

# Module3

# Syllabus

1. Basics of Learning theory
2. Similarity Based Learning
3. Regression Analysis

# 1. Basics of Learning Theory

- 1. Introduction to learning and its types**
  1. What is Learning?
  2. Classical and Adaptive Machine Learning
  3. Learning Types
- 2. Introduction to Computation Learning theory**
- 3. Design of a Learning System**
  1. Choosing a Training Experience
  2. Choosing a Targett Function
  3. Representation of a target function
  4. Function approximation
- 4. Introduction to Concept Learning**
  1. What is Concept Learning?
  2. Representation of a Hypothesis
  3. Hypothesis space
  4. Heuristic Space Search
  5. Generalization and Specialization
  6. Hypothesis Space Search Find S Algorithm
  7. Version Space and Candidate Elimination algorithm

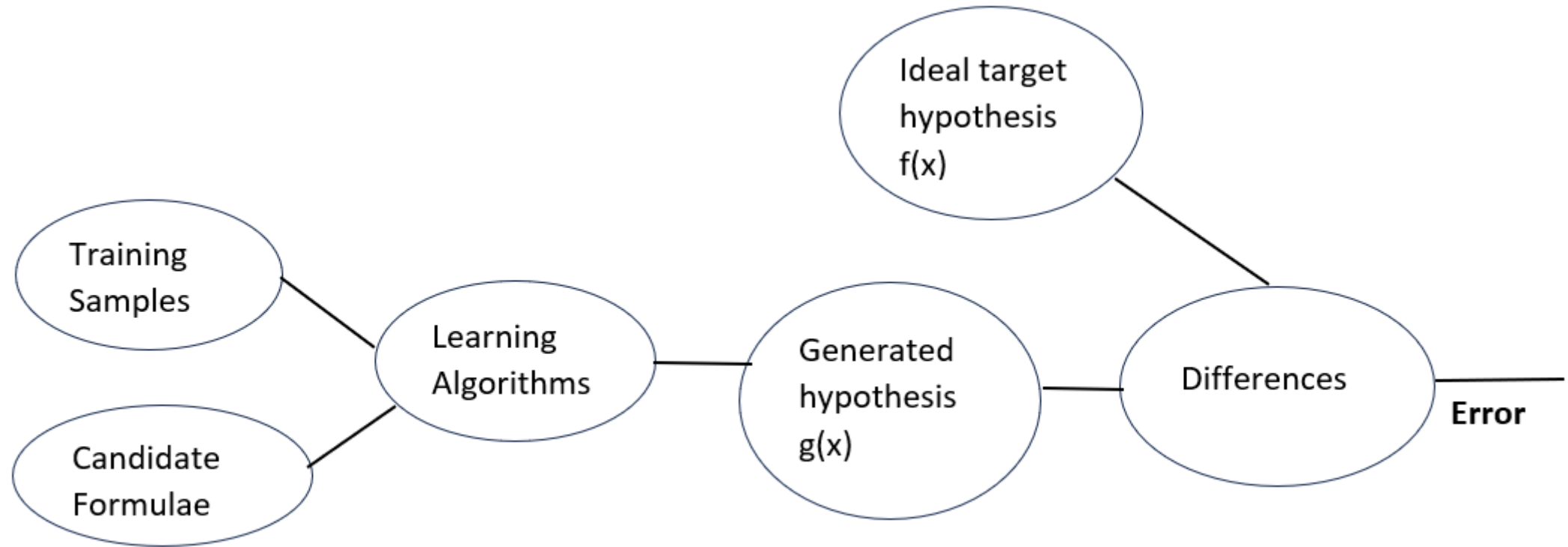
# 1. What is Learning?

- **Learning** is a process of acquiring knowledge and construct new ideas or concepts based on the experiences, study and training.
- ML is way of learning **general concept (hypothesis/formulae/pattern /insight )** from training examples without writing a program.
- There are two types of problems :
  1. **Well Posed problem**
  2. **Ill posed problem**
- Computers can solve well posed problems which have **well defined specification** and have the following components:
  1. **Class of Learning tasks (T)**
  2. **A measure of performance (P)**
  3. **A source of experience (E)**

# What is hypothesis?

- A hypothesis is a **statement or proposition that is formulated to explain a set of facts or phenomena.**
- It is a preliminary, **testable explanation** for observed phenomena, and it serves as the basis for further investigation and experimentation.
- In scientific research, hypotheses play a crucial role in the scientific method.
- In a research study, the hypothesis is often stated at the beginning and is tested through experimentation or data analysis. Depending on the results, the hypothesis may be supported, rejected, or modified.

# Learning Environment



Learning Model = Hypothesis Set + Learning algorithm

# Perception Learning algorithm

Let

$x$  be the input

$X$  is the input Space

$y$  be the out put (with class 0 or 1)

$Y$  bet the output space

$D$  be the input datasets

The simple Learning model can be given as

$h(x) = \text{sign}(\sum_{i=1}^D x_i w_i) + b > 0$  (Threshold) belongs to class 1  
and

$h(x) = \text{sign}(\sum_{i=1}^D x_i w_i) + b < 0$  (Threshold) belongs to  
Another Class

This simple mode is called **perception model**.

One can simplify this by making  $w_0 = b$  and fixing it as 1, then the model can further be simplified as:

$$h(x) = \text{sign}(w^T x)$$

# Learning Systems

## **Classical Learning system**

- Input,
- Process,
- Output

## **Adaptive Learning System**

- Reinforcement Learning
- Action
- Reward/Punishment
- Feedback

# Learning types

1. Learn by Memorization
2. Learn by examples (Experiences)
3. Learn by being taught (Passive or Active Learning)
4. Learning by critical thinking (Deductive Learning)
5. Self Learning (Reinforcement Learning/Self Directed learning)
6. Learning by solving Problems(Cognitive Learning)
7. Learning by generalizing explanations(Also called as explanation based learning)

# Questions that are basis for Computational Learning Theory(COLT)

1. How can learning System predict an unseen instance?
2. How do the hypothesis  $h$  is close to  $f$ , when hypothesis  $f$  itself is unknown?
3. How many samples are required?
4. Can we measure the performance of a learning system.
5. Is the solution obtained local or global.

## 2.Computational Learning Theory (COLT)

- **Deals** with formal method for learning systems
- **Deals** with frameworks for quantifying learning tasks and learning algorithms
- **Provides** fundamental basis for study of ML
- It uses many concepts from diverse areas such as Theoretical computer Science, AI and statistics.

# 3.Design of Learning system

A system that is built around a **learning algorithm** is called a learning system. The design of systems focuses on these **4 steps** :

## 1. Choosing a training experience

- Sample input and output
- Training Data Set/Supervisor

## 2. Choosing the target Function

- Type of knowledge required ( like legal moves /Move with largest score in chess)

### 3. Representation of target function

- Table/Collection or Rules or a neural network or a linear combination of features like  $V = w_0 + w_1x_1 + w_2x_2 + w_3x_3$

### 4. Choosing and Approximate Function

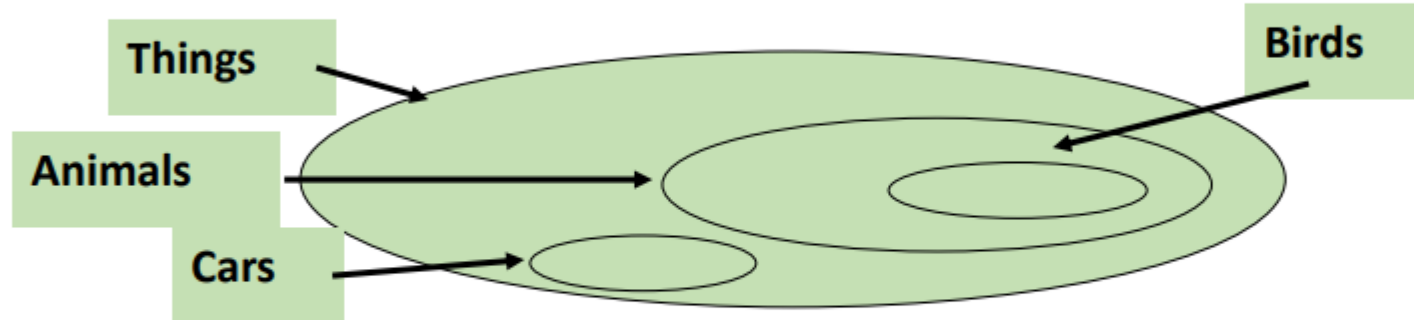
- The focus is to choose weights and fit the given training samples effectively. The aim is to reduce the error given as :
- $E = \sum [V_{\text{train}}(b) - V^{\wedge}(b)]^2$

# Components of Learning systems

1. **Performance** system to allow the game to play against itself
2. A **critic System** to generate the **samples**
3. A **generalizer system** to generate a **hypothesis** based on samples.
4. An **experimenter system** to generate a **new system** based on the currently learnt function.

# 4.0 What is Concept?

- A **Concept** is a subset of objects or events defined over a larger set [Example: The concept of a bird is the subset of all objects (i.e., the set of all things or all animals) that belong to the category of bird.]



- Alternatively, a concept is a boolean-valued function defined over this larger set [**Example:** a function defined over all animals whose value is true for birds and false for every other animal].

## 4.1 What is Concept learning ?

- Concept learning is a learning strategy of acquiring **abstract knowledge or inferring a general concept or deriving a category** from the given training samples.
- It is a process of **abstraction and generalization** from the data and helps to classify an object that has set of common and relevant features.
- It is a way of learning categories for object and to recognize new instances of those categories .

# Concept learning requires 3 things

1. **Input** : Labelled Training Data Set (Set of instances and concept /category)
2. **Output**: Target Concept of Target Function  $f(x)$  which maps from input  $x$  to output  $y$ .
3. **Test** – New instances to test the learning model

# Example: Sample Training Instances

Sl.NO	Horns	Tail	Tusks	Paws	Fur	Color	Hooves	Size	Elephant
1	No	Short	Yes	No	No	Black	No	Big	Yes
2	Yes	Short	No	No	No	Brown	Yes	Medium	No
3	No	Short	Yes	No	No	Black	No	Medium	Yes
4	No	Long	No	Yes	Yes	White	No	Medium	No
5	No	Short	Yes	Yes	Yes	Black	No	Big	Yes

**Independent attributes:** Horns, Tail, Tusks, Paws, Fur, Color, Hooves and Size

**Dependent attributes:** Elephant

**Target Concept** is to identify the animal is Elephant

## 4.2.Representation of hypothesis

- A hypothesis 'h' approximates a target function 'f'.
- Each hypothesis is represented as conjunction of attribute conditions  
Example : (Tail = Short)  $\wedge$  (Color = Black)----
- The set of hypothesis (h) is called hypotheses (H).
- Each attribute of hypothesis can take a value as either '?' or ' $\phi$ ' or can hold a single value
  - “?” denotes that the attribute can take any value [eg: Color = ?]
  - “ $\phi$ ” denotes that the attribute cannot take any value,i.e it represents a null value [eg: Horns =  $\phi$ ]
  - Single value denotes a specific single value from acceptable values of the attributes i.e. the attribute ‘Tail’ can take a value a short[ eg. Tail = short]

# Example

	Horns	Tail	Tusks	Paws	Fur	Color	Hooves	Size
<b>h =</b>	<b>&lt;No</b>	<b>?</b>	<b>Yes</b>	<b>?</b>	<b>?</b>	<b>Black</b>	<b>No</b>	<b>Medium</b>

The training data set given above has 5 training instances with 8 independent attributes and one dependent attribute. Here the different hypotheses that can be predicted for the target concept are:

	Horns	Tail	Tusks	Paws	Fur	Color	Hooves	Size
<b>h =</b>	<b>&lt;No</b>	<b>?</b>	<b>Yes</b>	<b>?</b>	<b>?</b>	<b>Black</b>	<b>No</b>	<b>Medium&gt;</b>

OR

<b>h=</b>	<b>&lt;No</b>	<b>?</b>	<b>Yes</b>	<b>?</b>	<b>?</b>	<b>Black</b>	<b>No</b>	<b>Big&gt;</b>
-----------	---------------	----------	------------	----------	----------	--------------	-----------	----------------

# Note : Most General and Most Specific Hypothesis

- $\langle ?, ?, ?, ?, ?, ?, ?, ? \rangle$  represents the **most general hypothesis** which **allows any value** to the attribute. Indicates **any animal** can be a elephant
- $\langle \Phi, \Phi, \Phi, \Phi, \Phi, \Phi, \Phi, \Phi \rangle$  represents the most specific hypothesis and will **not allow** any value for each of the attribute. This hypothesis indicates that **no animal can be an elephant**.

# Concept learning task of an elephant

Sl.NO	Horns	Tail	Tusks	Paws	Fur	Color	Hooves	Size	Elephant
1	No	Short	Yes	No	No	Black	No	Big	Yes
2	Yes	Short	No	No	No	Brown	Yes	Medium	No
3	No	Short	Yes	No	No	Black	No	Medium	Yes
4	No	Long	No	Yes	Yes	White	No	Medium	No
5	No	Short	Yes	Yes	Yes	Black	No	Big	Yes

**Input : 5 Instances with attributes**

**Target Concept/function 'c': Elephant --> {Yes, No}**

**Hypotheses H: Set of hypothesis each with conjunctions of literals as propositions.**

The hypothesis 'h' for the concept learning task of an Elephant is given as :  
**h = <No Short Yes ? ? Black No ?>**

This hypothesis h is expressed in **propositional logic** form as below:

**(Horns = No)  $\wedge$  (Tail = short)  $\wedge$  (Tusks = Yes) (Paws = ?)  $\wedge$  (Fur = ?)  $\wedge$  (Color = Black)  $\wedge$  (Hooves = No)  $\wedge$  (Size = ?)**

**Output : Learn hypothesis 'h' to predict an 'Elephant ' such that for a given test instance  $x$  ,  $h(x) = c(x)$**

**Note : Concept Learning can also be called as inductive learning that tries to induce a general function from specific training instances.**

## 4.2 Hypothesis Space

- Is the set of all hypothesis that approximates the target function  $f$ .
- The subset of hypothesis space that is consistent with all observed training instances is called as **Version Space**.
- Version Space represents the only hypotheses that are used for the **classification**.

## 4.2 Hypothesis Space

Sl.NO	Horns	Tail	Tusks	Paws	Fur	Color	Hooves	Size	Elephant
1	No	Short	Yes	No	No	Black	No	Big	Yes
2	Yes	Short	No	No	No	Brown	Yes	Medium	No
3	No	Short	Yes	No	No	Black	No	Medium	Yes
4	No	Long	No	Yes	Yes	White	No	Medium	No
5	No	Short	Yes	Yes	Yes	Black	No	Big	Yes

**Horns** – Yes, No (2)

**Tail** - Long, short (2)

**Tusks** – Yes, No (2)

**Paws** - Yes, No (2)

**Fur** - Yes, No (2)

**Color** - Brown, Black, White (3)

**Hooves** - Yes, No (2)

**Size** - Medium, Big (2)

Considering these values for each of the attribute there are

**$(2 \times 2 \times 2 \times 2 \times 3 \times 2 \times 2) = 384$**  distinct instances covering all the 5 instances in the training data set.

So we can generate  **$(4 \times 4 \times 4 \times 4 \times 5 \times 4 \times 4) = 81,920$**  distinct hypotheses when including two more values [ ?,  $\phi$ ]

# What is **heuristic** Space Search?

- Heuristic space search refers to a problem-solving approach in artificial intelligence and computer science where an algorithm explores a search space using **heuristic information to efficiently find a solution**. A search space is the set of all possible states or configurations that a system can be in, and the goal is to navigate through this space to reach a desirable state or solution.
- Finds a **Solution/hypothesis** to a problem using heuristic functions
- Example : A\* Search, Greedy Best FirstSearch, Hill climbing, genetic algorithms ,etc.

# Generalization and Specialization

1. Generalization – Specific to General learning
2. Specialization – General to specific Learning

# Find S Algorithm

**Input:** Positive instances in the training data set

**Output:** Hypothesis 'h'

**S1:** Initialize 'h' to the most specific hypothesis

$h = \langle \phi \ \phi \ \phi \ \phi \ \phi \ \phi \ \phi \ \phi \ \underline{\dots} \rangle$

**S2:** Generalize the initial hypothesis for the first positive instance

**S3:** For each subsequent instances:

**If** it is positive instance :

**Check** for each attribute value in the instance with the hypothesis 'h'

**If** the attribute value is the same as the hypothesis value, then do nothing,

**Else** if the attribute value is different than the hypothesis value, change it to '?' in 'h'

**Else if** it is a negative instance,

        Ignore it

# Example

<b>CGPA</b>	<b>Interactiveness</b>	<b>Practical Knowledge</b>	<b>Communication skills</b>	<b>Logical Thinking</b>	<b>Interest</b>	<b>Job Offer</b>
>=9	Yes	Excellent	Good	Fast	Yes	Yes
>=9	Yes	Good	Good	Fast	Yes	Yes
>=8	No	Good	Good	Fast	No	No
>=9	Yes	Good	Good	Slow	No	Yes

## Solution:

CGPA	Interactiveness	Practical Knowledge	Communication skills	Logical Thinking	Interest	Job Offer
>=9	Yes	Excellent	Good	Fast	Yes	Yes
>=9	Yes	Good	Good	Fast	Yes	Yes
>=8	No	Good	Good	Fast	No	No
>=9	Yes	Good	Good	Slow	No	Yes

### Step1:

Initialize 'h' to the most specific hypothesis. There are 6 attributes, so for each attribute we initially fill 'φ' in the initial hypothesis 'h'

h = < φ   φ   φ   φ   φ   φ >

### Step2:

Generalize the initial hypothesis for the first positive instance. I1 is a positive instance so generalize the most specific hypothesis h to include this positive instance. Hence,

h = < φ   φ   φ   φ   φ   φ >

I1:   >=9   Yes   Excellent   Good   Fast   Yes   **Yes (Positive Instance)**

h1=   >=9   Yes   Excellent   Good   Fast   Yes

CGPA	Interactiveness	Practical Knowledge	Communication skills	Logical Thinking	Interest	Job Offer
>=9	Yes	Excellent	Good	Fast	Yes	Yes
>=9	Yes	Good	Good	Fast	Yes	Yes
>=8	No	Good	Good	Fast	No	No
>=9	Yes	Good	Good	Slow	No	Yes

## Step3:

Scan the next instance I2 since I2 is positive instance. Generalize h to include positive instance I2. For each of the non matching attribute value in 'h' put a '?' to include this positive instance. The third attribute value is mismatching in h with I2 so put a ?.

h1	>=9	Yes	Excellent	Good	Fast	Yes
I2	>=9	Yes	Good	Good	Fast	Yes (Positive Instance)
h2	>=9	Yes	?	Good	Fast	Yes

**Now scan I3** . Since it is a negative instance ignore it. Hence the hypothesis remains the same without any change after scanning I3

h2	>=9	Yes	?	Good	Fast	Yes
I3	>=8	No	Good	Good	Fast	No (Negative Instance)
h3	>=9	Yes	?	Good	Fast	Yes

CGPA	Interactiveness	Practical Knowledge	Communication skills	Logical Thinking	Interest	Job Offer
>=9	Yes	Excellent	Good	Fast	Yes	Yes
>=9	Yes	Good	Good	Fast	Yes	Yes
>=8	No	Good	Good	Fast	No	No
>=9	Yes	Good	Good	Slow	No	Yes

Now scan I4 since it is positive instance, check for mismatch in the hypothesis h with I4. The 5th and 6th attribute value are mismatching, so add? To those attribute in 'h'.

**h3      >=9    Yes    ?      Good   Fast    Yes**

**I4:      >=9    Yes    Good   Good   Slow   No      Yes (Positive Instance)**

**h4      >=9    Yes    ?      Good   ?      ?**

Thus the final hypothesis generated with find S Algorithm is :

**h =      ( >=9    Yes    ?      Good   ?      ?)**

It includes all positive instances and obviously ignores any negative instance.

# Limitations of Find S Algorithm

1. Algorithm is consistent with positive instances and ignores negative instances.
2. Algorithm Finds only one unique hypothesis, wherein there may be many other hypotheses that are consistent with the training dataset.
3. Erroneous data set can mislead the algorithm in determining the consistent hypothesis since it ignores negative instances.

# Version Space

- The version space is the set of all hypotheses that are consistent with the observed training examples/training data sets.

A hypothesis  $h$  is **consistent** with a set of training examples  $D$  of target concept  $c$  if and only if  $h(x)=c(x)$  for each training example in  $D$ .

$$\textit{Consistent}(h, D) \equiv (\forall \langle x, c(x) \rangle \in D) h(x) = c(x)$$

The **version space**,  $VS_{H,D}$ , with respect to hypothesis space  $H$  and training examples  $D$ , is the subset of hypotheses from  $H$  consistent with all training examples in  $D$ .

$$VS_{H,D} \equiv \{h \in H \mid \textit{Consistent}(h, D)\}$$

# List then Eliminate Algorithm

**Input:** Version Space – a list of all hypothesis

**Output:** Set of consistent hypotheses

1. Initialize the version space with a list of hypotheses
2. For each training instance
  - a. Remove from version space any hypothesis that is inconsistent