Model Selection, Validation, and Regularization

Lecture 10

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

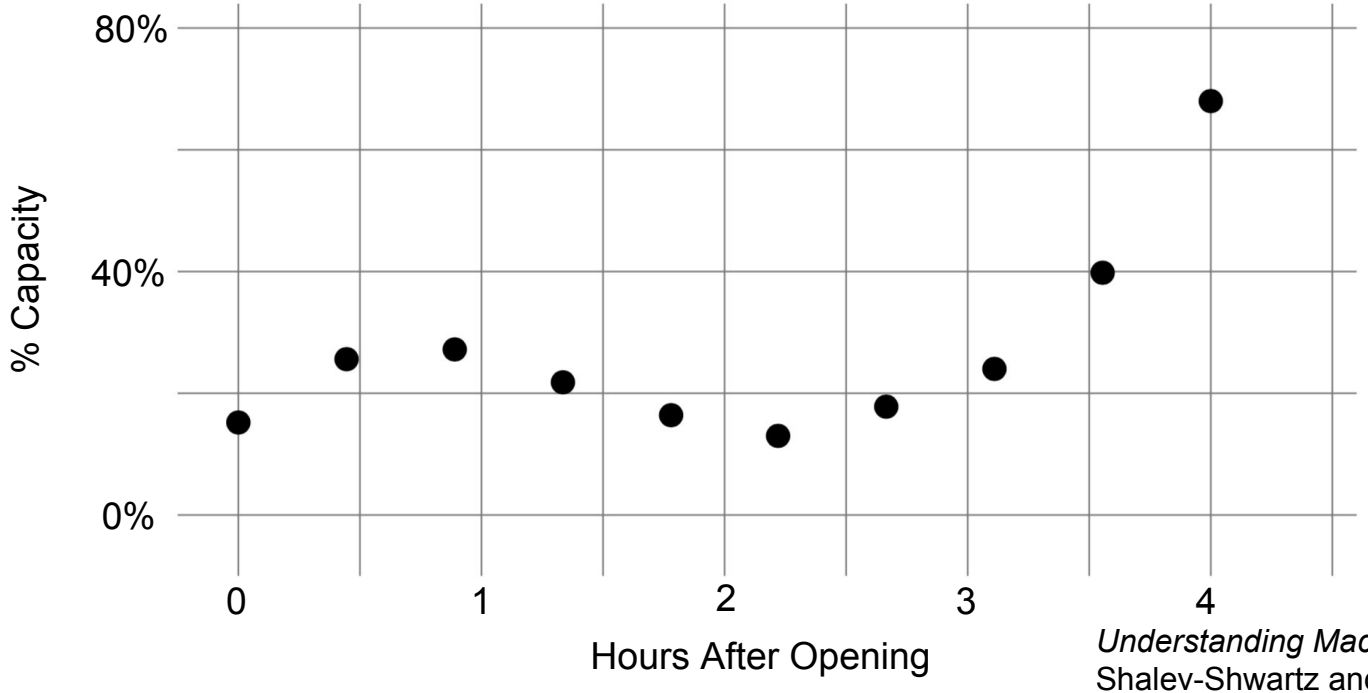
- Textbook: chapter 5

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

Motivating Example

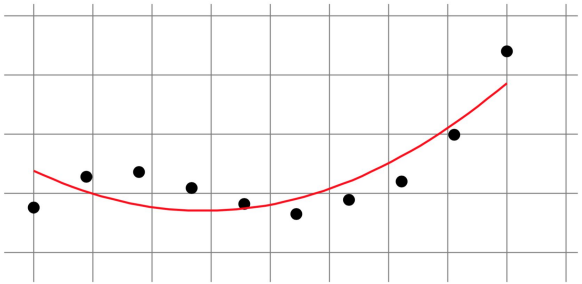
Let’s determine the popularity of Jo’s as a function of time:



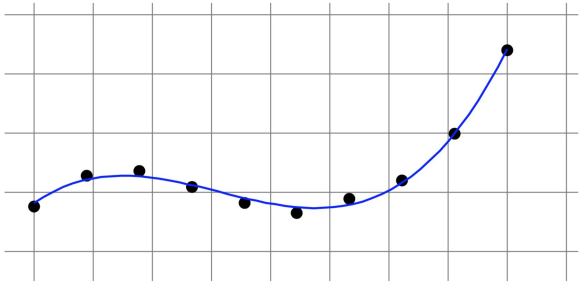
Understanding Machine Learning.
Shalev-Shwartz and Ben-David, 2014.

Let’s Model It

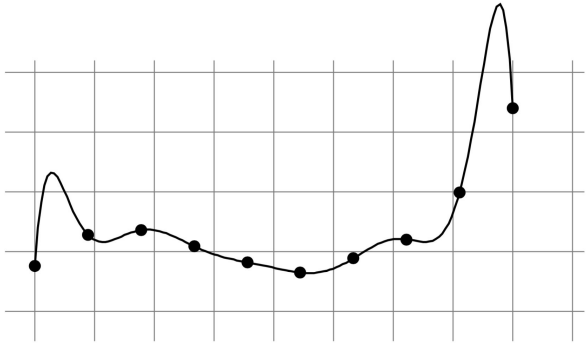
Polynomial regression of varying degrees:



Degree 2



Degree 3



Degree 10

Which one would you choose and why?

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

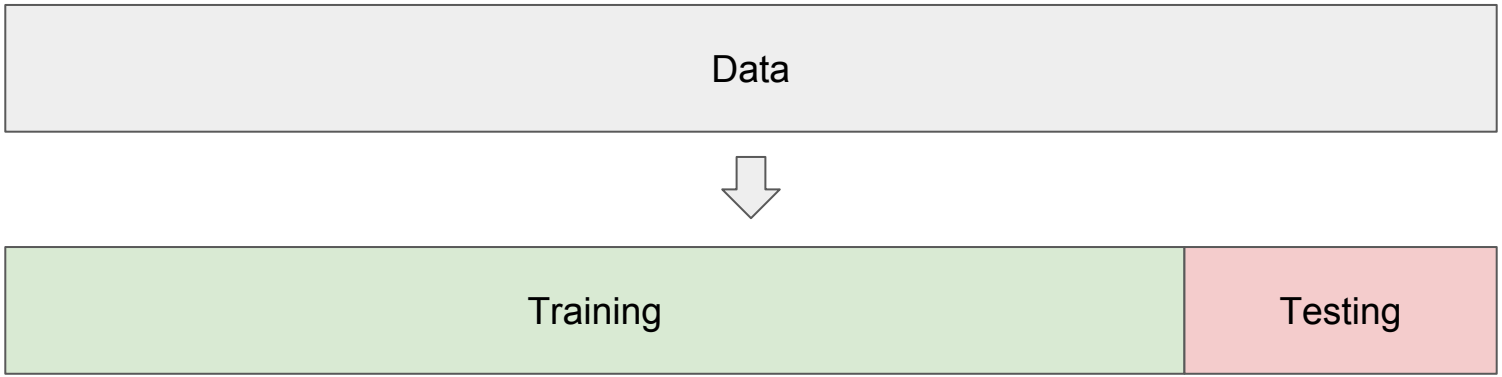
Model Selection and Validation

The Need for Validation

- As we increase polynomial order, we lower empirical risk
- But seems like overfitting!
- Q: How do we formalize this intuition and apply it to high-dimensional data?
- A: Find balance between approximation and estimation errors via validation

Previous Set Up

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

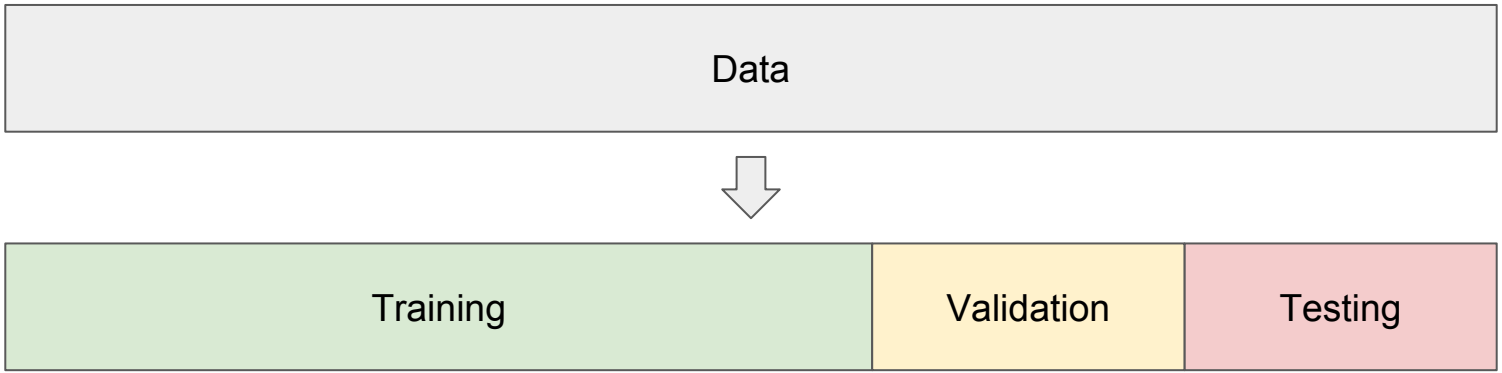


No Peeking!

- 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

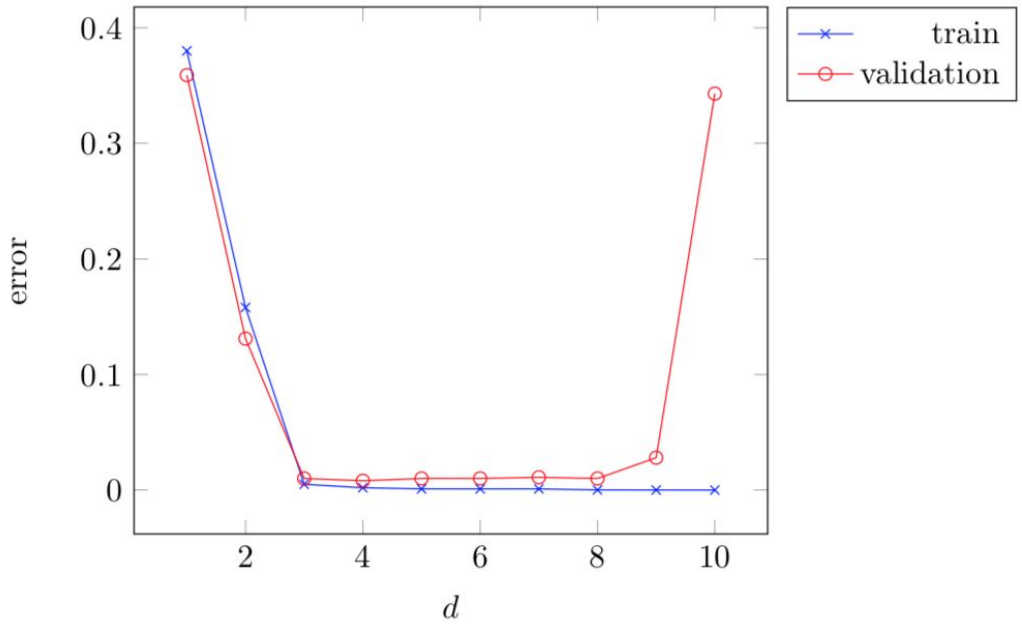
- Train different algorithms (or the same algorithm with different hyperparameters) on a given training set
- Now, to choose a single hypothesis from H we choose the one that minimizes the error over the validation set
- Error on the validation set approximates the true error

Model Selection Curves

Shows Training and Validation error as a function of complexity

Recall Jo's Example:

- Training error decreases monotonically
- Validation decreases then increases (Overfitting)



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

Bounding the Loss via Validation

Any hypothesis, maybe one from ERM

$\ell(h, (\mathbf{x}, y))$

THEOREM 11.1 *Let* h *be some predictor and assume that the loss function is in* $[0, 1]$. *Then, for every* $\delta \in (0, 1)$, *with probability of at least* $1 - \delta$ *over the choice of a validation set* V *of size* m_v *we have*

$$|L_V(h) - L_{\mathcal{D}}(h)| \leq \sqrt{\frac{\log(2/\delta)}{2\, m_v}}.$$

$$\frac{1}{m_v} \sum_{i=1}^{m_v} \ell(h, (\mathbf{x}_i^v, y_i^v))$$

Proof

- Recall Hoeffding's Inequality: $\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta-\mu\right|>\epsilon\right]\leq 2\exp\left(\frac{-2m\epsilon^2}{(b-a)^2}\right)$
- Define $\delta = 2\exp\left(-2m_v\epsilon^2\right)$
- Solve for ϵ: $\epsilon = \sqrt{\frac{\log(2/\delta)}{2m_v}}$
- Substitute ϵ and δ into Hoeffding's Inequality, where b = 1 and a = 0

Rearranging to Upper Bound on Loss

$$|L_V(h) - L_{\mathcal{D}}(h)| \leq \sqrt{\frac{\log(2/\delta)}{2m_v}}$$

implies

$$L_{\mathcal{D}}(h) \leq L_V(h) + \sqrt{\frac{\log(2/\delta)}{2m_v}}$$

Comparison with UC Upper Bound

Validation Upper Bound:

$$L_{\mathcal{D}}(h) \leq L_V(h) + \sqrt{\frac{\log(2/\delta)}{2m_v}}$$

Uniform Convergence Upper Bound

$$L_{\mathcal{D}}(h_S) \leq L_S(h_S) + \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2m}}$$

Question

What Went Wrong?

Steve plots a model selection curve for predicting the popularity of Andrew's. He considers polynomial regression with all maximum degrees 1 to 10 (inclusive). Using a validation set of size $m_v = 100$ he sees that degree 4 (h_4) has the best validation error of 0.1. Using the upper bound $L_{\mathcal{D}}(h) \leq L_V(h) + \sqrt{\frac{\log(2/\delta)}{2m_v}}$ he concludes

that with probability ≥95%, $L_{\mathcal{D}}(h_4) \leq L_V(h_4) + \sqrt{\frac{\log(2/0.05)}{200}} \leq 0.24$. However, when he (somehow magically) evaluates $L_{\mathcal{D}}(h_4)$, it is 0.26. What went wrong?

A: Nothing, it happens with <5% chance

B: Wrong value for δ

C: He didn't meet the bound's assumptions

D: Wrong value for m_v

Answer

Answer: He didn't meet the bound's assumptions (C)

- Tricky mistake!

- He evaluated $L_V(h)$ for all ten hypotheses in $\mathcal{H}_S$ (best of each kind on S)

- Just like the bound on the empirical risk minimizer, we have to account for how many hypotheses we evaluated on the validation data to pick h:

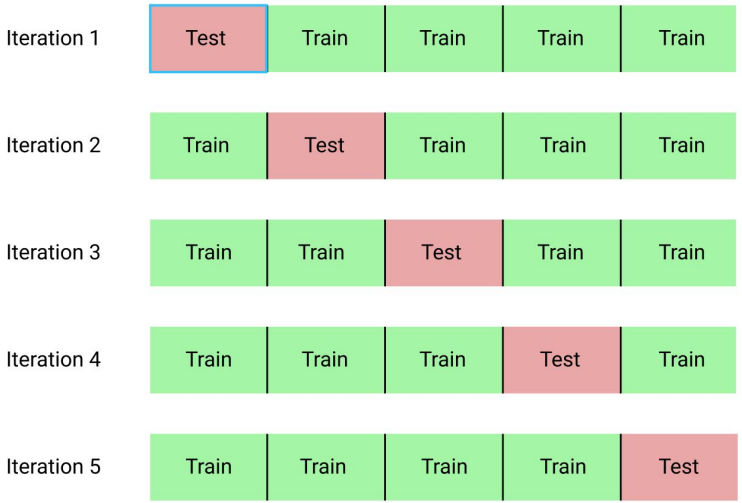
$$L_{\mathcal{D}}(h) \leq L_V(h) + \sqrt{\frac{\log|\mathcal{H}_S| + \log(2/\delta)}{2m_v}}$$

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



What if Learning Fails?

What if Learning Fails?

Plenty of options:

- Get a larger sample
- Change the hypothesis class by:
  - Enlarging it
  - Reducing it
  - Completely changing it
  - Changing the parameters you consider
- Change the feature representation of the data
- Change the optimization algorithm used to apply your learning rule

Need to smartly choose what is the issue: Approximation or Estimation error

Error Decomposition Using Validation

Using validation to see what is wrong (two types of 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*

Improving Approximation error:

- Increase size of *H* or change it
- Change featurization of data

Estimation error Depends on:

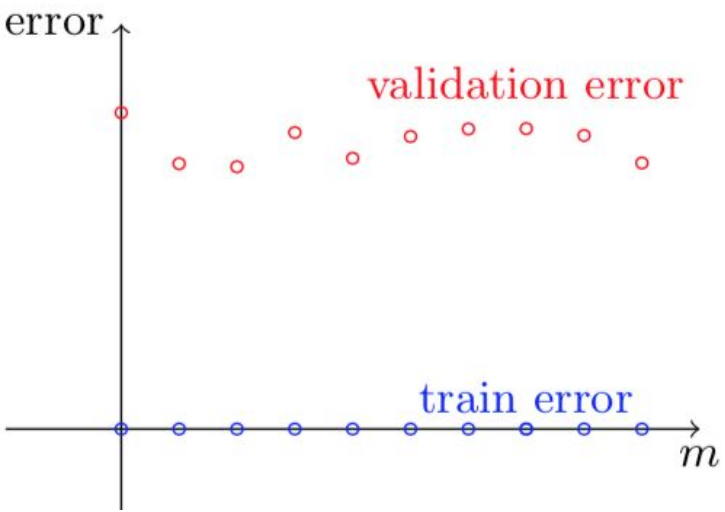
- Underlying distribution *D*
- Hypothesis class *H*
- Sample Size

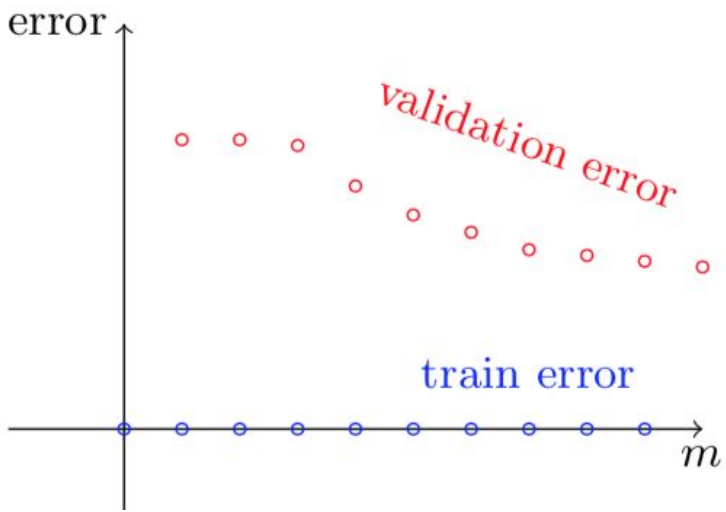
Improving Estimation error:

- Obtain more training samples
- Reduce *H*

Learning Curves

Train the algorithm on prefixes of the data of increasing sizes, and plot:





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

Learning Curves

- If approximation error is greater than 0 expect training error to grow and validation error to decrease as sample size increases

- If class is agnostic PAC learnable, they converge on the approximation error. This can be extrapolated from the curves as well.

Regularization

Fine-Tuning the Bias-Complexity Tradeoff

- Two types of error: approximation and estimation
- What tools do we have to adjust the spectrum?
  - Change the model, Change the representation
- 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:

$$R : \mathbb{R}^d \to \mathbb{R}$$

Regularized Loss Minimization: Combines both empirical risk and regularizer minimization:

$$\underset{w}{\operatorname{argmin}}(L_S(w) + R(w))$$

Simple(?) Regularizer

$$h_w(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \ldots + w_k x^k$$

$$R(w) = \lambda \max(\{k \quad \text{where } w_k \neq 0)$$

In words? Advantages? Challenges?

Tikhonov Regularization

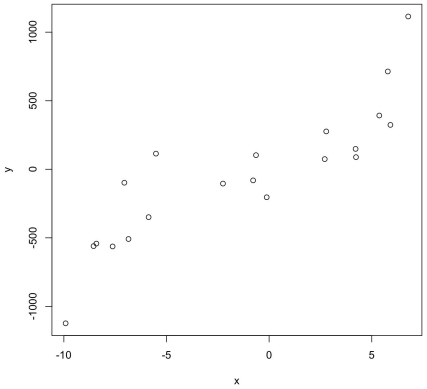
Also known as L2 regularization or weight decay

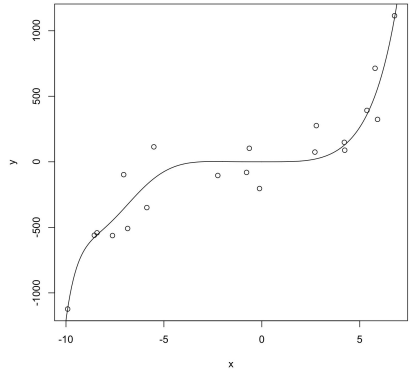
$$R(w) = \lambda ||w||_2^2 \qquad ||w||_2 = \sqrt{\sum_{i=1}^{d} w_i^2}$$

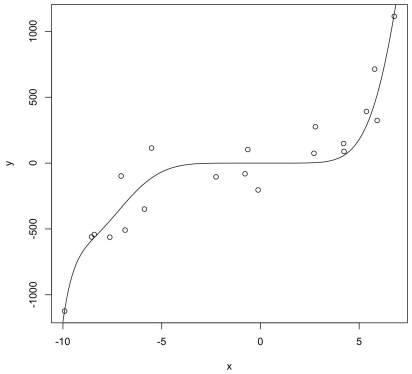
Ridge regression = linear/polynomial regression + Tikhonov regularization:

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

Ridge Regression Demo (degree=10)



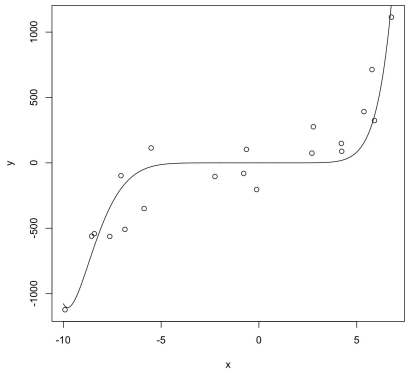


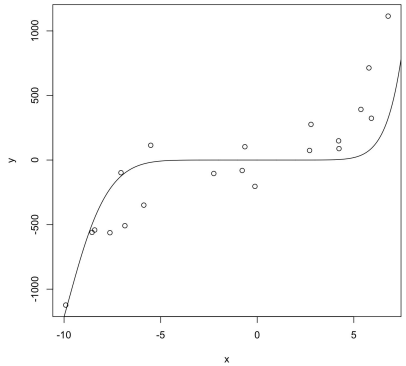


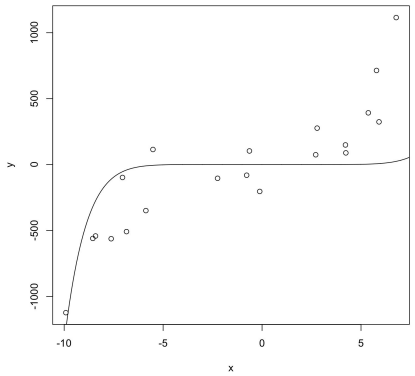
Regularization λ

10^-5

10^-4







10^-3

10^-2

10^-1

10^2: flat

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

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

Tikhonov Regularization for other Models

- We can add Tikhonov regularization to any risk function
- Gradient is a linear operator so we just add the gradient of R to the usual one
- For example, to use Tikhonov regularization for multiclass logistic regression:

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

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 Tikhonov 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

- Our final tool of learning theory: what makes a hypothesis class learnable? Can infinite hypothesis classes ever be learnable?

- Textbook: chapters 6, 9.1.3