UNSUPERVISED LEARNING

35.1 → What is clustering?

classification & Regression.

$D = \{x_i, y_i\}$ ← Predict y_i given x_i $\qquad y = f(x)$

$y_i \in \{0, 1\}$ → 2-class classification

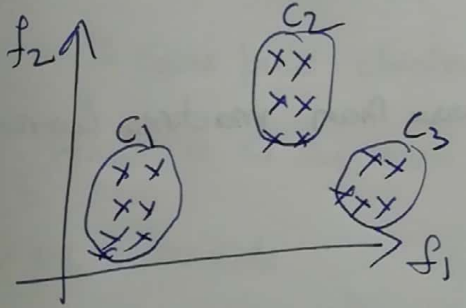
$y_i \in \mathbb{R}$ → regression.

In clustering

$D = \{x_i\}$ → no y_i's

Task: group/cluster similar data points

↳ points are close together & farther away from other points



(a) Point in a cluster are close together

(b) Points in diff clusters are far away

Similar: is problem specific

$D = \{x_i\}$ no y_i's

Q) How do you measure how well your algo. performed?

↳ well will know later

clustering: K-means, Hierarchical clustering, DBScan.

35.2 Unsupervised Learning

→ Clustering is always refered to as unsupervised learning

→ In case of classification & regression

$\{x_i, y_i\}$ we will a function $f(x) = y$

Here y_i is supervising/helping to find function $f(x)$

→ Semi-supervised learning : $D = D_1 \cup D_2$

$D_1 \rightarrow \{x_i, y_i\}$ $\quad$ $D_2 \rightarrow \{x_i\}$

Length of $|D_1| << |D_2|$

35.3 Applications

→ Deeply studied in Data mining course than machine learning

→ Tons of application

https://en.wikipedia.org/wiki/Cluster_analysis#Applications

Ex: ① eCommerce: Amazon.com, Alibaba, Ebay, flipkart

Task: group "similar" customers based on their purchasing behaviour

C1, (C2), C3, (C4), C5

(C4) → we can offer a discount (10%, 20%)

(C2) → we can offer a deal

purchasing behaviour → {money, credit card, products, geographical of customers

{estimate income per year

Eg ② Image segmentation (computer vision & image processing)

↳ grouping/clustering similar pixels into regions/segments

↳ Typical apply ML algo to perform obj detection

Eg: 3 Amazon Food Reviews

→ class label

review & text	Polarity (+ve, -ve)
r1	+ve
r2	-ve
r3	+ve
⋮	⋮

(Manually reviewed & labelled)
↳ Time Consuming
↳ Costly / Expensive

Imagine we have 1M reviews (no y_i's)

① cluster them into 10K groups using word-similarity
(BoW, tfidf, W2V)
↓
$C_1, C_2, C_3 \dots C_{10K}$

② review & label each cluster.

I will pick cluster C_i → pick one pt → +ve / -ve. All the points in C_i as +ve
↓
In real word (365 points)

③ ML- models

35.4 Metrics for clustering

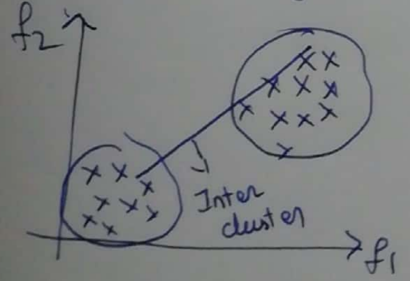
$D = \{x_i\}$; no y_i's

What are the performance measures ?

[Classfn & regression
↓
AUC
log-loss
Pre, Recall

y_i's ⇐ class label
⇐ regression label
{ ground truth }]

geom: what is a clustering result :



$D = \{x_i\}$ → 2 clusters
$x_i \in \mathbb{R}^2$

→ Intra cluster : within a cluster
→ Inter cluster : b/w clusters

In a ideal world

↳ Inter-cluster dist → V-high
↳ Intra-cluster dist → V. low

Dunn Index:

→ dist c_i, c_j

$$D = \frac{\uparrow \max_{i>j} d(i,j)}{\downarrow \max_{k} d'(k)}$$

↳ maximal inter cluster distance

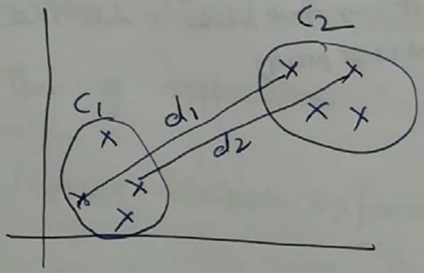
Intra cluster distance

K - cluster.

$\{c_1, c_2 \dots c_i, c_j, c_k\}$

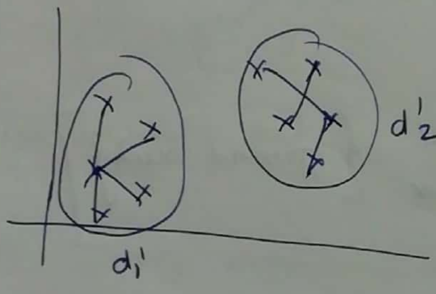
D is high ⇒ good clustering

d(i, j) = dist b/w C_i & C_j farthest point.



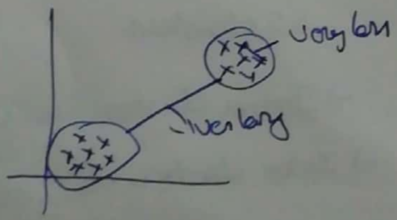
for Every pair: $\underbrace{d_1, d_2, d_3 \text{ ----- } d_k}$

Pick the max distance point



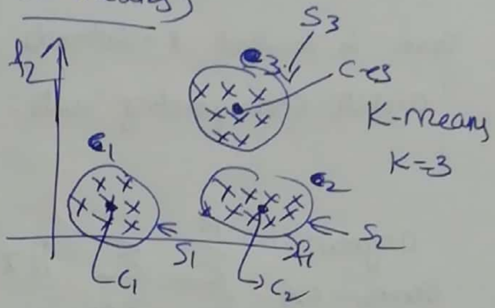
max (d_1', d_2')

Ideal clust.



→ many other measures / metrics for measuring clustering

35.5 Centroids, Geometric Intuition (k:means)



- → popular, simple
- → Basic K-means.
- → Variants of k means.

$\left\{ \begin{array}{l} k : \# \text{ of clusters.} \\ \uparrow \text{ hyper parameter} \rightarrow \boxed{\text{CV}} \text{ ideas lik} \end{array} \right.$

c_1, c_2, c_3 : centroids

$S_1, S_2, S_3 \Leftarrow$ sets

$S_1 \cap S_2 = \phi$ (null set)

$S_1 \cap S_3 = \phi$

$S_2 \cap S_3 = \phi$

→ when we say we want k-clusters we will get k-centroids & k-sets

k-clusture ÷ 1 2 3 - - - k

k-centroids = c_1, c_2, c_3 - - - c_k

k-sets : S_1, S_2 S_3 - - - S_k

$$c_i = \frac{1}{n} \sum_{x_j \in S_i} x_j \quad \Leftarrow \text{mean point of } S_i$$

K-means : centroid based clustering scheme

Big challange : How to find k-centroids, once we get centroids we can Easily find the sets by using nearest centroid idea.

35.6 Mathematical Formulation : objective function (K means)

$\mathcal{D}: \{x_1, x_2 \cdots x_n\}$

Task is to find k-Centroids : $c_1, c_2 \cdots c_k$ $\forall x_i \in S_j$

and its corresponding sets : $S_1, S_2 \cdots S_k$ $\forall_{i,j}\ S_i \cap S_j = \phi$

Intra cluster distance minimized.

$$\underset{(c_1, c_2 \cdots c_k)}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - c_i\|^2 \quad \text{s.t} \quad x \in S_i \quad \forall_{ij}\ S_i \cap S_j = \phi$$

Sq. dist of x from c_i

$$\left\{\begin{matrix} c_1, & c_2 & \cdots & c_k \\ \downarrow & \downarrow & & \downarrow \\ S_1 & S_2 & \cdots & S_k \end{matrix}\right. \rightarrow \text{using proximity idea}$$

→ In the above function there is no Intra cluster distance

→ V.V. hard to solve the above Equation.

↳ NP-hard

↳ Exponential time Complexity

In Computer Science : If we have a hard problem

↓

approximation algo → using some hacks

→ Lloyd's algo

35.7 K-means Algorithm

Lloyd's algo

Step 1 : Initialization : (random initialization)

→ randomly pick k pts from $\mathcal{D}$ and call them. $c_1, c_2, \cdots c_k$

Step 2 : Assignment

For each pt x_i in $\mathcal{D}$

→ Select the nearest c_j

→ Compute the dist (x_i, c_j) $\forall$ $j = 1, 2, \cdots k$

→ add x_i to set S_j corresponding to centroid c_j

→ we will have a point x_i assigned to a set S_j $\quad j = 1, 2, \dots k$

$i = 1$ to n.

Step3 ÷ Recompute centroids

→ recalculate / update c_j's as follows

$$c_j = \underbrace{\frac{1}{|S_j|} \sum_{x_i \in S_j} x_i}_{\text{mean-pt}}$$

→ # of points in set S_j

Step4 ÷ Repeat step 2 & step 3 untill convergence

$$\underset{\text{assignment}}{\text{step 2}} \quad \& \quad \underset{\text{Recalculate}}{\text{step 3}}$$

What is Convergence?

Centroids don't change much

old Centroid = $\{c_1, c_2 \dots c_k\}$

new centroid = $\{c_1', c_2' \dots c_k'\}$

$c_1 - c_1'$, $c_2 - c_1'$, $c_k - c_k'$

distance b/w new-centroid & old centroid is small

35.8. How to Initialize K-means++

Lloyd's algo + Initialization stage

↓

random ⟸ random : pick K-pts randomly from D
Initialization

↓

$c_1, c_2, c_3 \cdots c_k$

⇒ K-means has a problem called initialization sensitivity

→ final clusters & centroids are very much dependent on initialization po

How to deal with this problem

① repeat K-means multiple times with different initializations

↳ Pick the best clustering based on
{ smaller intra cluster distance
large inter cluster distance

② K-mean++

↳ Instead of random-init → it uses smart initialization

Initialization in K-means : (Task) pick $c_1, c_2, \cdots c_k$

① Pick the first centroid randomly → c_1 from D

② $\forall x_i \in D$ create a distribution as follows

dist

$x_i \longrightarrow \text{dist}^2 (x_i, \text{nearest centroid}) \Rightarrow ||(x_i - c_1)||^2$

x_1	d_1
x_2	d_2
x_3	d_3
⋮	⋮
x_n	d_n

pick a pt from $D - \{c_1\}$ with a prob. prop to d_i

→ This is called a probabilistic approach.

→ Pick a point with large distance, because it is farther away from the centroid c_1

(Q) why do This probalistically?

k-mean ÷ does get effected with outliers

35.9 Failure cases / Limitations

→ K-means has problems when clusters are of diffent

- Sizes
- Densities
- Non-globular shapes. (non-convex) 

→ k-means has problems when data has outliers.

Solution for all above problems is to increase the "k"

35.10 K- Medoids

→ Centroids may not be interpretable

Problem K-means ÷ $\underbrace{C_1, C_2, \cdots C_k}$

$D : \{x_1, x_2 \cdots x_n\}$

→ we will get data point instead of centroids

Partitioning around ~~methods~~ medoids (PAM) : k- medoids.

① Initialization ÷ k- means++ → Probabilistic method pick k pts from D

② Assignment ÷ closest medoid → same as in k mean.

$\{ x_i \in S_j$ if medoid is the closest medoid to x_i

③ update/recompute

k mean ÷ $c_j = \frac{1}{|S_j|} \sum_{x_i \in S_i} x_i$

k-medoids (PAM) ÷ (a) swap each medoid with a non medoid point
(b) If loss decrease keep the swap & undo the swap

loss in k-means

J_1

min

$$\sum_{i=1}^{k} \sum_{x \in S_j} ||x - \underbrace{m_j}_{\text{medoid}_j}||^2$$

$\overset{m_1}{x_1}, x_2, x_3 \quad x_4 \quad x_5 \quad \underset{m_2}{x_6} \quad x_7 \quad x_8 \quad x_9 \quad x_{10}$

(a) loss value

$x_1 = m_1 \ ; \ x_6 = m_2 \rightarrow \text{loss} \rightarrow l_1$

(b) swap $M_1 = x_2$

$m_2 = x_6$

↳ compute loss value $\left.\begin{array}{l} m_1 = x_2 \\ m_2 = x_6 \end{array}\right\} \longrightarrow \text{loss} - l_2$

If $l_2 < l_1$

$m_1 = x_2$

$m_2 = x_6$

Else

$m_1 = x_1$

$m_2 = x_6$

k-medoids

↳ Interpretability

↳ Kernalization ; sim; dist

35.11 Determining the right "k"

↑ hyper-parameter

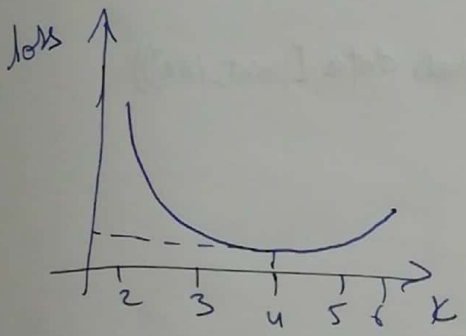
① domain - Knowledge : Food reviews

↳ +ve ~~&~~ -ve

(2 clusters)

Elbow method (or) knee method

$$\text{loss} = \sum_{i=1}^{k} \sum_{x \in S_i} ||x - c_i||^2 \rightarrow \text{minimize.}$$



best k is k=4

35.12 Code Samples

```
Sklearn.cluster.KMeans (n_cluster= 8, init = 'k-means ++', n_init= 10
max_iter =300, tol = 1e-4,)
```

k-medoids

```
from sklearn.metrics.pairwise import pairwise_distances
import numpy as np
import kmedoids
data:-
D= pairwisedistances (data, metric = 'euclidean')
# Split into 2 clusters
M,c = kmedoids.KMedoids (D,2)
```

```
print ('medoids')
for point_idx in M:
    print (data[Point_idx])

print (' ')
print ('clustering result:')
for label in C :
    for point_idx in C[label]:
        print('label {0}: {1}'. format (label, data [point_idx]))
```

35.13 Time & Space Complexity

$K\_means : O(\underset{\#pts}{n}\,\underset{\#clusters}{k}\,\underset{dim}{d}\,i) \rightarrow \#iterations$

Typically $\boxed{k \le 10}$

$\boxed{i \le 100}$

$\approx O(nd) \rightarrow$ linear time Complexity

Space : $O(nd + kd)$

$\approx \underset{\uparrow \text{linear}}{O(nd)}$

* K-means is quite fast