Scheme of Evaluation
Internal Assessment Test 1 – Oct.2023

Sub:	Artificial Intelligence & Machine Learning							**Code:**	18CS71
Date:	30/10/2023	Duration:	90mins	Max Marks:	50	**Sem:**	VII	**Branch:**	ISE

Note: Answer Any Five Questions

Question #	Description	Marks Distribution	Max Marks
1	• Candidate elimination algorithm • Inductive bias explanation	5 M 5 M	10 M
2	• Calculating the overall entropy • Calculating individual attribute gain values • Identifying root node and sub nodes • Constructing final decision tree	2 M 4 M 2 M 2 M	10 M
3	• Initializing specific hypothesis and generic hypothesis • For each positive and negative instances: Finding the maximally specific and generic hypothesis	2M 8M	10 M
4	• Calculating the overall entropy • Calculating individual attribute gain values • Identifying root node and sub nodes • Constructing final decision tree	2 M 4 M 2 M 2 M	10 M
5.a.	• Construction of decision tree for each Boolean expression	2*3	6 M
5. b.	• Explanation of pre-pruning and post pruning	2*2	4 M
6	• Back propagation algorithm • Deriving the training rule from output layer	4 M 6 M	10 M

CELEBRATING 30 YEARS CMRIT CMR INSTITUTE OF TECHNOLOGY, BENGALURU. ACCREDITED WITH A++ GRADE BY NAAC

Internal Assessment Test 1 Solutions– Oct.2023

Sub:	Artificial Intelligence & Machine Learning							**Code:**	18CS71
Date:	01/10/2023	Duration:	90mins	Max Marks:	50	**Sem:**	VII	**Branch:**	ISE

Note: Answer Any Five Questions

1. Write the algorithm for Candidate Elimination and explain the Inductive bias.

Solution:

Candidate Elimination Algorithm:

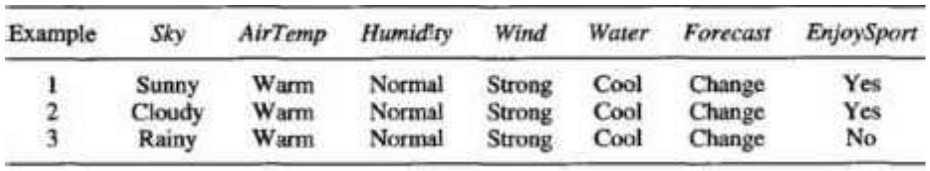
```
1. Initialize G to the set of maximally general hypotheses in H
2. Initialize S to the set of maximally specific hypotheses in H
3. For each training example d, do
a. If d is a positive example
        Remove from G any hypothesis inconsistent with d,
        For each hypothesis s in S that is not consistent with d,
            • Remove s from S
            • Add to S all minimal generalizations h of s such that h is consistent with d, and
              some member of G is more general than h
            • Remove from S, hypothesis that is more general than another hypothesis in S

b. If d is a negative example
    • Remove from S any hypothesis inconsistent with d
    • For each hypothesis g in G that is not consistent with d
            • Remove g from G
            • Add to G all minimal specializations h of g such that h is consistent with d, and
              some member of S is more specific than h
            • Remove from G any hypothesis that is less general than another in G
```

Inductive Bias:

1. A Biased Hypothesis Space

- Suppose we wish to assure that the hypothesis space contains the unknown target concept.
- The obvious solution is to enrich the hypothesis space to include every possible hypothesis.
- Consider EnjoySport example in which we restricted the hypothesis space to include only conjunctions of attribute values.



Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Cool	Change	Yes
2	Cloudy	Warm	Normal	Strong	Cool	Change	Yes
3	Rainy	Warm	Normal	Strong	Cool	Change	No

- Most specific hypothesis consistent with the first two examples
- It incorrectly covers the third (negative) training example
- The problem is that we have biased the learner to consider only conjunctive hypotheses.
- In this case we require a more expressive hypothesis space.

2. An unbiased learner

- The obvious solution to be a unbiased learner– design hypothesis space H to represent ***every teachable concept***;
    - It should capable of representing every possible subset of the instances X. In general, the set of all subsets of a set X is called the ***power-set*** of X.
    - In general, number of distinct subsets is $2_{|X|}$.
    - Thus, there are 2_{96}, or approximately distinct target concepts that could be defined over this instance space and that our learner might be called upon to learn.

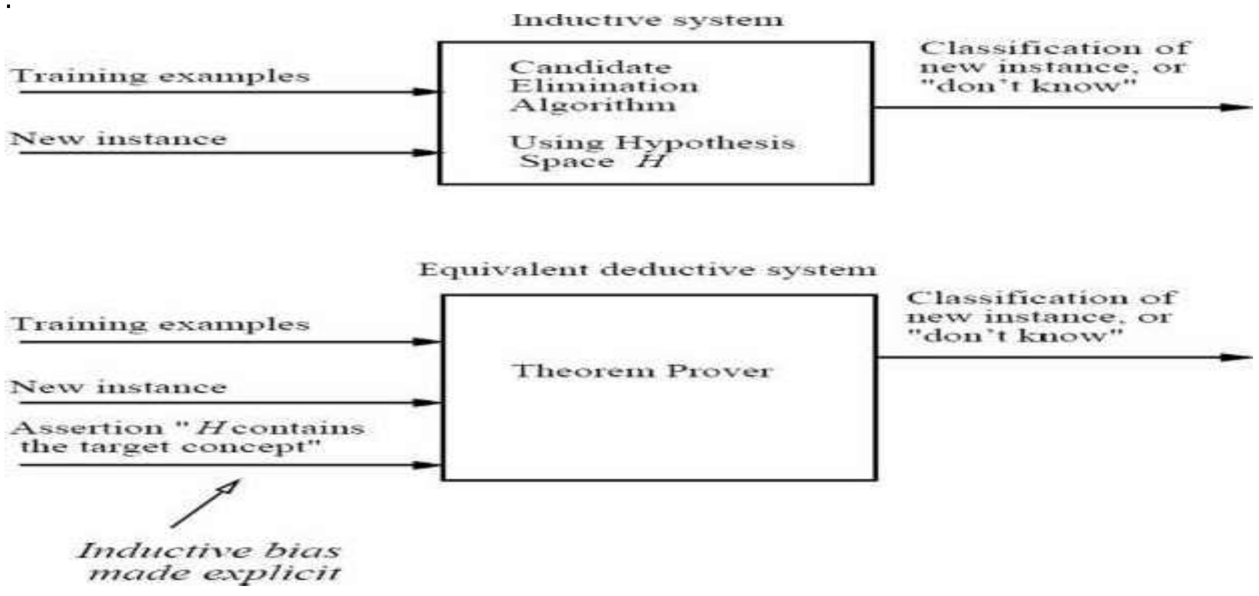
Our conjunctive hypothesis space is able to represent only 973 of these-a very biased hypothesis space indeed!

- Let us reformulate the *Enjoysport* learning task
- Let H' represent every subset of instances; that is, let H' correspond to the power set of X.
- One way to define such an H' is to allow arbitrary disjunctions, conjunctions, and negations of our earlier hypotheses.
- For instance, the target concept "Sky = Sunny or Sky = Cloudy" could then be described as $\langle Sunny, ?, ?, ?, ?, ? \rangle \vee \langle Cloudy, ?, ?, ?, ?, ? \rangle$

3. The Futility of Bias-Free Learning

- CEA generalizes observed training examples because it was biased by the implicit assumption that the target concept could be represented by a conjunction of attribute values.
- If this assumption is correct (and the training examples are error-free), its classification of new sample will also be correct.
- If this assumption is incorrect, however, it is certain that the CEA will mis-classify at least some instances from X.

.


Inductive system
Training examples
New instance
Candidate Elimination Algorithm
Using Hypothesis Space H
Classification of new instance, or "don't know"
Equivalent deductive system
Training examples
New instance
Assertion " H contains the target concept"
Theorem Prover
Classification of new instance, or "don't know"
Inductive bias made explicit

---

2. Construct decision tree using ID3 considering the following training examples.

Weekend	Weather	Parental Availability	Wealthy	Decision Class
H1	Sunny	Yes	Rich	Cinema
H2	Sunny	No	Rich	Tennis
H3	Windy	Yes	Rich	Cinema
H4	Rainy	Yes	Poor	Cinema
H5	Rainy	No	Rich	Home
H6	Rainy	Yes	Poor	Cinema
H7	Windy	No	Poor	Cinema
H8	Windy	No	Rich	Shopping
H9	Windy	Yes	Rich	Cinema
H10	Sunny	No	Rich	Tennis

Solution:

Step 1: Calculate the Entropy for the Whole Dataset

Entropy= $-\sum p_i \log_2(p_i)$

$= -\ (6/10)\log2(6/10)-(2/10)\log2(2/10)-(1/10)\log2(1/10)-(1/10)\log2(1/10) = 1.571$

Step 2: Calculate the Information Gain for Each Attribute

Information Gain= Entropy(S) - $\sum (|S||S_v| \text{Entropy}(S_v))$

Information Gain for Weekend:

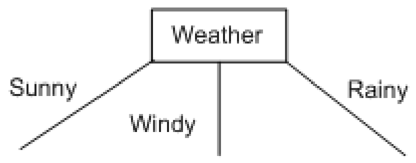
Since each instance of 'Weekend' is unique, each partition will contain exactly one instance, and the entropy of a single instance is always 0 (because there's no uncertainty).

Gain(S, weather) = 1.571 - ($|S_{sun}|$/10)*Entropy(S_{sun}) - ($|S_{wind}|$/10)*Entropy(S_{wind}) - ($|S_{rain}|$/10)*Entropy(S_{rain})
= 1.571 - (0.3)*Entropy(S_{sun}) - (0.4)*Entropy(S_{wind}) - (0.3)*Entropy(S_{rain})
= 1.571 - (0.3)*(0.918) - (0.4)*(0.81125) - (0.3)*(0.918) = 0.70

Gain(S, parental Availability) = 1.571 - ($|S_{yes}|$/10)*Entropy (S_{yes}) - ($|S_{no}|$/10)*Entropy (S_{no})
= 1.571 - (0.5) * 0 - (0.5) * 1.922 = 1.571 - 0.961 = 0.61

Gain(S, wealthy) = 1.571 - ($|S_{rich}|$/10)*Entropy (S_{rich}) - ($|S_{poor}|$/10)*Entropy (S_{poor})
= 1.571 - (0.7) * (1.842) - (0.3) * 0 = 1.571 - 1.2894 = 0.2816

Weather is having highest gain among the other attributes. So, root node is Weather



2nd level node identification:
There are three values for "Weather": Sunny, Windy, and Rainy.
E(Sunny) = -1/3log2(1/3)-2/3log2(2/3) = 0.918

Next, we set S to be S_{sunny} = {H1,H2,H10} (and, for this part of the branch, we will ignore all the other examples). In effect, we are interested only in this part of the table:

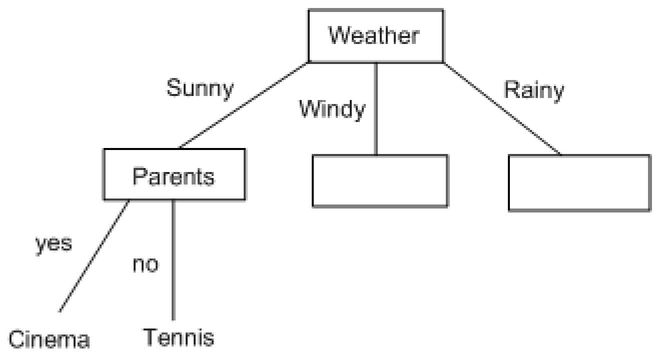
Weekend (Example)	**Weather**	**Parental availability**	**Wealthy**	**Decision (Category)**
H1	Sunny	Yes	Rich	Cinema
H2	Sunny	No	Rich	Tennis
H10	Sunny	No	Rich	Tennis

We need to calculate the values for Gain (S_{sunny}, parents) and Gain (S_{sunny}, money).

Gain(S_{sunny}, parental availability) = 0.918 - (|S_{yes}|/|S|)*Entropy(S_{yes}) - (|S_{no}|/|S|)*Entropy(S_{no})
= 0.918 - (1/3)*0 - (2/3)*0 = 0.918

Gain(S_{sunny}, wealthy) = 0.918 - (|S_{rich}|/|S|)*Entropy(S_{rich}) - (|S_{poor}|/|S|)*Entropy(S_{poor})
= 0.918 - (3/3)*0.918 - (0/3)*0 = 0.918 - 0.918 = 0

Thus, we select Parents attribute



E(Windy) = -3/4log2(3/4)-1/4log2(1/4) = 0.815

Next, we set S to be S_{windy} = {H3, H7, H8, H9}

Weekend (Example)	**Weather**	**Parental availability**	**Wealthy**	**Decision (Category)**
H3	Windy	Yes	Rich	Cinema
H7	Windy	No	Poor	Cinema
H8	Windy	No	Rich	Shopping
H9	Windy	Yes	Rich	Cinema

Gain(S_{Windy}, parental availability) = E(windy)- 2/4*E(Windy|Yes)-2/4* E(Windy|No)
= 0.815-0.5 = 0.315

Gain(S_{Windy}, wealthy) = E(windy)- 3/4*E(Windy|Rich)-1/4* E(Windy|Poor)
= 0.815-0.689 = 0.126
When Weather = Windy and Parental Availability = No the either Shopping or Cinema. Hence we need to further split the tree.
E(Windy&Parental Availability) = 1
Gain(Windy&Parental Availability|Wealthy) = E(Windy&Parental Availability) – 1/2 * E(Windy&Parental Availability|Wealthy=Rich)-1/2* E(Windy&Parental Availability|Wealthy=Poor) = 1

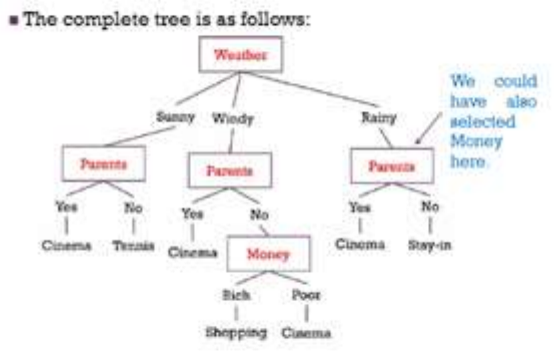
Thus, we select Money attribute.

E(Weather=Rainy) = -3/4log2(3/4)-1/4log2(1/4) = 0.815

Next, we set S to be S_{Rainy} = {H4, H5, H6}

Weekend (Example)	**Weather**	**Parental availability**	**Wealthy**	**Decision (Category)**
H4	Rainy	Yes	Poor	Cinema
H5	Rainy	No	Rich	Home
H6	Rainy	Yes	Poor	Cinema

Gain(Rainy|Parental Availability)=E(Rainy)-(2/3)*E(Rainy|Wealthy=Rich)-1/3* E(Rainy|Wealthy=Poor)
= 0.918
Thus, we can select either Parents or Money attribute



3. Consider the concept "Japanese Economy Car" with the following features

{Origin, Manufacturer, Color, Decade, Type}

Origin	Manufacturer	Color	Decade	Type	Example Type
Japan	Honda	Blue	1980	Economy	Positive
Japan	Toyota	Green	1970	Sports	Negative
Japan	Toyota	Blue	1990	Economy	Positive
USA	Chrysler	Red	1980	Economy	Negative
Japan	Honda	White	1980	Economy	Positive
Japan	Toyota	Green	1980	Economy	Positive
Japan	Honda	Red	1990	Economy	Negative

Apply Find-S and candidate elimination algorithms for the above dataset and list the observations.

Solution:

Find-S Algorithm:

S0=0,0,0,0,0
S1= Japan, Honda, Blue,1980,Economy +ve
2nd instance: Japan, Toyota, Green, 1970,Sports -ve
S2 = S1
3rd instance: Japan,Toyota,Blue,1990,Economy +ve
S3=Japan,?,Blue,?,Economy
4th instance: USA,Chrysler,Red,1980,Economy -ve
S4= S3=Japan,?, Blue,?,Economy
5th instance: Japan,Honda,White,1980,Economy +ve
S5= Japan,?,?,?,Economy
6th instance: Japan,Toyota,Green,1980,Economy +ve
S6= Japan,?,?,?, Economy
7th instance: Japan,Honda,Red,1990,Economy -ve
S7=S6

Candidate Elimination Algorithm:

S0= 0,0,0,0,0
G0=?,?,?,?,?

1st instance: S1= Japan, Honda, Blue,1980,Economy +ve
G1=?,?,?,?,?

2nd instance: Japan, Toyota, Green, 1970,Sports -ve
S2= Japan, Honda, Blue,1980,Economy
Try to make each ? with different possible pairs
G2= (USA,?,?,?,?)(?, Honda,?,?,?)(?, Chrysler,?,?,?)(?,?,Blue,?,?)(?,?, Red,?,?) (?,?,White,?,?) (?,?,?,1980,?) (?,?,?,1990,?)(?,?,?,?,Economy)

2nd, 4th, 7th and 9th pairs are consistant with S2 and rest all are inconsistent. Hence we eliminate them.
Final G2= (?,Honda,?,?,?) (?,?,Blue,?,?)(?,?,?, 1980,?)(?,?,?,?,Economy)

3rd instance: Japan,Toyota,Blue,1990,Economy +ve
S3=Japan,?,Blue,?,Economy
Make G3 more consistant pairs by removing less consistant pairs with Specific hypothesis.
G3 ={?,?,blue,?,?}{?,?,?,?,Economy} because first and 3rd pair is not consistant with specific hypothesis.

4th instance: USA,Chrysler,Red,1980,Economy -ve
S4= Japan,?, Blue,?,Economy
G4= (Japan,?,Blue,?,?)(?,Honda,Blue,?,?)(?,Toyota,Blue,?,?)(?,?,Blue,1970,?)(?,?,Blue,1990,?)(?,?,Blue,?,sports)(Japan,?,?,?,Economy)(?,Honda,?,?,Economy)(?,Toyota,?,?,Economy)(?,?,Blue,?,Economy)(?,?,Green,?,Economy)(?,?,White,?,Economy)(?,?,?,1970,Economy)(?,?,?,1980,Economy)
G4= (Japan,?,Blue,?,?)(Japan,?,?,?,Economy)(?,?,Blue,?,Economy)

5th instance: Japan,Honda,White,1980,Economy +ve
S5= Japan,?,?,?,Economy
G5= (Japan,?,?,?,Economy) bcz 1st and 3rd pairs are not consistant with 5th instance

6th instance: Japan,Toyota,Green,1980,Economy +ve
S6= Japan,?,?,?, Economy
G6=Japan,?,?,?,Economy

7th instance: Japan,Honda,Red,1990,Economy -ve
Whatever G6 and S6 we have got, it is consistant with negative instance. So S cannot be generalized and G cannot be specialized further.

Maximally specific hypothesis will be same for both Find-S and candidate elimination algorithms .Only generic hypothesis values will change in candidate elimination.

4. Create and explain the decision tree for the following transactions using ID3 algorithm.

Example	Colour	Toughness	Fungus	Appearance	Poisonous
1	Green	Soft	Yes	Wrinkled	Yes
2	Green	Hard	Yes	Smooth	No
3	Brown	Soft	No	Wrinkled	No
4	Brown	Soft	Yes	Wrinkled	Yes
5	Green	Soft	Yes	Smooth	Yes
6	Green	Hard	No	Wrinkled	No
7	Oran	Soft	Yes	Wrinkled	Yes

Solution:

Step 1: Calculate the overall Entropy of the Target Attribute (Poisonous)
Number of Poisonous (Yes): 4
Number of Not Poisonous (No): 3
Total: 7
E(Poisonous) = -((4/7)log2(4/7)+(3/7)log2(3/7)) ≈0.985

Step 2: Calculate the Information Gain for Each Attribute
For Colour:
Green: 3 Poisonous, 2 Not Poisonous, Brown: 1 Poisonous, 1 Not Poisonous, Orange: 1 Poisonous, 0 Not Poisonous

- E(Green) = -((2/4)log2(2/4)+(2/4)log2(2/4)) = 1
- E(Brown) = 1, E(Orange) = 0

Information Gain (Color) = E(Poisonous)-{(4/7)*E(Green)+(2/7)*E(Brown)+(1/7)*E(Orange)}
= 0.1280

For Toughness:

- E(Soft) = -((4/5)log2(4/5)+(1/5)log2(1/5)) = 0.7219
- E(Hard) = 0

Information Gain (Toughness) = E(Poisonous)-{(5/7)*E(Soft)+(2/7)*E(Hard)}
= 0.4955

For Fungus:

- E(Yes) = -((4/5)log2(4/5)+(1/5)log2(1/5)) = 0.7214
- E(No) = 0

Information Gain (Fungus) = E(Poisonous)-{(5/7)*E(Yes)+(2/7)*E(No)} = 0.4955

For Appearance:

- E(Wrinkled) = -((3/5)log2(3/5)+(2/5)log2(2/5)) = 0.9709
- E(Smooth) = 0

Information Gain (Appearance) = E (Poisonous)-{(5/7)*E(Wrinkled)+(2/7)*E(Smooth)}
= 0.2917

The attribute "Toughness" and "Appearance" has the highest information gain. We can choose either as root node. So, we have chosen "Toughness" as the root node of our decision tree.

```
[Toughness]
  |__ Soft --> [Poisonous = [1,3,4,5,7]]
  |__ Hard --> [Poisonous = [2,6]]
```

<u>2nd level Node Identification:</u>

Calculate the Entropy of the Target Attribute (Poisonous) based on Toughness
E(Soft) = -((4/5)log2(4/5)+(1/5)log2(1/5)) =0.7219

Information Gain (Soft=Colour) = E(Soft) – {(2/5)*E(Green)+(2/5)*E(Brown)+(1/5)*E(Orange)}
E(Green) = 0
E(Brown) =1, E(Orange) =0
Information Gain (Soft=Colour) = 0.7219-0-2/5*1-0 = 0.3219

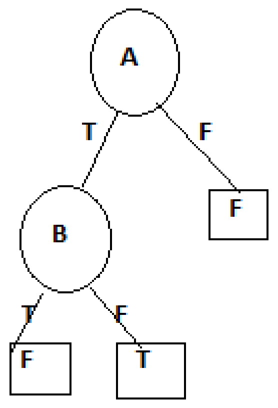
Information Gain (soft=Fungus) = E(Soft) – {(4/5)*E(Yes)+(1/5)*E(No)}
E(Yes) = 0,E(No) =0
Information Gain (Soft=Fungus) = 0.7219-0 = 0.7219

Information Gain (Soft=Appearance) = E(Soft) – {(4/5)*E(Wrinkled)+(1/5)*E(Smooth)}
E(Wrinkled) = -3/4log2(3/4)-1/4log2(1/4)=0.8112,E(Smooth) =0
Information Gain (Soft=Appearance) = 0.7219-4/5*0.8112 = 0.07294

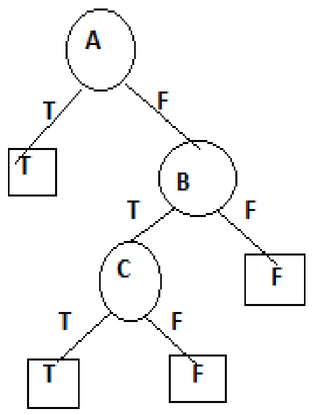
"Fungus" has the same highest information gain.

```
Toughness
  |__ Soft
       |__ [Fungus]
            |__ Yes  --> Yes
            |__ No  --> No
  |__ Hard  --> No
```

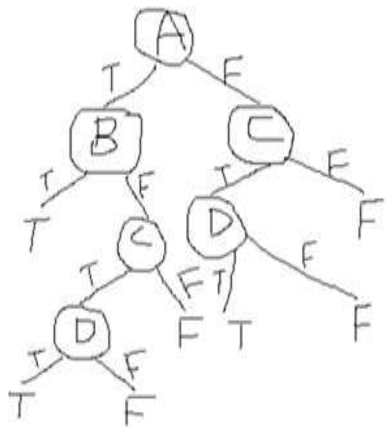
5.a. Construct Decision trees to represent the Boolean Functions:
a) A && ~ B b) A V [B && C] c) [A&&B] V [C&&D]
Solution:



(a)



(b)



(c)

5. b. Discuss the two approaches to prevent over fitting the data

Solution:

Approaches to avoiding overfitting in decision tree learning

- **Pre-pruning (avoidance):** Stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data
- **Post-pruning (recovery):** Allow the tree to overfit the data, and then post-prune the tree

1.1. Reduced-error pruning:

- ***Pruning*** a decision node consists of removing the subtree rooted at that node, making it a leaf node, and assigning it the most common classification of the training examples affiliated with that node
- Nodes are removed only if the resulting pruned tree performs no worse than-the original over the validation set.
- Reduced error pruning has the effect that any leaf node added due to coincidental regularities in the training set is likely to be pruned because these same coincidences are unlikely to occur in the validation set

1.2. Rule Post-Pruning :

Rule post-pruning is successful method for finding high accuracy hypotheses

Rule post-pruning involves the following steps:

1. Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing overfitting to occur.
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
4. Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

   For example, consider the decision tree above. The leftmost path of the tree in below figure is translated into the rule.

IF (Outlook = Sunny) ^ (Humidity = High) THEN *PlayTennis* = No

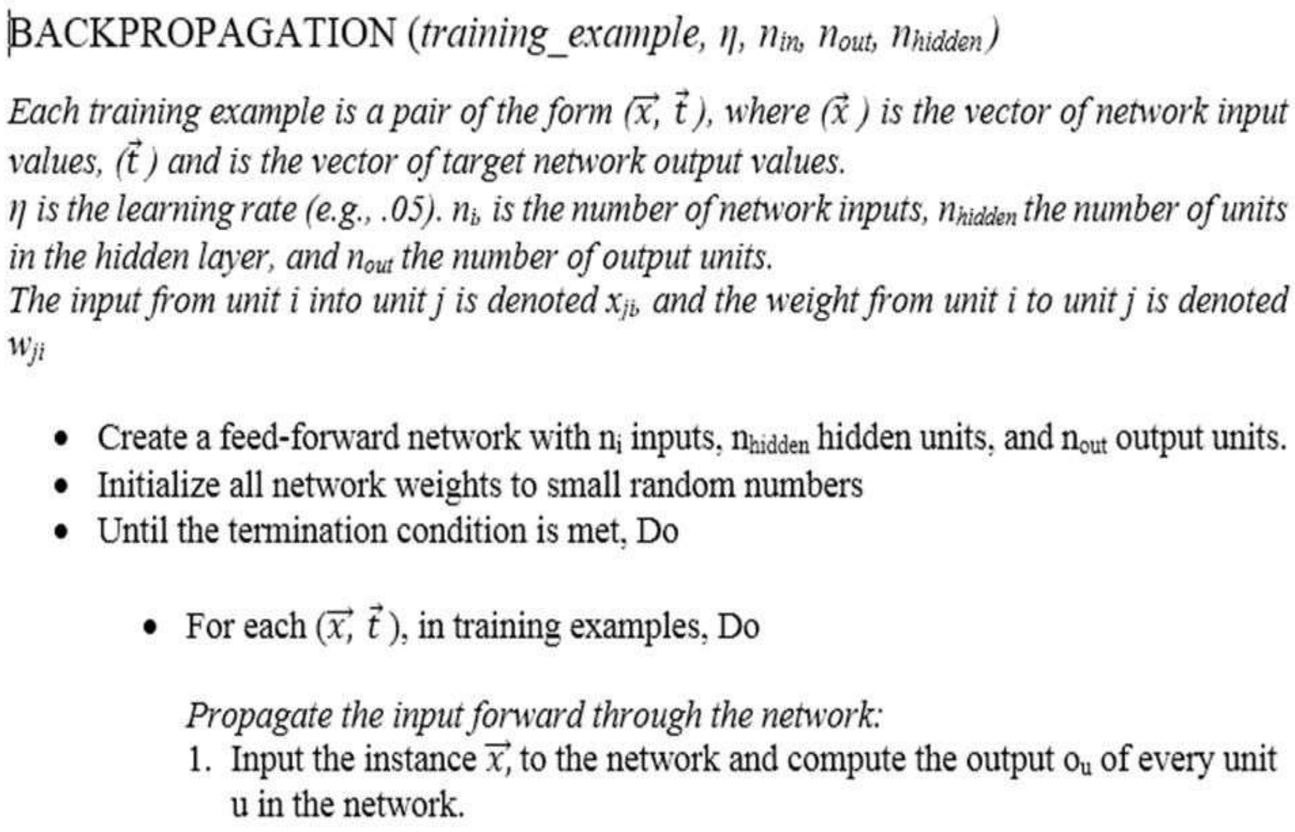
Given the above rule, rule post-pruning would consider removing the preconditions (Outlook = Sunny) and (Humidity = High)

It would select whichever of these pruning steps produced the greatest improvement in estimated rule accuracy.

---

6. Write an algorithm for back propagation which uses stochastic gradient descent method. Also derive the back propagation rule considering the output layer and training rule for output unit weights.

Solution:



BACKPROPAGATION (*training_example*, η, n_{in}, n_{out}, n_{hidden})

Each training example is a pair of the form $(\vec{x}, \vec{t})$, where $(\vec{x})$ is the vector of network input values, $(\vec{t})$ and is the vector of target network output values.
η is the learning rate (e.g., .05). n_i, is the number of network inputs, n_{hidden} the number of units in the hidden layer, and n_{out} the number of output units.
The input from unit i into unit j is denoted x_{ji}, and the weight from unit i to unit j is denoted w_{ji}

- Create a feed-forward network with n_i inputs, n_{hidden} hidden units, and n_{out} output units.
- Initialize all network weights to small random numbers
- Until the termination condition is met, Do

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

      Propagate the input forward through the network:
      1. Input the instance $\vec{x}$, to the network and compute the output o_u of every unit u in the network.

Propagate the errors backward through the network:

2. For each network output unit k, calculate its error term δ_k

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k)$$

3. For each hidden unit h, calculate its error term δ_h

$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} w_{h,k}\delta_k$$

4. Update each network weight w_{ji}

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

Where

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

Derivation

Case 1: Training Rule for Output Unit Weights.

- ***w**$_{ji}$* can influence the rest of the network only through ***net**$_j$*, ***net**$_j$* can influence the network only through ***o**$_j$*.

Therefore, we can invoke the chain rule again to write

$$\frac{\partial E_d}{\partial net_j} = \frac{\partial E_d}{\partial o_j}\frac{\partial o_j}{\partial net_j} \qquad \text{.....equ}(3)$$

To begin, consider just the first term in Equation (3)

$$\frac{\partial E_d}{\partial o_j} = \frac{\partial}{\partial o_j}\frac{1}{2}\sum_{k \in outputs}(t_k - o_k)^2$$

The derivatives $\frac{\partial}{\partial o_j}(t_k - o_k)^2$ will be zero for all output units k except when $k = j$. We therefore drop the summation over output units and simply set $k = j$.

$$\begin{aligned}\frac{\partial E_d}{\partial o_j} &= \frac{\partial}{\partial o_j}\frac{1}{2}(t_j - o_j)^2 \\ &= \frac{1}{2}2(t_j - o_j)\frac{\partial(t_j - o_j)}{\partial o_j} \\ &= -(t_j - o_j) \qquad \text{......equ}(4)\end{aligned}$$

Next consider the second term in Equation (3). Since $o_j = \sigma(net_j)$, the derivative $\frac{\partial o_j}{\partial net_j}$ is just the derivative of the sigmoid function, which we have already noted is equal to $\sigma(net_j)(1 - \sigma(net_j))$. Therefore,

$$\frac{\partial o_j}{\partial net_j} = \frac{\partial \sigma(net_j)}{\partial net_j}$$

$$= o_j(1 - o_j) \qquad \text{.......equ}(5)$$

Substituting expressions (4) and (5) into (3), we obtain

$$\frac{\partial E_d}{\partial net_j} = -(t_j - o_j)\ o_j(1 - o_j) \qquad \text{.......equ}(6)$$

and combining this with Equations (1) and (2), we have the stochastic gradient descent rule for output units

$$\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}} = \eta\ (t_j - o_j)\ o_j(1 - o_j)x_{ji} \qquad \text{.......equ}(7)$$