Bias-variance tradeoff (formally)

CS 229: Machine Learning
Emily Fox
Stanford University
January 17, 2024

1

Recap: Assessing performance

2

Measuring loss

Loss function:

$L(y, f_{\hat{w}}(\mathbf{x}))$

Cost of using ŵ at x when y is true

actual value

$\hat{f}(\mathbf{x})$ = predicted value ŷ

Examples: (assuming loss for underpredicting = overpredicting)

Absolute error: $L(y, f_{\hat{w}}(\mathbf{x})) = |y - f_{\hat{w}}(\mathbf{x})|$

Squared error: $L(y, f_{\hat{w}}(\mathbf{x})) = (y - f_{\hat{w}}(\mathbf{x}))^2$

Compute training error

Training error

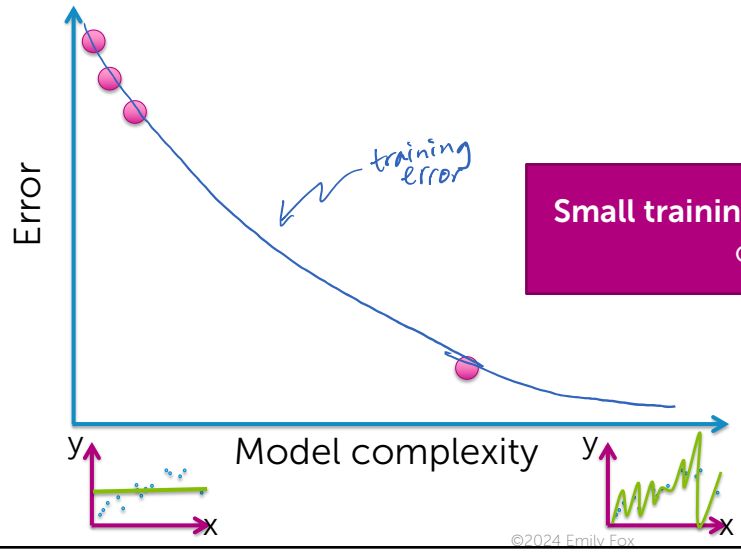
= avg. loss on houses in training set

$= \frac{1}{N} \sum_{i=1}^{N} L(y_i, f_{\hat{w}}(\mathbf{x}_i))$

N ← # obs in training set

fit using training data

Training error vs. model complexity



Small training error ≠> good predictions
overly optimistic

Generalization error definition

Really want estimate of loss over all possible (🏠,$) pairs

Formally:

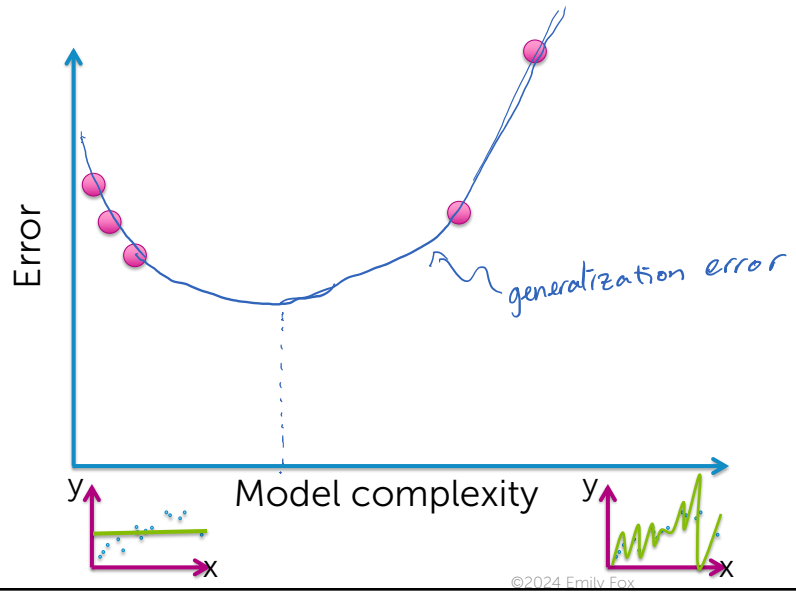
average over all possible (**x**,y) pairs weighted by how likely each is

generalization error = $E_{\mathbf{x},y}[L(y,f_{\hat{w}}(\mathbf{x}))]$

fit using training data

$= \int L(y, f_{\hat{w}}(x)) \underbrace{p(x,y)}_{p(y|x)p(x)} dx\,dy$

Generalization error vs. model complexity



Can't compute!

Forming a test set

Hold out some (🏠,$) that are *not* used for fitting the model

Training set

Test set

Compute test error

generalization error = $E_{x,y}[L(y, f_{\hat{w}}(\mathbf{x}))]$

$= \int L(y, f_{\hat{w}}(\mathbf{x}))\, p(y, \mathbf{x})\, dx dy$

$p(y|\mathbf{x})\, p(\mathbf{x})$

$\mathbf{x}, y \overset{iid}{\sim} p(x, y)$

Test error

= avg. loss on houses in test set

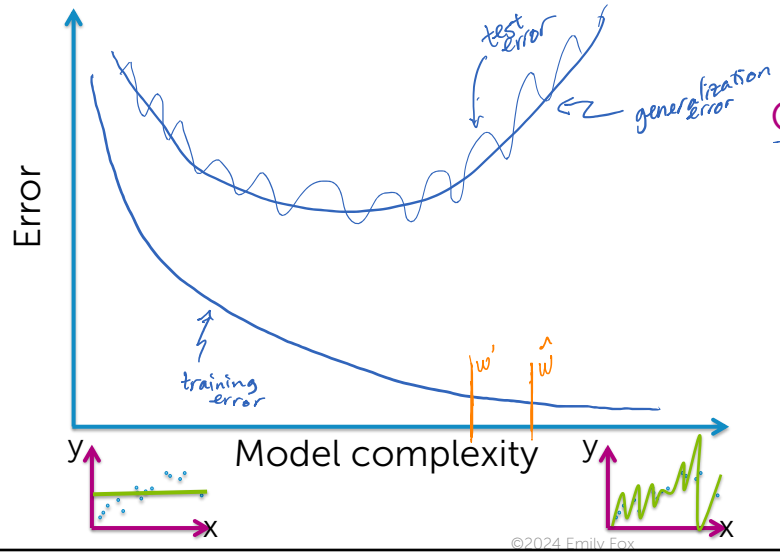
$= \frac{1}{N_{test}} \sum_{i \text{ in test set}} L(y_i, f_{\hat{w}}(\mathbf{x}_i))$

test points

fit using training data

has never seen test data!

Training, true, & test error vs. model complexity



Overfitting if: using $\hat{w}$

If there exists a model w/ est. params w' s.t.

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

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

3 sources of error + the bias-variance tradeoff

11

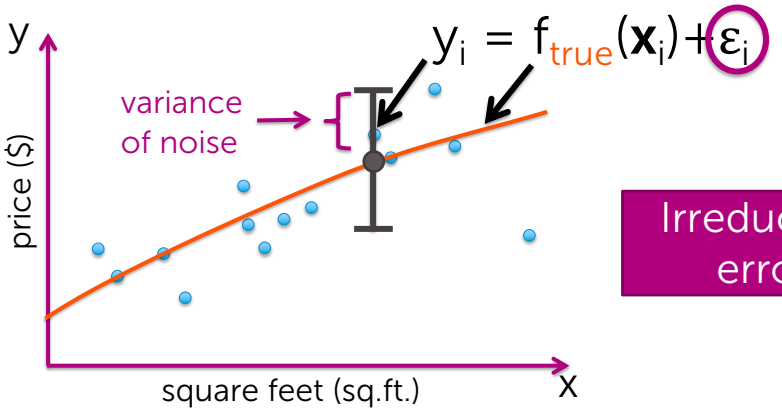
3 sources of error

In forming predictions, there are 3 sources of error:

1. Noise
2. Bias
3. Variance

12

Data inherently noisy



$y_i = f_{true}(\mathbf{x}_i) + \varepsilon_i$

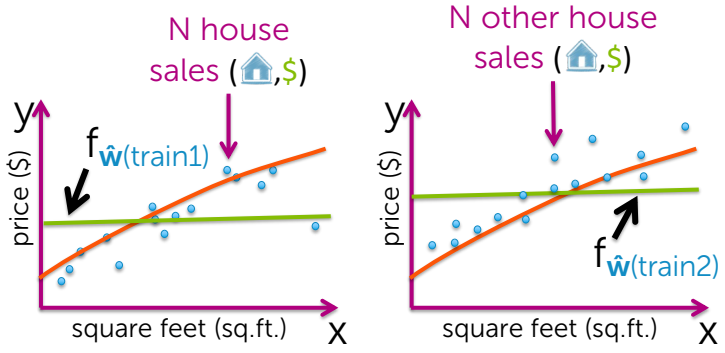
$E[\epsilon_i] = 0$
$Var(\epsilon_i) = \sigma^2$

Even if $f_{\hat{w}} = f_{true}$, will have error due to ϵ_i

Irreducible error

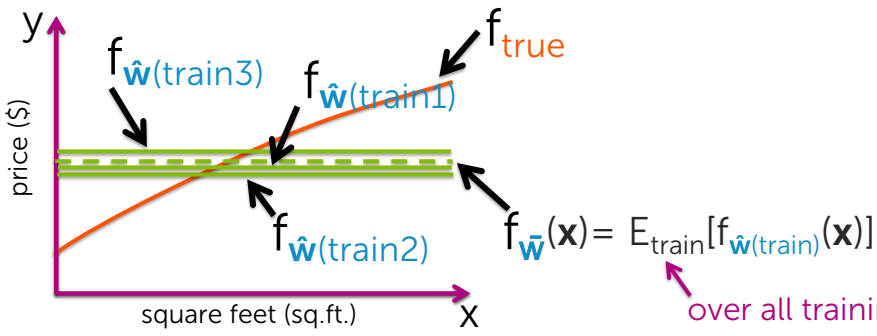
Bias contribution

Assume we fit a constant function



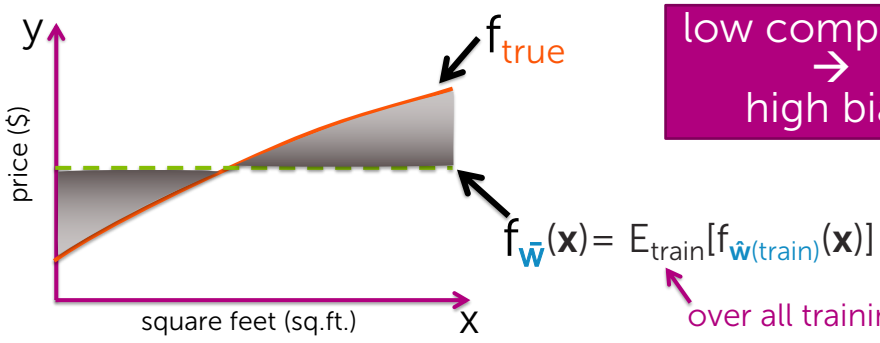
Bias contribution

Over all possible size N training sets, what do I expect my fit to be?



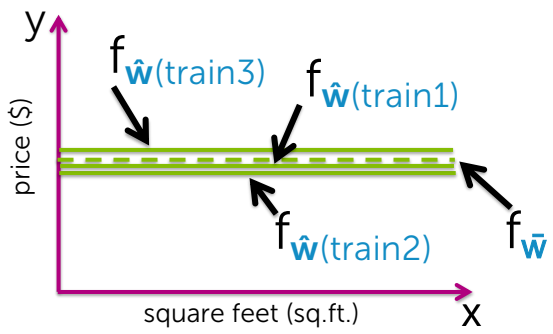
Bias contribution

$Bias(f_{\hat{w}}(x)) = f_{true}(x) - f_{\bar{w}}(x)$ ← Is our approach flexible enough to capture f_{true}? If not, error in predictions.



Variance contribution

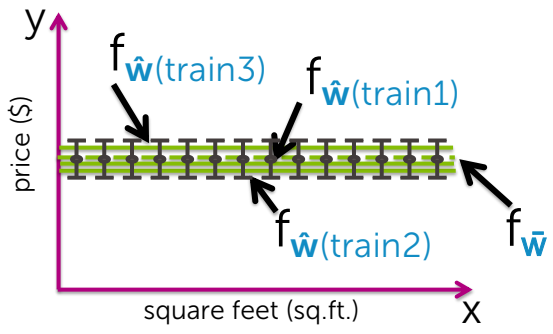
How much do specific fits vary from the expected fit?



Variance contribution

$Var(x) = E[(x - E[x])^2]$

How much do specific fits vary from the expected fit?



fit on a specific training dataset

what I expect to learn over all training sets

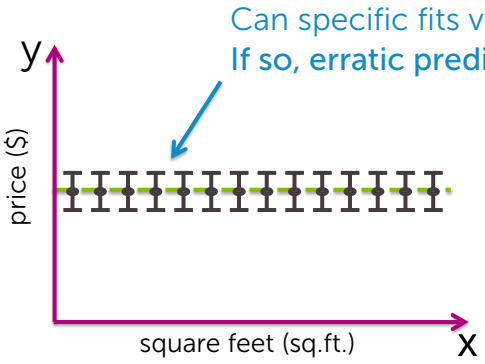
$$var(f_{\hat{w}}(\mathbf{x})) = E_{train}[(f_{\hat{w}(train)}(\mathbf{x}) - f_{\bar{w}}(\mathbf{x}))^2]$$

over all training sets of size N

deviation of specific fit from expected fit at **x**

Variance contribution

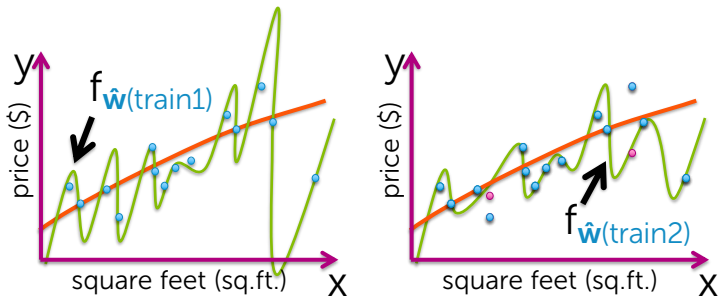
How much do specific fits vary from the expected fit?



low complexity
→
low variance

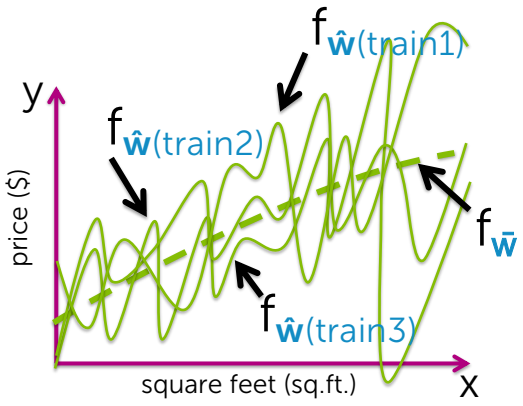
Variance of high-complexity models

Assume we fit a high-order polynomial

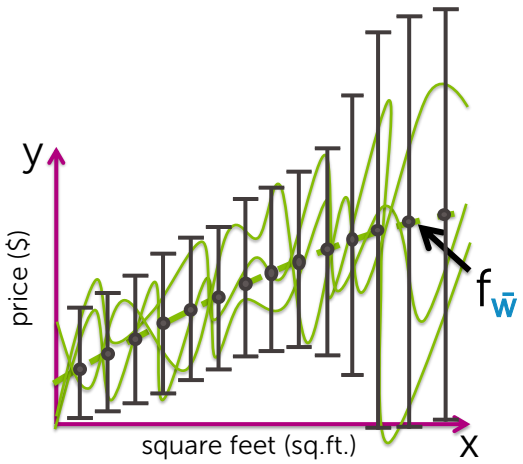


Variance of high-complexity models

Assume we fit a high-order polynomial

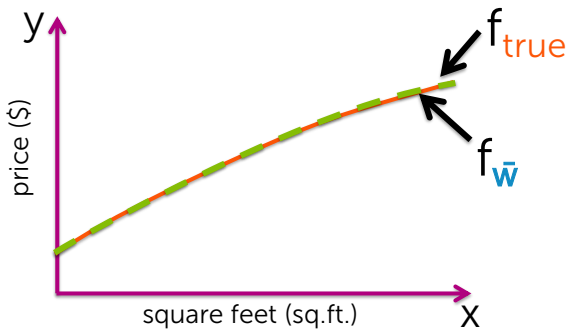


Variance of high-complexity models



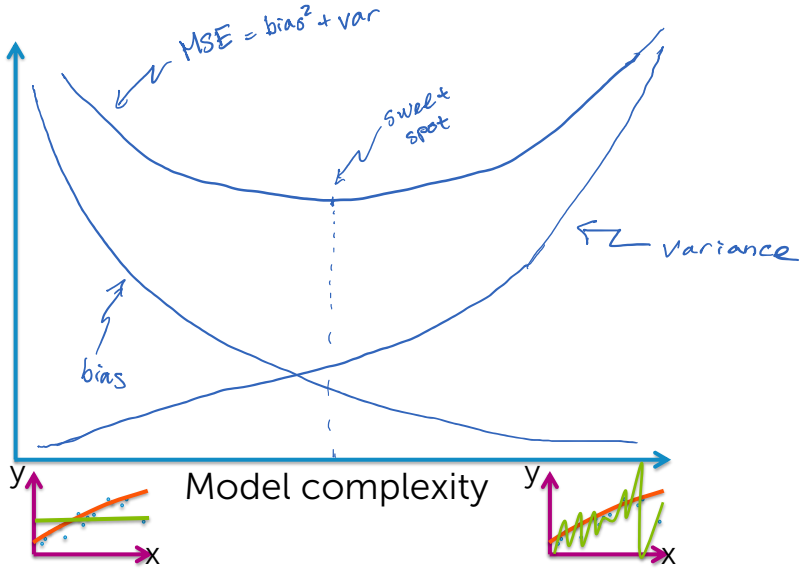
high complexity
→
high variance

Bias of high-complexity models

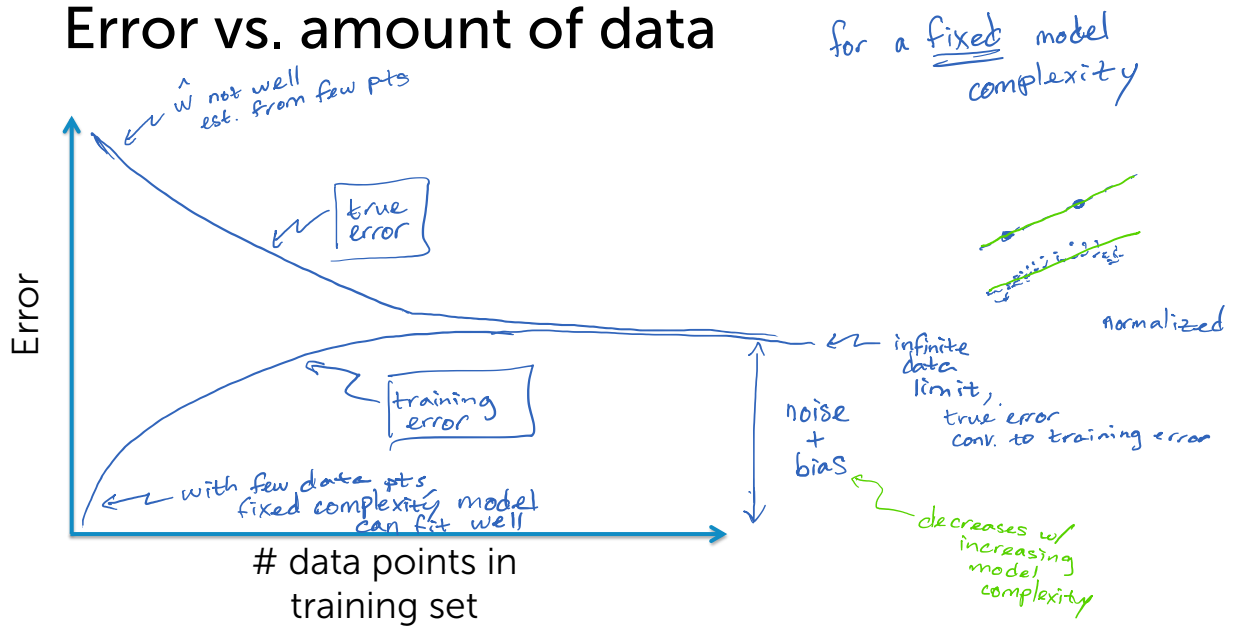


high complexity
→
low bias

Bias-variance tradeoff



Error vs. amount of data



Why 3 sources of error?
A formal derivation

Deriving expected prediction error

Expected prediction error
= E_{train} [generalization error of $\hat{\mathbf{w}}$(train)]
= E_{train} [$E_{\mathbf{x},y}$[L(y,$f_{\hat{\mathbf{w}}(train)}(\mathbf{x})$)]]

1. Look at specific **x**
2. Consider $L(y,f_{\hat{\mathbf{w}}}(\mathbf{x})) = (y-f_{\hat{\mathbf{w}}}(\mathbf{x}))^2$

Expected prediction error at **x**
= $E_{train}[E_{y|\mathbf{x}}\ [(y-f_{\hat{\mathbf{w}}(train)}(\mathbf{x}))^2]]$

Deriving expected prediction error

Expected prediction error at **x**
= $E_{train}[E_{y|\mathbf{x}}\ [(y-f_{\hat{\mathbf{w}}(train)}(\mathbf{x}))^2]$
= $E_{train}\ [E_{y|\mathbf{x}}\ [((y - f_{true}(\mathbf{x})) + (f_{true}(\mathbf{x}) - f_{\hat{\mathbf{w}}(train)}(\mathbf{x})))^2]$

$= E_{train}[E_{y|x}[(y-f)^2]] + 2\,E_{train}[E_{y|x}[(y-f)(f-\hat{f})]]$
$+ E_{train}[E_{y|x}[(f-\hat{f})^2]]$
$= \sigma^2 + MSE(\hat{f})$

$(y-f)^2 = \epsilon^2 = \sigma^2$ by defn
$(y-f) = \epsilon$, $E[\epsilon]E[f-\hat{f}] = 0$
$\triangleq MSE(\hat{f})$ mean square error

Shorthand: $f_{true} \to f$, $f_{\hat{w}(train)} \to \hat{f}$

$E[(a+b)^2] = E[a^2] + 2E[ab] + E[b^2]$

$E[ab] = E[a]E[b]$ if a, b uncorr (or, ind.)

Equating MSE with bias and variance

MSE(**x**)

$= E_{train}[(f_{true}(\mathbf{x}) - f_{\hat{\mathbf{w}}(train)}(\mathbf{x}))^2]$

$= E_{train}[((f_{true}(\mathbf{x}) - f_{\bar{\mathbf{w}}}(\mathbf{x})) + (f_{\bar{\mathbf{w}}}(\mathbf{x}) - f_{\hat{\mathbf{w}}(train)}(\mathbf{x})))^2]$

$= E_{train}[(f - \bar{f})^2] + 2\,E_{train}[(f - \bar{f})(\bar{f} - \hat{f})] + E_{train}[(\bar{f} - \hat{f})^2]$

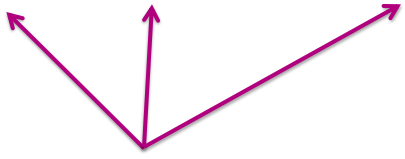
$= bias^2(\hat{f}) + var(\hat{f})$

Putting it all together

Expected prediction error at **x**

$= \sigma^2 + MSE(f_{\hat{\mathbf{w}}}(\mathbf{x}))$

$= \sigma^2 + [bias(f_{\hat{\mathbf{w}}}(\mathbf{x}))]^2 + var(f_{\hat{\mathbf{w}}}(\mathbf{x}))$



3 sources of error

Summary of assessing performance

What you can do now…

- Describe what a loss function is and give examples
- Contrast training, generalization, and test error
- Compute training and test error given a loss function
- Discuss issue of assessing performance on training set
- Define overfitting in terms of training and generalization (or, in practice, test) error
- Describe tradeoffs in forming training/test splits
- List and interpret the 3 sources of avg. prediction error
  - irreducible error, bias, and variance
- Derive avg. prediction error in terms of irreducible error, bias, and variance
- Sketch:
  - training & generalization/test error vs. model complexity
  - training & generalization error vs. # of training data points

Ridge Regression:
Regulating overfitting when using many features

CS 229: Machine Learning
Emily Fox
Stanford University
January 17, 2024

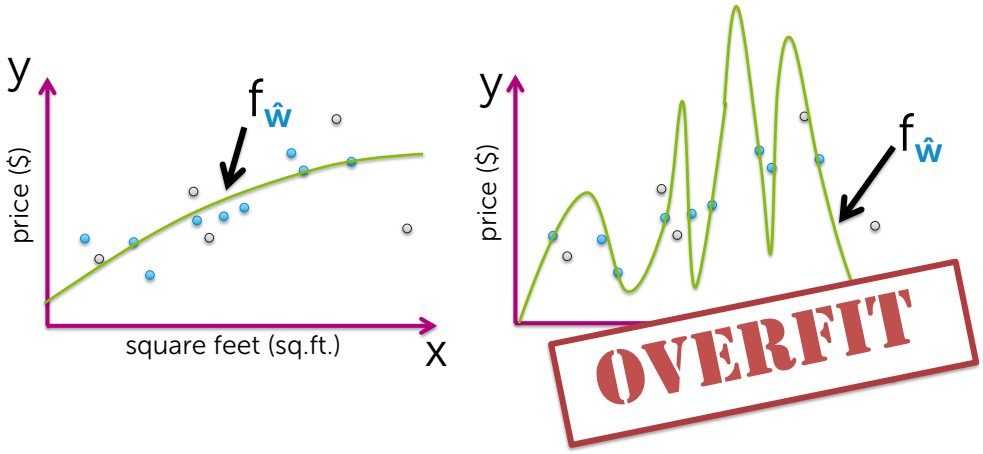
33

Overfitting of polynomial regression

34

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

37

Overfitting with many features

Not unique to polynomial regression

Generically, can happen with **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
- …

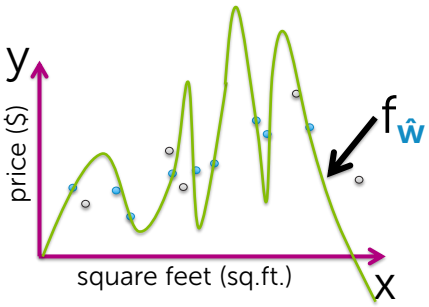
38

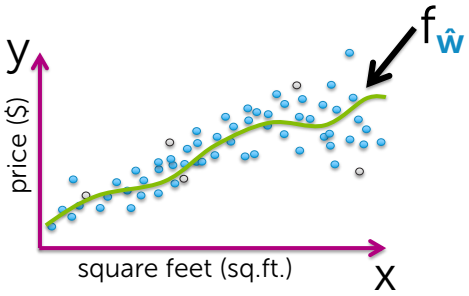
38

How does # of observations influence overfitting?

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

Many observations (N very large)
→ harder to overfit

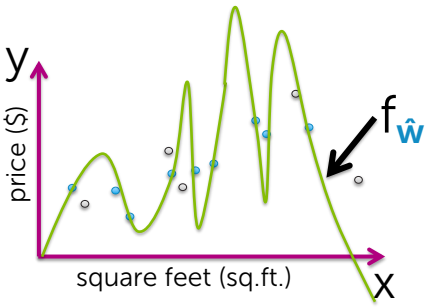




How does # of inputs influence overfitting?

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

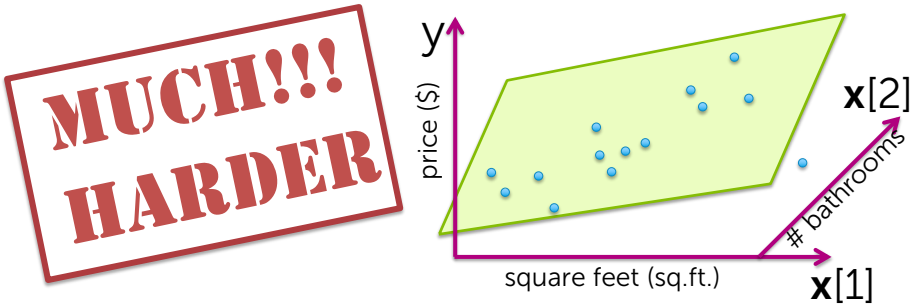
Data must include representative examples of all possible (sq.ft., $) pairs to avoid overfitting



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 (loss) function to prefer small coefficients

Desired total cost format

Want to balance:

i. How well function fits data

ii. Magnitude of coefficients

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

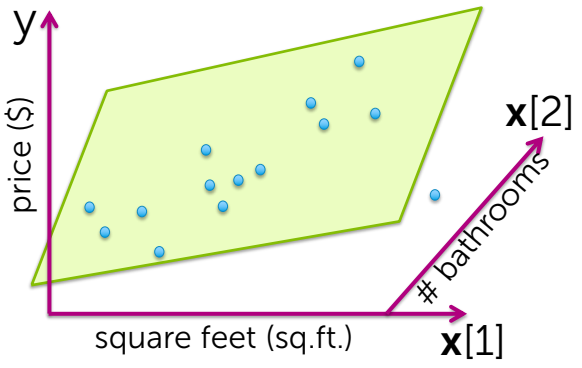
want to balance

measure quality of fit

small # = good fit to training data

small # = not overfit

Measure of fit to training data



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

pred. value using w

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? $\sum_{j=0}^{D} |w_j| \triangleq \|w\|_1$ 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 =

measure of fit + measure of magnitude of coefficients

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

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

If λ=∞:

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

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

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=∞)

In essence, λ controls model complexity

Small λ:

low bias, high variance

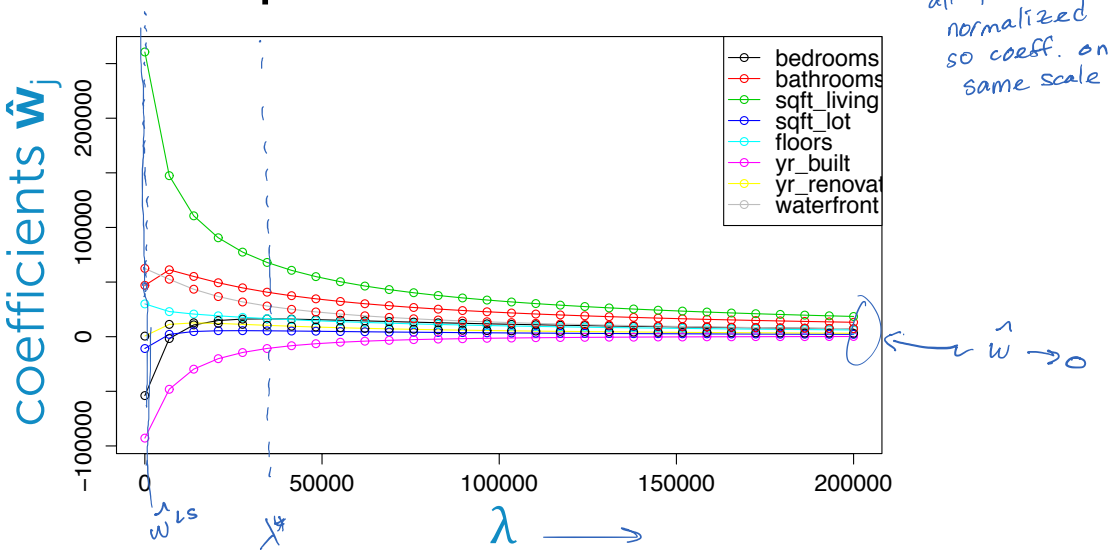
(e.g., standard least squares (RSS) fit of high-order polynomial for λ=0)

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)

53

Gradient of ridge regression cost

$$\|w\|_2^2 = w_0^2 + w_1^2 + \dots + w_D^2 = w^T w = [w_0 \dots w_D] \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}$$

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

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

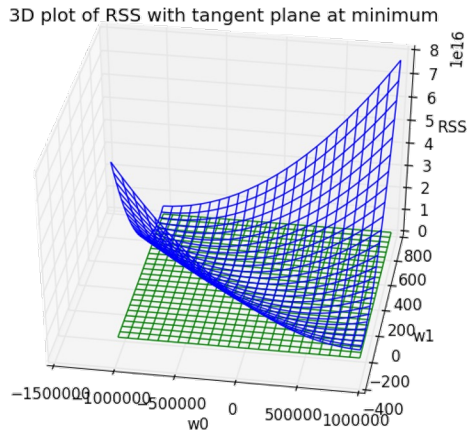
Why? By analogy to 1d case…

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

54

54

Ridge closed-form solution



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

add in

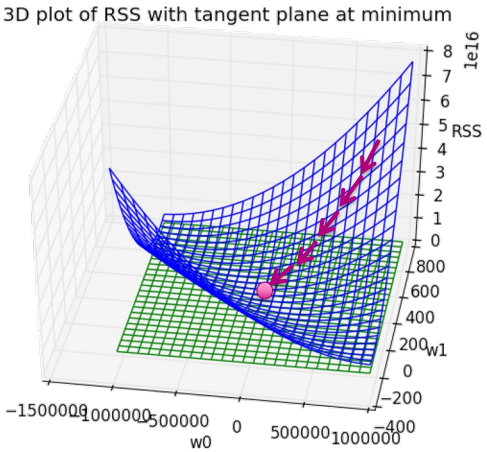
Solve for $\mathbf{w}$:

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

"regularizer"

Gradient descent for ridge regression

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



Algorithm:

while not converged

$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla \text{cost}(w^{(t)})$

$-2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w}^{(t)}) + 2\lambda\mathbf{w}^{(t)}$

$\leftarrow (1-2\eta\lambda) w^{(t)} - 2\eta H^T(y - H w^{(t)})$

How to choose λ

57

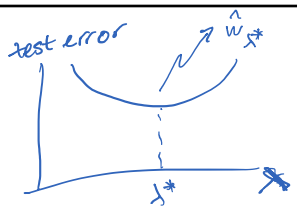
The regression/ML workflow

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

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

58

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

Training set	Test set

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

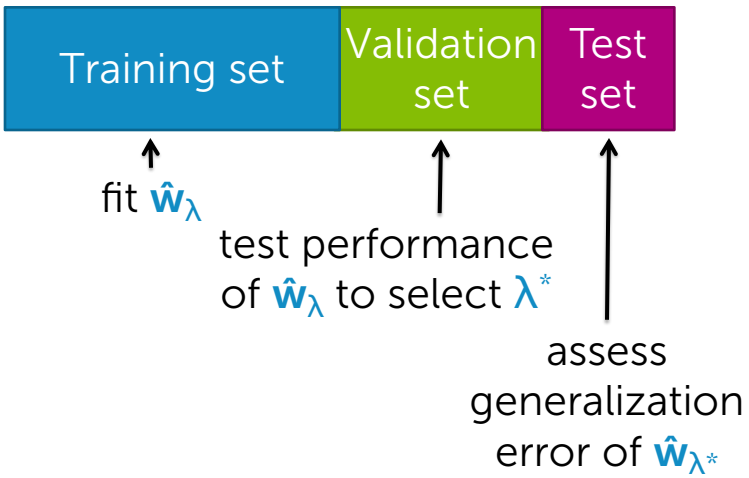
- λ^* was selected to minimize test error (i.e., λ^* was fit on test data)
- If test data is not representative of the whole world, then $\hat{\mathbf{w}}_{\lambda^*}$ will typically perform worse than test error indicates

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



Typical splits

80%	10%	10%
50%	25%	25%

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
- Handle intercept and scale of features with care

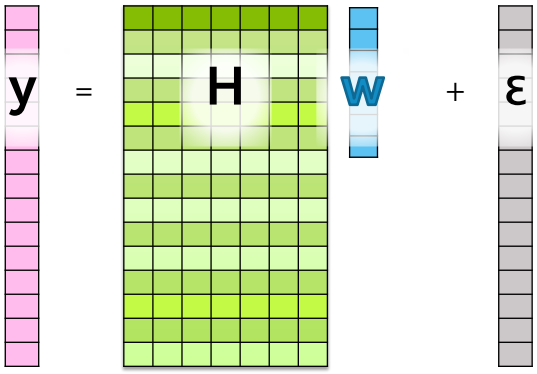
Fitting the ridge regression model (for given λ value) in more detail

Step 1:
Rewrite total cost in matrix notation

67

Recall matrix form of RSS

Model for all N observations together



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

68

68

Rewrite magnitude of coefficients in vector notation

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

$$= \begin{bmatrix} w_0 & w_1 & \cdots & w_D \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \\ \\ \\ \\ w_D \end{bmatrix}$$

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

Putting it all together

In matrix form, ridge regression cost is:

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

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

Step 2:
Compute the gradient

Gradient of ridge regression cost

$\|w\|_2 = \sqrt{w_0^2 + \dots + w_D^2} = \sqrt{w^T w}$

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

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

Why? By analogy to 1d case…

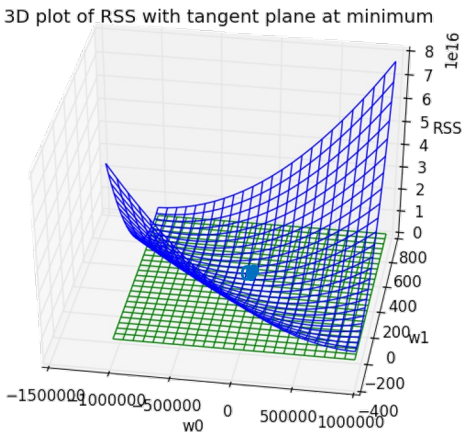
$\mathbf{w}^T\mathbf{w}$ analogous to w^2 and derivative of $w^2 = 2w$

72

Step 3, Approach 1:
Set the gradient = 0

73

Ridge closed-form solution



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

add

Solve for $\mathbf{w}$:

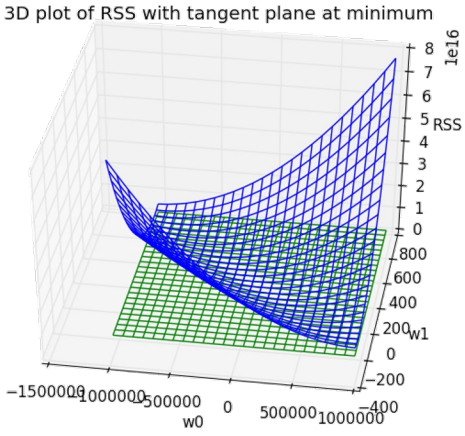
$$-H^T y + H^T H\hat{w} + \lambda I\hat{w} = 0$$

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

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

74

Interpreting ridge closed-form solution



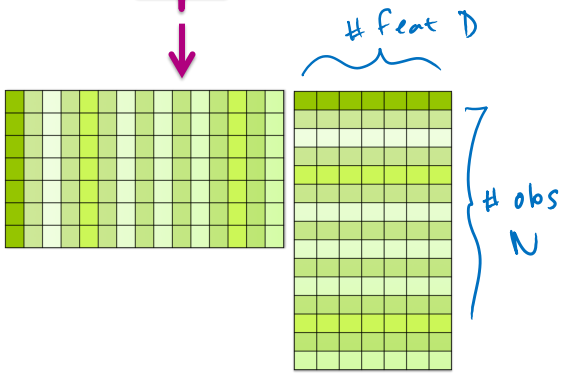
$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y}$$

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

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

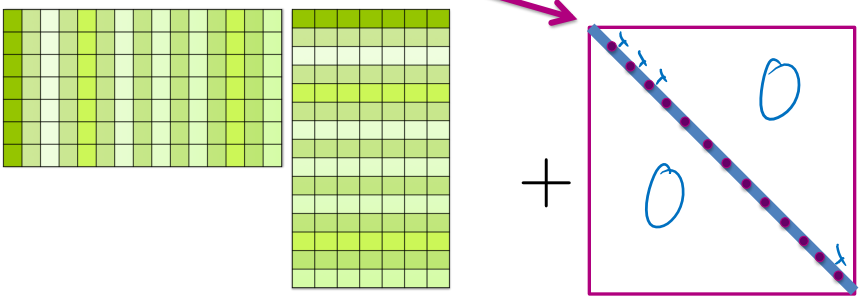


Invertible if:
- In general,
- (# linearly independent obs)
- N > D

Complexity of inverse:
- $O(D^3)$

Discussion of ridge closed-form solution

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



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

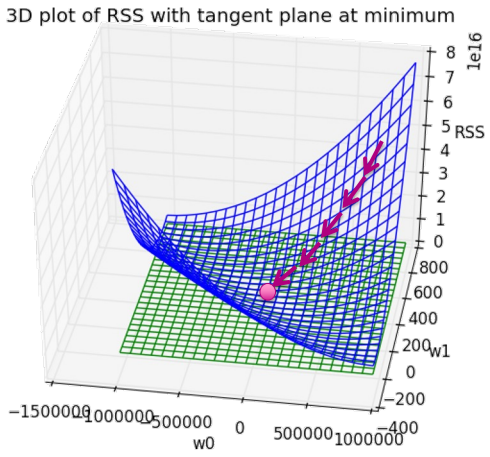
Invertible if:
Always if $\lambda > 0$, even if $N < D$

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

Step 3, Approach 2:
Gradient descent

Gradient descent for ridge regression

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



Algorithm:

while not converged

$w^{(t+1)} \leftarrow w^{(t)} - \eta \underbrace{\nabla \text{cost}(w^{(t)})}_{-2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w}^{(t)}) + 2\lambda\mathbf{w}^{(t)}}$

How to handle the intercept

Recall multiple regression model

Model:

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

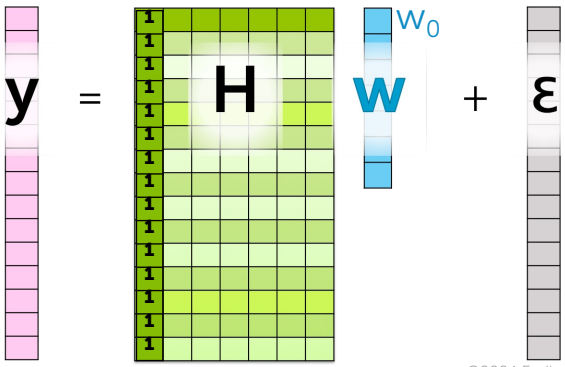
$= \sum_{j=0}^{D} w_j h_j(\mathbf{x}_i) + \varepsilon_i$

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

If constant feature (intercept)…

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

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

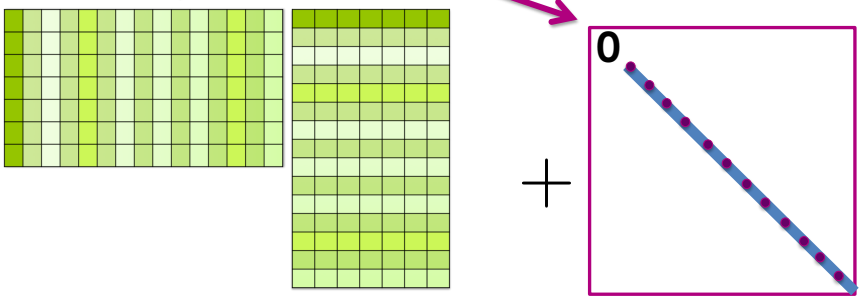
Modified ridge regression cost:

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

How to implement this in practice?

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

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



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

      $w_j^{(t+1)} \leftarrow (1-2\eta\lambda)w_j^{(t)} - \eta\ partial[j]$

  $t \leftarrow t + 1$

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)

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