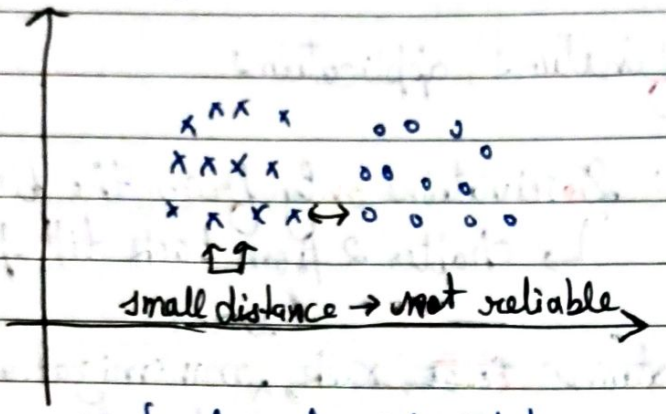
Clustering

It is a type of
→ Unsupervised learning.



⇒ We take a central value to determine if a data point belong to one cluster.

→ Clustering is an unsupervised machine learning technique used to group similar data points together based on their characteristics.

→ It is useful in Exploratory data Analysis to uncover patterns & relationship in data.

→ The fundamental ancept of similarity & dissimilarity are crucial in clustering as they define how points are grouped.

Similarity in clustering :-

Similarity measures how close or alike two data points are. It is often quantified using various distance or similarity metrics.

Common similarity measures are :-

(i) Euclidean distance
(ii) Cosine similarity
(iii) ~~Jaccard similarity~~ Jaccard similarity
(iv) Pearson Correlation coefficient

Dissimilarity in clustering :-

It measures how different two data points are from each other. It is often referred as distance in data space.

Common dissimilarity measures :-

(i) Manhattan distance
(ii) Mahanolobis distance
(iii) Hamming distance.

Minimum within clustered distance criteria

→ The minimum within ~~clustered~~ clustered distance aims to ensure that data points within a cluster are as similar as possible, which means, minimizing the distance between all data points in the same cluster.

→ This leads to tighter, more cohesive clusters.

Within cluster distance :-

It is measure of how far apart the data points within a single cluster are from each other.

The goal is to minimize this distance to ensure that all data points in a cluster are closely related or similar.

→ Within cluster sum of squares :-

For a cluster C_i, the within cluster distance is often calculated as the sum of distances between each point in the cluster and the centroid μ_i of that cluster.

$$WCSS = \sum_{x_j \in C_i} \| x_j - \mu_i \|^2$$

(i) Minimizing within cluster distance improves the homogeneity of the cluster.

(ii) Lower within cluster distance leads to compact clusters.

There is often a trade-off between this and creating well separated clusters.

Major clustering approaches :-

1. Partitioning based algo : k means & K-medoid — DBScan
2. Heirarchical algos : agglomerative or divisive
3. Density based algos : sparse or dense area (non linear clustering)
4. Grid based algos : multidimensional
5. Model based algos : density function
6. Graph based algos : minimum spanning tree.

(i) Partition based :

↖ we define no. of clusters 'k' prior.

↳ this is also a limitation of this method, that we need to decide 'k' in prior very

(ii) Heirarchical

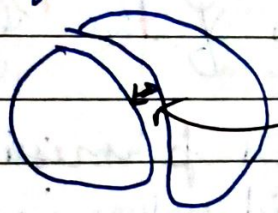
Agglomerative
⇓
top - down

↳ assume each data point is a class & keep joining similar points

Divisive
⇓
bottom - up

↳ assume all points are in same class and then split acc. to dissimilarity

(iii) Density instead of distance is used.



even though there is small distance but density is used to group points

⇒ Nowadays, fuzzy means clustering is used :-
↳ Fuzzy mean clustering → same point can belong to more than one class.

Syllabus

Partitioning : k mean & k medoid
Density based : DB-Scan

K-Means Clustering

(datapoint)

(i) Objects are defined in terms of set of attributes

$$A = (A_1, A_2 \ldots A_n)$$

where, each A_i is a continuous data type.

(ii) Distance computation: Any distance such as L1, L2, L3 or cosine similarity.

(iii) Minimum distance is the measure of closeness between an object and centroid

(iv) Centroid calculation: It is the mean value of each attribute values of all objects.

(v) Convergence criteria: Any one of the following are termination condition of the algorithm.

1. No. of max. iteration permissible
2. No change of centroid values in any cluster.
3. Zero or no significant movement of object from one cluster to another.

Q There are 15 data points, $k = 3$. Use Euclidean distance

Step-1 Plot the points and make guess on potential centroids

Step-2 Calculate distance of each point from centroids. Point belongs to the group whose centroid is nearest

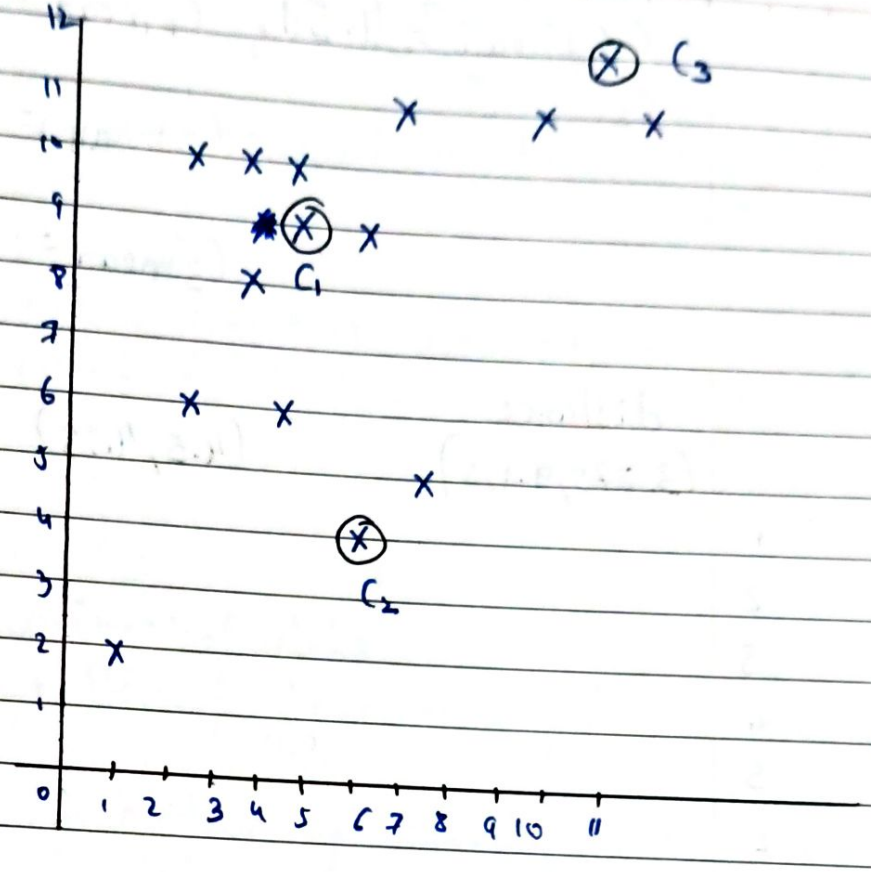
Step-3 Calculate new centroid by taking into account all points belonging to a group.

↳ Now begin next iteration.

S.No	X	Y
1	2	10
2	2	6
3	11	11
4	6	9
5	6	4
6	1	2
7	5	10
8	4	9
9	10	12
10	7	5
11	9	11
12	4	6
13	3	10
14	3	8
15	6	11

Q

Ans :-



Ans Let $C_1(4,9)$, $C_2(6,4)$ and $C_3(10,12)$

distance

	(4,9)	(6,4)	(10,12)	class
1	2.236	7.211	8.24	C_1
2	3.606	4.47	10	C_1
3	7.28	8.602	1.41	C_3
4	2	5	5	C_1
5	5.38	0	8.94	C_2
6	7.616	5.38	13.45	C_2
7	1.414	6.08	5.38	C_1
8	0	5.38	6.708	C_1
9	6.7	8.94	0	C_3
10	5	1.41	7.616	C_2
11	5.38	7.615	1.41	C_3
12	3	2.82	8.48	C_2
13	1.41	6.71	7.28	C_1
14	1.41	5	8.06	C_1
15	2.83	7	4.12	C_1

Date / /

~~C_1 mean $= C_1$, C_2~~ C_1 mean $= (3.875, 9.125)$

C_2 mean $= (4.5, 4.25)$

C_3 mean $= (10, 11.33)$

distance		
$(3.875, 9.125)$	$(4.5, 4.25)$	$(10, 11.33)$

Advantages

(i) k-means clustering is simple and can be used for a variety of objects

(ii) It is efficient from storage point of view as well as from time point of view.

Disadvantages :

(i) It does not work on categorical data.
(ii) If there is some outlier, the performance of k-means clustering degrades
(iii) It finds a local optima and may actually minimize global optima.

→ k-mean gets affected in presence of outliers
∴ k-medoid is used.

k-medoid (PAM)

Partition around medoid

(i) Initialize : Select k random points out of the n data points as medoid.

(ii) Associate each data point to the closest medoid by using any common distance measure.

(iii) Repeat until the cost decreases
- For each medoid 'm', for each data point 'o' which is not a medoid
- Swap 'm' and 'o', associate each data point to the closest medoid and recompute the cost
- if the total cost is more than in the previous step, then, undo the swap.

~~|7-3|+|6-4|~~ dont take bcz. cost high

e.g.

Sol :-

S.No	X_1	X_2	$C_1(3,4)$	$C_2(7,4)$		$C_1(3,4)$	$C_2(7,3)$	
1	7	6	6	②	C_2	6	③	C_2
2	2	6	③	7	C_1	③	8	C_1
3	3	8	④	8	C_1	④	9	C_1
4	8	5	6	②	C_2	6	③	C_2
5	7	4	4	⑥	C_2	4	①	C_2
6	4	7	④	6	C_1	④	7	C_1
7	6	2	5	③	C_2	5	②	C_2
8	7	3	5	①	C_2	5	⑥	C_2
9	6	4	3	①	C_2	3	②	C_2
10	3	4	⑥	4	C_1	⑤	5	C_1

Cost = 20

Cost = 22

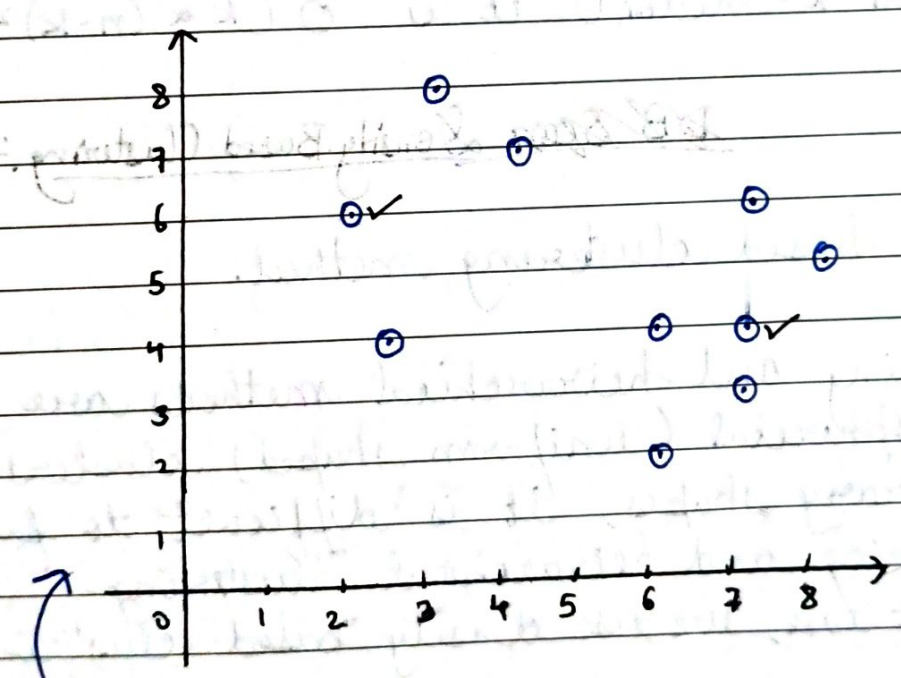
Q For a given $k=2$, cluster the above dataset using PAM. Choose (3,4) and (7,4) as medoid and suppose you are using manhattan distance measure.

Ans e.g. ~~[illegible]~~,

$$Cost = \underbrace{Cost\,((7,4),(7,6))}_{2} + Cost\,((7,4),(8,5)) + \cdots$$

$\therefore$ cost = 20

→ Now, replace (7,4) with some other point and then determine cost.



Draw graph to get better idea about desired medoids

k is chosen using WCSS using elbow method

k-mean v/s k-medoid

(i) → Both algorithms need predefined value for k, i.e. number of cluster prior to the training of algorithm.

(ii) Both algorithm arbitrarily choose the initial cluster centroid or medoid.

(iii) The k-medoid method is more robust than k-mean in the presence of outliers.

(iv) The time complexity of k-mean is $O(n * k * D)$ while in k-medoids it is $O(k * (n-k)^2)$

Density Based Clustering :-

→ Density based clustering method.

→ Partitioning and heirarchical methods are designed to find spherical (uniform shaped) clusters. In case of arbitrary shapes, it is difficult to find with partitioning and heirarchical clustering.
In that case, we use density based clustering methods.

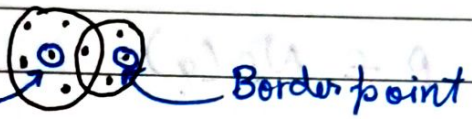
→ Basic idea : We make clusters where data points are dense.
A cluster is defined as a maximal set of density connected points.
It is based on two parameters
(1) ε (maximum radius of the neighbourhood)
(2) min points (minimum no. of points in an ε-neighbourhood of that point

∴ ε is max radius & min points is min. no. of points in a cluster.

→ Core point : 'minpoints' points in ε boundary

→ Border point : less than 'minpoints' points in ε boundary but some of these points are part of core point.

→ Outlier point : none of above two hold



⇒ ε - neighbourhood objects within a radius of ε of an object

$$N_\varepsilon(p) = \{ q \mid d(p,q) \leq \varepsilon \}$$

N_ε : neighbourhood of p

$d(p,q) \leq \varepsilon$: points with distance less than 'ε'

→ High density : ε - neighbourhood of an object contains at least minpoints of objects.

→ Low density : otherwise

→ Given ε and minpoints, categorize the objects into 3 exclusive groups

① A point is a core point if it has more than or equal minpoints objects within ε. These are the point that are at interior of the cluster.

② Border point has data objects fewer than minpoints within ε, but it is in neighbourhood of a core point.

③ Outlier (noise point) is any point that is neither a core point nor a boundary point.

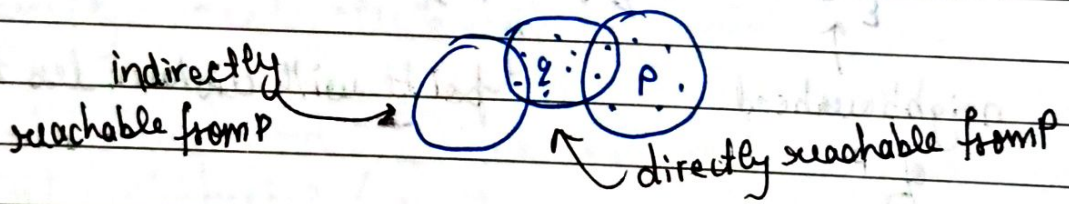
→ Directly density reachable:-

A point p is directly density reachable from a point q w.r.t. ε, minpoints, if p belongs to neighbourhood of q

i.e. $p \in N_\varepsilon(q)$

otherwise indirectly density reachable

→ Density reachable or indirectly density reachable



DB Scan

→ Density Based Spacial Clustering of Application with Noise

Algorithm :-

1. Arbitrarily select point p
2. Retrieve all points ~~does~~ which are density reachable from p w.r.t. ε and ~~for~~ minpoints.

3. If P is a core point, a cluster is formed
   If P is a border point, no points are density reachable from P then DBSCAN visits the next point of the dataset
4. Continue the process until all the points are accessed.

→ Determining ε and minpoints
   - Idea is that for points in a cluster, their k'th nearest neighbour are at roughly same distance..
   - Noise points have the k th nearest neighbour at farther distance.

DOMS Page No.
Date 13 / 10 / 24

Unit-5 : Feature extraction

Dimensionality Reduction

→ Why? to avoid curse of dimensionality.

→ Curse of dimensionality :- → no. of features i.e. dimension of feature vector

As no. of features increases, i.e., we fine tune to dataset, then it nicely follows the dataset.
∴ It will learn & classify very nicely.

But, when test dataset is used or the same algo is applied to other dataset then it is difficult for our model to classify properly

Therefore, no. of features should not be too much.

∴ if no. of features is too much, it shows too good accuracy at training data and low accuracy at test data.

Also, if features are too less, model will miss minor details and misclassification rate increases.

→ Bias and Variance

→ high bias & low variance → low bias and high variance

→ Dimensionality reduction : If two or more features give similar information then they can be reduced to one.

Benefits : (i) space required by system reduce
(ii) less computation
(iii) easy visualization
(iv) remove redundancy

Disadvantages : (i) some data might be lost
(ii) in PCA, principal components are unknown.

→ Approaches

- Feature selection
  - filter method
    - correlation
    - chi square
  - wrappers method
    - forward selection
    - backward selection
    - bidirection elimination
  - embedded methods
    - LASSO
- Feature extraction
  - PCA ✓ in syllabus
  - LDA
  - Kernel PCA
  - QDA

unit - 5

Principal Component Analysis (PCA)

→ Part of unsupervised machine learning

→ Dimensionality reduction + feature extraction

→ Increases interpretability and minimizes data loss

⇒ most important features are those which are independent, otherwise one feature can be used to derive the other.

e.g. $y = \beta_0 + \beta_1 x + e$ in linear regression

(β_0: intercept, β_1: slope, e: residual error)

then, $y \propto x$ and, therefore, we need not store y in data table and dimensionality is reduced by 1.

e.g. weight & height but height is not the only factor. There can be other factors like disease, heredity, eating habits etc. ∴ β_0 & β_1 are not constant and there is residual error 'e'.

Let there are 'n' no. of features
'r' no. of features are correlated
∴ (n-r) features are independent.

e.g. if features are $x_1, x_2, x_3, x_4, x_5, x_6$

if $x_4 = \alpha_{14} x_1 + \alpha_{24} x_2 + \alpha_{34} x_3$
$x_5 = \alpha_{15} x_1 + \alpha_{25} x_2 + \alpha_{35} x_3$
$x_6 = \alpha_{16} x_1 + \alpha_{26} x_2 + \alpha_{36} x_3$

then 6 features can be reduced to 3: x_1, x_2 and x_3

→ Basic terminologies of PCA

- variance → spread of data
- covariance → dependencies & relationship
- standardization → scaling each data within a range i.e. normalization

Covariance matrix, $cov(x, y) = \frac{\sum (x - \bar{x})(y - \bar{y})}{n}$

→ As we know, most of real life functions follows parametric distribution.

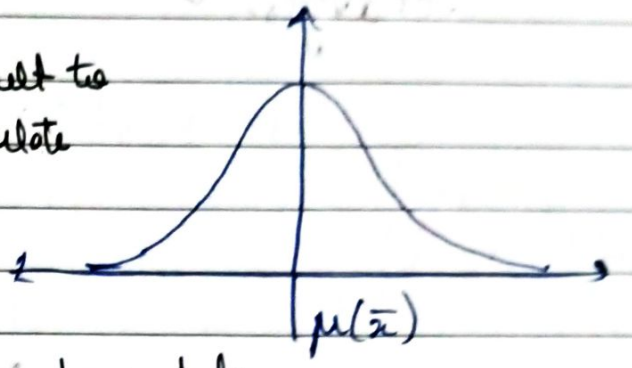
→ Model is based on predictions which use central features such as mean.

→ μ and $\bar{x}$ are different (difficult to calculate)

μ = mean of population

$\bar{x}$ = expected value of mean

↳ sample which is a representative of population



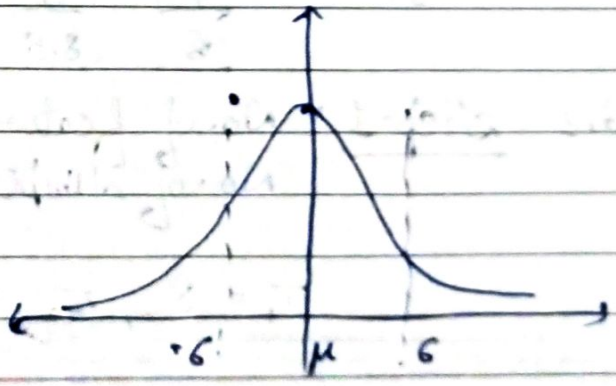
e.g. if population is 20% children, 30% > 60, 50% younger } then sample is a small part of population with similar distribution.

∴ use ground truth to find distribution & then use it to generate a small sample.

∴ $\mu \approx \bar{x}$ but $\mu \neq \bar{x}$

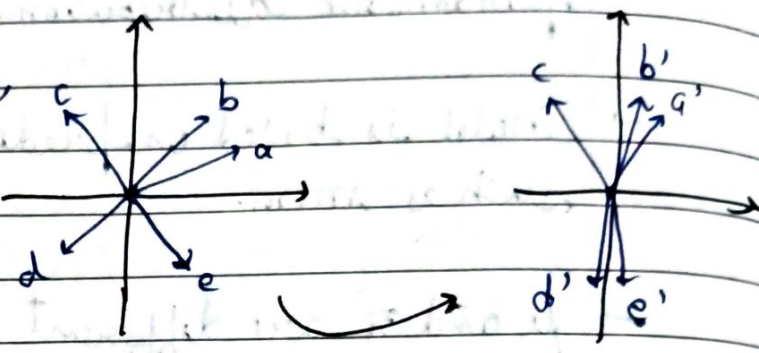
→ Around 60-70% data is from $-\sigma$ to σ.

∴ Covariance is used to find deviation from mean.



→ Eigenvalues & eigenvectors

↳ even after the transformation, its orientation remains same even if its magnitude changes.



linear transformation

↳ direction of all vectors changes except 'c' ∴ 'c' is eigenvector.

→ $\frac{x-\mu}{\sigma} \times$, $\frac{z-\mu}{\sqrt{\sigma}}$ ✓

Normal distribution

↳ standard form of normalization

Q

Sample	x_1	x_2
S_1	4	11
S_2	8	4
S_3	13	5
S_4	7	14
	8	8.5

Ans Step-1 : No. of features = 2
No. of samples = 4

$\bar{x}_1 = 8$, $\bar{x}_2 = 8.5$

Step-2 : Find cov. matrix, $\Sigma = \frac{1}{(N-1)} \begin{bmatrix} cov(a,a) & cov(a,b) \\ cov(b,a) & cov(b,b) \end{bmatrix}$

...

$\therefore \Sigma = \begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}$

Step-3 : Calculate eigenvalue & eigenvector

$|S - \lambda I| = 0$

$\therefore \begin{vmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{vmatrix} = 0 \quad \therefore \lambda^2 - 37\lambda + 201 = 0$

you can use calculator

on solving quad. eqn

$\therefore \lambda_1 = 30.38, \quad \lambda_2 = 6.62$

important

★ Always calculate eigenvector according to highest eigenvalue first.

∴ arrange eigenvalues in descending order.

↳ Note that eigenvalues are always unique but eigenvectors are not ∴ after this step, students might get different answers i.e. sign might be different in answers.

Step-4 : $(A - \lambda I) U_1 = 0$

A ← cov. matrix, λ ← eigenvalue, U_1 → eigenvector

$\therefore U_1 = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ apply any method to find u_1 & u_2

in this example, $\frac{u_1}{11} = \frac{u_2}{14-\lambda_1}$

∴ one possible $U_1 = \begin{bmatrix} 11 \\ -16.38 \end{bmatrix}$

Normalization :

$$\bar{u}_1 = \frac{11}{\sqrt{(11)^2 + (-16.38)^2}} = 0.5575$$

$$\bar{u}_2 = \frac{-16.38}{\sqrt{(11)^2 + (-16.38)^2}} = -0.8862$$

similarly

$$\therefore U_2 = \begin{bmatrix} 0.88308 \\ 0.5574 \end{bmatrix}$$

★ **Step-5**

transpose, x_1 of S_1, $\bar{x}_1$

first principal component → $P_{11} = U_1^T \begin{bmatrix} 4-8 \\ 11-8.5 \end{bmatrix} = -4.305$ ← first eigenvector

x_2 of S_1, $\bar{x}_2$

$P_{12} = U_1^T \begin{bmatrix} x_1 - \bar{x}_1 \\ x_2 - \bar{x}_2 \end{bmatrix} = 3.7859$

(x_1 of S_2, x_2 of S_2)

$P_{13} = 5.692$

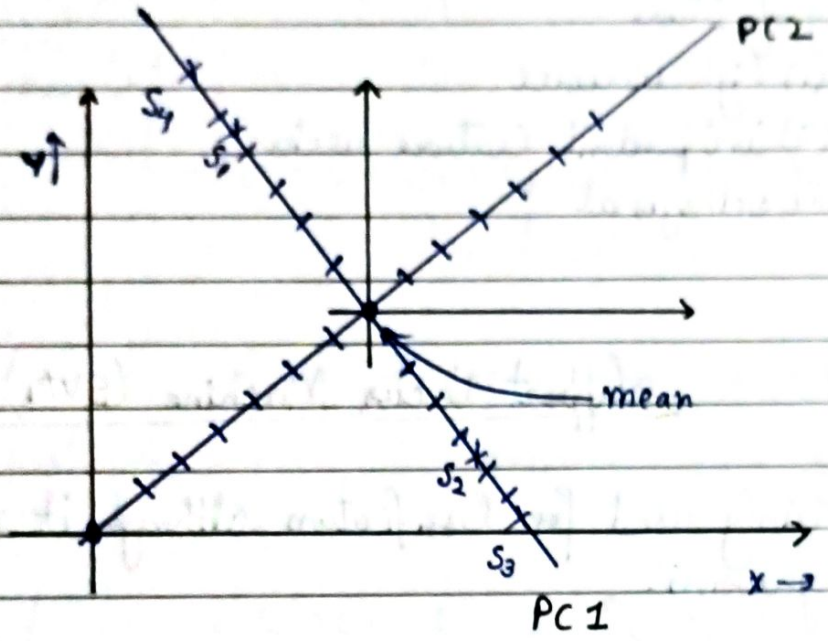
$P_{14} = -5.128$

$$\therefore \boxed{PCT = -4.305 \quad 3.7359 \quad 5.692 \quad -5.123}$$

→ if it is given in question to calculate first principal component

- only find PC, then plot it
- only use largest value of λ

Plotting :-

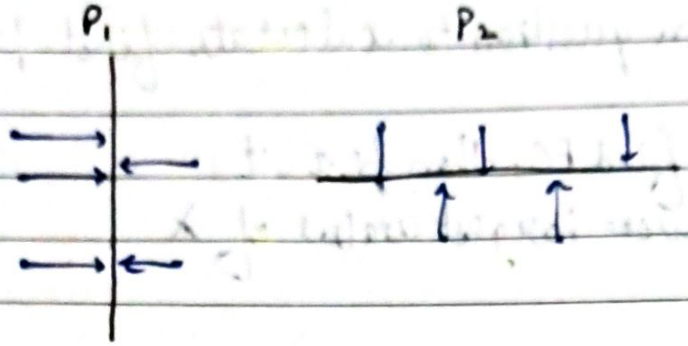


→ we can take any of the two new perpendicular lines as PC1 and PC2, main importance is that they pass through mean & are perpendicular

∴ in this example, we reduced 2D into 1D bcz. max info (90%) is coming from PC1.

Date 14 / 10 / 24

e.g.



P2 is better than P1 because diff. data pts. are at diff. location

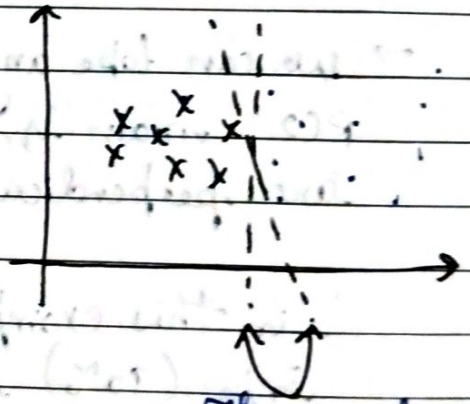
Eligibility of plane
- (i) high variance
- (ii) independent feature vector
- (iii) orthogonal

Support Vector Machine (SVM)

→ It is mainly used for classification although it can also be used for regression.

→ Our goal is to ~~cte~~ create a decision boundary

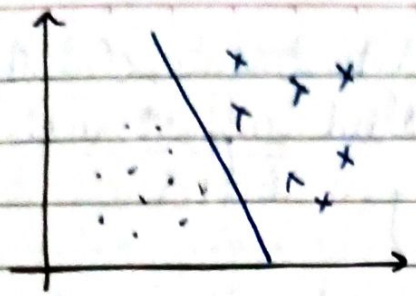
→ We need a criteria to determine best decision boundary.



There can be infinite types of decision boundaries.

→ Let us relate SVM with perceptron.

In perceptron
- (i) choose random decision boundary
- (ii) calculate error
- (iii) update decision boundary
- repeat till min. error.

→ 

$wx + b = 0$

weight vector — i/p vector — bias vector

The distance of boundary from itself is zero

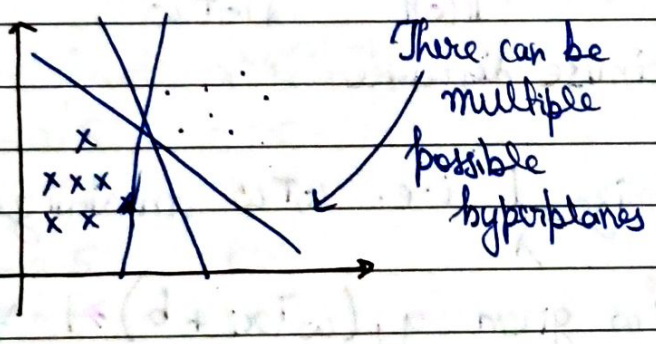
LHS : $wx+b > 0$ ∴ +ve class

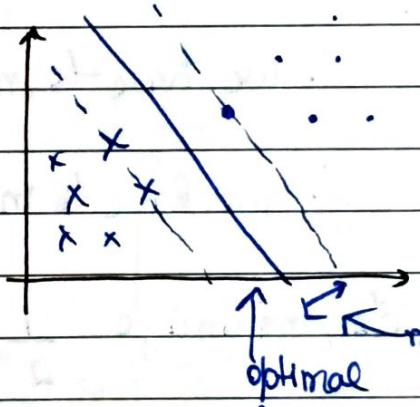
RHS : $wx+b < 0$ ∴ -ve class

↳ Margin : Best decision boundary has the largest boundary

→ 2D : line
3D : plane
>3D : hyperplane

} In general we call it hyperplane in n-dimensional space.

→ 



→ There is chance that some data comes whose features are close to boundary.

↳ ∴ smaller the margin, more difficult to classify unique features

↳ greater margin ensures that in order to change nature there is larger distance to travel to cross threshold.

Linear classifier : $f(x, w, b) = \text{sign}(wx+b)$

↳ +ve or -ve class

→ Margin is determined by the data points closest to the boundary. These points are called support vectors and the help in determining which boundary is better.

∴ farther the support vectors are from boundary, bigger the margin and better boundary.

→ ~~Margin can be defined as max width the~~

→ since we only consider support vectors, it requires less calculation.

Linear SVM :

~~Distance = X Width = Z~~

important : derivation for distance in linear SVM.

→ linear SVM distance = $\frac{2}{\|w\|} = \frac{2}{\sqrt{w^T w}} = \lambda$

∴ we have to maximise distance

∴ we have to minimize $\frac{1}{\lambda}$ i.e. $w^T w$ minimize

→ do, minimize $\frac{1}{2} w^T w$ given $y_i(w^T x_i + b) \geq 1 \quad \forall x_i, y_i$

This is the optimization problem.

⇒ In case of noise, we use more powerful kernels to avoid overfitting.

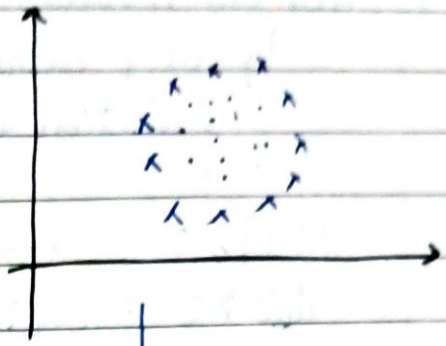
Soft margin classification : minimize $\frac{1}{2} w^T w + C \sum_{k=1}^{R} \varepsilon_k$

constant to avoid overfitting

error for each misclassification

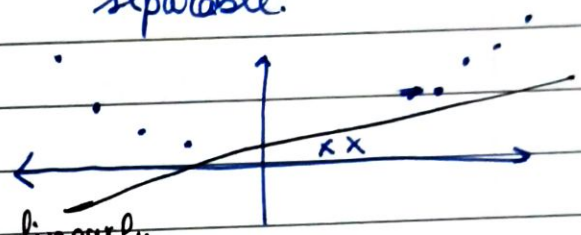
Non-linear SVM

→ Sometimes data is not linearly separable & we need non-linear hyperplane.



→ We increase dimension i.e. make higher dimension space to separate points easily.

transform using $z = x^2 + y^2$ then they are linearly separable.



e.g.

not linearly separable

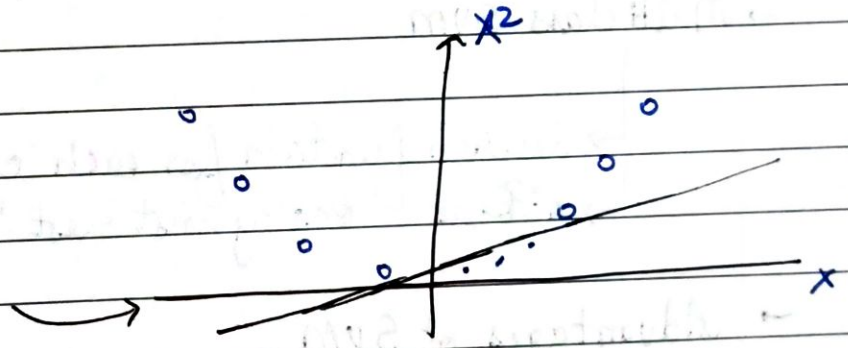
x^2

~~(kernel function)~~

linearly separable

e.g. Red points x^2 ~~Blue points~~

-0.5	0.25
-1	1
-2.5	6.25
-3.5	12.25
2.5	6.25
3.5	12.25
4.0	16



Blue points

~~0.25~~ 0.5	0.25
1	1
1.5	2.25

→ Non linear boundary increases complexity ∴ we map to higher dimension where it is linearly separable.

→ Mercer's theorem
↳ every semi-positive definite symmetric function is a kernel function.

$|det| \geq 0$: semidefinite
$det > 0$: definite positive.

We can use gram matrix to show.

→ Examples of kernel fn $k(x_i, x_j) = x_i^T x_j$ ↳ linear
↳ polynomial
↳ gaussian
↳ sigmoid

→ kernel fn performs the function of dot product in non linear SVM.

→ Multi class SVM
↳ decision function for each class.
↳ "one against rest", "one against one"

→ Advantages of SVM
↳ effective in high dimension
↳ good space complexity
↳ custom kernel fn
↳ good decision boundary with large margin

→ Weakness of SVM
↳ sensitive to noise
↳ SVM underperform in small train than test data
↳ no probability explanation
↳ not effective in large dataset

Q Consider hyperplane $y = x_1 - 2x_2$
predict class for
(i) (1,0)
(ii) (1,1)

Ans distance $w^T x + b$

here, $x_1 - 2x_2 = w^T x \therefore w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

$\therefore$ (i) $\begin{bmatrix} 1 \\ -2 \end{bmatrix}\begin{bmatrix} 1 & 0 \end{bmatrix} = 1 > 0 \therefore$ +ve class ($w^T x_1$)

(ii) $\begin{bmatrix} 1 \\ -2 \end{bmatrix}\begin{bmatrix} 1 & 1 \end{bmatrix} = -1 < 0 \therefore$ -ve class. ($w^T x_2$)

Step-1 : find support vectors
Step-2 : augment bias term

eg. Support vectors : (2,3) for +ve class $= (S_1)$
(4,5) for -ve class $= (S_2)$

let bias is 1

$\therefore \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 5 \\ 1 \end{bmatrix}$

Now, we have to find Lagrangian multipliers

$\alpha_1 S_1 S_1 + \alpha_2 S_1 S_2 = +1$ (S_1 is common & +ve)

$\alpha_1 S_2 S_1 + \alpha_2 S_2 S_2 = -1$ (S_2 is common & -ve)

$\alpha_1 \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}\cdot\begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}\cdot\begin{bmatrix} 4 \\ 5 \\ 1 \end{bmatrix} = 1 \therefore \alpha_1 (4+9+1) + \alpha_2 (8+15+1) = 1$

$\therefore 14\alpha_1 + 24\alpha_2 = 1$

$24\alpha_1 + 46\alpha_2 = -1$

∴ Now we can calculate Lagrangian multipliers α_1 and α_2.

$$w = \sum_i \alpha_i \cdot \overline{s_i} = \alpha_1 \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} 4 \\ 5 \\ 1 \end{bmatrix} = \text{a vector on putting } \alpha_1 \text{ and } \alpha_2$$

Let $w = \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix}_{3\times 1}$ ∴ Here we will get actual value of b

∴ eqn: $y = w^T x + b$ ← Substitute w & b here

↳ This is the eqn of hyperplane.

if A is a set then $|P(A)| = 2^{|A|}$
↑ powerset

DOMS Page No. Date 22 / 10 / 24

Clustering

Hard — each data point belongs only to single class

Soft → a membership function is defined such that point belongs to multiple classes

→ Crisp set :-

⇒ $A = \{a_1, a_2, a_3 \ldots a_n\}$ is a set of events

⇒ $A = \{x \mid p(x)\}$, p : property

⇒ using char. function,

$$\mu_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{otherwise} \end{cases}$$

e.g. set $A = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and property 'p' says even, then $A = \{2, 4, 6, 8\}$

→ Fuzzy set : boundary is not crisp.

Membership Function

⇒ Fuzzy sets : sets with imprecise or vague boundaries

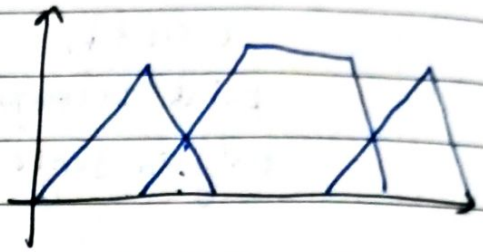
↳ It is a potential tool for handling imprecision and uncertainties

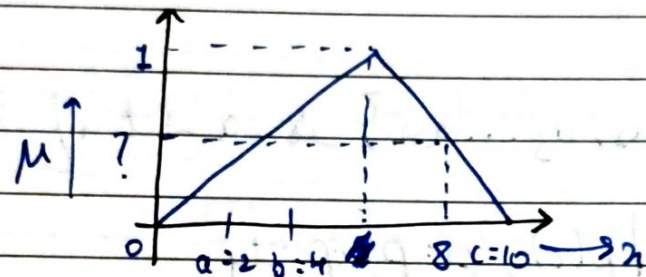
crisp set

Fuzzy set is more general concept of classical set.

How to find the membership function

e.g. fuzzy set will have overlapping areas





Q Calculate membership function μ at $x = 8$

Ans General formula

$$\mu_{triangle} = \max\left[\min\left(\left(\frac{x-a}{b-a}\right), \left(\frac{c-x}{c-b}\right), 0\right)\right]$$

$$\therefore \mu = \max\left[\min\left[\frac{2-2}{2}, \frac{10-x}{6}\right], 0\right]$$

$$= \max\left(\min\left(3, \tfrac{1}{3}\right), 0\right) = \frac{1}{3}$$

For trapezium, compare slope with 1 as well

and d = 10

Q a=2, b=4, c=8, at $x = 3.5$

Ans $\mu = \max\left[\min\left[\left(\frac{x-a}{b-a}\right), \left(\frac{d-x}{d-c}\right)\right], 0\right]$

$= \max\left(\min\left(, \frac{1.5}{2}, \frac{6.5}{2}\right), 0\right)$

$= \max\left(\frac{1.5}{\frac{6.5}{2}}, 0\right) = \cancel{\quad} \times 3.45 = 0.75$

Fuzzy C Mean Clustering (FCM)

↳ It is example of soft clustering

~~↳ (i) membership fn~~

Steps : (i) plot to get an idea
(ii) assign random prob. for each point.
(iii) choose m = fuzziness parameter
↳ usually taken as 2.

Centroid :-

$$V_{ij} = \frac{\sum_{k=1}^{n} r_{ik}^{m} \cdot x_k}{\sum_{k=1}^{n} r_{ik}^{m}}$$

n = no. of cluster
r = fuzziness membership value
x = data point
m = fuzziness parameter

	(1,3)	(2,5)	(4,8)	(7,9)
C_1	0.9	0.7	0.2	0.1
C_2	0.1	0.3	0.8	0.9

Q

Ans $V_{11} = \frac{r_{11}^2 x_1 + r_{12}^2 x_2 + r_{13}^2 x_3 + r_{14}^2 x_4}{r_{11}^2 + r_{12}^2 + r_{13}^2 + r_{14}^2} = \frac{3.8}{1.9} = \frac{2.02}{1.35} = 1.496$

$$v_{12} = \frac{r_{11}^2 y_1 + r_{12}^2 y_2 + r_{13}^2 y_3 + r_{14}^2 y_4}{r_{11}^2 + r_{12}^2 + r_{13}^2 + r_{14}^2} = \frac{5.29}{1.35} = 3.918$$

$$v_{21} = \frac{8.42}{1.55} = 5.43$$

$$v_{22} = \frac{12.89}{1.55} = 8.316$$

∴ Centroids, C_1 : (1.496, 3.918)
C_2 : (5.43, 8.316)

(iv) Calculate the distance of each data point from centroids of C_1 and C_2

$D_{11} = 1.043$
$D_{12} = 6.92$
$D_{21} = 1.194$
$D_{22} = 4.77$
$D_{31} = 4.789$
$D_{32} = 1.464$
$D_{41} = 7.491$
$D_{42} = 1.712$

(v) Update membership values

$$r_{ki} = \left(\sum_{j=1}^{n} \left\{ \frac{d_{ki}^2}{d_{kj}^2} \right\}^{\frac{1}{(m-1)}} \right)^{-1}$$

⇒ Repeat until difference b/w new and old r is ± 0.01

8 Nov : Quiz (3rd + 5th unit)

Feature Selection

→ Raw data → Sensors → Features → Feature selection → preprocessing

→ After collecting data, we don't know which features are more important and which are irrelevant or redundant.
↳ Irrelevant features increase complexity & are not wanted
↳ Redundent features are only diplicates and dont provide any value.

e.g.

Price	Address	No. of rooms	House age	Owner	Area of rooms

Here, while determining price, 'Owner' is irrelevant.
Also, 'No. of rooms' is redundant due to already provided information 'Area of rooms'.

Feature Selection Models

- Supervised
  - Filter method
  - Wrapper method
  - Embedded method
- Unsupervised
  - Data dimension criteria

Filter method

→ Some considerations for feature selection :-

1. Classification frequency must be same as earlier.
2. Classification of most data points will remain same as earlier.

↳ we are reducing features to :-
  ↳ To reduce time & space complexity.
  ↳ To avoid overfitting.

Filter method :-

→ Univariate
  ↳ Each feature is independent
  ↳ so, we consider individual feature

Multivariate
  ↳ we take a batch of features.

→ It is usually not as good as wrapper or embedded method

→ It also uses score.

(i) Fisher score,

$$S_i = \frac{\sum_{j=1}^{K} \left[n_j (\mu_{ij} - \mu_i)^2 \right]}{\sum_{j=1}^{K} \left[n_j \rho_{ij}^2 \right]}$$

n_j → no. of features
μ_{ij} → mean for a sample
ρ_{ij}^2 → variance

$i \rightarrow$ feature
$j \rightarrow$ class
$\mu \rightarrow$ mean
$\rho \rightarrow$ variance

Data table :-

Instances / samples	x_1	x_2	x_3 ...	x_K
→ S_1				
→ S_2				
→ S_n				

Entropy : gives mutual information

↳ Randomness gives hidden patterns ∴ more the randomness, higher the information

∴ Information gain, $IG(F_i, C) = H(F_i) - H(F_i/C)$

where, $H(F_i) = \left[- \sum_j P(x_j) \log (P(x_j)) \right]$

$$H(F_i/C) = \left[- \sum_k P(C_k) \sum_j P(x_j/C_k) \log_2 (P(x_j/C_k)) \right]$$

In such cases, we determine a threshold value, and determine

$$SU(F_i/C) = \frac{2\, IG(F_i/C)}{H(F_i) + H(C)}$$

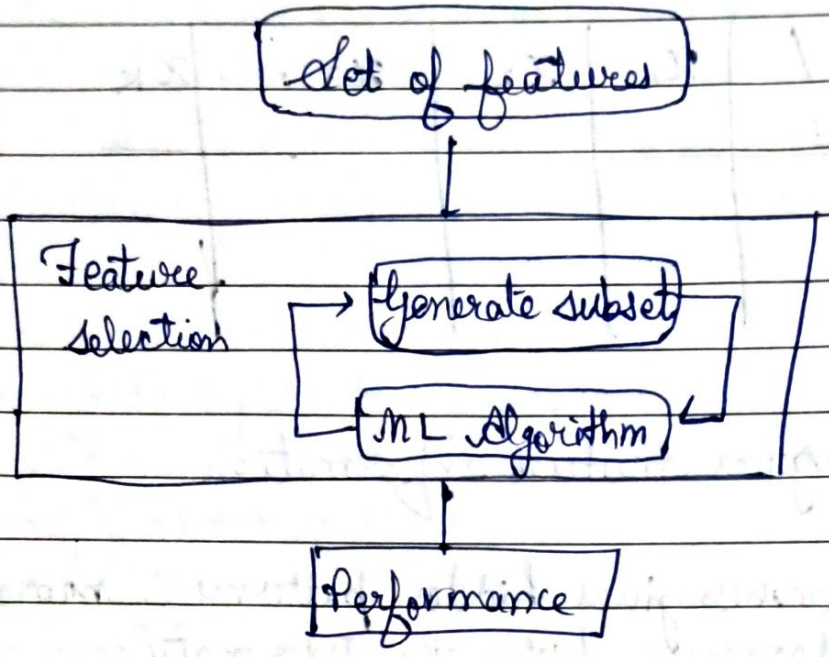
∴ if $SU(F_i/C) > I$ where I is threshold, we call it pre dominant value.

∴ Such features are selected.

⇒ Filter methods are independent of specific classifier.

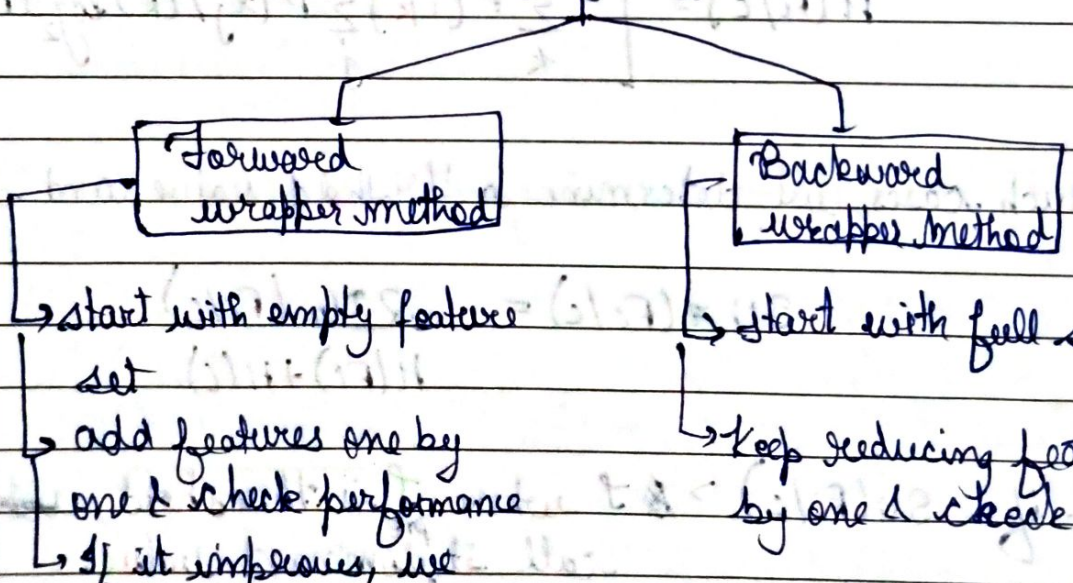
→ also called greedy approach

Wrapper methods



If performance improves : we take that feature
" remains same : redundant feature : ignore
reduces : irrelevant feature : ignore

→ wrapper methods are of two types

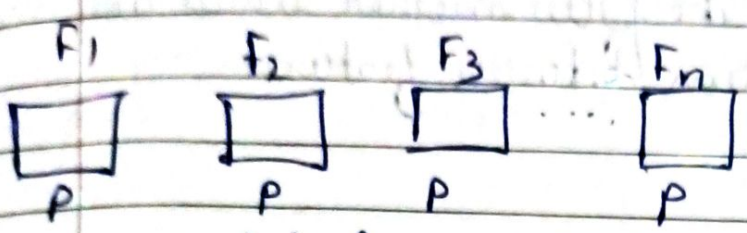


→ start with empty feature set
→ add features one by one & check performance
→ if it improves, we keep Fi otherwise we ignore

∴ for n features, in first iter we take n models.

→ start with full set
→ keep reducing features one by one & check performance

Forward wrapper



① n models for each feature
② Let us say Fi is best performance ∴ now we make (n-1) models by combining $F_i F_j$ where $j \neq i$
③ Again take the model with max value of P. Let us say $F_i F_j$

∴ take $F_i F_j$ as fixed ∴ make (n-2) models
$F_i F_j F_1$, $F_i F_j F_2$ ... $F_i F_j F_{n-2}$
④ Again take model with max performance
⋮

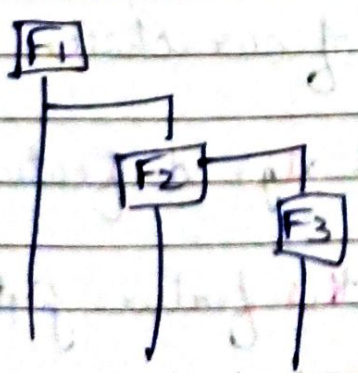
Repeat until no more combination is left.
⇒ Hence we recieve a subset of best performance model.

Backward wrapper

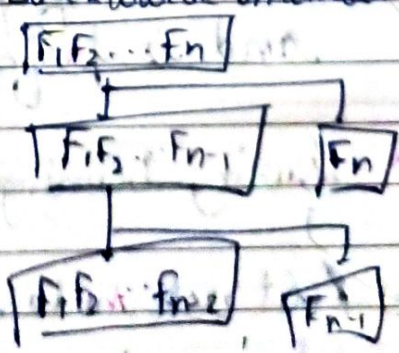
① $F_1 F_2 F_3 \cdots F_n$
↓
ML model
↓
② $F_1 F_2 \cdots F_{n-1}$ | $F_1 F_2 \cdots F_{n-2} F_n$ | ...
↓
③ Take the model with best performance

→ We take a threshold value to stop when sufficient features have been obtained.

Forward method :



Backward method

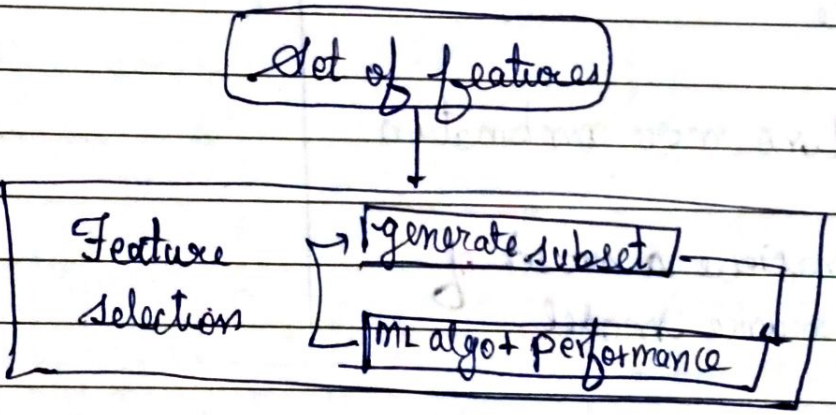


⇒ Wrapper method uses relation b/w various features using statistical methods, unlike filter method which does not take into account relation between features.

Statistical methods

- (i) Correlation coefficient: If x_1 and x_2 are highly correlated, one of them is redundant.
- (ii) chi-square test
- (iii) ANOVA
- (iv) Variance thresholding

Embedded method :- (intrinsic method)



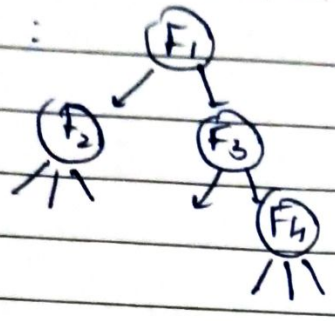
⇒ We are calculating performance as well within model.

① ∴ Selection of feature happens simultaneously with and is performed by ML algo of our choice.

② During training, some steps of ML algo do feature selection

③ At each node split they choose better feature just like decision tree.

Decision tree :



→ initially, features are random
→ matrices are calculated which determine which feature is better at a particular level.

⇒ Filter methods are less used due to time & space completity
⇒ wrapper & embedded methods are widely used.
⇒ embedded is considered best method due to its efficiency.