Empirical Risk Minimization (ERM)

Setup: given training samples (x_i, y_i), $i = 1, \dots, n$, and a collection of possible predictors $\mathcal{H}$, and a loss function ℓ that measures how well $h(x)$ predicts y

Example: $h_\theta(x) = x^T\theta$, $\mathcal{H} = \{h_\theta : \theta \in \Theta\}$, $\ell(\hat{y}, y) = (\hat{y} - y)^2$

$\Rightarrow \ell(h_\theta(x), y) = (x^T\theta - y)^2$

Other common losses include:

$\ell(\hat{y}, y) = |\hat{y} - y|$

$\ell(\hat{y}, y) = \log(1 + e^{-\hat{y}y})$ (logistic loss)

$$\ell(\hat{y}, y) = \begin{cases} 0 & \text{if } \hat{y} = y \\ 1 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 0 & \hat{y}y > 0 \\ 1 & \hat{y}y < 0 \end{cases}$$ (the 2nd equality holds if $y, \hat{y} \in \{-1, 1\}$)

$\ell(\hat{y}, y) = e^{-\hat{y}y}$ (exponential loss)

$\ell(\hat{y}, y) = \max(0, 1 - \hat{y}y)$ (hinge loss)

Risk: the Risk of a predictor h is $R(h) := \mathbb{E}_{X,Y}[\ell(h(X), Y)]$

If $(x_i, y_i) \overset{iid}{\sim} p(X, Y)$, then

$\hat{R}_n(h) := \sum_{i=1}^{n} \ell(h(x_i), y_i)$ is the empirical risk

Empirical Risk Minimization:

choose $\hat{h}_n$ to minimize sum of losses on all training data:

$$\hat{h}_n = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(h(x_i), y_i)$$

we write $\hat{h}_n$ to reflect that it is a function of our n training samples

The smallest risk we might incur is $R^* := \min_h \mathbb{E}[\ell(h(X), Y)]$ — this is not restricted to any model class $\mathcal{H}$. We might ask, how small is the risk of our $\hat{h}_n$ compared to R^*?

i.e., how big is $\mathbb{E}R(\hat{h}_n) - R^*$?

↑ this expectation is with respect to the distribution of the n training samples

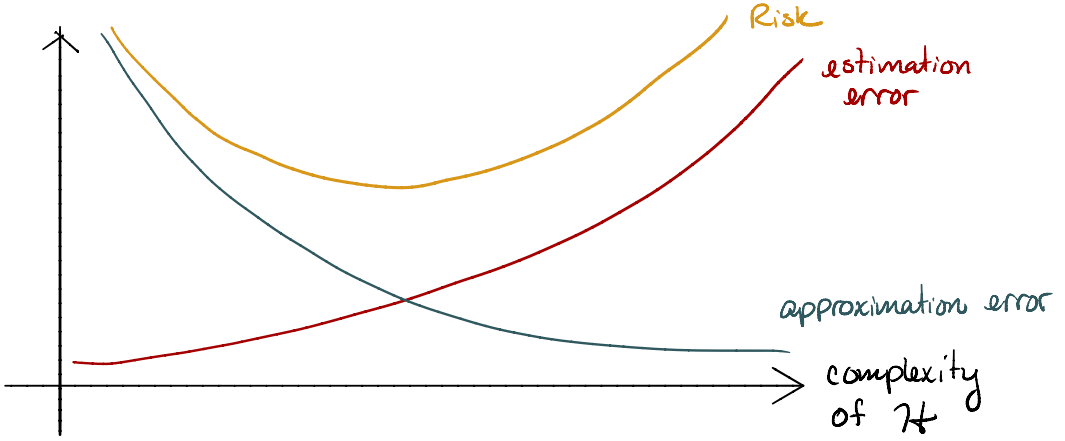
If we knew the distribution of the data, we could choose $h \in \mathcal{H}$ to minimize the risk directly without relying on finite training data. We use this concept to analyze ERM:

$$\mathbb{E}R(\hat{h}_n) - R^* = \underbrace{\mathbb{E}R(\hat{h}_n) - \min_{h \in \mathcal{H}} R(h)}_{\text{estimation error}} + \underbrace{\min_{h \in \mathcal{H}} R(h) - R^*}_{\text{approximation error}}$$

estimation error: error associated with using n training samples instead of full knowledge of the joint distribution of X, Y

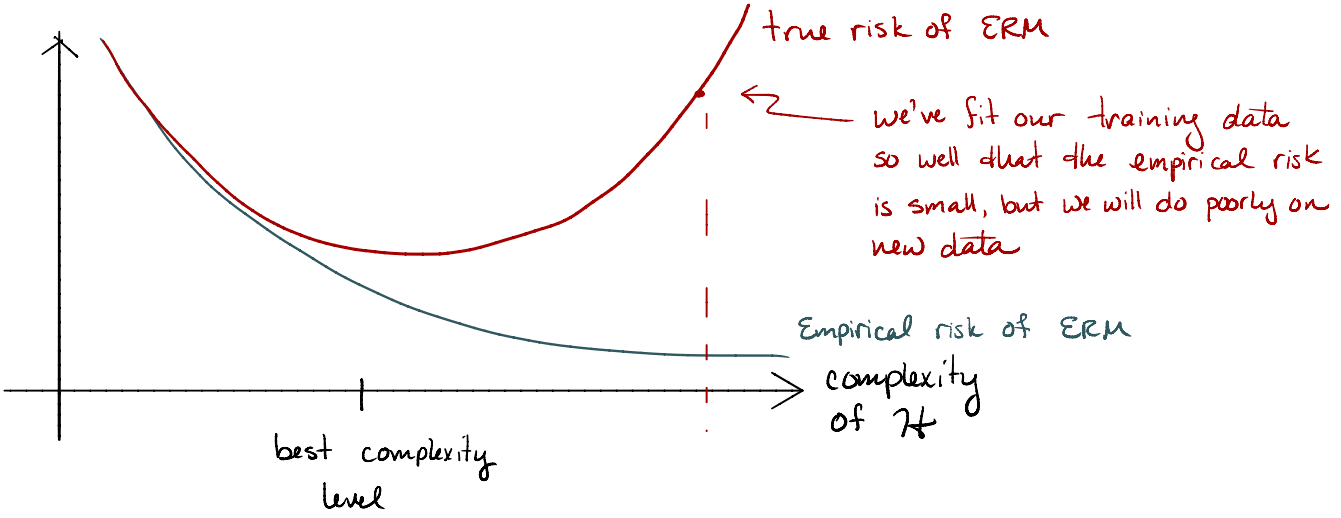
approximation error: error associated with restricting our attention to model class $\mathcal{H}$.

When $\mathcal{H}$ is very complex, approximation error may be smaller because we can closely approximate complex predictors, but estimation error may be large because we have to select on predictor from among many with only finite data, which can make it difficult to distinguish the relative risk of different predictors.



Another perspective: if $\mathcal{H}$ is large, then the empirical risk can be made very small with some choices of $h \in \mathcal{H}$, even when the true risk is large. That is $\hat{R}_n(h) - R(h)$ may be very large for some $h \in \mathcal{H}$ when $\mathcal{H}$ is large.

This leads to OVERFITTING.



How should we avoid overfitting?

A: Choose $\mathcal{H}$ to not be too complex
(e.g. require $\mathcal{H}$ to correspond to linear predictors)

B: instead of ERM, use Complexity regularized ERM

Complexity Regularized ERM:

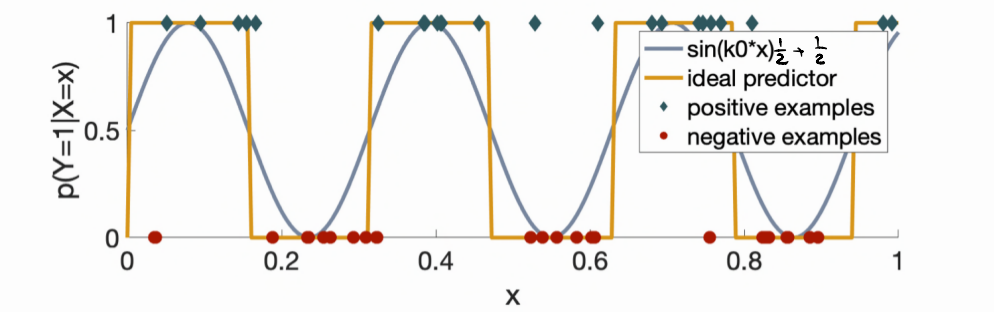
$$\hat{h}_n = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \left\{ \hat{R}_n(h) + C(h) \right\}$$

where $C(h)$ measures the complexity of h

typically C is chosen to mimic the behavior of the estimation error, so we can avoid models with large estimation error, and hopefully achieve a good balance between approximation and estimation error

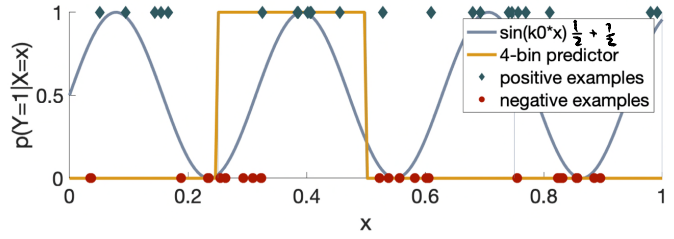
Example: $X \in [0,1]$, $Y|X \sim \text{Bernoulli}\left(\sin(k_0 X)\frac{1}{2} + \frac{1}{2}\right)$

if $k_0 = 20$:

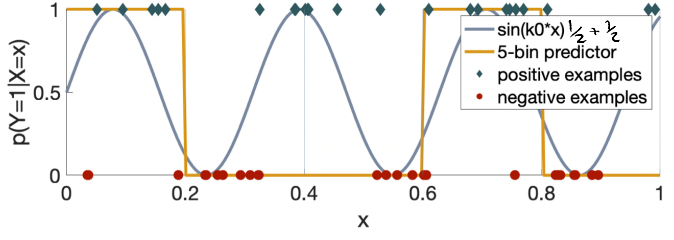


estimation strategy: divide $[0,1]$ into B equi-sized bins. for each bin, perform a majority vote to choose h.

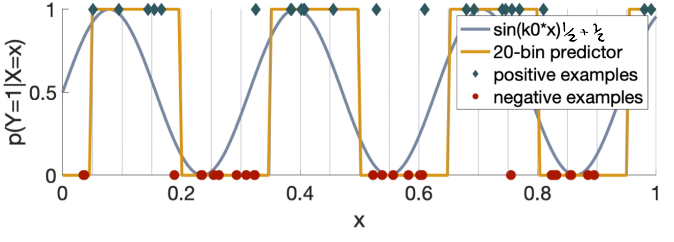
($\mathcal{H}_B$ is set of all B-bin predictors. Majority vote to select $h \in \mathcal{H}_B$ is empirical risk minimization)



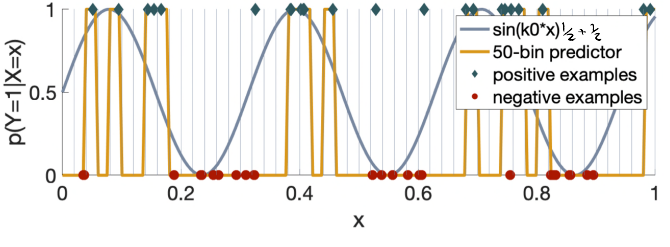
B = 4 high approx error, low est. error



B = 5 high approx error, low est. error



B = 20 med. approx error, med. est. error



B = 50 low approx error, high est. error

OVERFITTING

Example: (continued)

A. we could restrict ourselves to $\mathcal{H}_{20}$ from the beginning

(this may be difficult because we wouldn't know a priori that $B=20$ is optimal)

B. Let $\mathcal{H} := \bigcup_{B \geq 1} \mathcal{H}_B$

$$\hat{h} = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \; \hat{R}_n(h) + C(h) \quad \text{where } C(h) = B \text{ if } h \in \mathcal{H}_B$$

$$= \min\{B : h \in \mathcal{H}_B\}$$

equivalently:

Let $\hat{h}_{n,B} := \underset{h \in \mathcal{H}_B}{\operatorname{argmin}} \; \hat{R}_n(h)$

$$\hat{B} = \underset{B}{\operatorname{argmin}} \; \hat{R}_n(\hat{h}_{n,B}) + B$$

$$\hat{h}_n = \hat{h}_{n,\hat{B}}$$

Complexity Regularization Examples:

- **Method of Sieves**

  define $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \cdots$ where $|\mathcal{H}_i| := \#\{h \in \mathcal{H}_i\}$ grows with i

  if we have n training samples, use $\mathcal{H}_n$

  ex: with our binning method, we could always let $B = \sqrt{n}$

- **Bayesian methods**

  think of $C(h) = -\log p(h)$ = negative log prior likelihood

  that is, $p(h) = e^{-C(h)}$

  $C(h)$ is large (i.e. small prior probability) when h is complex.

Minimum description length (MDL)

represent each h using a string of bits, where more bits are required to represent more complex h. Let $C(h)$ = # bits needed to describe h — the "DESCRIPTION LENGTH"

Ex: binned estimators. how might we "describe" an estimate h_B from the earlier example? recall h_B has B bins (equal sizes) and each bin has a +1 or 0 label assigned to it. Imagine we restrict $B \leq n$. Then we need $\log n$ bits to encode the value of B and an addition B bits to encode the labels for each bin. $\Rightarrow C(h) = \log n + B$ for $h \in \mathcal{H}_B$.

Sparsity regularization

Sparsity regularization is often used in the context of linear predictors:

$h(x) = h_\theta(x) = x^T\theta$ or $g(x^T\theta)$ for some g.

θ here has the same dimension as x: $x \in \mathbb{R}^p \Rightarrow \theta \in \mathbb{R}^p$

however, less "complex" h_θ correspond to sparser θ — that is, θ with more zero valued entries.

Let $\|\theta\|_0 = \#\{i : \theta_i \neq 0\}$ = # nonzero entries in θ ℓ_0-norm

To "describe" h_θ, we need only encode θ. To do this, we need $\log p$ bits to encode the value of $\|\theta\|_0$ plus $\|\theta\|_0 \log p$ bits to encode <u>which</u> entries are nonzero, plus b bits to represent the value of each (quantized) nonzero entry of θ

$$\Rightarrow C(h_\theta) = \log p + \|\theta\|_0 \log p + \|\theta\|_0 \cdot b$$

This leads to ℓ_0 regularized estimation:

$$\hat{\theta} = \arg\min_{\theta} \hat{R}_n(h_\theta) + \lambda \|\theta\|_0$$

(λ: tuning parameter)

If we observe $y_i = \langle x_i, \theta \rangle + \varepsilon_i$, the ℓ_0 regularized least squares estimate is $\hat{\theta} = \arg\min_{\theta} \|y - X\theta\|_2^2 + \lambda \|\theta\|_0$

Unfortunately, this optimization problem is NP-hard.

However, under certain conditions on X and ε, we can instead compute the LASSO estimate

$$\hat{\theta} = \arg\min_{\theta} \|y - X\theta\|_2^2 + \lambda \|\theta\|_1$$

and arrive at the same estimate

Cross validation

Cross validation does not always explicity perform complexity regularization, but it nevertheless can be used to reduce overfitting.

We start by taking our n samples and dividing them into a training set $S_T = \{(x_i, y_i)\}_{i=1}^{m}$ and validation set $S_V = \{(x_i, y_i)\}_{i=m+1}^{n}$

Now assume we have multiple model classes $\mathcal{H}_\lambda$ indexed by a scalar parameter $\lambda > 0$, where bigger λ = more complexity. Then we might adopt the following strategy:

$$\hat{h}_\lambda = \underset{h \in \mathcal{H}_\lambda}{\operatorname{argmin}} \hat{R}_m^{(T)}(h), \quad \text{where } \hat{R}_m^{(T)}(h) = \sum_{i=1}^{m} \ell(h(x_i), y_i)$$

$= \text{error on } S_T$

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmin}} \hat{R}_{n-m}^{(V)}(\hat{h}_\lambda), \quad \text{where } \hat{R}_{n-m}^{(V)}(h) = \sum_{i=m+1}^{n} \ell(h(x_i), y_i)$$

$= \text{error on } S_V$

$$\hat{h} = \hat{h}_{\hat{\lambda}}$$

Example: $\mathcal{H}_\lambda = \{h_\theta : \|\theta\|_0 \leq \lambda\}$ or $\mathcal{H}_\lambda = \{h_\theta : \|\theta\|_1 \leq \lambda\}$

Cross validation and sparsity regularization

observe (y_i, x_i), $i = 1, \ldots, n$.

$$\hat{\theta}_\lambda = \operatorname*{argmin}_\theta \hat{R}_m^{(T)}(h_\theta) + \lambda \|\theta\|_0$$ ← we could also use $\|\theta\|_1$ here

$$\hat{\lambda} = \operatorname*{argmin}_\theta \hat{R}_{n-m}^{(V)}(h_{\hat{\theta}_\lambda})$$

$$\hat{\theta} = \hat{\theta}_{\hat{\lambda}}, \quad \hat{h} = h_{\hat{\theta}}$$

Other forms of complexity regularization

- decision trees (next lecture)
- structural risk minimization
- Vapnik–Cervonenkis dimension
- Rademacher complexity