COS324: Introduction to Machine Learning

Lecture 5: Efficient Learning

Prof. Elad Hazan & Prof. Yoram Singer

Recap & Today

- So far:
  1. Online learning model & algorithms
  2. PAC learnability: a general model for learning
  3. Learnability of finite hypothesis classes
- Today: what can be learned **efficiently**?

Learning Theory & the Scientific Method

Noam Chomsky, June 2011:

"It's true there's been a lot of work on trying to apply statistical models to various linguistic problems. I think there have been some successes, but a lot of failures. There is a notion of success... which I think is novel in the history of science. It interprets success as approximating unanalyzed data."

PAC Learnability

Fix $\varepsilon, \delta \in (0,1)$. An hypothesis class $\mathcal{H}$ is PAC learnable if there exists a learning algorithm which receives $m_{\mathcal{H}}(\varepsilon, \delta)$ i.i.d samples from any unkown distribution $\mathcal{D}$, and returns an hypothesis h for which $\mathcal{L}_{\mathcal{D}}(h) = \Pr_{(\mathrm{x},y)\sim\mathcal{D}}[h(\mathbf{x}) \neq y] \leq \varepsilon$ with probability $1-\delta$.

$m_{\mathcal{H}}$ is termed the sample complexity of learning $\mathcal{H}$

Realizable & Agnostic PAC Learnability

	PAC	Agnostic PAC
Dist	$\mathcal{D}$ over $\mathcal{X}$	$\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$
Truth	$h^\star \in \mathcal{H}$	not in class, may not exist
Risk	$\mathcal{L}_\mathcal{D}(h) = \Pr_{\mathrm{x} \sim \mathcal{D}}[h(\mathbf{x}) \neq h^\star(\mathbf{x})]$	$\mathcal{L}_\mathcal{D}(h) = \Pr_{(\mathrm{x},y) \sim \mathcal{D}}[h(\mathbf{x}) \neq y]$
Input	$S = \{(\mathbf{x}^i, h^\star(\mathbf{x}^i))\} \sim \mathcal{D}$	$S = \{(\mathbf{x}^i, y^i)\} \sim \mathcal{D}$
Goal	$\mathcal{L}_\mathcal{D}(\mathcal{A}(S)) \leq \varepsilon$	$\mathcal{L}_\mathcal{D}(\mathcal{A}(S)) \leq \min_{h \in \mathcal{H}} \mathcal{L}_\mathcal{D}(h) + \varepsilon$
|Sample|	$O\left(\log(\|\mathcal{H}\|/\delta)/\varepsilon\right)$	$O\left(\log(\|\mathcal{H}\|/\delta)/\varepsilon^2\right)$

Infinite Hypothesis Classes

- VC (Vapnik-Chervonenkis) dimension: “effective size” of hypothesis class (infinite or finite).
- VD dimensions is typically, but not always, equal to number of weights / parameters.
- Finite classes, $\text{VC}\dim(H) = \log |H|$
- Axis-aligned rectangles in $\mathcal{R}^d$, $\text{VC}\dim(H) = O(d)$
- Hyperplanes in $\mathcal{R}^d$, $\text{VC}\dim(H) = d + 1$
- Polygons in the plane, $\text{VC}\dim(H) = \infty$

Fundamental Theorem of Statistical Learning

(Without proof)

A realizable learning problem $(\mathcal{X}, \mathcal{Y}, \mathcal{H})$ is PAC-learnable if and only if its VC dimension is finite. Furthermore, it is learnable with sample complexity of

$$O\left(\frac{\mathrm{VC}\dim(H) + \log\frac{1}{\delta}}{\varepsilon}\right)$$

using the ERM algorithm, with sample

$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \sim \mathcal{D}$ of size $|S| = m$:

$$\text{Return } \hat{h} = \arg\min_{h \in \mathcal{H}} \left\{ \sum_{i \in S} \mathbb{1}[h(\mathbf{x}_i) \neq y_i] \right\}$$

Overfitting

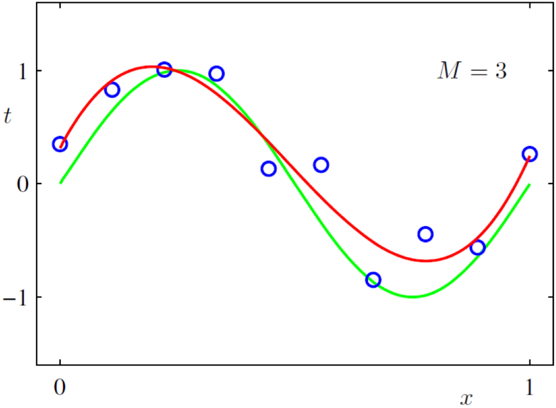
Sample complexity:

$$O\left(\frac{\mathrm{VC\,dim}(H) + \log\frac{1}{\delta}}{\varepsilon}\right)$$

is tight and explains the phenomenon of overfitting...

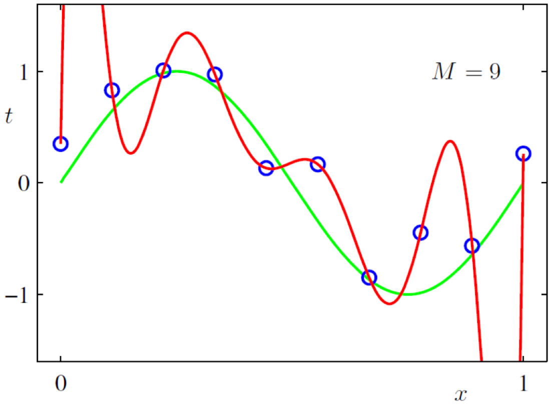
Learning & Overfitting

- We are given examples (x^i, y^i) where $x^i \in \mathbb{R}$ and $y^i = \sin(2\pi x) + \xi$ where $\xi \sim \mathcal{N}(0, 0.05)$
- We do not know the form of the function and decide to use M-degree polynomial to fit the examples



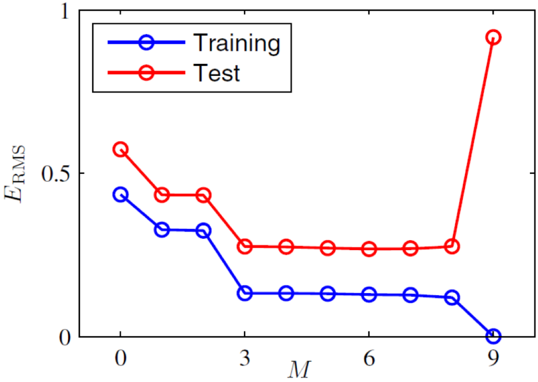
Taken from *Machine Learning and Pattern Recognition*, C.M. Bishop

Learning & Overfitting (cont.)



Taken from *Machine Learning and Pattern Recognition*, C.M. Bishop

Approximation Error on Unseen Examples



Taken from *Machine Learning and Pattern Recognition*, C.M. Bishop

Occam's Razor

William of Occam (circa 1287 — 1347), controversial theologian: "plurality should not be posited without necessity", i.e. "the simplest explanation is best explanation"

Modern-day version in Learning-Theoretic notation:

Theorem: A realizable learning problem $\mathcal{L} = (\mathcal{X}, \mathcal{Y}, \mathcal{H})$ is PAC-learnable if and only if its VC-dimension is finite, in which case it is learnable with sample complexity

$$O\left(\frac{\dim(H) + \log \frac{1}{\delta}}{\varepsilon}\right)$$

using the ERM algorithm.

Complex Hypothesis Classes

- Python programs of <= 10000 bytes:

$$|H| \approx 10000^{10000}$$

sample complexity is $O\left(\frac{\log|H|+\log\frac{1}{\delta}}{\varepsilon}\right) \approx O(\frac{50K}{\varepsilon})$ - not too bad!

- Can we find an efficient learning algorithm?
- Is learning equivalent to the halting problem?
- The *main* issue with PAC learning is computational efficiency
- Next topic: hypothesis classes that permit *efficient* learning through convex optimization

Efficiently Learnabble Hypothesis Spaces

Boolean formulae

Table: default

x_1	x_2	x_3	x_4	$f(\mathbf{x})$
1	0	0	1	0
0	1	0	0	1
0	1	1	0	1
1	1	1	1	0

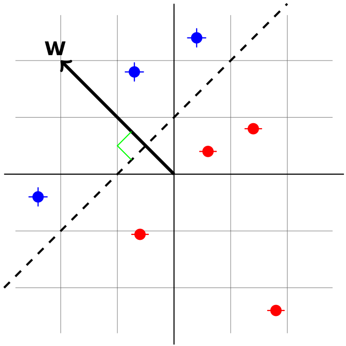
$$f(\mathbf{x}) = \bar{x}_4 \wedge x_2 \wedge \bar{x}_1$$

(Homework...)

Linear Classifiers

- Domain: Euclidean space $\mathbf{x} \in \mathcal{X} = \mathbb{R}^d$
- Hypothesis class: thresholding of linear predictors

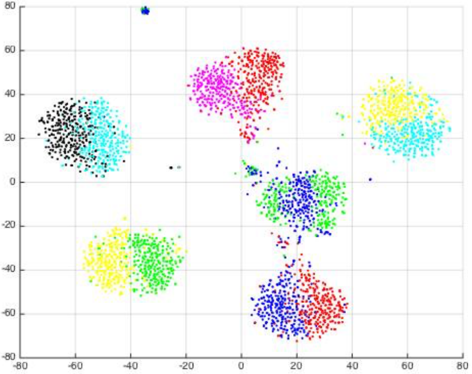
$$h_{\mathrm{w}}(x) = \mathrm{sign}\left(\mathbf{w} \cdot \mathbf{x} - b\right)$$



b is called a bias term. Assume it is zero w.l.o.g.

Practical Importance of Linear Classifiers

Many phenomena in nature are close to being linearly separable



Linear Classification: Learnability

Theorem the sample complexity of learning linear classifiers is

$$O\left(\frac{d + \log\frac{1}{\delta}}{\varepsilon}\right)$$

since $\mathrm{VC}\dim(H) = d + 1$

- Reduction to finite hypothesis class: Let $\mathcal{H}$ be all hyperplanes in $\mathbb{R}^d$ with norm at most one and precision ε,

$$\mathcal{H} = \{\mathrm{sign}(\mathbf{w}\cdot\mathbf{x}) \mid \|\mathbf{w}\| \leq 1,\, \mathbf{w} = (\varepsilon n_1, \varepsilon n_2, \ldots, \varepsilon n_d) : n_j \in \mathbb{Z}\}$$

$$|\mathcal{H}| \approx \left(\frac{1}{\varepsilon}\right)^d$$

- From learnability of finite hypothesis classes, linear classifiers are learnable with sample complexity

$$O\left(\frac{d\log\frac{1}{\varepsilon} + \frac{1}{\delta}}{\varepsilon}\right)$$

Linear Classification: Algorithmic Intractability

- ERM amounts to: given m vectors, $\mathbf{x}^1, \ldots, \mathbf{x}^m \in \mathbb{R}^d$ and m labels $y^1, \ldots, y^m \in \{-1, +1\}$ find a linear classifier $\mathbf{w}$ such that

$$\forall i \ , \ \operatorname{sign}\left(\mathbf{w} \cdot \mathbf{x}^i\right) = y^i$$

- Reduces to linear programming, assuming realizability

- Without realizability, finding $\mathbf{w}$ which minimizes disagreement $\operatorname{sign}\left(\mathbf{w} \cdot \mathbf{x}^i\right) \neq y^i$ is NP-hard

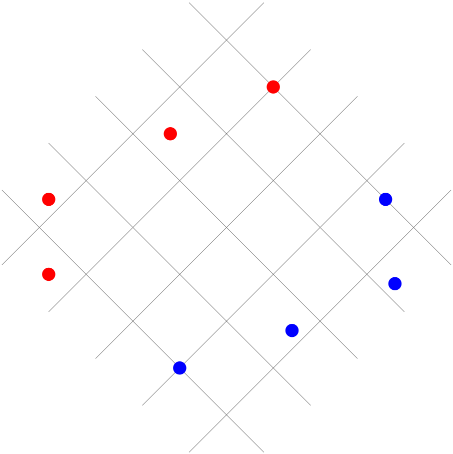
The Perceptron Algorithm

Iterate:

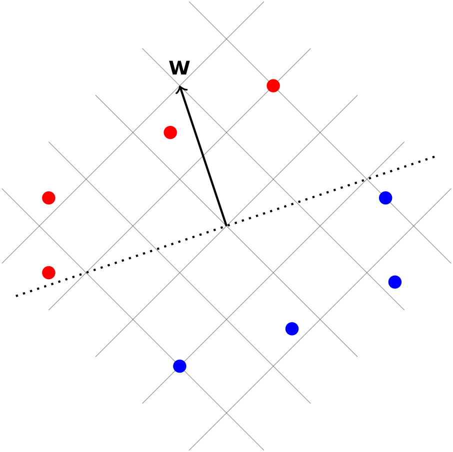
1. Find vector $\mathbf{x}^i$ such that $\text{sign}(\mathbf{w} \cdot \mathbf{x}^i) \neq y^i$
2. Add $\mathbf{x}^i$ to $\mathbf{w}$:

$$\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t + y^i \mathbf{x}^i$$

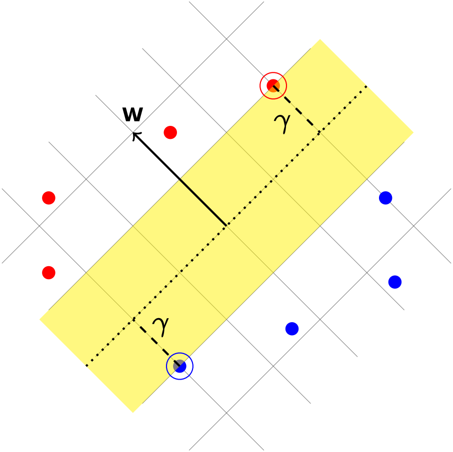
Classification Margin



Classification Margin



Classification Margin



The Perceptron Algorithm

Theorem (Novikoff 1962)

The perceptron algorithm returns a separating hyperplane for a realizable set of examples after at most $1/\gamma^2$ *iterations, where* γ *is the margin of the examples.*

Analysis of The Perceptron Algorithm

Let $\mathbf{w}^*$ be the optimal hyperplane such that $\forall i$, $y^i \mathbf{w}^* \mathbf{x}^i \geq \gamma$

From the Perceptron update we get the following two inequalities:

$$\begin{aligned} \mathbf{w}^{t+1} \cdot \mathbf{w}^* &= (\mathbf{w}^t + y^i \mathbf{x}^t) \cdot \mathbf{w}^* \\ &\geq \mathbf{w}^t \cdot \mathbf{w}^* + \gamma \end{aligned}$$

$$\begin{aligned} \|\mathbf{w}^{t+1}\|^2 &= \|\mathbf{w}^t + y^i \mathbf{x}^t\|^2 \\ &= \|\mathbf{w}^t\|^2 + y^t \mathbf{x}^t \cdot \mathbf{w}^t + \|y \mathbf{x}^t\|^2 \\ &\leq \|\mathbf{w}^t\|^2 + 1 \end{aligned}$$

Thus,

$$1 \geq \frac{\mathbf{w}^t}{\|\mathbf{w}^t\|} \mathbf{w}^* \geq \frac{t\gamma}{\sqrt{t}} = \sqrt{t}\gamma$$

And conclude that,

$$t \leq \frac{1}{\gamma^2}$$

Summary

- Overfitting, Occam's razor, theory of theories...
- main bottleneck: computational!
- Learning hyperplanes

Next time: a general framework for **efficient** learning