USN										

CMRIT — CMR Institute of Technology, Bengaluru. Accredited with A++ Grade by NAAC

Internal Assessment Test 2 - May 2025

Sub:	**Machine Learning**					Sub Code:	**BCS602**	Branch:	**ISE**
Date:	**23/05/2025**	Duration:	**90 min**	Max Marks:	**50**	Sem/Sec:	**VI / A, B & C**		**OBE**

Answer any FIVE FULL Questions — MARKS | CO | RBT

1a Briefly describe the scope of Reinforcement learning. [3] CO5 L1

1b Consider the given training dataset of 4 instances which contains the student's performance and their likelihood of getting a job offer or not. Apply Candidate Elimination Method. [7] CO2 L3

CGPA	Interactiveness	Practical Knowledge	Communication Skill	Logical Thinking	Interset	Job Offer
≥ 9	Yes	Excellent	Good	Fast	Yes	Yes
≥ 9	Yes	Good	Good	Fast	Yes	Yes
≥ 8	No	Good	Good	Fast	No	No
≥ 9	Yes	Good	Good	Slow	No	Yes

2 Derive Linear regression model with necessary equations. [10] CO3 L1

3a Consider the student performance dataset given in the table. Based on the performance of a student, classify whether a student will pass or fail using K -NN. Given the test case (6.1, 40,5). Assume k = 3 [5] CO3 L3

S. No	CGPA	Assessment	Project Submitted	Result
1.	9.2	85	8	Pass
2.	8	80	7	Pass
3.	8.5	81	8	Pass
4.	6	45	5	Fail
5.	6.5	50	4	Fail
6.	8.2	72	7	Pass
7.	5.8	38	5	Fail
8.	8.9	91	9	Pass

3b Consider the given training dataset T and construct a decision tree using C4.5 method. **(Note: One Iteration is enough)** [5] CO3 L3

S.No	Credit Score	Income	Collateral	Approve Loan
1.	High	High	High	Yes
2.	High	High	No	Yes
3.	Medium	High	Yes	Yes
4.	Low	Low	No	No

5.	Low	High	Yes	No

Q. No.	Question	Marks	CO	Level
4	Consider a perceptron to represent the Boolean function AND with the initial weigths **w_1 = 0.3, w_2 = -0.2, learning rate α = 0.2, and bias Θ = 0.4.** Use Step function to calculate the output and derive the perceptron that performs AND operation.	[10]	CO4	L3
5a	Draw the architecture of Fully Connected and Multilayer Perceptron Network	[04]	CO4	L2
5b	Assess a student's performance using Naïve Bayes algorithm with the dataset provided in the table. Given test data {(CGPA ≥ 9, Interactiveness = Yes, Practical Knowledge = Average}, apply Naïve Bayes theorem	[06]	CO4	L3

S.No	CGPA	Interactiveness	Practical Knowledge	Job Offer
1.	≥ 9	Yes	Very Good	Yes
2.	≥ 8	No	Good	Yes
3.	≥ 9	No	Average	No
4.	< 8	No	Average	No
5.	≥ 8	Yes	Good	Yes

Q. No.	Question	Marks	CO	Level
6a	Consider the following set of data given in the table. Apply Single Linkage algorithm	[05]	CO5	L3

Objects	X-Coordinate	Y - coordinate
1	1	4
2	2	8
3	5	10
4	12	18
5	14	28

Q. No.	Question	Marks	CO	Level
6b	If the given coordinates of the objects are (0,3) and (5.8), calculate the Euclidean, Manhattan and Chebyshev distance.	[03]	CO5	L3
6c	Explain Markov Decision Process	[02]	CO5	L1

Faculty Signature CCI Signature HOD Signature

SOLUTION

Ans- 1a The scope of **Reinforcement Learning (RL)** involves teaching agents to make sequences of decisions by interacting with an environment to maximize cumulative rewards. It covers:

1. **Learning from Interaction**: Agents learn optimal behaviors by exploring and exploiting outcomes of actions.
2. **Dynamic Environments**: Applied where outcomes depend on both current actions and evolving states.

3. **Applications**: RL is used in robotics, game playing (e.g., AlphaGo), autonomous vehicles, recommendation systems, finance, and healthcare.

4. **Key Techniques**: Includes value-based methods (like Q-learning), policy-based methods, and deep reinforcement learning (combining RL with deep learning).

In summary, RL's scope spans both theoretical and practical domains, enabling machines to learn decision-making strategies in complex, uncertain environments.

Ans 1(b) To apply the **Candidate Elimination Algorithm**, we maintain two sets:

- **S** (the most specific hypothesis)
- **G** (the most general hypothesis)

We generalize **S** only when it fails to cover a **positive** instance and specialize **G** only when it incorrectly covers a **negative** instance.

Step-by-step Execution

Initial Hypotheses

- **S =** First positive instance:
  ⟨≥ 9, Yes, Excellent, Good, Fast, Yes⟩
- **G =** Most general:
  ⟨?, ?, ?, ?, ?, ?⟩

Instance 2 (Positive):

⟨≥ 9, Yes, Good, Good, Fast, Yes⟩

Compare with **S** and generalize it:

- Practical Knowledge: Excellent → Good → generalize to ?

Updated S:
⟨≥ 9, Yes, ?, Good, Fast, Yes⟩

G remains unchanged.

Instance 3 (Negative):

⟨≥ 8, No, Good, Good, Fast, No⟩

This negative example is **covered by G** but **not by S**, so we **specialize G** to exclude this instance.

We specialize G by making hypotheses that exclude this instance while still covering S.

S = ⟨≥ 9, Yes, ?, Good, Fast, Yes⟩

Possible specializations of G that exclude the negative:

- CGPA: ≥ 9
- Interactiveness: Yes
- Interest: Yes

So new **G** becomes:

- ⟨≥ 9, ?, ?, ?, ?, ?⟩
- ⟨?, Yes, ?, ?, ?, ?⟩
- ⟨?, ?, ?, ?, ?, Yes⟩

Remove those that do **not** cover **S**:

- ⟨≥ 9, ?, ?, ?, ?, ?⟩ — OK
- ⟨?, Yes, ?, ?, ?, ?⟩ — OK
- ⟨?, ?, ?, ?, ?, Yes⟩ — OK

So all 3 stay.

Instance 4 (Positive):

⟨≥ 9, Yes, Good, Good, Slow, No⟩

Compare with **S = ⟨≥ 9, Yes, ?, Good, Fast, Yes⟩**

S doesn't cover due to:

- Logical Thinking: Fast ≠ Slow
- Interest: Yes ≠ No

So generalize S:

- Logical Thinking → ?
- Interest → ?

Updated S = ⟨≥ 9, Yes, ?, Good, ?, ?⟩

Now filter **G** to keep only those that still cover the updated **S** and the current positive instance.

From previous G:

1. ⟨≥ 9, ?, ?, ?, ?, ?⟩ — OK
2. ⟨?, Yes, ?, ?, ?, ?⟩ — OK

3. ⟨?, ?, ?, ?, ?, Yes⟩ — **Reject** (doesn't cover Interest = No)

Final G =

- ⟨≥ 9, ?, ?, ?, ?, ?⟩
- ⟨?, Yes, ?, ?, ?, ?⟩

Final Version Space

- **S = ⟨≥ 9, Yes, ?, Good, ?, ?⟩**
- **G = {⟨≥ 9, ?, ?, ?, ?, ?⟩, ⟨?, Yes, ?, ?, ?, ?⟩}**

ANS-2 1) **Objective of Linear Regression**

To model the relationship between a **dependent variable** y and one or more **independent variables** x, assuming a linear relationship:

$$y=\beta 0+\beta 1x+\epsilon$$

Where:

- y: actual output (dependent variable)
- x: input feature (independent variable)
- β0,β1 : regression coefficients (intercept and slope)
- ϵ: error term

2) Hypothesis Function

y^=h(x)=β0+β1x where y^ is the predicted output

3) Cost Function (Mean Squared Error – MSE

$$J(\beta 0, \beta 1) = \frac{1}{2m} \sum_{i=1}^{m} (h(xi) - yi)^2$$

Where:

- m: number of training examples
- $h(x_i)$: predicted value
- y_i: actual value

4. Gradient Descent (to minimize cost)

Update rules for coefficients:

$$\beta_o = \beta_o - \alpha \frac{\partial J}{\partial \beta_o}, \qquad \beta_1 = \beta_1 - \alpha \frac{\partial J}{\partial \beta_1}$$

Compute gradients:

$$\frac{\partial J}{\partial \beta_o} = \frac{1}{m} \sum_{i=1}^{m} (h(xi) - yi) , \frac{\partial J}{\partial \beta_1} = \frac{1}{m} \sum_{i=1}^{m} (h(xi) - yi)(xi) \ ,$$

Ans 3 (a)

S. No	CGPA	Assessment	Project Submitted	Result
1.	9.2	85	8	Pass
2.	8	80	7	Pass
3.	8.5	81	8	Pass
4.	6	45	5	Fail
5.	6.5	50	4	Fail
6.	8.2	72	7	Pass
7.	5.8	38	5	Fail
8.	8.9	91	9	Pass

Use Euclidean Distance Formula

$$Distance = \sqrt{(x_1 - x_1)^2 + (y - y_1)^2}$$

Compute Distances

S.No	CGPA	Assess	Proj	Result	Distance to (6.1, 40, 5)
1	9.2	85	8	Pass	$\sqrt{[(9.2-6.1)^2 + (85-40)^2 + (8-5)^2]} \approx 45.3$
2	8.0	80	7	Pass	$\sqrt{[(8-6.1)^2 + (80-40)^2 + (7-5)^2]} \approx 40.1$
3	8.5	81	8	Pass	$\sqrt{[(8.5-6.1)^2 + (81-40)^2 + (8-5)^2]} \approx 41.9$
4	6.0	45	5	Fail	$\sqrt{[(6.0-6.1)^2 + (45-40)^2 + (5-5)^2]} \approx 5.0$
5	6.5	50	4	Fail	$\sqrt{[(6.5-6.1)^2 + (50-40)^2 + (4-5)^2]} \approx 10.2$
6	8.2	72	7	Pass	≈ 33.6
7	5.8	38	5	Fail	$\sqrt{[(5.8-6.1)^2 + (38-40)^2 + (5-5)^2]} \approx 2.2$
8	8.9	91	9	Pass	≈ 51.2

---

4. Select 3 Nearest Neighbors

Sorted distances:

1. S7 – Distance ≈ 2.2 – **Fail**
2. S4 – Distance ≈ 5.0 – **Fail**
3. S5 – Distance ≈ 10.2 – **Fail**

---

5. Majority Voting

All 3 nearest neighbors are **Fail**.

Final Classification: FAIL

So, the student with (6.1, 40, 5) is predicted to **Fail** using K-NN with k=3k = 3k=3.

Ans 3(b) Step 1: Dataset Summary

S.No	Credit Score	Income	Collateral	Approve Loan
1	High	High	High	Yes
2	High	High	No	Yes
3	Medium	High	Yes	Yes
4	Low	Low	No	No
5	Low	High	Yes	No

 Total instances: **5**

 Class label: **Approve Loan (Yes/No)**

Step 2: Entropy of Dataset (D)

We compute the **Entropy of the entire dataset** DDD:

- Yes: 3 instances
- No: 2 instances

Entropy(D)= $-\frac{3}{5}\log\frac{3}{5}-\frac{2}{5}\log\frac{2}{5}$=0.971

Step 3: Choose Attribute with Highest Gain Ratio

We compute **Gain Ratio** for each attribute:

- C4.5 uses **Gain Ratio** = Information Gain / Split Information

Attribute: Credit Score

Values: High, Medium, Low

- High → Instances 1,2 → Yes, Yes → Entropy = 0
- Medium → Instance 3 → Yes → Entropy = 0
- Low → Instances 4,5 → No, No → Entropy = 0

Expected Entropy =$\frac{2}{5}.0+\frac{1}{5}.0=0$

Gain = 0.971−0=0.9710.971 - 0 = 0.9710.971−0=0.971

SplitInfo =- ($\frac{2}{5}\log\frac{2}{5}+\frac{1}{5}\log\frac{1}{5}+\frac{2}{5}\log\frac{2}{5}$)= 1.522

Gain Ratio=0.971/ .522=0.638

Attribute: Income

Values: High (4 instances), Low (1 instance)

- High → Yes, Yes, Yes, No → 3 Yes, 1 No

$$Entropy=-\frac{3}{4}\log\frac{3}{4}-\frac{1}{4}\log\frac{1}{4}$$

- Low → No → Entropy = 0

  Expected Entropy =$\frac{4}{5}.0.811+\frac{1}{5}.0=0.649$

  Gain = 0.971−0.649=0.3220.971 - 0.649 = 0.3220.971−0.649=0.322

  SplitInfo ≈ 0.722

  Gain Ratio≈0.322/0.722≈0.44

  Attribute: Collateral

  Values: High, No, Yes

- High → Instance 1 → Yes → Entropy = 0
- No → Instances 2, 4 → Yes, No → Entropy ≈ 1.0
- Yes → Instances 3, 5 → Yes, No → Entropy ≈ 1.0

Expected Entropy =$\frac{1}{5}.0+\frac{2}{5}.1+\frac{2}{5}.1=0.8$

Gain = 0.971−0.8=0.171

SplitInfo ≈ 1.522

Gain Ratio≈0.171/ 0.522= 0.112

Step 4: Choose Attribute with Highest Gain Ratio

- **Credit Score** has the highest gain ratio ≈ **0.638**

Final tree- 1 Iteartion

Credit Score?

├── High → Approve Loan = Yes

├── Medium → Approve Loan = Yes

└── Low → Approve Loan = No

Ans -4 To derive a **Perceptron for the AND function**, we'll follow these steps:

Given:

- **Initial Weights**:
  w1=0.3w_1 = 0.3w1=0.3, w2=−0.2w_2 = -0.2w2=−0.2
- **Learning Rate**: α=0.2\alpha = 0.2α=0.2
- **Threshold (Bias)**: θ=0.4\theta = 0.4θ=0.4
- **Activation Function**: Step function:

$$f(net) = \begin{Bmatrix} 1 & if\ net \geq \theta \\ 0 & if\ net < \theta \end{Bmatrix}$$

Step 1: Truth Table for AND

x_1	x_2	Target (t)
0	0	0
0	1	0
1	0	0
1	1	1

Step 2: Perceptron Training Loop (One Epoch)

We'll go through each input, compute output, compare with target, and adjust weights:

Case 1: (0, 0) → Target: 0

- Net input = 0∗0.3+0∗(−0.2)=0
- Since 0<θ(0.4)0 <θ(0.4), Output = 0 → No weight chang

Case 2: (0, 1) → Target: 0

Net = 0∗0.3+1∗(−0.2)=−0.20 → Output = 0 → No weight change

Case 3: (1, 0) → Target: 0

- Net = *0.3 + 0*(-0.2) = 0.3 → Output = 0 → No weight change

Case 4: (1, 1) → Target: 1

- Net = 1∗0.3+1∗(−0.2)=0.1 → Output = 0 Incorrect

Update weights:

Δw1=α·(t−o)· x1=0.2· (1−0)·1=0.2

Δw2=0.2·(1−0)·1=0.2

New weights:

w1=0.3+0.2=0.5

w2=−0.2+0.2=0.0

New Weights After One Epoch:

- w1=0.5w_1 = 0.5w1=0.5
- w2=0.0w_2 = 0.0w2=0.0
- Threshold = 0.4

x_1	x_2	Net = $w_1x_1 + w_2x_2$	Output	Target

0	0	0.0	0	0
0	1	0.0	0	0
1	0	0.5	1 ❌	0
1	1	0.5	1 ✅	1

Conclusion:
You **need another epoch** to fix (1,0). Repeat weight update for that input:

- Net = 0.5 → Output = 1 (should be 0)

$\Delta w1=0.2\cdot(0-1)\cdot 1=-0.2 \Rightarrow w1=0.5-0.2=0.3$
$\Delta w2=0.2\cdot(0-1)\cdot 0=0 \Rightarrow w2=0.0$

Final Weights After Convergence:

- w1=0.3w_1 = 0.3w1=0.3
- w2=0.0w_2 = 0.0w2=0.0
- Threshold = 0.4

This perceptron **correctly classifies** the AND function.

Ans-5

- **Fully Connected Network** refers to the connectivity: each neuron is connected to every neuron in the next layer.
- **Multilayer Perceptron** is a type of Fully Connected Network that has **at least one hidden layer** and uses nonlinear activation functions.

Fully connected neural network architecture

```
Input Layer      Hidden Layer       Output Layer
 (X1)  ● ───────────────┬→ ● (H1) ──┬──────────┬─────→ ● (O1)
 (X2)  ● ─┘      │   ● (H2) ─┘   │
 (X3)  ● ─────────────┘    ● (H3) ──────────┘
```

□ Each input node connects to every node in the next layer.
□ Typically consists of:

- Input layer
- One or more hidden layers
- Output layer

Multilayer Perceptron (MLP) Architecture

```
Input Layer      Hidden Layer(s)          Output Layer
  X1  ● ─┬──────────┬→ ● H1 ──┬─────────┬→ ● H4 ──┬───→ ● Y1
  X2  ● ─┘    │   ● H2 ┘    │   ● H5 ┘
  X3  ● ───────────┘   ● H3 ──────────┘    ● H6

      [Hidden Layer 1]     [Hidden Layer 2]
```

□ **Activation Functions** like ReLU, sigmoid, or tanh are applied in hidden layers.
□ Typically trained using backpropagation and gradient descent.

Key Differences / Notes:

Feature	**Fully Connected Network**	**Multilayer Perceptron**
Layers	May include only 1 layer	Always includes ≥1 hidden layer
Activation Functions	Not always applied	Non-linear activations used
Depth	Shallow or deep	Deep (≥1 hidden layers)

Ans-6 (a) **Single Linkage Algorithm**
Step-1 Find Euclidean distance

Pair	Distance
(1,2)	$\sqrt{(2-1)^2 + (8-4)^2} = \sqrt{1 + 16} = \sqrt{17} \approx$ **4.12**
(1,3)	$\sqrt{(5-1)^2 + (10-4)^2} = \sqrt{16 + 36} = \sqrt{52} \approx$ **7.21**
(1,4)	$\sqrt{(12-1)^2 + (18-4)^2} = \sqrt{121 + 196} = \sqrt{317} \approx$ **17.80**
(1,5)	$\sqrt{(14-1)^2 + (28-4)^2} = \sqrt{169 + 576} = \sqrt{745} \approx$ **27.29**
(2,3)	$\sqrt{(5-2)^2 + (10-8)^2} = \sqrt{9 + 4} = \sqrt{13} \approx$ **3.61**
(2,4)	$\sqrt{(12-2)^2 + (18-8)^2} = \sqrt{100 + 100} = \sqrt{200} \approx$ **14.14**
(2,5)	$\sqrt{(14-2)^2 + (28-8)^2} = \sqrt{144 + 400} = \sqrt{544} \approx$ **23.32**
(3,4)	$\sqrt{(12-5)^2 + (18-10)^2} = \sqrt{49 + 64} = \sqrt{113} \approx$ **10.63**
(3,5)	$\sqrt{(14-5)^2 + (28-10)^2} = \sqrt{81 + 324} = \sqrt{405} \approx$ **20.12**
(4,5)	$\sqrt{(14-12)^2 + (28-18)^2} = \sqrt{4 + 100} = \sqrt{104} \approx$ **10.20**

Step 2: Apply Single Linkage Clustering

Single Linkage: At each step, merge the two clusters that have the **smallest minimum distance** between any two members.

Initial Clusters:

- {1}, {2}, {3}, {4}, {5}

  Step 1: Merge Closest Pair
- Closest: (2,3) → Distance ≈ 3.61
  → New Cluster: {2,3}

  Clusters:
- {1}, {2,3}, {4}, {5}

  Step 2: Next Closest
- (1,2) ≈ 4.12
  → Merge {1} and {2,3} → {1,2,3}

  Clusters:
- {1,2,3}, {4}, {5}

  Step 3: Next Closest
- Min({1,2,3}, {4}) = min(17.80, 14.14, 10.63) = **10.63 (3,4)**
  → Merge {1,2,3} and {4} → {1,2,3,4}

  Clusters:
- {1,2,3,4}, {5}

  Step 4: Final Merge
- Min({1,2,3,4}, {5}) = min(27.29, 23.32, 20.12, 10.20) = **10.20 (4,5)**
  → Merge all → {1,2,3,4,5}

Dendrogram Order (Approximate Distances)

1. Merge (2,3) → 3.61
2. Merge (1) to (2,3) → 4.12
3. Merge (4) to (1,2,3) → 10.63
4. Merge (5) to (1,2,3,4) → 10.20

Ans 6(b) Let's calculate the **Euclidean**, **Manhattan**, and **Chebyshev** distances between the two points:

- **Point A**: (0, 3)
- **Point B**: (5, 8)
  1. Euclidean Distance $d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} = \sqrt{(5-0)^2 + (8-3)^2} = \sqrt{25 + 25} = \sqrt{50}$
  2. Manhattan Distance- d=|x2−x1|+|y2−y1|=|5−0|+|8−3|=5+5=10
  3. Chebyshev Distance

d=max(|x2−x1|,|y2−y1|)=max(|5−0|,|8−3|)=max(5,5)=5

Ans 6 (c) A **Markov Decision Process (MDP)** is a mathematical framework used to describe a fully observable environment in decision-making problems, especially in reinforcement learning. It provides a formal way to model **sequential decision-making** where outcomes are partly random and partly under the control of a decision maker.

Components of an MDP:

An MDP is defined by a **5-tuple**:

(S,A,P,R,γ)(S, A, P, R, \gamma)(S,A,P,R,γ)

1. **S – States**
   The set of all possible states the environment can be in.
   Example: In a grid-world, each grid cell is a state.
2. **A – Actions**
   The set of all actions available to the agent.
   Example: Up, Down, Left, Right.
3. **P – Transition Probability**
   P(s′|s,a)P(s'|s, a)P(s′|s,a): The probability of transitioning to state s′s's′ from state sss after taking action aaa.
   This satisfies the **Markov property**: the future is independent of the past given the present.
4. **R – Reward Function**
   R(s,a)R(s, a)R(s,a): The immediate reward received after performing action aaa in state sss.
   It defines the goal of the agent—to maximize the cumulative reward.
5. **γ – Discount Factor**
   0≤γ≤10 \leq \gamma \leq 10≤γ≤1: A factor that determines the importance of future rewards.
   - If γ = 0 → only immediate reward matters.
   - If γ ≈ 1 → long-term rewards are also important.

Goal of MDP:

To find a **policy** π(a|s)\pi(a|s)π(a|s) that defines the best action to take in each state in order to **maximize the expected cumulative reward** over time (often called the **return**).

Value Function:

1. **State Value Function Vπ(s)V^\pi(s)Vπ(s)**
   Expected return when starting in state sss and following policy π\piπ.
2. **Action Value Function Qπ(s,a)Q^\pi(s, a)Qπ(s,a)**
   Expected return after taking action aaa in state sss, and then following policy π\piπ.

Applications:

- Reinforcement learning
- Game playing (e.g., chess, Go)
- Robotics and control systems
- Finance and operations research