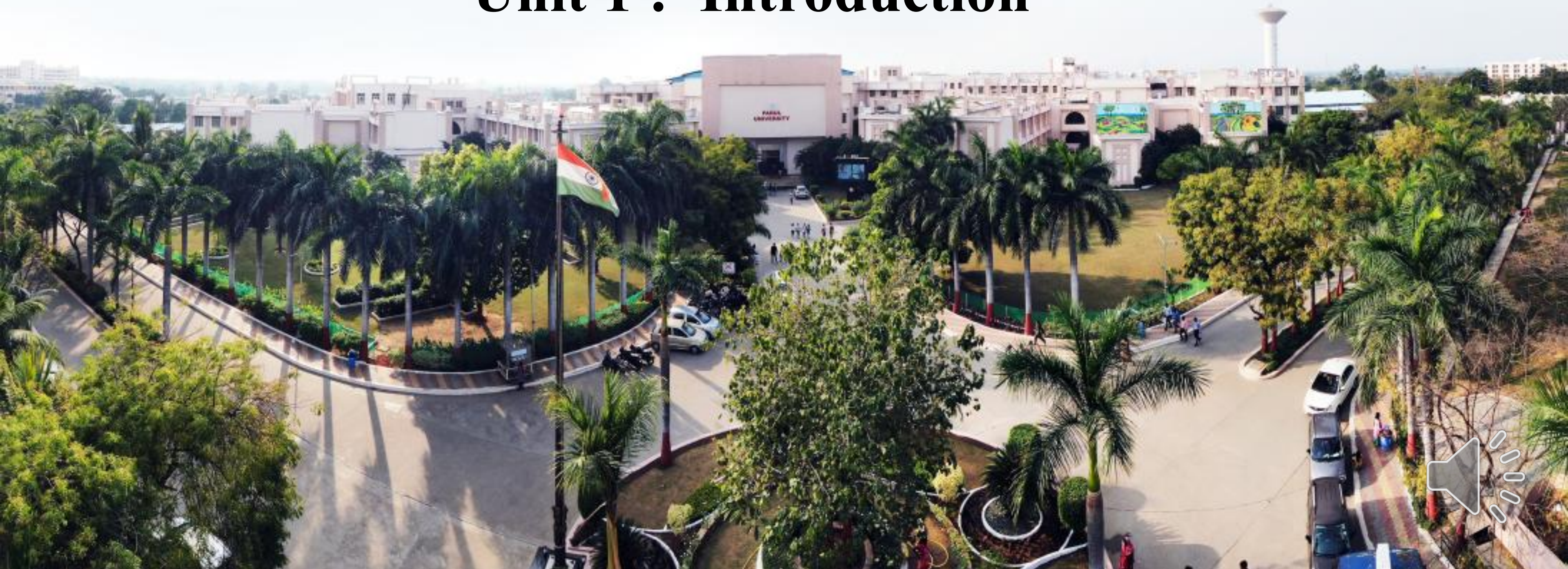
PARUL INSTITUTE OF ENGINEERING & TECHNOLOGY

PARUL UNIVERSITY, VADODARA (GUJARAT)

Department of AI and AIDS

Machine Learning (303105353)

Unit 1 : Introduction



Outline

- Introduction to Machine Learning
- Learning Paradigms
- PAC learning
- Basics of Probability
- Version Spaces.

Introduction to Machine Learning

•Machine learning is a subset of artificial intelligence
•It enables the machine to automatically learn from data, improve performance from past experiences, and make predictions.
•Machine learning contains a set of algorithms that work on a huge amount of data.
•Data is fed to these algorithms to train them, and on the basis of training, they build the model & perform a specific task.
•Machine learning uses various algorithms for **building mathematical models and making predictions using historical data or information.**
•It is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

Introduction to Machine Learning

What is Machine Learning?

In the real world, we are surrounded by humans who can learn everything from their experiences with their learning capability, and we have computers or machines which work on our instructions. But can a machine also learn from experiences or past data like a human does? So here comes the role of **Machine Learning**.

Human
I can learn everything automatically from experiences. Can u learn?
Machine
Yes, I can also learn from past data with the help of Machine learning

Introduction to Machine Learning

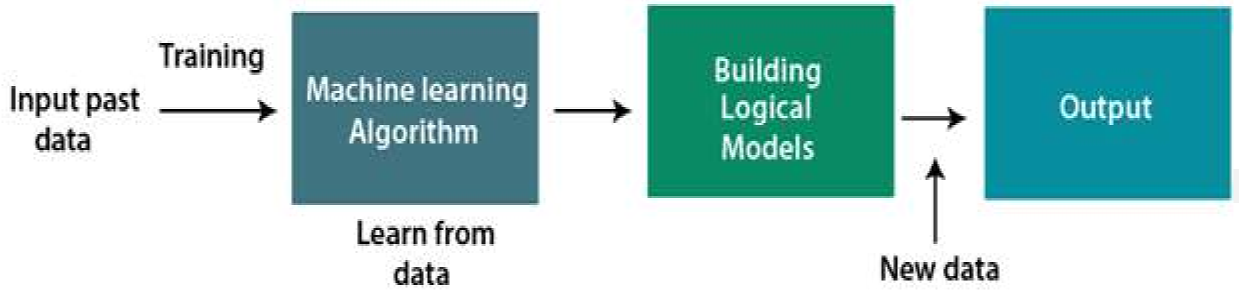
•**“Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.”**

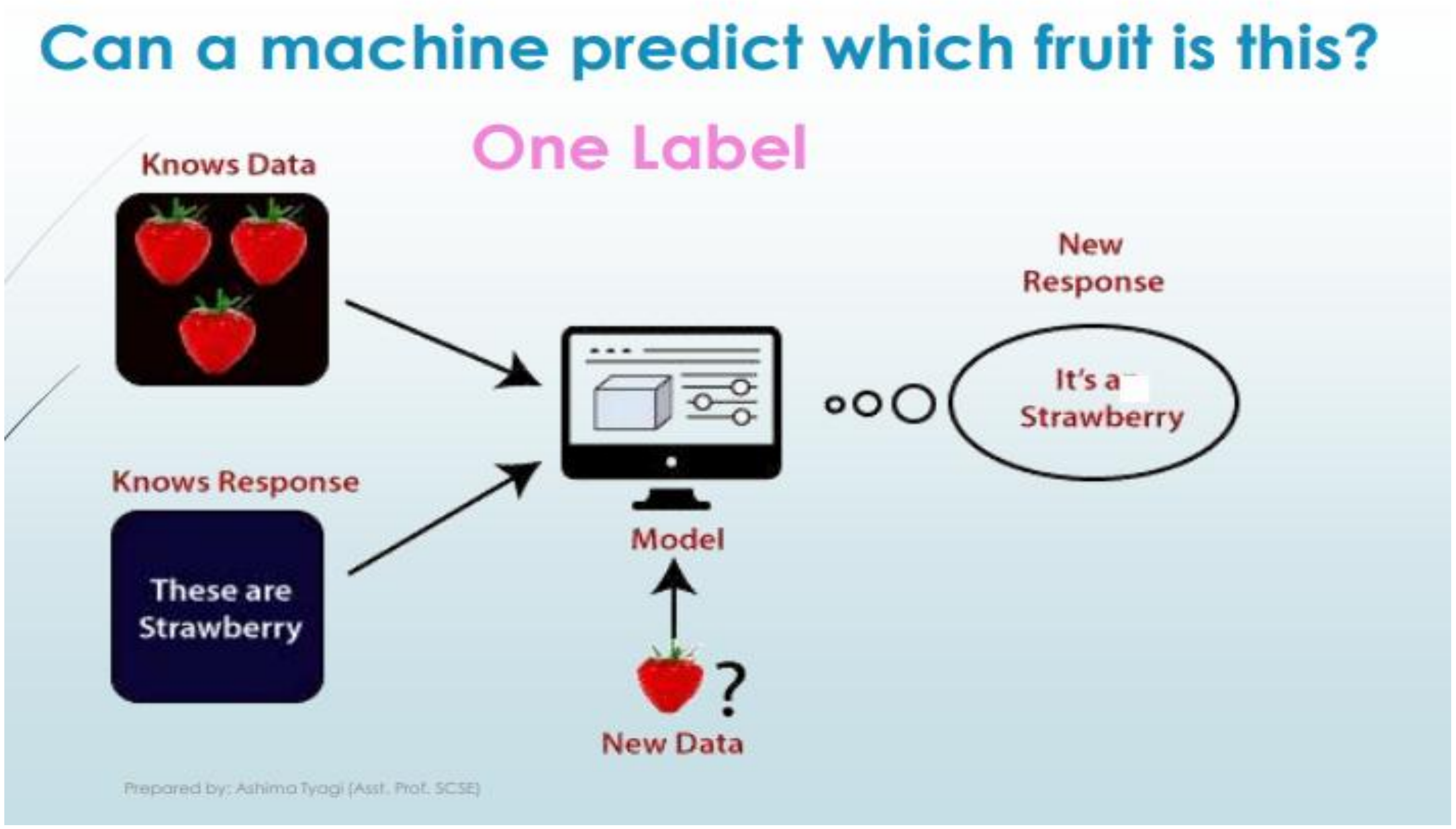
•With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive 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

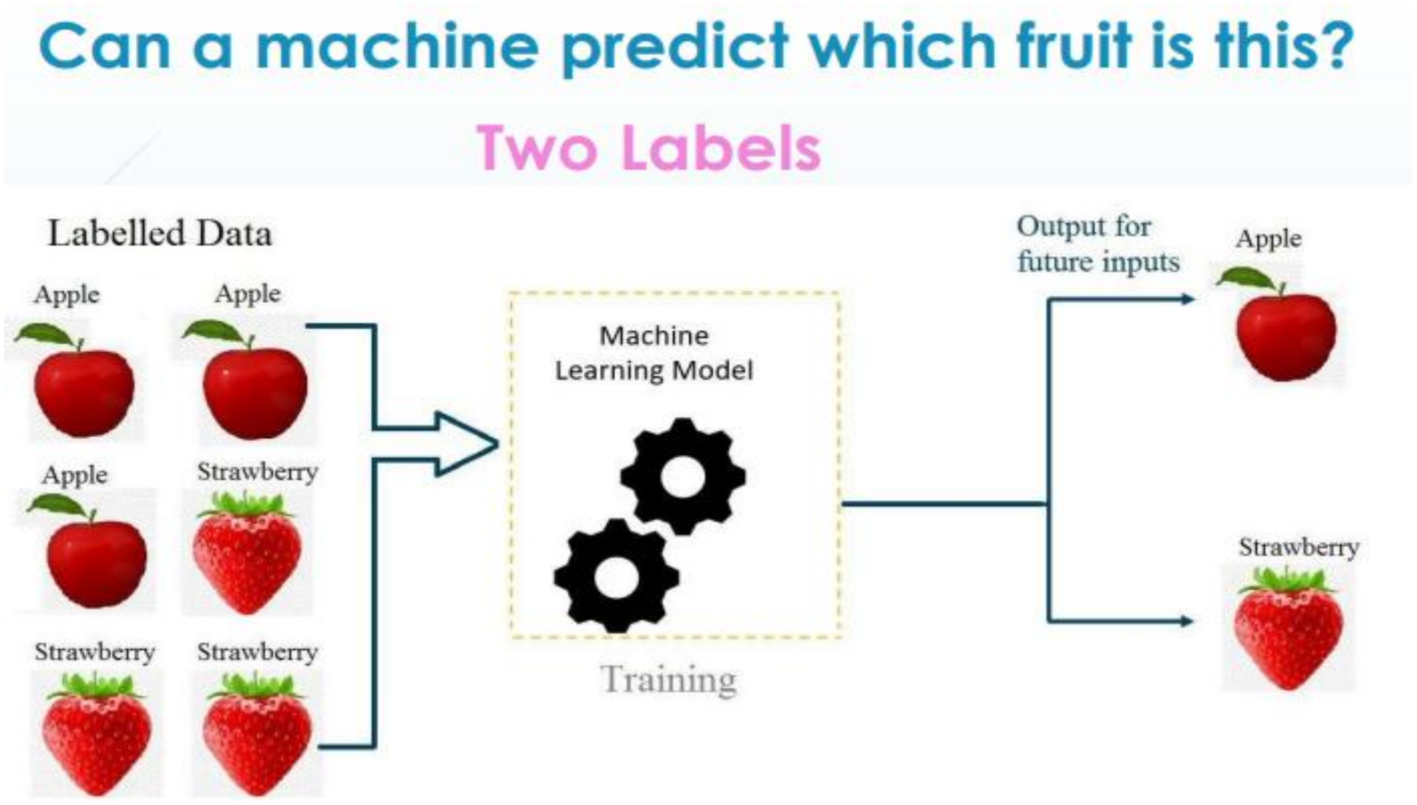
- A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it**. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output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





Can a machine predict which fruit is this?

Prepared by: Ashima Tyagi (Asst. Prof. SCSE)


Can a machine predict which fruit is this?
Two Labels
Labelled Data
Apple
Apple
Apple
Strawberry
Strawberry
Strawberry
Machine Learning Model
Training
Output for future inputs
Apple
Strawberry

Machine Learning

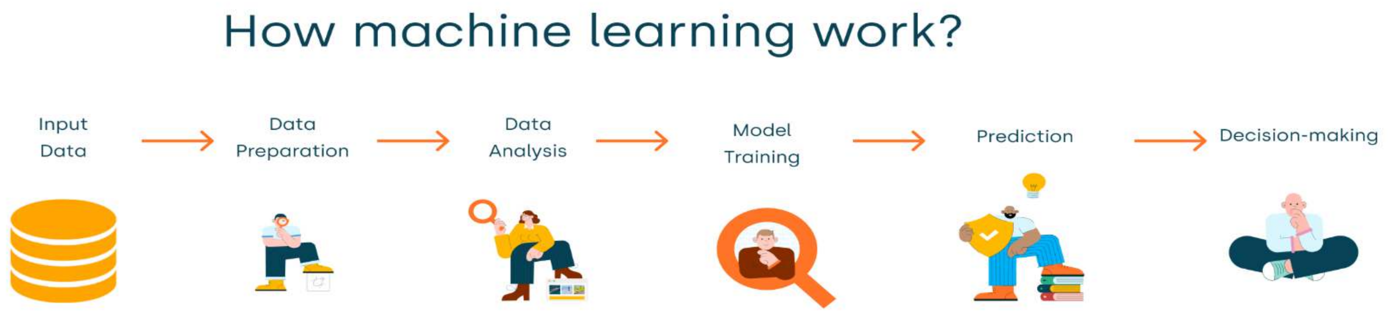
Machine Learning

- **Herbert Alexander Simon**: "Learning is any process by which a system improves performance from experience."
- "Machine Learning is concerned with computer programs that automatically improve their performance through experience. "

Herbert Simon
Turing Award 1975
Nobel Prize in Economics 1978

How does Machine Learning work

- Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:



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

- The need for machine learning is increasing day by day. The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly. As a human, we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.
- We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically. The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function. With the help of machine learning, we can save both time and money.

Need for Machine Learning

- The importance of machine learning can be easily understood by its uses cases, currently, machine learning is used in self-driving cars, cyber fraud detection, face recognition, and friend suggestion by Facebook, etc. Various top companies such as Netflix and Amazon have built machine learning models that are using a vast amount of data to analyze the user interest and recommend product accordingly.

Applications of Machine Learning

Sample applications of machine learning:

- ❑ **Web search**: ranking page based on what you are most likely to click on.
- ❑ **Computational** biology: rational design drugs in the computer based on past experiments.
- ❑ **Finance**: decide who to send what credit card offers to. Evaluation of risk on credit offers. How to decide where to invest money.
- ❑ **E-commerce**: Predicting customer churn. Whether or not a transaction is fraudulent
- ❑ **Space exploration**: space probes and radio astronomy.

Applications of Machine Learning

- **Robotics**: how to handle uncertainty in new environments. Autonomous. Self-driving car.
- **Information extraction**: Ask questions over databases across the web.
- **Social networks**: Data on relationships and preferences. Machine learning to extract value from data.
- **Debugging**: Use in computer science problems like debugging. Labor intensive process. Could suggest where the bug could be.

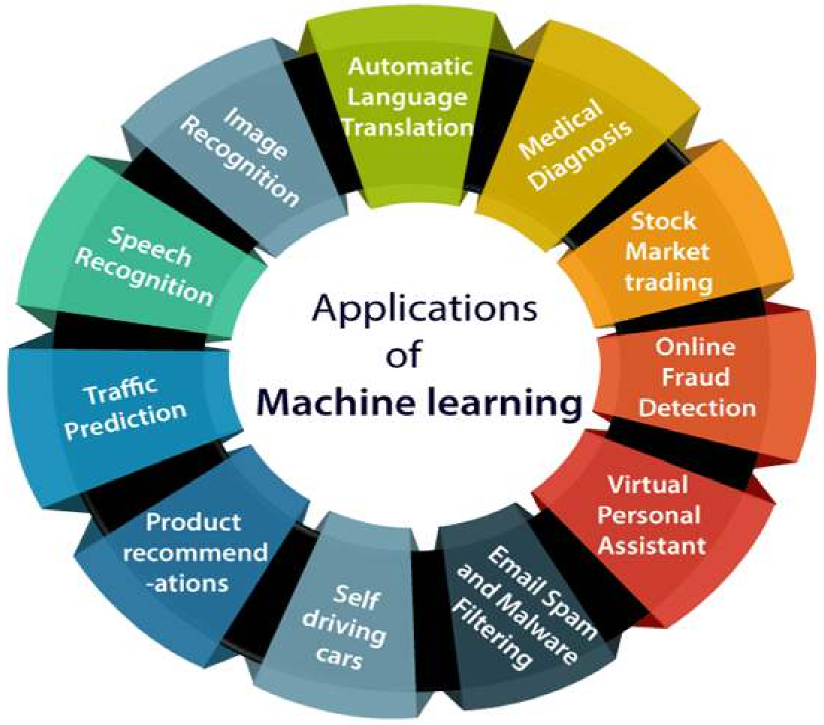
Applications of Machine Learning

But there are much more examples of ML in use

- ❑ **Prediction** — Machine learning can also be used in the prediction systems. Considering the loan example, to compute the probability of a fault, the system will need to classify the available data in groups.
- ❑ **Image recognition** — Machine learning can be used for face detection in an image as well. There is a separate category for each person in a database of several people.
- ❑ **Speech Recognition** — It is the translation of spoken words into the text. It is used in voice searches and more. Voice user interfaces include voice dialing, call routing, and appliance control. It can also be used a simple data entry and the preparation of structured documents.
- ❑ **Medical diagnoses —** ML is trained to recognize cancerous tissues.
- ❑ **Financial industry and trading** — companies use ML in fraud investigations and credit checks.

Applications of Machine Learning

But there are much more examples of ML in use

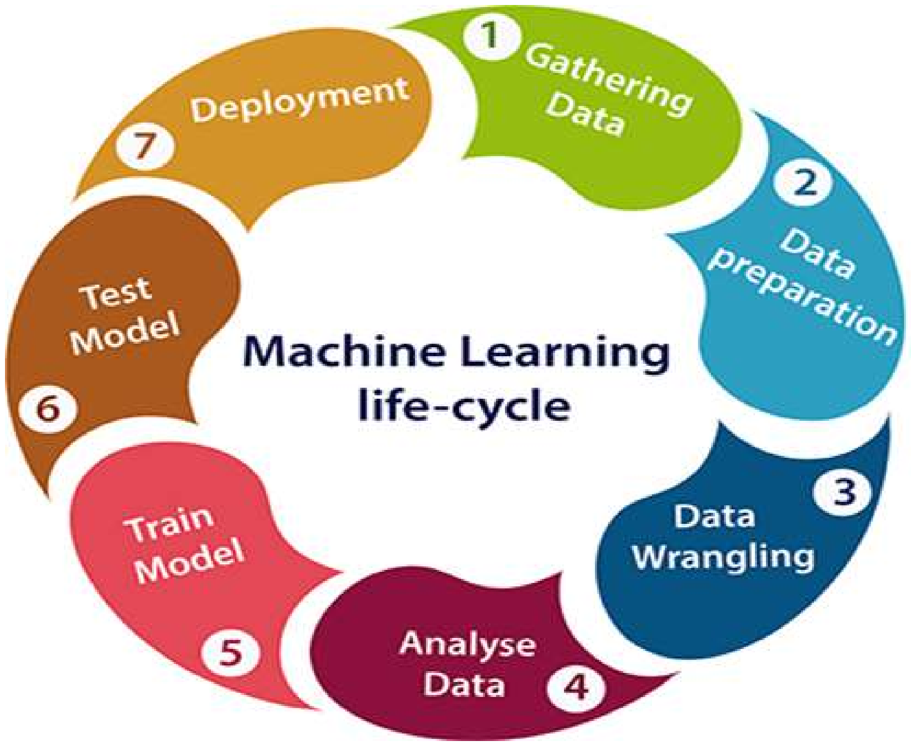

Applications of Machine learning
Automatic Language Translation
Medical Diagnosis
Stock Market trading
Online Fraud Detection
Virtual Personal Assistant
Email Spam and Malware Filtering
Self driving cars
Product recommend -ations
Traffic Prediction
Speech Recognition
Image Recognition

Applications of Machine Learning

5. Self-driving cars:

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

Machine learning Life cycle



Machine learning Life cycle

1. Gathering Data:
Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.
In this step, we need to identify the different data sources, as data can be collected from various sources such as **files**, **database**, **internet**, or **mobile devices**. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.
This step includes the below tasks:

- **Identify various data sources**
- **Collect data**
- **Integrate the data obtained from different sources**

By performing the above task, we get a coherent set of data, also called as a **dataset**. It will be used in further steps.

Machine learning Life cycle

2. Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

Data exploration:

It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.

A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

Data pre-processing:

Now the next step is preprocessing of data for its analysis.

Machine learning Life cycle

3. Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

Missing Values

Duplicate data

Invalid data

Noise

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

Machine learning Life cycle

4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

Selection of analytical techniques

Building models

Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as **Classification**, **Regression**, **Cluster analysis**, **Association**, etc. then build the model using prepared data, and evaluate the model.

Hence, in this step, we take the data and use machine learning algorithms to build the model.

Machine learning Life cycle

7. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.

If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

Learning Paradigms in Machine Learning

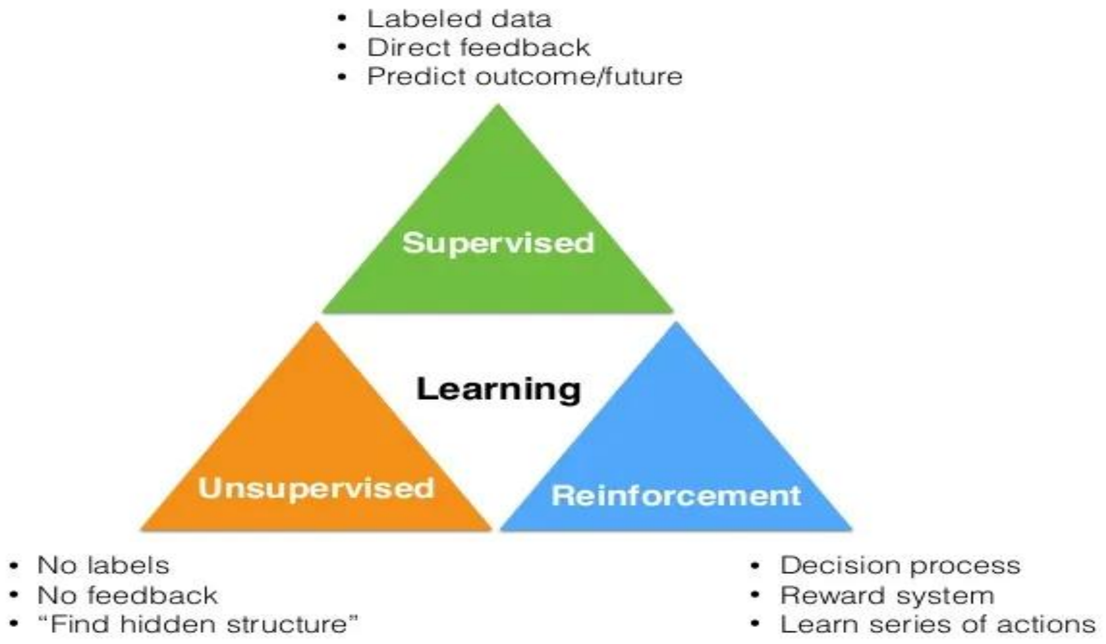
Learning Paradigms basically states a particular pattern on which something or someone learns.

Learning Paradigms related to machine learning, i.e how a machine learns when some data is given to it, its pattern of approach for some particular data.

Machine learning is commonly separated into three main learning paradigms:

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

Learning Paradigms in Machine Learning



Learning Paradigms in Machine Learning

1) 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**.

Learning Paradigms in Machine Learning

- The main goal of the supervised learning technique is to map the input variable(x) with the output variable(y). Some real-world applications of supervised learning are Risk Assessment, Fraud Detection, Spam filtering, etc.

Learning Paradigms in Machine Learning

2) 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 si[illegible]
- In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

Learning Paradigms in Machine Learning

Advantages of Unsupervised Learning

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

Disadvantages of Unsupervised Learning

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

Learning Paradigms in Machine Learning

3) 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.

Learning Paradigms in Machine Learning

Advantages and Disadvantages of Reinforcement Learning

Advantages

- It helps in solving complex real-world problems which are difficult to be solved by general techniques.
- The learning model of RL is similar to the learning of human beings; hence most accurate results can be found.
- Helps in achieving long term results.

Disadvantage

- RL algorithms are not preferred for simple problems.
- RL algorithms require huge data and computations.
- Too much reinforcement learning can lead to an overload of states which can weaken the results.

PAC - Learning

- **PAC** stand for **Probably approximately correct**
- **Probably approximately correct (PAC) learning** is a framework for mathematical analysis of machine learning algorithm.
- In the other words PAC learning is a theoretical framework for analyzing the generalization performance of machine learning algorithms.

Goal: With High Probability ("Probably"),the select hypothesis will have lower error ("Approximately Correct"")

In the PAC model ,we specify two small parameter ,ε (epsilon) and δ (delta) and require that with probability at least (1-δ) a system learn a concept with error at most ε.

PAC - Learning

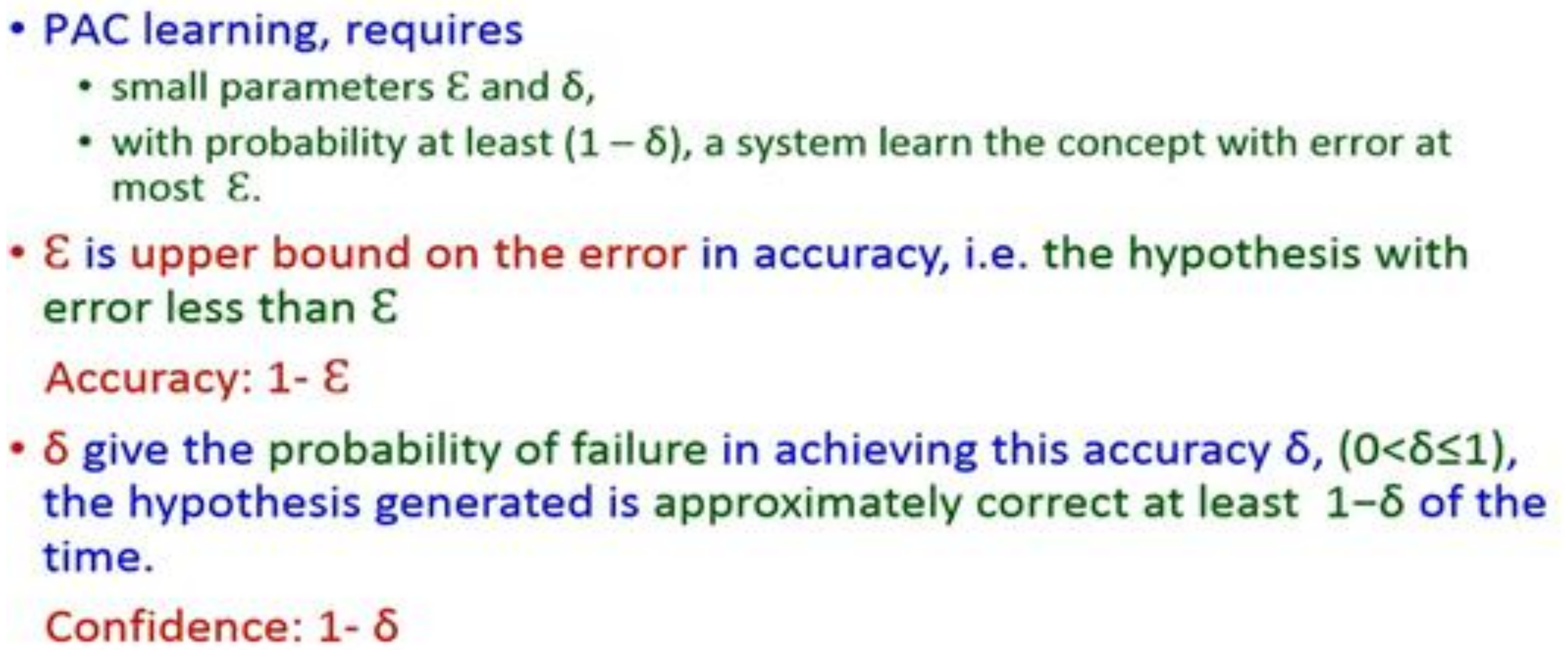
ε and δ parameters:

- ε gives an upper bound on the error in the accuracy with which h approximated(Accuracy : 1-ε)
- δ gives the probability of failure in the achieving this accuracy (Confidence : 1-δ)

PAC - Learning

- A good learner will learn with high probability and close approximation to the target concept
- With high probability, the selected hypothesis will have lower the error ("Approximately Correct") with the parameter ε and δ

PAC - Learning



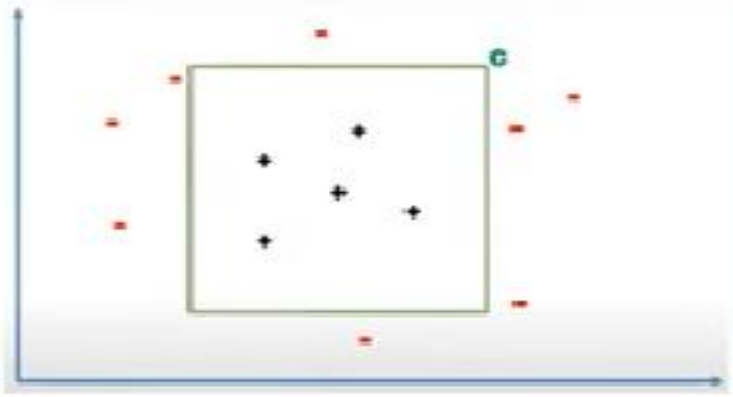
- PAC learning, requires
  - small parameters Ɛ and δ,
  - with probability at least (1 – δ), a system learn the concept with error at most Ɛ.
- Ɛ is upper bound on the error in accuracy, i.e. the hypothesis with error less than Ɛ

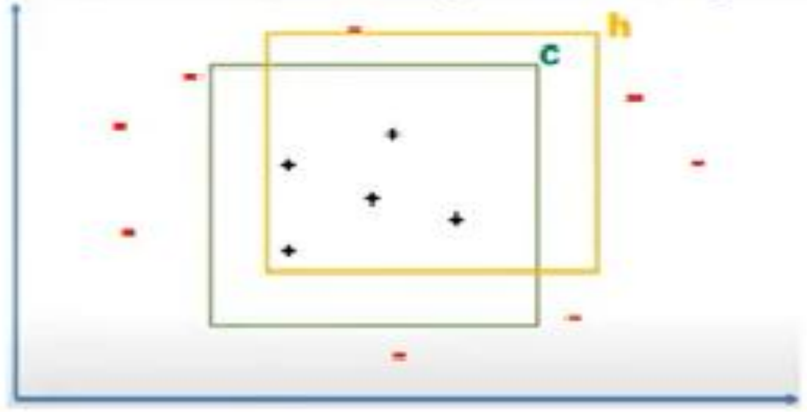
  Accuracy: 1- Ɛ
- δ give the probability of failure in achieving this accuracy δ, (0<δ≤1), the hypothesis generated is approximately correct at least 1–δ of the time.

  Confidence: 1- δ

PAC – Learning Example

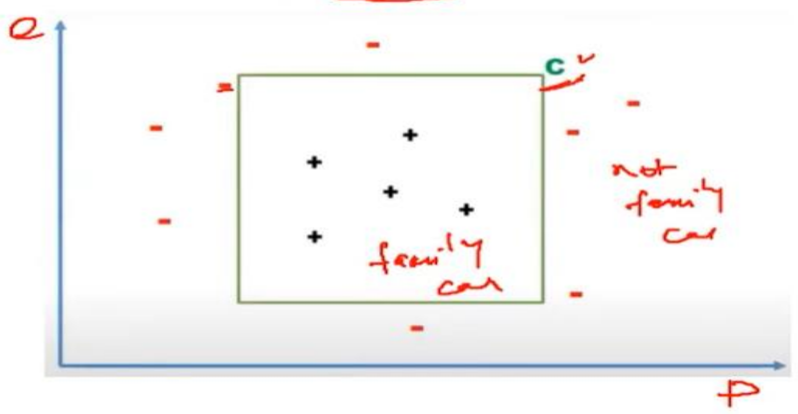
- N number of Car having Price and Engine power, as training set, (p,e), find the car is family car or not.
- An algorithm gives answer whether the car is family car or not.
- C – Target function
- Instances within rectangle represents family cars and outside are not family cars
- Hypothesis h – closely approximate C, and there may be error region.

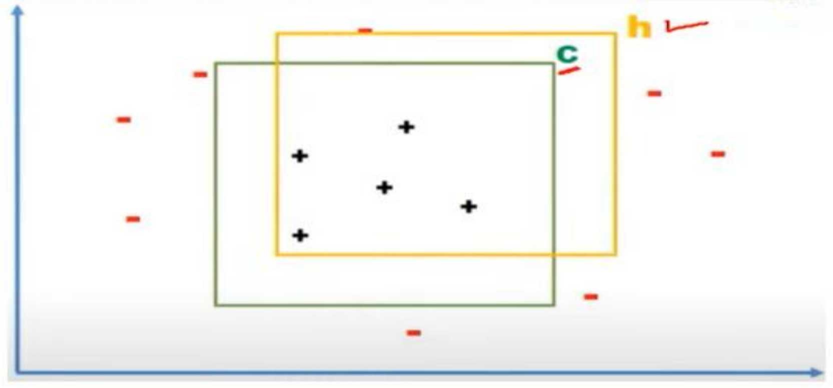




PAC – Learning Example

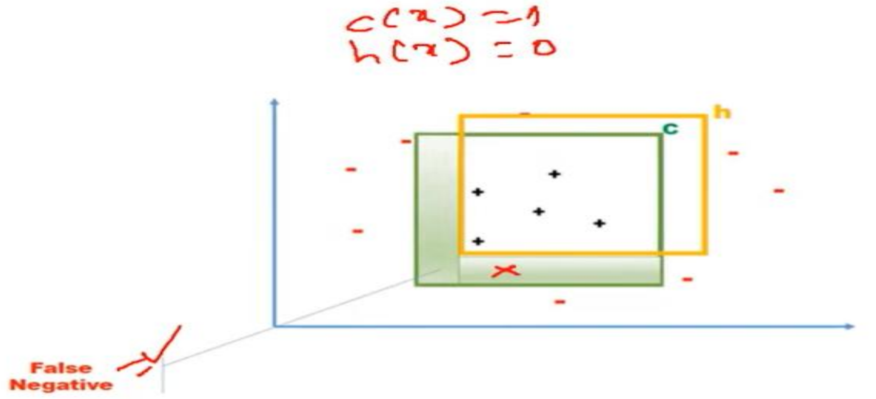
- N number of Car having Price and Engine power, as training set, (p,e), find the ca is family car or not.
- An algorithm gives answer whether the car is family car or not.
- C – Target function
- Instances within rectangle represents family cars and outside are not family cars
- Hypothesis h – closely approximate C, and there may be error region.

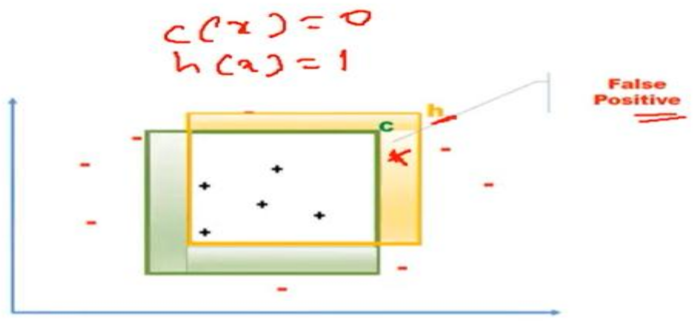




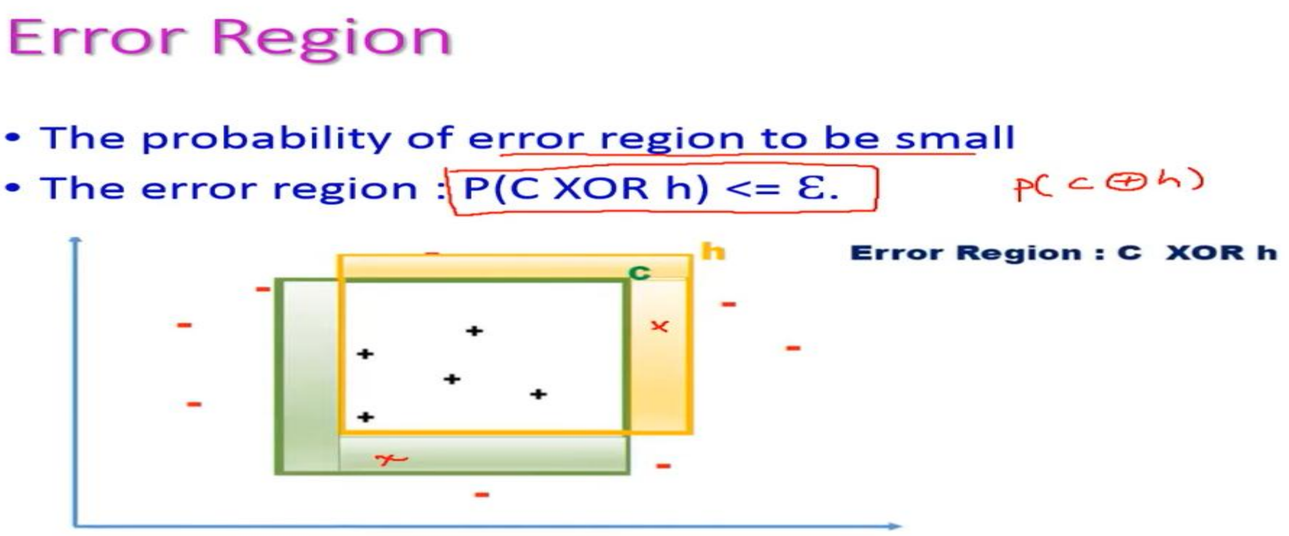
False Negative and False Positive

- Instances lies on shaded region are positive/negative according to our actual function 'C', but those are negative/positive based on the hypothesis h. Hence it is called as false negative or false positive





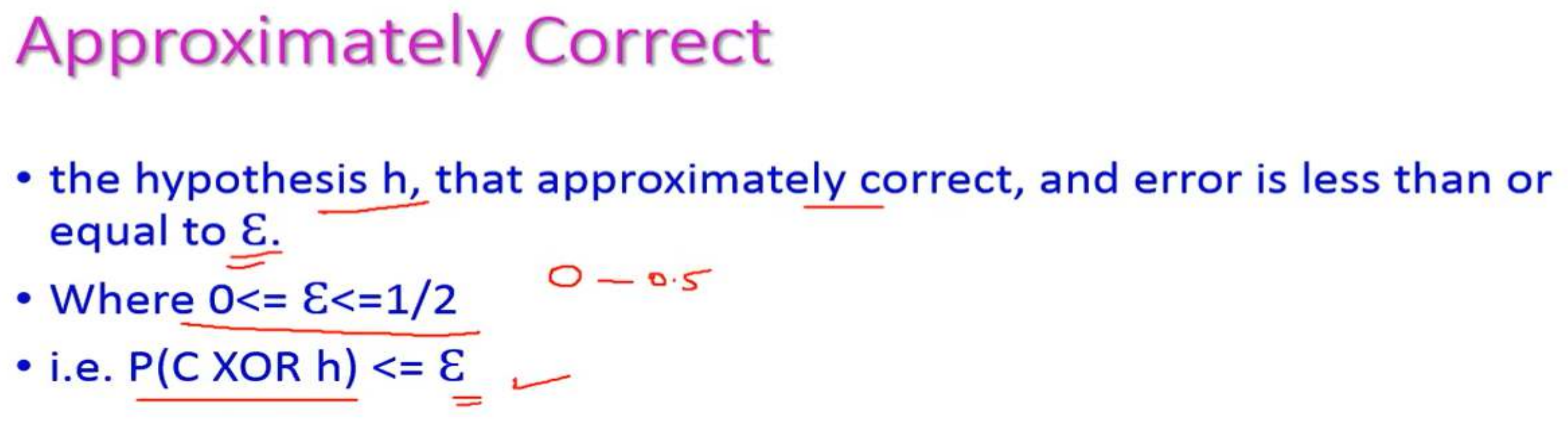
PAC – Learning Example



Error Region

- The probability of error region to be small
- The error region : $\boxed{P(C \text{ XOR } h) <= \mathcal{E}.}$ $P(c \oplus h)$

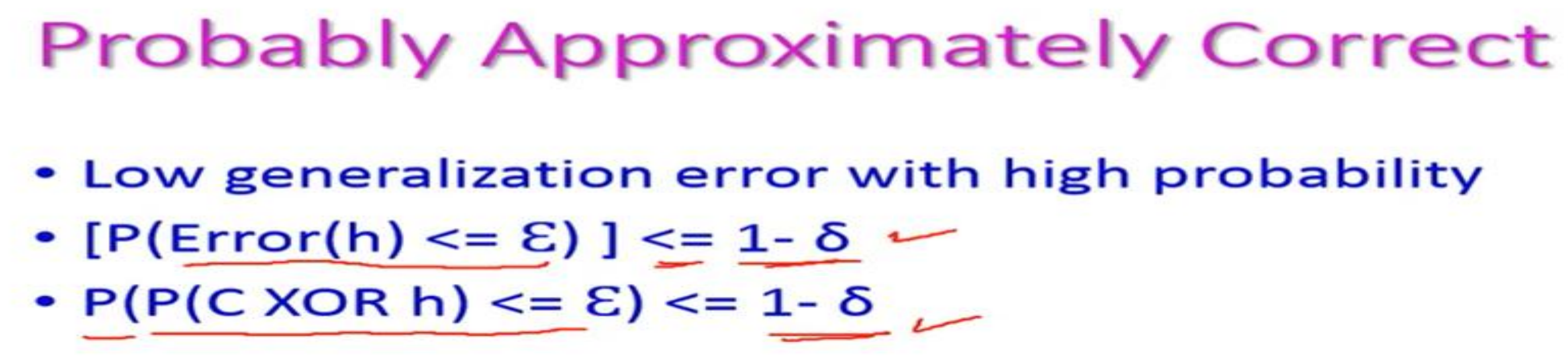
PAC – Learning Example



Approximately Correct

- the hypothesis h, that approximately correct, and error is less than or equal to Ɛ.
- Where 0<= Ɛ<=1/2 0 – 0.5
- i.e. P(C XOR h) <= Ɛ ✓

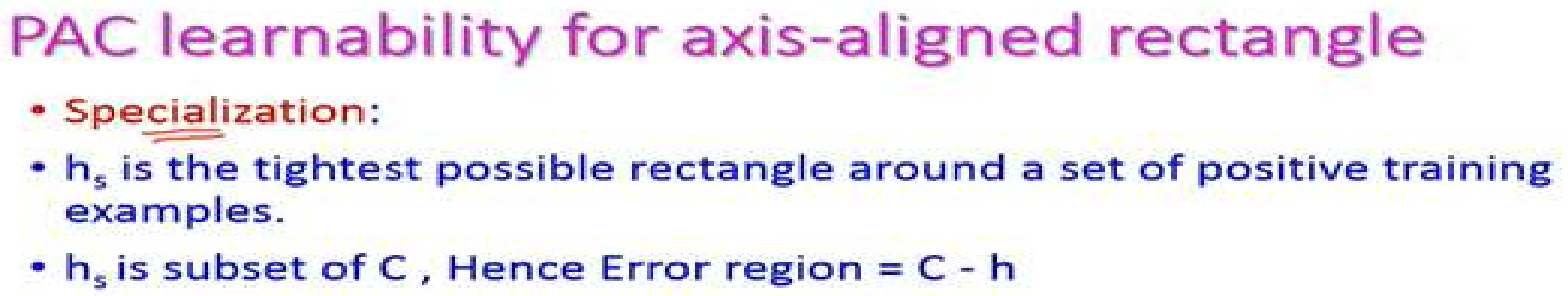
PAC – Learning Example



Probably Approximately Correct

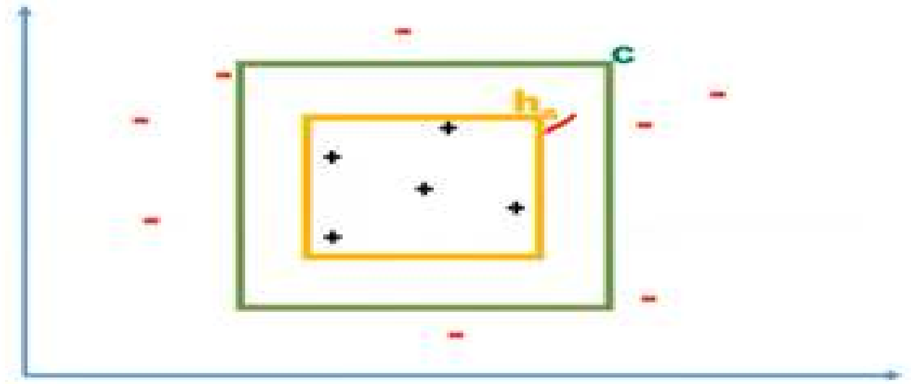
- Low generalization error with high probability
- [P(Error(h) <= Ɛ)] <= 1- δ ✓
- P(P(C XOR h) <= Ɛ) <= 1- δ ✓

PAC – Learning Example



PAC learnability for axis-aligned rectangle

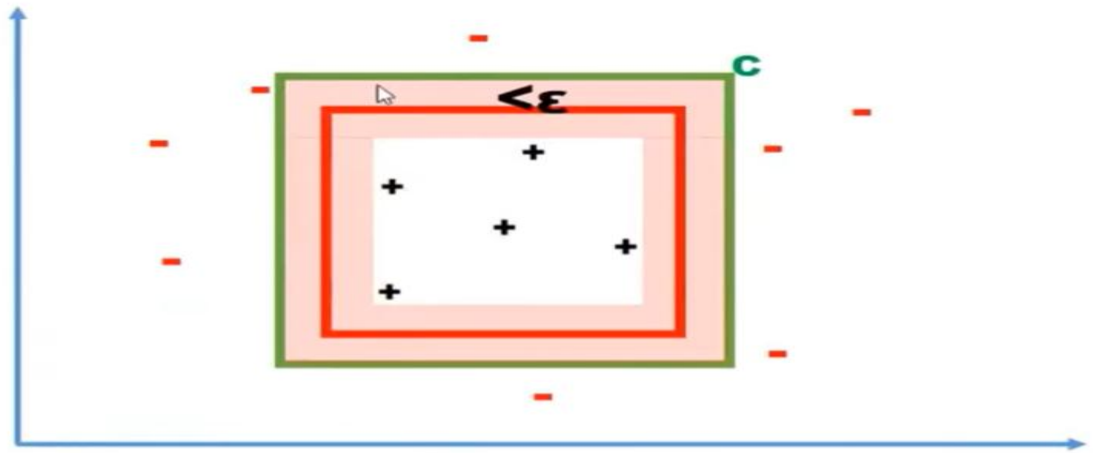
- Specialization:
- h_s is the tightest possible rectangle around a set of positive training examples.
- h_s is subset of C , Hence Error region = C - h



PAC – Learning Example

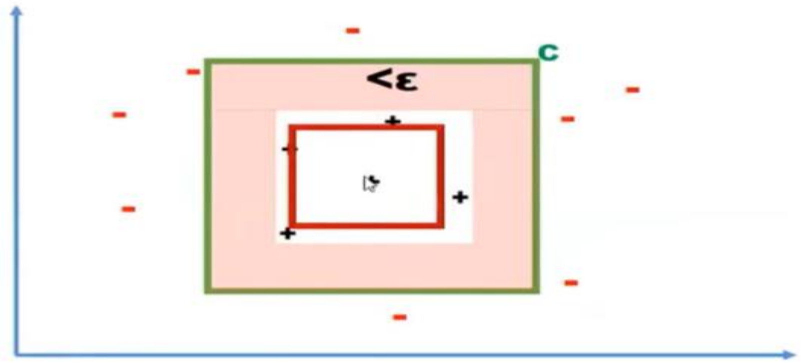
Approximately correct

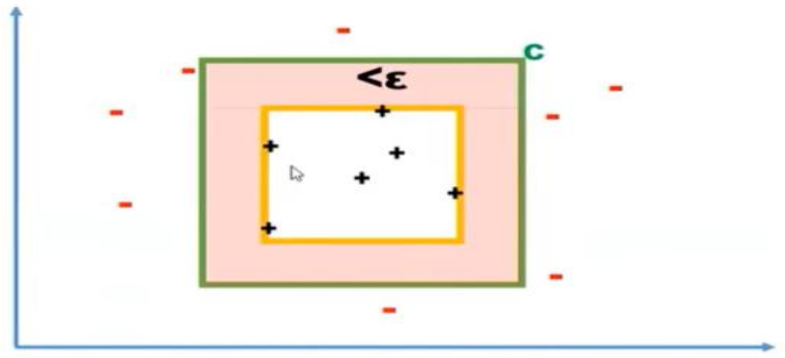
- If an hypothesis lies between h and c (shaded region) then it is approximately correct.



PAC – Learning Example

- If the generated hypothesis does not touch any of these region
- Error region is greater than Ɛ and not approximately correct, because the error region got increased.
- Atleast one +ve example at each side of the rectangle

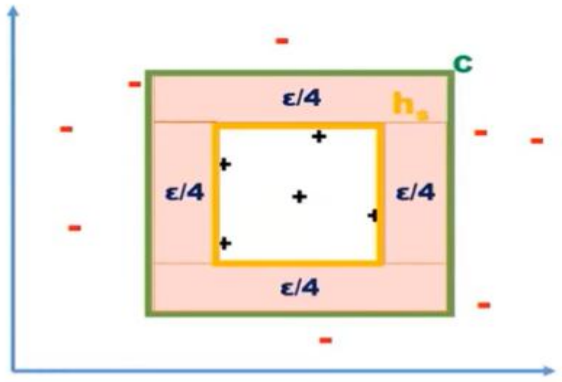




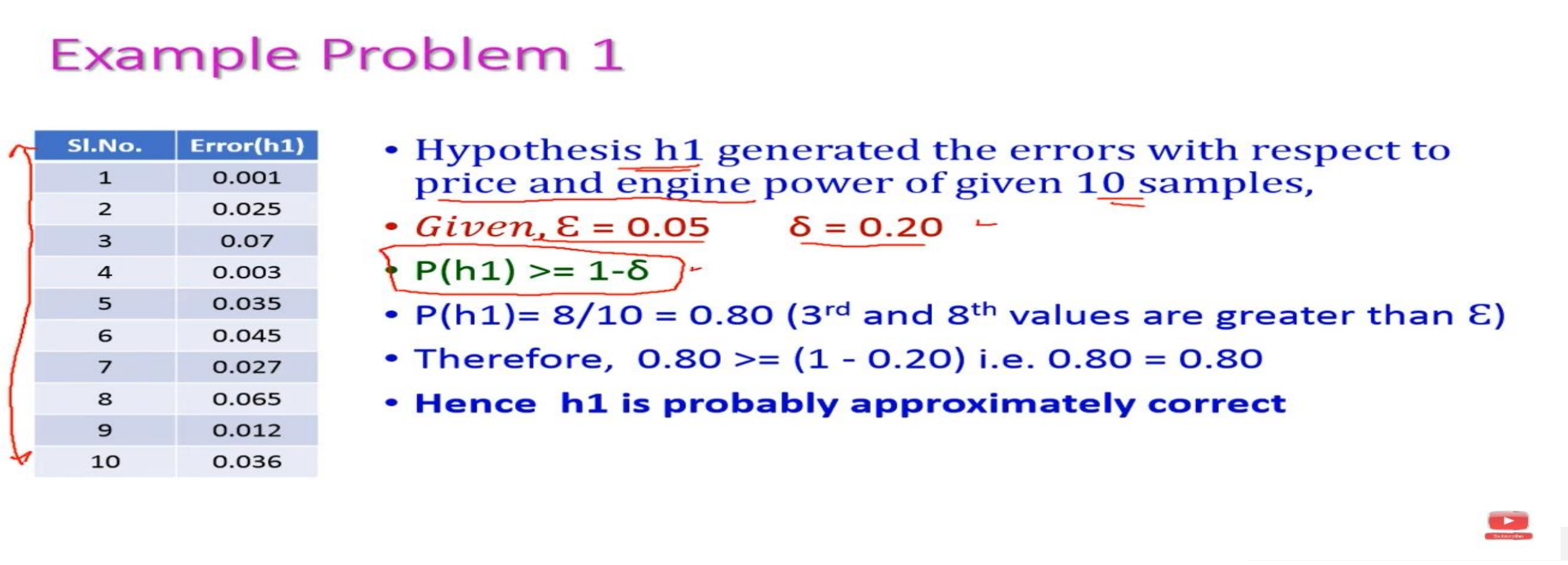
PAC – Learning Example

Error Region

- Error Region = sum of four rectangular strips < $\mathcal{E}$
- Each strip is at most $\mathcal{E}/4$
- Probability of positive example falling in any one of the strip (error region = $\mathcal{E}/4$)
- Probability that a randomly drawn positive example misses a strip = $1- \mathcal{E}/4$
- $P(m \text{ instance miss a strip}) = (1- \mathcal{E}/4)^m$
- $P(m \text{ instances miss any strip}) < 4(1- \mathcal{E}/4)^m$
- Finally we get $m > 4/ \mathcal{E} \log 4/ \delta$



PAC – Learning Example

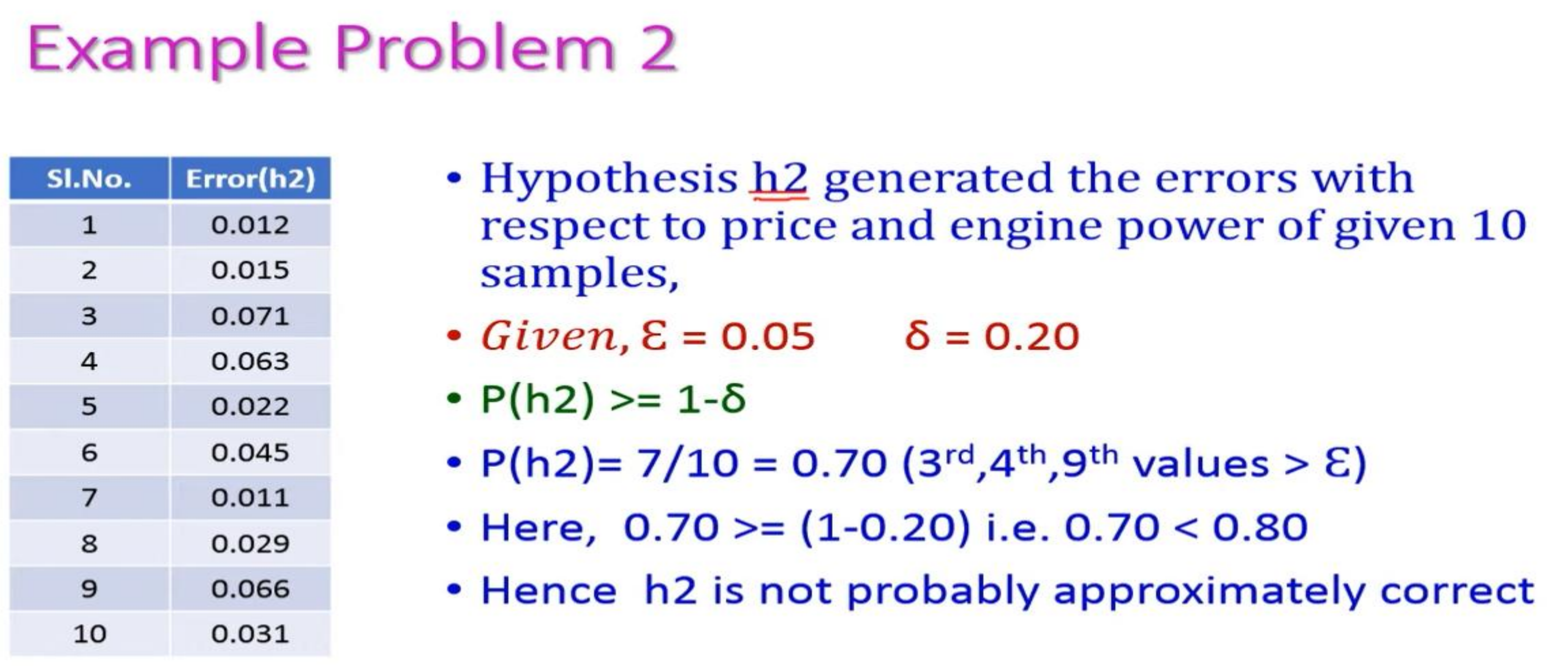


Example Problem 1

Sl.No.	Error(h1)
1	0.001
2	0.025
3	0.07
4	0.003
5	0.035
6	0.045
7	0.027
8	0.065
9	0.012
10	0.036

- Hypothesis h1 generated the errors with respect to price and engine power of given 10 samples,
- *Given*, $\mathcal{E} = 0.05$ $\delta = 0.20$
- $\boxed{P(h1) >= 1-\delta}$
- P(h1)= 8/10 = 0.80 (3rd and 8th values are greater than $\mathcal{E}$)
- Therefore, 0.80 >= (1 - 0.20) i.e. 0.80 = 0.80
- **Hence h1 is probably approximately correct**

PAC – Learning Example

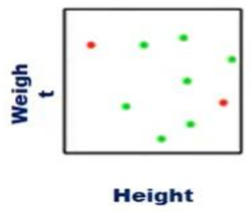


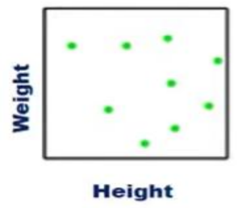
Example Problem 2

Sl.No.	Error(h2)
1	0.012
2	0.015
3	0.071
4	0.063
5	0.022
6	0.045
7	0.011
8	0.029
9	0.066
10	0.031

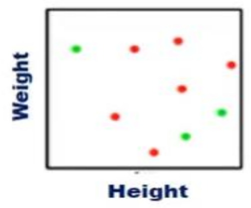
- Hypothesis h2 generated the errors with respect to price and engine power of given 10 samples,
- *Given*, $\varepsilon = 0.05$ $\delta = 0.20$
- $P(h2) >= 1-\delta$
- $P(h2)= 7/10 = 0.70$ (3rd,4th,9th values > ε)
- Here, $0.70 >= (1-0.20)$ i.e. $0.70 < 0.80$
- Hence h2 is not probably approximately correct

PAC – Learning Example





⋮



Error(H1)	Error(h2)
0.04	0.04
0.03	0.035
0.09	0.039
0.06	0.06
0.025	0.025
0.049	0.059
0.04	0.04
0.03	0.03
0.05	0.55
0.043	0.043

$\varepsilon = 0.05$ $\delta = 0.20$

P (H1) = 8/10=0.80

P (H1) = 8/10=0.80 >=1- 0.20

Hence H1 is probably approximately correct

P(H2)= 7/10=0.70

P(H2)= 7/10=0.70 <1-0.20

Hence H2 is not probably approximately correct

Issues with Machine Learning

Machine learning (ML) has proven to be a transformative technology, but it comes with its share of challenges and issues. These can be broadly categorized into technical challenges, ethical considerations, and practical limitations.

1. Data-Related Issues

Data Quality: Machine learning models depend heavily on the quality of data. Missing values, noisy data, or inconsistent formatting can degrade performance.

Data Quantity: Many ML algorithms require large amounts of labeled data for training, which might not always be available.

Data Bias: Datasets can reflect societal biases, leading to unfair or discriminatory predictions.

Imbalanced Data: When certain classes or categories are underrepresented in the data, models may perform poorly on these underrepresented groups.

Data Privacy: Collecting and using personal data can lead to privacy violations if not handled responsibly.

Issues with Machine Learning

2. Algorithmic Challenges

•**Overfitting**: The model performs well on training data but fails to generalize to unseen data.

•**Underfitting**: The model is too simplistic to capture the underlying patterns in the data.

•**Model Interpretability**: Complex models, such as deep neural networks, can act as black boxes, making it hard to understand their decision-making processes.

•**Hyperparameter Tuning**: Selecting the right parameters (e.g., learning rate, number of layers) can be a time-consuming trial-and-error process.

•**Scalability**: Some algorithms do not scale well with large datasets or high-dimensional data.

Issues with Machine Learning

3. Computational and Resource Constraints

•**High Computational Costs**: Training state-of-the-art models, especially deep learning models, requires substantial computational power.

•**Energy Consumption**: Large-scale ML models can have significant energy footprints, raising environmental concerns.

•**Latency Issues**: Real-time applications (e.g., autonomous driving, financial trading) demand low-latency predictions, which can be challenging for complex models.

Issues with Machine Learning

5. Ethical and Social Concerns

•**Bias and Fairness**: ML models can perpetuate or amplify societal biases if not properly addressed.

•**Transparency**: Lack of transparency in model decisions can erode trust, especially in critical applications like healthcare or criminal justice.

•**Job Displacement**: Automation through ML can lead to job loss in certain industries, raising socio-economic concerns.

•**Security**: ML systems are vulnerable to adversarial attacks, where small, carefully designed perturbations can cause incorrect predictions.

Issues with Machine Learning

6. Research and Knowledge Gaps

•**Generalization Across Domains**: Models trained on specific tasks often struggle to transfer knowledge to new domains.

•**Explainability**: Developing methods to make ML models more interpretable remains an ongoing research challenge.

•**Ethical AI Standards**: There's a lack of universal standards or frameworks to ensure ethical AI development and deployment.

How to Address These Machine Learning Issues

•**Better Data Practices**: Focus on data cleaning, augmentation, and collection to improve quality and diversity.

•**Regularization Techniques**: Use methods like dropout, L1/L2 regularization to combat overfitting.

•**Explainable AI (XAI)**: Invest in tools and methods to make models more interpretable.

•**Ethical Frameworks**: Adopt guidelines like "fairness, accountability, and transparency" in AI projects.

•**Continuous Monitoring**: Implement robust monitoring systems to detect model drift or failures in production.

Concept Learning in Machine Learning

Concept learning refers to the process of inferring a general description of a target concept from a set of examples. It is a fundamental component of many machine learning algorithms, especially in the context of supervised learning. Concept learning helps machines identify and classify objects, phenomena, or situations based on their characteristics.

The Concept Learning Task

The primary goal of concept learning is to find a hypothesis $h \in H$ such that

$\forall x,\ h(x)=C(x)$

where x represents an instance, and $C(x)$ is the true classification.
In practice, the goal is to find h that closely approximates C when C is unknown.

Key Elements of Concept Learning

•**Hypothesis Space (H)**:
The set of all possible hypotheses that can be considered to describe the target concept. Each hypothesis represents a potential generalization of the examples.

•**Target Concept (C)**:
The actual concept or rule we aim to learn. It is typically unknown and is approximated through hypothesis selection.

•**Training Examples**:
A set of labeled data points comprising:

•**Positive Examples**: Instances belonging to the target concept.

•**Negative Examples**: Instances not belonging to the target concept.

•**Learner**:
An algorithm or model that selects hypotheses from the hypothesis space to best fit the training examples.

•**Generalization and Specialization**:

•**Generalization**: Broadening a hypothesis to include more instances.

•**Specialization**: Narrowing a hypothesis to exclude certain instances.

Key Algorithms for Concept Learning

Candidate Elimination Algorithm:
- Uses the **version space** approach.
- Maintains two boundaries:
  - **General Hypothesis (G)**: This represents the most general description of the concept.
  - **Specific Hypothesis (S)**: This represents the most specific description of the concept.
- Refines GGG and SSS iteratively as new examples are presented.

Find-S Algorithm:
- Starts with the most specific hypothesis.
- Iteratively generalizes the hypothesis to cover positive examples while excluding negative examples.
- Simple but prone to overfitting.

Challenges in Concept Learning

•**Ambiguity in Hypothesis Space**:
•Multiple hypotheses can equally describe the training data.
•Need for criteria (e.g., Occam's Razor) to select simpler hypotheses.

•**Incomplete or Noisy Data**:
•Noise in data can lead to incorrect generalizations or specializations.
•Missing examples can limit the learner's ability to define the concept accurately.

•**Complex Concepts**:
•Complex relationships or high-dimensional data can make hypothesis search computationally expensive.

•**Overfitting**:
•A hypothesis may perfectly fit the training data but fail to generalize to new instances.

Applications of Concept Learning

- **Image Recognition**: Learning visual concepts like "cat" or "car" from labeled images.
- **Natural Language Processing**: Understanding linguistic concepts such as sentiment or topic categories.
- **Medical Diagnosis**: Identifying conditions based on symptoms and test results.
- **Fraud Detection**: Learning patterns of fraudulent transactions.

Improving Concept Learning

•**Feature Engineering**: Ensure relevant features are included to improve learning accuracy.

•**Regularization**: Prevent overfitting by penalizing overly complex hypotheses.

•**Active Learning**: Use unlabeled data to query labels for ambiguous instances.

•**Hybrid Approaches**: Combine concept learning with probabilistic or neural methods to handle complex data

Version Space

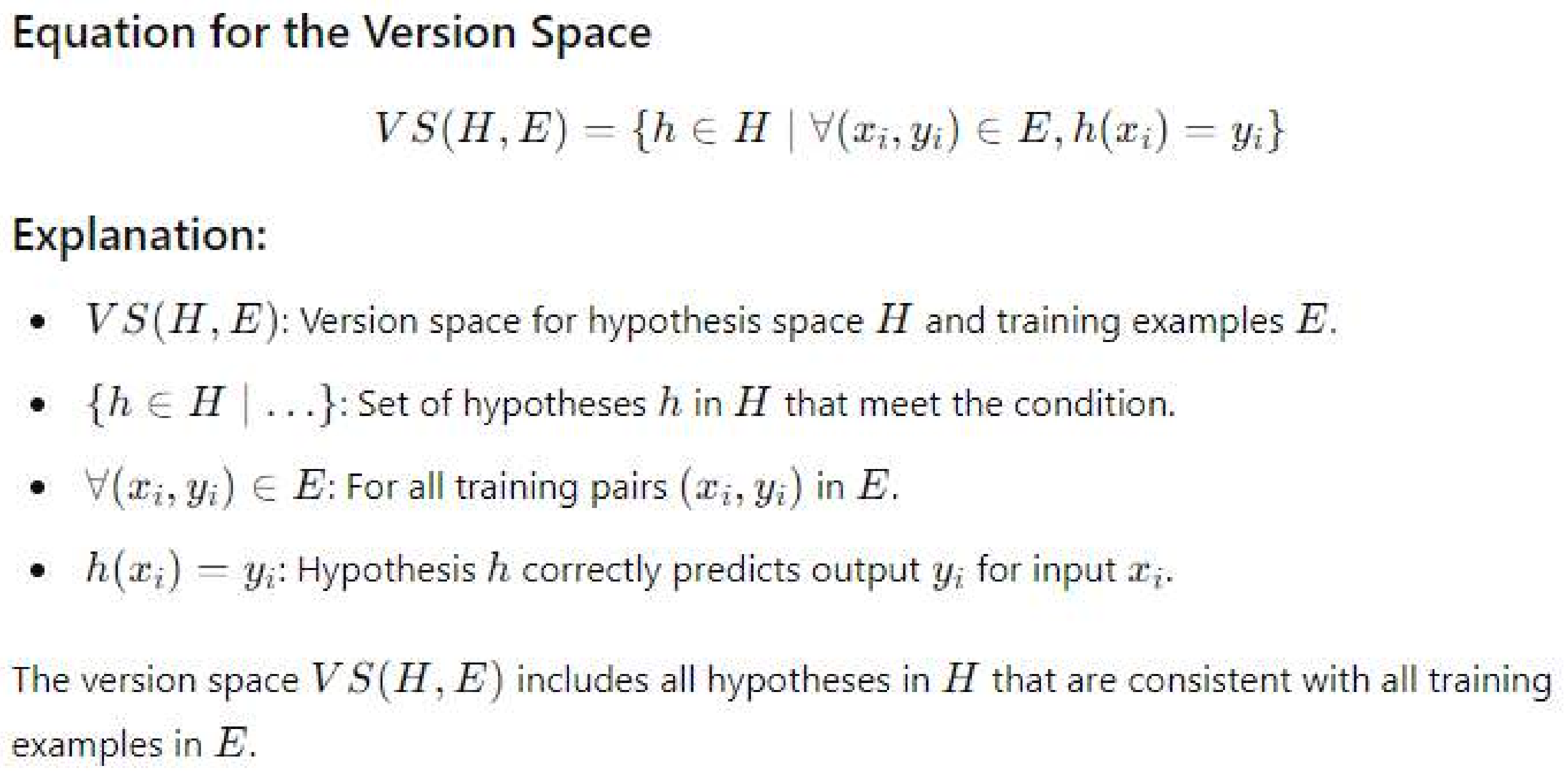
A **Version Space** is the subset of all possible hypotheses in the hypothesis space H that are consistent with the training examples. A hypothesis is considered consistent if it correctly classifies all the training examples.

$$\textbf{Version Space (VS)} = \{h \in H \mid \forall (x,y) \in D,\ h(x) = y\}$$

where:

- H: Hypothesis space
- D: Training data
- (x, y): An example x with label y

Version Space



Equation for the Version Space

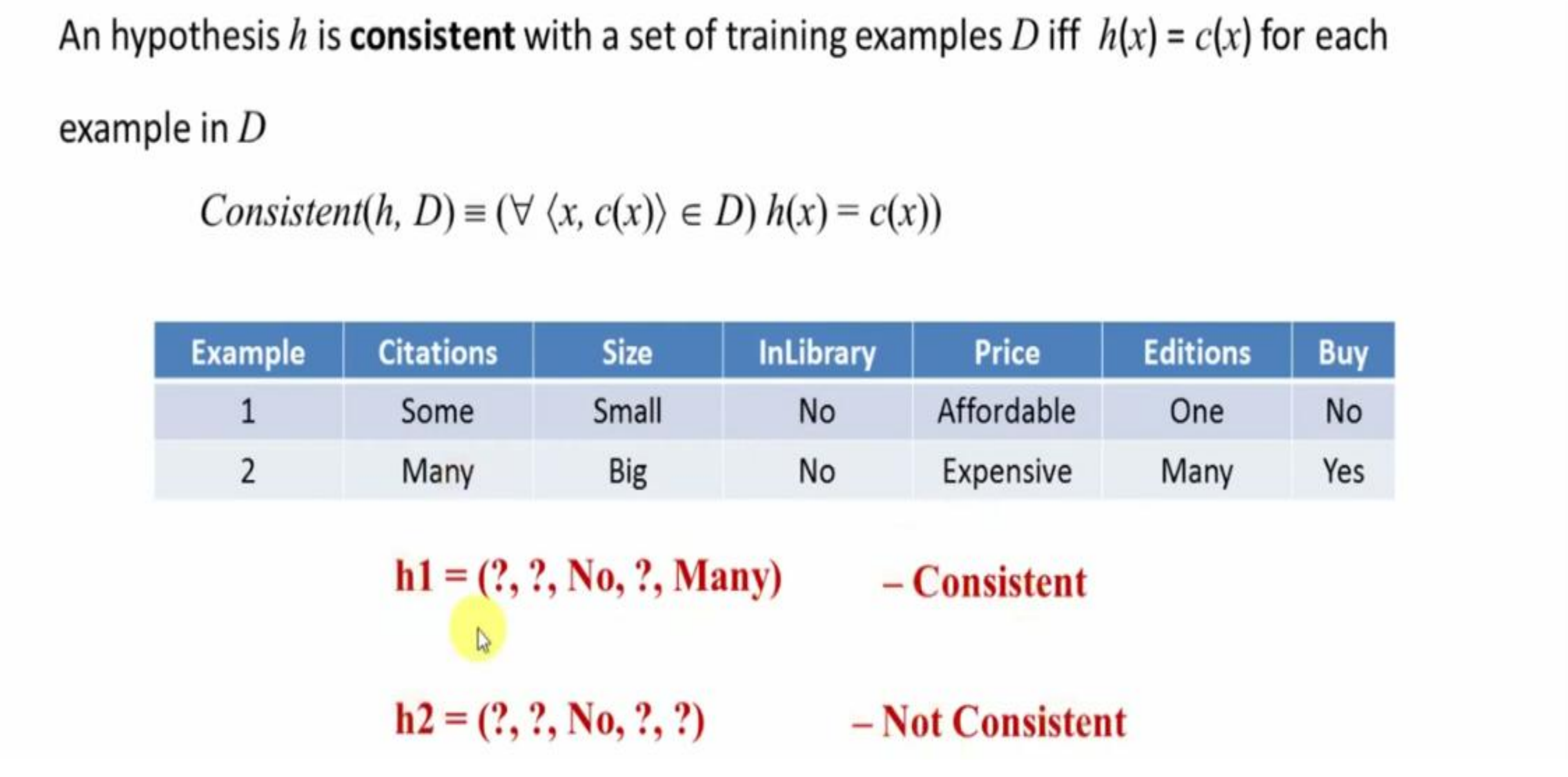
$$VS(H,E) = \{h \in H \mid \forall (x_i, y_i) \in E, h(x_i) = y_i\}$$

Explanation:

- $VS(H,E)$: Version space for hypothesis space H and training examples E.
- $\{h \in H \mid \ldots\}$: Set of hypotheses h in H that meet the condition.
- $\forall (x_i, y_i) \in E$: For all training pairs (x_i, y_i) in E.
- $h(x_i) = y_i$: Hypothesis h correctly predicts output y_i for input x_i.

The version space $VS(H,E)$ includes all hypotheses in H that are consistent with all training examples in E.

Consistent Hypothesis



An hypothesis h is **consistent** with a set of training examples D iff $h(x) = c(x)$ for each example in D

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

Example	Citations	Size	InLibrary	Price	Editions	Buy
1	Some	Small	No	Affordable	One	No
2	Many	Big	No	Expensive	Many	Yes

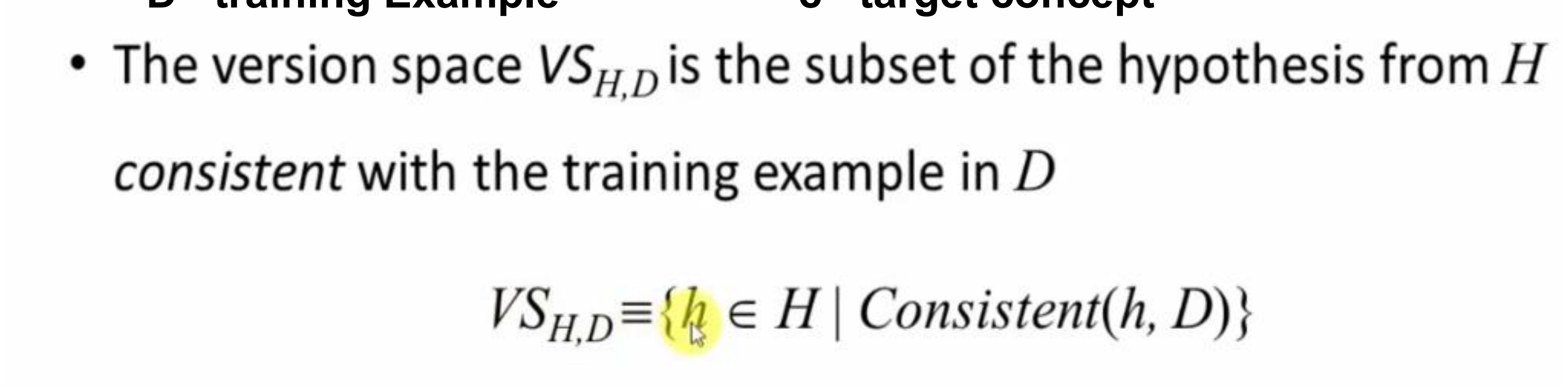
h1 = (?, ?, No, ?, Many) **– Consistent**

h2 = (?, ?, No, ?, ?) **– Not Consistent**

Version Space

- The Candidate-Elimination algorithm represents the set of all hypotheses consistent with the observed training examples.
- This subset of all hypotheses is called the ***version space*** with respect to the hypothesis space H and the training examples D, because it contains all possible versions of the target concept.

- **H= hypothesis space** **Consistent h(x)=c(x)**
  D= training Example **c= target concept**



- The version space $VS_{H,D}$ is the subset of the hypothesis from H *consistent* with the training example in D

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

General and Specific Boundaries

The version space is defined by its **general boundary** (G) and **specific boundary** (S):

1. **General Boundary (G)**:
The set of the most general hypotheses in H that are consistent with the training data. These hypotheses cover as many positive examples as possible without misclassifying negative examples.

2. **Specific Boundary (S)**:
The set of the most specific hypotheses in H that are consistent with the training data. These hypotheses cover only the positive examples and nothing more.

The version space lies between S and G, encompassing all hypotheses that are more general than S and more specific than G.

Candidate Elimination Algorithm

The **Candidate Elimination Algorithm** refines S and G iteratively as new examples are presented. The goal is to narrow the version space until it converges to the target concept.

Steps of the Algorithm

1.Initialization:

1. Start with S as the most specific hypothesis in H.
2. Start with G as the most general hypothesis in H.

For each training example (x, y):

•If y is positive:

- Remove from G any hypothesis that fails to classify x as positive.
- Generalize S as minimally as possible to include x, while ensuring consistency with all prior examples.

•If yyy is

Candidate Elimination Algorithm

•If y is negative:

  •Remove from SSS any hypothesis that incorrectly classifies x as positive.

  •Specialize G as minimally as possible to exclude x, while ensuring consistency with all prior examples.

•**Stop**:

•When S and G converge to a single hypothesis, that hypothesis is the learned target concept.

•If S and G become disjoint, the data is inconsistent.

•If yyy is

Example

Consider a concept learning task with the following hypothesis space:

Feature	Values
Color	Red, Green, Blue
Shape	Circle, Triangle, Square
Size	Small, Medium, Large

Training Examples:

Example	Color	Shape	Size	Class
1	Red	Circle	Small	Positive
2	Green	Circle	Medium	Negative
3	Red	Triangle	Small	Positive

•If yyy is

Example

Step-by-step process:

1. Initialization:

1. S=[Red, Circle, Small] (most specific).
2. G=[?, ?, ?] (most general).

2. Processing Example 2 (Negative):

1. S remains unchanged as it doesn't classify the negative example.
2. G is specialized to exclude [Green, Circle, Medium], yielding:
   1. [Color ≠ Green, ?, ?]
   2. [?, Shape ≠ Circle, ?]
   3. [?, ?, Size ≠ Medium].

3. Processing Example 3 (Positive):

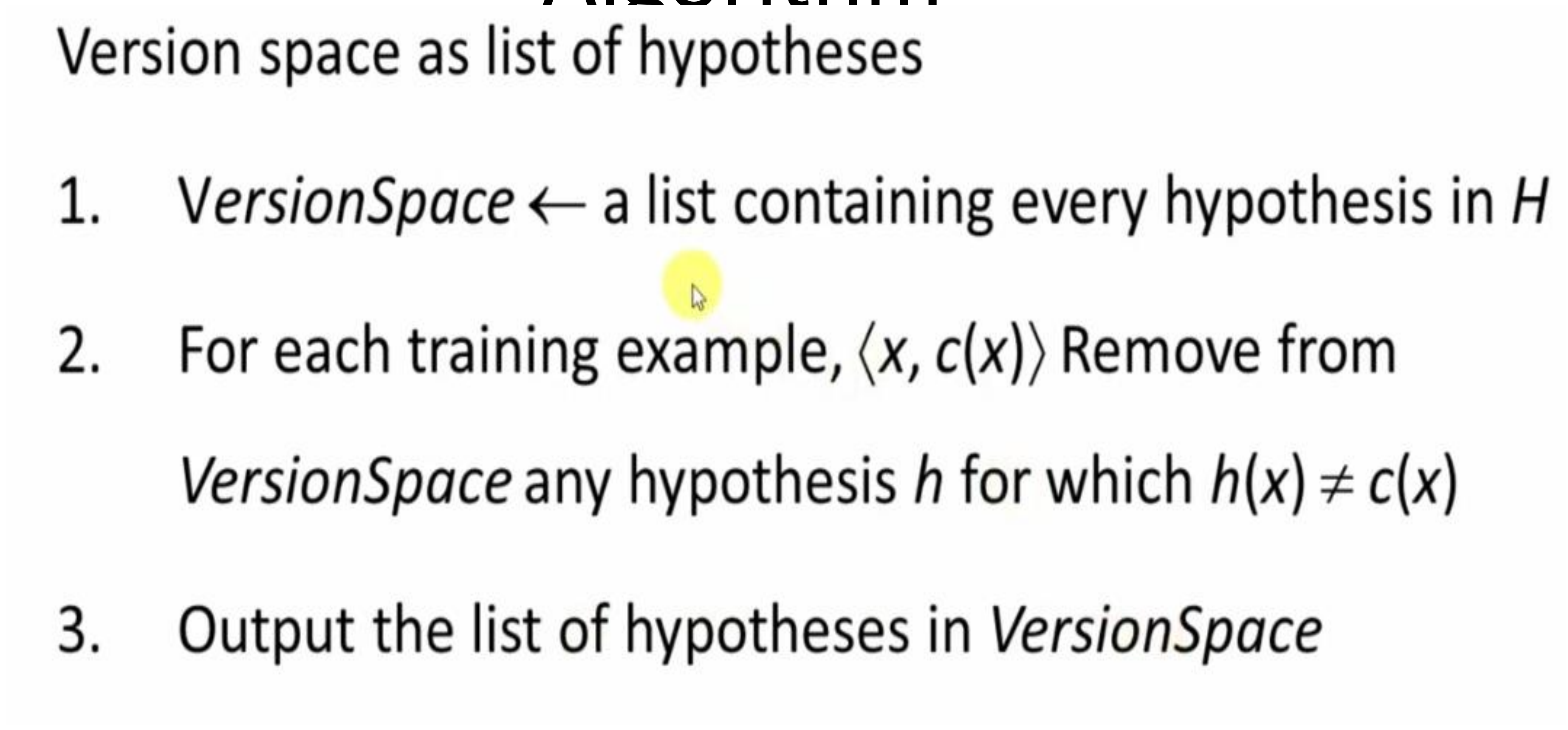
- S is generalized to [Red, ?, Small to include the new positive example.
- G is adjusted for consistency, refining further if necessary.

This process continues until S and G converge.

List-Then-Eliminate Algorithm

- List-Then-Eliminate algorithm initializes the version space to contain all hypotheses in H, then eliminates any hypothesis found inconsistent with any training example.
- The version space of candidate hypotheses thus shrinks as more examples are observed, until ideally just one hypothesis remains that is consistent with all the observed examples.
  - Presumably, this is the desired target concept.
  - If insufficient data is available to narrow the version space to a single hypothesis, then the algorithm can output the entire set of hypotheses consistent with the observed data.
- List-Then-Eliminate algorithm can be applied whenever the hypothesis space H is finite.
  - It has many advantages, including the fact that it is guaranteed to output all hypotheses consistent with the training data.
  - Unfortunately, it requires exhaustively enumerating all hypotheses in H - an unrealistic requirement for all but the most trivial hypothesis spaces.

List-Then-Eliminate Algorithm



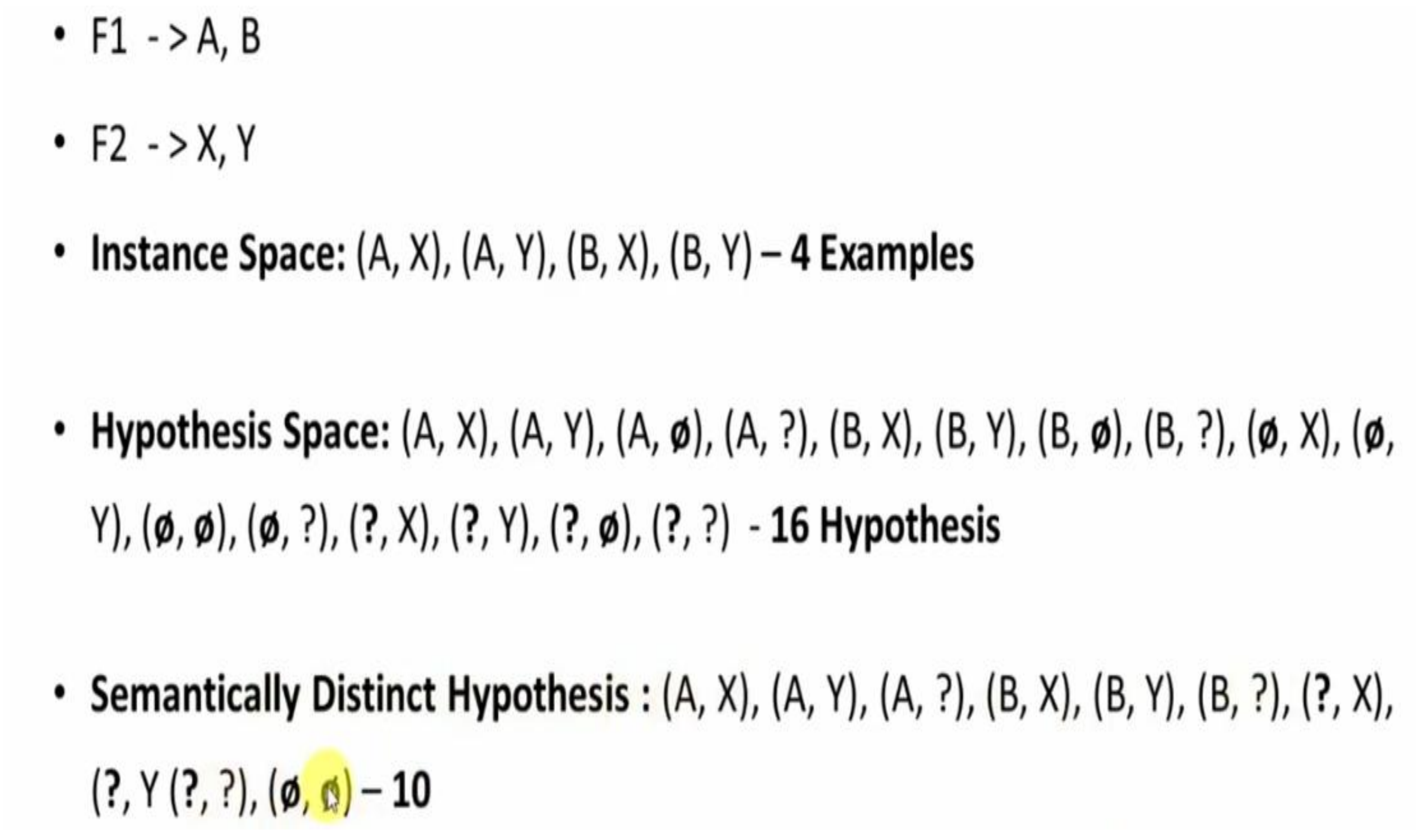
Version space as list of hypotheses

1. *VersionSpace* $\leftarrow$ a list containing every hypothesis in *H*

2. For each training example, $\langle x, c(x) \rangle$ Remove from *VersionSpace* any hypothesis *h* for which $h(x) \neq c(x)$

3. Output the list of hypotheses in *VersionSpace*

Example



- F1 -> A, B
- F2 -> X, Y
- **Instance Space:** (A, X), (A, Y), (B, X), (B, Y) – **4 Examples**
- **Hypothesis Space:** (A, X), (A, Y), (A, ø), (A, ?), (B, X), (B, Y), (B, ø), (B, ?), (ø, X), (ø, Y), (ø, ø), (ø, ?), (?, X), (?, Y), (?, ø), (?, ?) - **16 Hypothesis**
- **Semantically Distinct Hypothesis :** (A, X), (A, Y), (A, ?), (B, X), (B, Y), (B, ?), (?, X), (?, Y (?, ?), (ø, ø) – **10**

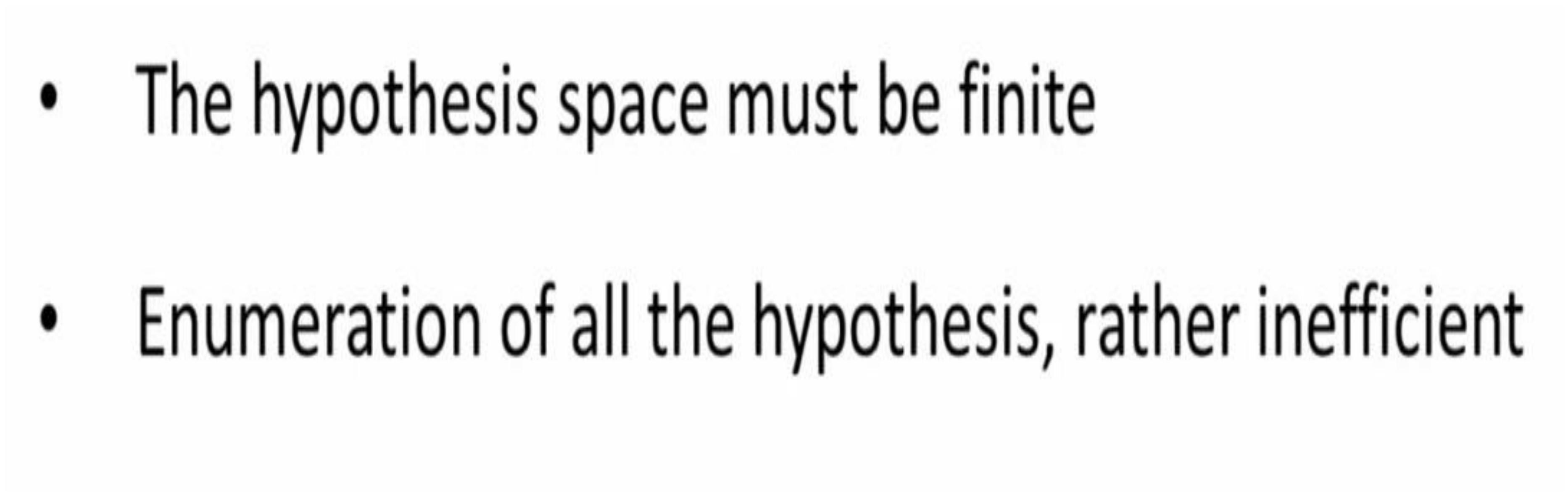
- Version Space: ~~(A, X)~~, ~~(A, Y)~~, (A, ?), ~~(B, X)~~, ~~(B, Y)~~, ~~(B, ?)~~, ~~(?, X)~~, ~~(?, Y)~~ (?, ?), ~~(ø, ø)~~,

- Training Instances

F1	F2	Target
A	X	Yes
A	Y	Yes

- Consistent Hypothesis are: (A, ?), (?, ?)

Problems with List Then Eliminate Algorithm



- The hypothesis space must be finite
- Enumeration of all the hypothesis, rather inefficient

Candidate-Elimination Algorithm

- The Candidate-Elimination algorithm computes the version space containing all hypotheses from H that are consistent with an observed sequence of training examples.
- It begins by initializing the version space to the set of all hypotheses in H; that is, by initializing the G boundary set to contain the most general hypothesis in H

  G0 □ { <?, ?, ?, ?, ?, ?> }

  and initializing the S boundary set to contain the most specific

  hypothesis S0 □ { <0, 0, 0, 0, 0, 0> }
- These two boundary sets delimit the entire hypothesis space, because every other hypothesis in H is both more general than S0 and more specific than G0.
- As each training example is considered, the S and G boundary sets are generalized and specialized, respectively, to eliminate from the version space any hypotheses found inconsistent with the new training example.
- After all examples have been processed, the computed version space contains all the hypotheses consistent with these examples and only these hypotheses.

Candidate-Elimination Algorithm

- Initialize G to the set of maximally general hypotheses in H
- Initialize S to the set of maximally specific hypotheses in H
- For each training example d, do
  - If d is a positive example
    - Remove from G any hypothesis inconsistent with d ,
    - For each hypothesis s in S that is not consistent with d ,-
      - Remove s from S
      - Add to S all minimal generalizations h of s such that
        - » h is consistent with d, and some member of G is more general than h
      - Remove from S any hypothesis that is more general than another hypothesis in S
  - If d is a negative example
    - Remove from S any hypothesis inconsistent with d
    - For each hypothesis g in G that is not consistent with d
      - Remove g from G
      - Add to G all minimal specializations h of g such that
        - » h is consistent with d, and some member of S is more specific than h
      - Remove from G any hypothesis that is less general than another hypothesis in G

Candidate - Elimination Algorithm

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

Initialize the generic and specific boundary

For each training example *d*, do:

If d is ***positive*** example

- Remove from *G* any hypothesis *h* inconsistent with *d*
- For each hypothesis *s* in *S* not consistent with *d*:
  - Remove *s* from *S* ✓
  - Add to *S* all minimal generalizations of *s* consistent with *d*

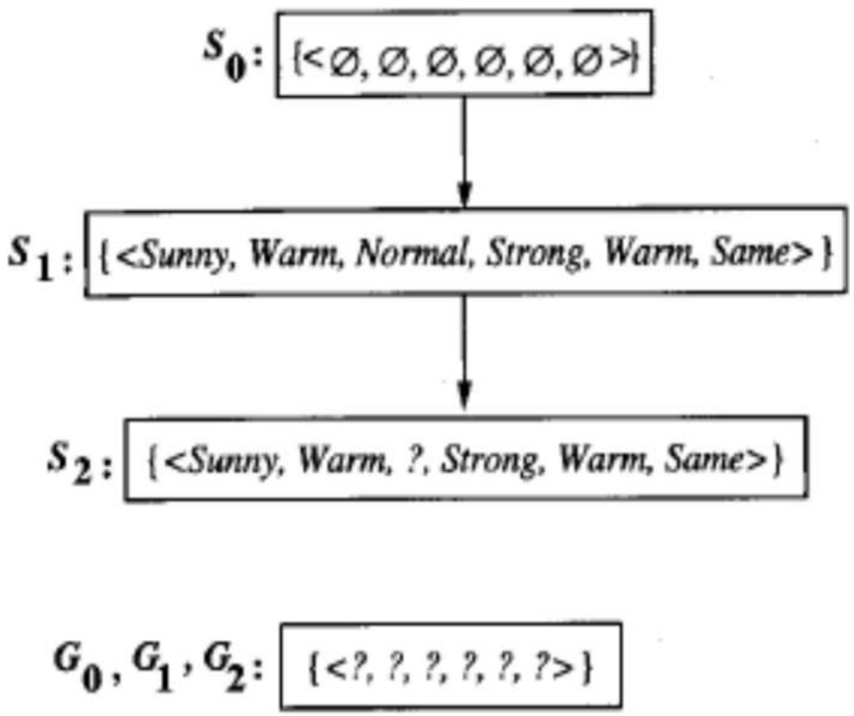
If d is ***negative*** example

- Remove from *S* any hypothesis *h* inconsistent with *d*
- For each hypothesis *g* in *G* not consistent with *d*:
  - Remove *g* from *G* ✓
  - Add to *G* all minimal specializations of *g* consistent with *d*

S_0: $\boxed{\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset. \emptyset \rangle}$

G_0: $\boxed{\langle ?, ?, ?, ?, ?, ? \rangle}$

Candidate-Elimination Algorithm


S_0: {<∅, ∅, ∅, ∅, ∅, ∅>}
S_1: {<Sunny, Warm, Normal, Strong, Warm, Same>}
S_2: {<Sunny, Warm, ?, Strong, Warm, Same>}
G_0, G_1, G_2: {<?, ?, ?, ?, ?, ?>}

•*S0* and *G0* are the initial boundary sets corresponding to the most specific and most general hypotheses.

•Training examples 1 and 2 force the *S* boundary to become more general.

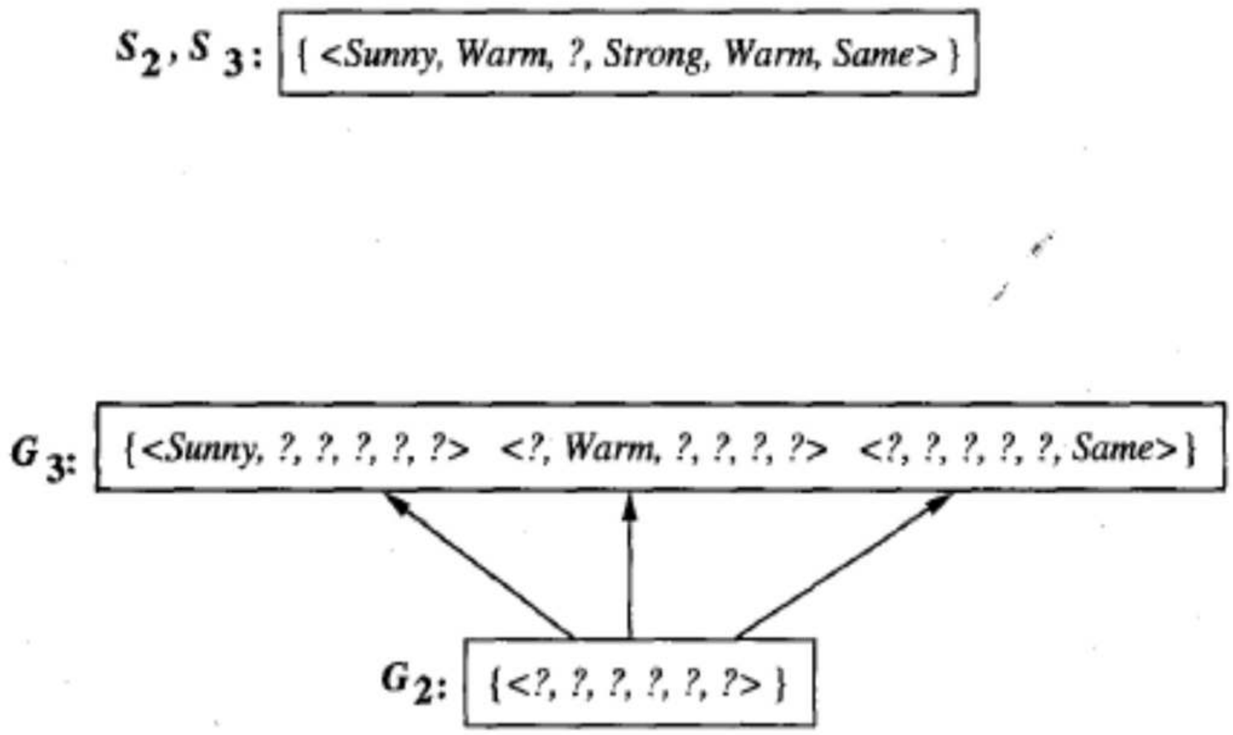
- They have no effect on the *G*

boundary

Training examples:

1. <*Sunny, Warm, Normal, Strong, Warm, Same*>, *Enjoy Sport* = *Yes*
2. <*Sunny, Warm, High, Strong, Warm, Same*>, *Enjoy Sport* = *Yes*

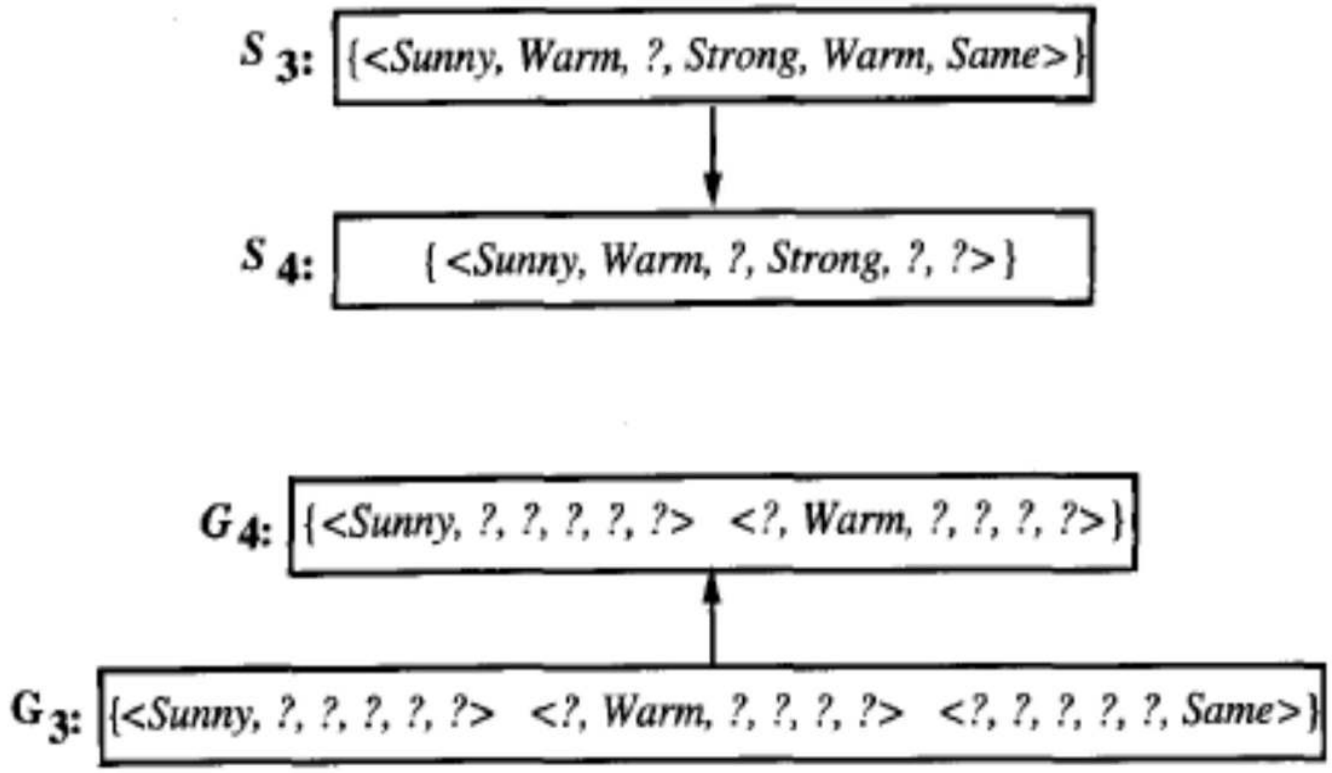
Candidate-Elimination



Training Example:

3. *<Rainy, Cold, High, Strong, Warm, Change>*, *EnjoySport=No*

Candidate-Elimination



Training Example:

4. *<Sunny, Warm, High, Strong, Cool, Change>, EnjoySport = Yes*

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

S_0: ⟨∅, ∅, ∅, ∅, ∅. ∅⟩

S_1: ⟨Sunny,Warm, Normal, Strong, Warm, Same⟩

S_2: S_3: ⟨Sunny,Warm, ?, Strong, Warm, Same⟩

S_4 ⟨Sunny, Warm, ?, Strong, ?, ?⟩

vtupulse.com

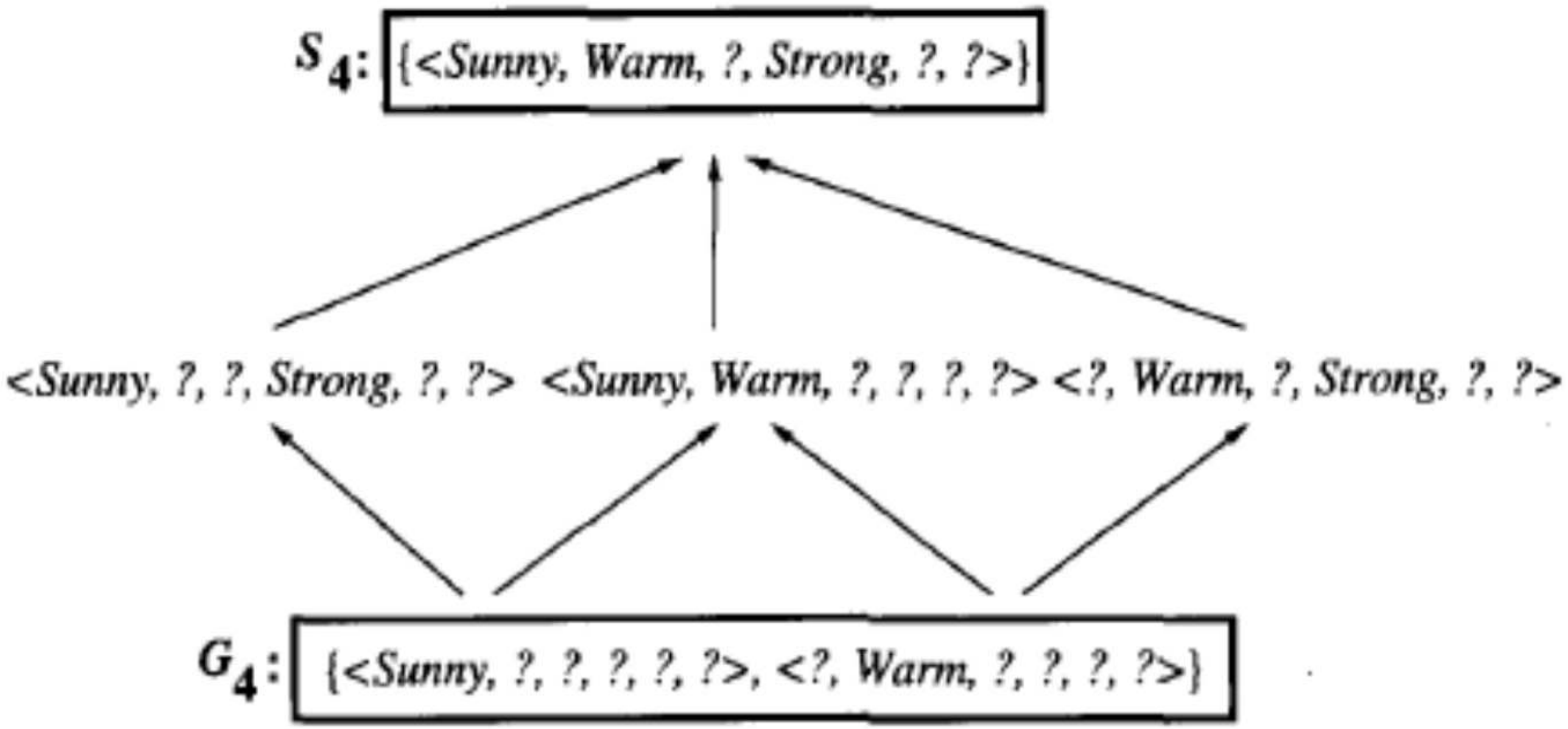
G_4: ⟨Sunny, ?, ?, ?, ?, ?⟩ ⟨?, Warm, ?, ?, ?, ?⟩

G_3: ⟨Sunny,?,?,?,?,?⟩ ⟨?,Warm,?,?,?,?⟩ ⟨?,?,Normal,?,?,?⟩ ⟨?, ?,?,?,Cool,?⟩ ⟨?,?,?,?,?,Same⟩

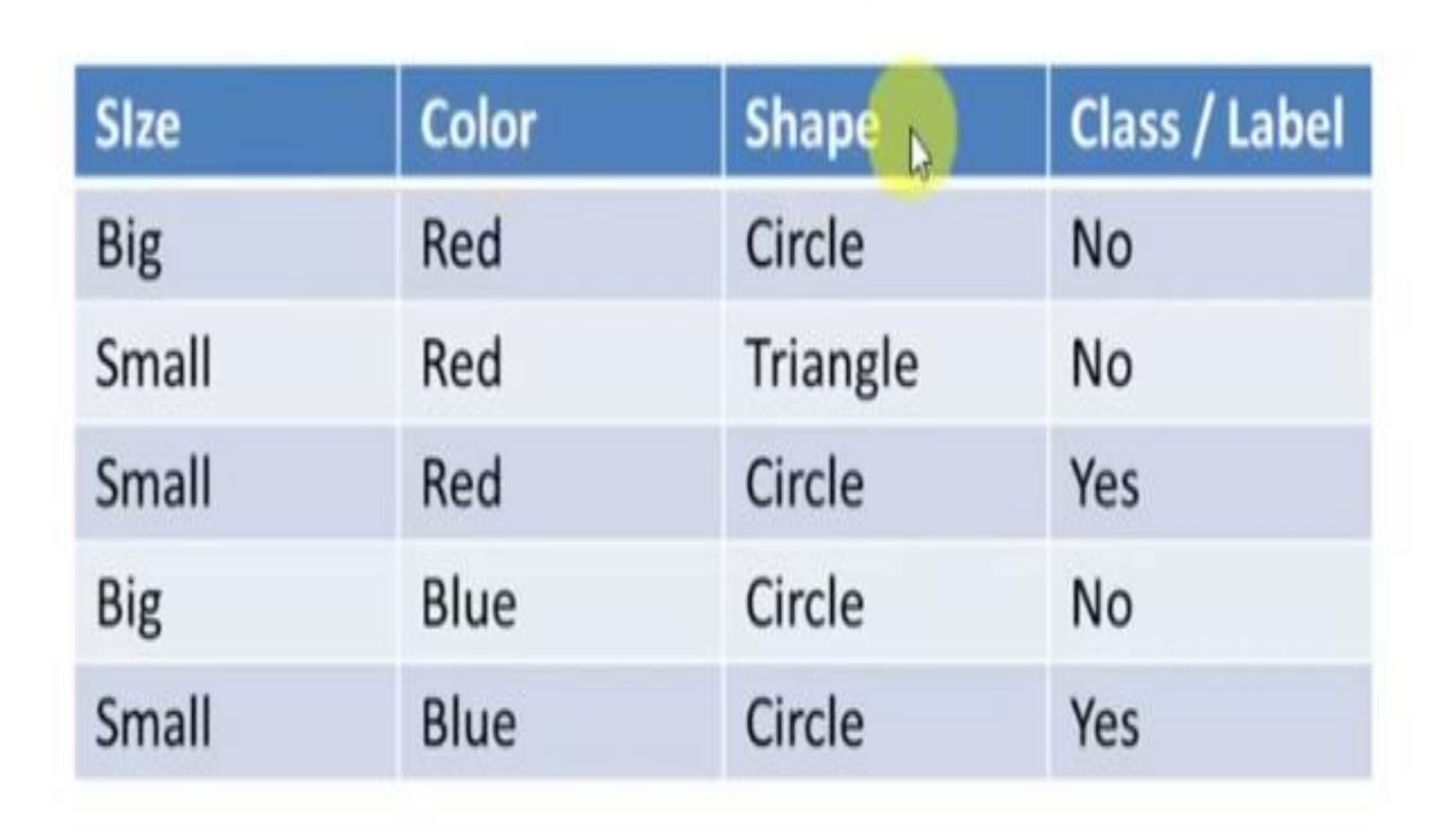
G_0: G_1: G_2: ⟨?, ?, ?, ?, ?, ?⟩ Mahesh Huddar

12:47

Candidate-Elimination Algorithm – Example Final Version Space



Example 2



SIze	Color	Shape	Class / Label
Big	Red	Circle	No
Small	Red	Triangle	No
Small	Red	Circle	Yes
Big	Blue	Circle	No
Small	Blue	Circle	Yes

Candidate Elimination Algorithm Solved Example - 2

SIze	Color	Shape	Class / Label
Big	Red	Circle	No
Small	Red	Triangle	No
Small	Red	Circle	Yes
Big	Blue	Circle	No
Small	Blue	Circle	Yes

S0: (0, 0, 0)

S1: (0, 0, 0)

S2: (0, 0, 0)

S3: (Small, Red, Circle)

S4: (Small, Red, Circle)

S5: (Small, ?, Circle)

S: G: (Small, ?, Circle)

G5: (Small, ?, Circle)

G4: (Small, ?, Circle)

G3: (Small, ?, Circle)

G2: (Small, Blue, ?) (Small, ?, Circle) (?, Blue, ?) (Big, ?, Triangle) (?, Blue, Triangle)

G1: (Small, ?, ?) (?, Blue, ?) (?, ?, Triangle)

G0: (?, ?, ?)

vtupulse.com

•If yyy is

Features and Limitation

Advantages of Candidate Elimination

•Systematically narrows the hypothesis space.

•Guarantees the target concept lies within S and G if data is noise-free.

Limitations

•Computationally expensive for large hypothesis spaces.

•Sensitive to noisy or inconsistent data.

•Requires a finite hypothesis space.

Inductive Bias

Inductive bias refers to the set of assumptions a learning algorithm makes to generalize from the training data to unseen instances. These assumptions guide the algorithm in selecting one hypothesis over others in the hypothesis space, particularly when multiple hypotheses are consistent with the training data.

Without inductive bias, a learning algorithm cannot generalize beyond the given data, as any pattern would be equally plausible.

Importance of Inductive Bias

•**Facilitates Generalization**: Inductive bias helps the model make predictions about unseen data by preferring certain hypotheses.

•**Restricts Hypothesis Space**: It narrows the search space, making learning computationally feasible.

•**Enables Learning with Limited Data**: Strong bias allows models to learn effectively even with a small dataset.

Types of Inductive Bias

•**Restrictive Bias (Strong Bias)**:
•Limits the hypothesis space significantly.
•Examples: Linear models that assume linear relationships between inputs and outputs.

•**Advantages**: Easier to learn; reduces overfitting.

•**Disadvantages**: May fail if the true target concept does not match the bias.

•**Preference Bias (Weak Bias)**:
•Does not eliminate hypotheses but ranks them based on preferences.
•Examples: Neural networks with a preference for simpler (lower capacity) models via regularization.

•**Advantages**: More flexible and capable of learning complex patterns.
•**Disadvantages**: Higher risk of overfitting; computationally more expensive.

Examples of Inductive Bias in Algorithms

- **Linear Regression**:
- Assumes the relationship between features and target is linear.

- **Decision Trees**:
- Assumes that concepts can be represented hierarchically with feature splits.

- **K-Nearest Neighbors (KNN)**:
- Assumes instances that are close in the feature space belong to the same class.

- **Neural Networks**:
- Assumes that the target concept can be approximated by the network's architecture (e.g., layers and activation functions).

- **Support Vector Machines (SVMs)**:
- Assumes that the data is separable by a hyperplane in a transformed feature space.

Trade-offs with Inductive Bias

•**Bias-Variance Tradeoff**:
•Strong inductive bias reduces variance but increases bias, potentially leading to under fitting.
•Weak inductive bias reduces bias but increases variance, potentially leading to overfitting.

•**Flexibility vs. Learnability**:
•Strong bias simplifies the learning process but limits the model's expressiveness.
•Weak bias enhances expressiveness but requires more data to learn effectively.

No Free Lunch Theorem

The **No Free Lunch (NFL) Theorem** states that no single inductive bias is universally superior across all tasks. The performance of a bias depends on how well it aligns with the actual target concept. This underscores the importance of selecting an appropriate algorithm and inductive bias based on the specific problem.

Thank You

Parul® University

www.paruluniversi
ty.ac.in