

PAC Learning

Lecture 1

Probability

Events, probabilities,
conditional probabilities,
union bound

Random variables,
expected value,
linearity of expectation
Markov's inequality

Algorithms

Approx. 3-SAT,

Big idea: we only want the average

Linearity of Expectation
 $E[X+Y] = E[X] + E[Y]$ (Independence added)
"average sum = sum of averages"

let $Y_i = \begin{cases} 1 & \text{if } i\text{th clause satisfied} \\ 0 & \text{otherwise} \end{cases}$ ($i=1, \dots, m$)

$E[Y_i] = 7/8$ (1 clause case)

$E[\text{\# clauses satisfied}] = E\left[\sum_{i=1}^m Y_i\right] = \sum_{i=1}^m E[Y_i] = 7/8 m$

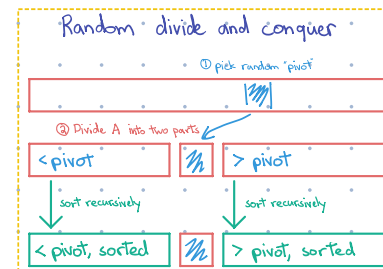
let $Y_i = \begin{cases} 1 & \text{if } i\text{th clause satisfied} \\ 0 & \text{otherwise} \end{cases}$

$Z = \sum_{i=1}^m Y_i = \text{\# clauses satisfied}$

$E[Z] = \frac{1}{2} E[Z | x_1 = \text{true}] + \frac{1}{2} E[Z | x_1 = \text{false}]$

$\Rightarrow$ either $E[Z | x_1 = \text{true}]$ or $E[Z | x_1 = \text{false}]$ is $\geq E[Z]$!

QuickSort,



Expected running time of QuickSort

For each $i, j \in [n]$, w/ $i < j$

$X_{ij} = \begin{cases} 1 & \text{if } i\text{th smallest element is compared to } j\text{th smallest} \\ 0 & \text{otherwise} \end{cases}$

$\sum_{i < j} X_{ij}$ = total # comparisons made by the algorithm
 $\approx$ total running time

$E[\sum_{i < j} X_{ij}]$ = average running time

$E[\sum_{i < j} X_{ij}] = \sum_{i < j} E[X_{ij}]$ (by linearity of expectation, average doesn't care if merged)

now analyze $E[X_{ij}]$ in isolation

$E[X_{ij}] = \Pr[X_{ij} = 1]$
 $= \Pr[\text{ith element is compared to } j\text{th}]$

$\uparrow$

ith elem is compared w/ jth

iff

ith or jth elem selected before any element in between

happens w/ probability $\frac{2}{j-i+1}$

$E[X_{ij}] = \frac{2}{j-i+1}$

$E[\sum_{i < j} X_{ij}] = \sum_{i < j} \frac{2}{j-i+1}$ (by linearity of expectation)

$= \sum_{i=1}^n \sum_{j=i+1}^n \frac{2}{j-i+1} = \sum_{k=1}^n \sum_{i=1}^k \frac{2}{k+1-i+1}$

$\leq 2n \sum_{k=1}^n \frac{1}{k} = 2n H_n$ with harmonic numbers

Quickselect (HW)

Lecture 2

Probability

Hash Functions,

Universal Hash Functions

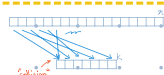
$h: [n] \rightarrow k$ s.t. for all pairs $i, j \in [n]$,
 $\Pr[h(i) = h(j)] \leq \frac{1}{k}$ (much weaker than ideal hash)

e.g. $h(x) = (ax + b) \bmod p \bmod k$

where • p prime, $> k$

• a in $\{1, \dots, p-1\}$ unifs. random

• b in $\{0, \dots, p-1\}$ unifs. random



Markov's Inequality

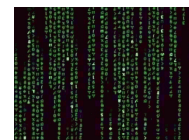
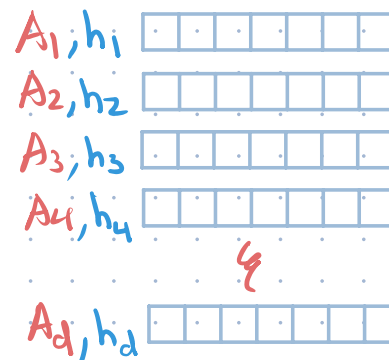
Markov's Inequality (special case)

Let $X \geq 0$ be nonnegative RV

$$\Pr[X \geq 2E[X]] \leq \frac{1}{2}$$

Algorithms

Frequency Estimation, Heavy Hitters (in streams)

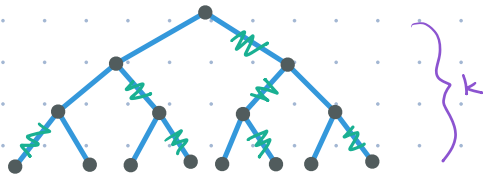


estimate f_i w/
 $\min_{j=1, \dots, d} A_j[h_j(i)]$

Lecture 4

Probability

Galton-Watson processes



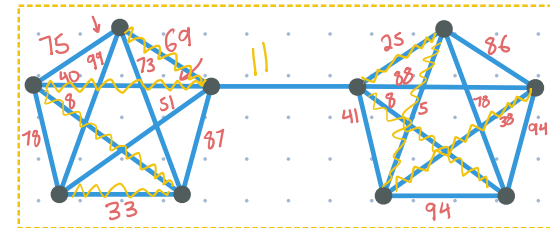
let T = complete binary tree of height k ,
w/ every edge deleted w/ prob $\leq 1/2$

Fact: $P[\text{some leaf connected to root}] \geq \frac{1}{k}$

"Galton-Watson process"

Algorithms

Random Contractions for min cut



Randomized contractions

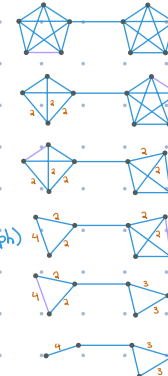
Input: $G=(V,E)$, weights $c \in \mathbb{R}_{>0}^E$

① while $|E| > 1$

② sample $e \in E$ in proportion to c

③ "contract" e

④ return set of edges (in original graph)
contracted to remaining edge



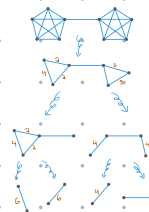
Amplification by branching

Instead of restarting at the beginning, restart midway through

High-level algorithm

① randomly contract until prob.
of ruining the min-cut is $\sim 1/2$

② Branch:
make two copies of current graph
recurse on both copies



PLAYER	Red	1
SCORE		10
POWDER		100
TIME		1:00

Would you like to
DRIVE the game?

Lecture 5

Probability

Multiplicative Chernoff Bound

Multiplicative Chernoff bound

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

$$\epsilon \in (0, 1)$$

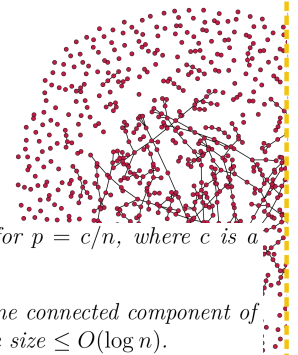
- $P[X_1 + \dots + X_n \geq (1 + \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3}$
- $P[X_1 + \dots + X_n \leq (1 - \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2}$

Algorithms (/ results)

Random graphs & the giant component

Erdős-Rényi graphs $G(n, p)$

- n vertices
- each edge sampled independently w/ probability p



Theorem 5.3. Consider a random graph $G \sim \mathcal{G}(n, p)$ for $p = c/n$, where c is a constant.

1. If $c > 1$, then with high probability, there is exactly one connected component of G with $\Omega(n)$ vertices, and all other components have size $\leq O(\log n)$.
2. For $c < 1$, then with high probability, all connected components of G has size $< O(\log n)$.

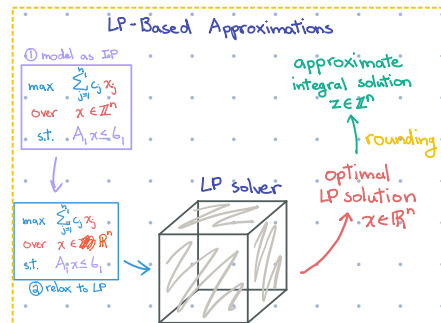
Lecture 6

Probability (& techniques)

Approximations via linear programming & randomized rounding

Linear programs (LPs)
similar to IPs, except no integrality constraints

min/max $\sum_{j=1}^n c_j x_j$ linear objective
over $x \in \mathbb{R}^n$ linear integral variables
st. $A_1 x \leq b_1$ linear constraints
 $A_2 x = b_2$
 $A_3 x \geq b_3$



Continuous optimization vs Discrete optimization

randomization

For each variable x_i , independently, randomly set $x_i = t$ with probability g_i , and $x_i = f$ otherwise.

$$E[\# \text{ sat. clauses}] \geq (1 - 1/e) \text{OPT}$$

k	oblivious	LP	average
1	1/2	$z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
2	3/4	$(3/4)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
3	7/8	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
4	15/16	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
5	31/32	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
...	...	...	...
k	$1 - 1/2^k$	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$

Can we get $\frac{3}{4}$ -APX?

Algorithms

Max Sat Set Cover Covering Integer Programs

Rounding LP solution
 $x \in [0, 1]^n \mapsto F \subseteq \{S_1, \dots, S_n\}$

For each set S_j , independently, take S_j with probability x_j .

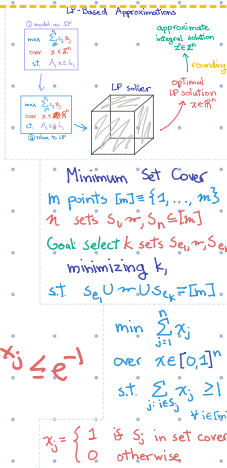
$$E[|F|] = \sum x_j \leq \text{OPT}$$

Fix a point i .

$$E[\# \text{ sets in } F \text{ covering } i] = \sum_{j: i \in S_j} x_j \geq 1$$

$$P[F \text{ not cover } i] = \prod_{j: i \in S_j} (1 - x_j) \leq e^{-\sum_{j: i \in S_j} x_j} \leq e^{-1}$$

$$P[F \text{ cover } i] \geq 1 - e^{-1}$$



Lecture 7

Probability

Continuous Random Variables

Variance, Chebyshev's Variance Reduction

Median trick

Variance

$$\text{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

Chebyshev's inequality

$$P[|X - E[X]| \geq d] \leq \frac{\text{Var}(X)}{d^2}$$

"median trick", "median of means"

① take mean to reduce variance, get constant prob. of error

② take median to amplify success prob. exponentially

Algorithms

Distinct Elements

1, 2, 3, ..., 14, 15, 16, ...

Algorithm w/
limited space

Goal: estimate # of distinct elements in the stream

Suppose you had an ideal hash function

$$h: [m] \rightarrow [0, 1]$$

and the following facts:

let $Y_1, \dots, Y_k \in [0, 1]$ independent and uniformly random

$$X = \min\{Y_1, \dots, Y_k\}$$

$$\text{then } E[X] = \frac{1}{k+1}$$

$$\textcircled{1} X \leftarrow 1$$

② for each element e

$$\textcircled{a} X \leftarrow \min\{X, h(e)\}$$

$$/* E[X] = \frac{1}{k+1}, k = (\text{\# distinct elements}) */$$

$$\textcircled{3} \text{ return } \frac{1}{X} - 1$$

$$\text{recall } E[X] = \frac{1}{k+1}$$

$$\text{goal: show } |X - E[X]| \leq \frac{\epsilon}{k+1}$$

w/ good prob.

$$P[|X - E[X]| \geq \frac{\epsilon}{k+1}]$$

$$\leq \text{Var}(X) \cdot \left(\frac{k+1}{\epsilon}\right)^2 \leftarrow \text{Chebyshev's inequality}$$

where

$$\text{Var}(X) = \frac{k}{(k+1)^2(k+2)} \approx \frac{1}{k^2}$$

Q: how to reduce $\text{Var}(X)$?

we can get $(1 \pm \epsilon)$ -APX

w/ 10% error prob. in $O(\frac{1}{\epsilon^2})$ space

what about high probability?

① make q copies $Y_1, \dots, Y_q$ of

② return median of $Y_1, \dots, Y_q$

Algo correct $\Leftrightarrow \geq \frac{1}{2}$ of Y_i 's correct

let $Z = \# \text{correct } Y_i$'s. $E[Z] = \frac{q}{2}$

$$P[Z \leq (1 - \frac{\epsilon}{4})E[Z]] \leq e^{-(\frac{\epsilon}{4})^2 E[Z]/2}$$

$$\leq \frac{1}{n^{O(n)}} \text{ for } q = O(\log n)$$

ideal hash $h: [m] \rightarrow [0, 1]$

let $Y_1, \dots, Y_k \in [0, 1]$

indep. and unifs. rand.

$$X = \min\{Y_1, \dots, Y_k\}$$

$$\text{then } E[X] = \frac{1}{k+1}$$

$$\text{and } E[X^2] = \frac{2}{(k+1)(k+2)}$$

$$\text{Var}(X) = E[(X - E[X])^2]$$

$$X_1, \dots, X_k \text{ indep. copies of } X$$

$$\bar{X} = \frac{1}{k}(X_1 + \dots + X_k)$$

$$\text{Var}(\bar{X}) = \frac{1}{k} \text{Var}(X)$$

$$P[X_1 + \dots + X_k \leq (1 - \epsilon)k] \leq e^{-\epsilon^2 k/4}$$

Lecture 8

Probability

Gaussian random variables

Gaussian random variables

$$x \sim \mathcal{N}(\mu, \sigma^2)$$

mean

variance

Def: the nicest, most pleasant, convenient random variable ever

Fact let $x \sim \mathcal{N}(\mu, \sigma^2)$, $d \in \mathbb{R}$.

then $x+d \sim \mathcal{N}(\mu+d, \sigma^2)$

$dx \sim \mathcal{N}(d\mu, d^2\sigma^2)$

Fact let $x_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$, $x_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$

then $x_1 + x_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$

Fact let $x \sim \mathcal{N}(0, 1)$, $t < \frac{1}{2}$

then $E[e^{tx^2}] = \frac{1}{\sqrt{1-2t}}$

Algorithms

Dimensionality reduction

Bag-of-words

literally any sentence is fine

any ↑ is literally sentence English
five encodes text in high dimensions

Lemma 8.8. Let $A \sim \mathcal{N}^{k \times d}$ be a random matrix where each coordinate A_{ij} is an independently drawn sample from $\mathcal{N}$. Let $\epsilon > 0$ be sufficiently small. For any vector x ,

$$\mathbb{P}\left[(1-\epsilon)\|x\|^2 \leq \frac{1}{k}\|Ax\|^2 \leq (1+\epsilon)\|x\|^2\right] \geq 1 - 2e^{-k/8}$$

$$k \left\{ \left[\frac{1}{k} \|Ax\|^2 \right] \leftarrow \alpha_i \right.$$

$$E[\|Ax\|^2] = \sum_{i=1}^k E[(Ax)_i^2] = k\|x\|^2$$

$$E[(\sum_{i=1}^k A_{ij} x_j)^2] = \|x\|^2$$

$$A_{ij} x_j \sim \mathcal{N}(0, x_j^2)$$

$$\sum_{j=1}^n A_{ij} x_j \sim \mathcal{N}(0, \|x\|^2)$$

$$(Ax) \sim \mathcal{N}^k(0, \|x\|^2)$$

$$\Pr[\|Ax\|^2 \geq k\|x\|^2 \cdot (1+\epsilon)]$$

$$\leq e^{-\epsilon^2 k/8}$$

Dimensionality Reduction - high level

$O(\log n)$ dimensions suffice

where $n = \# \text{ points}$

$$\mathbb{R}^{\text{very large}} \xrightarrow{f(x)} \mathbb{R}^{O(\log n)}$$

One can map n points into $\mathbb{R}^{O(\log n/\epsilon^2)}$ while preserving all distances up to $(1 \pm \epsilon)$ -APX

Lecture 9

Probability

Locality-sensitive hashing

Fix $\theta \in [0, \pi]$. given q , either:

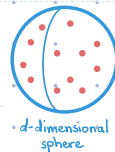
(a) output $p \in P$ w/ $\angle(p, q) \leq 4\theta$

(b) declare no $p \in P$ w/ $\angle(p, q) \leq \theta$

define $h: S^{d-1} \rightarrow \{-1, 1\}^k$ by

$$h_i(x) = \text{sign}(\langle g_i, x \rangle) \text{ w/ } g_i \sim N^d$$

$$k = \pi(\log(n)/2\theta)$$



d-dimensional sphere

when querying y :

- if there is $x \in P$ w/ $\angle(x, y) \leq \theta$, then it collides w/ probability $\geq 1/\sqrt{n}$

- expect to collide w/ at most one point $x \in P$ w/ $\angle(x, y) > 2\theta$

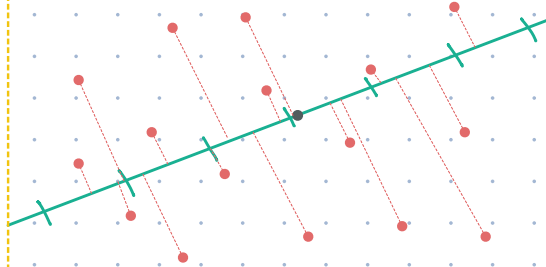
Lemma let $x, y \in S^{d-1}$

then $g \sim N^d$

$$P[\text{sign}(\langle g, x \rangle) \neq \text{sign}(\langle g, y \rangle)] = \frac{\angle(x, y)}{\pi}$$

Random line embedding & Buckets

$$h(x) = \lfloor \langle g, x \rangle + d \rfloor \text{ w/ } g \sim N^d, d \in [0, 1] \text{ unis. rand.}$$

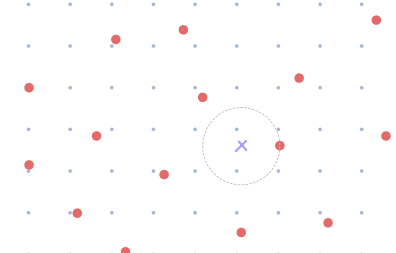


Algorithms

Approximate nearest neighbors

Nearest neighbor search

Input: n points $P \subset \mathbb{R}^d$



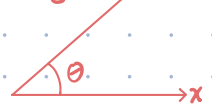
Query: given $q \in \mathbb{R}^d$, find closest point in P to q

Types of distances

- Euclidean distance

$$\|x - z\|_2 = \sqrt{\sum_i (x_i - z_i)^2}$$

- Angle



Lecture 10

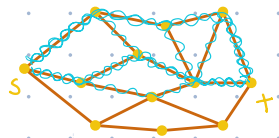
Probability (and techniques)

Algorithms

LP duality

L_1 -metric embeddings

Max. Flow
Min Cut



Path Packing LP

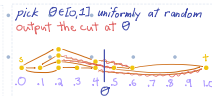
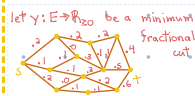
maximize $\sum_{p \in P} x_p$
over $x: P \rightarrow \mathbb{R}_{\geq 0}$
(all s-t paths)

s.t. $\sum_{p \in P} x_p \leq c_e$ for all edges e
(capacity of e)

Path Covering LP
(i.e. cut)

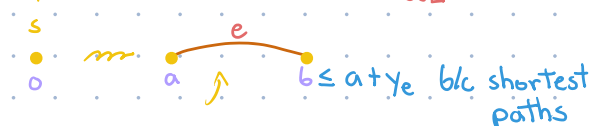
minimize $\sum_{e \in E} c_e y_e$
over $y: E \rightarrow \mathbb{R}_{\geq 0}$

s.t. $\sum_{e \in P} y_e \geq 1$ for all $p \in P$
(continuous analog of s-t cut)



$E[\text{capacity of cut}]$

$$= \sum_{e \in E} c_e \Pr[e \text{ cut}] \leq \sum_{e \in E} c_e y_e$$



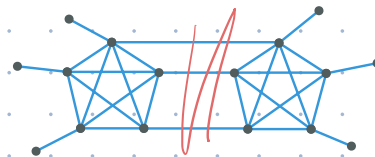
minimum capacity
of any fractional cut

Sparsest Cut

Special case: uniform sparsest cut

w/ uniform demands $b(u,v) = 1 \quad \forall u,v$

$$\text{"sparsity"} = \frac{c(\delta(s))}{|S| |T|}$$



useful primitive for graph partitioning, divide+conquer.

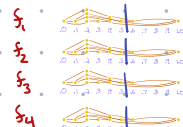
Q: how to round LP to sparse cut?

min $\sum_{e \in E} c_e d(u,v)$ over metrics $d: V \times V \rightarrow \mathbb{R}_{\geq 0}$
s.t. $\sum_{i,j \in T} b(i,j) d(i,j) \geq 1$

Suppose $d(u,v)$ was of the form

$$d(u,v) = \sum_{i=1}^k |\xi_i(u) - \xi_i(v)| \quad \text{for } \xi_i: V \rightarrow [0,1]^k$$

(assume $\min \xi_i(u) = 0, \max \xi_i(u) = 1$)



" L_1 -metric"

Lecture 11

Probability (and techniques)

Algorithms

Other metric-based problems

(sometimes via correspondence between cuts $\leftrightarrow$ metrics)

Buy-at-bulk network design
 k-median clustering
 metric labeling
 group Steiner tree
 metrical task systems
 vehicle routing
 hierarchical placement
 distributed k-server
 covering Steiner tree
 &



Approximation by tree metrics

Input: undirected $G=(V,E)$ w/ edge weights $w: E \rightarrow \mathbb{R}_{\geq 0}$

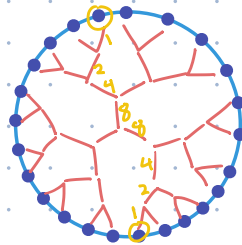
(High-level)
 Goal: compute tree T (containing V as nodes)
 so that shortest paths on T are close to T

$$d_T(u,v) \geq d_G(u,v)$$

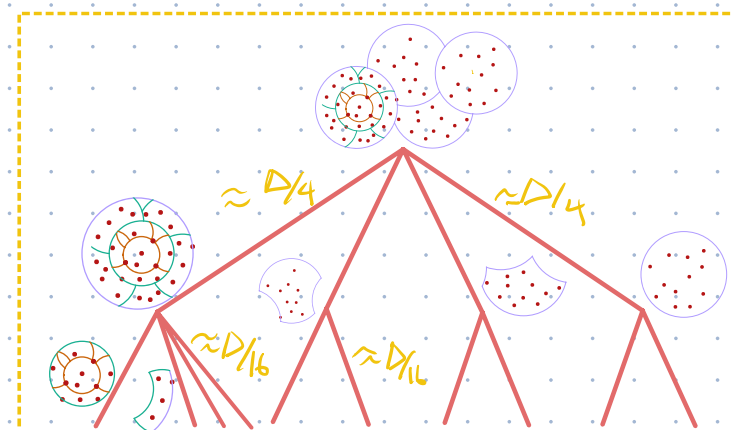
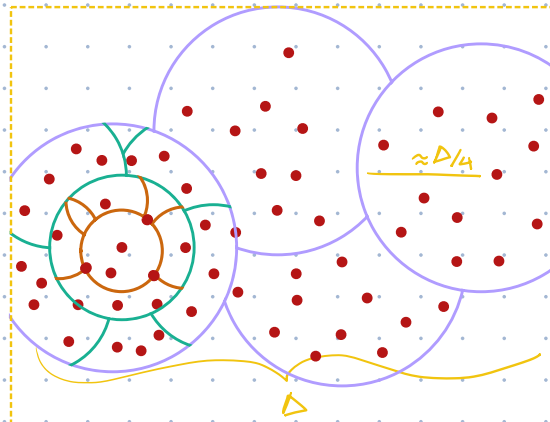
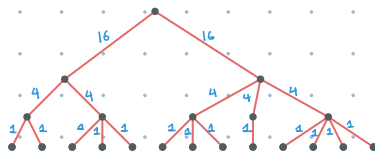
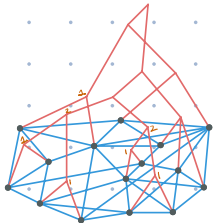
$$\text{For } e=f_u, v, \quad (\text{stretch } e) = \frac{d_T(u,v)}{d_G(u,v)}$$

(distance in T)
 (distance in G)

ideally
 $(\text{stretch } e) = \text{polylog}(n) \quad \forall e$



Theorem In randomized polynomial time,
 randomized hierarchical dominating tree metric w/
 $E[\text{stretch } e] \leq O(\log n) \quad \forall e \in E$



Lecture 12

Probability (and techniques)

Additive Chernoff Inequality

Double-sampling

VC dimension

Additive Chernoff bound

- let $X_1, \dots, X_n \in [0, 1]$ be independent
- $\mu = E[X_1 + \dots + X_n] / n$
- $\epsilon \in (0, 1)$
- $P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \leq e^{-2\epsilon^2 n}$
- $P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n}$

ϵ -sample $|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$

let Q sample k points w/ repetition
 $k = O\left(\frac{\ln(g(P))}{\epsilon^2}\right) \Rightarrow \epsilon$ -sample $k = O\left(\frac{\ln(g(P))}{\epsilon^2}\right) \Rightarrow \epsilon$ -net* (w/ constant probability)

ϵ -sample theorem [VC71]
 $k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon$ -sample (w/ constant probability)

e.g. $g(n) = O(n^d)$
 $k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon^2}\right) \Rightarrow \epsilon$ -sample
↑ independent of $|P|$ (!)

$\mu(r) = \frac{|P \cap r|}{|P|}$
 $\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$
 $R \Delta Q = \{r \cap q : r \in R, q \in Q\}$
 $P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n}$ $g(n) = \max\{|R \Delta Q| : Q \in \mathcal{Q}, |Q| \leq n\}$

Algorithms

ϵ -samples and ϵ -nets

ϵ -sample $|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$

let Q sample k points w/ repetition
 $k = O\left(\frac{\ln(g(P))}{\epsilon^2}\right) \Rightarrow \epsilon$ -sample $k = O\left(\frac{\ln(g(P))}{\epsilon^2}\right) \Rightarrow \epsilon$ -net* (w/ constant probability)

ϵ -net theorem [HW87]
 $k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon$ -net (w/ constant probability)

e.g. $g(n) = O(n^d)$
 $k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon^2}\right) \Rightarrow \epsilon$ -net
↑ independent of $|P|$ (!)

$\mu(r) = \frac{|P \cap r|}{|P|}$
 $\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$
 $R \Delta Q = \{r \cap q : r \in R, q \in Q\}$
 $P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n}$ $g(n) = \max\{|R \Delta Q| : Q \in \mathcal{Q}, |Q| \leq n\}$

Additive Chernoff bound

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]/n$$

$$\epsilon \in (0, 1)$$

$$\bullet \quad P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \leq e^{-2\epsilon^2 n}$$

$$\bullet \quad P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n}$$

(proof at end if time permits)

ϵ -sample

Q is an ϵ -sample if

$$|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$$

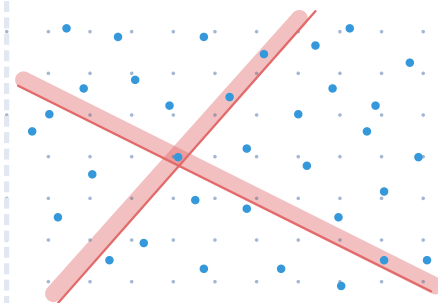
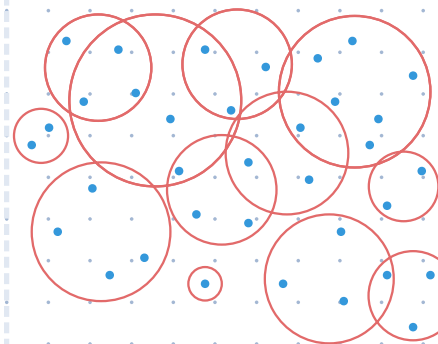
ϵ -net

Q is an ϵ -net if

$$\mu_Q(r) > 0 \text{ whenever } \mu(r) \geq \epsilon$$

(ϵ -sample $\Rightarrow$ ϵ -net)

(P, R)
points ranges
(subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

(!) $|Q| < (1-\epsilon)|P|$ is impossible w/out any assumptions on R

e.g. $R = 2^P$ (=all subsets of P)

many natural range spaces cannot induce (nearly) all subsets of P

let $R \wedge Q = \{r \cap Q : r \in R\}$ for $Q \subseteq P$

growth function $g: \mathbb{N} \rightarrow \mathbb{N}$

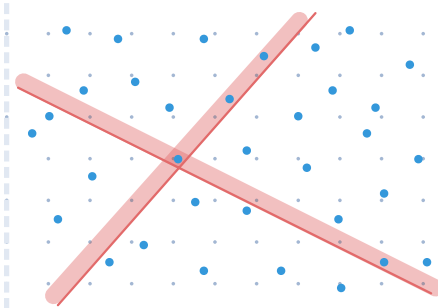
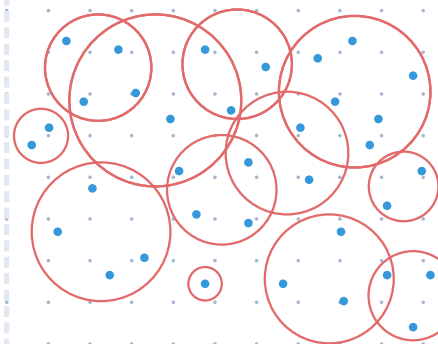
$$g(n) = \max \{ |R \wedge Q| : Q \subseteq P, |Q| \leq n \}$$

e.g. disks have $g(n) \leq O(n^3)$ (which we prove later)

(P, R)

points

ranges
(subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

ϵ -sample $|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$

let Q sample k points w/ repetition

$$k = O\left(\frac{\ln(g(|P|))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample} \quad k = O\left(\frac{\ln(g(|P|))}{\epsilon}\right) \Rightarrow \epsilon\text{-net} \quad (\text{w/ constant probability})$$

ϵ -sample theorem

[VC71]

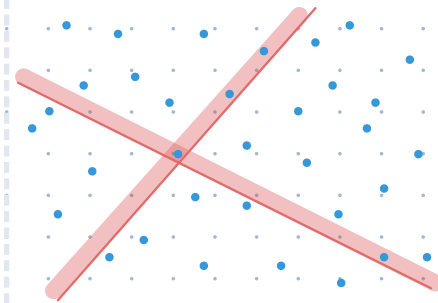
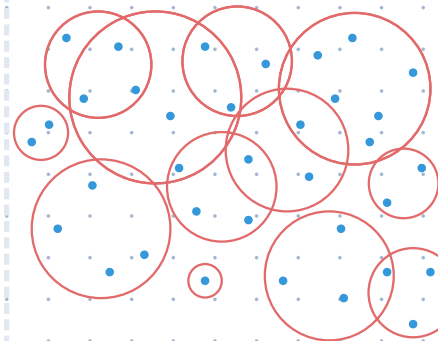
$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample} \quad (\text{w/ constant probability})$$

e.g. $g(n) = O(n^d)$

$$k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample}$$

↑ independent of $|P|$ (!)

(P, R)
 ↑ points ↑ ranges
 (subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

$$R \wedge Q = \{r \cap Q : r \in R\}$$

$$P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n} \quad g(n) = \max \{|R \wedge Q| : Q \subseteq P, |Q| \leq n\}$$

ϵ -sample $|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$

let Q sample k points w/ repetition

$$k = O\left(\frac{\ln(g(|P|))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample} \quad k = O\left(\frac{\ln(g(|P|))}{\epsilon}\right) \Rightarrow \epsilon\text{-net} \quad (\text{w/ constant probability})$$

ϵ -net theorem

[HW87]

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

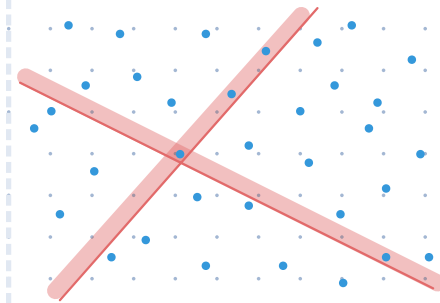
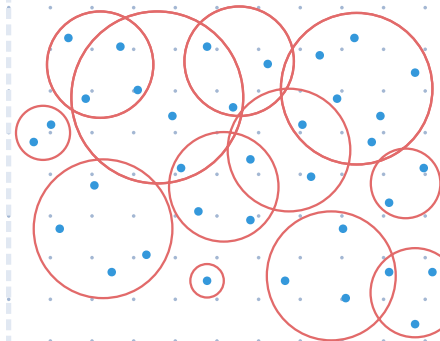
(w/ constant probability)

e.g. $g(n) = O(n^d)$

$$k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

↑ independent of $|P|$ (!)

(P, R)
 ↑ points ↑ ranges
 (subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

$$R \wedge Q = \{r \cap Q : r \in R\}$$

$$P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n} \quad g(n) = \max \{|R \wedge Q| : Q \subseteq P, |Q| \leq n\}$$

Lecture 13

Probability (and techniques)

Algorithms (models and applications)

Consistency model

PAC learning model

Generalization error

Occam's Razor

Generalization by growth

(Example)

Consistency model

- Domain X (space of examples)
 - Labels $\{0,1\}$
 - Concepts $c: X \rightarrow \{0,1\}$
 - Concept class $\mathcal{C}$ (family of concepts)
-

Assume X is labeled by unknown concept c
from known concept class $\mathcal{C}$

(High-level)

goal: identify unknown concept from
labeled subset of data

Consistency model

- Domain X (space of examples)
- Labels $\{0,1\}$
- Concepts $c: X \rightarrow \{0,1\}$
- Concept class $\mathcal{C}$ (family of concepts)

Assume X is labeled by unknown c from known $\mathcal{C}$
(High-level)
goal: identify unknown c from labeled data

Input: training set $(x_1, y_1), \dots, (x_m, y_m)$ w/ $y_i = c(x_i)$

a concept $c' \in \mathcal{C}$ is consistent if $c'(x_i) = y_i \quad \forall i$

a learning algo in the consistency model either

- outputs a consistent $c' \in \mathcal{C}$, or
- decides no such concept exists

e.g. Disjunctions $f(x_1, \dots, x_n) = x_{i_1} \vee \bar{x}_{i_2} \vee \dots$

$X = \{0, 1\}^n$, concepts $c \in \mathcal{C}$ defined by disjunctions

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

True concept: $c^*(x_1, \dots, x_n) = \bar{x}_2 \vee \bar{x}_5$

e.g. Disjunctions $X = \{0,1\}^n$, concepts $c \in \mathcal{C}$ are disjunctions

Learning algo: given $\{(x_1, y_1), \dots, (x_m, y_m)\}$, define

$$C(z_1, \dots, z_n) = z_{i_1} \vee \bar{z}_{i_2} \vee \dots \text{ where:}$$

- include z_j if $x_{i,j} = 0 \ \forall \ y_i = 0$
- include $\bar{z}_j$ if $x_{i,j} = 1 \ \forall \ y_i = 0$

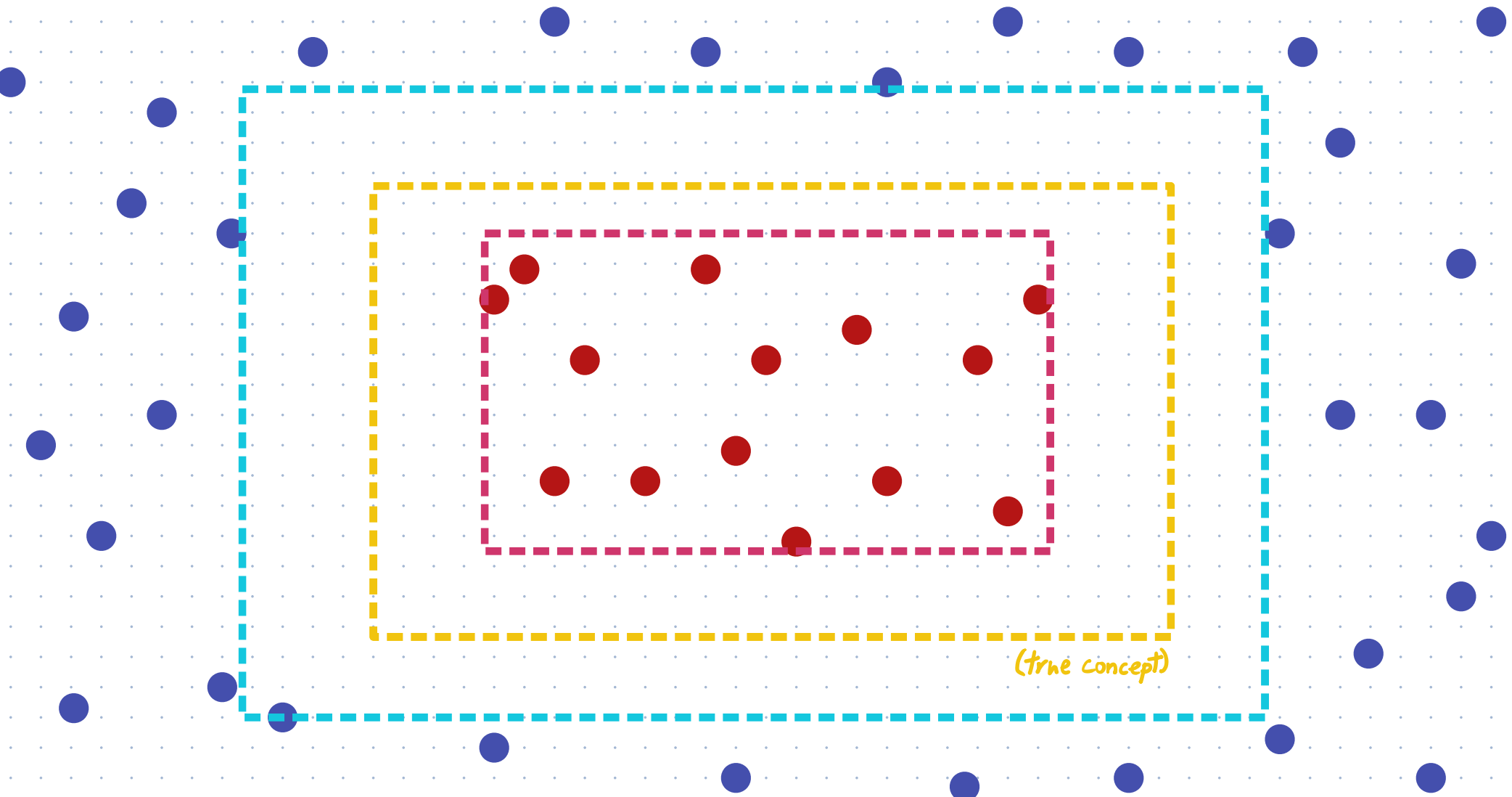
x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

$$\Rightarrow c(z) = \bar{z}_2 \vee \bar{z}_4 \vee \bar{z}_5$$

claim: if $c^*(z_1, \dots, z_n)$ is true concept,
then c^* is a "subformula" of c
(contains subset of symbols)

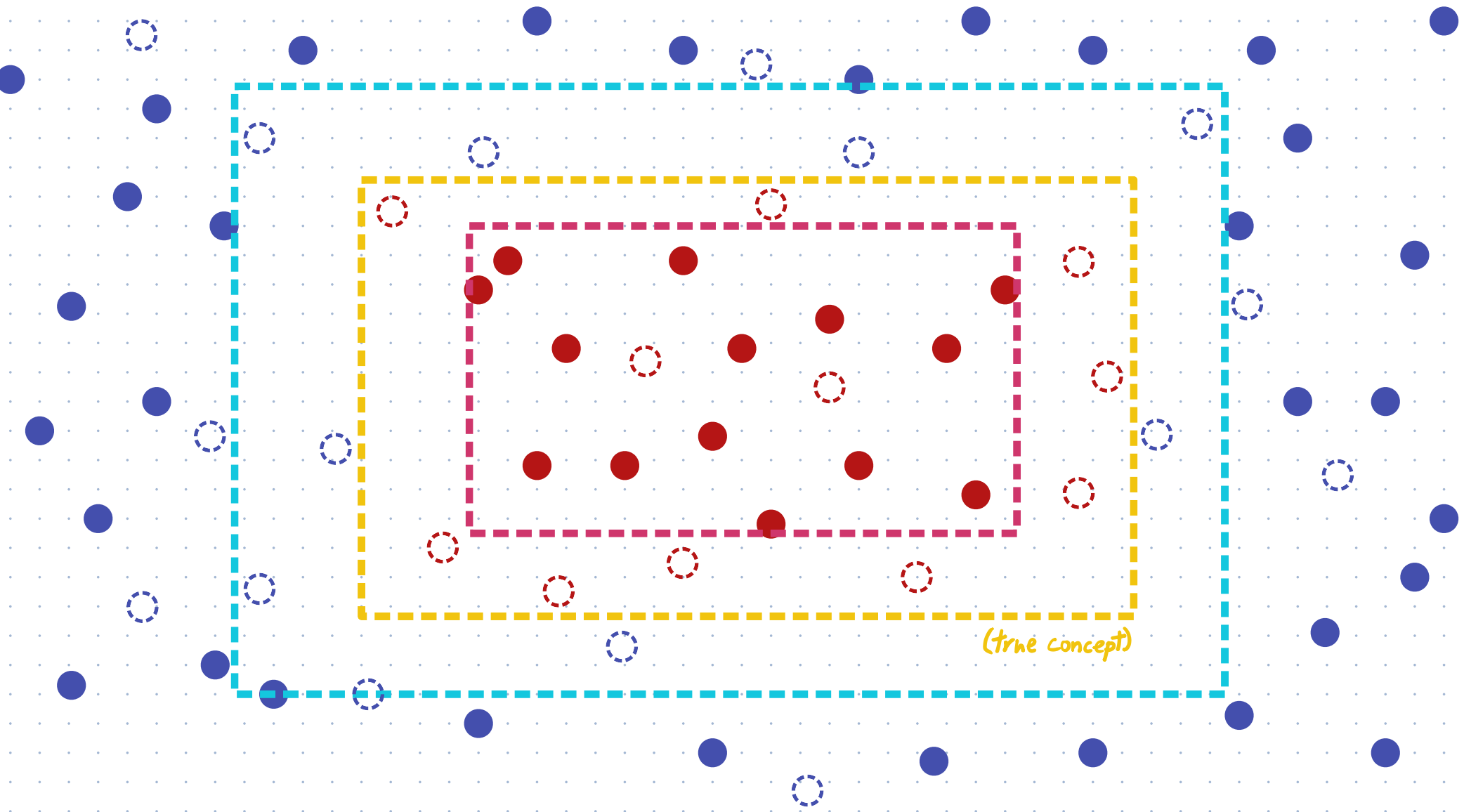
corollary: learning algo is consistent

e.g. Rectangles $X=[0,1]^2$, concepts induced by rectangles



consistent learner: output **smallest** or **largest** rectangle containing exactly the positively labeled points

e.g. Rectangles $X=[0,1]^2$, concepts induced by rectangles



consistent learner: output *smallest* or *largest* rectangle containing exactly the positively labeled points

Are we really "learning"?

Consistency model

- Domain X (space of examples)
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$ (family of concepts)

Assume X is labeled by unknown c from known $\mathcal{C}$

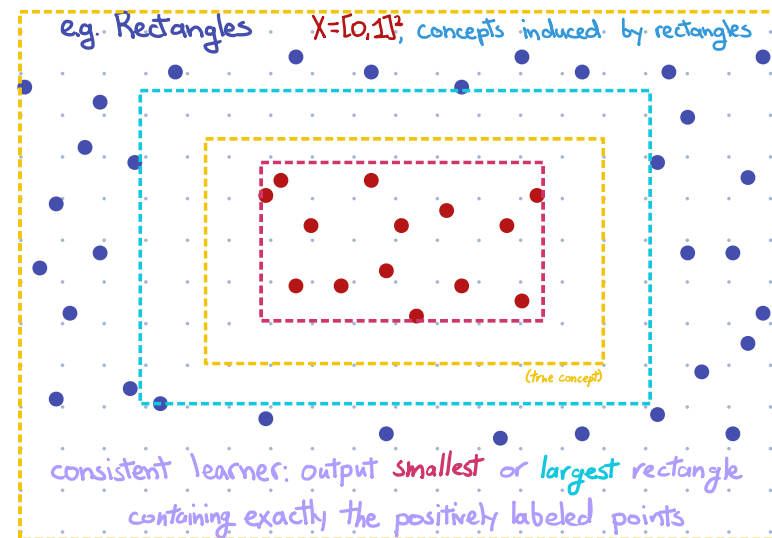
(high-level) goal: identify unknown c from labeled data

Input: training set $(x_1, y_1), \dots, (x_m, y_m)$ w/ $y_i = c(x_i)$

a concept $c' \in \mathcal{C}$ is consistent if $c'(x_i) = y_i \quad \forall i$

a learning algo in the consistency model either

- outputs a consistent $c' \in \mathcal{C}$, or
- decides no such concept exists



e.g. Disjunctions $X = \{0,1\}^n$, concepts $c \in \mathcal{C}$ are disjunctions

Learning algo: given $\{(x_i, y_i), \dots, (x_m, y_m)\}$, define

$c(z_1, \dots, z_n) = z_{i_1} \vee \bar{z}_{i_2} \vee \dots$ where:

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

- include z_j if $x_{i,j} = 0 \quad \forall y_i = 0$

- include $\bar{z}_j$ if $x_{i,j} = 1 \quad \forall y_i = 0$

claim: if $c^*(z_1, \dots, z_n)$ is true concept, then c^* is a "subformula" of c (contains subset of symbols)

$\Rightarrow c(z) = \bar{z}_2 \vee \bar{z}_4 \vee \bar{z}_5$

corollary: learning algo is consistent

consistency model does not ask
output concept c' to match
true concept c^* outside of training set.

leads to overfitting

(probably approximately correct)

Valiant, 1984

PAC learning

- Domain X (space of examples)
- Labels $\{0,1\}$
- Concepts $c: X \rightarrow \{0,1\}$
- Concept class $\mathcal{C}$ (family of concepts)

Assume X is labeled by unknown c from known $\mathcal{C}$

additionally:

- family of hypotheses $\mathcal{H}$ (family of functions $h: X \rightarrow \{0,1\}$
(possibly $\mathcal{H} = \mathcal{C}$)
- Assume points $x \in X$ drawn from fixed distribution $\mathcal{D}$
(but unknown)

for hypothesis $h \in \mathcal{H}$, define true classification error as

$$\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

(High-level)

goal: find $h \in \mathcal{H}$ minimizing $\text{err}(h)$

(probably approximately correct)

Valiant, 1984

PAC learning

- Domain X (space of examples)
- Labels $\{0,1\}$
- Concepts $c: X \rightarrow \{0,1\}$
- Concept class $\mathcal{C}$ (family of concepts)

Assume X is labeled by unknown c from known $\mathcal{C}$

- Hypotheses $\mathcal{H}$
- $x \in X$ from fixed distribution $\mathcal{D}$ (but unknown)
- $\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$

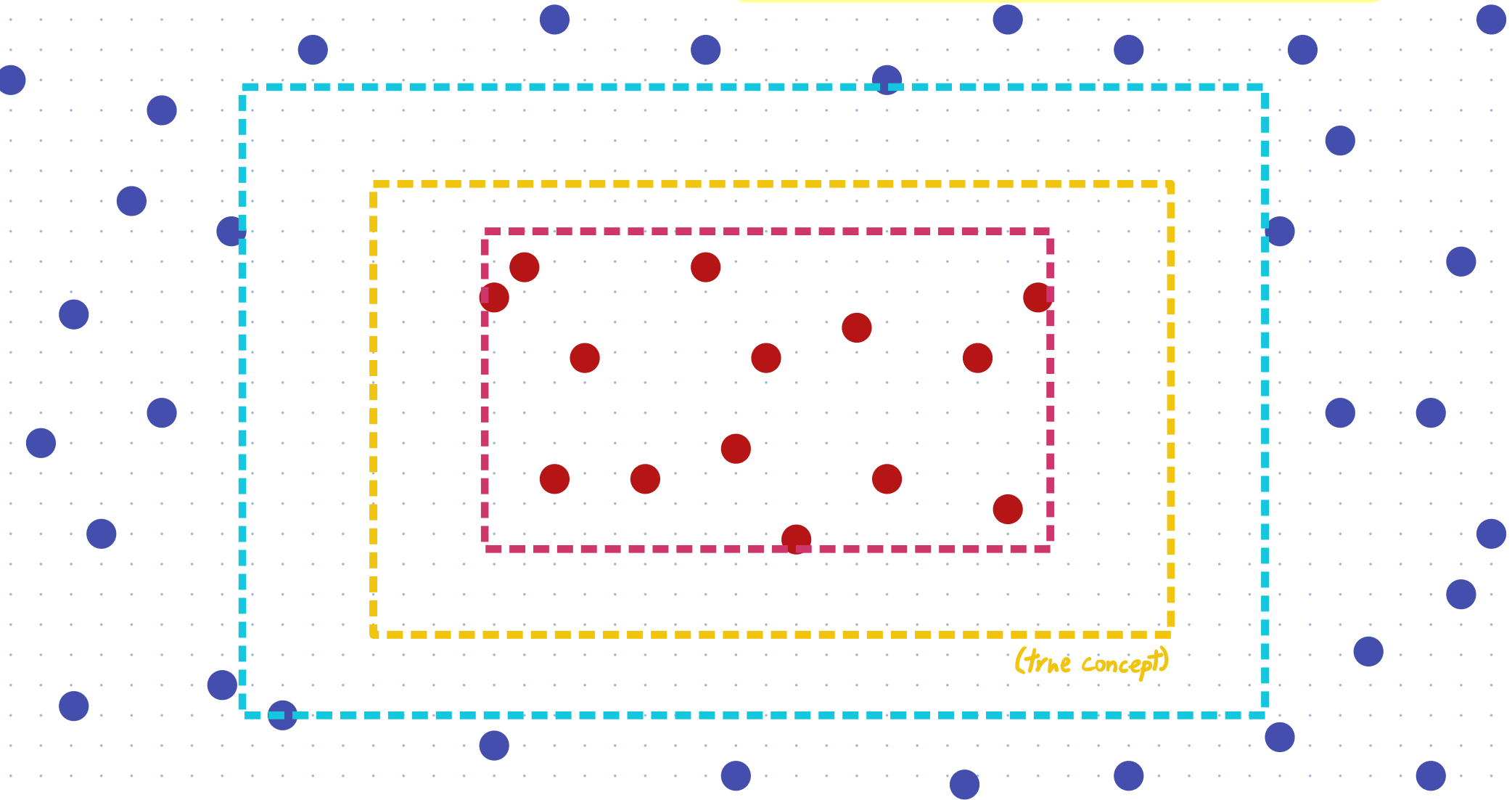
"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$ w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,

hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

e.g. Rectangles $X=[0,1]^2$, $\mathcal{D}=\text{uniform over } [0,1]^2$



For large m , sample more points closer to boundary
of true concept $\Rightarrow$ smallest, largest rectangles converge
(analysis via VC dimension later)

Generalization for finite hypotheses

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • Fixed distribution $\mathcal{D}$
(not unknown)
- $\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Theorem

for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob. $\geq 1 - \delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

Corollary consistent learning $\Rightarrow$ PAC learning
for fixed and finite $|\mathcal{H}|$

Theorem

for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob. $\geq 1-\delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

proof

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$
- Fixed distribution $\mathcal{D}$
(but unknown)
- $\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1-\delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$ • Concept class $\mathcal{C}$
unknown $\mathcal{C}$ from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

e.g. Disjunctions $f(x_1, \dots, x_n) = x_{i_1} \vee \bar{x}_{i_2} \vee \dots$

$X = \{0, 1\}^n$, concepts $c \in \mathcal{C}$ defined by disjunctions

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

True concept: $c^*(x_1, \dots, x_n) = \bar{x}_2 \vee \bar{x}_5$

e.g. Disjunctions

$X = \{0, 1\}^n$, concepts $c \in C$ are disjunctions

Learning algo: given $\{(x_1, y_1), \dots, (x_m, y_m)\}$, define

$$C(z_1, \dots, z_n) = z_{i_1} \vee \bar{z}_{i_2} \vee \dots \text{ where:}$$

- include z_j if $x_{i,j} = 0 \ \forall \ y_i = 0$
- include $\bar{z}_j$ if $x_{i,j} = 1 \ \forall \ y_i = 0$

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

$$\Rightarrow c(z) = \bar{z}_2 \vee \bar{z}_4 \vee \bar{z}_5$$

claim: if $c^*(z_1, \dots, z_n)$ is true concept,
then c^* is a "subformula" of c
(contains subset of symbols)

corollary: learning algo is consistent

Theorem

for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob. $\geq 1-\delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

for disjunctions,

$$|\mathcal{H}| \leq$$

e.g. Disjunctions $f(x_1, \dots, x_n) = x_{i_1} \vee \bar{x}_{i_2} \vee \dots$

$X = \{0,1\}^n$, concepts $c \in \mathcal{C}$ defined by disjunctions

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

True concept: $c^*(x_1, \dots, x_n) = \bar{x}_2 \vee \bar{x}_5$

Learning algo: given $\{(x_1, y_1), \dots, (x_m, y_m)\}$, define

$c(z_1, \dots, z_n) = z_{i_1} \vee \bar{z}_{i_2} \vee \dots$ where:

- include z_j if $x_{j,1} = 0 \ \forall y_i = 0$
- include $\bar{z}_j$ if $x_{j,1} = 1 \ \forall y_i = 0$

x_i	y_i
(0, 1, 1, 0, 1)	0
(0, 0, 1, 0, 1)	1
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 1)	0
(1, 1, 0, 0, 0)	1

claim: if $c^*(z_1, \dots, z_n)$ is true concept,
then c^* is a "subformula" of c

(contains subset of symbols)

$$\Rightarrow c(z) = \bar{z}_2 \vee \bar{z}_5 \vee \bar{z}_5$$

corollary: learning algo is consistent

Uniform convergence for finite $\mathcal{H}$

for fixed $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$,

define $\text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$

Theorem: for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon^2)$,
w/ prob $\geq 1-\delta$, we have

$$|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \quad \forall h \in \mathcal{H}$$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • Fixed distribution $\mathcal{D}$
(but unknown)
- $\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1-\delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

Implication: algo minimizing $\text{err}_S(h)$ is PAC learner

Theorem: for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon^2)$,
w/ prob $\geq 1-\delta$, we have

$$|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \quad \forall h \in \mathcal{H}$$

Proof:

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$
- Fixed distribution $\mathcal{D}$
(but unknown)

$$\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1-\delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$ • Concept class $\mathcal{C}$
unknown $\mathcal{C}$ from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\bullet \text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\bullet \text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

Occam's razor

- Aristotle: "Nature operates in the shortest way possible."
- William of Occam: "Plurality is never to be posited without necessity."
- Isaac Newton: "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances. Therefore, to the same natural effects we must, as far as possible, assign the same causes."
- Bertrand Russell: "Whenever possible, substitute constructions out of known entities for inferences to unknown entities."
- Albert Einstein: "Everything should be made as simple as possible, but not simpler."
- Richard Feynman: "The truth always turns out to be simpler than you thought."
- The Big Aristotle: "If you don't stick to simplicity, you'll die a horrible death."

(probably approximately correct)
PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$ • Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • Fixed distribution $\mathcal{D}$
(not unknown)

- $\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$
- $\text{err}_{\mathcal{S}}(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Occam's razor

for $h \in \mathcal{H}$, let $|h| = \# \text{ bits to encode } h$

Theorem: suppose $|h| \leq b \ \forall h \in \mathcal{H}$.

• for $m \geq O((b + \ln(1/\delta))/\epsilon)$, w/ prob $\geq 1 - \delta$,
all consistent $h \in \mathcal{H}$ have $\text{err}(h) \leq \epsilon$

• for $m \geq O((b + \ln(1/\delta))/\epsilon^2)$, w/ prob $\geq 1 - \delta$,
 $|\text{err}_S(h) - \text{err}(h)| \leq \epsilon \ \forall h \in \mathcal{H}$

(probably, approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0, 1\}$
- Labels $\{0, 1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0, 1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

Thm. for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob $\geq 1 - \delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

Theorem: for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon^2)$,
w/ prob $\geq 1 - \delta$, we have
 $|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \ \forall h \in \mathcal{H}$

- Aristotle: "Nature operates in the shortest way possible."
- William of Occam: "Plurality is never to be posited without necessity."
- Isaac Newton: "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances. Therefore, to the same natural effects we must, as far as possible, assign the same causes."
- Bertrand Russell: "Whenever possible, substitute constructions out of known entities for inferences to unknown entities."
- Albert Einstein: "Everything should be made as simple as possible, but not simpler."
- Richard Feynman: "The truth always turns out to be simpler than you thought."
- The Big Aristotle: "If you don't stick to simplicity, you'll die a horrible death."

Generalization bounds by growth

each hypothesis $h \in \mathcal{H}$

$$\Rightarrow \text{range } h^{-1}(1) = \{x \in X : h(x) = 1\}$$

if $\mathcal{H}$ has simple geometry,
can we get better bounds?

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\bullet \text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\bullet \text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

Thm. for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob. $\geq 1 - \delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

Theorem: for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon^2)$,
w/ prob $\geq 1 - \delta$, we have
 $|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \quad \forall h \in \mathcal{H}$

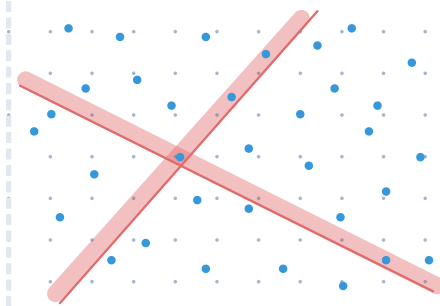
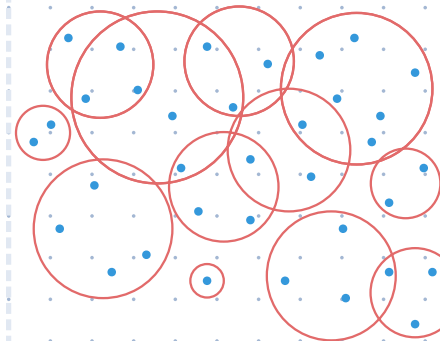
let $R \wedge Q = \{r \cap Q : r \in R\}$ for $Q \subseteq P$

growth function $g: \mathbb{N} \rightarrow \mathbb{N}$

$$g(n) = \max \{ |R \wedge Q| : Q \subseteq P, |Q| \leq n \}$$

e.g. disks have $g(n) \leq O(n^3)$ (which we prove later)

(P, R)
 ↑ points ↑ ranges
 (subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

ϵ -sample

Q is an ϵ -sample if

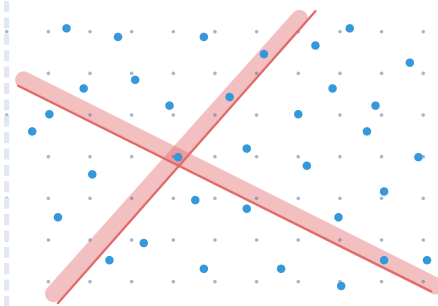
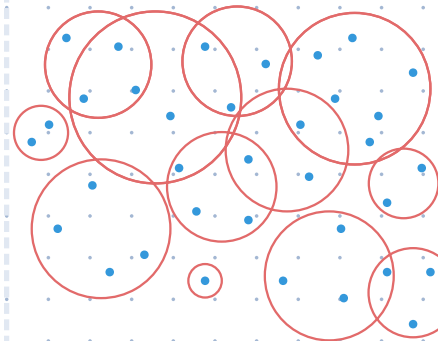
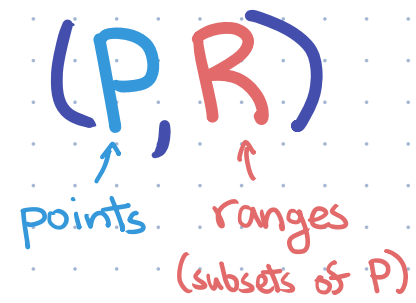
$$|\mu_Q(r) - \mu(r)| \leq \epsilon \quad \forall r \in R$$

ϵ -net

Q is an ϵ -net if

$$\mu_Q(r) > 0 \text{ whenever } \mu(r) \geq \epsilon$$

(ϵ -sample $\Rightarrow$ ϵ -net)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

ϵ -sample theorem

[VC71]

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample}$$

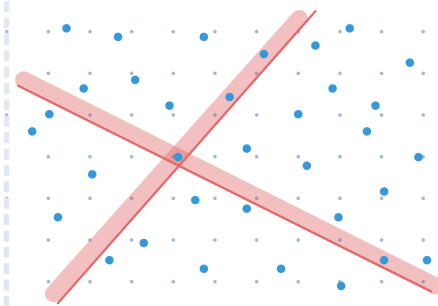
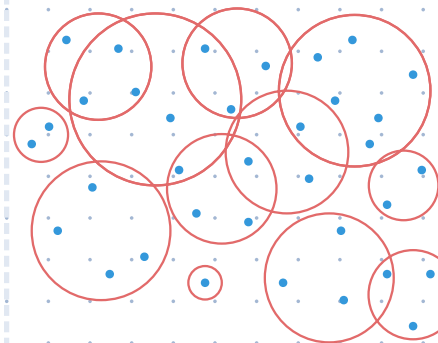
(w/ constant probability)

e.g. $g(n) = O(n^d)$

$$k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample}$$

↑ independent of $|P|$ (!)

(P, R)
 ↑ points ↑ ranges
 (subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

$$R \wedge Q = \{r \cap Q : r \in R\}$$

$$P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n}$$

$$g(n) = \max \{|R \wedge Q| : Q \subseteq P, |Q| \leq n\}$$

ϵ -net theorem

[HW87]

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

(w/ constant probability)

e.g. $g(n) = O(n^d)$

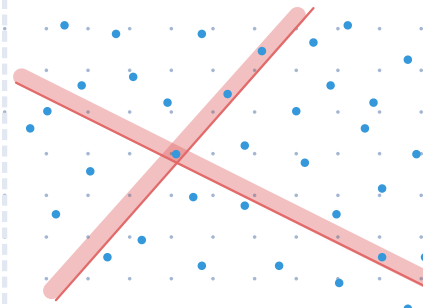
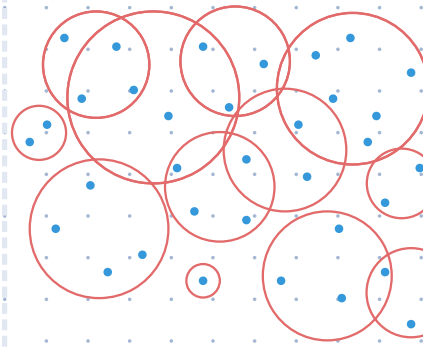
$$k \geq O\left(\frac{d \ln(d/\epsilon)}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

↑ independent of $|P|$ (!)

(P, R)

points

ranges
(subsets of P)



$$\mu(r) = \frac{|P \cap r|}{|P|}$$

$$\mu_Q(r) = \frac{|Q \cap r|}{|Q|}$$

$$R \wedge Q = \{r \cap Q : r \in R\}$$

$$P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right], P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \leq e^{-2\epsilon^2 n} \quad g(n) = \max \{|R \wedge Q| : Q \subseteq P, |Q| \leq n\}$$

Generalization bounds by growth

Theorem: let $\mathcal{H}$ have growth $g(n)$

• if $m \geq O(\ln(g(2m)/\delta)/\epsilon)$, then w/ prob $1-\delta$, all consistent $h \in \mathcal{H}$ have $\text{err}(h) \leq \epsilon$

• if $m \geq O(\ln(g(2m)/\delta)/\epsilon)$, then

$|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \quad \forall h \in \mathcal{H}$ w/ prob $1-\delta$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$
- Concept class $\mathcal{C}$
unknown c from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • Fixed distribution $\mathcal{D}$
(but unknown)

$$\text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1-\delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

Additive Chernoff

$$\left. \begin{aligned} P\left[\frac{X_1 + \dots + X_n}{n} \geq \mu + \epsilon\right] \\ P\left[\frac{X_1 + \dots + X_n}{n} \leq \mu - \epsilon\right] \end{aligned} \right\} \leq e^{-2\epsilon^2 n}$$

Thm. for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon)$, w/ prob. $\geq 1-\delta$,
all consistent $h \in \mathcal{H}$ has $\text{err}(h) \leq \epsilon$

Theorem: for $m \geq O(\ln(|\mathcal{H}|/\delta)/\epsilon^2)$,
w/ prob $\geq 1-\delta$, we have
 $|\text{err}(h) - \text{err}_S(h)| \leq \epsilon \quad \forall h \in \mathcal{H}$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$ • Concept class $\mathcal{C}$
unknown $\mathcal{C}$ from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\bullet \text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\bullet \text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample}$$

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

(probably approximately correct)

PAC learning

- Domain X
- Concepts $c: X \rightarrow \{0,1\}$
- Labels $\{0,1\}$ • Concept class $\mathcal{C}$
unknown $\mathcal{C}$ from known $\mathcal{C}$
- Hypotheses $\mathcal{H}$ • fixed distribution $\mathcal{D}$
(but unknown)

$$\bullet \text{err}(h) = \Pr_{x \sim \mathcal{D}} [h(x) \neq c(x)]$$

$$\bullet \text{err}_S(h) = \frac{1}{m} |\{i: h(x_i) \neq y_i\}|$$

"PAC learning algo" for $\epsilon, \delta \in (0,1)$

Input: $(x_1, y_1), \dots, (x_m, y_m) \sim \mathcal{D}$
w/ $m = \text{poly}(1/\epsilon, 1/\delta)$

Output: w/ probability $\geq 1 - \delta$,
hypothesis $h \in \mathcal{H}$ w/ $\text{err}(h) \leq \epsilon$

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon^2}\right) \Rightarrow \epsilon\text{-sample}$$

$$k \geq O\left(\frac{\ln(g(2k))}{\epsilon}\right) \Rightarrow \epsilon\text{-net}$$

Chapter 13

PAC learning

In this chapter we discuss foundations of learning theory particularly as it relates to randomized analysis. We start from the *consistent learning model*, which is appealingly simple but does not capture *generalization error*. This leads to the *probabilistic approximately correct (PAC)* learning model, which crucially incorporates randomization to correct this deficiency. From there we explore the notion of generalization error and study its correspondence to concentration inequalities that we have already studied.

13.1 The consistency model

In this chapter we are interested *classification algorithms* that assign labels to unlabeled data, generally based on previous experience analyzing a limited set of labeled training data. Instead of the design and analysis of such algorithms, we focus on developing *models* that allow us to more sensibly declare if such an algorithm is really “learning” or not.

Our first model of learning is called the *consistency model*. Let X denote the space of all possible examples, also called the *domain*. We consider binary classification problems where each $x \in X$ is labeled 0 or 1. A mapping $c : X \rightarrow \{0, 1\}$ from examples to labels is called a *concept*. We assume that X is labeled by an unknown concept c from a known collection of concepts $\mathcal{C} \subseteq 2^X$, also called a *concept class*.

The high-level goal is to identify the underlying true concept c given access to a limited set of data. More precisely, the input consists of a set of labeled examples

$$(x_1, y_1), \dots, (x_m, y_m)$$

where $x_i \in X$ and $y_i = c(x_i) \in \{0, 1\}$. A concept c' is *consistent* if $c'(x_i) = y_i$ for all i . A *learning algorithm in the consistency model* is one that, given the labeled examples above, either

- (a) outputs a concept $c' \in \mathcal{C}$ consistent with the examples, or

(b) decides that no such concept exists.

The concept class $\mathcal{C}$ is *learnable in the consistency model* if there is a learning algorithm in the consistency model for it.

Of course we are only interested in *efficient* learning algorithms. The concept class $\mathcal{C}$ is typically extremely large or even infinite, so directly inspecting all concepts in $\mathcal{C}$ is too slow.

It is obviously desirable to be able to identify a concept $c' \in \mathcal{C}$ that agrees with all the labeled input data. However, the consistency model does not ask c' to agree the true concept c beyond the labeled data. This can lead to some undesirable behavior, such as overfitting, as in the following examples.

Example: disjunctions. Recall that a boolean formula $f(x_1, \dots, x_n)$ is a *disjunction* if it is the disjunction ($\vee$) of x_i 's and $\bar{x}_i$'s. Let $X = \{0, 1\}^n$, and let $\mathcal{C}$ be the class of disjunctions over n -variables. That is, each $c \in \mathcal{C}$ corresponds to a disjunction $f_c(x_1, \dots, x_n) = x_{i_1} \vee \bar{x}_{i_2} \vee \dots \vee \bar{x}_{i_k}$, where $c(x) = 1$ iff $f_c(x) = \mathbf{true}$.

Given a set of labeled points $\{(x_i, y_i)\}$ in $\mathbb{R}^d$, we can define a disjunction $f(x_1, \dots, x_n) = x_{j_1} \vee \dots \vee x_{j_k}$ where we include:

(a) Variable x_j iff $x_{i,j} = 1$ for all samples x_i with $y_i = 1$.

(b) Negated variable $\bar{x}_j$ iff $x_{i,j} = 0$ for all samples x_i with $y_i = 0$.

Observe that f will correctly label all of the data in the sample, and the algorithm to construct f is efficient. Thus this concept class is efficiently learnable in the consistency model.

In general, if $f^*(x_1, \dots, x_n)$ is the disjunction defining the true concept c^* , then the function f we construct is a “superset” of f^* in terms of the symbols x_i and $\bar{x}_i$ it contains. Therefore if f is much larger than f^* , then it will overfit the training sample and generally label many more x 's as **true** than f^* .

Example: rectangles. Let $X = [0, 1]^2$, and let $\mathcal{C}$ be the set of concepts defined by rectangles in $[0, 1]^2$. That is, each $c \in \mathcal{C}$ is associated with a rectangle $R_c = [a, b] \times [c, d]$, and $c(x) = 1$ iff $x \in R_c$.

Given a set of labeled points $\{(x_i, y_i)\}$, it is easy to identify the smallest rectangle R_1 containing all (and only) the x_i 's labeled $y_i = 1$. Likewise one can identify the largest rectangle R_2 containing the same x_i 's. So by outputting either R_1 or R_2 (or anything in between), $(X, \mathcal{C})$ is efficiently learnable in the consistency model.

All we know about the true concept is that it is induced by a rectangle R^* such that $R_1 \subseteq R^* \subseteq R_2$. If the difference between R_1 and R_2 is very big (particular when the data is limited and/or not uniformly spread), then neither R_1 nor R_2 may be a very good classifier on real data.

13.2 The PAC learning model

The problem with the consistency model is that it does not incorporate *generalization* beyond the given training examples. This weakness is address in our next model, called the *probably approximately correct (PAC) learning* model and introduced in [Val84a; Val84b].

The PAC learning model is an extension of the consistent learning model. We retain the same setup where X denotes the space of examples, $\{0, 1\}$ the set of labels, and $\mathcal{C}$ the concept class. The new ingredients are as follows. First, we assume that the training examples are independently and identically distributed from a fixed (but unknown) distribution $\mathcal{D}$. Second, we define a second family of labeling functions, $\mathcal{H} \subseteq 2^X$, called the *hypothesis class*. (Possibly $\mathcal{H} = \mathcal{C}$.) Our algorithms output a hypothesis $h \in \mathcal{H}$ that, ideally, agrees as much as possible with the unknown concept c . The classification error of a hypothesis h is measured by

$$\text{err}(h) \stackrel{\text{def}}{=} \mathbf{P}_{x \sim \mathcal{D}}[h(x) \neq c(x)].$$

We now define a *PAC learning algorithm* for $(\mathcal{C}, \mathcal{D})$. Let $\epsilon, \delta \in (0, 1)$ be fixed parameters. A PAC learning algorithm takes as input $m = \text{poly}(1/\epsilon, 1/\delta)$ (labeled) samples drawn from $\mathcal{D}$, and produces a hypothesis h . The output hypothesis h is randomized since it reflects a randomized training set; besides, the algorithm itself is allowed to use additional randomization. To meet the criteria of PAC learning, the hypothesis h must have classification error at most ϵ with probability $1 - \delta$. This requirement can be stated compactly as:

$$\mathbf{P}_h[\text{err}(h) \leq \epsilon] = \mathbf{P}_h \left[\mathbf{P}_{x \sim \mathcal{D}}[h(x) \neq c(x)] \leq \epsilon \right] \geq 1 - \delta. \quad (13.1)$$

(13.1) is a mathematical expression of generalization. Unlike the consistency model of learning, PAC learning does not require h to be particularly accurate on the sample set. (13.1) is only concerned with h 's performance with respect to the entire distribution $\mathcal{D}$.

Example: rectangles. Recall the earlier example where the concepts are induced by rectangles in $[0, 1]^2$. There we observed that we have efficient consistent learning algorithms by outputting either the smallest rectangle R_1 or largest rectangle R_2 containing all of the positively labeled examples. We also observed that the true rectangle R^* lied somewhere between R_1 and R_2 , which is a weak guarantee if R_1 and R_2 differ greatly.

Suppose $\mathcal{D}$ drew points $x \in [0, 1]^2$ uniformly from $[0, 1]$. As the number of samples m grows large, we expect points that our closer and closer to each side of the boundary

of R^* , and R_1 and R_2 should converge over time. One can show that this concept is PAC learnable by directly calculating the generalization error for either the smallest or largest enclosing rectangle. We will prove this later (perhaps with weaker constants) when we more generally analyze concept classes based on their VC dimension.

13.3 Generalization for finite hypothesis classes

We first analyze the generalization error for a learning algorithm in the consistent learning model, as a function of the number of hypotheses, $|\mathcal{H}|$. In particular these bounds only hold when $\mathcal{H}$ is finite. In the following theorem, we need m to be proportional to $\ln(|\mathcal{H}|)$ to guarantee low generalization error.

Theorem 13.1. *Let $\epsilon, \delta \in (0, 1)$ and consider a training set of size m drawn from $\mathcal{D}$. If $m \geq \ln(|\mathcal{H}|/\delta)/\epsilon$, then with probability at least $1 - \delta$, every consistent hypothesis $h \in \mathcal{H}$ has true error $\text{err}(h) \leq \epsilon$.*

Proof. Call a hypothesis $h \in \mathcal{H}$ *good* if $\text{err}(h) \leq \epsilon$, and *bad* if $\text{err}(h) > \epsilon$. It suffices to show that with probability at least $1 - \delta$, all bad hypotheses are inconsistent with the training data.

Fix a bad hypothesis h . For each sample $(x_i, y_i) \sim \mathcal{D}$, we have $\mathbf{P}[h(x_i) = y_i] < 1 - \epsilon$. The probability that h agrees on all m independent samples $(x_1, y_1), \dots, (x_m, y_m)$ is

$$(1 - \epsilon)^m \leq e^{-\epsilon m} \leq \frac{\delta}{|\mathcal{H}|}.$$

By the union bound, the probability that some bad hypothesis h is consistent is at most

$$\sum_{h \text{ bad}} \mathbf{P}[h \text{ consistent}] \leq |\mathcal{H}| \cdot \frac{\delta}{|\mathcal{H}|} \leq \delta,$$

as desired. □

Uniform convergence. For a training set $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$ and hypothesis h , let

$$\text{err}(h | S) = \frac{1}{m} |\{i : h(x_i) \neq y_i\}|$$

denote the *empirical or training error rate* of h with respect to S . Most learning algorithms try to select $h \in \mathcal{H}$ minimizing the training error $\text{err}(h | S)$, in the hopes that it would minimize $\text{err}(h)$ as well. Here we run two kinds from risk when the empirical error $\text{err}(h | S)$ deviates from the true error $\text{err}(h)$.

1. A very good hypothesis h with low $\text{err}(h)$ may have $\text{err}(h | S)$ much larger than $\text{err}(h)$, dissuading the algorithm from selecting h .
2. A very bad hypothesis h with high $\text{err}(h)$ may have $\text{err}(h | S)$ much smaller than $\text{err}(h)$, tempting the algorithm from selecting h .

Thus, for minimizing $\text{err}(h | S)$ to be a good strategy for minimizing $\text{err}(h)$, we need $\text{err}(h | S)$ to be close to $\text{err}(h)$ for all the hypotheses h . This is given by the following guarantee.

Theorem 13.2. *Let $\epsilon, \delta \in (0, 1)$ and consider a training set S of size m . If $m \geq \ln(2|\mathcal{H}|/\delta)/2\epsilon^2$, then with probability at least $1 - \delta$,*

$$|\text{err}(h) - \text{err}(h | S)| \leq \epsilon$$

for all $h \in \mathcal{H}$.

Proof sketch. We apply the additive Chernoff bound to each $h \in \mathcal{H}$, and take the union bound. $\square$

Example: disjunctions Recall the concept class $\mathcal{C}$ of disjunctions over $X = \{0, 1\}^n$. Let $\mathcal{H} = \mathcal{C}$, and recall that we have a consistent learning algorithm for $\mathcal{C}$. We have $|\mathcal{H}| = |\mathcal{C}| = 3^n$ since each disjunction f is defined by selecting, for each $i \in [n]$, whether x_i , $\bar{x}_i$, or neither appears in f . By ??, for $m = O((n + \log(1/\delta))/\epsilon^2)$, the algorithm produces a hypotheses h with $\text{err}(h) \leq \epsilon$ with probability at least $1 - \delta$. That is, for fixed n , the class of disjunctions is PAC-learnable.

13.4 Occam's razor

Occam's razor refers the general principle that one should prefer simpler explanations. It has been echoed by many thinkers in history; for example:

- Aristotle: "Nature operates in the shortest way possible."
- William of Occam: "Plurality is never to be posited without necessity."
- Isaac Newton: "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances. Therefore, to the same natural effects we must, as far as possible, assign the same causes."
- Bertrand Russell: "Whenever possible, substitute constructions out of known entities for inferences to unknown entities."

- Albert Einstein: “Everything should be made as simple as possible, but not simpler.”
- Richard Feynman: “The truth always turns out to be simpler than you thought.”
- The Big Aristotle: “If you don't stick to simplicity, you'll die a horrible death.”

For learning algorithms, one way to measure the complexity of a hypothesis class $\mathcal{H}$ is via its *descriptive bit complexity*. Fix some system by which hypotheses $h \in \mathcal{H}$ are encoded in bit strings. For a hypothesis $h \in \mathcal{H}$, let $|h|$ denote the number of bits needed to describe h . The *descriptive complexity* of $\mathcal{H}$ is defined as the maximum number of bits, $|h|$, need to describe any hypothesis $h \in \mathcal{H}$. The following gives generalization bounds in terms of the bit complexity of $\mathcal{H}$.

Corollary 13.3. *Let $\mathcal{H}$ be a family of hypotheses of descriptive complexity b . Let $\epsilon, \delta \in (0, 1)$ and consider a training set S of size m .*

- (i) *If $m \geq (b \ln(2) + \ln(1/\delta))/\epsilon$, then with probability at least $1 - \delta$, all consistent hypotheses $h \in \mathcal{H}$ have $\text{err}(h) \leq \epsilon$.*
- (ii) *If $m \geq (b \ln(2) + \ln(2/\delta))/2\epsilon^2$, then with probability at least $1 - \delta$, all hypotheses $h \in \mathcal{H}$ have $|\text{err}(h | S) - \text{err}(h)| \leq \epsilon$.*

Proof. We have $|\mathcal{H}| \leq 2^b$ because b bits can describe at most 2^b hypothesis. □

Regularization by bit complexity. Corollary 13.3 says that hypothesis classes of low bit complexity have better generalization error. Another way to express this theme is to say that every hypothesis $h \in \mathcal{H}$ has generalization error proportional to its bit complexity, as follows.

Corollary 13.4. *Let $\epsilon, \delta \in (0, 1)$ and consider a training set of size m drawn from $\mathcal{D}$. With probability at least $1 - \delta$, we have*

$$|\text{err}(h) - \text{err}(h | S)| \leq O\left(\sqrt{|h| + \ln(1/\delta)/m}\right)$$

for all $h \in \mathcal{H}$.

Proof. For $i \in \mathbb{N}$, let $\mathcal{H}_i$ be the subset of hypothesis of bit complexity between 2^{i-1} and 2^i :

$$\mathcal{H}_i \stackrel{\text{def}}{=} \{h \in \mathcal{H} : 2^{i-1} < |h| \leq 2^i\}.$$

Let $\delta_i = \delta/2^i$. By corollary 13.3, for each $\mathcal{H}_i$, we have

$$\begin{aligned} |\text{err}(h) - \text{err}(h | S)| &\leq O\left(\sqrt{\ln(|\mathcal{H}_i|) + \ln(\delta_i)/m}\right) \\ &= O\left(\sqrt{\ln(2^i) + \ln(2^i/\delta)/m}\right) \\ &= O\left(\sqrt{(|h| + \ln(1/\delta))/m}\right) \end{aligned}$$

for all $h \in \mathcal{H}_i$ with probability of error at most δ_i . Taking the union bound over all i , the probability of error over all of $\mathcal{H} = \bigcup_i \mathcal{H}_i$ is $\sum_i \delta_i = \delta \sum_i 1/2^i = \delta$. $\square$

13.5 Stronger generalization bounds via growth rate

Many learning algorithms select from a hypothesis class $\mathcal{H}$ that is geometrically fairly simple. A recurring example has been a simple consistent learner for the concept class of rectangles, which of course generalizes to higher dimensions. The *perceptron algorithm* looks for a hyperplane that the data. This algorithm is particularly useful when combined with techniques such as the *kernel trick* that first embed the data into a higher-dimensional space where the labeled data becomes linearly separable.

Recall the previous chapter that the measure of simple geometric objects is particularly well-concentrated under random sampling. Applying those techniques – namely, ϵ -nets and ϵ -samples – to learning gives the following generalization bounds.

To extend the notion of a growth function to a hypotheses class $\mathcal{H}$, we interpret $\mathcal{H}$ as a family of ranges corresponding to the positively labeled sets,

$$\{h^{-1}(1) : h \in \mathcal{H}\}.$$

Recall that the *growth function* $g(n)$ of $\mathcal{H}$ is the maximum number of distinct subsets that can be induced by $\mathcal{H}$ over a set of n points.

Theorem 13.5. *Let $\mathcal{H}$ be a family of hypotheses with growth function $g(n)$. Let $\epsilon, \delta \in (0, 1)$. Consider a data set of size m .*

- (i) *If $m \geq O(\ln(g(2m)/\delta)/\epsilon)$, then with probability at least $1 - \delta$, all consistent hypotheses $h \in \mathcal{H}$ have $\text{err}(h) \leq \epsilon$.*
- (ii) *If $m \geq O(\ln(g(2m)/\delta)/\epsilon^2)$, then all $h \in \mathcal{H}$ have $|\text{err}(h | S) - \text{err}(h)| \leq \epsilon$.*

Proof. For each hypothesis h , let $\bar{h}(x) = 1 - h(x)$ denote the complementary hypothesis. Increasing $|\mathcal{H}|$ and the growth function $g(n)$ by at most a factor of 2 (why?), we assume that that $\mathcal{H}$ contains the complement $\bar{h}$ of each hypothesis $h \in \mathcal{H}$.

Consider the (possibly infinite) point set

$$P = X \times \{0, 1\} = \{(x, y) : x \in X, y \in \{0, 1\}\}$$

We associate each hypothesis $h \in \mathcal{H}$ with the range

$$r_h = \{(x, 1 - h(x))\} \subseteq P.$$

The distribution $\mathcal{D}$ induces a measure $\mu_{\mathcal{D}}$,

$$\mu_{\mathcal{D}}(r_h) = \mathbf{P}_{(x,y) \sim \mathcal{D}}[(x, y) \in r_h]$$

The measure of a range r_h equals the true error of h :

$$\mu_{\mathcal{D}}(r_h) = \mathbf{P}_{(x,y) \sim \mathcal{D}}[(x, y) \in r_h] = \mathbf{P}_{(x,y) \sim \mathcal{D}}[h(x) \neq y] = \text{err}(h).$$

Let $R_{\mathcal{H}} = \{r_h : h \in \mathcal{H}\}$. Let $f(m)$ denote the growth rate of $R_{\mathcal{H}}$. The key claim is that

$$f(m) \leq g^2(n). \quad (13.2)$$

Assuming (13.2) is true, the theorem follows from the ϵ -net and ϵ -sample theorems from chapter 12, as we now explain.

For item (i), call a hypotheses bad if $\text{err}(h) > \epsilon$; we want to show that for a sample of the claimed size m , with probability at least $1 - \delta$, every bad hypotheses is inconsistent with the data. Here we observe that since $\text{err}(h) = \mu_{\mathcal{D}}(r_h)$, this is equivalent to saying that the sample set is an ϵ -net. The claim now follows from theorem 12.3.

For item (ii), observe that $\text{err}(h) = \mu_{\mathcal{D}}(r_h)$, and for a training set S , $\text{err}(h|S) = \mu_S(r_h)$. Thus item (ii) is equivalent to saying that S is an ϵ -sample with probability at least $1 - \delta$. The sufficient bounds for m now follow from theorem 12.2.

It remains to prove (13.2). Fix a S set of m labeled points $\{(x_1, y_1), \dots, (x_m, y_m)\}$. Let $S'_0 = \{x_i : (x_i, 1) \in S\}$ denote the set of input points labeled 0 and let $S_0 = S'_0 \times \{0\}$ denote the corresponding labeled pairs. Likewise let $S'_1 = \{x_i : (x_i, 0) \in S\}$ and $S_1 = S'_1 \times \{1\}$ be the corresponding sets for label 1.

The total number of intersections induced by $R_{\mathcal{H}}$ on S , $|R_{\mathcal{H}} \wedge S|$, is bounded above by

$$|R_{\mathcal{H}} \wedge S| \leq |R_{\mathcal{H}} \wedge S_0| \cdot |R_{\mathcal{H}} \wedge S_1|,$$

since any two intersections in $|R_{\mathcal{H}} \wedge S|$ must differ on either S_0 or S_1 . We claim that both of the terms on the RHS are bounded above by $g(m)$, which proves the claim.

Consider $|R_{\mathcal{H}} \wedge S_0|$. For any $h \in \mathcal{H}$, we have

$$r_h \wedge S_0 = \{(x, 0) \in S : h(x) = 1\} = (h^{-1}(1) \wedge S'_0) \times \{0\}.$$

That is,

$$|R_{\mathcal{H}} \wedge S_0| = |\mathcal{H} \wedge S'_0| \leq g(m).$$

Similarly, for $|R_{\mathcal{H}} \wedge S_1|$, for each hypothesis h we have

$$r_h \wedge S_1 = \{(x_i, 1) \in S : h(x_i) = 0\} = (\bar{h}^{-1}(1) \wedge S'_1) \times \{1\}.$$

Consequently

$$|R_{\mathcal{H}} \wedge S_1| = |\mathcal{H} \wedge S'_1| \leq g(m).$$

This completes the proof of the claim, hence the proof of the theorem. $\square$

Example: rectangles. Consider the concept class $\mathcal{C}$ of rectangles in $\mathbb{R}^2$. We have a consistent learning algorithm by outputting any rectangle that contains only the positively labeled points of the training set.

Rectangles have constant VC-dimension, hence polynomial growth $g(n) = n^{O(1)}$ (cf. Sauer's lemma, theorem 12.5). So we need $O(\ln(1/\delta)/\epsilon)$ to guarantee that with probability at least $1 - \delta$, the rectangle we output has true error at most ϵ . That is, rectangles are PAC-learnable.

More generally, any concept class with fixed VC-dimension and a consistent learning algorithm is PAC-learnable.

13.6 Additional notes and materials

These notes are based on the first 10 lectures of [Sch19] and sections 5.1–5.7 of [BHK20]. We refer the interested reader to [Sch19] for additional background on these and other topics in theoretical machine learning.

13.7 Exercises