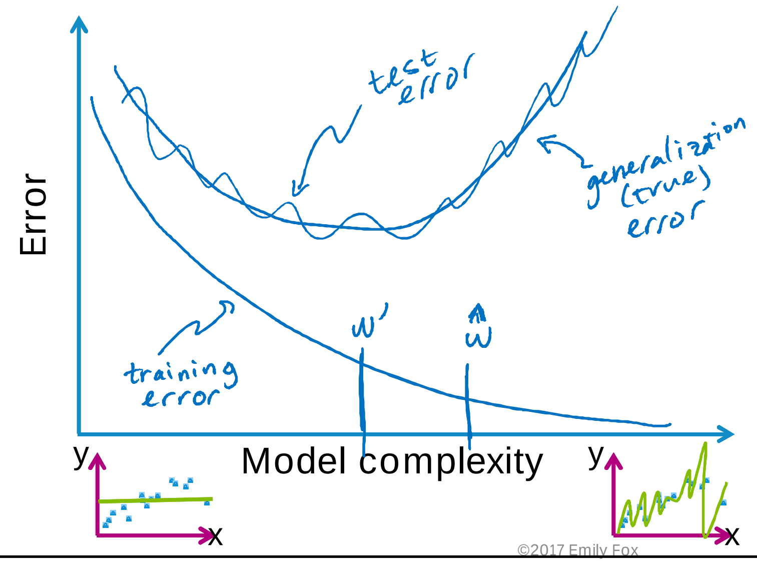
Ridge Regression:

Regulating overfitting when using many features

CSE 446: Machine Learning
Emily Fox
University of Washington
January 13, 2017

Training, true, & test error vs. model complexity

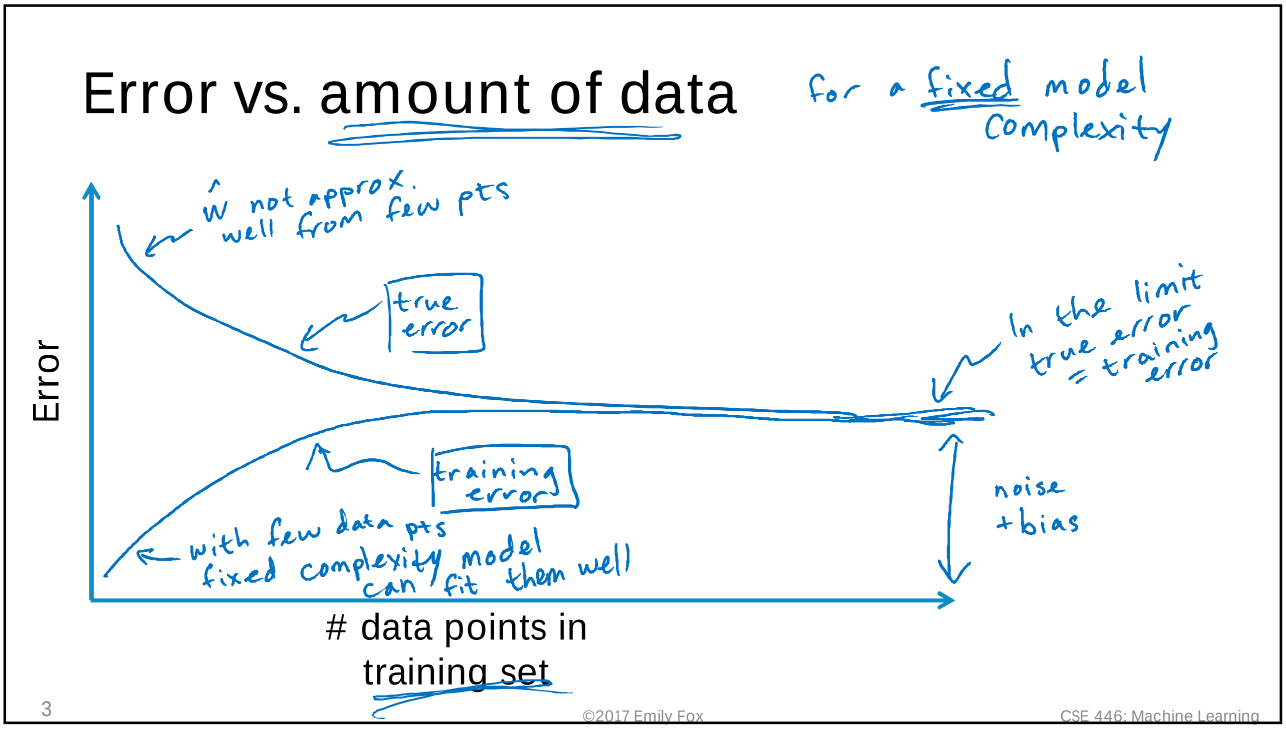


Overfitting if:

If there exists a model with estimated params w' such that

① training error $(\hat{w})$ < training error (w')

② true error $(\hat{w})$ > true error (w')



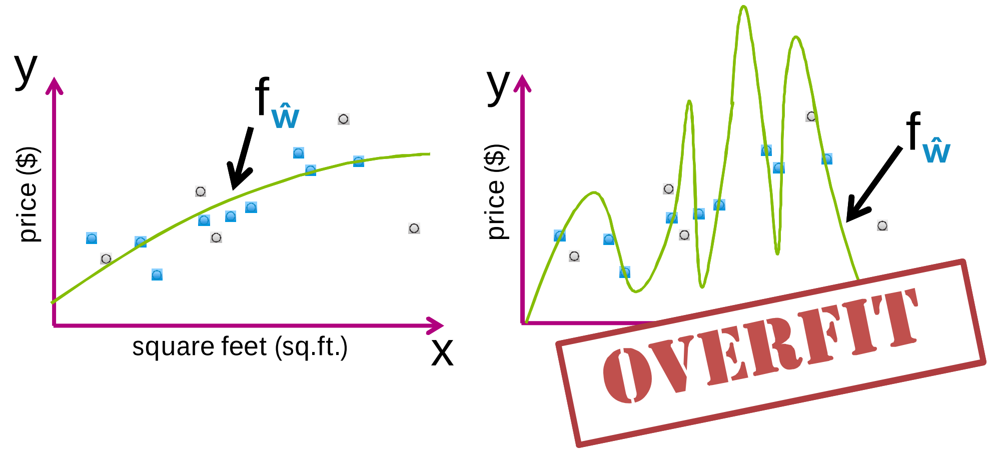
Error vs. amount of data

for a fixed model complexity

Overfitting of polynomial regression

Flexibility of high-order polynomials

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



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)

Or, generically, lots of features (D large)

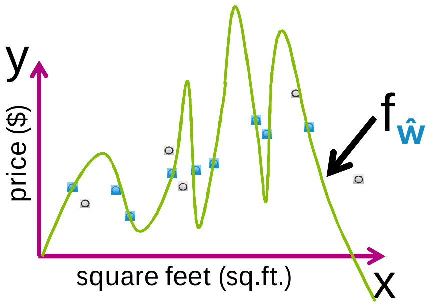
$$y_i = \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i) + \varepsilon_i$$

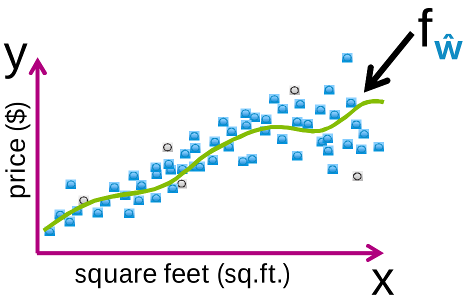
- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- ...

How does # of observations influence overfitting?

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

Many observations (N very large)
→ harder to overfit



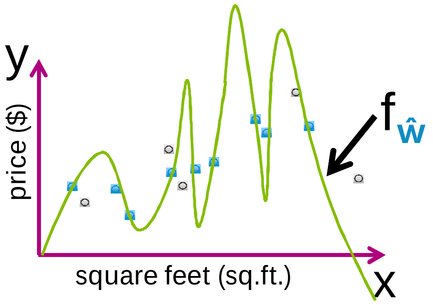


How does # of inputs influence overfitting?

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

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

HARD

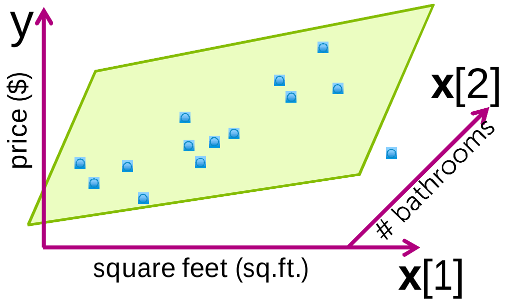


How does # of inputs influence overfitting?

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

MUCH!!! HARDER



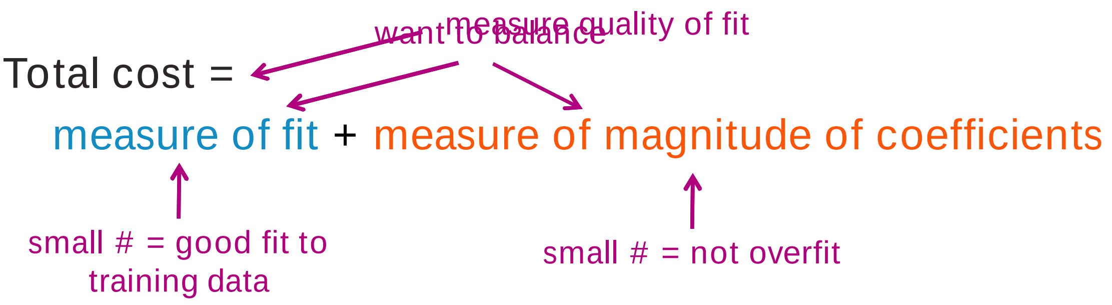
Adding term to cost-of-fit to prefer small coefficients

Desired total cost format

Want to balance:

i. How well function fits data

ii. Magnitude of coefficients



want to balance

measure quality of fit

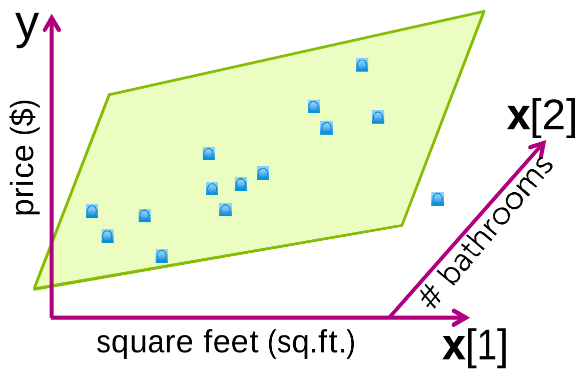
Total cost =

measure of fit + measure of magnitude of coefficients

small # = good fit to training data

small # = not overfit

Measure of fit to training data



pred. value using w

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

small RSS → model fitting training data well

Measure of magnitude of regression coefficient

What summary # is indicative of size of regression coefficients?

- Sum? $w_0 = 1{,}527{,}301$ $w_1 = -1{,}605{,}253$ X

  $w_0 + w_1 =$ small #

- Sum of absolute value? $\sum_{j=0}^{D} |w_j| \triangleq \|w\|_1$

  L_1 norm... discuss next lecture

- Sum of squares (L_2 norm) $\sum_{j=0}^{D} w_j^2 \triangleq \|w\|_2^2$

  L_2 norm... focus of this lecture

Consider specific total cost

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

Consider specific total cost

Total cost =

$$\underbrace{\text{measure of fit}}_{\text{RSS}(\mathbf{w})} + \underbrace{\text{measure of magnitude of coefficients}}_{\|\mathbf{w}\|_2^2}$$

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 $\lambda=0$:

reduces to min $RSS(w)$, as before (old soln)

$\rightarrow \hat{w}^{LS}$ (least squares)

If $\lambda=\infty$:

For solns where $\hat{w} \neq 0$, then total cost $= \infty$

If $\hat{w} = 0$, then total cost $= RSS(0)$ $\longrightarrow \hat{w} = 0$

If λ in between:

Then $0 \le \|\hat{w}\|_2^2 \le \|\hat{w}^{LS}\|_2^2$

Consider resulting objective

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

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

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 $\lambda=\infty$)

Small λ:

low bias, high variance

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

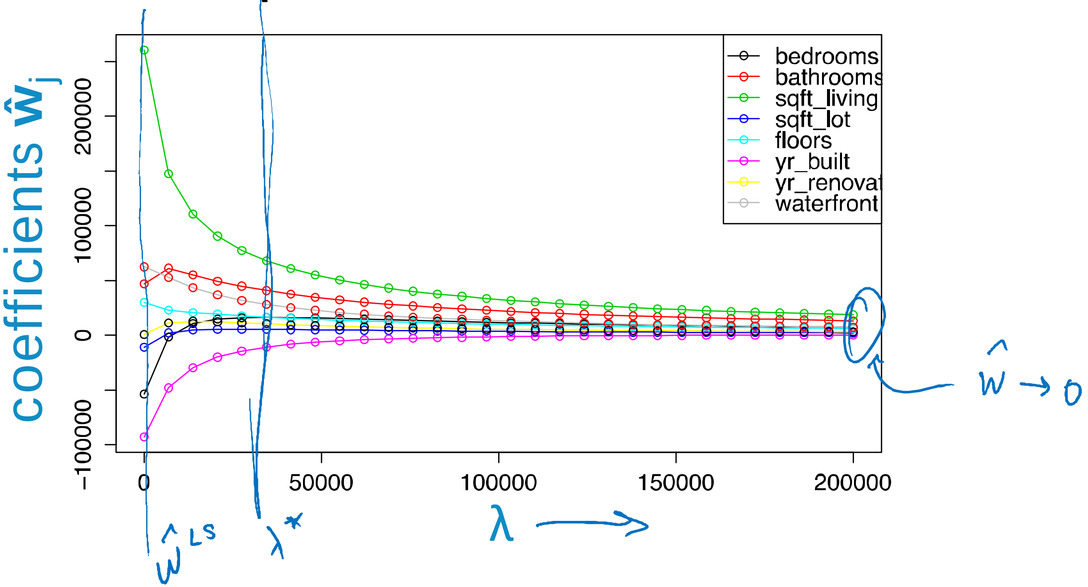
In essence, λ controls model complexity

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

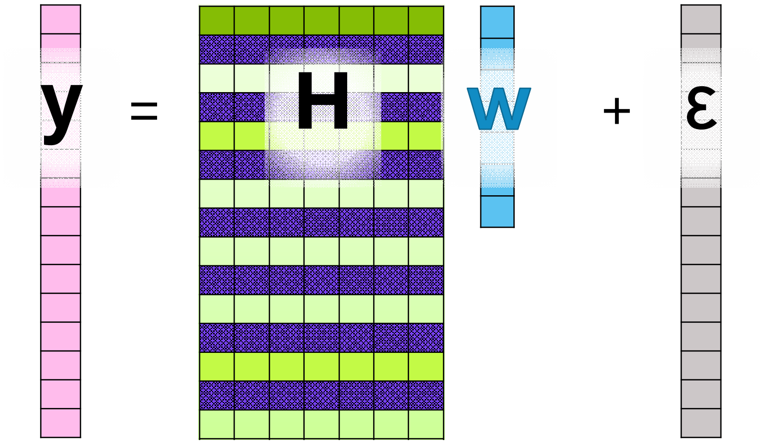


Fitting the ridge regression model (for given λ value)

Step 1:
Rewrite total cost in matrix notation

Recall matrix form of RSS

Model for all N observations together



Recall matrix form of RSS

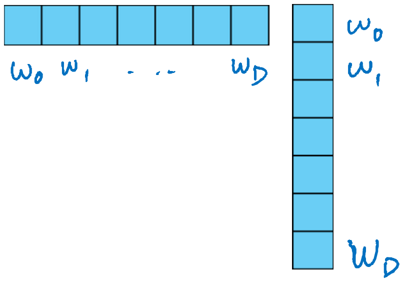
$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - h(\mathbf{x}_i)^T \mathbf{w})^2$$

$$= (\mathbf{y}-\mathbf{H}\mathbf{w})^T(\mathbf{y}-\mathbf{H}\mathbf{w})$$

Rewrite magnitude of coefficients in vector notation

$$||\mathbf{w}||_2^2 = w_0^2 + w_1^2 + w_2^{\;2} + \ldots + w_D^2$$

=



$$= \mathbf{w}^T \mathbf{w}$$

Putting it all together

In matrix form, ridge regression cost is:

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

$$= (\mathbf{y}-\mathbf{H}\mathbf{w})^T(\mathbf{y}-\mathbf{H}\mathbf{w}) + \lambda\mathbf{w}^T\mathbf{w}$$

Step 2:
Compute the gradient

Gradient of ridge regression cost

$\|w\|_2$

$= \sqrt{w_1^2+\dots+w_p^2}$

$= \sqrt{w^T w}$

$$\nabla[\text{RSS}(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2] = \nabla[(\mathbf{y}-\mathbf{H}\mathbf{w})^T(\mathbf{y}-\mathbf{H}\mathbf{w}) + \lambda\mathbf{w}^T\mathbf{w}]$$

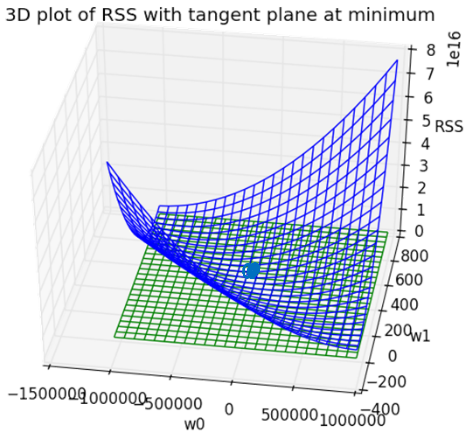
$$= \underbrace{\nabla[(\mathbf{y}-\mathbf{H}\mathbf{w})^T(\mathbf{y}-\mathbf{H}\mathbf{w})]}_{-2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w})} + \lambda\underbrace{\nabla[\mathbf{w}^T\mathbf{w}]}_{2\mathbf{w}}$$

Why? By analogy to 1d case…

$\mathbf{w}^T\mathbf{w}$ analogous to w^2 and derivative of $w^2=2w$

Step 3, Approach 1:
Set the gradient = 0

Ridge closed-form solution



add

$$\nabla \text{cost}(\mathbf{w}) = -\cancel{2}\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w}) + \cancel{2}\lambda\mathbf{I}\mathbf{w} = 0$$

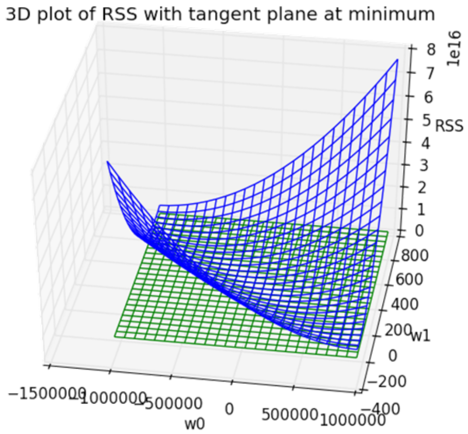
Solve for **w**:

$$-H^Ty + \underline{H^THw}\hat{} + \underline{\lambda I}\,\hat{w} = 0$$

$$(H^TH+\lambda I)\hat{w} = H^Ty$$

$$\hat{w}^{ridge} = (H^TH+\lambda I)^{-1}H^Ty$$

Interpreting ridge closed-form solution



$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y}$$

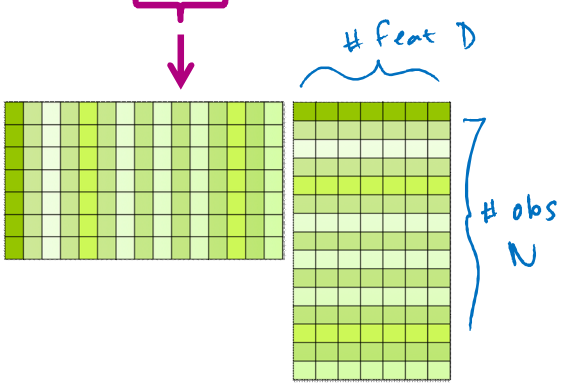
If $\lambda=0$: $\hat{w}^{ridge} = (H^T H)^{-1} H^T y = \hat{w}^{LS}$ ✓

↗ old soln

If $\lambda=\infty$: $\hat{w}^{ridge} = 0$ ✓ ← because it's like dividing by ∞

Recall discussion on previous closed-form solution

$$\hat{\mathbf{w}} = (\underbrace{\mathbf{H}^T\mathbf{H}})^{-1}\mathbf{H}^T\mathbf{y}$$



Invertible if:

In general,

(# linearly independent obs)

N > D

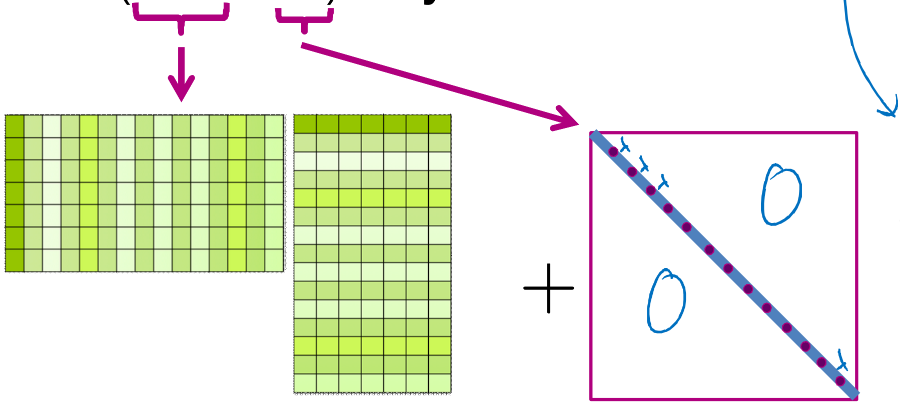
Complexity of inverse:

$O(D^3)$

Discussion of ridge closed-form solution

λI is making $H^T H + \lambda I$ more "regular"
→ "regularization"

$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y}$$



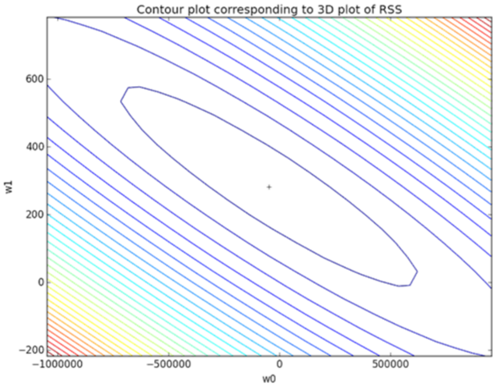
Invertible if:
Always if λ>0, even if N < D

Complexity of inverse:
$O(D^3)$…
big for large D!

Step 3, Approach 2: Gradient descent

Elementwise ridge regression gradient descent algorithm

$\nabla \text{cost}(\mathbf{w}) = -2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w}) + 2\lambda\mathbf{w}$



Update to jth feature weight:

$$w_j^{(t+1)} \leftarrow w_j^{(t)} - \eta \, *$$

as before

$$\left[-2\sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))\right.$$

new term

$$\left. + 2\lambda w_j^{(t)}\right]$$

Recall previous algorithm

```
init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), $t=1$
while $||\nabla \text{RSS}(\mathbf{w}^{(t)})|| > \varepsilon$
    for $j=0,\ldots,D$
        partial[j] $= -2\sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$
        $w_j^{(t+1)} \leftarrow w_j^{(t)} - \eta$ partial[j]
    $t \leftarrow t + 1$
```

Summary of ridge regression algorithm

```
init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), $t=1$
**while** $||\nabla RSS(\mathbf{w}^{(t)})|| > \varepsilon$
    **for** $j=0,\ldots,D$
        $\text{partial}[j] = -2\sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i-\hat{y}_i(\mathbf{w}^{(t)}))$
        $w_j^{(t+1)} \leftarrow (1-2\eta\lambda)w_j^{(t)} - \eta\ \text{partial}[j]$
    $t \leftarrow t + 1$
```

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

Training set
Test set

1. Model selection

For each considered λ :

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

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

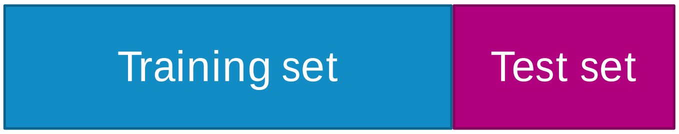
iii. Choose λ^* to be λ with lowest test error

Overly optimistic!

2. Model assessment

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

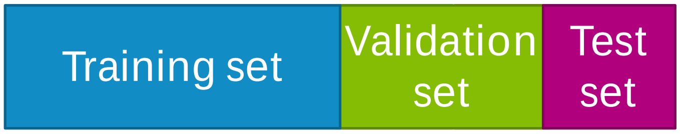
Hypothetical implementation



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

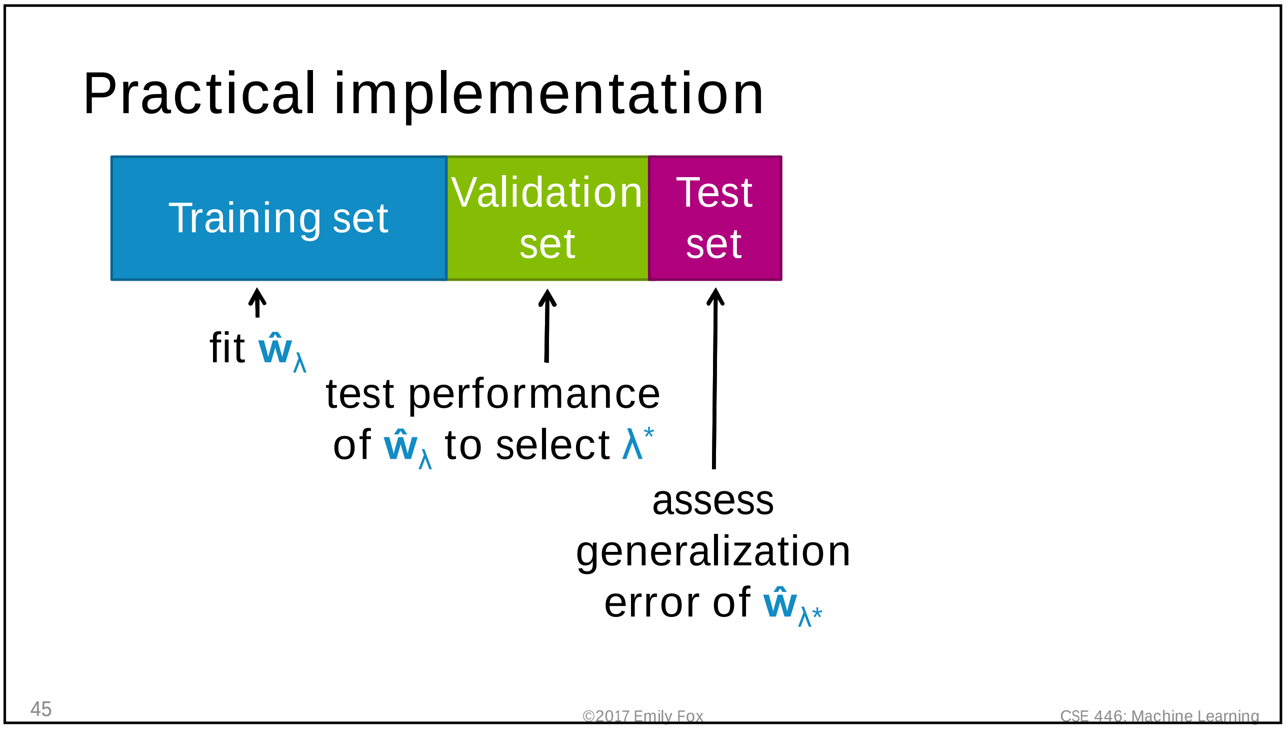
- λ^* was selected to minimize test error (i.e., λ^* was fit on test data)
- If test data is not representative of the whole world, then $\hat{\mathbf{w}}_{\lambda^*}$ will typically perform worse than test error indicates

Practical implementation



Solution: Create two “test” sets!

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

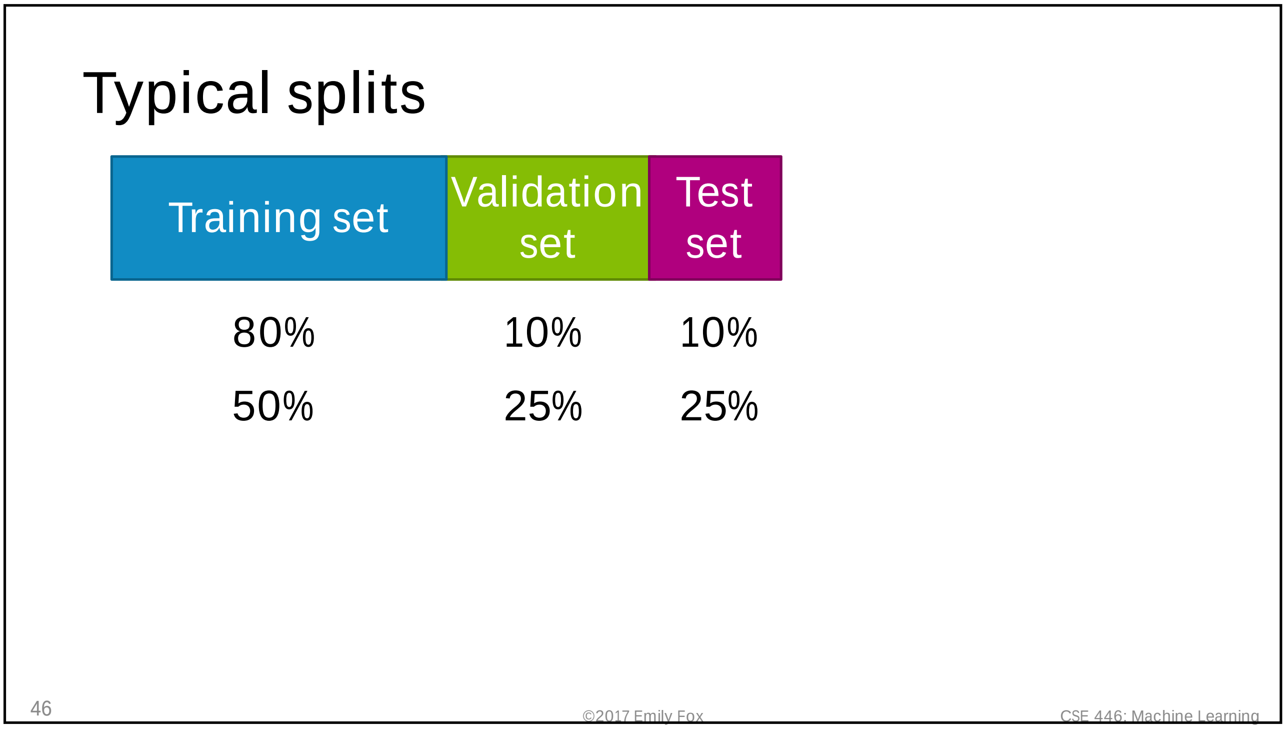


Practical implementation

fit $\hat{\mathbf{w}}_\lambda$

test performance of $\hat{\mathbf{w}}_\lambda$ to select λ^*

assess generalization error of $\hat{\mathbf{w}}_{\lambda^*}$



Typical splits

Training set	Validation set	Test set
80%	10%	10%
50%	25%	25%

How to handle the intercept

OPTIONAL

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., $\mathbf{x}[2]$

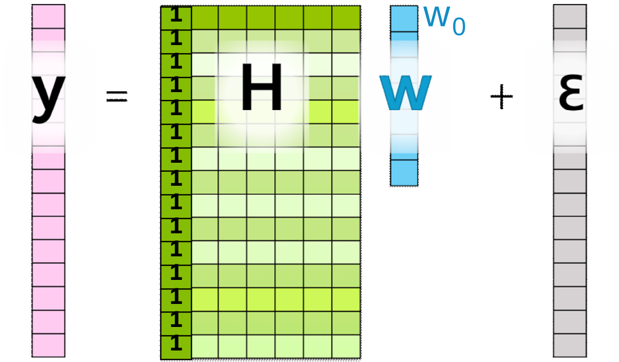
…

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

If constant feature…

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

In matrix notation for N observations:



Do we penalize intercept?

Standard ridge regression cost:

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

strength of penalty

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

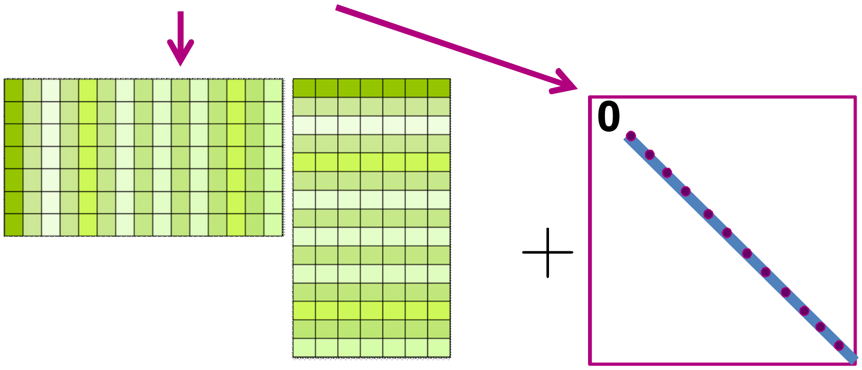
Modified ridge regression cost:

$$\text{RSS}(w_{0,}\mathbf{w}_{rest}) + \lambda\|\mathbf{w}_{rest}\|_2^2$$

How to implement this in practice?

Option 1: Don’t penalize intercept – Closed-form solution –

$$\hat{\mathbf{w}} = (\underbrace{\mathbf{H}^T\mathbf{H}} + \underbrace{\lambda\mathbf{I}^{mod}})^{-1}\mathbf{H}^T\mathbf{y}$$



Option 1: Don’t penalize intercept – Gradient descent algorithm –

```
while $||\nabla RSS(\mathbf{w}^{(t)})|| > \varepsilon$
    for j=0,...,D
        partial[j] = $-2\sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$
        if j==0
            $w_0^{(t+1)} \leftarrow w_0^{(t)} - \eta$ partial[j]
        else
            $w_j^{(t+1)} \leftarrow (1-2\eta\lambda)w_j^{(t)} - \eta$ partial[j]
    t ← t + 1
```

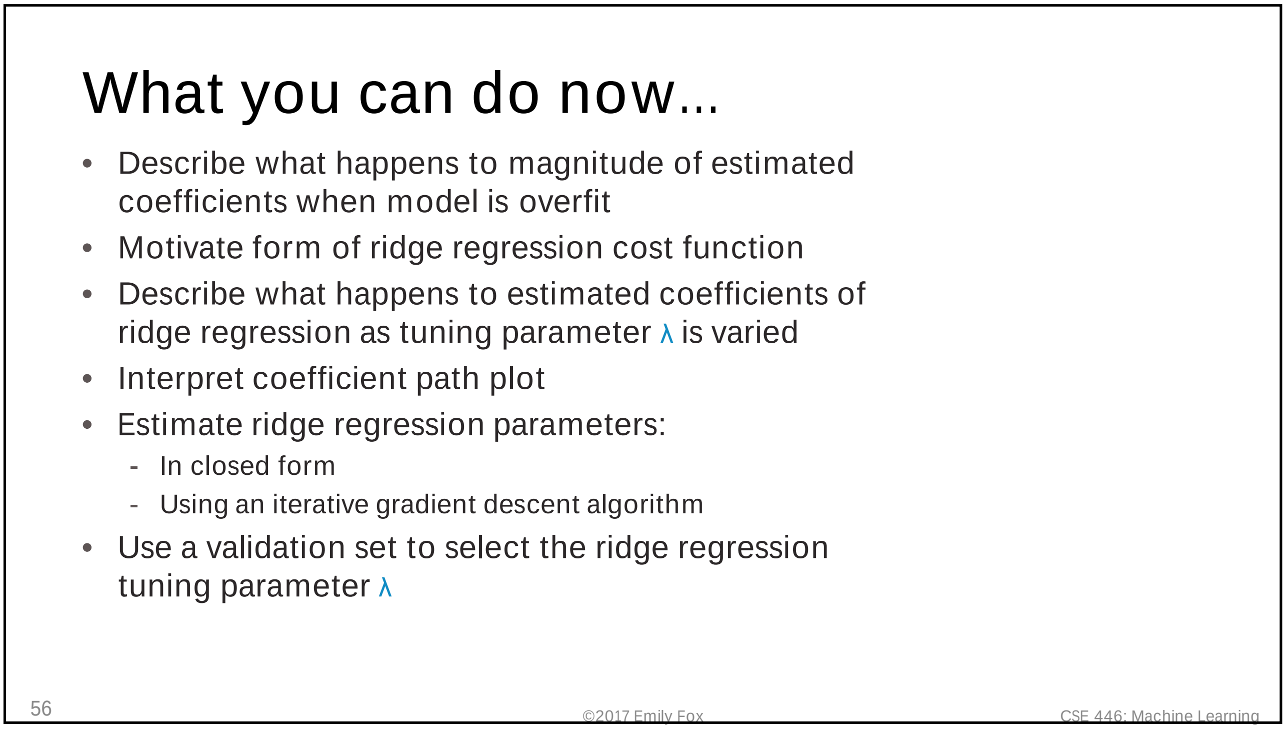
Option 2: Center data first

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)

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
- Estimate ridge regression parameters:
  - In closed form
  - Using an iterative gradient descent algorithm
- Use a validation set to select the ridge regression tuning parameter λ