MACHINE LEARNING CS60050 - part 1

Scribed by: Moumita Mishra 20ID92R01

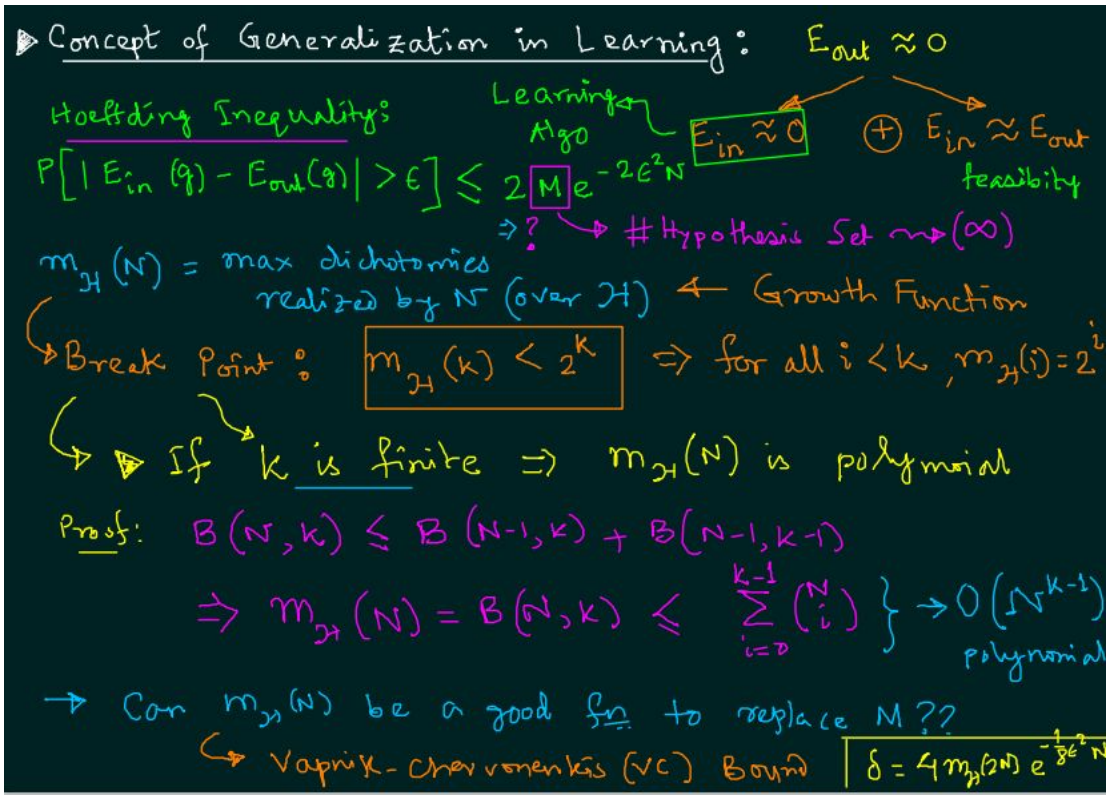
Note for class on Feb 24 2021

Concept of generalization of learning:

Generalization refers to your model's ability to adapt properly to new, previously unseen data, drawn from the same distribution as the one used to create the model.

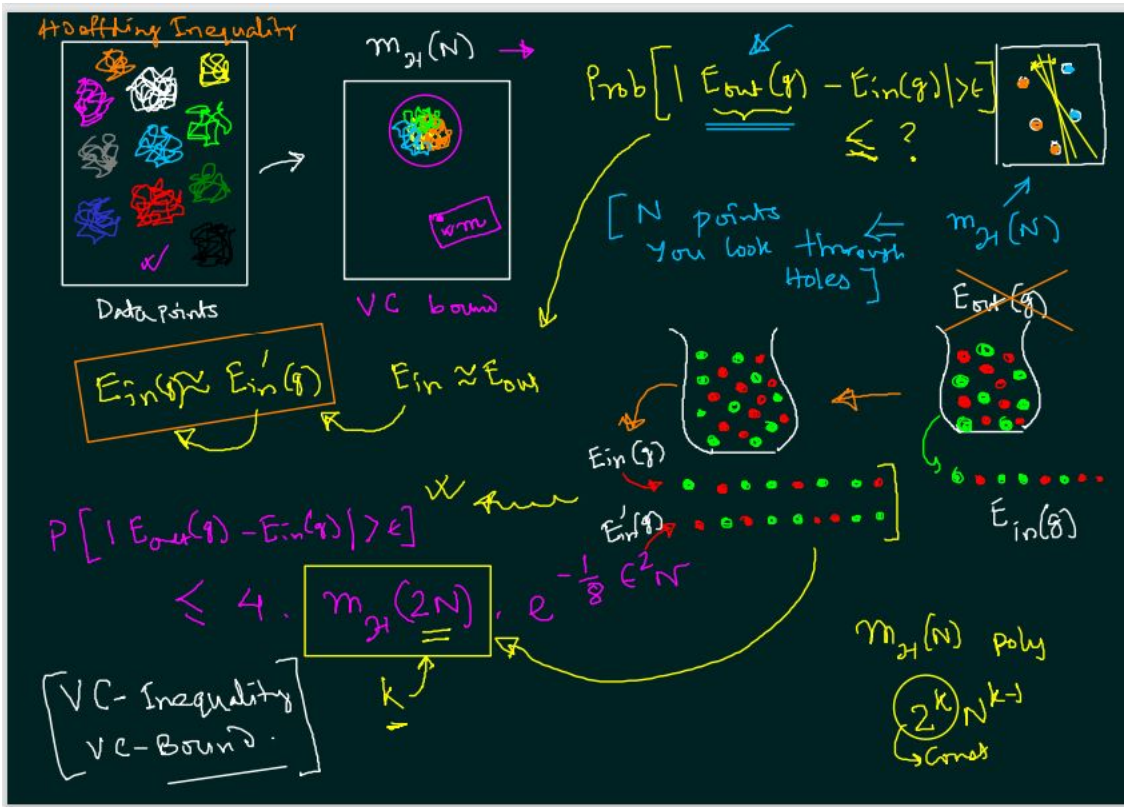
Hoeffding's inequality is a powerful technique—perhaps the most important inequality in learning theory—for bounding the probability that sums of bounded random variables are too large or too small. We will state the inequality, and then we will prove a weakened version of it based on our moment generating function calculations earlier

To qualify the relationship between generalization error and empirical error, we use a bound for the difference of both items. Therefore we use Hoeffding's inequality, one of the most powerful tools in learning theory:



Hoeffding's inequality belongs to the concentration inequalities, that give probabilistic bounds for a random variable to be concentrated around its mean. The intuition is as follows: Suppose, we have some random variables. We know, that if we take the average, we usually should get something close to the expectation. How can we apply this to our learning problem. The Hoeffding Inequality provides a way to characterize the discrepancy between Ein and Eout. Suppose we only have one possible hypothesis h.

Immediately from substituting Ein and Eout, we obtain Learning bounds for infinite hypothesis sets We have derivated bounds for finite hypothesis sets. But in machine learning, the hypothesis sets are usually infinite. Consider for example all hypotheses that are parameterized with real numbers, for example the Perceptron or the SVM. There are infinitely many hyperplanes in spaces of whatever dimension. In example we saw, that we can learn from a finite set of training examples, even when the hypothesis set is infinite. In this case, however the bound in formula will go to infinity. To do this, we need a less generous estimation for |Ein − Eout| than the union bound. We will show that many of the hypotheses in H are similar, meaning that the "bad" regions with |Ein(hi) − Eout(hi)| ≥ are in fact often overlapping and so the generalization error can be bound by a smaller term.



Growth Function The idea will be the following: We show that the hypotheses in a hypothesis set H can be "similar" to each other and therefore their "bad events" with poor generalization can overlap. Therefore, we define the growth function, that formalizes the number of "effective" hypotheses in a hypothesis set. It tells us how many different hypotheses that H can yield, if one only considers a finite sample of points x1, ..., xn. Through the growth function, we finally want to replace the factor |H| in the generalization bound 4.2.6. The smaller the growth function, the more similar the hypotheses in H are and the more overlap we have. We will again focus on binary classification here, that is on hypotheses that map X to Y = {−1, 1}.

VC Dimension We now derive an upper bound for the growth function $\Pi H(m)$, for all $m \in N$.For proving the polynomial bound, we define a new combinatorial quantity, the VC dimension. The VC (Vapnik-Chervonenkis) dimension is a single parameter that characterizes the growth function.

VC-Dimension: $d_{vc}(\mathcal{H}) = k - 1$

↳ largest N that can get SHATTERED by $\mathcal{H}$

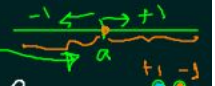
→ $N \leq d_{vc}(\mathcal{H}) \Rightarrow \mathcal{H}$ can SHATTER N points

→ $k > d_{vc}(\mathcal{H}) \Rightarrow k$ is a break point for $\mathcal{H}$.

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{k-1} \binom{N}{i} = \sum_{i=0}^{d_{vc}} \binom{N}{i} \rightsquigarrow O(N^{d_{vc}})$$

↳ Ex:

① Positive Rays: $d_{vc} = 1$ $k = 2$



② Positive Interval: $d_{vc} = 2$ $k = 3$

→ ③ 2D-perceptrons: $d_{vc} = 3$ $k = 4$

④ Convex Set: $d_{vc} = \infty$ $k = \infty$

$g \approx f \Rightarrow g$ generalizes well

MACHINE LEARNING CS60050 - part 2

Scribed by: Debayan Bhattacharya 20ET91R02

Note for class on Feb 24 2021

1. Positive Rays: $d_{vc} = 1, k = 2$
2. Positive Interval: $d_{vc} = 2, k = 3$
3. 2D-perceptrons: $d_{vc} = 3, k = 4$
4. Convex set: $d_{vc} = \infty, k = \infty$

Therefore, from learning diagram, $g \approx f \implies$ g generalizes well. It is independent of the learning algorithm. It is only dependent on the hypothesis set because it is a perceptron and it is dependent on the final hypothesis that it is going to produce.

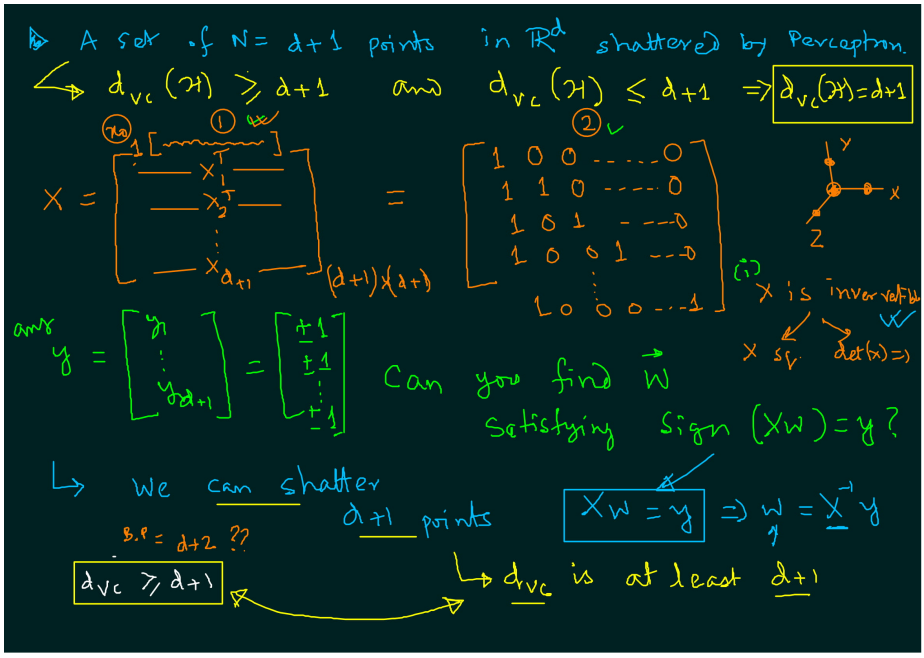
Now an interesting thing I am going to explain. Look at 2D perceptron. See that the VC dimension is 3. I will just try to prove for 2D perceptron in d-dimensions the VC dimension will be $d + 1$.

A set of $N = d + 1$ points in d-dimensional space R^d can be shattered by perceptrons. If I can prove these two $d_{VC}(H) \geq d + 1$ and $d_{VC}(H) \leq d + 1$. These two implies the fact that this should be equal to $d + 1$ that means $N = d + 1$ points can be shattered only but not more than that. Beyond which it is a break point. So, this will give $\boxed{d_{VC}(H) = d + 1}$ which I want to prove.

Let us say that I have N points as you can know that N points in d dimension is a matrix of order $(d+1) \times (d+1)$. Suppose this is my input matrix. Now, I will take these points to most generic way possible. At first point everyone is zero. Try to think about in 3D. Then I will take $X = 1$, the other ones all are zero. Then I will take $Y = 1$ and other ones are zero. Then I will take $Z = 1$ and the other ones are zero. This is the most generic setting I can make with all my $d + 1$ points. The important point to look here or the fact that I will use in my proof is that this $matrix X$ is invertible. It is invertible because X is a square matrix. It is invertible because $det(X) = 1$.

Now the next thing is that: Can we shatter these set of $d + 1$ points?
For any y we choose, can you find a vector w satisfying $sign(Xw) = y$? Then I say that all these points are shattered because for any arbitrary choice of y if I can find a w that can produce that y then all of these points can be labelled

arbitrarily, all possible dichotomies are possible. So it can be shattered. So this is very easy. Instead of taking the sign to be satisfied, I just make $Xw = y$. So my w is $w = X^{-1}y$. Why am I making this one? Instead of sign, if I make Xw itself equal to $+1$ or -1 then their sign will also match. That means we can shatter $d + 1$ points.



So now my trivia is that what does it prove?

A. Does it prove that $d_{VC} = d + 1$

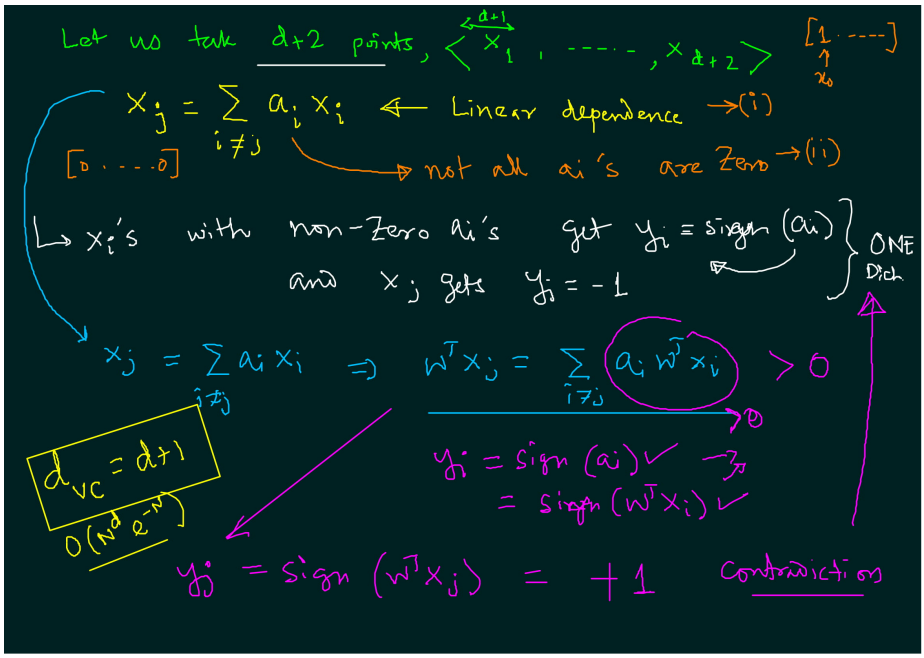
B. Does it prove $d_{VC} \geq d + 1$

C. Does it prove $d_{VC} \leq d + 1$ or,

D. Does it prove nothing? No conclusion?

If you can shatter $d + 1$ points, then d_{VC} is atleast $d + 1$ because any other points less than $d + 1$ you can also shutter as per my VC dimension definition i.e. if we can shutter $d + 1$ point, then d point we can shatter, $d - 1$ point we can shatter, $d - 2$ point we can shatter. So d_{VC} is atleast $d + 1$ in that case. So we have proved B. So we have proved $d_{VC}(H) \geq d + 1$.

We need to prove that $d + 2$ is a break point. That means I need to prove that we can not shatter $d + 2$ number of points. Let us take $d + 2$ points.



My claim is that I have to prove that not all possible combinations of this dichotomies are possible with these $d+2$ points. One immediate observation is that we now have more number of points than the number of dimensions. Therefore, for a particular j we must have a linear dependence with respect to all other $d+1$ number of $X_i s$.

$$X_j = \sum_{i \neq j} a_i X_i \qquad Linear Dependence$$

It is just like telling that suppose I give you 3 equations with X and $Y s$. Definitely the 3rd equation is a linear combination of other two equations.

Second factor: Not all $a_i s$ are zero. Why? Because if all of $X_i s$ are zero then X_j will produce a All-Zero vector.That is not possible. So some a_i should be 1. So, not all a_i's are zero.

When I say I could not realize all dichotomy with $d+1$ points, I will give one counter example dichotomy. That will suffice.

So consider the following. Let's say $X_i s$ with non-zero $a_i s$ get $y_i = sign(a_i)$ and Xj gets $Y_j = -1$. This is ONE dichotomy. Can I construct this? I am trying to show you that one such dichotomy, even if I wish to construct, I will be prevented. So, that means all possible dichotomies are not realizable.

$$w^T X_j = \sum_{i \neq j} a_i w^T X_i$$

My dichotomy suggested my $y_i = sign(a_i)$. Perceptron says $y_i = sign(w^T X_i)$. Therefore the signs should match. This means $Sum(a_i w^T X_i) > 0$. So $y_j = sign(w^T X_j) = +1$ leading to a contradiction w.rt. the dichotomy which we have assumed. Therefore not all possible dichotomies we can realize with $d+2$ points. So finally we reach that a d-D percentron has an $d+1$ VC dimension. It is an enormous result because I can now say that I am bounded with my generalisation boundary with an order of $N_d e^{-N}$.

Therefore I can say that I am bounded with my generalization by that factor though I can take infinitely possible hypothesis and this d is not large as we see in 2D percptron it is 4 and in $d-D$ it is $d+1$ only and it should be the case because percentrons are very simple model to get realized. That's why it should not be very very generic.

Intuitively what does d_{VC} suggest? One thing is the fact that it suggests the **Degrees of freedom**. First of all you have an analog set of knobs or weights in the perceptrons which you made it in terms of binary kind of a $d+1$ notion. Thats why we could reduce the generalization by a very succinct pattern.
So one thing is that your degrees of freedom is infinite because of its analog nature of weights. You reduce it with an equivalent number of binary degrees of freedom.

The second thing: what indeed it suggest? What are your number of attributes or parameter that you learn?

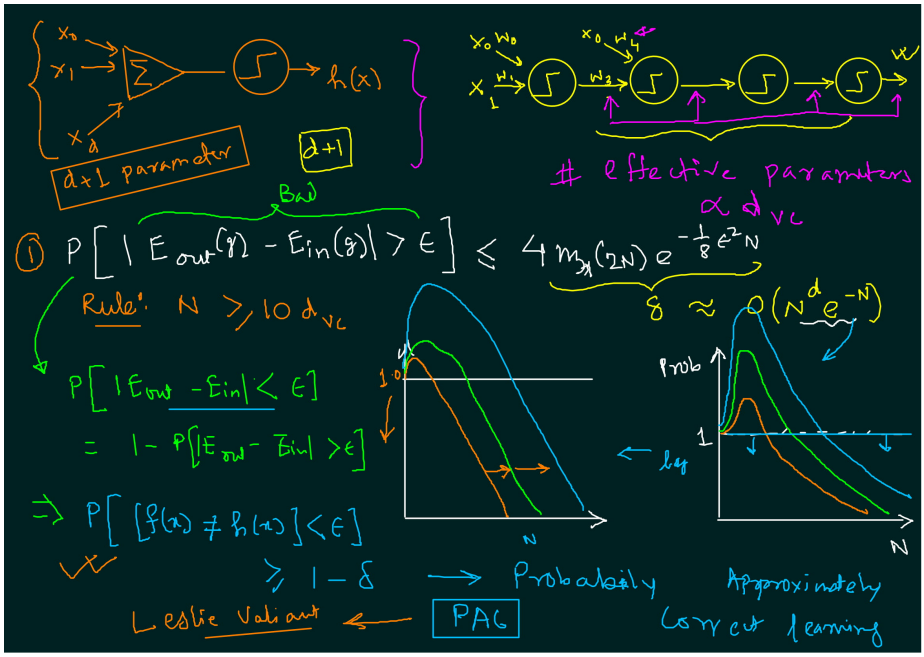
Here it is 2 because I want to learn a range from a to b. Number of parameters have an impact on VC dimensions. Similarly here you see that it is 3 or rather in general case that it is d+1.

Now if you try to consider the figure of perceptron, you can understand why it is $d+1$.
You can see here this is $X_0, X_1, X_2, ..X_d$ and it is dictating my perceptron output which is obviously $h(X)$ and you can see there are $d+1$ parameters. So, definitely VC dimension has a significance in terms of the freedom that we give to make a decision i.e. the number of parameters in our hand. It may not be the case that it is directly related to the set of parameters

Let's say I cascade the perceptron one after another as we do in neural networks, instead of the logistic units I cascade those. You may easily argue that even if it is a single input w.r.t the mandatory X_0, you can easily see here the number of parameters are exponential. So, the number of parameters are expo-

nential. However the outcome that we will get does not vary when we get an outcome given in the model.

Beyond the boundary of one perceptron, if you cascade it without knowing anything like a non-ML expert then it is not the number of parameters matter in the VC dimension but it is the number of effective parameters which is actually proportional to the VC dimension.



Now look at the fact. Sometimes we call this as δ or the confidence that we can bound it or generalize it and you can see that this δ term is nothing but order of $\mathcal{O}(N_d e^{-N})$.

Now when your N is small, w.r.t the d, it rises up and then it comes down. If you look at the nature of the curve, it looks like this in the figure. If it's VC dimension is more, then the curve is more steeper and e^{-N} takes over and it diminishes. But our range of importance is below this 1. More than 1 probability, we do not care to learn in ML course. So, what we will do, we try to make this scale as log scale and see the lower part to blow up.

So the increment of d_{VC} and N, the rule of thumb is to take $N \geq 10 d_{VC}$ so that you can get as less possible with this 1.0 probability line. This is a rule of thumb to get a meaningful generalization.

The other thing: looking at the equation, I can reconstruct the equation. L.H.S. is probability of something bad is happening. So, now how good I am generalizing? It is nothing but the following equation.

$$\mathcal{P}[|E_{out} - E_{in}| < \epsilon] = 1 - \mathcal{P}[|E_{out} - E_{in} > \epsilon]$$

This will give us

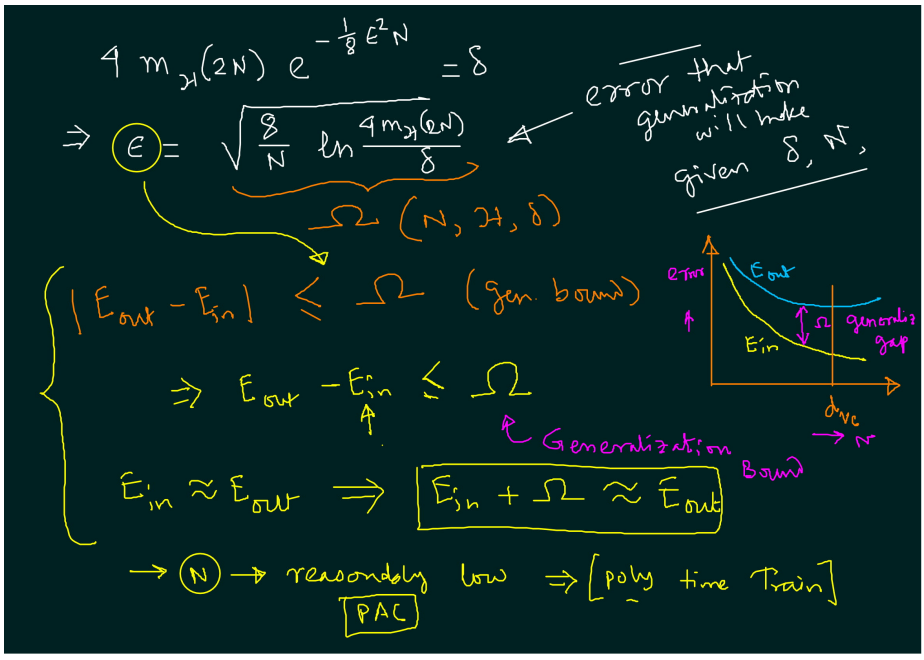
$$\mathcal{P}[f(x) \neq h(x)] < \epsilon] \geq 1 - \delta$$

I can say that your function will match the actual hypothesis or the target function that we have. The concept is known as **Probably Approximately Correct(PAC) Learning**. This has been invented by Leslie Valiant who is a Turing awardee due to only this concept where she has shown that Learning is feasible.

$$4m_H(2N)e^{-\frac{1}{8}\epsilon^2 N} = \delta$$

$$\epsilon = \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

This is the error that generalization will make given δ and number of points N. Let us call it as $\Omega(N, H, \delta)$ and therfore $|E_{out} - E_{in}| \leq \Omega$



So, this is the generalization bound which is pictorially represented by the graph in the slide.
You need to figure out a sweet spot i.e. dimension of VC and in between the

difference of that is the Generalization bound.

So, just removing the absolute values we get

$$E_{out} - E_{in} \leq \Omega$$

The reason of writing this is E_{in} is purposefully the term that you want to minimize through your learning algorithm. But you can't do it for E_{out}. So, it is mostly the case that E_{in} is almost close to zero where you have certain E_{out}. So, L.H.S. is proper negation and I want to generalize it w.r.t certain bound called generalization bound(Ω). So that is the gap that I would like to foresee in my learning and I would try to reduce this gap for the probability there.
ϵ is the error between E_{in} and E_{out} that I want to reduce. So obviouly my $E_{in}+\epsilon$ is what my E_{out} should be. That's how I generalize.

$$E_{in} \approx E_{out} \implies E_{in} + \Omega \approx E_{out}$$

So, this error generalizes my E_{out}. That's what the generalization boundary is all about.

This is a mathematical jugglery to show you that there is a notion of generalization. What you learn plus the error must match the E_{out}. ϵ gets minimized if your $m_H(2N)$ is polynomial.

What happens when number of samples N is reasonably low?
You could train it in polynomial time i.e. in polynomial time I could train the model.
If N is very much high, my hypothesis space is such generic and shatter all possible points then polynomial time training is not possible. That's why the notion of PAC learning says that it is probably approx correct w.r.t. polynomial time learning algorithms in the hypothesis sets.