Ridge Regression:

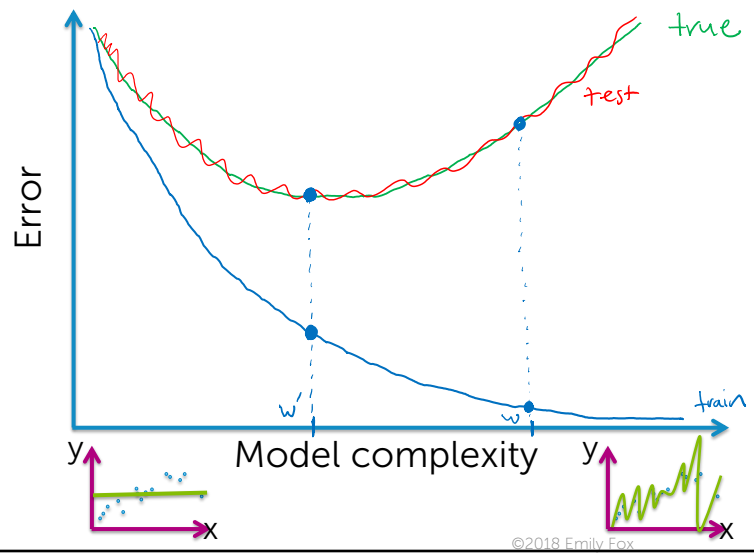
Regulating overfitting when using many features

CSE/STAT 416: Intro to Machine Learning

Emily Fox
University of Washington

April 10, 2018

Training, true, & test error vs. model complexity



test is a noisy approx of true error

Overfitting if:

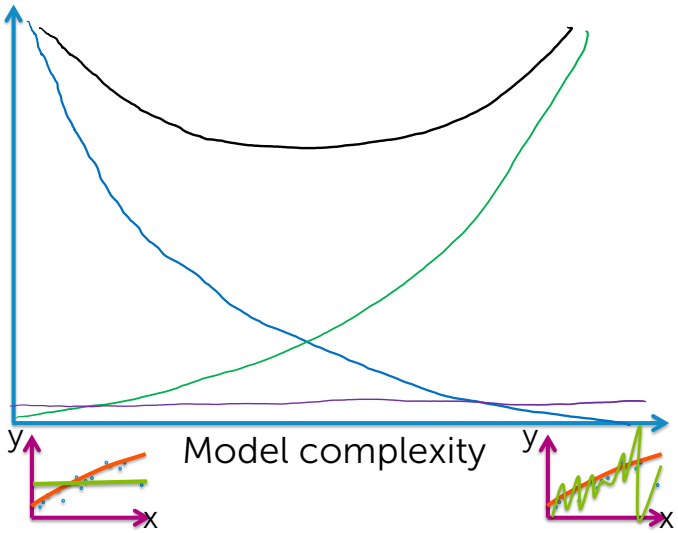
You learn a model w/ param w, but there exists a model w/ param w' such that

1) true error(w') < true error(w)

2) train error(w) < train error(w')

Bias-variance tradeoff

error = bias^2 + variance + noise



Simple Models

Bias: High

Variance: Low

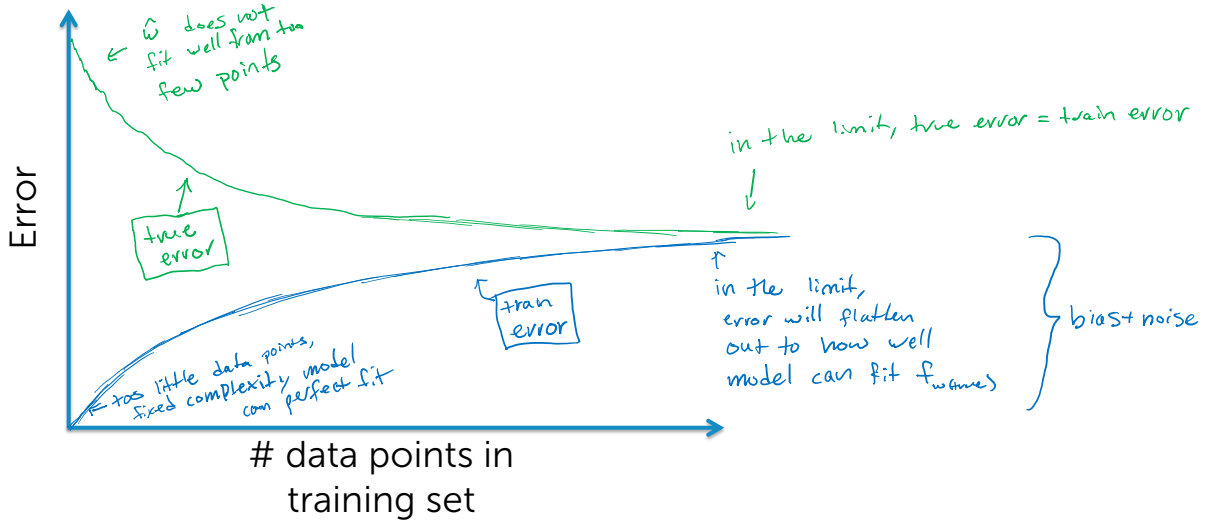
Complex Models

Bias: Low

Variance High

Error vs. amount of data

for fixed model complexity



Summary of assessing performance

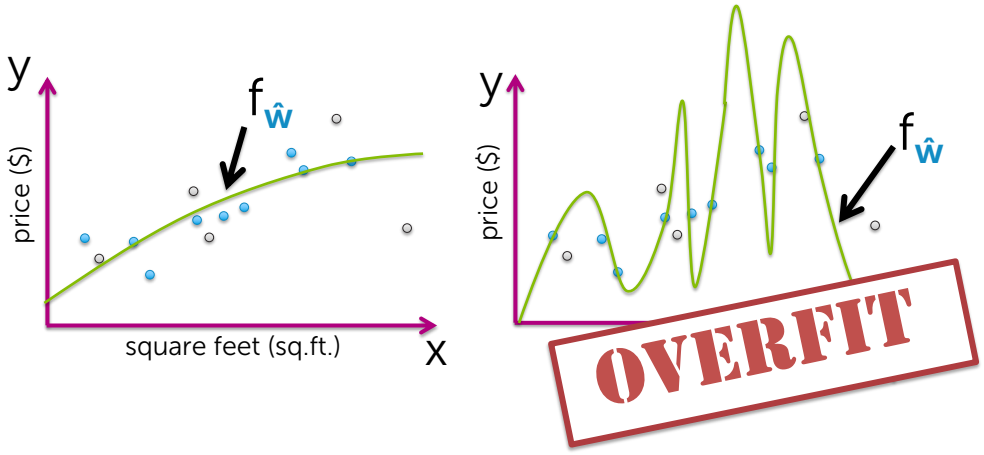
What you can do now…

- Describe what a loss function is and give examples
- Contrast training and test error
- Compute training and test error given a loss function
- Discuss issue of assessing performance on training set
- Describe tradeoffs in forming training/test splits
- List and interpret the 3 sources of avg. prediction error
  - Irreducible error, bias, and variance

Overfitting of polynomial regression

Flexibility of high-order polynomials

$$y_i = w_0 + w_1 x_i + w_2 x_i^2 + \dots + w_p x_i^p + \varepsilon_i$$



Symptom of overfitting

Often, overfitting associated with very large estimated parameters $\hat{\mathbf{w}}$

Overfitting of linear regression models more generically

Overfitting with many features

Not unique to polynomial regression,
but also if **lots of inputs** (d large)

- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- …

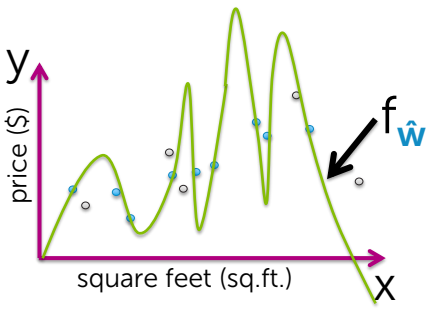
Or, generically,
lots of features (D large)

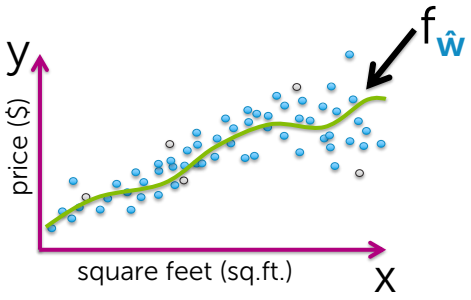
$$y_i = \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i) + \varepsilon_i$$

How does # of observations influence overfitting?

Few observations (N small)
→ rapidly overfit as model complexity increases

Many observations (N very large)
→ harder to overfit

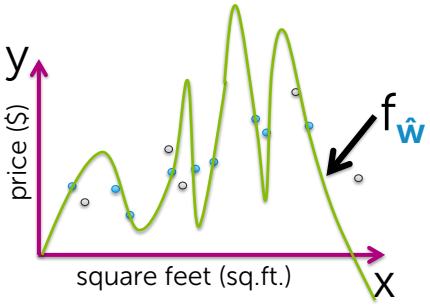




How does # of inputs influence overfitting?

1 input (e.g., sq.ft.):

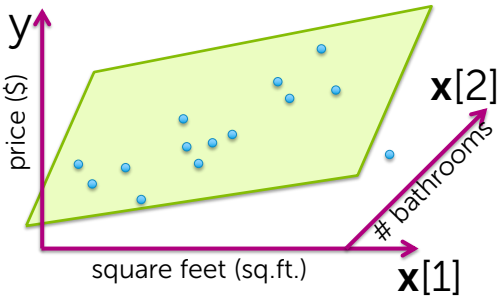
Data must include representative examples of all possible (sq.ft., $) pairs to avoid overfitting



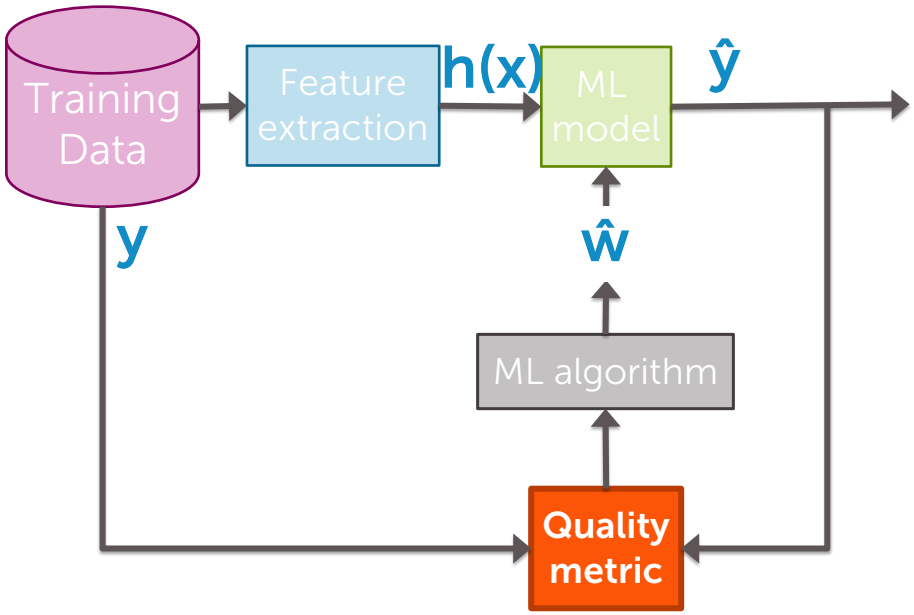
How does # of inputs influence overfitting?

d inputs (e.g., sq.ft., #bath, #bed, lot size, year,...):

Data must include examples of all possible (sq.ft., #bath, #bed, lot size, year,...., $) combos to avoid overfitting



Adding term to cost-of-fit to prefer small coefficients



Desired total cost format

Want to balance:

i. How well function fits data

ii. Magnitude of coefficients

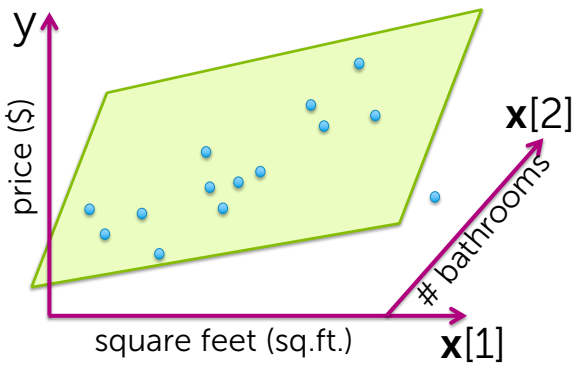
Total cost = measure quality of fit + measure of magnitude of coefficients

want to balance

measure of fit: small # = good fit to training data

measure of magnitude of coefficients: small # = not overfit

Measure of fit to training data



$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - h(\mathbf{x}_i)^T \mathbf{w})^2$$

$$= \sum_{i=1}^{N} (y_i - \hat{y}_i(\mathbf{w}))^2$$

small RSS → model fitting training data well

Measure of magnitude of regression coefficient

What summary # is indicative of size of regression coefficients?

- Sum? $w_0 = 1,527,301$ $w_1 = -1,605,253$ $w_0 + w_1 =$ small # ✗
- Sum of absolute value? $\sum_{j=0}^{D} |w_j| \triangleq \|\mathbf{w}\|_1$ ← L_1 norm… discuss next lecture
- Sum of squares (L_2 norm) $\sum_{j=0}^{D} w_j^2 \triangleq \|\mathbf{w}\|_2^2$ ← focus of this lecture

$\mathbf{w} = [w_0\ w_1\ \cdots\ w_D]$

Consider specific total cost

Total cost =
measure of fit + measure of magnitude of coefficients

Consider specific total cost

Total cost =
measure of fit + measure of magnitude of coefficients

RSS($\mathbf{w}$) $\qquad$ $\|\mathbf{w}\|_2^2$

Consider resulting objective

What if $\hat{\mathbf{w}}$ selected to minimize

$$\text{RSS}(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$

tuning parameter = balance of fit and magnitude

If λ=0:
reduces to min RSS(w), as before (old soln)
$\longrightarrow \hat{w}^{LS}$ (least squares)

If λ=∞:
For solns $\hat{w} \neq 0$, then total cost = ∞
If $\hat{w} = 0$, then total cost = RSS(0) $\longrightarrow \hat{w} = 0$

If λ in between:
Then $0 \leq \|\hat{w}\|_2^2 \leq \|\hat{w}^{LS}\|_2^2$

Consider resulting objective

What if $\hat{\mathbf{w}}$ selected to minimize

$$\text{RSS}(\mathbf{w}) + \lambda ||\mathbf{w}||_2^2$$

tuning parameter = balance of fit and magnitude

Ridge regression
(a.k.a L_2 regularization)

Bias-variance tradeoff

Large λ:

high bias, low variance

(e.g., $\hat{\mathbf{w}}$ =0 for λ=∞)

Small λ:

low bias, high variance

(e.g., standard least squares (RSS) fit of high-order polynomial for λ=0)

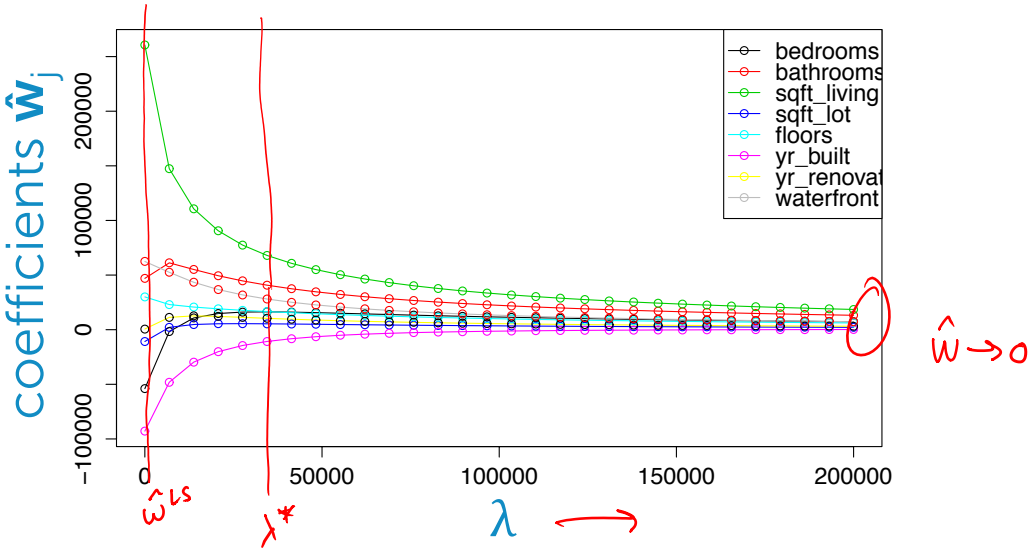
In essence, λ controls model complexity

Revisit polynomial fit demo

What happens if we refit our high-order polynomial, but now using **ridge regression**?

Will consider a few settings of λ …

Coefficient path



How to choose λ

The regression/ML workflow

1. **Model selection**
   Need to choose tuning parameters λ controlling model complexity

2. **Model assessment**
   Having selected a model, assess generalization error

Hypothetical implementation

1. **Model selection**

For each considered λ :

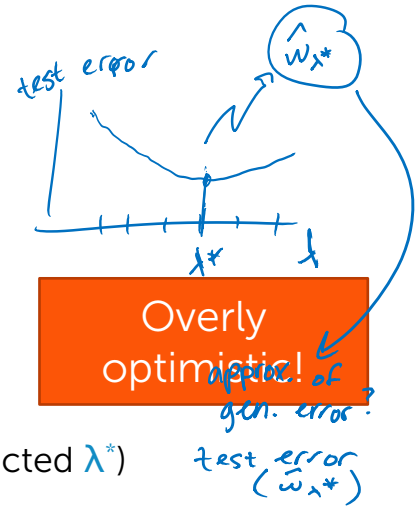
i. Estimate parameters $\hat{w}_\lambda$ on **training data**

ii. Assess performance of $\hat{w}_\lambda$ on **test data**

iii. Choose λ^* to be λ with lowest test error

2. **Model assessment**

Compute test error of $\hat{w}_{\lambda^*}$ (fitted model for selected λ^*) to approx. true error



Hypothetical implementation

Issue: Just like fitting $\hat{w}$ and assessing its performance both on training data

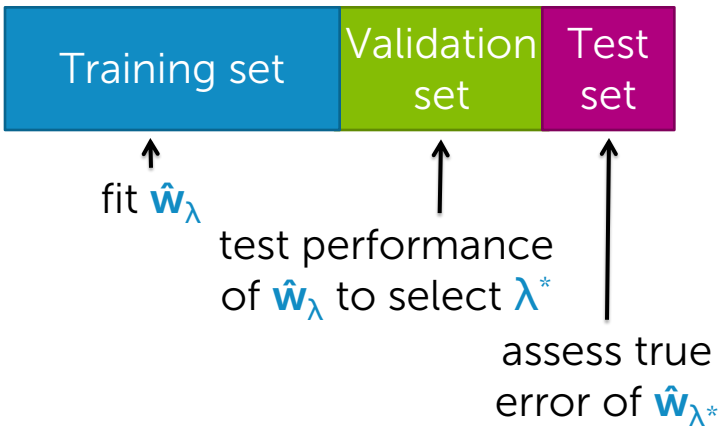
- λ^* was selected to minimize test error (i.e., λ^* was fit on test data)
- If test data is not representative of the whole world, then $\hat{w}_{\lambda^*}$ will typically perform worse than test error indicates

Practical implementation

Solution: Create two "test" sets!

1. Select λ^* such that $\hat{w}_{\lambda^*}$ minimizes error on validation set
2. Approximate true error of $\hat{w}_{\lambda^*}$ using test set

Practical implementation



Typical splits

80% 10% 10%

50% 25% 25%

How to handle the intercept

Recall multiple regression model

Model:

$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \ldots + w_D h_D(\mathbf{x}_i) + \varepsilon_i$

$= \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i) + \varepsilon_i$

feature 1 = $h_0(\mathbf{x})$… often 1 (constant)

feature 2 = $h_1(\mathbf{x})$… e.g., $\mathbf{x}[1]$

feature 3 = $h_2(\mathbf{x})$… e.g., $\mathbf{x}[2]$

…

feature D+1 = $h_D(\mathbf{x})$… e.g., $\mathbf{x}[d]$

Assume our 1st feature is 1 (include intercept)

Do we penalize intercept?

Standard ridge regression cost:

$$\text{RSS}(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$

λ: strength of penalty

$\|\mathbf{w}\|_2^2$: intercept $[w_0\ w_1\ \ldots\ w_D]$

Encourages intercept w_0 to also be small

Do we want a small intercept?

Conceptually, not indicative of overfitting…

Option 1: Don't penalize intercept

Modified ridge regression cost:

$$\text{RSS}(w_0, \mathbf{w}_{\text{rest}}) + \lambda ||\mathbf{w}_{\text{rest}}||_2^2$$

$\{w_1 \dots w_D\}$

Option 2: Center data first

If data are first centered about 0, then favoring small intercept not so worrisome

Step 1: Transform y to have 0 mean

Step 2: Run ridge regression as normal
(closed-form or gradient algorithms)

Feature normalization

Normalizing features

Scale training **columns** (not rows!) as:

$$\underline{h_j}(\mathbf{x}_k) = \frac{h_j(\mathbf{x}_k)}{\sqrt{\sum_{i=1}^{N} h_j(\mathbf{x}_i)^2}}$$

Normalizer: Z_j

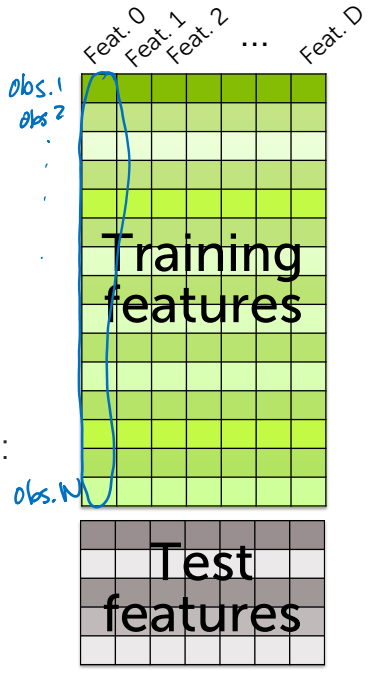
Apply same training scale factors to test data:

$$\underline{h_j}(\mathbf{x}_k) = \frac{h_j(\mathbf{x}_k)}{\sqrt{\sum_{i=1}^{N} h_j(\mathbf{x}_i)^2}}$$

Normalizer: Z_j

apply to test point

summing over training points



Summary for ridge regression

What you can do now…

- Describe what happens to magnitude of estimated coefficients when model is overfit
- Motivate form of ridge regression cost function
- Describe what happens to estimated coefficients of ridge regression as tuning parameter λ is varied
- Interpret coefficient path plot
- Use a validation set to select the ridge regression tuning parameter λ
- Handle intercept and scale of features with care

