1.12.19.

Machine learning:

A computer program is said to learn w.r.t task 'T' with the performance measure of 'P' and learning experience 'E' if there is improvement in learning performance measure 'P' w.r.t to task 'T' & learning experience 'E'.

*Designing a learning system:

→ machine learning:

making the machine to learn by giving data and work accordingly.

*Types of machine learnings:

→ Supervised learning (Target point/class label is given)
→ Unsupervised learning (class label is not given)
→ Reinforcement learning (Providing feedback to our model)
→ Semi Supervise learning (Partial classlabels are available)

*Well-posed problems:

Problems that are solved using machine learning and they are:

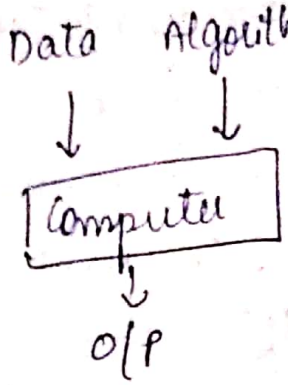
1) Hand written words
2) Driverless cars
3) checkers problem
4) Predicting faults

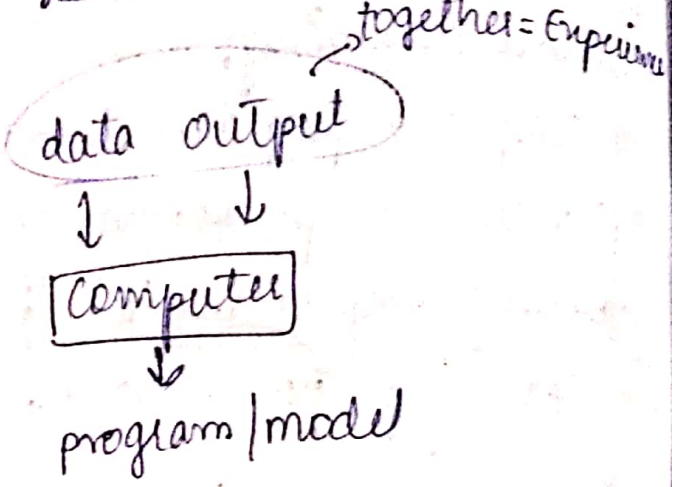
Steps for Problem solving

- → Task
- → Performance measure
- → Learning Experience.

*Designing a learning systems:

14/12/19



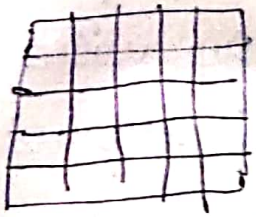


* Regression: Predicting no. of defects.

* classification: Predicting whether s/w has defects or not i.e whether it is working or not.

* cluster: set of similar objects.

* Designing a learning system:



① Choose the learning Experience.

Example:

Checkers problem →
- games played against themselves.
- games played against experts.
- Table of possible moves from a particular point.

② Target function: It depicts how to classify the characteristics (like for example for email system - separation of emails into Inbox & Spam box).

Here, in checkers problem, the target function

is to choose an efficient / appropriate move:

choose move: $B \rightarrow M$

Evaluating function: For each move, depicting its outcome, whether the move which has selected leads to either winning state or loosing state or draw state.

Generic way: $V(b) = B \longrightarrow R$

B ↓ set of all possible moves.

R → set of Real no's

Example: $v(b) = 100 \Rightarrow$ winning state

$v(b) = -100 \Rightarrow$ losing state

$v(b) = 0 \Rightarrow$ draw state

Target function: criteria to built our system.

(3) Representation of Target Function:

x_1: Black cells.

x_2: Blue cells.

x_3: No. of black kings.

x_4: no. of blue kings.

$V(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$ (linear regression model).

(w_0 ↓ interept, x_1 ↓ dependent)

(Similar to $y = m_1x_1 + m_2x_2 + \cdots c$ (i.e $y = mx + c$))

* The previous eq is one of the various models.

* To develop a Target function, best suitable model must be chosen (in Representation of Target function).

$V(b) = 0.5 + 0.7x_1 + 0.3x_2 + 0.7x_3 + 0.3x_4$

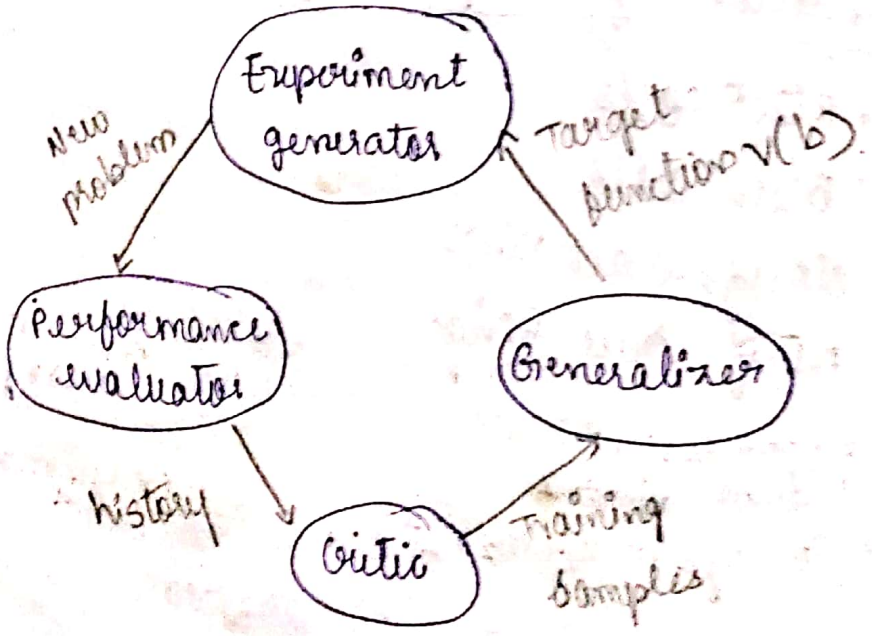
* Approximation Algorithm : It is responsible for generating values of intercepts $(w_1, w_2, w_3, w_4, w_0)$ (or) Constants.

Ex: $V(b) = 0.5 + 0.7x_1 + 0.3x_2 + 0.7x_3 + 0.3x_4$.

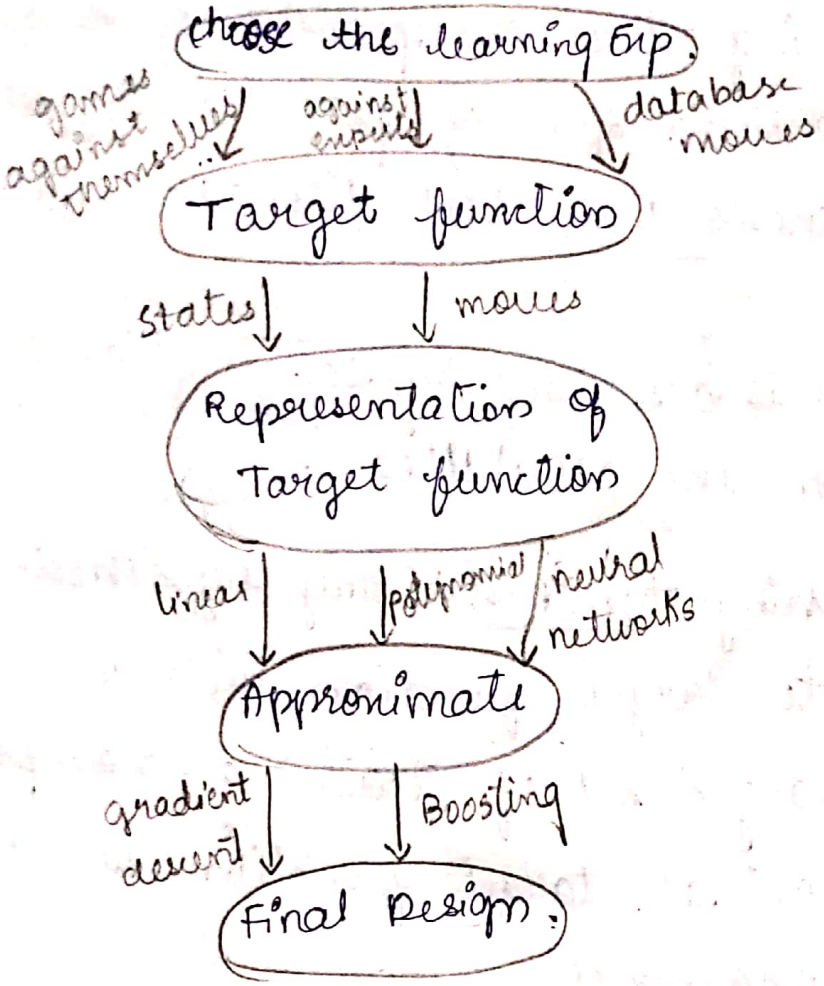
We will give x values we need to check for the $V(b)$ wether positive or negative and make a move.

Final design

~~Hitator~~ : Every model of ML has the same 4 modules i) Critic module
ii) Generalizer
iii) Experiment Generator
iv) Performance Evaluator.



* Design of learning system:



* Issues in machine learning:

1. What are the various algorithms available and how to select the best algorithm for our class..?
2. How much of training data is sufficient?
3. How to choose learning experience for the model?
4. How to define target function to the model?
5. How the learner alters itself to better represent the target function?

Testing data: Data which is used to test our model. evaluate ML model.

Training data: Data which is used to train our model. construct our ML

16/12/19.

* Concept learning: It is the problem of searching through pre-defined space of potential hypothesis for the hypothesis that best fits the training examples.

* Hypothesis: It is described by conjunction of constraints on the attributes.

* Inductive learning hypothesis: Any hypothesis found to approximate target function well over sufficiently larged set of training examples will also approximate target function well over unobserved examples.

* Positive samples: The sample for which class label has boolean value 1.

* Negative samples: The sample for which class label has boolean value 0.

→ C(x)=1, if training sample is +ve; C(x)=0 if training sample is -ve

In order make our machine learn, both positive & negative samples are to be considered.

attributes/features

Ex	Sky	Airtemp	humidity	wind	water	forecast	Enjoy sport
1	Sunny	warm	Normal	Strong	warm	same	yes
2	Sunny	warm	high	Strong	warm	same	yes
3	Rainy	Cold	high	Strong	warm	change	No
4	Sunny	warm	high	Strong	Cool	change	yes

supervise learning.

Let h_j, h_k be boolean valued functions defined over X, then h_j is more general than or equal to h_k which can be represented as $(h_j \geq_g h_k)$

↳ stands for general

iff $\forall_x \in X \; [h_k(x) = 1] \rightarrow (h_j(x) = 1)$

Let 2 samples → (these 2 are nothing but hypothesis).

① <sunny, warm, ?, ?, warm, same> → specific

② <sunny, ?, ?, ?, ?, ?> → generic.

If sample is specific, then it also satisfies to be a generic sample.

Find S algorithm:

* Finding a maximally specific hypothesis (h).

Step-1: Initialize h to most specific hypothesis H

Step-2: For each positive training instance x
→ For each attribute constraint a_i, if satisfied by x, then do nothing else, replace a_i and in 'h' by next most general constraint that is satisfied by x.

Step-3: Output the hypothesis h.

NOTE: Find S algorithm will not consider negative training samples.

Example:

⇒1) h <∅, ∅, ∅, ∅, ∅, ∅>

1st sample ← h<sunny, warm, Normal, strong, warm, same>

2nd sample ← h<Sn, warm, ?, strong, warm, same> (diff)

3rd sample ← ~~h<Sn, warm~~ is -ve so ignored.

4th sample ← h<sunny, warm, ?, strong, ?, ?>

Algo:

*A hypothesis 'h' is consistent with set of training examples D, iff $h(x) = c(x)$.

→for each example $\langle x, c(x) \rangle$ in D.

consistant $(h, D) = (\forall \langle x, c(x) \rangle \in D)$

1st sample circle big blue Yes.

2nd sample triangle small red NO.

Hypothesis -1 → [<?, big, ?> → Yes]

Hypothesis -2 → [<?, small, ?> → Yes]

Here, 1st hypothesis is consistent because, it is correctly classifying the above samples (i.e if 2nd attribute is big, it has class label as Yes, otherwise ⇒ NO.

∴ 1st sample & 2nd are classified correctly w.r.t Hyp-1).

* ~~Version Space~~: Denoted by $VS_{H,D}$, whereas 2nd hyp is not consistent because it is not classifying given samples correctly. (∵ hyp-2 says, if 2nd attribute is small, then class label is Yes, but the samples are not satisfied).
∴ Hyp-1 falls under version space.

*Version Space: Denoted by $VS_{H,D}$, w.r.t hypothesis space H, & training examples D. is subset of hypothesis from H consistent in Training example in D. $[VS_{H,D} \equiv \{h \in H \mid Consistent(h, D)\}$.

* List then eliminate algorithm;

Step 1: Initialize version space with a list containing every hypothesis in h.

Step 2: For every training example $x, c(x)$ remove from version space, any hypothesis h, for which

$$h(x) \neq c(x)$$

Step 3: Output the list of hypothesis in Version Space.

* General boundary:

→ General boundary 'G' w.r.t hypothesis space H, and training data D is set of maximally general members of H consistent with D.

* Specific boundary:

→ Specific boundary S' w.r.t hypothesis space H, and training data D is set of minimally general

(maximally specific) members of H which is consistent with D.

*Candidate elimination algorithm:

1) It is applicable for both positive and negative samples.
2) If it is +ve sample, we would work with specific boundary → $S_0 = \{\theta, \theta, \theta, \theta, \theta, \theta\}$.
3) If it is -ve sample, we would work with general boundary → $G_0 = \{?, ?, ?, ?, ?, ?\}$.

Example

	Sky	Airtemp	humidity	wind	water	forecast	Enjoy sport
1.	Sunny	warm	normal	strong	warm	same	Yes.
2.	Sunny	warm	high	strong	warm	same	Yes
3.	Rainy	cold	high	strong	warm	change	NO.
4.	Sunny	warm	high	strong	cool	change	Yes.

Initialize specific boundary.

Step1: $S_0 = \{\theta, \theta, \theta, \theta, \theta, \theta\}$

$G_{10} = \{?, ?, ?, ?, ?, ?\}$ → initialize generic boundary.

Step2: For first training sample,

it is +ve sample, so work with specific boundary

$S_1 = \{$Sunny, warm, normal, strong, warm, same$\}$.

$G_{11} = \{?, ?, ?, ?, ?, ?\}$ → same as G_{10} (no change).

Now, check whether S_1 & G_{11} is consistent with 1st sample or not.

Step 3: For 2nd sample:

it is true, work with specific boundary.

$S_2 = \{Sunny, warm, ?, Strong, Warm, same\}$

$G_{12} = \{?,?,?,?,?,?\}$ → same as G_1 (no change)

→ check whether S_2, G_{12} is consistent with 1st & 2nd samples (or) not

Step 4: For 3rd training sample.

→ it is -ve sample, work with Generic boundary.

$S_3 = \{Sunny, warm, ?, strong, warm, same\}$

(no change, with same as S_2).

$G_{13} = \{\langle Sunny, ?,?,?,?,? \rangle \langle ?, Warm, ?,?,?,? \rangle$

$\langle ?, ?, normal, ?,?,? \rangle$ (it is not satisfying 2nd sample (ie high → yes), so inconsistent, hence it is removed), $\langle ?,?,?,?, Cool, ? \rangle$ (it is not satisfying 1st sample (ie warm → yes) so inconsistent, hence it is removed

$\langle ?,?,?,?,?, same \rangle \}$.

[we are checking only for 1st, 2nd & 3rd samples bc it is consistent

(∵ for 4th sample (cool → yes) it becomes inconsistent]

∴ $G_{13} = \{\langle sunny, ?,?,?,?,? \rangle \langle ?, warm, ?,?,?,? \rangle$

$\langle ?,?,?,?,?, same \rangle$.

&

Step 5: For fourth sample.

it is +ve sample, so work with specific boundary

$S_4 = \{\langle sunny, warm, ?, strong, ?, ? \rangle\}$

G_{14} - 4th sample is +ve, we don't apply general boundary.

- we consider same G_3 & check whether consistent for all 4 samples (or) not.

$\therefore G_14 = \{\langle sunny, ?, ?, ?, ?, ? \rangle, \langle ?, warm, ?, ?, ?, ? \rangle,$

$\langle ?, ?, ?, ?, ?, same \rangle\}$

(It is satisfying 1,2,3 samples, but not 4th sample. Hence eliminated)

$\therefore G14 = \{\langle sunny, ?, ?, ?, ?, ? \rangle, \langle ?, warm, ?, ?, ?, ? \rangle\}$

⇒ Finally:

$S = \{\langle sunny, warm, ?, strong, ?, ? \rangle\}$

$G_1 = \{\langle sunny, ?, ?, ?, ?, ? \rangle, \langle ?, warm, ?, ?, ?, ? \rangle\}$

Version Space:

S $\boxed{\langle sunny, warm, ?, strong, ?, ? \rangle}$

$\langle sunny, ?, ?, strong, ?, ? \rangle$ $\langle sunny, warm, ?, ?, ?, ? \rangle$

$\langle ?, warm, ?, strong, ?, ? \rangle$ ← intermediate hypothesis

G $\boxed{\langle sunny, ?, ?, ?, ?, ? \rangle \langle ?, warm, ?, ?, ?, ? \rangle}$

Q) Size | Color | Shape | label
big | red | circle | NO
small | red | triangle | No
small | red | circle | Yes
big | blue | circle | No (wrong)
small | blue | circle | Yes.

Sol: Step 1: $S_0 = \{\theta, \theta, \theta\}$
$G_0 = \{?, ?, ?\}$

Step 2: Take 1st sample.
→ it is -ve sample so, we work on general boundary

$S_1 = \{\theta, \theta, \theta\}$ → same as S_0

$G_1 = \{$<small, ?, ?> <?, blue, ~~triangle~~ ? > <?, ?, triangle>$\}$

↳ we need to check consistency of all these attributes for only 1st sample.

Step 3: Take 2nd sample.
→ it is -ve sample, so we work on general boundary.

$S_2 = \{\theta, \theta, \theta\}$ → same as S_1

$G_2 = \{$<big, ?, ?> <?, blue, ?> <?, ?, Circle>$\}$

(it is not consistent in 1st sample i.e (big → NO) hence eliminated)

(it is not consistent in 1st sample i.e (Circle → NO) hence eliminated)

↳ By checking consistency on 1st ~~two~~ samples we get

$G_2 = \{$ <?, blue, ?> $\}$.

Step 4: Take 3rd sample.

→ it is +ve sample, work on specific boundary.

$S_3 = \{ small, red, circle \}$.

$G_3 = \{ <?, blue, ?> \}$ → same as G_2.

Check consistency for 1st three samples.

Here $G_3 = \{ <?, blue, ?> \}$

(it is not satisfying sample 3 i.e red → Yes
hence eliminated

$\therefore G_3 = \{ <?, ?, ?> \}$

Step 5: Take 4th sample.

→ it is -ve sample, work on general boundary.

$S_4 = \{ small, red, circle \}$ → same as S_3

$G_4 = \{ <small, ?, ?> \}$

↓
it is not

30/12/19.

28) Size color shape label.

big red circle NO.

Small red ▵le NO.

Small red circle Yes

big blue circle NO.

Small blue circle yes.

Sol: Step1: Initially;

$S_0 = \{\theta, \theta, \theta\}$.

$G_0 = \{?, ?, ?\}$.

Step 2: Take 1st sample.

→ it is -ve sample, so work with Generic boundary.

$S_1 = \{\theta, \theta, \theta\}$ ⇒ same as S_0.

$G_1 = \{\langle small, ?, ? \rangle \langle ?, blue, ? \rangle \langle ?, ?, triangle \rangle\}$ satisfies 2nd sample also

Step 3: Take 2nd sample.

→ it is -ve samples, so work with Generic boundary.

$S_2 = \{\theta, \theta, \theta\}$ ⇒ same as S_1

$G_{12} = \{\langle small, blue, ? \rangle \langle small, ?, circle \rangle \langle ?, blue, ? \rangle$

$\langle big, ?, ▵le \rangle \langle ?, blue, ▵le \rangle\}$

Step4: 3rd sample is +ve sample, so Specific boundary.

$S_3 = \{small, red, circle\}$.

$G_3 = \{\langle small, blue, ? \rangle \langle Small, ?, circle \rangle \langle ?, blue, ? \rangle$

$\langle$big, ?, circle$\rangle$ $\langle$?, blue, triangle$\rangle$ }

$\therefore G_3 = \{$small, ?, circle$\}$

Step 5: It is -ve sample, so Generic boundary.

S_4 = same = {small, red, circle}.

$G_4 = \{\langle$small, ?, circle$\rangle\}$

Step 6: +ve sample. $\therefore S_5 = \{$small, ?, circle$\}$

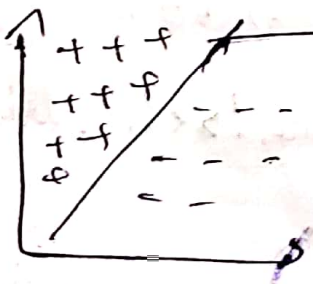
satisfies all samples.

$G_5 = \{\langle$small, ?, circle$\rangle\}$ → satisfies all samples.

finally intermediate sample = $\{\langle$small, ?, circle$\rangle\}$

it is the best hypothesis.

* The final o/p of machine learning model is hypothesis; Eg:



→ This line is hypothesis here

* Remarks on Version Space & Candidate Elimination

1) Will the candidate elimination algorithm converges to the correct hypothesis (Yes it converges, but there's a problem when there is an error in the data set).

2) Which training example should the learner request next.

3) How can partially learned concepts be used.

21/12/19:

* Inductive Bias:

⇒ Using Find S find the hypothesis.

Sunny	Warm	Normal	Strong	Cool	Change Yes
Cloudy	Warm	Normal	Strong	Cool	Change Yes.
Rainy	Warm	Normal	Strong	Cool	Change No

Sol: Find S can be applied only for +ve samples.

Sample 1: $S_1 = \langle$ Sunny, Warm, Normal, Strong, Cool, Change $\rangle$

Sample 2: $S_2 = \langle$?, Warm, Normal, Strong, Cool, Change $\rangle$

Sample 3: ∴ It is -ve sample, so find S can't be applied

∴ But S_2 hyp is wrongly classifying the 3rd sample.

∴ Hence the hypothesis S_2 is not consistent.

∴ We use disjunction hypothesis.

i.e $\{\langle$ Sunny, ?, ?, ?, ?, ? $\rangle$ V $\langle$ Cloudy, ?, ?, ?, ?, ? $\rangle\}$

↓ OR operation

Now this is a consistent hyp. ∴ it satisfies the given whole data set.

Inductive bias: Inductive bias of L (learning algorithm) is any minimal set of assertions 'B' such that for any Target concept 'c' & corresponding Training Samples D_c,

$$(\forall x_i \in X)\left[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)\right]$$

X → set of all training samples

B → set of assertions

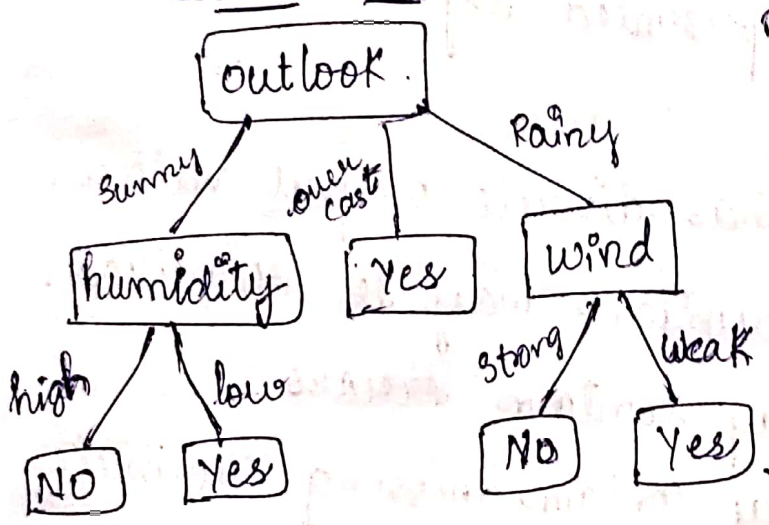
NOTE: $y \vdash z$ means $\Rightarrow z$ is provable by using y.

23/12/19 MODULE-2 DECISION TREES:

→ Decision Tree is also a supervised learner.

→ Definition: Decision tree is a classifier which consists of nodes, where each node is representing an attribute and branch represents test condition and leaf node represents the outcome.

Decision tree:



depicting whether a person can play game or not.

hypothesis for above decision tree:

$\langle outlook = Sunny \overset{and}{\wedge} humidity = low \rangle \overset{or}{\vee} \langle outlook = overcast \rangle$

$\vee \langle outlook = Rainy \wedge wind = weak \rangle$

Formal

Definition: Decision Tree learning is a method of approximating discrete (countable finite) value target function in which learned function is represented by decision tree.

27/12/19

*Appropriate problems for decision tree learning

1) Instances are represented by attribute value pairs.
2) Target function has discrete output values.
3) Disjunctive descriptions may be required.
4) Training data may contain errors.
5) Training data may contains missing attribute values.

*Data preprocessing: process of data cleaning, finding values of missed attributes and Error handling.

Feature Extraction: Extracting the required feature among various features.

*EOE

* ID3 algorithm is used

Entropy: It characterizes the impurity (randomness in data) of arbitrary collection of examples (complexity)

high entropy ⇒ high impurity → (means not a best classifier)

Formula:

$$\text{Entropy}(S) = -P_{\oplus} \log_2 P_{\oplus} - P_{\ominus} \log_2 P_{\ominus}$$

$P_{\oplus}$ ⇒ probability of +ve sample.

$P_{\ominus}$ ⇒ probability of -ve sample.

* Entropy: is the measure of randomness.

* Information Gain: 1) Effectiveness of attribute in classifying the data (mainly used for splitting the nodes)

2) It is an estimation of reduction in entropy w.r.t that attribute.

(For every attribute, information gain is calculated. The attribute which has highest information gain will be selected as the root node).

Formula:

⇒

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{V \in \text{values}(A)} \frac{|S_V|}{|S|} \text{Entropy}(S_V)$$

A → attribute S → entire sample.

→ (i.e either +ve samples or -ve samples)

NOTE:

1) If all samples belong to the same class, then Entropy is 0. (stopping condition in Decision tree)

2) Equal no. of the +ve & -ve samples, then Entropy = 1

3) Unequal no. of the +ve & -ve samples, then entropy lies b/w 0 & 1.

9) Construction of decision Tree: (using ID3 algorithm)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	NO
D2	Sunny	Hot	High	Strong	NO.
D3	Overcast	Hot	High	Weak	Yes.
D4	Rain	mild	High	Weak	Yes.
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	NO.
D7	Overcast	Cool	Normal	Strong	Yes.
D8	Sunny	mild	High.	Weak	NO.
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	mild	Normal	Weak	yes.
D11	Sunny	mild	Normal	Strong	Yes.
D12	Overcast	mild	High	Strong	Yes.
D13	Overcast	Hot	Normal	Weak	Yes.
D14	Rain	mild	High	Strong	No.

Sol: Step-1: Entropy of entire sample i.e Entropy (S) is to be calculated.

Step-2: Find out the root node

→ To find the root node, first we calculate Information gain of each attribute, and the attribute which is having high Information gain is taken as root

node.

Information Gain:

$$Gain(S,A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

→ $Entropy(S) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$$= -\frac{9}{14} \times \log_2\left(\frac{9}{14}\right) - \frac{5}{14} \log_2 \times \frac{5}{14} = 0.9403$$

$Entropy(S) = 0.9403$ → (remains same for all attributes)

(Attribute) (values)

① For A = Outlook, V = Sunny, overcast, Rain.

$Gain(S, Outlook) = Entropy(S) - \left[\frac{5}{14} Entropy(Outlook_{Sunny}) + \frac{4}{14} Entropy(Outlook_{overcast}) + \frac{5}{14} Entropy(Outlook_{rain})\right]$

(probability of Sunny in total sample; prob of overcast; prob. of rain)

→ $Gain(S, Outlook) = 0.9403 - \left[\frac{5}{14} Entropy(Outlook_{Sunny}) + \frac{4}{14} Entropy(Outlook_{overcast}) + \frac{5}{14} Entropy(Outlook_{rain})\right]$

i) $Entropy(Outlook_{Sunny}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$$= -\frac{2}{5} \times \log_2\left(\frac{2}{5}\right) - \frac{3}{5} \times \log_2\left(\frac{3}{5}\right) = 0.9710$$

ii) $Entropy(Outlook_{overcast}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

=> Here since, all the samples (overcast) belong to same class (i.e +ve samples), then entropy is '0'.

iii) $Entropy(Outlook_{Rain}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$

$$= -\frac{3}{5} \times \log_2\left(\frac{3}{5}\right) - \frac{2}{5} \times \log_2\left(\frac{2}{5}\right) = 0.9710$$

$\therefore Gain(S, Outlook) = 0.9403 - \left[\frac{5}{14}(0.9710) + \frac{4}{14}(0) + \frac{5}{14}(0.9710)\right]$

$\therefore Gain(S, Outlook) = 0.2467$

② A = Temperature, V = Hot, Mild, Cold

$\rightarrow Gain(S, Temperature) = Entropy(S) - \left[\frac{4}{14} Entropy(Temperature_{Hot}) + \frac{6}{14} Entropy(Temperature_{Mild}) + \frac{4}{14} Entropy(Temperature_{Cool})\right]$

i) $Entropy(Temperature_{Hot}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$

$= -\frac{2}{4}\log_2\left(\frac{2}{4}\right) - \frac{2}{4}\log_2\left(\frac{2}{4}\right)$ => Here equal no of +ve & -ve samples, so Entropy = 1

ii) $Entropy(Temperature_{Mild}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$

$= -\frac{4}{6}\times\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\times\log_2\left(\frac{2}{6}\right) = 0.9183$

iii) $Entropy(Temperature_{Cool}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$= -\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) = 0.8113$

$\therefore Gain(S, Temperature) = 0.9403 - \left[\frac{4}{14}(1) + \frac{6}{14}(0.9183) + \frac{4}{14}(0.8113)\right]$

$\therefore Gain(S, Temperature) = 0.0292$

⑤ A = Humidity, V = High, Normal.

$Gain(S, Humidity) = Entropy(S) - \left[\frac{7}{14} Entropy(Humidity_{High}) + \frac{7}{14} Entropy(Humidity_{Normal})\right]$

i) $Entropy(Humidity_{High}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$= -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right) = 0.9852$

ii) Entropy(Humidity Normal) $= -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$= -\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right) = \boxed{0.5917}$.

$\therefore$ Gain(S, Humidity) $= 0.9403 - \left[\frac{7}{14}(0.9852) + \frac{7}{14}(0.5917)\right]$.

$\boxed{\therefore \text{Gain}[S, \text{Humidity}) = 0.1519}$.

④ For A = Wind, V = weak, strong.

$\text{Gain}(S, \text{Wind}) = \text{Entropy}(S) - \left[\frac{8}{14}\text{Entropy}(\text{Wind}_{\text{weak}}) + \frac{6}{14}\text{Entropy}(\text{Wind}_{\text{strong}})\right]$

i) $\text{Entropy}(\text{Wind}_{\text{weak}}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

$= -\frac{6}{8}\times\log_2\left(\frac{6}{8}\right) - \frac{2}{8}\times\log_2\left(\frac{2}{8}\right) = \boxed{0.8113}$ //

ii) $\text{Entropy}(\text{Wind}_{\text{strong}}) = -P\oplus \log_2 P\oplus - P\ominus \log_2 P\ominus$.

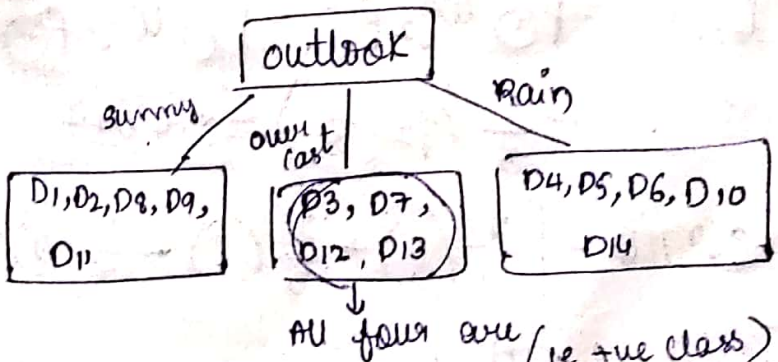
~~$-\frac{3}{6}\times\log$~~ [Here equal no. of +ve & -ve samples, so Entropy is $\boxed{1}$ //].

$\therefore \text{Gain}(S, \text{Wind}) = 0.9403 - \left[\frac{8}{14}(0.8113) + \frac{6}{14}(1)\right] = \boxed{0.0481}$ //.

$\boxed{\therefore \text{Gain}(S, \text{Wind}) = 0.0481}$.

$\therefore$ Among four attributes, Outlook ~~Attribute~~ is having the highest Information Gain value. Hence Outlook is taken as Root Node.

$\Longrightarrow$

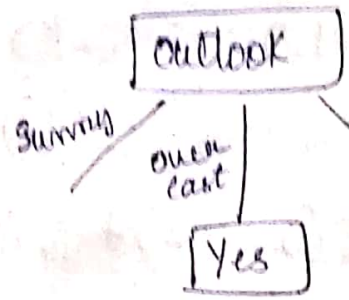


All four are (ie +ve class) of same class.

Hence stopping condition (acc to NOTE(1))

$\therefore$ Hence it becomes leaf node.

ie



Partial decision tree.

→ Step 3

Now for left side i.e for D1, D2, D8, D9, D11.

(D1, D2, D8, D9, D11) Find the entropy for total sample.

Entropy(S) = $-P(+)\log_2 P(+) - P(-)\log_2 P(-)$.

$= -\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right) = 0.9710$

→ Now find Information Gain for attributes Temperature, Humidity, Wind. (because Outlook is already a root node)

① A = Temperature, V = hot, cold, mild.

Gain(S, Temperature) = Entropy(S) − $\left[\frac{2}{5}\text{Entropy}\left(\text{Temperature}_{\text{hot}}\right) + \frac{2}{5}\text{Entropy}\left(\text{Temp}_{\text{mild}}\right) + \frac{1}{5}\text{Entropy}\left(\text{Temp}_{\text{cool}}\right)\right]$

i) Entropy(Temperature$_{\text{hot}}$) = 0 [because all samples (hot) belong to same class (-ve)]

ii) Entropy($\text{Temp}_{\text{mild}}$) = 1 [because eq no. of +ve & -ve samples]

iii) Entropy($\text{Temp}_{\text{cool}}$) = $-P(+)\log_2 P(+) - P(-)\log_2 P(-)$

$= -\frac{1}{5}\log_2\left(\frac{1}{5}\right) = 0.4644 = 0$ (because only one class i.e +ve)

Gain(S, Temp) = 0.9710 − $\left[\frac{2}{5}(0) + \frac{2}{5}(1) + \frac{1}{5}(0)\right]$

⇒ Gain(S, Temp) = 0.571

(2) A = Humidity ; V = high, Normal

$$Gain(S, Humidity) = Entropy(S) - \left[\frac{3}{5} Entropy(Humidity_{high}) + \frac{2}{5} Entropy(Humidity_{Normal})\right]$$

$$= 0.971 - \left[\frac{3}{5}(0) + \frac{2}{5}(0)\right]$$

$$Gain(S, Humidity) = 0.9710$$

(3) A = wind, V = weak, strong.

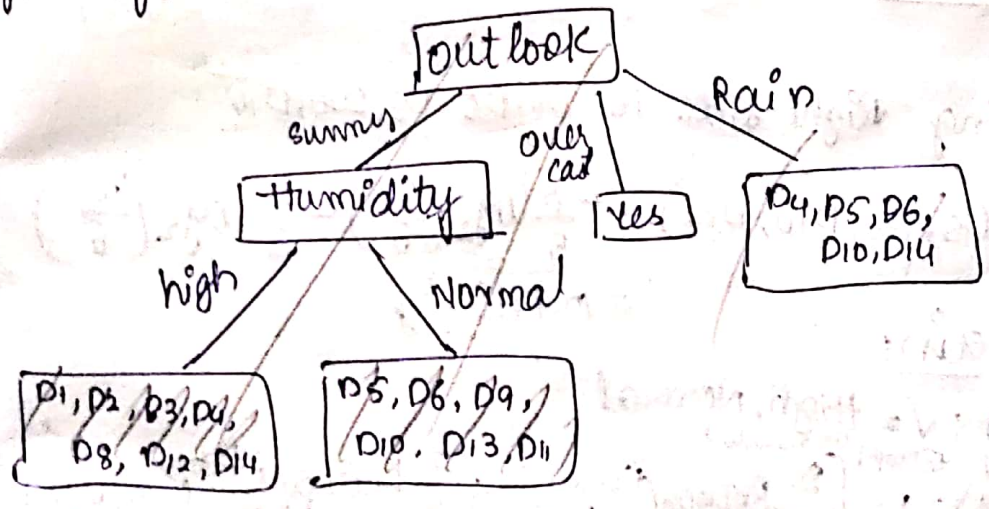
$$Gain(S, wind) = Entropy(S) - \left[\frac{3}{5} Entropy(Wind_{weak}) + \frac{2}{5} Entropy(Wind_{strong})\right]$$

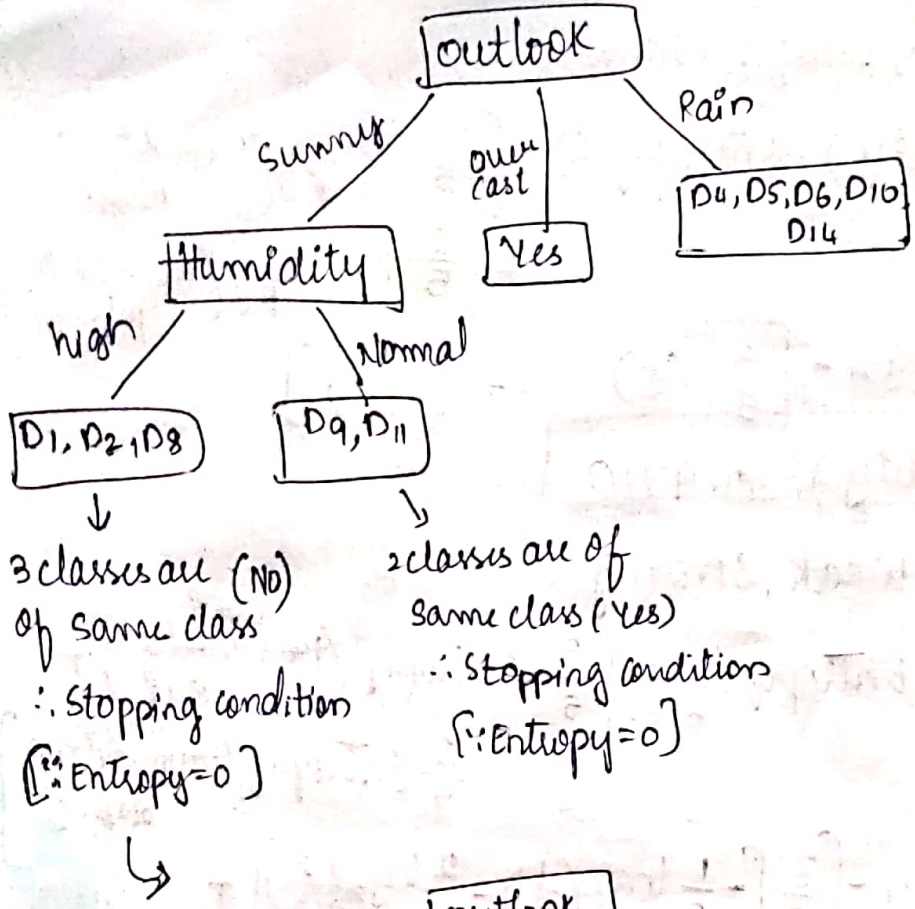
$$= 0.971 - \left[\frac{3}{5}\left[-\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{2}{3}\log_2\left(\frac{2}{3}\right)\right] + \frac{2}{5}[0]\right]$$

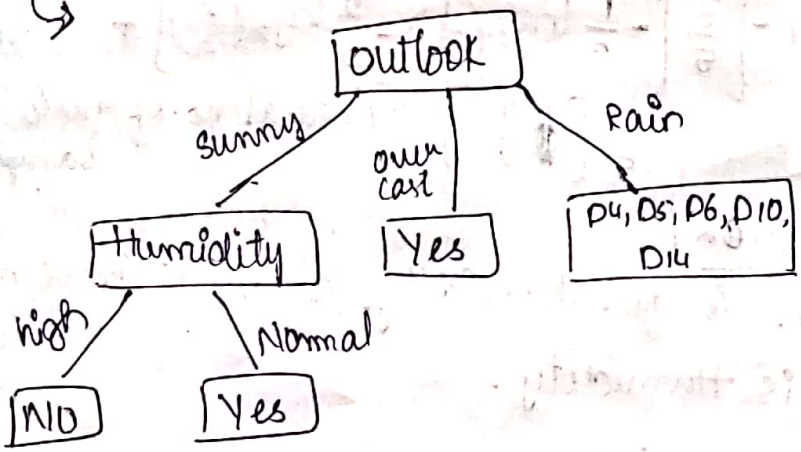
(equal no. of true & true samples)

$$Gain(S, wind) = 0.02$$

∴ Highest Gain is for humidity. so, next (root node) for left side is Humidity.







Step 4: Considering Right side ie D4, D5, D6, D10, D14

$$\text{Entropy}(D4, D5, D6, D10, D14) = -\frac{3}{5}\log_2\left(\frac{3}{5}\right) - \frac{2}{5}\log_2\left(\frac{2}{5}\right)$$

$$= 0.9710.$$

* Information Gain:

(I) ~~A~~ = ~~Humidity~~ ; V = High, Normal. (D4, D5, D6, D10, D14)

$$IG(S, \text{Humidity}) = \underset{0.9710}{1} - \left[\frac{2}{5}\text{Entropy}(\text{Hum}_{\text{high}}) + \frac{3}{5}\text{Entropy}(\text{Hum}_{\text{Normal}})\right]$$

$$IG(S, \text{Humidity}) = 1 - \left[\frac{2}{5}(1) + \frac{3}{5}\text{Entropy}(\text{hum}_{\text{normal}})\right]$$

$$\therefore \text{Ent}(\text{Hum}_{\text{normal}}) = -\frac{2}{3}\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) = 0.9183$$

$$IG(S, \text{Humidity}) = 0.0200$$

④ A = Temp; V = Hot, mild, cool.

$IG(S, Temp) = E(6, 10, 14) - \left[\frac{3}{5} E(Temp_{mild}) + \frac{2}{5} E(Temp_{cool})\right]$

$E(Temp_{mild}) = -\frac{2}{3}\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) = 0.9183$

$E(Temp_{cool}) = 1$ (equal no. of samples +ve & -ve)

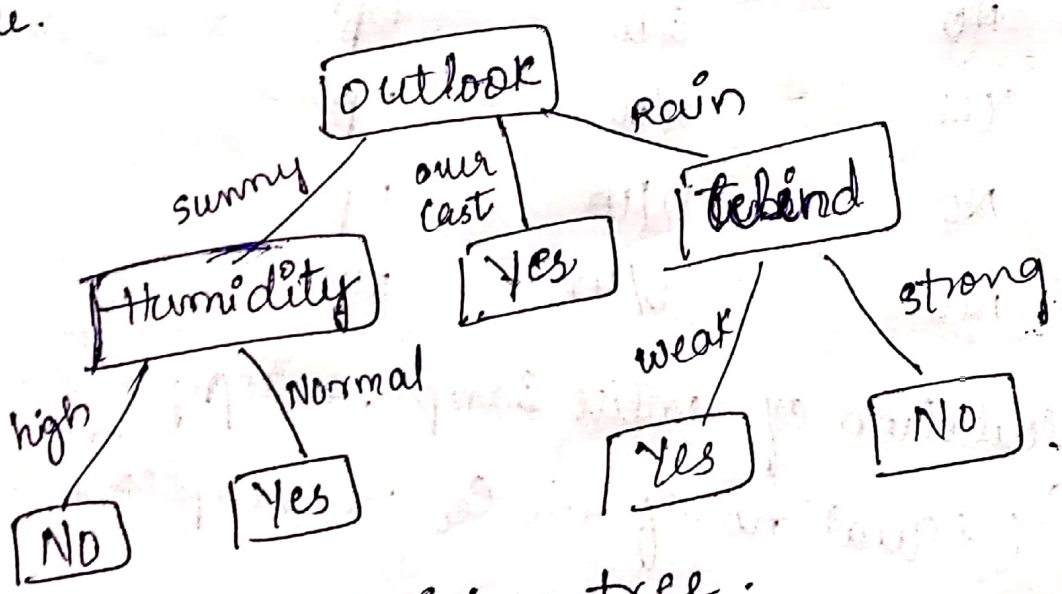
$\boxed{IG(S, Temp) = 0.02}$

⑤ $IG(S, wind) = 0.9710 - \left[\frac{3}{5} E(wind_{weak}) + \frac{2}{5} E(wind_{strong})\right]$

$IG(S, wind) = 0.9710 - \left[\frac{3}{5}(0) + \frac{2}{5}(0)\right]$

$\boxed{IG(S, wind) = 0.9710}$.

Here IG is more for wind, so it is taken as root node.



final decision tree.

2/1/20

2) Construction of Decision Tree:

	Age	Competition	Type	Profit
1.	old	Yes	S/W	down
2	old	NO	S/W	down
3.	old	NO	H/W	down
4	mild	Yes	S/W	down
5	mild	Yes	H/W	down
6	mild	NO	H/W	up
7	mild	NO	S/W	up
8	new	Yes	S/W	up
9	new	NO	H/W	up
10	new	NO	S/W	up

Sol: Step-1: Calculation of entire Sample entropy

$E(S) = 1$ [∵ equal no. of +ve & -ve samples].

Step-2: Information Gain: Calculation

(1)

$$IG(S, Age) = 1 - \left[\frac{3}{10} Entropy(Age_{old}) + \frac{4}{10} E(Age_{mild}) + \frac{3}{10} E(Age_{new})\right]$$

$$= 1 - \left[\frac{3}{10}(0) + \frac{4}{10} E(Age_{mild}) + \frac{3}{10}(0)\right]$$

(→ same class; → same class)

$$E(Age_{mild}) = -\frac{2}{3}\left[\log_2 \frac{2}{3}\right] - \frac{1}{3}\log_2 \frac{1}{3} = 0.9183.$$

$$IG(S, Age) = 1 - \left[\frac{3}{10}(0) + \frac{4}{10}(0.9183) + \frac{3}{10}(0)\right]$$

$$\boxed{IG(S, Age) = 0.6.}$$

Sol (2) for A = Competition ; V = Yes, NO.

$IG(S, Competition) = 1 - \left[\frac{4}{10} Entropy(Comp_{Yes}) + \frac{6}{10} E(Comp_{NO})\right]$

$Entropy(Comp_{Yes}) = -\frac{1}{4}\log_2(\frac{1}{4}) - \frac{3}{4}\log_2(\frac{3}{4}) = 0.8113$.

$Entropy(Comp_{NO}) = -\frac{4}{6}\log_2(\frac{4}{6}) - \frac{2}{6}\log_2(\frac{2}{6}) = 0.9183$

$IG(S, Comp) = 0.1245$

(3) for A = Type ; V = SIW, HIW.

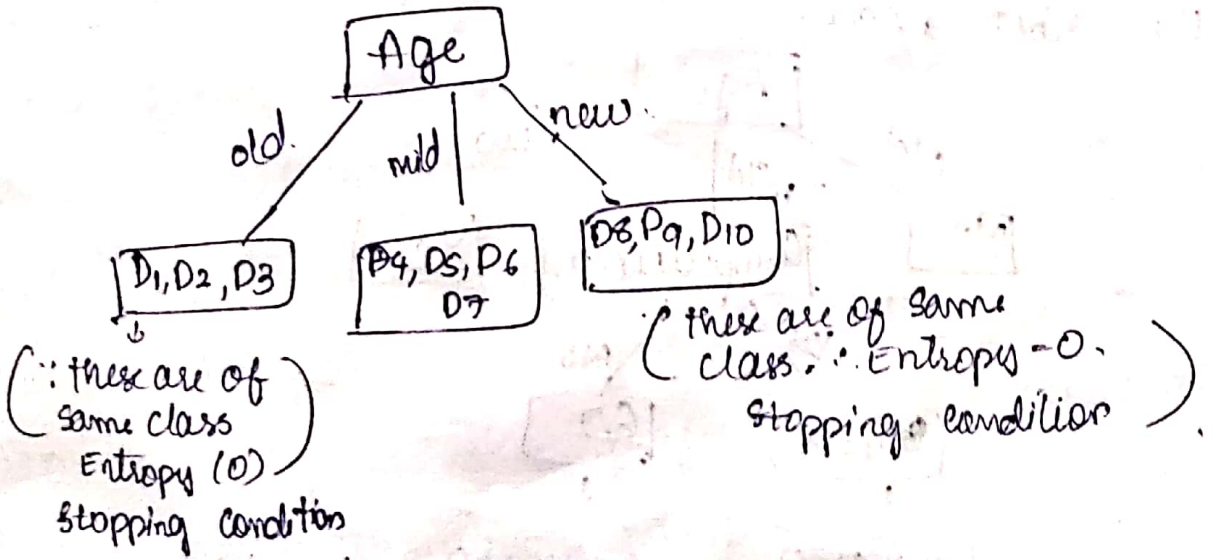
$IG(S, Type) = 1 - \left[\frac{6}{10} E(Type_{SIW}) + \frac{4}{10} E(Type_{HIW})\right]$

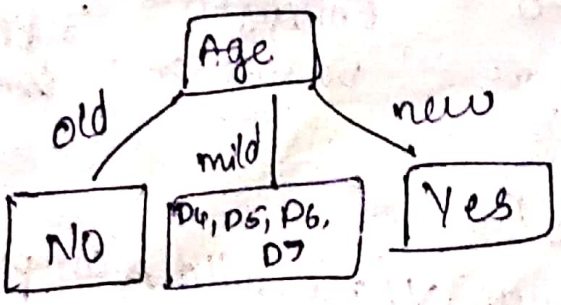
$E(Type_{SIW}) = 1$ [∵ equal no. of samples (Yes & No)]

$E(Type_{HIW}) = 1$ [∵ " " "]

$IG(S, Type) = 0$

(3) ∴ Highest Information Gain is for Age attribute. Hence it becomes the root node.





Step 3:- for the instances D_4, D_5, D_6, D_7.

Calculate Entropy $\Rightarrow E(4,5,6,7) = 1$ [$\because$ equal no. of +ve & -ve samples]

Now Calculate IG for Competition & Type.

① for A = competition ; V = Yes, No [for (D_4, D_5, D_6, D_7)]

$IG(S, \text{Competition}) = 1 - \left[\frac{2}{4}(\text{Entropy}(\text{Comp}_{\text{Yes}})) + \frac{2}{4}\text{Entropy}(\text{Comp}_{\text{NO}})\right]$

$IG(S, \text{comp}) = 1 - \left[\frac{2}{4}(0) + \frac{2}{4}(0)\right]$

$\boxed{IG(S, \text{comp}) = 1}$

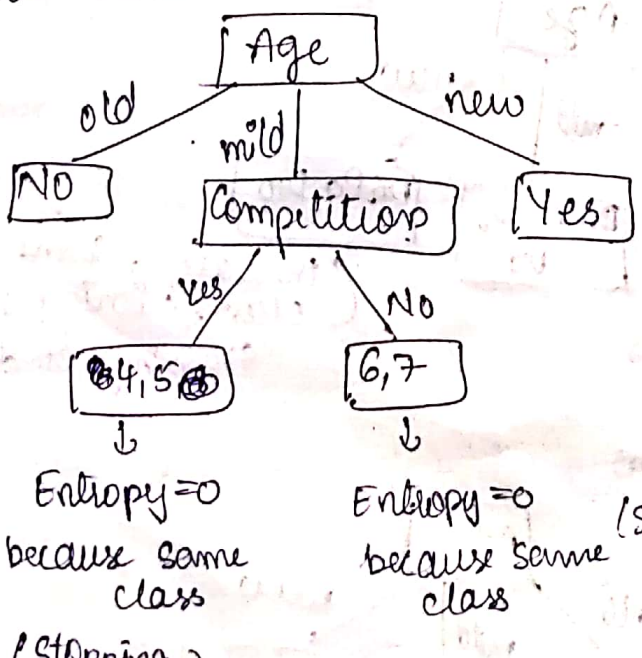
② for A = Type ; V = H/W, S/W.

$IG(S, \text{Type}) = 1 - \left[\frac{2}{4}\text{Entropy}(\text{Type}_{S/W}) + \frac{2}{4}.E(\text{Type}_{H/W})\right]$

$= 1 - \left[\frac{2}{4}(1) + \frac{2}{4}(1)\right]$

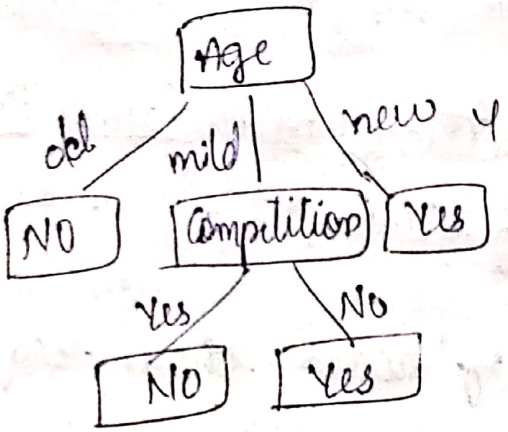
$\boxed{IG(S, \text{Type}) = 0}$

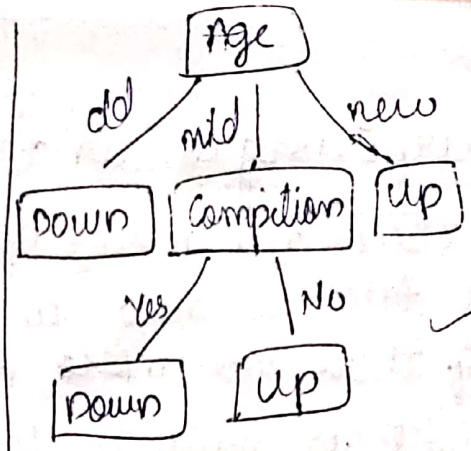
∴ Highest Information Gain is for Competition. Hence it is the root node.



(stopping cond)

∴ Hypothesis = (age = new) ∨ (age = mild ∧ competition = NO)





final decision tree

* ~~Hypothesis~~ Space Search in Decision Tree learning

*) In decision tree: (bias means preference / constraint)

1) One of the bias is selecting simple tree rather than all the other complex tree (ie a preference).

2) Another bias is using information Gain - to select best possible node (ie here the bias is a constraint).

* Hypothesis Space Search in Decision Tree learning:

1) Iterative Dicotomizer 3 (ID3) is complete space for finite descrete valued functions relative to available attributes.

2) ID3 maintains only single current hypothesis as it searches through space of decision trees.

3) ID3 in its pure form performs no back tracking.

4) ID3 uses all training examples at each step (information gain) in search to make statistically based decisions regarding how to refine current hypothesis

6/11/20

Overfitting & Underfitting: in Decision Tree:

If there are more no. of attributes in a sample, then the data is said to be overfitting the model. ∴ If there are more no. of nodes than required (which is said to be more specific), than the, it is called as overfitting.

If there are less no. of attributes in a sample, than required, then the data is said to be underfit the model. ∴ If there are less no. of nodes/ attributes (which is said to more generic), then it is called Underfitting.

6/11/20

* Inductive bias in Decision tree:

1) Restriction Bias: Categorical restriction on set of hypothesis considered.

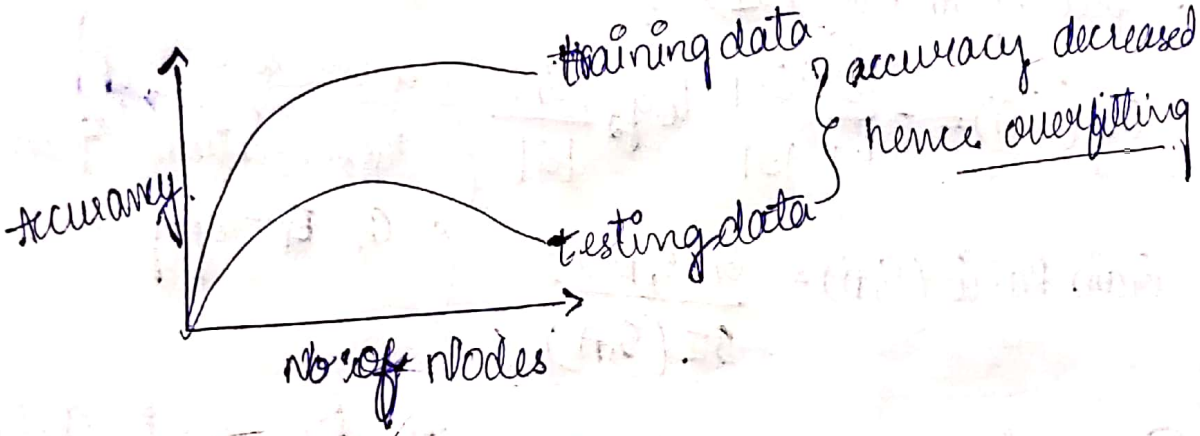
2) Preferance Bias: Preference for certain hypotheses over others. (In decision tree → preferring shorter trees i.e less no. of nodes.)

→ Most of the ML models use only any one of the biases.

* Issues in Decision tree:

1) Avoid overfitting in the data.

Overfitting: Given a hypothesis space (H). A hypothesis (h) belongs to H ($h \in H$) is said to overfit the training data if there exists some alternative hypothesis $h' \in H$ such that h has smaller error than h' over training examples, but h' has smaller error than h over entire distribution of instances.



* How to Avoid Overfitting: the method is - Reducing error pruning

1) Reduce error pruning → (reducing the no. of nodes in decision tree)
   - Post pruning (simpler/out memory in wastage)
   - Pre pruning (cost-efficient)

2) Rule Post pruning:

1) Infer decision tree from training set until the tree fits, the data.
2) Convert the learned tree into equivalent set of

rules.

3) Prune the rules by removing any pre condition (antecedent in the) that improves accuracy.

4) Sort the pruned rules by their estimated accuracy

(2) Incorporating continuous valid valued attributes. (like converting continuous values to discrete)

(3) Handling training examples with missing attribute values.

(4) Alternative measures for selecting attributes:

→ split information (SI)

$$SI(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

$$\text{Gain Ratio}(S,A) = \frac{G(S,A)}{SI(S,A)}$$

} alternatives for Information Gain & Entropy.

(5) Handling the attributes with different cost:

$$\frac{\text{Gain}^2(S,A)}{\text{Cost}(A)}$$

↳ Time required for finding the value of an attribute.

* Use of Bias: It is used to fit the training data in constructing the model.

* Artificial Neural Networks (ANN)

* Problems suited for Neural Networks:

1) Instances are represented by many attribute value pairs.

2) Target function output may be discrete, real valued, continous vector of several real valued (or) discrete valued attributes.

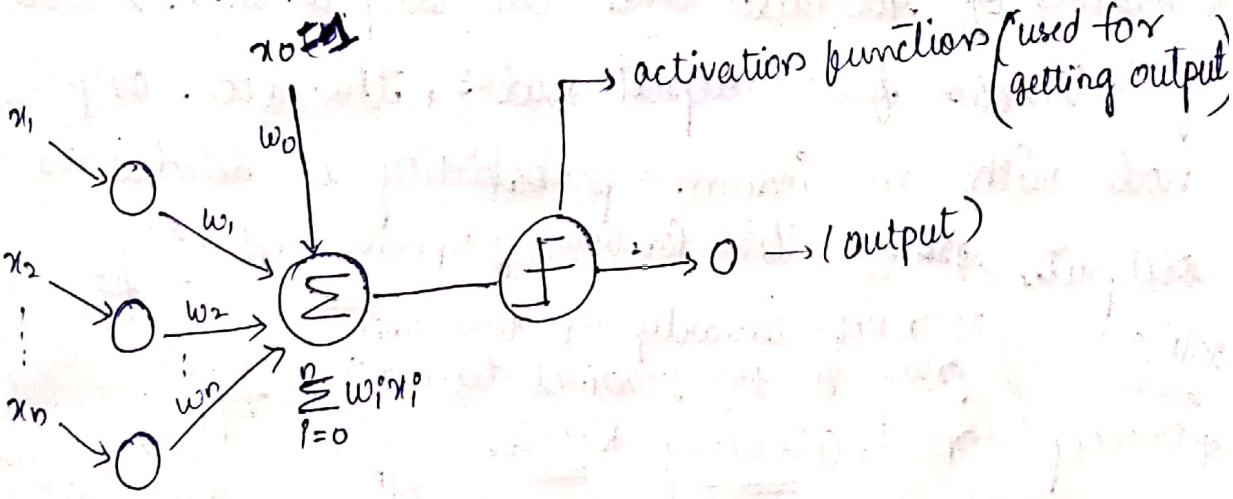
3) Training examples may contain errors.

4) Long training times are acceptable.

5) Fast evaluation of learned target function may be required.

6) Ability of humans to understand learned target function is not important.

* Perceptron (one type of Neural Network)



$$\text{output}(O) = \begin{cases} 1 & \text{if } \sum_{i=0}^{n} w_i x_i > 0 \\ -1 & \text{otherwise} \end{cases}$$

→ Perceptron is used for representing single output. o/p depends on activation function.

→ Perceptron is used for performing logical operations (Best suitable for logical op. except for XOR)

Example: ① Perform AND operation on

T = 1
F = -1 if weights w_1 & w_2 are 0.5 and $w_0 = -0.8$

Sol: For performing we use $\sum_{i=1}^{2} x_i w_i$

Here $x_0 = 1$ (always)

inputs

$= x_0 w_0 + x_1 w_1 + x_2 w_2$

$= 1(-0.8) + (1)(0.5) + (-1)(0.5) \Rightarrow -0.8 + 0.5 - 0.5$

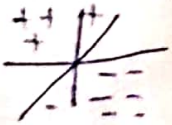
$= -0.8$.

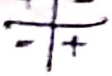
i.e if $\sum_{i=1}^{2} x_i w_i = -1$, if $\sum_{i=1}^{2} x_i w_i$ is < 0 (i.e F)

* Representation of Neural Network:

Hidden layer: performs all the computations on the input data to be processed.

→ If the image size is 30×30, the input layer consists of 900 nodes and the output layer consists of 2 or more than output nodes, the node output node with maximum probability is selected as output.

Soln: It is suitable for linearly separable unit → 

for non-linearly separable unit prefer to for gradient descent → 

* Perceptron learning Rules:

$w_i \leftarrow w_i + \Delta w_i$

where as $\Delta w_i \leftarrow \eta (t - o) x_i$ → inputs.

target o/p

output given by model (perceptron)

↳ constant → learning rate.

→ This learning rule is best suitable for linearly separable unit.

*Gradient descent or delta Rule:

In gradient descent approach, we repeatedly calculate error which is given by -

Error function
$$E(\vec{w}) = \frac{1}{2} \sum_{d \in D} (td - od)^2$$

td → target o/p of a sample
od → o/p of a sample given by our model (actual o/p)

D → Data set
d → data sample.

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \quad w_i \leftarrow w_i + \Delta w_i$$

$$E(\vec{w}) = \frac{1}{2} \sum_{d \in D} (td - od)^2$$

$$\Rightarrow \text{then } \frac{\partial E}{\partial w} = \frac{1}{2} \frac{\partial}{\partial w} \left[\sum_{d \in D} (td - od)^2 \right]$$

$$= \frac{1}{2} \times 2 \sum_{d \in D} (td - od) \frac{\partial}{\partial w_i} (td - od) \quad (\because od = w_i x_i)$$

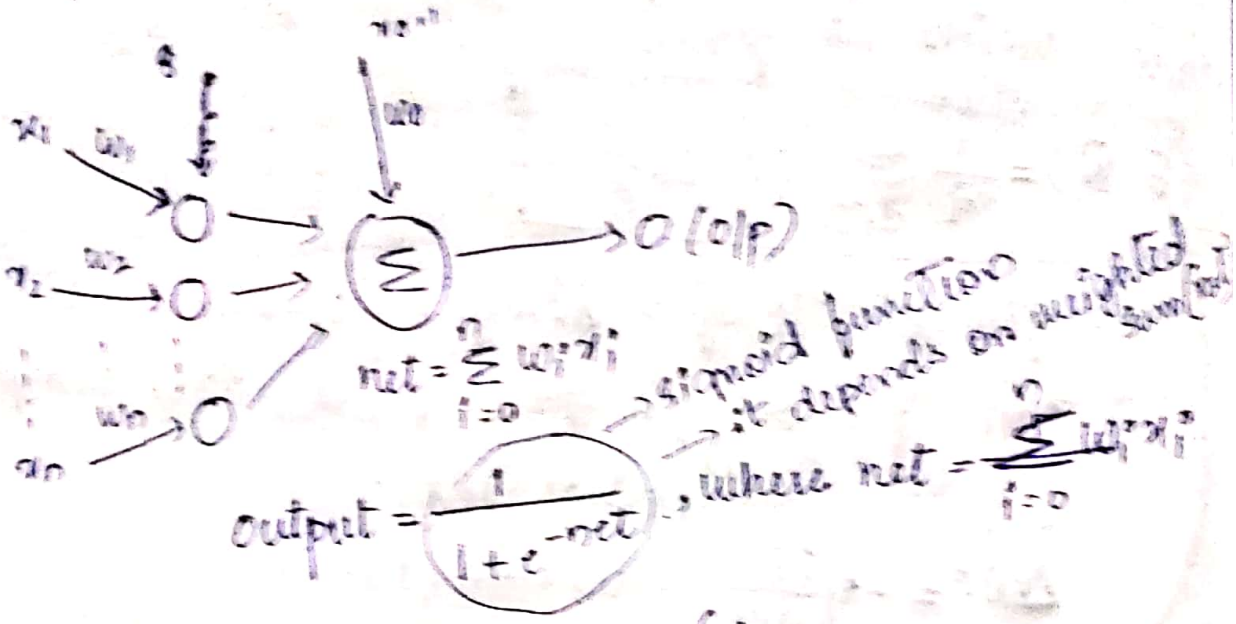
$$\frac{\partial}{\partial w} (td - od) = 0 - \frac{\partial}{\partial w} (w_i x_i) = -x_i$$

$$= \sum_{d \in D} (td - od)(-x_i)$$

$$\Delta w_i = \eta \sum_{d \in D} (td - od) x_i$$

2/1/20

*Sigmoid: It is an activation function.



⇒ Here output lies b/w 0 & 1

NOTE: In Neural Networks, output at a particular neuron, is considered as the input to the next layer neuron.

*Back Propagation Algorithm:

1) Create a fed forward network with 'n' inputs, 'h' hidden units and 'o' output units.

2) Initialize all network weights to small random numbers.

Example: Between −0.5 and +0.5

3) Until the termination condition is met, perform the following steps.

→ For each $(\vec{x}, \vec{t})$ in training examples

i) Input the instance $\vec{x}$ to the network and compute the output of every unit in the network.

ii) For each network output unit 'k' calculate its error δ_k.

$$\delta_k = O_k(1-O_k)(t_k - O_k)$$

iii) For each hidden unit calculate its error.

$$\delta_h = O_h(1-O_h)\sum_{k \in \text{output}} W_{kh}\,\delta_k$$

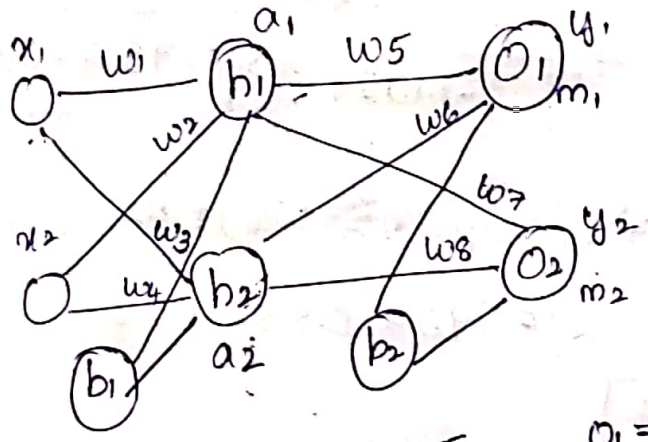
O_h = output at 'h'

Update each network weight

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

$$\Delta_{ji} \leftarrow \eta\,\delta_j\,x_{ji}$$

10)



$x_1 = 0.05$ $\quad b_1 = 0.35$ $\quad O_1 = 0.01$

$x_2 = 0.10$ $\quad b_2 = 0.60$ $\quad O_2 = 0.99$

$w_1 = 0.15$ $\quad w_5 = 0.40$ $\quad \eta = 0.6$

$w_2 = 0.20$ $\quad w_6 = 0.45$

$w_3 = 0.25$ $\quad w_7 = 0.50$

$w_4 = 0.30$ $\quad w_8 = 0.55$

Sol: i) $h_{inet} = x_1 w_1 + x_2 w_2 + b_1$

$= (0.05)(0.15) + (0.10)(0.20) + 0.35$

$h_{inet} = 0.3775$.

$h_1 output = \frac{1}{1+e^{-h_1 net}} = \frac{1}{1+e^{-0.3775}} \Rightarrow$ **0.59337 = $b_1 out$**

$\rightarrow net\, h_2 = x_1 w_3 + x_2 w_4 + b_1 = (0.05 \times 0.25) + (0.10 \times 0.30) + 0.35$

$net\, h_2 = 0.3925$

$output\, h_2 = \frac{1}{1+e^{-net\, h_2}} = \frac{1}{1+e^{-0.3925}} \Rightarrow$ **0.5969 = $output\, h_2$**

$\rightarrow net\, O_1 = (output\, h_1 \times w_5) + (output\, h_2 \times w_6) + b_2$

$= (0.5933 \times 0.4) + (0.5969 \times 0.45) + 0.60$

$net\, O_1 = 1.1059$

$output\, O_1 = \frac{1}{1+e^{-net\, O_1}} = \frac{1}{1+e^{-1.1059}} \Rightarrow$ **0.7514 = output O_1**

$\rightarrow net\, O_2 = (output\, h_2 \times w_7) + (output\, h_2 + w_8) + b_2$

$= (0.5933 \times 0.5) + (0.5969 \times 0.55) + 0.6$

$net\, O_2 = 1.2249$

$Output\, O_2 = \frac{1}{1+e^{-net\, O_2}} = \frac{1}{1+e^{-1.2249}} \Rightarrow$ **0.7729 = output O_2**

$\Rightarrow$ actual outputs: $O_1 = 0.7514$

$O_2 = 0.7729$

But given target o/ps = $O_1 = 0.01$

$O_2 = 0.99$

$\therefore$ checking the error values.

E_{O1} = Error at output1 (O_1)

$= \frac{1}{2}(td - od)^2 = \frac{1}{2}(0.01 - 0.7514)^2 = 0.2748$

Similarly, $E_{O_2} = \frac{1}{2}(0.99 - 0.7729)^2 = 0.0235$

Since the error values are not approx to '0', we try to adjust the weights (back propogation).

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{O_1}} \times \frac{\partial out_{O_1}}{\partial net_{O_1}} \times \frac{\partial net_{O_1}}{\partial w_5}$$

By some derivations $\Rightarrow$ $\frac{\partial E_{total}}{\partial out_{O_1}} = out_{O_1} - Target_{O_1} = 0.7414$

$$\frac{\partial out_{O_1}}{\partial net_{O_1}} = out_{O_1}(1 - out_{O_1}) = 0.1868.$$

$$\frac{\partial net_{O_1}}{\partial w_5} = out_{h_1} = 0.5969.$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.7414 \times 0.1868 \times 0.5969 = 0.0826.$$

23/1/20

*Remarks on Back Propogation Algorithm:

1) Convergence and local minima:-

→ In order to converge, add momentum in every iteration.

$$\Delta w_{ji} = \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1)$$; where $0 < \alpha < 1$

n = no. of iterations (n^{th} iteration)

2) Representational power of feed forward network.

3) Hypothesis Space

4) Generalization and overfitting.

* Evaluating Hypothesis:

1) Sample Error: Denoted by $error_S(h)$ of hypothesis h w.r.t target function f and data sample 'S' is $error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x))$ where,

n = no. Of samples in S.

f(x) = target function.

h(x) = actual o/p given by model.

→ $\delta(f(x), h(x)) = 1$, if $f(x) \neq h(x)$; else 0.

2) True Error: Denoted by $error_D(h)$ of hypothesis w.r.t target function 'f' and distribution 'D', probability that 'h' will misclassify an instance drawn at random according 'D'.

$\Rightarrow error_D(h) = probability_{x \in D}[f(x) \neq h(x)]$

24/1/20

$$error_D(h) = error_S(h) \pm Z_n \sqrt{\frac{error_S(h) * (1 - error_S(h))}{n}}$$

*Thumb rule for Estimation (→ Estimation of True Error & here Estimator is Sample Error):

$$n\, error_S(h)(1 - error_S(h)) \geq 5.$$

*Definitions:

① A random variable can be viewed as name of an experiment with a probabilistic outcome. It's values is the outcome of the experiment.

② Probability distribution for a random variable by specifies the probability $Pr(Y = y_i)$ that y will take that on the value y_i for each possible value of y_i

(3) Expected value (or) mean of a random variable Y is $E(Y) = \sum y_i \Pr(Y = y_i)$ (probability)

(4) Variance of random variable is the width or dispersion of the distribution about its mean.

(5) Central limit theorem:

It is the theorem stating that, sum of large no. of distributed random variables approximately follow normal distribution.

(6) Estimator: A random variable that estimates a random variable 'p'.

(7) n%. confidence interval, estimator estimate for parameter 'p' is an interval that includes 'p' with n% of probability.

*General approach for deriving confidence intervals

(1) Identify the underline population 'p' to be estimated. Eg: True Error

(2) Define the estimator 'y'. → Eg: Sample Error.

(3) Determine the probability distribution by that the once estimator Y including mean & variance.

(4) Determine n% confidence interval by finding the threshold values l and u such that D_y falls between l and u

In this case (Normal distribution) → continuous data.

i.e if $n \to \infty$

then Interval is $\boxed{y \pm Z_N \sigma}$

y = estimator

Z_n = constant

σ is variance.

Confidence interval.

25/1/2020

25/1/2020 MODULE-III

* Bayesian learning and Instance based learning;

Bayesian learning:

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

where $P(h|D)$ = post probability

$P(h)$ = prior probability.

$P(D|h)$ = likelihood

$P(D)$ = marginal value.

*) Find the probability of king in face card;

sol: $P(\text{king}|\text{face card}) = \frac{P(\text{face}|\text{king}) * P(\text{king})}{P(\text{face})}$

$$= \frac{1 * 4/52}{12/52} = \frac{4}{12} = \frac{1}{3}.$$

* Features of Bayesian learning:

1) Each observed training example can increamently decrease or increase the estimated probability that a hypothesis is correct.

2) Prior knowledge can be combined with observed data to determine final probability of the hypothesis.

3) Bayesian methods can accomodate hypothesis that make probabilistic predictions.

4) New instances can be classified by combining the predictions of multiple hypotheses weighted by their probabilities.

Q)

Fruit	yellow	Sweet	long	Total
Orange	350	450	0	650
Banana	400	300	350	400
Others	50	100	50	150
total	800	850	400	1200

Ans: The new instance is

$I = \{yellow, sweet, long\}$

First we have to find out probability with every fruit attribute and then with fruit.

① $P(Yellow/orange) = \dfrac{P(Orange/Yellow) \times P(yellow)}{P(Orange)}$

$= \dfrac{\frac{350}{800} \times \frac{800}{1200}}{\frac{650}{1200}} = 0.66$ //

$P(sweet/orange) = \dfrac{P(Orange/sweet) \times P(Sweet)}{P(Orange)}$

$= \dfrac{\frac{450}{850} \times \frac{850}{1200}}{\frac{650}{1200}} = 0.90$ //

$P(long/orange) = 0$,

New $P(\text{fruit}/\text{orange}) = 0.66 \times 0.90 \times 0$

$= 0$

(2) $P(\text{Yellow}/\text{banana}) = \dfrac{P(\text{banana}/\text{Yellow}) \times P(\text{Yellow})}{P(\text{banana})}$

$= \dfrac{\frac{400}{500} \times \frac{500}{1200}}{\frac{400}{1200}} = 1.$

$P(\text{Sweet}/\text{banana}) = \dfrac{P(\text{banana}/\text{Sweet}) \times P(\text{Sweet})}{P(\text{banana})}$

$= \dfrac{\frac{300}{850} \times \frac{850}{1200}}{\frac{400}{1200}} = 3/4$

$P(\text{long}/\text{banana}) = \dfrac{P(\text{banana}/\text{long}) \times P(\text{long})}{P(\text{banana})}$

$= \dfrac{\frac{350}{400} \times \frac{400}{1200}}{\frac{400}{1200}} = \frac{7}{8}$

New $P(\text{fruit}/\text{banana}) = 1 \times 3/4 \times 7/8 = 0.65$

(3) $P(\text{Yellow}/\text{others}) = \dfrac{P(\text{others}/\text{Yellow}) \times P(\text{Yellow})}{P(\text{others})}$

$= \dfrac{\frac{50}{800} \times \frac{800}{1200}}{450/1200} = 1/3$

$$P(\text{sweet}/\text{others}) = \frac{P(\text{others}/\text{sweet}) \times P(\text{sweet})}{P(\text{others})}$$

$$= \frac{\frac{100}{850} \times \frac{850}{1200}}{\frac{150}{1200}} = \frac{1}{3}$$

$$P(\text{long}/\text{others}) = \frac{P(\text{others}/\text{long}) \times P(\text{long})}{P(\text{others})}$$

$$= \frac{\frac{50}{400} \times \frac{400}{1200}}{\frac{150}{1200}} = \frac{1}{3}$$

Now $P(\text{fruit}/\text{others}) = \frac{1}{3} \times \frac{2}{3} \times \frac{1}{3} = 0.07$

∴ probability of banana is higher.

8)

* We have Prior knowledge that over the entire population only 0.008 have cancer.
* The lab test returns the correct positive results in only 98% of the cases, and the correct negative results in only 97% of the cases.
* Suppose we now observe the new patient for whom lab test returns the positive result. Should we diagnose the patient or not?

Sol: $P(\text{cancer}) = 0.008$ $\quad P(\oplus/\sim\text{cancer}) = 0.97$

$P(\sim\text{cancer}) = 0.992$ $\quad P(\oplus/\sim\text{cancer}) = 0.03$

$P(\oplus/\text{cancer}) = 0.98$

$P(\ominus/\text{cancer}) = 0.02$

Now

$$\rightarrow h_{MAP} = \arg\max_{h \in H} P(D/h)\,P(h)$$

$$h_{MAP} = \arg\max_{h \in H} P(\oplus/\text{cancer})\,P(\text{cancer})$$

$$h_{MAP} = \arg\max_{h \in H} P(\oplus/\sim\text{cancer})\,P(\sim\text{cancer})$$

① $P(\oplus/\text{cancer})\,P(\text{cancer}) = 0.98 \times 0.008$

$= 0.0078$

② $P(+ / \sim cancer) \times P(\sim cancer) = 0.03 \times 0.992$

$= 0.0298$

$h_{MAP} = \sim cancer$.

We have the 2nd case as high probability so, we don't need to diagnoize the patient.

* $P(h/D) = \frac{P(D/h) * P(h)}{P(D)}$

$P(h)$ = Prior probability that hypothesis h hold

$P(D/h)$ = probability of data D given some world in which hypothesis 'h' holds.

$P(D)$ = probability of D given no knowledge about which hypotheses holds.

* Example of text classification:

Text	Category
1) A great game	Sports
2) The election was over	Non Sport
3) Very clean match.	Sports
4) A clean but forgettable game	Sports
5) It was a close election	Non Sports

Sol: Sen = A very close game

$P(Sen / Sports) = ?$

P(Sen | Non-Sports) = ?

P(A | Sports) = ?

P(very | Sports) = ?

P(Close | Sports) = ?

P(game | Sports) = ?

total no. of words in sports = 11

total no. of words in non-sports = 9

Total no. of uniqie words = 14

* Laplace Smoothing:

probability of word = $\frac{\text{word Count} + 1}{\text{Total no. of words} + \text{no. of uniqie words}}$

$P(A|\text{Sports}) = \frac{\text{No. of times word 'A' occured in Sports} + 1}{\text{Total no. of words in sports} + \text{uniqie word}}$

$= \frac{2+1}{11+14} = \frac{3}{25} = 0.12$

$P(\text{very}|\text{Sports}) = \frac{1+1}{11+14} = \frac{2}{25} = 0.08$

$P(\text{Close}|\text{Sports}) = \frac{0+1}{11+14} = \frac{1}{25} = 0.04$

$P(\text{game}|\text{Sports}) = \frac{2+1}{11+14} = \frac{3}{25} = 0.12$

$= P(A|\text{Sports}) * P(\text{very}|\text{Sports}) * P(\text{close}|\text{sports}) * P(\text{game}|\text{sports})$

$= 0.12 \times 0.08 \times 0.04 \times 0.12$

$= 0.00004608$

$P(A/\text{non-sports}) = \frac{1+1}{9+14} = \frac{2}{23} = 0.086.$

$P(\text{very}/\text{non-sports}) = \frac{0+1}{9+14} = \frac{1}{23} = 0.043$

$P(\text{Close}/\text{non-sports}) = \frac{1+1}{9+14} = \frac{2}{23} = 0.086.$

$P(\text{game}/\text{non-sports}) = \frac{0+1}{9+14} = \frac{1}{23} = 0.043$

$= P(A/\text{sports}) \times P(\text{very}/\text{sports}) \times P(\text{close}/\text{sports}) \times P(\text{game}/\text{sports}) =$

$= 0.12 \times 0.08 \times 0.04 \times 0.12$

$= 0.00004608.$

$P(A/\text{non-sports}) = \frac{1+1}{9+14} = \frac{2}{23} = 0.086.$

$P(\text{very}/\text{non-sports}) = \frac{0+1}{9+14} = \frac{1}{23} = 0.043$

$P(\text{close}/\text{non-sports}) = \frac{1+1}{9+14} = \frac{2}{23} = 0.086.$

$P(\text{game}/\text{non-sports}) = \frac{0+1}{9+14} = \frac{1}{23} = 0.043$

$= 0.086 \times 0.043 \times 0.086 \times 0.043$

$= 0.0000136752$

This sentence belongs to sports class.

* Maximum Posterioric hypothesis: (h_{MAP}).

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\ p(h|D)$$

$$= \underset{h \in H}{\operatorname{argmax}}\ \frac{p(D|h) * p(h)}{p(D)}$$

$p(D)$ = Constant.

$$= \underset{h \in H}{\operatorname{argmax}}\ p(D|h) * p(h)$$

When prior probability $p(h)$ is same for every hypothesis

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\ p(D|h)$$

30/1/2020

* Brute-force Bayes Concept learning:

1) For each hypothesis 'h' in 'H'. Calculate posterior probability $p(h|D) = \frac{p(D|h) * p(h)}{p(D)}$

2) Output the hypothesis h_{MAP} with highest posterior probability $h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\ p(D|h)$

3) Assumptions in this approach;

i) Training data T is noise free (No Error)

1) Target concept c is contained in hypothesis space
2) we have no prior reason to believe any hypothesis, is more probable than any other.

$$P(h|D) = \frac{P(D/h) * P(h)}{P(D)} \qquad P(D/h) = \begin{cases} 1 & \text{if } d(i) = h(i) \ \forall d_i \in D \\ 0 & \text{otherwise} \end{cases}$$

if $P(D/h) = 1$ (i.e when hyp is consistent)

then $P(h|D) = \frac{1 \times P(h)}{P(D)}$

$P(h)$ = prob. of a hyp that it classifies the dataset correctly $= \frac{1}{|H|}$

$P(D)$ = prob of a data set that it holds a hyp $= \frac{|\text{Consistent hyp}|}{|H|}$

(∵ set of consistent hyp = version space) ∴ $P(D) = \frac{|VS_{H,D}|}{|H|}$

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

else if $P(h|D) = 0$ (i.e for inconsistent hyp)

then $P(D/h) = \frac{P(h|D) \cdot P(h)}{P(D)} = \frac{0 \times P(h)}{P(D)} = 0$

$\boxed{P(D/h) = 0}$ (for inconsistent hyp) & $\boxed{P(D/h) = \frac{1}{VS_{H,D}}}$ for consistent hyp

* Maximum likelihood and least square Error hypothesis.

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

$$D = \{d_1, d_2, \ldots, d_m\}$$

→ If instances are mutually independent it follows normal distribution, hence we consider probability density function,

$$= \underset{h \in H}{\arg\max} \prod_{i=1}^{m} p(d_i/h)$$

$$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - h(x_i))^2}$$

Apply log

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

Maximum likelihood hypothesis for predicting hypothesis:

$$P(D|h) = P(d_i|h, x_i) = \begin{cases} h(x_i) & \text{if } d_i = 1 \\ 1 - h(x_i) & \text{if } d_i = 0 \end{cases}$$

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i}$$

Minimum description length principle:

$$h_{MAP} = \max_{h \in H} P(D|h) P(h)$$

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}} LC_1(H) + LC_2(D|H).$$