Regression: 

Predicting House Prices

STAT/CSE 416: Intro to Machine Learning

Hunter Schafer (slides by Emily Fox)
University of Washington

April 5, 2018

Generic linear regression model

Model:

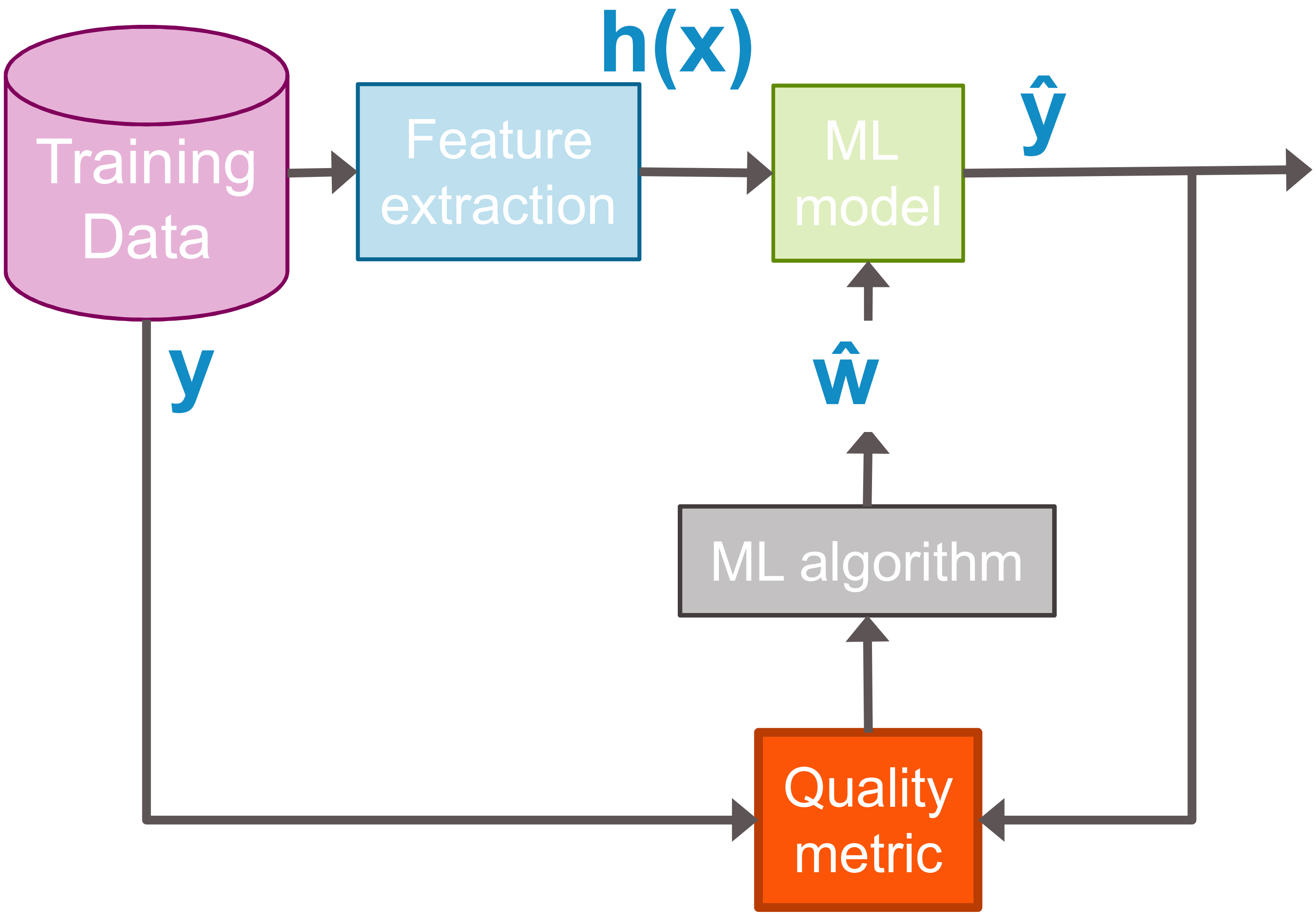
$$y_i = w_0\, h_0(x_i) + w_1\, h_1(x_i) + \ldots + w_D\, h_D(x_i) + \varepsilon_i$$

$$= \sum_{j=0}^{D} w_j\, h_j(x_i) + \varepsilon_i$$

feature 1 = $h_0(x)$ … e.g., 1
feature 2 = $h_1(x)$ … e.g., x[1] = sq. ft.
feature 3 = $h_2(x)$ … e.g., x[2] = #bath
or, log(x[7]) x[2] = log(#bed) x #bath

…

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


Training Data
Feature extraction
h(x)
ML model
ŷ
y
ŵ
ML algorithm
Quality metric

Measuring loss

Loss function:

$$L(\underbrace{y}_{\text{actual value}},\underbrace{f_{\hat{w}}(x)}_{\hat{f}(x) = \text{predicted value } \hat{y}})$$

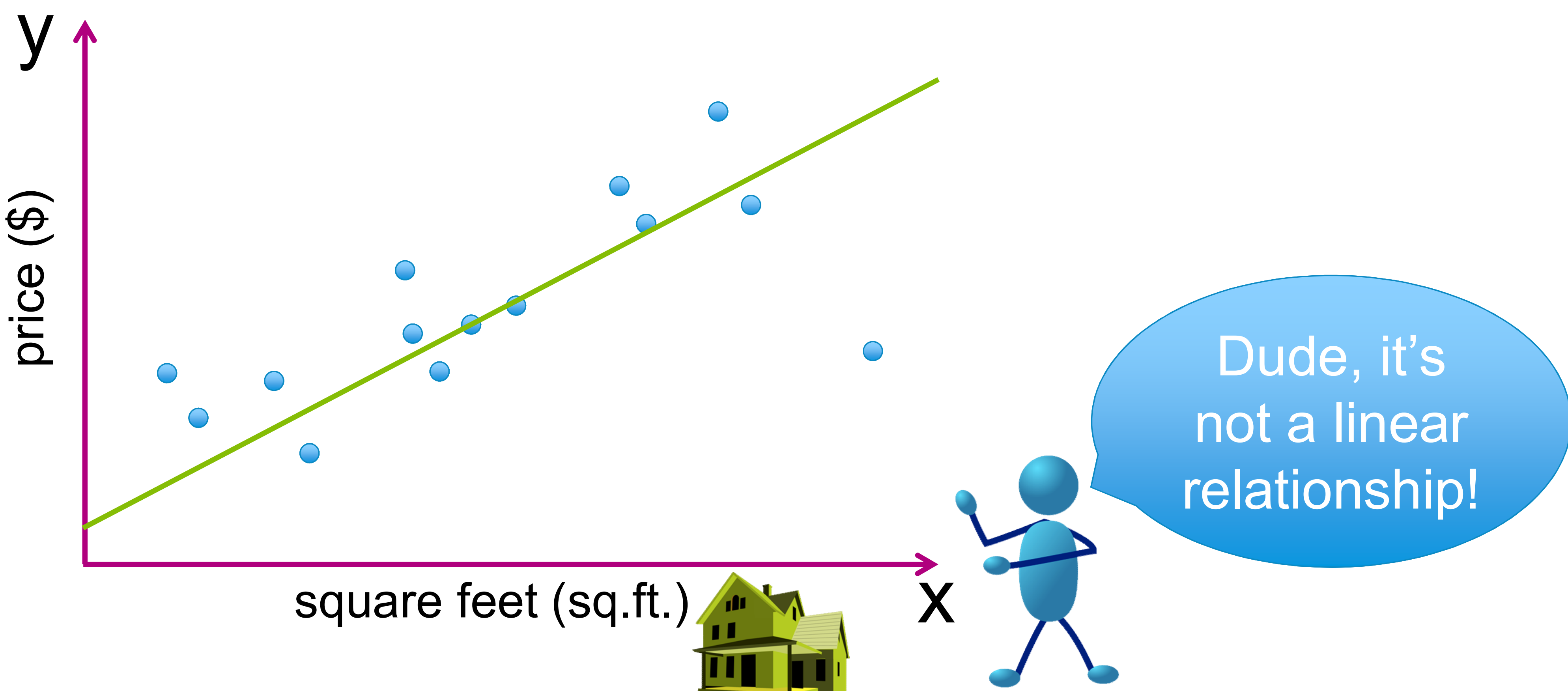
Cost of using $\hat{w}$ at x when y is true

Examples: (assuming loss for underpredicting = overpredicting)

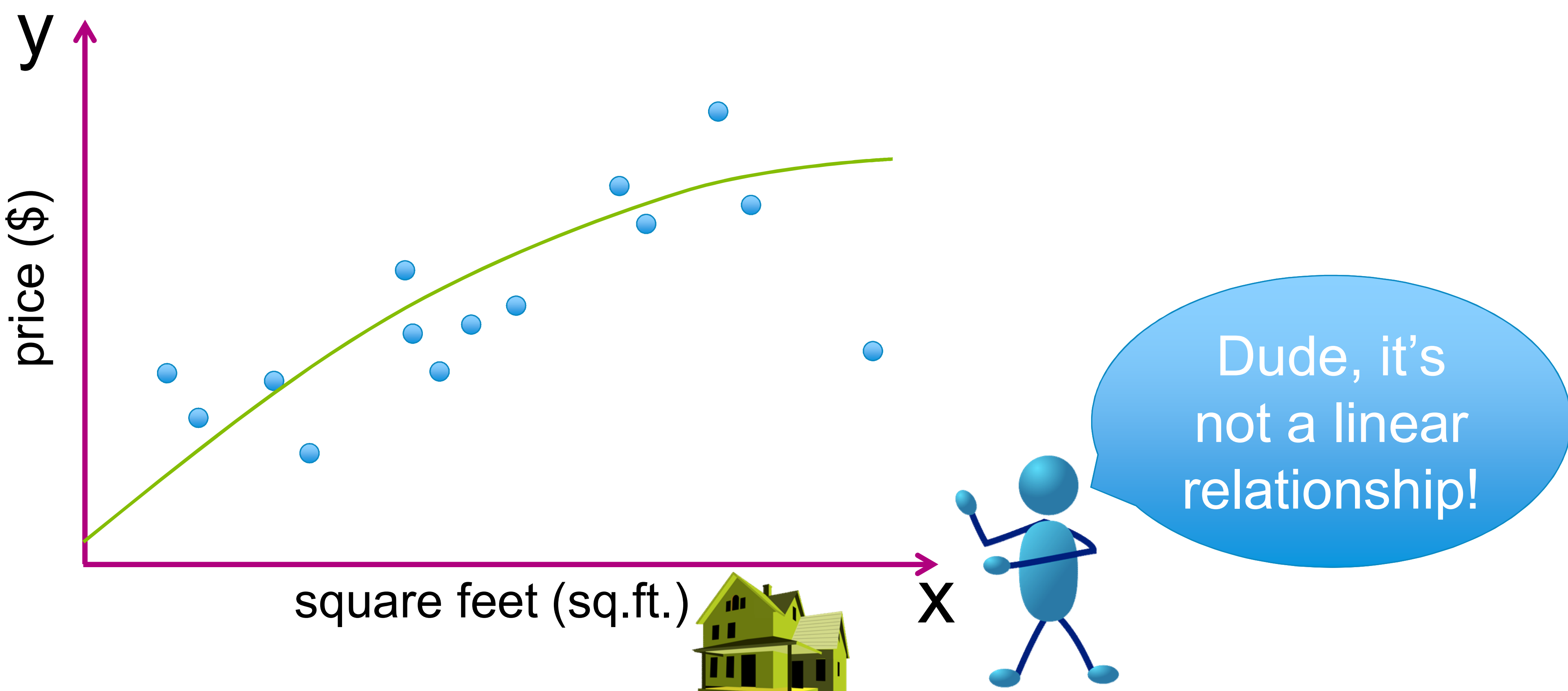
Absolute error: $L(y,f_{\hat{w}}(x)) = |y-f_{\hat{w}}(x)|$

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

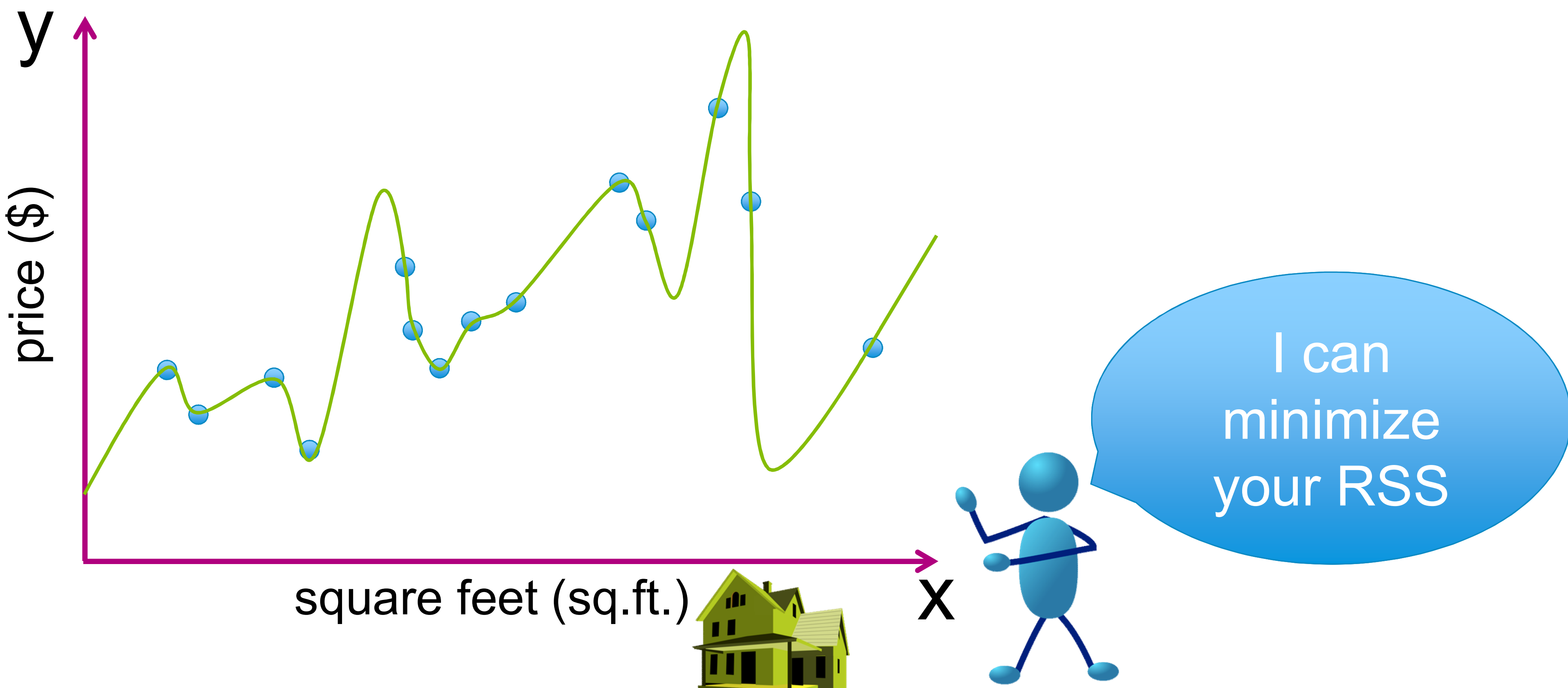
Fit data with a line or … ?



What about a quadratic function?

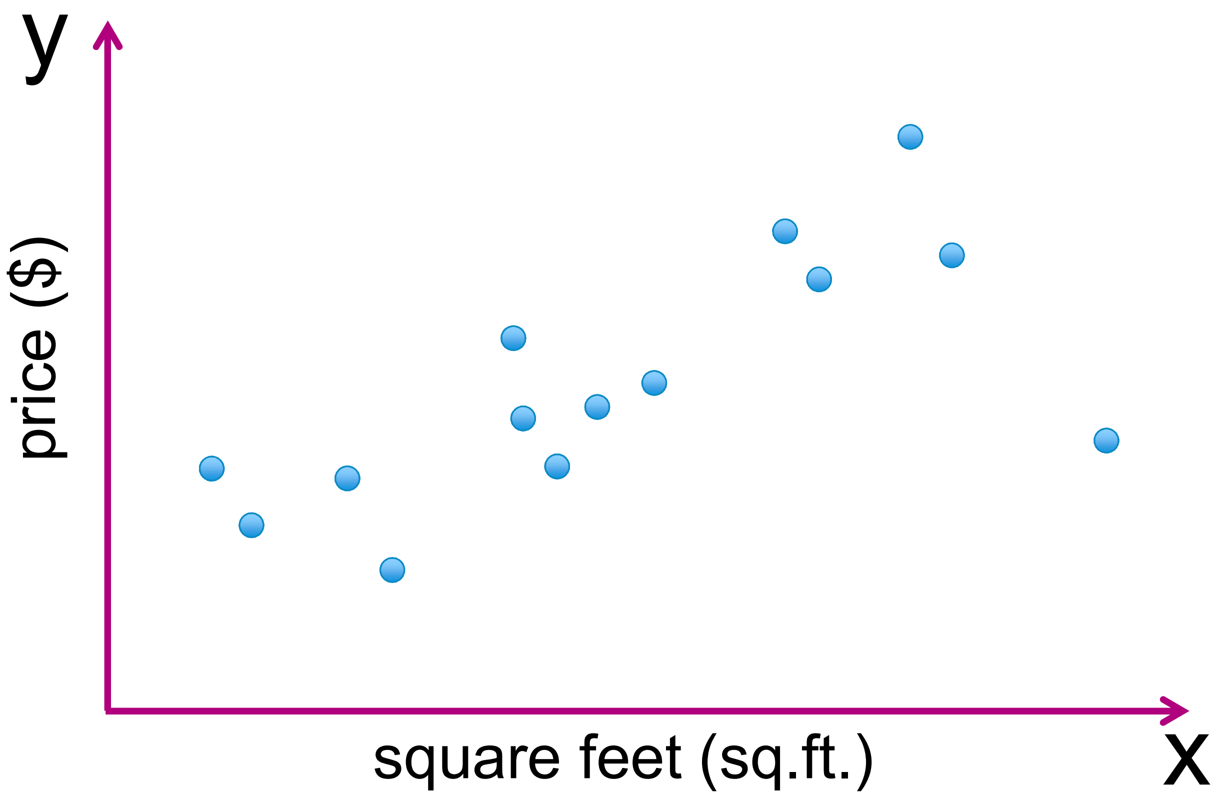


Even higher order polynomial

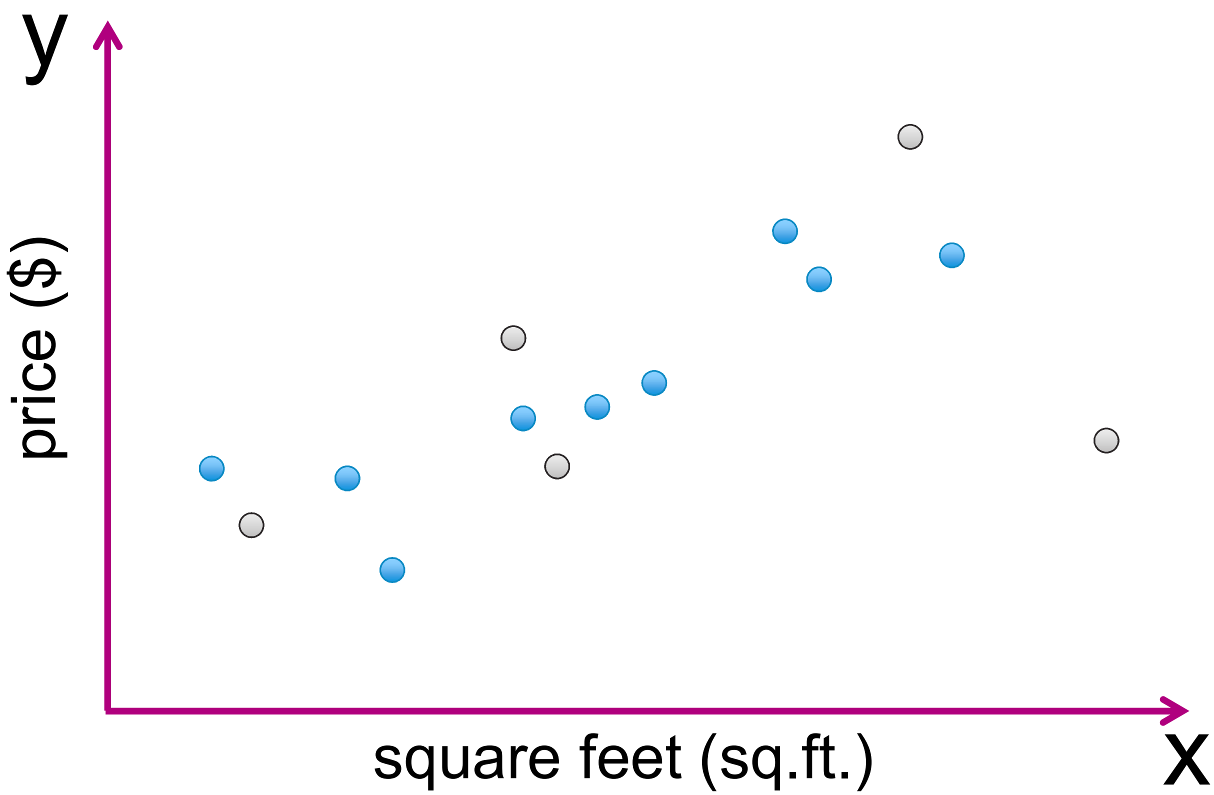


Assessing the loss
Part 1: Training error

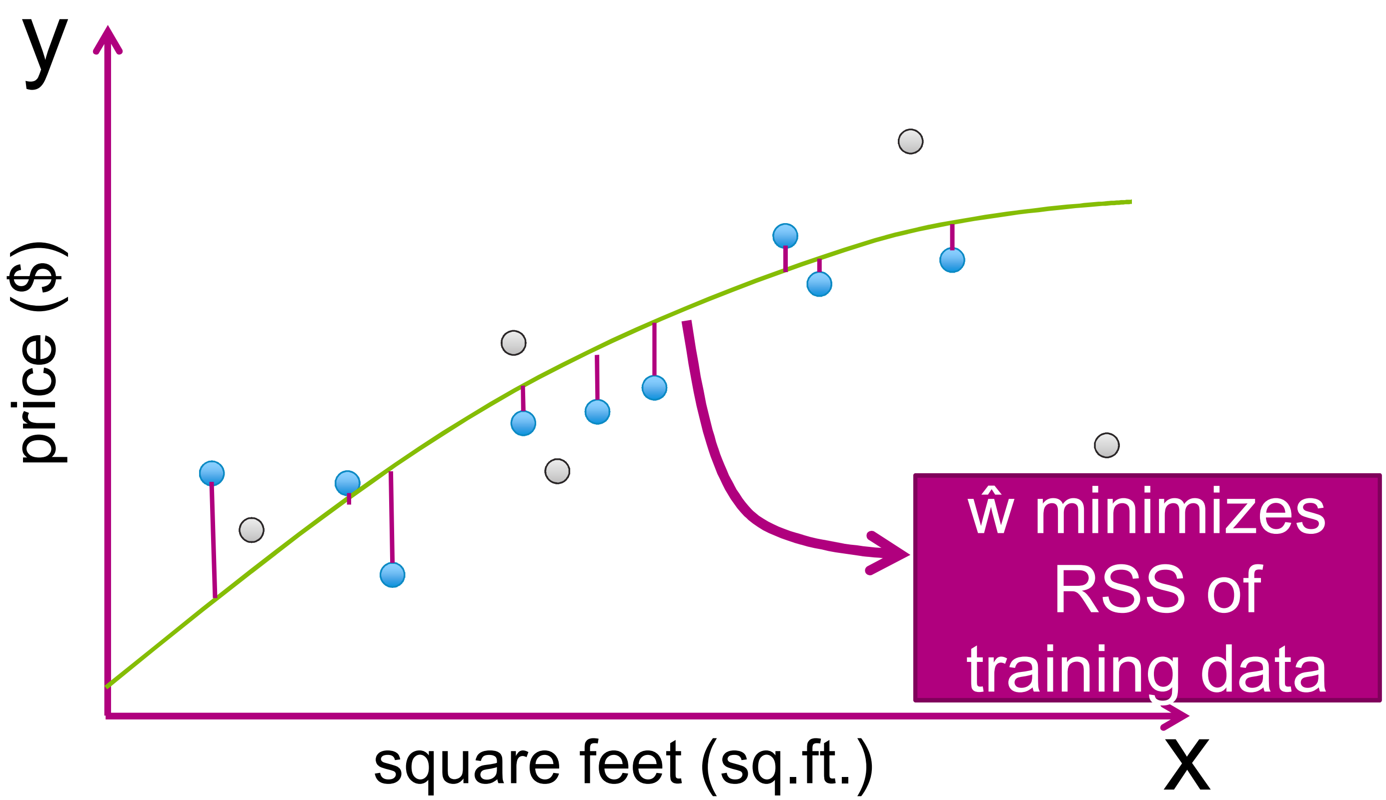
Define training data



Define training data



Example:
Fit quadratic to minimize RSS



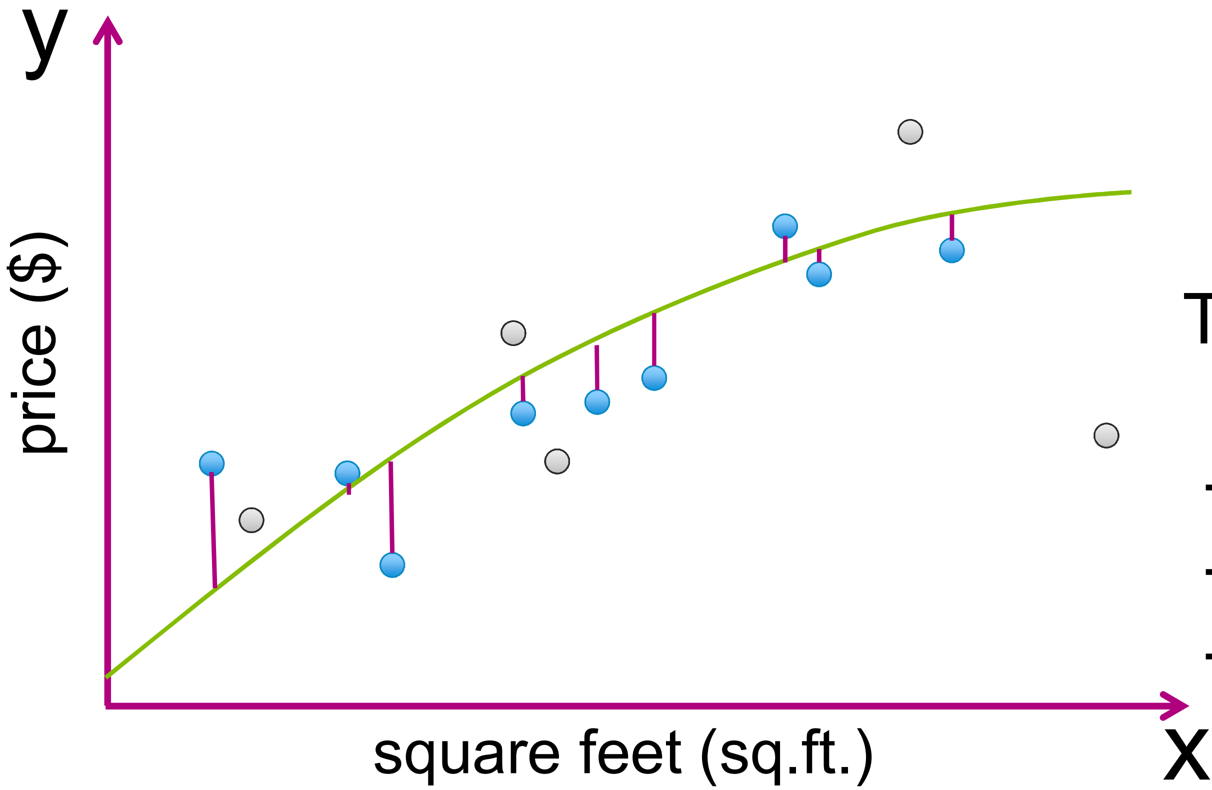
Compute training error

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

2. Training error
   = avg. loss on houses in training set
   $$= \frac{1}{N}\sum_{i=1}^{N} L(y_i,f_{\hat{w}}(x_i))$$

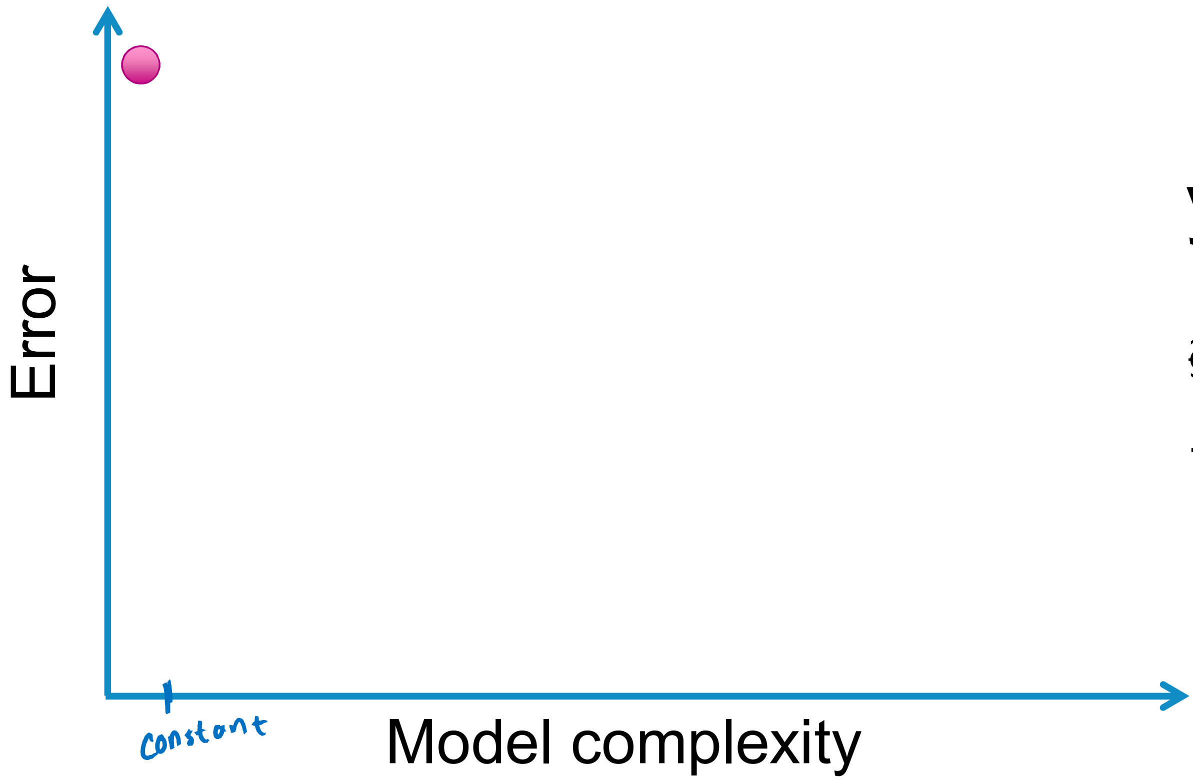
fit using training data

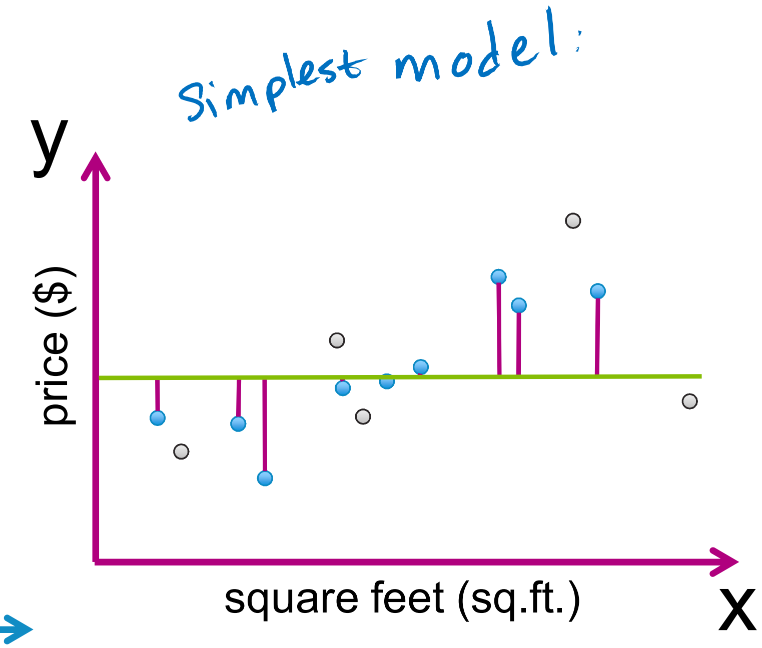
Example:
Use squared error loss $(y-f_{\hat{w}}(x))2$



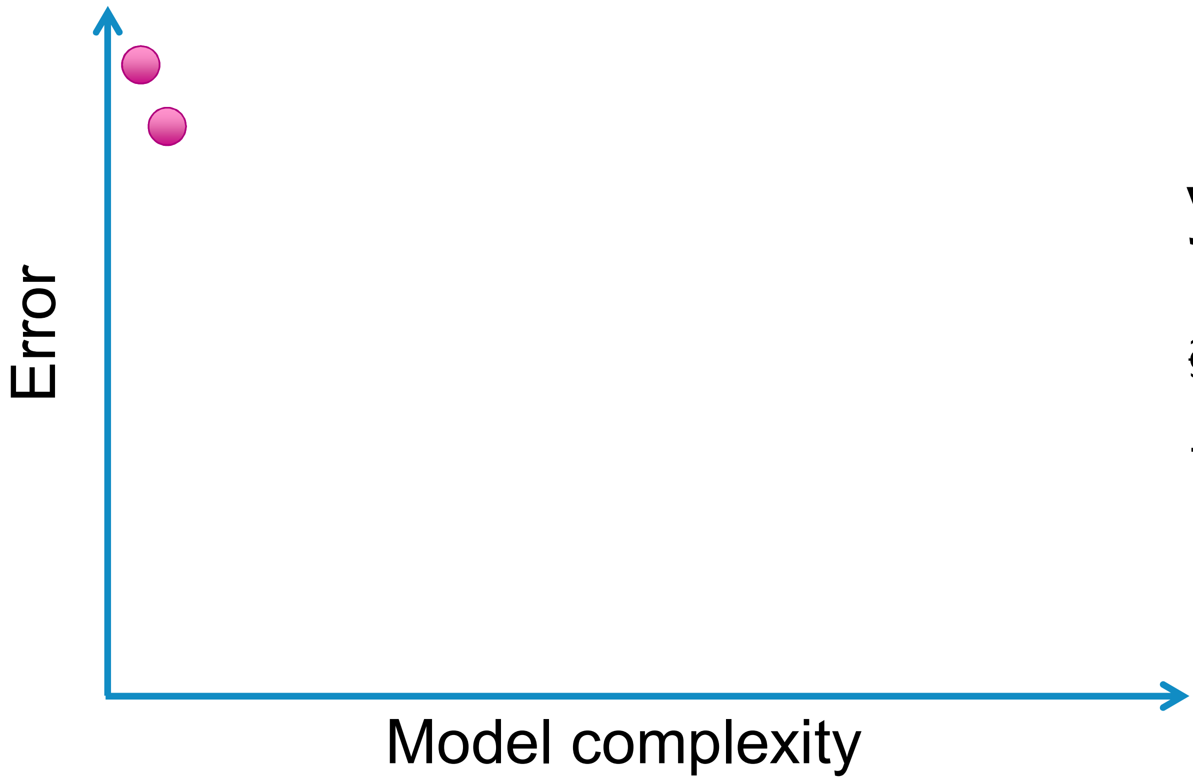
Training error ($\hat{w}$) = 1/N *
$[(\$_{train\ 1}-f_{\hat{w}}(sq.ft._{train\ 1}))^2$
$+ (\$_{train\ 2}-f_{\hat{w}}(sq.ft._{train\ 2}))^2$
$+ (\$_{train\ 3}-f_{\hat{w}}(sq.ft._{train\ 3}))^2$
+ … include all training houses]

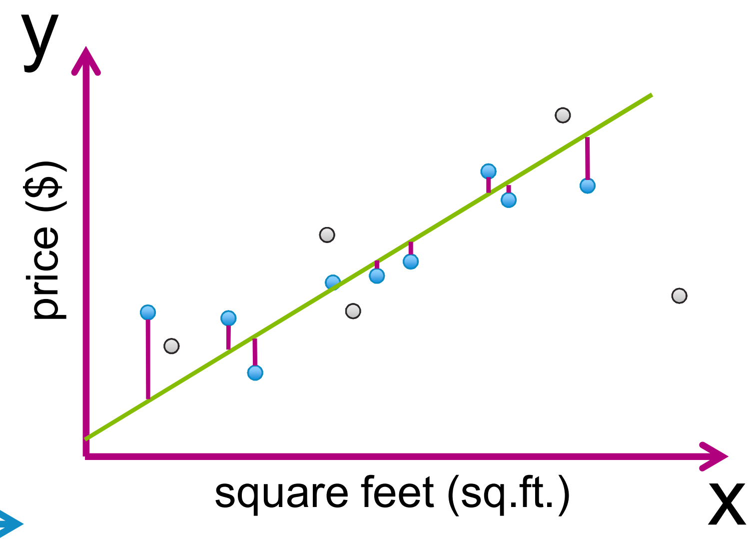
Training error vs. model complexity



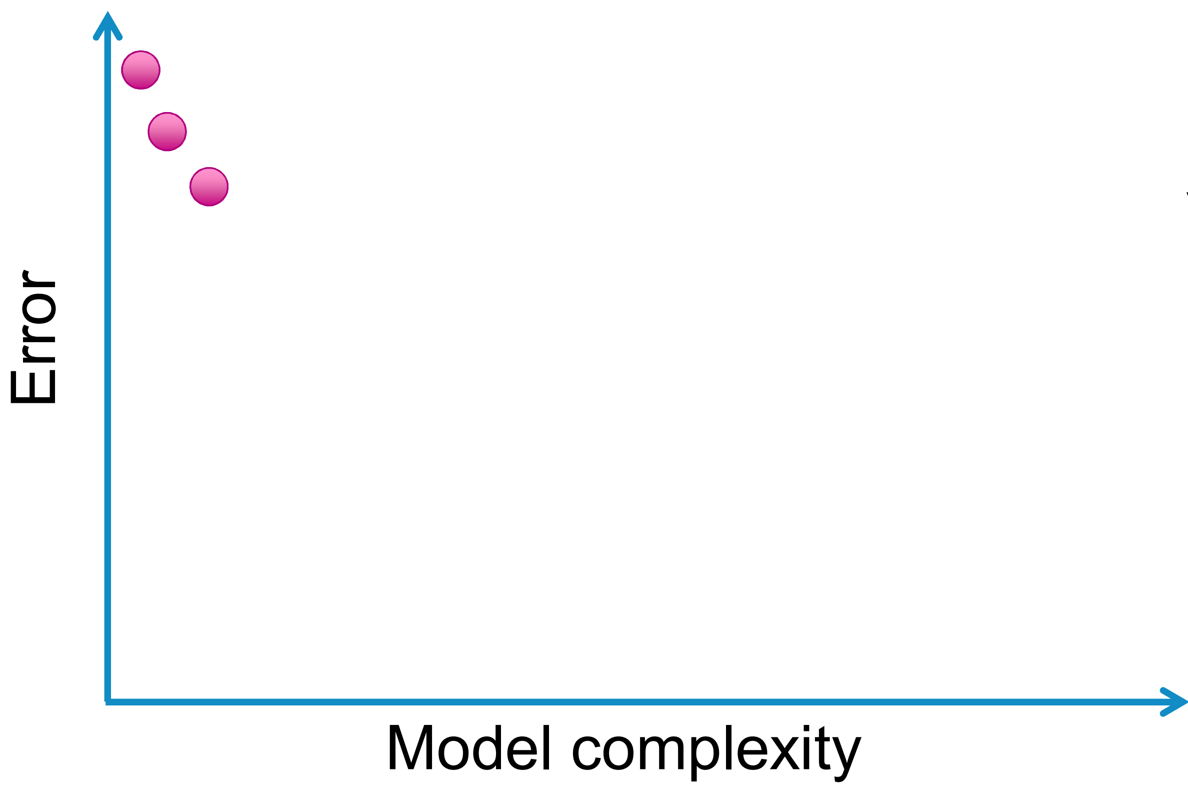


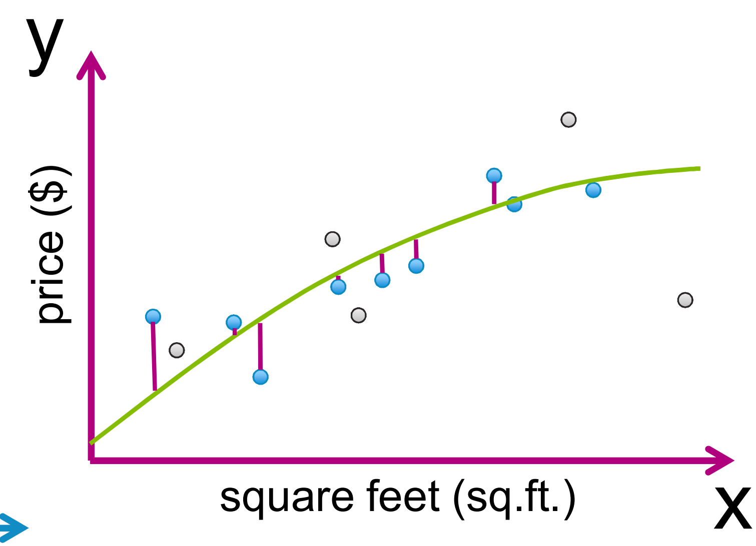
Training error vs. model complexity



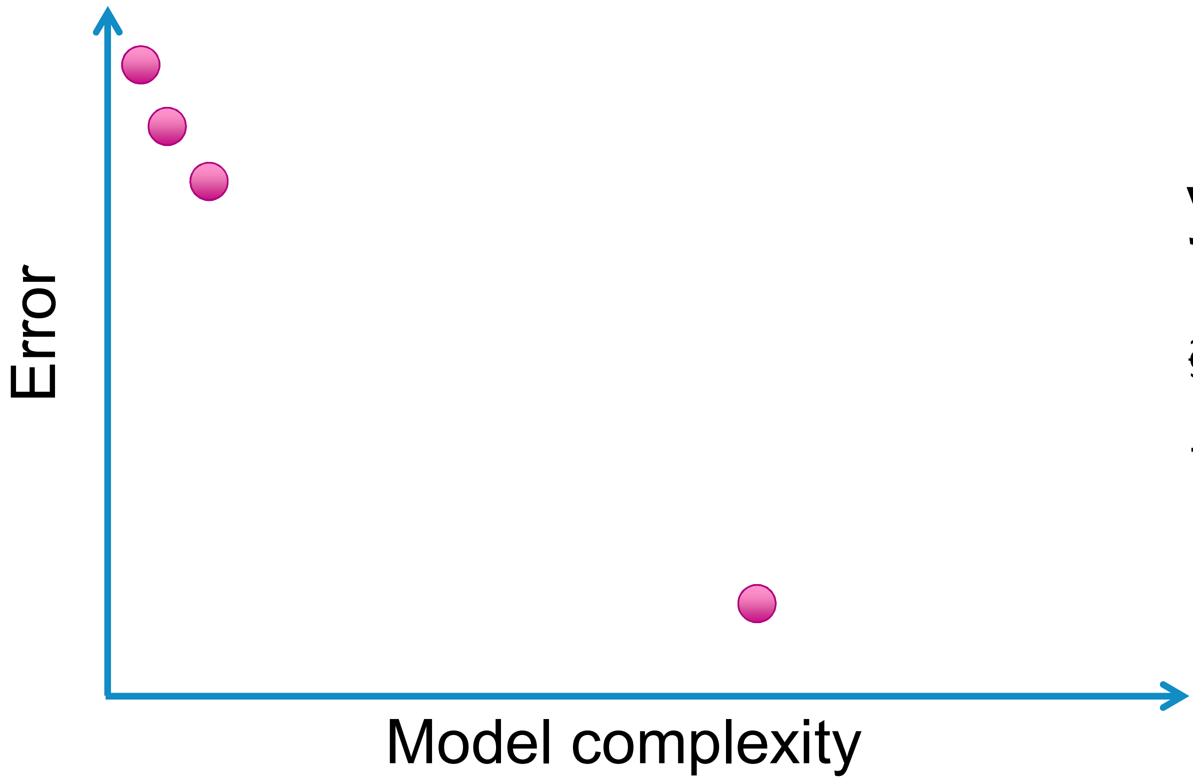


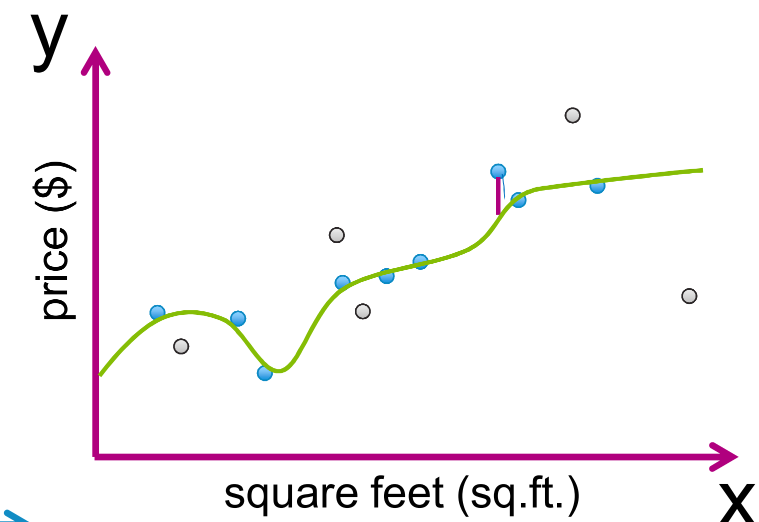
Training error vs. model complexity



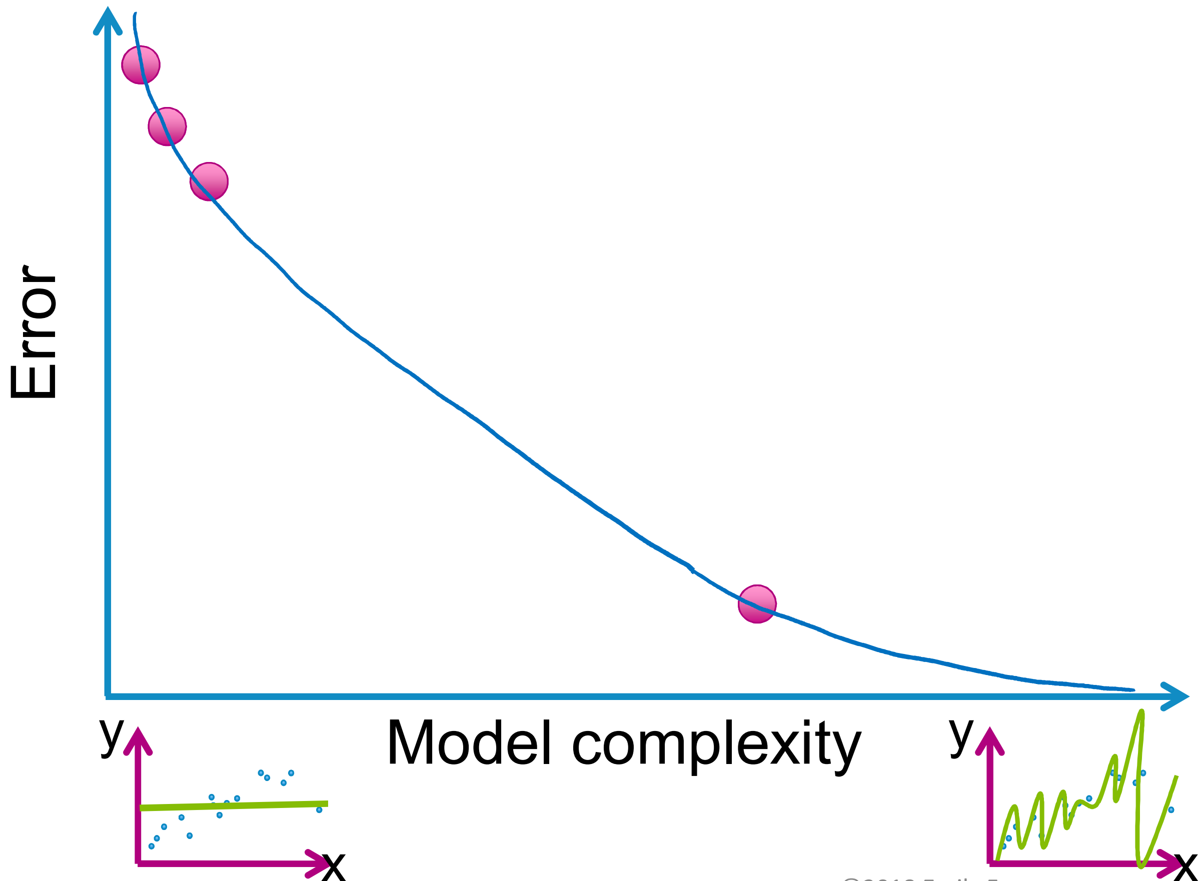


Training error vs. model complexity





Training error vs. model complexity



Assessing the loss
Part 2: Generalization (true) error

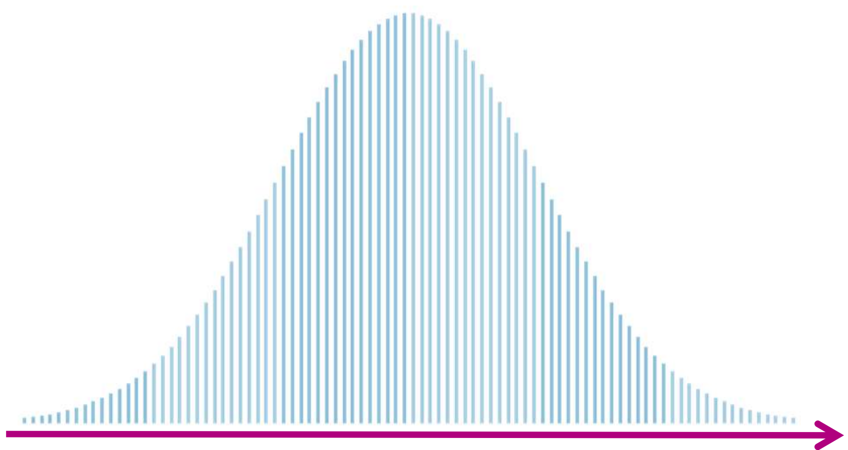
Generalization error

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

Lots of houses
in neighborhood,
but not in dataset

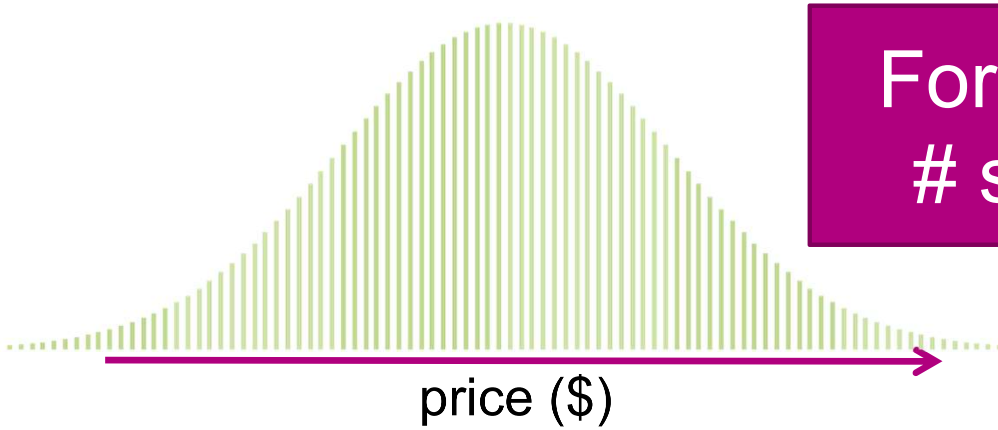
Distribution over houses

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?



For fixed # sq.ft.

Generalization error definition

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

Formally:

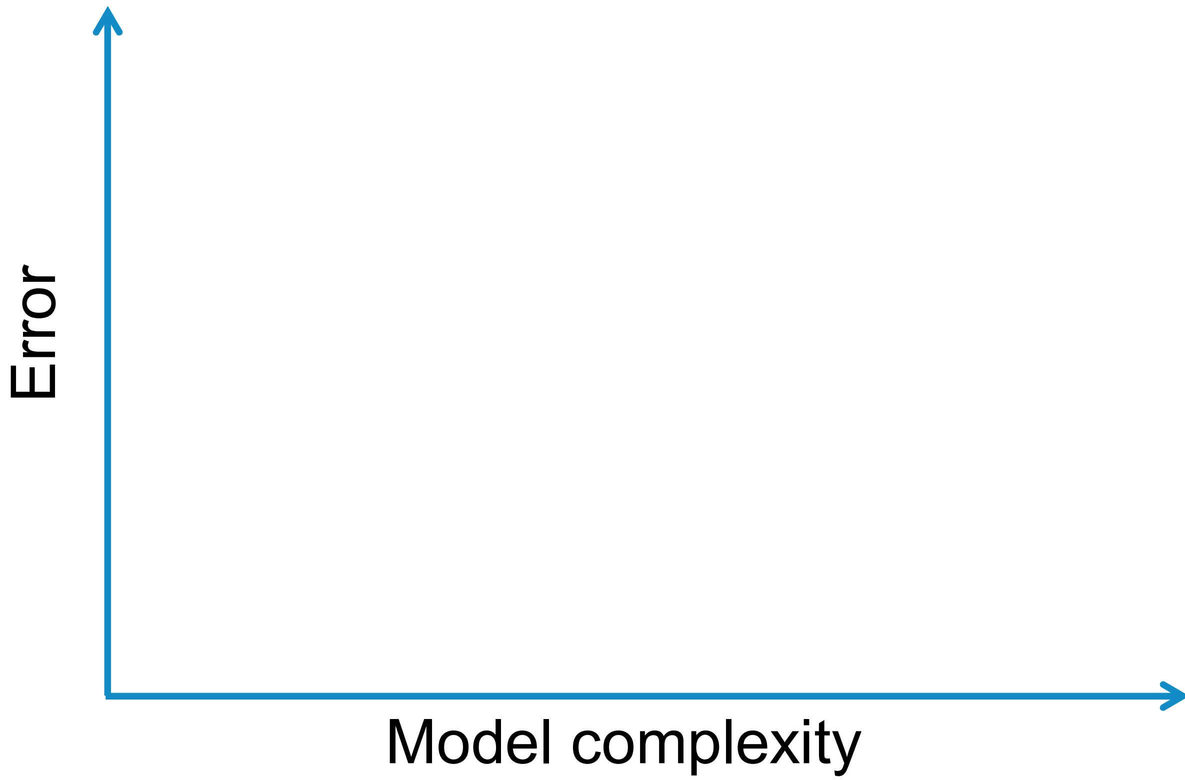
average over all possible (x,y) pairs weighted by how likely each is

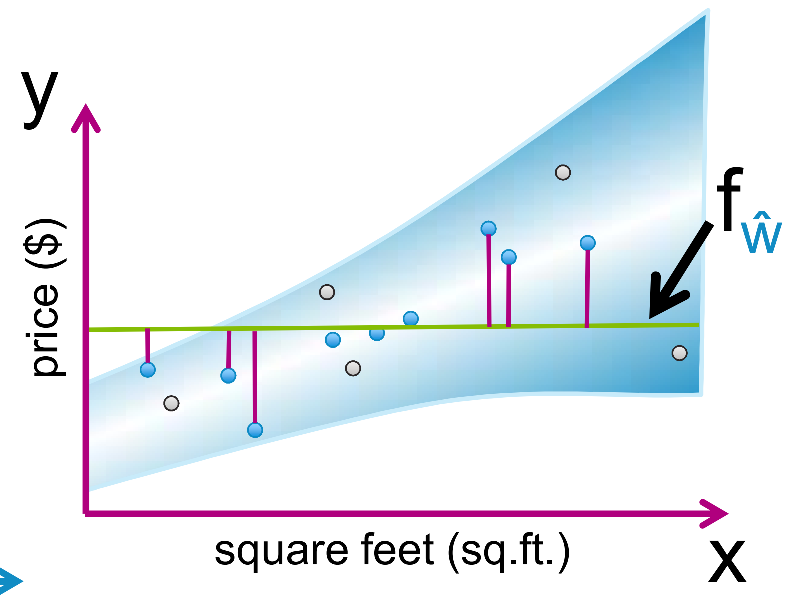
generalization error = $E_{x,y}[L(y,f_{\hat{w}}(x))]$

fit using training data

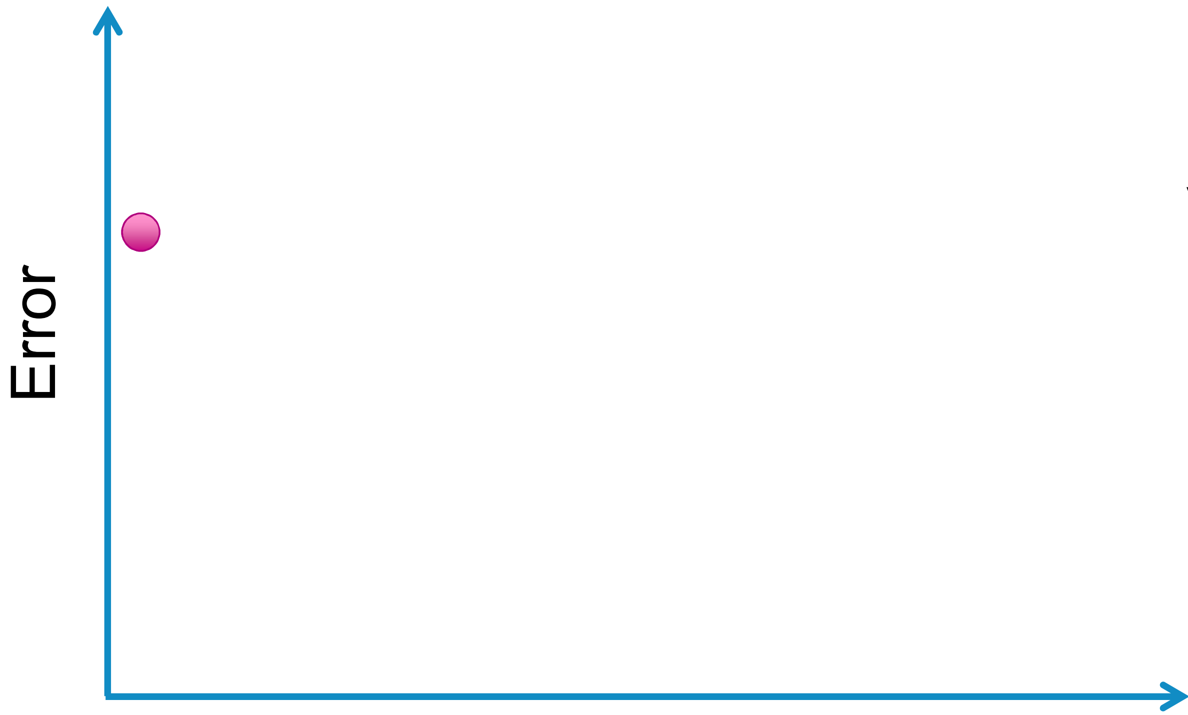
$$= \int L(y, f_{\hat{w}}(x))\, p(x,y)\, dx\, dy$$

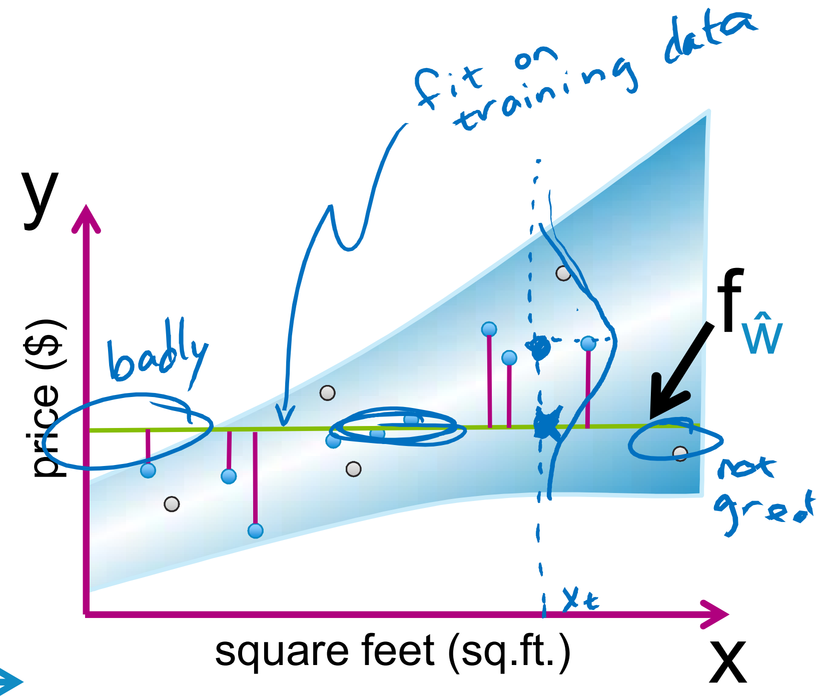
Generalization error vs. model complexity



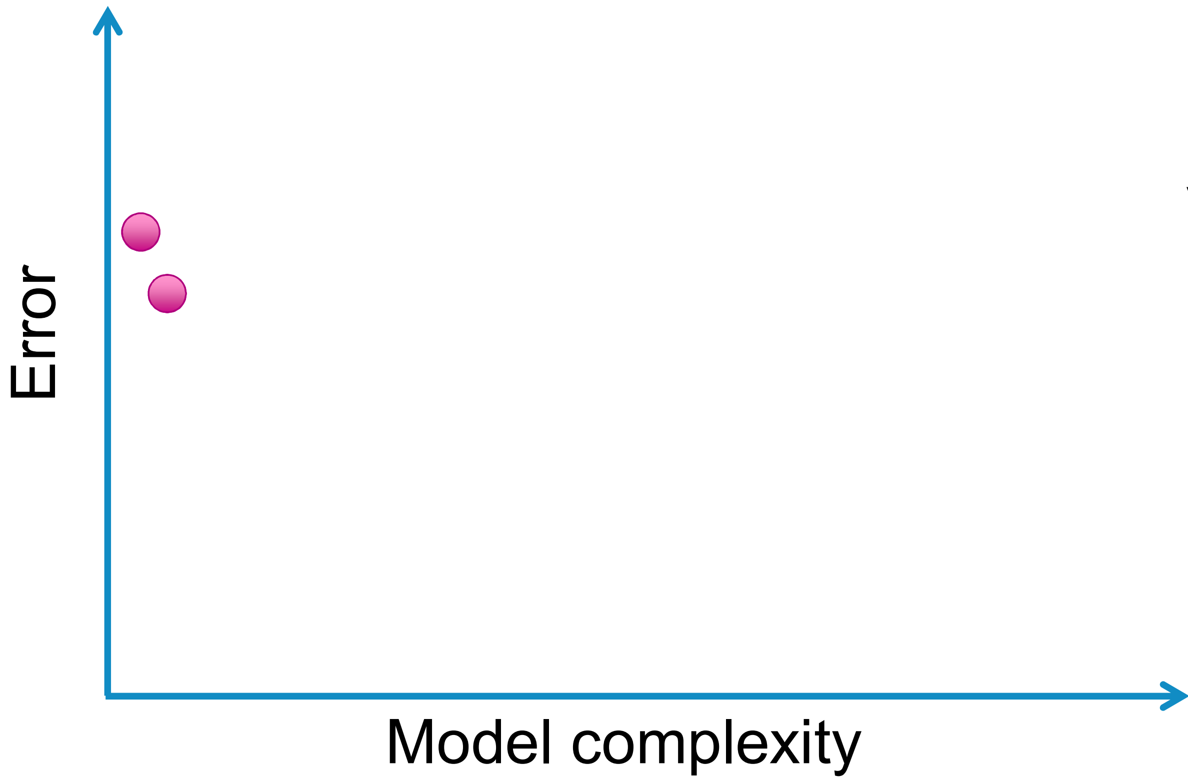


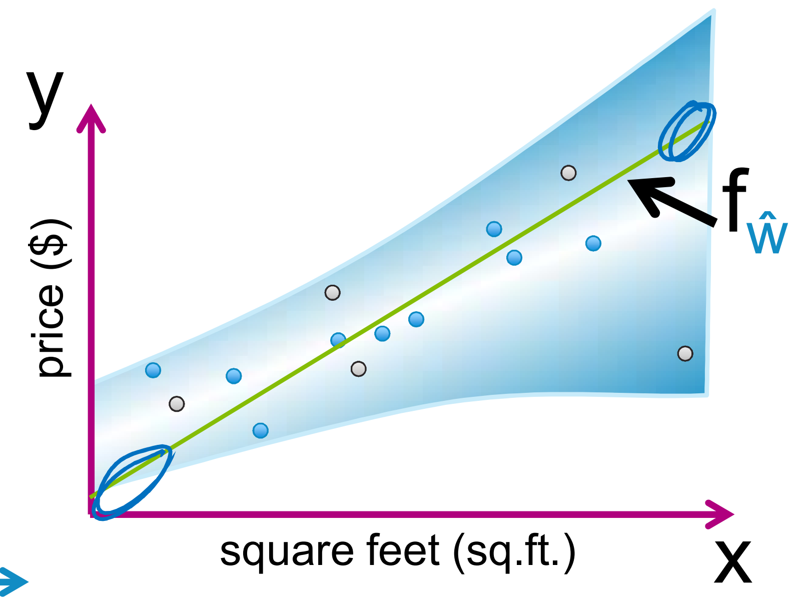
Generalization error vs. model complexity



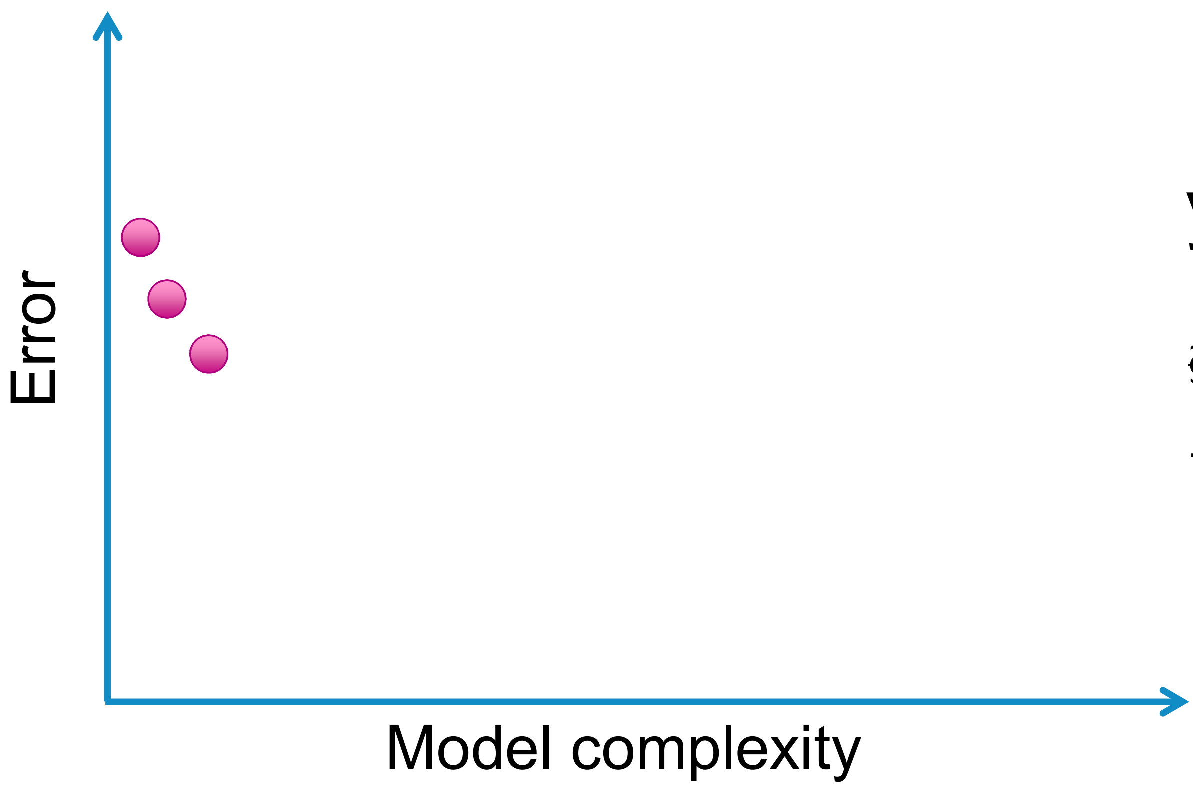


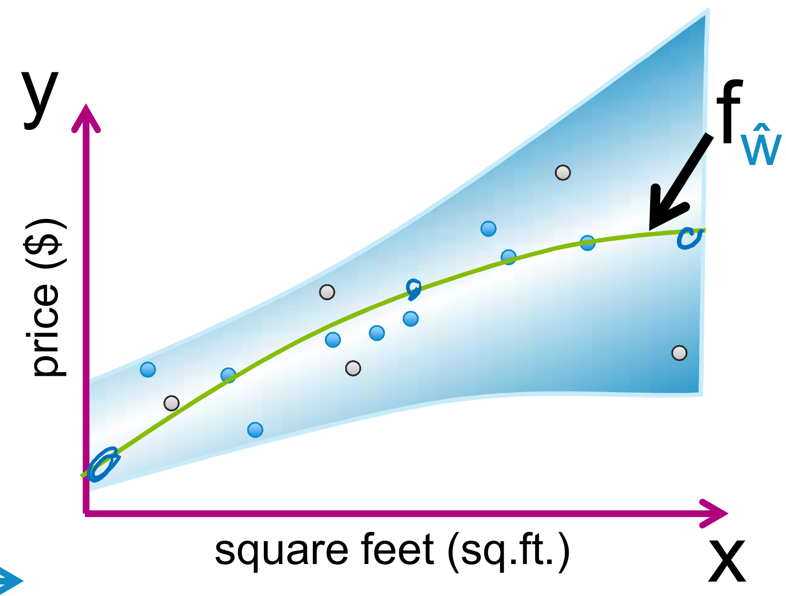
Generalization error vs. model complexity



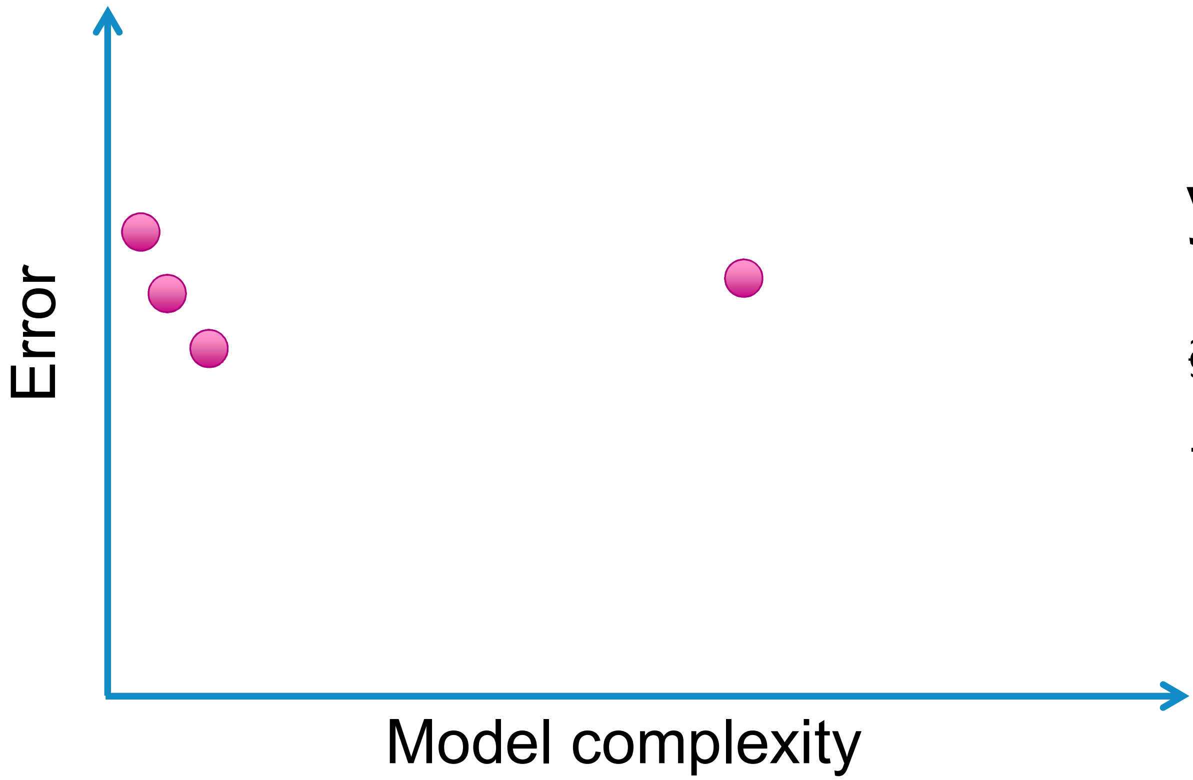


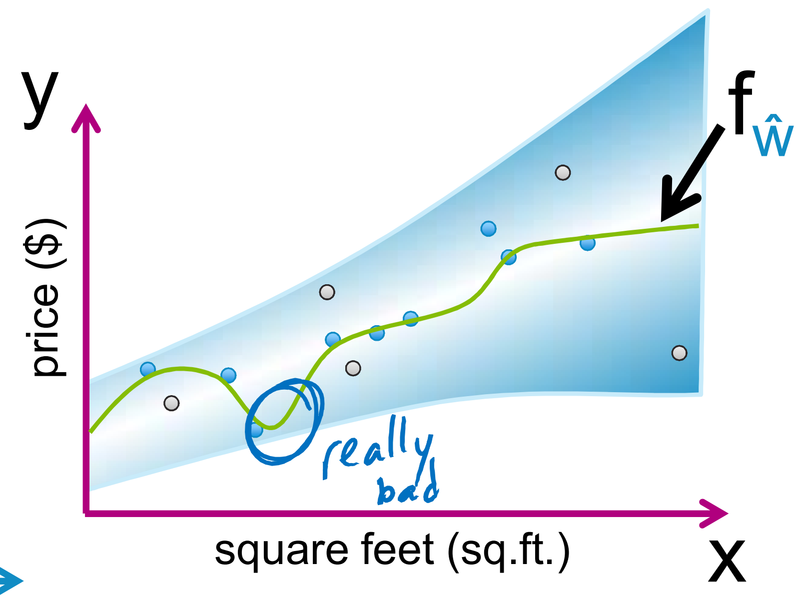
Generalization error vs. model complexity



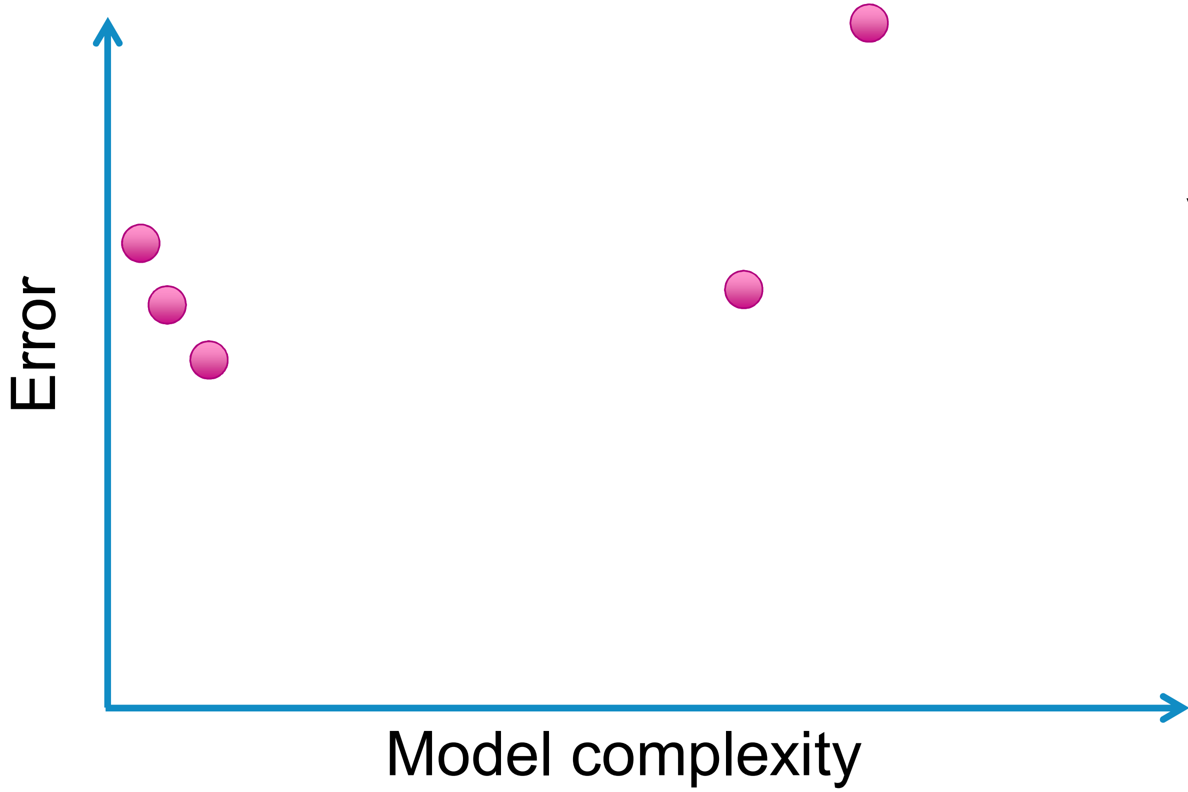


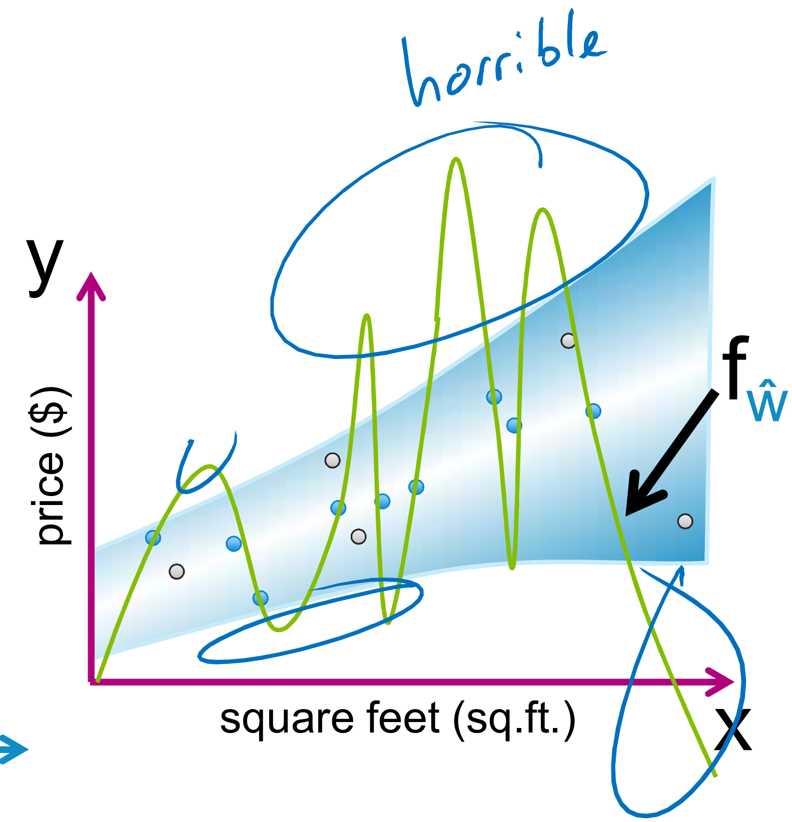
Generalization error vs. model complexity



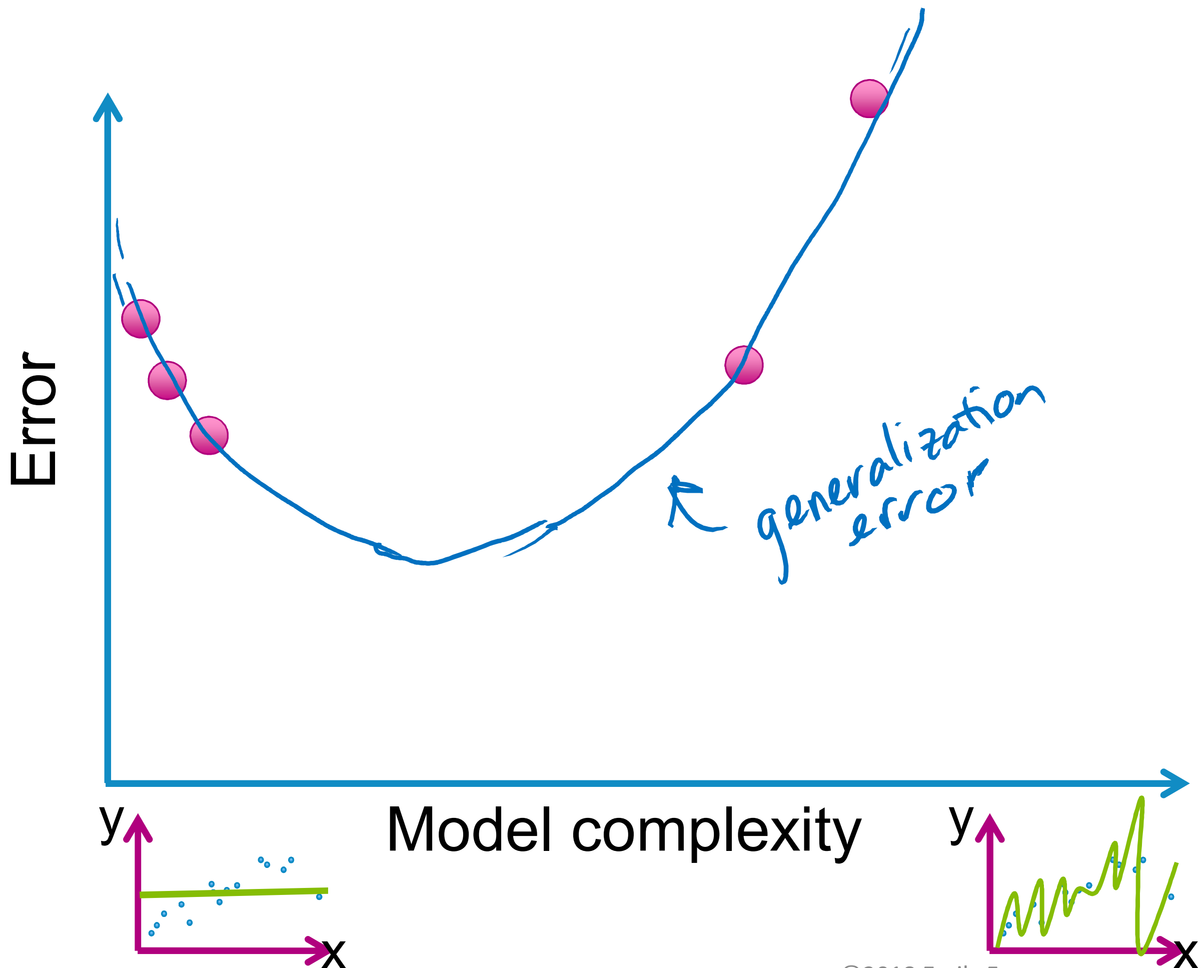


Generalization error vs. model complexity





Generalization error vs. model complexity



Can't compute!

Assessing the loss
Part 3: Test error

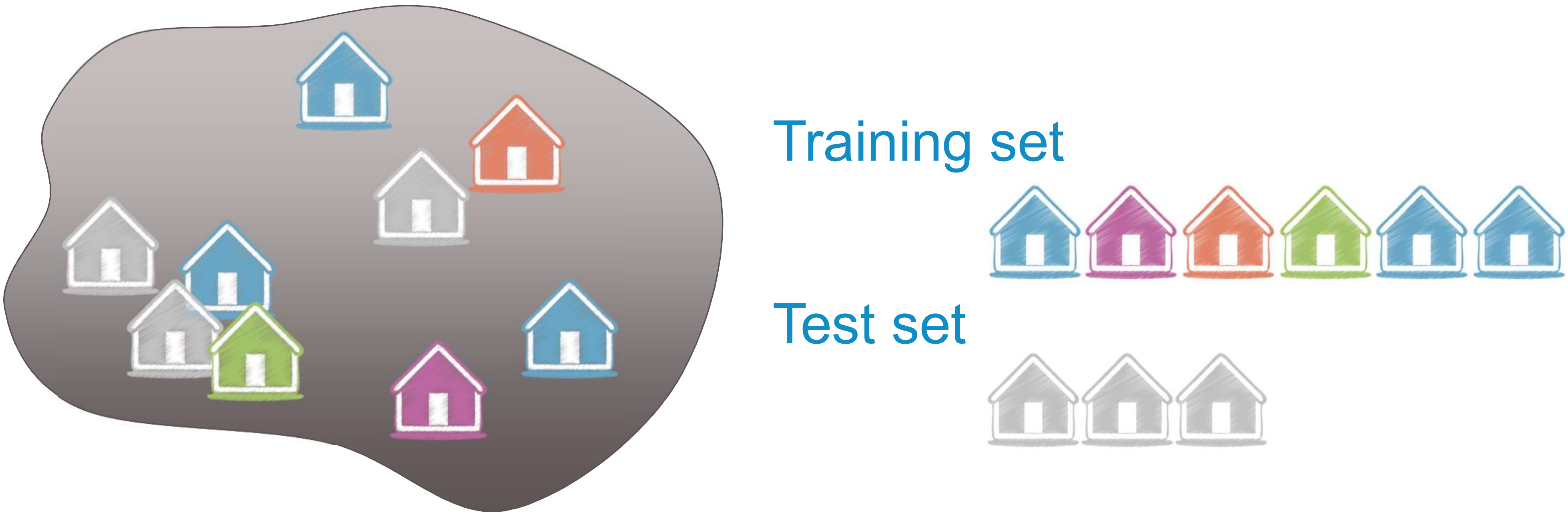
Approximating generalization error

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



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

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

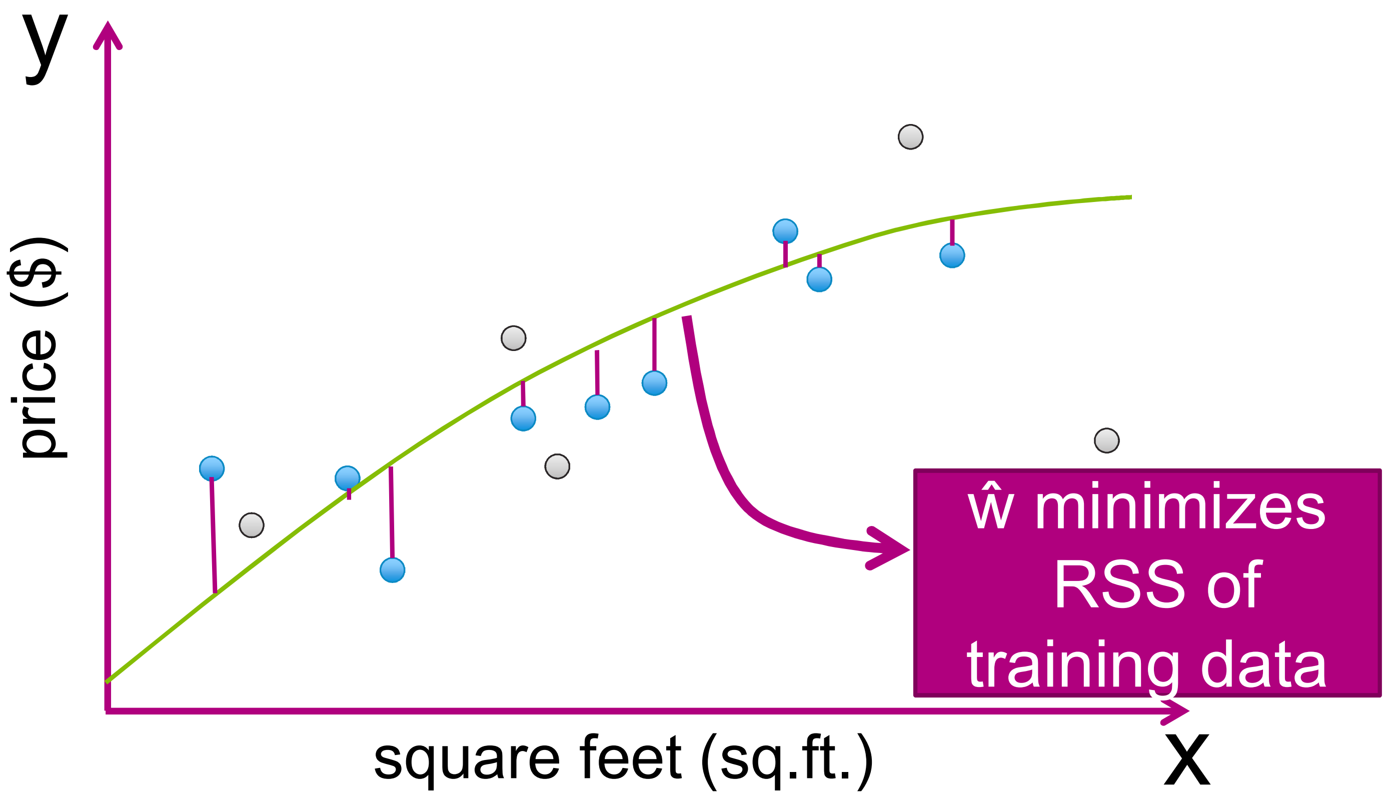
fit using training data

has never seen test data!

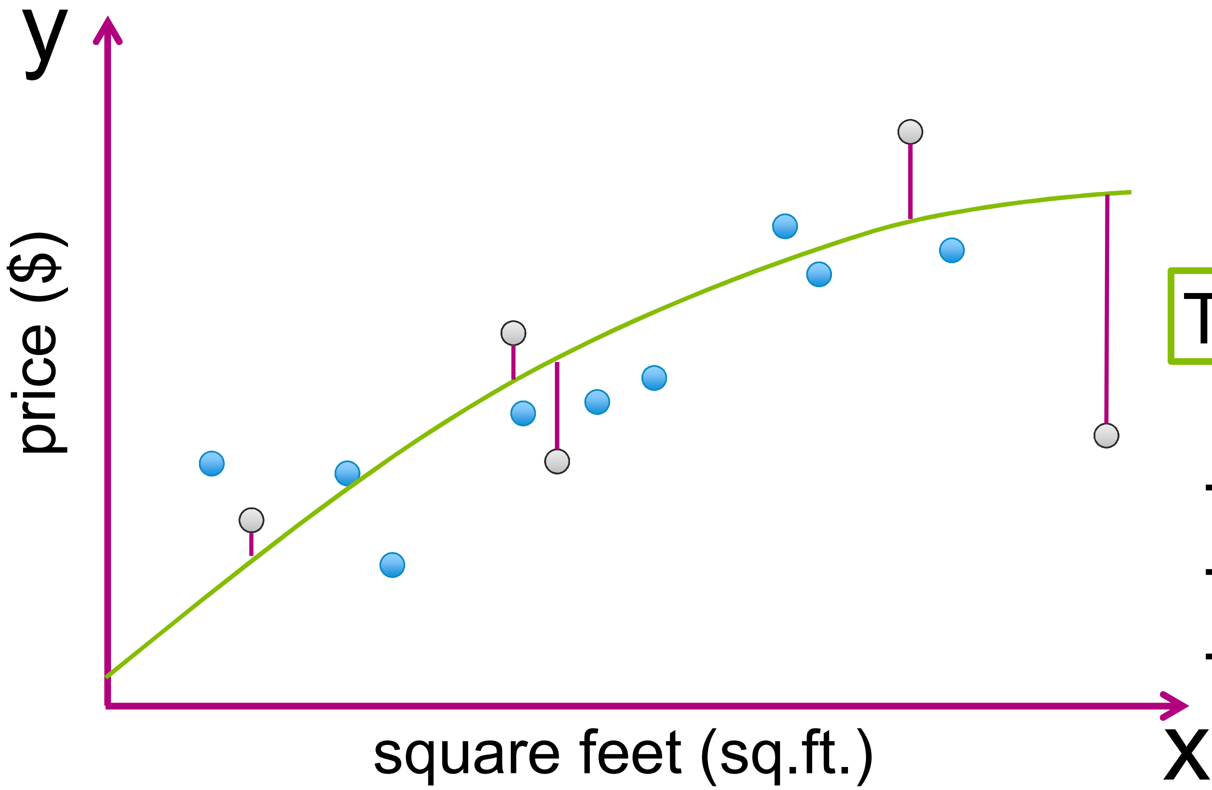
super important!

Don't train on test data

Example:
As before, fit quadratic to training data



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



$\boxed{\text{Test error }(\hat{w})} = 1/N_{test} *$

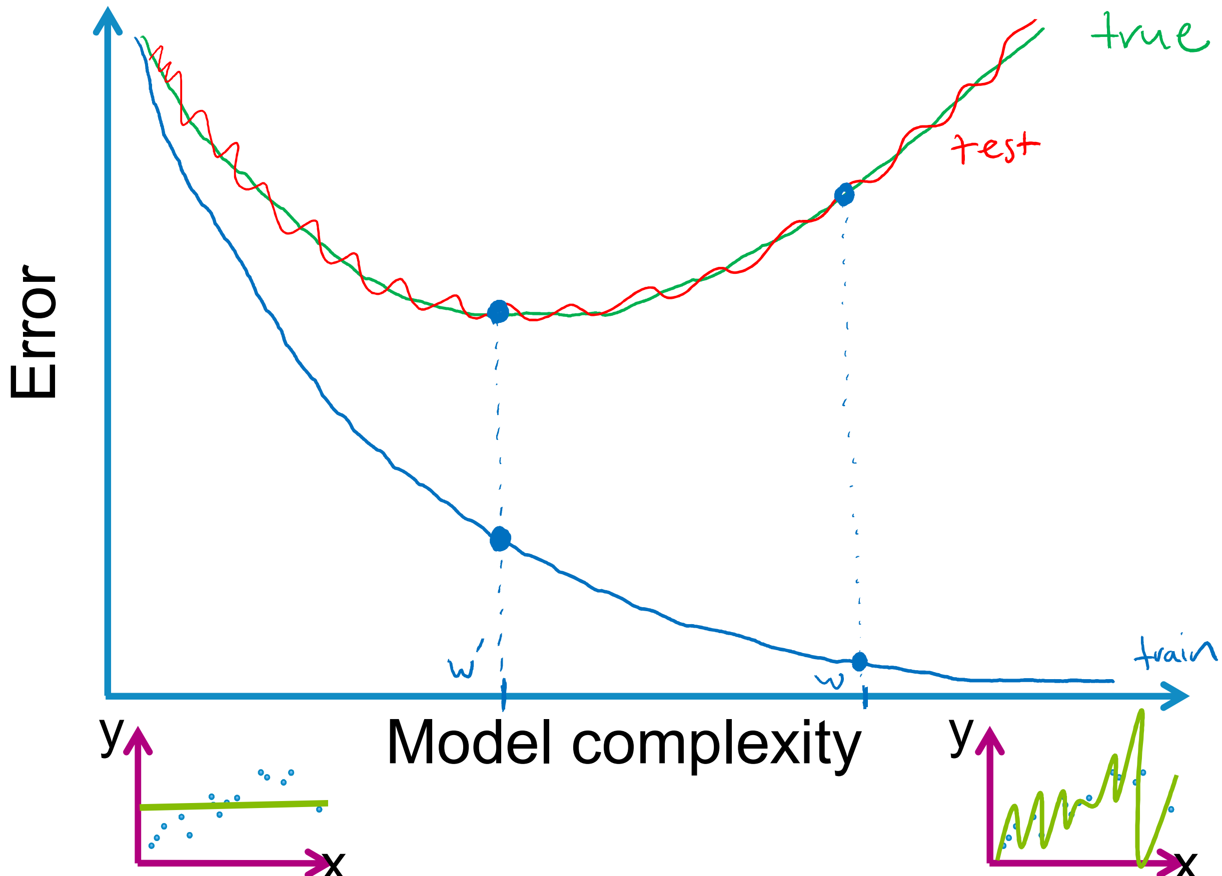
$[(\$_{test\ 1}-f_{\hat{w}}(sq.ft._{test\ 1}))^2$

$+ (\$_{test\ 2}-f_{\hat{w}}(sq.ft._{test\ 2}))^2$

$+ (\$_{test\ 3}-f_{\hat{w}}(sq.ft._{test\ 3}))^2$

+ … include all test houses]

Training, true, & test error vs. model complexity



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')

Training/test split

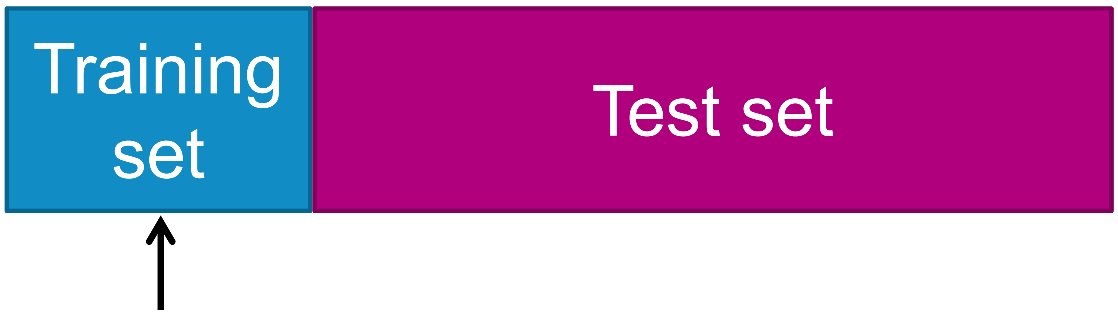
Training/test splits

whole dataset

Training set | Test set

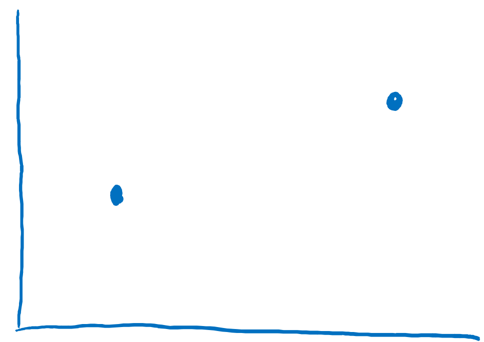
how many? vs. how many?

Training/test splits

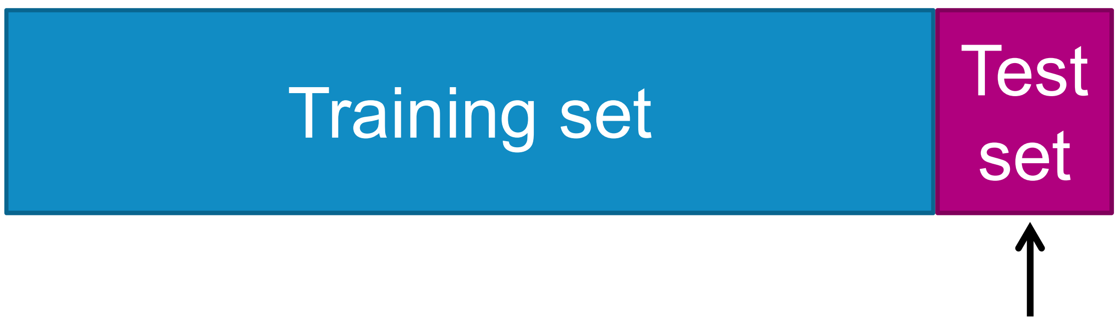


Too few → $\hat{w}$ poorly estimated

Only 2 train points
- could be linear
- could be quadratic
- who knows?

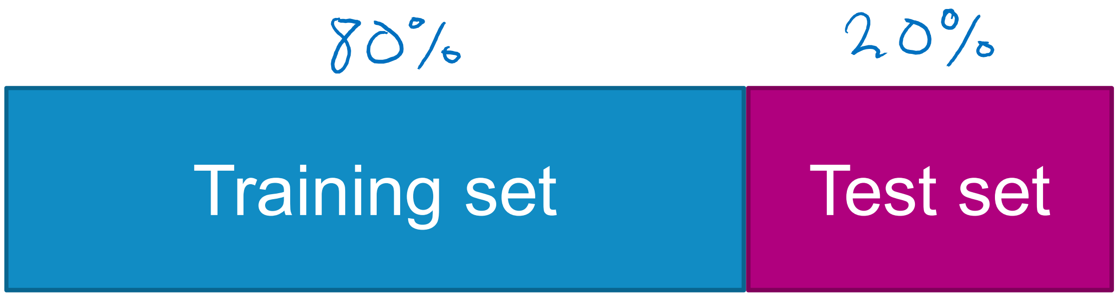


Training/test splits



Too few → test error bad approximation of true error

Training/test splits



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

If this leaves too few for training, other methods like cross validation (will see later…)

3 sources of error + the bias-variance tradeoff

3 sources of error

In forming predictions, there are 3 sources of error:

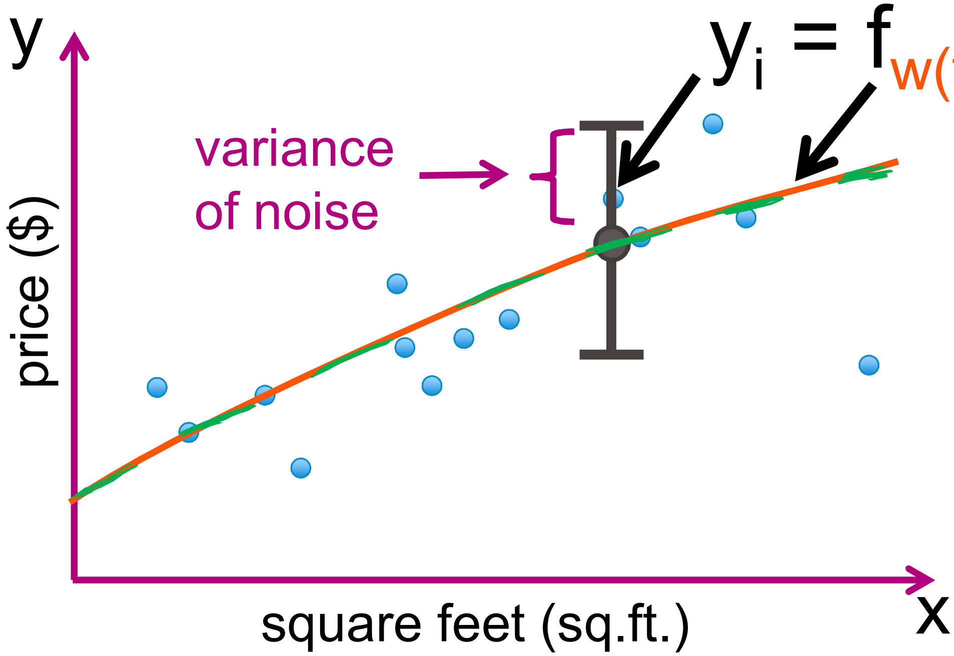
1. Noise
2. Bias
3. Variance

How your model deals with
- Signal
- Noise

Data inherently noisy

$E[\varepsilon_i] = 0$

$Var(\varepsilon_i) = \sigma^2$



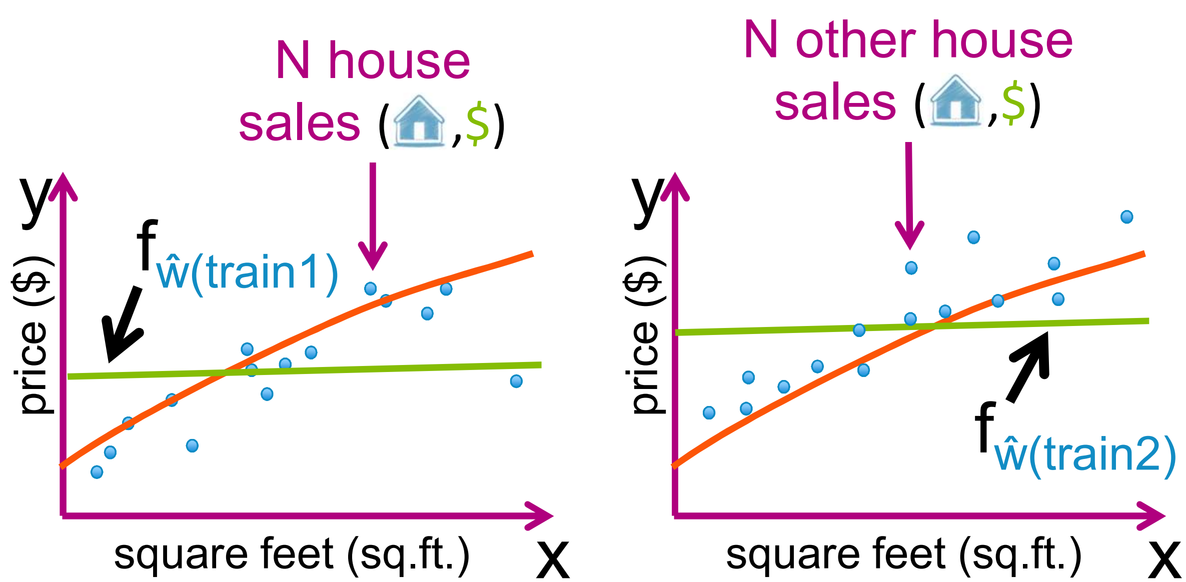
$y_i = f_{w(true)}(x_i) + \varepsilon_i$

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

Irreducible error

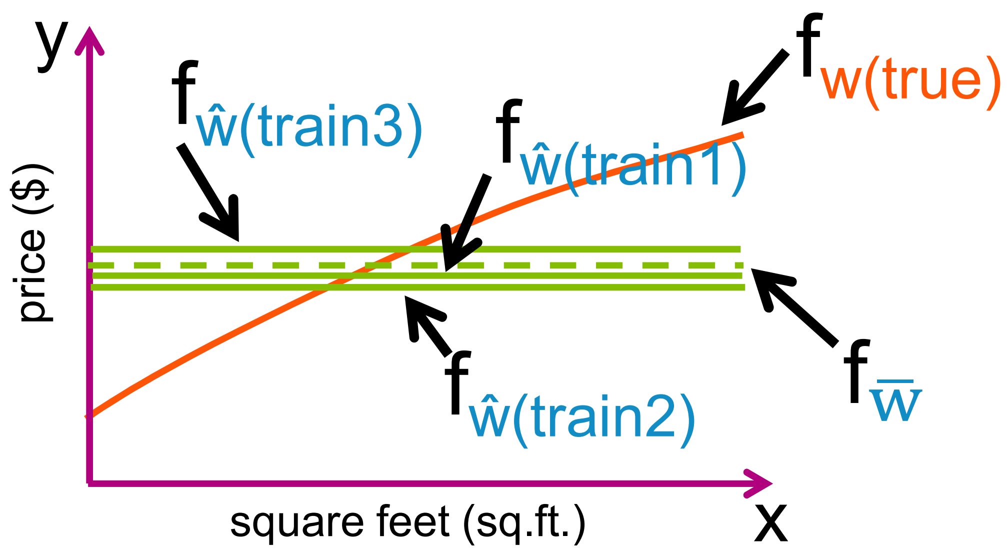
Bias contribution

Assume we fit a constant function



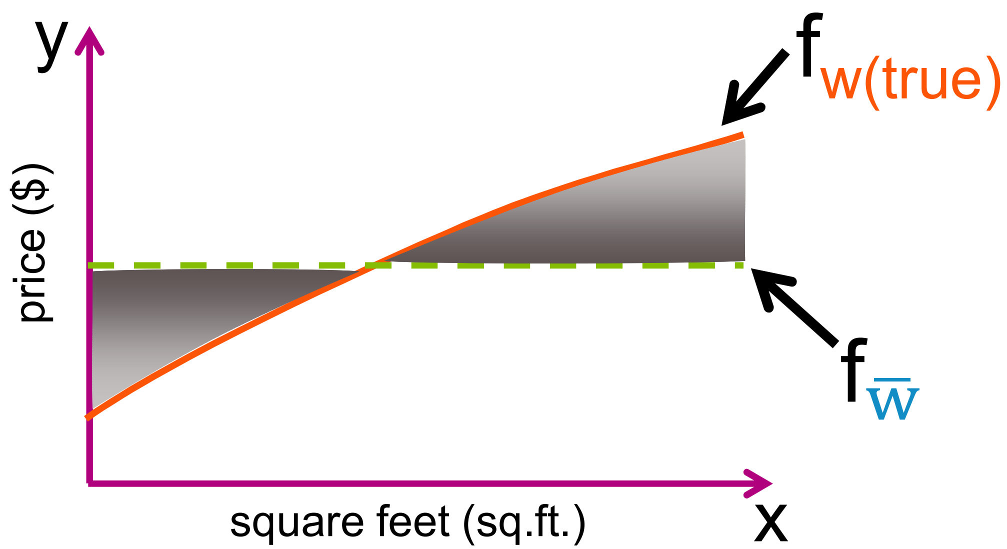
Bias contribution

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



Bias contribution

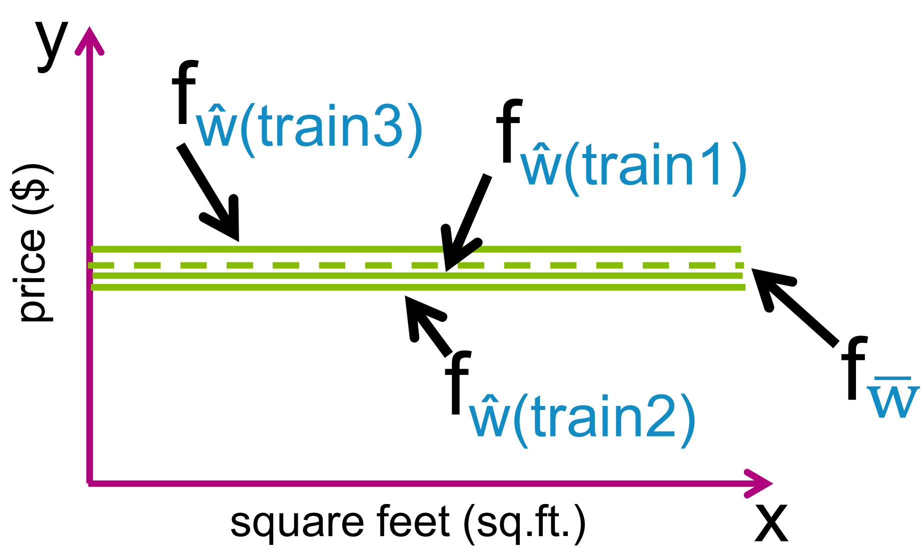
$\text{Bias}(x) = f_{w(true)}(x) - f_{\bar{w}}(x)$ ← Is our approach flexible enough to capture $f_{w(true)}$? If not, error in predictions.



low complexity → high bias

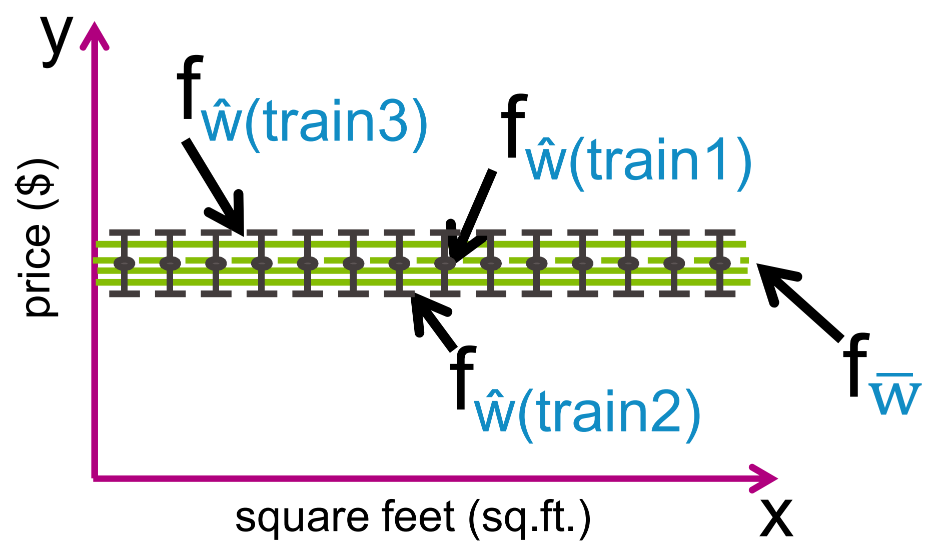
Variance contribution

How much do specific fits vary from the expected fit?



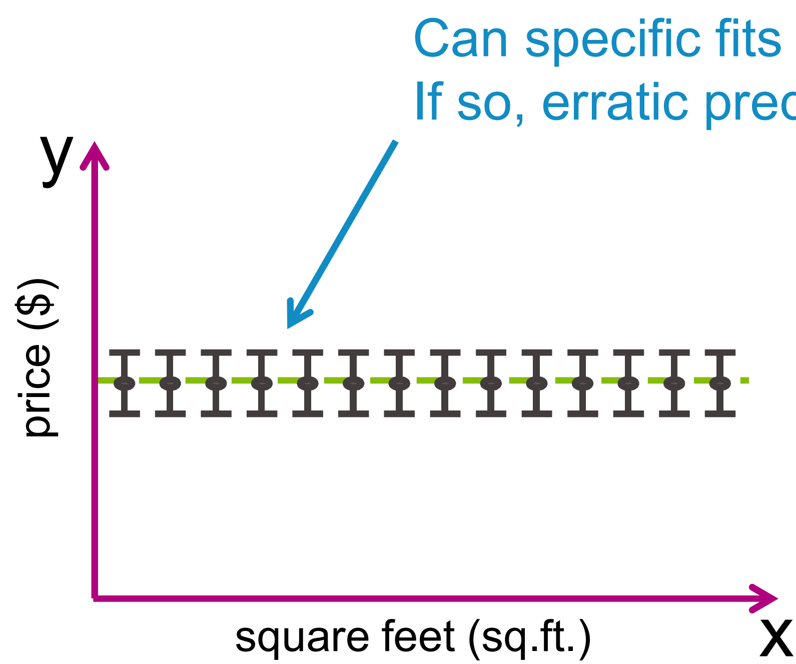
Variance contribution

How much do specific fits vary from the expected fit?



Variance contribution

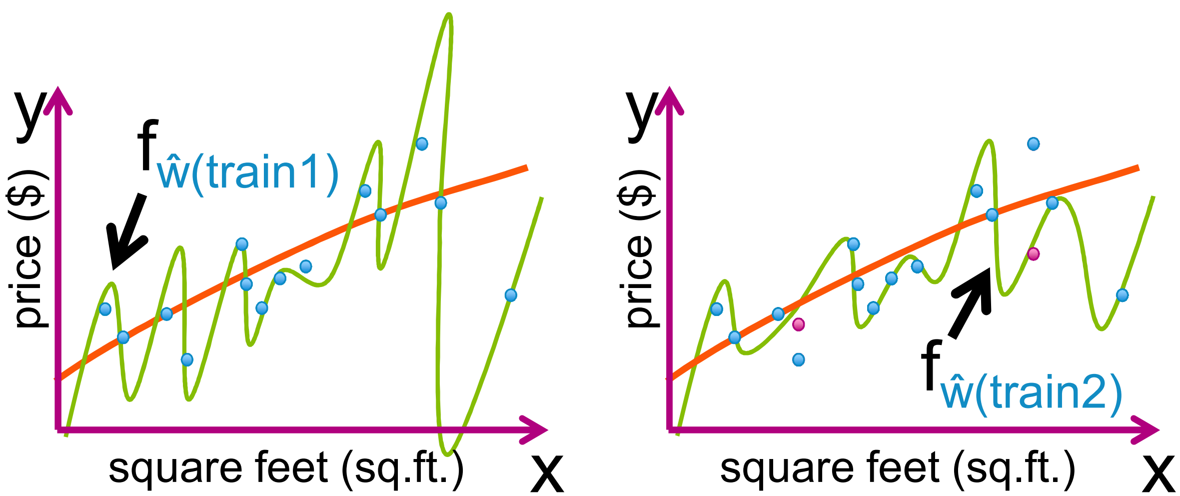
How much do specific fits vary from the expected fit?



low complexity
→
low variance

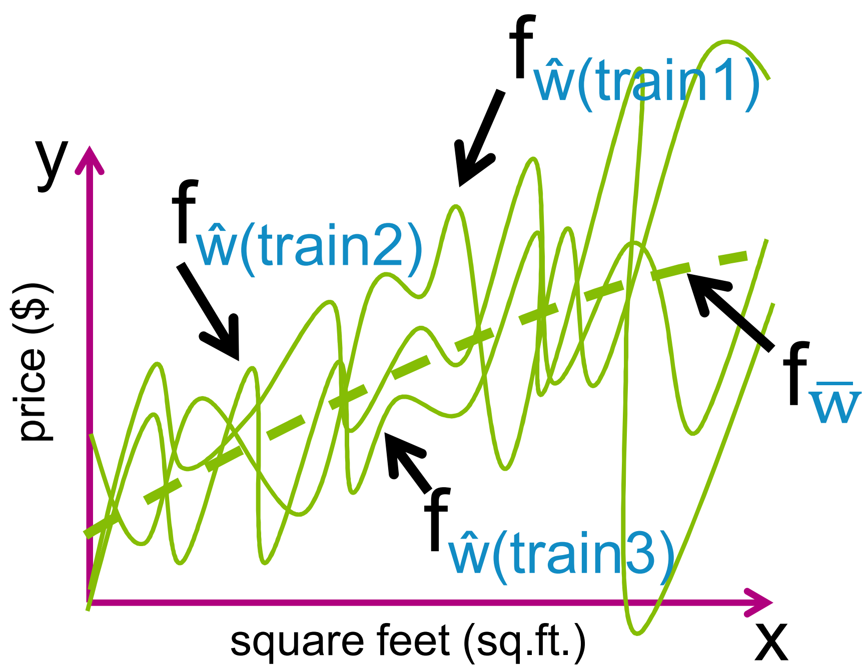
Variance of high-complexity models

Assume we fit a high-order polynomial

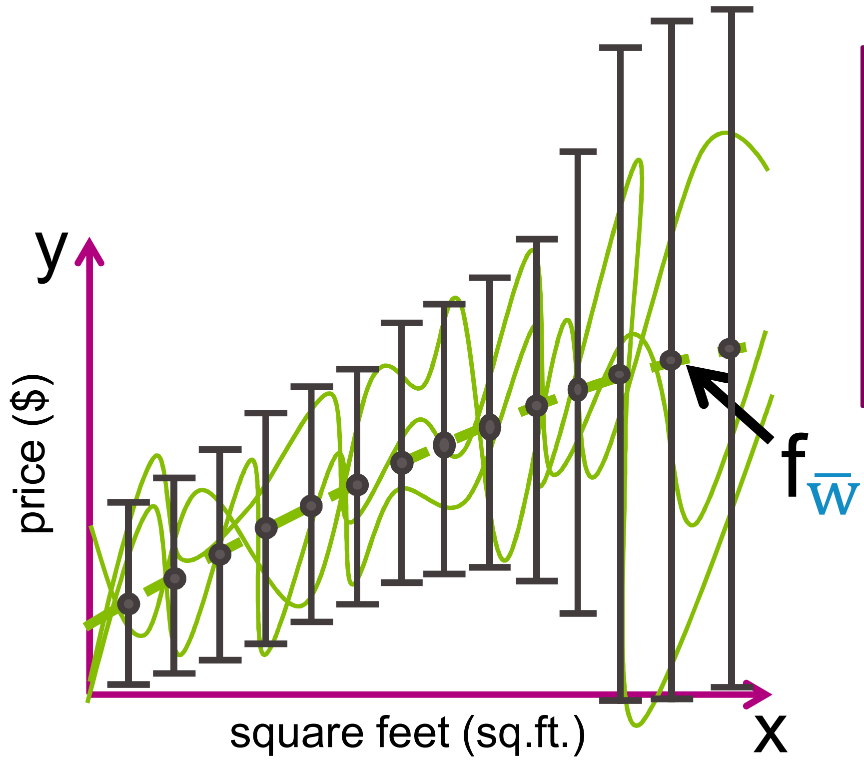


Variance of high-complexity models

Assume we fit a high-order polynomial

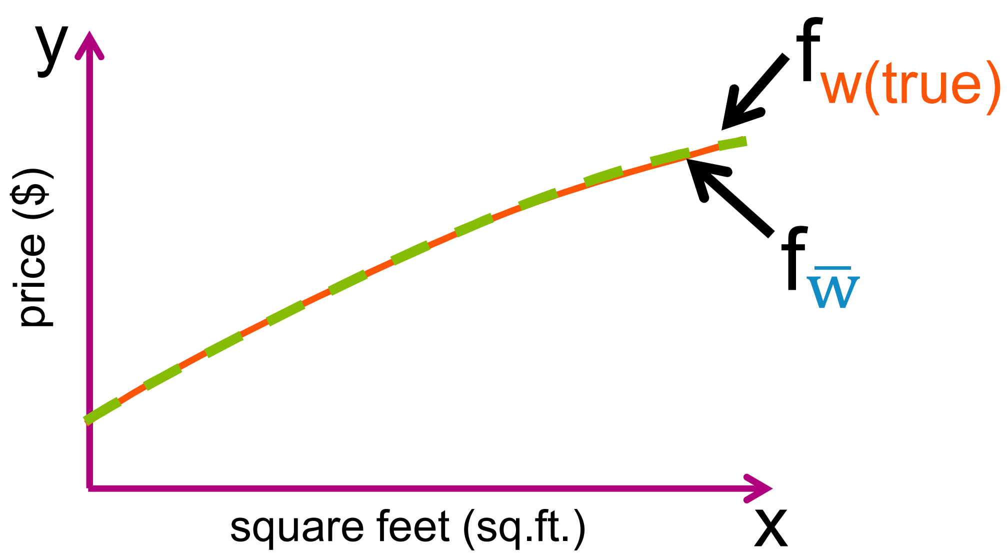


Variance of high-complexity models



high complexity
→
high variance

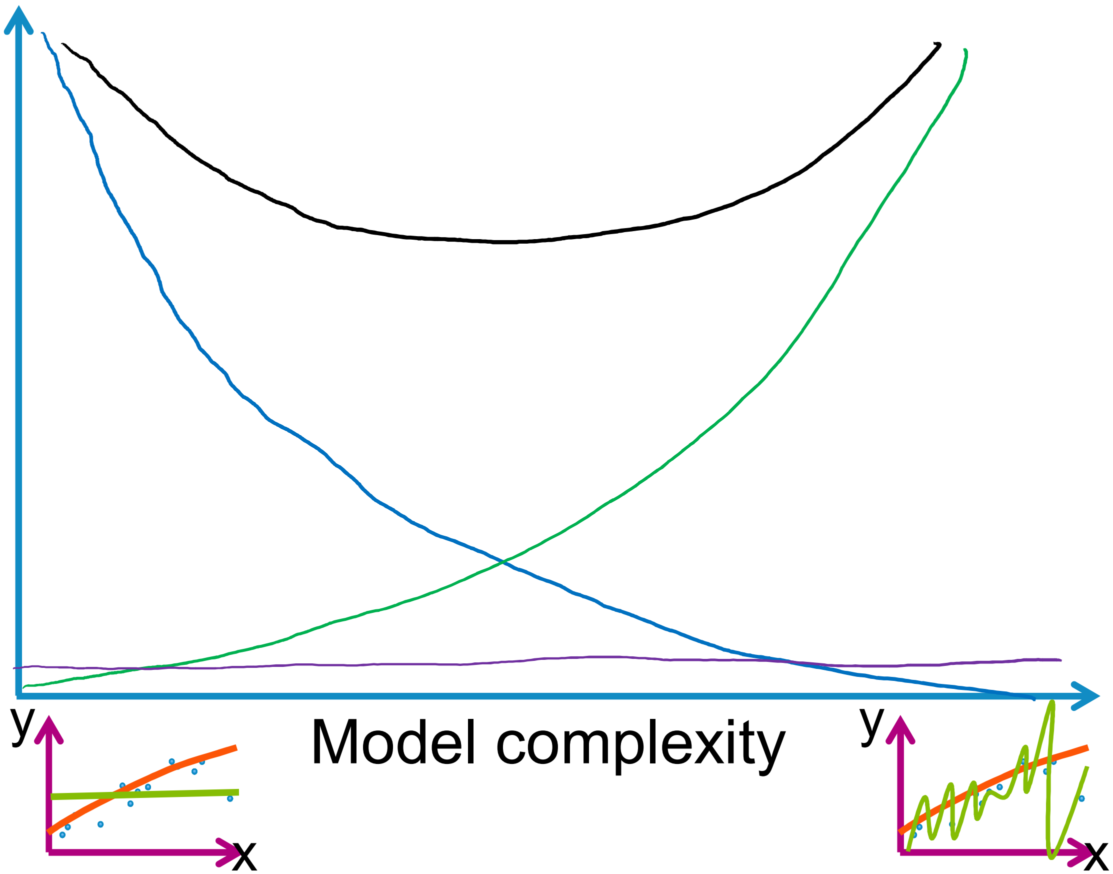
Bias of high-complexity models



high complexity
→
low bias

Bias-variance tradeoff

error = bias^2 + variance + noise



Simple Models

Bias: High

Variance: Low

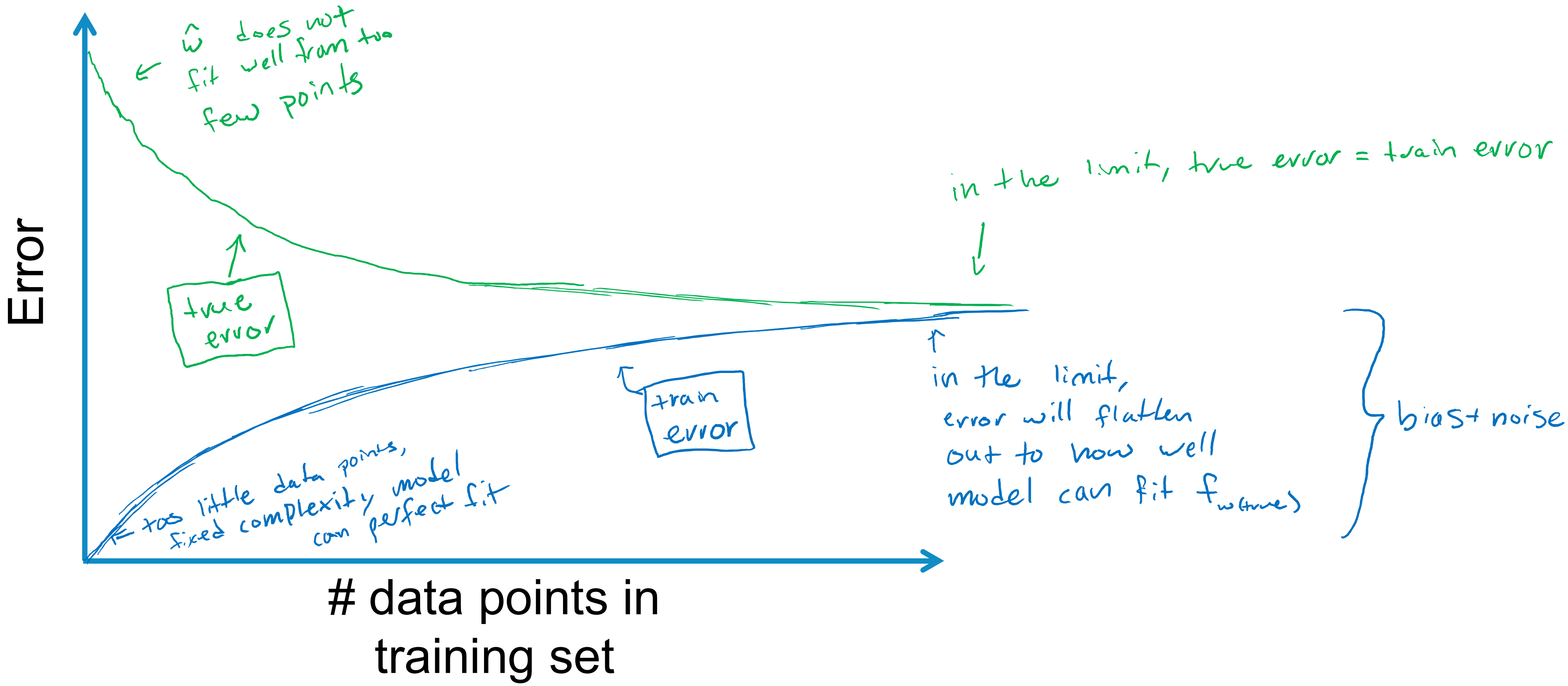
Complex Models

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