Biyani's Think Tank

Concept Based Notes

Artificial Intelligence & Machine Learning

Ms. Shbna Ali

Asst. Professor (Dept. of IT)

Biyani Girls College, Jaipur

BCA Semester V

Published by :

Think Tanks

Biyani Group of Colleges

Concept & Copyright :

©Biyani Shikshan Samiti Sector-3, Vidhyadhar Nagar, Jaipur-302 023 (Rajasthan)

Ph : 0141-2338371, 2338591-95 Fax : 0141-2338007

E-mail : acad@biyanicolleges.org

Website :www.gurukpo.com; www.biyanicolleges.org

ISBN :

Edition: 2025

Leaser Type Setted by :
Biyani College Printing Department

Preface

I am glad to present this book, especially designed to serve the needs ofthe students. The book has been written keeping in mind the general weakness in understanding the fundamental concepts of the topics. The book is self- explanatory and adopts the ―Teach Yourself‖ style. It is based on question- answer pattern. The language of book is quite easy and understandable basedon scientific approach.

Any further improvement in the contents of the book by making corrections, omission and inclusion is keen to be achieved based on suggestions from the readers for which the author shall be obliged.

I acknowledge special thanks to Mr. Rajeev Biyani, *Chairman* & Dr. Sanjay Biyani, *Director* (*Acad.*) Biyani Group of Colleges, who are the backbones and main concept provider and also have been constant source of motivation throughout this Endeavour. They played an active role in coordinating the various stages of this Endeavour and spearheaded the publishing work.

I look forward to receiving valuable suggestions from professors of various educational institutions, other faculty members and students for improvement of the quality of the book. The reader may feel free to send in their comments and suggestions to the under mentioned address.

Author

.

BCA-75T-301: Artificial Intelligence & Machine Learning

UNIT-I

General Issues and overview of AI: Concept of Intelligence, AI intelligent agents, Characteristics of AI, Comparison of AI, Machine Learning and Deep Learning. Defining problem as a State Space Search, Search and Control Strategies, Production systems, Problems - Water Jug problem, Block words Problem, Monkey & Banana problem, Applications of AI.Ethical considerations in AI development& deployment and MYCIN.

Unit-II

Searching- Searching for solutions, uniformed search strategies Breadth first search, depth first Search. Informed search strategies (Heuristic search) Generate-and-test, Hill climbing, Best First Search, Constraint Satisfaction,A*, AO* Algorithms, Problem reduction, Game Playing-Adversial search.
Knowledge Representation Definition of Knowledge, Types of knowledge (Procedural and Declarative knowledge), Approaches to Knowledge Representation, Knowledge representation using Propositional and Predicate logic, Conversion to clause form, Resolution in Propositional logic, Resolution in Predicate logic.

Unit-III

Concept: Machine Learning, Machine Learning Foundations-Overview, Applications, Types of Machine Learning, Basic Concepts in Machine Learning - Examples of Machine Learning.
Supervised Learning: Introduction, Linear Models of Classification - Decision Trees, Naïve Bayes Classification, Linear Regression - Logistic Regression - Bayesian Logistic Regression - Probabilistic Models Neural Network- Feed Forward Network Functions - Error Back Propagation - Regularization.

Unit-IV

Unsupervised Learning Clustering, Association rule mining, K-Means Clustering, EM (Expectation Maximization), Mixtures of Gaussians, EM algorithm in General, The Curse of Dimensionality, Dimensionality Reduction, Factor Analysis, Principal Component Analysis.
Probabilistic Graphical Models: Directed Graphical Models, Bayesian Networks, Exploiting Independence Properties, From Distributions to Graphs, Examples - Markov Random Fields - Inference In Graphical Models - Learning - Naïve Bayes Classifiers - Markov Models - Hidden Markov Models.

[illegible]
Dy. Registrar (Acad.)
University of Rajasthan
JAIPUR

Unit - I
General Issues and overview of AI

Q1. What is the Concept of Intelligence in AI?

Answer: "It is a branch of computer science by which we can create intelligent machines which can behave like a human, think like humans, and able to make decisions."Artificial Intelligence exists when a machine can have human based skills such as learning, reasoning, and solving problems With Artificial Intelligence you do not need to preprogram a machine to do some work, despite that you can create a machine with programmed algorithms which can work with own intelligence, and that is the awesomeness of AI. It is believed that AI is not a new technology, and some people says that as per Greek myth, there were Mechanical men in early days which can work and behave like humans.

Need of Artificial Intelligence-

Before Learning about Artificial Intelligence, we should know that what is the importance of AI and why should we learn it. Following are some main reasons to learn about AI: With the help of AI, you can create such software or devices which can solve real-world problems very easily and with accuracy such as health issues, marketing, traffic issues, etc.

- With the help of AI, you can create your personal virtual Assistant, such as Cortana, Google Assistant, Siri, etc.
- With the help of AI, you can build such Robots which can work in an environment where survival of humans can be at risk.
- AI opens a path for other new technologies, new devices, and new Opportunities.

Goals of Artificial Intelligence:

Following are the main goals of Artificial Intelligence:

1. Replicate human intelligence
2. Solve Knowledge-intensive tasks
3. An intelligent connection of perception and action
4. Building a machine which can perform tasks that requires human intelligence such as:

- ✓ Proving a theorem
- ✓ Playing chess
- ✓ Plan some surgical operation
- ✓ Driving a car in traffic

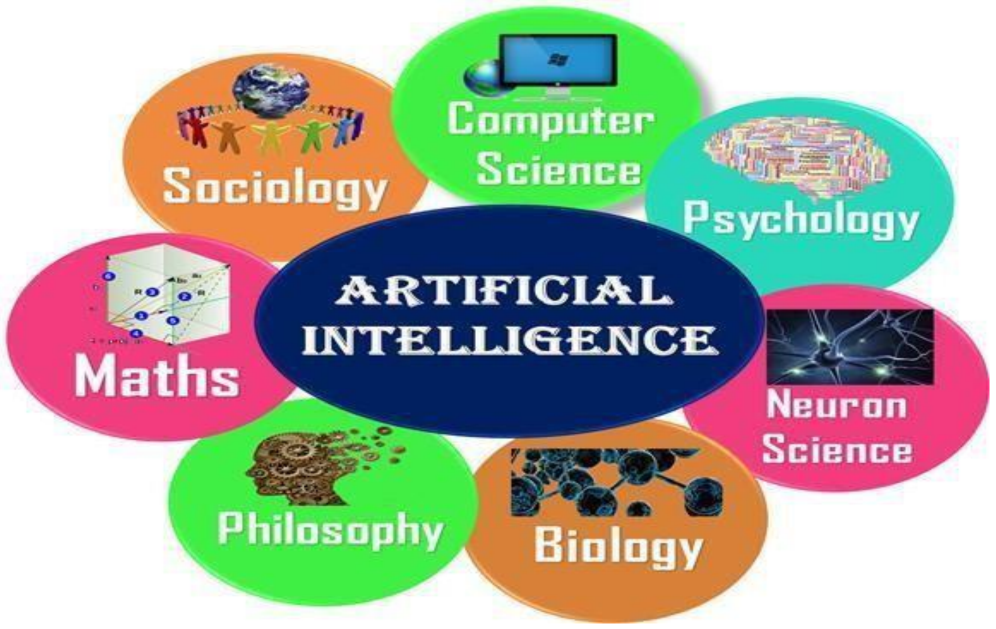
5. Creating some system which can exhibit intelligent behavior, learn new things by itself, demonstrate, explain, and can advise to its user.

What Comprises to Artificial Intelligence?

Artificial Intelligence is not just a part of computer science even it's so vast and requires lots of other factors which can contribute to it. To create the AI first we should know that how intelligence is composed, so the Intelligence is an intangible part of our brain which is a combination of **Reasoning, learning, problem-solving perception, language understanding, etc**.

To achieve the above factors for a machine or software Artificial Intelligence requires the following discipline:

- Mathematics
- Biology
- Psychology
- Sociology
- Computer Science
- Neurons Study
- Statistics



Advantages of Artificial Intelligence:

Following are some main advantages of Artificial Intelligence:

o **High Accuracy with less errors:** AI machines or systems are prone to less errors and high accuracy as it takes decisions as per pre-experience or information.

o **High-Speed:** AI systems can be of very high-speed and fast-decision making, because of that AI systems can beat a chess champion in the Chess game.

o **High reliability:** AI machines are highly reliable and can perform the same action multiple times with high accuracy.

o **Useful for risky areas:** AI machines can be helpful in situations such as defusing a bomb, exploring the ocean floor, where to employ a human can be risky.

o **Digital Assistant:** AI can be very useful to provide digital assistant to the users such as AI technology is currently used by various E- commerce websites to show the products as per customer requirement.

o **Useful as a public utility:** AI can be very useful for public utilities such as a self-driving car which can make our journey safer and hassle-free, facial recognition for security purpose, Natural language processing to communicate with the human in human-language, etc.

Disadvantages of Artificial Intelligence:

Every technology has some disadvantages, and thesame goes for Artificial intelligence. Being so advantageous technology still, it has some disadvantages which we need to keep in our mind while creating an AI system. Following are the disadvantages of AI:

o **High Cost:** The hardware and software requirement of AI is very costly as it requires lots of maintenance to meet current world requirements.

o **Can't think out of the box:** Even we are making smarter machines with AI, but still they cannot work out of the box, as the robot will only do that work for which they are trained, or programmed.

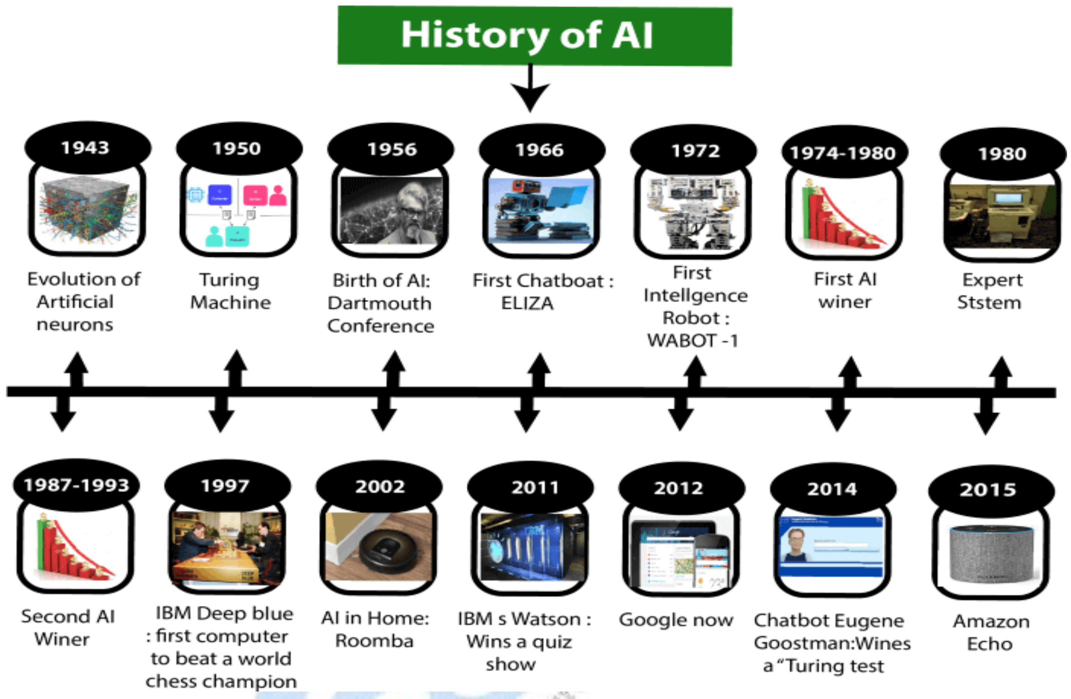
o **No feelings and emotions:** AI machines can be an outstanding performer, but still it does not have the feeling so it cannot make any kind of emotional attachment with human, and may sometime be harmful for users if the proper care is not taken.**Increase dependency on machines:** With the increment of technology, people are getting more dependent on devices and hence they are losing their mental capabilities.

o **No Original Creativity:** As humans are so creative and can imagine some new ideas but still AI machines cannot beat this power of human intelligence and cannot be creative and imaginative.

Q.2 Define History of Artificial Intelligence?

Ans: Artificial Intelligence is not a new word and not a new technology for researchers. This technology is much older than you would imagine.

Maturation of Artificial Intelligence (1943-1952)


History of AI
1943
Evolution of Artificial neurons
1950
Turing Machine
1956
Birth of AI: Dartmouth Conference
1966
First Chatboat : ELIZA
1972
First Intellgence Robot : WABOT -1
1974-1980
First AI winer
1980
Expert Ststem
1987-1993
Second AI Winer
1997
IBM Deep blue : first computer to beat a world chess champion
2002
AI in Home: Roomba
2011
IBM s Watson : Wins a quiz show
2012
Google now
2014
Chatbot Eugene Goostman:Wines a "Turing test
2015
Amazon Echo

- **Year 1943:** The first work which is now recognized as AI was done by Warren McCulloch and Walter pits in 1943. They proposed a model of **artificial neurons**.
- **Year 1949:** Donald Hebb demonstrated an updating rule for modifying the connection strength between neurons. His rule is now called **Hebbian learning**.
- **Year 1950:** The Alan Turing who was an English mathematician and pioneered Machine learning in 1950. Alan Turing publishes **"Computing Machinery and Intelligence"** in which he proposed a test. The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence, called a **Turing test**.

The birth of Artificial Intelligence (1952-1956)

- **Year 1955:** An Allen Newell and Herbert A. Simon created the "first artificial intelligence program"Which was named as **"Logic Theorist"**. This program had proved 38 of 52 Mathematics theorems, and find new and more elegant proofs for some theorems.
- **Year 1956:** The word "Artificial Intelligence" first adopted by American Computer scientist John

McCarthy at the Dartmouth Conference. For the first time, AI coined as an academic field.
At that time high-level computer languages such as FORTRAN, LISP, or COBOL were invented. And the enthusiasm for AI was very high at that time.

The golden years-Early enthusiasm (1956-1974)

- **Year 1966:** The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first chatbot in 1966, which was named as ELIZA.
- **Year 1972:** The first intelligent humanoid robot was built in Japan which was named as WABOT-1.

The first AI winter (1974-1980)

- The duration between years 1974 to 1980 was the first AI winter duration. AI winter refers to the time period where computer scientist dealt with a severe shortage of funding from government for AI researches.
- During AI winters, an interest of publicity on artificial intelligence was decreased.

A boom of AI (1980-1987)

- **Year 1980:** After AI winter duration, AI came back with "Expert System". Expert systems were programmed that emulate the decision-making ability of a human expert.
- In the Year 1980, the first national conference of the American Association of Artificial Intelligence **was held at Stanford University**.

The second AI winter (1987-1993)

- The duration between the years 1987 to 1993 was the second AI Winter duration.
- Again Investors and government stopped in funding for AI research as due to high cost but not efficient result. The expert system such as XCON was very cost effective.

The emergence of intelligent agents (1993-2011)

- **Year 1997:** In the year 1997, IBM Deep Blue beats world chess champion, Gary Kasparov, and became the first computer to beat a world chess champion.
- **Year 2002:** for the first time, AI entered the home in the form of Roomba, a vacuum cleaner.
- **Year 2006:** AI came in the Business world till the year 2006. Companies like Facebook, Twitter, and Netflix also started using AI.

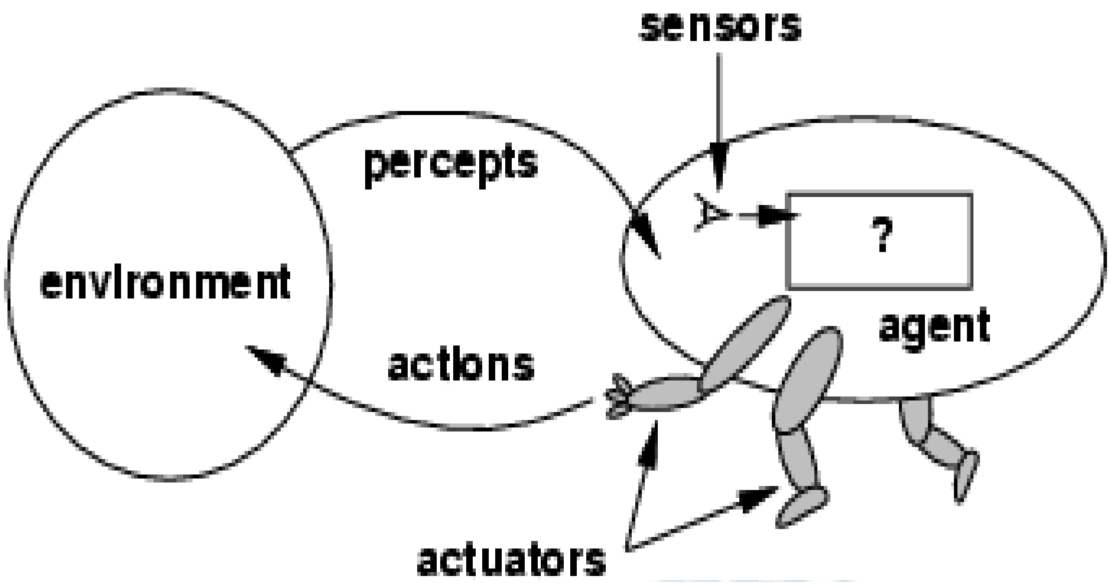
Deep learning, big data and artificial general intelligence (2011-present)

- **Year 2011:** In the year 2011, IBM's Watson won jeopardy, a quiz show, where it had to solve the complex questions as well as riddles. Watson had proved that it could understand natural language and can solve tricky questions quickly.
- **Year 2012:** Google has launched an Android app feature "Google now", which was able to provide information to the user as a prediction.
- **Year 2014:** In the year 2014, Chatbot "Eugene Goostman" won a competition in the infamous "Turing test."
- **Year 2018:** The "Project Debater" from IBM debated on complex topics with two master debaters and also performed extremely well.
- Google has demonstrated an AI program "Duplex" which was a virtual assistant and which had taken hairdresser appointment on call, and lady on other side didn't notice that she was talking with the machine.

Q3: What is an AI Intelligent Agent?

Ans: An AI Intelligent Agent is an autonomous entity (usually a software program or robot) that can perceive its environment, make decisions, and take actions to achieve specific goals. These agents use sensors to collect data, process that information, and perform actions using actuators or outputs.

In simple terms, an intelligent agent in AI is like a smart decision-maker that acts on behalf of a user or system, continuously learning and adapting to reach its objectives.



Agent:

An *Agent* is anything that can be viewed as perceiving its environment through sensors andacting upon that environment through actuators.

- A *human agent* has eyes, ears, and other organs for sensors and hands, legs, mouth,and other body parts for actuators.
- A *robotic agent* might have cameras and infrared range finders for sensors andvarious motors foractuators.
- A *software agent* receives keystrokes, file contents, and network packets as sensoryinputs and acts on the environment by displaying on the screen, writing files, and sending network packets.

Percept:

We use the term percept to refer to the agent's perceptual inputs at any given instant.

Percept Sequence:

An agent's percept sequence is the complete history of everything the agent has ever perceived.

Agent function:

Mathematically speaking, we say that an agent's behavior is described by the agent functionthat maps any **given percept sequence to an action**.

Components of an Intelligent Agent -

An AI intelligent agent usually has these core components:

1 **Sensors:** Devices or tools that gather information from the environment (e.g., cameras, microphones, GPS, or software inputs).

2 **Actuators:** Tools or software that allow the agent to take action (e.g., motors in robots, message responses in chatbots).

3 **Perception Module:** Processes raw data from the sensors into useful information.

4 **Decision-Making Module:** Analyzes the data and determines the best course of action.

5 **Learning Module (optional):** Allows the agent to improve its performance over time based on past experiences (common in machine learning-based agents).

Types of Intelligent Agents in AI

1. Simple Reflex Agents:

These respond to the current situation based on predefined rules (e.g., if-else logic).

Example: A thermostat that turns on the heater if the temperature drops below a certain level.

2. Model-Based Reflex Agents:

These keep track of the world using internal models and respond based on both current input and history.

Example: A self-driving car using a map and sensor data to navigate.

3. Goal-Based Agents:

These choose actions by considering the outcomes and working toward specific goals.

Example: A chess-playing AI trying to checkmate the opponent.

4. Utility-Based Agents:

These maximize their performance using a utility function (a measure of success).

Example: A recommendation system that shows the most relevant content based on user satisfaction.

5. Learning Agents:

These improve their performance over time through experience.

Example: A personal assistant app that learns your preferences and habits.

Real-World Examples of Intelligent Agents

- **Virtual Assistants:** Like Siri, Alexa, and Google Assistant. They listen (sensors), process your voice command (decision-making), and respond (actuators).
- **Autonomous Vehicles:** They sense the road using cameras and LiDAR, decide how to navigate, and control the car.
- **Recommendation Systems:** Platforms like Netflix or Amazon use intelligent agents to suggest movies or products.
- **Smart Home Devices:** Thermostats, lights, and alarms that adapt to user behavior.

Q.4. What are the Characteristics of Artificial Intelligence (AI)?

Ans: Artificial Intelligence (AI) is a branch of computer science that enables machines to perform tasks that typically require human intelligence. These tasks include learning from experience, reasoning, understanding language, problem-solving, and making decisions.

The key characteristics of AI systems help define their behavior and capabilities. Let's explore them one by one with relevant examples:

1. Learning Ability

Learning is a fundamental characteristic of AI. It refers to the ability of an AI system to acquire data, process it, and use it to improve future performance without being explicitly programmed.

Types of Learning:

- **Supervised Learning:** AI learns from labeled data.
- **Unsupervised Learning:** AI finds patterns in unlabeled data.
- **Reinforcement Learning:** AI learns through feedback (rewards or penalties).

2. Reasoning

Reasoning allows AI to draw conclusions from facts and apply logic to solve problems. It includes both deductive and inductive reasoning.

3. Problem-Solving
Problem-solving is the AI‘s ability to identify the best solution among many possible options. AI uses techniques like heuristic search, rule-based systems, and optimization algorithms to solve complex problems.

4. Perception
Perception allows AI to interpret the world around it using sensory data such as images, sounds, or touch. This is often achieved through technologies like computer vision, speech recognition, and sensor processing.

5. Language Understanding (Natural Language Processing – NLP)
Language understanding enables AI to read, understand, interpret, and generate human language. This allows seamless communication between humans and machines.
.

6. Autonomy
Autonomy means the AI system can perform tasks without human guidance. Autonomous systems can make their own decisions based on pre-learned patterns and real-time data.

7. Adaptability
Adaptability refers to the AI system‘s ability to change its behavior or adjust its operations based on changes in the environment or data.

8. Interaction with Environment
AI systems can interact with their environment through sensors and actuators. They collect data, analyze it, and take actions to modify or respond to their surroundings.

9. Decision Making
AI systems can analyze data, evaluate possible actions, and make decisions. This process often involves probability, logic, and optimization to choose the best action.

10. Goal-Oriented Behavior
AI systems often operate with specific goals in mind. They plan and execute actions to achieve these goals efficiently.

11. Self-Correction and Improvement

An AI system can analyze its performance and learn from mistakes to improve in the future. This is known as feedback learning or continuous improvement.

12. Planning

Planning allows AI to set objectives, define steps, and organize actions in a logical sequence to achieve desired results.

13. Emotional Intelligence (Emerging Characteristic)

Some advanced AI systems are designed to recognize, interpret, and respond to human emotions. This helps in building more natural and empathetic human-machine interactions.

Q.5 Compare AI, Machine Learning, and Deep Learning.

Ans. **Comparison of AI, Machine Learning, and Deep Learning**

S.no.	**Aspect**	**Artificial Intelligence (AI)**	**Machine Learning (ML)**	**Deep Learning (DL)**
1	**Definition**	The broader concept of machines being able to carry out tasks smartly.	A subset of AI where machines learn from data without explicit programming.	A subset of ML using neural networks with multiple layers for learning.
2	**Scope**	Broad – includes reasoning, decision-making, perception, learning, etc.	Narrower – focuses only on learning from data.	More specific – focuses on deep neural network architectures.
3	**Goal**	To simulate human intelligence.	To enable systems to learn and make predictions from data.	To enable systems to learn complex patterns and representations from data.

4	**Techniques Used**	Rule-based systems, logic, learning, NLP, robotics, etc.	Statistical models, decision trees, SVM, clustering, regression.	Neural networks, CNNs, RNNs, transformers, etc.
5	**Data Dependency**	Can work with less or structured data.	Requires a decent amount of structured data.	Requires a large amount of unstructured data.
6	**Human Intervention**	High – often needs manual rule creation and programming.	Moderate – algorithms learn from data but may need feature engineering.	Low – minimal feature engineering, as systems learn features automatically.
7	**Hardware Requirements**	Basic computing resources.	Moderate – standard CPUs can handle ML tasks.	High – needs powerful GPUs or TPUs for training models.
8	**Examples**	Virtual assistants, chatbots, robotics, game AI.	Spam filters, recommendation systems, fraud detection.	Image recognition, voice assistants, language translation.
9	**Learning Approach**	Includes both learning and non-learning approaches.	Learning-based (supervised, unsupervised, reinforcement).	Uses deep learning algorithms (mainly supervised and unsupervised).

10	**Accuracy**	Varies depending on system design and approach.	Generally accurate with good data and tuning.	Very high accuracy with large datasets.
11	**Complexity**	Can be simple or highly complex.	Moderate complexity.	High complexity due to layered architectures.

Q.6. What are Search and Control Strategies in AI?

Ans: In Artificial Intelligence, especially in problem-solving and planning, **search** is the process of finding a sequence of actions that leads from an initial state to a goal state. However, just having a search method isn't enough — we need strategies to **control how the search is carried out** efficiently. That‘s where **search strategies** and **control strategies** come into play.

Search Strategy-

There are two types of strategies that describe a solution for a given problem:

Uninformed Search (Blind Search)-

This type of search strategy does not have any additional information about the states except the information provided in the problem definition. They can only generate the successors and distinguish a goal state from a non-goal state. These type of search does not maintain any internal state, that‘s why it is also known as Blind search.

There are following types of uninformed searches:

1. Breadth-first search
2. Uniform cost search
3. Depth-first search
4. Depth-limited search
5. Iterative deepening search
6. Bidirectional search

Informed Search (Heuristic Search) -This type of search strategy contains some additional information about the states beyond the problem definition. This search uses problem-specific knowledge to find more efficient solutions. This search maintains some sort of internal states via heuristic functions (which provides hints), so it is also called heuristic search.

There are following types of informed searches:

1. Best first search (Greedy search)
2. A search

Control Strategy:-

A **control strategy** is a set of rules or guidelines that govern how the search is conducted. It ensures that the search is **efficient**, **systematic**, and **goal-directed**. It controls **which node to expand next**, **what to do with visited nodes**, and **how to manage memory**.

Types of Control Strategies:

Control strategies are generally evaluated based on the following characteristics:

S.No.	**Control Feature**	**Explanation**
1	**Completeness**	Will the strategy always find a solution if one exists?
2	**Time Complexity**	How much time will the strategy take to find the solution?
3	**Space Complexity**	How much memory does the strategy require?
4	**Optimality**	Does the strategy find the least-cost or best solution?

Common Control Strategies:

S.No.	**Control Strategy**	**Description**	**Example**
1	**Forward Search**	Starts from the initial state and explores toward the goal state.	Solving a maze from entrance to exit
2	**Backward Search**	Starts from the goal state and works backward to reach the initial state.	Planning tasks backward from a deadline

3	**Depth Limiting**	Restricts how deep the search tree can go.	Prevents infinite loops in DFS
4	**Iterative Deepening**	Repeatedly uses depth-limited search with increasing limits.	Combines advantages of BFS and DFS
5	**Bidirectional Search**	Simultaneously searches from both initial and goal states to meet in the middle.	Efficient in large state spaces (like graphs)

Q.7. What is Production System Explain in Detail.

Ans: A **Production System** in AI is a formal framework used to model intelligent problem-solving and reasoning. It is structured to simulate how humans think and make decisions by applying a set of rules to a collection of data. These systems use rules to process information and make logical conclusions, often used in fields like expert systems, automated decision-making, and planning. Let's break down the key components, working mechanisms, and types of production systems in greater detail.

Key Components of a Production System

A **Production System** in AI consists of the following elements:

1. **Production Rules (or Rules of Inference)**:

   Format: These rules are typically expressed in an "if-then" format (i.e., condition-action pairs).

   Structure:

   Condition (IF): Specifies when the rule should be applied (e.g., the presence of certain facts).

   Action (THEN): Specifies what action to take if the condition is true (e.g., updating knowledge or deriving new facts).

   Example:

- If a person is under 18 years old, then classify as a minor.
- If the patient has a fever and cough, then suspect flu.

2. **Knowledge Base**:

   The knowledge base is a collection of facts, symbols, or data that represent knowledge about the world.

   It contains **facts** (representing real-world information) and **rules** (which encode reasoning processes).

It is dynamic: as rules are applied, the knowledge base is updated with new facts or conclusions.

3. **Working Memory**:

   This is a temporary storage that holds the current state of the system, representing the facts or data that are being processed at a given moment.

   It is the area where information from the knowledge base and the results of rule applications are stored.
4. **Inference Engine**:

   The inference engine is the component that actually applies the rules to the knowledge base and working memory to derive new information.

   It selects which rule to apply, applies the rule to the working memory, and updates the working memory accordingly.

   The inference engine may also decide whether the application of a rule is a "goal" state, meaning it has reached a solution.
5. **Control Strategy**:

   The control strategy governs the sequence in which rules are applied and how the system decides which rule to use next.

   It‘s a mechanism for deciding the next action:

- **Forward Chaining**: The system starts from the initial facts (data) and applies rules to infer new facts, working forward from the known information.
- **Backward Chaining**: The system starts with a goal (desired outcome) and works backward, applying rules to figure out what facts are needed to achieve that goal.

Production Systems Work

Let‘s break down the working of a production system:

1. **Initialization**:

   The system begins with an initial set of facts stored in the working memory and a set of production rules stored in the knowledge base.

   Rule Selection:

   The inference engine looks at the conditions of the rules and identifies which rules can be applied to the facts in the working memory. If the condition of a rule matches the current facts, the rule is "fired."

2. **Rule Application**:

   Once a rule is selected, the corresponding action is executed, often updating the knowledge base or working memory. The inference engine may add new facts or modify existing facts.
3. **Iteration**:

   The process repeats: the system continuously applies rules, updating its working memory and knowledge base, until a goal is achieved or no more rules can be applied (indicating that the process has either found a solution or cannot proceed).
4. **Termination**:

   A production system typically terminates when the system reaches a predefined goal state or when no further rules can be applied. In some cases, the system may enter a state where no applicable rules exist, causing it to halt (indicating failure).

Example of a Production System in Action

Consider an AI system designed to assist in medical diagnosis:

- **Production Rules**:

Rule 1: If the patient has a fever, then it may be an infection.

Rule 2: If the patient has a cough, then it may be a respiratory illness.

Rule 3: If the patient has both a fever and cough, then it may be the flu.

- **Knowledge Base**:

Facts: Patient has a fever.

Rules: As listed above.

- **Working Memory**:

  Contains current facts such as "Patient has a fever."

- **Inference Engine**:

The inference engine would check the rules:

Rule 1 applies because the patient has a fever, so the system infers that it may be an infection.

Rule 2 does not apply yet because the patient doesn't have a cough.

If the patient later reports a cough, Rule 3 could be triggered.

Q.8. Define Water Jug Problem in Detail.

Ans. Water Jug Problem –

The **Water Jug Problem** is a classic puzzle involving two jugs with different capacities. The goal is to use a set of operations to measure out a specific amount of water.

- ✓ Given:

i. Jug A with capacity a liters

ii. Jug B with capacity b liters

iii. Target: measure exactly c liters of water

- ✓ **Allowed Operations:**

i. **Fill** either jug completely.

ii. **Empty** either jug.

iii. **Pour** water from one jug to the other, until one is either full or empty.

- ✓ **State Representation:**

Each state is a pair (x, y):

i. x is the current amount of water in Jug A

ii. y is the current amount of water in Jug B

- ✓ **Necessary Condition for Solution:**

i. A solution exists **only if**:

ii. c is divisible by the **GCD (greatest common divisor)** of a and b,

and $c \leq \max(a, b)$

- ✓ **BFS Algorithm for Water Jug Problem**
- ✓ **Goal: Find a path from (0, 0) to any state (x, y) where x == c or y == c.**

- ✓ **Algorithm:**

1. Create a queue to store the states (x, y).
2. Create a set to keep track of visited states to avoid repetition.
3. Enqueue the initial state (0, 0).

```
4. While the queue is not empty:
   a. Dequeue the front state (x, y).
   b. If x == c or y == c → Goal is achieved → Return the path or "Success".
   c. If (x, y) is already visited → continue
   d. Mark (x, y) as visited.
   e. Generate all possible next states using the allowed operations:
      - Fill Jug A: (a, y)
      - Fill Jug B: (x, b)
      - Empty Jug A: (0, y)
      - Empty Jug B: (x, 0)
      - Pour A → B: min(x, b - y) poured → (x - poured, y + poured)
      - Pour B → A: min(y, a - x) poured → (x + poured, y - poured)
   f. Enqueue all these new states if not visited.
5. If no solution is found → Return "No solution".
```

✓ **Example (4-liter and 3-liter jugs, target = 2):**

i. Jug A: 4 liters

ii. Jug B: 3 liters

iii. Target = 2 liters

✓ **BFS will explore**:

Initial State -(0,0)

→ (4,0) [Fill A]

→ (0,3) [Fill B]

→ (4,3) [Fill A, Fill B]

→ (1,3) [Pour A → B]

→ (1,0) [Empty B]

→ ...

Eventually → (2,0) or (0,2)

✓ **Time and Space Complexity**

✓ Time Complexity: **O(a b), since the number of possible states is at most a b**

✓ Space Complexity: **O(a b) for visited states and queue**

Unit – 02

Searching & Knowledge Representation

Q.1 Explain Uninformed Search in Detail.

Ans: Uninformed search (also known as blind search) refers to search strategies in artificial intelligence where the algorithm searches through the state space without any domain-specific knowledge or information about the goal state. Essentially, the algorithm doesn't know anything about the specific nature of the problem other than the basic structure of the states and the possible transitions between them.

In uninformed search, the algorithm explores the state space using only the available information about the problem and operates without heuristics. It tries all possible options, often in a systematic way, until it reaches the goal state.

Key Characteristics of Uninformed Search:

1. No Prior Knowledge: Uninformed search doesn‘t have any additional information about the goal state, other than the fact that it exists somewhere in the state space.
2. Explores All Possible States: The algorithm may explore all states, including states that are far from the goal, before it reaches the correct solution.
3. General Approach: Uninformed search algorithms are applicable to a wide range of problems but can be inefficient for large or complex problems.

Common Types of Uninformed Search Algorithms

1. Breadth-First Search (BFS)

BFS is an uninformed search algorithm that explores all nodes at the present depth level before moving on to nodes at the next depth level.

- How it works:

1. Start at the root node (initial state).
2. Explore all neighboring nodes (states) at the current depth.
3. Move to the next level, exploring neighbors of all nodes at that level.
4. Continue this process until the goal state is found or all states are exhausted.

✓ **Properties:**

- Complete: BFS is guaranteed to find the solution if one exists (i.e., it won‘t miss the goal).
- Optimal: It will find the shortest path to the goal if the path cost is uniform.

- Time Complexity: O(b^d), where `b` is the branching factor (number of children per node) and `d` is the depth of the shallowest goal.
- Space Complexity: O(b^d), as it needs to store all the nodes in memory.

Example of BFS :-

✓ **Algorithm: Breadth-First Search**

➢ **Input:**

A problem instance with:

- INITIAL-STATE
- ACTIONS(state)
- GOAL-TEST(state)
- CHILD-NODE(problem, node, action)

Output:

A solution (path to goal) or failure

```
1. Create node with:
      node.STATE ← problem.INITIAL-STATE
      node.PATH-COST ← 0
2. If problem.GOAL-TEST(node.STATE) then
         return SOLUTION(node)
3. Initialize frontier ← FIFO queue with node as the only element
4. Initialize explored ← empty set
5. Loop do:
      a. If EMPTY?(frontier) then
            return failure
      b. node ← POP(frontier)     // shallowest node
      c. Add node.STATE to explored
      d. For each action in problem.ACTIONS(node.STATE) do:
         i.   child ← CHILD-NODE(problem, node, action)
         ii.  If child.STATE not in explored or frontier then
                 1. If problem.GOAL-TEST(child.STATE) then
                       return SOLUTION(child)
                 2. INSERT(child, frontier)
```

BFS illustrated:

Step 1: Initially frontier contains only one node corresponding to the source state A.

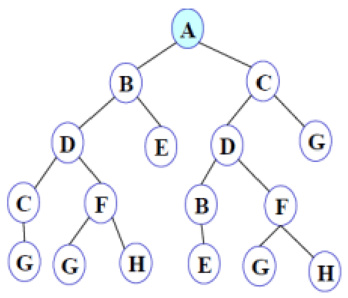


Figure 1

Frontier: A

Step 2: A is removed from fringe. The node is expanded, and its children B and C are generated. They are placed at the back of fringe.

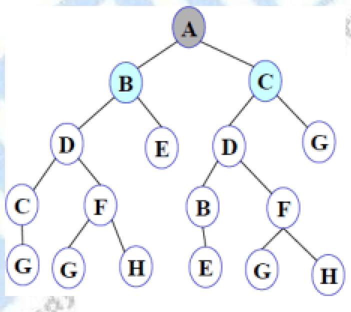


Figure 2

Frontier: B C

Step 3: Node B is removed from fringe and is expanded. Its children D, E are generated and putat the back of fringe.

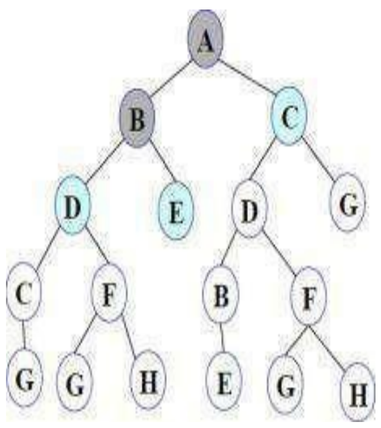
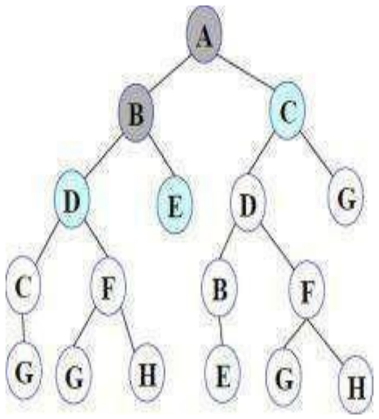


Figure 3

Frontier: C D E



Step 4: Node C is removed from fringe and is expanded. Its children D and G are added to theback of fringe.

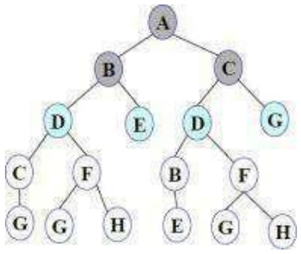


Figure 4

Frontier: D E D G

Step 5: Node D is removed from fringe. Its children C and F are generated and added to the backof fringe.

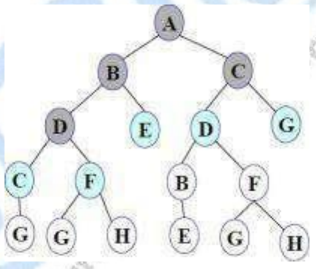


Figure 5

Frontier: E D G C F

Step 6: Node E is removed from fringe. It has no children.

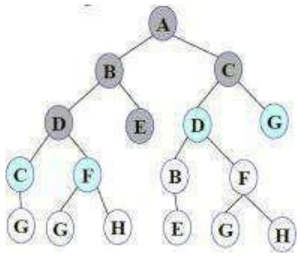


Figure 6

Frontier: D G C F

Step 7: D is expanded; B and F are put in OPEN.

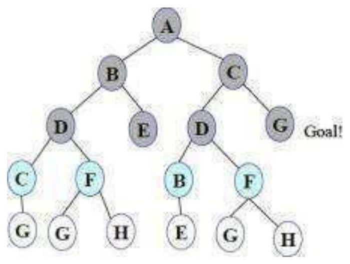


Figure 7

Frontier: G C F B F

Step 8: G is selected for expansion. It is found to be a goal node. So the algorithm returns the path A C G by following the parent pointers of the node corresponding to G. The algorithm terminates.

Advantages:

- One of the simplest search strategies
- Complete. If there is a solution, BFS is guaranteed to find it.
- If there are multiple solutions, then a minimal solution will be found
- The algorithm is optimal (i.e., admissible) if all operators have the same cost.Otherwise, breadth first search finds a solution with the shortest path length.
- **Time complexity** : $O(b^d)$
- **Space complexity** : $O(b^d)$
- **Optimality** :Yes

Disadvantages:

- Requires the generation and storage of a tree whose size is exponential the depth of theshallowest goal node.
- The breadth first search algorithm cannot be effectively used unless the search space isquite small.

Q.2. Explain Depth-First Search (DFS) in Detail.

We may sometimes search the goal along the largest depth of the tree, and move up only when further traversal along the depth is not possible. We then attempt to find alternative offspring of the parent of the node (state) last visited. If we visit the nodes of a tree using the above principlesto search the goal, the traversal made is called depth first traversal and consequently the search strategy is called ***depth first search***.

Main Concept:-

Start at a node → Visit it → Recursively (or using a stack) visit all its unvisited neighbors → Go as deep as possible → Backtrack when no unvisited neighbors are left.

RECURSIVE-DLS(node, problem, limit)

Input:

- node: current node in the search tree
- problem: the search problem
- limit: the remaining depth allowed

Output:

- A solution, 'cutoff', or 'failure'

Algorithm:

```
RECURSIVE-DLS(node, problem, limit)
1.  if problem.GOAL-TEST(node.STATE) then
2.  return SOLUTION(node)
3.  else if limit = 0 then
4.  return 'cutoff'
5.  else
6.  cutoff_occurred ← false
7.  for each action in problem.ACTIONS(node.STATE) do
8.  child ← CHILD-NODE(problem, node, action)
9.  result ← RECURSIVE-DLS(child, problem, limit − 1)
10. if result = 'cutoff' then
11. cutoff_occurred ← true
12. else if result ≠ 'failure' then
13. return result
14. if cutoff_occurred then
15. return 'cutoff'
```

16. else

17. return 'failure'

DFS illustrated:

Step 1: Initially fringe contains only the node for A.

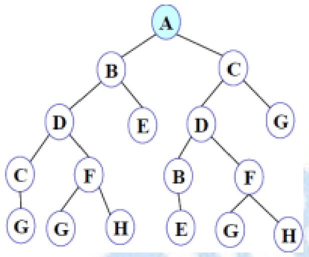


Figure 1

FRINGE: A

Step 2: A is removed from fringe. A is expanded and its children B and C are put in front of fringe.

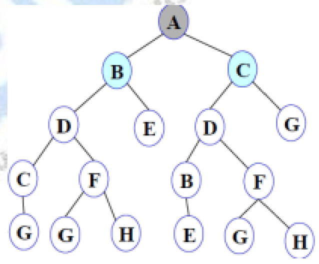
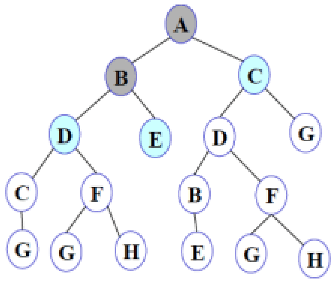


Figure 2

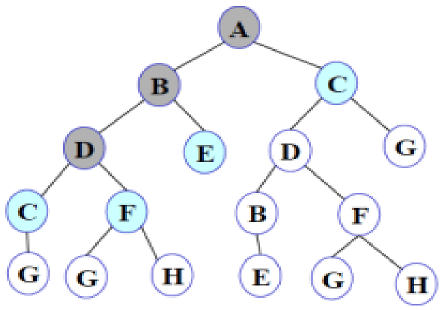
FRINGE: B C



Step 3: Node B is removed from fringe, and its children D and E are pushed in front of fringe.

Figure 3

FRINGE: D E C



Step 4: Node D is removed from fringe. C and F are pushed in front of fringe.

Figure 4

FRINGE: C F E C

Step 5: Node C is removed from fringe. Its child G is pushed in front of fringe.

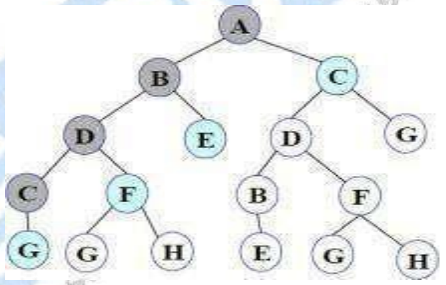


Figure 5

FRINGE: G F E C

Step 6: Node G is expanded and found to be a goal node.

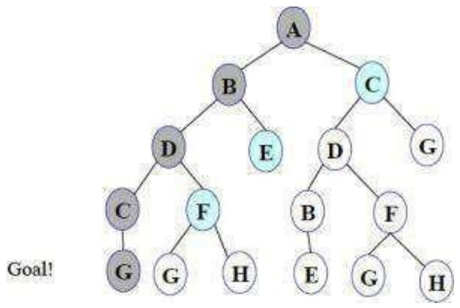


Figure 6

FRINGE: G F E C

The solution path A-B-D-C-G is returned and the algorithm terminates.

Depth first search

1. takes exponential time.
2. If N is the maximum depth of a node in the search space, in the worst case the algorithm will

   take time O(b).

3. The space taken is linear in the depth of the search tree, O(bN).

Note that the time taken by the algorithm is related to the maximum depth of the search tree. If the search tree has infinite depth, the algorithm may not terminate. This can happen if the search space is infinite. It can also happen if the search space contains cycles. The latter case can be handled by checking for cycles in the algorithm. Thus **Depth First Search is not complete.**

Time and Space Complexity

- **Time Complexity:** O(V + E), where V = vertices, E = edges
- **Space Complexity:** O(V) for visited set + recursion stack

Applications of DFS

- Path finding
- Cycle detection
- Topological sorting
- Connected components
- Puzzle solving (e.g., Sudoku, N-Queens)
-

Q. 3. **Define Informed search/Heuristic search in detail.**

Ans. Informed search/Heuristic search

A *heuristic* is a method that

- might not always find the best solution ***but*** is guaranteed to find a good solution inreasonable time. By sacrificing completeness it increases efficiency.

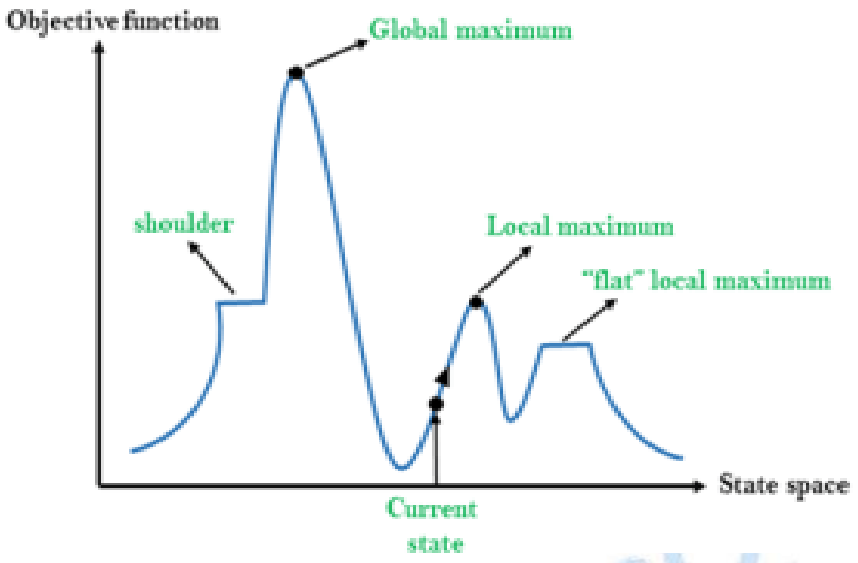
- Useful in solving tough problems which
  - could not be solved any other way.
  - solutions take an infinite time or very long time to compute.

Hill Climbing Algorithm

- Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution tothe problem. It terminates when it reaches a peak value where no neighbor has a higher value.
- It is also called greedy local search as it only looks to its good immediate neighbor stateand not beyond that.
- Hill Climbing is mostly used when a good heuristic is available.
- In this algorithm, we don't need to maintain and handle the search tree or graph as it onlykeeps a single current state.

  The idea behind hill climbingis as follows.

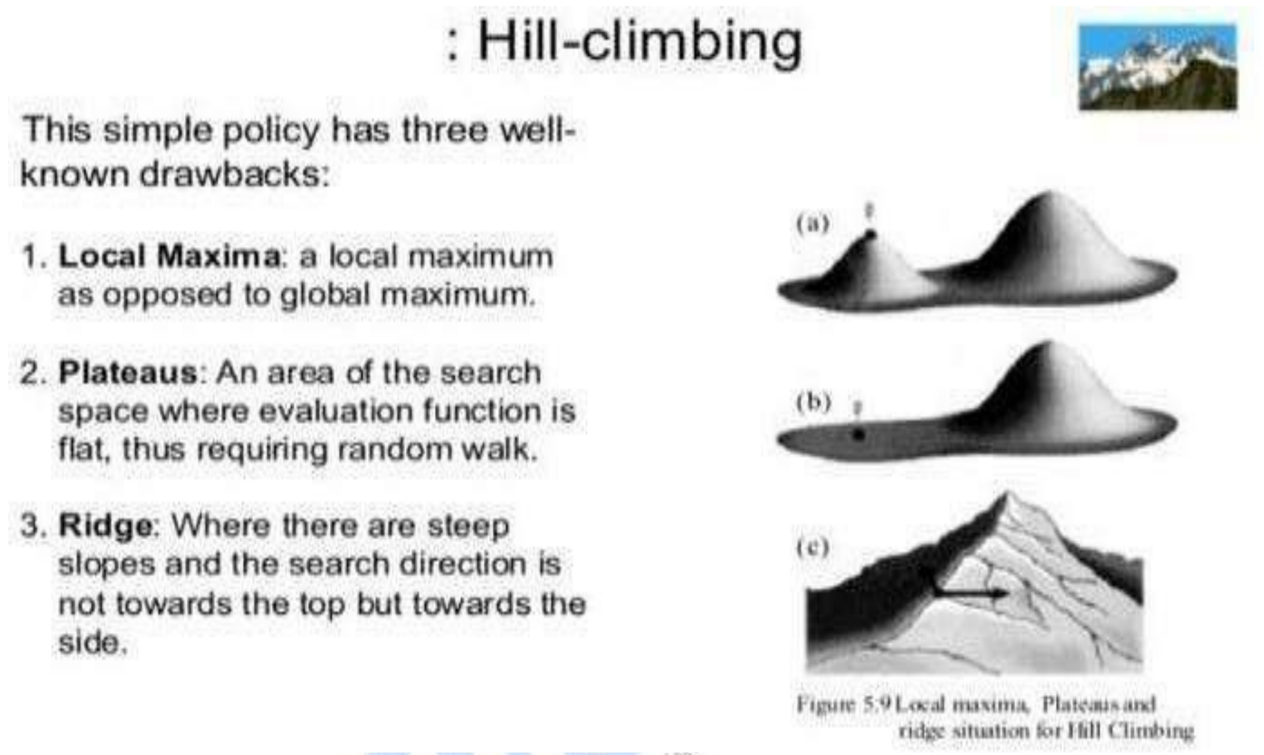
1. Pick a random point in the search space.
2. Consider all the neighbors of the current state.
3. Choose the neighbor with the best quality and move to that state.
4. Repeat 2 thru 4 until all the neighboring states are of lower quality.
5. Return the current state as the solution state.



Different regions in the state space landscape:

- **Local Maximum:** Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.
- **Global Maximum:** Global maximum is the best possible state of state space landscape. It has the highest value of objective function.
- **Current state:** It is a state in a landscape diagram where an agent is currently present.
- **Flat local maximum:** It is a flat space in the landscape where all the neighbor states of current states have the same value.
- **Shoulder:** It is a plateau region which has an uphill edge.

Problems in Hill Climbing Algorithm*:*



: Hill-climbing

This simple policy has three well-known drawbacks:

1. **Local Maxima**: a local maximum as opposed to global maximum.
2. **Plateaus**: An area of the search space where evaluation function is flat, thus requiring random walk.
3. **Ridge**: Where there are steep slopes and the search direction is not towards the top but towards the side.

Figure 5.9 Local maxima, Plateaus and ridge situation for Hill Climbing

Best First Search:

- A combination of depth first and breadth first searches.
- Depth first is good because a solution can be found without computing all nodes and breadth first is good because it does not get trapped in dead ends.
- The best first search allows us to switch between paths thus gaining the benefit of both approaches. At each step the most promising node is chosen. If one of the nodes chosen generates nodes that are less promising it is possible to choose another at the same level and in effect the search changes from depth to breadth. If on analysis these are no better than this previously unexpanded node and branch is not forgotten and the search method reverts to the
- **OPEN** is a priority queue of nodes that have been evaluated by the heuristic function but which have not yet been expanded into successors. The most promising nodes are at the front.

- **CLOSED** are nodes that have already been generated and these nodes must be stored because agraph is being used in preference to a tree.

Algorithm:

1. Start with OPEN holding the initial state
2. Until a goal is found or there are no nodes left on open do.

- Pick the best node on OPEN
- Generate its successors
- For each successor Do
- If it has not been generated before ,evaluate it ,add it to OPEN andrecord its parent

- If it has been generated before change the parent if this new path is betterand in that case update the cost of getting to any successor nodes.

- If a goal is found or no more nodes left in OPEN, quit, else return to 2.

Q.4.Define Constraint Satisfaction Problems in detail.

Ans: Sometimes a problem is not embedded in a long set of action sequences but requires picking the best option from available choices. A good general-purpose problem solving technique is to list the constraints of a situation (either negative constraints, like limitations, or positive elements that you want in the final solution). Then pick the choice that satisfies most of the constraints.

Formally speaking, a **constraint satisfaction problem (or CSP**) is defined by a set of variables, X1;X2;
: : :
;Xn, and a set of constraints, C1;C2; : : : ;Cm. Each variable Xi has anonempty domain Di of possible values. Each constraint Ci involves some subset of tvariables and specifies the allowable combinations of values for that subset. A state of theproblem is defined by an assignment of values to some or all of the variables, {Xi = vi;Xj =vj ; : : :} An assignment that does not violate any constraints is called a consistent or
legalassignment. A complete assignment is one in which every variable is mentioned, and a solution to a CSP is a complete assignment that satisfies all the constraints. Some CSPs also require a solution that

maximizes an objectivefunction.

CSP can be given an incremental formulation as a standard search problem as follows:

1. **Initial state**: the empty assignment fg, in which all variables are unassigned.

2. **Successor function**: a value can be assigned to any unassigned variable, provided that it doesnot conflict with previously assigned variables.
3. **Goal test**: the current assignment is complete.
4. **Path cost**: a constant cost for every step

Examples:

The best-known category of continuous-domain CSPs is that of **linear programming** problems, where constraints must be linear inequalities forming a *convex* region.

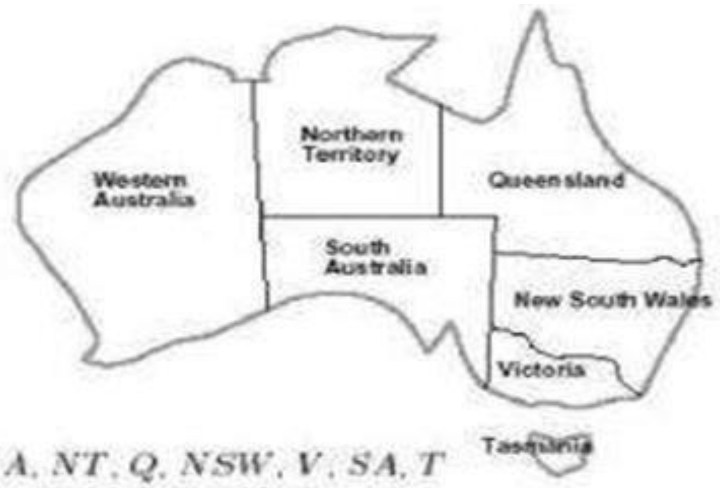
$$\begin{array}{cccc} & T & W & O \\ + & T & W & O \\ \hline F & O & U & R \end{array}$$

Crypt arithmetic puzzles.

Example: The map coloring problem.

- The task of coloring each region red, green or blue in such a way that no neighboringregions have the same color.
- We are given the task of coloring each region red, green, or blue in such a way that theneighboring regions must not have the same color.
- To formulate this as CSP, we define the variable to be the regions: WA, NT, Q, NSW, V, SA, and
- T. The domain of each variable is the set {red, green, blue}. The constraints require
- neighboring regions to have distinct colors: for example, the allowable combinations for WA and NTare the pairs

- {(red,green),(red,blue),(green,red),(green,blue),(blue,red),(blue,green)}. (The constraint can also berepresented as theinequality WA≠NT). Therearemany possible solutions,

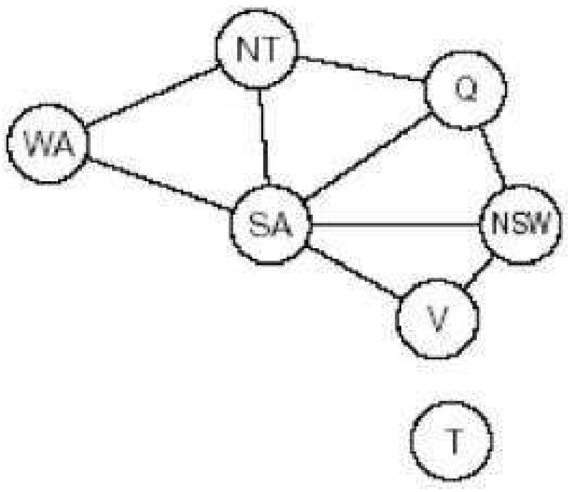


Variables WA, NT, Q, NSW, V, SA, T
Domains $D_i = \{red, green, blue\}$
Constraints: adjacent regions must have different colors
e.g., $WA \neq NT$ (if the language allows this), or
$(WA, NT) \in \{(red, green), (red, blue), (green, red), (green, blue), \ldots\}$

such as {WA = red, NT = green, Q = red, NSW = green, V = red, SA = blue, T = red}.Map of Australia showing each of its states and territories

Constraint Graph: A CSP is usually represented as an undirected graph, called constraint graph where the nodes are the variables and the edges are the binaryconstraints.

Constraint graph: nodes are variables, arcs show constraints



The map-coloring problem represented as constraintgraph. CSP can be viewed as a standard search problemas follows:

- **Initial state** : the empty assignment {},in which all variables are unassigned.
- **Successor function:** a value can be assigned to any unassigned variable, provided that it does not conflict with previously assigned variables.
- **Goal test:** the current assignment is complete.
- **Path cost:** a constant cost(E.g.,1) for every step.

AO*Search: (And-Or) Graph:

The Depth first search and Breadth first search given earlier for OR trees or graphs can be easily adopted by AND-OR graph. The main difference lies in the way termination conditions are determined, since all goals following an AND nodes must be realized; where as a single goal node following an OR node will do. So for this purpose we are using AO algorithm.

Like A algorithm here we will use two arrays and one heuristic function. **OPEN:**

It contains the nodes that has been traversed but yet not been marked solvable or unsolvable.

CLOSE:

It contains the nodes that have already been processed.6 7:The distance from current node to goal node.

Algorithm:

Step 1: Place the starting node into OPEN.

Step 2: Compute the most promising solution tree say T0.

Step 3: Select a node n that is both on OPEN and a member of T0. Remove it from OPEN and place it in CLOSE.

Step 4: If n is the terminal goal node then leveled n as solved and leveled all the ancestors of n assolved. If the starting node is marked as solved then success and exit.

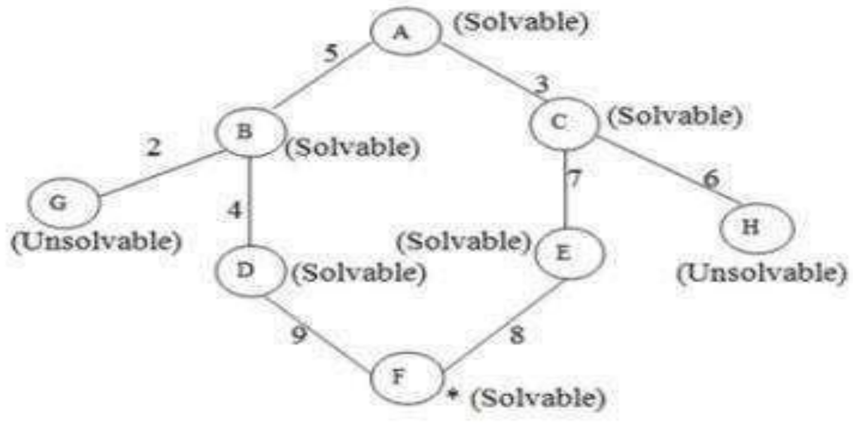
Step 5: If n is not a solvable node, then mark n as unsolvable. If starting node is marked as unsolvable,then return failure and exit.

Step 6: Expand n. Find all its successors and find their h (n) value, push them into OPEN. Step 7: Return to Step 2.

Step 8: Exit.

Implementation:

Let us take the following example to implement the AO algorithm.



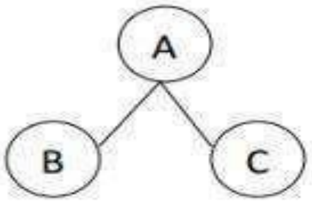
Figure

Step 1: In the above graph, the solvable nodes are A, B, C, D, E, F and the unsolvable nodes are G, H. Take A as the starting node. So place A into OPEN.

i.e. OPEN = A CLOSE = (NULL) Φ

The children of A are B and C which are solvable. So place them into OPEN and place A into the CLOSE.

i.e. OPEN = B C CLOSE = A

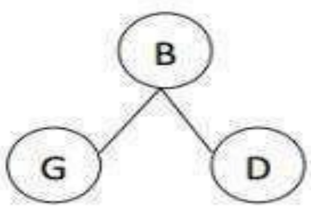


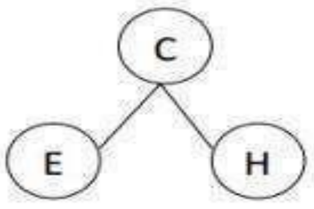
Step 3:

Now process the nodes B and C. The children of B and C are to be placed into OPEN. Also remove B and C from OPEN and place them into CLOSE.

So OPEN = G D E

C A B C





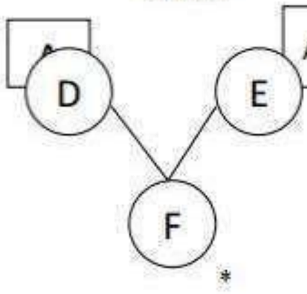
(O)

'O' indicated that the nodes G and H are unsolvable.

Step 4:

As the nodes G and H are unsolvable, so place them into CLOSE directly and process the nodes D and E.

i.e. OPEN = CLOSE =



A	B C		$G^{(O)}$	D E		$H^{(O)}$

Step 5:

Now we have been reached at our goal state. So place F into CLOSE.

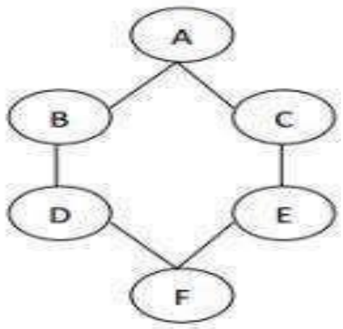
A	B C		$G^{(O)}$	D E		$H^{(O)}$	F

i.e. CLOSE =

Step 6:

Success and Exit

AO* Graph:



Figure

Q.5. Explain basic knowledge representation and reasoning:

Ans: Humans are best at understanding, reasoning, and interpreting knowledge. Human knows things,which is knowledge and as per their knowledge they perform various actions in the real world. But how machines do all these things comes under knowledge representation

- **There are three factors which are put into the machine, which makes it valuable:**
- **Knowledge:** The information related to the environment is stored in the machine.
- **Reasoning:** The ability of the machine to understand the stored knowledge.
- **Intelligence:** The ability of the machine to make decisions on the basis of the stored information.
- A knowledge representation language is defined by two aspects:
- The syntax of a language describes the possible configurations that can constitute sentences.
- The semantics determines the facts in the world to which the sentences refer.

For example, the syntax of the language of arithmetic expressions says that if x and y are expressions denoting numbers, then x > y is a sentence about numbers. The semantics of the languagesays that x > y is false when y is a bigger number than x, and true otherwise From the syntax and semantics, we can derive an inference mechanism for an agent that uses the language.

- Recall that the semantics of the language determine the fact to which a given sentence refers.Facts are part of the world,
- whereas their representations must be encoded in some way that can be physically stored within anagent. We cannot put the world inside a computer (nor can we put it inside a human), so all reasoning mechanisms must operate on representations of facts, rather than on the facts themselves.Because sentences are physical configurations of
- parts of the agent,

- Reasoning must be a process of constructing new physical configurations from old ones.
- Proper reasoning should ensure that the new configurations represent facts that actually follow from thefacts that the old configurations represent.
- We want to generate new sentences that are necessarily true, given that the old sentences are true.This relation between sentences is called entailment.
- In mathematical notation, the relation of entailment between a knowledge base KB and a sentence ais

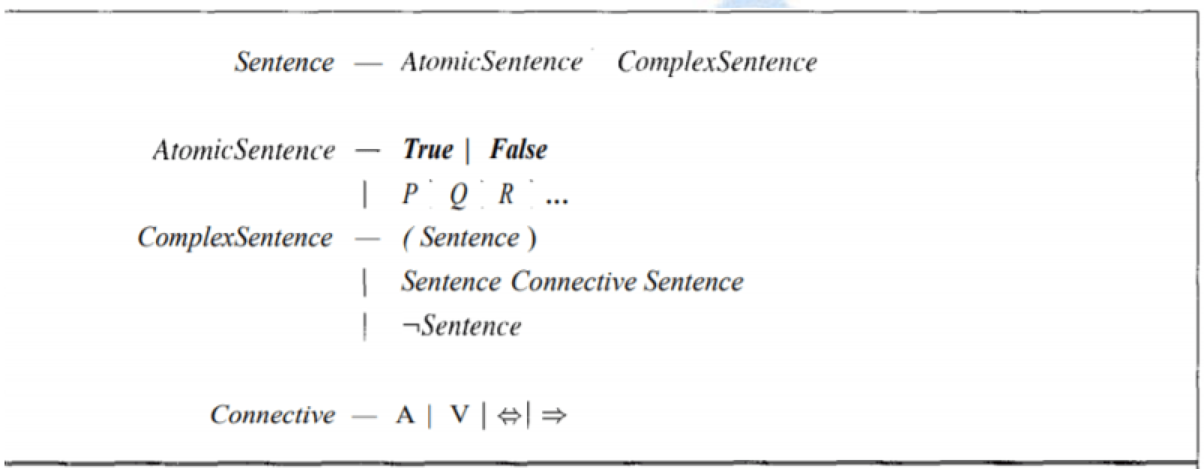
$$KB \models \alpha.$$

pronounced "KB entails a" and written as

- o An inference procedure can do one of two things:
- o given a knowledge base KB, it can generate new sentences a that purport to be entailed by KB.
- o E.g., x + y = 4 entails 4 = x + y
- o Entailment is a relationship between sentences (i.e., syntax) that is based on semantics

PROPOSITIONAL LOGIC:

- Propositional logic (PL) is the simplest form of logic where all the statements are made bypropositions.
- A proposition is a declarative statement which is either true or false.It is a technique of knowledge



```
        Sentence  —  AtomicSentence   ComplexSentence

  AtomicSentence  —  True | False
                  |  P   Q   R   ...
 ComplexSentence  —  ( Sentence )
                  |  Sentence Connective Sentence
                  |  ¬Sentence

      Connective  —  A | V | ⇔| ⇒
```

Figure 6.8 A BNF (Backus-Naur Form) grammar of sentences in propositional logic.

representation in logical and mathematical form

Syntax of propositional logic:

The symbols of prepositional logic are the logical constants True and False,

proposition symbolssuch as P and Q, the logical connectives A, V, <=>, =>and and parentheses,

All sentences are made by putting these symbols together using the following rules:

The logical constants True and False are sentences by themselves.

A prepositional symbol such as P or Q is a sentence by itself.

Wrapping parentheses around a sentence yields a sentence, for example, (P A Q).A sentence can be formed by combining simpler sentences with one of the five logical connectives::

Negation: A sentence such as ¬ P is called negation of P. A literal can be either Positive literal ornegative literal.

Example: P=Today is not Sunday -> ¬ p

1. **Conjunction:** A sentence which has ∧ connective such as, P ∧ Q is called a conjunction.Example: Rohan is intelligent and hardworking. It can be written as, P= Rohan is intelligent,
   Q= Rohan is hardworking. → P∧ Q.
2. **Disjunction**: A sentence which has ∨ connective, such as P ∨ Q. is called disjunction, where P andQ are the propositions.
   Example: "Ritika is a doctor or Engineer",
   Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as P ∨ Q.
3. **Implication:** A sentence such as P → Q, is called an implication. Implications are also known asif-then rules. It can be represented as If it is raining, then the street is wet.
   Let P= It is raining, and Q= Street is wet, so it is represented as P → Q
4. **Biconditional:** A sentence such as P⇔ Q is a Biconditional sentence, example If I ambreathing, then I am alive
   P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.Precedence of connectives:
   Precedence of connectives:

Semantics

- The semantics of prepositional logic is also quite straightforward. We define it by specifying the interpretation of the proposition symbols and constants, and specifying the meanings of the logical connectives.
- Validity
- Truth tables can be used not only to define the connectives, but also to test for valid sentences.
- Given a sentence, we make a truth table with one row for each of the possible combinations of truth values for the proposition symbols in the sentence.
- If the sentence is true in every row, then the sentence is valid. For example, the sentence ((P V H) A ¬H) => P
- Translating English into logic:
- User defines semantics of each propositional symbol
- P: It is Hot
- Q: It is Humid

- R:It is raining

Example:- If it is humid then it is hotQ->P

.If it is hot and humid , then it is raining(P A Q)->R

Limitations of Propositional logic:

- In propositional logic, we can only represent the facts, which are either true or false.
- PL is not sufficient to represent the complex sentences or natural language statements.
- The propositional logic has very limited expressive power.
- Consider the following sentence, which we cannot represent using PL logic.
- "Some humans are intelligent", or "Sachin likes cricket

Advantages of Propositional Logic

- The declarative nature of propositional logic, specify that knowledge and inference are separate, and inference is entirely domain-independent. Propositional logic is a declarative language because its semantics is based on a truth relation between sentences and possible worlds.
- It also has sufficient expressive power to deal with partial information, using disjunction and negation.
- Propositional logic has a third COMPOSITIONALITY property that is desirable in representation languages, namely, compositionality. In a compositional language, the meaning of a sentence is a function of the meaning of its parts. For example, the meaning of ―S1,4A S1,2‖ is related to the meanings of ―S1,4‖ and ―S1,2.

Drawbacks of Propositional Logic

- Propositional logic lacks the expressive power to concisely describe an environment with many objects.
- For example, we were forced to write a separate rule about breezes and pits for each square, such asB1,1⇔ (P1,2 ∨P2,1) .
- In English, itseemseasyenoughtosay,―Squares adjacenttopitsarebreezy.‖
- The syntax and semantics of English somehow make it possible to describe the environment concisely

SYNTAX AND SEMANTICS OF FIRST-ORDER LOGIC

Models for first-order logic :

The models of a logical language are the formal structures that constitute the possible worlds under consideration. Each model links the vocabulary of the logical sentences to elements of the possible world, so that the truth of any sentence can be determined. Thus, models for propositional logic link proposition symbols to predefined truth values. Models for first-order logic have objects. The domain of a model is the set of objects or domain elements it contains. The domain is required to be nonempty—every possible world must contain at least one object.
A relation is just the set of tuples of objects that are related.

Unary Relation: Relations relates to single Object Binary Relation: Relation Relates to multiple objects Certain kinds of relationships are best considered as functions, in that a given object must be related to exactly oneobject.

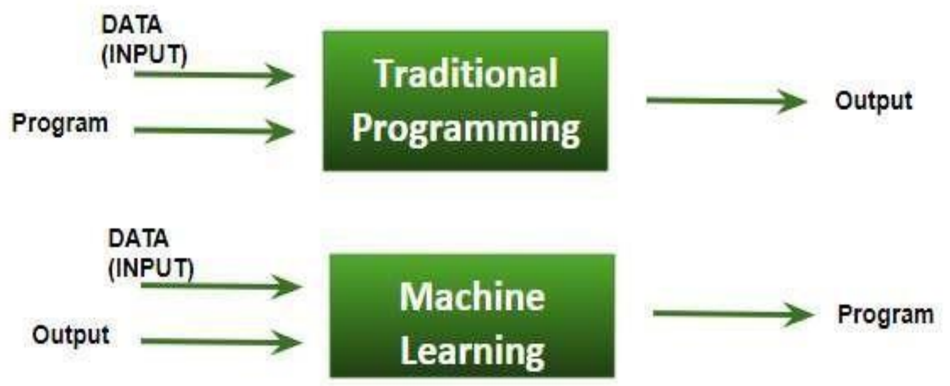
Unary Relation : John is a king Binary Relation :crown is on head of john , Richard is brother ofjohn Theunary "left leg" function includes the following mappings: (Richard the Lionheart) ->Richard's left leg (King John) ->Johns left Leg

UNIT- III

Machine-Learning & Supervised learning

Q.1. What is Machine Learning ?

Ans : In a very layman‘s manner, Machine Learning(ML) can be explained as automating and improving the learning process of computers based on their experiences without being actually programmed i.e. without any human assistance. The process starts with feeding good quality data and then training our



machines(computers) by building machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data do we have and what kind of task we are trying to automate. **Example: Training of students during exams.** While preparing for the exams students don't actually cram the subject but try to learn it with complete understanding. Before the examination, they feed their machine(brain) with a good amount of high - quality data (questions and answers from different books or teachers' notes, or online video lectures). Actually, they are training their brain with input as well as output i.e. what kind of approach or logic do they have to solve a different kinds of questions. Each time they solve practice test papers and find the performance (accuracy /score) by comparing answers with the answer key given, Gradually, the performance keeps on increasing, gaining more confidence with the adopted approach. That's how actually models are built, train machine with data (both inputs and outputs are given to the model), and when the time comes test on data (with input only) and achieve our model scores by comparing its answer with the actual output which has not been fed while training. Researchers are working with assiduous efforts to improve algorithms, and techniques so that these models perform even much better.

Definition: A computer program which learns from experience is called a machine learning program or simply a learning program .

Basic Difference in ML and Traditional Programming

- **Traditional Programming:** We feed in DATA (Input) + PROGRAM (logic), run it on the machine, and get the output.
- **Machine Learning:** We feed in DATA(Input) + Output, run it on the machine during training and the machine creates its own program(logic), which can be evaluated while testing.
- **What does exactly learning mean for a computer?** A computer is said to be learning from **Experiences** with respect to some class of **Tasks** if its performance in a given task improves with the Experience.
- 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** **Example:** playing checkers. **E** = the experience of playing many games of checkers **T** = the task of playing checkers. **P** = the probability that the program will win the next game In general, any machine learning problem can be assigned to one of two broad classifications: Supervised learning

and Unsupervised learning.

.

Definition of learning: 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 T, as measured by P , improves with experience E.

Examples

- Handwriting recognition learning problem
- Task T : Recognizing and classifying handwritten words within images
- Performance P : Percent of words correctly classified
- Training experience E : A dataset of handwritten words with given classifications
- A robot driving learning problem
- Task T : Driving on highways using vision sensors
- Performance P : Average distance traveled before an error
- Training experience E : A sequence of images and steering commands recorded while observing a human driver.

Q.2. Explain Classification of Machine Learning.

Ans: Machine learning implementations are classified into four major categories, depending on the nature
of the learning ―signal‖ or ―response‖ available to a learning system which are as follows:

A. Supervised learning:

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. The given data is labeled . Both *classification* and *regression* problems are supervised learning problems .

gender	age	label

M	48	sick
M	67	sick
F	53	healthy
M	49	sick
F	32	healthy
M	34	healthy
M	21	healthy

- Example — Consider the following data regarding patients entering a clinic . The data consists of
the gender and age of the patients and each patient is labeled as ―healthy‖ or ―sick‖.

B. Unsupervised learning:

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses. In unsupervised learning algorithms, classification or categorization is not included in the observations. Example: Consider the following data regarding patients entering a clinic. The data consists of the gender and age of the patients.

gender	age
M	48
M	67
F	53
M	49

F	34
M	21

As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find on the web in the form of marketing automation are based on this type of learning.

C. Reinforcement learning:

Reinforcement learning is the problem of getting an agent to act in the world so as to maximize its rewards.

A learner is not told what actions to take as in most forms of machine learning but instead must discover which actions yield the most reward by trying them. For example — Consider teaching a dog a new trick: we cannot tell him what to do, what not to do, but we can reward/punish it if it does the right/wrong thing.

When watching the video, notice how the program is initially clumsy and unskilled but steadily improves with training until it becomes a champion.

Semi-supervised learning:

Where an incomplete training signal is given: a training set with some (often many) of the target outputs missing. There is a special case of this principle known as Transduction where the entire set of problem instances is known at learning time, except that part of the targets are missing. Semi- supervised learning is an approach to machine learning that combines small labeled data with a large amount of unlabeled data during training. Semi-supervised learning falls between unsupervised learning and supervised learning.

Supervised learning

Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well labelled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labelled data.

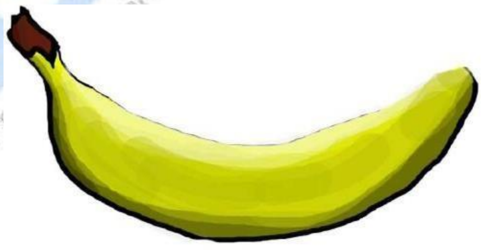
For instance, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all the different fruits one by one like this:

If the shape of the object is rounded and has a depression at the top, is red in color, then it will be labeled as –**Apple**.

If the shape of the object is a long curving cylinder having Green-Yellow color, then it will be labeled as –

Banana.

Now suppose after training the data, you have given a new separate fruit, say Banana from the basket, and asked to identify it.



Since the machine has already learned the things from previous data and this time has to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in the Banana category. Thus the machine learns the things from training data(basket containing fruits) and then applies the knowledge to test data(new fruit).

Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output.

In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher.

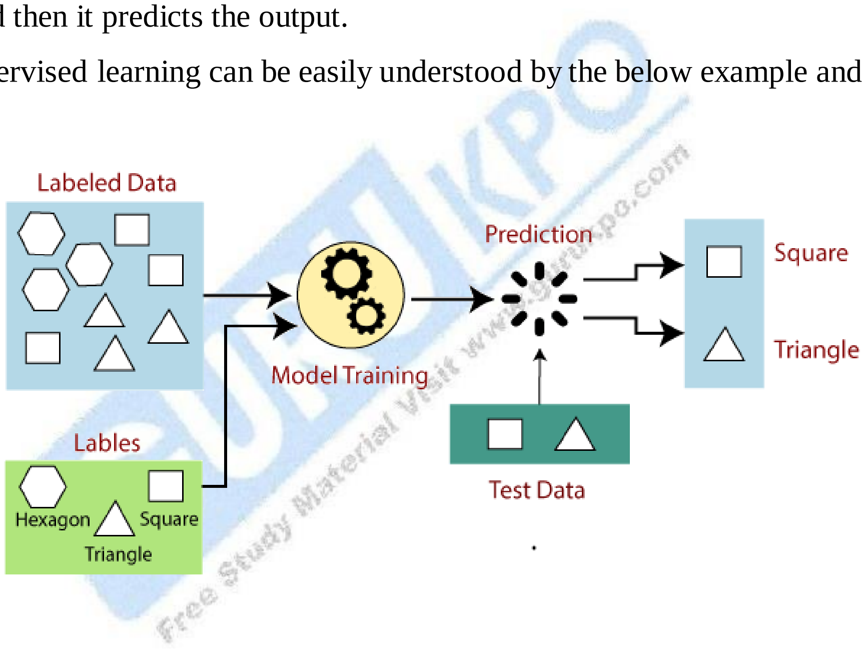
Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to **find a mapping function to map the input variable(x) with the output variable(y)**.

In the real-world, supervised learning can be used for **Risk Assessment, Image classification, Fraud Detection, spam filtering**, etc.

How Supervised Learning Works?

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

The working of Supervised learning can be easily understood by the below example and diagram:



Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a **Square**.
- If the given shape has three sides, then it will be labelled as a **triangle**.
- If the given shape has six equal sides then it will be labelled as **hexagon**.

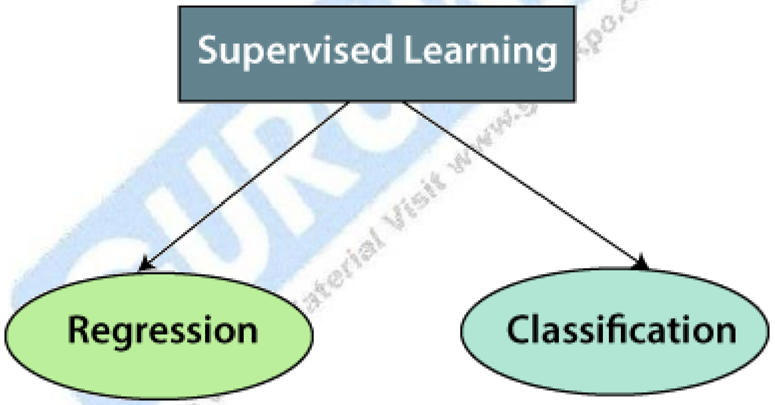
Now, after training, we test our model using the test set, and the task of the model is to identify the shape.

The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset, test dataset, and validation dataset**.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Types of supervised Machine learning Algorithms:



Supervised learning can be further divided into two types of problems:

1. Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

2. Classification

Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.
Spam Filtering,

- Random Forest
- Decision Trees
- Logistic Regression
- Support vector Machines

Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.
- Supervised learning model helps us to solve various real-world problems such as **fraud detection, spam filtering**, etc.

Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.
- In supervised learning, we need enough knowledge about the classes of object.

Supervised learning is classified into two categories of algorithms:

- **Classification**: A classification problem is when the output variable is a category, such as ―Red‖ or ―blue‖
, ―disease‖ or ―no disease‖.
- **Regression**: A regression problem is when the output variable is a real value, such as ―dollars‖ or ―weight‖.

Supervised learning deals with or learns with ―labeled‖ data. This implies that some data is already tagged
with the correct answer.

Types:-

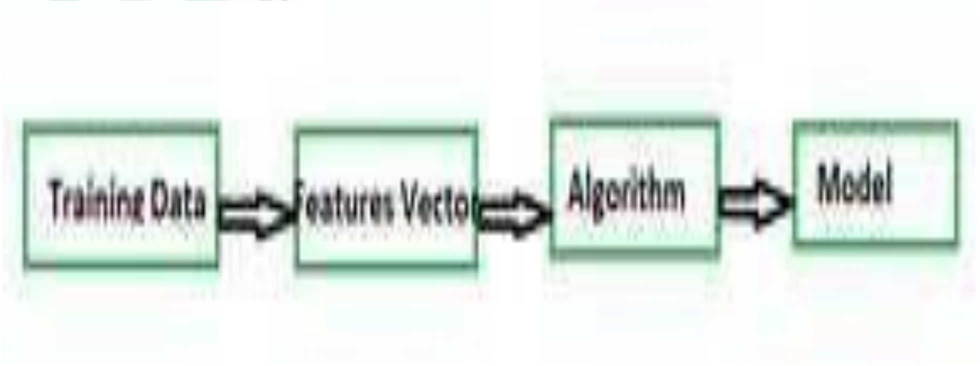
- Regression
- Logistic Regression
- Classification
- Naive Bayes Classifiers
- K-NN (k nearest neighbors)
- Decision Trees
- Support Vector Machine

Advantages:-

- Supervised learning allows collecting data and produces data output from previous experiences.
- Helps to optimize performance criteria with the help of experience.
- Supervised machine learning helps to solve various types of real-world computation problems.

Disadvantages:-

- Classifying big data can be challenging.
- Training for supervised learning needs a lot of computation time. So, it requires a lot of time.



Steps

<u>Unsupervised learning</u>

Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore the machine is restricted to find the hidden structure in unlabeled data by itself. **For instance**, suppose it is given an image having both dogs and cats which it has

never seen.

Thus the machine has no idea about the features of dogs and cats so we can‘t categorize it as _dogs and cats _. But it can categorize them according to their similarities, patterns, and differences, i.e., we can easily categorize the above picture into two parts. The first may contain all pics having **dogs** in them and the second part may contain all pics having **cats** in them. Here you didn‘t learn anything before, which means no training data or examples.

It allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with unlabelled data.

Unsupervised learning is classified into two categories of algorithms:

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

In the previous topic, we learned supervised machine learning in which models are trained using labeled data under the supervision of training data. But there may be many cases in which we do not have labeled data and need to find the hidden patterns from the given dataset. So, to solve such types of cases in machine learning, we need unsupervised learning techniques.

What is Unsupervised Learning?

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things. It can be defined as:

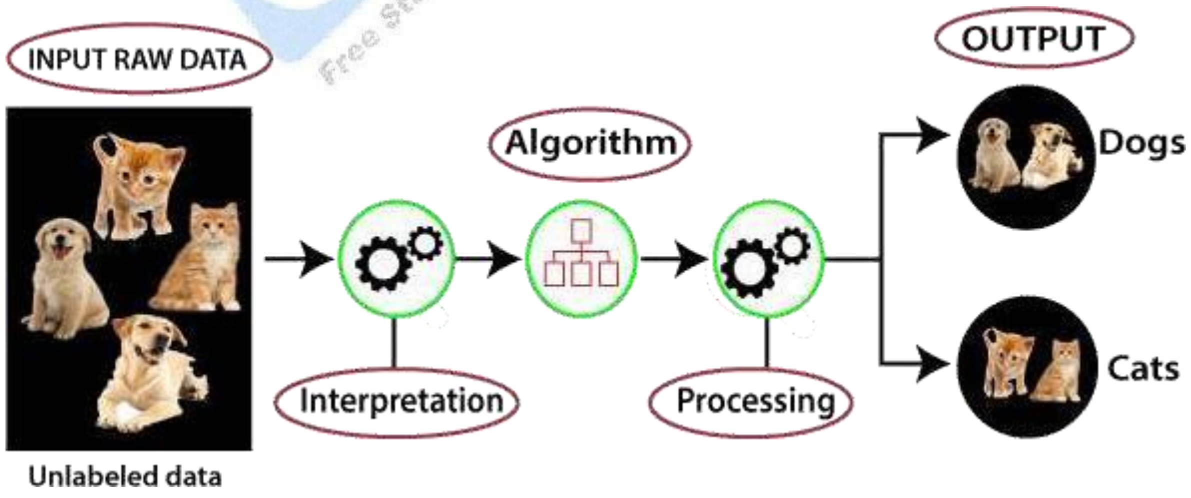
Q.3. Why use Unsupervised Learning?

Ans: Below are some main reasons which describe the importance of Unsupervised Learning:

- Unsupervised learning is helpful for finding useful insights from the data.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.
- In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

Working of Unsupervised Learning

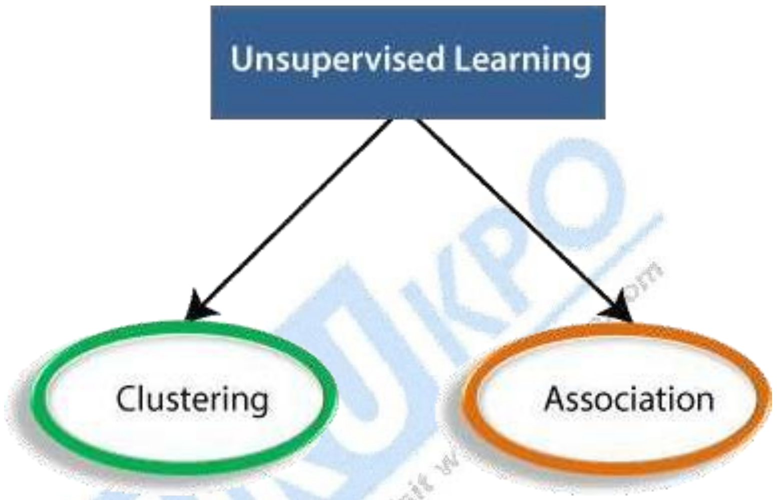
Working of unsupervised learning can be understood by the below diagram:



Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order

to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.
Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

Types of Unsupervised Learning Algorithm:
The unsupervised learning algorithm can be further categorized into two types of problems:



- **Clustering**: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.
- **Association**: An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

Unsupervised Learning algorithms:
Below is the list of some popular unsupervised learning algorithms:

- **K-means clustering**
- **KNN (k-nearest neighbors)**

- **Hierarchal clustering**
- **Anomaly detection**
- **Neural Networks**
- **Principle Component Analysis**
- **Independent Component Analysis**
- **Apriori algorithm**
- **Singular value decomposition**

Advantages of Unsupervised Learning

- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.
- Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

Types of Unsupervised Learning:-

Clustering

1. Exclusive (partitioning)
2. Agglomerative
3. Overlapping
4. Probabilistic

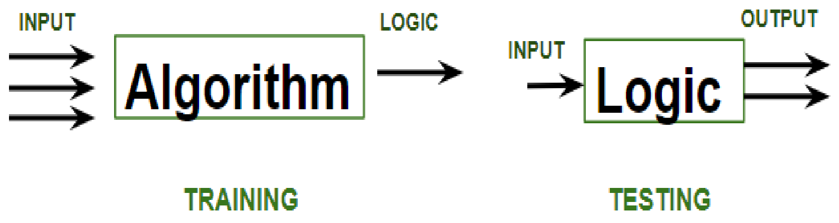
Clustering Types:-

1. Hierarchical clustering
2. K-means clustering
3. Principal Component Analysis
4. Singular Value Decomposition
5. Independent Component Analysis

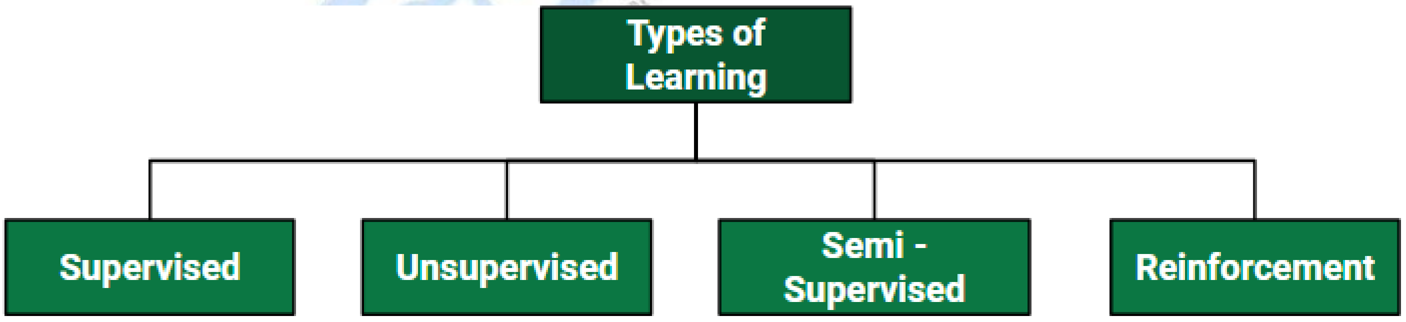
No. of classes	No. of classes is known	No. of classes is not known
Data Analysis	Uses offline analysis	Uses real-time analysis of data
Algorithms used	Linear and Logistics regression, Random forest, Support Vector Machine, Neural Network, etc.	K-Means clustering, Hierarchical clustering, Apriori algorithm, etc.

Q.4. Types of Learning – Supervised Learning

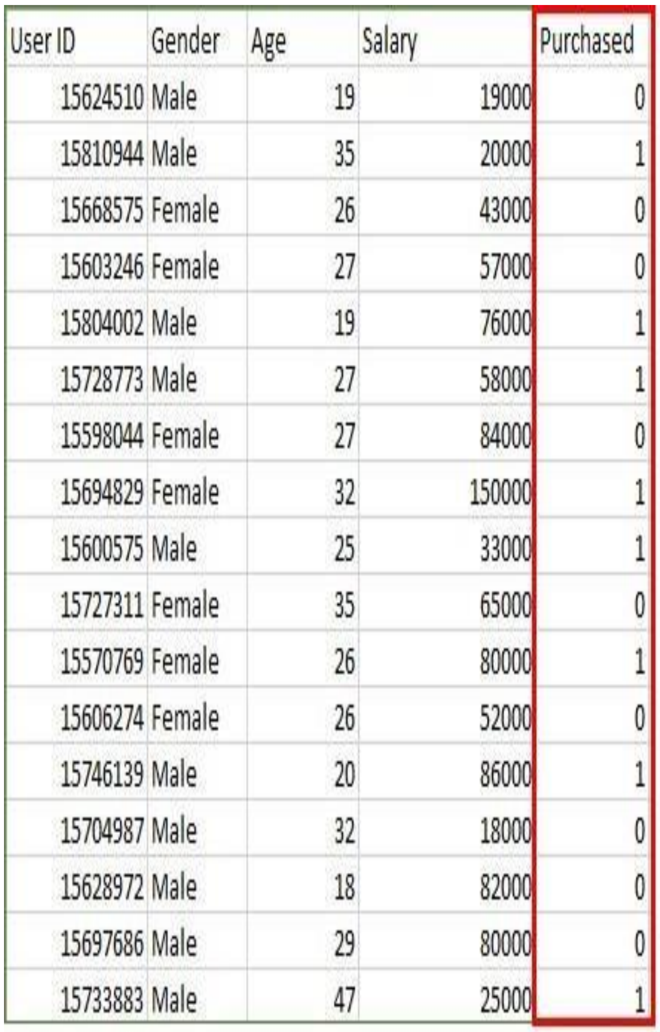
Ans. Let us discuss what is learning for a machine is as shown below media as follows:



A machine is said to be learning from **past Experiences**(data feed-in) with respect to some class of **tasks** if its **Performance** in a given Task improves with the Experience. For example, assume that a machine has to predict whether a customer will buy a specific product let‘s say ―Antivirus‖ this year or not. The machine will do it by looking at the **previous knowledge/past experiences** i.e the data of products that the customer had bought every year and if he buys Antivirus every year, then there is a high probability that the customer is going to buy an antivirus this year as well. This is how machine learning works at the basic conceptual level.

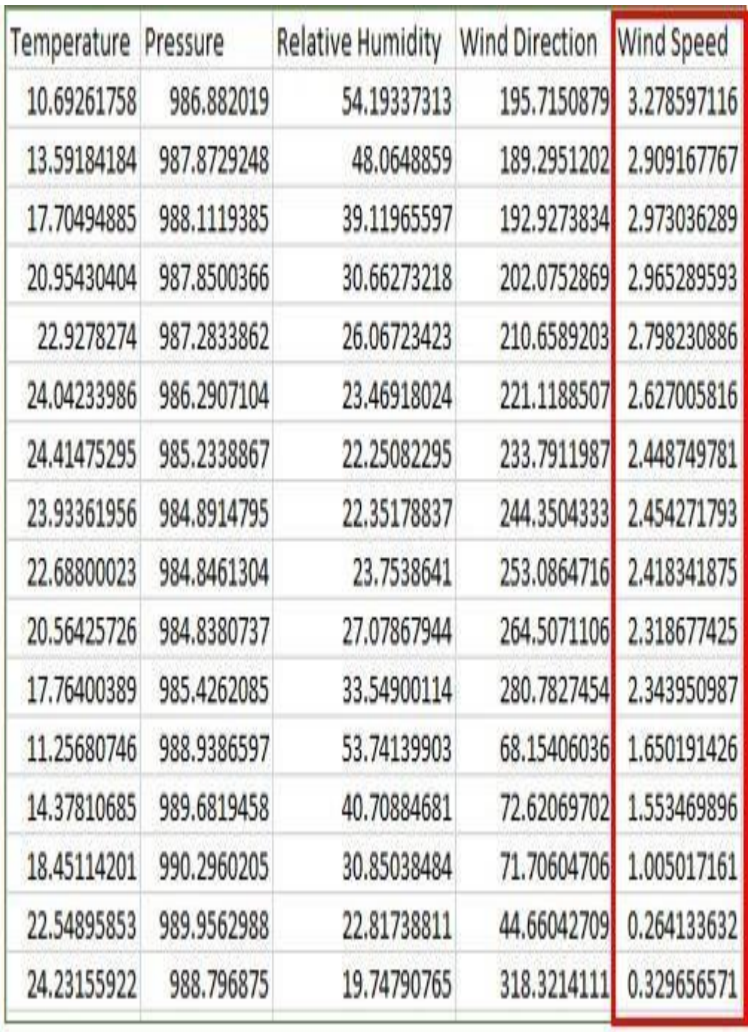


Supervised learning is when the model is getting trained on a labelled dataset. A **labelled** dataset is one that has both input and output parameters. In this type of learning both training and validation, datasets are labelled as shown in the figures below.



User ID	Gender	Age	Salary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	1
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	1
15728773	Male	27	58000	1
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	1
15727311	Female	35	65000	0
15570769	Female	26	80000	1
15606274	Female	26	52000	0
15746139	Male	20	86000	1
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1

Figure A: CLASSIFICATION



Temperature	Pressure	Relative Humidity	Wind Direction	Wind Speed
10.69261758	986.882019	54.19337313	195.7150879	3.278597116
13.59184184	987.8729248	48.0648859	189.2951202	2.909167767
17.70494885	988.1119385	39.11965597	192.9273834	2.973036289
20.95430404	987.8500366	30.66273218	202.0752869	2.965289593
22.9278274	987.2833862	26.06723423	210.6589203	2.798230886
24.04233986	986.2907104	23.46918024	221.1188507	2.627005816
24.41475295	985.2338867	22.25082295	233.7911987	2.448749781
23.93361956	984.8914795	22.35178837	244.3504333	2.454271793
22.68800023	984.8461304	23.7538641	253.0864716	2.418341875
20.56425726	984.8380737	27.07867944	264.5071106	2.318677425
17.76400389	985.4262085	33.54900114	280.7827454	2.343950987
11.25680746	988.9386597	53.74139903	68.15406036	1.650191426
14.37810685	989.6819458	40.70884681	72.62069702	1.553469896
18.45114201	990.2960205	30.85038484	71.70604706	1.005017161
22.54895853	989.9562988	22.81738811	44.66042709	0.264133632
24.23155922	988.796875	19.74790765	318.3214111	0.329656571

Figure B: REGRESSION

Both the above figures have labelled data set as follows:

- **Figure A:** It is a dataset of a shopping store that is useful in predicting whether a customer will purchase a particular product under consideration or not based on his/ her gender, age, and salary.

Input: Gender, Age, Salary

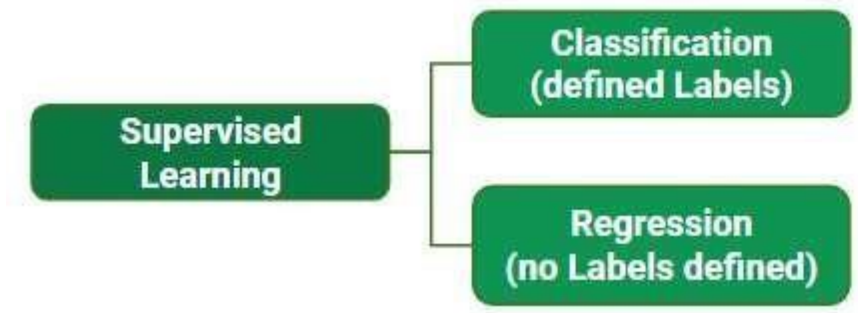
Output: Purchased i.e. 0 or 1; 1 means yes the customer will purchase and 0 means that the customer won‘t purchase it.

- **Figure B:** It is a Meteorological dataset that serves the purpose of predicting wind speed based on different parameters.

Input: Dew Point, Temperature, Pressure, Relative Humidity, Wind Direction

Output: Wind Speed

Types of Supervised Learning:


Supervised Learning
Classification (defined Labels)
Regression (no Labels defined)

A. **Classification:** It is a Supervised Learning task where output is having defined labels(discrete value). For example in above Figure A, Output – Purchased has defined labels i.e. 0 or 1; 1 means the customer will purchase, and 0 means that the customer won't purchase. The goal here is to predict discrete values belonging to a particular class and evaluate them on the basis of accuracy. It can be either binary or multi-class classification. In **binary** classification, the model predicts either 0 or 1; yes or no but in the case of **multi-class** classification, the model predicts more than one class. **Example:** Gmail classifies mails in more than one class like social, promotions, updates, and forums.

B. **Regression:** It is a Supervised Learning task where output is having continuous value. For example in above Figure B, Output – Wind Speed is not having any discrete value but is continuous in a particular range. The goal here is to predict a value as much closer to the actual output value as our model can and then evaluation is done by calculating the error value. The smaller the error the greater the accuracy of our regression model.

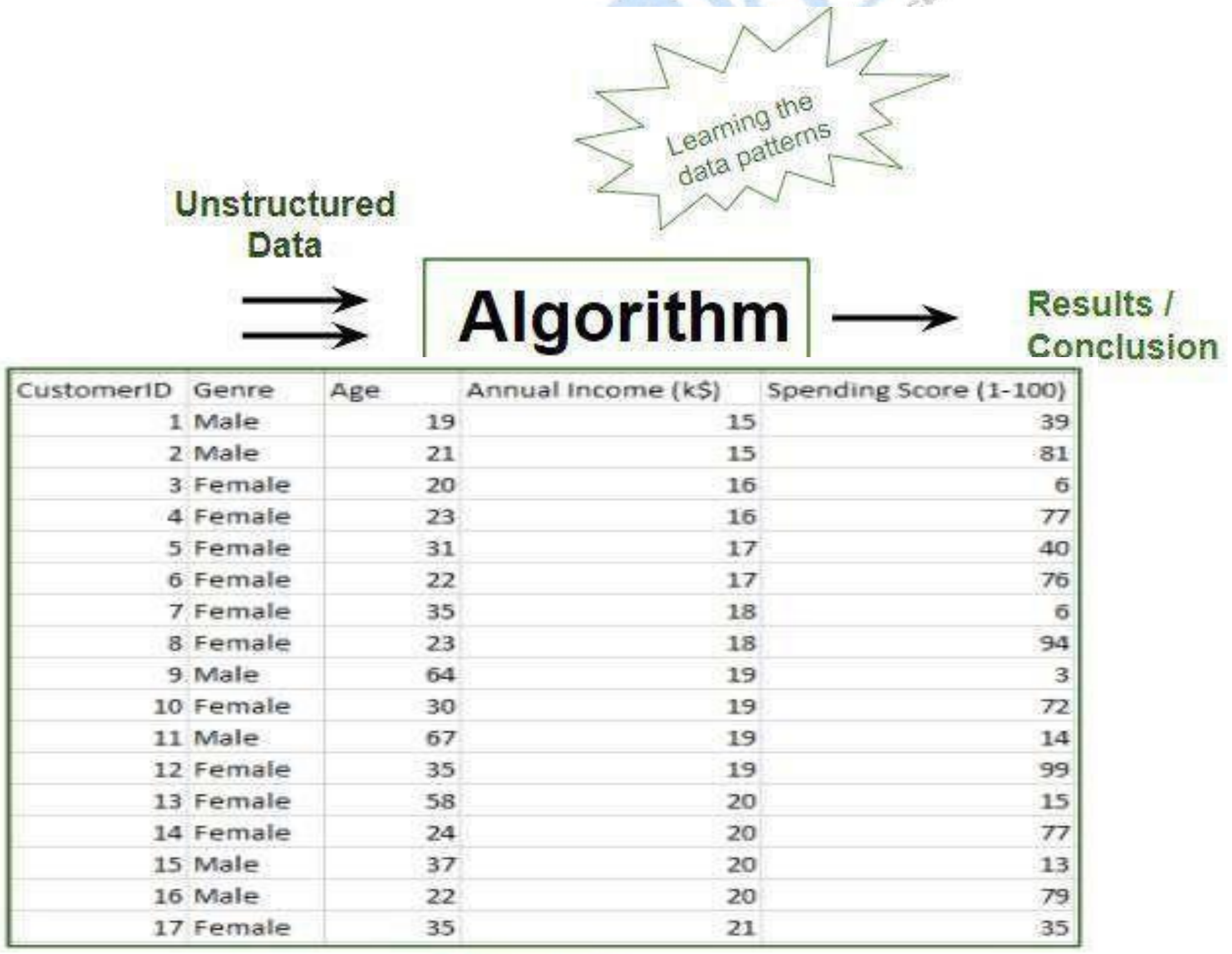
Example of Supervised Learning Algorithms:

- Linear Regression
- Logistic Regression
- Nearest Neighbor
- Gaussian Naive Bayes

- Decision Trees
- Support Vector Machine (SVM)
- Random Forest

Unsupervised Learning:

Or unsupervised machine learning analyzes and clusters unlabeled datasets using machine learning algorithms. These algorithms find hidden patterns and data without any human intervention, i.e., we don‘t give output to our model. The training model has only input parameter values and discovers the groups or patterns on its own. Data-set in Figure A is Mall data that contains information about its clients that subscribe to them. Once subscribed they are provided a membership card and the mall has complete information about the customer and his/her every purchase. Now using this data and unsupervised learning techniques, the mall can easily group clients based on the parameters we are feeding in.

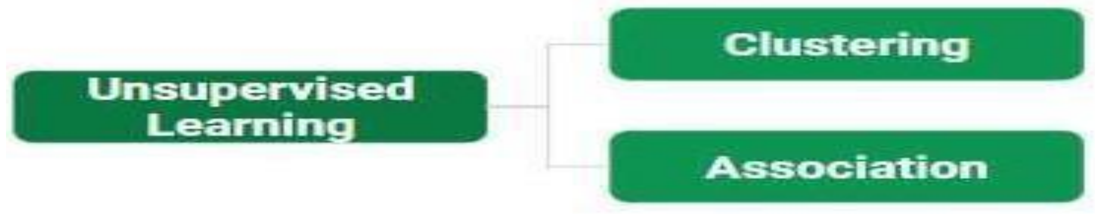


CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40
6	Female	22	17	76
7	Female	35	18	6
8	Female	23	18	94
9	Male	64	19	3
10	Female	30	19	72
11	Male	67	19	14
12	Female	35	19	99
13	Female	58	20	15
14	Female	24	20	77
15	Male	37	20	13
16	Male	22	20	79
17	Female	35	21	35

Figure A

The input to the unsupervised learning models is as follows:

- **Unstructured data**: May contain noisy(meaningless) data, missing values, or unknown data
- **Unlabeled data**: Data only contains a value for input parameters, there is no targeted value(output). It is easy to collect as compared to the labeled one in the Supervised approach.



Types of Unsupervised Learning are as follows:

- **Clustering:** Broadly this technique is applied to group data based on different patterns, such as similarities or differences, our machine model finds. These algorithms are used to process raw, unclassified data objects into groups. For example, in the above figure, we have not given output parameter values, so this technique will be used to group clients based on the in put parameters provided by our data.
- **Association:** This technique is a rule-based ML technique that finds out some very useful relations between parameters of a large data set. This technique is basically used for market basket analysis that helps to better understand the relationship between different products. For e.g. shopping stores use algorithms based on this technique to find out the relationship between the sale of one product w.r.t to another's sales based on customer behavior. Like if a customer buys milk, then he may also buy bread, eggs, or butter. Once trained well, such models can be used to increase their sales by planning different offers.

Some algorithms: *K-Means Clustering*

- *DBSCAN – Density-Based Spatial Clustering of Applications with Noise*
- *BIRCH – Balanced Iterative Reducing and Clustering using Hierarchies*
- *Hierarchical Clustering*
- Supervised and Unsupervised learning are the two techniques of machine learning. But both the techniques are used in different scenarios and with different datasets. Below the explanation of both learning methods along with their difference table is given.

- Supervised learning is a machine learning method in which models are trained using labeled data. In supervised learning, models need to find the mapping function to map the input variable (X) with the output variable (Y).

$$Y = f(X)$$

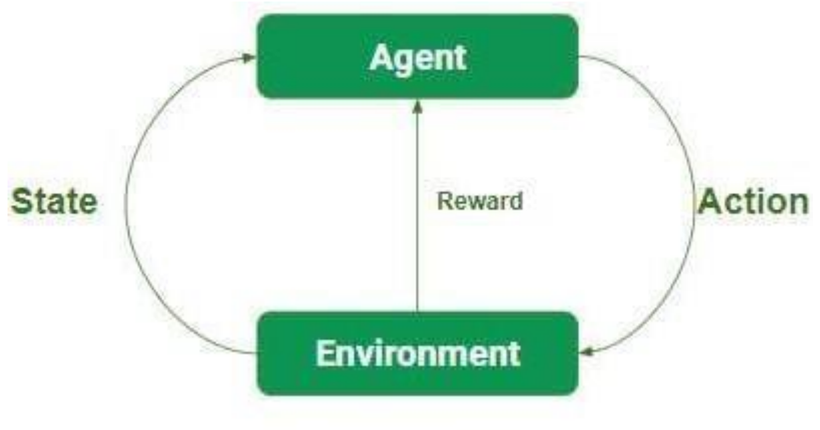
-
- Supervised learning needs supervision to train the model, which is similar to as a student learns things
in the presence of a teacher. Supervised learning can be used for two types of problems: **Classification** and **Regression**.
- **Example:** Suppose we have an image of different types of fruits. The task of our supervised learning model is to identify the fruits and classify them accordingly. So to identify the image in supervised learning, we will give the input data as well as output for that, which means we will train the model by the shape, size, color, and taste of each fruit. Once the training is completed, we will test the model by giving the new set of fruit. The model will identify the fruit and predict the output using a suitable algorithm.
- **Unsupervised Machine Learning:**
- Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data. The goal of unsupervised learning is to find the structure and patterns from the input data. Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own.
- **Learn more** Unsupervised Machine Learning
- Unsupervised learning can be used for two types of problems: **Clustering** and **Association**.
- **Example:** To understand the unsupervised learning, we will use the example given above. So unlike supervised learning, here we will not provide any supervision to the model. We will just provide the input dataset to the model and allow the model to find the patterns from the data. With the help of a suitable algorithm, the model will train itself and divide the fruits into different groups according to the most similar features between them.
- The main differences between Supervised and Unsupervised learning are given below:

Supervised Learning	**Unsupervised Learning**
Supervised learning algorithms are trained using labeled data.	Unsupervised learning algorithms

Supervised learning model takes direct feedback to check if it is predicting correct output or not.	Unsupervised learning model does not take any feedback.
Supervised learning model predicts the output.	Unsupervised learning model finds the hidden patterns in data.
In supervised learning, input data is provided to the model along with the output.	In unsupervised learning, only input data is provided to the model.
The goal of supervised learning is to train the model so that it can predict the output when it is given new data.	The goal of unsupervised learning is to find the hidden patterns and useful insights from the unknown dataset.
Supervised learning needs supervision to train the model.	Unsupervised learning does not need any supervision to train the model.
Supervised learning can be categorized in **Classification** and **Regression** problems.	Unsupervised Learning can be classified in **Clustering** and **Associations** problems.
Supervised learning can be used for those cases where we know the input as well as corresponding outputs.	Unsupervised learning can be used for those cases where we have only input data and no corresponding output data.
Supervised learning model produces an accurate result.	Unsupervised learning model may give less accurate result as compared to supervised learning.
Supervised learning is not close to true Artificial intelligence as in this, we first train the model for each data, and then only it can predict the correct output.	Unsupervised learning is more close to the true Artificial Intelligence as it learns similarly as a child learns daily routine things by his experiences.
It includes various algorithms such as Linear Regression, Logistic Regression, Support Vector Machine, Multi-class Classification, Decision tree, Bayesian Logic, etc.	It includes various algorithms such as Clustering, KNN, and Apriori algorithm.

Semi-supervised Learning:

As the name suggests, its working lies between Supervised and Unsupervised techniques. We use these techniques when we are dealing with data that is a little bit labeled and the rest large portion of it is unlabeled. We can use the unsupervised techniques to predict labels and then feed these labels to supervised techniques. This technique is mostly applicable in the case of image data sets where usually all images are not labeled.



REINFORCEMENT

Reinforcement Learning:

- In this technique, the model keeps on increasing its performance using Reward Feedback to learn the behavior or pattern. These algorithms are specific to a particular problem e.g. Google Self Driving car, AlphaGo where a bot competes with humans and even itself to get better and better performers in Go
- Game. Each time we feed in data, they learn and add the data to their knowledge which is training data. So, the more it learns the better it gets trained and hence experienced.
- Agents observe input.
- An agent performs an action by making some decisions.
- After its performance, an agent receives a reward and accordingly reinforces and the model stores in state-action pair of information.
- Temporal Difference (TD)
- Q-Learning
- Deep Adversarial Networks

Q.5. What are Linear Models of Classification?

Ans: Linear models of classification are a type of machine learning model used to classify data points into different categories based on a linear decision boundary. These models work by computing a weighted sum of the input features and applying a decision rule to determine the class label.

Core Idea:

A linear model assumes that the relationship between the input features and the output class is linear. In binary classification, the model tries to find a hyperplane (a line in 2D, a plane in 3D, etc.) that separates the classes.

Mathematical Representation:

For a binary classification problem, a linear model predicts the label (y) based on the input vector (mathbf{x} = [x_1, x_2, ..., x_n]) using:

[

y = text{sign}(mathbf{w}^T mathbf{x} + b)

]

Where:

- (mathbf{w}) = weight vector
- (b) = bias (intercept)
- (text{sign}) = sign function, which returns +1 or -1 (or 0, but typically binary classes are +1 and -1)

Common Examples of Linear Classifiers:

1. Logistic Regression (uses the sigmoid function to output probabilities)
2. Support Vector Machines (SVM) (linear kernel)
3. Perceptron (basic neural network model)
4. Linear Discriminant Analysis (LDA) (when assumptions hold)

Characteristics:

- Fast and efficient, especially with large datasets.
- Easy to interpret (feature weights indicate importance).
- Perform well when data is linearly separable.
- Can struggle with complex, non-linear decision boundaries.

Q.6. Explain Naïve Bayes Classification ?

Ans: Naïve Bayes Classification is a simple yet powerful probabilistic machine learning algorithm used for classification tasks. It is based on Bayes' Theorem and makes a key assumption: features are conditionally independent given the class — which is why it's called "naïve."

Core Idea:

Naïve Bayes uses probability to predict the most likely class for a given input. It calculates the probability of each class given the input features and chooses the class with the highest probability.

Bayes' Theorem:

[

P(C|X) = frac{P(X|C) cdot P(C)}{P(X)}

]

Where:

- (P(C|X)): Posterior probability of class (C) given features (X)
- (P(X|C)): Likelihood of features (X) given class (C)
- (P(C)): Prior probability of class (C)
- (P(X)): Marginal probability of fatures (X) (acts as a normalizing constant)

The "Naïve" Assumption:

Assumes all features are independent of each other given the class label:

[

P(X|C) = P(x_1|C) cdot P(x_2|C) cdot dots cdot P(x_n|C)

]

This simplifies computation significantly, even though it might not hold true in reality.

Types of Naïve Bayes:

1. Gaussian Naïve Bayes – For continuous data (assumes features follow a normal distribution)
2. Multinomial Naïve Bayes – For count data (e.g. word counts in documents)
3. Bernoulli Naïve Bayes – For binary/boolean features (e.g. word presence/absence)

Advantages:

- Very fast and scalable

- Works surprisingly well even with the "naïve" assumption

- Performs well with text classification (spam filtering, sentiment analysis, etc.)

Disadvantages:

- Assumes feature independence (rarely true in real-world data)
- Can perform poorly when features are highly correlated

Q.6. What is Logistic Regression?

Logistic Regression is a **supervised learning algorithm** used for **binary classification** — that is, it predicts whether something belongs to **class 0 or class 1** (e.g., spam or not spam, pass or fail, etc.).

Despite the name, **it's a classification algorithm**, not a regression one.

Core Idea:

Logistic regression models the **probability** that a given input xmathbf{x}x belongs to a certain class using the **logistic (sigmoid) function**.

P(y=1|x)=σ(wTx+b)P(y = 1 | mathbf{x}) = sigma(mathbf{w}^T mathbf{x} + b)P(y=1|x)=σ(wTx+b)

Where:

- σ(z)=11+e−zsigma(z) = frac{1}{1 + e^{-z}}σ(z)=1+e−z1 is the **sigmoid function**
- wmathbf{w}w = weights (learned during training)
- bbb = bias or intercept
- Output is a probability between 0 and 1

Decision Rule:

- If P(y=1|x)≥0.5P(y = 1 | mathbf{x}) geq 0.5P(y=1|x)≥0.5, predict class **1**
- If P(y=1|x)<0.5P(y = 1 | mathbf{x}) < 0.5P(y=1|x)<0.5, predict class **0**

Q.7. Why Use the Sigmoid Function?

The sigmoid squashes any real-valued number into the range (0, 1), making it perfect to model **probabilities**.

Cost Function:

Logistic regression uses **log loss** (or **binary cross-entropy**) to evaluate its predictions:

Loss=−[ylog(p)+(1−y)log(1−p)]text{Loss} = -[y log(p) + (1 - y) log(1 - p)]Loss=−[ylog(p)+(1−y)log(1−p)]

Where:

- yyy = actual label (0 or 1)
- ppp = predicted probability

Advantages:

- Simple and easy to interpret
- Outputs probabilities, not just labels
- Works well when the classes are linearly separable
- Fast training and prediction

Limitations:

- Assumes linear relationship between input features and the log-odds
- Not suitable for complex or non-linear relationships (unless you transform features)

Q.8. What are Probabilistic Models?

Ans: Probabilistic models are a type of machine learning model that use the principles of probability theory to represent and reason about uncertainty in data. Instead of giving just a "yes/no" answer, they give you probabilities for different outcomes.

Core Idea:

Rather than just predicting a class or value directly, a probabilistic model estimates:

[

P(Y | X)

]

Which means: ―What‘s the probability of output (Y), given input (X)?‖

This approach helps when you want to:

- Understand the confidence of predictions
- Handle uncertainty and noisy data

- Combine models or decisions (as in Bayesian networks)

Examples of Probabilistic Models:

1. Naïve Bayes Classifier – Uses Bayes' theorem to predict class probabilities.
2. Logistic Regression – Predicts the probability of a binary outcome using the sigmoid function.
3. Bayesian Networks – Graphical models that represent variables and their dependencies using probability.
4. Hidden Markov Models (HMMs) – Used for time series data, modeling hidden states and transitions.
5. Gaussian Mixture Models (GMMs) – Models data as a mixture of several Gaussian distributions.

Advantages:

- Models uncertainty naturally
- Can provide confidence scores for predictions
- Good for incomplete or noisy data
- Can be combined with Bayesian inference

Disadvantages:

- Often more computationally expensive
- May require strong assumptions (e.g., feature independence)
- Harder to interpret than some rule-based or decision-tree models

Unit – IV

1. What is Unsupervised Learning?

Ans: Unsupervised learning is a type of machine learning where the algorithm is provided with **unlabeled data**, and it tries to **discover patterns**, **structures**, or **relationships** in the data **without any explicit supervision**. Unlike supervised learning (where we know the correct answers), unsupervised learning explores the data and organizes it in a meaningful way without prior labels.

Why Use Unsupervised Learning?

Many real-world datasets come **without labels**. For example:

- Clustering customers based on their behavior (no predefined groups)

- Detecting unusual patterns in server logs (no ―anomaly‖ label)
- Discovering topics in a collection of documents

In such cases, **unsupervised learning** becomes essential because it helps in:

- Understanding the **structure** of the data
- **Preprocessing** or **compressing** data (e.g., dimensionality reduction)
- **Generating features** or insights for other models
- **Exploratory data analysis** (EDA)

Main Techniques in Unsupervised Learning

There are two primary categories of tasks in unsupervised learning:

1. Clustering

Group data points into clusters based on similarity.

2. Association Rule Mining

Find interesting relationships, correlations, or patterns among features in the data.

We‘ll dive into these subtopics in more detail in future sections — but let‘s explore the foundation first.

Key Differences from Supervised Learning

Feature	**Supervised Learning**	**Unsupervised Learning**
Labeled Data	Required	Not required
Goal	Predict output (target variable)	Discover patterns or structure
Examples	Classification, Regression	Clustering, Association, Dim. Red.
Output	Labels or continuous values	Clusters, rules, embeddings

Example: Clustering Customers

Imagine you work for an e-commerce company. You have a lot of data on customers — age, spending habits, types of products they buy — but you don‘t know which customers are similar.

With **unsupervised learning**, you can group similar customers into **clusters**, which can help:

- Target marketing strategies
- Personalize product recommendations
- Identify VIPs or at-risk customers

No prior labels are used — just patterns that emerge from the data itself.

Common Algorithms in Unsupervised Learning

Here are some of the most widely used algorithms in this space:

1. K-Means Clustering

- Divides data into *K* clusters based on distance (typically Euclidean).
- Each point is assigned to the nearest cluster center.
- Simple, fast, but assumes spherical clusters.

2. Hierarchical Clustering

- Builds a hierarchy of clusters.
- Can be **agglomerative** (bottom-up) or **divisive** (top-down).
- Results can be visualized using a **dendrogram**.

3. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- Groups together points that are close in terms of density.
- Can identify **outliers** as noise.
- Doesn't require specifying the number of clusters in advance.

4. Gaussian Mixture Models (GMMs)

- A **probabilistic** clustering method.
- Assumes data is generated from a mixture of several Gaussian distributions.
- Can model **overlapping clusters** better than K-Means.

5. PCA (Principal Component Analysis)

- A **dimensionality reduction** method.
- Transforms data into a new coordinate system that captures maximum variance.
- Often used for **visualizing high-dimensional data**.

6. Autoencoders

- Neural network-based models used for unsupervised **feature learning**.
- Try to compress and then reconstruct the input data.
- Useful for dimensionality reduction and anomaly detection.

Applications of Unsupervised Learning

Unsupervised learning is used across industries. Some key applications:

Domain	**Use Case**
E-commerce	Customer segmentation
Finance	Fraud detection, portfolio clustering

Healthcare	Disease subtype discovery
Marketing	Market basket analysis, customer journey
NLP	Topic modeling, word embeddings
Cybersecurity	Anomaly detection in network traffic

Challenges in Unsupervised Learning

1. **No Ground Truth:**
   No labeled data makes it hard to evaluate accuracy. Validation requires domain knowledge or heuristics.
2. **Choosing the Right Algorithm:**
   There‘s no one-size-fits-all. Clustering might work great on one dataset and fail on another.
3. **Interpreting Results:**
   Clusters or features extracted may not always be easy to explain.
4. **Scalability:**
   Some unsupervised algorithms can be computationally intensive (e.g., hierarchical clustering).
5. **Parameter Tuning:**
   Many algorithms require choosing parameters like number of clusters (K in K-Means), epsilon (in DBSCAN), or number of components (in PCA).

Intuition Behind Unsupervised Learning

Think of unsupervised learning as a **curious explorer**.

It walks into a completely unfamiliar city (dataset), with no map (labels), and tries to:

- **Group neighborhoods** based on looks and vibe (clustering),
- **Find connections** between places (associations),
- **Compress the map** to keep only the major roads (dimensionality reduction).

It's about **making sense of the unknown**.

2. What is Clustering in Unsupervised Learning?

Ans: Clustering is one of the most fundamental tasks in **unsupervised learning**. The core idea is simple but powerful: **group similar data points together** without using any labels. Each group formed is known as a **cluster**, and items within a cluster are more similar to each other than to those in other clusters.

It‘s widely used in data exploration, pattern recognition, customer segmentation, document organization, and even anomaly detection.

Why Use Clustering?

In many real-world situations, we have data but no idea how it‘s structured. Clustering helps us:

- Discover **hidden structures** or **groupings**.
- Understand data **distribution** and **density**.
- Perform **preprocessing** or **feature engineering** for other models.
- Summarize large datasets by identifying core patterns.
- Think of clustering as the data scientist‘s way of **letting the data speak for itself**.

Objective of Clustering

- Given a dataset X={x1,x2,...,xn}X = {x_1, x_2, ..., x_n}X={x1,x2,...,xn}, the goal is to divide it into KKK clusters C1,C2,...,CKC_1, C_2, ..., C_KC1,C2,...,CK such that:
- ∀i,j∈Ckforall i, j in C_k∀i,j∈Ck: similarity between xix_ixi and xjx_jxj is **high** (within cluster similarity)
- ∀xi∈Ck,xj∈Cl,k≠lforall x_i in C_k, x_j in C_l, k ne l∀xi∈Ck,xj∈Cl,k □=l: similarity is **low** (between cluster dissimilarity)
- In practice, similarity is often measured using distance metrics like **Euclidean**, **Manhattan**, or **cosine** distance.

Common Clustering Algorithms

Let‘s go through some of the most widely used clustering algorithms:

1. K-Means Clustering

- Partitions data into KKK clusters.
- Each cluster has a **centroid**, and each point is assigned to the nearest centroid.
- Iteratively updates centroids and point assignments.

Pros:

- Fast and scalable.
- Works well on spherical, equally sized clusters.

Cons:

- Requires choosing KKK in advance.

- Sensitive to outliers and initial centroids.
- Assumes clusters are isotropic and convex.

2. Hierarchical Clustering

- Builds a tree (dendrogram) of clusters.
- Two main types:
  - **Agglomerative:** Start with individual points and merge them.
  - **Divisive:** Start with all points in one cluster and split them.

Pros:

- Doesn‘t require choosing KKK in advance.
- Produces a full hierarchy of clusters.

Cons:

- Computationally expensive for large datasets.
- Not good with high-dimensional data.

3. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- Forms clusters based on density of data points.
- Two parameters: εvarepsilonε (radius) and MinPts (min points to form dense region).

Pros:

- Can find arbitrarily shaped clusters.
- Detects outliers as noise.
- Doesn‘t need KKK.

Cons:

- Struggles with varying densities.
- Needs careful tuning of parameters.

4. Gaussian Mixture Models (GMM)

- Assumes data is generated from a mixture of several **Gaussian distributions**.
- Uses the **Expectation-Maximization (EM)** algorithm to estimate parameters.

Pros:

- Provides **probabilistic** cluster memberships.
- Handles overlapping clusters better than K-Means.

Cons:

- More computationally intensive.

- Can converge to local optima.

Evaluation Metrics for Clustering

Clustering is **unsupervised**, so evaluation is not straightforward. However, we use:

1. Internal Evaluation (no ground truth):

- **Silhouette Score:** Combines cohesion and separation.
- **Dunn Index:** Ratio of smallest distance between clusters to largest intra-cluster distance.
- **Inertia (for K-Means):** Sum of squared distances to cluster centers.

2. External Evaluation (with ground truth):

- **Adjusted Rand Index (ARI)**
- **Normalized Mutual Information (NMI)**
- **Purity**

How to Choose the Number of Clusters?

This is a classic problem in clustering. Some strategies include:

- **Elbow Method**: Plot within-cluster sum of squares vs. KKK, and look for the ―elbow‖ point.
- **Silhouette Score**: Choose the KKK that gives the highest average silhouette score.
- **Gap Statistic**: Compares total within intra-cluster variation with expected under null reference.

Applications in Depth

Let‗s take a closer look at one practical application:

Customer Segmentation

In retail or e-commerce, companies often don‗t know their customers in advance. Clustering can group them based on:

- Purchase history
- Browsing behavior
- Frequency of interaction
- Recency of purchases

Each cluster might correspond to:

- Bargain shoppers
- Loyal frequent buyers
- Infrequent window-shoppers

Marketers can then **personalize** campaigns for each group.

Challenges in Clustering

- **Determining Number of Clusters:** Often unknown, and methods to estimate KKK can be unreliable.
- **High-Dimensional Data:** Clustering becomes hard due to the **curse of dimensionality**.
- **Interpreting Clusters:** What does a given cluster mean? Requires domain expertise.
- **Scalability:** Some algorithms (like hierarchical clustering) are not suitable for large datasets.
- **Sensitivity to Initialization:** K-Means and GMMs can yield different results depending on starting points.

Q.4 Explain in Directed Graphical Models

Ans: In Graphical Model Terms:

In a **Directed Graphical Model**, variables are represented as **nodes**, and **edges** (arrows) show **conditional dependencies** (who influences whom).

Factor Analysis as a DGM:

In **Factor Analysis**, you have:

- **Latent variables (Factors)**: F1,F2,...,FkF_1, F_2, ..., F_kF1,F2,...,Fk → **not observed**
- **Observed variables**: X1,X2,...,XpX_1, X_2, ..., X_pX1,X2,...,Xp

The **structure** of the graphical model is:

```
F1 ─┬→ X1
    ├→ X2
    ├→ X3
    ⋮
    └→ Xp
F2 ─┬→ X1
    ├→ X2
    ⋮
    └→ Xp
...
Fk ─┬→ X1
    ⋮
```

└→ Xp

This is a **bipartite graph**:

- Arrows go **from latent factors (F) → to observed variables (X)**
- Each **XiX_iXi** is conditionally dependent on all factors
- All observed variables are **conditionally independent given the factors**

Interpretation:

- The **latent variables (F)** are the **common causes** of the observed variables.
- The **errors (ϵiepsilon_iϵi)** are **independent** across XiX_iXi's, and not shown in the DGM unless explicitly modeled.

In Probabilistic Terms:

P(F,X)=P(F)·P(X|F)P(F, X) = P(F) cdot P(X mid F)P(F,X)=P(F)·P(X|F)

- P(F)P(F)P(F): Prior over latent factors (often standard normal)
- P(X|F)P(X mid F)P(X|F): Likelihood — observed variables are modeled as linear combinations of FFF plus noise

Terminology

1. Artificial Intelligence (AI)

The field of creating intelligent machines capable of mimicking human reasoning, learning, and problem-solving to perform tasks autonomously.

2. Intelligent Agent

An entity that perceives its environment and takes actions to maximize goal achievement, like robots or software agents.

3. Machine Learning (ML)

A subfield of AI where algorithms learn patterns from data without being explicitly programmed to perform specific tasks.

4. Deep Learning

A subset of ML using multi-layered neural networks to model complex patterns in data, especially for images, speech, and natural language.

5. State Space Search

A method of representing problems as a sequence of states and actions to find a solution from an initial to a goal state.

6. Production System

A system consisting of rules, a database, and a control strategy to infer conclusions from known facts, often used in expert systems.

7. Breadth-First Search (BFS)

An uninformed search strategy that explores all nodes at the current depth level before moving to the next.

8. Depth-First Search (DFS)

An uninformed strategy that explores a path as far as possible before backtracking and exploring alternate paths.

9. Heuristic Search

An informed search technique using domain-specific knowledge (heuristics) to find more efficient solutions than uninformed methods.

10. A Algorithm

A best-first search that uses cost and heuristic estimates to find the shortest path to a goal efficiently.

11. Hill Climbing

A local search algorithm that continually moves in the direction of increasing value or fitness to find local optima.

12. Constraint Satisfaction Problem (CSP)

A problem where a set of variables must satisfy specific constraints, such as Sudoku or scheduling problems.

13. Propositional Logic

A logic system using simple, declarative statements (true/false) to represent knowledge in a formal and structured way.

14. Predicate Logic

An extension of propositional logic allowing the use of variables and quantifiers, making it more expressive for knowledge representation.

15. Naïve Bayes Classifier

A probabilistic classifier based on Bayes‘ theorem, assuming features are conditionally independent given the class label.

16. Logistic Regression

A statistical model for binary classification using a logistic function to model the probability of a class label.

17. Neural Network

A machine learning model inspired by the human brain, consisting of layers of interconnected nodes (neurons) that learn complex patterns.

18. Back propagation
An algorithm for training neural networks by propagating the error backward and adjusting weights using gradient descent.

19. K-Means Clustering
An unsupervised learning algorithm that partitions data into k clusters based on similarity, using iterative refinement of cluster centers.

20. EM Algorithm
An iterative method for finding maximum likelihood estimates in probabilistic models with latent variables, like mixtures of Gaussians.

21. Factor Analysis
A statistical method that explains observed data in terms of a few underlying latent variables or factors.

22. PCA (Principal Component Analysis)
A dimensionality reduction technique that transforms data into uncorrelated components ordered by variance, capturing the most important structure.

23. Bayesian Network
A directed graphical model that represents conditional dependencies among variables using nodes and edges with probability distributions.

24. Markov Model
A stochastic model where the next state depends only on the current state, not on the sequence of previous states.

25. Hidden Markov Model (HMM)
A probabilistic model with hidden states that emit observable events, widely used in speech and sequence analysis.