Model Selection, Validation, and Regularization

Lecture 8

Last Time

- The ***no-free-lunch theorem*** tells us that there is no universal learning algorithm that will work best on all problems.
- Further, for every algorithm, there is a problem it fails on, even though another succeeds
- Instead, for every learning problem we must balance the bias-complexity tradeoff using prior knowledge
- Textbook: chapter 5

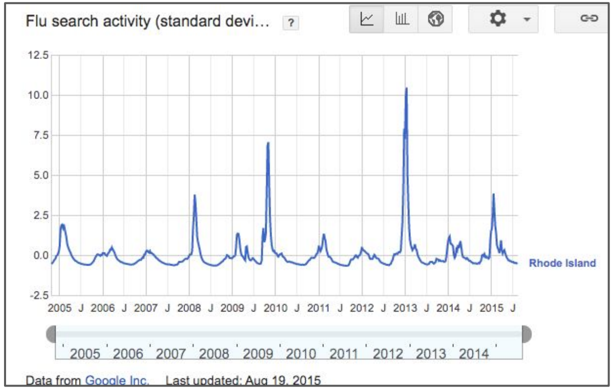
This Class

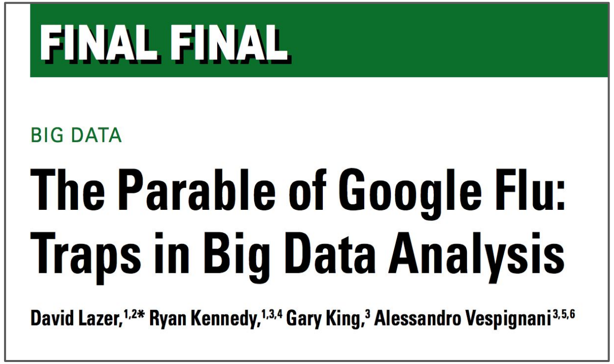
- How do we balance the bias-complexity tradeoff in practice?
- Textbook: chapters 11.0, 11.2, 11.3, 13.0, 13.1, 13.4

Motivation

Example: Google Flu Trends

- Used search trends to predict flu epidemics in 25 different countries
- Paper reported that model predicted outbreaks up to 10 days before CDC models
- Massively overestimated some flu outbreaks and missed others



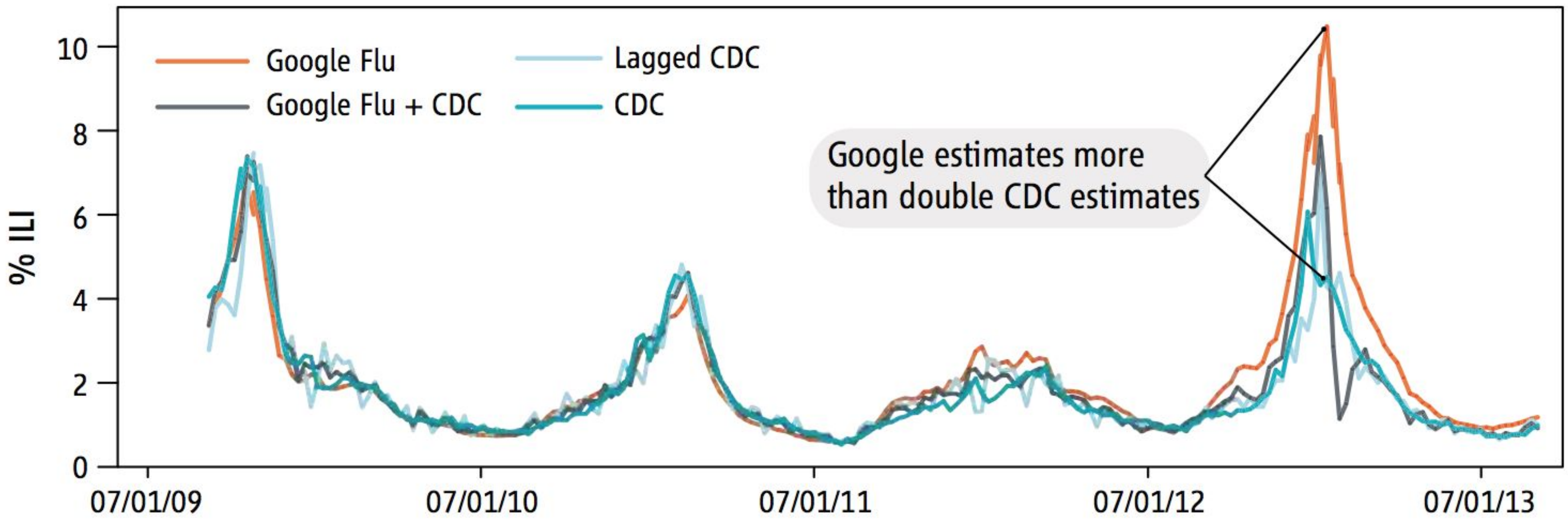

FINAL FINAL

BIG DATA

The Parable of Google Flu: Traps in Big Data Analysis

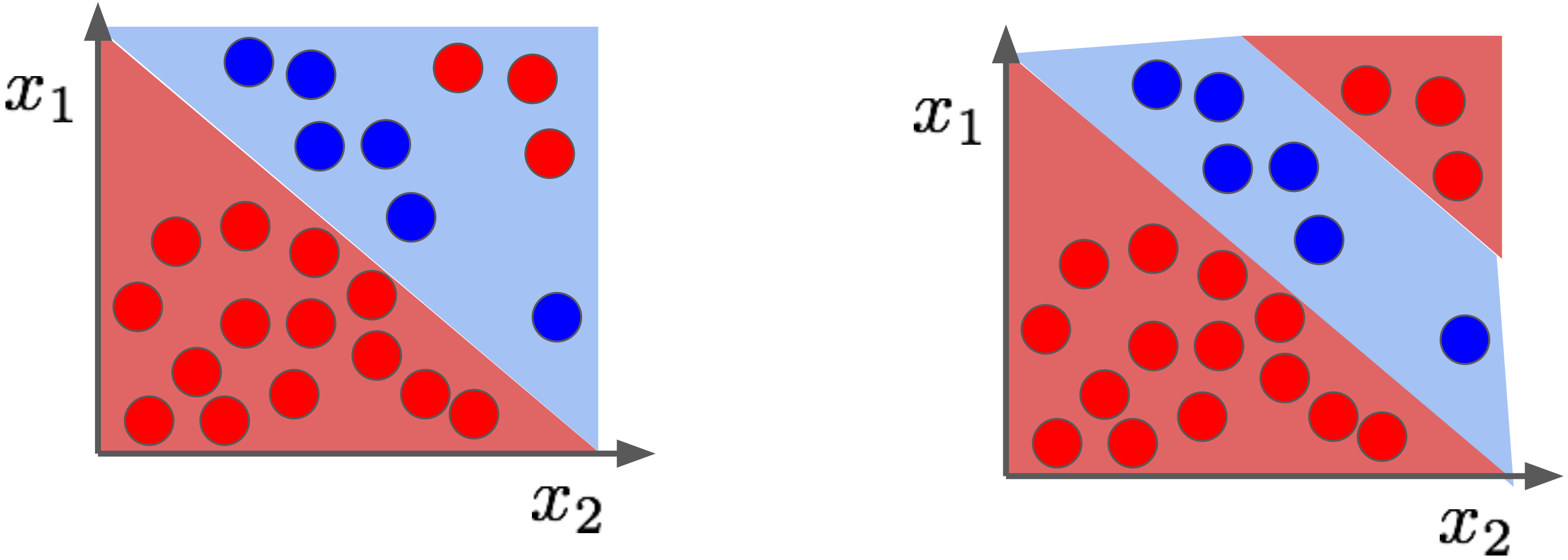
David Lazer,[1,2*] Ryan Kennedy,[1,3,4] Gary King,[3] Alessandro Vespignani[3,5,6]

Big Data Hubris

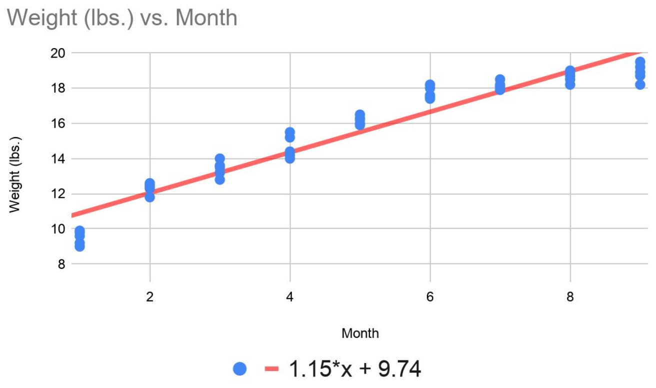


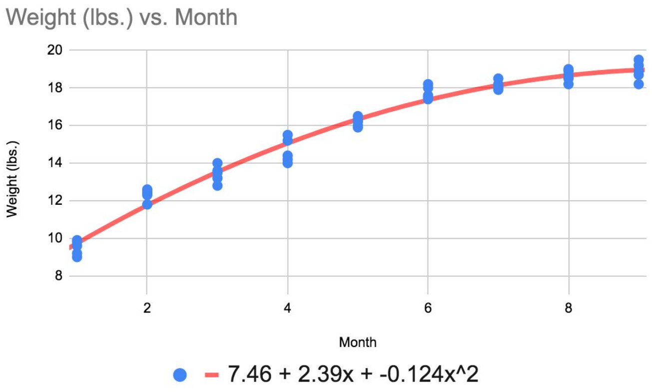
Model Selection and Validation

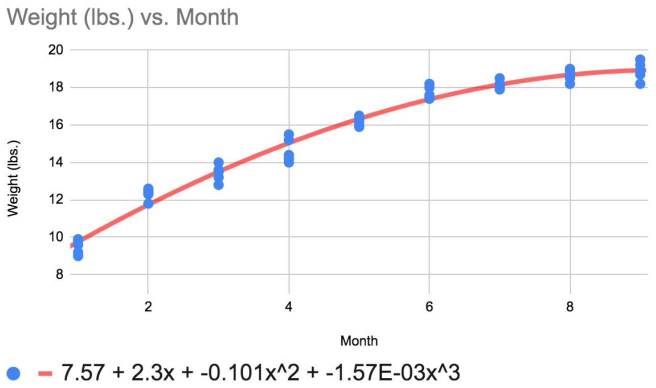
Which Would You Choose?

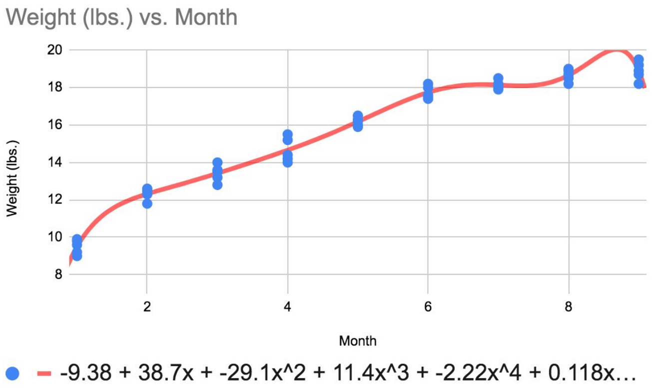


Which Would You Choose?







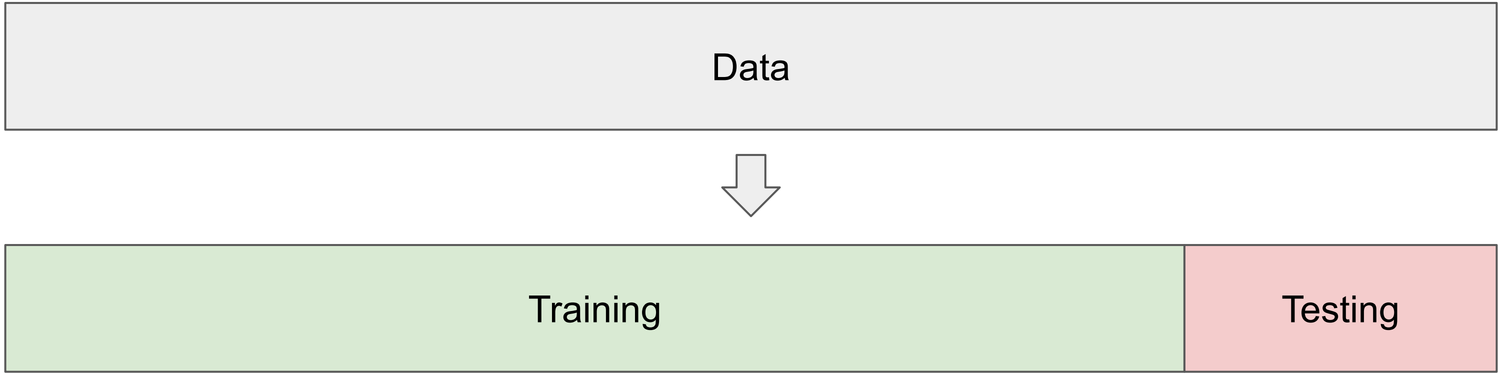


Everyone Needs a Little Validation (Data)

- As we increase polynomial order, we lower empirical risk
- But seems like overfitting!
- We can balance between bias and complexity using a set of validation data

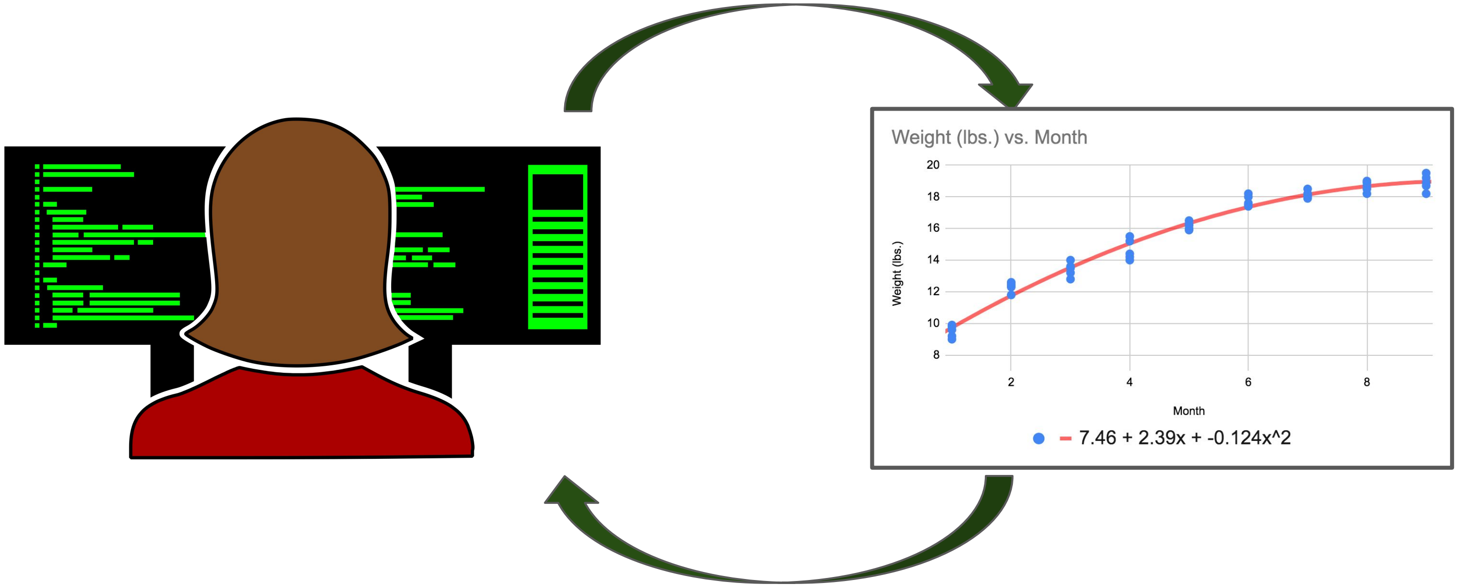
Previous Set Up

So far we've held out a test set to get an estimate of $L_{\mathcal{D}}(h)$



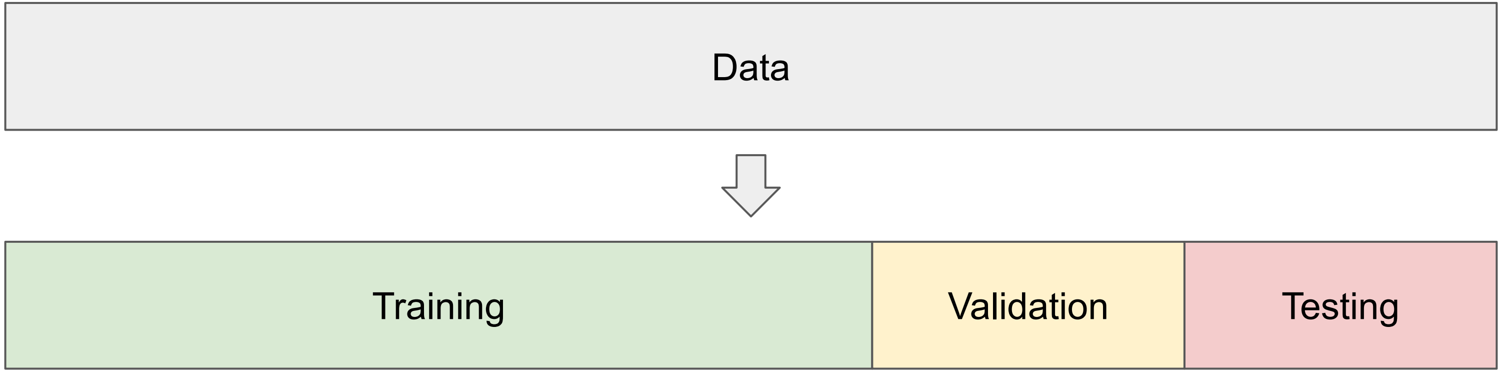
No Peeking!

- If we evaluate multiple hypotheses on the test set, and then pick the best one, then it is no longer an unbiased estimate of $L_{\mathcal{D}}(h)$



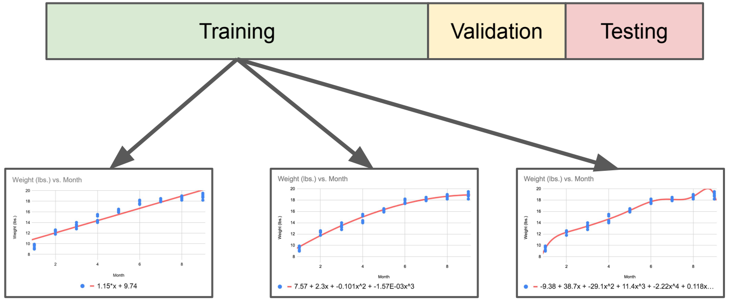
Training-Validation-Test Split

Use training data to train, validation data to select the best model, and testing data for a estimation of true error

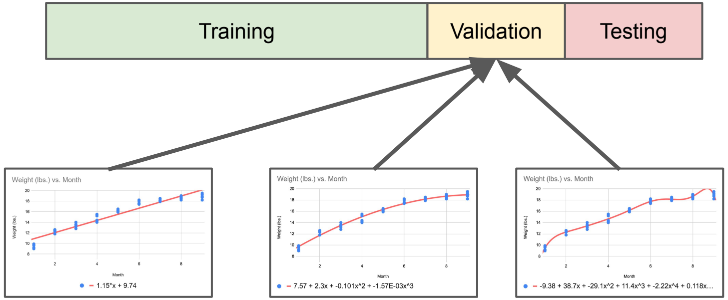


Model Selection with Validation

- Train different algorithms (or the same algorithm with different hyperparameters) on a given training set



- Next, we choose the hypothesis that minimizes the error over the validation set



Model Selection with Validation

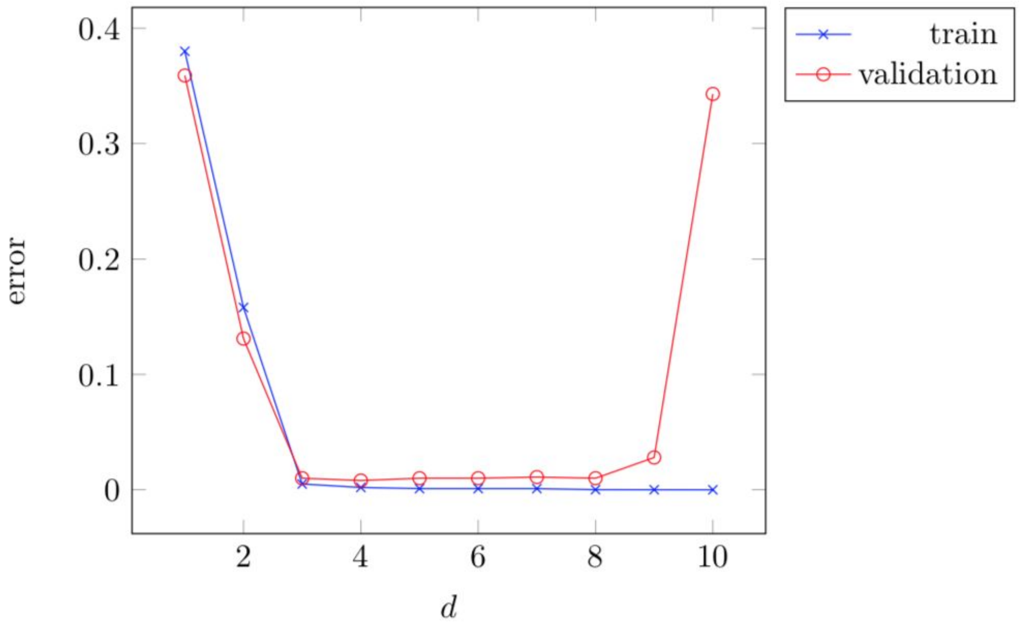
- Once we've selected our model, we can evaluate *only* that model *once* on the test set, if we want an unbiased estimate of $L_{\mathcal{D}}(h)$

Model Selection Curves

Shows Training and Validation error as a function of complexity

In baby weight example:

- Initially, train and validation error is high (underfitting)
- As we increase complexity:
    - Training error decreases
    - Validation decreases then increases (overfitting)



Understanding Machine Learning.
Shalev-Shwartz and Ben-David, 2014.

k-fold Cross Validation

Previous methods work great when you have a ton of data
What if you don’t want to “waste data” on those?

General idea of k-fold across all sets:

1. Split data into *k* subsets of equal size
2. For each fold, train on the union of all other folds and estimate error using the fold
3. Average the error across all folds

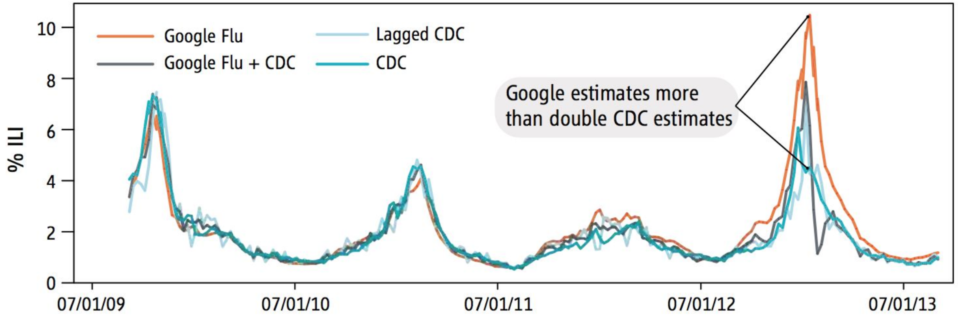
Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

Question

Why was Model Selection Hard for Google Flu?

What challenges made model selection difficult for Google Flu Trends?

(Select all that apply.)



A: Limited amount of validation data

B: K-fold cross valid. not applicable

C: Examples violate i.i.d. assumption

D: Low hypothesis class bias

Answer

Answer: ACD

- A: Limited amount of validation data
  - Number of weeks for which data is collected is limited, cannot get more
- B: K-fold cross valid. not applicable
  - Doesn’t solve all problems, but could split weeks into K folds and predict held-out data
- C: Examples violate i.i.d. assumption
  - Lots of causes! One example: changes in search algorithm change user behavior
- D: Low hypothesis class bias
  - With so many possible search times, high probability of finding model that fits train and validation data

What if Learning Fails?

What if Learning Fails?

- Need to smartly choose what is the issue: approximation or estimation error

- Recall:

$$\epsilon_{app} = \min_{h \in H} L_D(h)$$

$$\epsilon_{est} = L_D(h_S) - \epsilon_{app}$$

What do these depend on?

Types of Error and their Dependencies

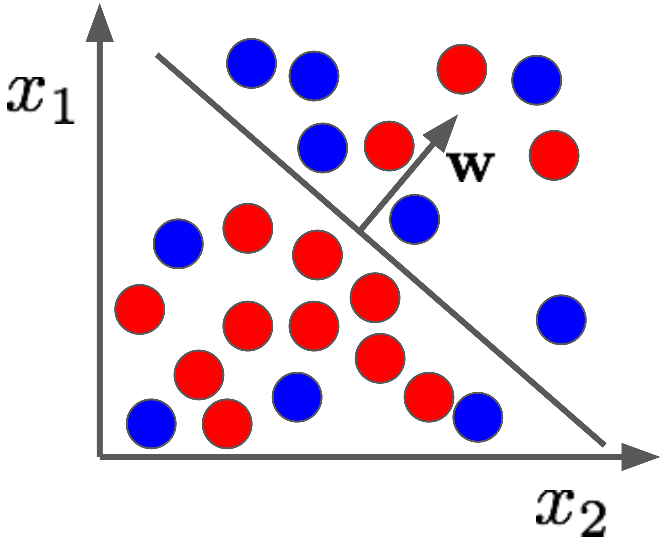
Approximation Error Depends on:

- Underlying distribution D
- Hypothesis class H

Estimation error Depends on:

- Underlying distribution D
- Hypothesis class H
- Sample Size

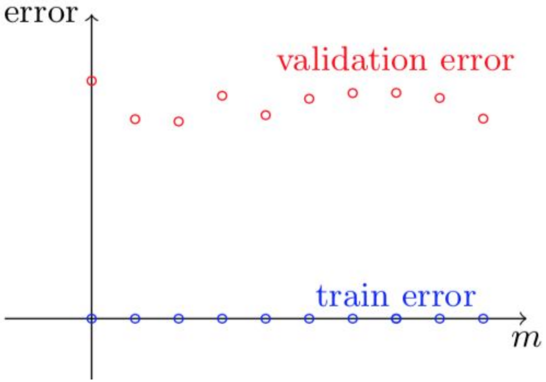
If Empirical Risk is Large...



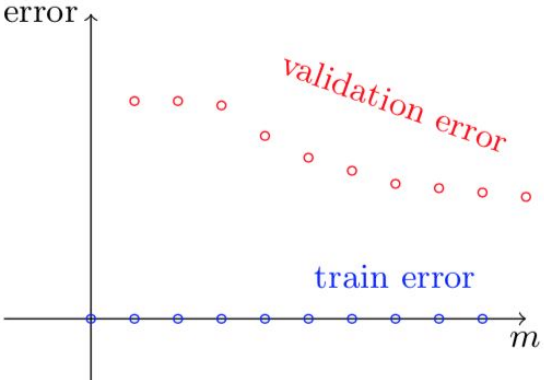
- If $L_S(h_S)$ is large, more training data won't help
- Approximation error is likely large!

If Empirical Risk is Small...

- Plot a learning curve
  - Train the algorithm on prefixes of the data of increasing sizes, and plot



Weak dependence on m
(approximation error)

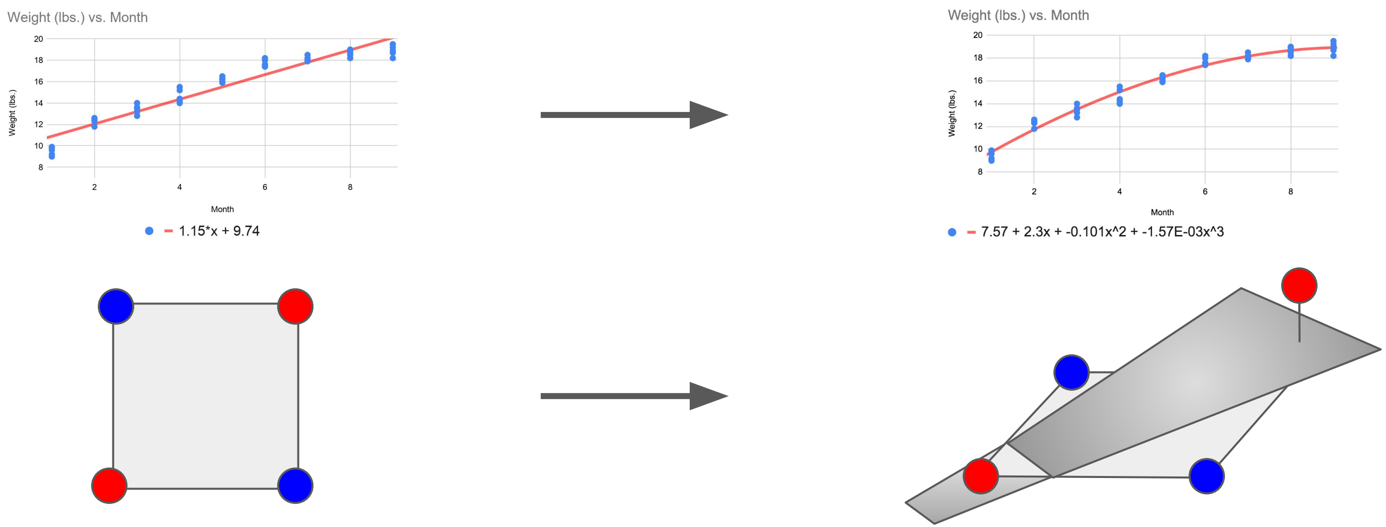


Strong dependence on m
(estimation error)

Understanding Machine Learning.
Shalev-Shwartz and Ben-David, 2014.

Reducing Approximation Error

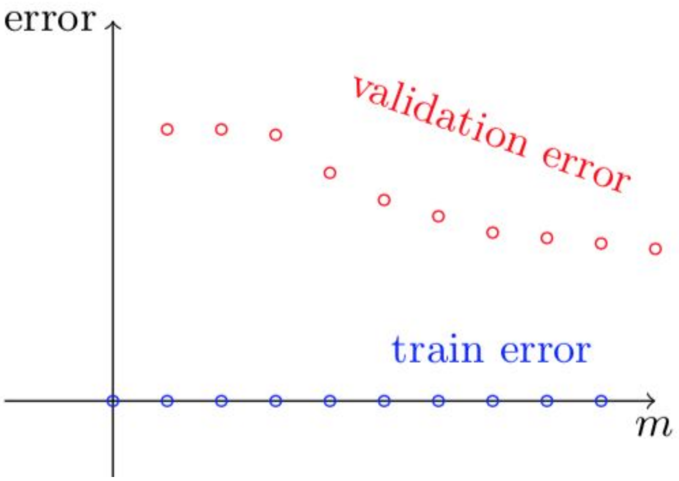
- Only thing we have control over is hypothesis class!
- Need to increase hypothesis class complexity



Reducing Estimation Error

- Collecting more training data could help
  - Not always practical!

- Reducing hypothesis class complexity could also help

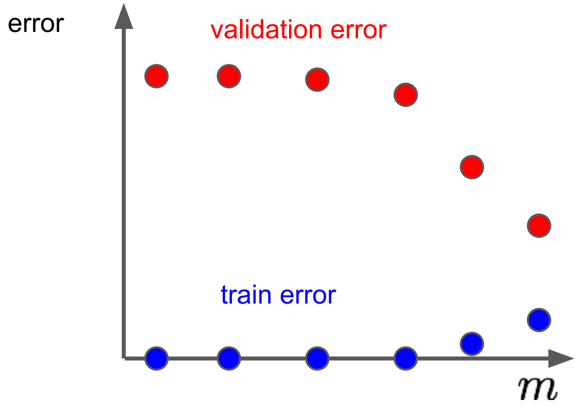


Strong dependence on m
(estimation error)

Question

What Should We Do?

Suppose we get a learning curve like this:



What should we try first to reduce error?

A: Collect more training data

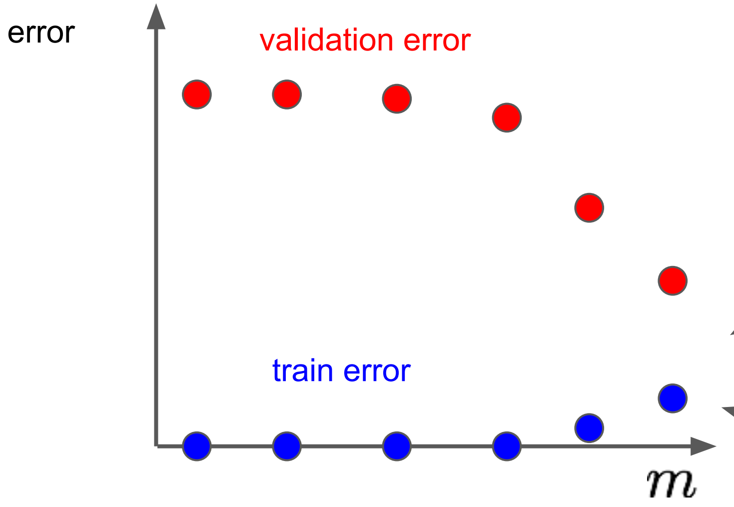
B: Decrease hypothesis complexity

C: Increase hypothesis complexity

D: None of the above

Answer

Answer: Increase complexity (C)



Extrapolating from curves suggests most error is approximation

Non-zero training error means approximation error is non-zero

Regularization

Fine-Tuning the Bias-Complexity Tradeoff

- So far, moving in the bias-complexity spectrum has required engineering new hypothesis classes

- What if we don't want to throw all of our hard work away? Can we keep our representation (training data and hypothesis class) and adjust the tradeoff?

Regularization

- A regularizer balances between empirical risk and simpler hypotheses:

$$R : \mathcal{H} \to \mathbb{R}$$

- Regularized loss minimization: combines both empirical risk and regularizer:

$$h_S \in \underset{h \in \mathcal{H}}{\arg\min} \; L_S(h) + R(h)$$

A Simple(?) Regularizer

- $h_w(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \ldots + w_k x^k$
- $R(w) = \lambda \max(\left\{k \quad \text{where } w_k \neq 0\right)$

- What does this mean in words?
  - Advantages?
  - Challenges?

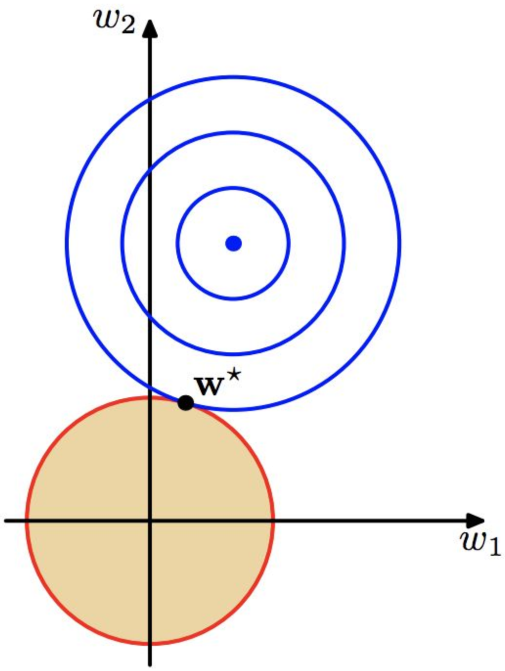
L2 Regularization

- A.K.A. Tikhonov regularization or weight decay

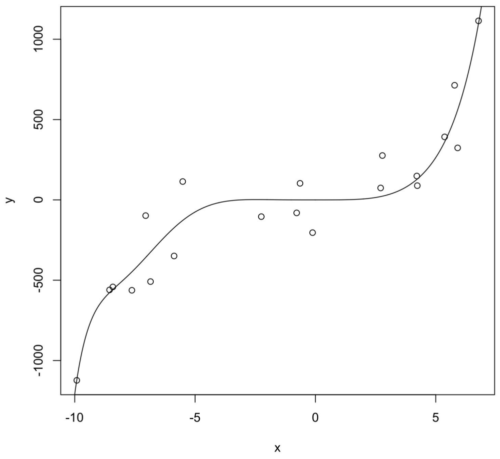
$$R(\mathbf{w}) = \lambda \|\mathbf{w}\|_2^2 \qquad \|\mathbf{w}\|_2^2 = \sum_{i=1}^{d} w_i^2$$

- ***Ridge regression*** = linear/polynomial regression + L2 regularization:

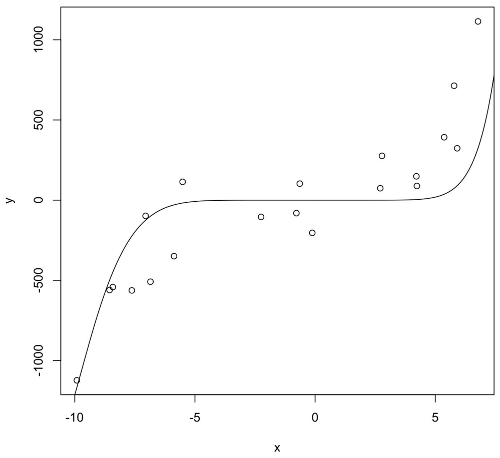
$$\underset{\mathbf{w}\in\mathbb{R}^d}{\operatorname{argmin}} \left(\lambda \|\mathbf{w}\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \right)$$



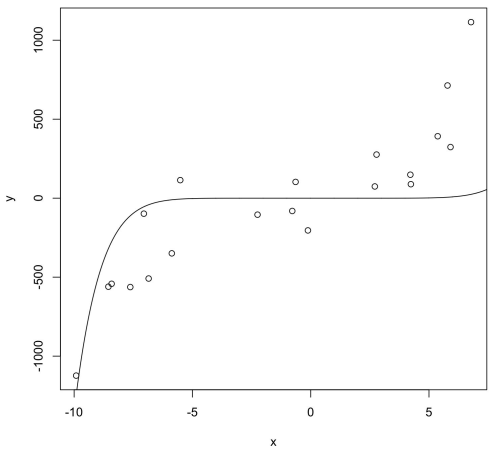
Ridge Regression Demo (degree=10)



$\lambda = 10^{-4}$



$\lambda = 10^{-1}$



$\lambda = 10^{2}$

ERM for Ridge Regression

- Gradient of the empirical risk is $(2\lambda m I + A)\mathbf{w} - \mathbf{b}$ where

$$A = \left(\sum_{i=1}^{m} \mathbf{x}_i \mathbf{x}_i^\top\right) \qquad \mathbf{b} = \sum_{i=1}^{m} y_i \mathbf{x}_i$$

- Setting equal to 0 and solving for $\mathbf{w}$ gives

$$\mathbf{w} = (2\lambda m I + A)^{-1}\mathbf{b}$$

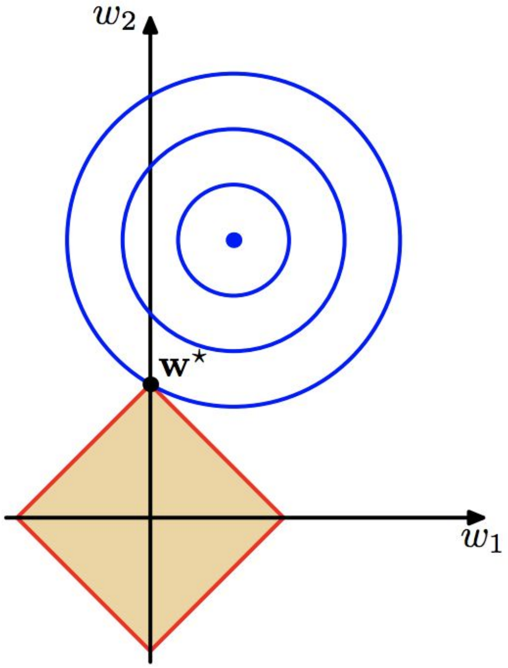
L1 Regularization

- Sparsity inducing regularizer:

$$R(\mathbf{w}) = \lambda \|\mathbf{w}\|_1 \qquad\qquad \|\mathbf{w}\|_1 = \sum_{i=1}^{d} |w_i|$$

- ***Lasso regression*** = linear/polynomial regression + L1 regularization:

$$\arg\min_{\mathbf{w} \in \mathbb{R}^d} \left(\lambda \|\mathbf{w}\|_1 + \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \right)$$



SGD for Regularized Losses

- Gradient is a linear operator so we just add the gradient of R to the usual one
- For example, to use L2 regularization for multiclass logistic regression:

$$\frac{\partial L_S(h_{\mathbf{w}}) + R(h_{\mathbf{w}})}{\partial w_{st}} = \frac{1}{m}\sum_{i=1}^{m}(h_{\mathbf{w}}(\mathbf{x}_i)_s - \mathbf{1}[y_i = s])x_{it} + 2\lambda w_{st}$$

- For L1 regularization for multiclass logistic regression:

$$\frac{\partial L_S(h_{\mathbf{w}}) + R(h_{\mathbf{w}})}{\partial w_{st}} = \frac{1}{m}\sum_{i=1}^{m}(h_{\mathbf{w}}(\mathbf{x}_i)_s - \mathbf{1}[y_i = s])x_{it} \pm \lambda$$

  - If a weight w_i is ever 0, treat the partial derivative of R as 0

Review

- A held-out ***validation set*** is a critical tool for model selection
- It helps assess where on the bias-complexity tradeoff a hypothesis is
- ***Regularizers*** like L2 regularization give us a knob λ to adjust bias-complexity tradeoff for a fixed hypothesis class
- Textbook: chapters 11.0, 11.2, 11.3, 13.0, 13.1, 13.4

Next Class

- New hypothesis classes: “boost” a fixed class into a more complex one
- Textbook: chapter 10