USN | | | | | | | | | |

Internal Assessment Test 1 – September 2018

Sub:	Machine Learning				Sub Code:	15CS73	Branch:	CSE
Date:	22.09.2018	Duration:	90 min's	Max Marks: 50	Sem / Sec:	7 / A,B,C		OBE

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	What is Machine Learning? List some applications of Machine Learning.	[04]	CO1	L2
(b)	Illustrate with an example to design a machine learning system.	[06]	CO1	L2
2 (a)	What is concept learning? Explain the process of finding maximally specific hypothesis for the below mentioned dataset.	[02+03]	CO1,4	L3

Restaurant	Meal	Day	Cost	Target Function
Sam's	breakfast	Friday	cheap	yes
Hilton	lunch	Friday	expensive	no
Sam's	lunch	Saturday	cheap	yes
Denny's	breakfast	Sunday	cheap	no
Sam's	breakfast	Sunday	expensive	no

		MARKS	CO	RBT
(b)	Compare Find-S algorithm with Candidate Elimination Algorithm.	[05]	CO1	L2
3	Write the Candidate Elimination Algorithm. Apply the algorithm for the following data to learn the concept of "**Japanese Economy Car**" and to find the version space.	[10]	CO1	L3

Origin	***Manufacturer***	***Color***	***Decade***	***Type***	***TargetFunction***
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*

		MARKS	CO	RBT
4 (a)	Explain Inductive bias through Candidate Elimination Algorithm.	[06]	CO1	L2
(b)	Discuss the issues in machine learning.	[04]	CO1	L2
5 .	Write the ID3 algorithm for creating decision tree with an example.	[10]	CO2	L2

6 (a) Write the Appropriate problems used for decision tree learning & List the applications of decision tree learning. [5] CO2 L2

(b) Consider the following set of training examples [5]

Instance	A1	A2	Classification
1	T	T	+
2	T	T	+
3	T	F	-
4	F	F	+
5	F	T	-
6	F	T	-

i. What is the entropy of this collection of training examples with respect to the target function classification?

ii. What is the information gain of A2 relative to these training examples? CO2 L2

7 Discuss the issues of ID3 in detail. [10] CO2 L2

8. Explain the inductive bias in decision tree learning [10] CO2 L2

15CS73 - Machine Learning

Solutions for IAT -1

1 (a) What is Machine Learning? List some applications of Machine Learning. [04]

Machine Learning is an approach or subset of Artificial Intelligence that is based on the idea that machines can be given access to data along with the ability to learn from it.

For example :Computers learning from medical records which treatments are the most effective for new disease.

1.Learning to recognize spoken words.
All of the most successful speech recognition systems employ machine learning in some form.
For example, the SPHINX system (e.g., Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal.

2. Learning to drive an autonomous vehicle.
Machine learning methods have been used to train computer-controlled vehicles to steer correctly when driving on a variety of road types. For example, the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars.

3.Learning to classify new astronomical structures.
Machine learning methods have been applied to a variety of large databases to learn

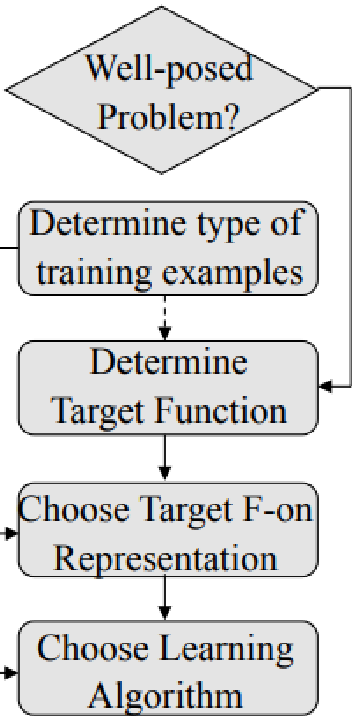
general regularities implicit in the data.

4. Learning to play world-class backgammon.

The most successful computer programs for playing games such as backgammon are based on machine learning algorithms.

1. (b) Illustrate with an example to design a machine learning system. [06]

Designing a machine learning system is explained with the help of learning to play checkers game in the world of checkers tournament. The performance measure adopted is : the percent of games it wins in this world tournament. The following figure lists the steps involved in the design of a learning system.


Well-posed Problem?
Determine type of training examples
Determine Target Function
Choose Target F-on Representation
Choose Learning Algorithm

- **Choosing the Training Experience**

The first design choice we face is to choose the type of training experience from which our system will learn. The type of training experience available can have a significant impact on success or failure of the learner. One key attribute is whether the training experience provides direct or indirect feedback regarding the choices made by the performance system.

A second important attribute of the training experience is the degree to which the learner

controls the sequence of training examples.

A third important attribute of the training experience is how well it represents the distribution

of examples over which the final system performance P must be measured.

- **Choosing the Target Function**

The next design choice is to determine exactly what type of knowledge will be learned and how this will be used by the performance program. Let us begin with a checkers-playing program that can generate the legal moves from any board state. The program needs only to learn how to choose the best move from among these legal moves.
The function **ChooseMove** and use the notation **ChooseMove : B $\rightarrow$ M** to indicate that this function accepts as input any board from the set of legal board states B and produces as output some move from the set of legal moves M.
Define the target value V(b) for an arbitrary board state b in B, as follows:
1. if b is a final board state that is won, then V(b) = 100
2. if b is a final board state that is lost, then V(b) = -100
3. if b is a final board state that is drawn, then V(b) = 0
4. if b is a not a final state in the game, then V(b) = V(b'), where b' is the best final board state that can be achieved starting from b and playing optimally until the end of the game

c) Choosing a Representation for the Target Function

for any given board state, the function c will be calculated as a linear combination of the following board features:
w0+w1*bp(b)+w2*rp(b)+w3*bk(b)+w4*rk(b)+w5*bt(b)+w6*rt(b)

bp(b): number of black pieces on board b
rp(b): number of red pieces on b
bk(b): number of black kings on b
rk(b): number of red kings on b
bt(b): number of red pieces threatened by black (i.e., which can be taken on black's next turn)
rt(b): number of black pieces threatened by red

where w_0 through W_6 are numerical coefficients, or weights, to be chosen by the learning algorithm.

- **Partial design of a checkers learning program:**

Task T: playing checkers
Performance measure P: percent of games won in the world tournament
Training experience E: games played against itself
Target function: V:Board $\rightarrow$ M
Target function representation

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

e) Choosing a Function Approximation Algorithm

In order to learn the target function f we require a set of training examples, each describing a specific board state b and the training value $V_{train}(b)$ for b.

f) ESTIMATING TRAINING VALUES

This rule for estimating training values can be summarized as:

Rule for estimating training values.

$$V_{train}(b) \leftarrow \hat{V}(Successor(b))$$

g) ADJUSTING THE WEIGHTS

LMS weight update rule.

For each training example $\langle b, V_{train}(b)\rangle$

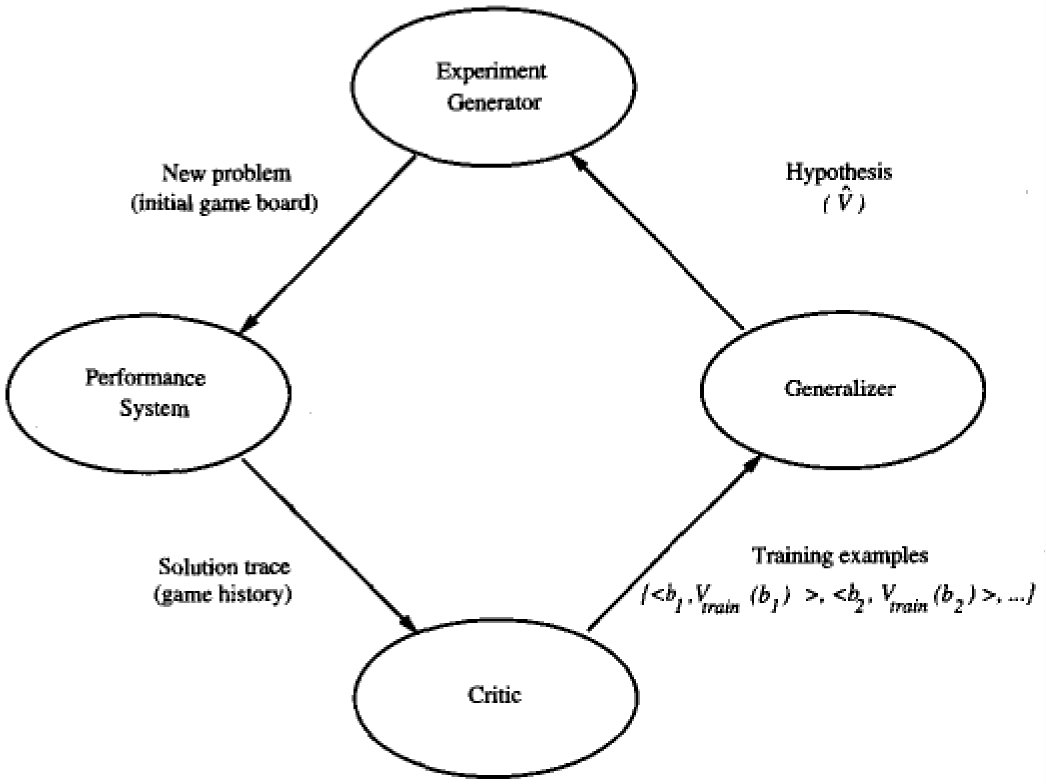
- Use the current weights to calculate $\hat{V}(b)$
- For each weight w_i, update it as

$$w_i \leftarrow w_i + \eta \ (V_{train}(b) - \hat{V}(b)) \ x_i$$

h) The Final Design

These four modules, summarized in Figure are as follows:

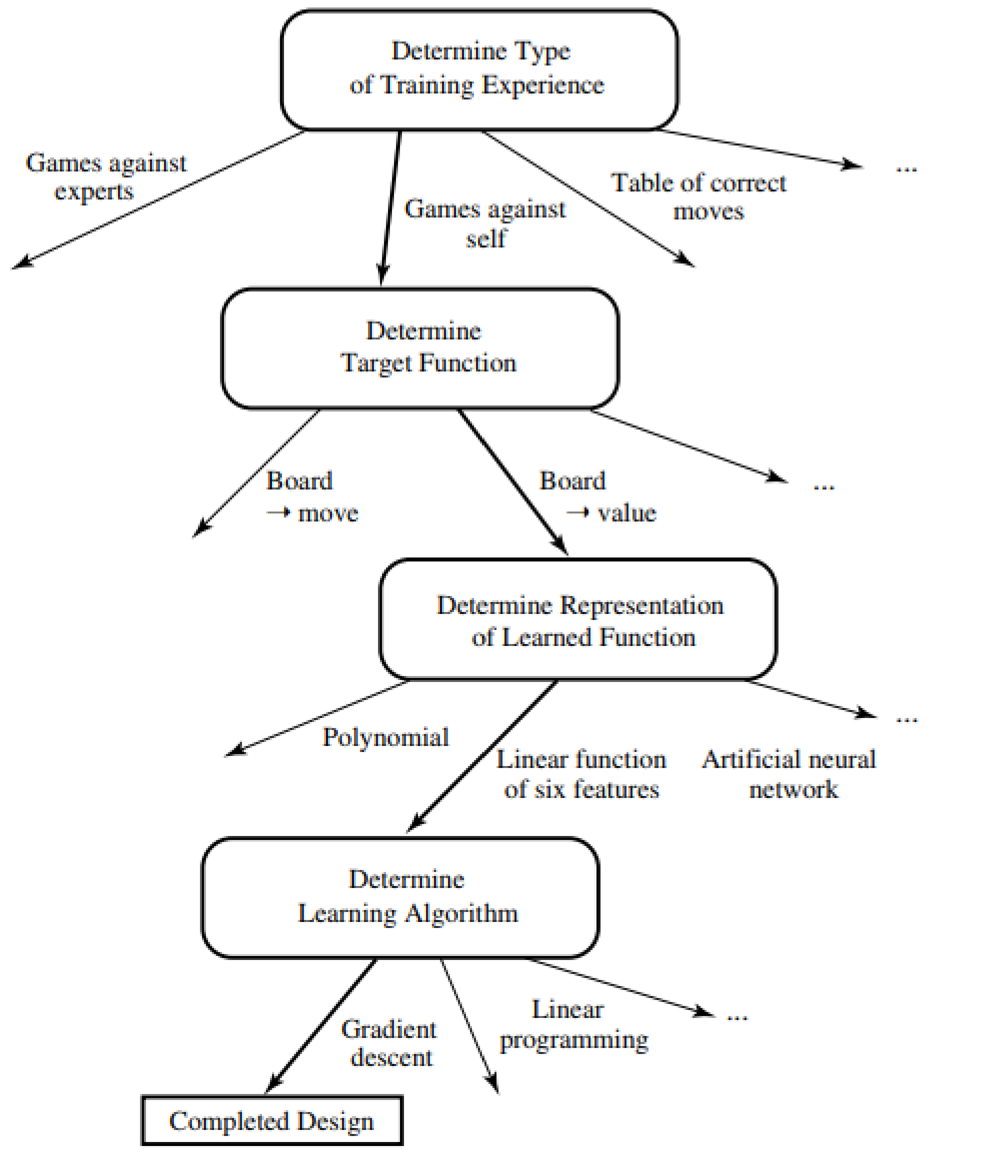
1.The **Performance System** is the module that must solve the given performance task, in this case playing checkers, by using the learned target function(s). It takes an instance of a new problem (new game) as input and produces a trace of its solution (game history) as output.


Experiment Generator
New problem (initial game board)
Hypothesis ($\hat{V}$)
Performance System
Generalizer
Solution trace (game history)
Training examples
$\{<b_1, V_{train}(b_1)>, <b_2, V_{train}(b_2)>, ...\}$
Critic

2.The **Critic** takes as input the history or trace of the game and produces as output a set of training examples of the target function. As shown in the diagram, each training example in this case corresponds to some game state in the trace, along with an estimate Vtraio,f the target function value for this example.
3.The **Generalizer** takes as input the training examples and produces an output hypothesis that is its estimate of the target function. It generalizes from the specific training xamples,hypothesizing a general function that covers these examples and other cases beyond the training examples.

4.The **Experiment Generator** takes as input the current hypothesis (currently learned function) and outputs a new problem (i.e., initial board state) for the Performance System to explore. Its role is to pick new practice problems that will maximize the learning rate of the overall system.

Summary of choices in designing the checkers learning program.


Determine Type of Training Experience
Games against experts
Games against self
Table of correct moves
...
Determine Target Function
Board → move
Board → value
...
Determine Representation of Learned Function
Polynomial
Linear function of six features
Artificial neural network
...
Determine Learning Algorithm
Gradient descent
Linear programming
...
Completed Design

2 (a) What is concept learning? Explain the process of finding maximally specific hypothesis for the below mentioned dataset. [02+03]

Concept Learning : Inferring a boolean-valued function from training examples of its input and output.

Restaurant	Meal	Day	Cost	Target Function
Sam's	breakfast	Friday	cheap	yes
Hilton	lunch	Friday	expensive	no
Sam's	lunch	Saturday	cheap	yes
Denny's	breakfast	Sunday	cheap	no
Sam's	breakfast	Sunday	expensive	no

Many concept learning algorithms utilize general-to-specific ordering of hypotheses
• General-to-Specific Ordering: –

h1 precedes (is more general than) h2, This is written as $h1 >_g h2$ [05]

1.Initialize hypothesis h to most specific hypothesis in the hypothesis space H
h0 =(∅, ∅, ∅, ∅)

2.For each positive training instance in the given training example, check whether the hypothesis covers the training instance or not .If it is not covered then replace each attribute by the next more general constraint that covers the given instance.

h1 = (Sam's, breakfast, Friday, cheap)

h2 =(Sam's, ?, ?,cheap)

Maximally specific hypothesis is h2.

hence **Restaurant** =Sam's and **Cost** = cheap then **Target Function** => yes

2(b) Compare Find-S algorithm with Candidate Elimination Algorithm.

- Find-S outputs a hypothesis from H that is consistent with the training examples.
- Find-S outputs one of the many hypotheses from H that might fit with the training data.
- Find-S is sensitive to noise that is (almost always) present in training examples.
- There is no guarantee that h returned by Find-S is the only h that fits the data, because it considers only positive training examples.

- Candidate Elimination Algorithm(CEA) addresses several of the limitations of Find-S Algorithm.

- The subset of all hypotheses is called the ***Version Space*** (VS) is generated with respect to hypotheses space H and the training examples D .

- Candidate Elimination Algorithm outputs the set of all hypotheses consistent with the training examples including the negative training examples.
- CEA uses more_ general_than partial ordering to provide the consistent hypotheses.

3 **Write the Candidate Elimination Algorithm.** [10]

```
G :maximally general hypotheses in H

S :maximally specific hypotheses in H

For each training example d=<x,c(x)>

Case 1 : If d is a positive example
    Remove from G any hypothesis that is 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
        consistent with d
        Some member of G is more general than h
            Remove from S any hypothesis that is more general than another
            hypothesis in S

Case 2: If d is a negative example
    Remove from S any hypothesis that is 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
      consistent with d
      Some member of S is more specific than h
            Remove from G any hypothesis that is less general than another
            hypothesis in G
```

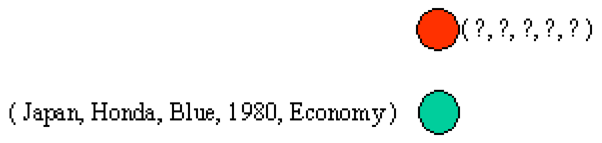
Apply the algorithm for the following data to learn the concept of "Japanese Economy Car" and to find the version space.

Origin	***Manufacturer***	***Color***	***Decade***	***Type***	***Target Function***
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*

<u>Solution:</u>

1. **Positive Example**: (Japan, Honda, Blue, 1980, Economy)

Initialize G to a singleton set that includes everything. Initialize S to a singleton set that includes the first positive example.	G = { (?, ?, ?, ?, ?) }
	S = { (Japan, Honda, Blue, 1980, Economy) }

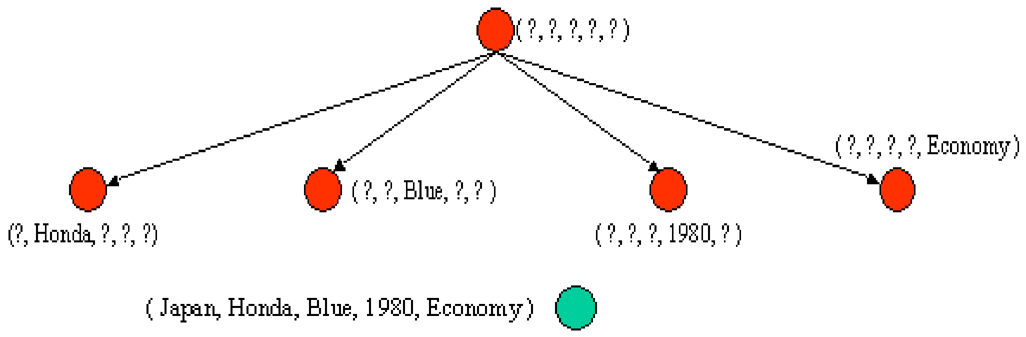


- These models represent the most general and the most specific heuristics one might learn.
  The actual heuristic to be learned, "Japanese Economy Car", probably lies between them somewhere within the version space.

2. **Negative Example**: (Japan, Toyota, Green, 1970, Sports)

Specialize G to exclude the negative example.

G =	{ (?, Honda, ?, ?, ?), (?, ?, Blue, ?, ?), (?, ?, ?, 1980, ?), (?, ?, ?, ?, Economy) }
S =	{ (Japan, Honda, Blue, 1980, Economy) }

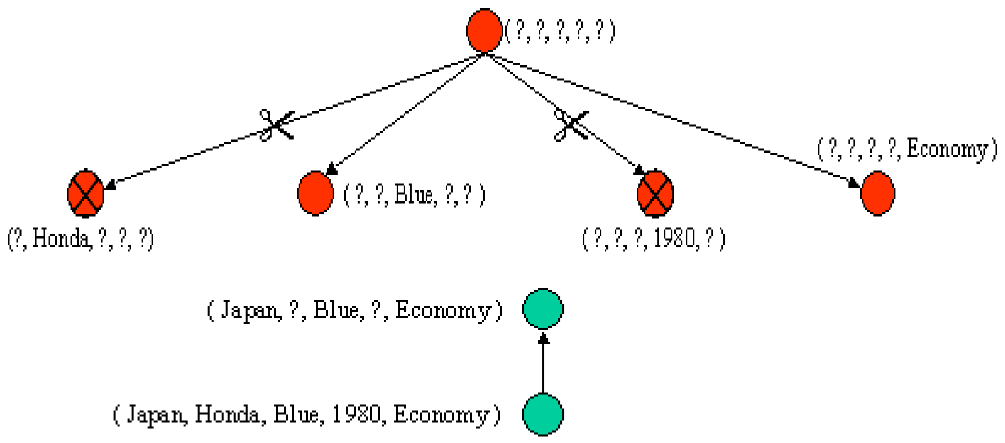


Refinement occurs by generalizing S or specializing G, until the heuristic hopefully converges to one that works well.

3. **Positive Example**: (Japan, Toyota, Blue, 1990, Economy)

Prune G to exclude descriptions inconsistent with the positive example. (Prune = ✂)
Generalize S to include the positive example.

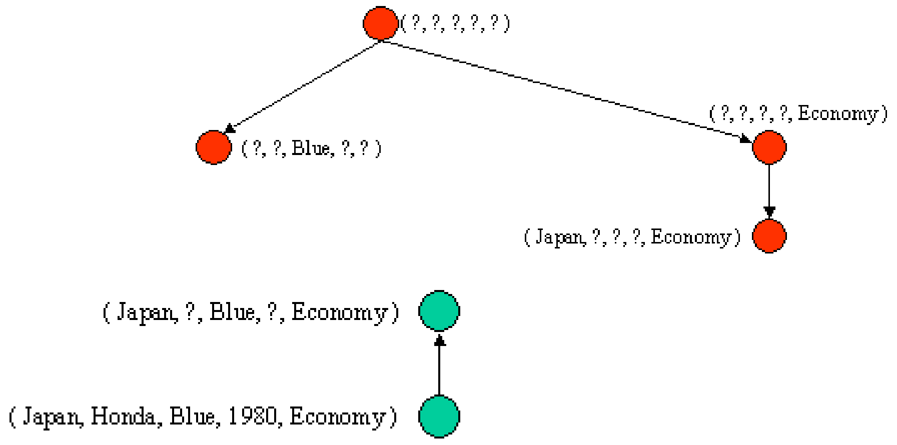
G =	{ (?, ?, Blue, ?, ?), (?, ?, ?, ?, Economy) }
S =	{ (Japan, ?, Blue, ?, Economy) }



4. **Negative Example**: (USA, Chrysler, Red, 1980, Economy)

Specialize G to exclude the negative example (but stay consistent with S)

G =	{ (?, ?, Blue, ?, ?), (Japan, ?, ?, ?, Economy) }
S =	{ (Japan, ?, Blue, ?, Economy) }

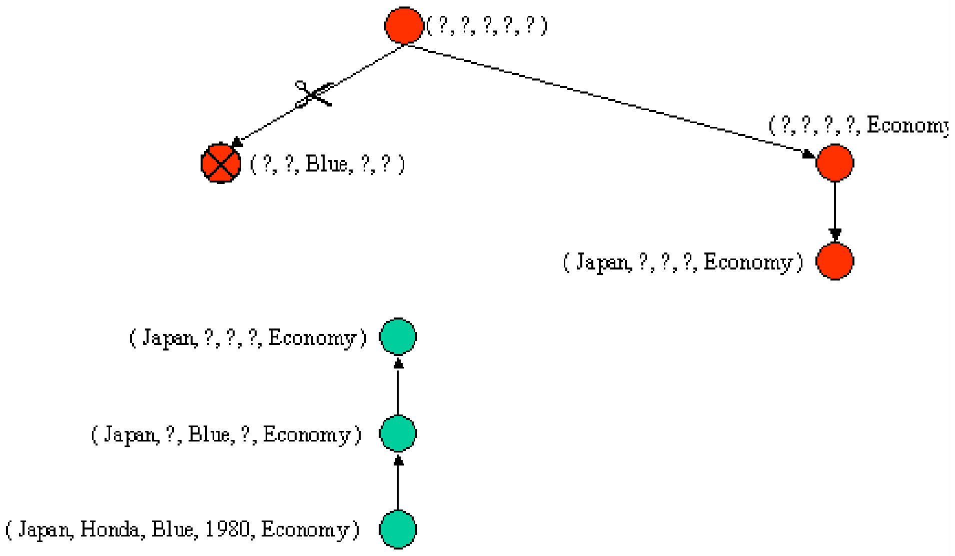


(?, ?, ?, ?, ?)
(?, ?, Blue, ?, ?)
(?, ?, ?, ?, Economy)
(Japan, ?, ?, ?, Economy)
(Japan, ?, Blue, ?, Economy)
(Japan, Honda, Blue, 1980, Economy)

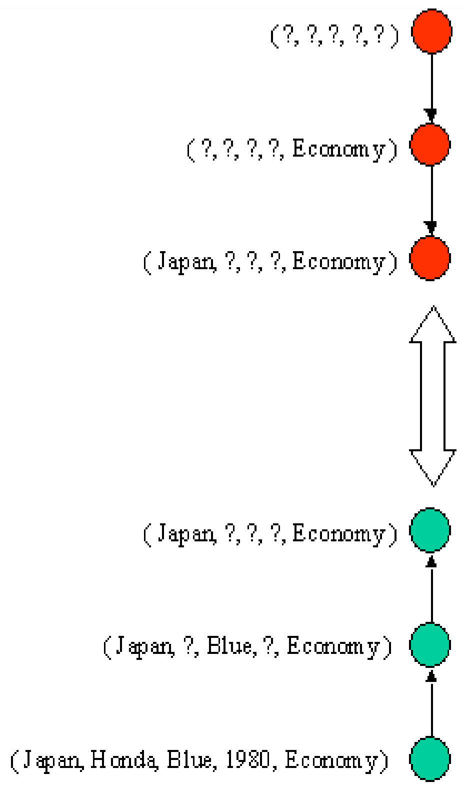
5. **Positive Example**: (Japan, Honda, White, 1980, Economy)

Prune G to exclude descriptions inconsistent with positive example.
Generalize S to include positive example.

G = { (Japan, ?, ?, ?, Economy) }
S = { (Japan, ?, ?, ?, Economy) }


(?, ?, ?, ?, ?)
(?, ?, Blue, ?, ?)
(?, ?, ?, ?, Economy
(Japan, ?, ?, ?, Economy)
(Japan, ?, ?, ?, Economy)
(Japan, ?, Blue, ?, Economy)
(Japan, Honda, Blue, 1980, Economy)

G and S are singleton sets and S = G.
Converged.
No more data, so algorithm stops.


(?, ?, ?, ?, ?)
(?, ?, ?, ?, Economy)
(Japan, ?, ?, ?, Economy)
(Japan, ?, ?, ?, Economy)
(Japan, ?, Blue, ?, Economy)
(Japan, Honda, Blue, 1980, Economy)

4 (a) Explain Inductive bias through Candidate Elimination Algorithm. [06]

Inductive inference is the process of reaching a general conclusion from specific examples. The general conclusion should apply to unseen examples.

In the **CANDIDATE-ELIMINATION** algorithm, we restricted the hypothesis space to include only **conjunctions of attribute values**.
The hypothesis space is unable to represent simple **disjunctive** target concept
For example :

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

Suppose the hypothesis h0 in the hypothesis space H is **h0=(?,Warm,Normal,Strong,Cool,Change)**

This hypothesis is maximally specific and consistent with the first 2 examples.
It covers the first 2 examples but incorrectly covers the third example. Because the third one is negative example.
This is because of the bias (considered only **conjunction** hypotheses).

We have not considered the simple **disjunctive** target concept such as:
Sky=Sunny **or** Sky = Cloudy.

Unbiased learner

The solution to the problem of assuring that the target concept is in the hypothesis space H is to provide a hypothesis space capable of representing every concept.
We need to cover every possible subset of the instance X .In general , the set of all subsets of a set X is called the power set of X.
For Example : In ***EnjoySport*** learning task, there are 6 attributes:
Sky = {sunny,cloudy,rain} = >3 different values
AirTemp= {warm,cold} = > 2 different values
Humidity = {} = > 2 different values
Wind={Strong, Weak} = > 2 different values
Water ={Warm,Cold} = > 2 different values
Forecast={Same,Change}= > 2 different values

The total no.of attributes in the instance space =3 x 2 x 2 x 2 x 2 x 2 = 96
The possible concept that can be defined over this set is 2^{96} (approximately $=10^{28}$ distinct target conept).but our conjunctive hypothesis space is able to represent only 973 (1 + 4x3x3x3x3x3=973 semantically distinct hypotheses).
Let us reformulate the Enjoysport learning task in an unbiased way by defining a new hypothesis space H' that can 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.

One advantage of viewing inductive inference systems in terms of their inductive bias is that it provides a nonprocedural means of characterizing their policy for generalizing beyond the observed data. A second advantage is that it allows comparison of different learners according to the strength of the inductive bias they employ. Consider, for example the following **three learning algorithms**, which are listed from weakest to strongest bias.

1. **ROTE-LEARNER:**
   Learning corresponds simply to storing each observed training example in memory. Subsequent instances are classified by looking them up in memory. If the instance is found in memory, the stored classification is returned. Otherwise, the system refuses to classify the new instance.
2. **CANDIDATE-ELIMINATION Algorithm**:
   New instances are classified only in the case where all members of the current version space agree on the classification. Otherwise, the system refuses to classify the new instance.
3. **FIND-S**:
   This algorithm, described earlier, finds the most specific hypothesis consistent with the training examples. It then uses this hypothesis to classify all subsequent instances.

The ROTE-LEARNER has no inductive bias. The classifications it provides for new instances follow deductively from the observed training examples, with no additional assumptions required.
The CANDIDATE-ELIMINATlON Algorithm has a stronger inductive bias: that the target concept can be represented in its hypothesis space. Because it has a stronger bias, it will classify some instances that the ROTE LEARNER will not. Of course the correctness of such classifications will depend completely on the correctness of this inductive bias.
The FIND-S algorithm has an even stronger inductive bias. In addition to the assumption that the target concept can be described in its hypothesis space, it has an additional inductive bias assumption: that all instances are negative instances unless the opposite is entailed by its other know1edge

4 (b) Discuss the issues in machine learning. [04]

It involves searching a very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by the learner.
1. Algorithms for learning general target function from specific training examples.
2. Setting at which the particular algorithm converge to the desired function, given sufficient training data
3. Algorithm that perform best for the given types of problems and its representations
4. The Amount of training data required.

5. The prior knowledge needed to guide the process of generalizing from example.

6. The best strategy for choosing a useful next training experience, to alter the complexity of the learning Problem.
7. The best way to reduce the learning task to one or more function approximation problems.

8.The representation to improve its ability to represent and learn the target function

5 Write the ID3 algorithm for creating decision tree with an example. [10]

ID3 - Algorithm

```
ID3(Examples, TargetAttribute, Attributes)
•  Create a Root node for the tree
•  If all Examples are positive, Return the single-node tree Root, with label =
•  If all Examples are negative, Return the single-node tree Root, with label
•  If Attributes is empty, Return the single-node tree Root, with label = mos
   value of TargetAttribute in Examples
•  Otherwise Begin
   – A ← the attribute from Attributes that best classifies Examples
   – The decision attribute for Root ← A
   – For each possible value, vi, of A,
      • Add a new tree branch below Root, corresponding to the test A =
      • Let $Examples_{vi}$ be the subset of Examples that have value vi for A
      • If $Examples_{vi}$ is empty
         – Then below this new branch add a leaf node with label = mos
           value of TargetAttribute in Examples
         – Else below this new branch add the subtree
              ID3($Examples_{vi}$, TargetAttribute, Attributes – {A})
•  End
•  Return Root
```

Example: [5]

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**

$$\text{Entropy(S)} \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

p_i is the proportion of S belonging to the class i

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

Values(A) is the set of all possible values for A

Entropy(S) = – (9/14) . $\log_2$(9/14) – (5/14) . $\log_2$(5/14) = 0.940

Which attribute to test at the root?

▪ ***Gain*(*S*, *Outlook*) = 0.246**

▪ ***Gain*(*S*, *Humidity*) = 0.151**

▪ ***Gain*(*S*, *Wind*) = 0.048**

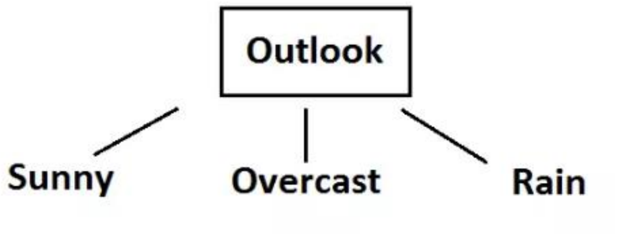
▪ ***Gain*(*S*, *Temperature*) = 0.029**

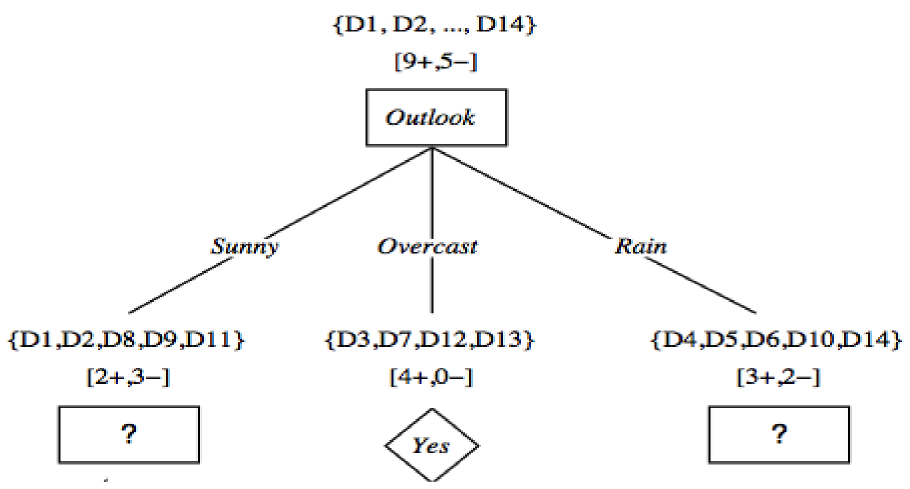
▪ ***Outlook* provides the best prediction for the target**

Lets grow the tree:

▪ **add to the tree a successor for each possible value of *Outlook***

▪ **partition the training samples according to the value of *Outlook***





Sunny outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Overcast outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
3	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes

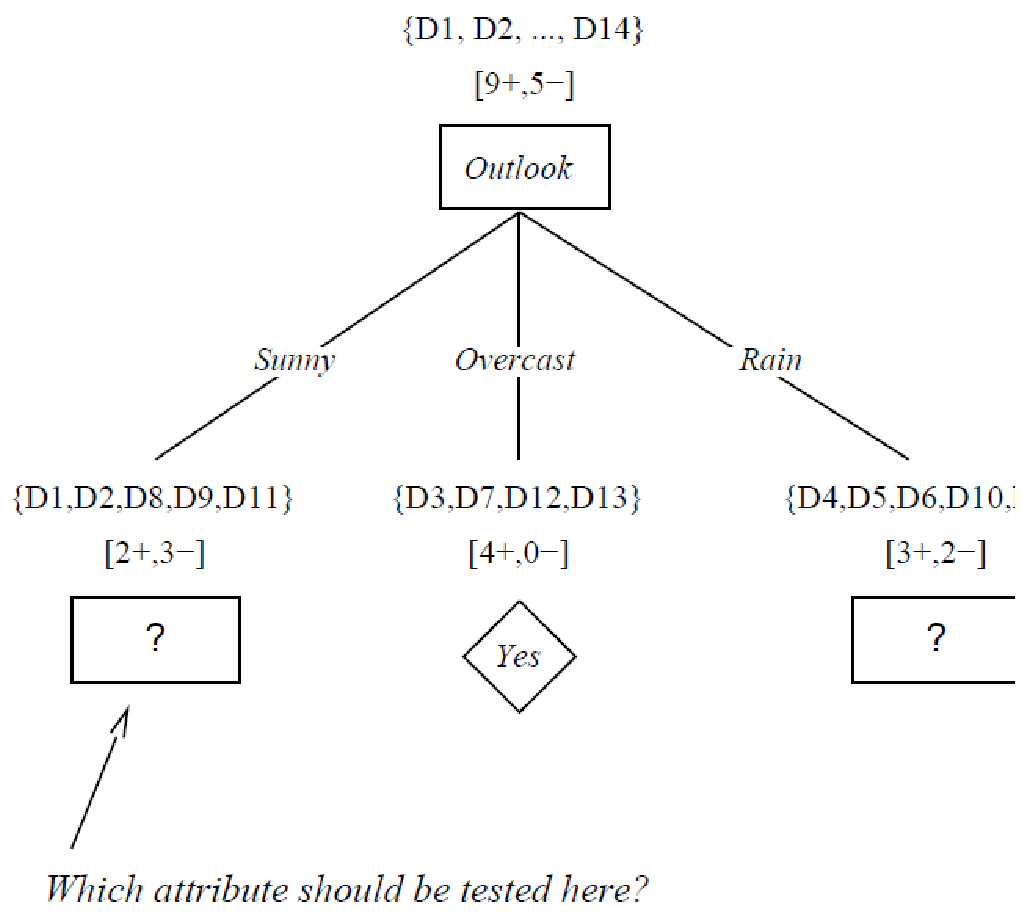
Rain outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes

14	Rain	Mild	High	Strong	No

Working on *Outlook=Sunny* node:
***Gain*(*SSunny*, *Humidity*) = 0.970 − 3/5 * 0.0 − 2/5 * 0.0 = 0.970**
***Gain*(*SSunny*, *Temp.*) = 0.970 − 2/5* 0.0 − 2/5 * 1.0 − 1/5 * 0.0 = 0.570**
***Gain*(*SSunny*, *Wind*) = 0.970 − 2/5 * 1.0 − 3.5* 0.918 = 0 .019**

- ***Humidity provides the best prediction for the target***
- **Lets grow the tree:**
  - **add to the tree a successor for each possible value of *Humidity***
  - **partition the training samples according to the value of *Humidity***

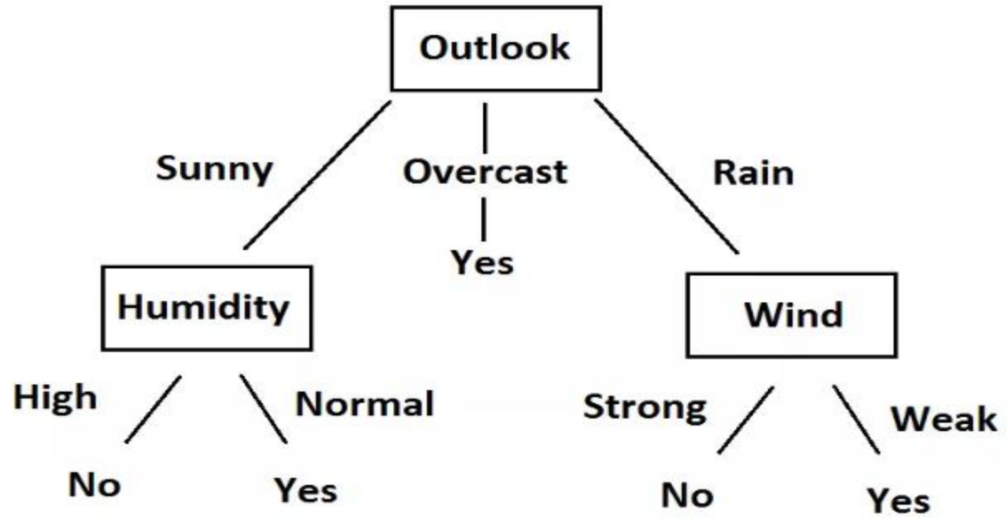


S_{sunny} = {D1,D2,D8,D9,D11}

$$Gain\ (S_{sunny}, Humidity) = .970 - (3/5)\ 0.0 - (2/5)\ 0.0 = .970$$

$$Gain\ (S_{sunny}, Temperature) = .970 - (2/5)\ 0.0 - (2/5)\ 1.0 - (1/5)\ 0.0$$

$$Gain\ (S_{sunny}, Wind) = .970 - (2/5)\ 1.0 - (3/5)\ .918 = .019$$



2.Decision will always be yes if outlook were overcast.

High Humidity on Decision – All No

Day	Humidity	Temp.	Wind	Decision
1	**High**	Hot	Weak	No
2	**High**	Hot	Strong	No

8	**High**	Mild	Weak	No

Normal Humidity on Decision –All Yes

Day	**Humidity**	**Temp.**	**Wind**	**Decision**
9	Normal	Cool	Weak	Yes
11	Normal	Mild	Strong	Yes

Weak Wind on decision – All yes ,returns the label as yes

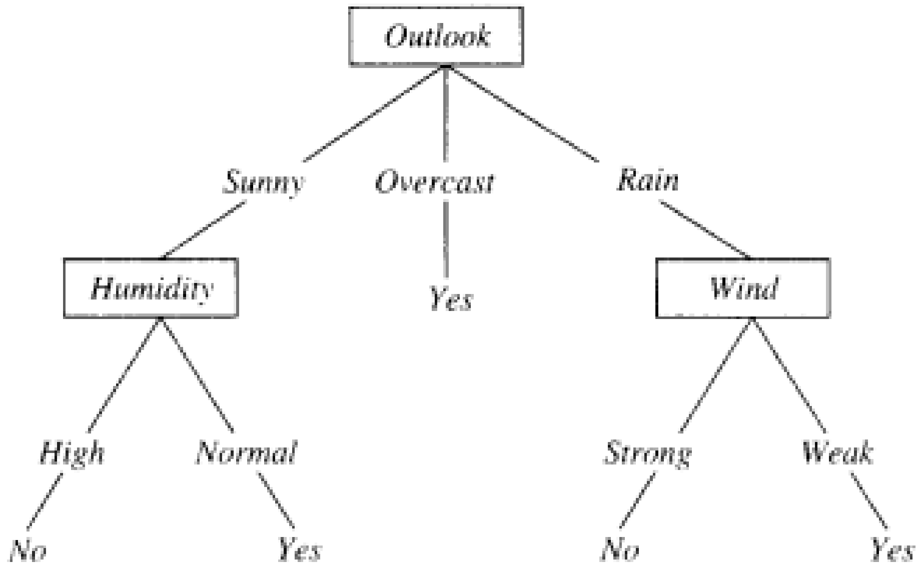
Day	**Outlook**	**Temp.**	**Humidity**	**Wind**	**Decision**
4	Rain	Mild	High	Weak	**Yes**
5	Rain	Cool	Normal	Weak	**Yes**
10	Rain	Mild	Normal	Weak	**Yes**

Strong Wind on Decision -All No, returns the label as No

Day	**Outlook**	**Temp.**	**Humidity**	**Wind**	**Decision**

6	Rain	Cool	Norm al	Strong	No
14	Rain	Mild	High	Strong	No

Hence the final Decision tree will be :



6 (a) **Write the Appropriate problems used for decision tree learning & List the [5] applications of decision tree learning.**

Decision tree learning is generally best suited to the problems:

- Instances are represented by attribute-value pairs
- The target function has discrete output values
- Disjunctive description may be required
- The training data may contain errors
- The training data may contain missing attribute values

Applications of decision tree learning:

1. Decision tree learning is applied to problems such as learning to classify medical patients by their diseases,equipment malfunctions by their causes
2. classifying the loan applicants by their likelihood of defaulting on payments
3. Prediction problems.

Consider the following set of training examples **[5]**

6.(b)

Instance	**A1**	**A2**	**Classification**
1	T	T	+
2	T	T	+
3	T	F	-
4	F	F	+
5	F	T	-
6	F	T	-

iii. What is the entropy of this collection of training examples with respect to the target function classification?

iv. What is the information gain of A2 relative to these training examples?

Solution:

i) S is the given collection ,

$$\text{Entropy(S)} \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

p_i is the proportion of S belonging to the class i

Entropy(S) = 1 ,*when equal number* of positive and negative examples.

ii) Information gain(S,A2)

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

Gain(S,A2) = 1- { (4/6)*Entropy(T) + (2/6)* Entropy(F)}
= 1- {(4/6)*(equal no.of + and -) + (2/6)*(equal no.of + and -)}
= 1-{(4/6)*1 + (2/6)*1}
= 1-{(4/6) + (2/6)}
= 1 – 1

Gain(S,A2) = 0

7 **Discuss the issues of ID3 in detail.** **[10]**

Issues in Decision Tree Learning: (Explanation must be given to all points)

- Determining how deeply to grow the decision tree
- Handling continuous attributes
- Choosing an appropriate attribute-selection measure
- Handling training data with missing attribute-values
- Improving computational efficiency

8. **Explain the inductive bias in decision tree learning** **[10]**

Inductive bias is the set of assumptions that along with the training data justify the classifications assigned by the learner to future instances.

Given H as the power set of instances X

• ID3 has preference of short trees with high information gain attributes near the root.

• ID3 has preference for certain hypotheses over others, with no hard restriction on the hypotheses space

Occam's Razor

• Prefer the simplest hypothesis that fits the data

• Argument in favor –

 • A short hypothesis that fits data unlikely to be a coincidence

 • A long hypothesis that first data might be a coincidence

• Argument Opposed –

 • There are many ways to define small sets of hypotheses

 • Two different hypotheses from the same training examples possible when applied by two learners that perceive these examples in terms of different internal representations

----