Subject Name: **Machine Learning**

Subject Code: **MCA-4014**

Subject Topic: **Computational Learning Theory (PAC Learning and VC Dimension)**

Abhishek Dwivedi

Assistant Professor

Department of Computer Application

UIET, CSJM University, Kanpur

Computational Learning Theory

- Computational learning theory, or *CoLT* for short, is a field of study concerned with the use of formal mathematical methods applied to learning systems.
- It seeks to use the tools of theoretical computer science to quantify learning problems. This includes characterizing the difficulty of learning specific tasks.
- Computational learning theory may be thought of as an extension or sibling of statistical learning theory, or *SLT* for short, that uses formal methods to quantify learning algorithms.
- **Computational Learning Theory** (*CoLT*): Formal study of learning tasks.
- **Statistical Learning Theory** (*SLT*): Formal study of learning algorithms.

- This division of learning tasks vs. learning algorithms is arbitrary, and in practice, there is a lot of overlap between the two fields.
- *One can extend statistical learning theory by taking computational complexity of the learner into account. This field is called computational learning theory or COLT.*
- The focus in computational learning theory is typically on supervised learning tasks. Formal analysis of real problems and real algorithms is very challenging. As such, it is common to reduce the complexity of the analysis by focusing on binary classification tasks and even simple binary rule-based systems. As such, the practical application of the theorems may be limited or challenging to interpret for real problems and algorithms.
- *The main unanswered question in learning is this: How can we be sure that our learning algorithm has produced a hypothesis that will predict the correct value for previously unseen inputs?*

Questions explored in computational learning theory might include:

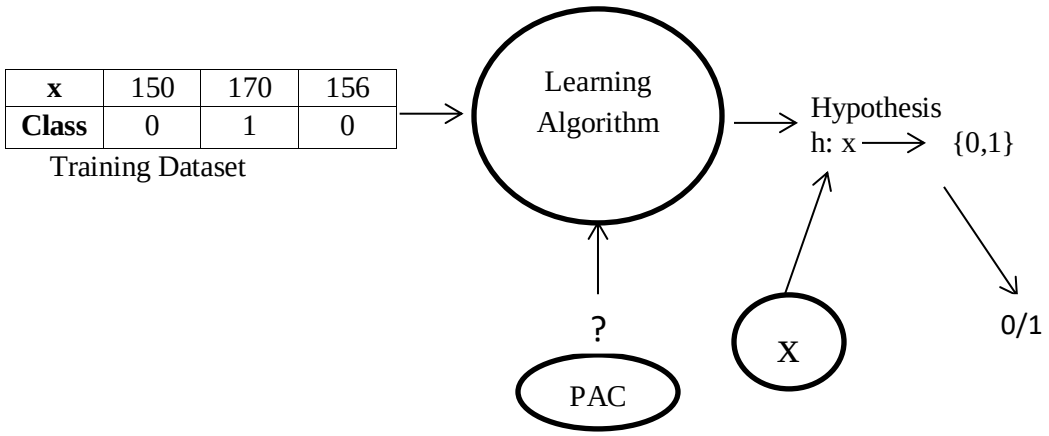
- How do we know a model has a good approximation for the target function?
- What hypothesis space should be used?
- How do we know if we have a local or globally good solution?
- How do we avoid overfitting?
- How many data examples are needed?

As a machine learning practitioner, it can be useful to know about computational learning theory and some of the main areas of investigation. The field provides a useful grounding for what we are trying to achieve when fitting models on data, and it may provide insight into the methods.

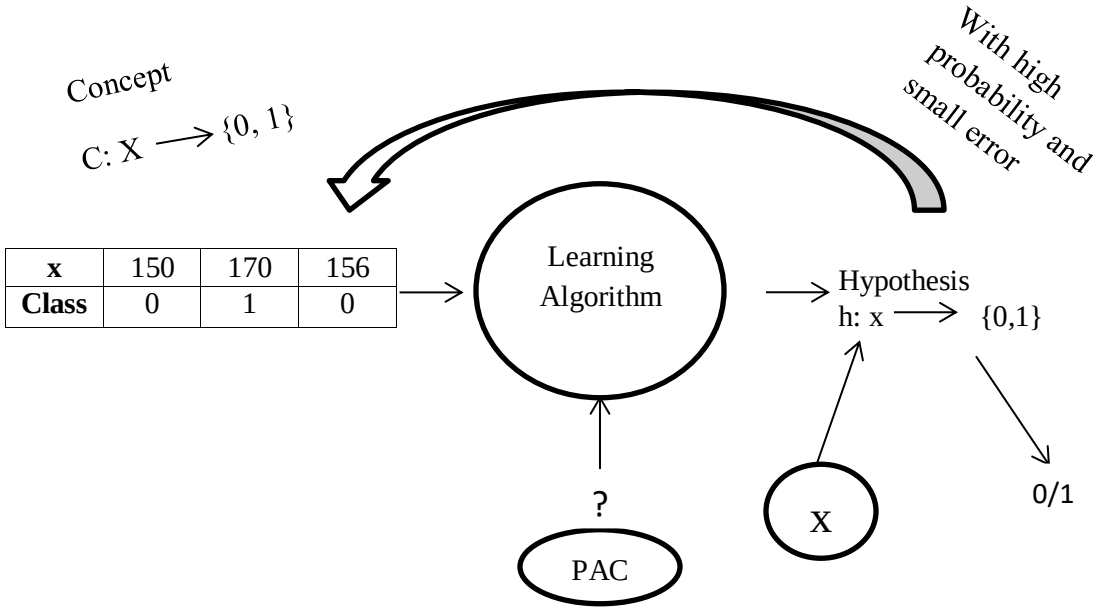
The most widely discussed areas of study from computational learning theory are:

- PAC Learning.
- VC Dimension.

Probably Approximately Correct Learning (PAC)



- PAC is a framework for analyzing learning algorithms mathematically
- With high Probability, PAC Learning algorithms find a hypothesis that is approximately identical to the hidden target concept.



Notation:-

- Instance Space, X
- Concept Class, C – Family of functions, c

$$c: X \longrightarrow \{0,1\}$$

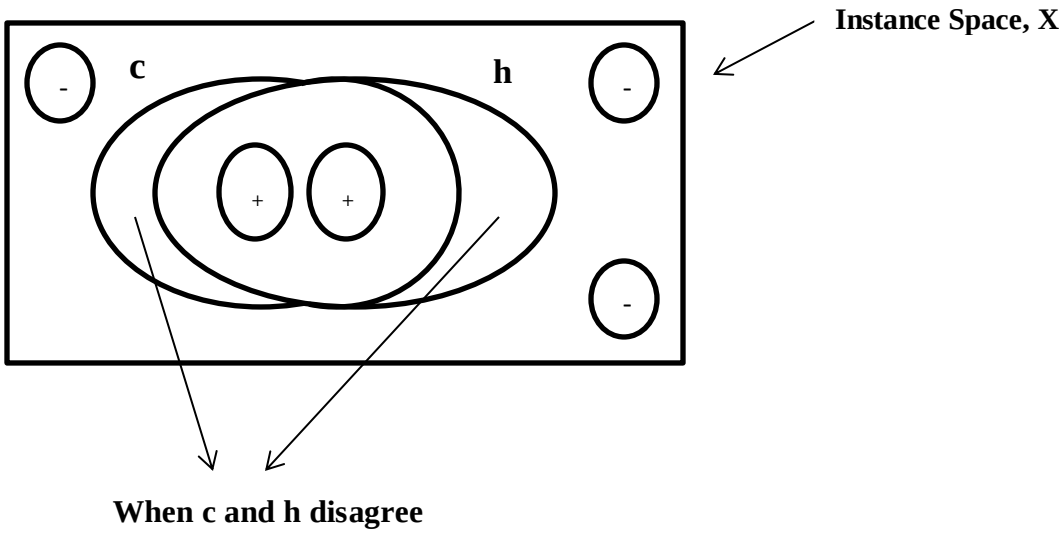
- Hypothesis Class, H – Family of functions, h

$$h: X \longrightarrow \{0,1\}$$

- Probability Distribution, F – Training Samples are generated randomly from X according to F
- Learning Algorithm, A

Definition:- A concept class C is PAC- learnable, if there is an algorithm A which for samples drawn with any probability distribution F and any concept c ϵ C, with high probability produce a hypothesis h ϵ H, whose error is small.

True Error of h:-



The true error of h w.r.t target concept C and distribution F,

$Error_{\mathbf{F}}(h) = P_{\mathbf{x \epsilon F}}(h(x) \neq c(x))$

Length (Dimension) of instance $\longrightarrow$ Dimensions of data (n)

Size of a concept: - Size of concept is the number of parameters used to represent a concept.

Suppose consider a concept like

Axis aligned rectangle is represented by 4 parameters ($a \leq x \leq b$) and ($c \leq y \leq d$) so the size of concept is 4.

Formal Definition:- A concept class C is said to be PAC learnable by algorithm L if for all $c \epsilon C$, distribution F over X, ε such that $0 < \varepsilon < ½$ and δ such that $0 < \delta < ½$, the learner, L will output a hypothesis h with probability at least $(1- \delta)$ and with $Error_{\mathbf{F}}(h) \leq \varepsilon$.

$P > 1- \delta$ and error $\leq \varepsilon$

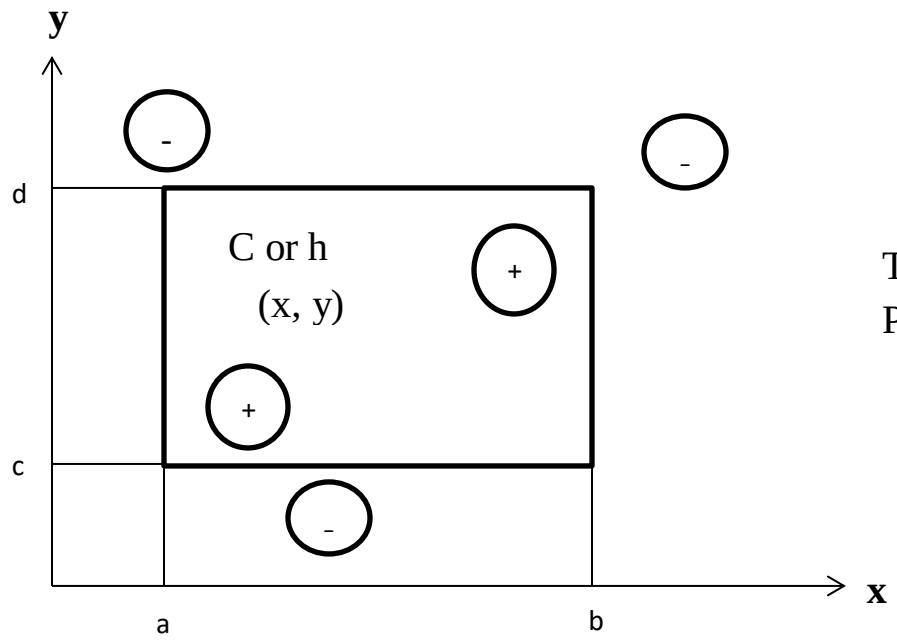
Example: - **Instance Space X** – set of all points (x, y) in a plane.

Length of instance – 2

Concept class C – Set of all axis aligned rectangle of the form $(a \leq x \leq b)$ and $(c \leq y \leq d)$, there are 4 parameters (a, b, c, d).

So **size of concept** – 4

Hypothesis Space H – Set of all hypothesis to be equal to the set C of concepts.



This Problem is a PAC learnable

VC Dimension (Theory of Learning Algorithms)

- Vapnik–Chervonenkis theory, or VC theory for short, refers to a theoretical machine learning framework developed by Vladimir Vapnik and Alexey Chervonenkis.
- VC theory learning seeks to quantify the capability of a learning algorithm and might be considered the premier sub-field of statistical learning theory.
- VC theory is comprised of many elements, most notably the VC dimension.
- The VC dimension quantifies the complexity of a hypothesis space, e.g. the models that could be fit given a representation and learning algorithm.
- One way to consider the complexity of a hypothesis space (space of models that could be fit) is based on the number of distinct hypotheses it contains and perhaps how the space might be navigated. The VC dimension is a clever approach that instead measures the number of examples from the target problem that can be discriminated by hypotheses in the space.
- *The VC dimension measures the complexity of the hypothesis space [...] by the number of distinct instances from X that can be completely discriminated using H*

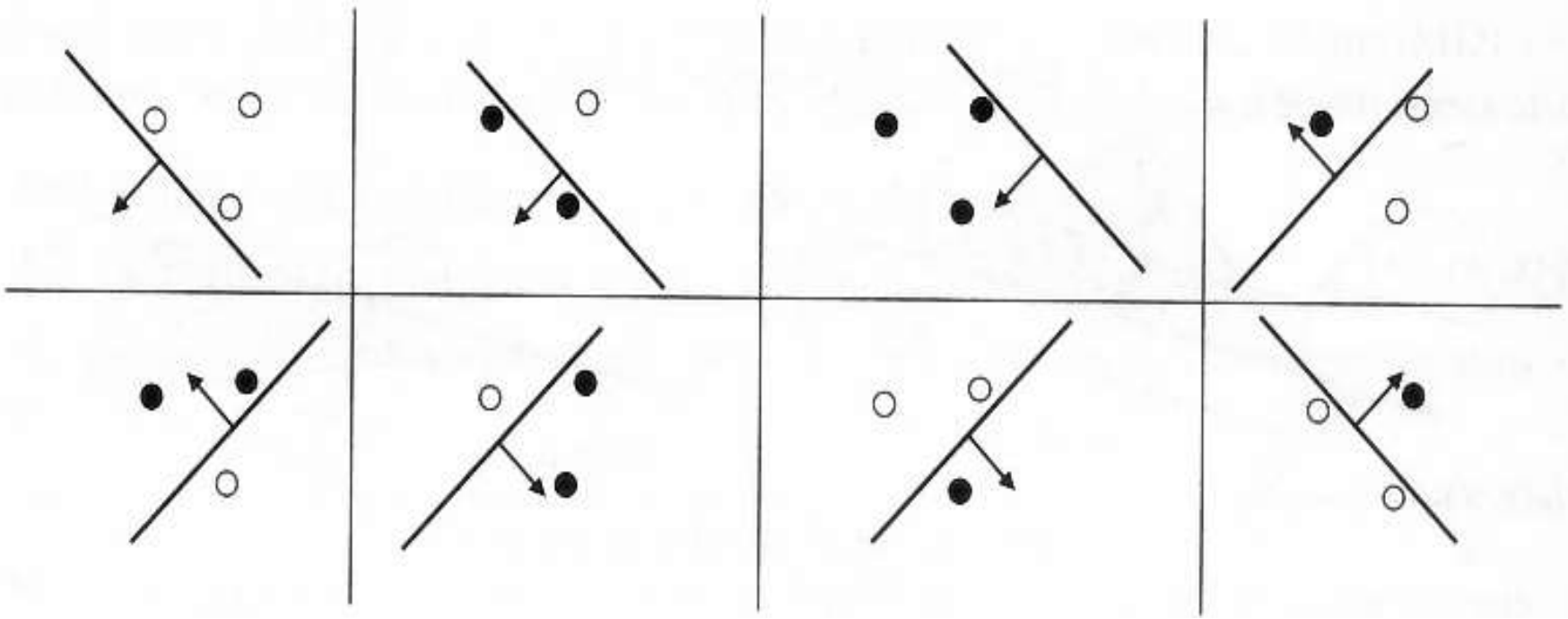
- The VC dimension estimates the capability or capacity of a classification machine learning algorithm for a specific dataset (number and dimensionality of examples).
- Formally, the VC dimension is the largest number of examples from the training dataset that the space of hypothesis from the algorithm can "*shatter*."
- *The VC dimension, VC(H), of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H.*

- The VC dimension of a classifier is defined by Vapnik and Chervonenkis to be the cardinality (size) of the largest set of points that the classification algorithm can **shatter** . This may seem like a simple definition, but it is easy to misinterpret, so I will now go into more detail here and explain the key terms in the definition. We will use 2-D examples for simplicity, but these ideas generalize to any number of dimensions.

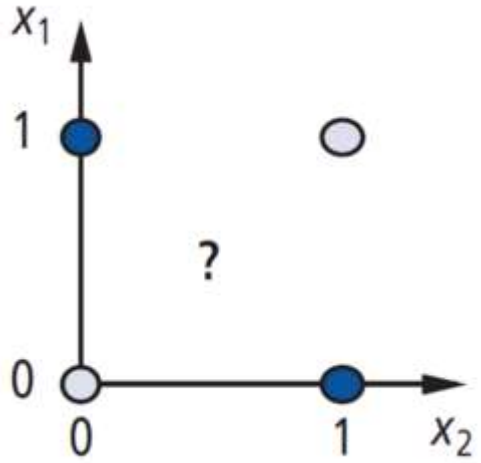
Shattering a set of points

- A configuration of **N** points on the plane is just any placement of **N** points. In order to have a VC dimension of *at least* **N**, a classifier must be able to shatter a *single* configuration of **N** points. In order to **shatter** a configuration of points, the classifier must be able to, for every possible **assignment** of positive and negative for the points, perfectly partition the plane such that the positive points are separated from the negative points. For a configuration of **N** points, there are **2^N** possible assignments of positive or negative, so the classifier must be able to properly separate the points in each of these.

- In the below example, we show that the VC dimension for a linear classifier is *at least* 3, since it can shatter this configuration of 3 points. In each of the $2^3 = 8$ possible assignment of positive and negative, the classifier is able to perfectly separate the two classes.



- Now, we show that a linear classifier is *lower than* 4. In this configuration of 4 points, the classifier is unable to segment the positive and negative classes in at least one assignment. Two lines would be necessary to separate the two classes in this situation. We actually need to prove that there *does not exist* a 4 point configuration that can be shattered, but the same logic applies to other configurations.



- Since we have now shown that the linear classifier's VC dimension is *at least* 3, and *lower than* 4, we can finally conclude that its VC dimension is *exactly* 3. Again, remember that in order to have a VC dimension of **N**, the classifier must only shatter *a single* configuration of **N** points.