CS 760: Machine Learning
PAC Learning

University of Wisconsin-Madison

Outline

- **PAC Learning**
  - definition, sample complexity bounds
- **VC-dimension**
  - motivation, definition, other complexity measures

Outline

- **PAC Learning**
  - definition, sample complexity bounds

- **VC-dimension**
  - motivation, definition, other complexity measures

Recall: **Generalization error decomposition**

Last time, we were interested in the quantity

$$err(\hat{h}) = \widehat{err}(\hat{h}) + [err(\hat{h}) - \widehat{err}(\hat{h})]$$

$err(\hat{h})$: true risk (test error)

$\widehat{err}(\hat{h})$: empirical risk (training error)

$[err(\hat{h}) - \widehat{err}(\hat{h})]$: generalization gap

If the generalization gap is small and our learning algorithm is empirical risk minimization then the true risk is also small

Today: a framework for bounding the generalization error and gap in terms of the number of samples

PAC Learning: Setup

- input space X (images, documents, feature vectors in $\mathbb{R}^2$, …)

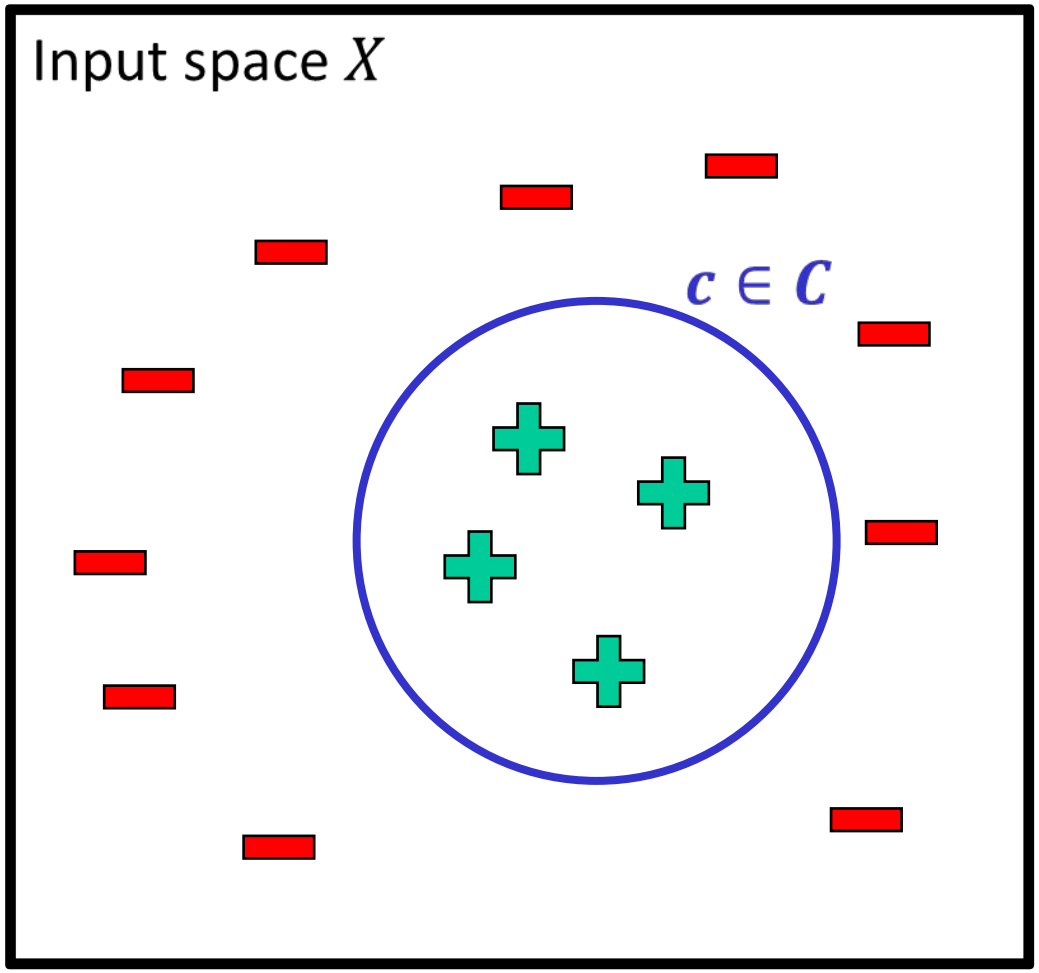
- output space $Y = \{0,1\}$
  - just binary classification in this class
  - extendable to regression / multi-class classification

- hypothesis class H
  - set of functions $f: X \mapsto Y$ that the learning algorithm can output

PAC Learning: Data

- concept class C
  - set of labeling functions $c: X \mapsto Y$
  - represents set of possible true labeling functions
  - example: circles in $\mathbb{R}^2$
  - NOT (always) the same thing as the hypothesis class H
  - ideally $H \supset C$

- data distribution D over $X \times Y$
  - unknown
  - get m samples (x_i, y_i), $i = 1, \ldots, m$ i.i.d. from D
  - $y_i = c(x_i)$ for some $c \in C$

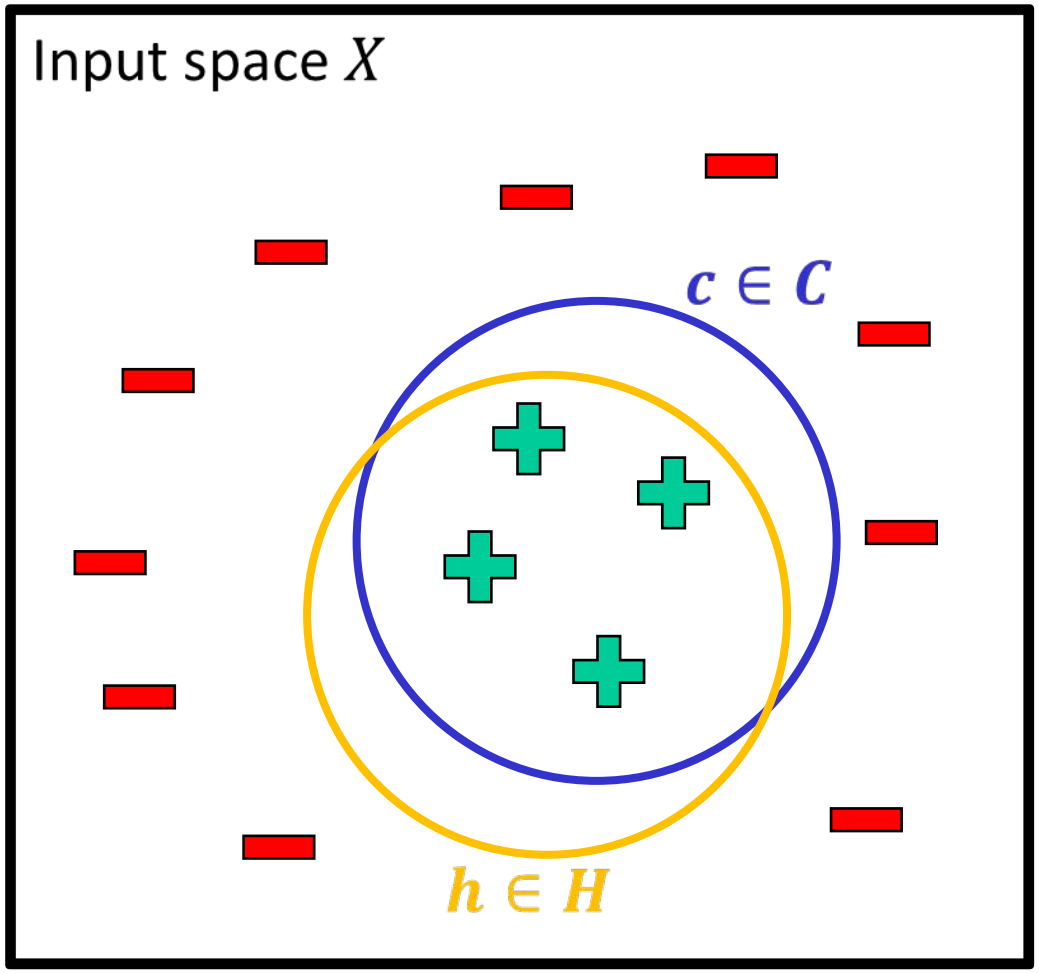


PAC Learning: Goal

Use the training data to **approximate** the unknown target concept $c \in C$ using a hypothesis $h \in H$

Evaluate using **true risk**

$$err(h) = P_{x\sim D}\{h(x) \neq c(x)\} = \mathbb{E}1_{\{h(x)\neq c(x)\}}$$



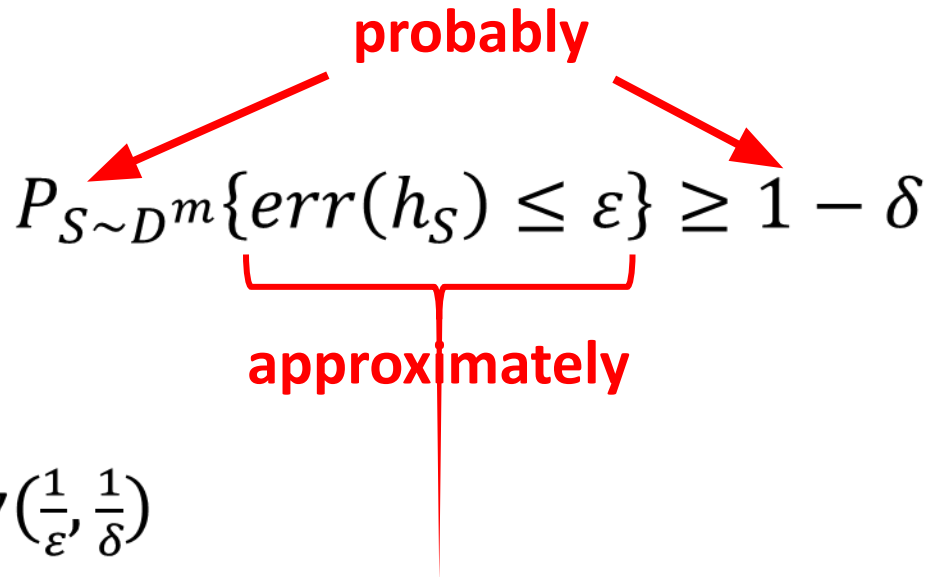
But we only have access to the **empirical risk** $\widehat{err}(h) = \frac{1}{m}\sum_{i=1}^{m} 1_{\{h(x)\neq c(x)\}}$

Thus any approximation must be **probabilistic**

PAC Learning: Definition

PAC = Probably Approximately Correct

A concept class C is **PAC-learnable** if there exists a learning algorithm that takes any dataset S of size m and outputs a hypothesis $h_S: X \mapsto Y$ that for any $\varepsilon, \delta > 0$ has the property



so long as $m \geq poly(\frac{1}{\varepsilon}, \frac{1}{\delta})$

PAC Learning: Definition

In words: C is PAC-learnable **if**

- there is a learning algorithm **and**
- a polynomial function $poly$: $(0,1)^2 \mapsto \mathbb{Z}_+$ **such that**
- the probability the algorithm outputs a model that is wrong on more than ε-fraction of samples from D is at most δ, **so long as**
- the number of samples m is $\geq poly(1/\varepsilon, 1/\delta)$

The polynomial function is called the **sample complexity**

Compare to **computational complexity**: a problem is **tractable** (i.e. in P) if there exists a polynomial time algorithm to solve it

Consistent case guarantee

Suppose there exists a **consistent** learning algorithm:

- always returns h_S with empirical risk $\widehat{err}(h_S) = 0$
- for example: empirical risk minimization on $H \supset C$

Then C is PAC-learnable with sample complexity $\frac{1}{\varepsilon}\log\frac{|H|}{\delta}$

In other words, given m samples, with probability $\geq 1 - \delta$,

$$err(h_S) = err(h_S) - \widehat{err}(h_S) \leq \frac{1}{m}\log\frac{|H|}{\delta}$$

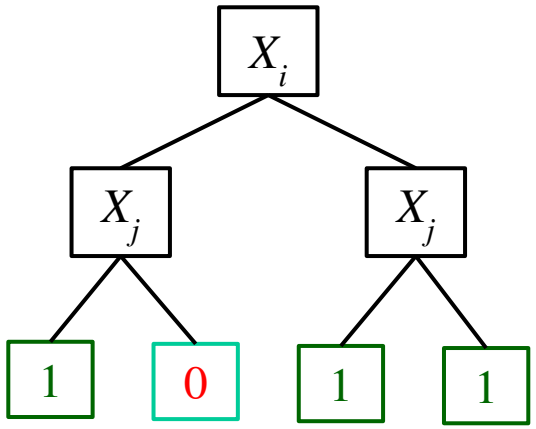
Consistent case: Example

Suppose we have n-dimensional binary data that is perfectly classified by a depth 2 decision tree:

- $|H| = \#\ of\ pairs \times \#\ of\ leaf\ labelings = \binom{n}{2} \times 2^4 = 16\binom{n}{2}$
- If we want success probability ≥ 0.99 and error rate ≤ 0.05 then

$$m \geq \frac{1}{0.05}\left(\log \frac{16n(n-1)}{2 \times 0.01}\right)$$

- If $n = 100$ then we $m \geq 318$ samples suffices.



Inconsistent case

Want to drop the consistency requirement $err(h_S) = 0$

Inconsistent case guarantee:

- w.p. $\geq 1 - \delta$, $err(h) \leq \widehat{err}(h) + \sqrt{\frac{1}{2m} \log \frac{2|H|}{\delta}}$ $\forall\, h \in H$

- quadratically worse error than consistent case

- uniform convergence bounds holds **simultaneously for all h**

PAC learning so far…

Formalizes learning task while allowing for imperfect learning due to randomness / approximation (parameterized via δ and ε)

Can also define **efficient** PAC learnability by requiring the learning algorithm to take polynomial-time compute

Shows how many samples are needed to learn over a finite hypothesis class

What if the hypothesis space is infinite? Can we still learn?

- linear models
- neural networks
- …

$$err(h) \leq \widehat{err}(h) + \sqrt{\frac{1}{2m}\log\frac{2|H|}{\delta}}$$

Outline

- **PAC Learning**
  - definition, sample complexity bounds

- **VC-dimension**
  - motivation, definition, other complexity measures

Infinite hypothesis classes

Most practical learning algorithms operate over infinite hypothesis classes

Basic PAC results give infinite sample complexity for $|H| = \infty$

Need a different way to quantify the capacity of the class

The **Vapnik-Chervonenkis (VC) dimension** does this by measuring how easy it is for function in H to fit arbitrary labels

Getting started: **Shattering**

Hypothesis space H **shatters** a set of points $S = \{x_1, \dots, x_k\} \in X$ if for every possible labeling $\{y_1, \dots, y_k\} \in \{0,1\}$ of S there exists a function $h \in H$ such that produces that labeling, i.e.

$$h(x_1) = y_1, \dots, h(x_k) = y_k$$

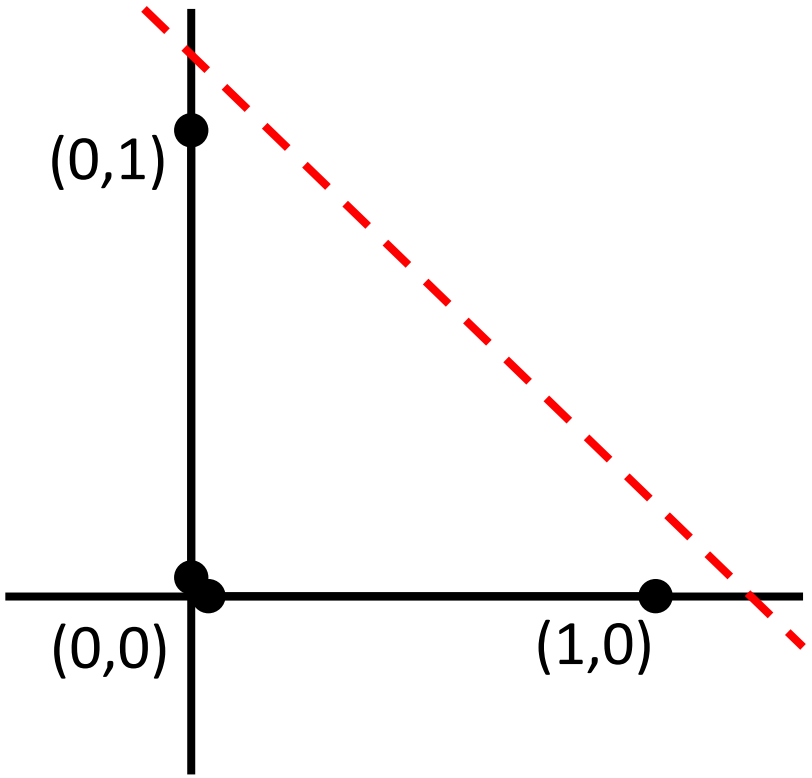
Demonstrates that H is expressive enough to make arbitrary distinctions between these a set of k points.

Shattering example: Lines in 2D

Hypotheses: $H = \{\text{sign}(w_1 x_1 + w_2 x_2 + b) : w_1, w_2, b \in \mathbb{R}\}$
Set of points: $S = \{(0,0), (1,0), (0,1)\}$

2^3 possible labelings:

- (0,0,0)
- (0,0,1)
- (0,1,0)
- (0,1,1)
- (1,0,0)
- (1,0,1)
- (1,1,0)
- (1,1,1)



VC dimension

The **VC dimension of a hypothesis class H** is the size of the largest set of points in X that can be shattered by H

Two step procedure to show $VC(H) = d$:

1. find a set of points $S \subset X$ of size $|S| = d$ that is shattered by H
   - i.e. find d points that can be labeled arbitrarily by functions $h \in H$
   - easier step: only need to find one set of shattered points, NOT all

2. show that no set of $d + 1$ points can be shattered

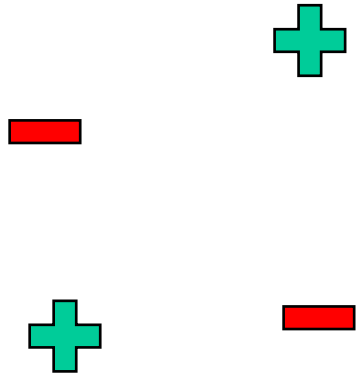
VC dimension example: Lines in 2D

Already demonstrated a set of three points that is shattered by H, so $VC(H) \geq 3$

Is there a set of four points shattered by H?

Case 1: collinear points

Case 2: one point inside convex hull



Case 3: quadrilateral

No! Thus $VC(H) < 4$, and so $VC(H) = 3$

What does the VC-dimension get us?

If H has VC dimension d and we draw m samples i.i.d. from D then with probability at least $1 - \delta$ the following is true for all $h \in H$:

$$err(h) \leq \widehat{err}(h) + \sqrt{\frac{2d}{m}\log\frac{em}{d}} + \sqrt{\frac{1}{m}\log\frac{1}{\delta}}$$

VC-dimension roughly takes the place of $\log|H|$ as a measure of the capacity of the hypothesis class.

VC dimensions of other classes

Linear classifiers in $\mathbb{R}^d$: $d + 1$

Finite hypothesis spaces: $\leq \log |H|$

L-layer ReLU networks with W weights: $O(WL \log W)$

Implications of VC-dimension

Bound suggests roughly $m \approx d = VC(H)$ examples suffice to start getting meaningful generalization

$$err(h) \leq \widehat{err}(h) + \sqrt{\frac{2d}{m}\log\frac{em}{d}} + \sqrt{\frac{1}{m}\log\frac{1}{\delta}}$$

- sometimes okay for linear models over sufficiently large data
- vacuous for modern deep nets (CNNs with millions of weights do well on CIFAR-10, a dataset with <100K examples)

Other measures of hypothesis class capacity

Rademacher complexity: how easily do functions $h \in H$ fit **random** labels?

Covering number: how many functions $h \in H$ are needed so that **the rest are close** to one of them?

Achievements of learning theory

PAC theory provides a way of thinking about the relationship between learning performance, sample size, and model capacity

- works for any data distribution
- capacity measures can be computed for most classes of interest
- many extensions to noisy labels, hypothesis classes that do not contain the target concept, etc.
- bounds generalization error in interpretable ways

Caveats

Drawbacks of most PAC guarantees:

- do not adapt to easy data (worst-case)
- does not consider the effect of optimization
- capacity measures might be misleading for modern deep nets

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Misha Khodak, Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Fred Sala, Josiah Hanna