Lecture 6: Regularization

Recall from previous lecture:

D is unknown distribution over $\mathbb{R}^d$, $f: \mathbb{R}^d \to \{0,1\}$ is ground truth, $f \in \mathcal{A}$ for some known concept class $\mathcal{A}$.

Given: labeled training set $(X_1,y_1),\dots,(X_n,y_n)$ where $X_i \sim D$ independent, $y_i = f(X_i)$.

Goal: Output $g \in \mathcal{A}$ with good performance:

perfect recovery: $g = f$

mostly focus on this → PAC learning: $\Pr_{X\sim D}[g(X) \neq f(X)] \le \varepsilon$ w.p. $1-\delta$.

Thm: If $\mathcal{A} = \{f_1,\dots,f_m\}$, then $n \geq \frac{1}{\varepsilon}(\log m + \log 1/\delta)$ suffices.

(aside: log = ln).

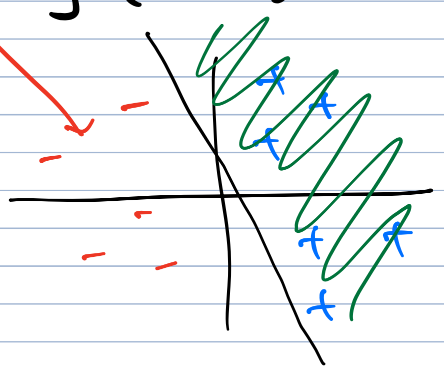
Thm: If $\mathcal{A} = \{\text{linear classifiers}\}$, then $n \geq \Omega\left(\frac{1}{\varepsilon}(d + \log 1/\delta)\right)$ suffices.

Same rates in agnostic setting ($f \notin \mathcal{A}$) but w/ $\frac{1}{\varepsilon^2}$, using ERM

empirical risk minimizer

Recall: linear classifiers are parametrized by $\theta \in \mathbb{R}^d$ unit vector,

$$f_\theta(x) = \text{sign}(\langle \theta, x \rangle) = \begin{cases} 1 & \text{if } \langle \theta, x\rangle \geq 0 \\ 0 & \text{o.w.} \end{cases}$$



Q: What can you do if $d \gg n$?

A: Nothing! (in the worst case)

A: Regularization (for many settings) (not classification!)

In this class, we will explore this through the lens of linear regression

Linear regression: $X_1,\dots,X_n \sim D$ in $\mathbb{R}^d$

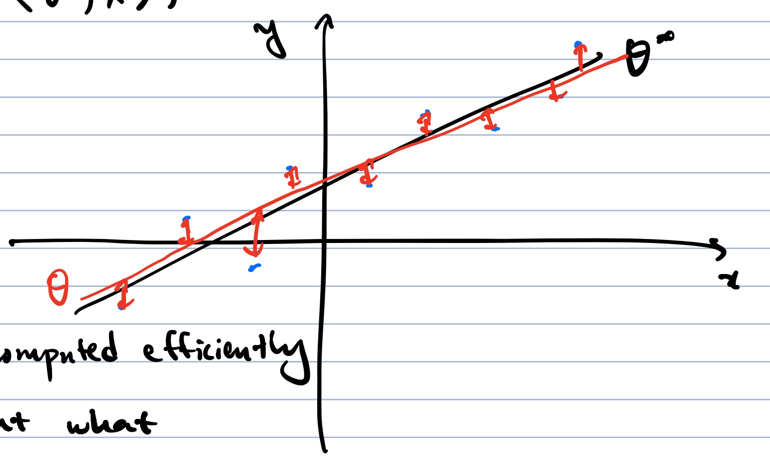
$y_1,\dots,y_n \in \mathbb{R}$ ← not $\{0,1\}$.

ground truth $\theta^* \in \mathbb{R}^d$ ← not unit

Promise: $y_i \approx \langle \theta^*, X_i \rangle$, $\forall i = 1,\dots,n$ ← some noise (often assumed to be Gaussian)

Goal: output θ that:
1) is close to θ^*, or
2) acts like θ^* for typical data

GenError: $\mathbb{E}_{x\sim D}\left(\langle \theta, x\rangle - \langle \theta^*, x\rangle\right)^2$



ERM for linear regression:

$$L_{train}(\theta) := \sum_{i=1}^{n} \left(\langle \theta, x_i\rangle - y_i\right)^2$$

Output: $\underset{\theta \in \mathbb{R}^d}{\operatorname{argmin}}\ L_{train}(\theta)$ ← can be computed efficiently

This works well when $n >> d$, but what happens if $n << d$?

In this case, there are many θ with the same training loss.

Why? In this case, $\operatorname{span}(\{x_1, \dots, x_n\})$ has $\dim \leq n << d$.

for any θ, and any $v \in (\operatorname{span}\{x_1, \dots, x_n\})^{\perp}$, ← $\dim = d - n$.

$\langle \theta, x_i\rangle = \langle \theta + v, x_i\rangle \ \forall i = 1, \dots, n$ $\qquad \langle v, x_i\rangle = 0 \ \forall i$

so $L_{train}(\theta) = L_{train}(\theta + v)$

How to choose good θ?

Intuition: we should favor <u>simple</u> solutions

"simple" depends on the problem though

Common, useful notions of simple:

- small (low norm) solutions
- sparse solutions

Regularization is a method that lets you favor such "simple" solutions.

In this class: ℓ_2 and ℓ_1 regularization

small solutions are simple (ℓ_2)

sparse solutions are simple. (ℓ_1)

ℓ_2-regularization aka "ridge regression", "Tikhonov regularization"
[Hoerl, Kennard '70]

$$L_{ridge}(\theta) = \underbrace{\sum_{i=1}^{n} \left(\langle \theta, x_i\rangle - y_i\right)^2}_{\text{error}} + \underbrace{\boxed{\lambda \|\theta\|_2^2}}_{\text{simplicity}}, \quad \lambda \geq 0 \text{ some parameter}$$

Output $\operatorname{argmin}_\theta L_{ridge}(\theta)$ ← can be computed efficiently example: $z_1, \cdots, z_n \in \mathbb{R}$, $\eta_1, \cdots, \eta_n \in \mathbb{R}$

$x_1 = (z_1, z_1, \eta_1)$ $y_1 = 10 z_1$

$x_2 = (z_2, z_2, \eta_2)$ $y_2 = 10 \cdot z_2$

$\vdots$ $\vdots$

$x_n = (z_n, z_n, \eta_n)$ $y_n = 10 z_n$

which $\theta \in \mathbb{R}^3$ is a solution w/ 0 error?

$\langle \theta, x_i \rangle = \theta_1 z_i + \theta_2 z_i + \theta_3 \eta_i$

so θ is a perfect fit as long as $\theta_1 + \theta_2 = 10$, θ_3 can be arbitrary.

Ridge regression will choose θ w/ minimal ℓ_2-norm

$\theta_3 = 0$ ← removes spurious feature!

min $\theta_1^2 + \theta_2^2$ s.t. $\theta_1 + \theta_2 = 10$.

$\theta_1 = \theta_2 = 5$

ridge solution (for some suitable λ)

$\theta \approx (5, 5, 0)$.

ℓ_1 (and ℓ_0) regularization

What if we want a sparse sol'n? eg. $(10, 0, 0)$

$$L_{\ell_0}(\theta) = \sum_{i=1}^{n} (\langle \theta, x_i \rangle - y_i)^2 + \lambda \|\theta\|_0$$

is limit (in a certain sense) of $\|\theta\|_p^p$ as $p \to 0$.

$$\|\theta\|_0 = |\{i : \theta_i \neq 0\}|$$

But this cannot be computed efficiently

least absolute shrinkage & selection operator

Instead: take the "convex proxy": ℓ_1 (aka LASSO) [Santosa, Symes '86] [Tibshirani, '96]

$$L_{lasso}(\theta) = \sum_{i=1}^{n} (\langle \theta, x_i \rangle^2 - y_i)^2 + \lambda \|\theta\|_1$$

eg $x_1 = (2z_1, z_1, \eta_1)$ $y_1 = 10 z_1$

$x_2 = (2z_2, z_2, \eta_2)$ $\vdots$

$\vdots$

$x_n = (2z_n, z_n, \eta_n)$ $y_n = 10 z_n$

What is the ℓ_1-minimizing solution?

$\theta_1 2 z_i + \theta_2 z_i = 10 z_i \to 2\theta_1 + \theta_2 = 10$

Again $\theta_3 = 0$. Now l_2: $\min \theta_1^2 + \theta_2^2$ s.t. $2\theta_1 + \theta_2 = 10$

$\underline{\theta_1 = 4}, \underline{\theta_2 = 2}$

has nonzero on both.

What about l_1?

$$\min |\theta_1| + |\theta_2|$$
$$\text{s.t. } 2\theta_1 + \theta_2 = 10.$$

$\theta_1 = 5, \boxed{\theta_2 = 0}$. → only 1 nonzero!

Why does l_1 enforce sparsity?

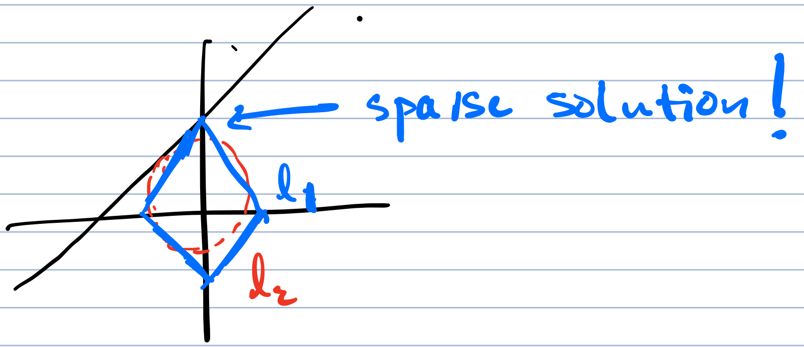
l_1 - regularization is the "soft version" of the following "hard" constraint

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$\min \|\theta\|_1$ such that

affine constraint

$X\theta = y$ ← set of θ w/ 0 train error



Why sparsity?

- In practice, there are often many spurious features! Useful to prune them

- sparsity offers statistical advantages!

Suppose ground truth is sparse, i.e.

$$\mathcal{A} = \{ \langle \theta, x \rangle : \|\theta\|_0 \le k \}.$$

Recall: Intuitively, generalization ≈ $\log|\#\text{distinct elts in } \mathcal{A}|$.

How many free parameters for $\mathcal{A}$?

How many free parameters in A?

A vector in $\mathcal{A}$ can be specified by:

1. Choose its nonzero coordinates S, $|S| \le k$.
2. Choose a k-dim vector on this support

$$\binom{n}{k} \cdot C^k \approx n^k \cdot C^k$$

rate for general vectors.

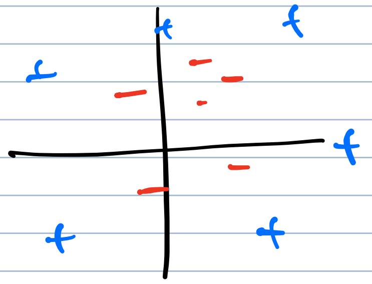
so generalization $\approx \log((nC)^k)$

$\approx k \log n \ll d$ if k small

Kernelization

What if linear is insufficiently expressive?

recall from last lecture:



no good linear separator

We can use a kernel $k: \mathbb{R}^d \to \mathbb{R}^m$, $m \gg d$.

hope there is a classifier in this higher space.

Polynomial kernel:

$$k(x_1, \dots, x_d) = (1, x_1, \dots, x_d, x_1^2, x_1 x_2, \dots, x_d^2)$$

(degree 2 polynomial kernel).

$$k: \mathbb{R}^d \to \mathbb{R}^{(d+1)^2}$$

For $\theta \in \mathbb{R}^{(d+1)^2}$

$$\langle \theta, k(x) \rangle = \sum_{i=0}^{d} \sum_{j=0}^{d} \theta_{ij} x_i x_j \quad (\text{set } x_0 = 1)$$

degree r kernel

$$k: \mathbb{R}^d \to \mathbb{R}^{\boxed{(d+1)^r}} \leftarrow \text{huge!}$$

Recipe: kernelize + regularize

Rule of thumb: want to be in regime where everything just barely works!