Lecture 21:

Module 5: An Introduction to Statistical Learning Theory:

We consider the problem of supervised learning in this module, but all the concept generalize to other forms of learnng (e.g. unsupervised learning).

A formal framework for learning theory:

(1) Data is assumed to be generated iid according to a distribution

input / features $\swarrow$ $\quad$ $\searrow$ label

$(x, y) \sim D(x,y)$. The input domain is denoted by $\mathcal{X}$ ($x \in \mathcal{X}$, e.g. $\mathcal{X} = \mathbb{R}^p$), the label domain is denoted by $\mathcal{Y}$ (i.e. $y \in \mathcal{Y}$, e.g. $\mathcal{Y} \in \{0, 1\}$), and

D is a distribution over $X \times Y$.

2: The learner output: the learner's goal is to find a predictive relation $h: X \longrightarrow Y$ which has low error/high accuracy. Formally speaking, for any function $h: X \longrightarrow Y$ we define its error as:

a function from X to Y

error / loss

$$L_D(h) = \Pr_{(x,y)\sim D}\{h(x) \neq y\}$$

D= distribution of the data

$$= \mathbb{E}_{(x,y)\sim D}\left[\mathbb{1}\{h(x)\neq y\}\right]$$

$$\mathbb{1}\{A\} = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{if } A \text{ is false} \end{cases}$$

Ideally, we would like to find a predictor h that has the smallest error; i.e.

our "gold standard" is to solve the following problem:

$$\underset{h}{\text{minimize}} \quad \underbrace{L_D(h)}_{\Pr_{(x,y)\sim D}\{h(x)\neq y\} = E_{(x,y)\sim D}[\mathbb{1}\{h(x)\neq y\}]} \tag{1}$$

However, this task is impossible since D is unknown.

(3) The learner's input: Training Data!

The only information that the learner has about the data distribution is a set of training samples:

$$S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

where each (x_i, y_i) is generated i.i.d.

from the distribution D. Hence, the learner has has to find "approximate" solutions to problem (1) using the training data.

Data distribution D

$$S=\{(x_1,y_1),\dots,(x_n,y_n)\} \longrightarrow \boxed{\text{learner}} \longrightarrow \underbrace{\hat{h}_S}$$

"approximate" solution to this problem.

ultimate goal:

solve:

$$\min_h \Pr\{h(x)\neq y\}$$

$S=\{(x_1,y_1),\dots,(x_n,y_n)\}$

$$\min_h \underbrace{E_{(x,y)\sim D}\left[\#\{h(x)\neq y\}\right]}$$

↳ estimate: $\frac{1}{n}\sum_{i=1}^{n} \#\{h(x_i)\neq y_i\}$

(4) Empirical Risk Minimization (ERM):

true error $\longrightarrow L_D(h) = \mathbb{E}_{(x,y)\sim D}\left[\mathbb{1}\{h(x)\neq y\}\right]$

unbiased estimate

training error $\longrightarrow L_S(h) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{h(x_i)\neq y_i\}$

ERM is the task of finding a predictor that minimizes the training error:

$$\min_h L_S(h)$$

---

instead of solving $\rightarrow \boxed{\min_h L_D(h)}$

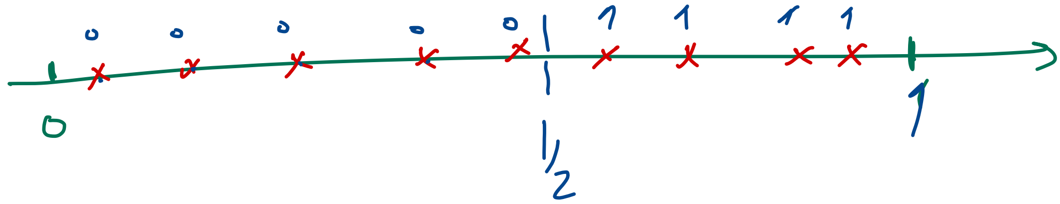
ERM solves $\longrightarrow \boxed{\min_h L_S(h)}$

claim: $\underbrace{\min_h L_S(h) = 0}_{\text{overfitting}}$

Example: Consider the following distribution on data:

$$\mathcal{X} = [0,1] \qquad \longrightarrow \qquad x \sim \text{uniform}[0,1]$$

$$\mathcal{Y} = \{0,1\} \qquad \boxed{y = \begin{cases} 0 & \text{if } x \leq 1/2 \\ 1 & \text{if } x > 1/2 \end{cases}}$$



$$\min_h L_D(h) = \min_h \Pr\{h(x) \neq y\}$$

$$\min_h \underbrace{L_S(h)}_{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{h(x_i) \neq y_i\}} =$$

$$\hat{h}_S = \begin{cases} y_i & \text{if } x = x_i \\ 0 & x \neq \{x_1, \dots, x_n\} \end{cases}$$

Proof of the claim:

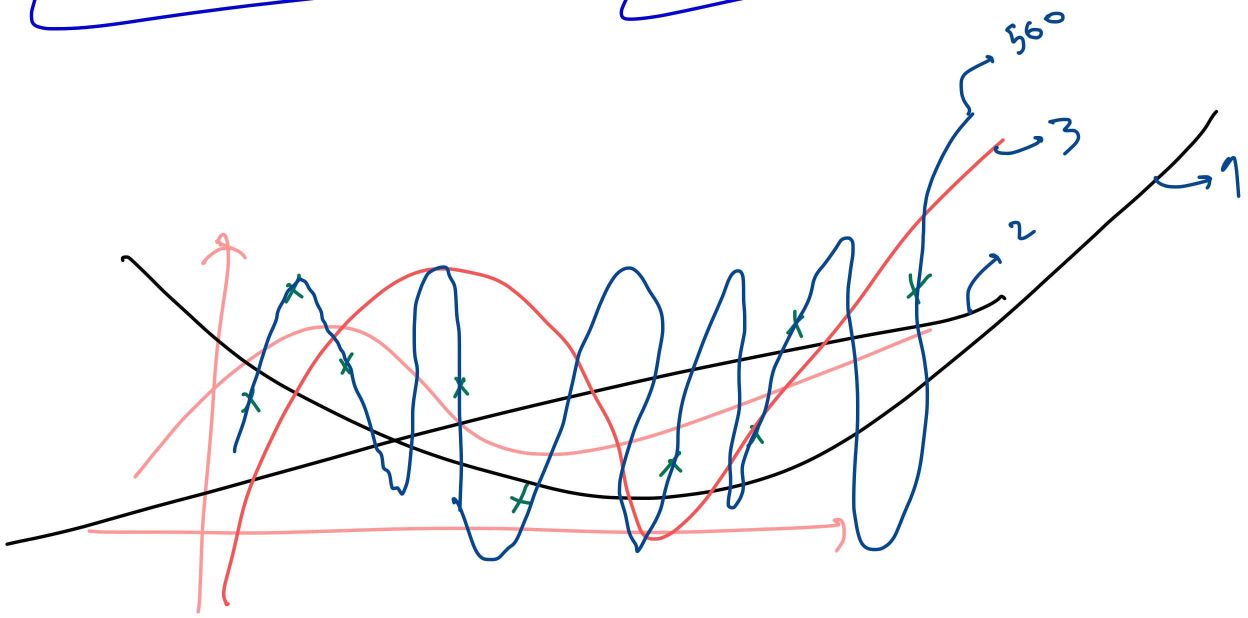
Define $\hat{h}_s(x) = \begin{cases} y_i & \text{if } x = x_i \\ 0 & \text{if } x \notin \{x_1, \ldots, x_n\} \end{cases}$

$$L_s(\hat{h}_s) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{\hat{h}_s(x_i) \neq y_i\}$$

$$= 0$$

$\boxed{\min_h L_D(h)} \longrightarrow \boxed{\min_h L_s(h)}$ → not good overfitting

Recall (Regression)

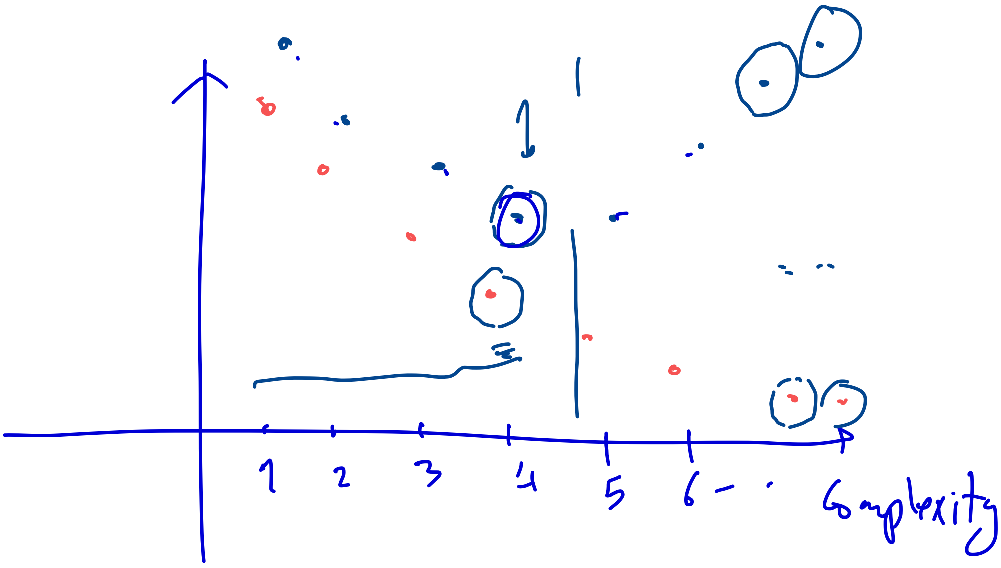


(5) Although ERM is natural, it cain miserably if we are not careful: It can overfit easily: the minimum of

$$\min_{h} L_S(h)$$

is always zero (we always overfit).

(6) We need to search for conditions under which there is a gaurantee that ERM does <u>not</u> overfit; namely, conditions under which when ERM has good performance on training data, it also has good performance over the underlying distribution.



$$\min_{h} L_S(h) \longrightarrow \min_{h \in \mathcal{H}} L_S(h)$$

restrict the class of functions/predictors (restrict the complexity)

$\Downarrow$

prevents overfitting

The above problem is denoted by

$ERM_{\mathcal{H}}$

$\mathcal{H}$ is the specific function class that we use to fit the data.

$$\begin{tikzcd}
S = \{(x_i, y_i)\}_{i=1}^{n} \arrow[r] & \boxed{\text{learner } (\mathcal{H})} \arrow[r] \arrow[d] & \hat{h}_S \\
& \min\limits_{h \in \mathcal{H}} L_S(h) &
\end{tikzcd}$$