UNIT 2

Random forest:

- working
- Advantages
- Disadvantages.

RANDOM ⇒ steps
Bagging → choosing small amt of data from original all data. (dataset)

⇒ feature selection

x_1, x_2, x_3, x_4 → columns of a dataset

$\sqrt{4} = 2$ features are choosen.

→ different decision trees

FOREST ⇒ collection of trees lead to forest.

⇒ many trees may encounter different errors / mistakes and so this will help to learn and know to address each of behaviour, so performance increased.

Leo Bierman ⇒ proposed Random forest.

① Random

2 factors.

(i) Bootstraping → enables to choose smaller subset of entire dataset

Randomness leads to different decision trees.

each decision tree has different type of mistakes.

③ Forest
collection of different decision trees

Eg: Assume dataset
100 training examples = N
$n_1, n_2, n_3 \ldots, n_n = 300$
↓ subsets of dataset
30 30. training examples because of bootstraping

WORKING:

① Building Random forest

ⓐ It is ensemble learning that is each decision tree is a weak learner but group of weak learning decision tree will give better performance.

ⓑ Assume N training examples from which $n_1, n_2 \ldots n_n$ training examples are chosen randomly.

ⓒ Assume N features out of which m features are choosen

ⓓ The decision trees are constructed based on finding the best split node.

② When the new data input arrives it is passed through each of the decision tree in Random forest., then each tree predicts the label, majority label will be choosen as class of new data input.

first step is Bootstraping
second step is Aggregation
} together called as BAgging

⇒ Regression using Random forest
step (A) remains same
in step(B), it is averaging instead of aggregation.
↓ majority

- Advantages.
① less training time even for larger dataset
② Accuracy is more in 2 cases.
ⓐ large dataset ⓑ Noisy data
③ used both for classification and regression

- Disadvantages.
① memory requirement more. (many DTs).
② Blackbox (internal factors of DT, we can't get to know).
③ output is limited to range present in training dataset.

26th
- Unsupervised learning

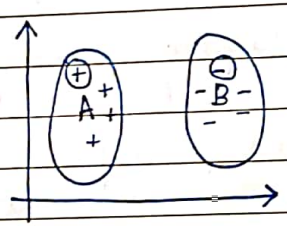
⇒ Obtaining the data-cleaning
⇒ cluster analysis.

DOMS Page No. Date / /

ⓐ no overlap (one & only one cluster)
ⓑ centroid (every cluster must have a centroid)
(Avg of numeric values in training set)
ⓒ similarity within the cluster compared to datapoints in other cluster.

⇒ Obtaining useful information from existing data.
→ Principle of unsupervised learning.

• cluster analysis Algorithm:



this is a 2 dimensional visualisation and is having 2 clusters beyond 3 dimensions. It is difficult for humans to visualise the data. and identify the clusters. Therefore cluster analysis algorithm are required.

2 different approaches: of unsupervised learning
① clustering
② Association (minimal dependency btw data).

K- Means algorithm

DOMS Page No. Date / /

I Input
II ① ② ③ ④ } Body of algorithm (4 substeps)
III Termination
IV Output

• INPUT
1.) Set of training egs without labelling
2.) no of clusters (no of k values)
k values → (constant) : ① user specified
② ML specified automatically.
(software ML generate automatically).

• BODY.
① Create k clusters, which are initially empty.
calculate the centroid of each cluster
② Consider a training example x, represented in the form of vector (numerical representation of data is called vector).
Calculate the distance of this training eg x from centroid of each cluster. (k clusters)
Let J denote the nearest cluster to the training example x.
③ Move x to J, if x is already in J, do nothing else relocate that x from that cluster to J.
Recalculate the new centroids of clusters.
④ Repeat steps ② and ③ till termination or stopping

condition is met.

III every training example is in its nearest cluster. i.e no relocation is required. – TERMINATION.

IV OUTPUT.
K clusters with similar examples.

⇒ Disadvantage:
supplying K value is tricky.

⇒ Advantage:
simple & easy to understand.

• Euclidian distance – used to calculate the distance b/w training example and centroids of clusters.

$$d(x,y) = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$$

data points	x_1	y_1	Distance to x_2, y_2 = 2, 10	x_3, y_3 = 5, 8	x_4, y_4 = 1, 2	cluster	new cluster
A1	2	10	0	3.6055	8.0622	1	
A2	2	5	5	4.242	3.1622	3	
A3	8	4	8.485	5	7.2801	2	
B1	5	8	3.6055	0	7.2111	2	
B2	7	5	7.071	3.6055	6.7082	2	
B3	6	4	7.211	4.1231	5.3851	2	
C1	1	2	8.062	7.2111	0	3	
C2	4	9	2.236	1.414	7.615	2	

① ② ③

Initial centroids : k=3
A(2,10)
B1(5,8)
C1(1,2).

2nd cluster (A3, B1, B2, B3, C2).
new centroid = 8+5+7+6+4 (avg) / 5
x = 6
y = 4+8+9+4+9/5 = 5

3rd cluster = (A2, C1) x: 2+1/2 = 3/2 = 1.5 = x
y = 2+2/2 = 3 = y
7-2 = 5
7/2 = 3.5

1st cluster (A1) = (2,10)
B1 = (6,6)
C1 = (1.5, 3.5)

Data points	x	y	I (2,10)	II (6,6)	III (1.5,3.5)	cluster	new cluster
A1	2	10	0	5.656	6.519	1	1
A2	2	5	5.	4.12	1.58	3	3
A3	8	4	8.485	2.82	6.51	2	2
B1	5	8	3.605	2.23	5.700	2	2
B2	7	5	7.071	1.414	5.70	2	2
B3	6	4	7.211	2	4.527	2	2
C1	1	2	8.062	6.40	1.58	3	3
C2	4	9	2.236	3.60	6.041	2.	(1)

1 ⇒ A1, C2 = $\frac{2+4}{2} = 3$, $\frac{10+9}{2} = 9.5$ = A1 = (3, 9.5).

2 ⇒ A3, B1, B2, B3 = B1 (6.6, 5.25)

C1 = (1.5, 3.5)

	I (3, 9.5)	II (6.5, 5.25)	III (1.5, 3.5)	New cluster
A1	1.118	6.543	6.519	1
A2	4.609	4.506	1.58	3
A3	7.433	1.952	6.51	2
B1	2.5	3.132	5.700	(1)
B2	6.020	0.559	5.70	2
B3	6.264	1.346	4.527	2
C1	7.762	6.388	1.58	3
C2	1.118	4.506	6.041	1

I (3.6, 9)	II (7, 4.33)	III (1.5, 3.5)	new cluster
1.88	7.55	6.519	1
4.308	5.044	1.58	3
6.66	1.05	6.51	2
1.72	4.17	5.700	1
5.25	0.67	5.70	2
5.54	1.05	4.527	2
7.46	6.43	1.58	3
0.4	5.55	6.041	1

Termination condition

There is no change in cluster, so all data points are in appropriate cluster.

- Hierarchical clustering : ALGORITHM.

Input section : Set of unlabelled training examples.

Step 1: each example will form one cluster, so for n training examples there are N clusters initially.

Step 2: calculate the distance btw 2 clusters. The last cluster to cluster distance will be merged.

∴ the clusters reduced from N to N−1.

Step 3: the second step is repeated till the termination condition (some threshold specified is reached) on the no. of clusters.

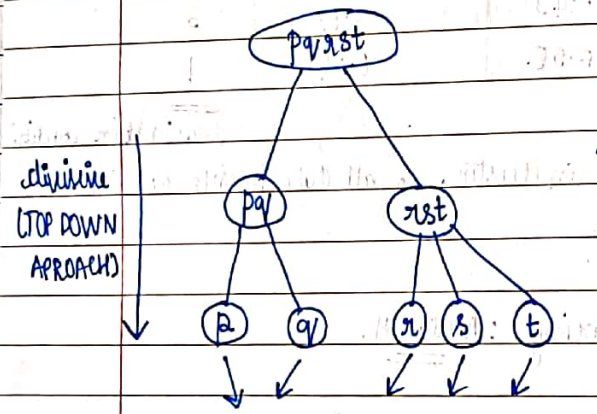
1 April 2024.

Hierarchical clustering :

2 types:

① Agglomerative
(Latin word means collecting)

② Divisive
splitting



- HAC

→ Hierarchical agglomerative clustering

3 types:

① single linkage } minimum distance btw clusters (min)
② complete linkage }
③ Average linkage

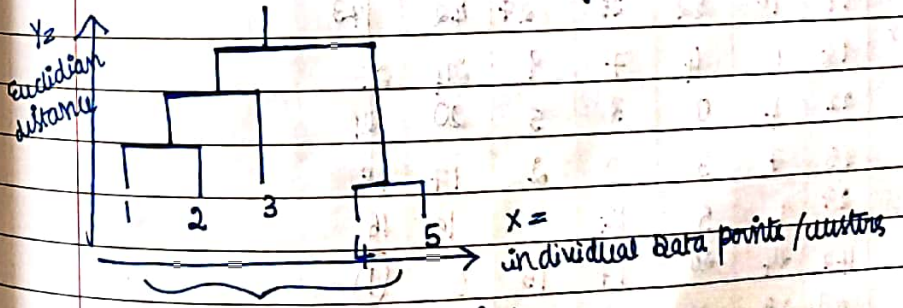
① single linkage
merges the two clusters and finds distance and chooses min
(r,s), (t)
(r,t), (s,t)
(a→b→c) ...

it calculates the minimum distance btw 2 clusters and uses f/n min. gives a longchain cluster).

② complete linkage
max f/n uses.
usually spherical in shape.
(r,s), (t)
(r,t), (s,t) max distance btw 2 options
max(r,y)
→ covers many eg.

③ Average linkage. (not there).

- Represented by Dendogram
pictorial representation of Hierarchical clustering.
2 greek words : Dendo = Tree
gram = Drawing



⇒ 1 Dimensional points (x points only given) (1 coordinate of a point is given)

⇒ 2 dimensional points. Both x & y coordinates of point P are given.

- Euclidian Distance $= \sqrt{(x_2 - x_1)^2} = (x_2 - x_1)$

1. y coordinate is absent
2. sq root & square gets cancelled.

⇒ Applications of HAC:

① fake news detection
   ⓐ identify the keywords.
   ⓑ Appropriate distance will be calculated and will be grouped under one cluster.

1) find the single linkage for the following 10 points.

	18	22	25	27	42	43
18	0	4	7	9	24	25
22	4	0	3	5	20	21
25	7	3	0	2	17	18
27	9	5	2	0	15	16
42	24	20	17	15	0	1
43	25	21	18	16	1	0

① is the minimum non zero distance

step 2: minimum distance = 1.
42 and 43 are clubbed together and eliminate 43 row and column.

II

	18	22	25	27	(42,43)
18	0	4	7	9	24
22	4	0	3	5	20
25	7	3	0	2	17
27	9	5	2	0	15
(42,43)	24	20	17	15	0

step 3:

III

	18	(22,)	(25,27)	(42,43)
18	0	4	7	24
(22,)	4	0	3	20
(27,25)	7	3	0	17
(42,43)	24	20	7	0

3 is min distance between clusters and (25,27) Hence the cluster (22) and (25,27) are merged into a single cluster equ to (22,(25,27))

IV

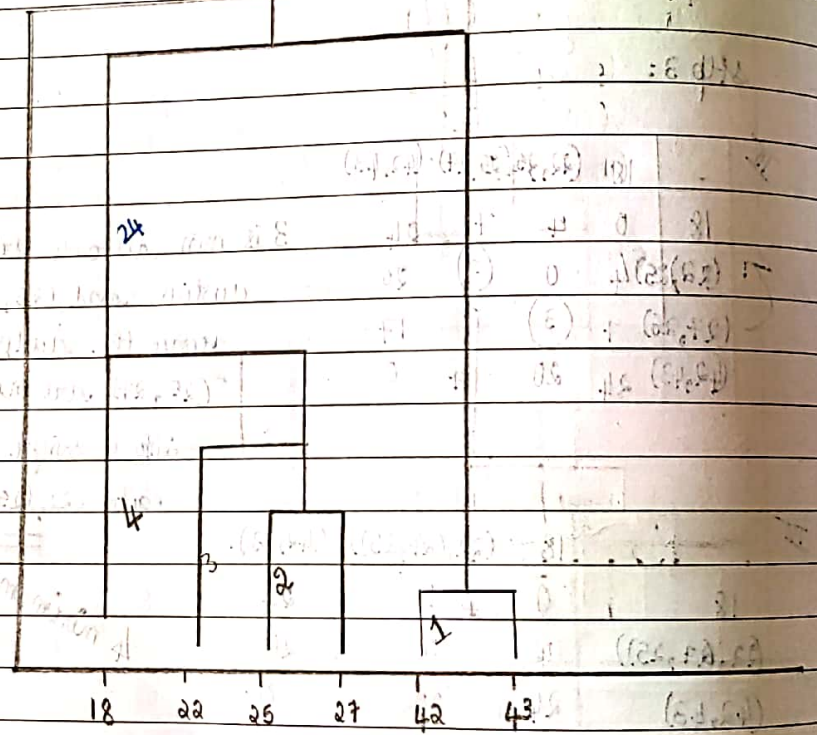
	18	(22,(27,25))	(42,43).
18	0	4	24
(22,(27,25))	4	0	20
(42,43)	24	20	0

4 minimum

IV

	(18,(22,(25,27)))	(42,43)
(18,(22,(25,27)))	0	24
(42,43)	24	0

(18,(22,(25,27))) , (42,43)



2) complete linkage
for the following data points : 1, 5, 8, 10, 2.

	1	5	8	10	2
1	0	4	7	9	1
5	4	0	3	5	3
8	7	3	0	2	6
10	9	5	2	0	8
2	1	3	6	8	0

1 2 3 4 5 → column nos

↓ row nos

(2,(1,5))

Distance btw (2)(1,5) is one cluster.

(2,1),(2,5)

max (x, y)

max [dist(x), dist(y)]

max(4,3)

= 4

Distance (3,(1,5)) ⇒

max((3,1)(3,5))

= (7,6)

= 7

	(1,5)	2	3	4
(1,5)	0	4	7	9
2	4	0	3	5
3	7	3	0	2
4	9	5	2	0

(1,5) = 1.

	(1,5)	2	(3,4)
(1,5)	0	4	9
2	4	0	5
(3,4)	9	5	0

d(1,5)(3,4) = max((1,5)3),((1,5),4)
max = max(7,9)
= 9

Distance max of (4,(1,5))
= (4,1)(4,5)
= (9,8)
= 9

d(2,(3,4)) = max(d(2,3), d(2,4))
= max(3,5)
= 5

d((3,4),2) = (d(3,2), d(4,2))
= (5,3) max = 5

single linkage - neighbour clustering
complete linkage - farthest neighbour clustering

	(1,5,2)	(3,4)
(1,5,2)	0	9
(3,4)	9	0

(1,5),2), (3,4).

max(d((1,5),2),3) and d((1,5),2),4)

= max((1,5) with (3,4) and (2) with (3,4)

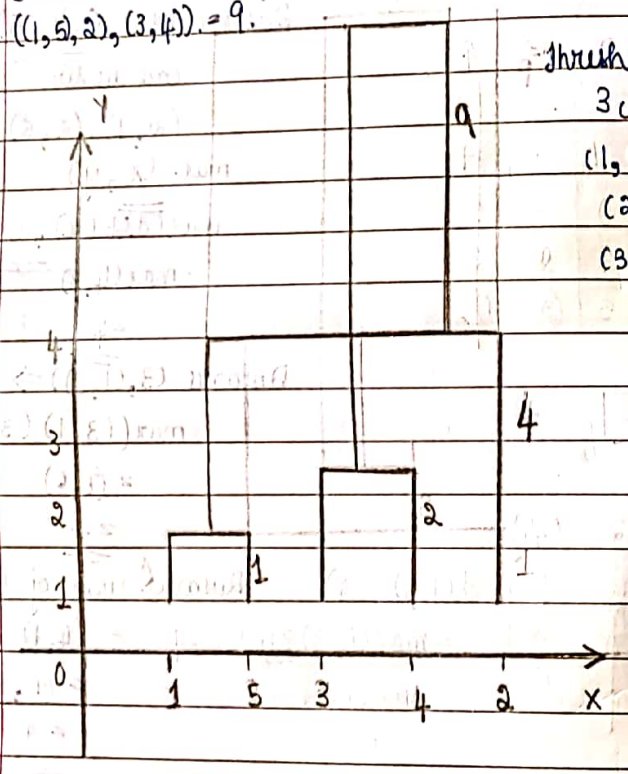
= max ((1,5)(3,4) & (2)(3,4)

= 5,9 = 9

2) (3,4) = 2

3) ((1,5),2) = 4.

4) ((1,5,2),(3,4)) = 9.

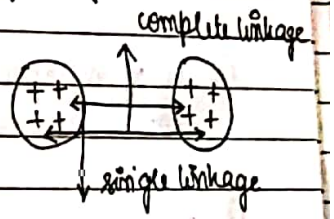


⇒ HAC (Hierarchical Agglomerative clustering)
looks for similarities between 2 points belonging to 2 different clusters.

similarly, (HDC)
Divisive hierarchical clustering looks for similarities btw 2 points where a pt belongs to cluster A and a pt belongs to cluster B.

→ single linkage / single nearest distance / Neighbour clustering
distance btw the closest members of 2 clusters A and B.

→ complete linkage / complete farthest distance / farthest neighbour clustering
distance btw 2 farthest members / pts of 2 diff clusters A & B.



• Dendrogram:
combining leaf nodes leading to cluster.

→ In HAC., we need not to have predefined no of clusters naming k mean clustering.

• Agglomerative consumes more time & space, it ensures nearby points (near) end up in same cluster.

• Identification of criminal activities } Applications
• Document analysis.

8 April

Unit3

ANN (Artificial neural network).

2 researchers : Mc Culloh
Pitts

functions of brain : recognising & discriminating

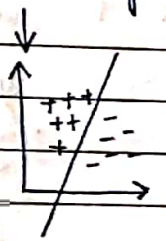
HUMAN BRAIN.	ANN
① Neuron switching time approx = 0.001sec.	② Many neural links threshold switching units
② No of neurons approx = 10^{10} or 10^{11}.	② many
③ connections per neuron is 104 to 10^5	③ many
④ scene recognition time approx is 0.1sec.	④ no specific point
⑤ much parallel computation	⑤ highly parallel.

- When to use ANN ?

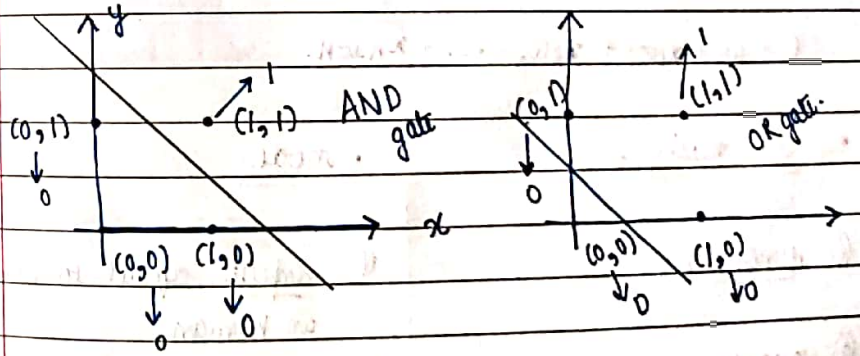
① Input is high dimensional
② noisy data
③ when target fn form is unknown

I st type of ANN is PERCEPTRON (Rosenblatt).

- simplest ANN
- proposed by Frank Rosenblatt.
- perceive - recognition automaton
- supervised ANN
- used with linear data (eg : linear classification)



- Graphical representation for logical gates.



AND and OR are linearly separable but XOR gate is not.
So perceptron can't be used with XOR gate.

Components of perceptron.

① Input layer (X) → $x_1, x_2, x_3 \ldots x_n$

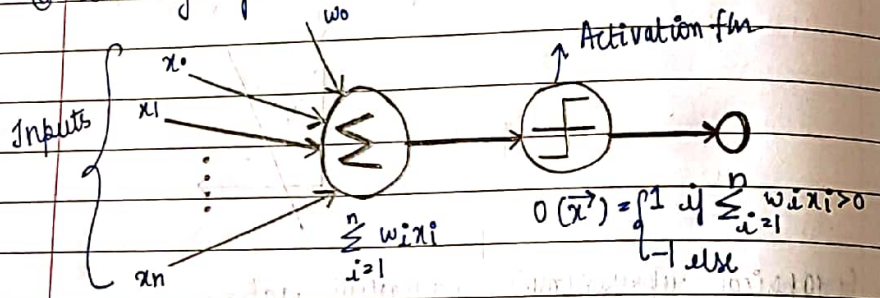
② weight (W) → $w_1, w_2, w_3 \ldots w_n$.

③ Bias.
It adds more flexibility to the network

④ Activation function

⑤ output

⑥ training algorithm.



$o = w_0 + x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$

• cell nucleus	• node.
① dendrites	① input equal to dendrites in human.
② synapse.	② weights
③ axon	③ output

• Perceptron training rule :

$w_i \leftarrow w_i + \Delta w_i$ → change in weight

$\Delta w_i = \eta (t-o) x_i$ → ith training eg (ith i/p).
(t-o) → error

η → eta → learning rate → lower the better

t → target value. o → predicted value.

• Gradient descent:

⇒ Gradient means change, descent means downward.

⇒ it is an optimization algorithm.
(reduce the error).

① $E[\vec{w}] = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$
error w.r.t weight vector

② $\nabla E[\vec{w}] = \left[\frac{\delta E}{\delta w_0}, \frac{\delta E}{\delta w_n} \right]$

③ $\Delta \vec{w} = -\eta \frac{\delta E}{\delta w_i}$ (change in weight)

• $\frac{\delta}{\delta w_i} \left(\frac{1}{2} \sum (t_d - o_d)^2 \right) = \frac{1}{2} \sum \frac{\delta}{\delta w_i} (t_d - o_d)$

$= \sum (t_d - o_d) \frac{\delta}{\delta w_i} (t_d - o_d)$ → target value is const

$= \sum (t_d - o_d) (-x_{i,d})$

$= \delta \frac{\sum (O - w_i x_i)}{\delta w_i}$

$= -x_i$

- Back-propagation algorithm (9 April 2024).

① multilayer network
② non-linearly separable decisions

INPUTS:

1.) training example → $\langle \vec{x}, \vec{t} \rangle$
$\vec{x}$ → each training example
$\vec{t}$ → target value associated with each training example.

2.) η_{in}
n → network in → inputs

3.) η_{hidden} → network → units

4.) η_{out} → network → output units

5.) η → eta → learning rate

All these are inputs of BP algorithm

STEPS:

I create NN II propagate @ forward → i/ps III update weights
ⓑ backward - errors

- Back propagation Algorithm of training example.

① Each training example is a pair of the form $\langle \vec{x}, \vec{t} \rangle$ where $\vec{x}$ is a vector of network input values, $\vec{t}$ where is a vector of network output values.
↓
target

Steps:

I create a feed forward network with η in inputs, η_{hidden} units, η_{out} output units.
Initialize all network weights to some small random value. (-0.5 to 0.5) until the termination condition is met.

② for each training example $\langle \vec{x}, \vec{t} \rangle$, do
propagate the input forward through the network.

II Propagate the input forward through the network.

ⓐ Input the instance $\vec{x}$ to the network and compute the output O_u of every unit u in the network - foward propogation.

ⓑ Propagate the errors backward through the network.
for each output network unit k, calculate its error term δ_k.

$\delta_k \leftarrow O_k(1 - O_k)(t_k - O_k)$ ← $\delta(-)$

where $O_k(1 - O_k)$ is the derivative of sigmoid function ($\sigma(-)$)

also called as sigmoid squashing fn

for each hidden unit h.

$$\delta_h \leftarrow O_h(1-O_h) \sum_{k \in \text{outputs}} w_{kh}\, \delta_k$$

- derivative of sigmoid activation fln w.r.t h. $\Rightarrow O_h(1-O_h)$.
- w_{kh} → weigh associated from hidden layer to output layer.

III Update each network weight w_{ji}

$$w_{ji} = w_{ji} + \Delta w_{ji}$$

where

$$\Delta w_{ji} = \eta\, \delta_j\, x_{ji}$$

where x_{ji} is input from ith unit to jth unit and the weight associated with the connection is w_{ji}.

δ_j → error associated with jth unit.

⇒ the stochastic gradient descent version of back propation algorithm for feed forward networks containing 2 layers of sigmoid units. it as discussed above.

- Stothastic → greek word → guess work.
meaning guess or aim
Here, the weights are updated independently of each other and done before the summation(Σ) unlike after Σ in gradient descent algorithm.

→ as updation is individual, performance is better.

- sample No. X Y

P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

①

	P1 (0.40, 0.53)	P2 (0.22, 0.38)	P3 (0.35, 0.32)	P4 (0.26, 0.19)	P5 (0.08, 0.41)	P6 (0.45, 0.30)
A (0.40, 0.53)	0	0.234	0.215	0.367	0.3417	0.23
B (0.22, 0.38)	0.234	0	0.1431	0.1974	0.1431	0.245
P3 (0.35, 0.32)	0.215	0.1431	0	0.1581	0.2846	0.10
P4 (0.26, 0.19)	0.367	0.1974	0.1581	0	0.284	0.29
P5 (0.08, 0.41)	0.3417	0.1431	0.2846	0.284	0	0.386
P6 (0.45, 0.30	0.235	0.243	0.10	0.211	0.386	0

②

	P1	P2	(P3, P6)	P4	P5
P1	0	0.23	0.215	0.3676	0.3417
P2	0.234	0	0.1431	0.1974	0.1431
(P3, P6)	0.215	0.1431	0	0.1581	0.284
P4	0.367	0.1974	0.1581	0	0.284
P5	0.3417	0.1431	0.284	0.284	0

③	P_1	$(P_2,(P_3,P_6))$	P_4	P_5
P_1	0	0.234	0.367	0.3417
$(P_2,(P_3,P_6))$	0.234	0	0.1974	0.1431
P_4	0.367	0.1974	0	0.284
P_5	0.3417	0.1431	0.284	0

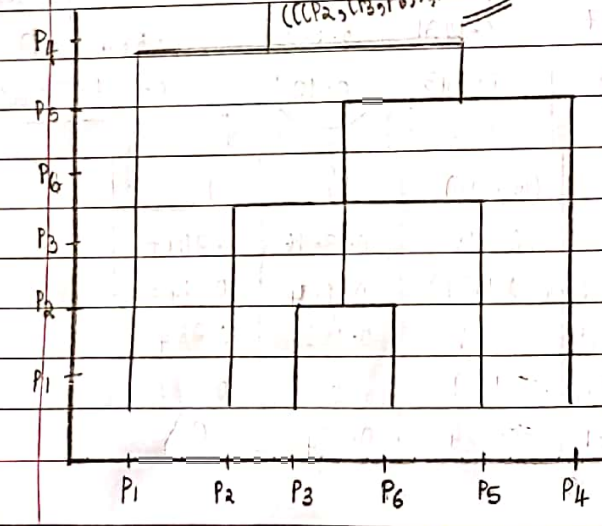
④	P_1	$((P_2,(P_3,P_6)),P_5)$	P_4
P_1	0	0.234	0.3676
$((P_2,(P_3,P_6)),P_5)$	0.234	0	0.1974
P_4	0.3676	0.1974	0

⑤

	P_1	$(((P_2,(P_3,P_6)),P_5),P_4)$
P_1	0	0.234
$(((P_2,(P_3,P_6)),P_5),P_4)$	0.234	0

(P1) and $(((P_2,(P_3,P_6)),P_5),P_4)$ are the 2 clusters

$((((P_2,(P_3,P_6)),P_5),P_4)$



- Naive Bayes Derivation

→ Bayes theorem

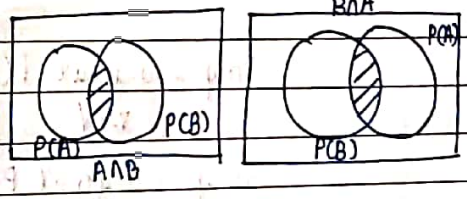
→ P(A|B): find, known/ occured.
→ more informed decision

P(A) → general. conditioned probability

⇒ Based on bayes theorm.

→ conditional probability: w the help of which we can make more informed decision (P(A|B))

⇒ general probability:



→ $P(h|D) = \frac{P(D|h)\,P(h)}{P(D)}$ — ①

$h_{map} = \underset{h \in H}{\operatorname{argmax}}\, P(h|D)$ — ②

Put ① in ②

$= \underset{h \in H}{\operatorname{argmax}} \frac{P(D|h)\,P(h)}{P(D)}$ — ③

in ③ P(D) is constant

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\ P(D|h)\,P(h) \rightarrow ④$$

where P(D) is constant

But the equation ④ cannot be used for real world examples.

⇒ ∴ modified equation is:

V_{map} = most probable target value.

Given the features $(a_1, a_2 \dots a_n)$ which describe the training example

$$V_{map} = \underset{v_j \in V}{\operatorname{argmax}}\ P(v_j | a_1, a_2 \dots a_n) \quad — ②$$

$$V_{map} = \underset{v_j \in V}{\operatorname{argmax}}\ \frac{P(a_1, a_2 \dots a_n | v_j) \cdot P(v_j)}{P(a_1, a_2 \dots a_n)} \quad — ③$$

where $P(a_1, a_2 \dots a_n)$ is constant

$$V_{map} = \underset{v_j \in V}{\operatorname{argmax}}\ P(a_1, a_2 \dots a_n | v_j) \cdot P(v_j) \quad — ④$$

$$\boxed{V_{map} = \underset{v_j \in V}{\operatorname{argmax}}\ P(v_j) \prod_i P(a_i | v_j)}$$

- Evaluation metrics: confusion matrix

Accuracy
Precision } TP.
Recall
F1-score

	Predicted No	Predicted Yes.
Actual No	TN = 45	FP = 5
Actual Yes	FN = 5	TP. = 95

① Accuracy

total Actual yes = 100

total Actual No = 50

$$\frac{\text{No. of correctly classified instances}}{\text{No. of All instances}} = \text{Accuracy}$$

$$\frac{TN + TP}{\text{all}} = \frac{140}{150} = 0.93 \times 100 = 93\%$$

② Precision

When the machine predicts yes, how often it is correct

$$\text{Precision} = \frac{\%\text{ of correctly labelled +ve instances (TP)}}{\text{All +ve labelled instances}}$$

$$= \frac{TP}{TP + FP} = \frac{95}{100} = 95\%$$

3) Recall. (also called as sensitivity or true positive rate)

Recall = % of correctly labelled +ve instances.
All +ve instances (or actual Yes).

$$= \frac{TP}{TP+FN} = \frac{95}{100} = 95\%$$

4) $\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$ = F-score.

$$= \frac{2 \times 95\% \times 95\%}{\frac{95\%}{100} + \frac{95}{100}} = \frac{2 \times 95 \times 95}{10000} \times \frac{100}{190}$$

$$= 0.95 = 95\%$$

1) find if the patient is having cough or not from the following:

P(cough) = 0.008

Report → + or − ve

P(+| cough) = (0.98)
P(−| cough) = 0.02

P(+ | noncough) = 0.03
P(−| noncough) = 0.97

2) Identify if the patient is having cough or not given report is +ve.

$$P(\text{cough} \mid +ve) = \frac{P(+ve \mid \text{cough})\, P(\text{cough})}{P(+ve)}$$ ✗ not used

$$= 0.0078$$

$$P(\text{noncough} \mid +ve) = P(+ve \mid \text{noncough}) \times P(\text{noncough})$$
$$= 0.03 \times 0.992$$
$$= 0.02976$$

Vmap = No cough.

If case: $P(\text{cough} \mid -ve) = P(-ve \mid \text{cough}) \times P(\text{cough})$
$$= 0.02 \times 0.008$$
$$= 0.00016$$

$$P(\overset{non}{\text{cough}} \mid -ve) = P(-ve \mid \text{non cough}) \times P(\text{noncough})$$
$$= 0.992 \times 0.97$$
$$= 0.96224$$

Vmap = No cough (highest value).

PCA : principal component analysis
It is a dimensionality reduction technique.
Algorithm has 5 steps:
① compute mean of all the features from the dataset having

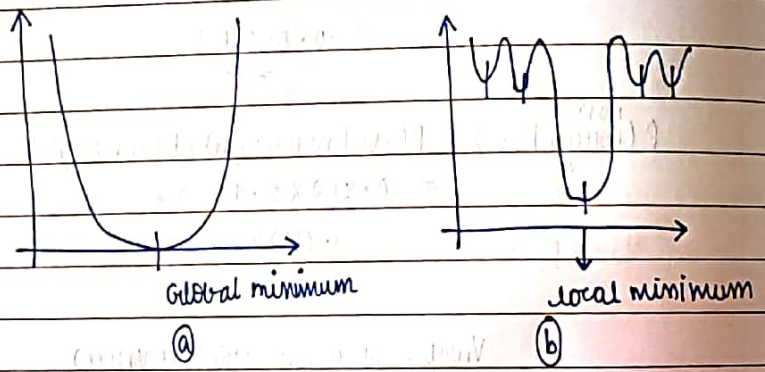
n features and N examples.

④ Calculate the covariance matrix
⑤ calculate eigen values & eigen vectors of covariance matrix
⑥ Normalize the eigen vectors.
⑦ Form new dataset by arranging eigen values in descending order. unit eigen vector corresponding to the largest eigen value = PCA

18 April • Mean squared error $[(y_i - \hat{y_i}) = \text{error})]$ (MSE)

$$\sum_{i=1}^{n} (y_i - \hat{y_i})^2 / n. \quad \text{or} \quad \sqrt{\frac{1}{n} \times \sum_{i=1}^{n} (y_i - \hat{y_i})^2}$$

⇒ RMSE → square root of MSE

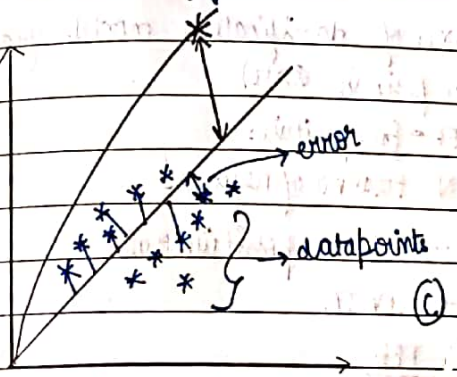


ⓐ ⓑ

mean squared error :

fig ⓐ is differentiable at each point and hence slope can be calculated at any point. It has only one global minimum, ∴ converges / convergence is faster.

Disadvantage: It can / is not robust in outlayers as shown in fig (c).



→ RMSE : standard to way to measure error.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \times \sum_{i=1}^{n} (y_i - \hat{y_i})^2}$$

only used for regression,
also used for hyper parameter tuning

- Hyper parameter : values which are initialized before the process. (High level parameter). Eg: weights.
- parameter : values which are initialized during the process

lowest RMSE = 0.

⇒ Error rate = 1 − Accuracy.
↓
how many miss classification done.

- ROC (Receiver operating characteristic curve).

→ It shows performance of classification model graphically.
X axis is FPR (False positive rate)
FPR = FP/N. FP = false positives
N = total no of negatives

Y axis is TPR is plotted (true possitive rate)
also called as SENSITIVITY.

$$TPR = \frac{TP}{P}$$ (total no of +ve egs).

→ Area under curve (AUC).
Higher the value of AUC, good is the performance of mode

AUC = 0 → Bad
= 1 → Good.

→ It is a global measure of ability of a test to determine if specific condition is present or not.