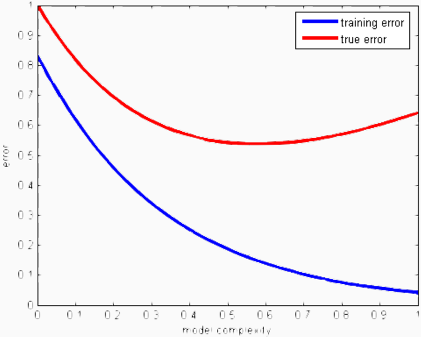
CS-GY 6923: Lecture 7
Taste of Learning Theory, PAC learning

NYU Tandon School of Engineering, Prof. Christopher Musco

THE FUNDAMENTAL CURVE OF ML

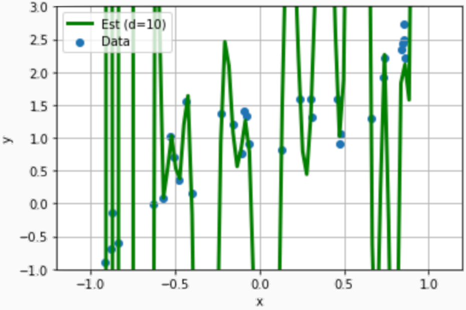
Key Observation: Due to overfitting, more complex models do not always lead to lower test error.



The more complex a model is, the more training data we need to ensure that we do not overfit.

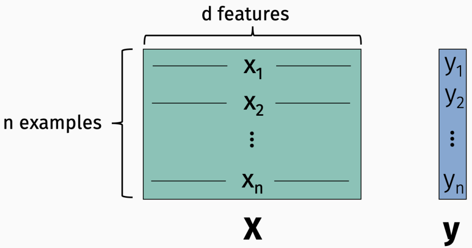
EXAMPLE: POLYNOMIAL REGRESSION

If we want to learn a degree q polynomial model, we will perfectly fit our training data if we have $n \leq q$ examples.



Need $n > q$ samples to ensure good generalization. How much more?

If we want to fit a multivariate linear model with d features, we will perfectly fit our training data if we have $n \leq d$ examples.



Need $> d$ samples to ensure good generalization.

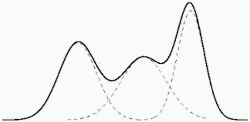
How much more?

Major goal in learning theory:

Formally characterize how much training data is required to ensure good generalization (i.e., good test set performance) when fitting models of varying complexity.

Statistical Learning Model:

- Assume each data example is randomly drawn from some distribution $(\mathbf{x}, y) \sim \mathcal{D}$.



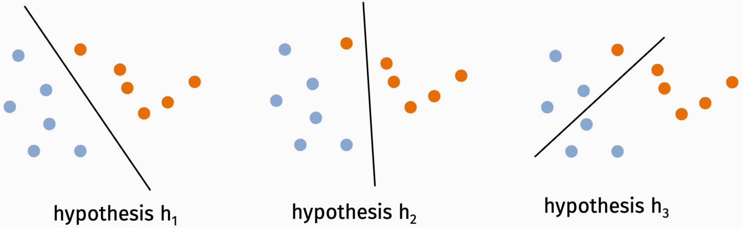
For today: We will only consider <u>classification problems</u> so assume that $y \in \{0, 1\}$.

Statistical Learning Model:

- Assume each data example is randomly drawn from some distribution $(\mathbf{x}, y) \sim \mathcal{D}$.
- Assume we want to fit our data with a function h (a "hypothesis") in some hypothesis class $\mathcal{H}$. For input $\mathbf{x}$, $h(\mathbf{x}) \to \{0, 1\}$.

You can think of h as a model, instantiated with a specific set of parameters. I.e. h is the same as f_θ.

Linear threshold functions:

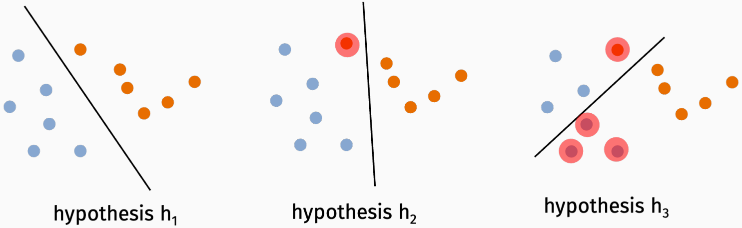


$\mathcal{H}$ contains all functions of the form:

$$h(\mathbf{x}) = \mathbb{1}[\mathbf{x}^T\boldsymbol{\beta} \geq \lambda]$$

.

Linear threshold functions:

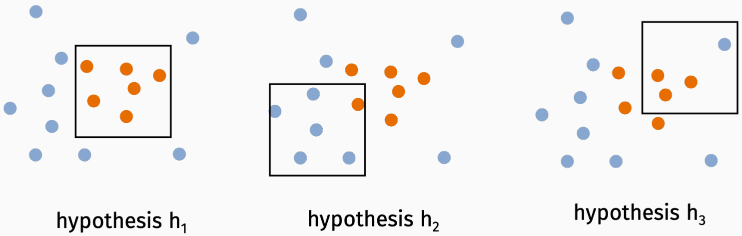


$\mathcal{H}$ contains all functions of the form:

$$h(\mathbf{x}) = \mathbb{1}[\mathbf{x}^T\boldsymbol{\beta} \geq \lambda]$$

.

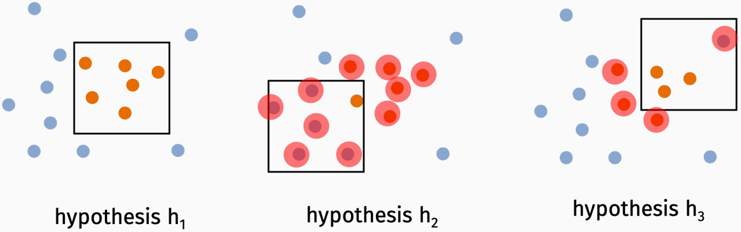
Axis aligned rectangles:



$\mathcal{H}$ contains all functions of the form:

$$h(\mathbf{x}) = \mathbb{1}[l_1 \leq x_1 \leq u_1 \text{ and } l_2 \leq x_2 \leq u_2]$$

Axis aligned rectangles:



$\mathcal{H}$ contains all functions of the form:

$$h(\mathbf{x}) = \mathbb{1}[l_1 \leq x_1 \leq u_1 \text{ and } l_2 \leq x_2 \leq u_2]$$

Disjunctive Normal Form (DNF) formulas:

Assume $\mathbf{x} \in \{0,1\}^d$ is binary.

$\mathcal{H}$ contains functions of the form:

$$h(\mathbf{x}) = (x_1 \wedge \bar{x}_5 \wedge x_{10}) \vee (\bar{x}_3 \wedge x_2) \vee \ldots \vee (\bar{x}_1 \wedge x_2 \wedge x_{10})$$

$\wedge$ = "and", $\vee$ = "or"

k-DNF: Each conjunction has at most k variables.

Same as "population risk" for the zero one loss:

- **Population ("True") Error**:

$$R_{pop}(h) = \Pr_{(\mathbf{x},y)\sim\mathcal{D}}[h(\mathbf{x}) \neq y]$$

- **Empirical Error**: Given a set of samples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m) \sim \mathcal{D}$,

$$R_{emp}(h) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$$

Goal is to find $h \in \mathcal{H}$ that minimizes <u>population error.</u>

Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \sim \mathcal{D}$ be our training set and let h_{train} be the empirical error minimizer:

$$h_{train} = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$$

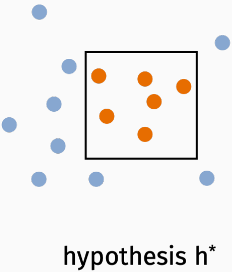
Let h^* be the population error minimizer:

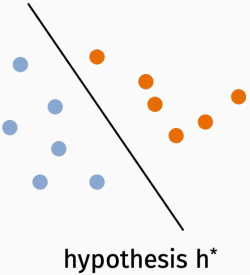
$$h^* = \arg\min_h R_{pop}(h) = \arg\min_h \Pr_{(\mathbf{x},y)\sim\mathcal{D}} [h(\mathbf{x}) \neq y]$$

Goal: Ideally, for some small ϵ, $R_{pop}(h_{train}) - R_{pop}(h^*) \leq \epsilon$.

Simplification for today: Assume we are in the realizable setting, which means that $R_{pop}(h^*) = 0$. I.e. there is some hypothesis in our class $\mathcal{H}$ that perfectly classifies the data.

Formally, for any $(\mathbf{x}, y)$ such that $\Pr_{\mathcal{D}}[\mathbf{x}, y] > 0$, $h^*(\mathbf{x}) = y$.





Extending to the case when $R_{pop}(h^*) \neq 0$ is not hard, but the math gets a little trickier. And intuition is roughly the same.

Probably Approximately Correct (PAC) Learning (Valiant, 1984):

For a hypothesis class $\mathcal{H}$, data distribution $\mathcal{D}$, and training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, let $h_{train} = \arg\min_h \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$.

In the realizable setting, how many training samples n are required so that, with probability $1 - \delta$,

$$R_{pop}(h_{train}) \leq \epsilon?$$

The number of samples n will depend on ϵ, δ, and the <u>complexity</u> of the hypothesis class $\mathcal{H}$. Perhaps surprisingly, it will not depend at all on $\mathcal{D}$.

Many ways to measure complexity of a hypothesis class. Today we will start with the simplest measure: the number of hypotheses in the class, $|\mathcal{H}|$.

Example: What is the number of hypothesis in the class of 3-DNF formulas on d dimensional inputs $\mathbf{x} = [x_1, \ldots, x_d] \in \{0, 1\}^d$?

$$h(\mathbf{x}) = (x_1 \wedge \bar{x}_5 \wedge x_{10}) \vee (\bar{x}_3 \wedge x_2) \vee \ldots \vee (\bar{x}_1 \wedge x_2 \wedge x_{10})$$

Caveat: Many hypothesis classes are infinitely sized. E.g. the set of linear thresholds

$$h(\mathbf{x}) = \mathbb{1}[\mathbf{x}^T\boldsymbol{\beta} \geq \lambda]$$

But you could imagine approximating $\mathcal{H}$ by a finite hypothesis class. E.g. take values in $\boldsymbol{\beta}, \lambda$ to lie on a finite grid of size C. Then how many hypothesis are there?

Formally moving from finite to infinite sized hypothesis classes is a huge area of learning theory (VC theory, Rademacher complexity, etc.)

Consider the realizable setting with hypothesis class $\mathcal{H}$, data distribution $\mathcal{D}$, training data set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, and $h_{train} = \arg\min_h \frac{1}{n}\sum_{i=1}^n \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$.

Theorem

If $n \geq \frac{1}{\epsilon}\left(\log|\mathcal{H}| + \log\frac{1}{\delta}\right)$, *then with probability* $1-\delta$,

$$R_{pop}(h_{train}) \leq \epsilon.$$

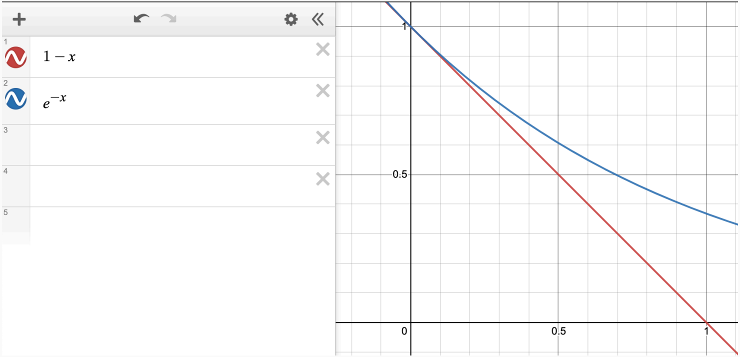
Roughly how many training samples are needed to learn 3-DNF formulas? To learn (discretized) linear threshold funtions?

Two ingredients needed for proof:

1. For any $\epsilon \in [0, 1]$, $(1 - \epsilon) \leq e^{-\epsilon}$.
2. **Union bound**. Basic but important inequality about probabilities.

ALGEBRAIC FACT

For any $\epsilon \in [0, 1]$, $(1 - \epsilon) \leq e^{-\epsilon}$.



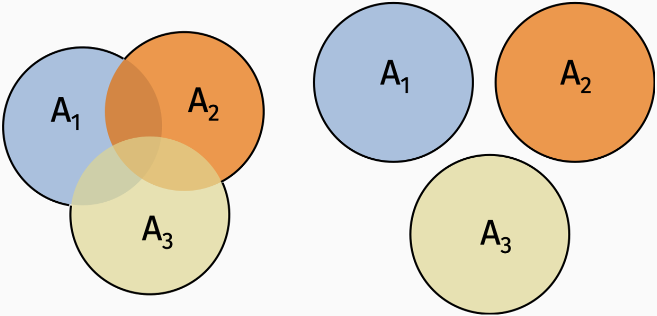
Raising both sides to $1/\epsilon$, we have the $(1 - \epsilon)^{1/\epsilon} \leq \frac{1}{e} \approx .37$.

The specific constant here won't be imporatnt.

Lemma (Union Bound)

For __any__ random events $A_1, \ldots, A_k$:

$$\Pr[A_1 \text{ or } A_2 \text{ or } \ \ldots \ \text{ or } A_k] \leq \Pr[A_1] + \Pr[A_2] + \ldots + \Pr[A_k].$$



Proof by picture.

What is the probability that a dice roll is odd, or that it is ≤ 3?

What is the probability that a dice roll is 1, or that it is ≥ 4?

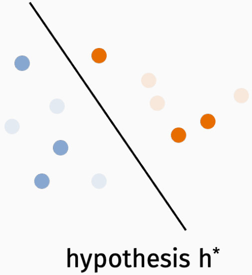
Consider the realizable setting with hypothesis class $\mathcal{H}$, data distribution $\mathcal{D}$, training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, and $h_{train} = \arg\min_h \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$.

Theorem

If $n \geq \frac{1}{\epsilon}\left(\log|\mathcal{H}| + \log\frac{1}{\delta}\right)$*, then with probability* $1-\delta$*,*

$$R_{pop}(h_{train}) \leq \epsilon.$$

First observation: Note that because we are in the realizable setting, we always select and h_{train} with $R_{train}(h_{train}) = 0$. There is always at least one $h \in \mathcal{H}$ such that $h(\mathbf{x}_i) = y_i$ for all i.



Proof approach: Show that for any fixed hypothesis h^{bad} with $R_{pop}(h^{bad}) > \epsilon$, it is very unlikely that $R_{train}(h^{bad}) = 0$. So with high probability, we will not choose a bad hypothesis.

Let h^{bad} be a fixed hypothesis with $R_{pop}(h) > \epsilon$. For $(\mathbf{x}, y)$ drawn from $\mathcal{D}$, what is the probability that $h^{bad}(\mathbf{x}) = y$?

What is the probability that for a training set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ drawn from $\mathcal{D}$ that $h^{bad}(\mathbf{x}_i) = y_i$ for all i? I.e. that $R_{train}(h^{bad}) = 0$.

Claim

For any fixed hypthesis h with $R_{pop}(h^{bad}) > \epsilon$, the probability that $R_{train}(h) = 0$ can be bounded by:

$$\Pr[R_{train}(h^{bad}) = 0] < e^{-\epsilon n}.$$

Set $n \geq \frac{1}{\epsilon}\log(|\mathcal{H}|/\delta)$. Then we have that for any fixed hypthesis h^{bad} with $R_{pop}(h^{bad}) > \epsilon$,

$$\Pr[R_{train}(h^{bad}) = 0] < \frac{\delta}{\mathcal{H}}.$$

Let $h_1^{bad}, \ldots, h_m^{bad}$ be <u>all hypthesis in $\mathcal{H}$ with $R_{pop}(h) > \epsilon$</u>. How large can m be? Certainly no more than $\mathcal{H}$!

$$\begin{aligned}
&\Pr[R_{train}(h_1^{bad}) = 0 \text{ or } \ldots \text{ or } R_{train}(h_m^{bad}) = 0] \\
&\leq \Pr[R_{train}(h_1^{bad}) = 0] + \ldots + \Pr[R_{train}(h_m^{bad}) = 0] \\
&< m \cdot \frac{\delta}{\mathcal{H}}
\end{aligned}$$

So with probability $1 - \delta$ (high probability) <u>no</u> bad hypotheses have 0 training error. Accordingly, it must be that when we choose a hypothesis with 0 training error, we are choosing a <u>good one</u>. I.e. one with $R_{pop}(h) \leq \epsilon$.

THINGS WE DIDN'T COVER TODAY

- How to deal with the non-realizable setting? E.g. where $\min_h R_{pop} \neq 0$?
- How to deal with infinite hypothesis classes (most classes in ML are)?
- How to find $h_{train} = \arg\min_h \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[h(\mathbf{x}_i) \neq y_i]$ in a computationally efficient way?

Have a good spring break!