CS 760: Machine Learning
Learning Theory

University of Wisconsin-Madison

Outline

- **Basic error decomposition**
  - goals of learning theory, different decompositions
- **Bias-variance tradeoff**
  - definition, intuition, sample complexity bounds

Outline

- **Basic error decomposition**
  - goals of learning theory, different decompositions

- **Bias-variance tradeoff**
  - definition, intuition, sample complexity bounds

Why learning theory?

Formal analysis of algorithms is important in all areas of CS:

- Example: binary search has time complexity $O(\log n)$
- Example: running gradient descent on a smooth and convex function yields an ε-suboptimal point in $O(1/\varepsilon)$ iterations

We desire a rigorous understanding of algorithms to

- be able to predict how an algorithm will work on new problems
- understand when a problem is inherently hard (lower bounds)
- understand when a problem can be learned efficiently (time, space, training set size)
- provide guarantees on performance under certain conditions

Learning Theory

- One basic approach: try to understand how the performance of a learned model depends on
  - the difficulty and amount of data
  - the complexity of the model class
  - the training procedure

- Error decomposition breaks down the total error of a model into different errors coming from each of these components

Bayes error

Given p(x,y) and loss function L, the **best one can do** is the Bayes optimal predictor

$$h^*(x) = \arg\min_{\hat{y}} \sum_y L(\hat{y}, y) p(y|x)$$

The corresponding Bayes error is

$$R^* = \mathbb{E}_x \left[\min_{\hat{y}} \sum_y L(\hat{y}, y) p(y|x) \right]$$

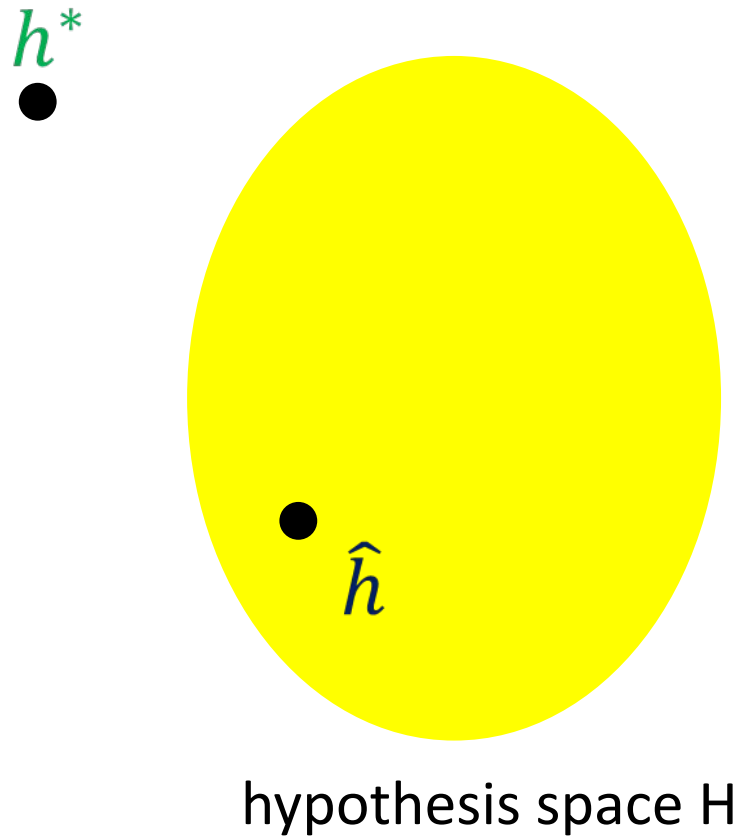
But h* is typically not in the hypothesis space of a learning algorithm. (e.g. no weights of a given neural net architecture can express h*)

Error decomposition

Suppose we have a hypothesis class H of candidate prediction functions

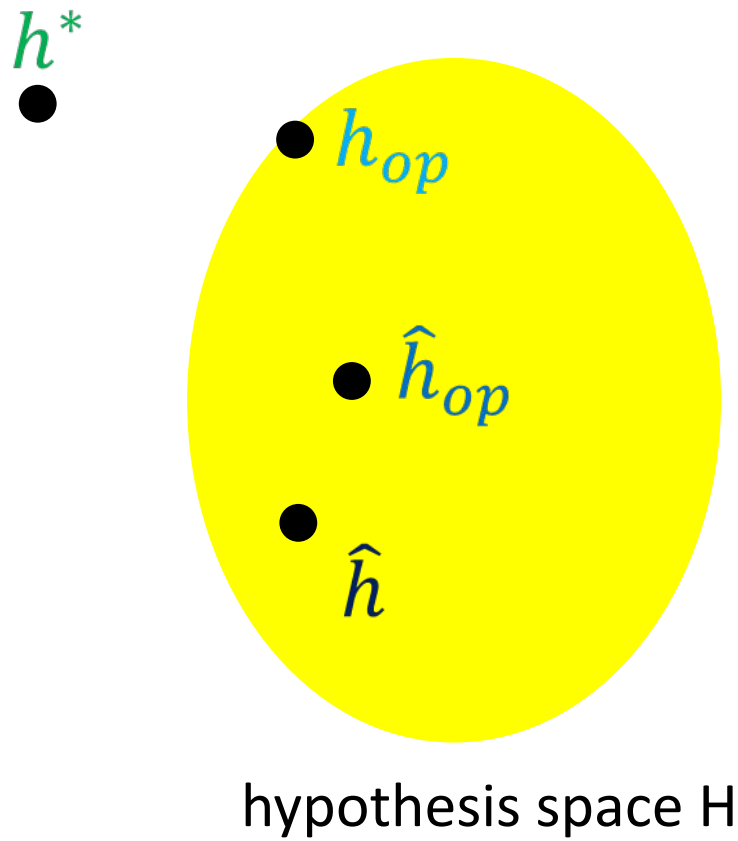
Let $err(h)$ be the expected error of hypothesis h on the test distribution, also known as the **risk**

We can try to understand why the error of the hypothesis $\hat{h}$ returned by a learning algorithm is larger than that of the optimal classifier h^* by **decomposing the error**



Error decomposition

- 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



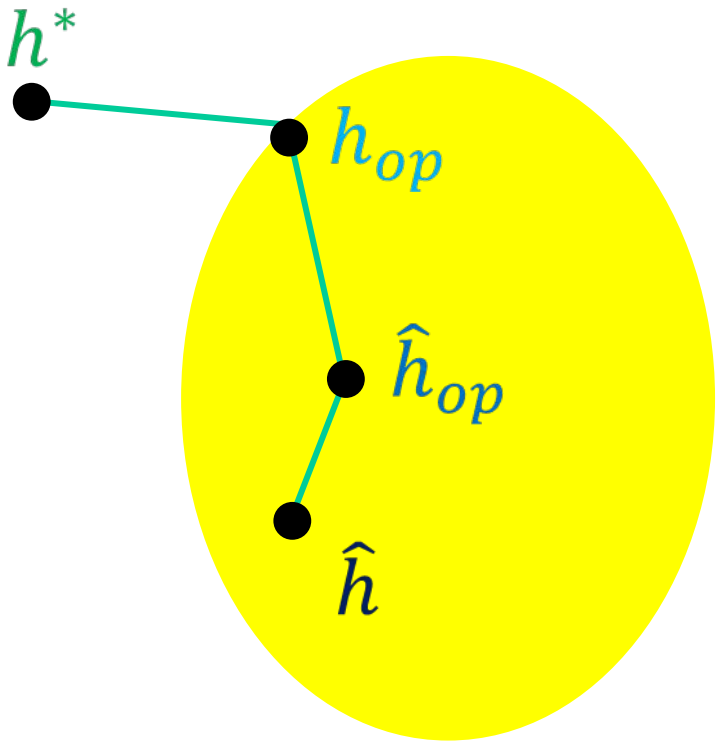
Error decomposition

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

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

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

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



hypothesis space H

Error decomposition

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

$$= err(h_{opt}) - err(h^*)$$ → **Approximation error** due to problem modeling (our choice of hypothesis class)

$$+ err(\hat{h}_{opt}) - err(h_{opt})$$ → **Estimation error** due to finite data

$$+ err(\hat{h}) - err(\hat{h}_{opt})$$ → **Optimization error** due to imperfect optimization

Error decomposition

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

highly data-dependent and so difficult to control mathematically without strong assumptions

Approximation error due to problem modeling (our choice of hypothesis class)

primary concern of (statistical) learning theory

Estimation error due to finite data

important but addressed by **optimization** theory, and in-practice we often get zero training error (assume $\hat{h} = \hat{h}_{opt}$)

Optimization error due to imperfect optimization

Bounding estimation error

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

empirical risk

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

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

I'm the minimizer

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

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

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

depends on hypothesis space and data, **not** learning algorithm

Another error decomposition

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

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

same quantity as before

- We can compute the training error $\widehat{err}(\hat{h})$: if it is small, then a small generalization gap implies small test error
- How do we bound the generalization gap?

Bounding the generalization gap

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

The supremum characterizes the **capacity** of the hypothesis class H to overfit the training data.

Learning theory tries to bound it by some function of the number of training examples and a measure of how "big" the hypothesis class is.

e.g. next class: $\sup_{h \in H} |err(h) - \widehat{err}(h)| \leq \tilde{O}\left(\sqrt{\frac{\text{VC-dimension}(H)}{\#\text{training examples}}}\right)$

Outline

- **Basic error decomposition**
  - goals of learning theory, different decompositions

- **Bias-variance tradeoff**
  - definition, intuition, sample complexity bounds

Yet another decomposition

The bias-variance decomposition separates the expected risk of a model training procedure (learning algorithm) into

- bias: expected error of the learned model
- variance: sensitivity of the algorithm to the training set
- irreducible error: inherent noisiness of the problem

Statistical way of understanding the tradeoff between approximation error (bias) and estimation error (variance)

Setup

Consider the task of learning a regression model given a training set $D = \left\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\right\} \subset X \times Y$

Assume data is generated by the model $y = f(x) + \varepsilon$, where ε is a random variable with mean zero and variance σ^2.

We use D to train a model $\hat{f}: X \mapsto Y$

What is the **expected MSE** of $\hat{f}$ at a fixed point $x \in X$?

Goal

Define the MSE at a fixed point $x \in X$ as

$$err_x(\hat{f}) = \mathbb{E}_{y|x}\left[(\hat{f}(x) - y)^2\right]$$

Related to the **risk** err but at a fixed input point rather than w.r.t. a joint distribution over (x, y) pairs:

$$err(\hat{f}) = \mathbb{E}_{(x,y)}\left[(\hat{f}(x) - y)^2\right]$$

Interested in **expected MSE** w.r.t. the randomness of drawing *D*:

$$\mathbb{E}_D\left[err_x(\hat{f})\right] = \mathbb{E}_D\mathbb{E}_{y|x}\left[(\hat{f}(x) - y)^2\right]$$

Separating out the irreducible error

$$\mathbb{E}\left[\left(\hat{f}(x) - y\right)^2\right]$$

$$= \mathbb{E}\left[\left(\hat{f}(x) - f(x) - \varepsilon\right)^2\right]$$

$$= \mathbb{E}\left[\left(\hat{f}(x) - f(x)\right)^2\right] + 2\mathbb{E}\left[\left(\hat{f}(x) - f(x)\right)\varepsilon\right] + \mathbb{E}[\varepsilon^2]$$

$$= \underbrace{\mathbb{E}\left[\left(\hat{f}(x) - f(x)\right)^2\right]}_{\text{(squared) bias + variance}} + 0 + \underbrace{\sigma^2}_{\text{irreducible error}}$$

Deriving the bias-variance decomposition

$$\mathbb{E}\left[\left(\hat{f}(x) - f(x)\right)^2\right]$$

$$= \mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}\left[\hat{f}(x)\right] + \mathbb{E}\left[\hat{f}(x)\right] - f(x)\right)^2\right]$$

$$= \mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)^2\right]$$ ⟵ variance

$$+ \left(\mathbb{E}\left[\hat{f}(x)\right] - f(x)\right)^2$$ ⟵ squared bias

$$+2\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)\left(\mathbb{E}\left[\hat{f}(x)\right] - f(x)\right)\right]$$

$$= \mathbb{E}\left[\hat{f}(x)\right]^2 - \mathbb{E}\left[\hat{f}(x)\right]^2 + \mathbb{E}\left[\hat{f}(x)\right]\mathbb{E}[f(x)] - \mathbb{E}\left[\hat{f}(x)\right]\mathbb{E}[f(x)] = 0$$

What have we derived?

$$\mathbb{E}_D\left[err_x(\hat{f})\right]$$
$$= \mathbb{E}_D\mathbb{E}_{y|x}\left[\left(\hat{f}(x) - y\right)^2\right]$$
$$= \left(\mathbb{E}_D\left[\hat{f}(x)\right] - f(x)\right)^2 + \mathbb{E}_D\left[\left(\hat{f}(x) - \mathbb{E}_D\left[\hat{f}(x)\right]\right)^2\right] + \sigma^2$$

bias: how far away is the average prediction from the true function?

variance: how different is the prediction on average across different samples of the dataset?

irreducible error

Understanding bias: $\mathbb{E}_D\left[\hat{f}(x)\right] - f(x)$

Large if $\hat{f}(x)$ is far away from $f(x)$ across different draws of the dataset D

Indicates that the learning algorithm does not fit the data well, i.e. is **underfitting**

Can be caused by:

- an inflexible model class, e.g. fitting a nonlinear f with a hypothesis class of linear models
- poor optimization, i.e. not minimizing the training error

Understanding variance: $\mathbb{E}_D\left(\hat{f}(x) - \mathbb{E}_D[\hat{f}(x)]\right)^2$

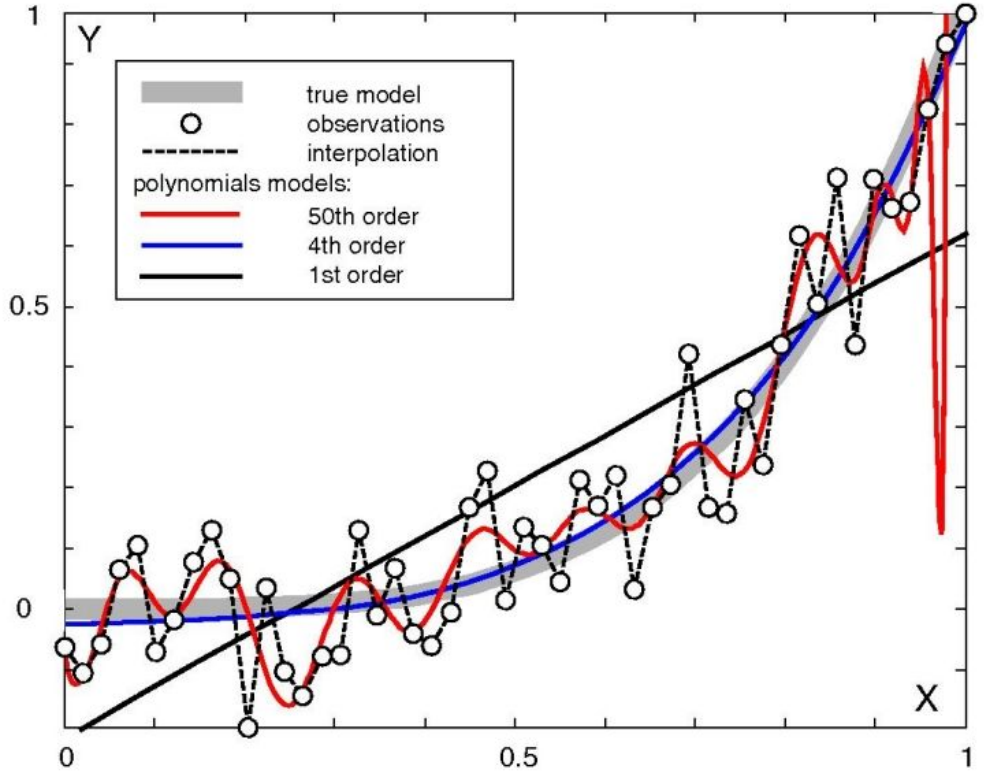
Large if the prediction varies $\hat{f}(x)$ significantly across different random draws of the dataset D

Indicates that the learning algorithm may be **overfitting**

Can be caused by using a high-capacity model that can adapt to random noise rather than the true signal f

Example: Polynomial Interpolation

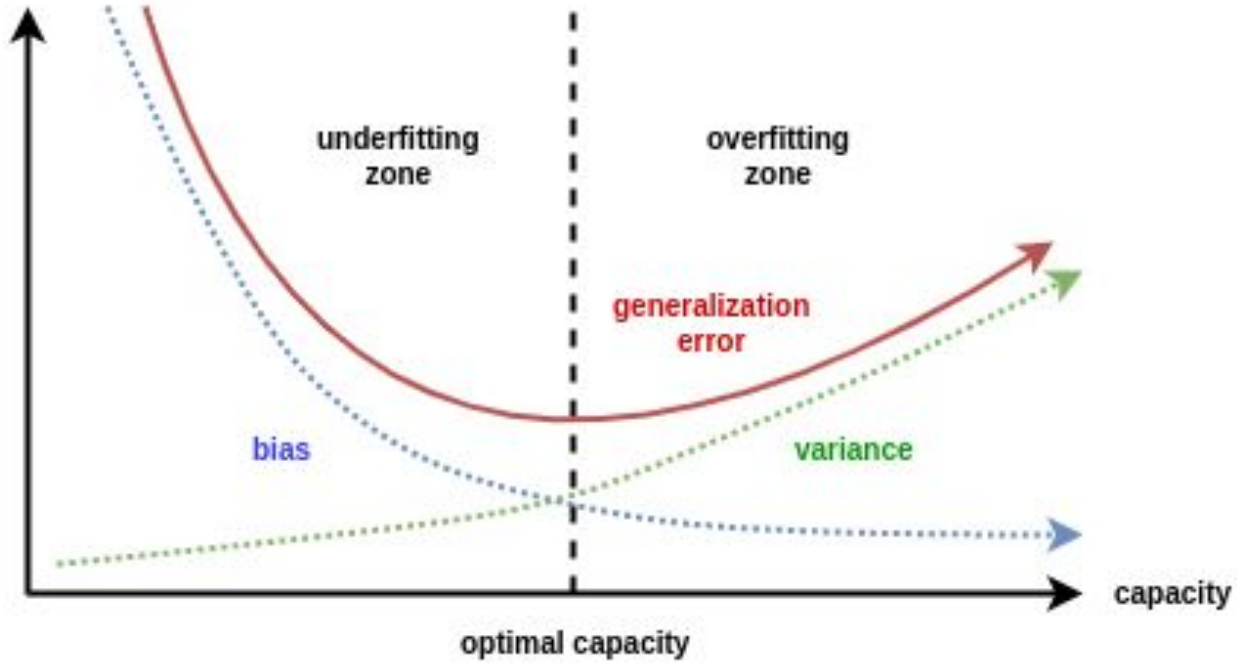
- 1st order polynomial has high bias, low variance
- 50th order polynomial has low bias, high variance
- 4th order polynomial represents a good trade-off



The bias-variance tradeoff

The B-V decomposition models predictive error as having two controllable components

- more expressive learners reduce bias but increase variance
- typically depicted via a capacity vs. error plot suggesting an optimal capacity
- can be extended beyond regression to classification



Break & Quiz

True or False: increasing the number of neighbors (k) in k-NN will typically **increase the bias** and **reduce the variance**

Answer: True

True or False: increasing the regularization strength in LASSO will typically **increase the bias** and **reduce the variance**

Answer: True

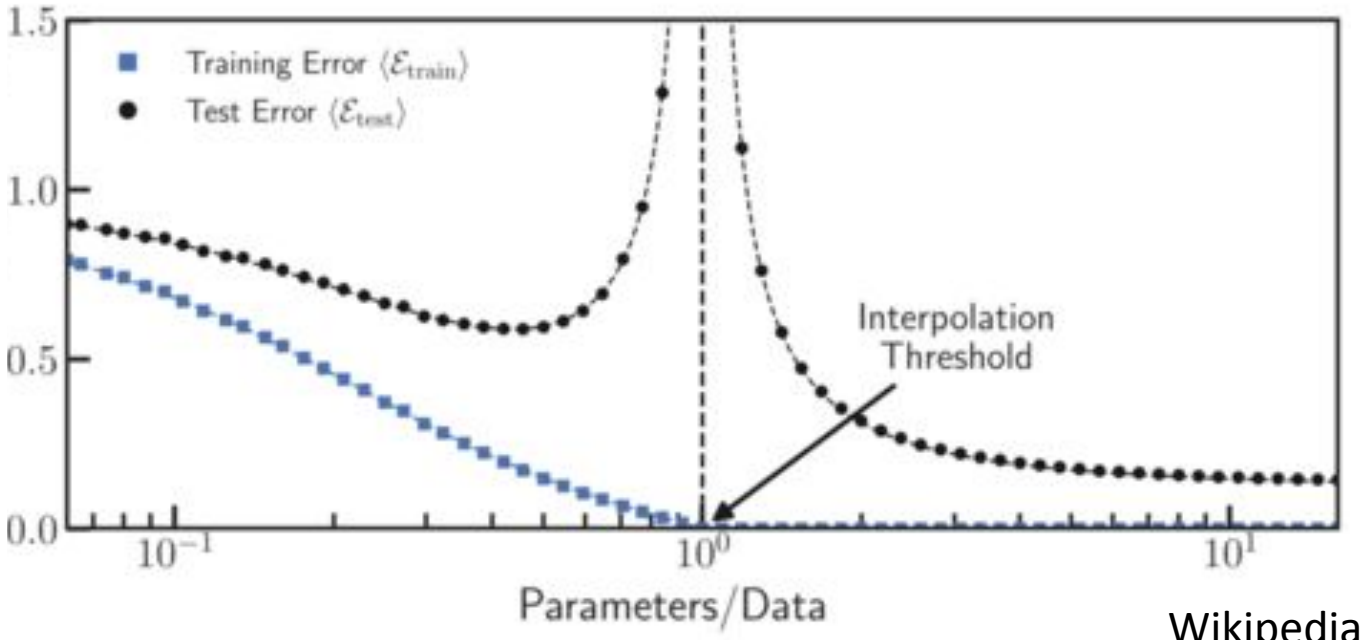
True or False: adding degree 2 polynomial features to a linear model will typically **increase the bias** and **reduce the variance**

Answer: False

Caveats

- There is not always a strict tradeoff: with ensemble methods we can often reduce bias and/or variance without increasing the other term

- Neural networks (and even simpler models) sometimes yield a **double descent** phenomenon, where error goes down, then up, **then down again** as model capacity increases



Wikipedia

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