Date :

Page No.

UNIT :- 1

AI
↓
ML
↓
DL

Q.1) What is machine learning?

Field of study that gives a computer capability do learn without being explicitly programmed.

Ex:- online shopping (show related product on my previous search)

→ Machine adapts to the user based on data.

* well posed learning problem :-

An agent solves a problem or task T, performance P and gain some experience E.

If P is measured at T it can improve E.

(learning by experience)

→ Examples

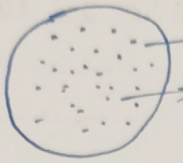
1. Playing checker Problem
2. Hand written recognition problem
3. Robo driving learning problem

Problem	Task (T)	Performance (P)	Experience (E)
1. Playing checkers learning	Playing again opponents to win game	Make perfect moves to win game	It plays itself to improve
2. Hand writing recognition learning	classifying the images & text	Better classification	A database of homework text.
3. Robot driving learning problem	drive the car in a 4 line highway	Source to destination the avg distance travelled	Image, Vehicles on Road

* Perspective of Machine learning :-

Perspective of machine learning involves searching very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by learner.

ex:



→ sorting algorithm
→ merge sort (ex- choose one algorithm, your choice)

* Issue in Machine learning :-

1. what algorithm should be used? (so many algorithms)
2. which algorithm perform best for which types of problems?
3. How much training data is sufficient? and testing data
4. what kind of method should be used to reduce learning overhead
5. what kind of method should be used?
6. For which type of data which method should be used

Designing a learning system :

To get a successful learning system, it should be designed. For a proper design, several steps should be followed for perfect & efficient

4 steps :

→ choosing the training experience
→ choosing the target function
→ choosing a representation for target function
→ choosing a
for approxim
function
→ final

Date :

Page No.

Step-1 choosing the training experience :-

In choosing a training experience, 3 attributes are taken
1. type of feedback (direct, indirect)
2. degree
3. distribution example.

1) whether the training experience provides direct or indirect feedback regarding the choices made by performance system

T → (P) → (E)
trainer
D ↓ Indi.

Direct feedback - more (Physical)
Indirect feedback (Recorded)

Ex:- checkers game, learning driving

→ You want to go with direct / Indirect. Its your choice.

2) degree do which learner will control the sequence of training

Example:- learning driving

with trainer help | with trainers partial help | completely on your own

3) How well it represent the distribution of example over which the performance of final system is measured
- more possible combinations
- more situations, more example
- learning over distributed range of example.

Example:- learning driving → trainer ← S1, S2.
→ without trainer

Step:2 Choosing the target function:-

What type of knowledge is learnt and how it is used by the performance system.

Ex:- checkers game

-> while moving diagonally set of all possible moves is called legal moves.

one move ⇒ target move.

Target function ⇒ V(b)
Board state ⇒ b
Legal move ⇒ b->B

1. b is final board state that is won then V(b) = 100
2. b is final board state that is lost, then V(b) = -100
3. b is final board state that is draw, then V(b) = 0
4. If b is not final state then V(b) = V(b').

b' ⇒ best final state.

Step:3 choosing a representation of target function

For any board state (b), we calculate function "c" as linear combination of following board features. i.e C(b)

Features

$x_1 - x_6$

x_1 = No. of black pieces on board
x_2 = No. of red pieces on board
x_3 = No. of black kings on board
x_4 = No. of red kings on board
x_5 = No. of black pieces threatened by red (black can be
x_6 = No. of red pieces threatened by black

$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6$

Representation of $\hat{V}(b)$

numerical coefficients (or) weight of each feature

additive constant

Date:
Page No.

Step 4 choosing a learning algorithm for approximating the target function:

To learn a target function (f) we need a set of training examples
(describe a particular board state (b) and training value (Vtrain(b)) for b.

ordered pair = (b, Vtrain(b))
(training example representation)

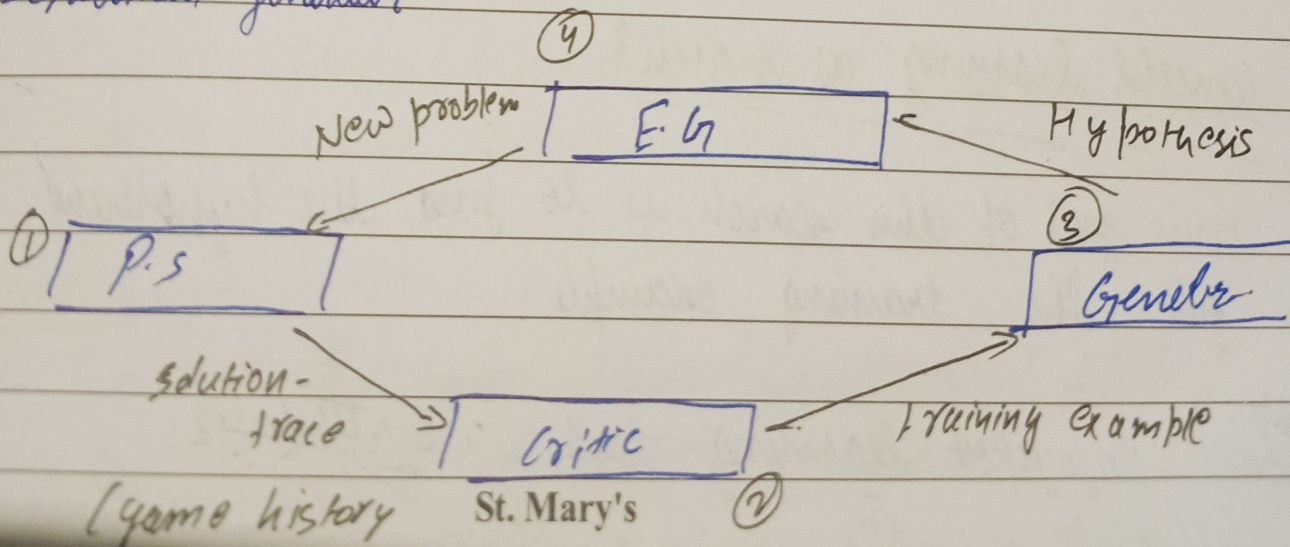
Example:- black won the game
(i.e $x_2 = 0$, which means no red
Vtrain(b) = +100
$b = (x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0)$

$< (x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0) +100 >$

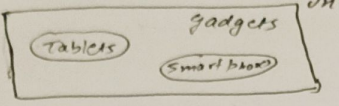
we need to do 2 steps in this phase

* Final Design

1. Performance system
2. Critic
3. Generalizer
4. Experiment generator



* Concept learning :-



* Features (Binary valued attribute)
size :- large, small → x_1
color: Black, Blue → x_2
screentype :- flat, folded → x_3
shape :- square, Rectangle → x_4

concept = $<x_1, x_2, x_3, x_4>$ →
Tablet = <large, black, flat, square>
smartphone = <small, blue, folded, rectangle>
No. of possible instances = 2^d

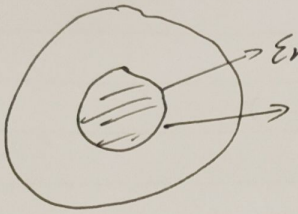
d = no. of features

Total possible concepts = 2^{2^d}

$2^4 = 16$
$2^{2^4} = 2^{16}$

<S, B, F, S>

Ex:-

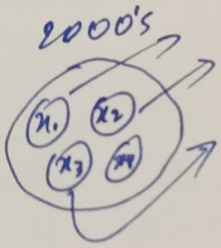


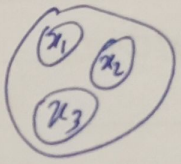
$<\phi, \phi, \phi, \phi>$ → Reject all
(most hypothesis)
specific

$<?, ?, ?, ?>$ → accept all
(most general hypothe

Main goal :- Finds all concept/hypothesis that are consistent.

Example:





$(x_1, x_2, x$
are co

* Concept learning as search!

Main goal of this search is to find the hypothesis that fits the training example.

ple :-

Enjoysport learning task :- 6 attributes

(① sky, ② Air temperature, ③ humidity, ④ wind, ⑤ water and

Date : Page No.

Sky :- 3 values — Rainy, Cloudy, Sunny
Remaining attributes have only - 2 values

∴ different instances possible = 3x2x2x2x2x2 = 96
syntactically distinct hypothesis = (5x4x4x4x4x4) = 5,120
(Additionally, 2 more values - ? and ϕ)
< ϕ, sunny, cloudy, rainy, ? >

Eg: semantically distinct hypothesis = 1 + (4x3x3x3x3x3)
= 973
(Null - taken as common only ① more value)

- After finding all the syntactically & semantically distinct hypothesis,

we search the best match from all these
(∵ much closer to our learning problem)

Finds Algorithm :-
= → ϕ (specific)
(Finding a maximally specific hypothesis)

- this algorithm considers only positive example { Yes => +ve, No => -ve }

* Representation :-
most specific hypothesis => ϕ
most general hypothesis => ?

=> Algorithm :-
step:-1) Initialise the most specific hypothesis (ϕ)
h0 = < ϕ, ϕ, ϕ, ϕ, ϕ > => 5 attributes

eg:2) for each +ve sample
for each attribute
if (value = hypothesis value) => Ignore
else Replace with the most general hypothesis ?

St. Mary's

Example:-

	origin	manufacturer	color	year	Type	class
1	JP	HO	Blue	1980	eco	+
2	JP	TO	green	1970	SPO	-
3	JP	TO	Blue	1990	eco	+
4	JP	AU	Red	1980	eco	-
5	JD	HO	White	1980	eco	+
6	JP	TO	green	1980	eco	+
7	JP	HO	Red	1980	eco	-

Solu

$h_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$

$h_1 = \langle$ 'JP', 'HO', 'Blue', '1980', 'eco' $\rangle$ { only (+) taken }

$h_2 = h_1$

$h_3 = \langle$ JP, ?, Blue, ?, eco $\rangle$

$h_4 = h_3$

$h_5 = \langle$ JP, ?, ?, ?, eco $\rangle$

$h_6 = \langle$ JP, ?, ?, ?, eco $\rangle$ → most general

* Disadvantages:-

- consider only +ve value
- h6 may not be the sole hypothesis that fits the complete

VERSION SPACE:-

Subset of hypothesis H consistent with the training examples

$VS_{H,D} = \{ h \in H \mid \text{consistent}(h, D) \}$

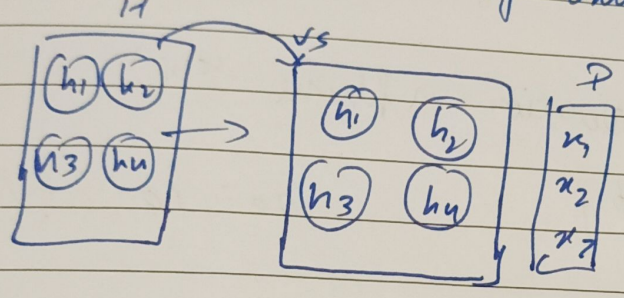
H = hypothesis

D = training example

Algorithm to obtain version space:

1. version space = list containing every hypothesis in H
2. From this step, we keep on removing inconsistent hypothesis from version space.
   for each training example, $\langle x, c(x) \rangle$ remove any hypothesis that is $h(x) \neq c(x)$
3. o/p the list of hypothesis into version space after checking for all training example.

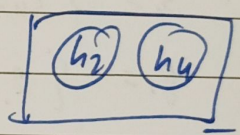
Example:



check consistency
$h(x) = c(x)$

$h_2(x_1) = c(x_1)$
$h_1(x_2) \neq c(x_2)$

$h_2(x_1) = c(x_1)$	$h_4(x_1) = c(x_1)$
$h_2(x_2) = c(x_2)$	$h_4(x_2) = c(x_2)$
$h_3(x_3) = c(x_3)$	$h_4(x_3) = c(x_3)$
$h_4(x_4) = c(x_4)$	$h_4(x_4) = c(x_4)$



are consistent.

CANDIDATE ELIMINATION ALGORITH:-

- uses the concept of version space
- consider both +ve and -ve values (yes and no)
- both specific and general hypothesis
- For positive sample, move from specific to general
- For negative sample, move from general to specific

$S = \{\phi, \phi, \phi, \phi, \phi, \phi\}$ ↑

$G = \{?, ?, ?, ?, ?, ?\}$ ↓

* Algorithm :-

1. Initialise with general and specific hypothesis (S and G)

$s = \{\phi, \phi, \phi, \phi, \ldots, \phi\}$ depends on no. of attribute

$g = \{?, ?, ?, ?, \ldots, ?\}$

2. For each example,
   if example is positive
   make specific to general.
   else example is negative
   make general to specific.

Example:- Enjoy sport (Question in phone pic)

$S_0 = \{\phi\,\phi\,\phi\,\phi\,\phi\,\phi\}$ $G_0 = \{\phi ? ? ? ? ? ? \}$

1) +ve

$s_1 = \{$'sunny', 'warm', 'normal', 'strong', 'worm', 'same'$\}$

$g_1 = \{? ? ? ? ? ?\}$

2) +ve, (specific to general)

$s_2 = \{$'sunny', 'warm', '?', 'strong', 'worm', 'same'$\}$

3) -ve (general to specific)

$s_3 = \{$'sunny', 'warm', ?, 'strong, 'worm', 'same'$\}$

$G_3 = \{$<'sunny' ? ? ? ? ? ?> <? 'warm' ? ? ? ? ?> <? ? ? ? ? 'same'>

only written change value with specific

4) $S_4 = \{$'sunny', 'warm', ? 'strong', ? ? $\}$

$G_4 = \{$< 'sunny' ? ? ? ? ?> <? 'warm' ? ? ? ?>$\}$

S_4 and G_4 final hypothesis

G_4 doesn't … same … S_4 have …

INDUCTIVE BIAS :-

→ Remarks on CE and vs algorithm :-

1) will the CE algorithm gives us correct hypothesis?
2) what training example should the learner request next?

→ Inductive learning :-

From example we derive rule

→ Deductive learning :-

Already existing rules are are applied to our examples.

* Biased hypothesis space :-

Does not consider all type of training examples

solution → include all hypothesis.

ex :- Sunny ∧ warm ∧ normal ∧ strong ∧ cool ∧ change → Yes

* Unbiased hypothesis space :-

Providing a hypothesis capable of representing set of all examples.

Possible Instance : 3V $2 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$

Target concept : 2^{96} (huge)

(Practically not possible)

* Idea of inductive bias :-

learner generalizes beyond the observed training examples to infer new examples.

'>' → inductively infered from

$x > y$ ⇒ y is inductively infered from x.

Example:

learning algorith $\Rightarrow L$

training data $D_c = \{x, c(x)\}$

New instance $= x_i$

Represented as $L(x_i, D_c)$

$(D_c \wedge x_i) > L(x_i, D_c)$

* Decision Tree learning :-

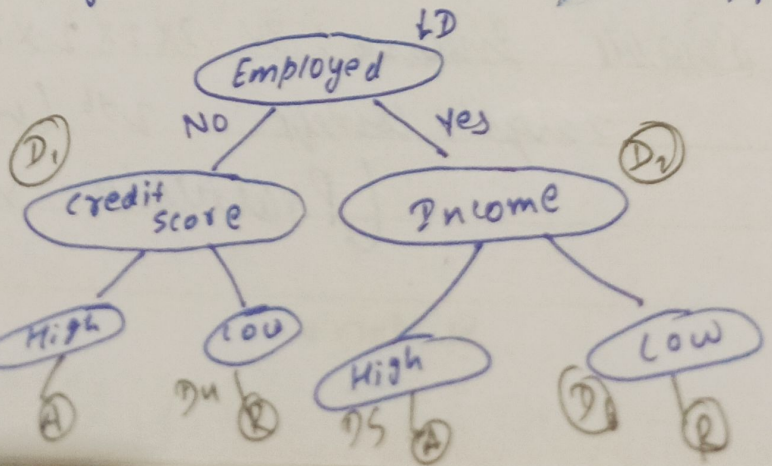
→ used in - tree structured classification & Regression

Dataset → Algorithm → classifies the data (decision tree algorithm)

2 types of node
① Decision node
② leaf node

Example :- loan system :-

(decides if the loan should be approved / Rejected)



Date :

Page No.

→ Algorithm (ID3)

1. In the given dataset choose the target attribute
2. calculate information gain of target attribute

$$IE_1 = \frac{-P}{P+N} \log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log_2\left(\frac{N}{P+N}\right)$$

3. for remaining attributes, find entropy

Entropy = IE, X Probability.

$$E(A) = \sum \frac{P_i + N_i}{P+N} I(P_i N_i)$$

4. calculate Gain = $IE_1 - E(A)$

Example:-

step 1 Target attribute = Profit.

step 2:- Information gain,

$$IE_1 = \frac{-P}{P+N} \log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log_2\left(\frac{N}{P+N}\right)$$

=> P = count (down) = 5

N = count (up) = 5

$$= \frac{-5}{10} \log_2\left(\frac{5}{10}\right) - \frac{5}{10} \log_2\left(\frac{5}{10}\right)$$

$$= -\left(\frac{1}{2} \log_2(2^{-1}) + \frac{1}{2} \log(2^{-1})\right)$$

$$= -\left(\frac{1}{2} \times -1 \log_2 2 + \frac{1}{2} \times -1 \log_2 2\right) = -\left(\frac{1}{2} \times -1 + \frac{1}{2} \times -1\right)$$

$$= -(-1) = 1$$

$IG_1 = 1$

③ Calculate entropy for remaining attributes.

$$E(A) = \sum \frac{P_i + N_i}{P+N} I(P_i, N_i) \quad [\text{IE}_i \times \text{Probability}]$$

Age

(i) Prepare a table for each attribute
rows - values of undertaken attribute (old, mid, new)
columns - values of target attribute (down, up)

	down	up
old	3	0
mid	2	2
new	0	3

Entropy = IE, x Probability

P = down count
N = up count.

$$IE_1(\text{old}) = -\left(\frac{3}{3}\log\left(\frac{3}{3}\right) + \frac{0}{3}\log\left(\frac{0}{3}\right)\right) = 0$$

Probability = $\frac{3}{10}$

Entropy (old) = $0 \times \frac{3}{10} = 0$

$$I\,E_1(\text{mid}) = -\left(\frac{2}{4}\log\left(\frac{2}{4}\right) + \frac{2}{4}\log\left(\frac{2}{4}\right)\right]$$

$$= -\left(\frac{1}{2}\log\left(\frac{1}{2}\right) + \frac{1}{2}\log\left(\frac{1}{2}\right)\right]$$

$$= -\left(-\frac{1}{2} + -\frac{1}{2}\right) = -(-1) = 1$$

Probability = $\frac{4}{10}$

Entropy = $1 \times \frac{4}{10} = 0.4$.

$$IE_1(\text{new}) = \left(-\frac{0}{3}\log\left(\frac{0}{3}\right) + \frac{3}{3}\log\left(\frac{3}{3}\right)\right] = 0$$

Probability = 3/10

Entropy = $0 \times \frac{3}{10} = 0$

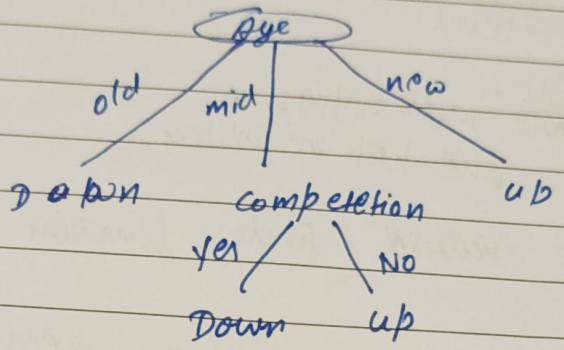
Entropy (Age) = E(0) + E(m) + E(N) = 0 + 0.4 + 0 = 0.4

Gain = $IE_1 - E(A) = 1 - 0.4 = 0.6$

Page No.

In the same way, calculate gain of other attributes

Gain (competition) = 0.149
Gain (type) = 0
Gain (Age) = 0.6
Highest gain → root node
(Age)



old → all down
mid → Some down and some up
new → all up
why competition → next highest gain
why not other attribute (Type)

Type ⇒ gain = 0 and even in DT also, no requirement with it

* Appropriate problem for decision tree learning:

DT algorithm is best suited to problems with the following characteristics

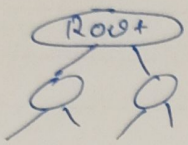
1. Instances are represented by attribute value pairs
2. The target function has discrete o/p values.
3. Training data can have errors.
4. May contain missing attribute value also

Hypothesis space in decision tree learning :-

ID_3 can be characterized as searching a space of hypotheses for one that fits the training examples.

ID_3 -> will search set of possible decision tree from available hypothesis

ID_3 perform simple to complex searching.



First start with empty tree and keep on adding.

- every discrete valued (finite) function can be described by some decision tree
- avoid major risk of searching incomplete hypotheses
- has only single current hypothesis.
- Backtracking is not possible
- can be extended easily to noisy data also.

* Inductive bias in Decision tree Algorithm:-

- set of assumptions

Inductive bias of ID_3 consists of describing the basis by which ID_3 chooses one consistent decision tree over all the possible DT's

ID_3 search strategy:

1. select in favour of shorter trees over longer ones.
2. selects element with highest IG, as root attribute over lower IG ones

* Types of inductive bias:

1. Restrictive bias - based on conditions.
2. ~~Rest~~ ~~Pref~~ Preference bias - based on priorities

ID3 ⇒ ~~Rest~~ Preferen
VS and CE ⇒ Restrictive

Why short hypothesis?
Prefer simpler hypothesis that fits the data

Issues :-

1. overfitting the data
2. Incorporating continuous → discrete valued algorithm
3. Determining depth of tree for correct classification
4. Handling attributes with different costs.
5. Alternative measures for selecting attributes.

UNIT-II

Introduction to artificial Neural Networks:- (ANN)

Neural functions are done artificially.

Example:-

When we touch a hot vessel

-> A node has two parts

1. Summation:

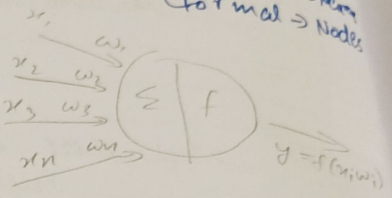
calculate the weighted sum of all the i/p

$= x_1w_1 + x_2w_2 + x_3w_3 + \dots + x_nw_n$

$= \Sigma x_i w_i$

Human brain - millions of neuron

{ Biological -> Neuron
Formal -> Nodes



once weighted sum is calculated
sent to activation function.

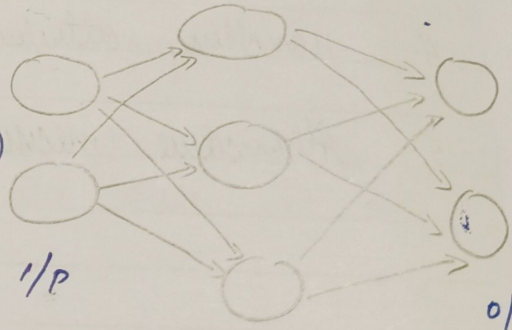
2. Activation:

generate the output based on (I/P) function / given
$\Sigma x_i w_i$

* Representation of Artificial Neural Networks:

ANN are divided into 3 parts

1. Input layer
2. Hidden layer
3. Output layer. (receive)



I/P

Hidden
(Important data taken from i/p & Process)

O/P
(Producing & Providing)

* Appropriate problem for Neural N/w learning:

Instances have many attribute value pairs

Target function has discrete values, continious values or combination of both.

Training example with error or missing values.

long training times are acceptable

Fast evaluation of the target function learnt.

Ability for humans to learn the target function learnt by machine is not important.

* PERCEPTRON :-

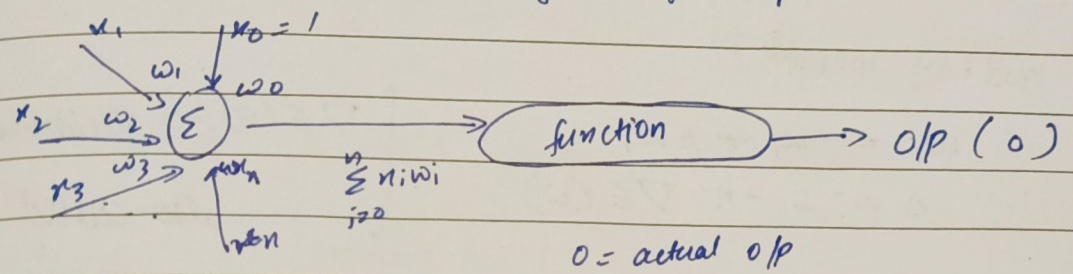
→ basic unit to build ANN

- takes real valued i/p, calculate linear combination of these inputs and generate output.

output = 1 if result > threshold
-1 otherwise

$$o(x_1, x_2 \dots x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + w_2 x_2 + \dots + w_n x_n > 0 \\ -1 & \text{otherwise} \end{cases}$$

$w_1, w_2 \dots w_n \rightarrow$ weights of inputs



o = actual o/p
t = target o/p

if actual = target ⇒ weights are fixed
otherwise ⇒ changed

How to change weights ?-
Initially, random weights

Formula to change weights :

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta (t - o) x_i$$

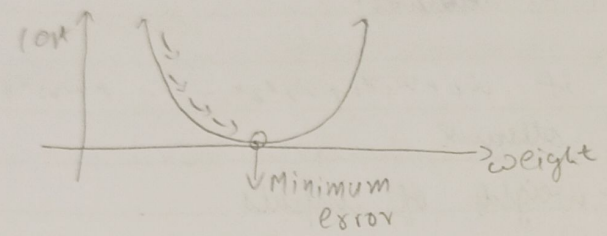
η = learning rate (0-1) (0-2)
t = target o/p
o = actual o/p
x_i = i/p associated with w_i

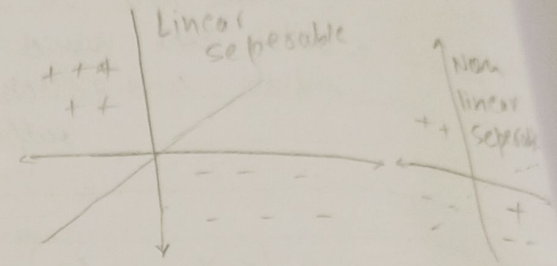
Decode
Q-14

* Gradient Descent And the Delta Rule:-

Delta Rule → used for non linearly seperable data
Perceptron training rule → works goods only for linearly seperable.

Main Idea of Delta Rule :-
→ uses gradient descent rule and finds out the best weight. → (Reduce) ↓





How to modify weights?

$$w_i \leftarrow w_i + \Delta w_i$$
$$\Delta w_i = -\eta \nabla E(w)$$

$\nabla E(w)$ → derivative of error w.r.t weights.
also called as gradient.

$$\nabla E(w) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n}\right]$$

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\left(\frac{1}{2}\sum_{d \in D}(t_d - o_d)^2\right) \Rightarrow \text{Formula to find error.}$$

$$E = \frac{1}{2}\sum_{d \in D}(t_d - o_d)^2$$

$$\frac{\partial E}{\partial w_i} = \frac{1}{2}\frac{\partial}{\partial w_i}\sum (t_d - o_d)^2$$
$$= \sum \frac{1}{2} \times 2 \times (t_d - o_d)' \frac{\partial}{\partial w_i}(t_d - o_d)$$
$$= \sum (t_d - o_d)\frac{\partial \cdot}{\partial w_i}(t_d - w_d x_d)$$

$$\nabla E(w) = \sum (t_d - o_d)(-x_d)$$
$$\therefore \Delta w_i = -\eta \times \sum (t_d - o_d)(-x_d)$$
$$\Delta w_i = \eta \sum (t_d - o_d) x_d$$
$$w_i \leftarrow w_i + \Delta w_i$$

$\frac{d}{dx}(a^n) = n \cdot a^{n-1}$

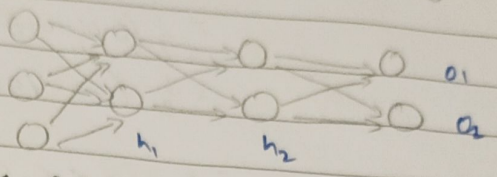
$o_d = \sum w_i x_i$
$w_0 x_0 + w_1 x_1 + \ldots$
$o_i = x_i w_i$
$o_d = x_d w_d$

$\therefore \frac{\partial}{\partial w}(w_d x_d) = x_d \frac{\partial}{\partial w}(w)$

Page No.

Multi layer Neural Networks:-

-> non linearly seperable data



BACK PROPAGATION ALGORITHM

(backward propagation of error) [go back and correct the errors]

when error occurs, we go in backward direction

i.e output → hidden layer → input layer.

Part 1:- calculate forward propagation error

question in phone pic 3

b_1 = bias b/w I/P → h

b_2 = bias b/w h → op

1) Calculate h_1 (in and out)

$h_1(in) = x_1 w_1 + x_2 w_2 + b_1$

$= 0.15 \times 0.25 + 0.2 \times 0.1 + 0.35$

$= 0.377$

$h_1(out) = 1/(1+e^{-h_1(in)})$

$= \frac{1}{1+e^{-(0.377)}} = 0.5932$

2) Calculate h_2 (in and out)

$h_2(in) = x_1 w_3 + x_2 w_4 + b_1$

$= 0.3925$

$h_2(out) = 0.5968$

3) Calculate o_1 (in and out)

$o_1(in) = h_1(out) w_5 + h_2(out) w_6 + b_2$

$= 1.105$

$o_1(out) = 1/(1+e^{-o_1(in)})$

$= 0.7513$

4) Calculate o_2 (in and out)

$o_2(in) = h_1(out) w_7 + h_2(out) w_8 + b_2$

$= 1.22484$

$o_2(out) = 1/(1+e^{-o_2(in)})$

$= 0.7729$

5) Calculate E_{Total} :-

$$E_{Total} = \sum \frac{1}{2}(target - o/p)^2$$

$$= E_{o1} + E_{o2} \quad \text{(error at } o_1 \& o_2\text{)}$$

$$= \frac{1}{2}(0.01 - 0.7513)^2 + \frac{1}{2}(0.99 - 0.7729)^2$$

$$= \frac{1}{2}(-0.7413)^2 + \frac{1}{2}(0.2171)^2$$

$$= 0.274 + 0.0235 = 0.29832 \text{ (approx)}$$

Part 2 :- Calculate backward propagation error (output layer → hidden layer)

w_5, w_6, w_7 and w_8

$\eta = 0.6$ learning rate

First let us adjust w_5

$$w_5^* = w_5 - \eta \frac{\partial E_{total}}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out\ o_1} \times \frac{\partial out\ o_1}{\partial net\ o_1} \times \frac{\partial net\ o_1}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial out\ o_1} = \underset{(o_1(out))}{out\ o_1} - \underset{(o_1 given)}{target\ o_1} = 0.751365 - 0.01 = 0.7413565$$

$$\frac{\partial out\ o_1}{\partial net\ o_1} = out\ o_1 * (1 - out\ o_1) = 0.751365(1 - 0.751365) = 0.186$$

$$\frac{\partial net\ o_1}{\partial w_5} = \underset{(h_1(out))}{out\ h_1} = 0.59326992$$

$$\frac{\partial E_{total}}{\partial w_5} = \ldots \times \ldots \times \ldots = 0.08216704$$

$$w_5^* = w_5 - \eta \frac{\partial E_{total}}{\partial w_5}$$

$$= 0.4 - 0.6 \times 0.08216704 = 0.35069972$$

Part 3: calculating backward propagation of error.
(hidden - input layer)

(w_1, w_2, w_3, w_4)

$\eta = 0.6$

just lets adjust w_1

$$w_1^* = w_1 - \eta \frac{\partial E_{total}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h_1}} \times \frac{\partial out_{h_1}}{\partial net_{h_1}} \times \frac{\partial net_{h_1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h_1}} = \frac{\partial E_{o1}}{\partial out_{h_1}} + \frac{\partial E_{o2}}{\partial out_{h_1}} \Rightarrow 0.036350306$$

$$\frac{\partial E_{o2}}{\partial net_{o_2}} \times \frac{\partial net_{o_2}}{\partial out_{h_1}} \Rightarrow \frac{\partial E_{o_2}}{\partial out_{o_2}} \times \frac{\partial out_{o_2}}{\partial net_{o_2}}$$

$$\frac{\partial out_{o_2}}{\partial net_{o_2}} = out_{o_2}(1 - out_{o_2})$$

$$= 0.7729(1 - 0.7729)$$

$$= 0.175510052$$

$$\frac{\partial E_{o_2}}{\partial out_{o_2}} = (out_{o_2} - target\ o_2)$$

$$= o_2 out - o_2 given$$

$$= 0.771292465 - 0.99$$

$$= 0.217071535$$

$$\frac{\partial E_{o_2}}{\partial out_{h_1}} = 0.217071535 \times 0.175500052 = \ldots$$

same process for $\frac{\partial E_{o1}}{\partial out_{h_1}}$

Other route:

$\frac{\partial net_{o_1}}{\partial out_{h_1}} = out_{o_2}$ from $h_1 \to w_5$

$= 0.50$

also use this

$$\frac{\partial out_{h_1}}{\partial net_{h_1}} = out(h_1)(1 - out\ h_1)$$

$$= 0.241300709$$

$$\frac{\partial net_{h_1}}{\partial w_1} = \frac{\partial}{\partial w_1}(w_1 x_1 + w_2 x_2 + b_1) = x_1 = 0.05$$

$$\frac{\partial E_{total}}{\partial w_1} = 0.036350306 \times 0.241300709 \times 0.05$$

$$= 0.00438568$$

$$w_1^* = w_1 - \eta \frac{\partial E_{total}}{\partial w_1} = 0.15 - 0.6 \times 0.00438568$$

$$= 0.149736859 2.$$

* Remarks on Back Propagation Algorithm :-

5 Remarks

1. Convergence & local minima (error back-propag local)
   ↳ interconnected to each other
2. Representational power of feed forward networks. (depth, nodes)
3. Hypothesis space search and inductive bias
4. hidden layer representation.
5. generalisation, overfitting and stopping criterion.
   (when to stop algorithm)
   generalise the error and overfit

Function :- Boolean, Continuous, Arbitrary

* AN ILLUSTRATIVE EXAMPLE: FACE RECOGNITION:

→ Face Recognition is a category of biometric software that maps individuals facial features and stores the data as a face print.

→ Machine learning and image processing are backbones of FR.

* Image Processing :-
  Involves the process of computer vision

→ Image Reading :- Read any image between the range of 0-255
  For any image, there are 3 colors - RGB

- open cv :- Python library to solve computer vision

* Machine learning :-

takes data as input and learn from that data
- Identifies the pattern
- ML does 3 major function in Face Recognition
  1. Deriving the feature vector
  2. Matching algorithm
  3. Face Recognition operation → 1) Face detection
     2) Face Analysis
     3) Image to data conversion
     4) Match finding.

* Face Recognition S/W :-

- Deepvision AI, Sense Time, Amazon Recognition, Facefirst, True face, Cognitec - -

Application :- Hospital security, Airline industry etc.

Advanced Topic in Neural networks.
3 advanced topics

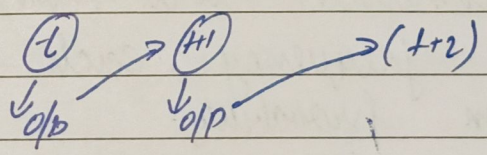
1) Alternative error functions :-

adding an extra penalty term to $E(\vec{w})$ can reduce the problem of overfitting.
- each weight is multiplied with a constant in each iteration
∴ we can redefine E as.

$$E(\vec{w}) = \frac{1}{2} \sum_{d \in D} \sum_{k \in O/P} (t_{kd} - o_{kd})^2 + \gamma \sum w_{ji}^2$$

2) Alternative error minimization procedures :-
2 procedures
1. line search 2. conjugate gradient descent method

3) Recurrent Neural N/w (RNN) :-
ANN that can be applied to time series data and uses the o/p of N/w units at time t as input to another unit at time $t+1$.



* Evaluating the hypothesis :-

Evaluating the accuracy of hypothesis is of basis of ML

For evaluating the accuracy of hypothesis, we mainly focus on 3 questions.

1) hypothesis is accurate over limited sample of data, then what about additional data.
2) we have 2 hypothesis - one is better than other when used on sample of data.
   Then will better hypothesis work good for general data also!
3) when we have limited data, what is the best way to use this data for both learning and estimating hypothesis

* Motivation :-

sometime, very precise hypothesis is required. In that case estimating the accuracy is very important.

Example:- Medical treatment.

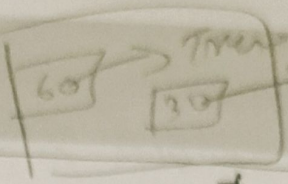
* Estimating hypothesis accuracy :-

Let us make some assumptions,

-> There is some space of possible instances X over which various target functions may be defined.
-> different instances will have different frequencies.
-> based on frequency, each instances in X will have some unknown probability.
-> Trainer will teach the machine about the training examples of target function. $f(x) = x^2$.
   Send a particular instances (x_i) along with target value $f(x)$
   Target function : $f : X \rightarrow \{0, 1\}$

Classifier each instances into diff. categories based on req.
Now, after classification, what is the probable error in this estimation we made.

Error 2 types ① Sample error ② True error



Sample Error

→ The error rate of hypothesis over a sample of data

Sample error = $error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x))$

$\delta(f(x), h(x)) = 1$ if $f(x) \neq h(x)$

$= 0$ otherwise

True Error

→ The error rate over of hypothesis over entire distribution D.

True error = $error_D(h)$

$= P[f(x) \neq h(x)]$

Basic of Sampling Theory :- (11 definitions)

1) Random variable :- Name of an experiment with probabilistic outcome.

ex :- $P(X=H) = 0.5$ $P(X=T) = 0.5$ $y_i \rightarrow$ random variable

2) Probability distribution :- Probability distribution for a random variable Y $P(Y=y_i)$ that Y will take value y_i for each possible y_i.

ex :- when two coins are tossed, let Y represent no. of heads

$P(Y \leq 1) = P(Y=0) + P(Y=1) = 0.25 + 0.5 = 0.75$

3) ~~Ex~~ Expected value (or) mean

$E(Y) = \sum y_i P(Y=y_i)$

4) Variance :- shows dispersion of data from its mean.

$Var(Y) = E[(Y - \mu_X)^2]$

Random variable — Mean

5) Standard deviation :-

$= \sqrt{\text{Variance}}$

$= \sqrt{Var(Y)}$

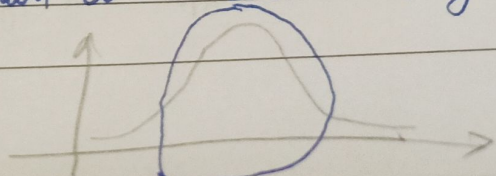
6) Binomial distribution :- ${}^nC_r$ ${}^nC_r$

gives probability of observing r heads in a series of n independent coin tosses

ex :- finding prob. of 2 heads in 3 tosses

7) Normal distribution :-

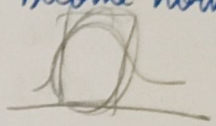
Bell shaped probability distribution that covers many continuous values.



8) Central limit theorem :-

If you are having population with mean μ and SD as σ, and you take large random samples from that population with replacement, then the distribution will approximately become normal.

example :- 80, 40, 60 ⇒ 60 (50)



9) Estimator

It is a random variable Y used to estimate some parameter of an underlying population. ex :- $P(Y = 50)$

10) Estimation Bias :- Sample mean $= \mu_y$

population mean $= \mu_p$

$EB = \mu_y - \mu_p$

11) Confidence Interval

Probability (P) at N%.

(Area under normal distribution curve in which data will fall 95% of the times)

* Difference in error of two hypothesis :-

* Hypothesis testing :

we calculate 2 mutually exclusive statements

true or true

↓ on population

↓ using sample data

and decide upon population belongs to which statement.

* Steps in hypothesis testing :-

1) make initial assumption (H_0)

2) collect data

3) Gather evidences to accept or reject null hypothesis (we should also assume an alternate hypothesis - H_1)

-> 2 errors are possible.

(Type 1 and Type 2 error)

→ Type 1 error
we implicitly know that h_0 is correct but we do not have enough evidence to prove that then, automatically, h_1 is taken as correct

→ Type 2 error
when we do not have enough evidence to prove the truth of h_0, without any choice we approve h_1.
(we do not know about truth of h_0 & h_1.)

	h_0	h_1
do not reject	OK	Type 2 error
Reject	Type 1 error	OK

Comparing Learning algorithm :-

we have so many algorithm in ML
there is always a confusion about which one to choose.
∴ we consider some factors.

1) time complexity — training & testing
2) space complexity — less space
3) sample complexity — no. of examples to train the m
4) unbiased data — not learn about everything 60-30 / 30
5) online and offline algorithm — online → new data automatically updated
offline → obtain the entire machine
6) Parallelizability → { parallel algorithm → complete multiple algorithm at once
Ex :- KNN
7) Parametricity { Parametric - fixed data
Non parametric - changing data.

UNIT: III

* BAYES THEOREM:-

Bayes theorem gives the probability of an event based on prior knowledge of conditions.

$$P(A/B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$

- $P(B/A)$ → Likelihood
- $P(B)$ → Marginal
- $P(A)$ → Prior
- $P(A/B)$ → hypothesis

Proof:- Let us take 2 events: A and B

$P(A/B)$ - Conditional Prob.

$$P(A/B) = \frac{P(A \cap B)}{P(B)}$$

$$P(B/A) = \frac{P(B \cap A)}{P(A)}$$

$$P(A/B) \cdot P(B) = P(A \cap B) \quad -(1)$$

$$P(B/A)\ P(A) = P(B \cap A) \quad -(2)$$

from (1) and (2)

$$P(A/B)\ P(B) = P(B/A)\ P(A)$$

$$P(A/B) = \frac{P(B/A)\ P(A)}{P(B)} \rightarrow \text{Bayes th.}$$

Terms explanation:-

A = hypothesis — $P(A/B)$ = Prob. of hypothesis before observing given data

B = given data — $P(B)$ = Prob. of given data

$P(A/B)$ = finding prob. of hypothesis when probability of training examples are given.

$P(B/A)$ = finding prob. of given data provided with prob. of hypothesis that is true.

Q:- $P(\text{king}/\text{Face})$ → Prob of face card when it king → as king is face card.

$$= \frac{P(\text{Face}/\text{king}) \cdot P(\text{king})}{P(\text{Face})}$$

Face - J, K, Q

$4 \times 3 = 12$

$$= \frac{\frac{4}{4} \times \frac{4}{52}}{\frac{12}{52}} = \frac{1 \times 1/13}{3/13} = \frac{1}{3}$$

BAYES THEOREM AND CONCEPT LEARNING :-

What is the relation between bayes theorem and concept learning ?

→ Bayes theorem calculates the probability of each possible hypothesis and outputs the most probable one.

Brute force Bayes concept learning.

1) For each hypothesis in H calculate posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \quad -①$$

2) output the hypothesis h_{MAP} (h_{ml}) - maximum likelihood with highest probability

$$h_{MAP} = \underset{h \in H}{\arg\max}\ P(h|D)$$

↳ we need values P(h) and P(D|h)

Some assumptions:

1) Training data D is noise free
2) Target concept c is present in hypothesis space H
3) we have no prior reason to believe that any hypothesis is more probable than any other.

$$P(h) = \frac{1}{|H|} \text{ for all } h \text{ in } H$$

$$P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 & \text{otherwise} \end{cases}$$

From ①

Case 1 : Hypothesis h is inconsistent i.e $P(D|h) = 0$

$$P(h|D) = \frac{0 \times P(h)}{P(D)} = 0 \qquad d_i \neq h(x_i)$$

Case 2 : Hypothesis h is consistent i.e $P(D|h) = 1$

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} = \frac{1 \times P(h)}{P(D)} = \frac{\frac{1}{|H|}}{|VS_{H,D}|} = \frac{1}{|VS_{H,D}|}$$

$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$

* MAXIMUM LIKELIHOOD AND LEAST SQUARED ERROR HYPOTHESIS:

D → set of target values

To find the maximum likelihood hypothesis in bayesian learning.

$$h_{ML}\ (h_{MAP}) = \underset{h \in H}{\operatorname{argmax}}\ p(h/D)$$ → { is the one which has the least of the sum squared error b/w the hypothesis }

Let us take training instances $(x_1, x_2, \ldots x_m)$ and coincides target values $D = (d_1, d_2, \ldots d_m)$

∴ we write $P(D/h)$ as product of $P(d_i/h)$

$$h_{MAP} = \underset{h}{\operatorname{argmax}} \prod_{i=1}^{m} P(d_i/h)$$

π - product

Σ - summation $\sum_{i=1}^{n} x_i$

By assuming ~~gaussian~~ ~~formula~~ normal distribution

$$f(x/\mu) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \rightarrow f(d_i/h) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(d_i - h(x_i))^2}{2\sigma^2}}$$

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-1}{2\sigma^2}(d_i - h(x_i))^2}$$

→ use $\ln$ → logarithm function

$$= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} \frac{1}{2\sigma^2} (d_i - h(x_i))^2$$

$\ln \frac{1}{\sqrt{2\pi\sigma^2}}$ → constant term (so delete this)

$$= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} - \frac{1}{2\sigma^2} (d_i - h(x_i))^2$$

(added - since the term is -ve to min)

$$= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{1}{2\sigma^2} (d_i - h(x_i))^2$$

(constant the delete)

$$= \underset{h}{\operatorname{argmin}} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

$$\therefore h_{MAP} \text{ or } h_{ML} = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

→ least squared error.

* Minimum DESCRIPTION LENGTH PRINCIPLE!-

Assumption :-

Representing a concept in minimum possible way
- Then it is said to be good one

Mathematically

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

Applying algorithm:

$$\arg\max_{h \in H} \log P(D/h) + \log P(h) \qquad \log(ab) = \log a + \log b$$

$$\arg\min_{h \in H} -\log P(D/h) - \log P(h)$$

(minimum length / short hypothesis is preferred)

Example:-

1) let us consider a probability of designing a code to transmit message drawn at random from a set D.
where probability of drawing an i^{th} msg $= P_i$
2) while transmitting, we want a code that minimize the expected no. of bits

To do this, we should assign shorter codes to the most probable we represent the length of message i w.r.t i as $L_C(i)$

↳ increases

$\therefore h_{MAP} = \arg\min L_{CH}(h) + L_{CD/h}(D/h)$

L_H -> optimal encoding for H

$L_{D/h}$ -> " " " D given h

$\therefore h_{MDL} = h_{MAP}$

Bayes OPTIMAL CLASSIFIER :-

Bayes optimal classifier is a probabilistic model that makes the most probable prediction for a new example.

$$P(A/B) = \frac{P(B/A)\,P(A)}{P(B)}$$

for a dataset,

$$x = \{x_1, x_2, x_3 \ldots x_n\} \{y\}$$

$$P(y/(x_1, x_2, \ldots x_n)) = \frac{[P(x_1/y) \cdot P(x_2/y) \ldots P(x_n/y)] \times P(y)}{P(x_1) \cdot P(x_2) \ldots P(x_n)}$$

$= P(y) \prod_{i=1}^{n} P(x_i/y)$

$P(x_1)P(x_2)\cdots P(x_n)$ → Probability of this value will constant, so it will delete from

$\Rightarrow P(y) \prod_{i=1}^{n} P(x_i/y)$

Ques

Play

Yes	9	9/14
NO	5	5/14
Total	14	100%

Total (sunny, hot)

Pic-9.5

$P(\text{Yes}/\text{sunny, hot}) = P(\text{Sunny}/\text{yes}) \times P(\text{hot}/\text{yes}) \times P(y$

$= \frac{2}{9} \times \frac{2}{9} \times \frac{9}{14} = 0.031$

$P(\text{NO}/\text{sunny, hot}) = P(\text{Sunny}/\text{NO}) \times P(\text{hot}/\text{No}) \times P(\text{No})$

$= \frac{3}{5} \times \frac{2}{5} \times \frac{5}{14} = 0.08571$

Total $= 0.031 + 0.08571 = 0.27$

$P(\text{Yes}) = \frac{0.031}{0.27} = 0.114$

$P(\text{NO}) = \frac{0.08571}{0.27} = 0.317$

} Probability of No is more. ∴ Player will not enjoy play.

Gibs algorithm :-

(Similar to Bayes theorem)

1) choose one hypothesis at random, according to $P(h/D)$
2) use this to classify new instance.

$2\,[\text{error gibs}] \leq 2E\,[\text{error Bayes optimal}]$

NAIVE BAYES CLASSIFIER :-

→ classification technique based on Bayes theorem with an assumption of independence among features

$P(A/B) = \frac{P(B/A)\,P(A)}{P(B)}$

Pic-6

fruit {yellow / Sweet, long

Ques Pic-6

$P(\text{Yellow}/\text{orange}) = \frac{P(\text{orange}/\text{Yellow})\,P(\text{Yellow})}{P(\text{orange})}$

$= \frac{\frac{350}{800} \times \frac{800}{1200}}{\frac{650}{1200}} = 0.53$

$P(\text{sweet} / \text{orange}) = \frac{P(O/S)\,P(S)}{P(O)} = 0.69$

$P(\text{Fruit} / \text{orange}) = P(Y/O) \times P(S/O) \times P(L/O)$

$= 0.53 \times 0.69 \times 0 = 0$

$P(\text{Fruit} / \text{Banana}) = P(Y/B) \times P(S/B) \times P(L/B)$

$= 1 \times 0.75 \times 0.89 = 0.65 > 0.65$ → more probable

$P(\text{Fruit} / \text{other}) = P(Y/O) \times P(S/O) \times P(L/O)$

$= 0.33 \times 0.66 \times 0.33 = 0.072$

∴ Banana is the yellow, long, sweet, and belong to the most probable one.

* BAYESIAN BELIER NETWORKS :-

- 2 important concepts
  (1) directed acyclic graph (DAG)
  (2) conditional probability Table (CPT)

* Directed acyclic graph.

RAIN → Node, Rain
↓ Dog bark / Rain
Dog bark
↓ Cat hide / Dog bark
Cat hide

* CPT (calculate always respect to parent node)

	R	~R
B	9/48	18/48
~B	3/48	18/48

$(D=T \& R=T) = 9/48 = 0.19$

$(B=T \& D=F) = 18/48 = 0.375$

$(B=F \& R=T) = 3/48 = 0.06$

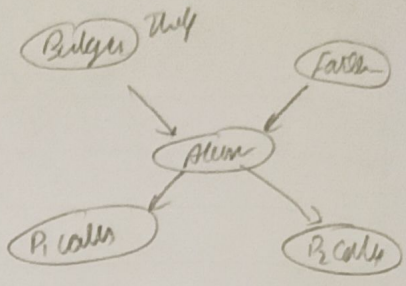
$(B=F \& R=F) = 18/48 = 0.375$

→ Bayesian belief n/w is a probabilistic graphical model (PGM) that represents conditional dependencies between random variable through DAG.
→ also suitable for representing probabilistic relation between multiple events (more than 2 events)

Given Probabilities are.

$P(B=T) = 0.001$ $P(E=T) = 0.002$

$P(B=F) = 0.999$ $P(E=T) = 0.998$



Probability of alarm:- (2 parents)

B	E	$P(A=T)$	$P(A=F)$
T	T	0.95	0.05
T	F	0.99	0.06
F	T	0.29	0.71
F	F	0.001	0.999

Probability of P_1 (1 parent)

A	$P(P_1=T)$	$P(P_1=F)$
T	0.90	0.10
F	0.05	0.95

Probability of P_2

A	$P(P_2=T)$	$P(P_2=F)$
T	0.70	0.30
F	0.01	0.99

Example

Find the probability of P_1 is T, P_2 is T, A is T, B is F and E is F

i.e $P(P_1, P_2, A, \sim B, \sim E)$

$= P(P_1/A)\, P(P_2/A)\, P(A/\sim B, \sim E)\, P(\sim B) \cdot P(\sim E)$

$= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 = 0.00062$

EM ALGORITHM: (Expectation - Maximisation)

-> used to find latent variable ——> not directly observed

-> basic for many unsupervised clustering algorithm

* steps involved in EM algorithm:

1. Initially, a set of initial values are considered
A set of incomplete data is given to system
2. Next step - expectation step - E step
Here, we use observed data to estimate or guess the values of miss / incomp data
3. Maximization step or M-step
Here, we use the complete data generated in preceding E-step to update the values.
4. we check if values are converging / not
If converging — stop. otherwise, Repeat step 2 and 3 till the convergence occurs

Date :

Page No.

* usage	+ Advantages	* Disadvantages
1) used to fill missing data 2) used for unsupervised clustering 3) used to discovers values of related variables	+ With each iteration, likelihood increases => E-step and M-step are easy to implement	1) slow convergence 2) may converge to local optimal only.

Instance Based Learning :-

memories and then apply (not use brain)

-> Instead of performing explicit generalization, it compares new problems with instance in training, which are stored in memory.

Example: spam mails — (spam filter all spam mails goes to spam with checking)

-> also called as memory based learning / lazy learning

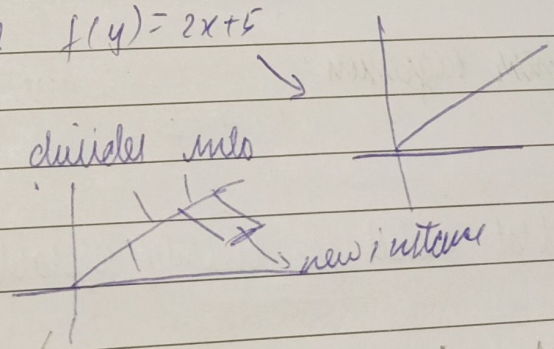
-> done with 3 different approaches.

Instance based learning

- lazy learners (KNN)
- Radial basis function (RBF)
- Case based Reasoning (CBR)

Example:- Initially, we have $f(y) = 2x+5$

If we have $f(y) = 2x^2+3x$ and divides into pockets



↳ this graph will guide by previous memory.

K-Nearest Neighbour Algorithm (KNN):- (example for lazy learn

Example:- given data query => x = (Maths = 6, CS = 8)
and k = 3 → nearest neighbour
classification - Pass / fail

	Maths	CS	Result
1)	4	3	F
2)	6	7	P ✓
3)	7	8	P ✓
4)	5	5	F
5)	9	9	P ✓

Euclidean distance (d)

$$d = \sqrt{|x_0 - x_{A1}|^2 + |x_{c_2} - x_{A_2}|^2}$$

o = observed value
a = actual value

Sol: calculate 1) $d_1 = \sqrt{(6-4)^2 + (8-3)^2} = \sqrt{2^2 + 5^2} = \sqrt{29} = 5.38$

2) $d_2 = \sqrt{(6-6)^2 + (8-7)^2} = \sqrt{0+1} = 1$

3) $d_3 = \sqrt{(6-7)^2 + (8-8)^2} = \sqrt{1+0} = 1$

4) $d_4 = \sqrt{(6-5)^2 + (8-5)^2} = \sqrt{1+9} = \sqrt{10} = 3.16$

5) $d_5 = \sqrt{(6-9)^2 + (8-9)^2} = \sqrt{4+0} = \sqrt{4} = 2$

3 nearest value

3 P, 0 F → Majority → Pass → Majority go pass

REGRESSION:-

Statistical tool used to understand and quantify the relation between 2 more variables.

* linear Regression:

$$y = \beta_0 + \beta_1 x + \varepsilon$$

→ best suited for linearly seperable data

y = dependent variable
x = Independent variable
β_0 = Constant / Intercept
β_1 = x - slope / coefficient
ε = error.

Date :

* local
Ma
→ LW
→ (

lazy learning

Date :

* Locally weighted Regression:-
To overcome the problem of non linearly seperable data
-> LWR algorithm assign weights to data to overcome the problem.
-> Computationally more expensive.
<-> finding weight for every data

Finding weights-? By kernel smoothing

$$D = ae^{-\frac{||X - X_0||}{2c^2}}$$

X => each training i/p
X_0 -> Value we are predicting
c -> Constant

- we construct a weight matrix (w)
-> for each training i/p (X) and for the value we are trying to predict (X_0)

weight matrix - diagonal matrix

$$\beta = (X' W X)^{-1} X^{-1} WY$$

β = model parameter

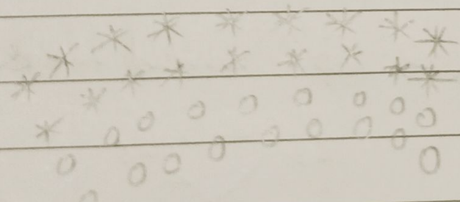
then, prediction can be defined as

$$y = \beta X_0 ,$$ y => Prediction.

* Drawback:-
1) Need to evaluate whole dataset everytime
2) computation cost is more.
3) memory requirement is more.

RADIAL BASIS FUNCTIONS:- -> used in ANN
-> has only one hidden nodes

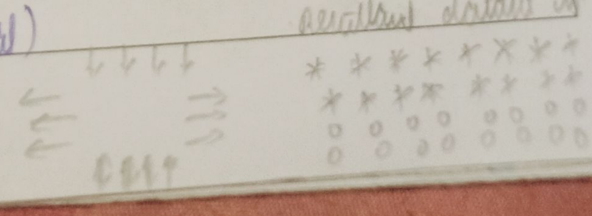
Example



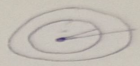
-> 2 class (standard circle)

-> data is not linearly seperable.

-> 2 steps. (not mandatory, only based on requirement)
1. Increase the dimensionality (2D-3D)
2. expand the direction (Horizontal)
compress the direction (vertical)



How RBF does this
→ Consider one center randomly
→ Draw concentric circle
↳ one center for many circle



To expand / compress, we use (3) functions.

1) Multiquadratic
$\phi(r) = (r^2 + c^2)^{1/2}$
$c > 0 \Rightarrow$ constant

2) Inverse multiquadratic
$\phi(r) = \frac{1}{(r^2+c^2)^{1/2}}$

3) Gaussian function
$\phi(r) = \exp\left[\frac{-r^2}{2\sigma^2}\right]$
$\sigma > 0 \Rightarrow$ constant

→ use any one of these for calculation

CASE BASED REASONING :- (CBR)

All instance based learner have 3 properties
1) lazy learners
2) classification is different for each instance
3) instance are represented with n dimensional euclidean space

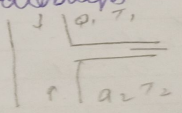
CBR:

- Everytime is considered as case and based on previous case we propose a solution.
- Instances are represented as symbols (not values)
↳ Even

⇒ CBR has 3 components.
1. Similarity function or distance measure
2. Approximation / adjustment of instances
3. Symbolic representations of instances.

For modelling CBR, we use CADET system
↳ (case based design tool) { has 75 prescriptive libraries }

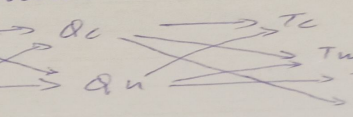
Ex:- Modern waterclips



Junction
$Q_1 \xrightarrow{+} Q_3$
$Q_2 \xrightarrow{+} Q_3$
$T_1 \xrightarrow{+} T_3$
$T_2 \xrightarrow{+} T_3$

use this
Make this

another lab : control temp and water flow



Date :

Page No.

REMARKS ON LAZY and EAGER ALGORITHMS:-

* Lazy learning:

1. Simply stores training data and waits until it get a test tuple
2. less training time, more prediction time.
3. ex: All instance based learning algorith

* Eager learning:-

1. when we give a training set, it constructs a model for classification before getting new example.
2. more training time, less prediction time.
3. ex:- Decision tree, Naive Bayes, ANN etc.