Internal Assessment Test 1 – NOV 2021

Machine Learning Solution

	MARKS	CO	RBT
1 (a) What you mean by Well Posed learning Problem? Explain with an example.	[05]	CO1	L2

Definition: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

To have a well-defined learning problem, three features needs to be identified:

1. The class of tasks
2. The measure of performance to be improved
3. The source of experience

Examples

1. ***Checkers game:*** A computer program that learns to play ***checkers*** might improve its performance as measured by its ability to win at the class of tasks involving playing checkers games, through experience obtained by playing games against itself.

A checkers learning problem:

- Task T: playing checkers
- Performance measure P: percent of games won against opponents
- Training experience E: playing practice games against itself

(b) Specify the learning task for Checker's Game and Hand Writing Recognition Problem	[05]	CO1	L1

A checkers learning problem:

- Task T: playing checkers
- Performance measure P: percent of games won against opponents
- Training experience E: playing practice games against itself

2. **A handwriting recognition learning problem:**
   - Task T: recognizing and classifying handwritten words within images
   - Performance measure P: percent of words correctly classified
   - Training experience E: a database of handwritten words with given classifications

2 (a) Describe the steps involved in Designing learning System. [10] | CO1 | L2

The basic design issues and approaches to machine learning are illustrated by designing a program to learn to play checkers, with the goal of entering it in the world checkers tournament

1. Choosing the Training Experience
2. Choosing the Target Function
3. Choosing a Representation for the Target Function
4. Choosing a Function Approximation Algorithm
    1. Estimating training values
    2. Adjusting the weights
5. The Final Design

1. **Choosing the Training Experience**

- The first design choice is to choose the type of training experience from which the system will learn.
- The type of training experience available can have a significant impact on success or failure of the learner.

There are three attributes which impact on success or failure of the learner

1. Whether the training experience provides ***direct or indirect feedback*** regarding the choices made by the performance system.

2. The degree to which the ***learner controls the sequence of training examples***

3. How well it represents the ***distribution of examples*** over which the final system performance P must be measured

2. **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's consider 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. We must learn to choose among the legal moves, the most obvious choice for the type of information to be learned is a program, or function, that chooses the best move for any given board state.

1. Let ***ChooseMove*** be the target function and the notation is

ChooseMove : B→ M

which 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.

2. An alternative target function is an ***evaluation function*** that assigns a ***numerical score***

to any given board state

Let the target function V and the notation

V : B → R

3. Choosing a Representation for the Target Function

Let's choose a simple representation - for any given board state, the function ***c*** will be calculated as a linear combination of the following board features:

- x_1: the number of black pieces on the board
- x_2: the number of red pieces on the board
- x_3: the number of black kings on the board
- x_4: the number of red kings on the board
- x_5: the number of black pieces threatened by red (i.e., which can be captured on red's next turn)
- x_6: the number of red pieces threatened by black

4. 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.

Each training example is an ordered pair of the form $(b, V_{train}(b))$.

For instance, the following training example describes a board state b in which black has won the game (note $x_2 = 0$ indicates that red has no remaining pieces) and for which the target function value $V_{train}(b)$ is therefore +100.

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

Function Approximation Procedure

1. Derive training examples from the indirect training experience available to the learner

2. Adjusts the weights w_i to best fit these training examples

5. The Final Design

The final design of checkers learning system can be described by four distinct program modules that represent the central components in many learning systems

1. **The Performance System** is the module that must solve the given performance task 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.

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

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 examples, 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 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.

3 (a) Comment the issues in Machine Learning.	[05]	CO1	L1

- What algorithms exist for learning general target functions from specific training examples? In what settings will particular algorithms converge to the desired function, given sufficient training data? Which algorithms perform best for which types of problems and representations?
- How much training data is sufficient? What general bounds can be found to relate the confidence in learned hypotheses to the amount of training experience and the character of the learner's hypothesis space?
- When and how can prior knowledge held by the learner guide the process of generalizing from examples? Can prior knowledge be helpful even when it is only approximately correct?
- What is the best strategy for choosing a useful next training experience, and how does the choice of this strategy alter the complexity of the learning problem?
- What is the best way to reduce the learning task to one or more

function approximation problems? Put another way, what specific functions should the system attempt to learn? Can this process itself be automated?

(b) Write the Find-S algorithm. Discuss the issues with the same. [05] CO2 L2

1. Initialize *h* to the most specific hypothesis in *H*

2. For each positive training instance *x*

For each attribute constraint

a_i in *h*

If the constraint a_i is satisfied by *x*

Then do nothing

Else replace a_i in *h* by the next more general constraint that is satisfied by *x*

Output hypothesis

Issues with Fins-S

1. Has the learner converged to the correct target concept?
2. Why prefer the most specific hypothesis?
3. Are the training examples consistent?

4 (a) Write Candidate Elimination algorithm. Apply the same to obtain final Version Space for the training example below. [10] CO2 L2

Initialize G to the set of maximally general hypotheses in H Initialize S to the set of maximally specific hypotheses in H

For each training example d, do

- 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 any hypothesis that is more general than another hypothesis in S

- 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 hypothesis in G

Example	Sky	Air Temp	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

5 (a) Discuss an Unbiased Learner with respect to Candidate elimination algorithm. [06] CO2 L2

- 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 teachable concept that is representing every possible subset of the instances X.
- The set of all subsets of a set X is called the power set of X

  - In the ***EnjoySport*** learning task the size of the instance space X of days described by the six attributes is 96 instances.
  - Thus, there are 2^{96} distinct target concepts that could be defined over this instance space and learner might be called upon to learn.
  - The conjunctive hypothesis space is able to represent only 973 of these - a biased hypothesis space indeed

  - 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
  - The target concept "Sky = Sunny or Sky = Cloudy" could then be described as
  - Modelling inductive systems by equivalent deductive systems.

- The input-output behavior of the CANDIDATE-ELIMINATION algorithm using a hypothesis space H is identical to that of a deductive theorem prover utilizing the assertion "H contains the target concept." This assertion is therefore called the inductive bias of the CANDIDATE-ELIMINATION algorithm.
- Characterizing inductive systems by their inductive bias allows modelling them by their equivalent deductive systems. This provides a way to compare inductive systems according to their policies for generalizing beyond the observed training data.

(b) Define a Decision Tree. Explain its representation. [04] CO1 L1

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.

DECISION TREE REPRESENTATION

- Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance.
- Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute.
- An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node.

- Decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances.
- Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions

For example, the decision tree shown in above figure
corresponds to the expression (Outlook = Sunny
∧ Humidity = Normal)
∨ (Outlook = Overcast)
∨ (Outlook = Rain ∧ Wind = Weak)

6 (a) Define an ID3 algorithm. Apply the same to obtain Decision Tree for the training example below. [10] CO2 L3

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

7 (a) Write short note on the following. [2+3+2+3] CO1, CO2 L1

a) Version Space

The version space is represented by its most general and least general members. These members form general and specific boundary sets that delimit the version space within the partially ordered hypothesis space.

Definition: The **general boundary** G, with respect to hypothesis space ***H*** and training data *D*,
is the set of maximally general members of *H* consistent with *D*

$$G \equiv \{g \in H \mid Consistent\ (g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)]\}$$

Definition: The **specific boundary** S, with respect to hypothesis space *H* and training data *D*,
is the set of minimally general (i.e., maximally specific) members of *H* consistent with *D.*

$$S \equiv \{s \in H \mid Consistent\ (s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)]\}$$

b) Entropy and Information gain

To define information gain, we begin by defining a measure called entropy. *Entropy measures the impurity of a collection of examples.*

Given a collection S, containing positive and negative examples of some target concept, the entropy of S relative to this Boolean classification is

$$Entropy\ (S) \equiv -p_{\oplus} log_2 p_{\oplus} - p_{\ominus} log_2 p_{\ominus}$$

INFORMATION GAIN MEASURES THE EXPECTED REDUCTION IN ENTROPY

- ***Information gain,*** is the expected reduction in entropy caused by partitioning the examples according to this attribute.
- The information gain, Gain(S, A) of an attribute A, relative to a collection of examples S, is defined as

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

c) Concept Learning

- Learning involves acquiring general concepts from specific training examples. Example: People continually learn general concepts or categories such as "bird," "car," "situations in which I should study more in order to pass the exam," etc.
- Each such concept can be viewed as describing some subset of objects or events defined over a larger set
- Alternatively, each concept can be thought of as a Boolean-valued function defined over this larger set. (Example: A function defined over all animals, whose value is true for birds and false for other animals).

Definition: Concept learning **-** Inferring a Boolean-valued function from training examples of its input and output

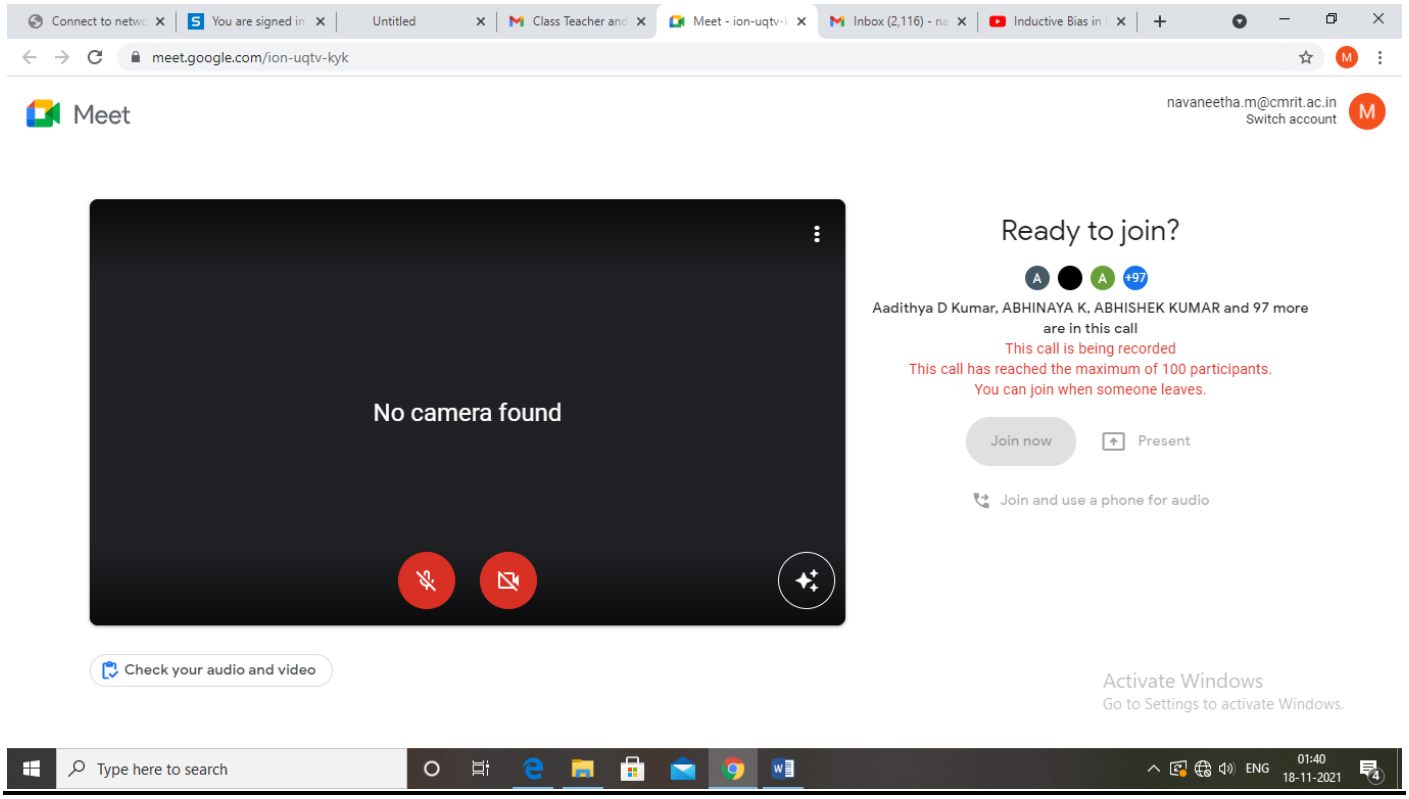
d) Limitations of Candidate elimination algorithm

What if the target concept is not contained in the hypothesis space?
Can we avoid this difficulty by using a hypothesis space that includes every possible hypothesis?
How does the size of this hypothesis space influence the ability of the algorithm to generalize to unobserved instances?
How does the size of the hypothesis space influence the number of training examples that must be observed?

Course Outcomes		Blooms Level	Modules covered	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO1	PSO2	PSO3	PSO4
CO1	Investigate concept learning, Identify the problems for machine learning. And select the either supervised, unsupervised or reinforcement learning.	**L1,L2,L3**	**1,2,3,4,5**	2	2	1	1	2	1	-	-	1	1	-	2	1	1	1	-
CO2	Explain theory of probability and statistics related to machine learning	**L1,L2,L3**	**1,2,3,4,5**	2	3	1	1	2	1	1	-	1	1	-	1	1	1	1	-
CO3	Understanding the concepts of ANN, Bayes classifier, k nearest neighbour	**L1,L2,L3**	**3,4,5**	2	3	2	1	2	2	1	-	1	1	-	2	1	1	1	-

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				
PSO3	Apply software engineering methods to design, develop, test and manage software systems.				
PSO4	Develop intelligent applications for business and industry				


Connect to netwo
You are signed in
Untitled
Class Teacher and
Meet - ion-uqtv-k
Inbox (2,116) - na
Inductive Bias in
meet.google.com/ion-uqtv-kyk
Meet
navaneetha.m@cmrit.ac.in
Switch account
No camera found
Check your audio and video
Ready to join?
Aadithya D Kumar, ABHINAYA K, ABHISHEK KUMAR and 97 more are in this call
This call is being recorded
This call has reached the maximum of 100 participants.
You can join when someone leaves.
Join now
Present
Join and use a phone for audio
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
01:40
18-11-2021