
Unit 4

# Planning

# What is Planning?


- The task of coming up with a sequence of actions that will achieve a goal is called planning.
- Planning Environments
  1. Classical Planning Environments
     - Fully observable, deterministic, static and finite, discrete.
  2. Non classical Planning Environments
     - Partially observable, stochastic with differen algorithms and agent designs. t

# Planning (Contd..)


- Difficulty in problem-solving agent
  - Performs irrelevant actions
    - Have to explore all the states
    - Ex: buy book with 10-digit ISBN $10^{10}$ actions
  - Finding heuristics function
    - Problem-solving agent lacks autonomy because it depends on human to supply heuristic function for each new problem.
  - No problem decomposition
- **All these problems are overcome by planning agent by representing the goal as conjunction of subgoals.**

# Planning Problem

The planning problem is actually the question how to go to next state or the goal state from the current state. It involves two things 'how' and 'when'.

- The planning problem is defined with:

1. Domain model
2. Initial state
3. Goal state (next state)

**The domain model** defines the actions along with the objects. It is necessary to specify the operators too that actually describe the action. Along with this, information about actions and state constraints while acting should also be given. This entirely formulates the domain model.

**The initial state** is the state where any action is yet to take place (the stage when the exam schedule is put up!).

**The final state** or the goal state is the state which the plan is intended to achieve.

# Planning Agents


- **problem-solving agents** are able to plan ahead - to consider the consequences of *sequences* of actions - before acting.
- **knowledge- based agents** can select actions based on explicit, logical representations of the current state and the effects of actions.
  - This allows the agent to succeed in complex, inaccessible environments that are too difficult for a problem-solving agent

– **Problem Solving Agents + Knowledge-based Agents**

**= Planning Agents**

# Simple Planning Agent

- Planning can be viewed as a type of problem solving in which the agent uses beliefs about actions and their consequences to search for a solution over the more abstract space of plans, rather than over the space of situations.

Algorithm of a simple planning agent:

1. Generate a goal to achieve
2. Construct a plan to achieve goal from current state
3. Execute plan until finished
4. Begin again with new goal

- The agent first generates a goal to achieve, and then constructs a plan to achieve it from the current state. Once it has a plan, it keeps executing it until the plan is finished, then begins again with a new goal.

# Languages for Planning Problems

- STRIPS
  - Stanford Research Institute Problem Solver
  - Historically important
- ADL
  - Action Description Languages
- PDDL
  - Planning Domain Definition Language

# Planning Languages


- Languages must represent..
  - States
  - Goals
  - Actions
- Languages must be
  - Expressive for ease of representation
  - Flexible for manipulation by algorithms

# Example of a robot

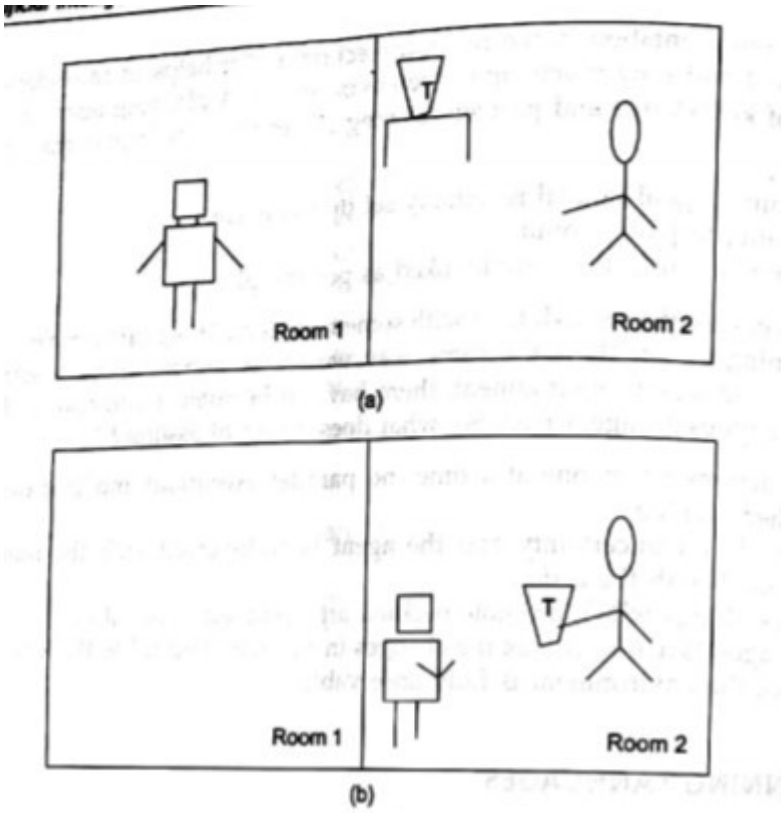



Figure 9.2 (a) Initial stage, (b) Goal state.

**State Representation:**

1. in(robot,room1)^in(tea,room2)^in(guest,room2)
2. in(robot,room1)^in(tea,room1)^in(guest,room2)
3. in(robot,room1)^in(tea,room2)^in(guest,room1) and so on

**Action Representation:**

Action1: Move-to-room2
Precondition: in(robot,room1)
Post-condition: add-List: in(robot,room2)
delete-list: in(robot,room1)

# Block World

- There are ‘N’ number of Blocks resting on table with specified sequence.
- Goal is to arrange in desired sequence.
- Available moves
  - Put block on table
  - Put a block on another block top
- State is represented using sequence of blocks in current pos.

# STRIPS

- STRIPS stands for "STanford Research Institute Problem Solver," was the planner used in Shakey, one of the first robots built using AI technology ,which is an action-centric representation ,for each action , specifies the effect of an action.

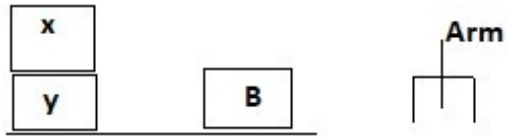
A **STRIPS planning problem** specifies:
1) an initial state *S*
2) a goal *G*
3) a set of STRIPS actions

The **STRIPS representation** for an action consists of
- the **precondition**, which is a set of assignments of values to features that must be true for the action to occur, and
- the **effect**, which is a set of resulting assignments of values to those primitive features that change as the result of the action.

# Block World Problem

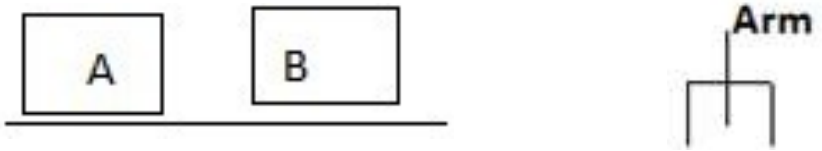
- Action List:

| SNo | Action | Precondition | Effect |
|---|---|---|---|
| 1 | Pickup(x) | Arm Empty<br>On(x, Table)<br>Clear(x) | Holding(x) |
| 2 | Putdown (x) | Holding(x) | Arm Empty<br>On(x, Table)<br>Clear(x) |
| 3 | Stack(x,y) | Holding(x)<br>Clear(y) | On(x,y)<br>Clear(x)<br>Arm Empty |
| 4 | Unstack(x,y) | On(x,y)<br>Clear(x)<br>Arm Empty | Holding(x)<br>Clear(y) |

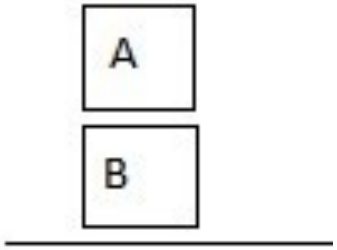


# Block World Problem

- Start State:




- Goal State:

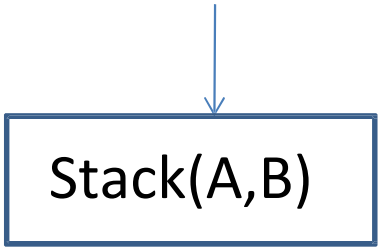


# Block World Problem

- Start State:
  - On(A, table)
  - On(B, table)
- Goal State:
  - On(A,B)

**Solution**:

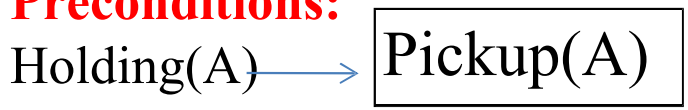
On(A,B)




**Preconditions:**

Holding(A)

Clear(B)

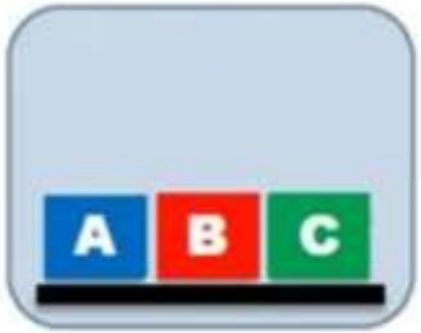



Post conditions:

Arm Empty

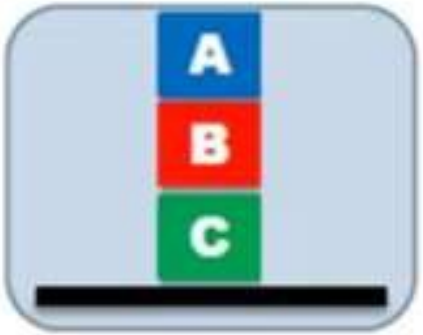
On(A, B)

Clear(A)



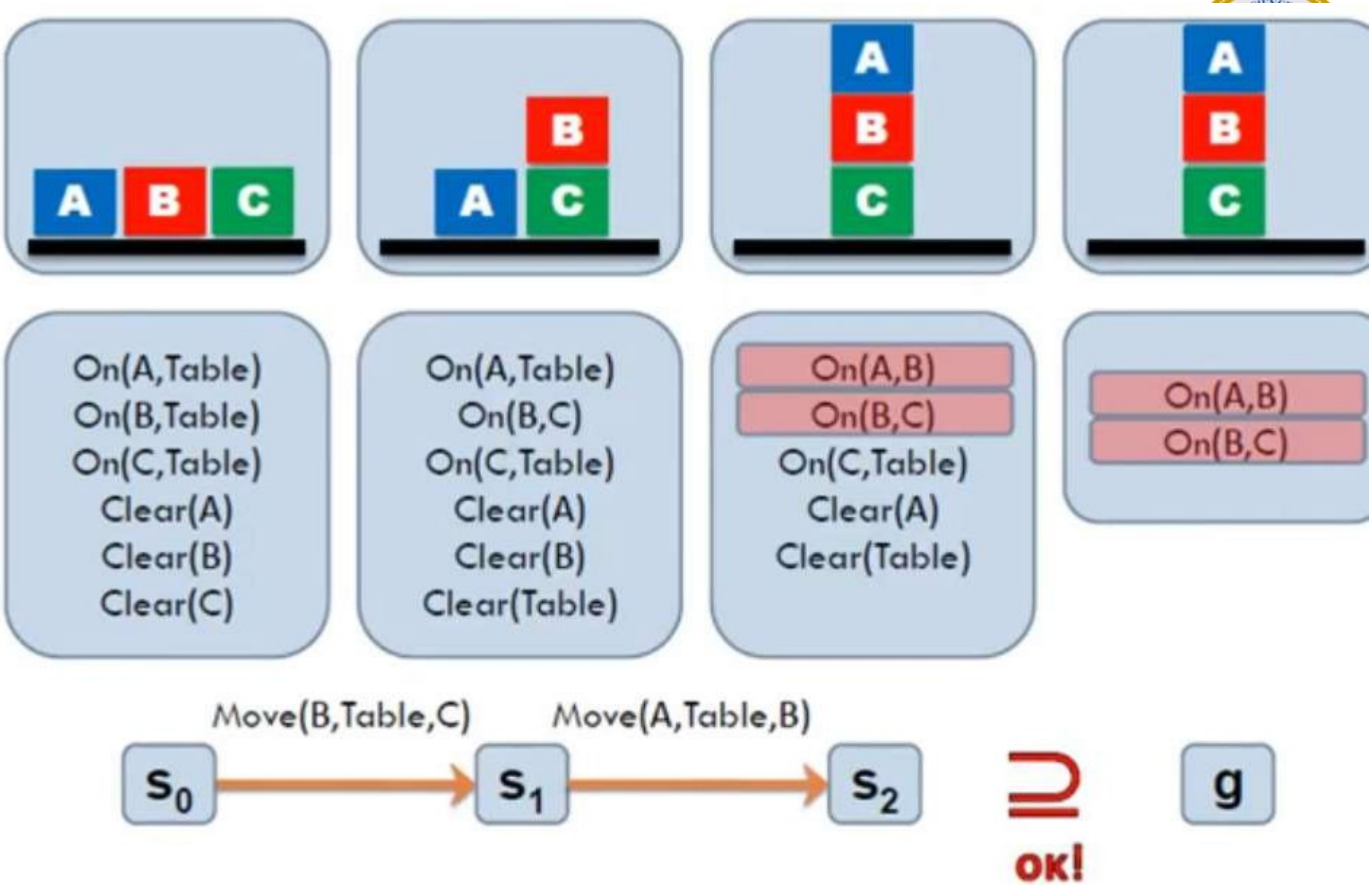

On(A,Table)
On(B,Table)
On(C,Table)
Clear(A)
Clear(B)
Clear(C)

$S_0$



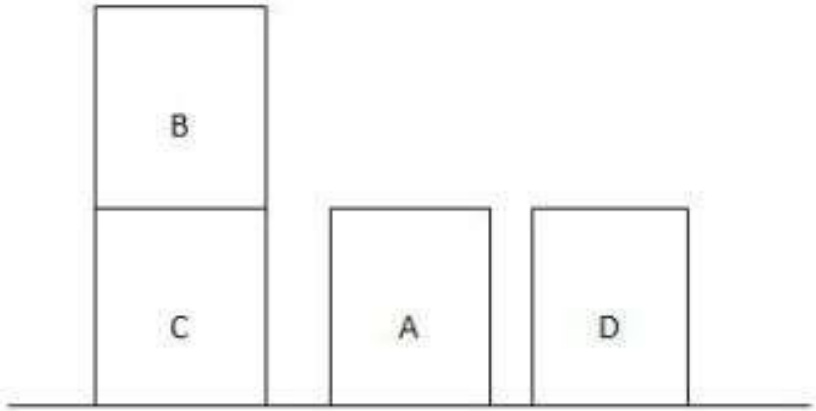

On(A,B)
On(B,C)

g


A B C
B
A C
A
B
C
A
B
C
On(A,Table)
On(B,Table)
On(C,Table)
Clear(A)
Clear(B)
Clear(C)
On(A,Table)
On(B,C)
On(C,Table)
Clear(A)
Clear(B)
Clear(Table)
On(A,B)
On(B,C)
On(C,Table)
Clear(A)
Clear(Table)
On(A,B)
On(B,C)
Move(B,Table,C)
Move(A,Table,B)
$s_0$
$s_1$
$s_2$
⊇
OK!
g

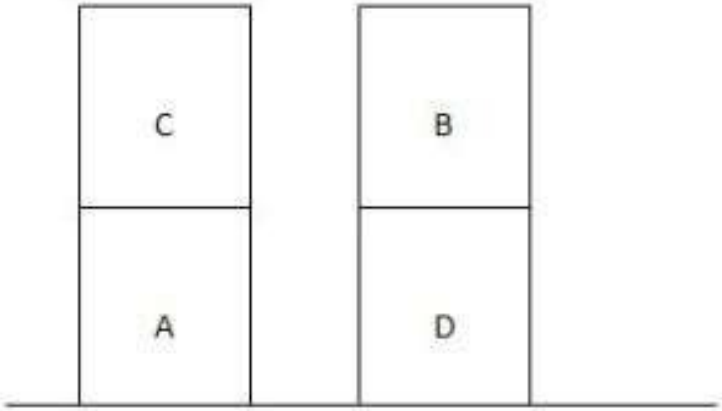
# GOAL STACK PLANNING

- **Basic Idea** to handle interactive compound goals uses goal stacks, Here the stack contains
  :
  - goals,
  - operators -- ADD, DELETE and PREREQUISITE lists
  - a database maintaining the current situation for each operator used.




**Initial State**

ON(B,C) ∧ ONTABLE(C) ∧ ONTABLE(A) ∧ ONTABLE(D)
CLEAR(B) ∧ CLEAR(A) ∧ CLEAR(D)



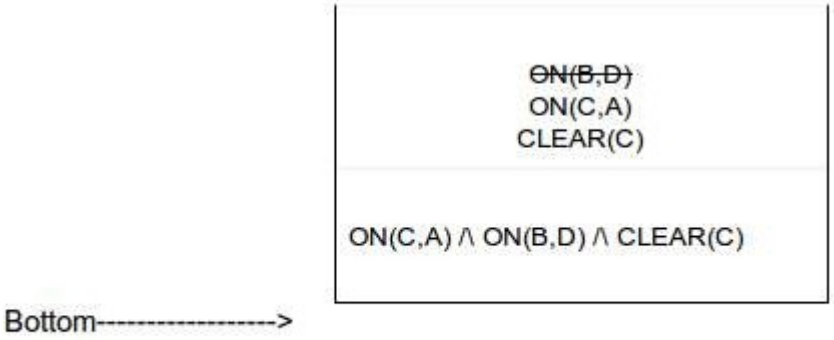

**Goal State**

ON(C,A) ∧ ON(B,D) ∧ ONTABLE(A) ∧ ONTABLE(D)
∧ CLEAR(C) ∧ CLEAR(B)

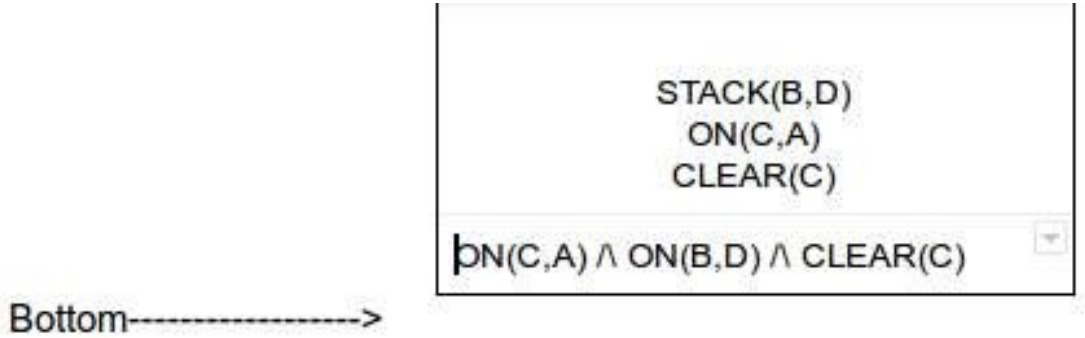
- First step is to push the goal into the stack.


- Next push the individual predicates of the goal into the stack.

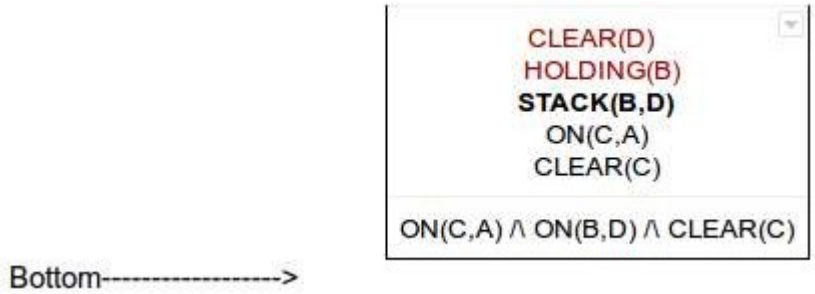
- Now pop an element out from the stack.



- ON(B,D) which is a predicate and it is not true in our current world. Push the relevant action which could achieve the subgoal ON(B,D) in to the stack.



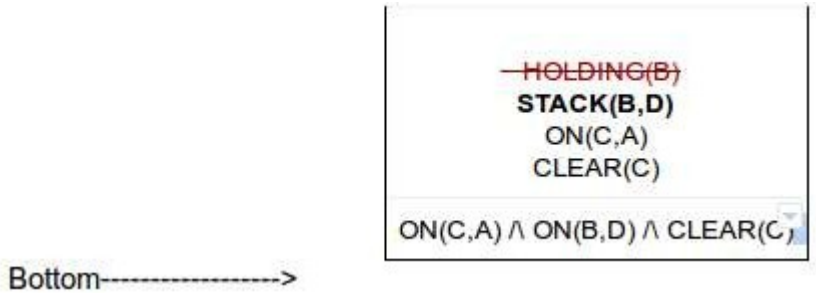
push the precondition of the action Stack(B,D) into the stack.



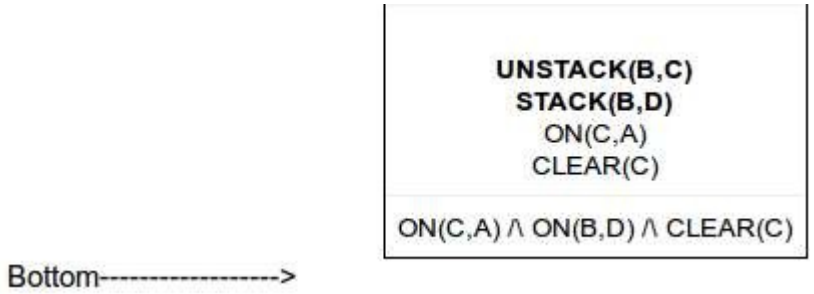

POP an element out from the stack

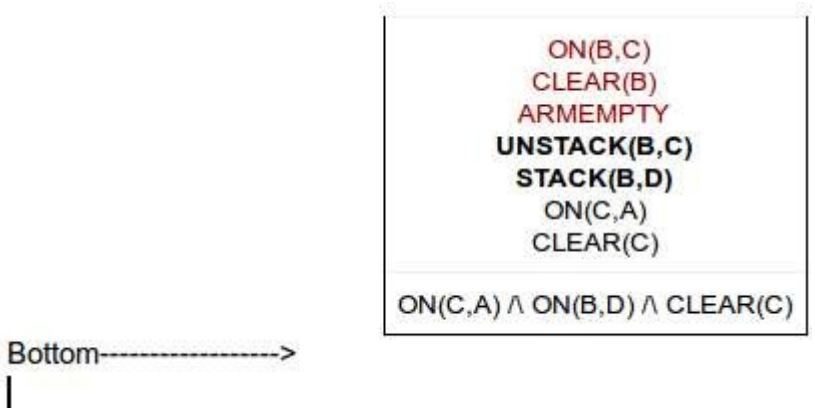
CLEAR(D) is true in the current world model

pop the stack

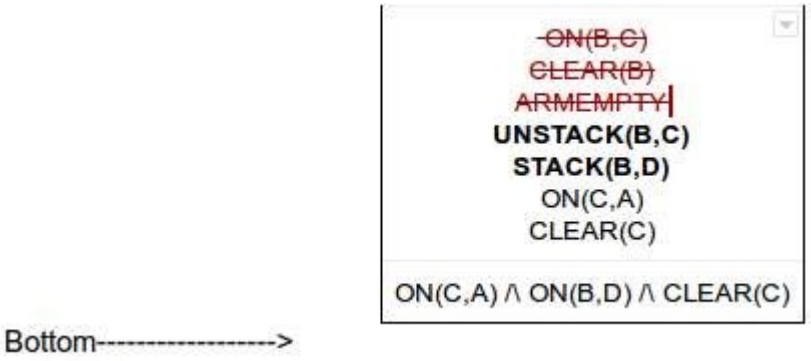


Lets push the action UNSTACK(B,C) into the stack.

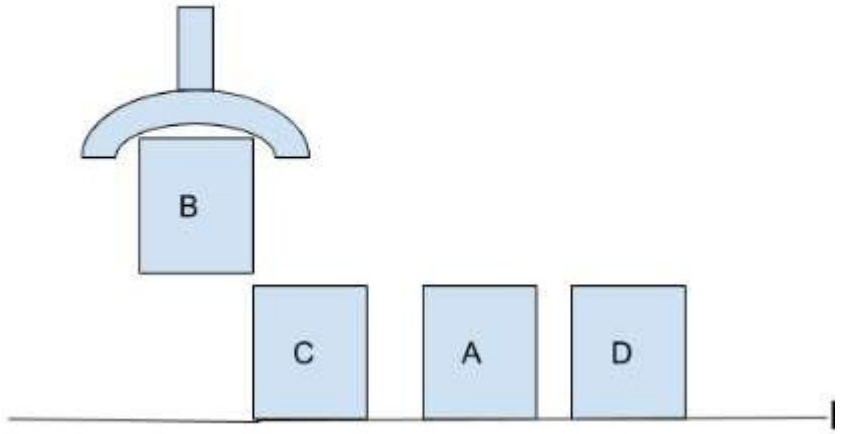


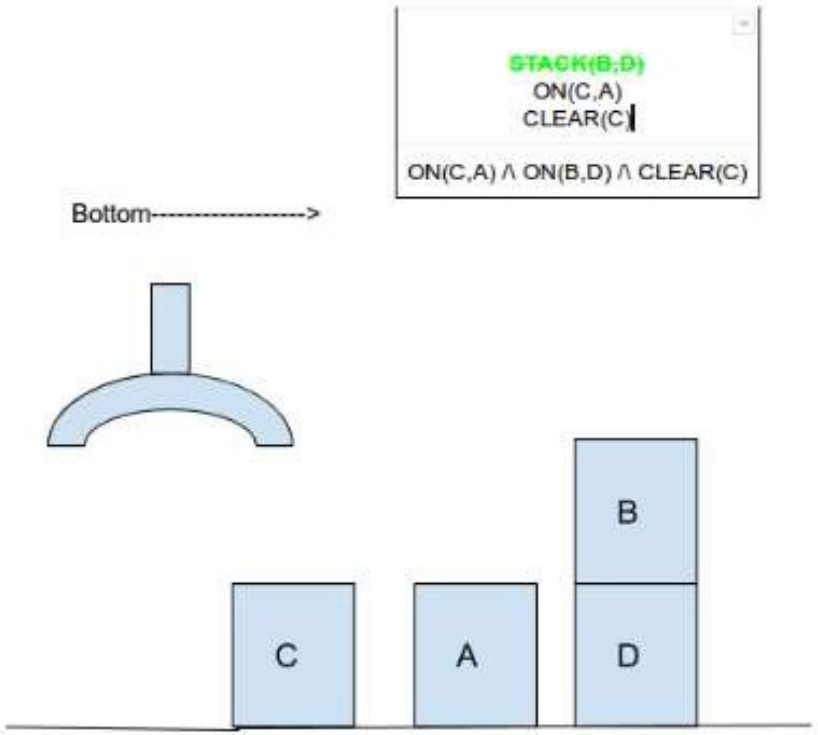
POP the stack.



Plan= { UNSTACK(B,C) }

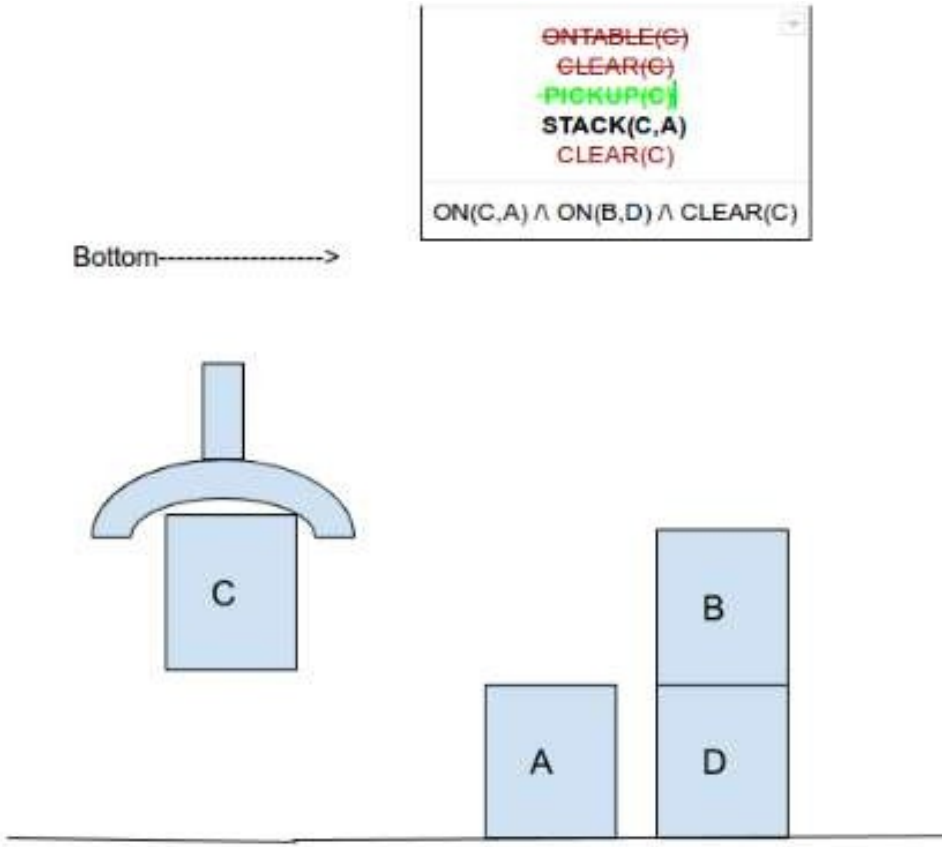


PLAN= { UNSTACK(B,C), STACK(B,D) }

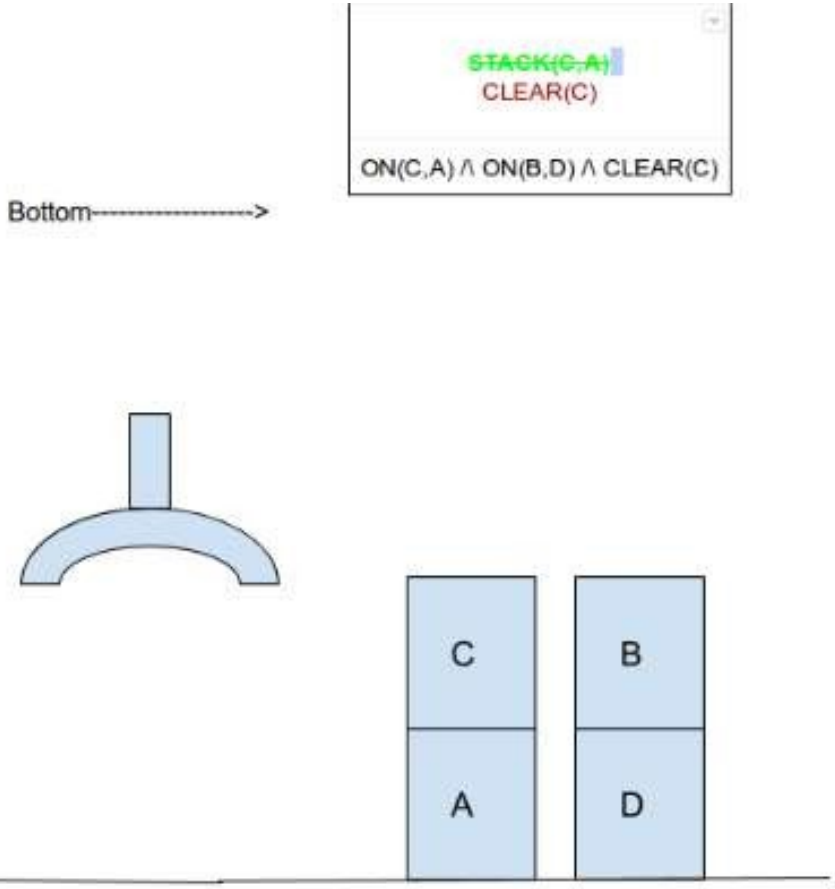



Bottom---------->

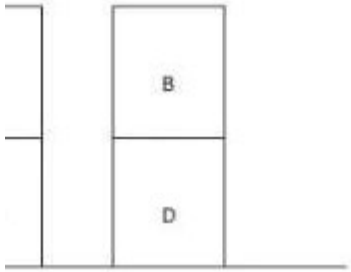
~~ON(C,A)~~
CLEAR(C)
ON(C,A) ∧ ON(B,D) ∧ CLE




PLAN= { UNSTACK(B,C), STACK(B,D) ,PICKUP(C) }




PLAN= { UNSTACK(B,D), STACK(B,D) ,PICKUP(C) ,STACK(C,A) }




**Goal State**

N(B,D) /\ ONTABLE(A) /\ ONTABLE(D)
\ CLEAR(B)

~~CLEAR(C)~~

~~ON(C,A) ^ ON(B,D) ^ CLEAR(C)~~

Bottom---------->

PLAN= { UNSTACK(B,D), STACK(B,D) ,PICKUP(C) ,STACK(C,A) }

# Means - Ends Analysis


- Search strategies either reason forward of backward
- Mixed strategy - solve the major parts of problem first and solve the smaller problems that arise when combining them together.
- Such a technique is called "Means - Ends Analysis".
- Means-Ends Analysis is problem-solving techniques used in Artificial intelligence for limiting search in AI programs.
- It is a mixture of Backward and forward search technique.
- The means -ends analysis process centers around finding the difference between current state and goal state.

# Means - Ends Analysis


**How means-ends analysis Works:**

- The means-ends analysis process can be applied recursively for a problem. It is a strategy to control search in problem-solving.

Following are the main Steps which describes the working of MEA technique for solving a problem.

1. **First, evaluate the difference between Initial State and final State.**
2. **Select the various operators which can be applied for each difference.**
3. **Apply the operator at each difference,which reduces the difference between the current state and goal state.**

# Example of Mean-Ends Analysis:

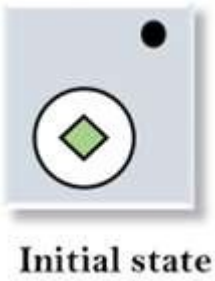
- Apply MEA to get the goal state.


- Solution:

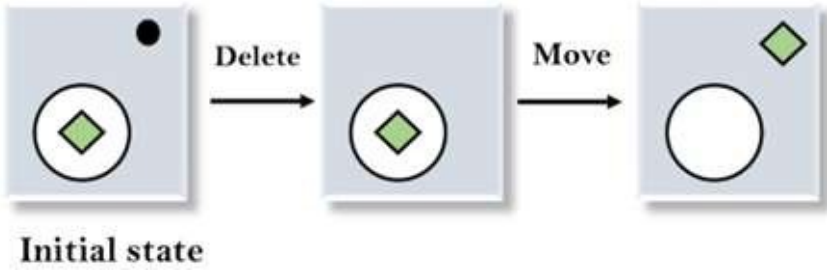
- To solve the above problem, first find the differences between initial states and goal states, and for each difference, generate a new state and will apply the operators. The operators we have for this problem are:
- **Move**
- **Delete**
- **Expand**

# Example of Mean-Ends Analysis:

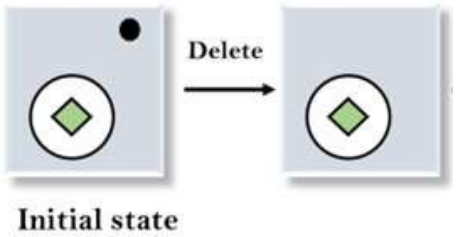
**Step 1: Evaluate Initial State**



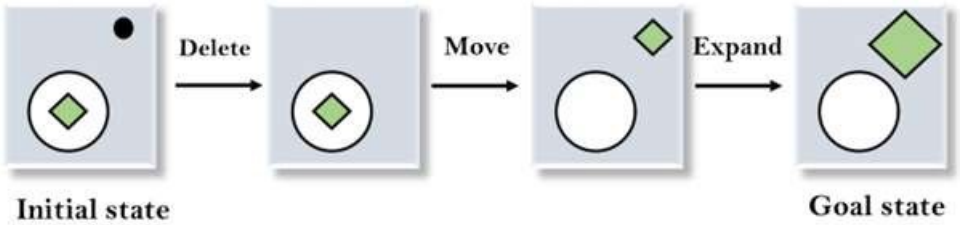

**Step 3: Apply Move Operator**




**Step 2: Apply Delete Operator**

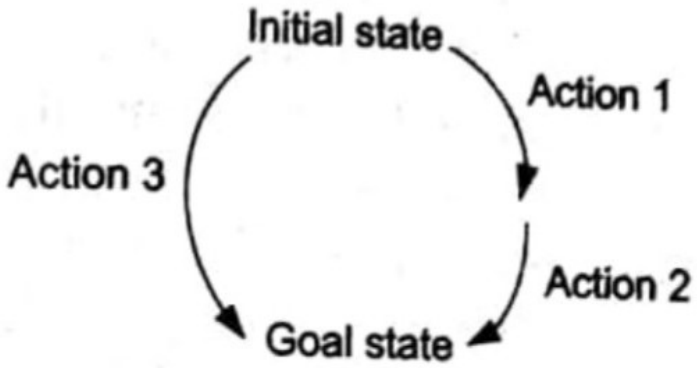



**Step 1: Apply Expand Operator**



# NON-LINEAR PLANNING

- A plan that consists of sub-problems, which are solved simultaneously is said non-linear plan.
- In case of the goal stack planning there are some problems. To achieve any goal, it could have an impact on the one that has been achieved.




Non-linear planning.

# Conditional Planning


- Conditional planning has to work regardless of the outcome of an action.
- The outcome of actions cannot be determined so the environment is said to be nondeterministic.
- It's a way to deal with uncertainty by checking what is actually happening in the environment at predetermined points in the plan. (Conditional Steps)

Example:

Check whether SFO airport (San Francisco International Airport) is operational. If so, fly there; otherwise, fly to some other place.

# Conditional Planning

- Three kind of

 Environments Fully Observable

 The agent always knows the current state Partially Observable

- The agent knows only a certain amount about the actual state. (much more common in real world)

 Unknown state

- The agent knows nothing about the current

# Conditional Planning in Fully Observable Environments


- Agent used conditional steps to check the state of the environment to decide what to do next.
- Plan information stores in a library Ex: Action(Left) Clean v Right

Syntax: If then plan_A else plan_B

# Reactive Planning

- Reactive planning is planning under uncertainty.
- Makes use of the if-then rules.
- The reactive planners are based on the concept that they should be able to handle an unknown situation too. So, the reaction rules are used that help them in doing so.
- A rule selection is based on the priority and a holding condition that maximises the priority.
- The rule which is at present in execution is said to be active whereas the and the ones with holding priority (we can call them possible competitors) are pre-active others are inactive.
- A B-tree structure is used in reactive planning, where the things are algorithm selects the rule. Sometimes, no rule can be selected. In such a case, dependent on the algorithm implementation for rule selection.

# LEARNING

- learning consists of various activities like understanding, memorisation, knowledge acquisition and inference.
- Learning is a continuous process.
- Learning from observation is required to construct meaningful classification of observed objects and situations.

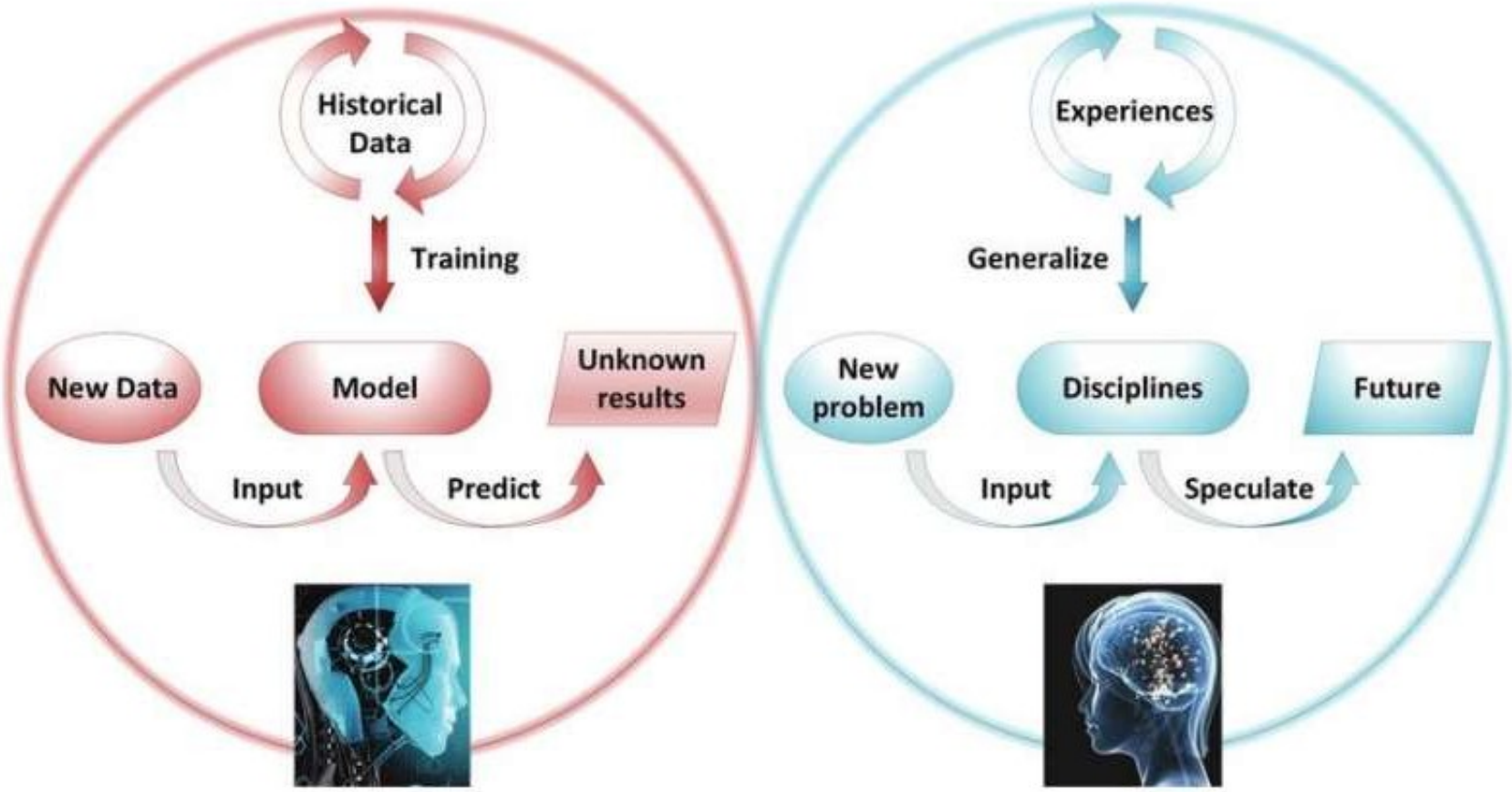
# Machine learning


- Machine learning is the branch of **artificial intelligence (AI)** which provide the ability to learning automatically learn and improve from experience. It was first introduced in **1959** by **Arthur Samuel**.

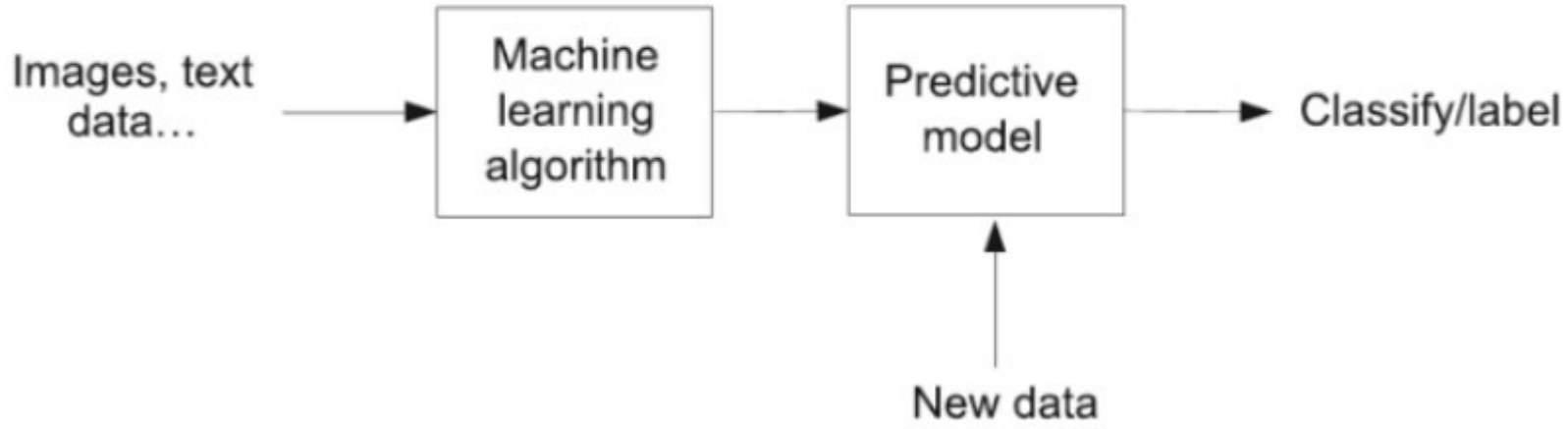
**Features of Machine Learning**

- Machine learning uses data to detect a various pattern in any dataset.
- It is a data-driven technology.
- It learns from past data and improves it automatically.
- Machine learning is similar to data mining, as it deals with a considerable amount of the data.

## Machine learning Vs Human Learning



# Machine Learning approach



# Scope of machine learning

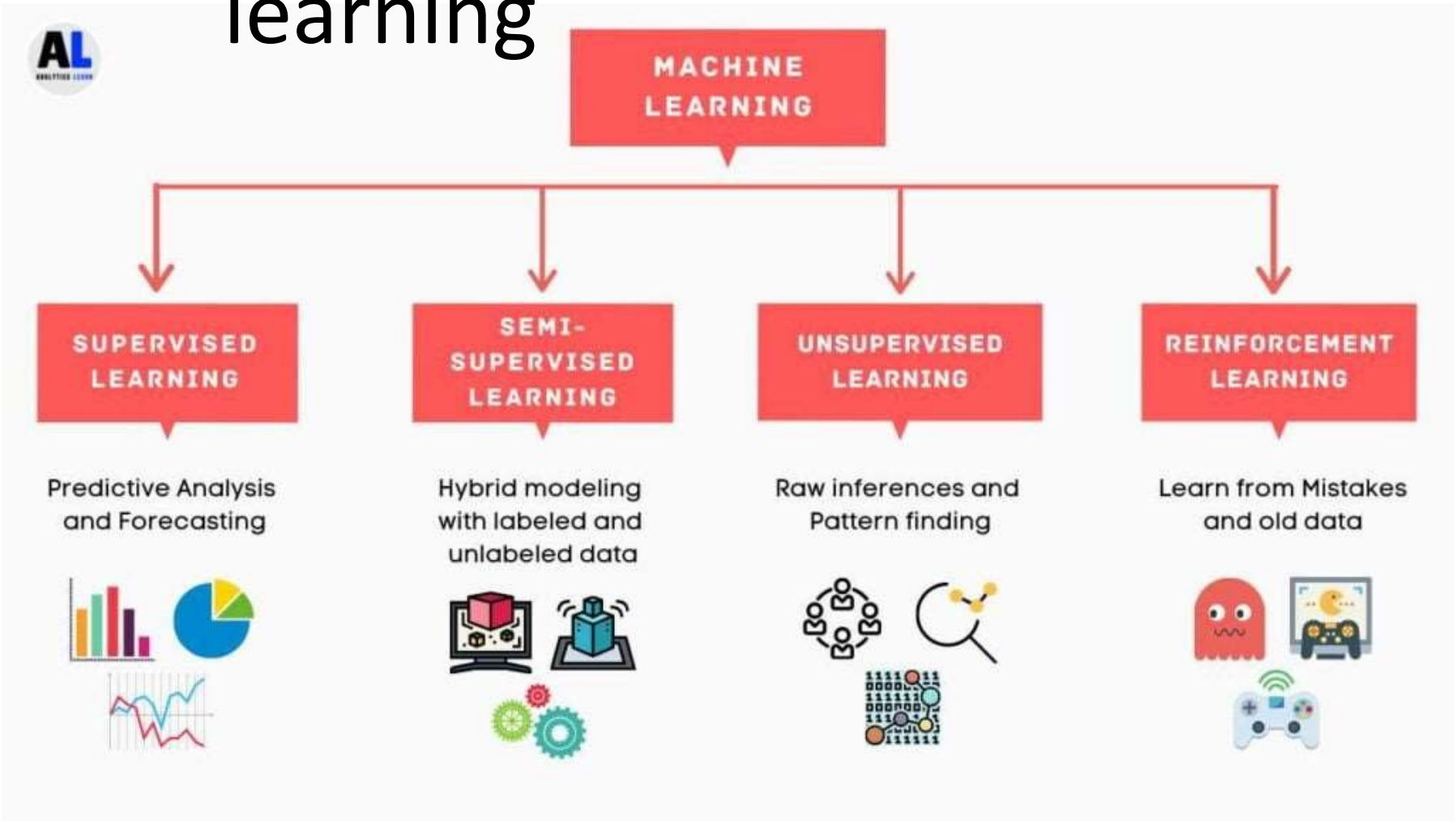

# GOALS OF MACHINE LEARNING

- To produce learning algorithms with practical value.
- Development and enhancement of computer algorithms and models to meet the decision making requirements in practical scenarios.
- To facilitate in building intelligent systems (IS) that can be used in solving real time problems.

# Challenges of Machine Learning

- Avaliability of limited learning data and unknown perspectives.
- Aquiring Accurate , compact and precise knowledge building.
- Require large working memory to store data.
- Focusing Too Much on Algorithms and Theories
- Monitoring and maintenance

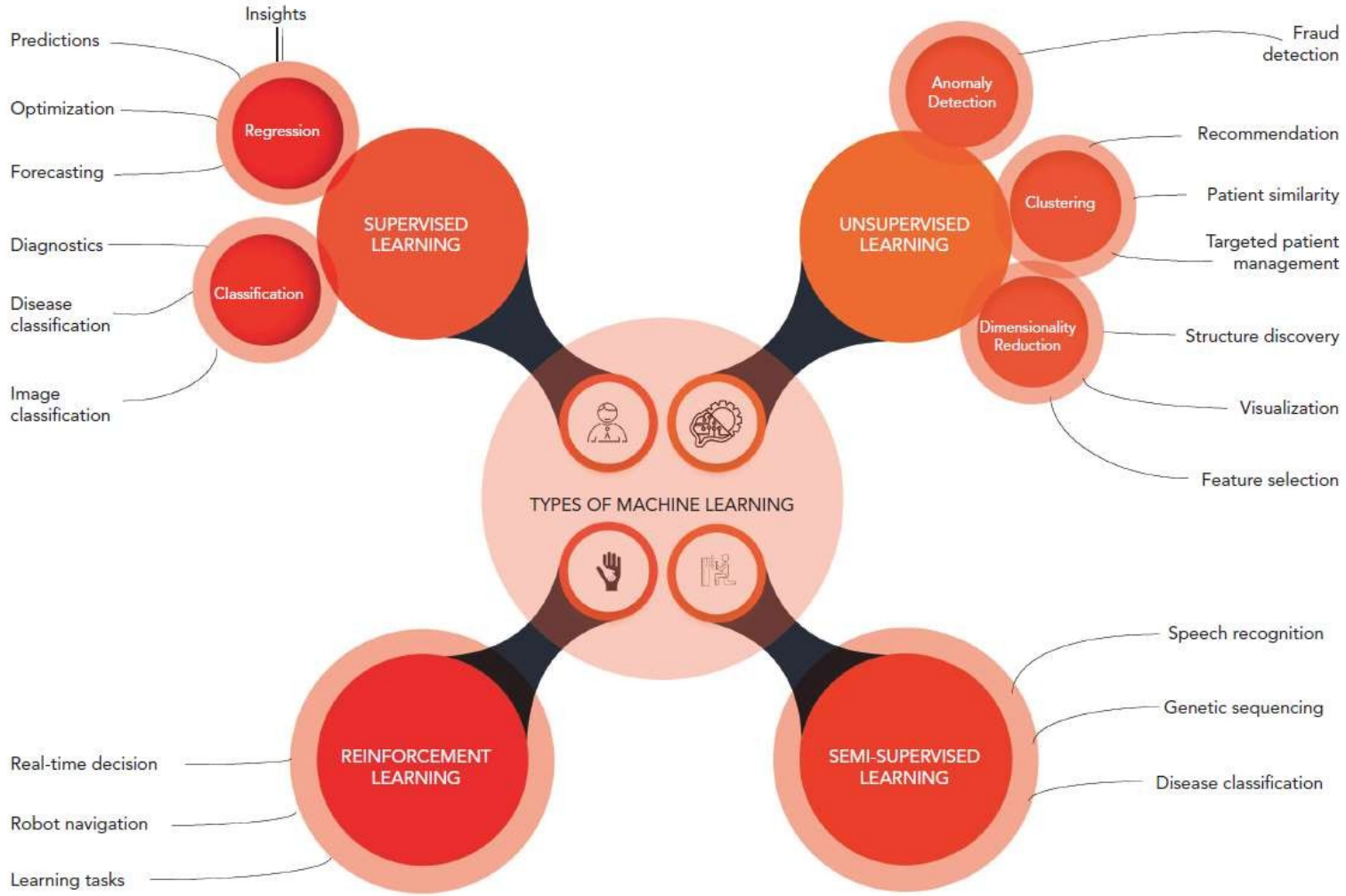
# Types of machine learning



# Styles of Learning

| Supervised | Unsupervised | Semi-Supervised | Reinforcement |
|---|---|---|---|
| • Data has **known labels** or output | • Labels or output unknown<br>• Focus on **finding patterns and gaining insight** from the data | • Labels or output known for **a subset of** data<br>• A blend of supervised and unsupervised learning | • Focus on **making decisions** based on previous experience<br>• Policy-making with feedback |
| • Insurance underwriting<br>• Fraud detection | • Customer clustering<br>• Association rule mining | • Medical predictions (where tests and expert diagnoses are expensive, and only part of the population receives them) | • Game AI<br>• Complex decision problems<br>• Reward systems |

Figure 1: Types of Machine Learning with Examples of Respective Use

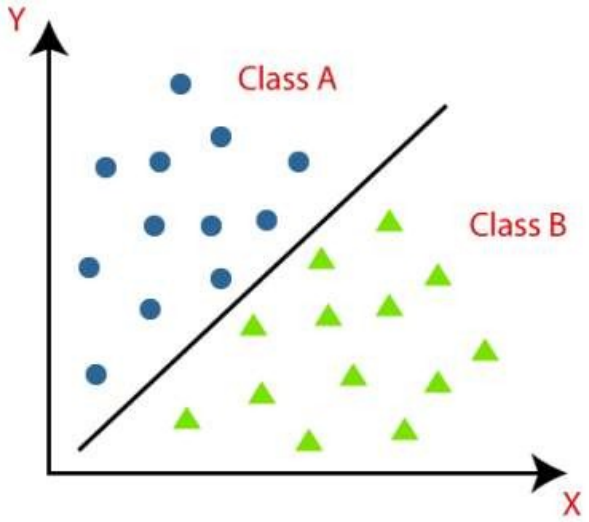


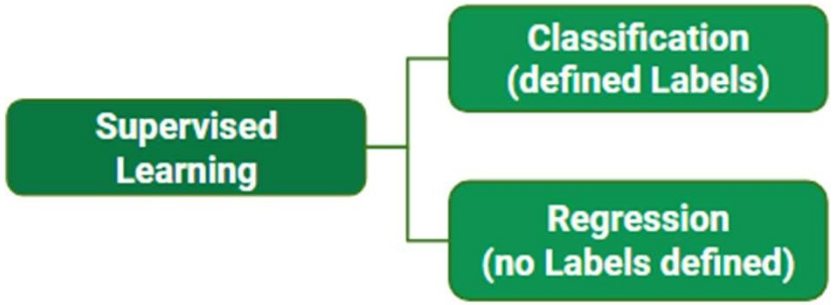
# Supervised learning

•Based on the labelled data set.

•supervised learning is the learning algorithm that is provided with the set of training data and the algorithm further induces the classifier to classify the unseen or new data.

• A line (hyperplane) which is generated after learning separating two classes class A and class the in two parts the classifier and the decision-making engine minimize the false positives and false negative.

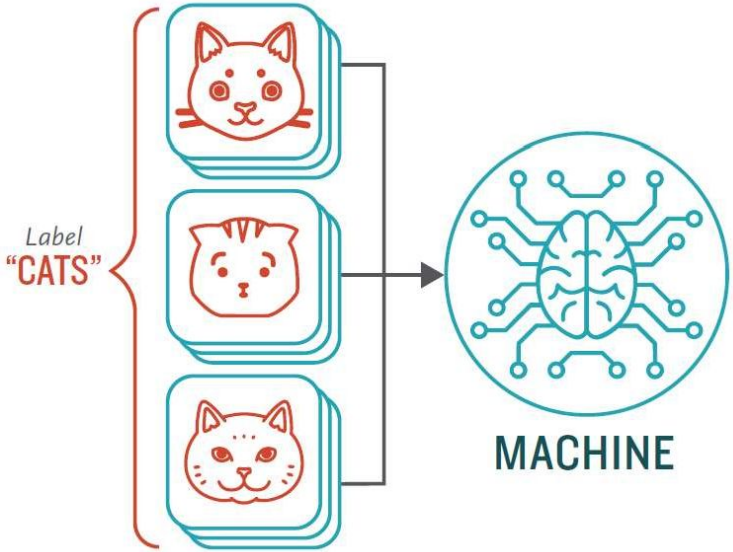


# How **Supervised** Machine Learning Works

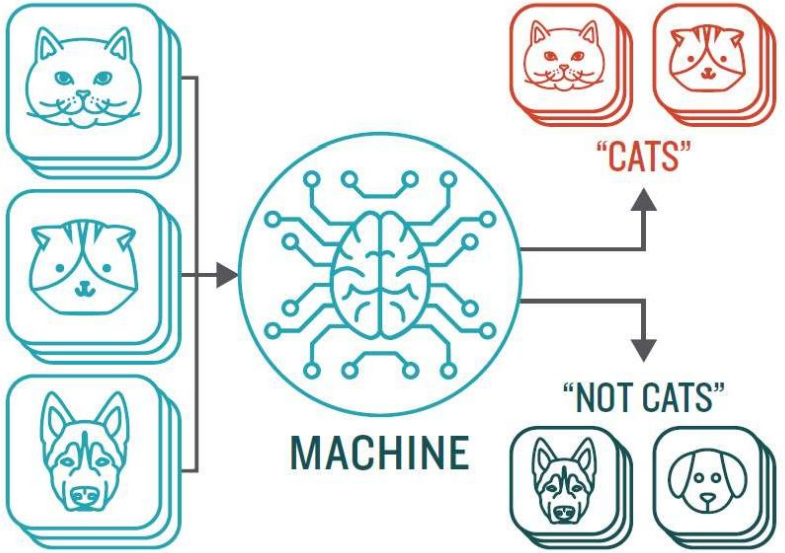
**STEP 1**

Provide the machine learning algorithm categorized or "labeled" input and output data from to learn

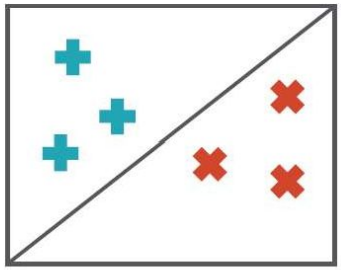



**STEP 2**

Feed the machine new, unlabeled information to see if it tags new data appropriately. If not, continue refining the algorithm

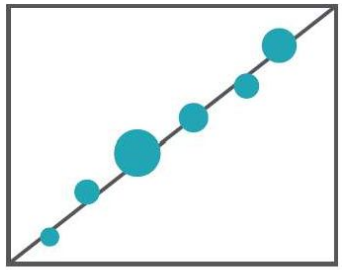



## TYPES OF PROBLEMS TO WHICH IT'S SUITED



### CLASSIFICATION

Sorting items into categories

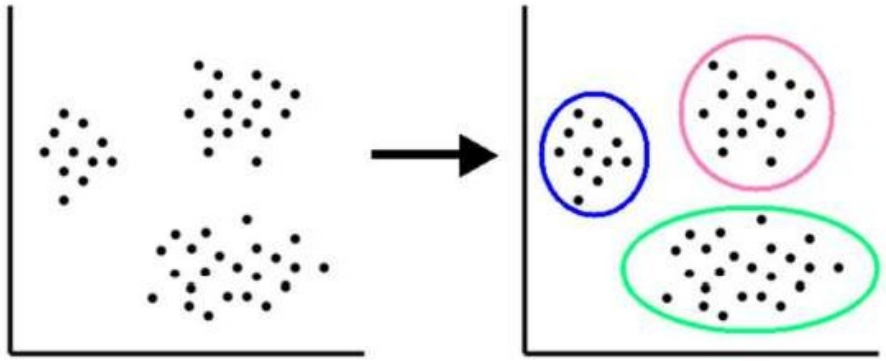


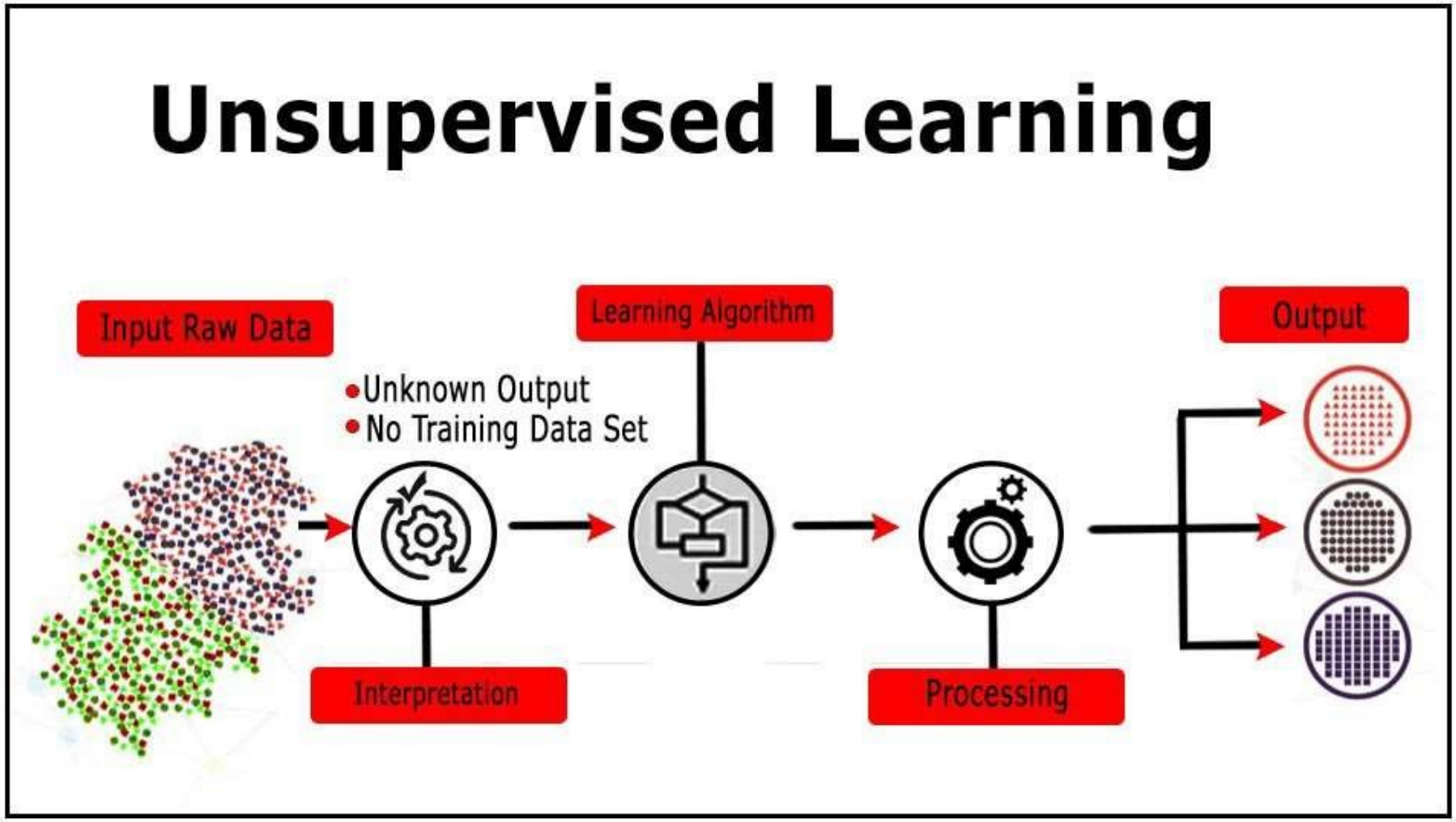
### REGRESSION

Identifying real values (dollars, weight, etc.)

# Unsupervised Learning

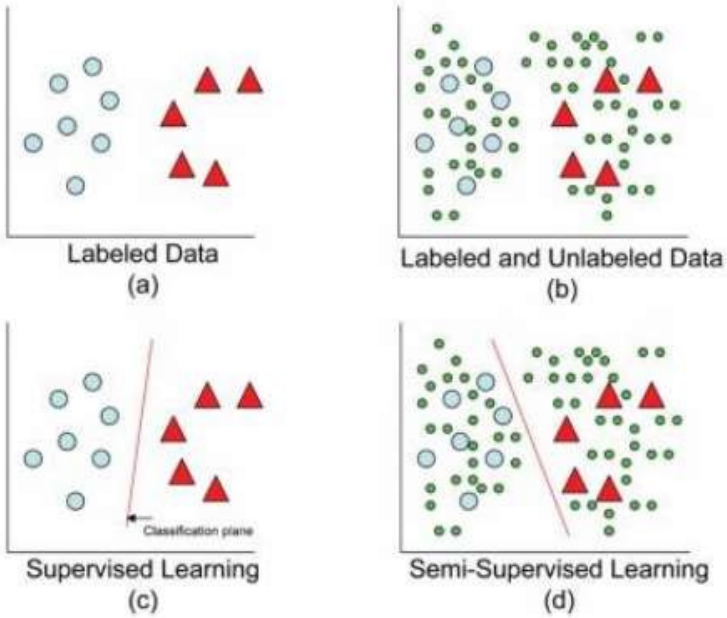
- Use unlabeled dataset.
- Learning is more based on similarities and differences which are visible. These differences and similarities are mathematically represented in unsupervised learning
- Grouping and categorisation of the objects is based on the understanding of similarities and visualisation of their relations
- Unsupervised learning performs hierarchical clustering.




Unsupervised Learning
Input Raw Data
Learning Algorithm
Output
•Unknown Output
•No Training Data Set
Interpretation
Processing

# Semi - Supervised Learning

- Semi supervised learning is developed to cope up with the issues of learning in supervised or unsupervised mode in isolation.
- Semi-supervised learning tries to learn from the labelled as well as unlabeled data.
- Let U be a set of unlabeled data and L be a set of labelled data.
- As the learning process the learning approach identifies the unlabeled data U with reference to a labelled data L and keeps on labelling the unlabeled data.
- This method is also called as self training in semi supervised learning.



# Learning Methods

- Artificial neural network based learning-Back propagation
- Support vector machines
- Reinforcement learning
- Adaptive learning
- Multi_agent based learning
- Ensemble learning
- Learning for decision making
- Distributed learning
- Speedup learning

# Artificial neural network based learning

ANN is a computational model that performs simulation of the human biological neurons.

The simulation is concerned with the functioning of neurons.

Artificial neurons have different inputs which are weighted as per the strength of signal and then computed by mathematical function which determines the activation of the neurons.

ANNs combined artificial neurons in order to process information the hidden layer is the one that Learns to recode for the input .

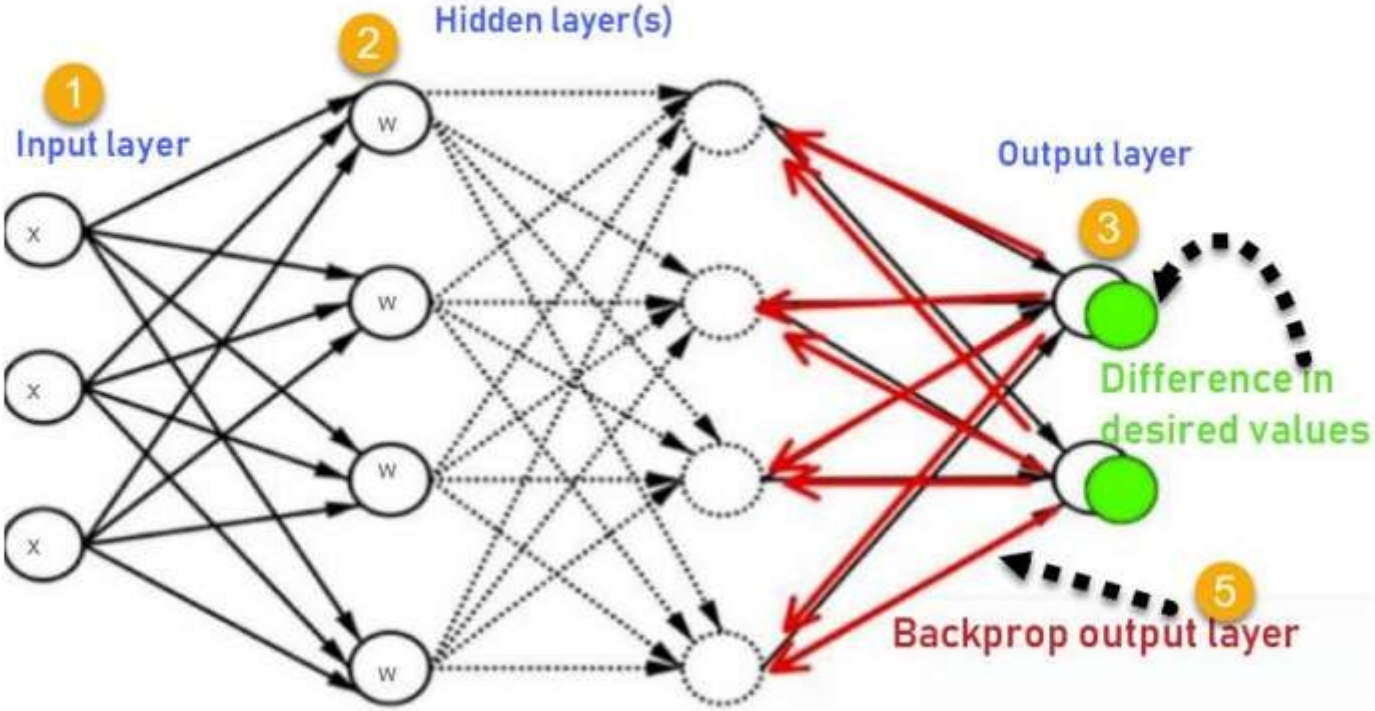

# Back- propagation Algorithm

Back-propagation is the essence of neural net training.

It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch (i.e., iteration).

Proper tuning of the weights allows you to reduce error rates and to make the model reliable by increasing its generalization.

# Backpropagation Algorithm



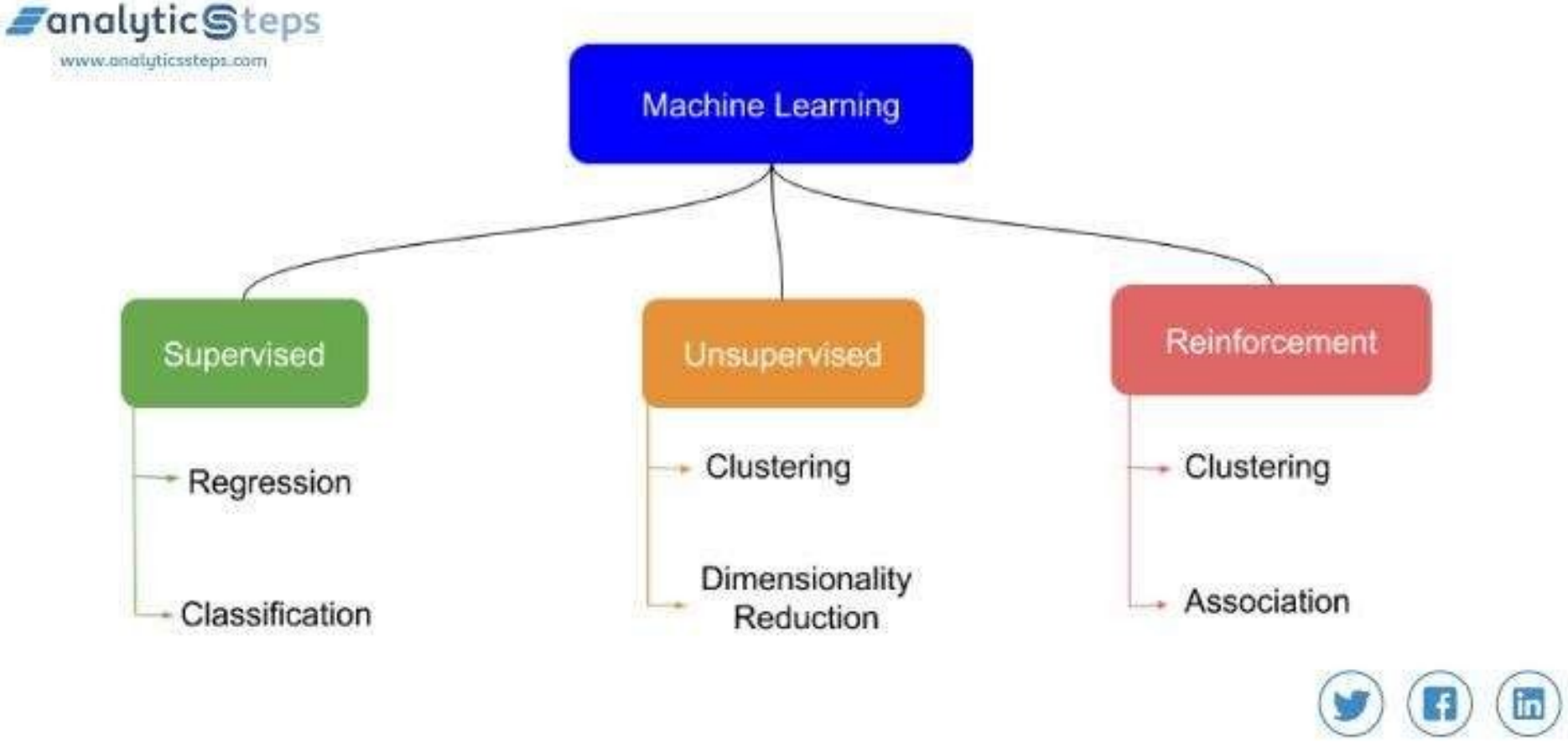
# Backpropagation Algorithm

1. Inputs X, arrive through the preconnected path
2. Input is modeled using real weights W. The weights are usually randomly selected.
3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
4. Calculate the error in the outputs.

```
ErrorB= Actual Output – Desired Output
```

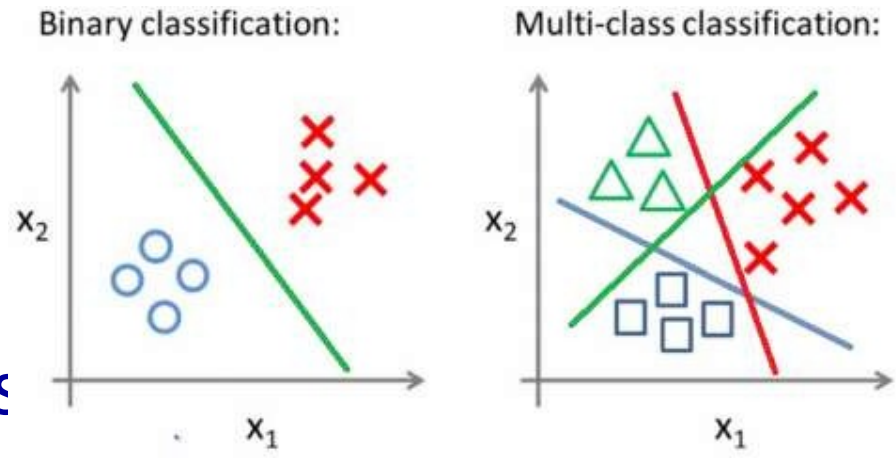
5. Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

Keep repeating the process until the desired output is achieved.


analyticSteps
www.analyticssteps.com
Machine Learning
Supervised
Regression
Classification
Unsupervised
Clustering
Dimensionality Reduction
Reinforcement
Clustering
Association

# What is Classification?

It is a process of categorizing a given set of feature vectors into two or more classes.

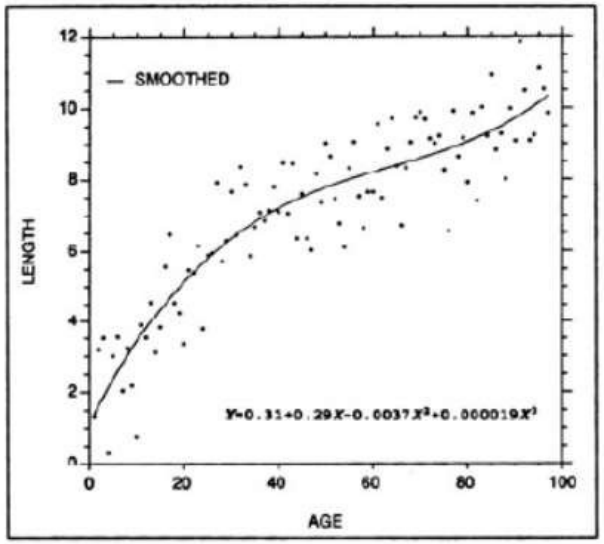



# Classification Techniques

- Logistic Regression
- Artificial Neural Networks (ANN)
- **Support Vector Machine (SVM)**
- Deep Neural Network (DNN)
- Convolution Neural Network (1D-CNN and 2D-CNN)

# Regression or Prediction

- **Prediction vs data classification**
  - Similarities: Both learn from a data set
  - Difference:
    - **In classification, each example has a class associated**
    - **In prediction, each example has a numerical value associated**




# Linear Regression Example

- Price of the house is dependent on size of the house

| Size of the house(sft) | Price(in Rs) |
|---|---|
| 1000 | 5 lakh |
| 2000 | 15 lakh |
| 800 | 4 lakh |

- What will be price of the house if size of the house is 1200?

# Support Vector Machines (SVM)

- Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset.
- SVM or Support Vector Machine is a linear model for classification and regression problems.
- Can solve linear and non-linear problems and work well for many practical problems.

The idea of SVM is simple:

- ***The algorithm creates a line or a hyperplane which separates the data into classes.***

# Motivation for Support Vector Machines


- The problem to be solved is one of the **supervised binary classification**. That is, we wish to categorize new unseen objects into two separate groups based on their properties and a set of known examples, which are already categorized.
- A good example of such a system is classifying a set of new *documents* into positive or negative sentiment groups, based on other documents which have already been classified as positive or negative.
- Similarly, we could classify new emails into spam or non-spam, based on a large corpus of documents that have already been marked as spam or non-spam by humans. SVMs are highly applicable to such situations.

# Motivation for Support Vector Machines

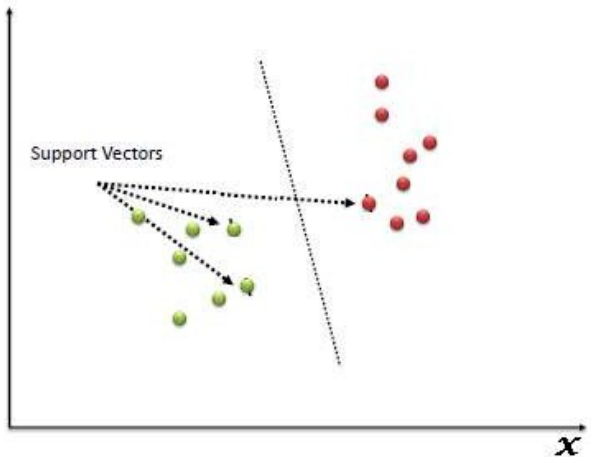

- A Support Vector Machine models the situation by creating a *feature space*, which is a finite-dimensional vector space, each dimension of which represents a "feature" of a particular object. In the context of spam or document classification, each "feature" is the prevalence or importance of a particular word.
- The **goal of the SVM** is to train a model that assigns new unseen objects into a particular category.
- It achieves this by creating a linear partition of the feature space into two categories.
- Based on the features in the new unseen objects (e.g. documents/emails), it places an object "above" or "below" the separation plane, leading to a categorization (e.g. spam or non-spam). This makes it an example of a non-probabilistic linear classifier. It is non-probabilistic, because the features in the new objects fully determine its location in feature space and there is no stochastic element involved.

# OBJECTIVES

- **Support vector machines (SVM)** are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
- It is a **machine learning** approach.
- They analyze the large amount of data to identify patterns from them.
- SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown in the image below.

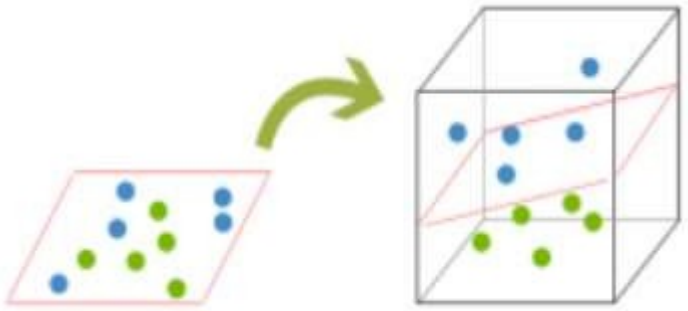
# Support Vectors

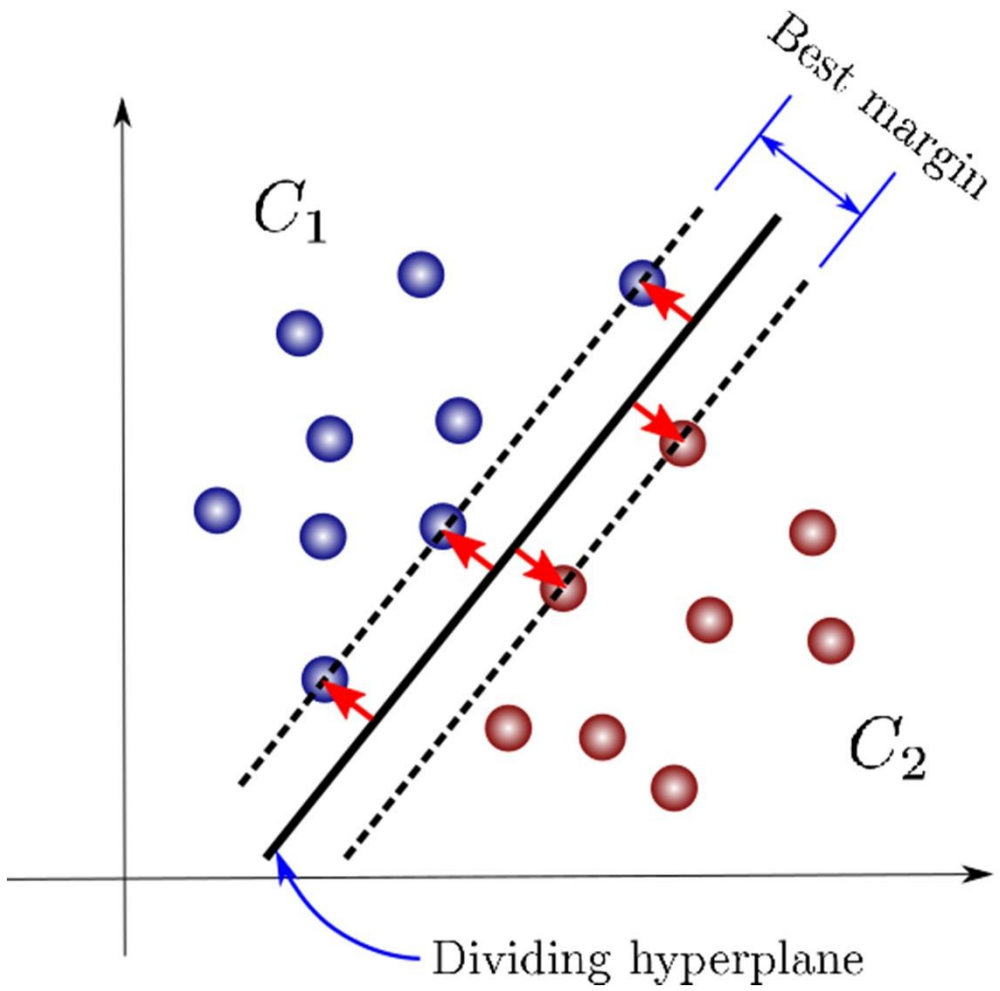
- ***Support vectors are the data points that lie closest to the decision surface (or hyperplane)***
- They are the data points most difficult to classify
- They have direct bearing on the optimum location of the decision surface
- We can show that the optimal hyperplane stems from the function class with the lowest "capacity" (VC dimension).
- Support vectors are the data points nearest to the hyperplane, the points of a data set that, ***if removed, would alter the position of the dividing hyperplane.*** Because of this, they can be considered the critical elements of a data set.



# What is a hyperplane?

- There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but ***we need to find out the best decision boundary that helps to classify the data points.*** This best boundary is known as the hyperplane of SVM.
- The dimensions of the hyperplane depend on the features present in the dataset.
- **2 features** □ a **straight line**. **3 features** □ a **2-dimension plane.**

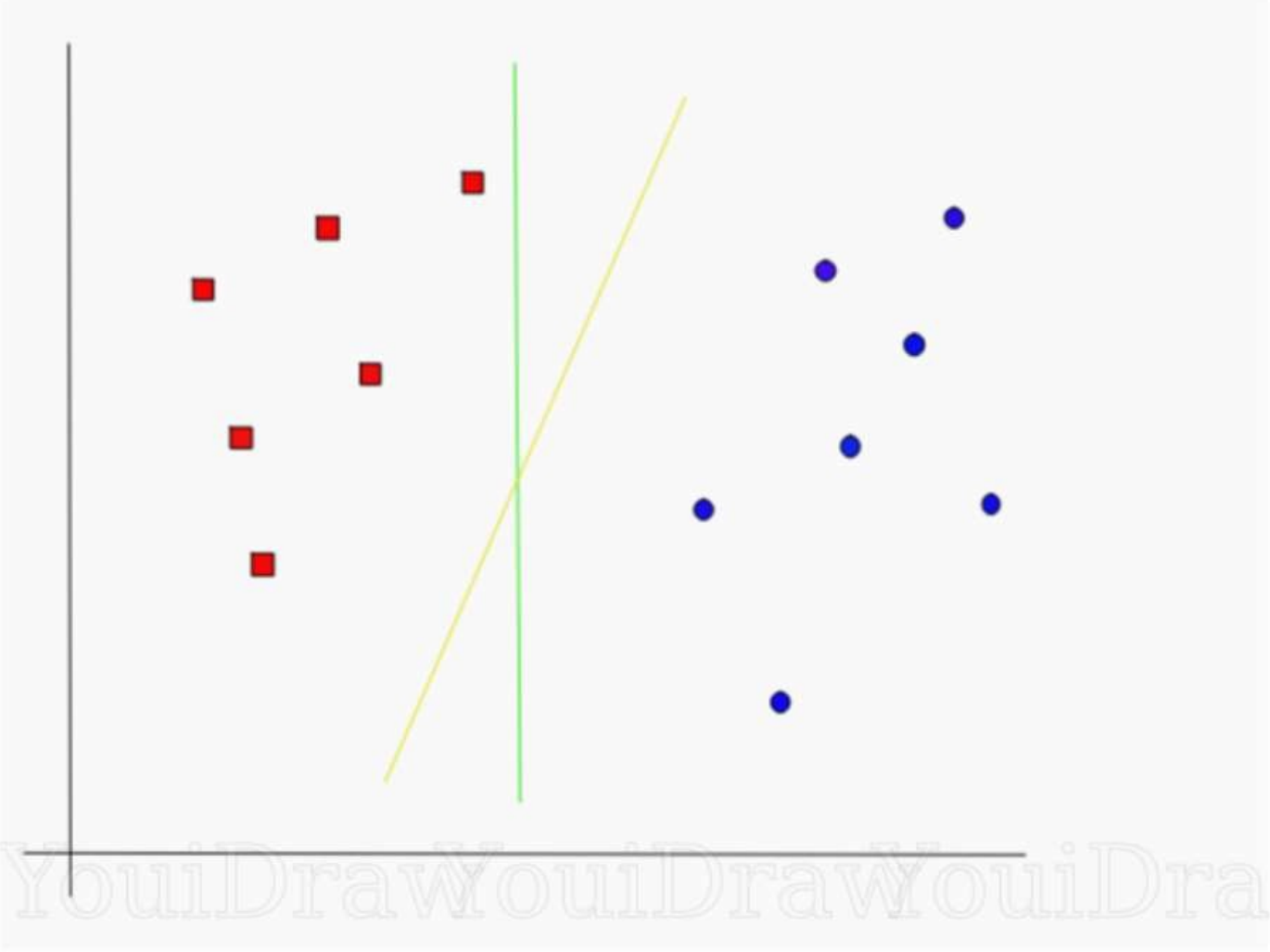



Best margin
$C_1$
$C_2$
Dividing hyperplane

# Types of SVM

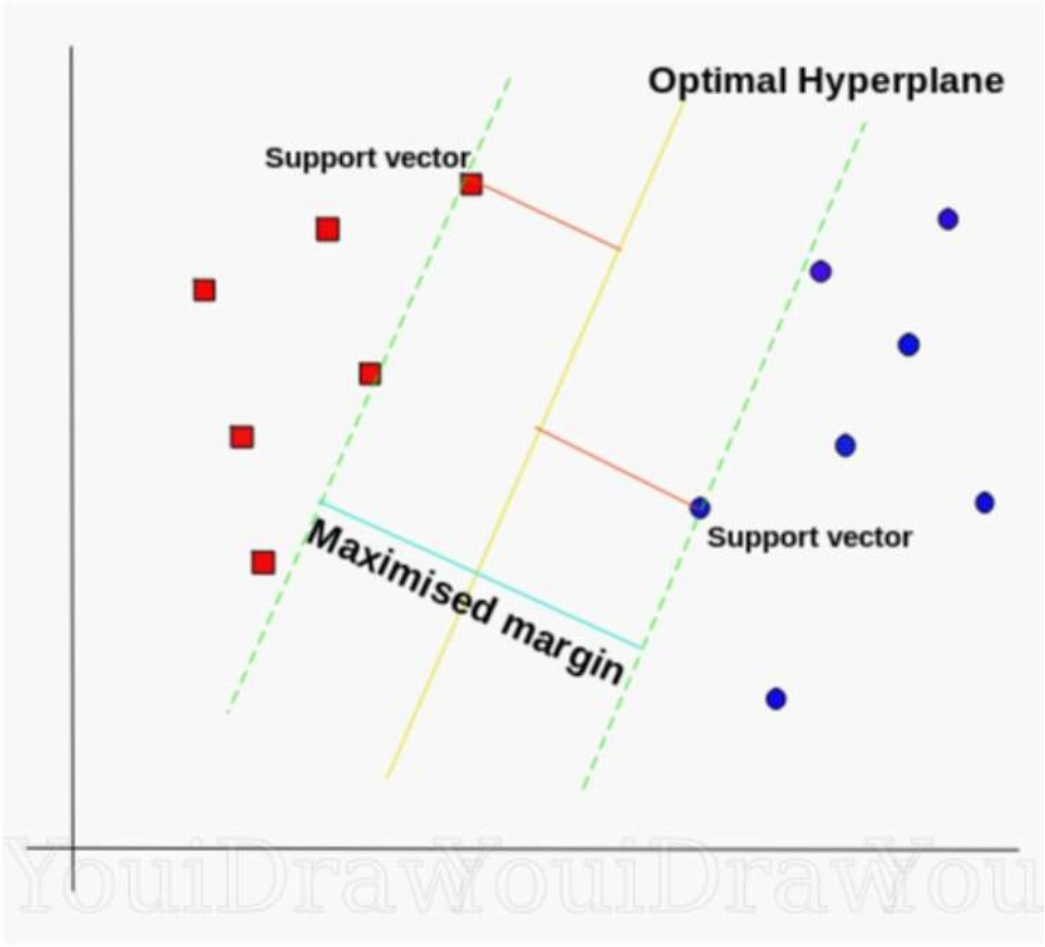
**SVM can be of two types:**

- **Linear SVM:** Linear SVM is used for linearly separable data
- **Non-linear SVM:** Non-Linear SVM is used for non- linearly separated data



Which line according to you best separates the data???

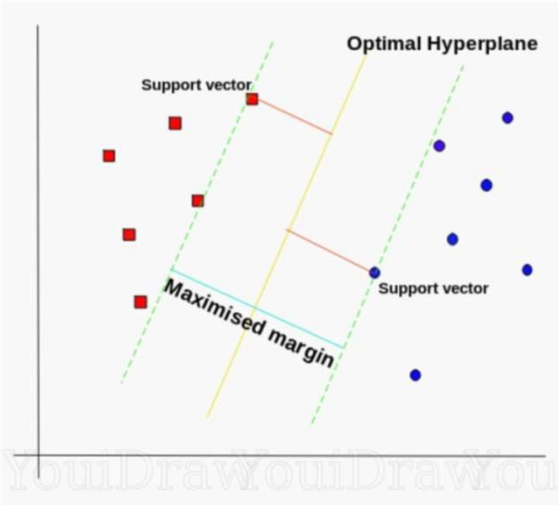
# SVM's way to find the best line




Optimal Hyperplane using the SVM algorithm

# How do we find the right hyperplane?

- How do we best segregate the two classes within the data?
- The distance between the hyperplane and the nearest data point from either set is known as the **margin**. The goal is to choose a hyperplane with the greatest possible margin between the hyperplane and any point within the training set, giving a greater chance of new data being classified correctly. **There will never be any data point inside the margin.**
- **The hyperplane for which the margin is maximum is the optimal hyperplane.**




Optimal Hyperplane using the SVM algorithm

Thus SVM tries to make a decision boundary in such a way that the separation between the two classes(that street) is as wide as possible.

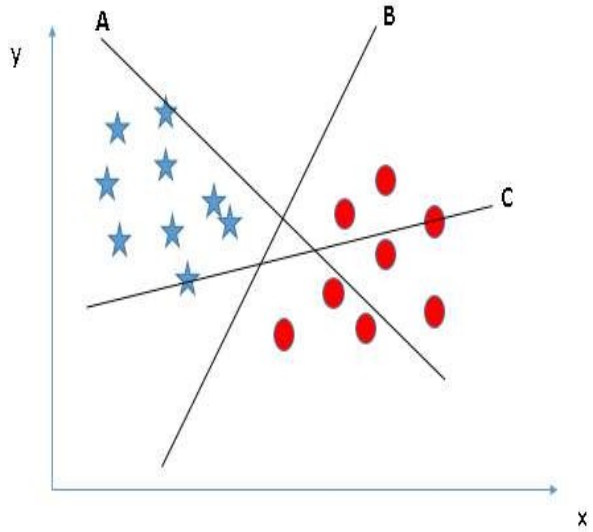
# How does it work? How can we identify the right hyper-plane?


- You need to remember a thumb rule to identify the right hyper-plane:

"Select the hyper-plane which segregates the two classes better".

# Identify the right hyperplane (Scenario-1):

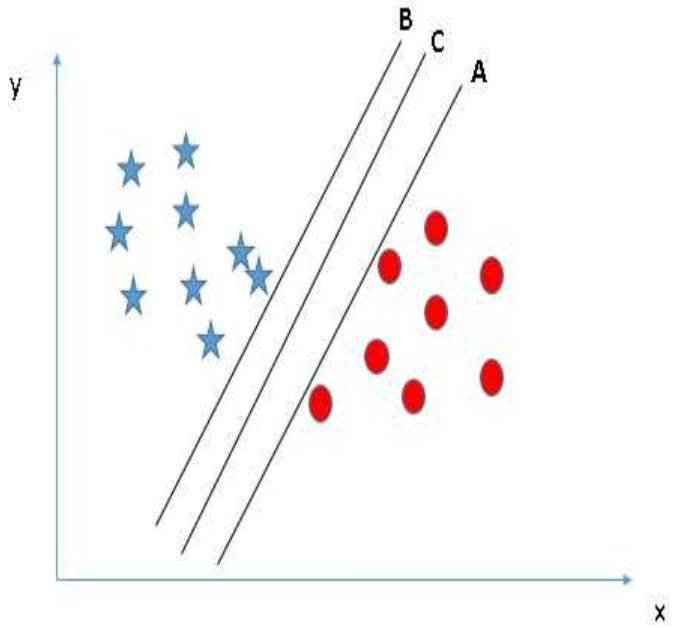
- Here, we have three hyperplanes (A, B and C). Now, identify the right hyperplane to classify star and circle.




- Hyperplane “B” has excellently performed this job.

# Identify the right hyperplane (Scenario-2):

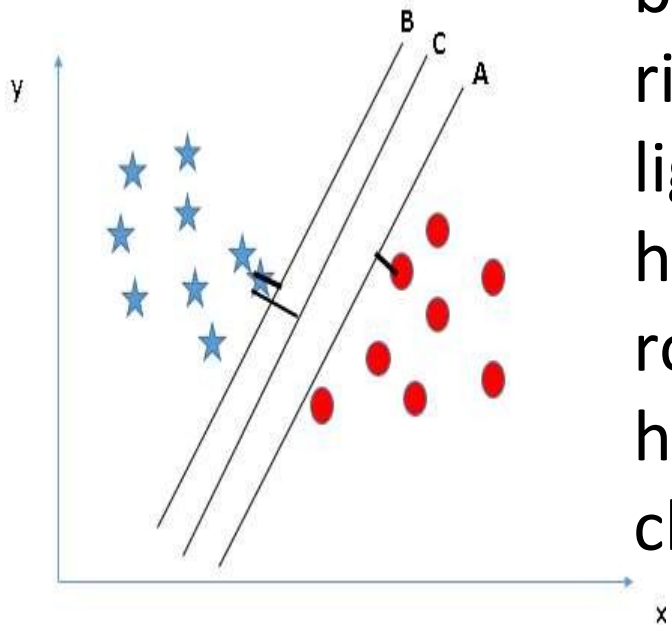
- Here, we have three hyperplanes (A, B and C) and all are segregating the classes well. Now, how can we identify the right hyperplane?




Here, maximizing the distances between nearest data point (either class) and hyperplane will help us to decide the right hyperplane.

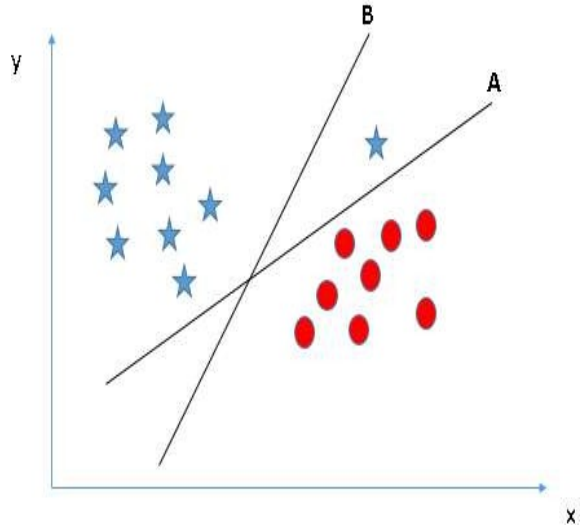
# Scenario-2

This distance is called as **Margin**. Let's look at the below snapshot:




We can see that the margin for hyperplane C is high as compared to both A and B. Hence, we name the right hyperplane as C. Another lightning reason for selecting the hyperplane with higher margin is robustness. If we select a hyperplane having low margin then there is high chance of missclassification.

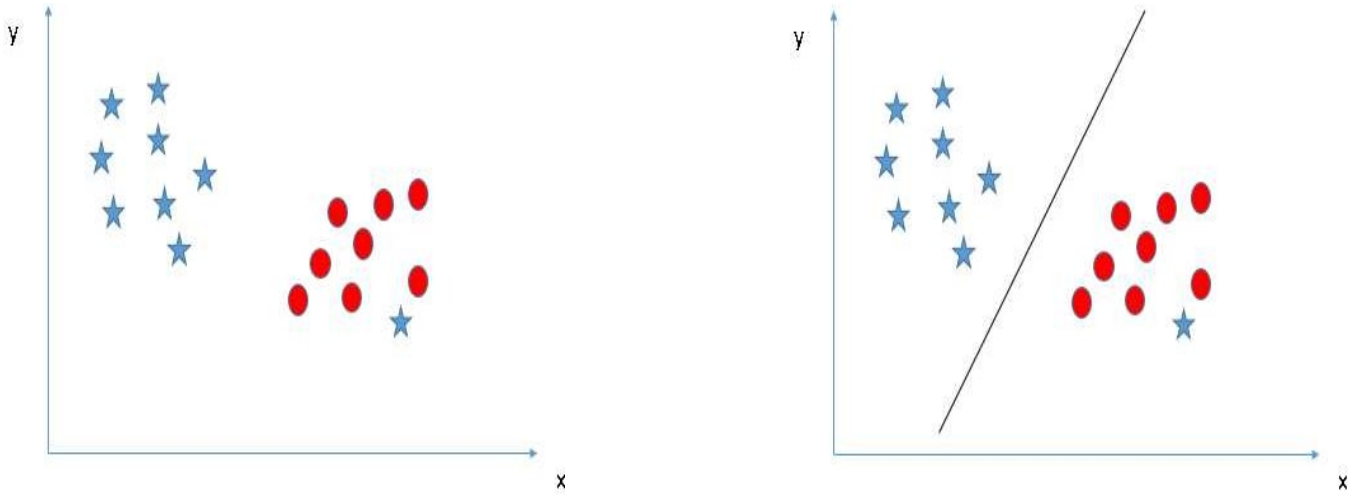
# Identify the right hyperplane (Scenario-3)




- Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But, here is the catch, SVM selects the hyperplane which **classifies the classes accurately prior to maximizing margin.**
- Here, hyperplane B has a classification error and A has classified all correctly. Therefore, the right hyperplane is **A.**

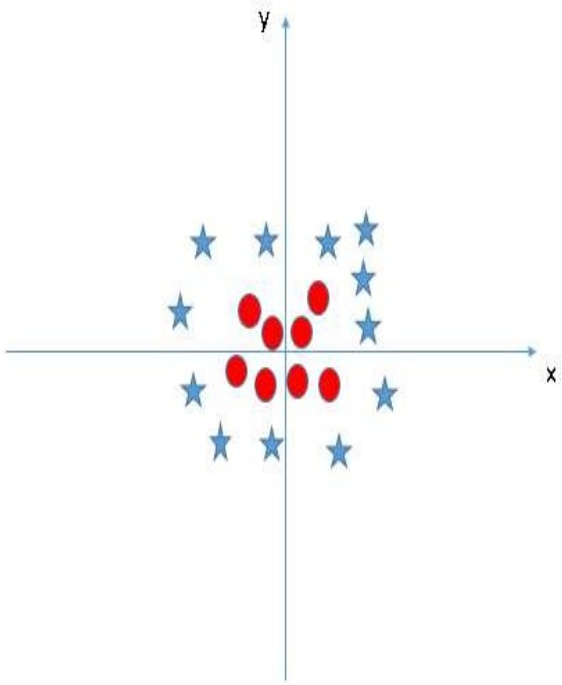
# Identify the right hyperplane (Scenario-4)

- We are unable to segregate the two classes using a straight line, as one of star lies in the territory of other (circle) class as an outlier.
- One star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyperplane that has maximum margin. Hence, we can say, SVM is robust to outliers.



# Find the hyperplane to segregate to classes (Scenario-5)

- In the scenario below we can't have linear hyperplane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyperplane. This is a non-linear data.

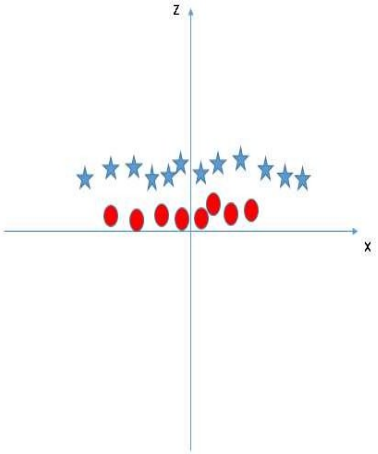



SVM can solve this problem. It solves this problem by introducing additional dimension. Here, we will add a new dimension

$z=x^2+y^2$.

# Scenario-5 (Contd.)

- Now, let's plot the data points on axis x and z:



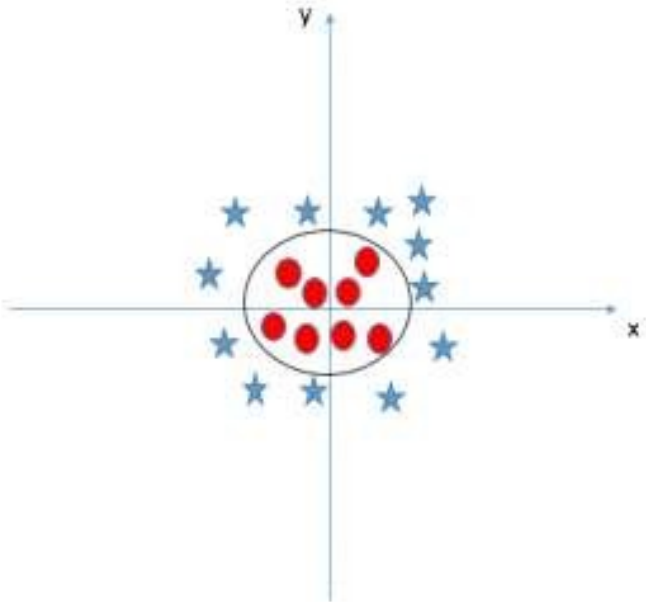

- In above plot, points to consider are:
  - All values for z would be positive always because z is the squared sum of both x and y
  - In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

# Scenario-5 (Contd..)

- Now the data is clearly linearly separable. Let the purple line separating the data in higher dimension be z=k, where k is a constant.
- Since, $\mathbf{z=x^2+y^2}$ we get $x^2 + y^2 = k$; which is an **equation of a circle**. So, we can project this linear separator in higher dimension back in original dimensions using this transformation.

# Scenario-5

- When we look at the hyperplane in original input space it looks like a circle:

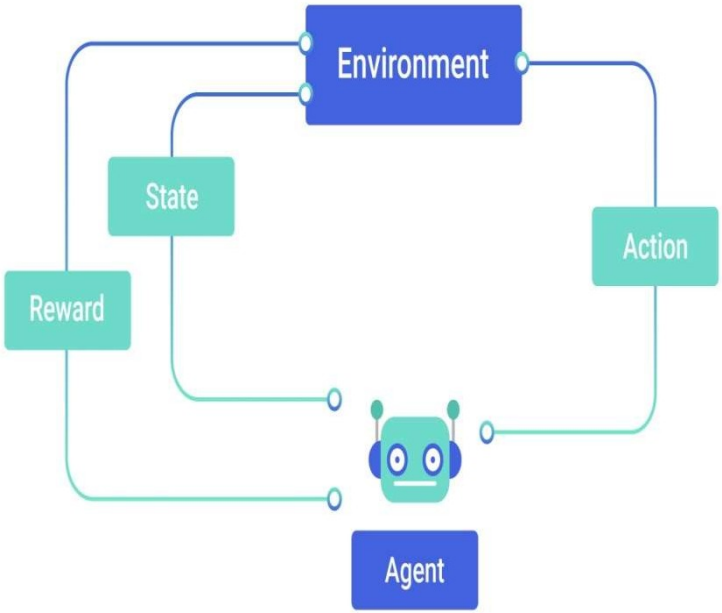


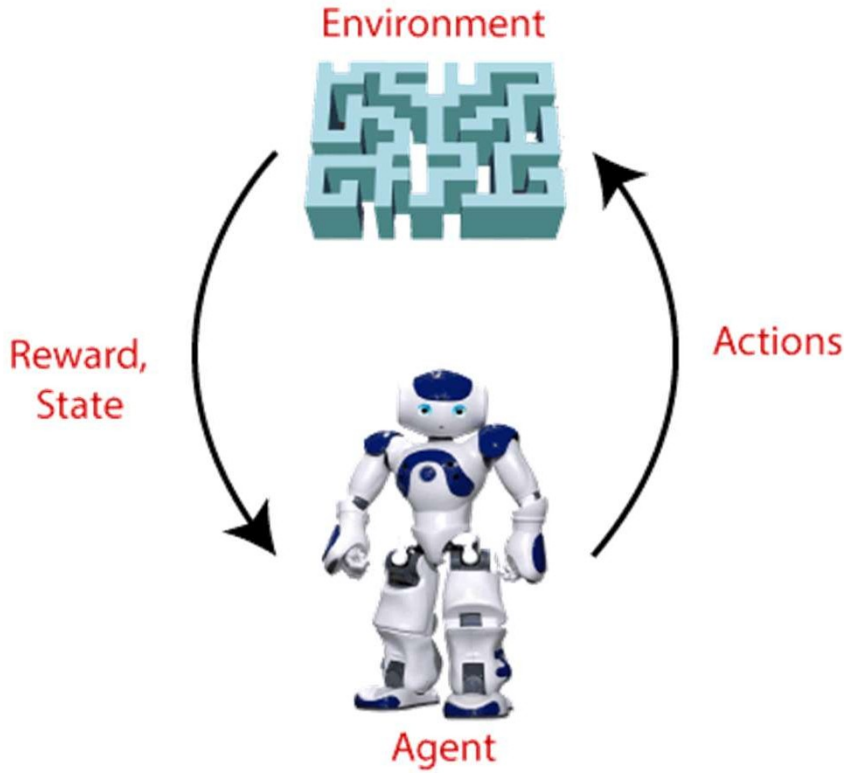
# What is Reinforcement Learning?


- Learning from interaction with an environment to achieve some long-term goal that is related to the state of the environment
- The goal is defined by reward signal, which must be maximised.
- Agent must be able to partially/fully sense the environment state and take actions to influence the environment state
- The state is typically described with a feature-vector

# Reinforcement learning

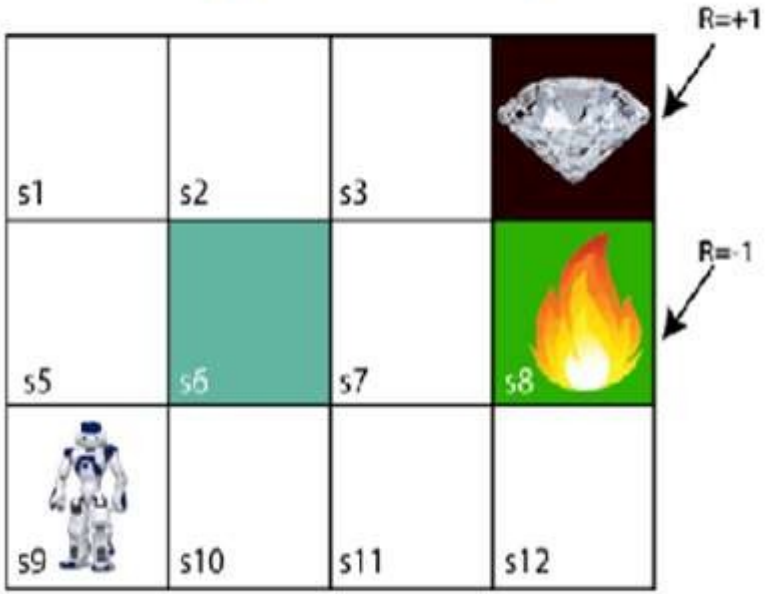
***"Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that."***

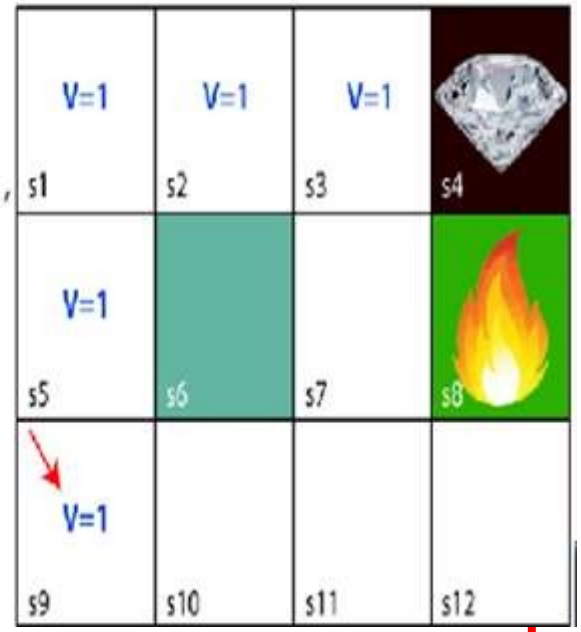


# Reinforcement Learning Example

Let's take an example of a maze environment that the agent needs to explore. Consider the below image:



# Terms used in Reinforcement Learning


- **Agent**
- **Environment**
- **Action**
- **State**
- **Reward**
- **Policy**
- **Value**
- **Q-value**

# Reinforcement Learning Systems

- Reinforcement learning systems have 4 main elements:
  - Policy
  - Reward signal
  - Value function
  - Optional model of the environment

# Policy


- A policy is a mapping from the perceived states of the environment to actions to be taken when in those states
- A reinforcement learning agent uses a policy to select actions given the current environment state

# On-policy versus Off-policy


- An on-policy agent learns only about the policy that it is executing
- An off-policy agent learns about a policy or policies different from the one that it is executing

# Reward Signal

- The reward signal defines the goal
- On each time step, the environment sends a single number called the reward to the reinforcement learning agent
- The agent's objective is to maximise the total reward that it receives over the long run
- The reward signal is used to alter the policy

# Value Function (1)


- The reward signal indicates what is good in the short run while the value function indicates ***what is good in the long run***
- The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting in that state
- Compute - using the states that follow the current state and the rewards available in those states

# Value Function (2)

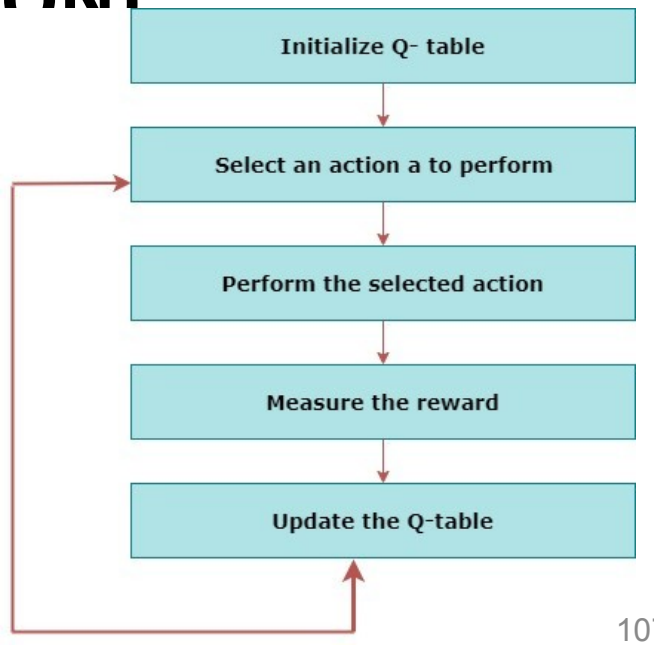

- Use the values to make and evaluate decisions
- Action choices - Based on value judgements
- Prefer actions - Bring about states of highest value instead of highest reward
- Rewards - given directly by the environment
- Values must continually be re-estimated from the sequence of observations that an agent makes over its lifetime

# Model

- Model - mimics the behavior of the environment.
- With the help of the model, one can make inferences about how the environment will behave.
- The model is used for planning, which means it provides a way to take a course of action by considering all future situations before actually experiencing those situations.
- The approaches for solving the RL problems **with the help of the model** are termed as the **model-based approach**. Comparatively, an approach **without using a model** is called a **model-free approach**.

# Reinforcement Learning Algorithms

- **Q-Learning**
- **State Action Reward State action (SARSA)**
- **Deep Q Neural Network (DQN)**



# Types of Reinforcement learning

- **Positive Reinforcement**
- **Negative Reinforcement**

**Positive Reinforcement:**

- The positive reinforcement learning means adding something to increase the tendency that expected behavior would occur again.

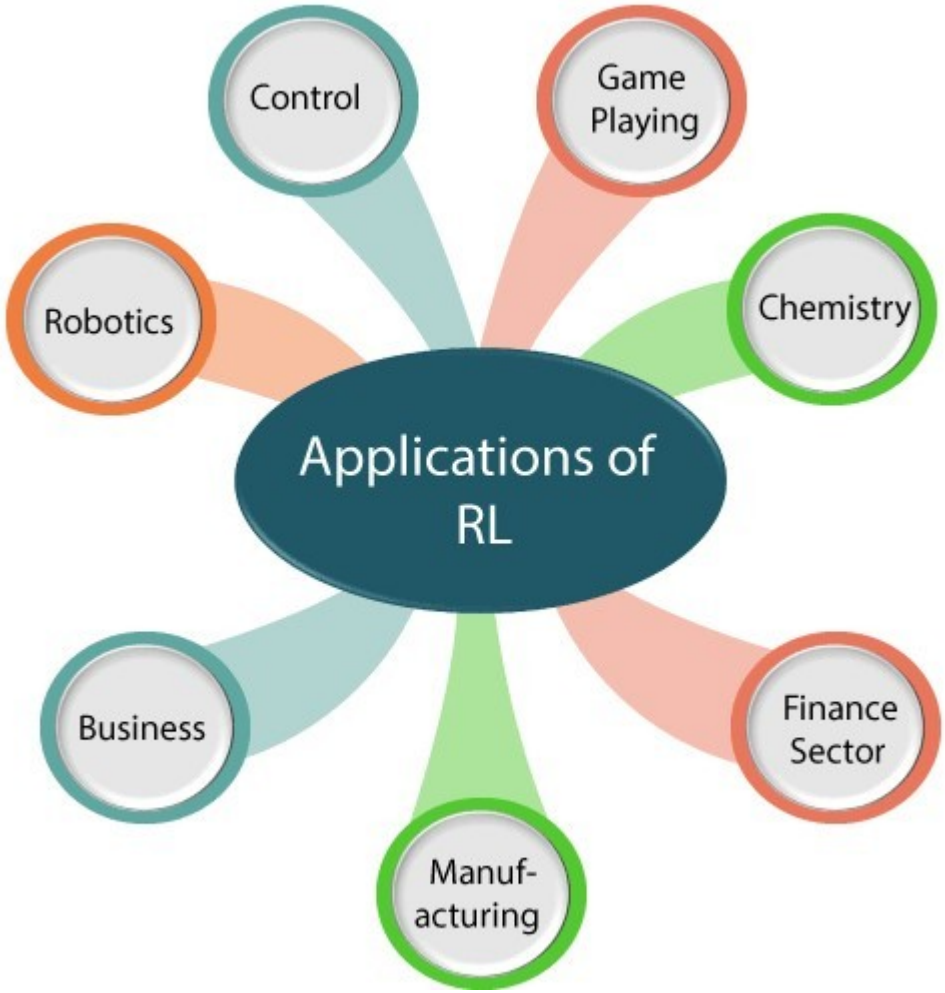
**Negative Reinforcement:**

- The negative reinforcement learning is opposite to the positive reinforcement as it increases the tendency that the specific behavior will occur again by avoiding the negative condition.

# Diff b/w RL and SL


| Reinforcement Learning | Supervised Learning |
|---|---|
| RL works by interacting with the environment. | Supervised learning works on the existing dataset. |
| The RL algorithm works like the human brain works when making some decisions. | Supervised Learning works as when a human learns things in the supervision of a guide. |
| There is no labeled dataset is present | The labeled dataset is present. |
| No previous training is provided to the learning agent. | Training is provided to the algorithm so that it can predict the output. |
| RL helps to take decisions sequentially. | In Supervised learning, decisions are made when input is given. |

# Reinforcement Learning Applications



# Advantages of Reinforcement Learning

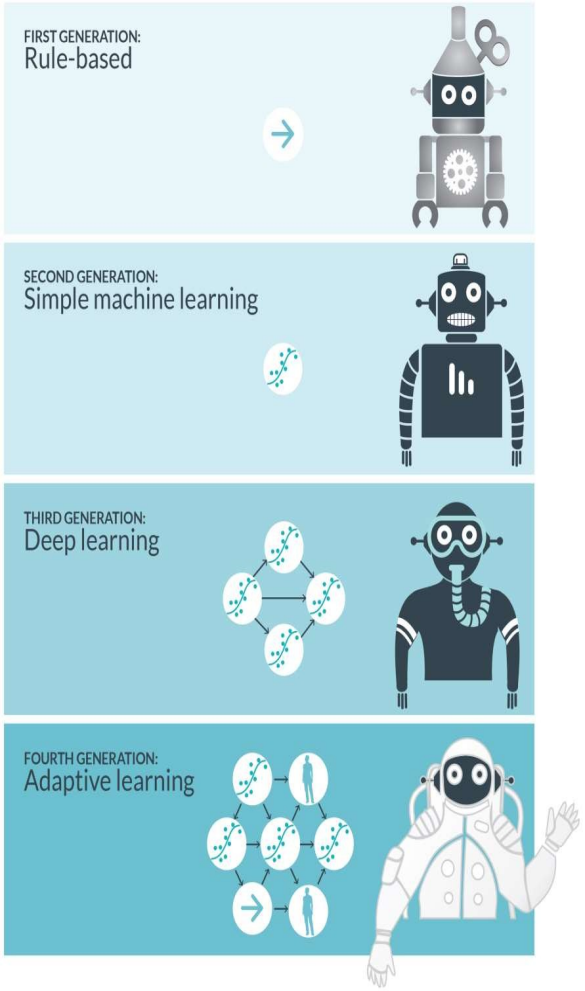

- It can solve higher-order and complex problems. Also, the solutions obtained will be very accurate.
- The reason for its perfection is that it is very similar to the human learning technique.
- Due to it's learning ability, it can be used with neural networks. This can be termed as **deep reinforcement learning**.
- Since the model learns constantly, a mistake made earlier would be unlikely to occur in the future.
- Various problem-solving models are possible to build using reinforcement learning.
- When it comes to creating simulators, object detection in automatic cars, robots, etc., reinforcement learning plays a great role in the models.
- **The best part is that even when there is no training data, it will learn through the experience it has from processing the training data.**
- For various problems, which might seem complex to us, it provides the perfect models to tackle them.

# Disadvantages of Reinforcement Learning


- The usage of reinforcement learning models for solving simpler problems won't be correct. The reason being, the models generally tackle **complex** problems.
- We will be wasting unnecessary processing power and space by using it for simpler problems.
- We need lots of data to feed the model for computation. Reinforcement Learning models require a lot of training data to develop accurate results.
- This consumes time and lots of computational power.
- When it comes to building models on real-world examples, the **maintenance cost is very high.**
- Like for building driverless vehicles, robots, we would require a lot of maintenance for both hardware and software.
- **Excessive training can lead to overloading** of the states of the model. This will result in the model for getting the result.
- This may happen if too much memory space goes out in processing the training data.

# Adaptive Learning

**Next Generation of machine Intelligence**



# Adaptive learning


- No learning method is complete in itself.
- Need to select the learning method based on the requirements.
- Need to develop a combination of some of the existing methods based on requirements.
- Adaptive machine learning algorithms are the machine learning models, where **the changes in the environment** help in selecting the algorithm or learning method.

# Adaptive learning (Contd..)

- As per the scenario, most suitable algorithm is selected.
- Moreover the development of especially fast adapting algorithms poses many different issues like selection of choices, handling equilibrium states and so on.
- The adaptive learning solves some of thecomplex problems for which a single learning method is not enough.
- This method is even more appropriate when environment is continuously changing and the real response is expected. time

•

# What is a Multi-Agent System?

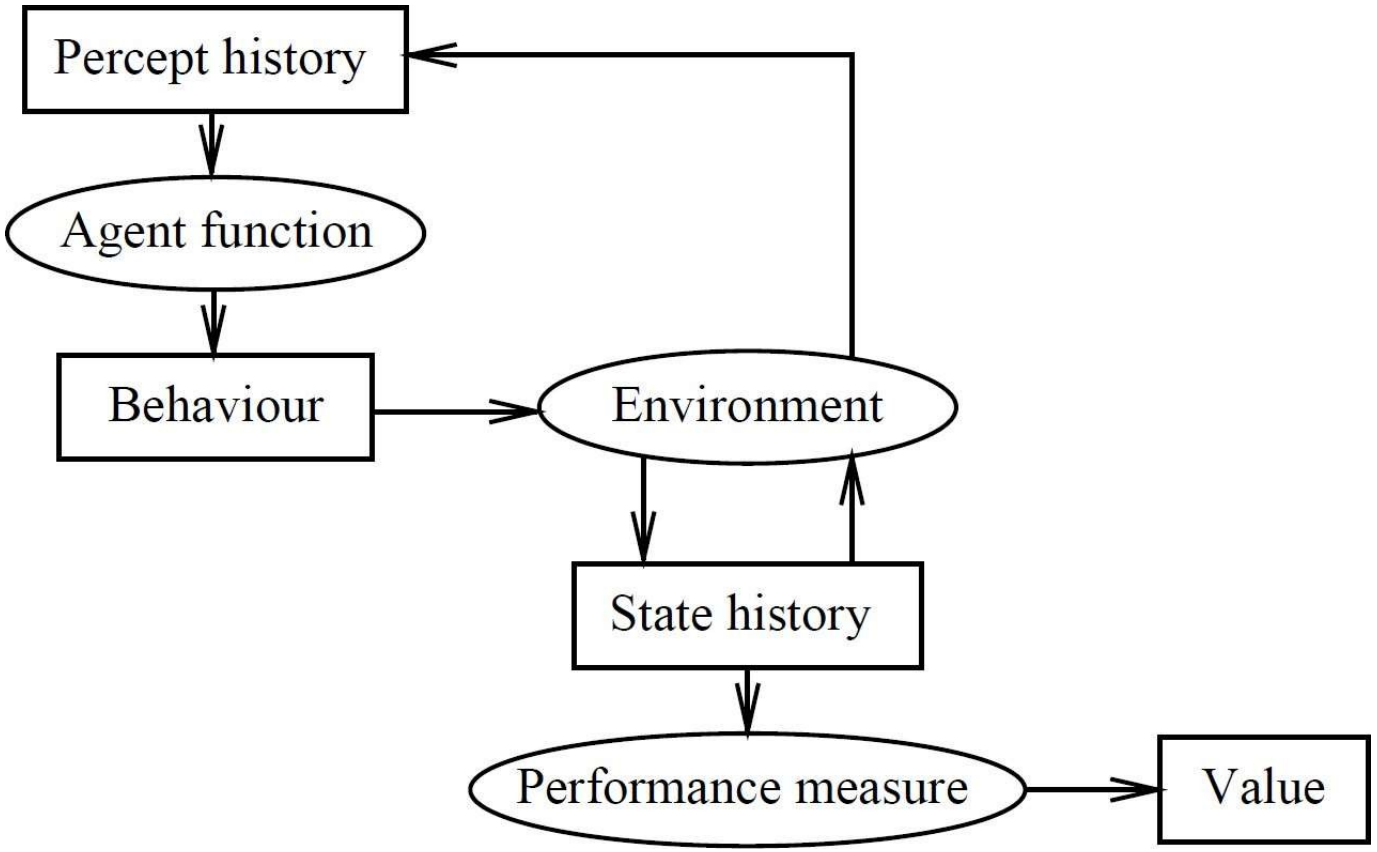

- A system with multiple autonomous entities, with distributed information, computational ability, and possibly divergent interests.

- Agents :: artificial or human, cooperative or self-interested

# Multiagent Systems, a Definition

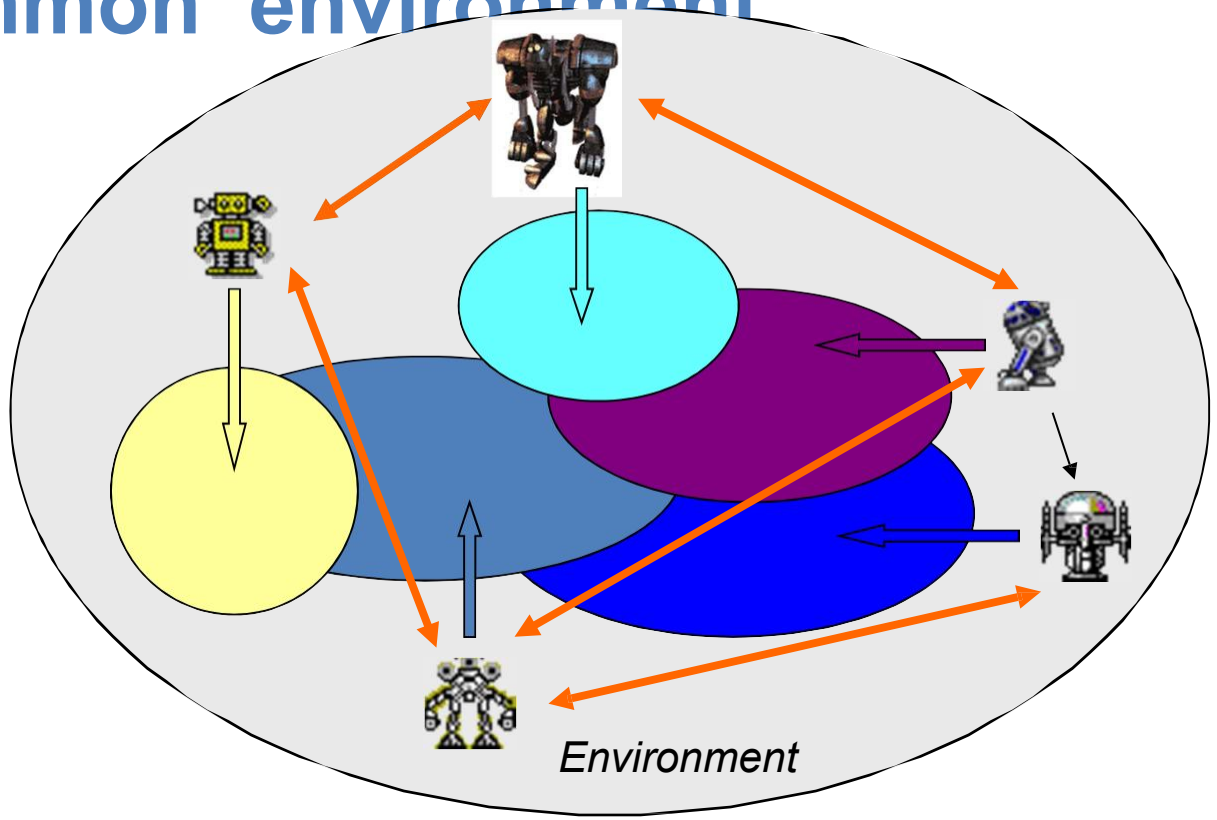

- A multiagent system is one that consists of a number of agents, which *interact* with one-another
- To *cooperate*, *coordinate*, and *negotiate* with each other

# Overview of an agent



# Multi-agent systems

## Many entities (agents) in a common environment




*Influenece area*

*Interactions*

# Learning in Multiagent Systems


- Intersection of DAI and ML

- Why bring them together?
  - There is a strong need to equip Multiagent systems with learning abilities
  - The extended view of ML as Multiagent learning is qualitatively different from traditional ML and can lead to novel ML techniques and algorithms

# Need for Multi agent Learning

- A single agent cannot handle learning in case of applications
- A team or group of agents possesses the potential to overcome the limitation of single agent and work in in coordination to accomplish a task.
- This can be two cases in multi agent based learning:

# Multi agent based learning


1) Where the agent tries to maximize its own utility

- Eg: Consider a manufacturing industry domain.

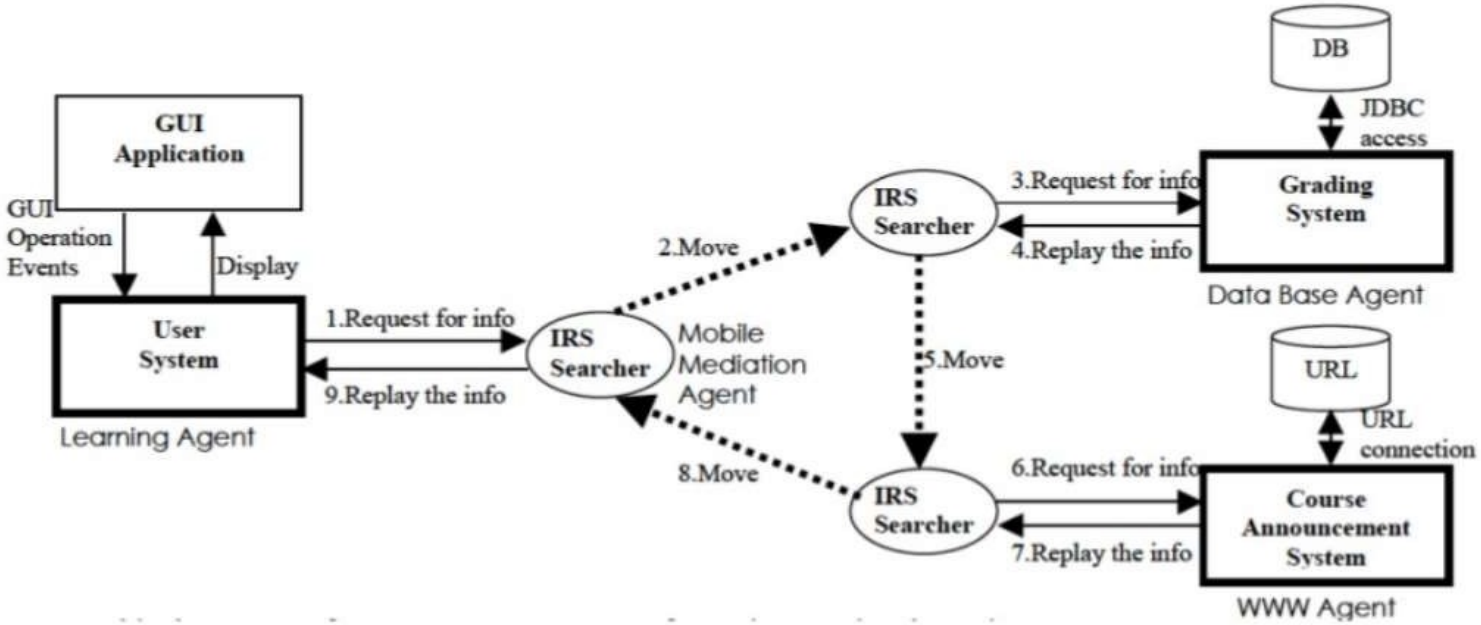
2) Where they worked in collaboration to achieve some common goals.

- Eg: Game playing. Can be related with the reinforcement learning, Where for each strategy of the agent some reward is achieved this is where each agent tries to maximize his own utility function.

# Multi_agent based learning Applications

## EDUCATION

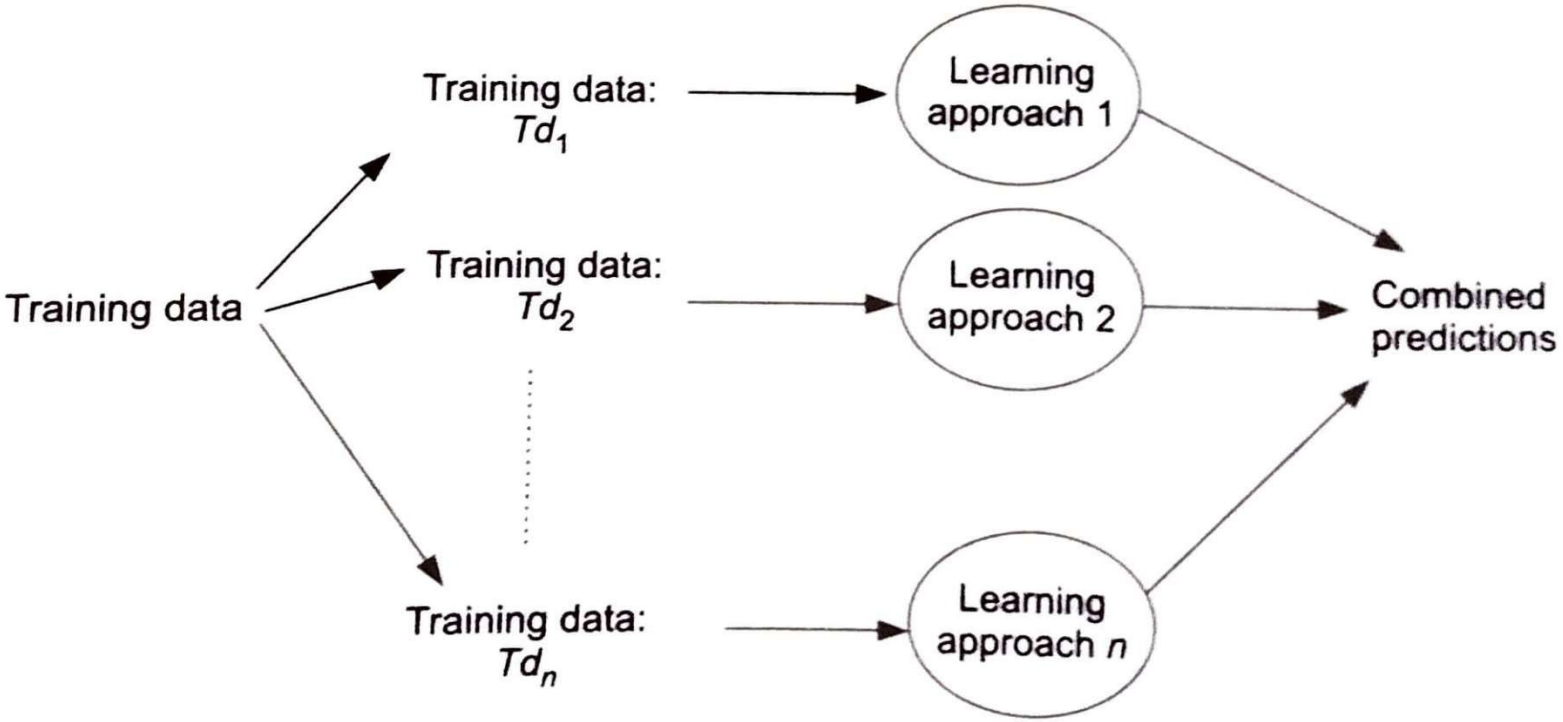
- A learning Multi-Agent system that mines the Web to advise students



# Advantages of a Multi-Agent Approach


- An MAS has the following advantages over a single agent or centralized approach:
- An MAS distributes computational resources and capabilities across a network of interconnected agents. Whereas a centralized system may be plagued by resource limitations, performance bottlenecks, or critical failures, an MAS is decentralized and thus does not suffer from the "single point of failure" problem associated with centralized systems.
- An MAS allows for the interconnection and interoperation of multiple existing legacy systems. By building an agent wrapper around such systems, they can be inporporated into an agent society.
- An MAS models problems in terms of autonomous interacting component-agents, which is proving to be a more natural way of representing task allocation, team planning, user preferences, open environments, and so on.
- An MAS efficiently retrieves, filters, and globally coordinates information from sources that are spatially distributed.
- An MAS provides solutions in situations where expertise is spatially and temporally distributed.
- An MAS enhances overall system performance, specifically along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse.

# Ensemble Learning




Ensemble method: The concept.

# Ensemble Learning (Contd…)


Ensemble learning method is the one where **multiple learners or learning algorithms are trained.**

- In most of the learning algorithms a single hypothesis drives the learning.
- In ensemble learning method the whole collection or ensemble of hypothesis is selected from the hypothesis space and their predictions are combined.
- In this approach, the learners or referred to as base learners.
- The most commonly used ensemble learning methods are

1) Boosting
2) Bagging.
3) Stacking

# Ensemble Learning (Contd…)


- Ensemble learning is an ML paradigm where numerous base models (which are often referred to as “weak learners”) are combined and trained to solve the same problem.
- This method is based on the theory that by correctly combining several base models together
- Together these ensemble models (aptly called “strong learners”) achieve better, more accurate results.
- There are three main ensemble techniques: bagging, boosting and stacking.

# Ensemble Learning (Contd..)


- **Bagging:** Bagging attempts to incorporate similar learners on small-sample populations and calculates the average of all the predictions. Generally, bagging allows you to use different learners in different populations. By doing so, this method helps to reduce the variance error.
- **Boosting:** Boosting is an iterative method that fine-tunes the weight of an observation according to the most recent classification. If an observation was incorrectly classified, this method will increase the weight of that observation in the next round (in which the next model will be trained) and will be less prone to misclassification. Similarly, if an observation was classified correctly, then it will reduce its weight for the next classifier. The weight represents how important the correct classification of the specific data point should be, as this enables the sequential classifiers to focus on examples previously misclassified. Generally, boosting reduces the bias error and forms strong predictive models, but at times they may overfit on the training data.
- **Stacking:** Stacking is a clever way of combining the information provided by different models. With this method, a learner of any sort can be used to combine different learners' outputs. The result can be a decrease in bias or variance determined by which combined models are used.

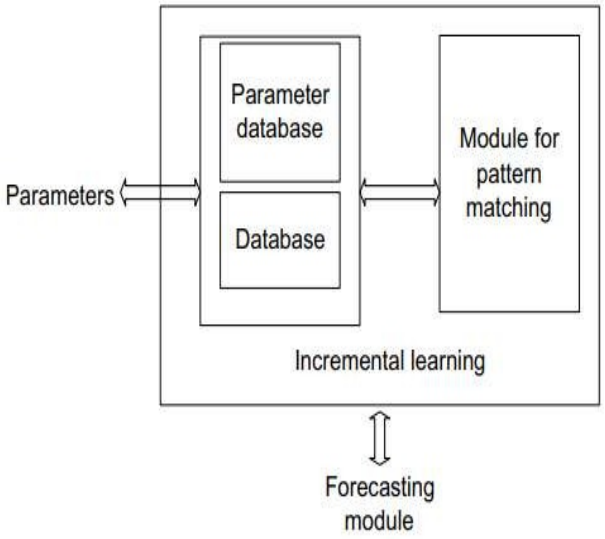
# Learning for decision making

AI systems for decision-making can be understood as lying along a spectrum according to their levels of autonomy.

- In some cases, human experts use AI techniques to support them in reasoning about a single, high-stakes decision.
- In other settings, it makes sense for an AI system to make autonomous decisions and
- In between there are 'mixed-initiative' systems that share

# Learning and Decision-Making Model

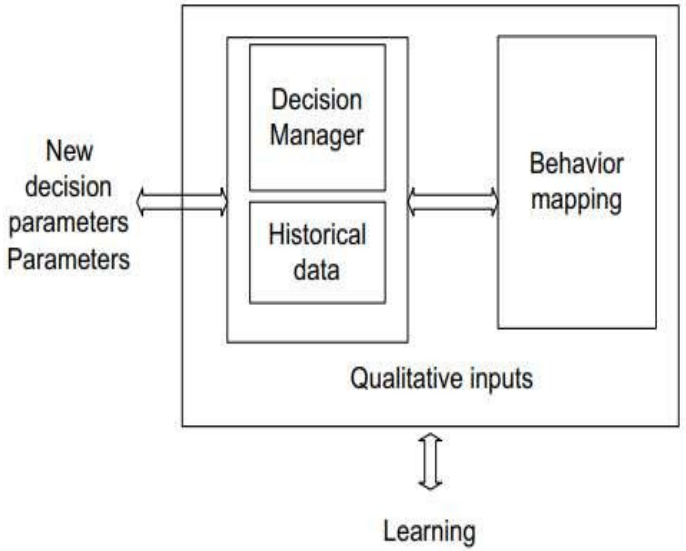
- The following figure gives the architecture of the new forecasting module.
- Its applications vary from health-care decision making to hospitality industry and revenue management. This forecasting tool will lie beneath the decision system.




Incremental learning and forecasting.

# Learning and Decision-Making Model

- Complete decision-system architecture for decision making based on incremental learning is depicted in figure below.
- The decision manager is responsible for decision making and works on historical data and behavior mapping. Qualitative inputs and incremental quantitative inputs facilitate decision making.




Incremental learning and decision making.

# Distributed learning

- In distributed learning the task of learning is distributed.
- **Need for distributed learning** - Arises due to large data sets and time constraints.
- More than one agent in different parts of the data set. There will be **distributed learning algorithms** taken part in each partition to get the desired outcome, which would then be combined.
- **Efficiency** of distributed learning is affected to look at. it is extremely important that outcome of distributed learning matches with the ones acheived under the absence of distributed environment.
- Multi agent systems canbe thought of as a **subset of distributed learning.**

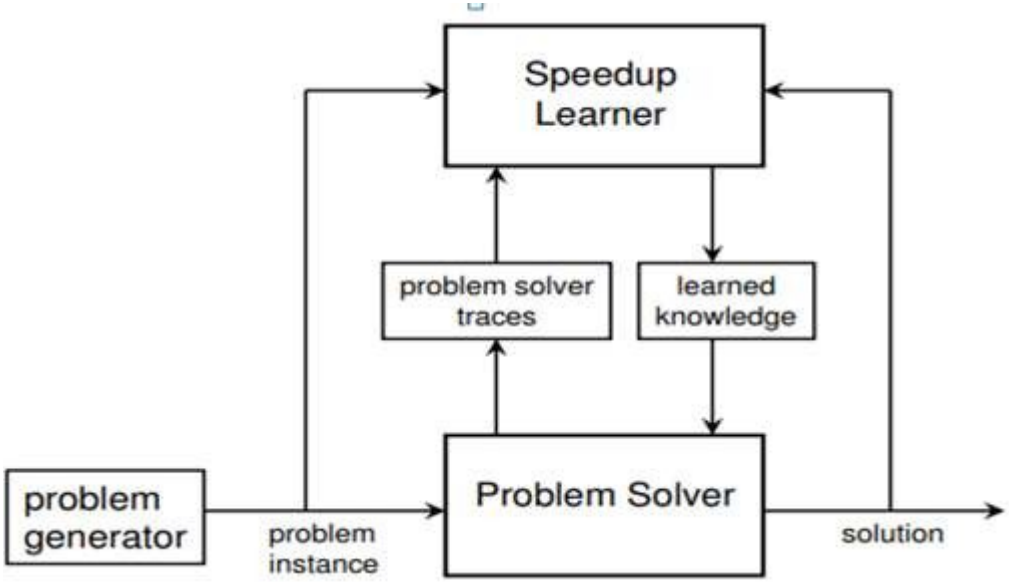
# Speedup learning

- Speed learning typically deals with speeding up problem solving by effective use of problem solving experience. Hence, problem solving experience is an input for speed of learning.
- In this learning,

1) There is no option with the environment.
2) New problems cannot be solved.

So,speed up learning accelerates the process experiences And prior observations.

# Speedup learning (Contd..)



# Speedup learning -Types

## Intra-Problem versus Inter-Problem Speedup

- Intra-problem speedup learning is when knowledge is learned during the solution of the current problem instance and is only applicable to speeding up the solution of the current instance. After a solution is found, the knowledge is discarded as it is not applicable to future instances.

- Inter-problem speedup learning is when the learned knowledge is applicable not only to the problem(s) it was learned on but also to new problems encountered in the future

# Speedup learning Examples


- Much of the speedup learning work arising from research in AI search and constraint satisfaction falls into the intra-problem paradigm.
- The most common forms of learning are deductive and are based on computing explanations of "search failures" that occur during the solution of a particular problem.
- Here a search failure typically corresponds to a point where the problem solver must backtrack.
- By computing and forming such failure explanations the problem solver is typically able to avoid similar types of failures in the future by detecting that a search path will lead to failure without fully exploring that path.


# Thank you