Machine Learning

- Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.
- Machine learning (ML) that focuses on developing systems that learn—or improve performance—based on the data they ingest.
- Artificial intelligence is a broad word that refers to systems or machines that resemble human intelligence.

What is Machine Learning?

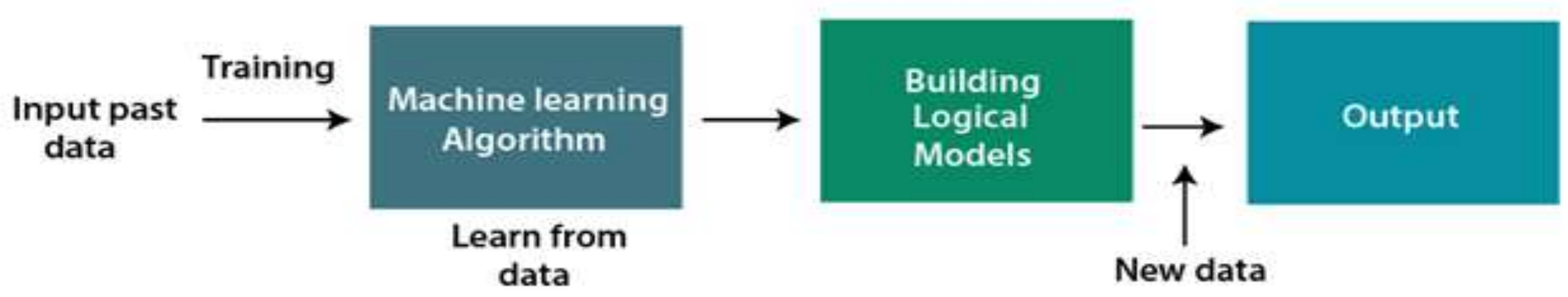
- Machine Learning is the field of study that gives computers the capability to learn **without being explicitly programmed**.
- It is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own.
- The term machine learning was first introduced by **Arthur Samuel** in **1959**.

- With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed.
- Machine learning brings computer science and statistics together for creating predictive models.
- Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.

A machine has the ability to learn if it can improve its performance by gaining more data.

How does Machine Learning work

- A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it**.
- The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.



Features of Machine learning

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

Need for Machine Learning

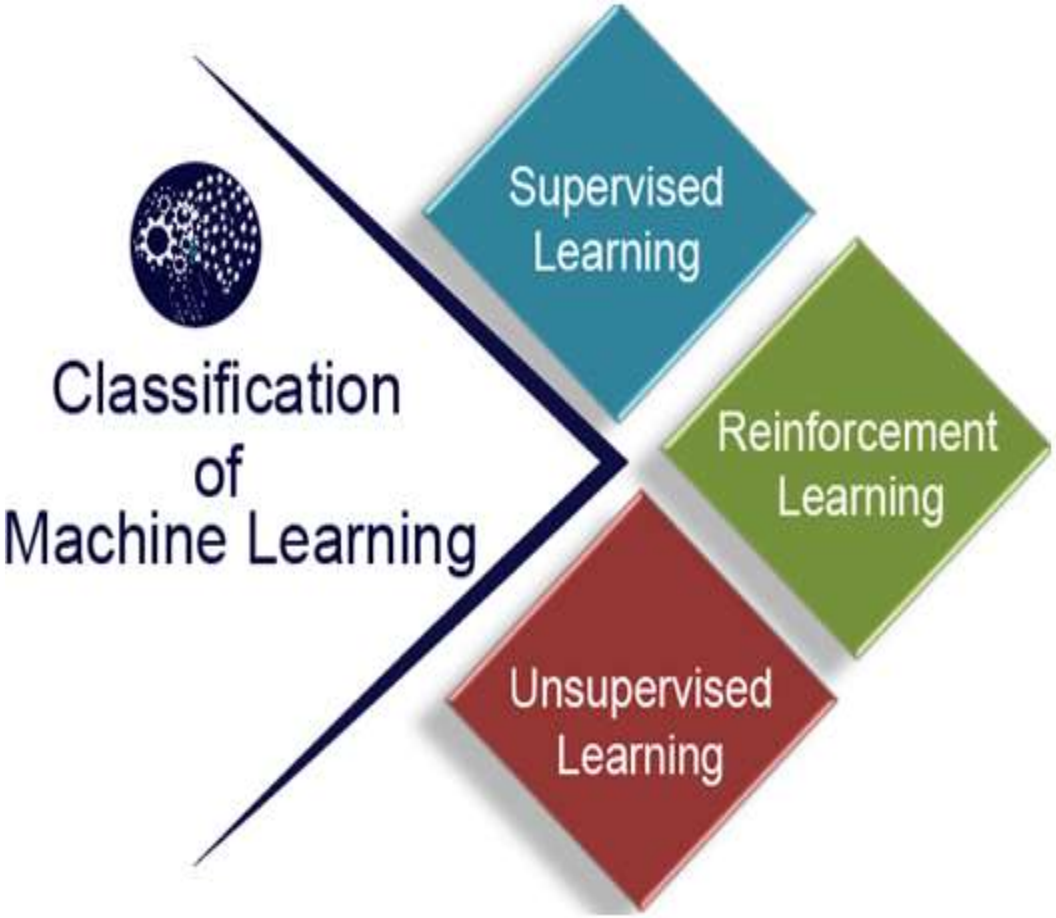
- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance.
- Finding hidden patterns and extracting useful information from data.

Currently, machine learning is used in **self-driving cars**, **cyber fraud detection**, **face recognition**, and **friend suggestion by Facebook**, etc.

Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. **Supervised learning**
2. **Unsupervised learning**
3. **Reinforcement learning**



Supervised Learning

- Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.
- The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.
- The goal of supervised learning is to map input data with the output data.

- The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.
- Supervised learning can be grouped further in two categories of algorithms:
- **Classification**
- **Regression**

Unsupervised Learning

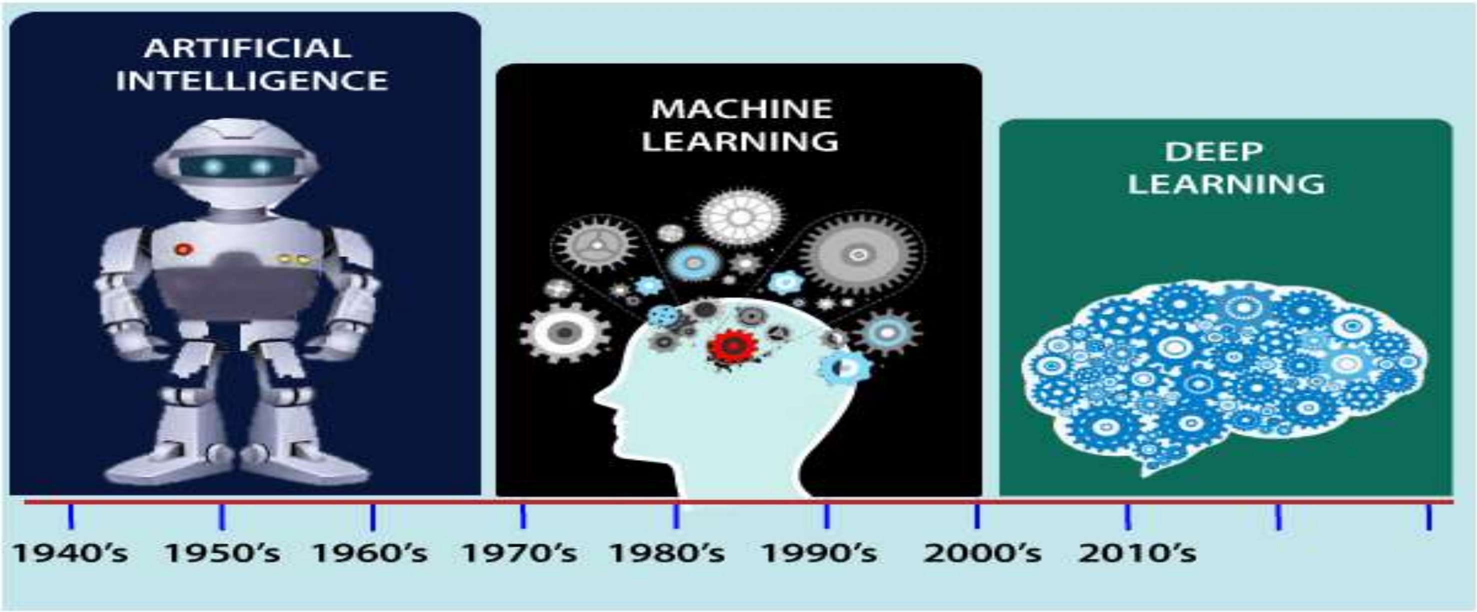
- Unsupervised learning is a learning method in which a machine learns without any supervision.
- The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision.
- The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

- In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classifieds into two categories of algorithms:
- **Clustering**
- **Association**

Reinforcement Learning

- Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
- The agent learns automatically with these feedbacks and improves its performance.
- In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.
- **The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.**

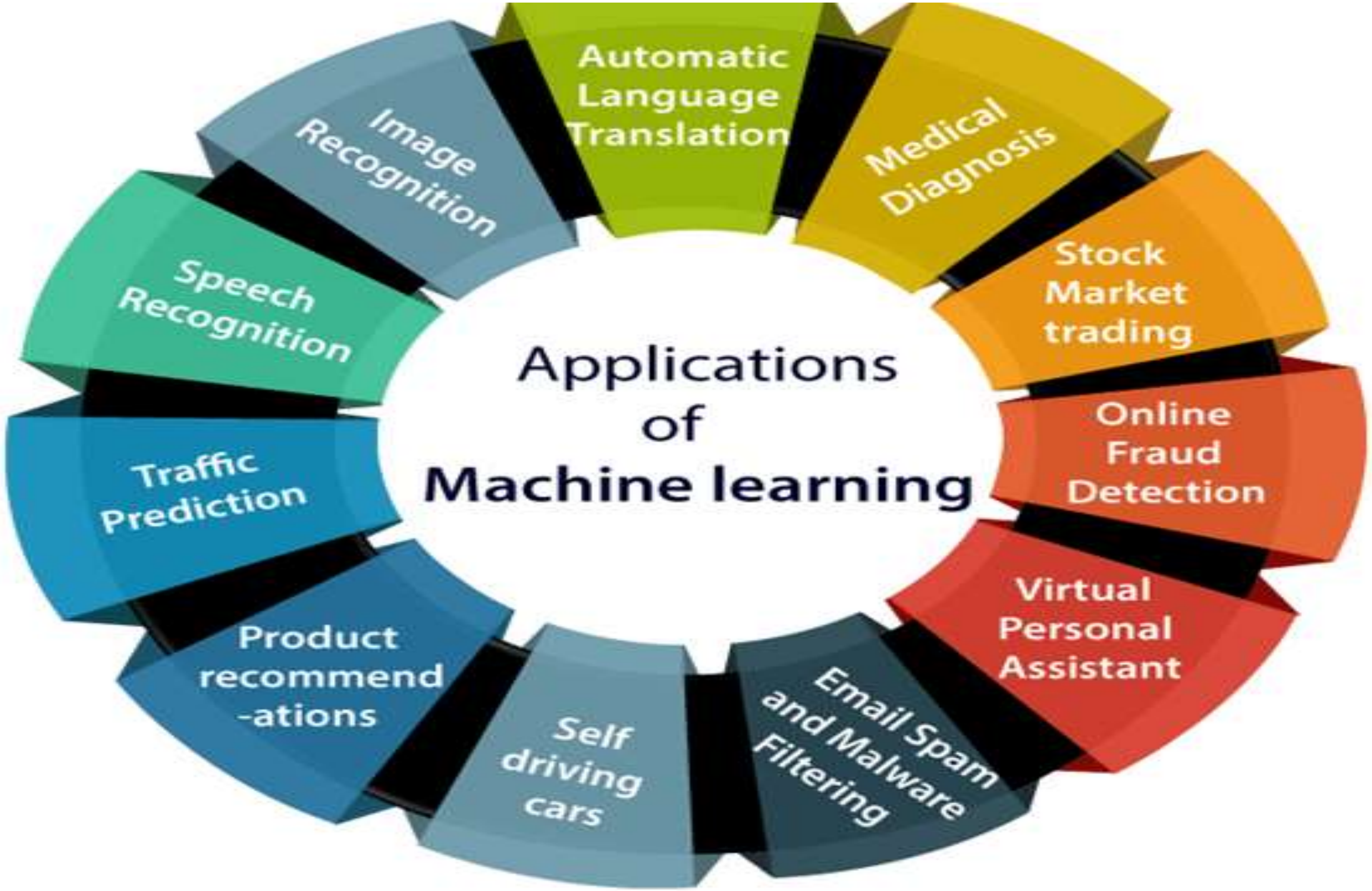
History of Machine Learning



Now machine learning has got a great advancement as **self-driving cars**, **Amazon Alexa**, **Catboats**, **recommender system**, **weather prediction**, **disease prediction**, **stock market analysis**, etc.

It includes **Supervised**, **unsupervised**, and **reinforcement learning with clustering**, **classification**, **decision tree**, **SVM algorithms**, etc.

Applications of Machine learning



Well Posed Learning Problem (Learning by example)

- A computer program is said to learn from experience E in context to some task T and some performance measure P, if its performance on T, as was measured by P, upgrades with experience E.
- Any problem can be segregated as well-posed learning problem if it has three traits –

➢Task

➢Performance Measure

➢Experience

- **Certain examples that efficiently defines the** well-posed **learning problem are –**

1. To better filter emails as spam or not

- Task – Classifying emails as spam or not
- Performance Measure – The fraction of emails accurately classified as spam or not spam
- Experience – Observing you label emails as spam or not spam

2. A checkers learning problem

- Task – Playing checkers game
- Performance Measure – percent of games won against opposer
- Experience **–** playing implementation games against itself

3. Fruit Prediction Problem

- Task – forecasting different fruits for recognition
- Performance Measure – able to predict maximum variety of fruits
- Experience – training machine with the largest datasets of fruits images

4. Face Recognition Problem

- Task – predicting different types of faces
- Performance Measure – able to predict maximum types of faces
- Experience – training machine with maximum amount of datasets of different face images

Perspective of ML

- It involves searching very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by the learner.
- **Hypotheses-** an assumption, an idea that is proposed for the sake of argument so that it can be tested to see if it might be true.

Issues/Challenges in ML

1. Poor Quality of Data- Unclean and noisy data can make the whole process extremely exhausting. The quality of data is essential to enhance the output.

2. Underfitting of Training Data- This process occurs when data is unable to establish an accurate relationship between input and output variables.

3. **Overfitting of Training Data-** Overfitting refers to a machine learning model trained with a massive amount of data that negatively affect its performance. This means that the algorithm is trained with noisy and biased data, which will affect its overall performance.

4. How much training data is sufficient?

5. How much testing data is required?

6. What algorithms should be used?

7. Which algorithm performs best for which type of problems.

8. What kind of methods to be used?

9. What methods should be used to reduce learning overhead.

10. For which type of data which methods should be used?

Designing a Learning System

- A computer program is said to be learning from experience (E), with respect to some task (T). Thus, the performance measure (P) is the performance at task T, which is measured by P, and it improves with experience E.”
- To get a successful learning system, it should be designed.
- For a proper design, several steps should be followed.

- **Steps for Designing Learning System are:**

Step 1. Choosing the Training Experience:

- The very important and first task is to choose the training data or training experience which will be fed to the Machine Learning Algorithm.
- It is important to note that the data or experience that we fed to the algorithm must have a significant impact on the Success or Failure of the Model. So Training data or experience should be chosen wisely.

Below are the attributes which will impact on Success and Failure of Data:

i. **The training experience will be able to provide direct or indirect feedback regarding choices.** For example: While Playing chess the training data will provide feedback to itself like instead of this move if this is chosen the chances of success increases.

ii. **Second important attribute is the degree to which the learner will control the sequences of training examples.** For example: when training data is fed to the machine then at that time accuracy is very less but when it gains experience while playing again and again with itself or opponent the machine algorithm will get feedback and control the chess game accordingly.

iii. **Third important attribute is how it will represent the distribution of examples over which performance will be measured.** For example, a Machine learning algorithm will get experience while going through a number of different cases and different examples. Thus, Machine Learning Algorithm will get more and more experience by passing through more and more examples and hence its performance will increase.

Step 2- Choosing target function: The next important step is choosing the target function. It means according to the knowledge fed to the algorithm the machine learning will choose NextMove function which will describe what type of legal moves should be taken.

- For example : While playing chess with the opponent, when opponent will play then the machine learning algorithm will decide what be the number of possible legal moves taken in order to get success.

Step 3- Choosing Representation for Target function:

- When the machine algorithm will know all the possible legal moves the next step is to choose the optimized move using any representation i.e. using linear Equations, Hierarchical Graph Representation, Tabular form etc.
- The NextMove function will move the Target move like out of these move which will provide more success rate.
- For Example : while playing chess machine have 4 possible moves, so the machine will choose that optimized move which will provide success to it.

Step 4- Choosing Function Approximation Algorithm:

- An optimized move cannot be chosen just with the training data.
- The training data had to go through with set of example and through these examples the training data will approximates which steps are chosen and after that machine will provide feedback on it.
- For Example : When a training data of Playing chess is fed to algorithm so at that time it is not machine algorithm will fail or get success and again from that failure or success it will measure while next move what step should be chosen and what is its success rate.

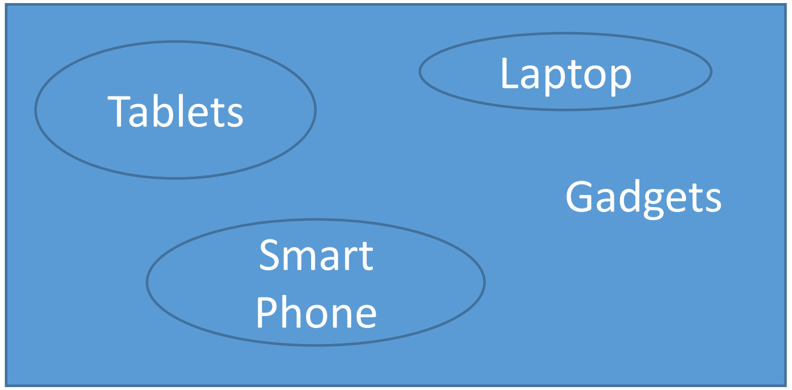
Step 5- Final Design:

- The final design is created at last when system goes from number of examples , failures and success , correct and incorrect decision and what will be the next step etc.
- Example: DeepBlue is an intelligent computer which is ML-based won chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

Concept Learning

- Concept Learning in Machine Learning can be thought of as a Boolean-valued function defined over a large set of training data.
- This simply defines: finding all the consistent hypothesis.
- Example:
- We learn about Gadgets (tablets and

smart phones). For this, we first need to understand the features of both the gadgets.



	Tablet	Smart Phone
X1: Size	Large	Small
X2: Colour	Black	Blue
X3: Screen Type	Flat	Folded
X4: Shape	Square	Rectangle

- Concept: <x1,x2,x3,x4>

Tablet: <Large, Black, Flat, Square>

Smart Phone: <Small, Blue, Folded, Rectangle>

Number of possible instances= 2^d   where, d= no. of features

Total possible concepts= 2^No. of possible instances.

- As we learn about tablets and smart phones we can apply the learning to the entire domain (concept).
- Therefore, we need to take the features which are consistent. If the features are not consistent than the hypothesis always varies and we can't get into some conclusion out of it.
- Main Goal→ is to find the all hypothesis that are consistent.
- In this term we use two types of hypothesis:

1. Most specific hypothesis: Reject All

   Denoted by ϕ

2. Most general hypothesis: Accept All

   Denoted by ?

Concept Learning as Search

- Main Goal→ is to find out the hypothesis that best fits the training example.
- Example: Find out the day in which sports can be enjoyed

Lets assume 6 attributes based on which this can be decided:

(sky, air temp, humidity, wind, water, forecast)

Now, sky: (sunny, rainy, cloudy) and remaining attributes are having only 2 values.

- Calculate **different instances possible**:

$$3*2*2*2*2*2 = 96$$

- Now, find out the **syntactically distinct hypothesis**. For this we need to add most general and specific hypothesis for all the attributes. i.e. add ϕ and ?

Therefore,

- Syntactically Distinct Hypothesis= 5*4*4*4*4*4 = 5120
- Now, find out the **semantically distinct hypothesis. i.e. null (ϕ) taken as common.**

= 1+ (4*3*3*3*3*3) = 973

- After finding all the possible hypothesis instances possible, we search the best match i.e. hypothesis that is much closes to our learning problem.

Find S Algorithm

- It is a basic concept learning algorithm in machine learning.
- Finding a maximally specific hypothesis that fits all the positive examples. (ϕ)
- This algorithm considers only positive values.
- The find-S algorithm starts with the most specific hypothesis and generalizes this hypothesis each time it fails to classify an observed positive training data.
- **?** indicates that any value is acceptable for the attribute.
- ϕindicates that no value is acceptable.

Most specific hypothesis: Denoted by ϕ

Most general hypothesis: Denoted by ?

- **Steps to be followed:**

1. Start with the most specific hypothesis. **h = {ϕ, ϕ, ϕ, ϕ, ϕ, ϕ}**
2. Take the next example and if it is negative, then no changes occur to the hypothesis.
3. If the example is positive and we find that our initial hypothesis is too specific then we update our current hypothesis to a general condition.
4. Keep repeating the above steps till all the training examples are complete.
5. After we have completed all the training examples we will have the final hypothesis when can use to classify the new examples.

Algorithm:

Step 1: Initialize h to the most specific hypothesis in H

Step 2: For each positive training instance x,

For each attribute constraint a, in h

If the constraint a, is satisfied by x

Then do nothing

Else replace a, in h by the next more general constraint that is satisfied by x

Step 3: Output the hypothesis h

- Example:

S.No	Origin	Manufacturer	Color	Year	Type	Class
1	Japan	HU	Blue	1980	Eco	Yes (+ve)
2	Japan	TO	Green	1970	Sports	No (-ve)
3	Japan	TO	Blue	1990	Eco	Yes
4	USA	AU	Red	1980	Eco	No
5	Japan	HU	White	1980	Eco	Yes
6	Japan	TO	Green	1980	Eco	Yes
7	Japan	HU	Red	1980	Eco	No

- h0= <ϕ, ϕ, ϕ, ϕ, ϕ> because: we are having 5 attributes
- h1= <‘JP’, ‘Hu’, ‘blue’, 1980, ‘eco’>
- h2= ignore because its class is negative
- h3= <‘JP’, ?, ‘blue’, ?, ‘eco’> // compare value from previous hypothesis
- h4= ignore because its class is negative
- h5= <‘JP’, ?, ?, ?, ‘eco’>
- **h6= <‘JP’, ?, ?, ?, ‘eco’> // Most general hypothesis**
- h7= ignore because its class is negative

- **Disadvantage**: It considers only +ve values.
- It may not be sole hypothesis that fits the complete data.

- Example

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

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

h0 = **<Ø, Ø, Ø, Ø, Ø, Ø>**

Solution:

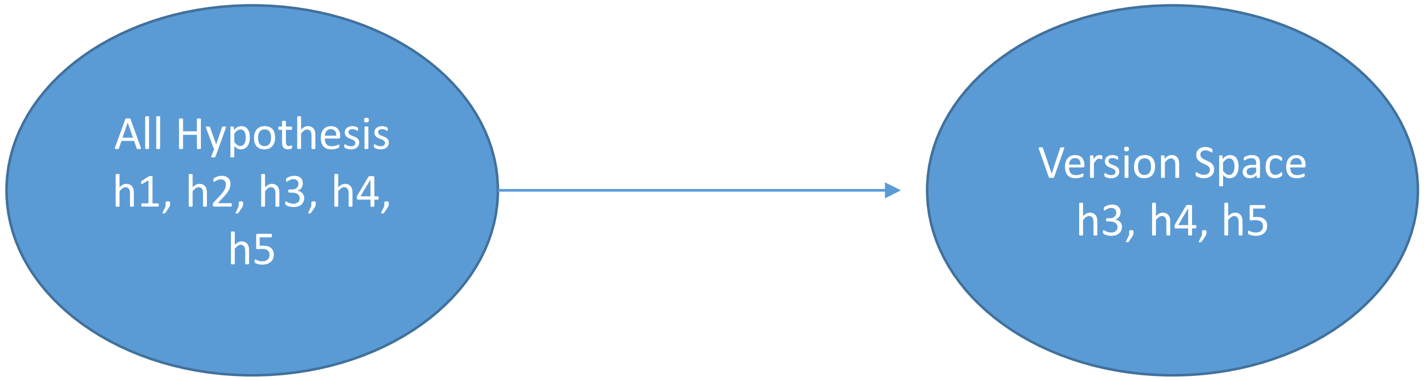
- h1 = <Sunny, Warm, Normal, Strong, Warm, Same>
- h2 = <Sunny, Warm, ?, Strong, Warm, Same>
- h3 is Negative example Hence ignored
- h4 = <Sunny, Warm, ?, Strong, ?, ?>
- **The final maximally specific hypothesis is**
- **<Sunny, Warm, ?, Strong, ?, ?>**

Version Space

- Subset of hypothesis H consistent with the training examples.
- VS(H,D)= {h subset of H| consistent (h,D)}
- Where, H- hypothesis, D= training example
- We need to verify:
- h(x)=c(x) i.e Hypothesis must derive target function.

- Algorithm:

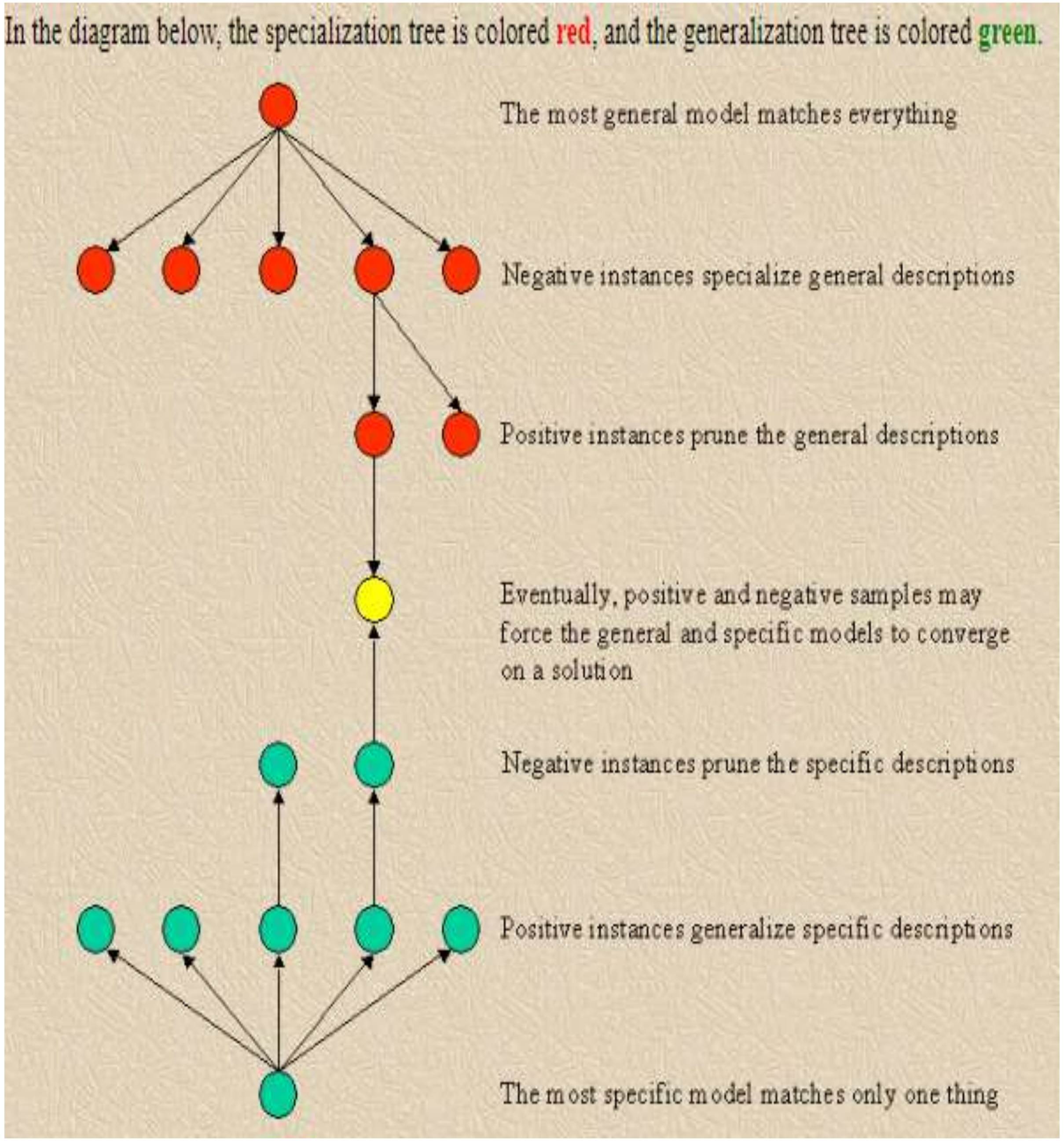
1. List containing every hypothesis in H
2. From this step, we remove inconsistent hypothesis from version space.
3. For each training example, if h(x)=c(x). Then remove that hypothesis.
4. Finally output the list of hypothesis into version space after checking for all the training examples.



Candidate Elimination Algorithm

- The candidate elimination algorithm incrementally builds the version space given a hypothesis space H and a set E of examples.
- The examples are added one by one; each example possibly shrinks the version space by removing the hypotheses that are inconsistent with the example.
- The candidate elimination algorithm does this by updating the general and specific boundary for each new example.

- This can be consider this as an extended form of the Find-S algorithm.
- Consider both positive and negative examples.
- Actually, positive examples are used here as the Find-S algorithm (Basically they are generalizing from the specification).
- While the negative example is specified in the generalizing form.



- **Terms Used:**
- **Concept learning:** Concept learning is basically the learning task of the machine (Learn by Train data)
- **General Hypothesis:** Not Specifying features to learn the machine.
- **G = {'?', '?','?','?'…}:** Number of attributes
- **Specific Hypothesis:** Specifying features to learn machine (Specific feature)
- **S= {ϕ,ϕ, ϕ,…ϕ}: The number** of pi depends on a number of attributes.
- **Version Space:** It is an intermediate of general hypothesis and Specific hypothesis. It not only just writes one hypothesis but a set of all possible hypotheses based on training data-set.

Algorithm:

```
Step1: Load Data set
Step2: Initialize General Hypothesis  and Specific  Hypothesis.
Step3: For each training example
Step4: If example is positive example
          if attribute_value == hypothesis_value:
             Do nothing
          else:
             replace attribute value with '?' (Basically
generalizing it)
Step5: If example is Negative example
          Make generalize hypothesis more specific.
```

- Example:

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Algorithmic steps:

```
Initially : G = [[?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?,
?, ?, ?],
                 [?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?,
?, ?, ?]]
            S = [Null, Null, Null, Null, Null, Null]
```

```
For instance 1 : <'sunny','warm','normal','strong','warm ','same'>
and positive output.
            G1 = G
            S1 = ['sunny','warm','normal','strong','warm ','same']

For instance 2 : <'sunny','warm','high','strong','warm ','same'> and
positive output.
            G2 = G
            S2 = ['sunny','warm',?,'strong','warm ','same']
```

```
For instance 3 : <'rainy','cold','high','strong','warm ','change'>
and negative output.
            G3 = [['sunny', ?, ?, ?, ?, ?], [?, 'warm', ?, ?, ?, ?],
[?, ?, ?, ?, ?, ?],
                  [?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?,
?, ?, 'same']]
            S3 = S2

For instance 4 : <'sunny','warm','high','strong','cool','change'>
and positive output.
            G4 = G3
            S4 = ['sunny','warm',?,'strong', ?, ?]
```

Output :

```
G = [['sunny', ?, ?, ?, ?, ?], [?, 'warm', ?, ?, ?, ?]]
S = ['sunny','warm',?,'strong', ?, ?]
```

The Candidate Elimination Algorithm (CEA) is an improvement over the Find-S

Advantages of CEA over Find-S:

1. Improved accuracy: CEA considers both positive and negative examples to generate the hypothesis, which can result in higher accuracy when dealing with noisy or incomplete data.
2. Flexibility: CEA can handle more complex classification tasks, such as those with multiple classes or non-linear decision boundaries.
3. More efficient: CEA reduces the number of hypotheses by generating a set of general hypotheses and then eliminating them one by one. This can result in faster processing and improved efficiency.
4. Better handling of continuous attributes: CEA can handle continuous attributes by creating boundaries for each attribute, which makes it more suitable for a wider range of datasets.

Disadvantages of CEA in comparison with Find-S:

1. More complex: CEA is a more complex algorithm than Find-S, which may make it more difficult for beginners or those without a strong background in machine learning to use and understand.
2. Higher memory requirements: CEA requires more memory to store the set of hypotheses and boundaries, which may make it less suitable for memory-constrained environments.
3. Slower processing for large datasets: CEA may become slower for larger datasets due to the increased number of hypotheses generated.
4. Higher potential for overfitting: The increased complexity of CEA may make it more prone to overfitting on the training data, especially if the dataset is small or has a high degree of noise.

Inductive Bias

- The phrase "inductive bias" refers to a collection of (explicit or implicit) assumptions made by a learning algorithm in order to conduct induction, or generalize a limited set of observations (training data) into a general model of the domain.
- From Candidate-Elimination Algorithm, we get two hypotheses, one specific and one general at the end as a final solution.
- **Now, we need to check if the hypothesis we got from the algorithm is actually correct or not, also make decisions like what training examples should the machine learn next.**

The fundamental questions for inductive reference:

- What happens if the target concept isn’t in the hypothesis space?
- Is it possible to avoid this problem by adopting a hypothesis space that contains all potential hypotheses?
- What effect does the size of the hypothesis space have on the algorithm’s capacity to generalize to unseen instances?
- What effect does the size of the hypothesis space have on the number of training instances required?

- **Inductive Learning:**
- This basically means learning from examples.
- We are given input samples (x) and output samples (f(x)) in the context of inductive learning, and the objective is to estimate the function (f). i.e from examples rules are derived.
- The goal is to generalize from the samples and map such that the output may be estimated for fresh samples in the future.

- **Examples:**

Assessment of credit risk:

- The x represents the customer’s properties.
- Whether or whether the f(x) has been accepted for credit.

The diagnosis of disease:

- The x represents the patient’s characteristics.
- The f(x) is the illness they are afflicted with.

Face recognition: is a technique for recognizing someone’s face.

- Bitmaps of people’s faces make up the x.
- The f(x) is used to give the face a name.

- **Deductive Learning:**
- Learners are initially exposed to concepts and generalizations, followed by particular examples and exercises to aid learning.
- Already existing rules are applied to the training examples.

- **Biased Hypothesis Space:**
- It does not include all types of training instances.
- The issue is that we have skewed the learner's thinking to only evaluate conjunctive possibilities. i.e does not consider all types of training examples.
- In this instance, a more expressive hypothesis space is required.

- **Unbiased Hypothesis Space:**
- The obvious answer to the challenge of ensuring that the target idea is represented in hypothesis space H is to create a hypothesis space that can represent any reachable notion.
- It provides a hypothesis capable of representing set of all examples. (which is actually not possible)

- So, Inductive bias refers to a set of assumptions made by a learning algorithm in order to conduct induction or generalize a limited set of observations (training data) into a general model of the domain.
- Induction would be impossible without such a bias, because observations may generally be extended in a variety of ways.
- Predictions for new scenarios could not be formed if all of these options were treated equally, that is, without any bias in the sense of a preference for certain forms of generalization (representing previous information about the target function to be learned).
- i.e. solution to unbiased is to define a biased hypothesis but in such a way that it will consider all the training examples.

- The idea of inductive bias is to let the learner generalize beyond the observed training examples to deduce new examples.
- ‘ > ’ -> Inductively inferred from.

- For example,
- x > y means y is inductively deduced from x.

- **Types of Inductive Bias:**
- **Maximum conditional independence:** It aims to maximize conditional independence if the hypothesis can be put in a Bayesian framework. The Naive Bayes classifier employs this bias.
- **Minimum cross-validation error:** Select the hypothesis with the lowest cross-validation error when deciding between hypotheses.
- **Maximum margin:** While creating a border between two classes, try to make the boundary as wide as possible. In support vector machines, this is the bias. The idea is that distinct classes are usually separated by large gaps.
- **Minimum hypothesis description length:** When constructing a hypothesis, try to keep the description as short as possible.

- **Minimum features:** features should be removed unless there is strong evidence that they are helpful. Feature selection methods are based on this premise.
- **Nearest neighbors:** Assume that the majority of the examples in a local neighborhood in feature space are from the same class.
- If the class of a case is unknown, assume that it belongs to the same class as the majority of the people in its near vicinity.
- The k-nearest neighbor’s algorithm employs this bias. Cases that are close to each other are assumed to belong to the same class.