Introduction to Learning Theory

CS 760@UW-Madison

Goals for the lecture

you should understand the following concepts

- error decomposition

- bias-variance tradeoff

- PAC learnability
- consistent learners and version spaces
- sample complexity

Error Decomposition

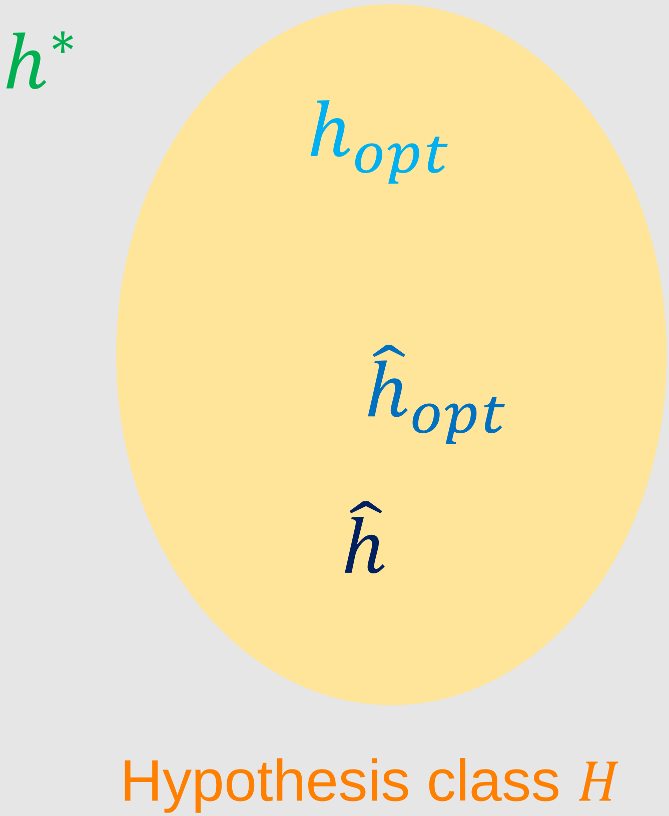
How to analyze the generalization?

- Key quantity we care in machine learning: the error on the future data points (i.e., the expected error on the whole distribution)
- Divide the analysis of the expected error into steps:
  - What if full information (i.e., infinite data) and full computational power (i.e., can do optimization optimally)?
  - What if finite data but full computational power?
  - What if finite data and finite computational power?

- Example: error decomposition for prediction in supervised learning

Bottou, Léon, and Olivier Bousquet. "The tradeoffs of large scale learning." *Advances in neural information processing systems*. 2008.

Error/risk decomposition



h^*

h_{opt}

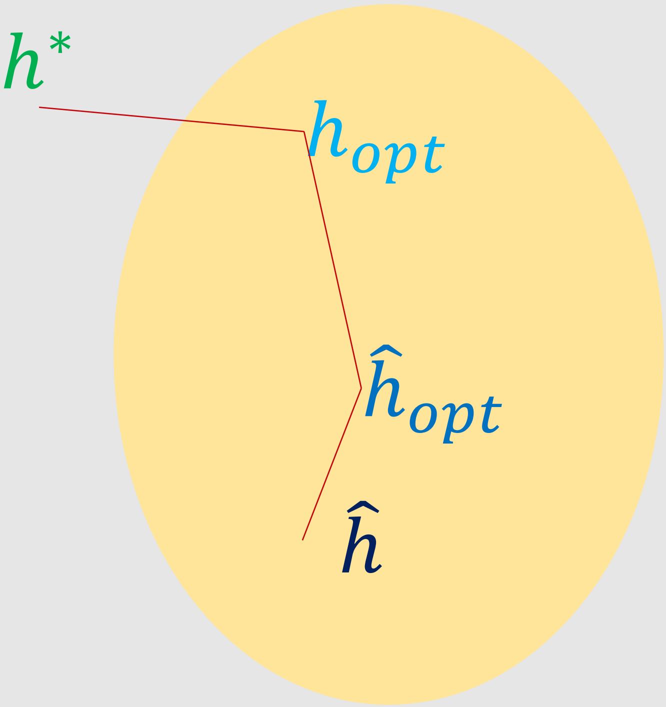
$\hat{h}_{opt}$

$\hat{h}$

Hypothesis class H

- h^*: the optimal function (Bayes classifier)
- h_{opt}: the optimal hypothesis on the data distribution
- $\hat{h}_{opt}$: the optimal hypothesis on the training data
- $\hat{h}$: the hypothesis found by the learning algorithm

Error/risk decomposition



Hypothesis class H

$$err(\hat{h}) - err(h^*)$$

$$= err(h_{opt}) - err(h^*)$$

$$+ err(\hat{h}_{opt}) - err(h_{opt})$$

$$+ err(\hat{h}) - err(\hat{h}_{opt})$$

Error/risk decomposition

Approximation error

Estimation error

Optimization error

$$
\begin{aligned}
& err(\hat{h}) - err(h^*) \\
&= err(h_{opt}) - err(h^*) \\
&+ err(\hat{h}_{opt}) - err(h_{opt}) \\
&+ err(\hat{h}) - err(\hat{h}_{opt})
\end{aligned}
$$

“the fundamental theorem of machine learning”

Error/risk decomposition

- approximation error: due to problem modeling (the choice of hypothesis class)
- estimation error: due to finite data
- optimization error: due to imperfect optimization

$$err(\hat{h}) - err(h^*)$$

$$= err(h_{opt}) - err(h^*)$$

$$+ err(\hat{h}_{opt}) - err(h_{opt})$$

$$+ err(\hat{h}) - err(\hat{h}_{opt})$$

More on estimation error

$$err(\hat{h}_{opt}) - err(h_{opt})$$

$$= err(\hat{h}_{opt}) - \widehat{err}\,(\hat{h}_{opt})$$

$$+ \widehat{err}\,(\hat{h}_{opt}) - err(h_{opt})$$

$$\leq err(\hat{h}_{opt}) - \widehat{err}\,(\hat{h}_{opt})$$

$$+ \widehat{err}\,(h_{opt}) - err(h_{opt})$$

$$\leq 2 \sup_{h \in H} |err(h) - \widehat{err}(h)|$$

Another (simpler) decomposition

$$err(\hat{h}) = \widehat{err}(\hat{h}) + \underbrace{\left[err(\hat{h}) - \widehat{err}(\hat{h})\right]}_{\text{Generalization gap}}$$

$$\leq \widehat{err}(\hat{h}) + \sup_{h \in H} |err(h) - \widehat{err}(h)|$$

- The training error $\widehat{err}(\hat{h})$ is what we can compute
- Need to control the generalization gap

Bias-Variance Tradeoff

Defining bias and variance

- consider the task of learning a regression model $f(\boldsymbol{x};\, D)$ given a training set $D = \left\{(x^{(1)}, y^{(1)}),\ldots,(x^{(m)}, y^{(m)})\right\}$

indicates the dependency of model on D

- a natural measure of the error of f is

$$E\left[\left(y - f(\mathbf{x}; D)\right)^2 \mid D\right]$$

where the expectation is taken with respect to the real-world distribution of instances

Defining bias and variance

- this can be rewritten as:

$$E\left[\left(y - f(\boldsymbol{x}; D)\right)^2 \mid \boldsymbol{x}, D\right] = E\left[\left(y - E[y \mid \boldsymbol{x}]\right)^2 \mid \boldsymbol{x}, D\right]$$
$$+ \left(f(\boldsymbol{x}; D) - E[y \mid \boldsymbol{x}]\right)^2$$

error of f as a predictor of y

noise: variance of y given $\boldsymbol{x}$; doesn’t depend on D or f

Defining bias and variance

- now consider the expectation (over different data sets D) for the second term

$$E_D\left[\left(f(\boldsymbol{x};D) - E[y\,|\,\boldsymbol{x}]\right)^2\right] =$$

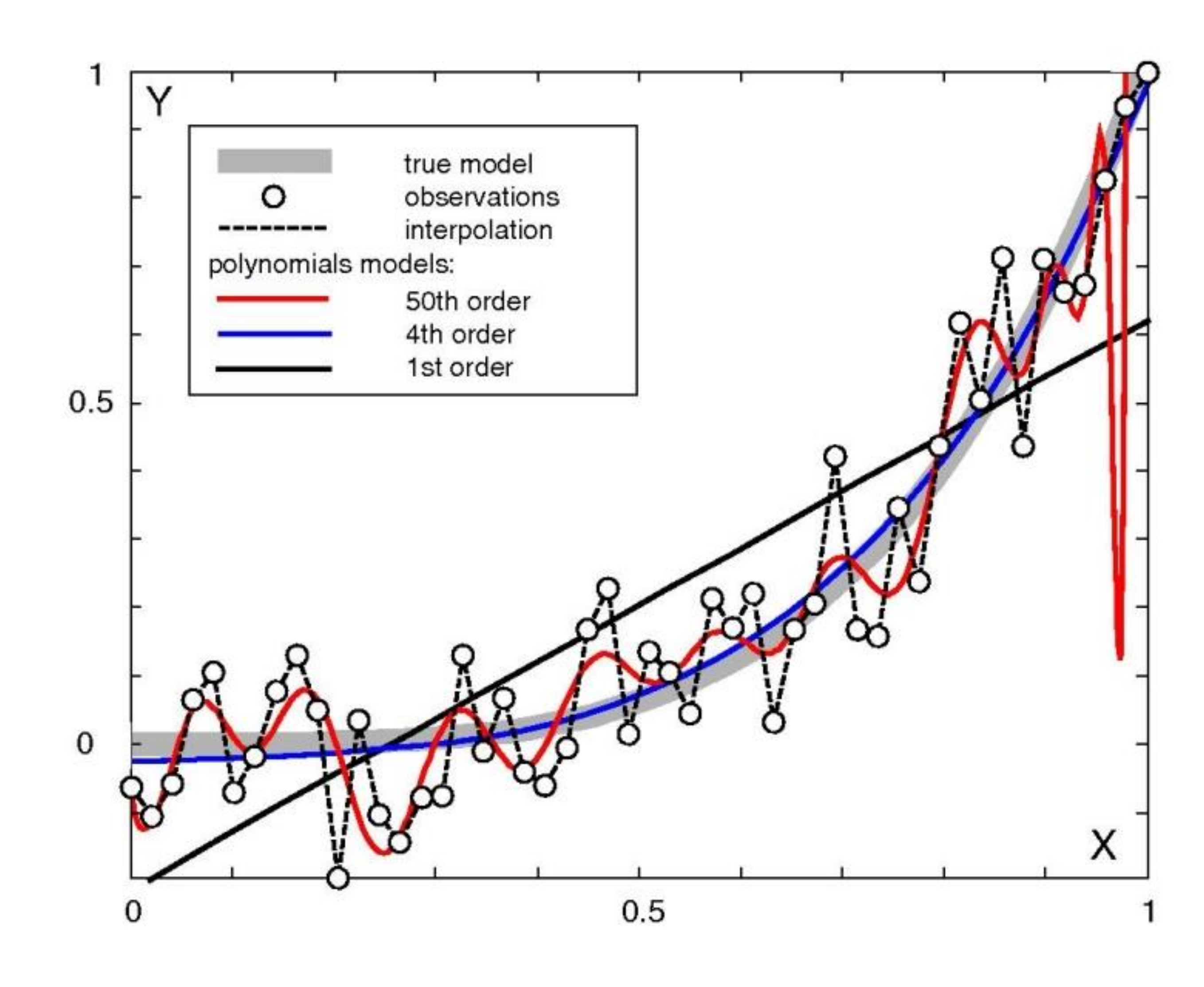
$$\left(E_D\left[f(\boldsymbol{x};D)\right] - E\left[y\,|\,\boldsymbol{x}\right]\right)^2 \qquad \text{bias}$$

$$+\,E_D\left[\left(f(\boldsymbol{x};D) - E_D\left[f(\boldsymbol{x};D)\right]\right)^2\right] \qquad \text{variance}$$

- bias: if on average $f(\boldsymbol{x};D)$ differs from $E\,[y\,|\,\boldsymbol{x}]$ then $f(\boldsymbol{x};D)$ is a biased estimator of $E\,[y\,|\,\boldsymbol{x}]$
- variance: $f(\boldsymbol{x};D)$ may be sensitive to D and vary a lot from its expected value

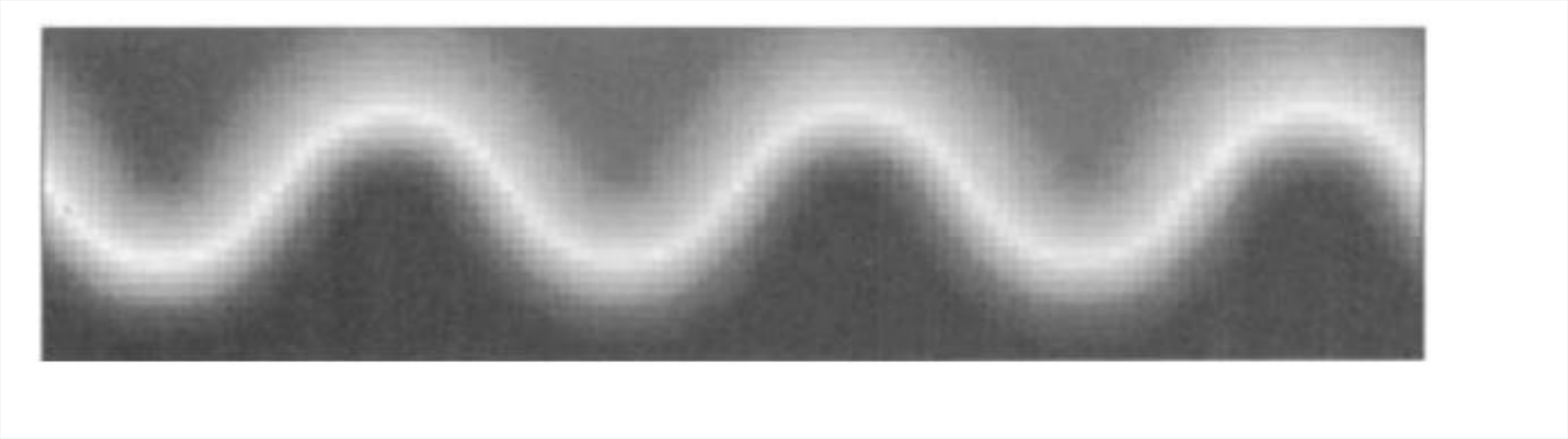
Bias/variance for polynomial interpolation

- the 1st order polynomial has high bias, low variance
- 50th order polynomial has low bias, high variance
- 4th order polynomial represents a good trade-off

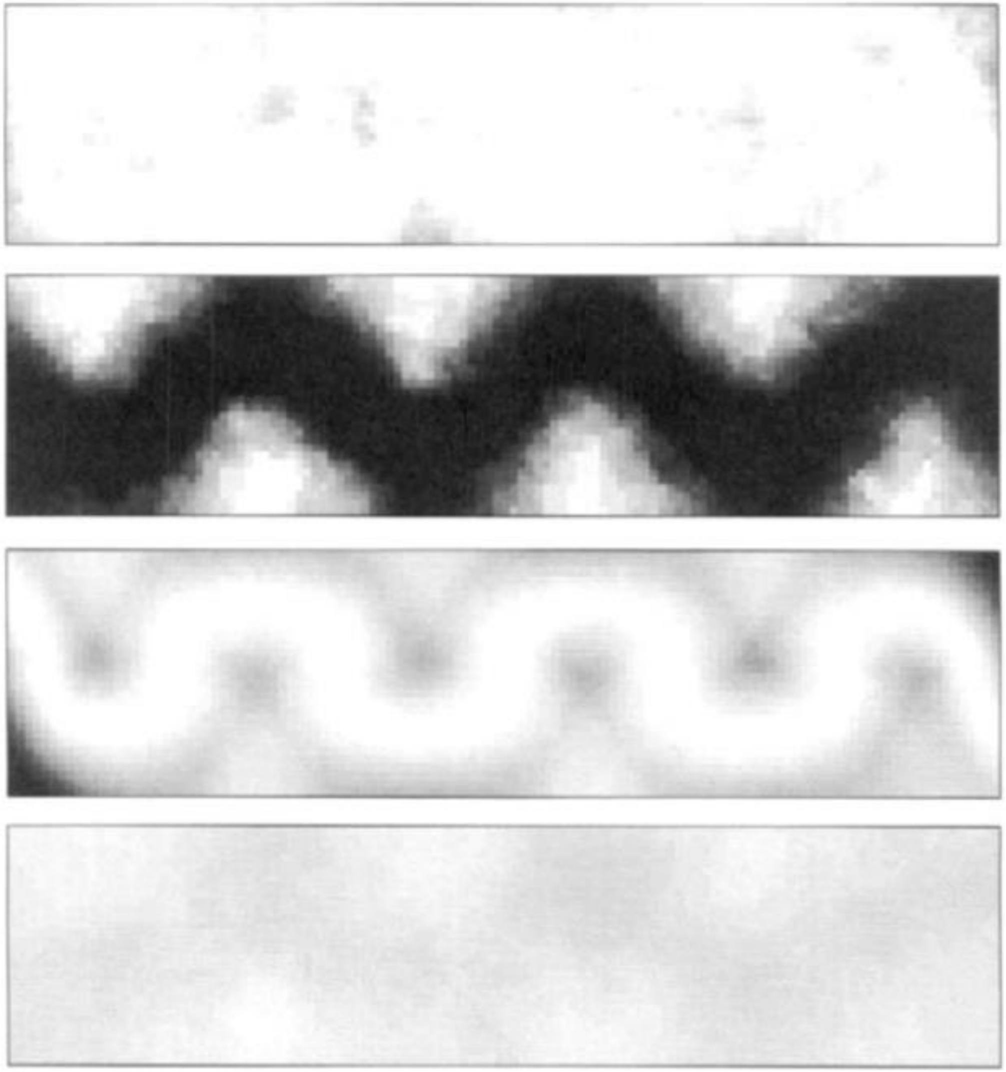


Bias/variance trade-off for k-NN regression

- consider using k-NN regression to learn a model of this surface in a 2-dimensional feature space



Bias/variance trade-off for k-NN regression



bias for 1-NN

variance for 1-NN

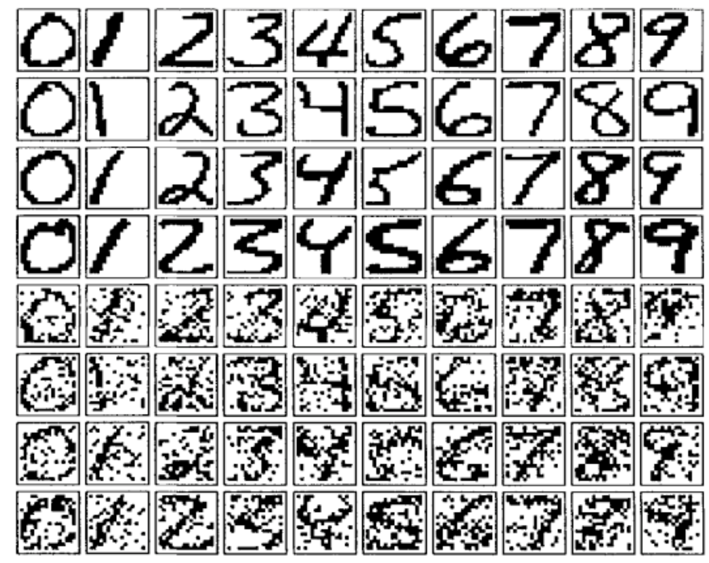
bias for 10-NN

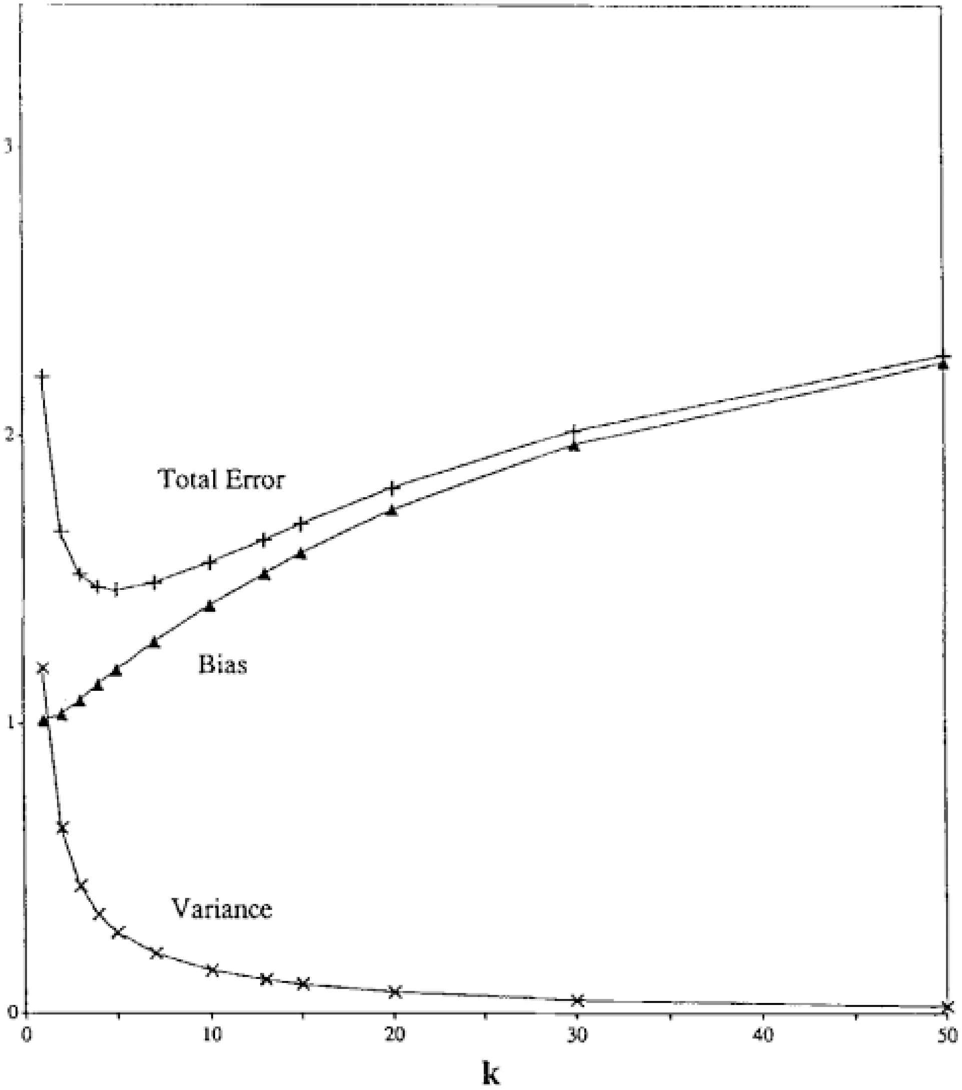
variance for 10-NN

darker pixels
correspond to
higher values

Bias/variance trade-off

- consider k-NN applied to digit recognition





Bias/variance discussion

- predictive error has two controllable components
  - expressive/flexible learners reduce *bias*, but increase *variance*

- for many learners we can trade-off these two components (e.g. via our selection of k in k-NN)

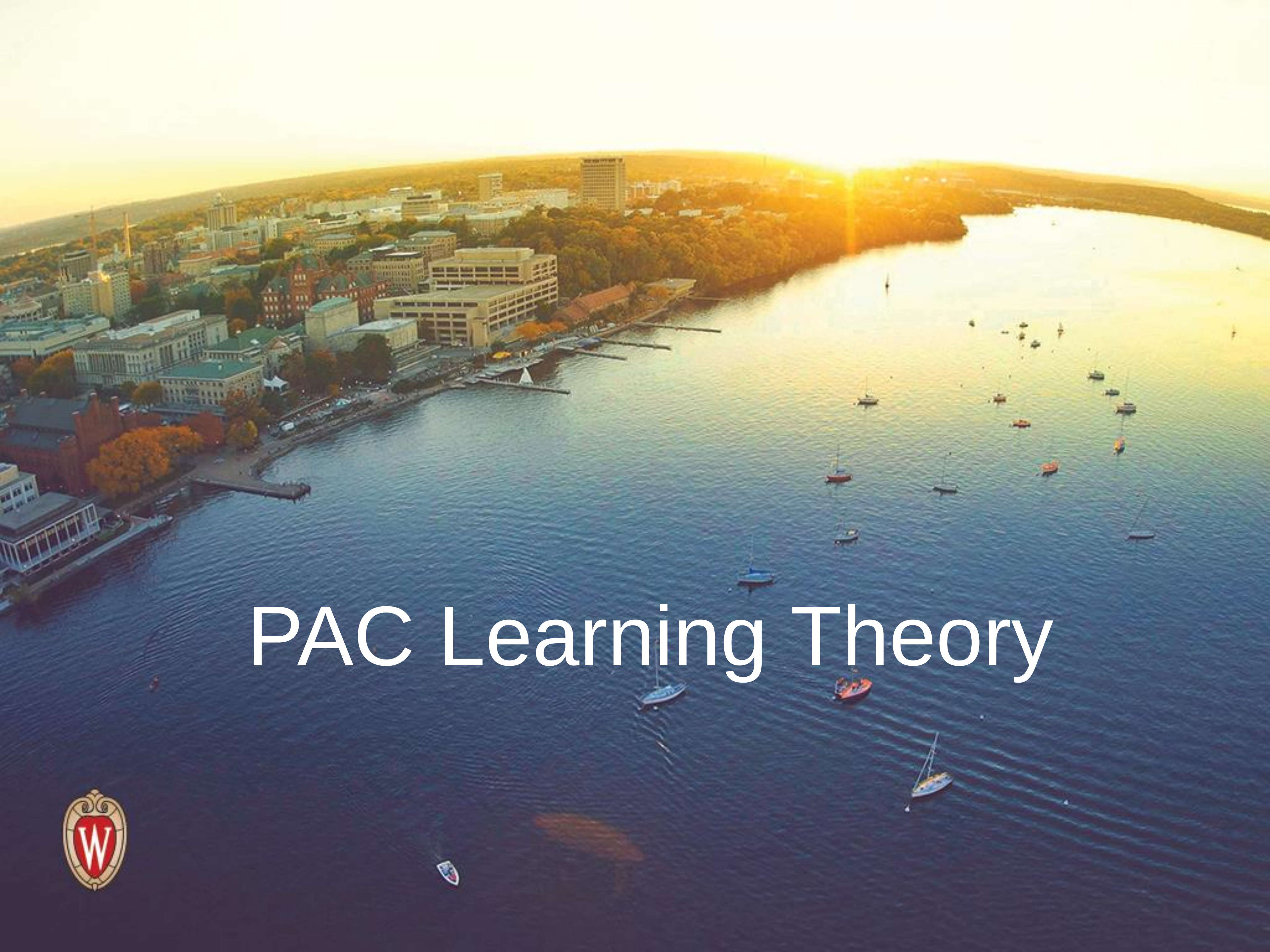
- the optimal point in this trade-off depends on the particular problem domain and training set size

- this is not necessarily a strict trade-off; e.g. with ensembles we can often reduce bias and/or variance without increasing the other term

Bias/variance discussion

the bias/variance analysis

- helps explain why simple learners can outperform more complex ones
- helps understand and avoid overfitting



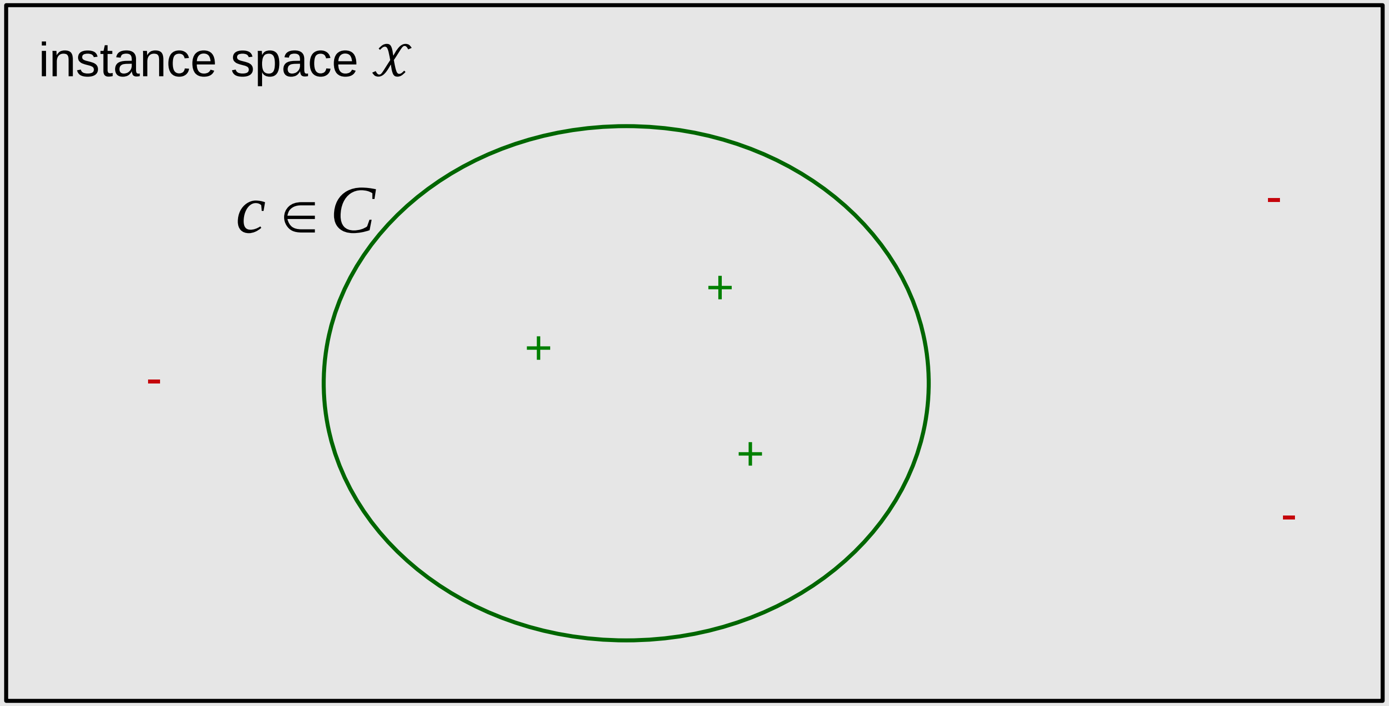
PAC Learning Theory

PAC learning

- Overfitting happens because training error is a poor estimate of generalization error
  → Can we infer something about generalization error from training error?

- Overfitting happens when the learner doesn’t see enough training instances
  → Can we estimate how many instances are enough?

Learning setting



- set of instances $\mathcal{X}$
- set of hypotheses (models) H
- set of possible target concepts C
- unknown probability distribution $\mathcal{D}$ over instances

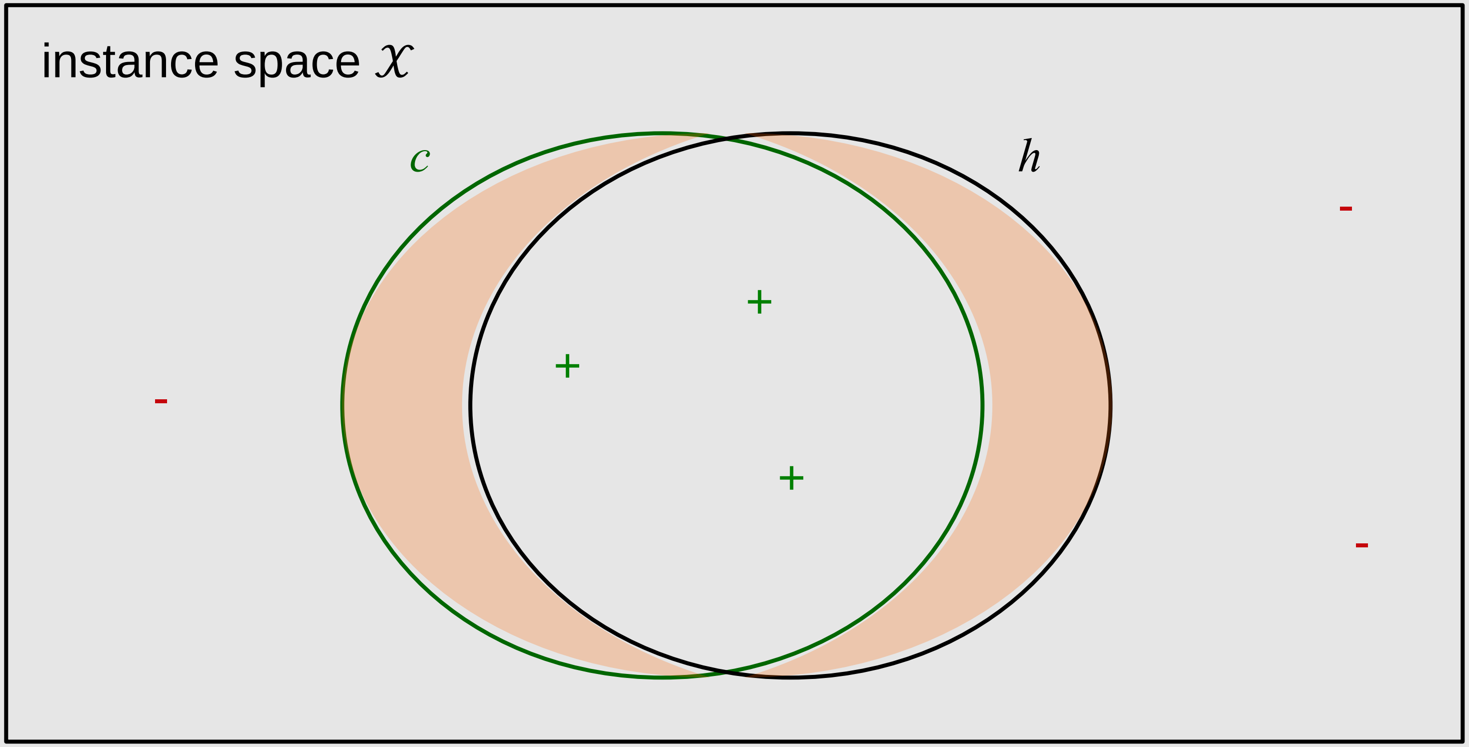
Learning setting

- learner is given a set D of training instances $\langle \boldsymbol{x}, c(\boldsymbol{x}) \rangle$ for some target concept c in C
  - each instance $\boldsymbol{x}$ is drawn from distribution $\mathcal{D}$
  - class label $c(\boldsymbol{x})$ is provided for each $\boldsymbol{x}$
- learner outputs hypothesis h modeling c

True error of a hypothesis

the *true error* of hypothesis h refers to how often h is wrong on future instances drawn from $\mathcal{D}$

$$error_{\mathcal{D}}(h) \equiv P_{x \in \mathcal{D}}\left[c(\boldsymbol{x}) \neq h(\boldsymbol{x})\right]$$



Training error of a hypothesis

the *training error* of hypothesis h refers to how often h is wrong on instances in the training set D

$$error_D(h) \equiv P_{x\in D}[c(x) \neq h(x)] = \frac{\sum_{x\in D} \delta(c(x) \neq h(x))}{|D|}$$

Can we bound $error_{\mathcal{D}}(h)$ in terms of $error_{\mathrm{D}}(h)$?

Is approximately correct good enough?

To say that our learner L has learned a concept, should we require $error_{\mathcal{D}}(h) = 0$?

this is not realistic:

- unless we've seen every possible instance, there may be multiple hypotheses that are consistent with the training set
- there is some chance our training sample will be unrepresentative

Probably approximately correct learning?

Instead, we'll require that

- the error of a learned hypothesis h is bounded by some constant ε
- the probability of the learner failing to learn an accurate hypothesis is bounded by a constant δ

Probably Approximately Correct (PAC) learning [Valiant, *CACM* 1984]

- Consider a class C of possible target concepts defined over a set of instances $\mathcal{X}$ of length n, and a learner L using hypothesis space H

- C is PAC learnable by L using H if, for all

  $c \in C$

  distributions $\mathcal{D}$ over $\mathcal{X}$

  ε such that $0 < \varepsilon < 0.5$

  δ such that $0 < \delta < 0.5$

- learner L will, with probability at least $(1\text{-}\delta)$, output a hypothesis $h \in H$ such that $error_{\mathcal{D}}(h) \leq \varepsilon$ in time that is polynomial in

  $1/\varepsilon$

  $1/\delta$

  n

  size(c)

PAC learning and consistency

- Suppose we can find hypotheses that are consistent with m training instances.
- We can analyze PAC learnability by determining whether
  1. m grows polynomially in the relevant parameters
  2. the processing time per training example is polynomial

Version spaces

- A hypothesis h is *consistent* with a set of training examples D of target concept if and only if $h(\boldsymbol{x}) = c(\boldsymbol{x})$ for each training example $\langle \boldsymbol{x}, c(\boldsymbol{x}) \rangle$ in D

$$consistent(h, D) \equiv (\forall \langle x, c(x) \rangle \in D)\ \ h(x) = c(x)$$

- The version space $VS_{H,D}$ with respect to hypothesis space H and training set D, is the subset of hypotheses from H consistent with all training examples in D

$$VS_{H,D} \equiv \{h \in H \mid consistent(h, D)\}$$

Exhausting the version space

- The version space $VS_{H,\mathrm{D}}$ is ε-exhausted with respect to c and *D* if every hypothesis $h \in VS_{H,\mathrm{D}}$ has true error $< \varepsilon$

$$\left(\forall h \in VS_{H,\mathrm{D}}\right) error_{\mathcal{D}}(h) < \varepsilon$$

Exhausting the version space

- Suppose that every h in our version space $VS_{H,D}$ is consistent with m training examples
- The probability that $VS_{H,D}$ is not ε-exhausted (i.e. that it contains some hypotheses that are not accurate enough)

$$\le |H| e^{-\varepsilon m}$$

Proof:

$(1-\varepsilon)^m$ probability that some hypothesis with error > ε is consistent with m training instances

$k(1-\varepsilon)^m$ there might be k such hypotheses

$|H|(1-\varepsilon)^m$ k is bounded by $|H|$

$\le |H| e^{-\varepsilon m}$ $(1-\varepsilon) \le e^{-\varepsilon}$ when $0 \le \varepsilon \le 1$

Sample complexity for finite hypothesis spaces

[Blumer et al., *Information Processing Letters* 1987]

- we want to reduce this probability below δ

$$|H| e^{-em} \pounds d$$

- solving for *m* we get

$$m \geq \frac{1}{e}\left(\ln|H| + \ln\left(\frac{1}{d}\right)\right)$$

log dependence on *H*

ε has stronger influence than δ

PAC analysis example: learning conjunctions of Boolean literals

- each instance has n Boolean features
- learned hypotheses are of the form $Y = X_1 \wedge X_2 \wedge \neg X_5$

How many training examples suffice to ensure that with prob ≥ 0.99, a consistent learner will return a hypothesis with error ≤ 0.05 ?

there are 3^n hypotheses (each variable can be present and unnegated, present and negated, or absent) in H

$$m \geq \frac{1}{.05}\left(\ln\left(3^n\right) + \ln\left(\frac{1}{.01}\right)\right)$$

for n=10, $m \geq 312$ for n=100, $m \geq 2290$

PAC analysis example: learning conjunctions of Boolean literals

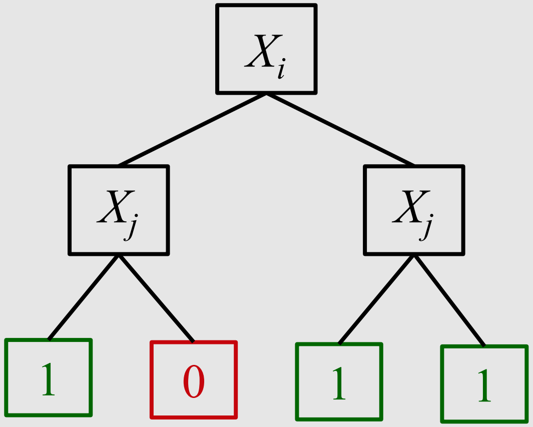
- we’ve shown that the sample complexity is polynomial in relevant parameters: $1/\varepsilon$, $1/\delta$, n

- to prove that Boolean conjunctions are PAC learnable, need to also show that we can find a consistent hypothesis in polynomial time (the FIND-S algorithm in Mitchell, Chapter 2 does this)

```
FIND-S:
    initialize $h$ to the most specific hypothesis   $x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \ldots\ x_n \wedge \neg x_n$
    for each positive training instance $\boldsymbol{x}$
            remove from $h$ any literal that is not satisfied by $\boldsymbol{x}$
    output hypothesis $h$
```

PAC analysis example: learning decision trees of depth 2

- each instance has n Boolean features
- learned hypotheses are DTs of depth 2 using only 2 variables



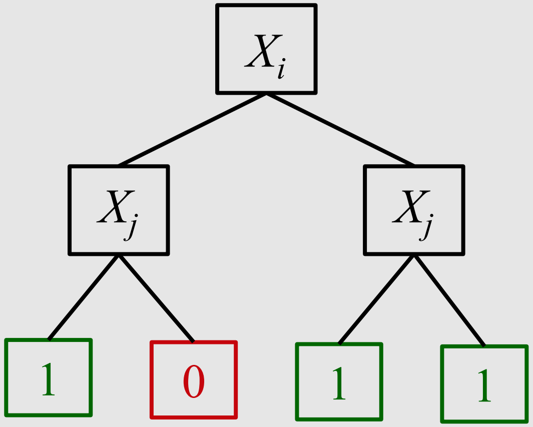
$$|H| = \underbrace{\binom{n}{2}}_{\text{\# possible split choices}} \times \underbrace{16}_{\text{\# possible leaf labelings}} = \frac{n(n-1)}{2} \times 16 = 8n(n-1)$$

# possible split choices

possible leaf labelings

PAC analysis example: learning decision trees of depth 2

- each instance has n Boolean features
- learned hypotheses are DTs of depth 2 using only 2 variables



How many training examples suffice to ensure that with prob ≥ 0.99, a consistent learner will return a hypothesis with error ≤ 0.05 ?

$$m \geq \frac{1}{.05}\left(\ln\left(8n^2 - 8n\right) + \ln\left(\frac{1}{.01}\right)\right)$$

for $n=10$, $m \geq 224$ for $n=100$, $m \geq 318$

PAC analysis example: K-term DNF is not PAC learnable

- each instance has n Boolean features
- learned hypotheses are of the form $Y = T_1 \vee T_2 \vee \ldots \vee T_k$ where each T_i is a conjunction of n Boolean features or their negations

$|H| \leq 3^{nk}$, so sample complexity is polynomial in the relevant parameters

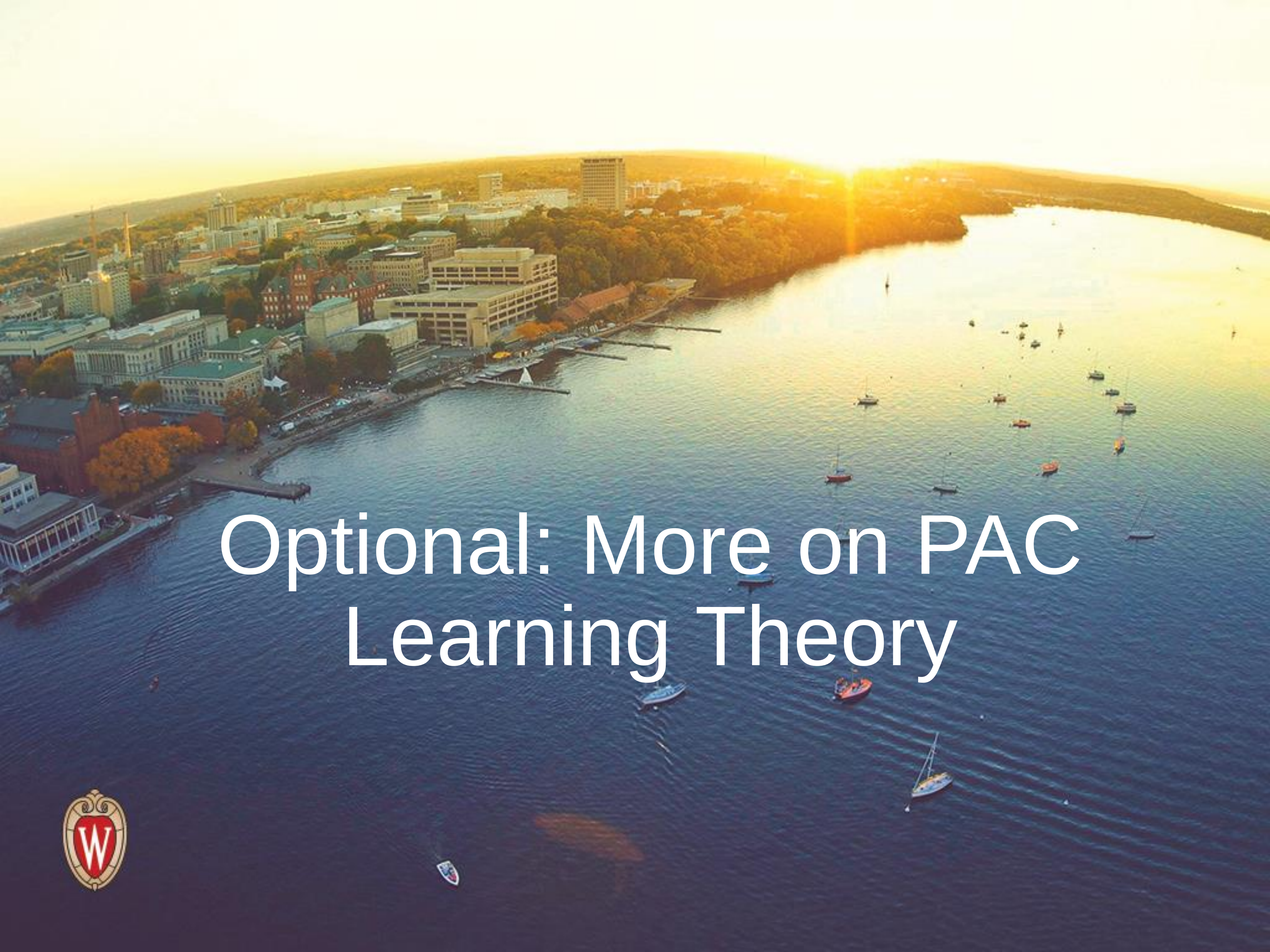
$$m \geq \frac{1}{e}\left(nk \ln(3) + \ln\left(\frac{1}{d}\right)\right)$$

however, the computational complexity (time to find consistent h) is not polynomial in m (e.g. graph 3-coloring, an NP-complete problem, can be reduced to learning 3-term DNF)

Comments on PAC learning

- PAC analysis formalizes the learning task and allows for non-perfect learning (indicated by ε and δ)
  - Requires polynomial computational time
- finding a consistent hypothesis is sometimes easier for larger concept classes
  - e.g. although k-term DNF is not PAC learnable, the more general class k-CNF is

- PAC analysis has been extended to explore a wide range of cases
  - the target concept not in our hypothesis class: see optional material
  - infinite hypothesis class (VC-dimension theory): see optional material
  - noisy training data
  - learner allowed to ask queries
  - restricted distributions (e.g. uniform) over $\mathcal{D}$
  - etc.
- most analyses are worst case
- sample complexity bounds are generally not tight



Optional: More on PAC Learning Theory

What if the target concept is not in our hypothesis space?

- so far, we've been assuming that the target concept c is in our hypothesis space; this is not a very realistic assumption

- *agnostic learning* setting
  - don't assume $c \in H$
  - learner returns hypothesis h that makes fewest errors on training data

Hoeffding bound

- we can approach the agnostic setting by using the Hoeffding bound
- let $Z_1 \ldots Z_m$ be a sequence of m independent Bernoulli trials (e.g. coin flips), each with probability of success $E[Z_i] = p$
- let $S = Z_1 + \cdots + Z_m$

$$P[S < (p - \varepsilon)m] \leq e^{-2m\varepsilon^2}$$

Agnostic PAC learning

- applying the Hoeffding bound to characterize the error rate of a given hypothesis

$$P\left[error_{\mathcal{D}}(h) > error_{D}(h) + \varepsilon\right] \leq e^{-2m\varepsilon^2}$$

- but our learner searches hypothesis space to find h_{best}

$$P\left[error_{\mathcal{D}}(h_{best}) > error_{D}(h_{best}) + \varepsilon\right] \leq |H|e^{-2m\varepsilon^2}$$

- solving for the sample complexity when this probability is limited to δ

$$m \geq \frac{1}{2\varepsilon^2}\left(ln|H| + ln\left(\frac{1}{\delta}\right)\right)$$

What if the hypothesis space is not finite?

- **Q:** If H is infinite (e.g. the class of perceptrons), what measure of hypothesis-space complexity can we use in place of $|H|$?

- **A:** the largest subset of $\mathcal{X}$ for which H can guarantee zero training error, regardless of the target function.

this is known as the *Vapnik-Chervonenkis dimension* (VC-dimension)

Shattering and the VC dimension

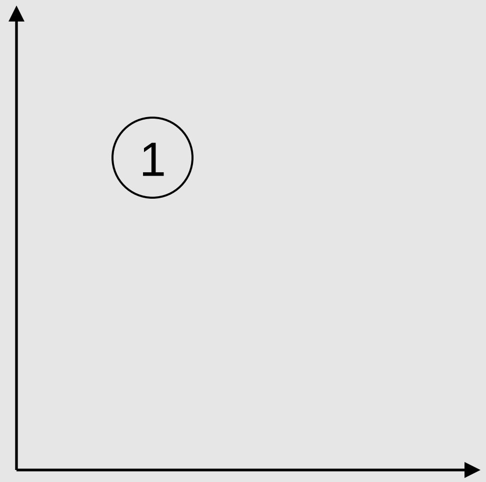
- a set of instances D is *shattered* by a hypothesis space H iff for every dichotomy of D there is a hypothesis in H consistent with this dichotomy

- the *VC dimension* of H is the size of the largest set of instances that is shattered by H

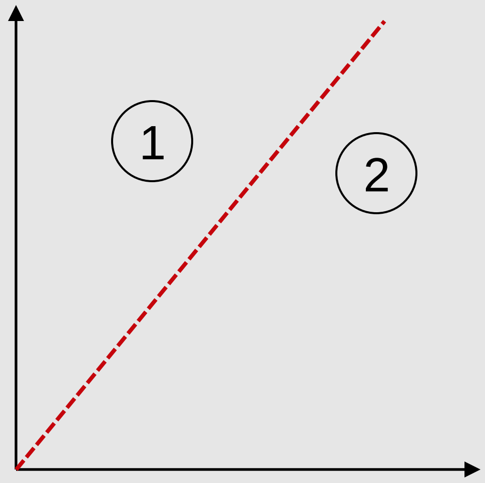
Infinite hypothesis space with a finite VC dimension

consider: H is set of lines in 2D (i.e. perceptrons in 2D feature space)

can find an h consistent with 1 instance no matter how it's labeled



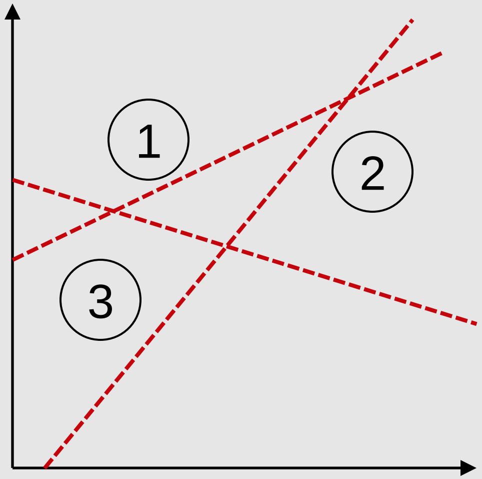
can find an h consistent with 2 instances no matter labeling



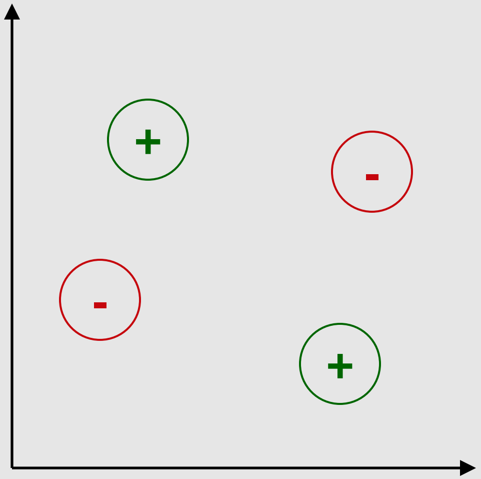
Infinite hypothesis space with a finite VC dimension

consider: H is set of lines in 2D

can find an h consistent with 3 instances no matter labeling (assuming they're not colinear)



cannot find an h consistent with 4 instances for some labelings



can shatter 3 instances, but not 4, so the VC-dim(H) = 3

more generally, the VC-dim of hyperplanes in n dimensions = n+1

VC dimension for finite hypothesis spaces

for finite H, VC-dim$(H) \leq \log_2|H|$

Proof:

suppose VC-dim$(H) = d$

for d instances, 2^d different labelings possible

therefore H must be able to represent 2^d hypotheses

$2^d \leq |H|$

$d =$ VC-dim$(H) \leq \log_2|H|$

Sample complexity and the VC dimension

- using VC-dim(H) as a measure of complexity of H, we can derive the following bound [Blumer et al., *JACM* 1989]

$$m \geq \frac{1}{e}\left(4\log_2\left(\frac{2}{d}\right) + 8\text{VC-dim}(H)\log_2\left(\frac{13}{e}\right)\right)$$

m grows log × linear in ε (better than earlier bound)

can be used for both finite and infinite hypothesis spaces

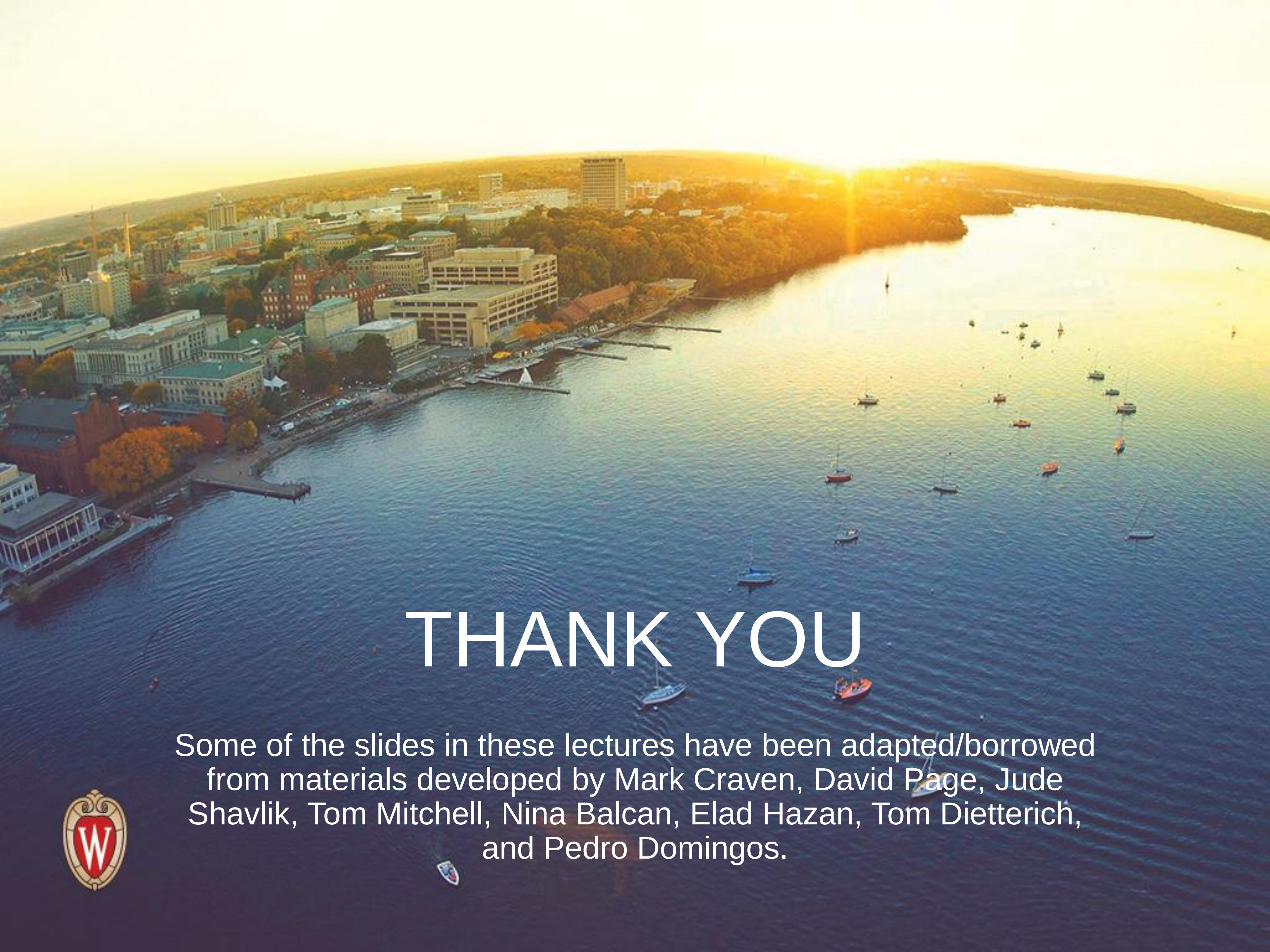
Lower bound on sample complexity

[Ehrenfeucht et al., *Information & Computation* 1989]

- there exists a distribution $\mathcal{D}$ and target concept in C such that if the number of training instances given to L

$$m < \max\left[\frac{1}{e}\log\left(\frac{1}{d}\right), \frac{\text{VC-dim}(C) - 1}{32e}\right]$$

then with probability at least δ, L outputs h such that $error_{\mathrm{D}}(h) > \varepsilon$



THANK YOU

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, and Pedro Domingos.