Regression recap

CS 229: Machine Learning
Emily Fox
Stanford University
January 10, 2024

1

How much is my house worth?

2

How much is my house worth?

3

Data

input *output*


(x_1 = sq.ft., y_1 = $)


(x_2 = sq.ft., y_2 = $)


(x_3 = sq.ft., y_3 = $)


(x_4 = sq.ft., y_4 = $)


(x_5 = sq.ft., y_5 = $)

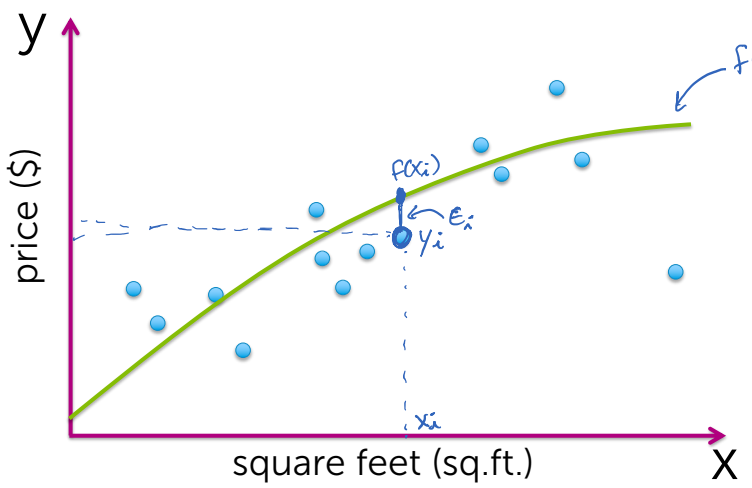
⋮

Input vs. Output:
- y is the quantity of interest
- assume y can be predicted from x

Example of supervised learning

4

Model – How we *assume* the world works



Regression model:

$y_i = f(x_i) + \epsilon_i$

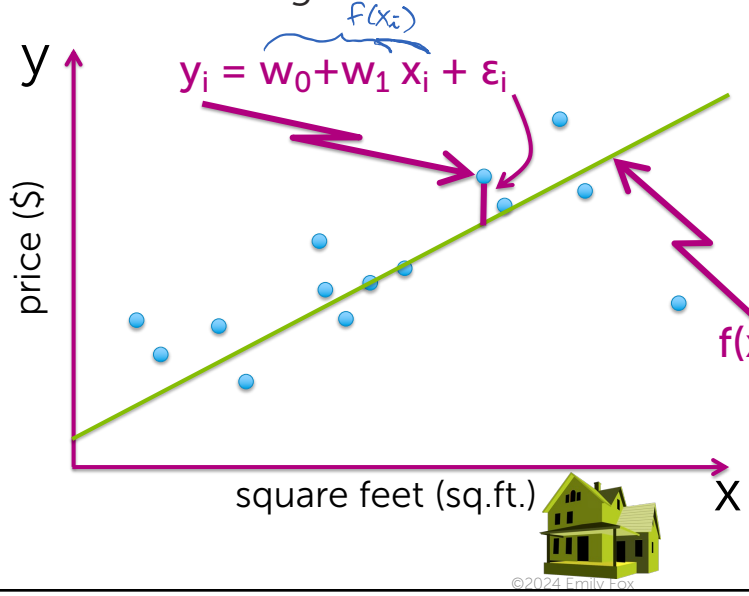
$E[\epsilon_i] = 0$

expected value

5

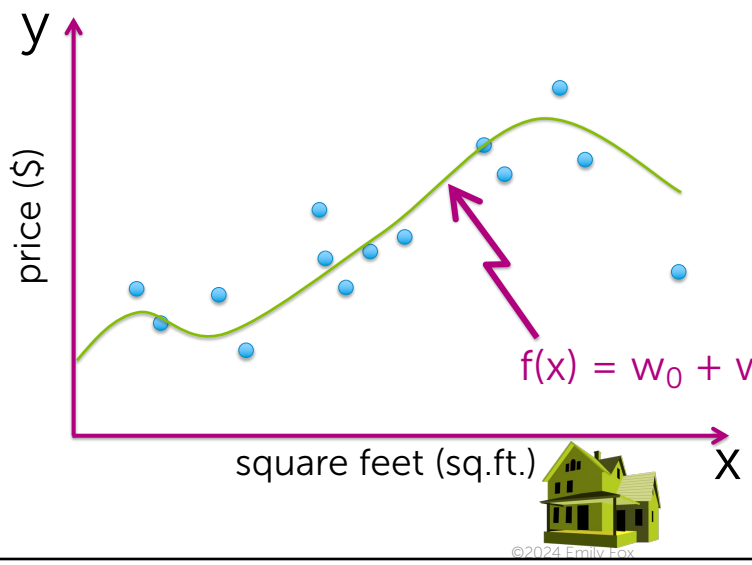
Use a simple **linear** regression model

Fit a line through the data



6

Polynomial regression (still "linear" regression)

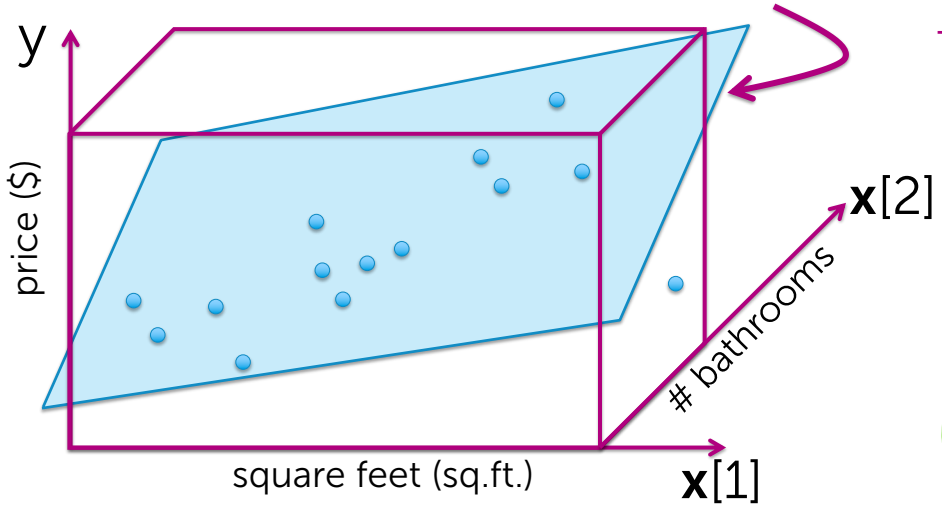


$f(x) = w_0 + w_1 x + w_2 x^2 + \dots + w_p x^p$

7

Considering higher-dim input spaces

$f(\mathbf{x}) = w_0 + w_1 \text{ sq.ft.} + w_2 \#\text{bath}$



- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- Time since last remodel
- Zipcode
- …

8

Generic linear regression model

Model:

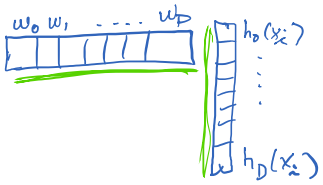
$$y_i = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \ldots + w_D h_D(\mathbf{x}_i) + \varepsilon_i$$

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

vector input

features

feature



feature 1 = $h_0(\mathbf{x})$ … e.g., 1

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

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

2nd component of input vector

or, log($\mathbf{x}[7]$) $\mathbf{x}[2]$ = log(#bed) x #bath

…

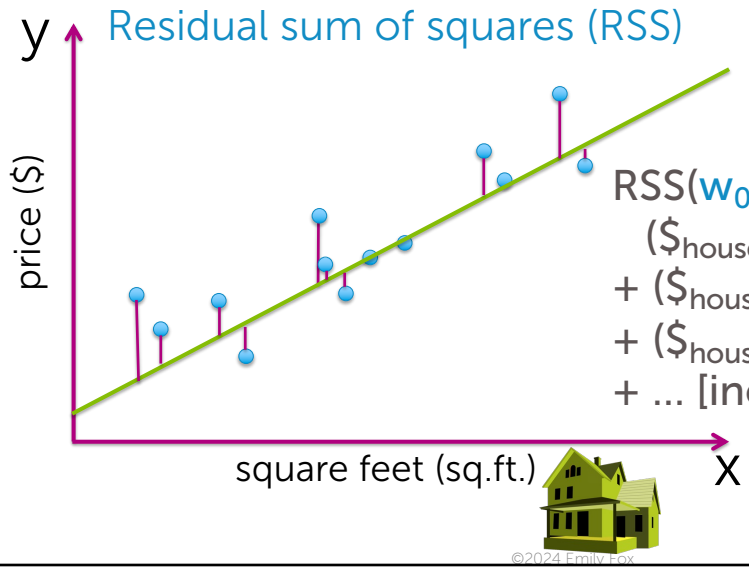
feature D+1 = $h_D(\mathbf{x})$ … some other function of $\mathbf{x}[1]$,…, $\mathbf{x}[d]$

9

Fitting the linear regression model

10

"Cost" of using a given line (Loss function)

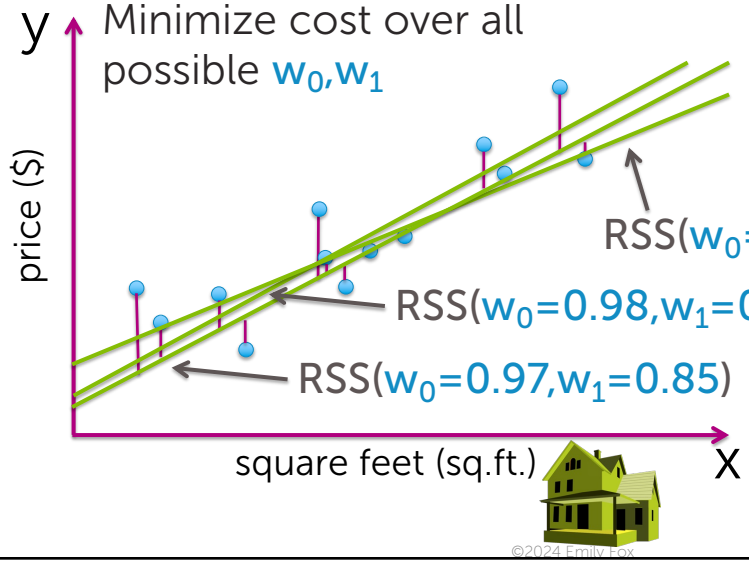


predicted price

$$RSS(w_0,w_1) = (\$_{house\ 1} - [w_0 + w_1 sq.ft._{house\ 1}])^2 + (\$_{house\ 2} - [w_0 + w_1 sq.ft._{house\ 2}])^2 + (\$_{house\ 3} - [w_0 + w_1 sq.ft._{house\ 3}])^2 + \ldots \text{ [include all houses]}$$

11

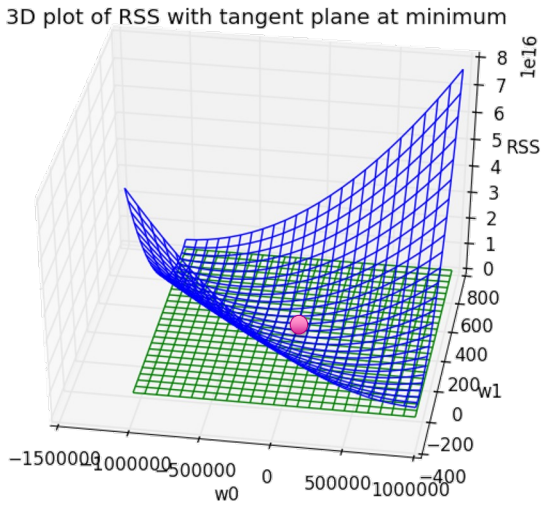
Find "best" line



$$RSS(w_0,w_1) = \sum_{i=1}^{N} (y_i - [w_0 + w_1 x_i])^2$$

12

Minimizing the cost



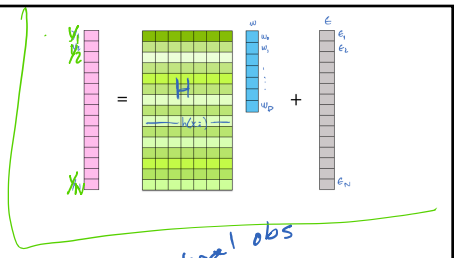
Minimize function over all possible w_0, w_1

$$\min_{w_0,w_1} \sum_{i=1}^{N} (y_i - [w_0 + w_1 x_i])^2$$

RSS(w_0, w_1) is a function of 2 variables

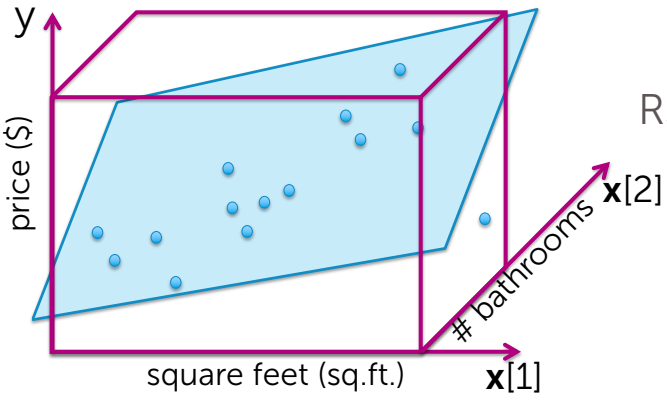
13

RSS for multiple regression



actual obs

e_i



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

pred. value

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

ϵ

show this at home

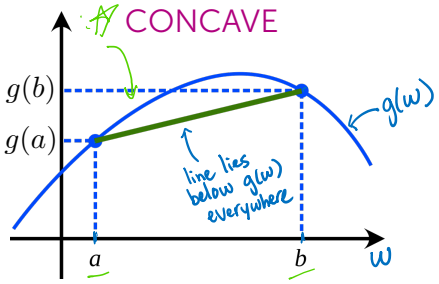
14

Background on optimization

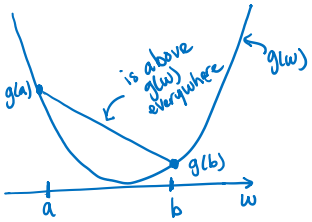
15

Convex/concave functions

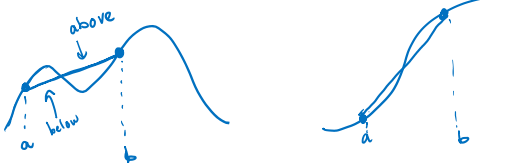
CONCAVE



CONVEX



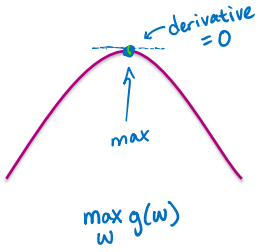
NEITHER



16

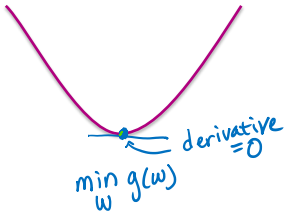
Finding the max or min analytically

CONCAVE



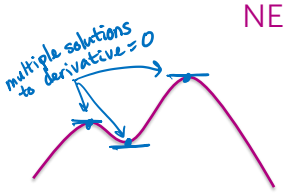
$\max_w g(w)$

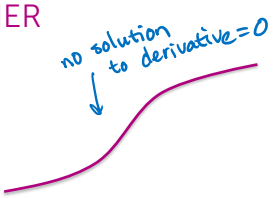
CONVEX



$\min_w g(w)$

NEITHER





Example:

$g(w) = 5-(w-10)^2$

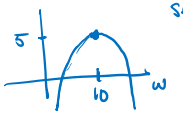
$\frac{dg(w)}{dw} = 0 - 2(w-10)^1 \cdot 1$

$= -2w + 20$

set derivate = 0:

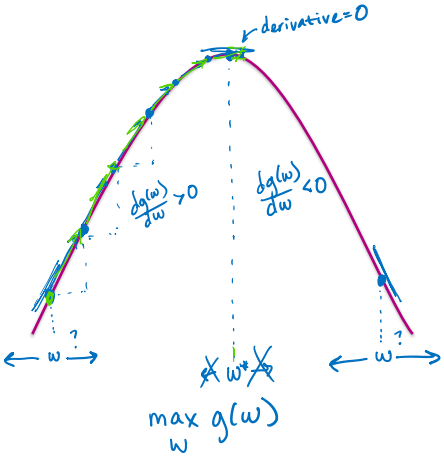
$-2w + 20 = 0$

$w = 10$



17

Finding the max via gradient ascent ("hill climbing")



How do we know whether to move w to right or left? (Inc. or dec. the value of w?)

while not converged

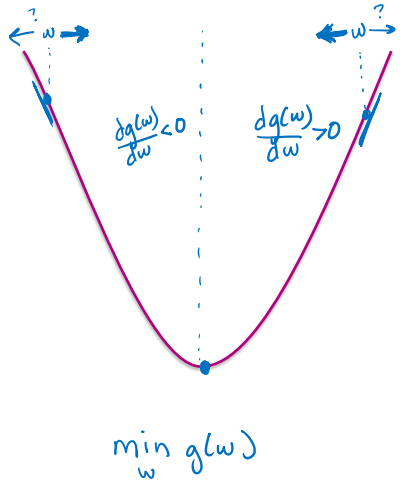
$w^{(t+1)} \leftarrow w^{(t)} + \eta \frac{dg(w)}{dw}$

iteration t

stepsize

18

Finding the min via gradient descent ("hill descent")



When derivative is positive, we want to decrease w and when derivative is negative, we want to increase w

Algorithm:

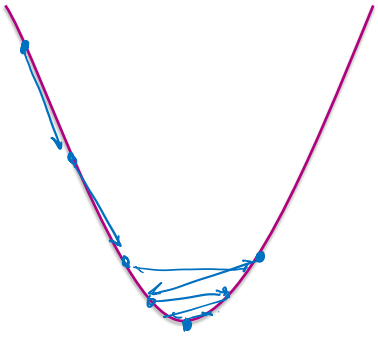
while not converged

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \left.\frac{dg}{dw}\right|_{w^{(t)}}$$

19

Choosing the stepsize— Fixed stepsize

η

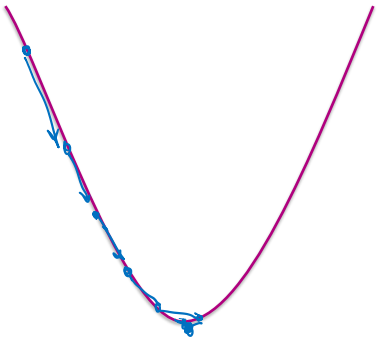


$\eta = 0.1$

20

Choosing the stepsize—
Decreasing stepsize

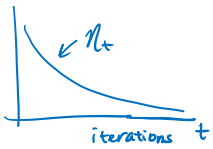
or stepsize schedule



Common choices:

$$\eta_t = \frac{\alpha}{t}$$

$$\eta_t = \frac{\alpha}{\sqrt{t}}$$



21

Convergence criteria

For convex functions, optimum occurs when

$$\frac{dg(w)}{dw} = 0$$

In practice, stop when

$$\left|\frac{dg(w)}{dw}\right| < \epsilon$$

↑ threshold to be set

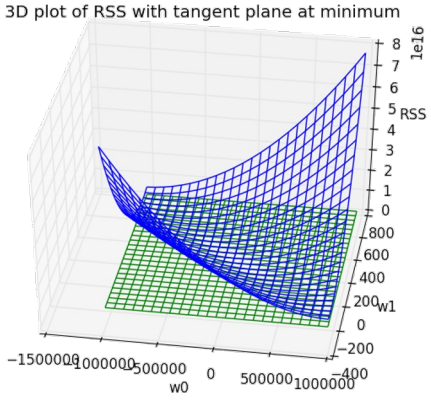
Algorithm:

while not converged

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \left.\frac{dg}{dw}\right|_{w^{(t)}}$$

22

Moving to multiple dimensions: Gradients



$$\nabla g(w) = \begin{bmatrix} \frac{\partial g}{\partial w_0} \\ \frac{\partial g}{\partial w_1} \\ \vdots \\ \frac{\partial g}{\partial w_p} \end{bmatrix}$$

gradient

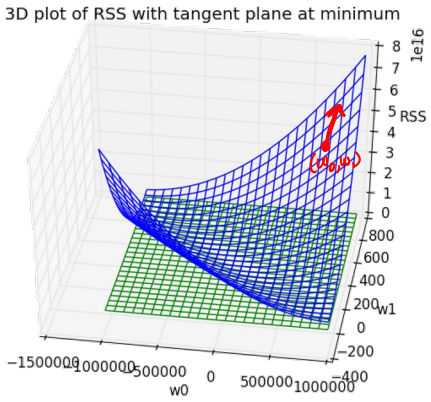
$[w_0, w_1, \ldots, w_p]$

$(p+1)$-dimensional vector

partial derivative is like a derivate with respect to w_i treating all other variables as constants

23

Gradient example



$$g(w) = 5w_0 + 10w_0w_1 + 2w_1^2$$

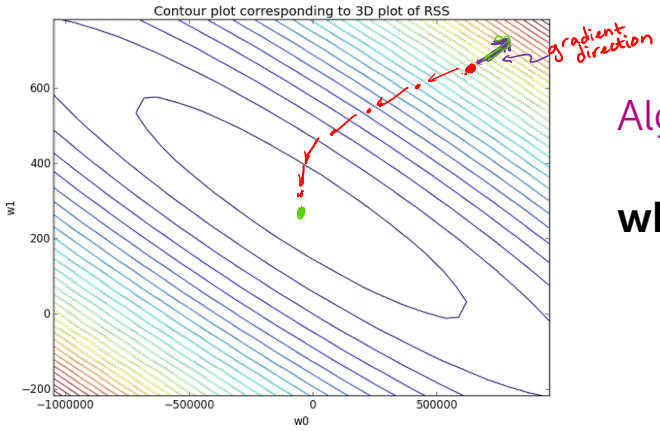
$$\frac{\partial g}{\partial w_0} = 5 + 10w_1$$

$$\frac{\partial g}{\partial w_1} = 10w_0 + 4w_1$$

$$\nabla g(w) = \begin{bmatrix} 5 + 10w_1 \\ 10w_0 + 4w_1 \end{bmatrix}$$

24

Gradient descent



Algorithm:

while not converged

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

$\begin{bmatrix} \\ \\ \\ \end{bmatrix} \leftarrow \begin{bmatrix} \\ \\ \\ \end{bmatrix} - \eta \begin{bmatrix} \\ \\ \\ \end{bmatrix}$

Convergence: $\|\nabla g(w)\| < \epsilon$

25

Back to regression:
Take the gradient of RSS with respect to **w**

26

Gradient of RSS

$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x[1]} \\ \frac{\partial f}{\partial x[2]} \\ \vdots \\ \frac{\partial f}{\partial x[m]} \end{bmatrix}$ ($m \times 1$) vector of partials

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

← vector of dim D+1 (length of w)

Why? By analogy to 1D case:

$$\frac{d}{dw}(y-hw)(y-hw) = \frac{d}{dw}(y-hw)^2 = 2(y-hw)(-h)$$
$$= -2h(y-hw)$$

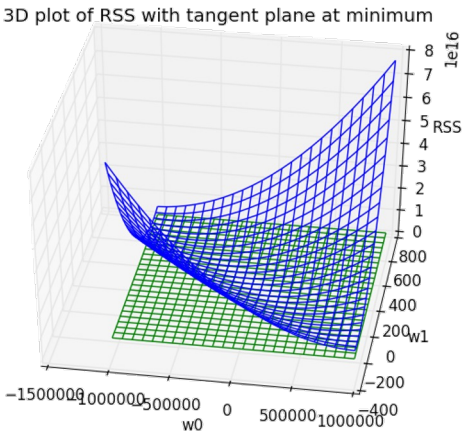
(scalars)

27

Approach 1:
Set the gradient = 0

28

Closed-form solution



$$\nabla RSS(\mathbf{w}) = -2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w}) = 0$$

Solve for $\mathbf{w}$:

$$-2H^Ty + 2H^TH\hat{w} = 0$$

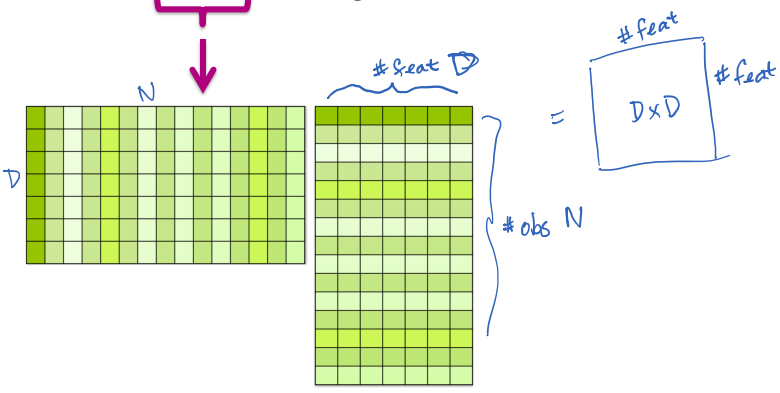
$$H^TH\hat{w} = H^Ty$$

$$\hat{w} = (H^TH)^{-1}H^Ty$$

29

Closed-form solution

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



Invertible if: (really, # of linearly ind. obs.)

In most cases $N > D$

Complexity of inverse:

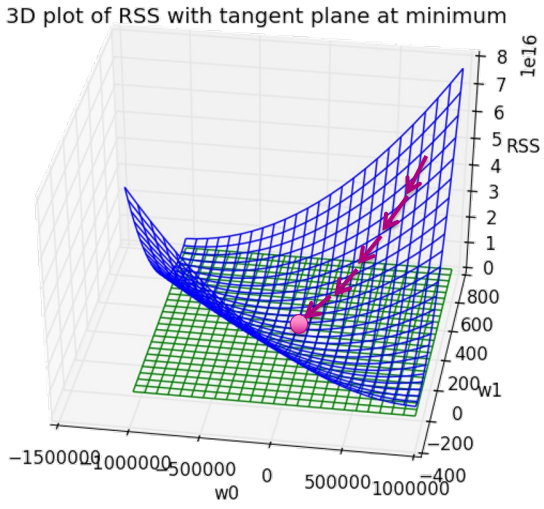
$O(D^3)$

30

Approach 2:
Gradient descent

31

Gradient descent



Algorithm:

while not converged

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

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

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

$\hat{y}(w^{(t)})$ best guess of obs. using $w^{(t)}$

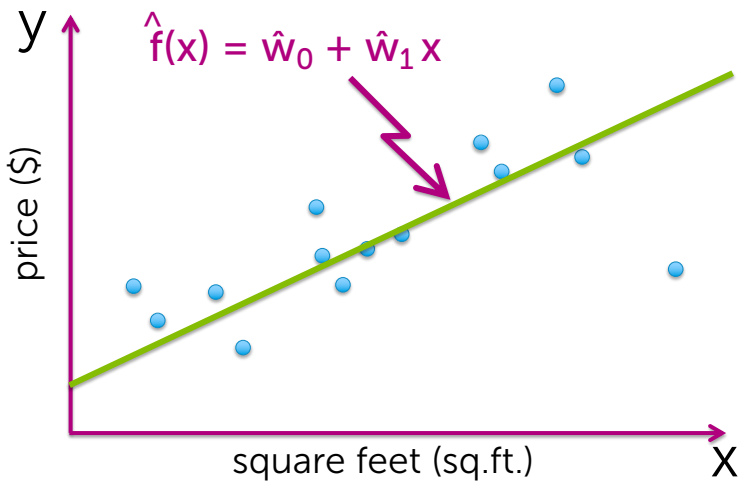
$\hat{\mathbf{w}}$

32

Using the fitted line

33

Model vs. fitted line



Regression model:

$y_i = w_0 + w_1 x_i + \varepsilon_i$

model params (unknown vars)

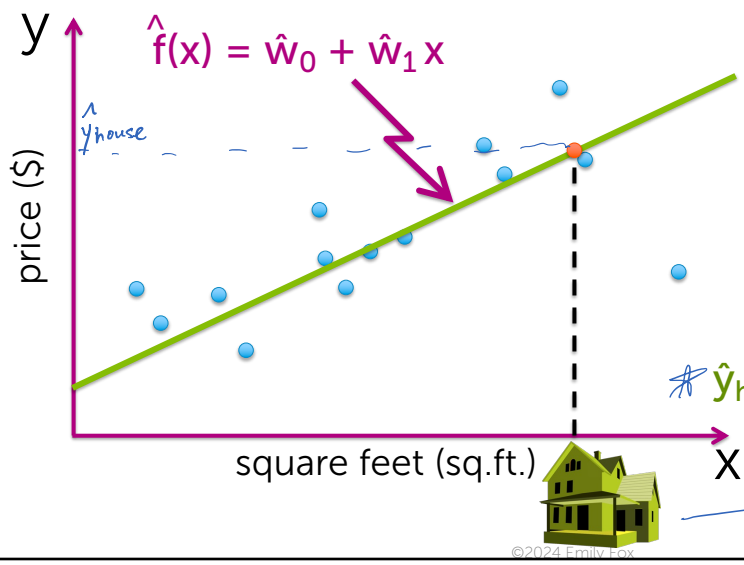
Estimated parameters:

$\hat{w}_0$, $\hat{w}_1$

estimated values (actual #'s)

34

Seller:
Predicting your house price



Regression model:

$y_i = w_0 + w_1 x_i + \varepsilon_i$

Best guess of your house price:

$\hat{y}_{house} = \hat{w}_0 + \hat{w}_1$ sq.ft.$_{house}$

35

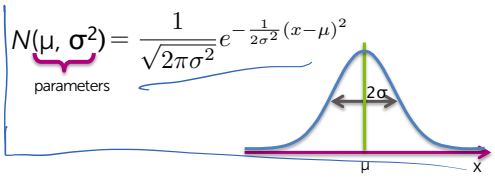
Why min RSS?

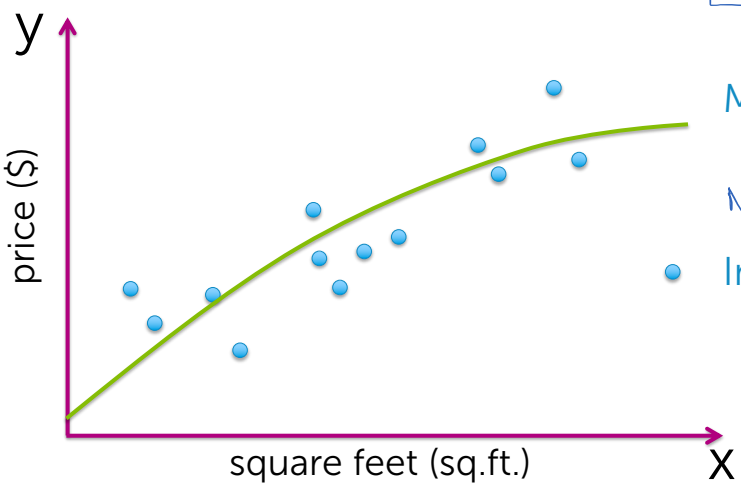
36

Assuming Gaussian noise

$N(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$

parameters





Model for ε_i:

So far: $E[\epsilon_i] = 0$

Now, stronger assumption: $\epsilon_i \sim N(0, \sigma^2)$

Implied distribution on y_i:

$y_i = h^T(x_i) w + \epsilon_i$

"given" const

$y_i | x_i; w \sim N(h^T(x_i) w, \sigma^2)$

37

Maximum likelihood estimate of params

$\arg\max_w p(\mathbf{y} | \mathbf{X}; w) \overset{\epsilon_i \text{ iid}}{=} \arg\max_w \prod_{i=1}^N p(y_i | x_i; w) = \arg\max_w \ln \{ \}$ ← strictly inc. fcn

Maximize log-likelihood wrt **w**

$$\arg\max_w \ln p(\mathbf{y} \mid \mathbf{X}; \mathbf{w}) = \ln \left(\frac{1}{\sigma\sqrt{2\pi}} \right)^N \prod_{i=1}^N e^{\frac{-(y_i - \mathbf{h}(\mathbf{x}_i)^\top \mathbf{w})^2}{2\sigma^2}}$$

$$= \arg\max_w \; N \ln(\cdot) - \sum_{i=1}^N \frac{(y_i - h^T(x_i) w)^2}{2\sigma^2}$$

no w dep.

$$= \arg\min_w \sum_{i=1}^N (y_i - h^T(x_i) w)^2$$ ← MLE for Gauss model = min RSS! (LS line)

38

Summary of linear regression

39

What you can do now…

- Describe the input (features) and output (real-valued predictions) of a regression model
- Write a regression model using multiple inputs or features thereof
- Cast both polynomial regression and regression with multiple inputs as regression with multiple features
- Calculate a cost metric (e.g., RSS)
- Estimate model parameters of a general multiple regression model to minimize RSS:
  - In closed form
  - Using an iterative gradient descent algorithm
- Exploit the estimated model to form predictions
- Relate minimizing RSS to maximizing likelihood of a Gaussian model

40

Assessing performance, Bias-variance tradeoff

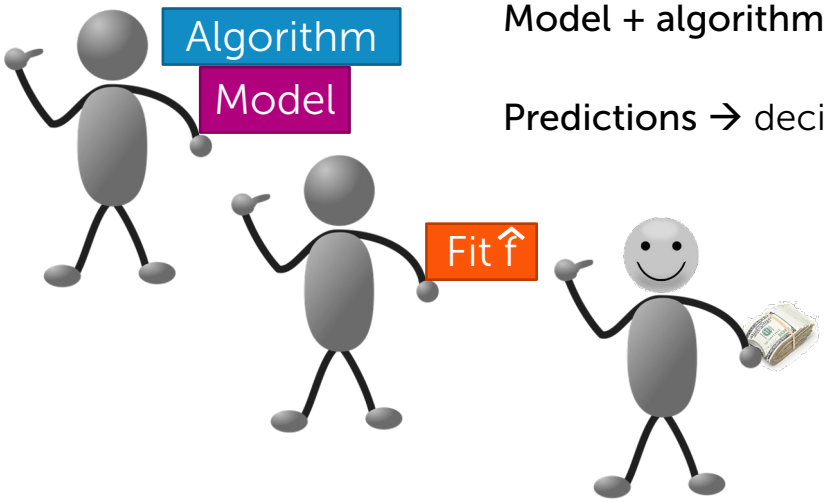
CS 229: Machine Learning
Emily Fox
Stanford University
January 10, 2024

41

Assessing performance

42

Make predictions, get $, right??



Model + algorithm → fitted function

Predictions → decisions → outcome

Or, how much am I losing?

Example: Lost $ due to inaccurate listing price

- Too low → low offers
- Too high → few lookers + no/low offers

How much am I losing compared to perfection?

Perfect predictions: Loss = 0

My predictions: Loss = ???

Measuring loss

Loss function:

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

Cost of using ŵ at x when y is true

actual value

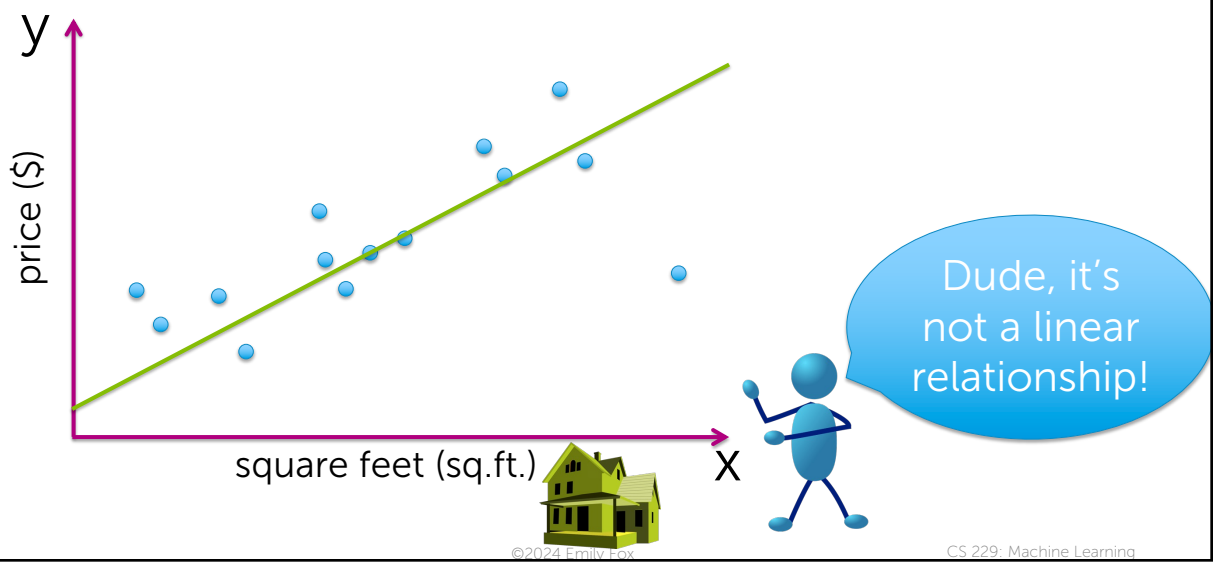
$\hat{f}(\mathbf{x})$ = predicted value ŷ

Examples: (assuming loss for underpredicting = overpredicting)

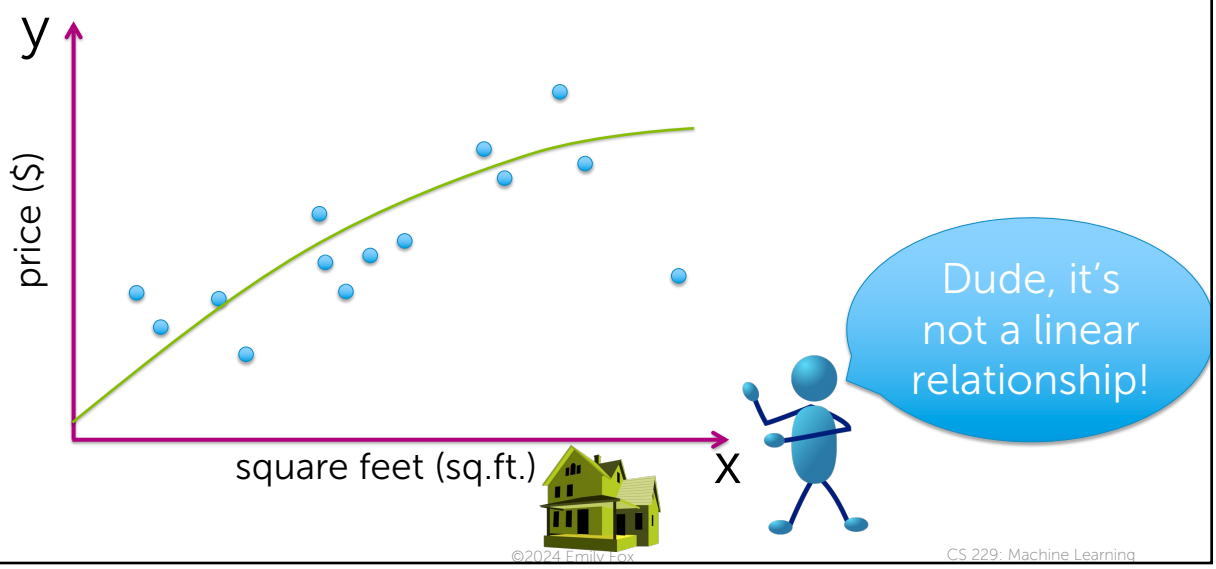
Absolute error: $L(y, f_{\hat{w}}(\mathbf{x})) = |y - f_{\hat{w}}(\mathbf{x})|$

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

Fit data with a line or … ?



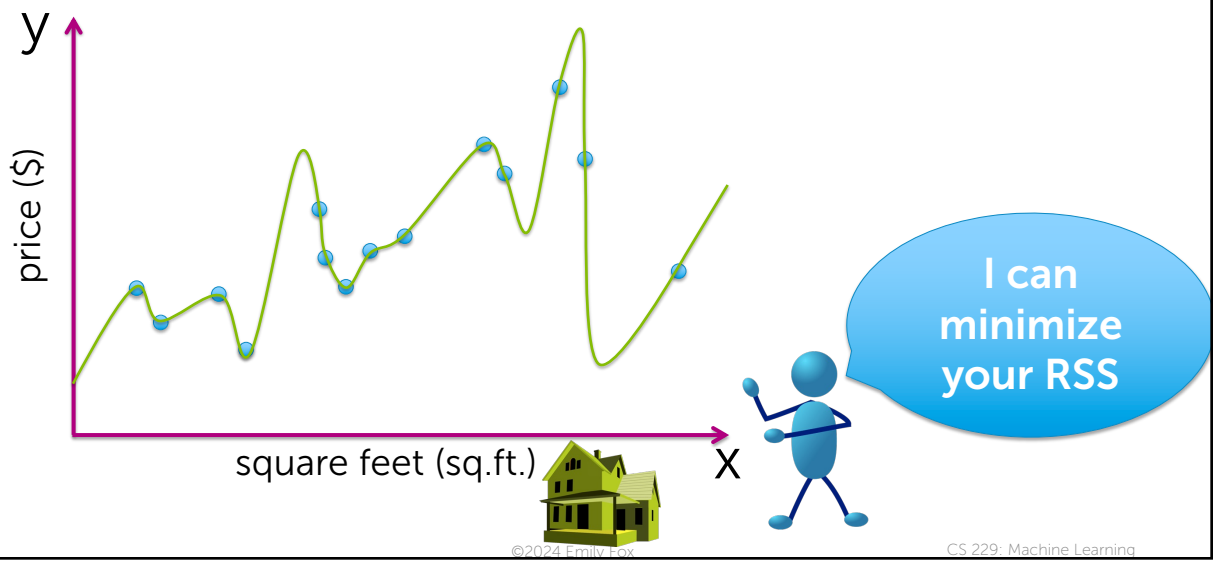
What about a quadratic function?



47

47

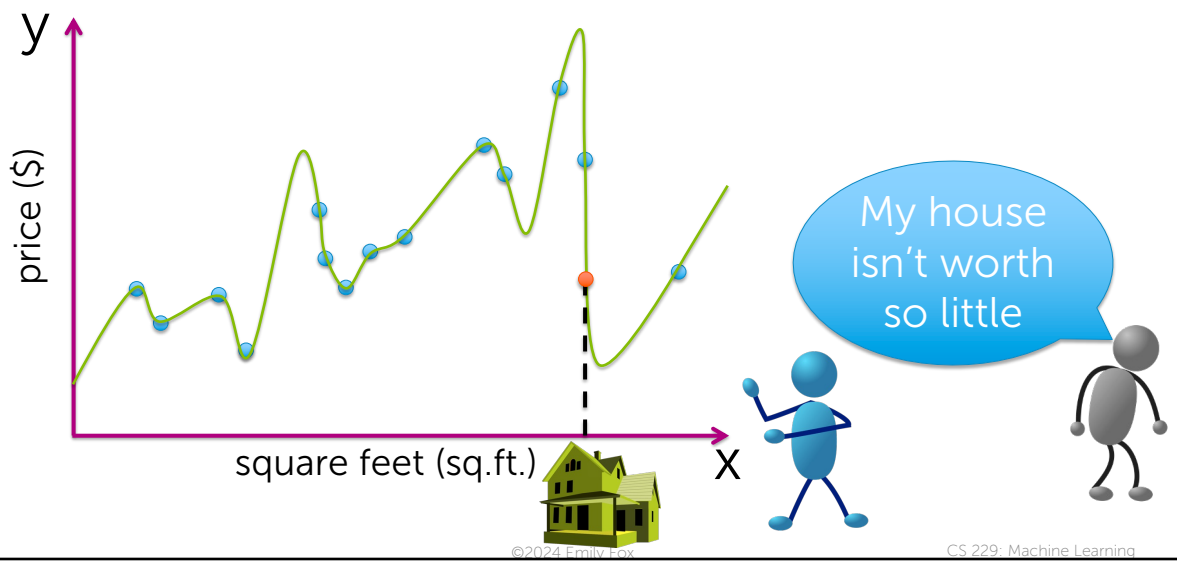
Even higher order polynomial



48

48

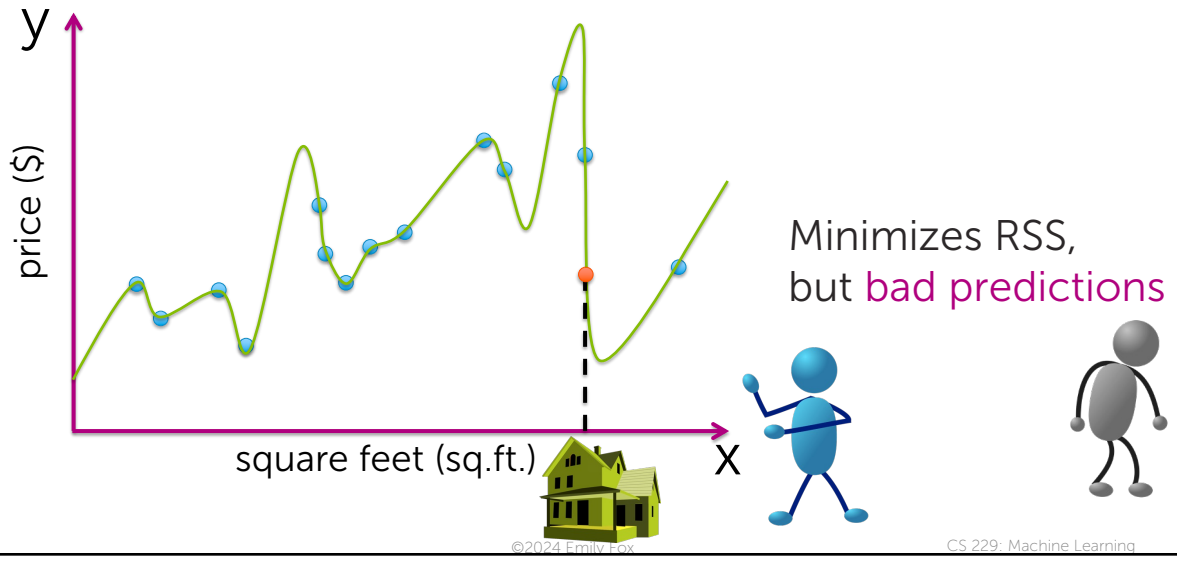
Do you believe this fit?



49

49

Do you believe this fit?



50

50

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful." George Box, 1987.

51

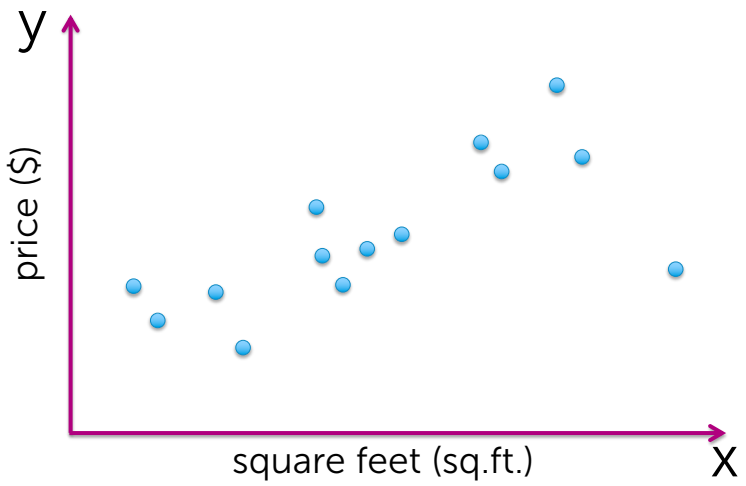
Assessing the loss

52

Assessing the loss
Part 1: Training error

53

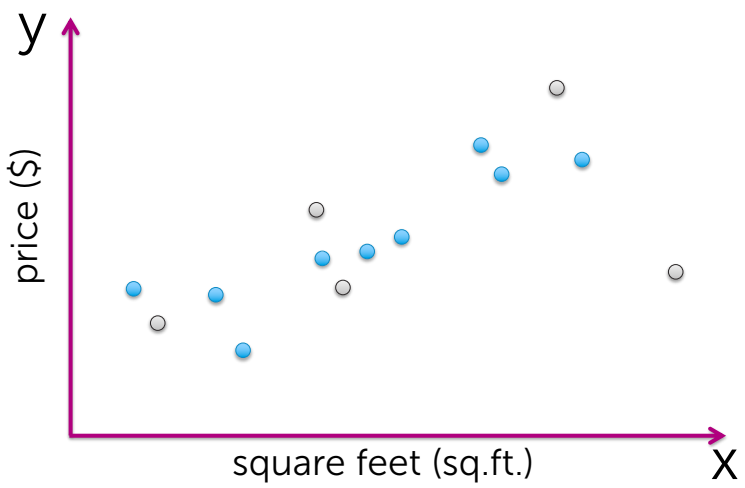
Define training data



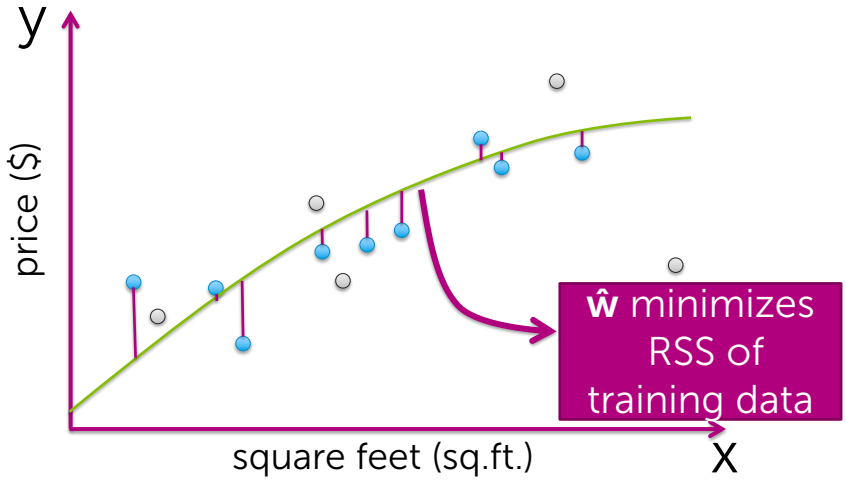
54

54

Define training data



Example: Fit quadratic to minimize RSS



Compute training error

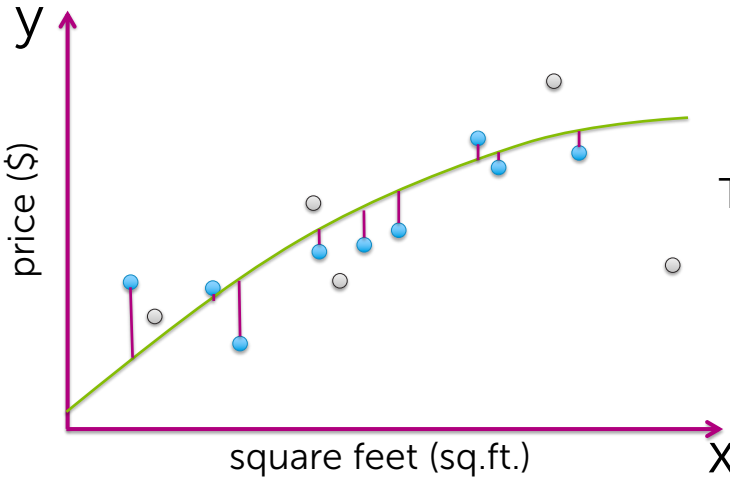
1. Define a loss function $L(y, f_{\hat{w}}(\mathbf{x}))$
   - E.g., squared error, absolute error,…

2. 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

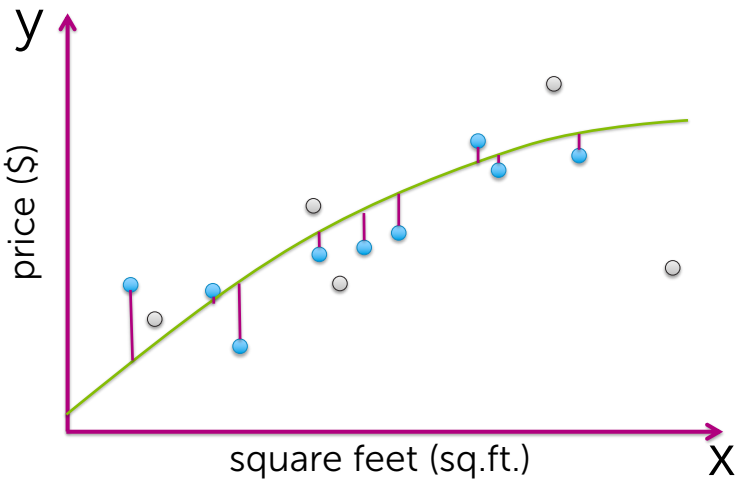
   $\hat{w}$ ← fit using training data

Example:
Use squared error loss $(y-f_{\hat{w}}(\mathbf{x}))^2$



Training error ($\hat{w}$) = 1/N *
[($\$_{train\ 1}$ - $f_{\hat{w}}(sq.ft._{train\ 1})$)^2
+ ($\$_{train\ 2}$ - $f_{\hat{w}}(sq.ft._{train\ 2})$)^2
+ ($\$_{train\ 3}$ - $f_{\hat{w}}(sq.ft._{train\ 3})$)^2
+ … include all training houses]

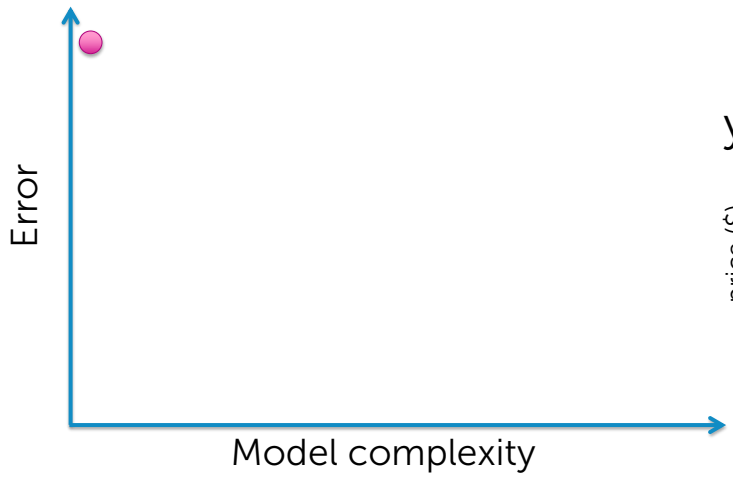
Example:
Use squared error loss $(y-f_{\hat{w}}(x))2$

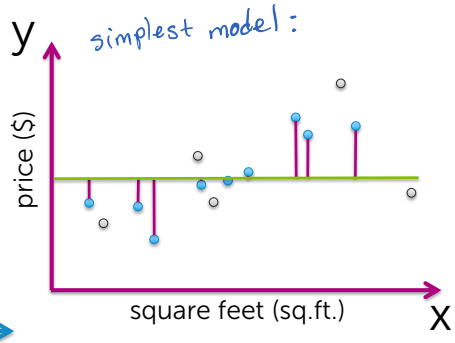


Training error ($\hat{w}$) = $\frac{1}{N}\sum_{i=1}^{N}(y_i-f_{\hat{w}}(x_i))^2$

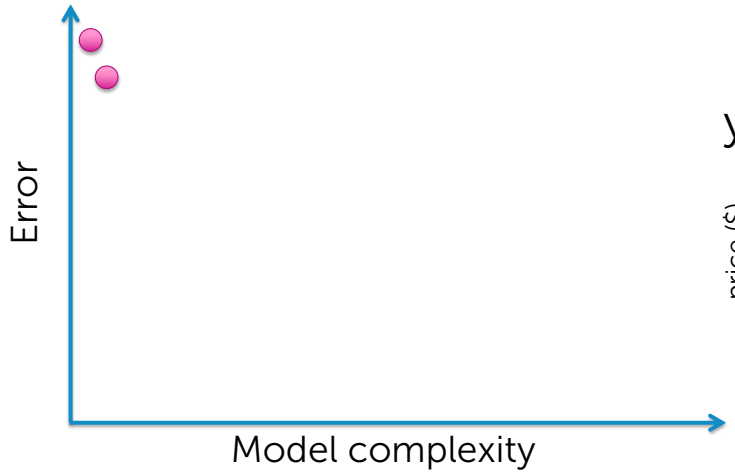
RMSE = $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-f_{\hat{w}}(x_i))^2}$

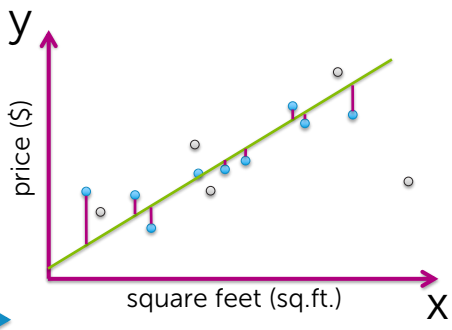
Training error vs. model complexity



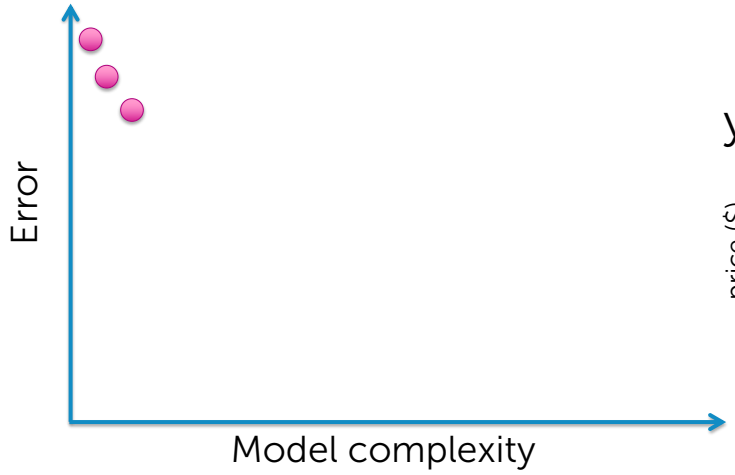


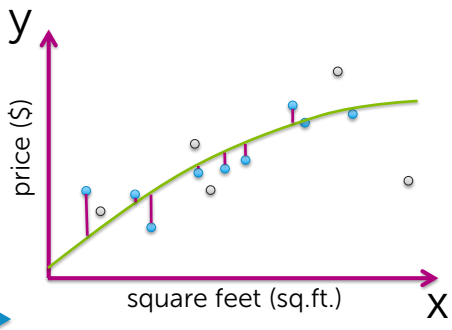
Training error vs. model complexity



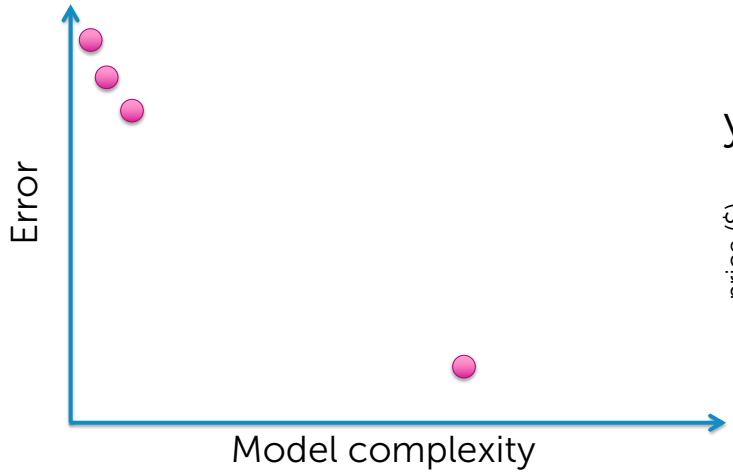


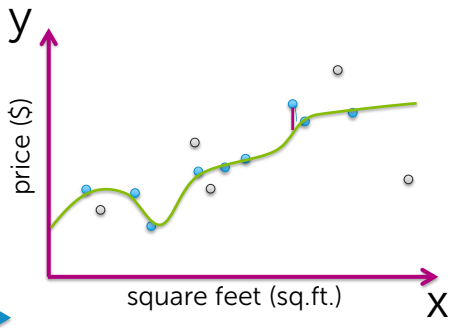
Training error vs. model complexity



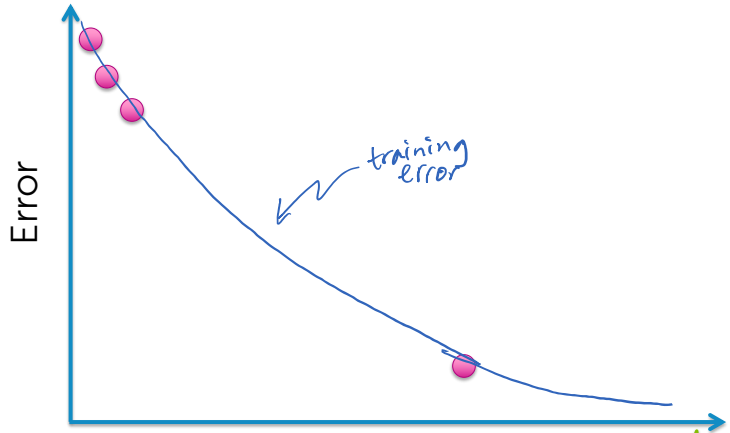


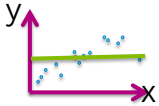
Training error vs. model complexity





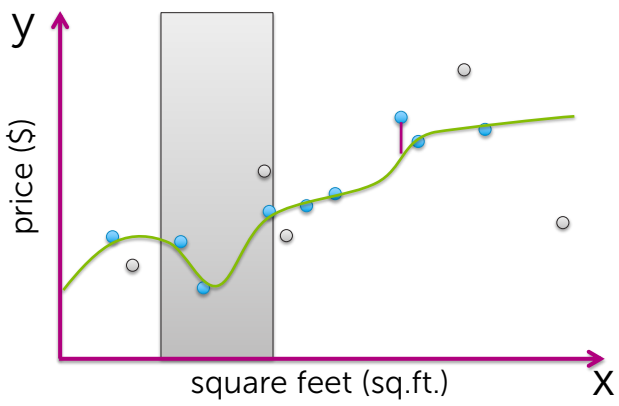
Training error vs. model complexity





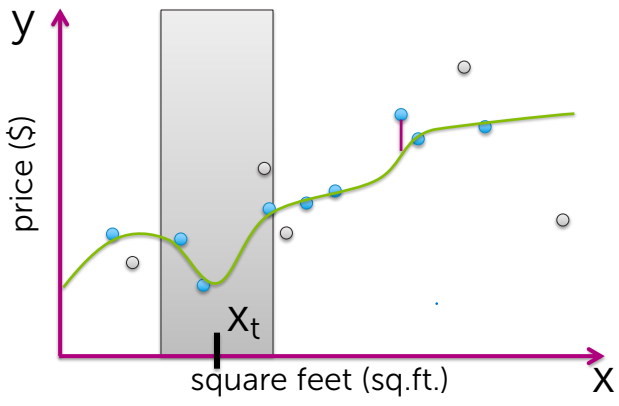
Is training error a good measure of predictive performance?

How do we expect to perform on a new house?



Is training error a good measure of predictive performance?

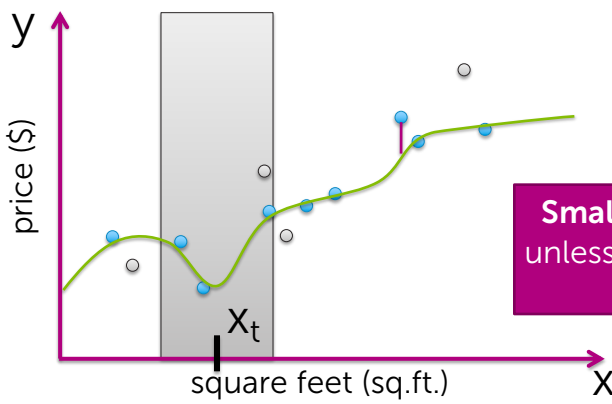
Is there something particularly bad about having x_t sq.ft.??



Is training error a good measure of predictive performance?

Issue:

Training error is overly optimistic…$\hat{\mathbf{w}}$ was fit to training data

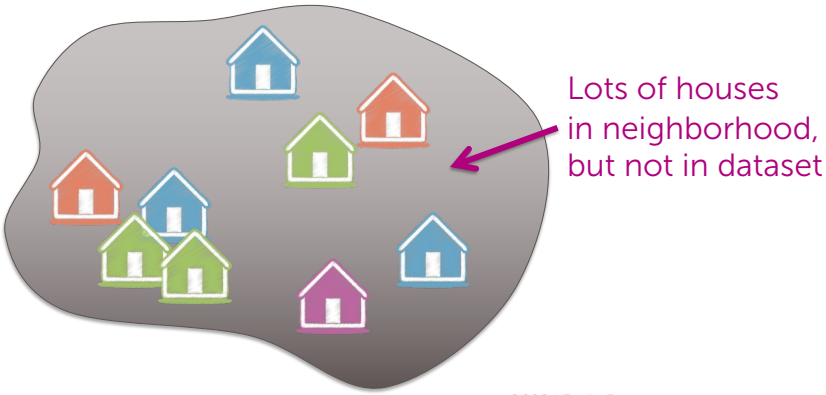


Small training error ≠> good predictions
unless training data includes everything you might ever see

Assessing the loss
Part 2: Generalization (true) error

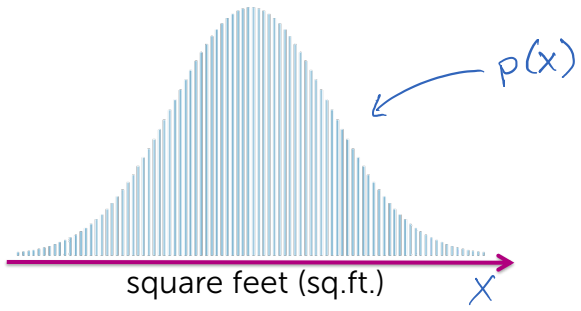
Generalization error

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



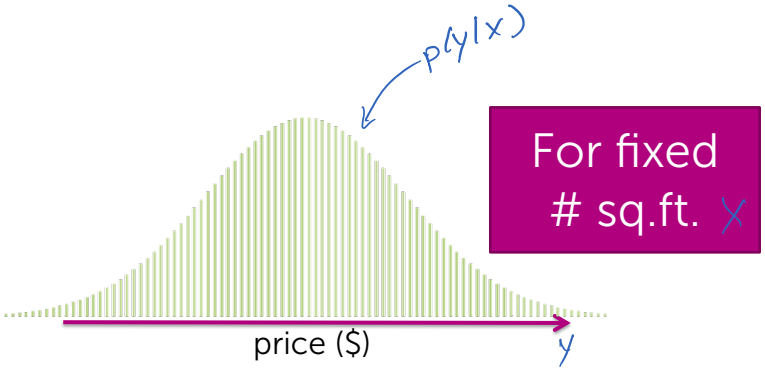
Distribution over houses

In our neighborhood, houses of what # sq.ft. (🏠)
are we likely to see?



Distribution over sales prices

For houses with a given # sq.ft. (🏠), what house prices $ are we likely to see?



Generalization error definition

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

Formally:

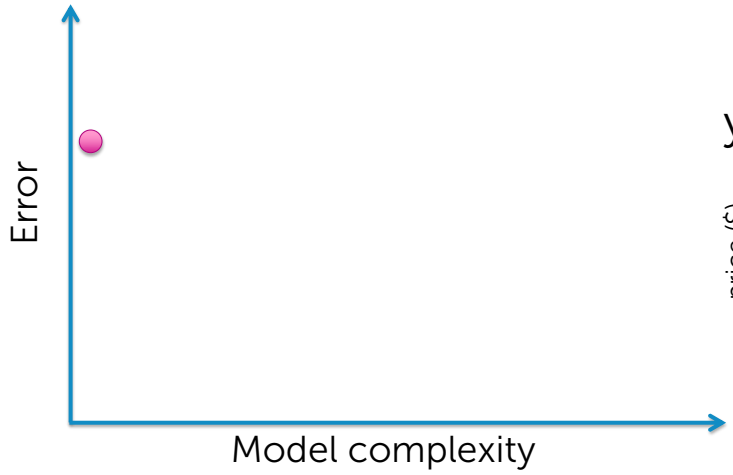
average over all possible (**x**,y) pairs weighted by how likely each is

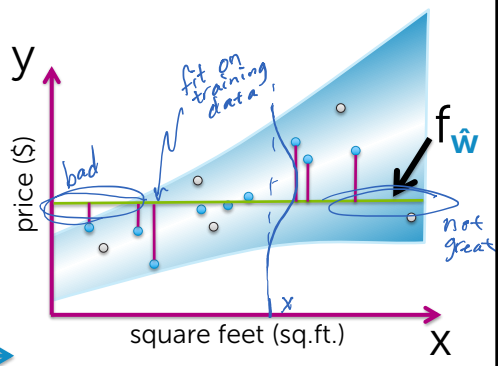
$$\text{generalization error} = E_{\mathbf{x},y}[L(y, f_{\hat{w}}(\mathbf{x}))]$$

fit using training data

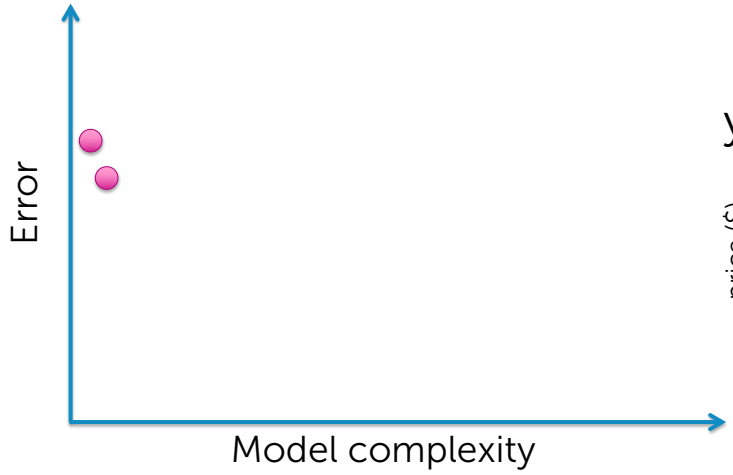
$$= \int L(y, f_{\hat{w}}(x)) \underbrace{p(x,y)}_{p(y|x)p(x)} dx\,dy$$

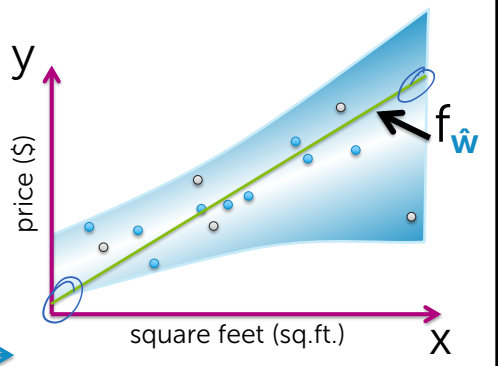
Generalization error vs. model complexity



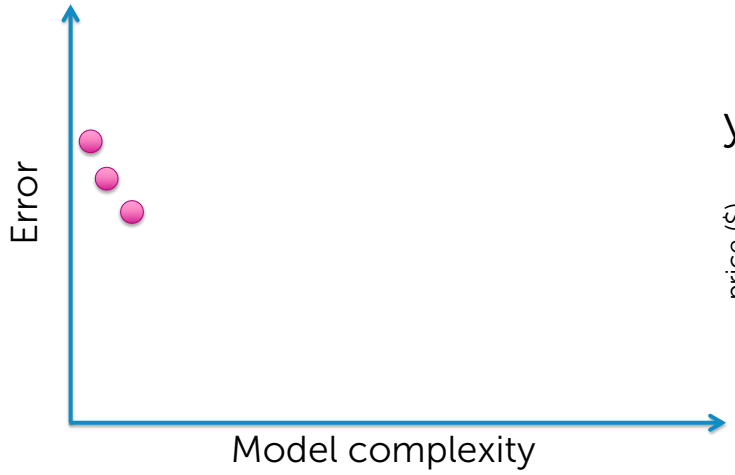


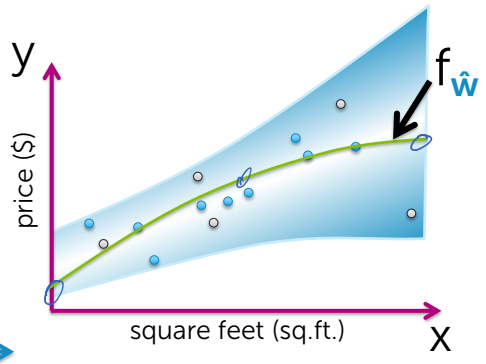
Generalization error vs. model complexity



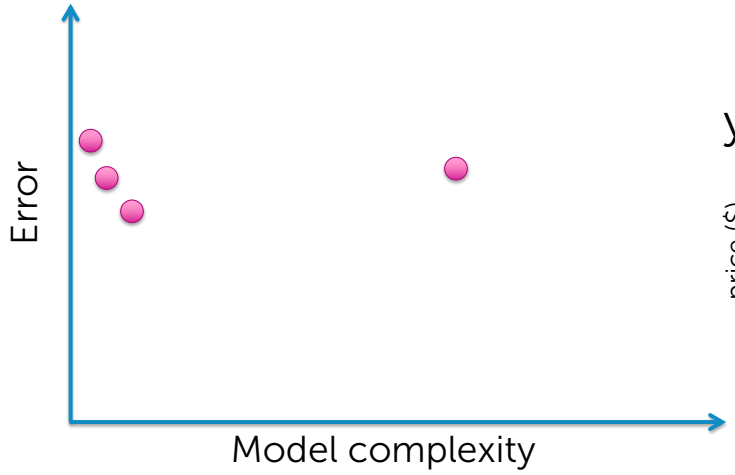


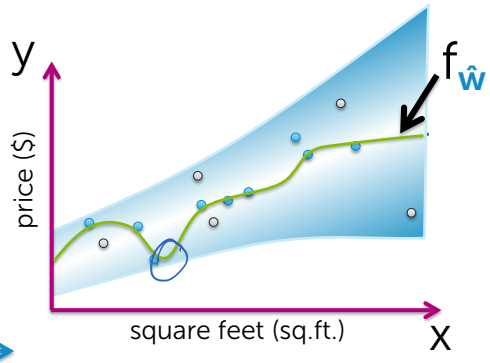
Generalization error vs. model complexity



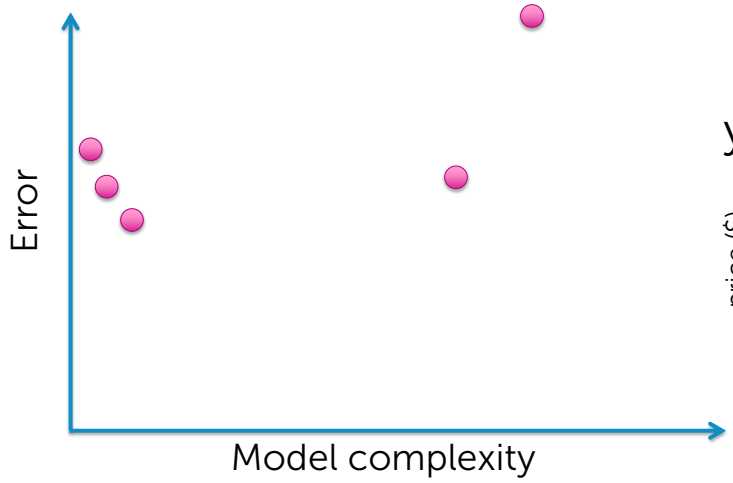


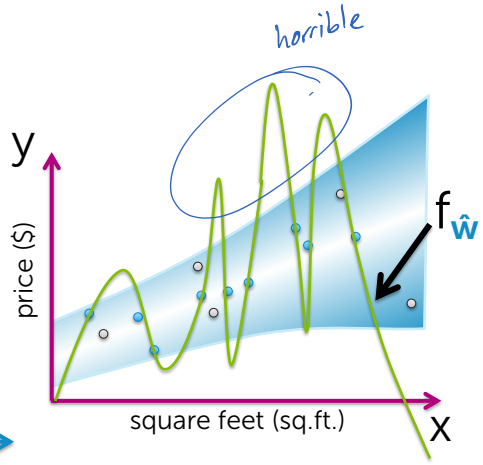
Generalization error vs. model complexity



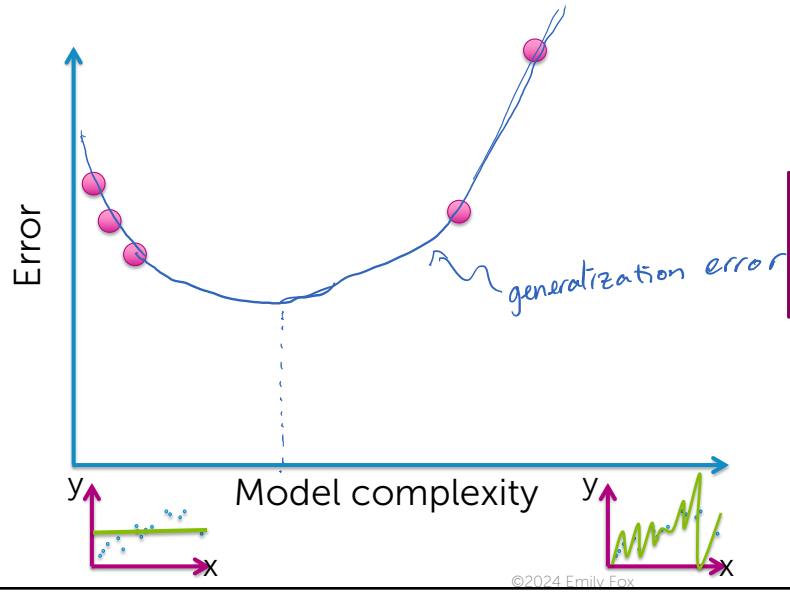


Generalization error vs. model complexity





Generalization error vs. model complexity



Can't compute!

Assessing the loss
Part 3: Test error

Approximating generalization error

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

Approximate by looking at houses not in training set

Forming a test set

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

Training set

Test set

Forming a test set

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

Proxy for "everything you might see"

Test set

Compute test error

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

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

$p(y|x)\, p(x)$

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

Test error

= avg. loss on houses in test set

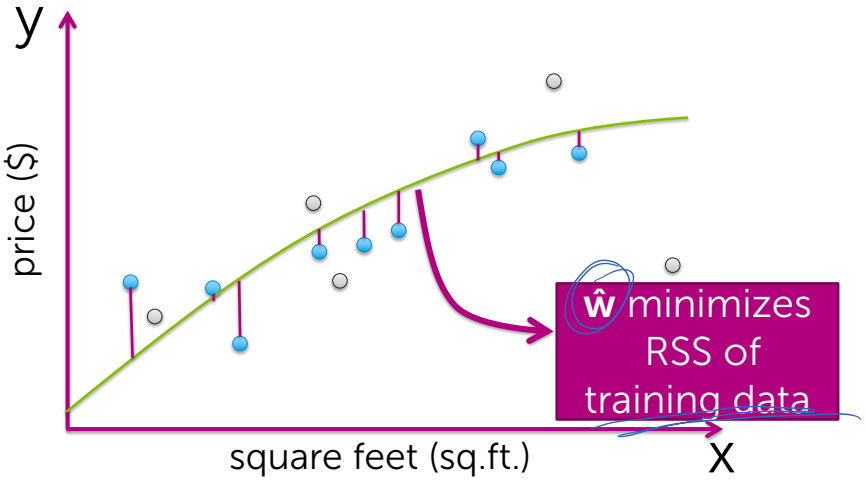
$$= \frac{1}{N_{test}} \sum_{i \text{ in test set}} L(y_i, f_{\hat{w}}(x_i))$$

test points

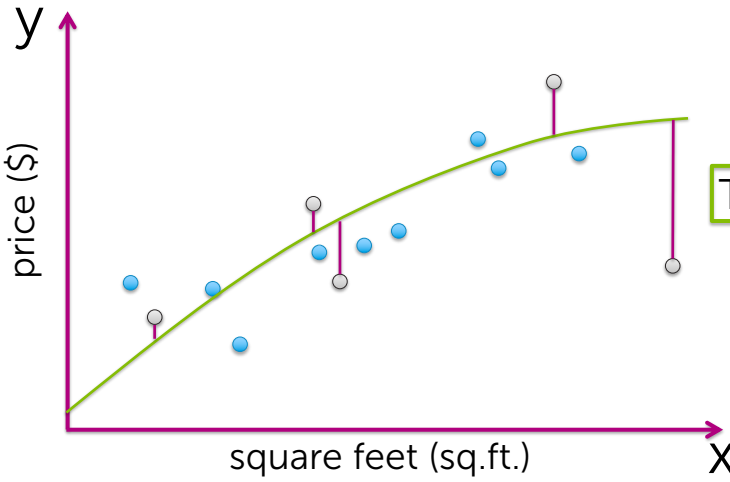
fit using training data

has never seen test data!

Example:
As before, fit quadratic to training data

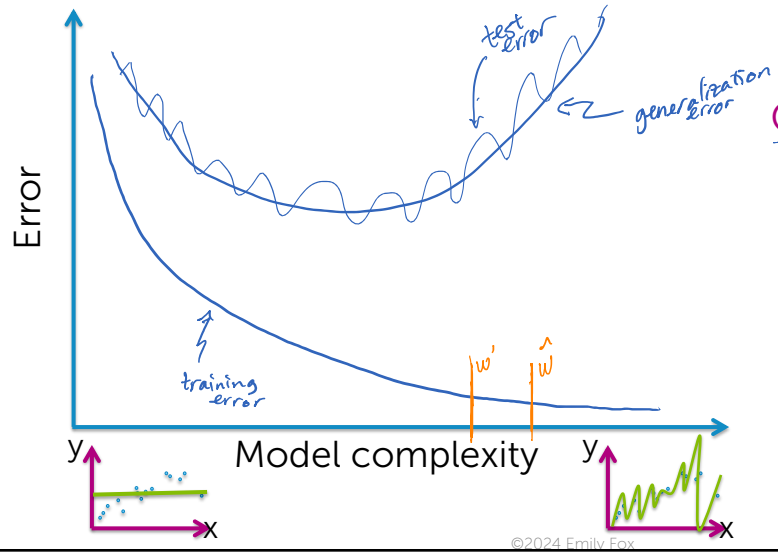


Example:
As before, use squared error loss $(y-f_{\hat{w}}(x))2$



Test error ($\hat{w}$) = $1/N_{test}$ *
$[(\$_{test\ 1}-f_{\hat{w}}(sq.ft._{test\ 1}))^2$
$+ (\$_{test\ 2}-f_{\hat{w}}(sq.ft._{test\ 2}))^2$
$+ (\$_{test\ 3}-f_{\hat{w}}(sq.ft._{test\ 3}))^2$
+ … include all test houses]

Training, true, & test error vs. model complexity



Overfitting if: using $\hat{w}$

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

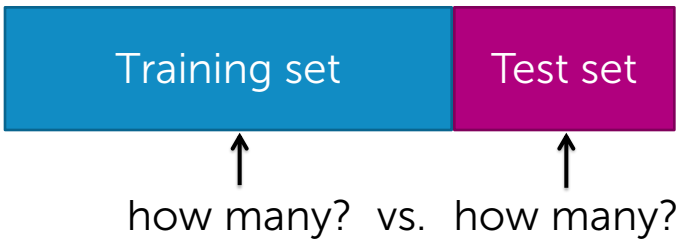
(1) training error ($\hat{w}$) < training error (w')

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

Training/test split

87

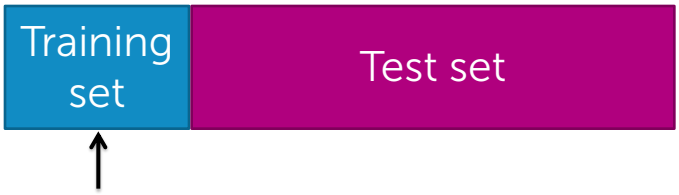
Training/test splits



88

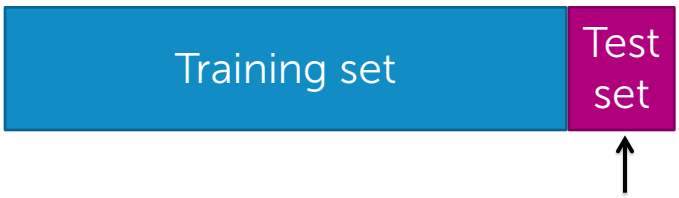
88

Training/test splits



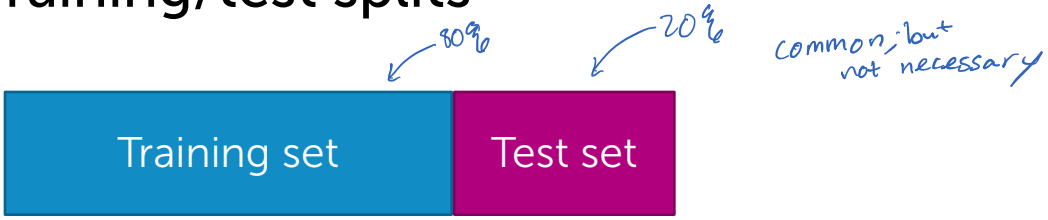
Too few → $\hat{w}$ poorly estimated

Training/test splits



Too few → test error bad approximation of generalization error

Training/test splits



Typically, just enough test points to form a reasonable estimate of generalization error

If this leaves too few for training, other methods like **cross validation**

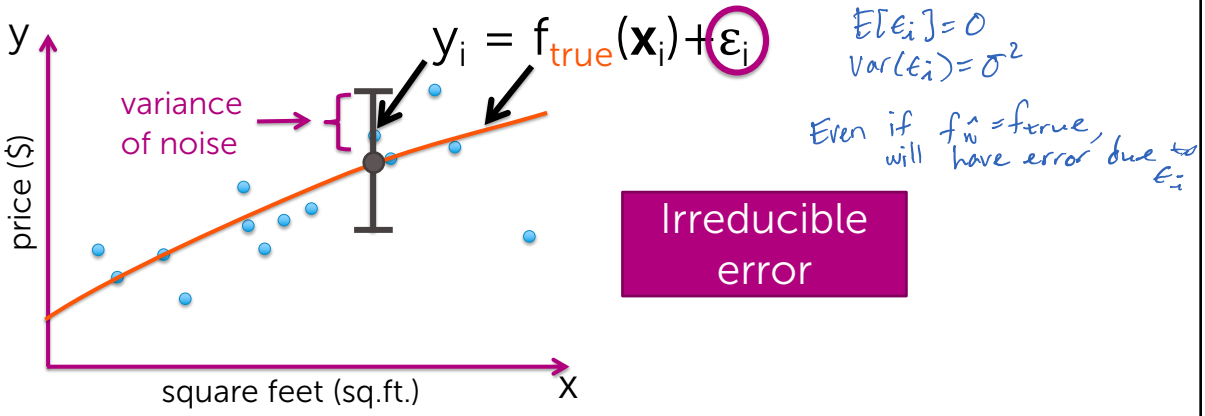
3 sources of error + the bias-variance tradeoff

3 sources of error

In forming predictions, there are 3 sources of error:

1. Noise
2. Bias
3. Variance

Data inherently noisy



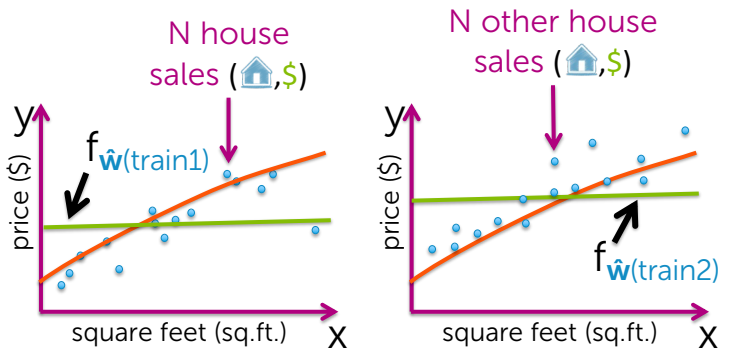
$E[\epsilon_i] = 0$
$Var(\epsilon_i) = \sigma^2$

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

Irreducible error

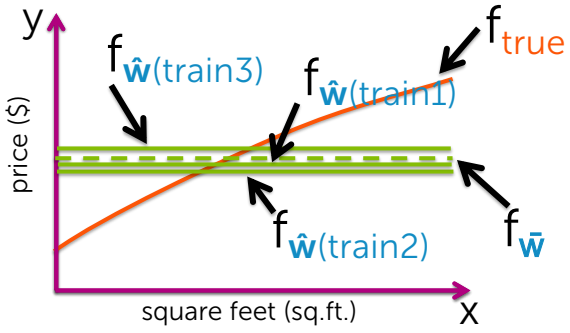
Bias contribution

Assume we fit a constant function



Bias contribution

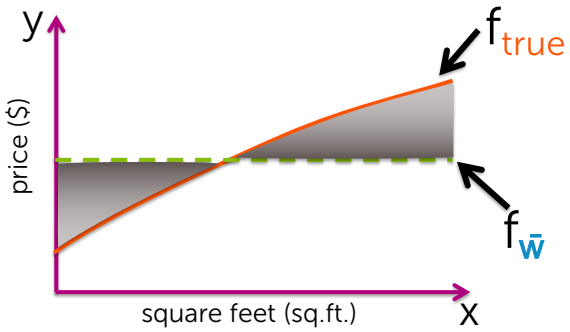
Over all possible size N training sets, what do I expect my fit to be?



Bias contribution

$\text{Bias}(\mathbf{x}) = f_{true}(\mathbf{x}) - f_{\bar{\mathbf{w}}}(\mathbf{x})$

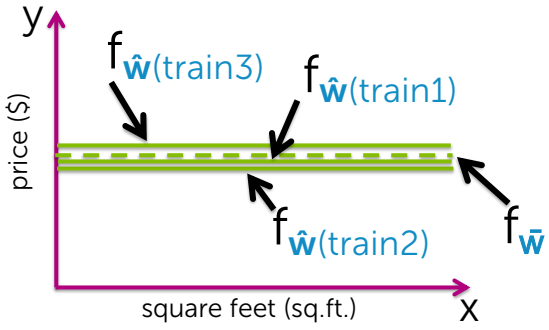
Is our approach flexible enough to capture f_{true}?
If not, error in predictions.



low complexity → high bias

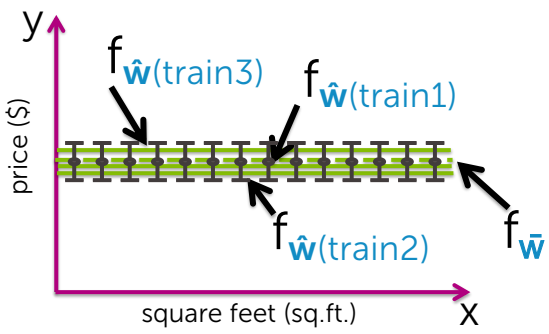
Variance contribution

How much do specific fits vary from the expected fit?



Variance contribution

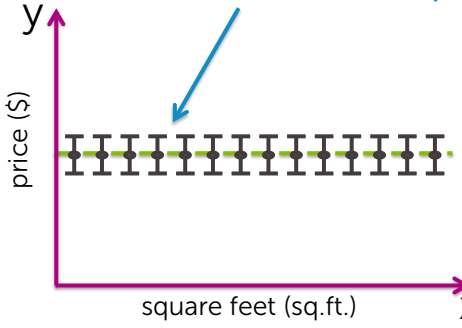
How much do specific fits vary from the expected fit?



Variance contribution

How much do specific fits vary from the expected fit?

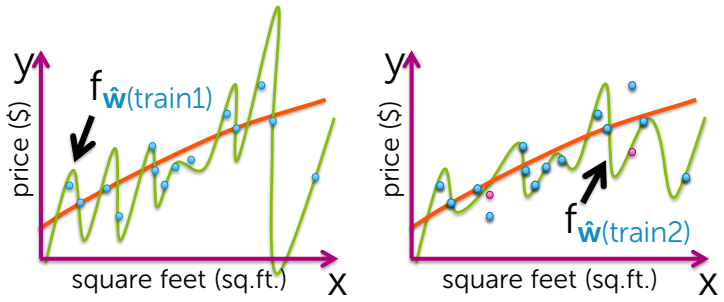
Can specific fits vary widely?
If so, erratic predictions



low complexity
→
low variance

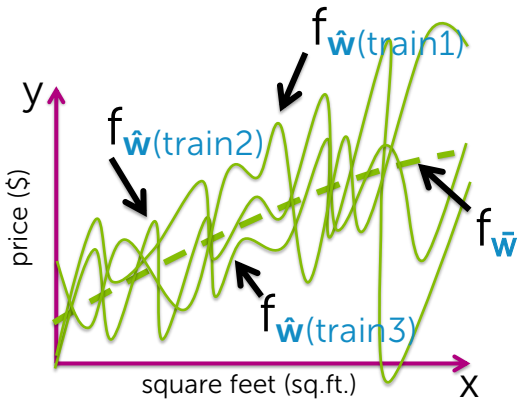
Variance of high-complexity models

Assume we fit a high-order polynomial

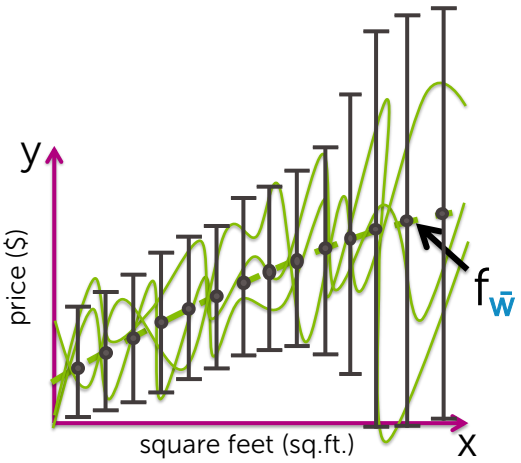


Variance of high-complexity models

Assume we fit a high-order polynomial

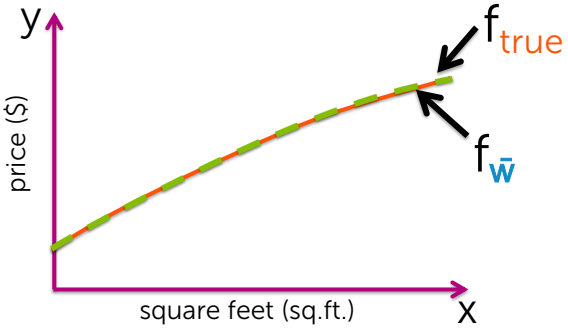


Variance of high-complexity models



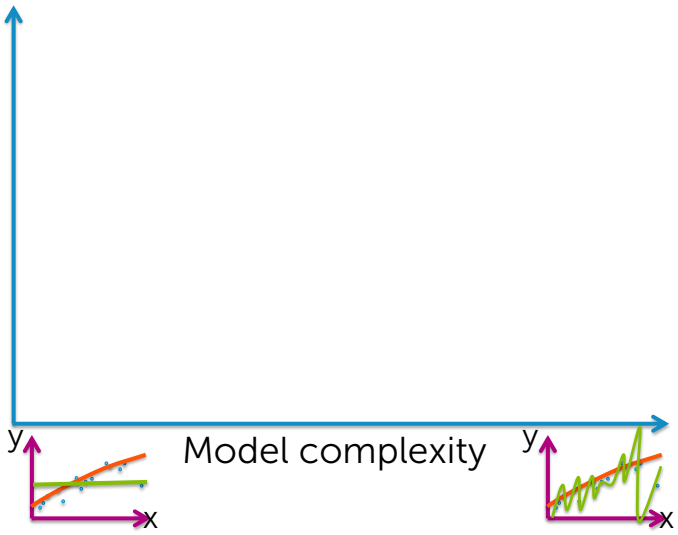
high complexity
→
high variance

Bias of high-complexity models

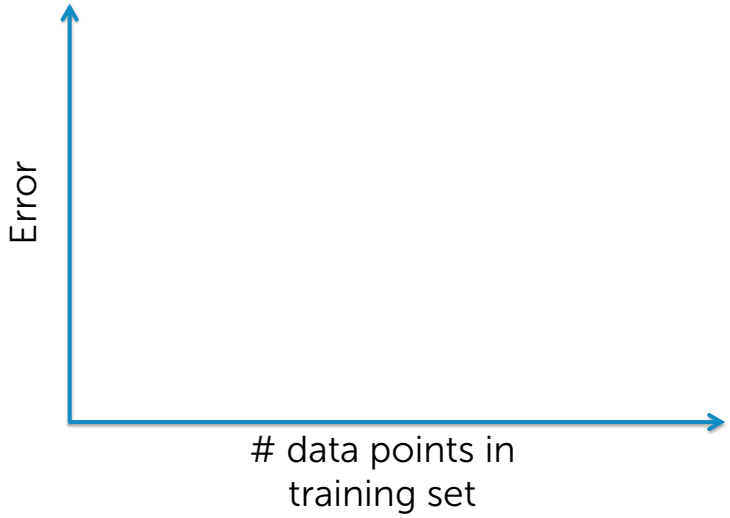


high complexity
→
low bias

Bias-variance tradeoff



Error vs. amount of data



Why 3 sources of error?
A formal derivation

107

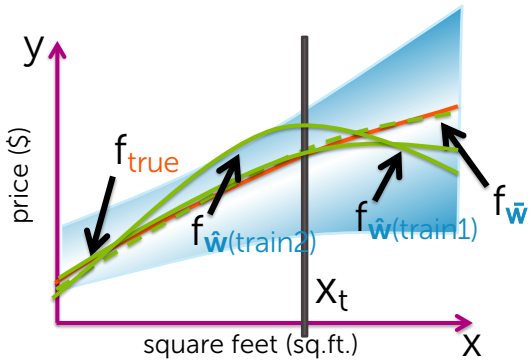
Bias of function estimator, formally

Average estimated function = $f_{\bar{\mathbf{w}}}(\mathbf{x})$

True function = $f_{true}(\mathbf{x})$

$E_{train}[f_{\hat{\mathbf{w}}(train)}(\mathbf{x})]$

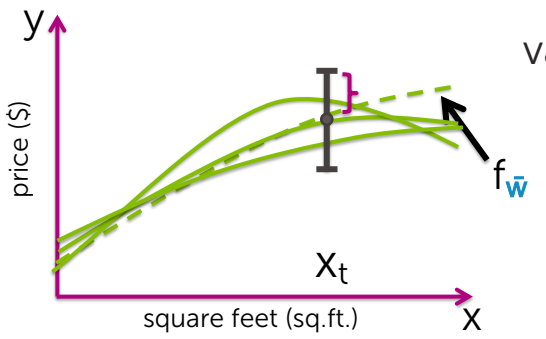
over all training sets of size N



$$bias(f_{\hat{\mathbf{w}}}(\mathbf{x}_t)) = f_{true}(\mathbf{x}_t) - f_{\bar{\mathbf{w}}}(\mathbf{x}_t)$$

108

Variance of function estimator, formally



fit on a specific training dataset

what I expect to learn over all training sets

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

over all training sets of size N

deviation of specific fit from expected fit at x_t

Deriving expected prediction error

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

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

Expected prediction error at x_t
= $E_{train}[E_{y_t|x_t}[(y_t - f_{\hat{w}(train)}(x_t))^2]]$

Deriving expected prediction error

Expected prediction error at $\mathbf{x}_t$

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

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

Equating MSE with bias and variance

$MSE[f_{\hat{\mathbf{w}}(train)}(\mathbf{x}_t)]$

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

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

Putting it all together

Expected prediction error at $\mathbf{x}_t$

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

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

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