Solution of IAT-2 – Dr. P. N. Singh

Internal Assessment Test 2 – Nov. 2022

Sub:	ARTIFICIALINTELLIGENCEANDMACHINELEARNING				Sub Code:	18CS71	Branch:	CSE
Date:	01/12/2022	Duration:	90 mins	Max Marks: 50	Sem/Sec:	7/A,B,C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	a)Discuss the shortcomings of candidate elimination algorithm	[4+3+3]	CO2	L3

a) **Short comings of Candidate Elimination Algorithm**

1. least commitment strategy - the algorithm modifies the S and G sets as little as possible when accommodating new examples.
2. performs an exhaustive search of the space of all possible classification rules.
3. does not tolerate any noise: the G and S sets "pass" each other.

b)**Apply candidate elimination algorithm on the training samples given below. Last column in the dataset is the target attribute**

1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	cold	high	Strong	Warm	Change	No
3	Sunny	Warm	High	Strong	Warm	Same	Yes
4	Rainy	Cold	High	Strong	Warm	Same	No
5	Sunny	Warm	High	Strong	Cool	Same	Yes

Step 1
The most generic is : [['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
The most specific is : ['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
Step 2
The most generic is : [['?', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', 'Normal', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', 'Same']]
The most specific is : ['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
Step 3
The most generic is : [['?', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', 'Same']]
The most specific is : ['Sunny' 'Warm' '?' 'Strong' 'Warm' 'Same']
Step 4
The most generic is : [['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
The most specific is : ['Sunny' 'Warm' '?' 'Strong' 'Warm' 'Same']
Step 5
The most generic is : [['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
The most specific is : ['Sunny' 'Warm' '?' 'Strong' '?' 'Same']

2	Construct a decision tree from the following training dataset.	[10]	CO2	L3

Sl. No	Attribute 1	Attribute 2	Attribute 3	Traget Class
1	T	T	A	Yes
2	F	F	B	Yes
3	T	T	C	No
4	F	T	A	Yes
5	T	F	C	No
6	T	F	B	Yes
7	T	T	C	No
8	F	T	A	Yes
9	F	F	C	No
10	F	T	B	No

The Decision Tree

For Attribute 1:

S_t[2+, 3-] So, $E(S_t)$ = -2/5* log2(2/5) – 3/5*log2(3/5) = 0.971

S_f[3+,2-] So, $E(S_f)$ = 0.971

Info Gain = 1-(5/10)*(0.971)- (5/10)*(0.971) = 0.029

For Attribute 2:

S_t[3+, 3-] So, $E(S_t)$ = 1

S_f[2+,2-] So, $E(S_f)$ = 1

Info Gain = 1-(5/10)*1- (5/10)*1 = 0.0

For attribute 3:

S_a[3+, 0-] So, $E(S_a)$ =0

S_b[2+,1-] So, $E(S_b)$ = -2/3*log2(2/3) – 1/3*log2(1/3) = 0.9183

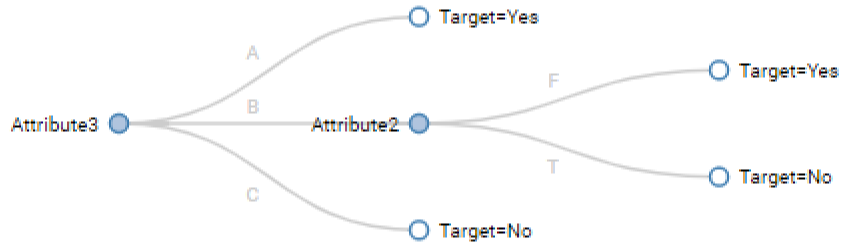
S_c[0+,4-] So, $E(S_c)$ = 0

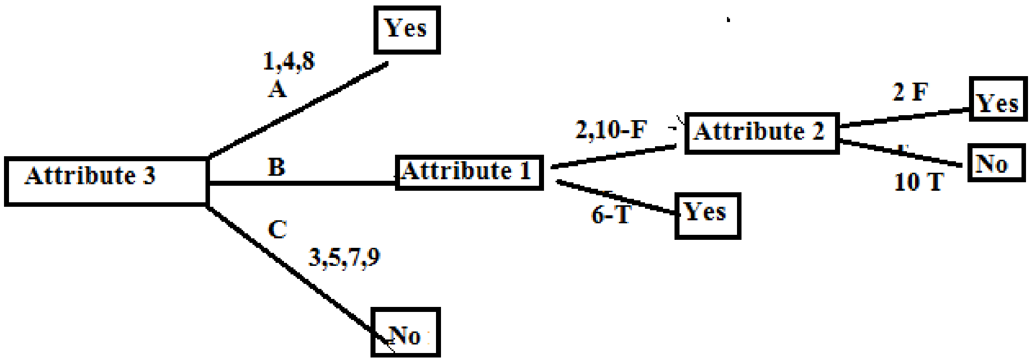
Info Gain = 1 – 0 – 3/10 * 0.9183 = 0.7245

Now,

Info gain of Attribute 3 is 0.7245 is the highest to be the root node. And Info Gain of Attribute 1 is 0.029 is the second level node.

After calculating Entropy and Information Gain the decision tree is (ID3).





This tree can be accepted but ID3 advocates for the previous one

3	a) What do you mean by gain and entropy? How it is used to build the decision tree?	[4+6]	CO2	L3

Mathematically, the information gain can be computed by the equation as follows:

Information Gain = $E(S_1) - E(S_2)$

– **$E(S_1)$** denotes the entropy of data belonging to the node before the split.

– **$E(S_2)$** denotes the weighted summation of the entropy of children nodes by considering the weights as the proportion of data instances falling in specific children nodes.

$$G = \text{Total Entropy} - |S_v|\ |S|\ \text{entropy}\ (S_v)$$

Entropy:

Entropy is a measure of the randomness in the information being processed. The higher the entropy, the harder it is to draw any conclusions from that information.

$$E = -\sum_{i=1}^{N} P_i \log_2 P_i$$

If targets values are only positive or only negative then the entropy will be 0

If in target values positive and negative values are equal, then the entropy will be 1

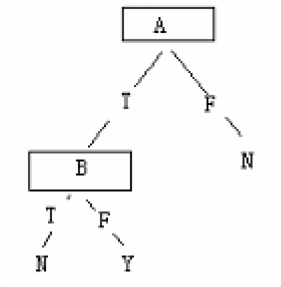
To Build decision tree:

1. An attribute with the highest information gain from a set should be selected as the parent (root) node.
2. Build child nodes for every value of attribute A.
3. Repeat iteratively until you finish constructing the whole tree.

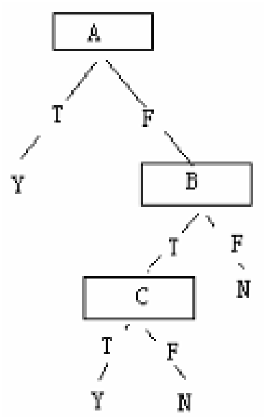
b) Construct the decision tree to represent the following Boolean functions:

i) $A \wedge \neg B$ ii) $A \vee [B \wedge C]$

i) $A \wedge \neg B$

<table>
<tr>
<td></td>
<td>


ii) A∨ [B∧ C]



</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>
Derive the Gradient Descent Rule. Write Stochastic Gradient Descent algorithm for training a linear unit.

- Consider locally weighted regression in which the target function f is approximated near x_q using a linear function of the form

$$\hat{f}(x) = w_0 + w_1 a_1(x) + \cdots + w_n a_n(x)$$

Where, $a_i(x)$ denotes the value of the i^{th} attribute of the instance x

- Derived methods are used to choose weights that minimize the squared error summed over the set D of training examples using gradient descent

$$E \equiv \frac{1}{2} \sum_{x \in D} (f(x) - \hat{f}(x))^2$$

Which led us to the gradient descent training rule

$$\Delta w_j = \eta \sum_{x \in D} (f(x) - \hat{f}(x)) a_j(x)$$

Where, η is a constant learning rate

Stochastic Gradient Descent Algorithm:

- **Stochastic** refers to the property of being well described by a random probability distribution
- Gradient Descent is a generic optimization algorithm capable of finding optimal solutions to a wide range of problems.
- The general idea is to tweak parameters iteratively in order to minimize the cost function.
- An important parameter of Gradient Descent (GD) is the size of the steps, determined by the learning rate hyperparameters. If the learning rate is too small, then the algorithm will have to go through many iterations to converge, which will take a long time, and if it is too high we may jump the optimal value.
- In SGD, it uses only a single sample, i.e., a batch size of one, to perform each
</td>
<td>[10]</td>
<td>CO3</td>
<td>L2</td>
</tr>
</table>

	iteration. The sample is randomly shuffled and selected for performing the iteration. **SGD algorithm:** for i in range (m) : $\theta_j = \theta_j - \alpha\left(\widehat{y}^i - y^i\right) x_j^i$			
5	a) List the appropriate problems for neural network learning. b) Show the derivation of back propagation training rule for output unit weights. **a) Appropriate Problems for NN Learning** 1. Instances are represented by many attribute-value pairs. 2. The target function output may be discrete-valued, real-valued, or a vector of several real-valued or discrete-valued attributes. 3. The training examples may contain errors. • Long training times are acceptable. 4. Fast evaluation of the learned target function may be required. 5. The ability of humans to understand the learned target function is not important. **b) derivation of Back propagation training rule for output units weight:** First Total Error $E_{total} = \sum \frac{1}{2}(target - output)^2$ Now calculating backward propagation error: 1. Adjusting weights between Output layer and hidden Layer For new weight of w_5 : Using partial differentiation $\frac{\partial E_{total}}{\partial w_5}$ is read as "the partial derivative of E_{total} with respect to w_5". You can also say "the gradient with respect to w_5". By applying the chain rule: $\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$ Partial derivative of logistic function = 1 minus output $Out_{O1} = 1/(1+e^{-neto1})$ = $out_{o1} = \frac{1}{1+e^{-net_{o1}}}$ Putting all together $\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$	[4+6]	CO1	L1
6	a) Explain Naïve Bayes classifier. b) The data set given shows whether an equipment is faulty or not.	[4+6]	CO2	L3

Color	Weight	Smoke	Faulty
Red	Heavy	No	Yes
Yellow	Light	No	No
Red	Heavy	Yes	Yes
Yellow	Heavy	Yes	Yes
Blue	Light	No	Yes
Yellow	Light	No	No
Blue	Heavy	Yes	No
Red	Heavy	No	Yes
Blue	Heavy	No	Yes
Yellow	Light	Yes	Yes

Using this "Equipment data set", decide the class of the equipment with <color=Yellow, Weight=Heavy, Smoke=No> using naive Bayes classifier

a) **Naïve base classifier:**
Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.
Bayes Theorem: P(A|B) = P(B|A).P(A) / P(B)
P(A|B) : Probability of A given that event B already occur
P(B|A) : Probability of B given that event A already occur

b)
Probability for Faulty Yes = 7/10 = 0.7
Probability for Faulty No = 3/10 = 0.3
To decide the class of <color=Yellow, Weight=Heavy, Smoke=No>
Using Bayes Theorem **P(A|B) = P(B|A).P(A) / P(B)**

Color	Yes	No
Red	3/7	0/3
Yellow	2/7	2/3
Blue	2/7	1/3

Weight	Yes	No
Heavy	5/7	1/3
Light	2/7	2/3

Smoke	Yes	No
Yes	3/7	1/3
No	4/7	2/3

Now, P(Yes| color=Yellow, Weight = Heavy, Smoke = No) =
P(Yes)*(P color=Yellow|Yes)* P(Weight=Heavy | Yes)* P(Smoke=No | Yes)
=7/10*2/7*5/7*4/7 = 0.0816
And P(No| color=Yello, Weight = Heavy, Smoke = No) =
P(No)*(P color=Yellow|No)* P(Weight=Heavy | No)* P(Smoke=No|No)
=3/10*2/3*1/3*2/3 = 0.044
So,

	P(Yes| color=Yello, Weight = Heavy, Smoke = No) > P(No| color=Yello, Weight = Heavy, Smoke = No) And thus class predicted is “Yes”			