, x_4, x_5\}$ and $k = 5$

Now partition the data in different ways,

One way:

$\boxed{x_1, x_2, x_5} \quad \boxed{x_3} \quad \boxed{x_4}$

Another way:

$\boxed{x_1, x_4} \quad \boxed{x_5} \quad \boxed{x_2, x_3}$

There are multiple ways to parition the boxes. In fact, one could argue how many different possible parition ways are possible:

$\rightarrow$ We want to partition 5 points into 3 boxes

$\rightarrow$ # possible ways: 3^5 (this also include possibilities which leads to empty boxes.)

In General, If we want to partition N data points into K clusters then Number of possible ways of partitioning are

$$k^N$$

Notation

$\rightarrow x_1, x_2, \cdots, x_n$ {Data-points}

$\rightarrow z_1, z_2, \cdots, z_n \quad z_i = \{1, 2, \cdots, k\}$ { Cluster Indicator }

If my 10^{th} data point goes into 4^{th} cluster then $z_{10} = 4$.

Question-1: Given a cluster assignment, how good is it?

$$\rightarrow F(z_1, \cdots, z_n) = \sum_{i=1}^{n} || x_i - \mu_{z_i} ||^2 \quad (Objective\ Function)$$

$\rightarrow \mu_{z_i}$ is the mean of z_i cluster.

$$\mu_k = \frac{\sum_{i=1}^{n} x_i 1(z_i = k)}{\sum_{i=1}^{n} 1(z_i = k)} \qquad 1(u) = \begin{cases} 1 & if\ u\ true \\ 0 & o|w \end{cases}$$

Example:

I have 5 data points and 2 clusters. (N = 5, K = 2)

$\rightarrow x_1, x_2, x_3, x_4, x_5$

$\rightarrow z_1 = 1, z_2 = 2, z_3 = 1, z_4 = 1, z_5 = 2$

$$\mu_1 = \frac{x_1 + x_3 + x_4}{3} ; \mu_2 = \frac{x_2 + x_5}{2}$$

Goal:

$$\min_{\{z_1, \cdots, z_n\}} \sum_{i=1}^{n} || x_i - \mu_{z_i}||^2$$

Problem:

Too many possibilities, **NP-Hard Problem**

LLYOD'S ALGORITHM / K-MEANS ALGORITHM

$$z_1^0,\ z_2^0,\ \cdots,\ z_n^0\ ; \qquad z_i^0\ \in \{\ 1,\ 2,\ \cdots,\ k\}$$

0 here represent the iteration number

<u>UNTIL CONVERGENCE</u>

- Compute Means

$$\forall\ k \qquad \mu_k^t\ =\ \frac{\sum_{i=1}^n x_i 1\left(z_i^t\ = k\right)}{\sum_{i=1}^n 1\left(z_i^t\ = k\right)}$$

- Re-assignment step

$$\forall\ i \quad z_i^{t+1} = \underset{k}{argmin}\ ||\ x_i - \mu_k^t||^2$$

→ This step will calculate the distance of data-point to the mean of every cluster and if it found any k which has strictly less distance or be closer to the data point, then that data-point will be assigned to that k.
→Assuming if the mean of current assignment is smallest, then don't reassign.

FACT:

Llyod's algorithm converges
→ Converged Solution may not be optimal.
→ But produces "reasonable" clusters in practice

What is this algorithm in brief:
We start with an potentially arbitrary partition of the data points into K clusters.
and then, every point looks at the distance of the point to its own mean and compares it with a mean of other clusters and if it finds a cluster whose mean is closer to it in terms of distance squared, then it jumps to that cluster. And this happens for every point and that is how a reassignment of points to partition or clusters happen. And this goes on until Convergence.

Convergence: It is the state of the algorithm where every data-point has settled in the most optimal cluster i.e. Now no data-point want to jump to any other cluster.

Questions:
1. Does the algorithm converge ?
2. What is the nature of clusters ?
3. How do we do Initialise this algorithm ?
4. What is the K that we should use that works for this data set ?

1.CONVERGENCE

Does lloyd's algorithm converge ? → Yes

PROOF

FACT-1:

Let $x_1,\ x_2,\ \cdots,\ x_l\ \in\ \mathbb{R}^d$

- **What we are asking is ?** → we have a bunch of points, we want a v, which minimises the sum of the distance squared , or the average of the distance squared and let call it v^*.
- v^* is nothing but a point in R^d from which the sum of squared distance of some bunch of data points is minimum

$$v^* \ =\ \underset{v\,\in\,\mathbb{R}^d}{argmin} \sum_{i=1}^{l} ||x_i - v||^2 \tag{1}$$

- We can view the above objective as a function of v, take derivative and set it to zero and solve for v.
- The Answer to above Question that what is the point from which the sum of squared distance of some bunch of data-points is minimum is the mean of those data points.

$$v^* \ =\ \frac{1}{l}\sum_{i=1}^{l} x_i \quad (mean\ of\ those\ bunch\ of\ data\ points) \tag{2}$$

Now, we going to prove why lloyd's algorithm actually converges in a very nice way.

→ Let's assume we are at some iteration t of llyod's algorithm

→Let's assume our current assignment of points to clusters look likes as follows

$$z_1^t,\ z_2^t,\ \cdots,\ z_n^t\ ; \qquad z_i^t \in \{1,\ \cdots,\ k\}$$

where t correspponds the iteration number and sub-script corresponds to the particular data point.

Now, let's define Mean of cluster k in iteration t

$$\mu_k^t \leftarrow Mean\ of\ cluster\ k\ in\ iteration\ t$$

- We are assuming that at step t, the algorithm does not converge. Well, if the algorithm converges in step t , then there is nothing to prove.

→ Let's say we update our assignment to next steps assignments

$$z_1^{t+1},\ z_2^{t+1},\ \cdots,\ z_n^{t+1} \qquad z_i^{t+1} \in \{1,\ \cdots,\ k\}$$

- How do we argue it is a better partition than the previous partition ?
  - Well, we have to look at our objective function(whose main focus is to quantify the goodness of partition) which given a partition will tell you how good that partition is by assigning it a number

by assigning it a number.
  - We want smallest value for that objective function. The partition that gives us the smallest value is the best partition.

$$\underbrace{\sum_{i=1}^{n} ||x_i - \mu^t_{z_i^{t+1}}||^2}_{mean\ of\ cluster\ where\ x_i\ wants\ to\ go\ to} < \underbrace{\sum_{i=1}^{n} ||x_i - \mu^t_{z_i^t}||^2}_{mean\ of\ current\ cluster\ where\ x_i\ assigned\ to} \tag{3}$$

What is $\sum_{i=1}^{n} ||x_i - \mu^t_{z_i^t}||^2$ quantity ?

- We are in the t^{th} iteration.
- Every point is being measured in this expression to the mean of the box to which it is assigned to.
- We are computing the distance of every point to the mean of the box to which it is assigned to.
- x_i is assigned to the box z_i^t.
  - This is by definition where x_i is assigned, z_i^t is a cluster indicator of the i-th data point in the t-th iteration. So, i-th data point goes to the z_i^t which is some number between 1 to k.
- So, basically, What this is capturing is just the sum of distances of each point to its own mean in the i-th iteration.

$$F\left(z_1^t, \ \cdots, \ z_n^t\right) = \sum_{i=1}^{n} ||x_i - \mu^t_{z_i^t}||^2$$

What is $\sum_{i=1}^{n} ||x_i - \mu^t_{z_i^{t+1}}||^2$ quantity ?

- This quantity is trying to capture the distance of each point to the mean it is being assigned to in the next round.
- We are still measuring the distance of each data point to the current mean but not of the same box to which it is assigned currently but to the box to which it will be assigned in the next iteration.
- Well, some data-points do not want to jump, for those data points which is happy with their own mean in such case $z_i^{t+1} = z_i^t$.
- But there may be some data points which might find that there is a mean which I am closer to so I want to jump and that is when the reassignment actually happens.
- It is an intermediate quantity

Why does a point want to jump ?

- Well, this is because the distance to where it is comparing itself to is strictly lesser than it is currently assigned to.

Why does the above two expression have this kind of relationship ?

- This is simply by Algorithm choice
- Algorithm makes this choice to make a jump only when it finds a different cluster where the mean is strictly closer to than the current cluster.

the mean is strictly closer to then the current cluster.

But remember,

- This quantity $\sum_{i=1}^{n} ||x_i - \mu^t_{z_i^{t+1}}||^2$ is not the objective value of the partition that we would get in the $t+1$ round.
- Why ?
  - I have not made the jump yet, But the objective value in the $t+1$ round will be counted or approved only when for that specific round no other jump will happen.
  - So, this expression is just depicting the situation saying, I am closer to the other guy, and I am going to measure my distance to the other guy.
  - Whereas, If you really want to measure the objective function value in the next round you should compare each point to the mean that you get in the next round, so after you have made the assignments. (i.e. after lot of jumping every point has been reassigned for this round)
  - So, this is an intermediate Quantity

How does the objective function change after we make the partition change ?
→ If it becomes smaller then our partition get better otherwise not.

$$\sum_{i=1}^{n} ||x_i - \mu^{t+1}_{z_i^{t+1}}||^2 \leqslant \sum_{i=1}^{n} ||x_i - \mu^{t}_{z_i^{t+1}}||^2 \tag{4}$$

What is this $\sum_{i=1}^{n} ||x_i - \mu^{t+1}_{z_i^{t+1}}||^2$ Quantity ?

- This quantity is measuring the sum of squared distance of each data point to the mean of the clusters or box in $t+1$ round .
- Partition objective value in the $t+1$ evaluated in the t+1 round
-

$$F\left(z_1^{t+1}, \ \cdots , \ z_n^{t+1}\right) = \sum_{i=1}^{n} ||x_i - \mu^{t+1}_{z_i^{t+1}}||^2$$

In equation (4) we are comparing th

where

$$\sum_{i=1}^{n} ||x_i - \mu^{t+1}_{z_i^{t+1}}||^2 = \sum_{i=1}^{n} \sum_{k=1}^{k} ||x_i - \mu^{t+1}_{k}||^2 \cdot \mathbb{1}\left(z_i^{t+1} = k\right)$$

What does this $\sum_{i=1}^{n} \sum_{k=1}^{k} ||x_i - \mu^{t+1}_{k}||^2 \cdot \mathbb{1}\left(z_i^{t+1} = k\right)$ mean ?

- This is just the breakdown of the entire contribution of all the data points, which is what is summed here into each cluster.

for every k

$$\sum_{i \in C_k} ||x_i - \mu_k^{t+1}||^2 \leqslant \sum_{i \in C_k} ||x_i - v||^2$$

$$\leqslant \sum ||x_i - \mu_k^t||^2$$

C_k → It is the cluster whose index number is k.

what have we gained by all of this ?
→ The objective function strictly reduces after each re-assignment.

$$F\left(z_1^{t+1}, \cdots, z_n^{t+1}\right) < F\left(z_1^t, \cdots, z_n^t\right)$$

Why does the above relation means that algorithm should converge ?

- All we are saying, we are in some partition, which has some objective value and now reassignment steps happens and the objective value strictly reduces.
- The Question is that why strict reduction in objective vlaue imply the algorithm should converge ?
    - Algorithm convergence means that you are reaching a specific parition where you know every point is happy with its own mean, no reassignment happens after that.
- The answer is:
    - The finiteness of number of partitions is what is allowing us to say algorithm has to converge.
    - There are only "FINITE" number of assignments
    - Algorithm must converge.

→ Every time a reassignment happens in the lloyd's of the K-means algorithm, our objective function, which is measured as the sum of the distance of the data points to the mean of the clusters to which it is assigned distance squared, to which it is assigned, reduces strictly monotonically and that implies at some point, you will hit a partition where every point is happy with its own, and it will not change anymore, because it cannot infinitely keep reducing when you have only a finite bunch of possibilities.

→ It does not mean algorithm is always going to take k^n rounds to converge.
→ We are also not talking about the goodness of the partitions or the clustering that will result from this algorithm.

2. Nature of clusters

Question: What kind of clusters are being produced by the llyod's algorithm ?

To understand this, let's look at the simple case, when $k = 2$.

- Llyod's algorithm produces 2 clusters with means μ_1 and μ_2.

Question: What can we say about the points assigned to cluster 1 and cluster 2 ?

Let's say we are first focusing on cluster 1. The argument for cluster 2 will be similar.

By Algorithm construction and its converging nature,

For points in cluster 1,

$$||x - \mu_1||^2 \ \leqslant \ ||x - \mu_2||^2 \tag{5}$$

The above statement means, that points in cluster 1 are closer to the mean of cluster 1 than the mean of cluster 2.

If we solve the above equation,

$$||x||^2 + ||\mu_1||^2 - 2x^T\mu_1 \leqslant ||x||^2 + ||\mu_2||^2 - 2x^T\mu_2 \tag{6}$$

$$||\mu_1||^2 - 2x^T\mu_1 \leqslant ||\mu_2||^2 - 2x^T\mu_2$$

$$x^T(\mu_2 - \mu_1) \ \leqslant \ \frac{||\mu_2||^2 - ||\mu_1||^2}{2} \tag{7}$$

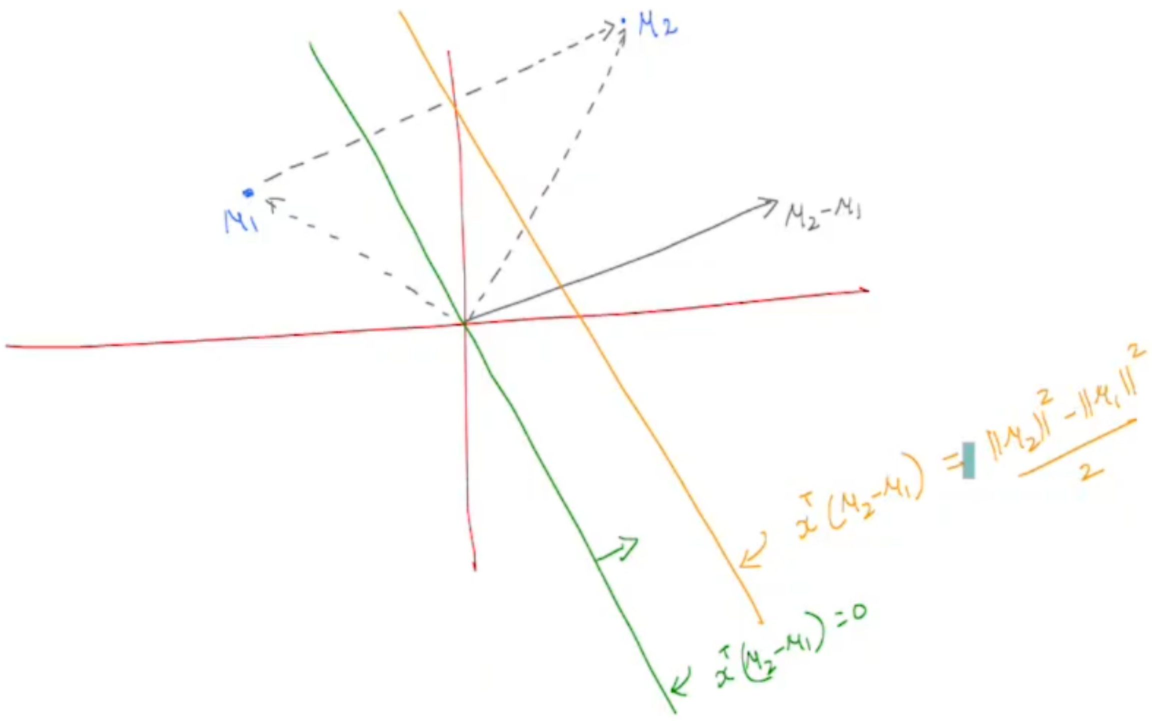
The result that has come out is some kind of $x^T w \ \leqslant \ b$.

The statement $x^T(\mu_2 - \mu_1) \ \leqslant \ \frac{||\mu_2||^2 - ||\mu_1||^2}{2}$ is only saying that $x^T(\mu_2 - \mu_1)$ is atmost $\frac{||\mu_2||^2 - ||\mu_1||^2}{2}$.

What is $\mu_2 - \mu_1$? I mean which vector is this ?

→ The answer is in the figure

→ Another thing, to notice, we have drawn a line perpendicular to $\mu_2 - \mu_1$. Here, $x^T(\mu_2 - \mu_1) = 0$.



The data points which is in cluster 1 must satisy equation (7) so they must lie in a region which is less than $\frac{||\mu_2||^2 - ||\mu_1||^2}{2}$.

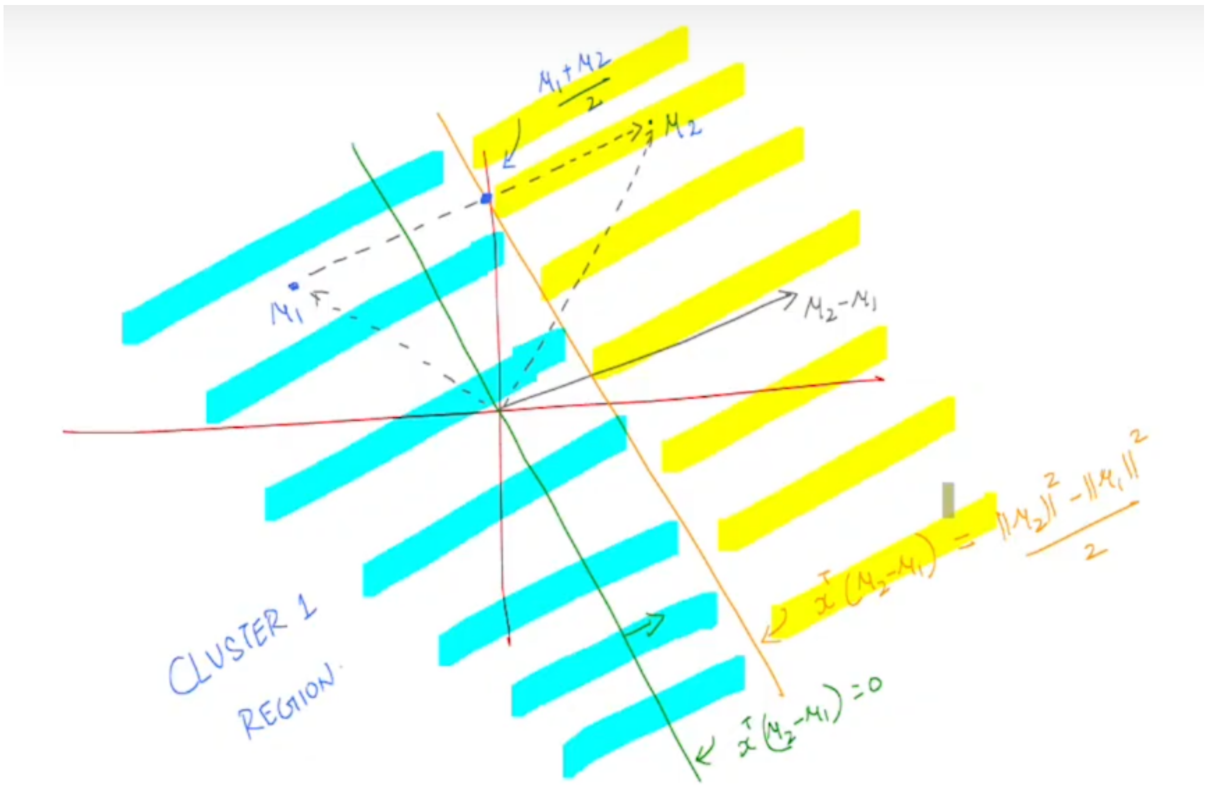
There comes a question, How much did I move to right means from green line to yellow line ? we can find the answer if have any point that satisfy the equation:

$$Let\ x = \frac{\mu_1 + \mu_2}{2}$$

$$Then,$$

$$x^T(\mu_2 - \mu_1) = \left(\frac{\mu_1 + \mu_2}{2}\right)^T (\mu_2 - \mu_1)$$

$$= \frac{||\mu_2||^2 - ||\mu_1||^2}{2}$$



$\frac{\mu_1+\mu_2}{2}$
μ_2
μ_1
$\mu_2-\mu_1$
$x^T(\mu_2-\mu_1) = \frac{\|\mu_2\|^2-\|\mu_1\|^2}{2}$
CLUSTER 1
REGION.
$x^T(\mu_2-\mu_1)=0$

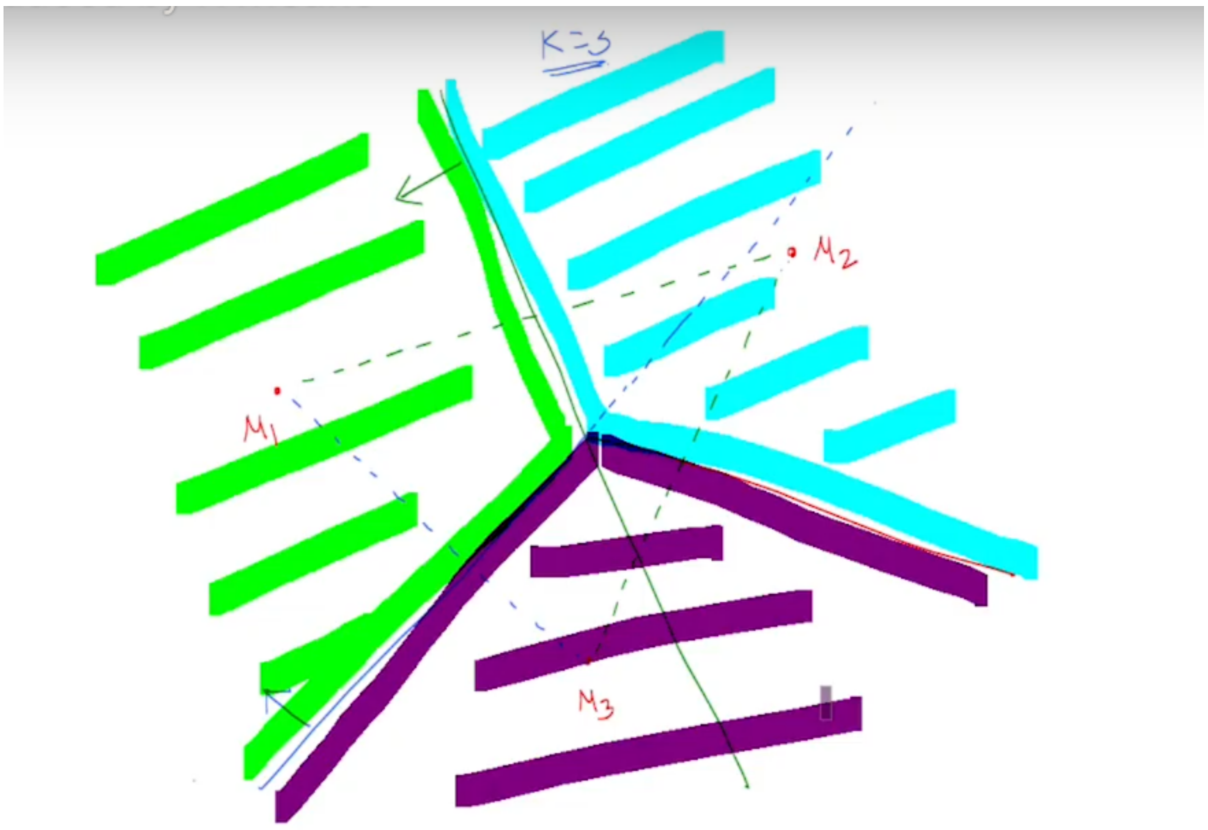
Question: What happens when k=3 ?

Well, now algorithm has to converge not with just 2 means but three means.

For the illustration purpose we are taking 2-Dimensional data but all of we are dsoing is applicable to high dimensional data.

Let suppose, we have three clusters $C_1,\ C_2,\ C_3$ and $\mu_1,\ \mu_2,\ \mu_3$ be their respective means. Then by convergence criteria of llyod's algorithm we know that every point assigned to C_1 is closer to μ_1 than μ_2 or μ_3 .

Where are all the points that are closer to μ_1 than to two other means ?
→ The answer to this question is already mentioned i.e. It is just the left hand side region of the perpendicular bisector of the imaginary line joining the two means.



We are dividing the entire space in this case $\mathbb{R}^2$ which is where the data points are into K different regions using intersections of half space(half space in two dimension means you draw a line and then
it divides the entire space into two parts.

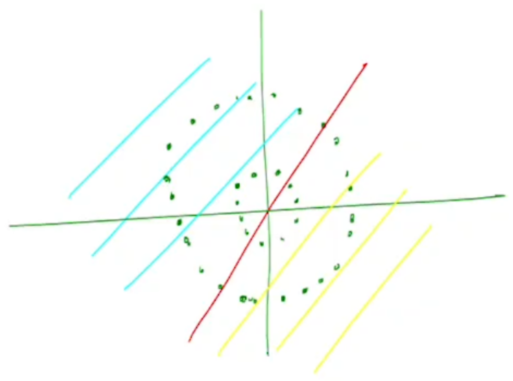
Cluster regions are intersections of HALF SPACES.

These half spaces has name → VORONOI REGIONS

The nature of clusters are voronoi partition.

Well, we have seen K-means algorithm uses voronoi partition means a line or intersection of line that separates the clusters . It means if the clustering can be more beneficial in non-linear way then K-means algorithm cannot properly find cluster (i.e. non-linear seperation) but it will give some line

For example: in the below picture We have data points in two circles by intitution we can see that there can be two clusters one is outer concentric circle and another one is inner concentric circle but K-means cannot identify it, where it will give some line as a partition which will not serve our purpose.



How to solve this Problem ?
→ Like we have seen in Kernel PCA, when we have non-linear relationship in dimension d then we
can go to higher Dimension D using Kernel Function or Kernel Matrix Where these non-linear relation may behave as linear.
→ Similarly, there is a way we can Kernelize K-means and it is called the Kernel K-means algorithm .
→ Kernel K - Means clustering algorithm can help in Spectral clustering.

In summary, we can understand it like this, There are no voronoi regions in the low dimensional space, that gives me natural clustering of this data but then if we map this data to some high dimensional space, then natural clusters would kind of separate out into different voronoi regions and in the high dimensional space, we can perform simple llyod's algorithm.

3.INITIALISATION

We could initialize it however we want, but some initialisation might lead us to a better cluster.

Question: What is the good initialisation so that we can end up in good clusters ?

→ Random initialisation can also lead to good cluster but that will not be efficient.

Then what is a good idea?

We will pick K-means uniformly at random from our dataset and then I will put each mean into thier box separately.

For example, I have 1000 data-points and I picked data point number 5, 72, 58, 65, 90. and then I put 5 in C_1, 72 in C_2 , and so on. and for now I am assuming them as mean of their respective cluster.

Now we will put the remaining data points into these boxes or clusters based on, like, we will compare the distance of the data point with every mean and then we will simply put that data point into that box which has closer distance than remaining ones.

This does not mean algorithm is converged now, but at least it is a good start.

This was a way that a lot of people do.

Another principled way is K-Means++ Algorithm.

K- Means++ Algorithm

- K-means++ algorithm is nothing but K-Means algorithm with a better initialization.

Idea:

- The Idea is still the same, we still want to pick k points from the dataset.
- Instead of picking these means randomly we can think this what exactly we want these means to look like in the end of the algorithm, If we think carefully we will find that these means have most distance between them i.e. They are as away as possible from each other.

This ++ algorithm intialise itself in a iterative fashion:

- Choose first mean μ_1^0 uniformly at random from $\{x_1, \ \cdots , \ x_n\}$. (super script is iteration number)

$$for\, l \ = \ 2, \ ....., \ k$$

$$choose\ \mu_l^0 \qquad probabilisticaly \ \propto \ to\ score$$

score is a +ve number. Higher the score, more the probability of picking that.

$$\forall\, x \qquad S(x) \;=\; \min_{j=1,\ldots,l-1} ||x - \mu_j^0||^2$$

Guarentee why this algorithm works fine ?

$$E\left[\sum_{i=1}^{n} ||x_i - \mu_{z_i}||^2\right] \;\leqslant\; O(log\, k)\left[\min_{z_1,\cdots,z_n} \sum_{i=1}^{n} ||x_i - \mu_{z_i}||^2\right]$$

The Downside of this Algorithm is that it requires a lot of time in doing this initialization.

Computationally, It is not much efficient.

4. Choice of K

Let see what happens to our objective value if we choose k = n i.e. k equal to the number of data points, in this case

$$F(z_1, \cdots, z_n) = \sum_{i=1}^{n} ||x_i - \mu_{z_i}||^2 = 0$$

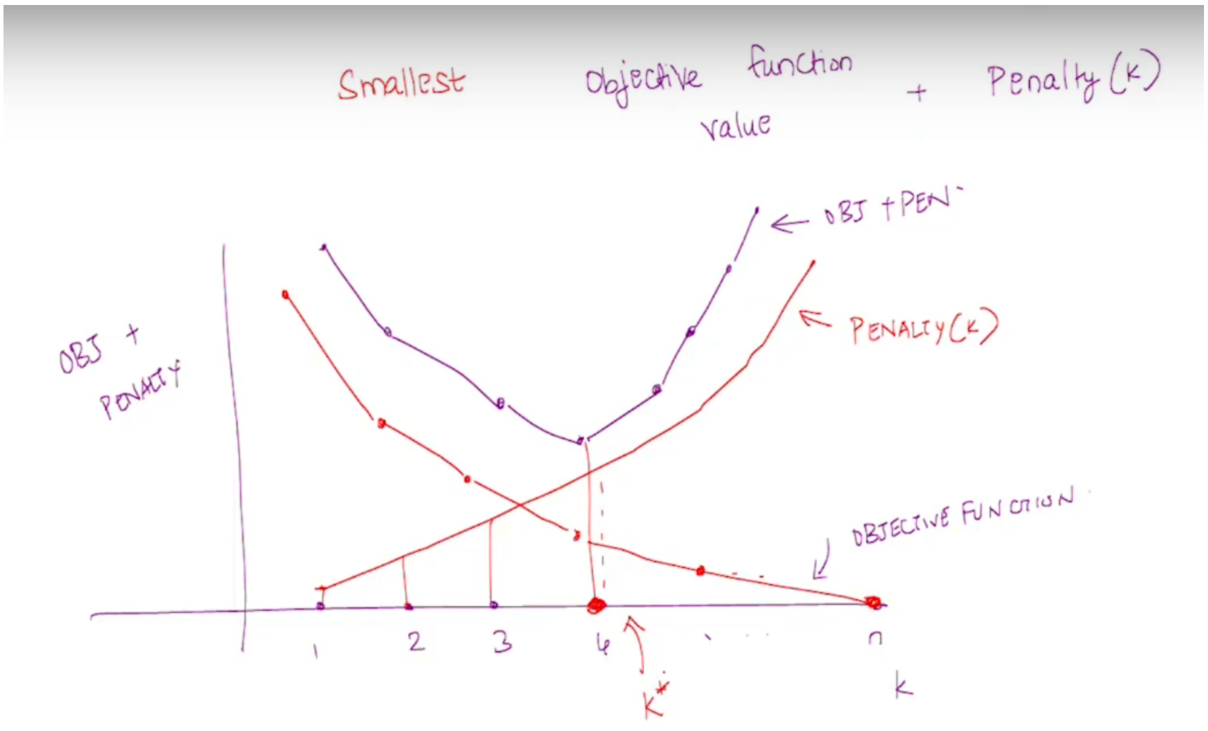
→ Making K large can definitely reduce the objective value but that is not our goal.

→we want K to be as small as possible.

We want to find k that has the smallest objective functions + penality(K).

Penalty is just a function of K and think of it as how much we are penalizing for asking for one more cluster

We want Objective function to be as small as possible and Penalty(K) also as small as possible by the thing is With the increase of number of K, Objective value starts decreasing but penalty(K) starts increasing so we have to find a optimal vlaue where they both balances.



Some Common criterion for choosing obj. func + penalty(K)

A.I.C - Akaike Information Criterion

$$\left[2K - 2log\left(L(\theta^*)\right)\right]$$

B.I.C - Bayesian Information Criterion

$$\left[K\, log(n)\; -\; 2\, log\left(L\left(\theta^{*}\right)\right)\;\right]$$

Summary:

- Convergence → Yes
- Nature of Clusters → Voronoi Regions
- Initialisation → K-Means++
- Choice of K → OBJ + Penality(K)

Working out an example for K-Means

1. Example

Consider the following dataset in $\mathbb{R}^2$.

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

It is a toy dataset. (Toy dataset $\rightarrow$ Something which we can handle and visualize)

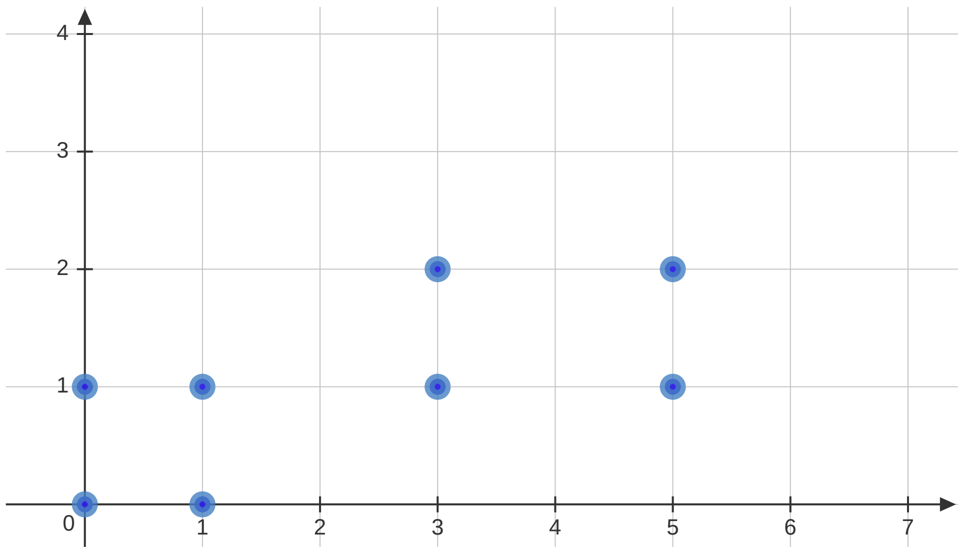
We will try to cluster this dataset.

First thing we need to do is, visualize this dataset.

1.1. Visualize the dataset

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

Shape of the dataset is $d \times n$, where $d = 2, n = 8$.



By looking at the data, I am saying A good choice of k would be 2.

$\rightarrow$This is not possible in practice, because in practice we can't visualize the dataset.

1.2. How many Cluster assignments are possible with k means and n data points ?

$$k^n$$

For this problem, if we have k clusters, then number of assignments would be k^8.

1.3. Run k-means with $k = 2$ and $z^0 = [1\ 2\ 2\ 2\ 2\ 2\ 2\ 2]$. Plot the voronoi regions. To which cluster does $(2, 2)$ belong ? Find the value of the objective function at the end.

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

Step-0: Initialisation

The initial cluster assignment is as follows:

The first point belongs to cluster-1 and rest of the points belongs to cluster-2

$$z^0 = [1\ 2\ 2\ 2\ 2\ 2\ 2\ 2]$$

Now, we are going to find mean of each cluster.

$$\mu_1^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mu_2^0 = \begin{bmatrix} 18/7 \\ 8/7 \end{bmatrix}$$

$$= \frac{2}{7}\begin{bmatrix} 9 \\ 4 \end{bmatrix} = \begin{bmatrix} 2.57 \\ 1.14 \end{bmatrix}$$

$\mu_1 \longrightarrow$ mean of the first cluster

$\mu_2 \longrightarrow$ mean of the second cluster

Step-1: First iteration of k-means

Step-1.1: Compute the cluster assignments (Computing z)

$x_i \rightarrow$ The data point

$z_i \rightarrow$ The cluster indicator variable of the the data point

$||x_i - \mu_1||^2 \rightarrow$ the distance of the data point from mean of cluster-1

$||x_i - \mu_2||^2 \rightarrow$ The distance of the data point from mean of cluster-2

Then we will compare both the distances and then Assign this data point accordingly to z_{t+1} .

x_i	z_i^0	$\|\|x_i - \mu_1^0\|\|^2$	$\|\|x_i - \mu_2^0\|\|^2$	z_{t+1}^0
$(0, 0)$	1	smaller		1
$(0, 1)$	2	1 smaller	> 1	1
$(1, 0)$	2	smaller		1

Now, we have a new cluster assignments.

$$z^1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Step-1.2. Compute the cluster means (Computing μ)

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

$$\mu_1 = \begin{bmatrix} (0+0+1+1) \ / \ 4 \\ (0+1+0+1) \ / \ 4 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}$$

$$\mu_2 = \begin{bmatrix} 4 \\ 1.5 \end{bmatrix}$$

At the end of first iteration $\mu_1 = (0.5, 0.5)$ and $\mu_2 = (4, 1.5)$

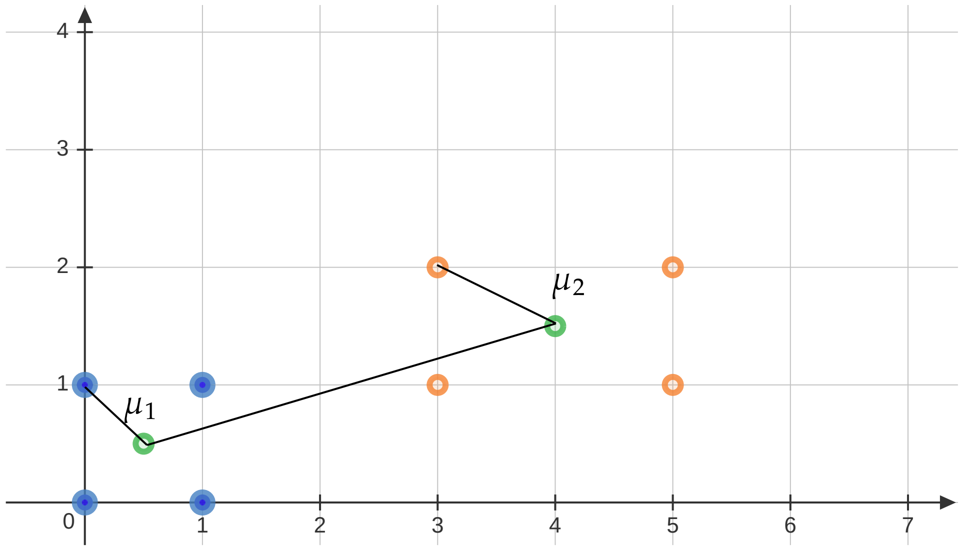
Step-2: Repeat step-1 until the algorithm converges.

Step-1: again

Now mean are: $\mu_1^1 = (0.5, 0.5)$ and $\mu_2^1 = (4, 1.5)$

x_i	z_i^1	z_{t+1}^1
$(0, 0)$	1	1
$(0, 1)$	1	1
$(1, 0)$	1	1
$(1, 1)$	1	1
$(3, 1)$	2	2

we will do this by geometry,
we will plot the means.



Step-2.1: There is no change in the cluster assignments. It means that algorithm is converged.

We see that $z_1 = z_2$. It means we can stop at this state.

Now,

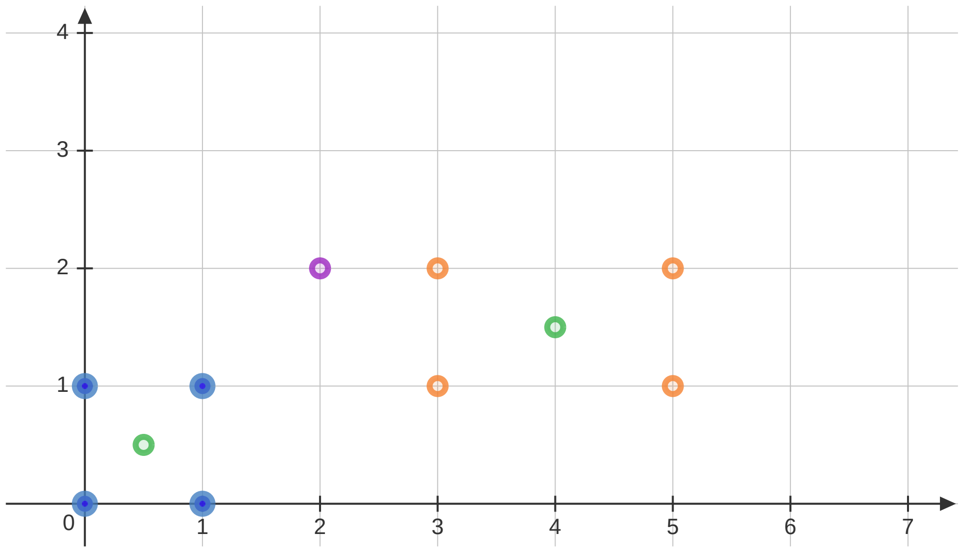
The Value of Objective Function:

$$F(D) = \sum_{i=1}^{n} ||x_I - \mu_{z_i}||^2$$

$$F(D) = \left(0.5^2 + 0.5^2\right) \times 4 + \left(1^2 + 0.5^2\right) \times 4$$
$$= 2 + 5 = 7$$

D → Data
$f(D)$ captures intra-cluster distances (within-cluster distances) and not inter-cluster (btw-two clusters)

Now, we are focusing on that point (2,2) lies in which cluster.



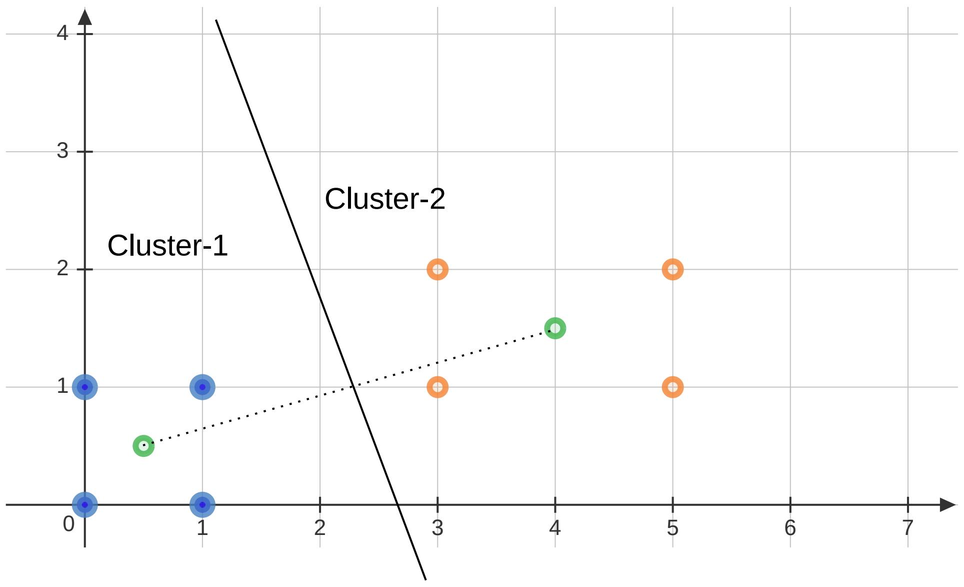
For this, we will calculate the distance btw the point (2,2) and the mean of the clusters.

$$Distance\ from\ \mu_1 \ = \ ||(2,2)\ -\ (0.5,0.5)\ ||^2 \ = \ 2\ \times 1.5^2 \ = \ 4.5$$

$$Distance\ from\ \mu_2 \ = \ ||(2,2)\ -\ (4,1.5)\ ||^2 \ = \ 2^2 + 0.5^2 \ = \ 4.25$$

So, point (2,2) will go to cluster- 2.

Now, we will look at the vornonoi regions.



The Voronoi regions are half-planes

1.4. Run k-means with $k = 2$ and $z^0 = [1\ 2\ 1\ 2\ 1\ 2\ 1\ 2]$. Plot the voronoi regions. To which cluster does $(2, 2)$ belong ? Find the value of the objective function at the end.

Now, we are going to run our K-Means once again, but with different initialisation.

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

Step-0: Initialisation

The initial cluster assignment is as follows:

$$z^0 = [1\ 2\ 1\ 2\ 1\ 2\ 1\ 2]$$

Now, we are going to find mean of each cluster.

$$\mu_1^0 = \frac{1}{4}\begin{bmatrix} 9 \\ 2 \end{bmatrix} = [2.25,\ 0.5]^T$$

$$\mu_2^0 = \frac{1}{4}\begin{bmatrix} 9 \\ 6 \end{bmatrix} = [2.25,\ 1.5]^T$$

$\mu_1 \longrightarrow$ mean of the first cluster

$\mu_2 \longrightarrow$ mean of the second cluster

Step-1: First iteration of k-means

Step-1.1: Compute the cluster assignments (Computing z)

$x_i \rightarrow$ The data point

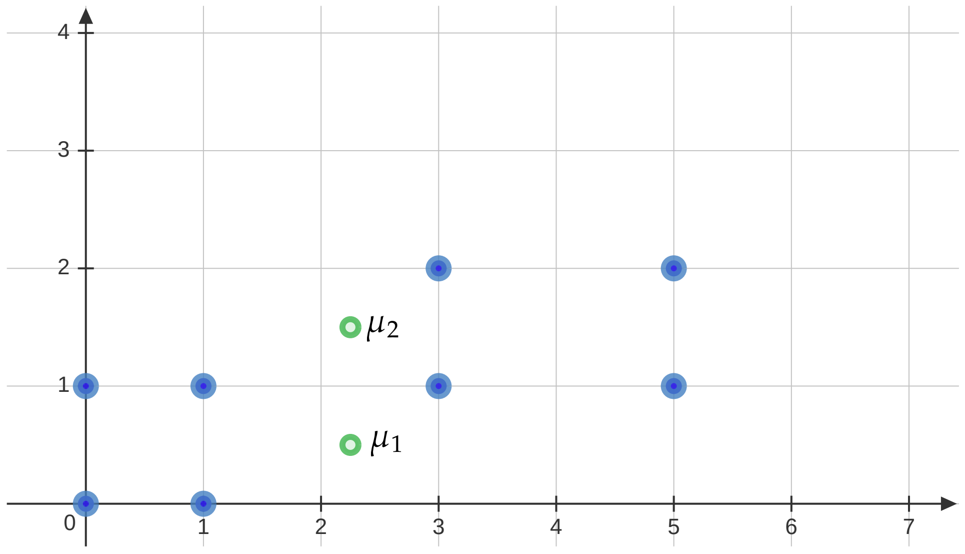
$z_i \rightarrow$ The cluster indicator variable of the the data point

$||x_i - \mu_1||^2 \rightarrow$ the distance of the data point from mean of cluster-1

$||x_i - \mu_2||^2 \rightarrow$ The distance of the data point from mean of cluster-2

Then we will compare both the distances and then Assign this data point accordingly to z_{t+1} .

mean $\mu_1 = [2.25,\ 0.5]^T$ *and* $\mu_2 = [2.25,\ 1.5]^T$



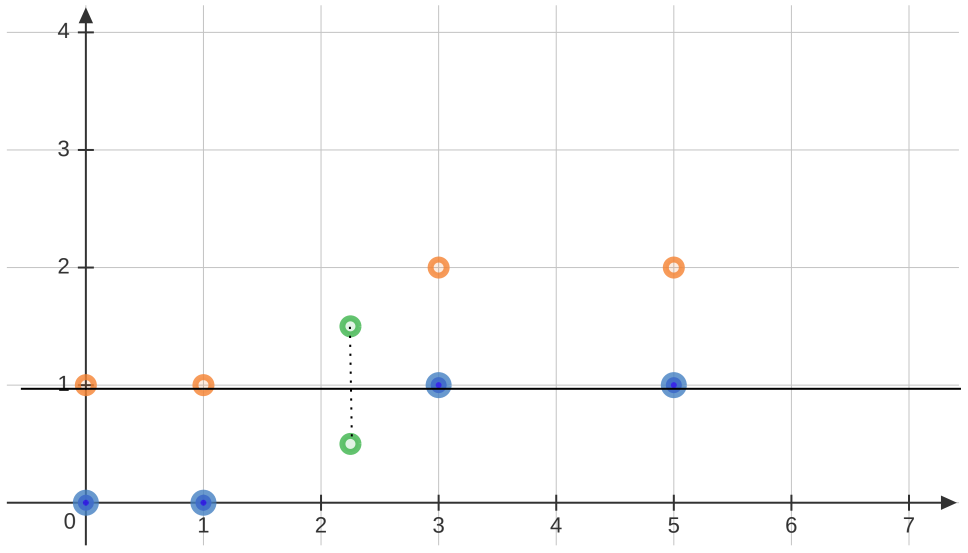
If there are ties $(d_1 = d_2)$, keep as it .

$$\begin{aligned} z_1 &= 1 \\ z_2 &= 2 \\ z_3 &= 1 \\ z_4 &= 2 \\ z_5 &= 1 \\ z_6 &= 2 \\ z_7 &= 1 \\ z_8 &= 2 \end{aligned}$$

Now, cluster assignments come out be same so our algorithm will stop.

$$z^0 = [1\ 2\ 1\ 2\ 1\ 2\ 1\ 2]$$

$$z^1 = [1\ 2\ 1\ 2\ 1\ 2\ 1\ 2]$$



MAIN TAKEAWAY: If we change the initialisation, the clusters are dependent on the initialization .

Now, we have to compute the objective function value:

$$f(D_{blue}) = (2.25)^2 + (0.5)^2 = 5.31$$

$$and\ in\ the\ final$$

$$f(D) > 7$$

So, it will be bad clustering compared to previous one.

Now, we will find to which cluster $(2, 2)$ will go to $\rightarrow$ Cluster-2 (from diagram)

Now, we will work with the final initialisation i.e. $z^0 = [1\ 1\ 1\ 1\ 1\ 1\ 2\ 2]$.

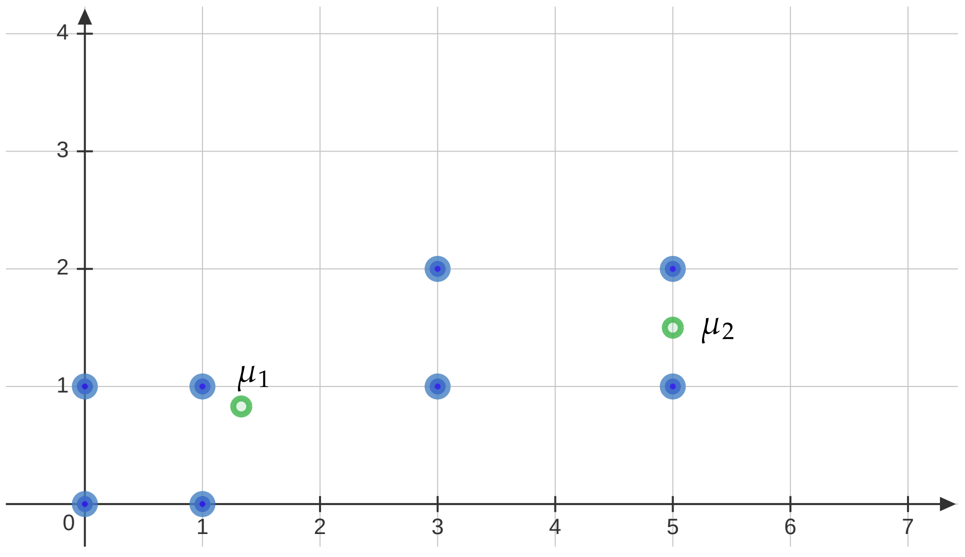
1.5. Run k-means with $k = 2$ and $z^0 = [1\ 1\ 1\ 1\ 1\ 1\ 2\ 2]$. Plot the voronoi regions. To which cluster does $(2, 2)$ belong ? Find the value of the objective function at the end.

$$X = \begin{bmatrix} 0 & 0 & 1 & 1 & 3 & 3 & 5 & 5 \\ 0 & 1 & 0 & 1 & 1 & 2 & 1 & 2 \end{bmatrix}$$

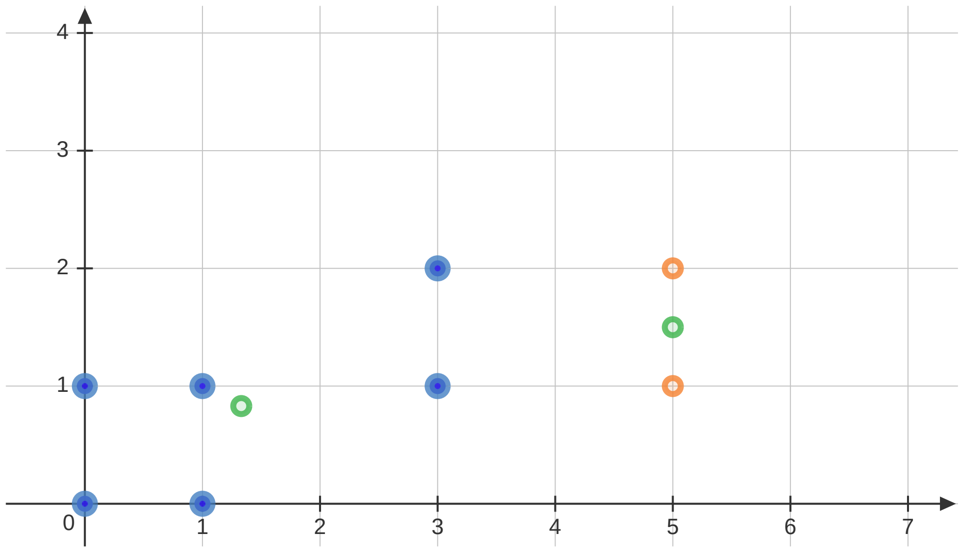
Now, we are going to find mean of each cluster.

$$\mu_1^0 = \frac{1}{6}\begin{bmatrix} 8 \\ 5 \end{bmatrix} = [\,1.33\ \ 0.83\,]^T$$

$$\mu_2^0 = \frac{1}{2}\begin{bmatrix} 10 \\ 3 \end{bmatrix} = [\,5,\ 1.5]^T$$



After doing all things as before:



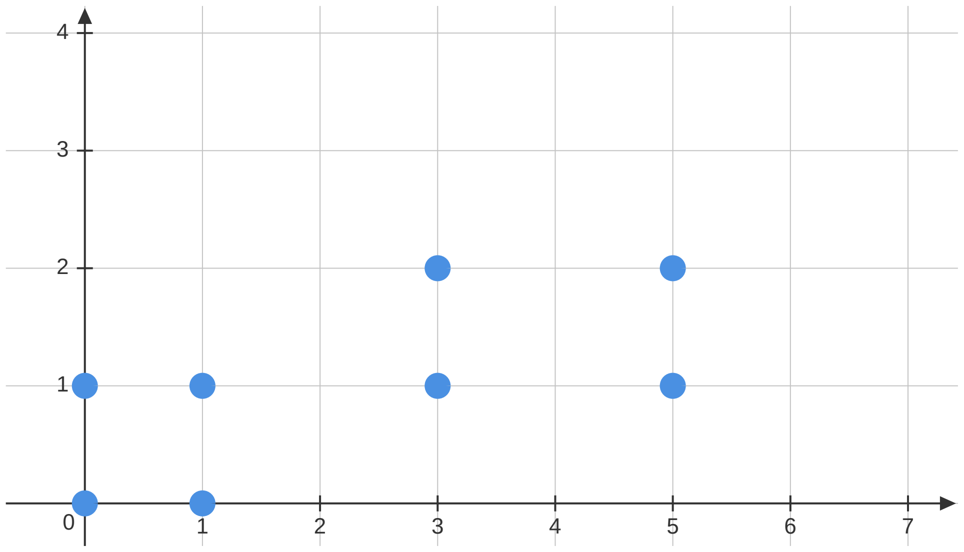
1.6. Run k-means with k= 2, but by initialising the first mean $\mu_1 = \begin{bmatrix} 100 \\ 100 \end{bmatrix}$ and $\mu_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. What do you observe ?

$\rightarrow$ Every data point will go to cluster 2 and no data point will go to cluster 1 .

Different ways of initialization

- Choose z values for all n-data points
- Choose k points in the dataset at random and make them the initial means.
- Choose some k points in $\mathbb{R}^d$ as the means.

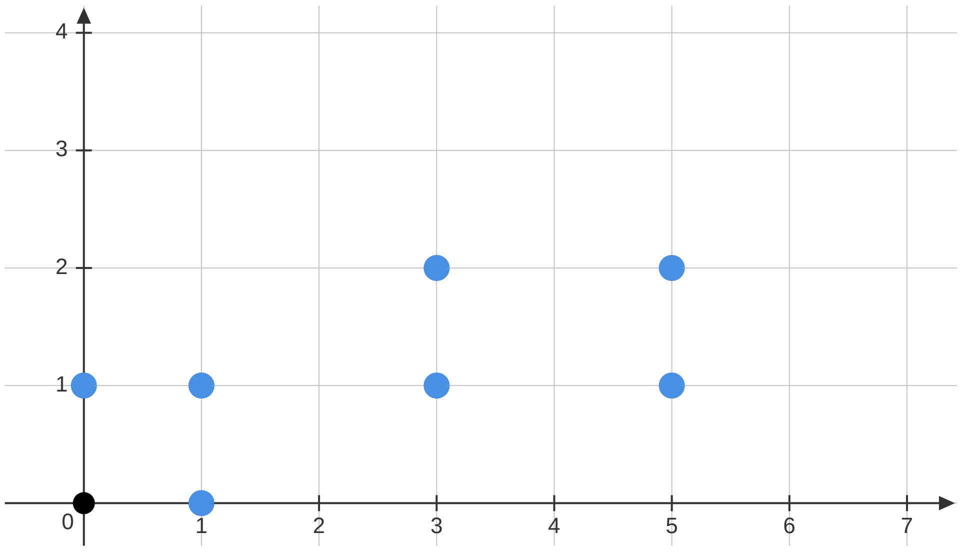
1.7. For k =3, run a simulation of the K-means++ algorithm. Compute the probabilities of choosing different points as the 3 means.



Step-1: Choose a point uniformly at random

Let's say

$$(0, 0)$$



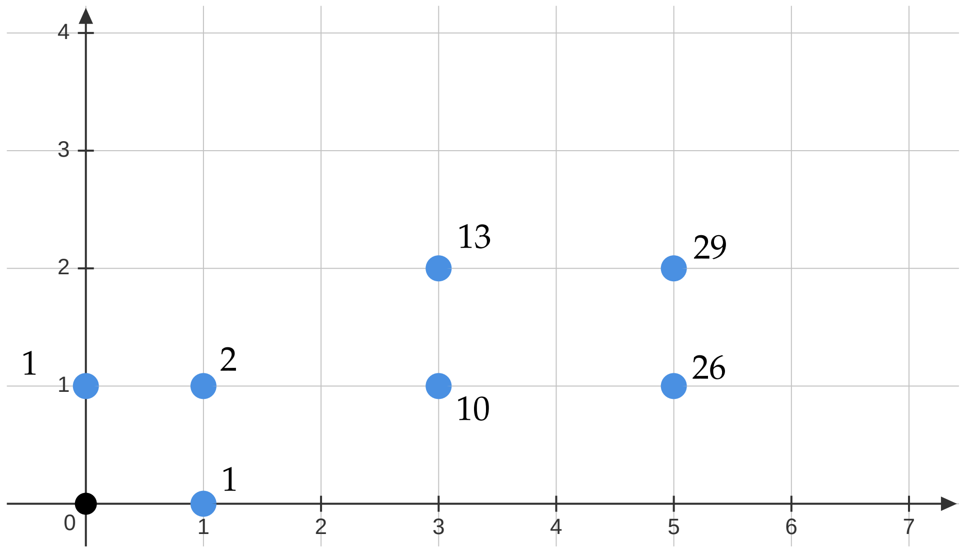
$$P(x_1) \;=\; \frac{1}{8}$$

$$\mu_1 \;=\; (0, 0)$$

Step-2: Choose the second mean

Step 2.1: Compute the scores for the remaining 7 data- points. from μ_1 .

Score $\rightarrow$ squared distance from the previous mean.



Step-2.2 → Form the probability distribution over the 7-data points using these scores.

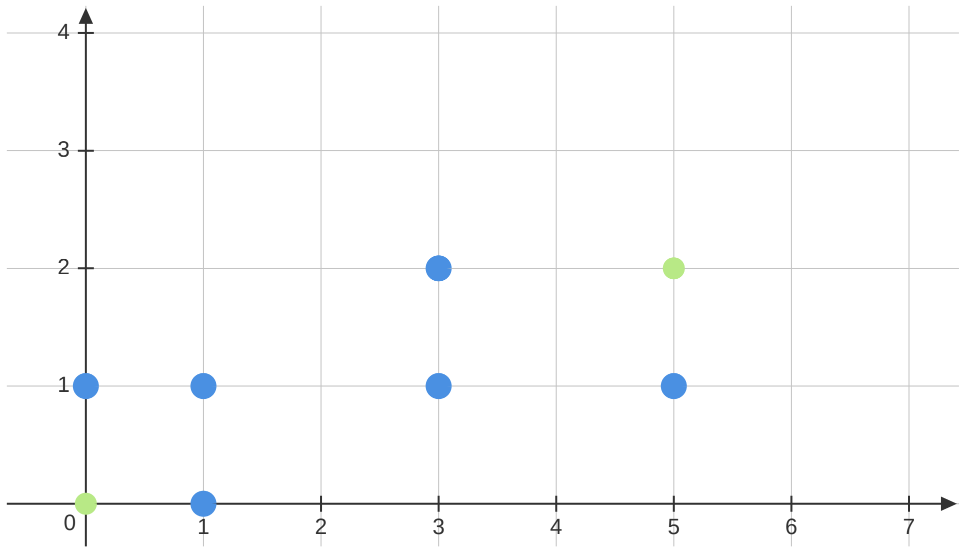
$$P(x_i) = \frac{||x_i - \mu||^2}{\sum_{i=1}^{n-1} ||x_i - \mu||^2}$$

x_i	$P(\mu_2 = x_i \mid \mu_1 = (0,0))$ → probability of x_i as being chosen for second mean
(0,1)	1/82
(1,0)	1/82
(1,1)	2/82
(3,1)	10/82
(3,2)	13/82
(5,1)	26/82
(5,2)	29/82

Basically, we are trying to find the next mean as far away as possible from the first mean.

Step-2.3 : Sample a point from this distribution

For this run , let us assume that $\mu_2 = x_8 = (5,2)$

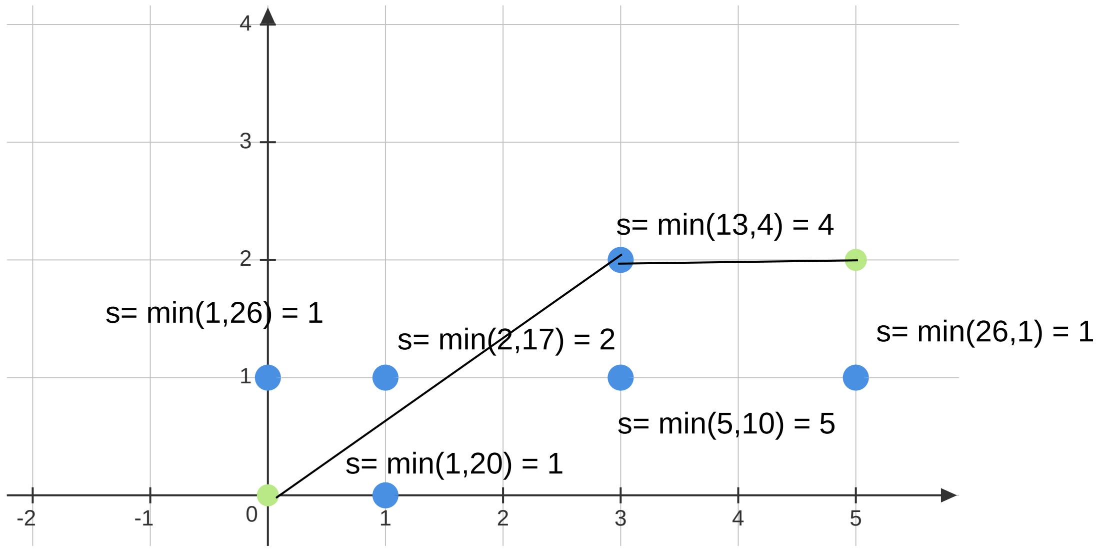


A → choose the first mean

B → Choose the second mean

Step-3: Choose the third mean

Step-3.1: Compute the scores of each of the six points to the two means



Form the probability distribution:

x_i	$P(\mu_2 = x_i \vert\, \mu_1 = (0,0))$ → probability of x_i as being chosen for second mean
(0,1)	1/14
(1,0)	1/14
(1,1)	2/14
(3,1)	5/14

Now, the third mean is $\mu_3 = (3, 1) = x_5$

Now, Probability of choosing different means:

$$P(A) = \frac{1}{8}$$

$$P(B|A) = \frac{29}{82}$$

$$P(C|A, B) = \frac{5}{14}$$

$$P(A, B, C) = \frac{1}{8} \times \frac{29}{82} \times \frac{5}{14}$$

Questions

Question to include from where: Assignments, PA, GA, Tutorials, Solve with us, ChatGpt.

---

1. How many configurations are possible for partitioning 100 data points into 10 clusters if these clusters are allowed to be empty?

$$K = 10, N = 100$$

$$Number\ of\ Configurations = K^N = 10^{100}$$

---

2. A cluster will be good if:
   - The points in the cluster are close to the mean of the cluster.
   - The points in the cluster are homogeneous.

---

3. If z1,z2,...,zn represent the clusters assigned to data points x1,x2,...,xn respectively, then which of the following would represent the mean (μk) of cluster k ?

$$\mu_k = \frac{\sum x_i\, 1(z_i = k)}{\sum 1(z_i = k)}$$

---

4. If z1,z2,...,zn represent the clusters assigned to data points x1,x2,...,xn respectively, and μk represents the mean of cluster k, then which of the following would be a good measure for ‘goodness’ of partitions?

$$Goodness\ of\ partition = objective\ function = F(z_1, \cdots, z_n)$$

$$F(z_1, \cdots, z_n) = min \sum_{i=1}^{n} ||x_i - \mu_{z_i}||^2$$

---

5. Does k-means always converge? → Yes
6. The solution produced by k-means algorithm is always optimal. → No

---

7. Two facts about K-means Algorithm:
   - K-means algorithm can not automatically determine the value of K. It has to be provided as an input parameter.
   - Finding optimal clusters is an NP-hard problem.

---

8. Consider a bunch of points $x_1, \cdots, x_n$. Which of the following would represent a point v whose sum of squared distances from the points $x_1, \cdots, x_l$ is the minimum?

$$v = \frac{1}{l}\sum_{i=1}^{l} x_i$$

---

9. It the value of objective function strictly reduces, it indicates that → The partitions can not repeat.

---

10. There are only a finite number of partition configurations.

11. If μ_1 and μ_2 are means of two clusters C_1 and C_2 in k-means, then for all the data points settling in C_1 → The mean μ_1 is closer than μ_2.

---

12. For every cluster, it's Voronoi region will be formed from intersection of k−1k−1 half spaces if kk is the total number of clusters. → True

---

13. Correct Statements:
    - K-means can not efficiently cluster data points that are not linearly separable.
    - Kernel K-means is used to cluster data points that are not linearly separable.

---

14. Which of the following is true with respect to optimality of clusters?
    - Getting optimal clusters is an NP-hard problem, so no technique exists that can guarantee optimal clusters

---

15. To obtain good clusters, how should the initial means of clusters be?
    - Means should be as far apart as possible.

---

16. K-means++ initializes the clusters' means
    - In an iterative fashio
    - Probabilistically, by assigning a different probability to each data point.

---

17. During different iterations of the initialization step, k-means++ assigns highest probabiliity to those points to be chosen as clusters' means for which
    - The closest cluster mean (of already chosen clusters) is the farthest.

---

18. Due to a smart initialization of cluster means, K-means++ is able to produce optimal clusters. → No.

---

19. K-means algorithm automatically determines the value of K as per the given data. → No

---

20. Which of the following is correct with respect to the value of k?
    - The value of k should neither be very small nor very large.

21. If obj represents the value of objective function and P represnts penalty, then the best value of k will be that for which → (obj+P) is minimum.

22. Which of the following techniques help(s) in fixing a suitable value of penalty?
- Akaike Information Criterion
- Bayesian Information Criterion

23. Which of the following sequences is correct for K-Means algorithm?
- Specify the number of clusters
- Assign clusters centres randomly
- Assign each data point to the nearest cluster centres.
- Re-assign cluster centres
- Re-assign each data point to nearest cluster centres.

24. If $F\left(x_1^t, \ \cdots, \ x_n^t\right)$ represents the value of objective function in iteration tt of Lloyd's algorithm, then which of the following is true?

$$F\left(x_1^{t+1}, \ \cdots, \ x_n^{t+1}\right) < F\left(x_1^t, \ \cdots, \ x_n^t\right)$$

25. If μ1 and μ2 are means of two clusters in k-means, then the boundary between the two clusters will be

→ Perpendicular to the line joining μ1 and μ2 and at the point $\frac{\mu_1 + \mu_2}{2}$

26. With respect to Lloyd's algorithm, choose the correct statements:
- The partition configurations can not repeat themselves.
- Objective function after making the re-assignments strictly reduces.
- Change of value of objective function indicates that the partition configuration has changed.

27. For 1000 data points, out of k = 1, 10 and 100, which value of k is likely to result in the maximum value of the objective function? → 1

28. For 100 data points, if k = 100, what will be the value of the objective function? → 0

29. Important note:
- One initialization might get stuck in local minima, while another may lead to global minima.
- The initialization of cluster centres may affect the number of iterations K-means takes to converge.

30.Outliers are data points that deviate significantly from the rest of data points. Knowing the way Lloyd's algorithm works, do you think it is sensitive to outliers? → Yes

---

31. Which of the following statements are True?

- K-means is sensitive to cluster center initializations.
- Bad initialization can lead to poor convergence speed.
- Bad initialization can lead to bad overall clustering.

---

32. If the data set has two features x1 and x2, which of the following are true for K means clustering with k = 3?

    1. If x1 and x2 have a correlation of 1, the cluster centres will be in a straight line.
    2. If x1 and x2 have a correlation of 0, the cluster centres will be in straight line.

$\rightarrow$ statement 1 is correct as when x_1 and x_2 have perfect correlation then all data points lie in a straight line so their centroids will also lie in a straight line.

---

33. Correct Statements:

- At the end of k-means, the objective function settles in a local minima and reaching global minima may not be guaranteed.

General Questions

---

1. **What is the primary objective of the K-means clustering algorithm?**
    - The primary objective of the K-means clustering algorithm is to partition a dataset into kk distinct clusters, where each data point belongs to the cluster with the nearest centroid.
    - This is achieved by minimizing the within-cluster variance, which is the sum of the squared distances between each data point and its corresponding cluster centroid.
    - The algorithm iteratively adjusts the centroids and reassigns data points to clusters until convergence, ensuring that the total distance within clusters is as small as possible.
    - The goal is to create clusters that are compact (minimizing variance) and well-separated from each other.

---

2. **Explain the steps involved in the K-means algorithm. How does it iterate to find the final clusters?**
    - Determine the Number of Clusters (k):
        - Decide how many clusters you want to partition your data into. This is often based on prior knowledge or using methods like the elbow method or silhouette analysis.
    - Initialize Centroids:
        - **Random Initialization**: Select k random points from the dataset as the initial centroids.
        - **K-means++ Initialization**: Select the first centroid randomly, then choose subsequent centroids based on a probability proportional to the squared distance from each point to the nearest existing centroid. This helps in better initialization and often leads to improved clustering results.
    - Assign Data Points to Clusters:
        - For each data point, calculate the distance to each centroid (commonly using Euclidean distance).
        - Assign each data point to the cluster whose centroid is nearest (i.e., has the smallest distance).
    - Update Centroids:
        - After all points have been assigned to clusters, recalculate the centroids of each cluster. The new centroid is typically the mean of all data points assigned to that cluster.
    - Check for Convergence:
        - Compare the new centroids with the previous centroids. If the centroids do not change significantly or if the assignments of data points to clusters remain unchanged, the algorithm has converged.
    - Iterate:
        - Repeat the assignment and update steps until convergence is reached. This iterative process helps in minimizing the within-cluster variance.

Convergence Criteria

The algorithm typically converges when:

- The centroids do not change significantly between iterations.
- The assignments of data points to clusters remain constant.
- A predefined number of iterations is reached.

---

3. **What are the key assumptions made by the K-means algorithm regarding the data?**
   - Linear relationship
     - K-means assumes that the clusters are spherical and can be separated by linear boundaries.
     - This means it works best when the data points within each cluster are approximately equidistant from the centroid.
   - Equal Cluster Size
     - It generally assumes that ALL clusters have same number of data points.
     - K-means can struggle with clusters of varying sizes or densities.
   - Feature Independence
     - K-means assumes that features are independent of each other. Correlation between features can affect the clustering outcome.
   - Homogeneous Clusters
     - K-means assumes that clusters are compact and well-separated. It may not perform well if the clusters are intertwined or have complex shapes.
   - Noisy Data
     - K-means is sensitive to outliers and noise in the data. Outliers can disproportionately affect the position of centroids, leading to incorrect clustering.

---

4. How does the choice of the initial centroids affect the results of the K-means algorithm? What strategies can be used to improve centroid initialization?

Impact of Initial Centroids on K-means Results

- Convergence Speed
  - If the initial centroids are poorly chosen, the algorithm may take longer to converge.
- Local Minima
  - K-means can converge to local minima based on the initial centroid positions. If the centroids start far from the optimal locations, the final clusters may not represent the best partitioning of the data.
- Quality of clusters
  - The choice of initial centroids directly influences the quality of the resulting clusters. Bad initialization can lead to clusters that are not well-separated or do not capture the underlying structure of the data.

Strategies to Improve Centroid Initialization

- K-means++ Initialization
  - This method enhances the initialization process by selecting the first centroid randomly
    and then choosing subsequent centroids based on a probability distribution that favors points farther away from existing centroids.
  - This approach tends to result in better initial placements and reduces the chances of getting stuck in local minima.
- Random Initialization with Multiple Runs
  - Run the K-means algorithm multiple times with different random initializations and select the best clustering outcome based on the lowest objective function value (sum of squared distances). This approach increases the likelihood of finding a better clustering solution.
- Using Domain Knowledge
  - If prior knowledge about the data is available, you can use that information to select initial centroids. For example, you might choose centroids based on known cluster

    characteristics or by selecting points that are representative of different regions of the data space.

- Hierarchical Clustering for Initialization
  - Performing a hierarchical clustering algorithm first can help identify potential centroids. You can then use the centroids from these clusters as initial points for K-means.
- Elbow Method or Silhouette Analysis
  - These methods can help determine an appropriate number of clusters (kk), which can guide the initial centroid selection process. By selecting centroids that correspond to significant data points in these analyses, you can improve initial placement

---

5. What are the strengths and weaknesses of K-means clustering compared to other clustering algorithms like hierarchical clustering or DBSCAN?

Strengths of K-means

- Simplicity and Speed
  - K-means is straightforward to implement and computationally efficient, especially for large datasets.
  - Its time complexity is typically O(n·k·i) where n is the number of data points, k is the number of clusters, and i is the number of iterations.
- Scalability
  - It works well with large datasets and can handle a significant number of clusters efficiently.
- Interpretability
  - The output is easy to understand: you get k clusters with their centroids, which can be useful for summarizing data.

Weaknesses of K-means

- Sensitivity to Initialization
  - The final results can be heavily influenced by the initial placement of centroids, leading to poor convergence or local minima.
- Assumption of Spherical Clusters
  - K-means assumes that clusters are spherical and of similar size, which can be a limitation when the actual clusters have irregular shapes or varying densities.
- Fixed Number of Clusters
  - You must specify the number of clusters kk beforehand, which may not be straightforward without prior knowledge of the data.
- Sensitivity to Outliers
  - Outliers can significantly affect the position of centroids, leading to misleading clustering results