Introduction to Clustering

KMeans Algorithm

[Unsupervised Learning]

CASE STUDY

You are a data scientist at Amazon, and you need to decide different marketing strategies for different types of customers.

How will you identify the different customer groups and assign a group to every customer ?

DATA:

	ID	n_clicks	n_visits	amount_spent	amount_discount	days_since_registration	profile_information
0	1476	130	65	213.905831	31.600751	233	235
1	1535	543	46	639.223004	5.689175	228	170
2	1807	520	102	1157.402763	844.321606	247	409
3	1727	702	83	1195.903634	850.041757	148	200
4	1324	221	84	180.754616	64.283300	243	259

→ Quiz

Q: What is the difference b/w this and a classification problem?

→ Here we have no target.

In binary classification, we need:

$$D = \left\{ (\vec{x}_i, y_i)_{i=1}^{\hat{}} \; ; \; \vec{x}_i \in \mathbb{R}^d \; ; \; y_i \in \{0, 1\} \right\}$$

multi-classification → $y_i \in S$

finite set of classes

regression → $y_i \in \mathbb{R}$

Unsupervised Learning

However, we do not have a 'target' in our problem.

$$D = \{ \vec{x}_i ; \; x_i \in R^d \}$$

Such probems are called unsupervised learning

Q: Is this a valid use case then?

→ Offcourse:

Example:

→ customer / product segmentation

→ grouping similar images in galexy

→ detecting similar stocks automatically

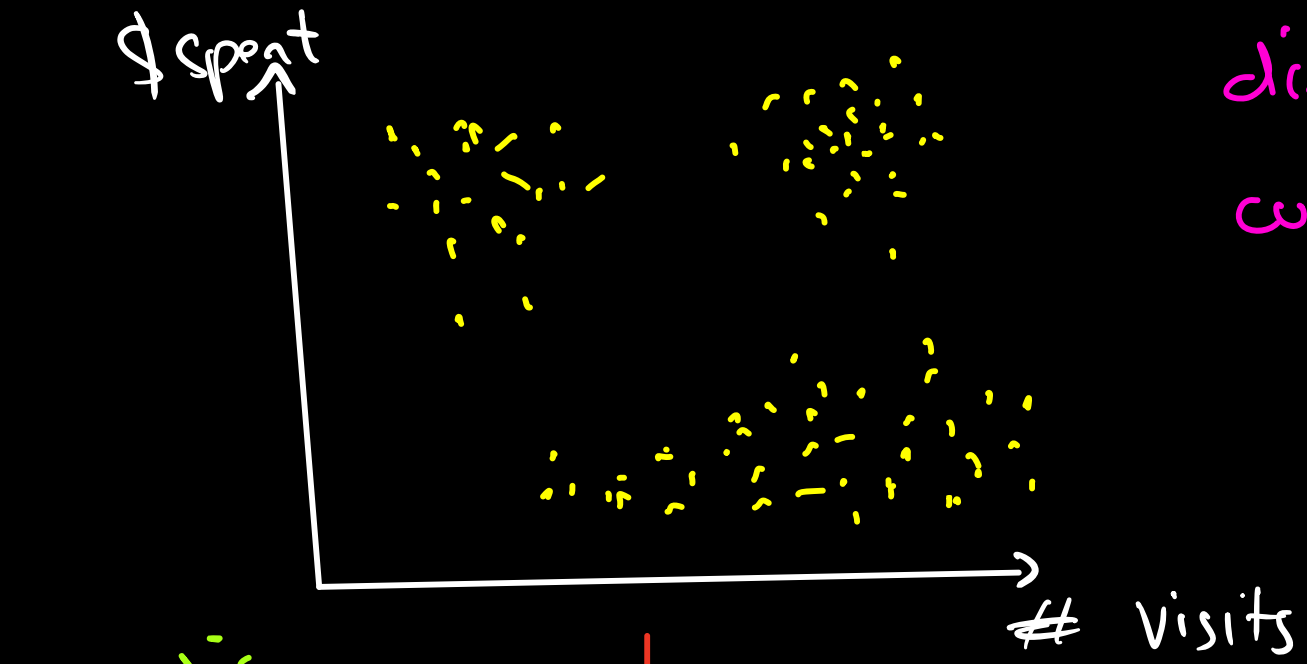
The process of grouping any kind of data based on similarity in their features, automatically, without human expertise, is called **Clustering**

This is one type of unsupervised - learning

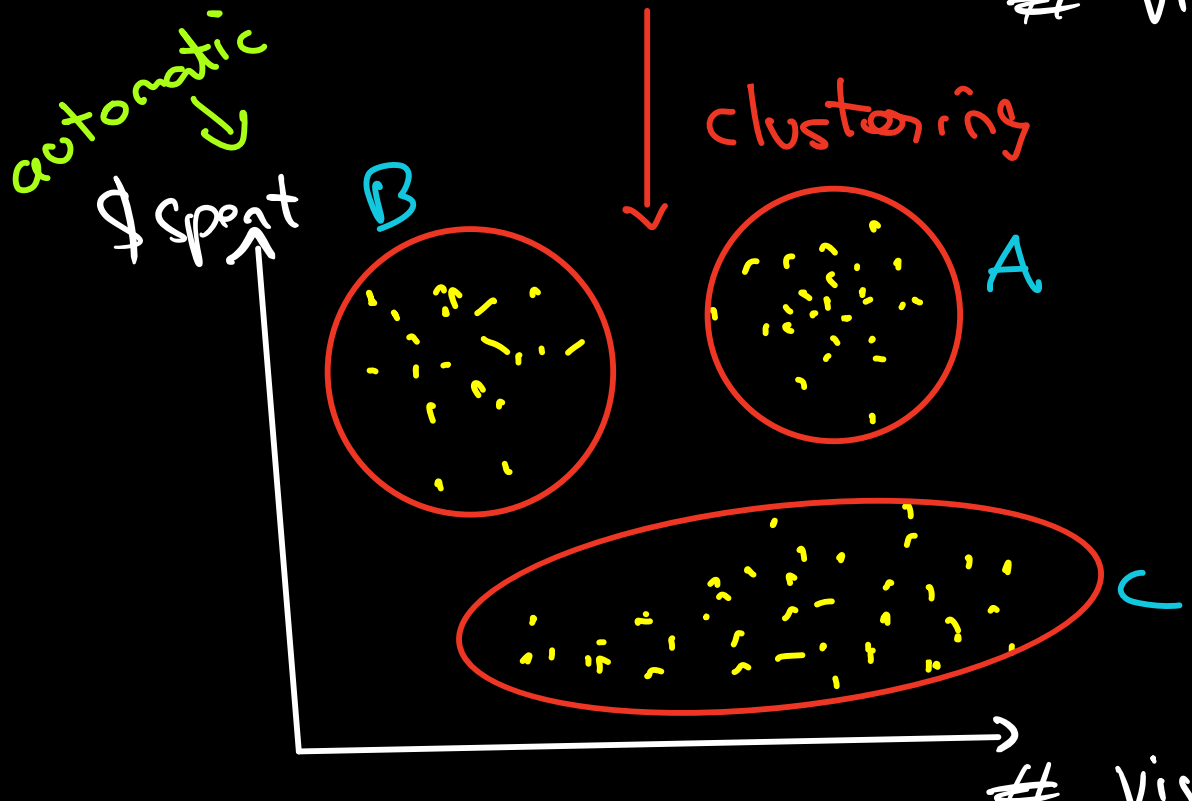
Other examples of unsupervised learning:

- → Association rules / Recommenders
- → Dimensionality reduction (Eg: PCA)
- → Anomaly detection
- → Encoding (word-to-vec)

Clustering



Q: Can you spot different types of customers?



Intuitions

A → Heavy shoppers
Visit a lot / Spend a lot

B → Rich people
spend whenever they visit

C → Window shoppers

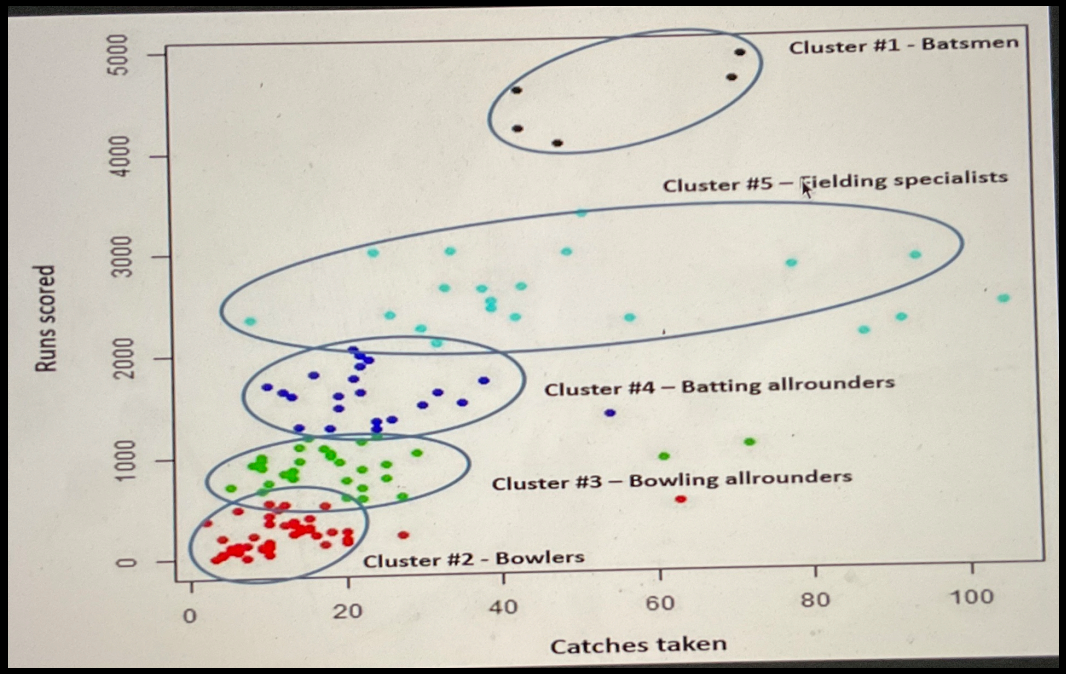
Q: What would you recommend for each group??

→ No ads needed for 'A'

→ Only ads needed for 'B'

→ Ads + discount for 'C'

Another example



Intuitively, clustering is dividing a population into groups such that the pts in one group are similar to each other.

Each group is called a **cluster**

So the task is

→ group similar points

→ define "similarity"

↓

Eg: distance on a scatter plot

Q: How do you decide if a cluster is good or bad??

→ There is no ground truth data!
Hence nothing to compare with

→ It should simply make 'business sense'

Advanced applications of clustering

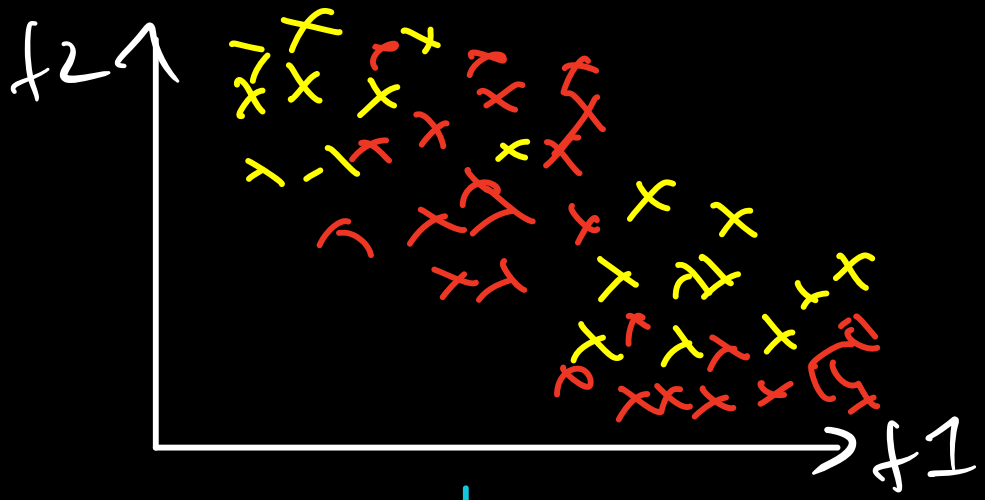
→ Validation of domain knowledge / human opinion

→ Search algorithms and recommender sys.

→ Feature creation

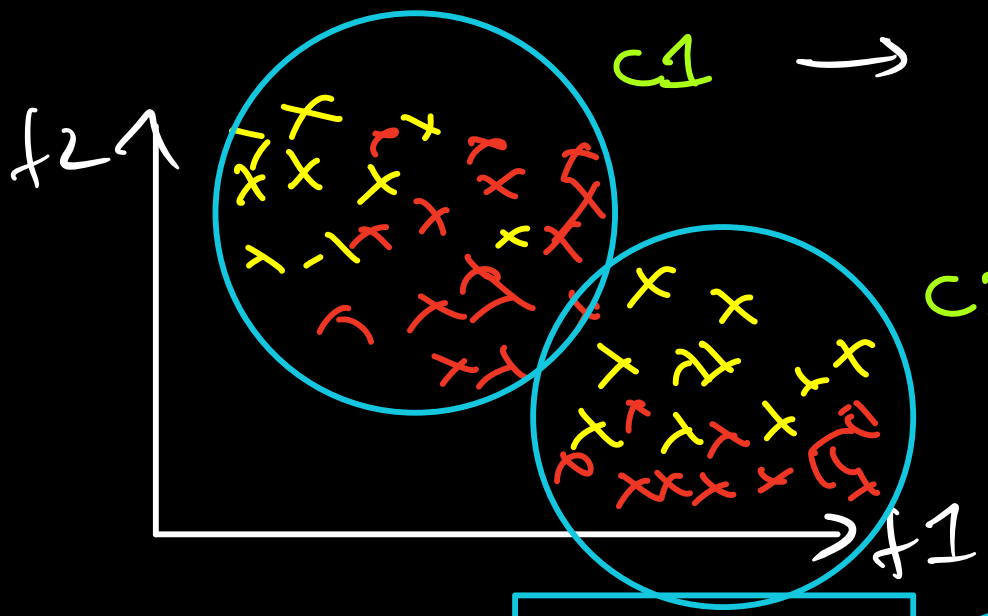
→ Clustering with supervised learning to improve model accuracy!!

Example:



Build a classifier for this problem.

Q: Any guesses??



C1 → Logistic Regression 1

C2 → Logistic Regression 2

[cascading]

input → Cluster

LR1 → Yellow / Red

LR2 → Yellow / Red

How to perform clustering

Q: Any ideas ??

So far we have seen a pattern

Geometrical idea

↳ Convert to math

↓

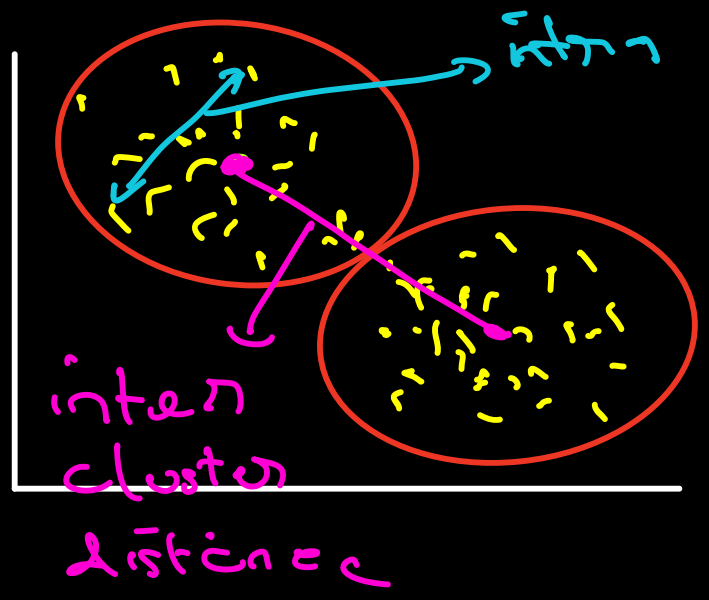
Design an optimisation function

↓

optimize

Q: Can we construct a loss / gain function for clustering ??
[Hint: ANOVA / f statistic]

→ We assuming that data comes from different populations here.



Ideally, we need to assign clusters in a way that:

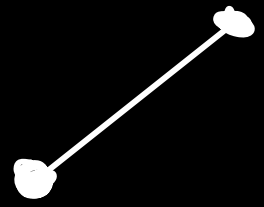
inter-cluster ↑ (max)

intra-cluster ↓ (min)

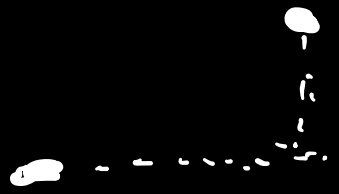
Q: What is distance mathematically ??

→ Euclidian distance

→ Low dimensions



→ Manhattan distance
→ Medium dimensions



→ Cosine distance
→ high dimensions

→ etc..

So the choice of the distance metric is a hyperparameter for many clustering algorithms.

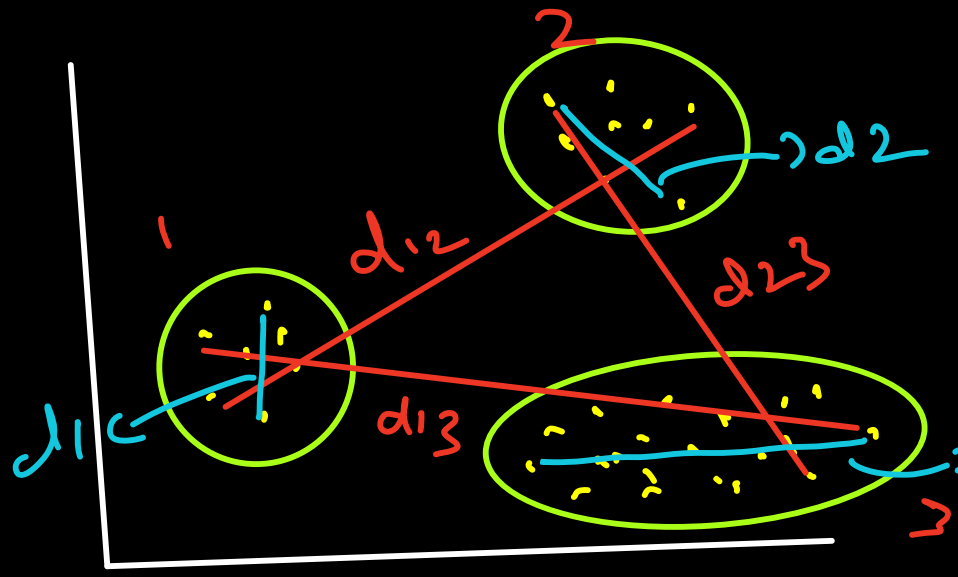
Now,
Using this we can write some sort of metric which we may optimise

Dunn Index

↳ slightly simpler metric than f-statistic.

$$D = \frac{\text{min inter cluster dist}}{\text{max intra cluster dist}}$$

→ worst case scenarios are taken



$\min(d_{12}, d_{13}, d_{23})$

$= d_{23}$

$\max(d_1, d_2, d_3)$

$= d_3$

∴ Dunn Index = d_{23} / d_3

If dunn index is high, clusters are well seperated and spread out, else they are not.

→ Note: dunn index is unbound, so it can only be interpreted in a relative sense.

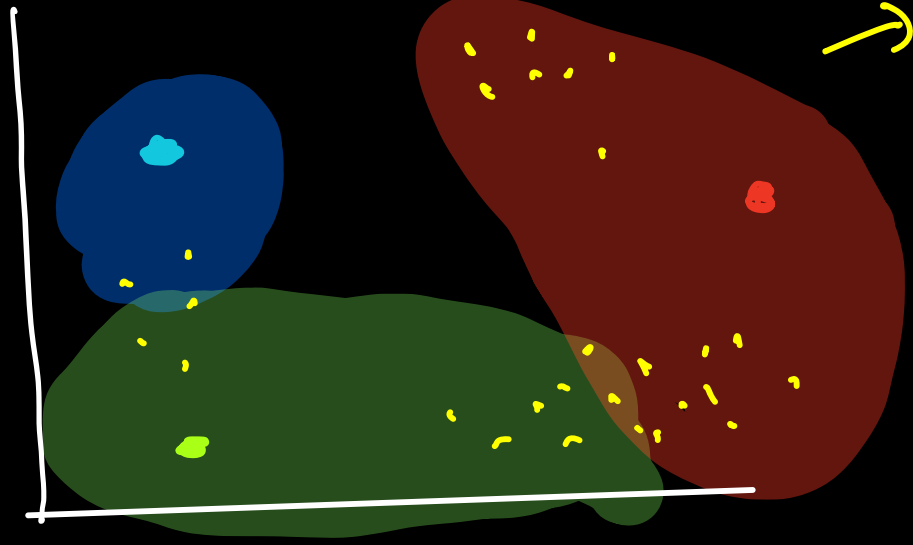
→ Quiz : High or low?

Optimisation

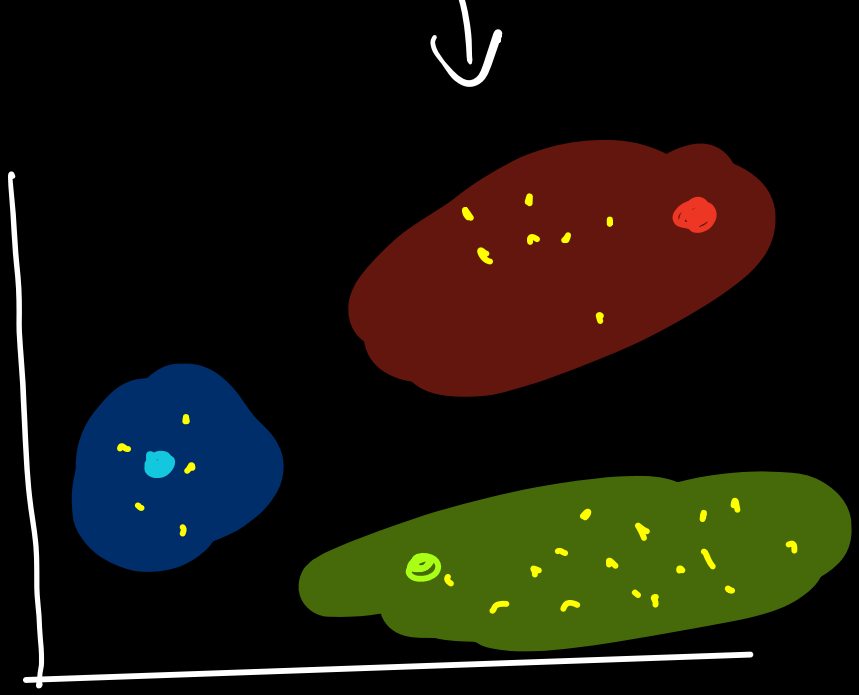
Now that we have a way to compare different cluster possibilities, we need to find a way to find the best clusters

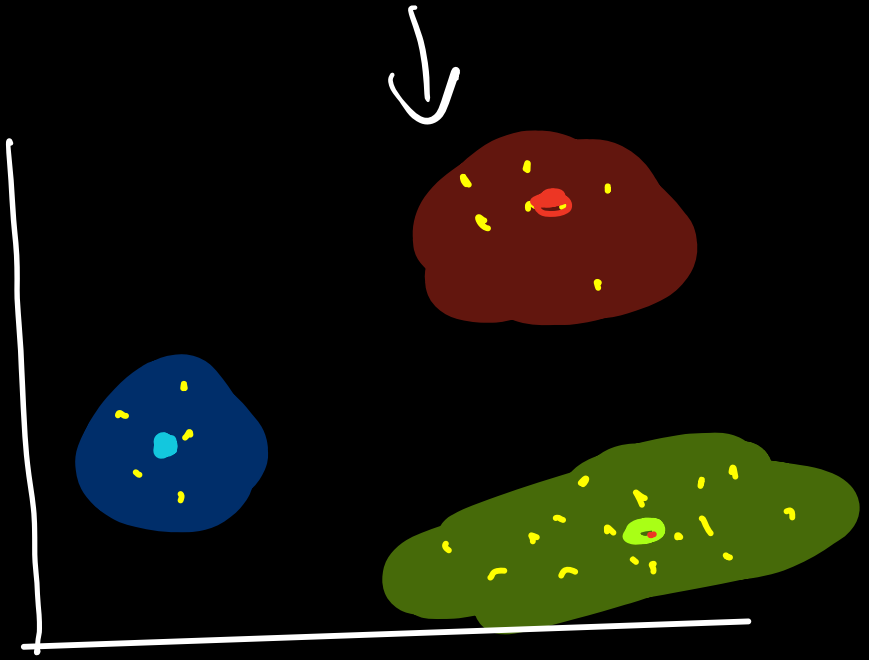
Steps: → Decide # clusters (manual)

→ Initialise cluster centers randomly
→ To each point assign the cluster whose center is closest
→ Calculate dun index
→ update the centroid location
→ Repeat until best centroids are found.



→ Dun ?!





↳ stop !!

Optimisation

$$\max_{[\ ?\ ]} \quad f(?) \longrightarrow ??$$

What is the parameter & the func^{n} ?!

→ Params → centroids

→ f = ?? dunn index (clusters)

cluster centroids = $C = \begin{bmatrix} \vec{c_1} \\ \vec{c_2} \\ \vdots \\ \vec{c_k} \end{bmatrix}$

$g(C, X)$ = assignment function based on nearest centroid

$\downarrow$

argmin (distance (C, X)) | $d(x_i, c_i)$ = dunn index

$$L: \max_{c} d(X, \underbrace{\text{argmin}}_{\text{not differentiable}}(\underbrace{\text{distance}(c, x)}_{\text{some distance metrix are not differentiable}}))$$

Hence we cannot solve this using calculus or gradient descent.

Clustering by nature is a discrete problem and hence cant be optimised easilly.

[There are ways, but they are all hard]

Lloyd's Algorithm (K-Means Clustering)

Steps:

→ Randomly initialize K centers

→ assign points to this center to get your cluster.

→ find the centroids of these clusters

→ Re-assign points

→ Repeat until new centers = prev centers

→ animation / code

Determining the best 'K'

→ Visually

→ Domain knowledge

└ Slack groups

→ Avg reply time, # users, # msgs

→ support channels (sales / ops)

→ discussion forums (students)

→ internal teams (employees)

→ Elbow method

Q: which is best K??

a) 3

b) 2

c) 14

→ We can visually see that its a) 3

Is How do you tell that numerically?

=> Metrics!!

1) WCSS

2) Silhoutte Score

WCSS (inertia)

Within - cluster sum of square

$$WCSS = \sum_{i=1}^{K} \sum_{j=1}^{m_i} distance(x_j, c_i)$$

distance of pts from their respective cluster centroids (should be low)

[unbounded]

Quiz: When will inertia go to zero?

Silhouette Score

$$S(x_i) = \frac{b - a}{\max(b, a)}$$

a = avg distance of x_i with points in its own cluster

b = avg distance of all point of nearest cluster with x_i

How close is the point to the points neighbouring cluster compared to own cluster.

$$\max \sum_i S(x_i)$$

[Bounded]

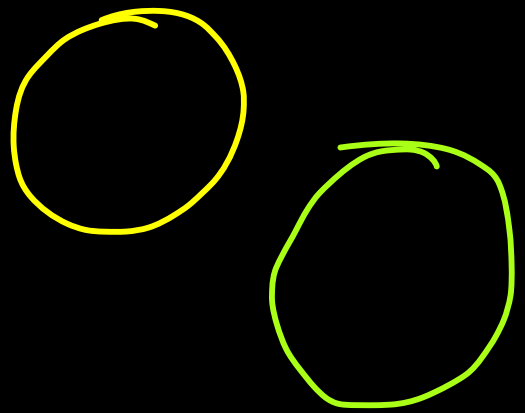
$S_i(x_i) = 1$, $b > a$ & $a = 0$

$s_i(x_i) = -1$, $a > b$ & $b = 0$

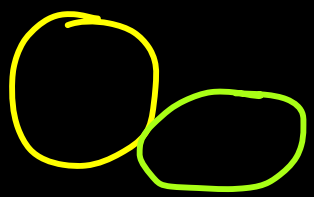
(incorrectly alotted clusters)

x_i → x

$0 > s_i > 1$



$s_i \sim 0$



$-1 < s_i < 0$