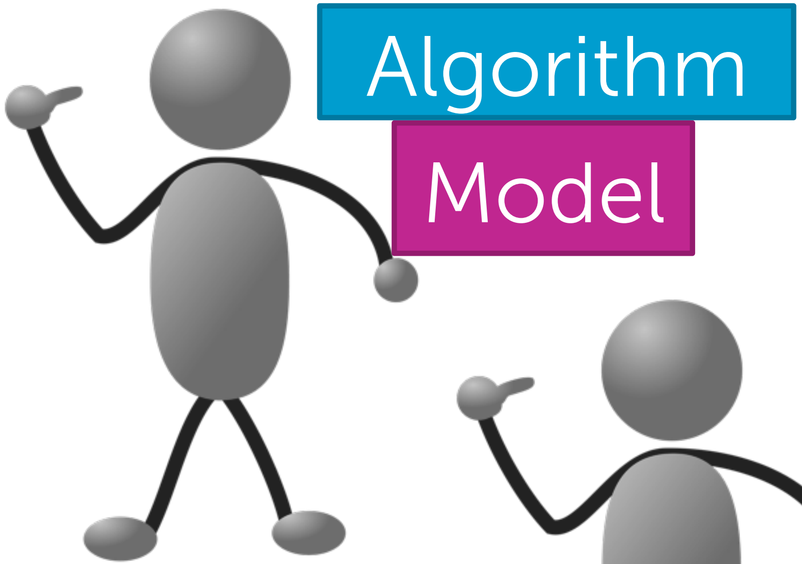
Assessing Performance

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington

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:

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

Cost of using ŵ at x when y is true

Examples:

(assuming loss for underpredicting = overpredicting)

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

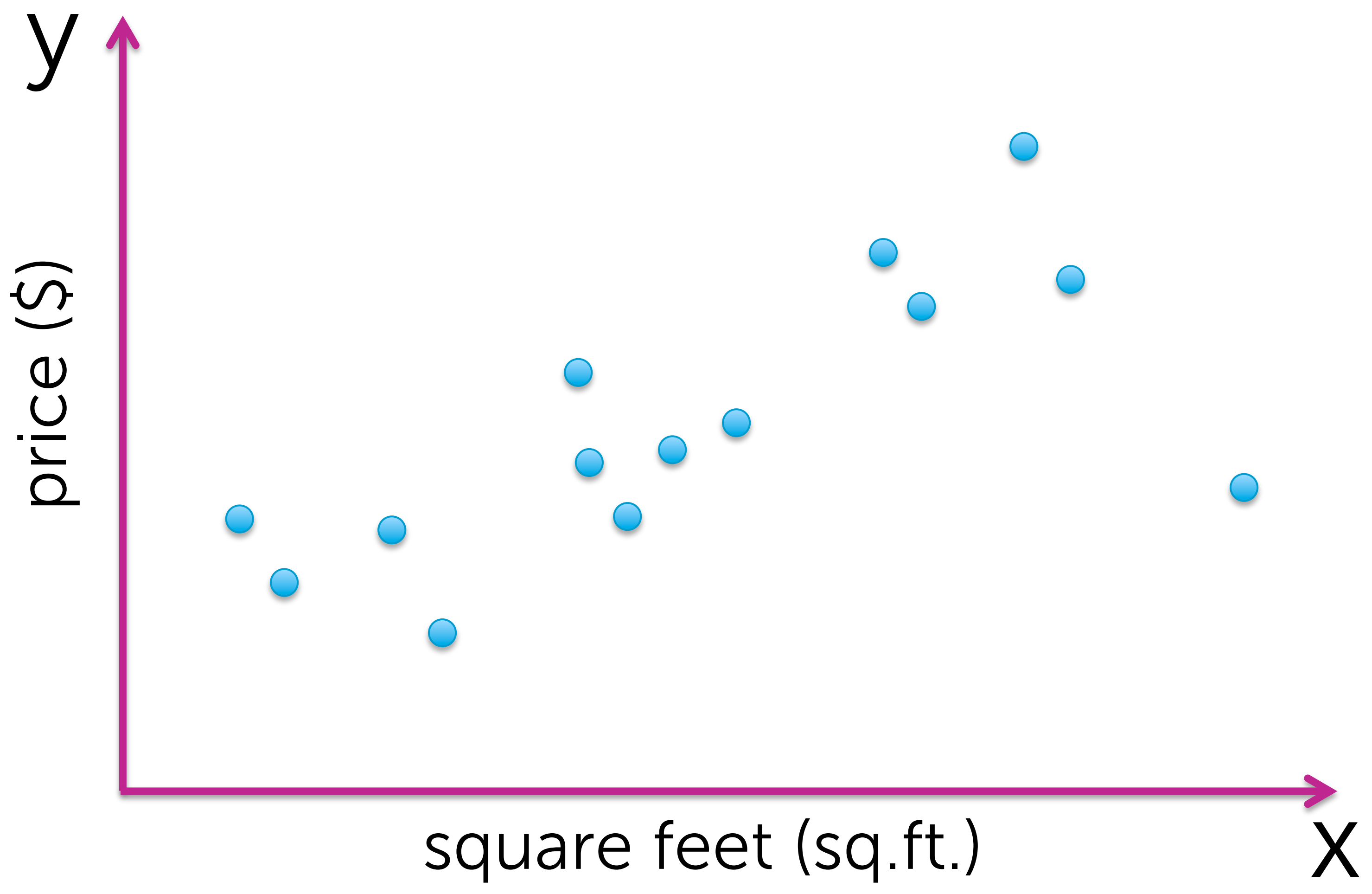
Squared error: $L(y,f_{\hat{w}}(\mathbf{x})) = (y-f_{\hat{w}}(\mathbf{x}))^2$

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

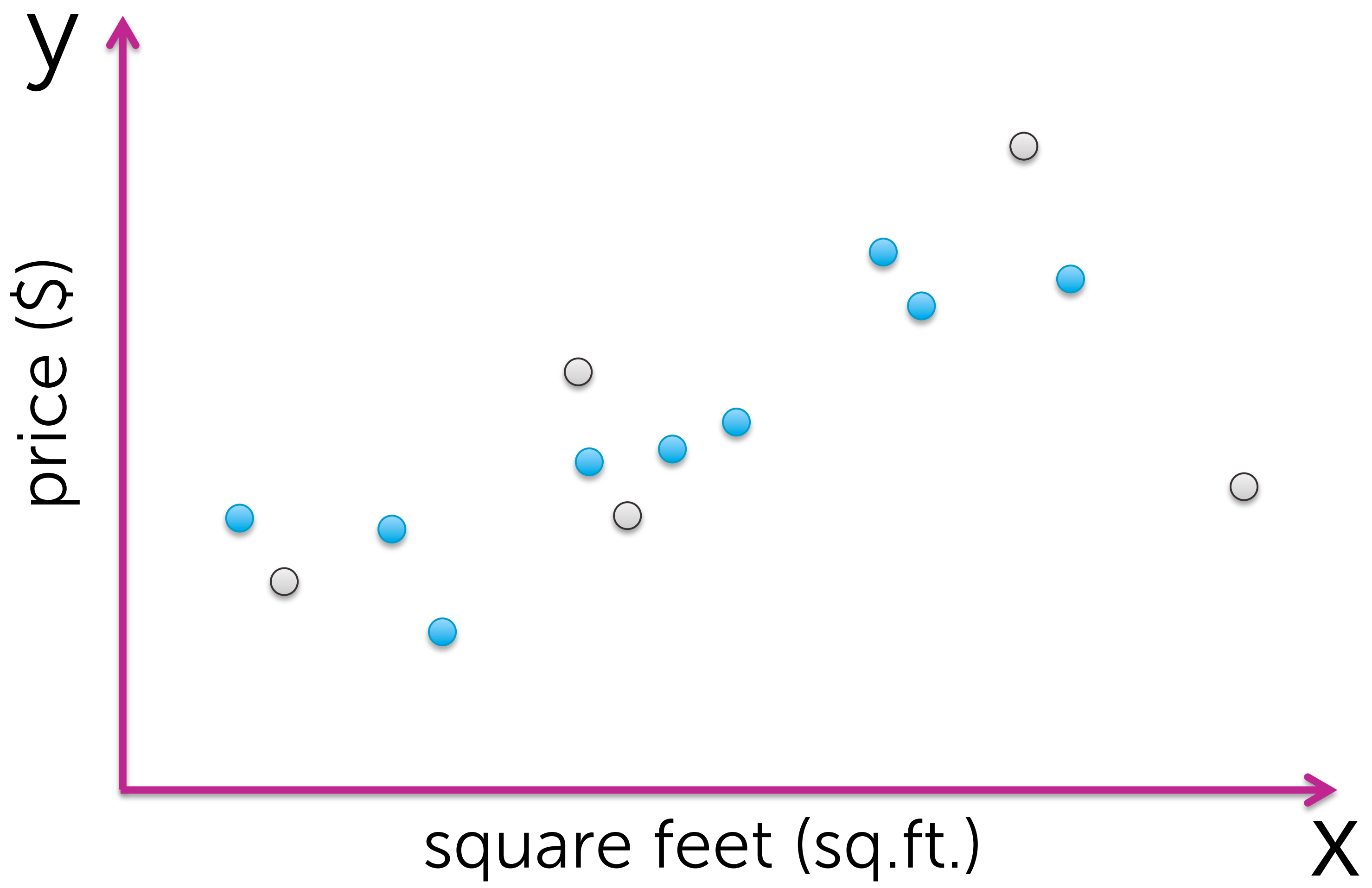
Assessing the loss

Assessing the loss
Part 1: Training error

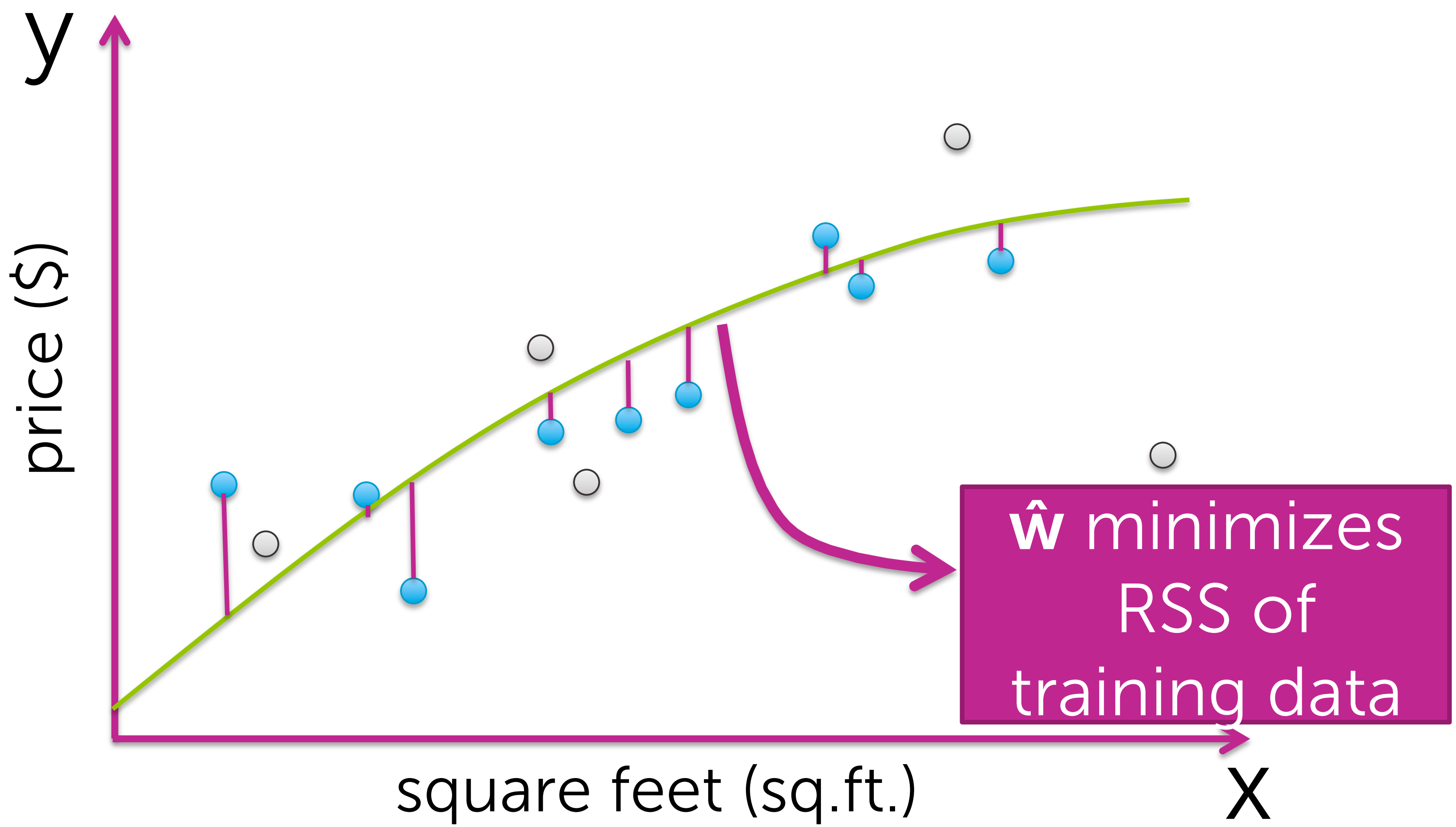
Define training data



Define training data



Example:
Fit quadratic to minimize RSS



Compute training error

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

2. Training error

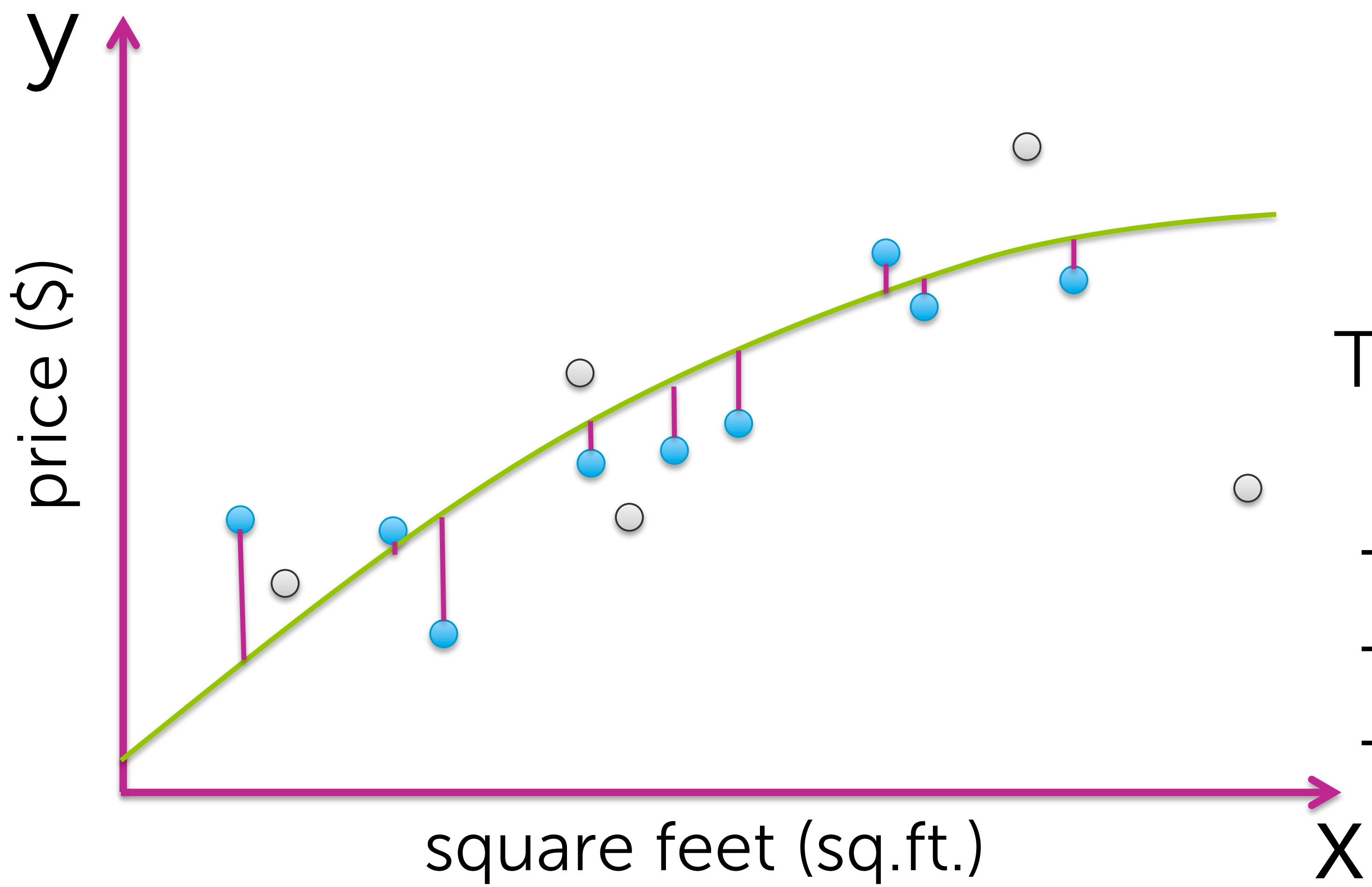
   = avg. loss on houses in training set

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

fit using training data

Example:

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



Training error ($\hat{\mathbf{w}}$) = 1/N *

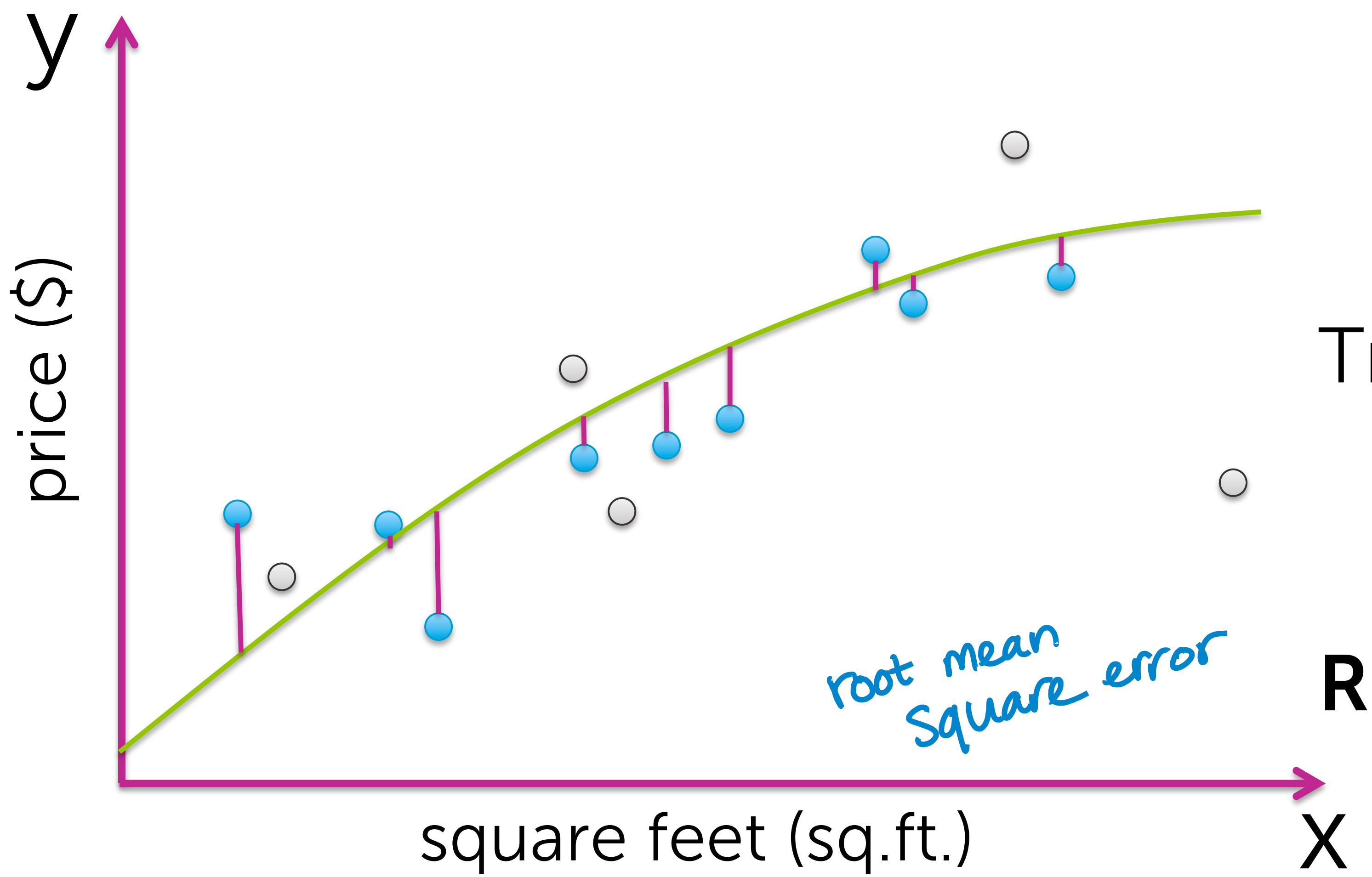
$[(\$_{train\ 1}-f_{\hat{\mathbf{w}}}(sq.ft._{train\ 1}))^2$

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

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

+ ... include all training houses]

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



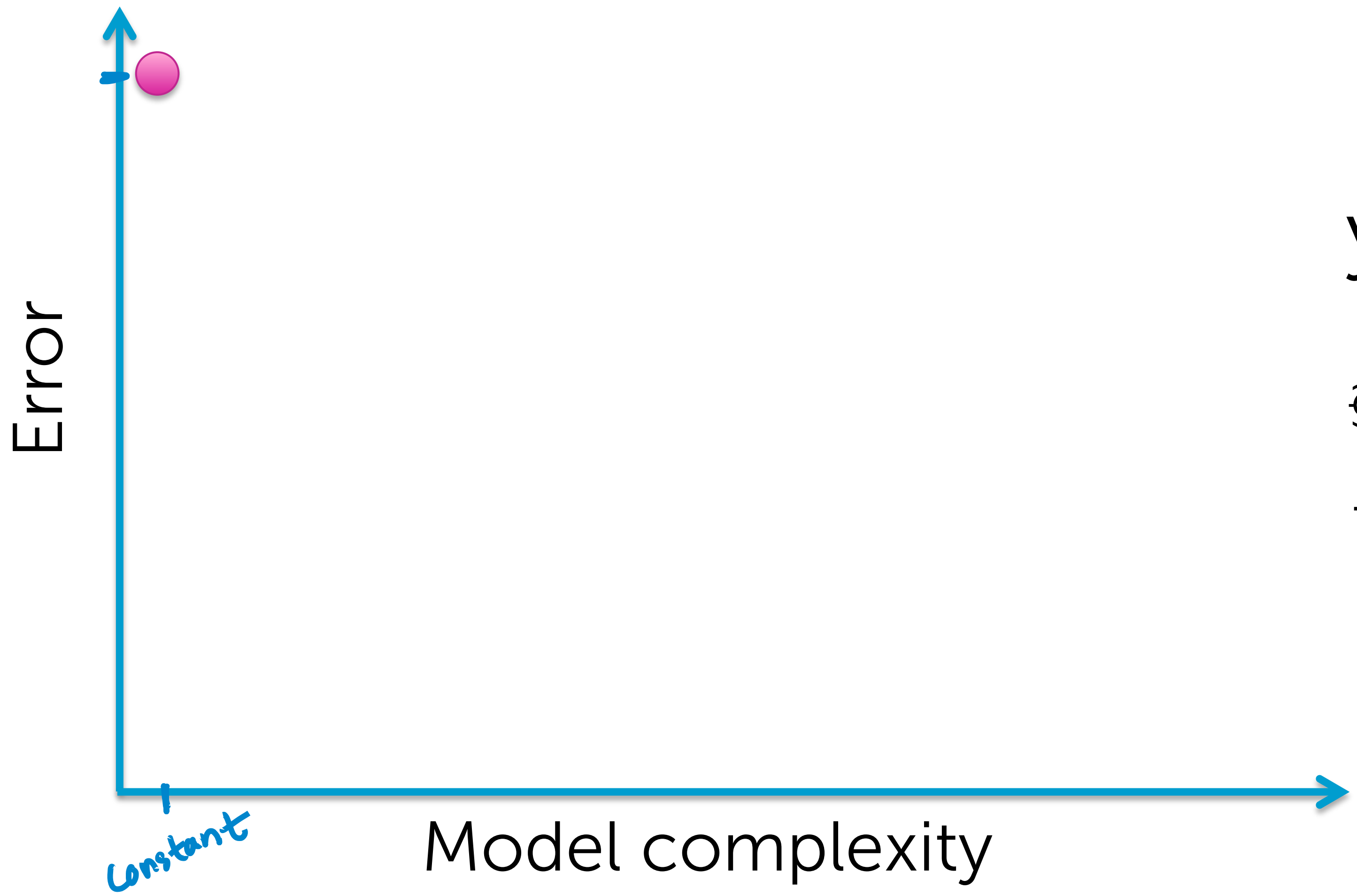
Training error ($\hat{\mathbf{w}}$) =

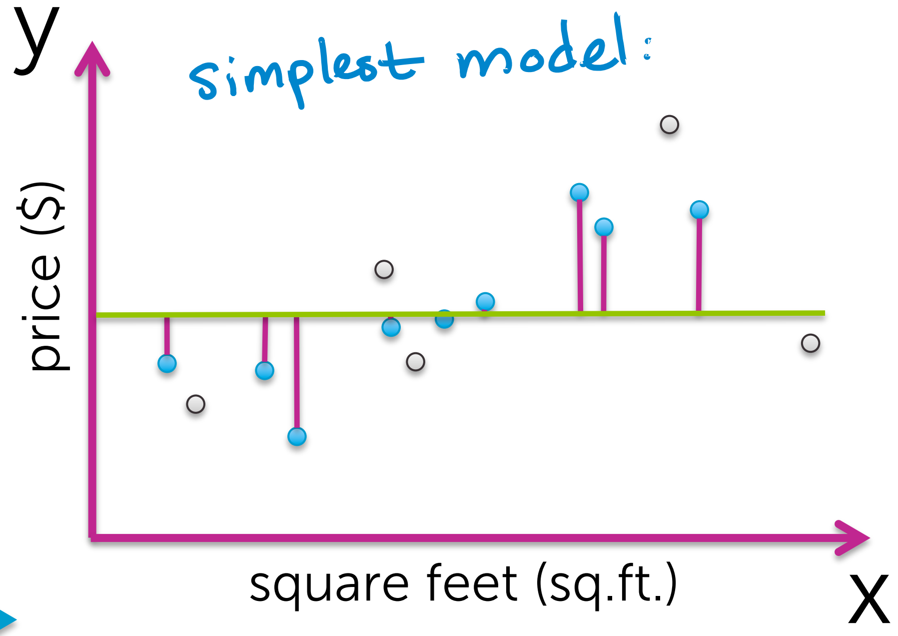
$$\frac{1}{N}\sum_{i=1}^{N}(y_i-f_{\hat{\mathbf{w}}}(\mathbf{x}_i))^2$$

RMSE =

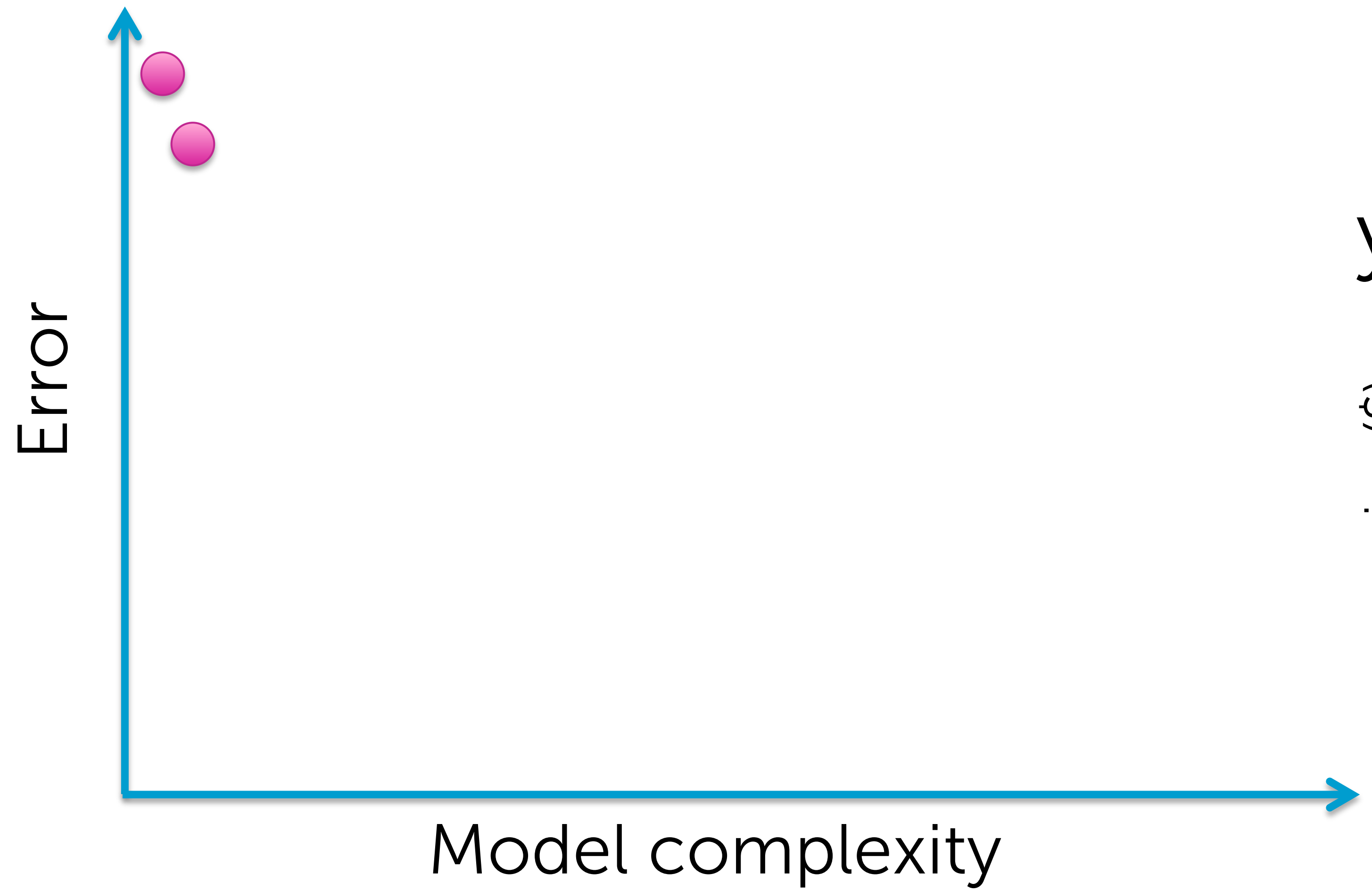
$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-f_{\hat{\mathbf{w}}}(\mathbf{x}_i))^2}$$

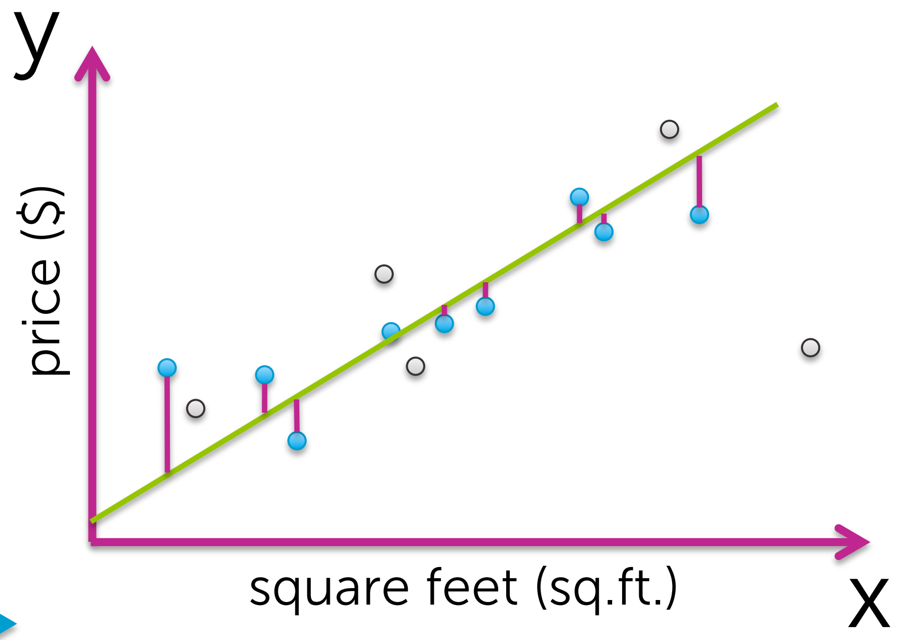
Training error vs. model complexity



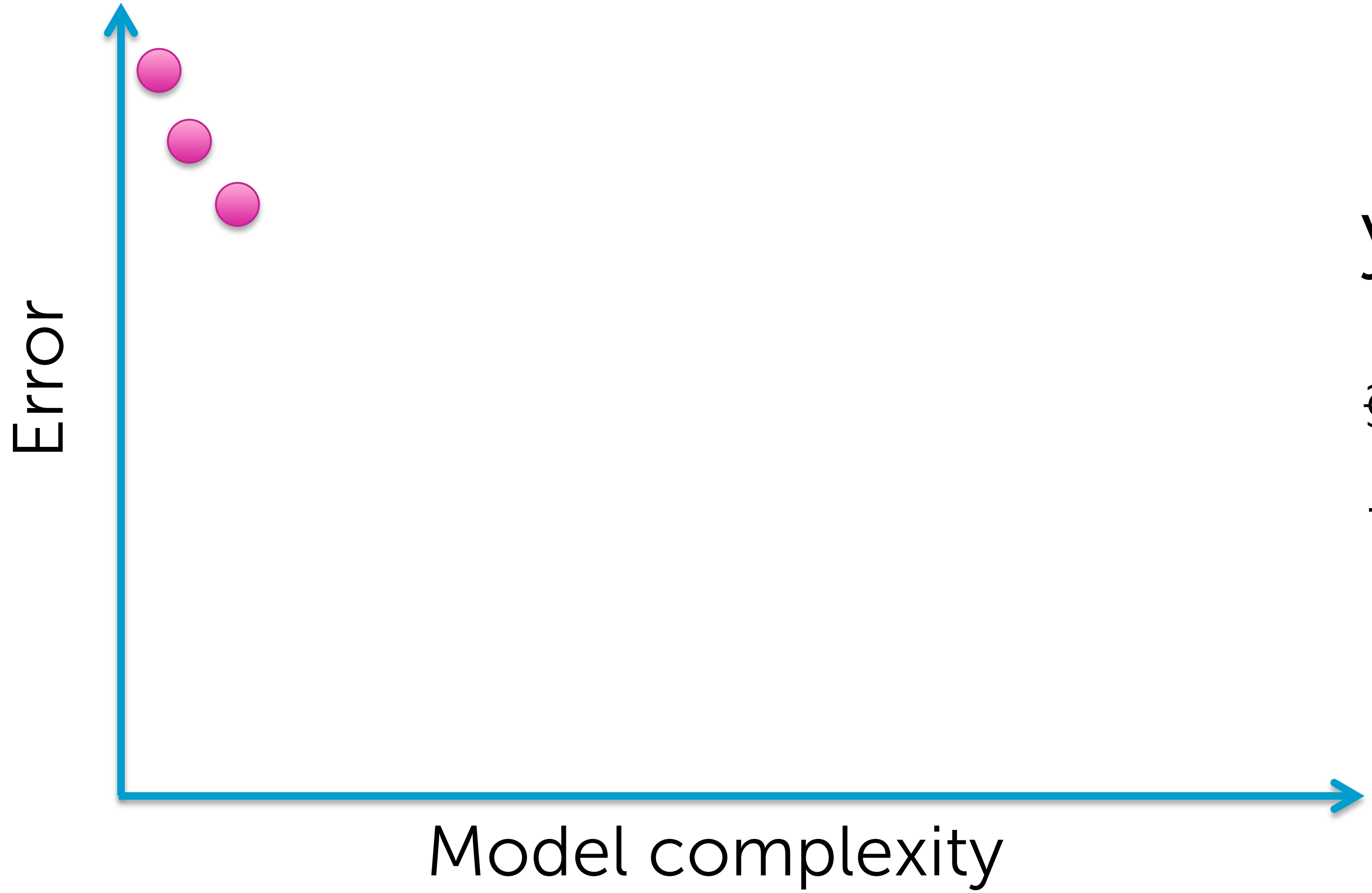


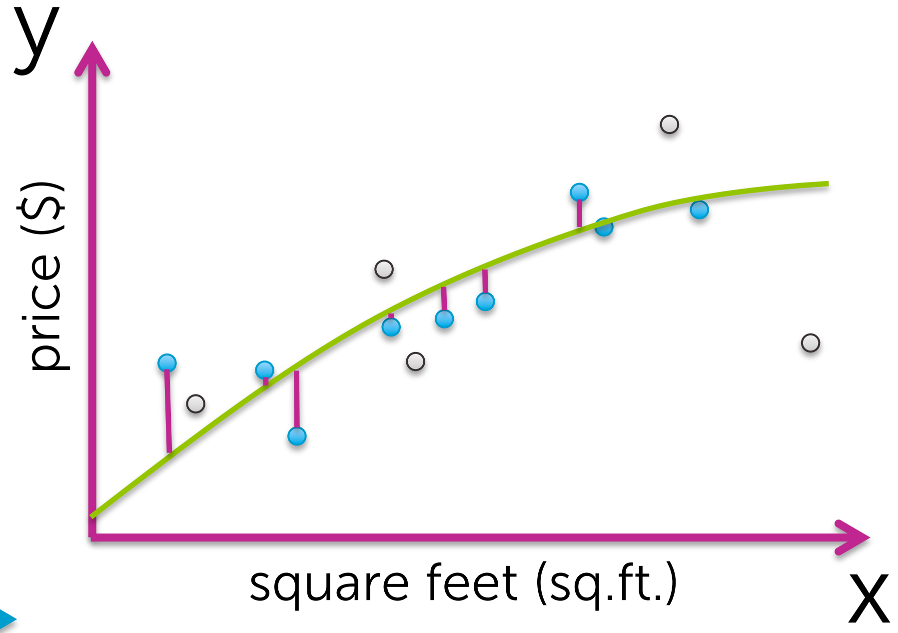
Training error vs. model complexity



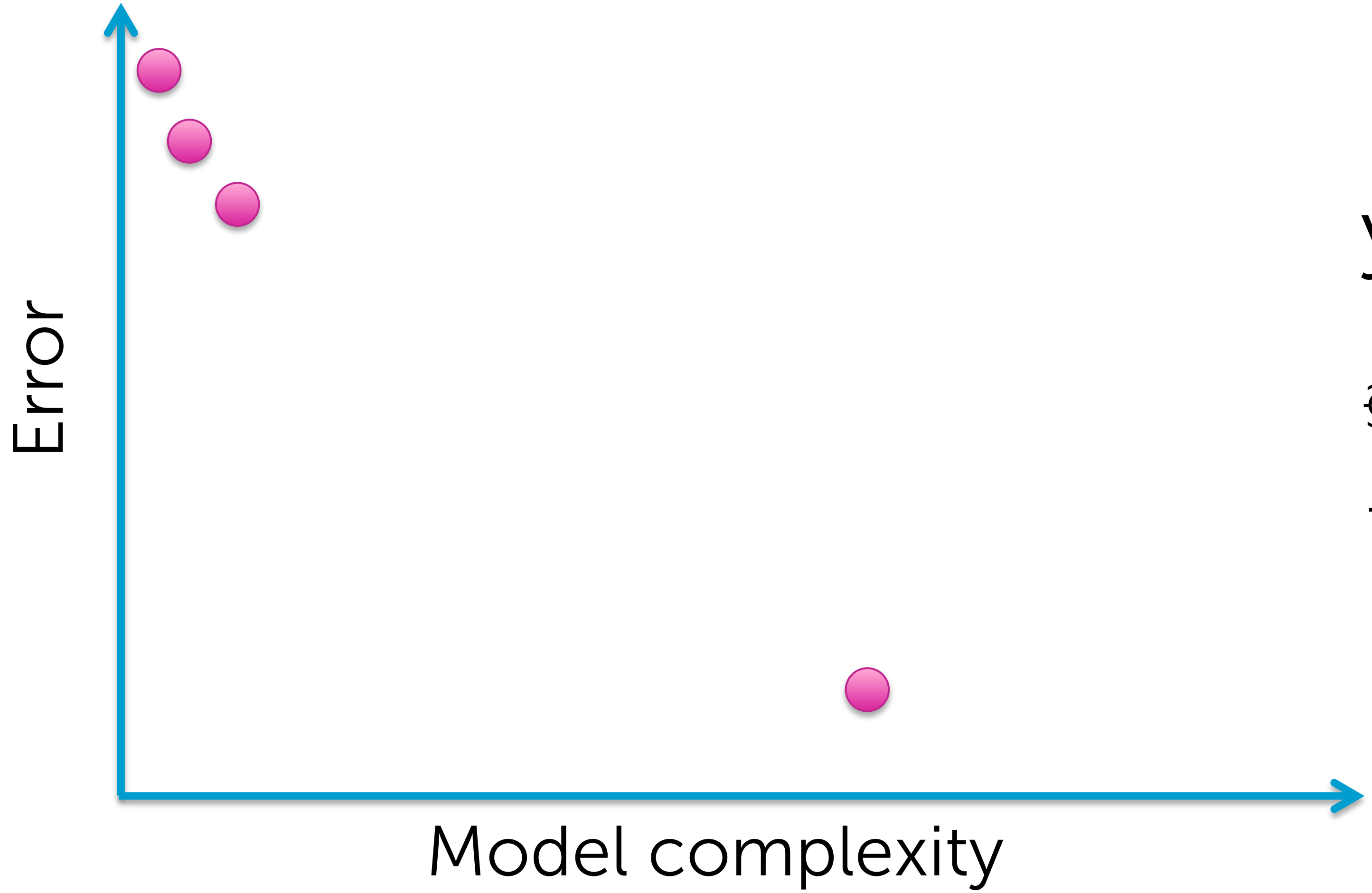


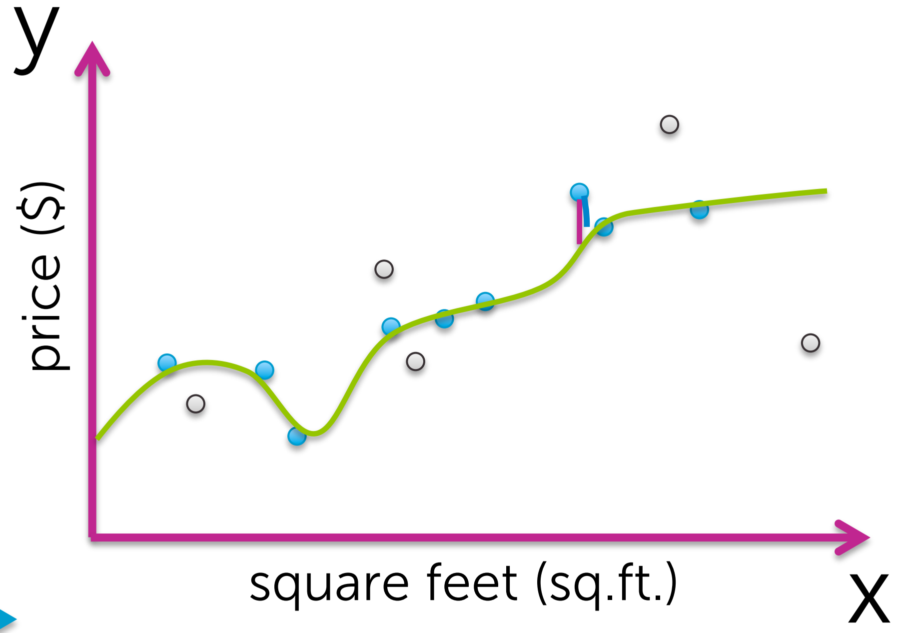
Training error vs. model complexity



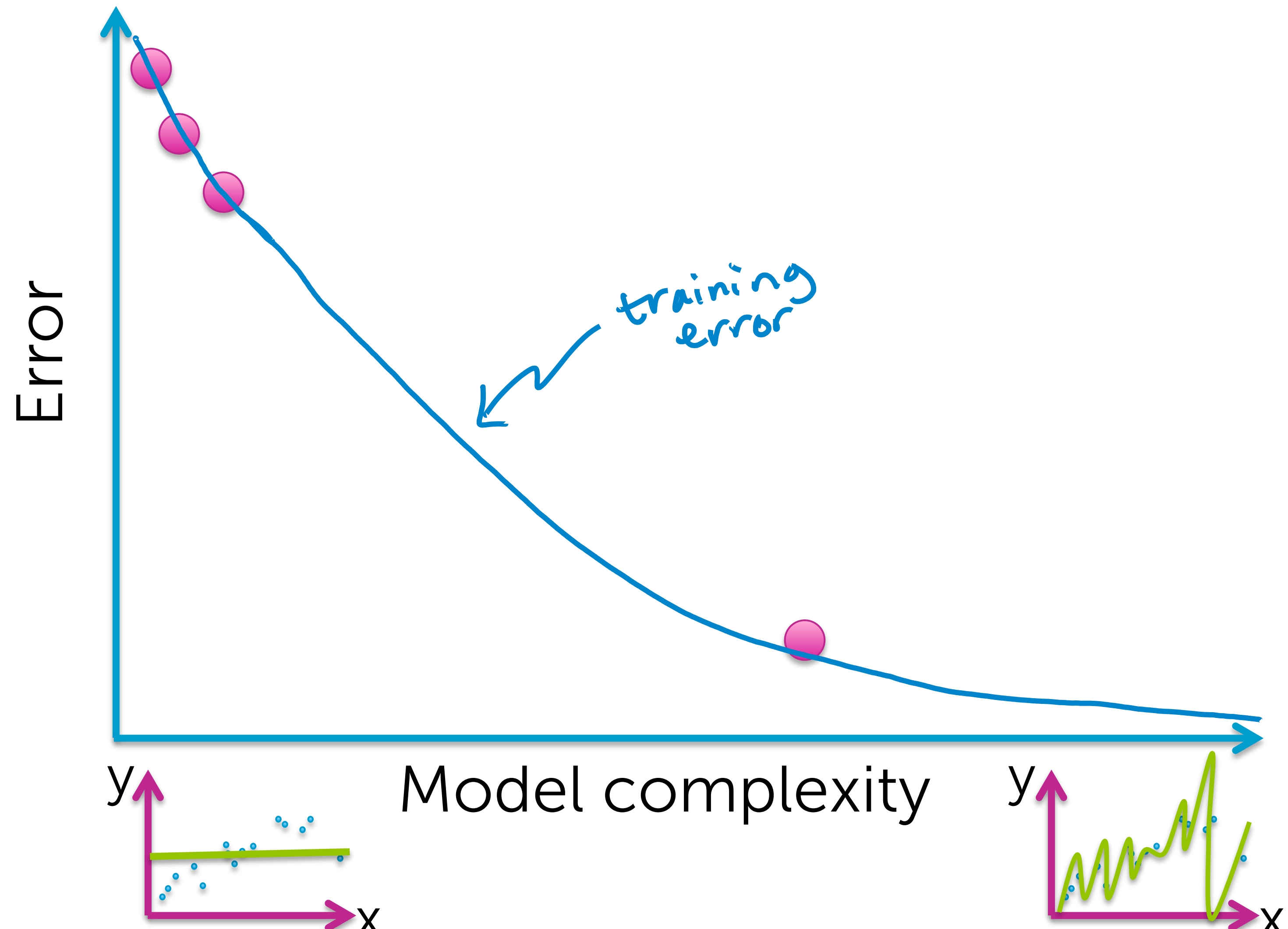


Training error vs. model complexity



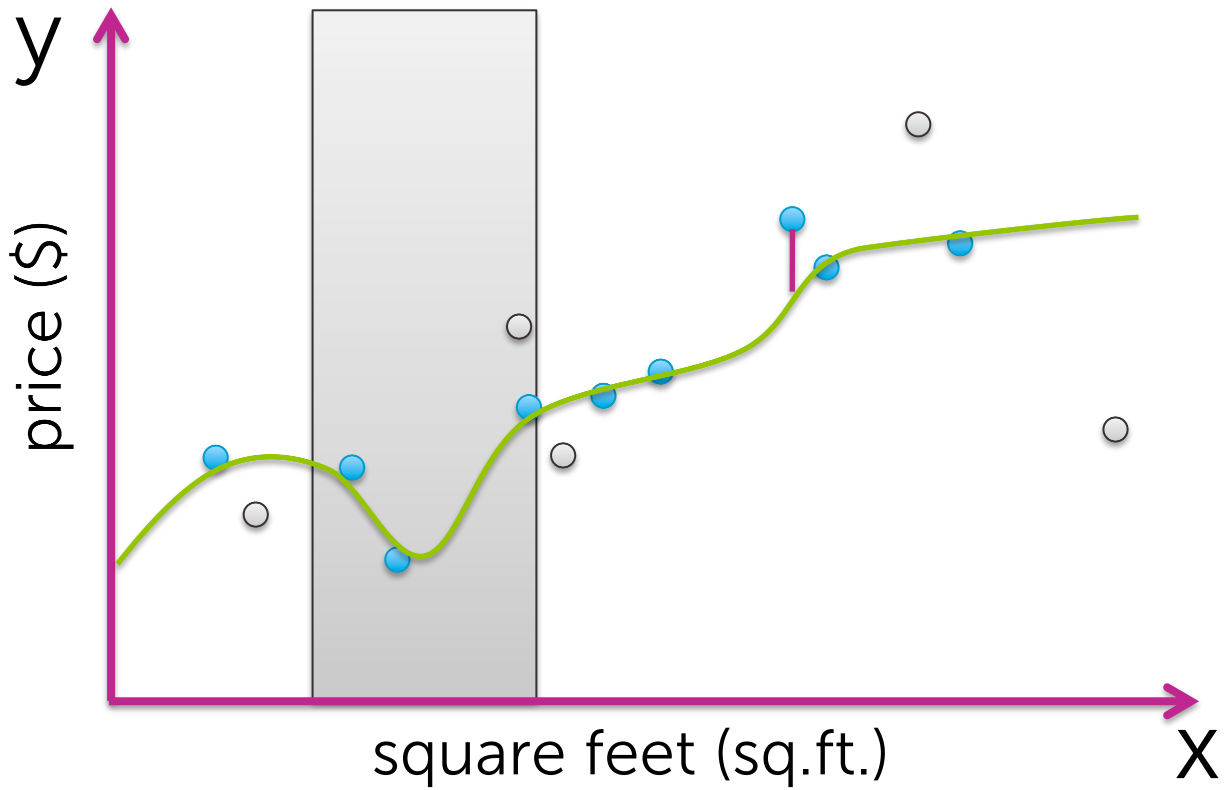


Training error vs. model complexity



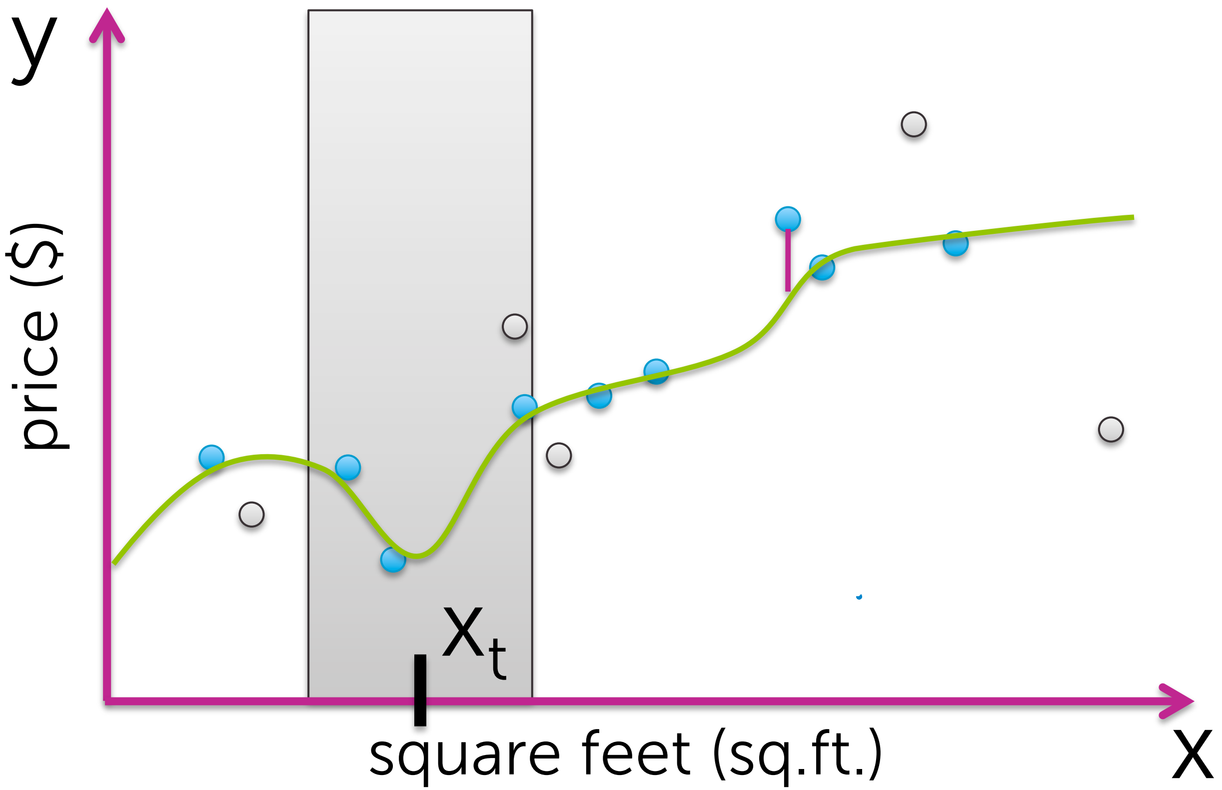
Is training error a good measure of predictive performance?

How do we expect to perform on a new house?



Is training error a good measure of predictive performance?

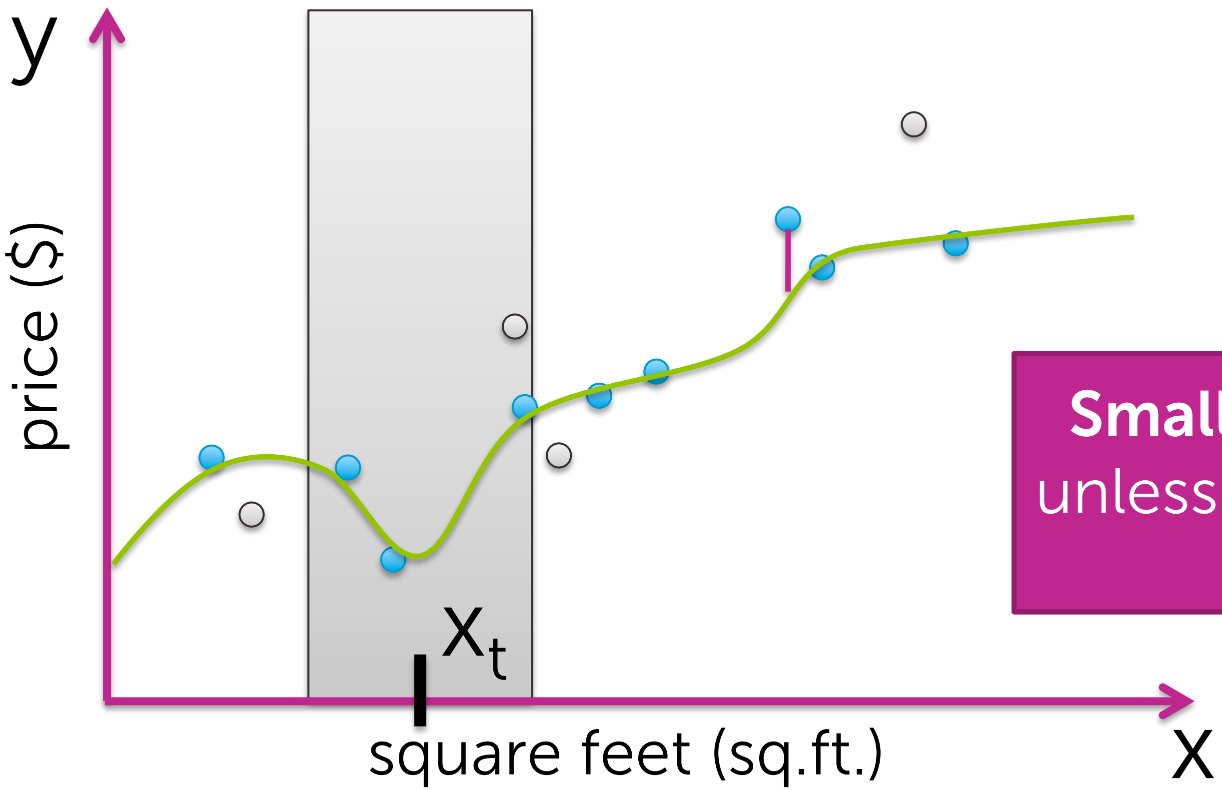
Is there something particularly bad about having x_t square feet???



Is training error a good measure of predictive performance?

Issue: Training error is overly optimistic

because $\hat{\mathbf{w}}$ 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

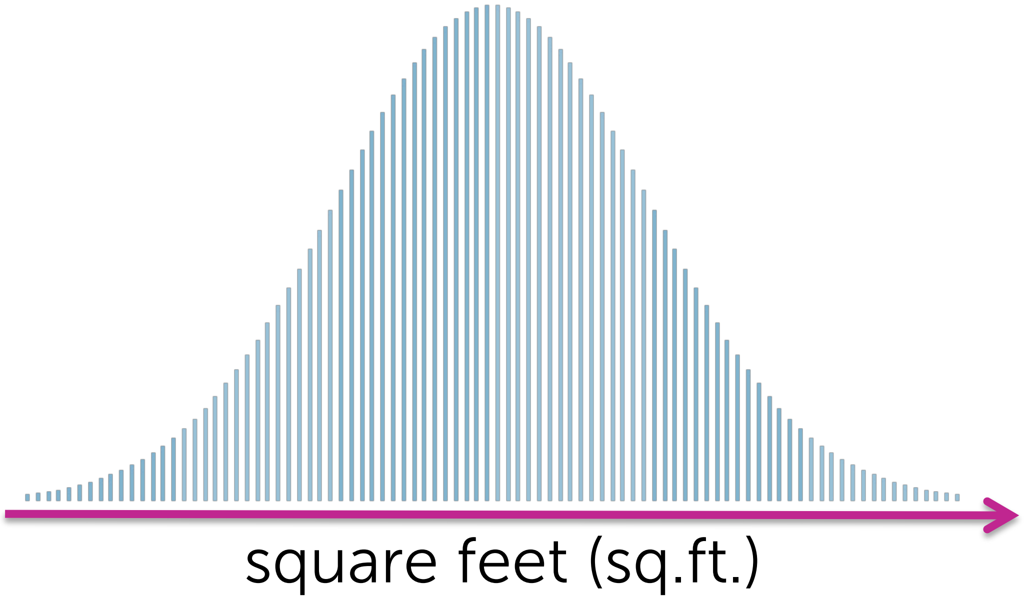
Generalization error

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

Lots of houses
in neighborhood,
but not in dataset

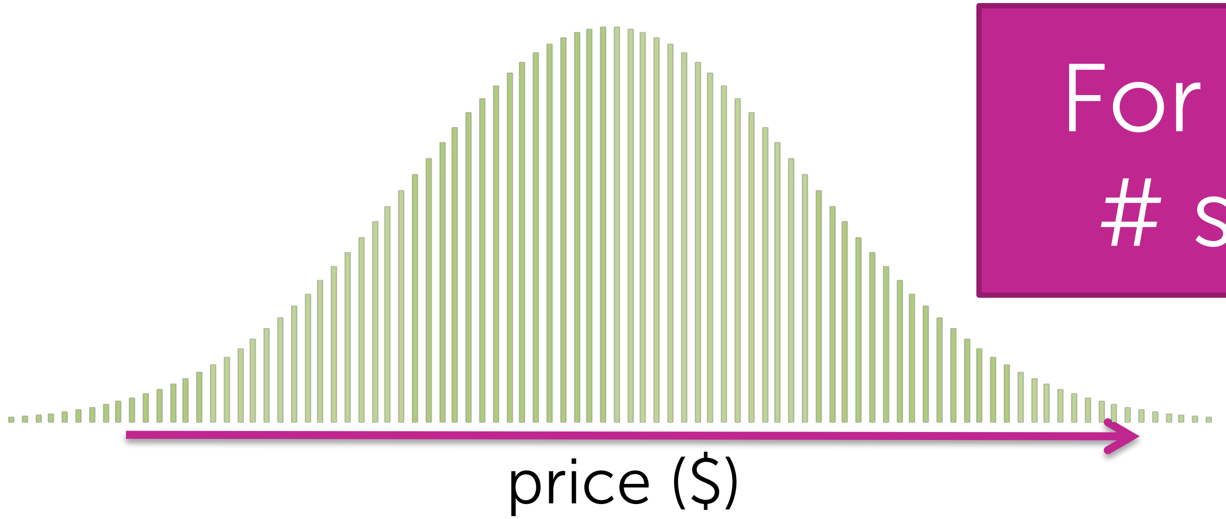
Distribution over houses

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

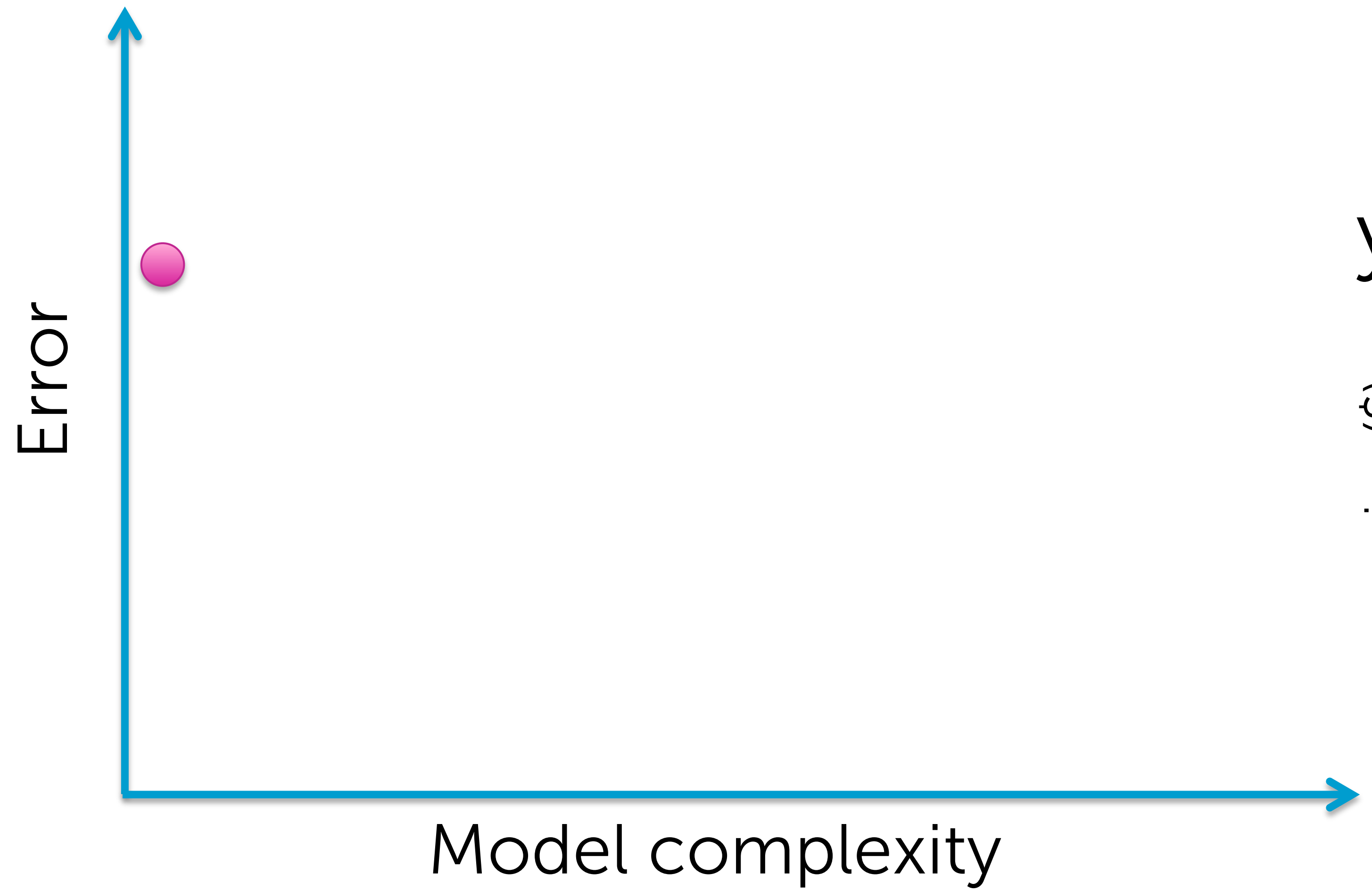
Formally:

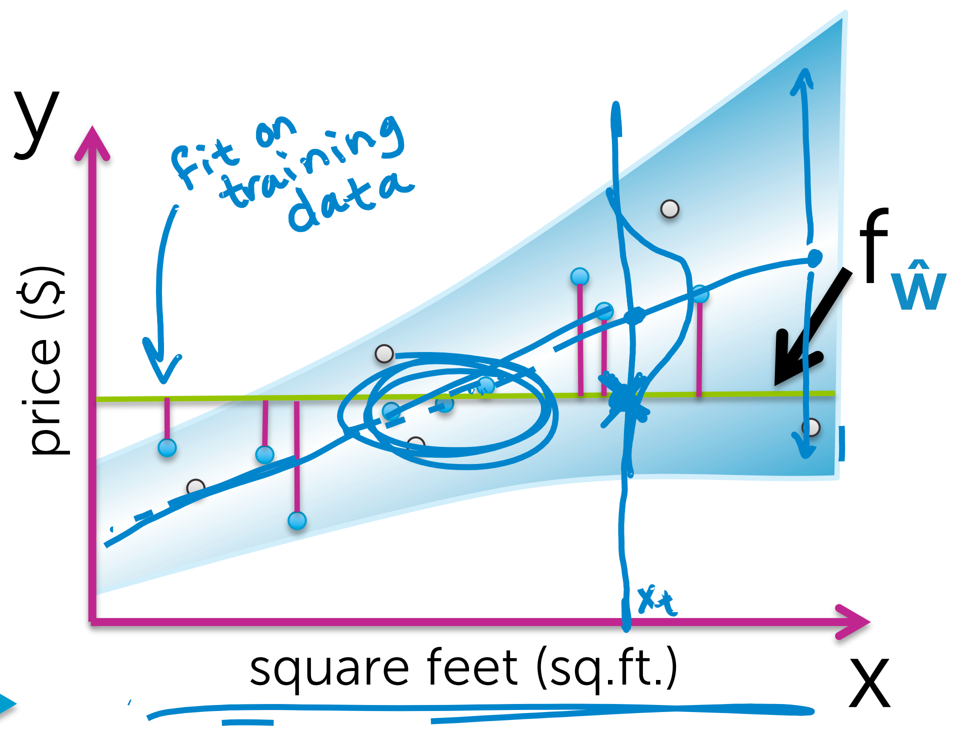
average over all possible
($\mathbf{x}$,y) pairs weighted by
how likely each is

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

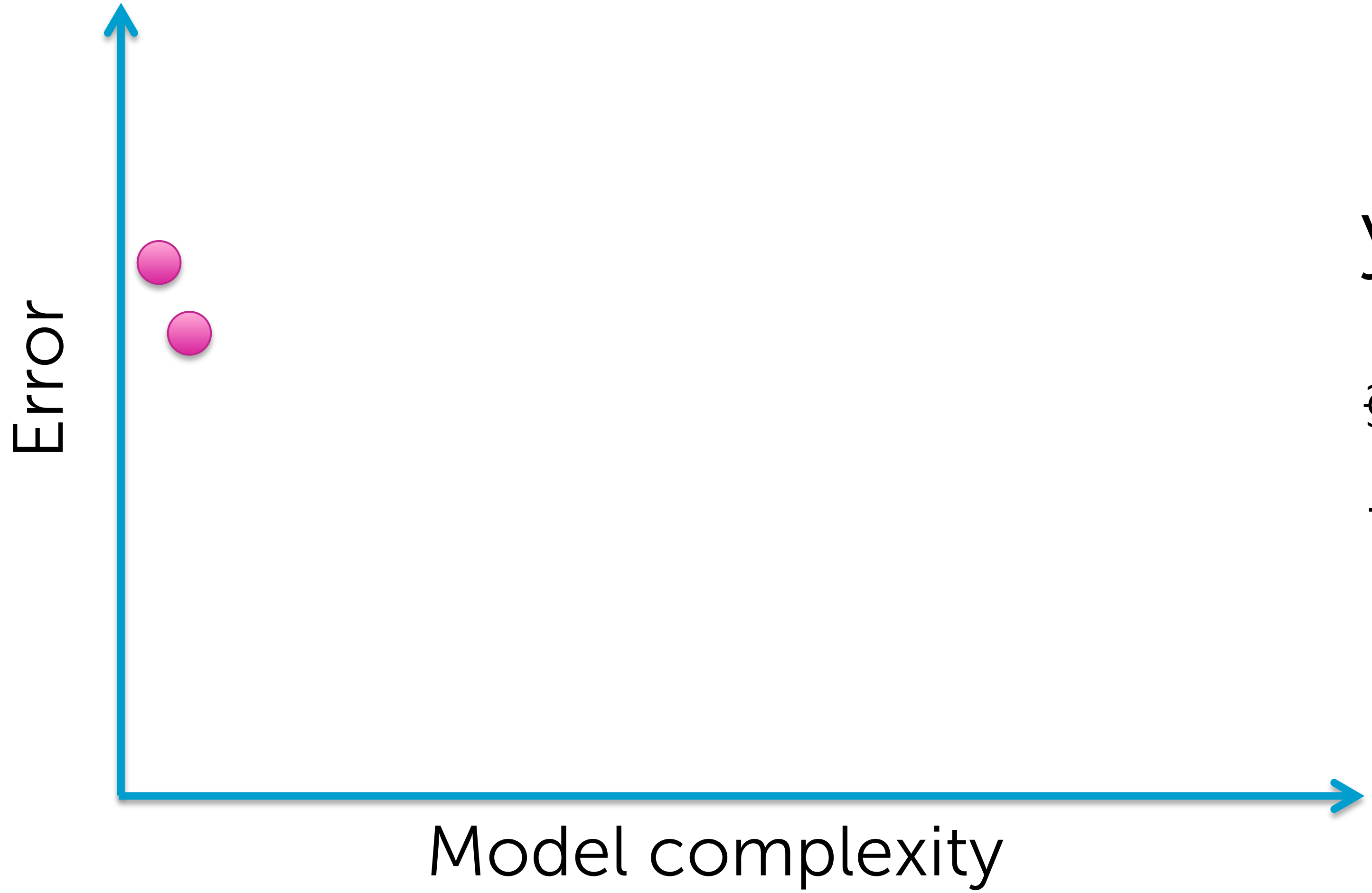
fit using training data

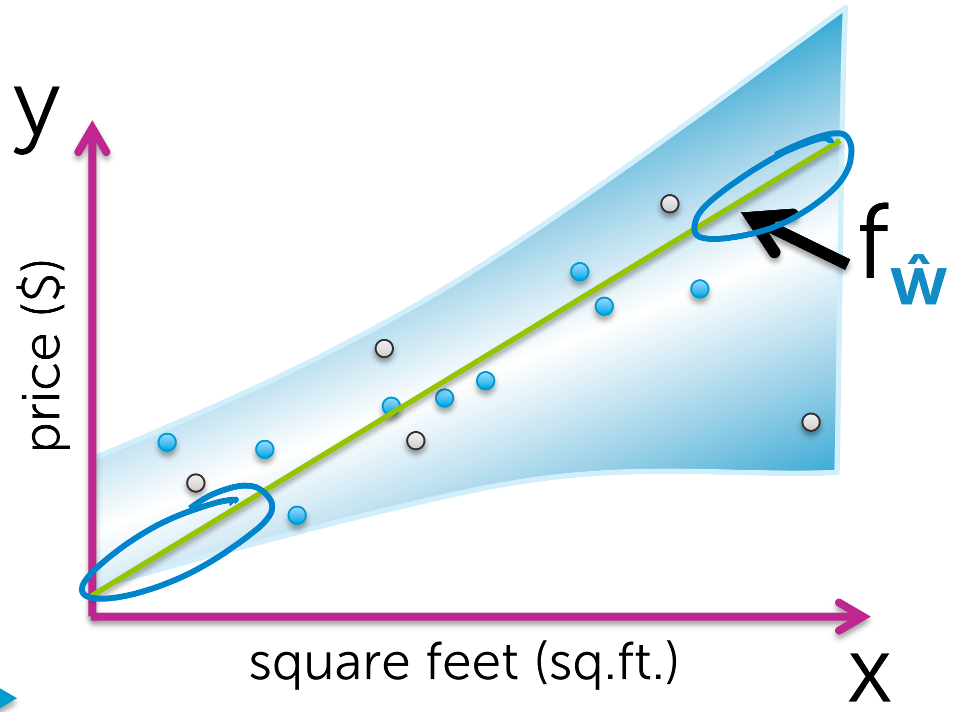
Generalization error vs. model complexity



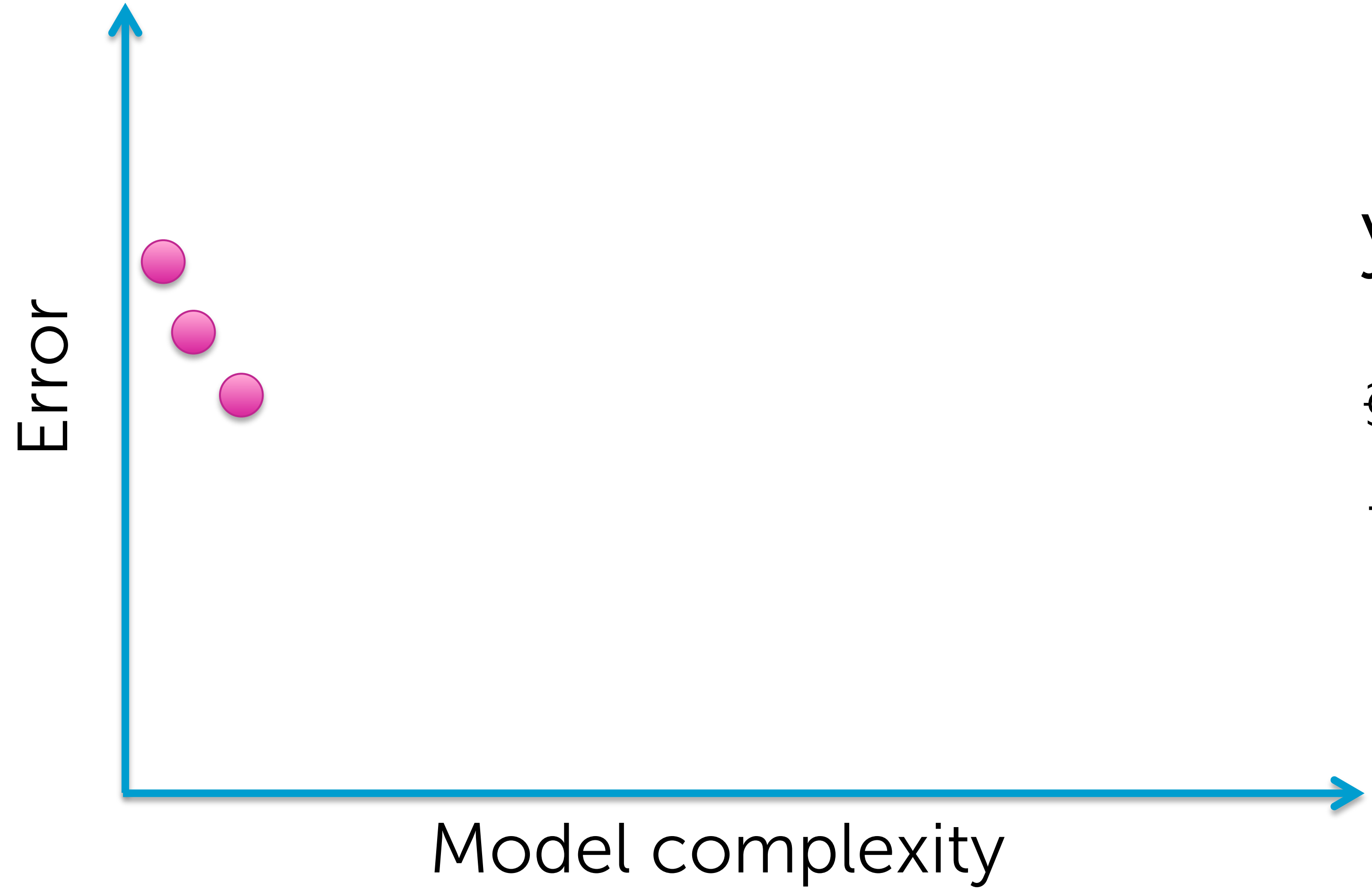


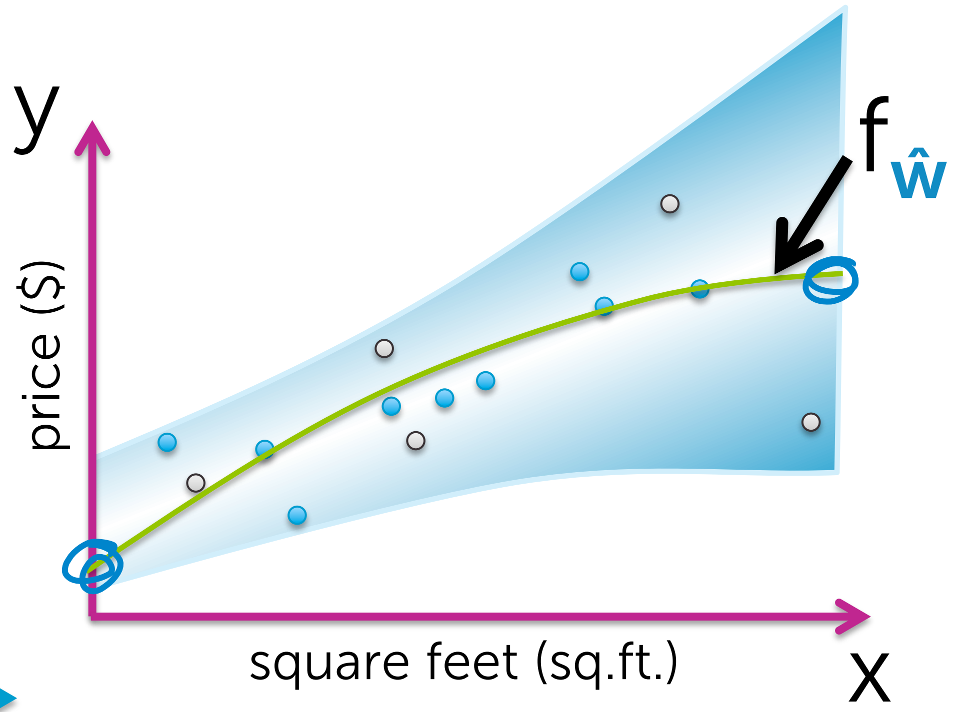
Generalization error vs. model complexity



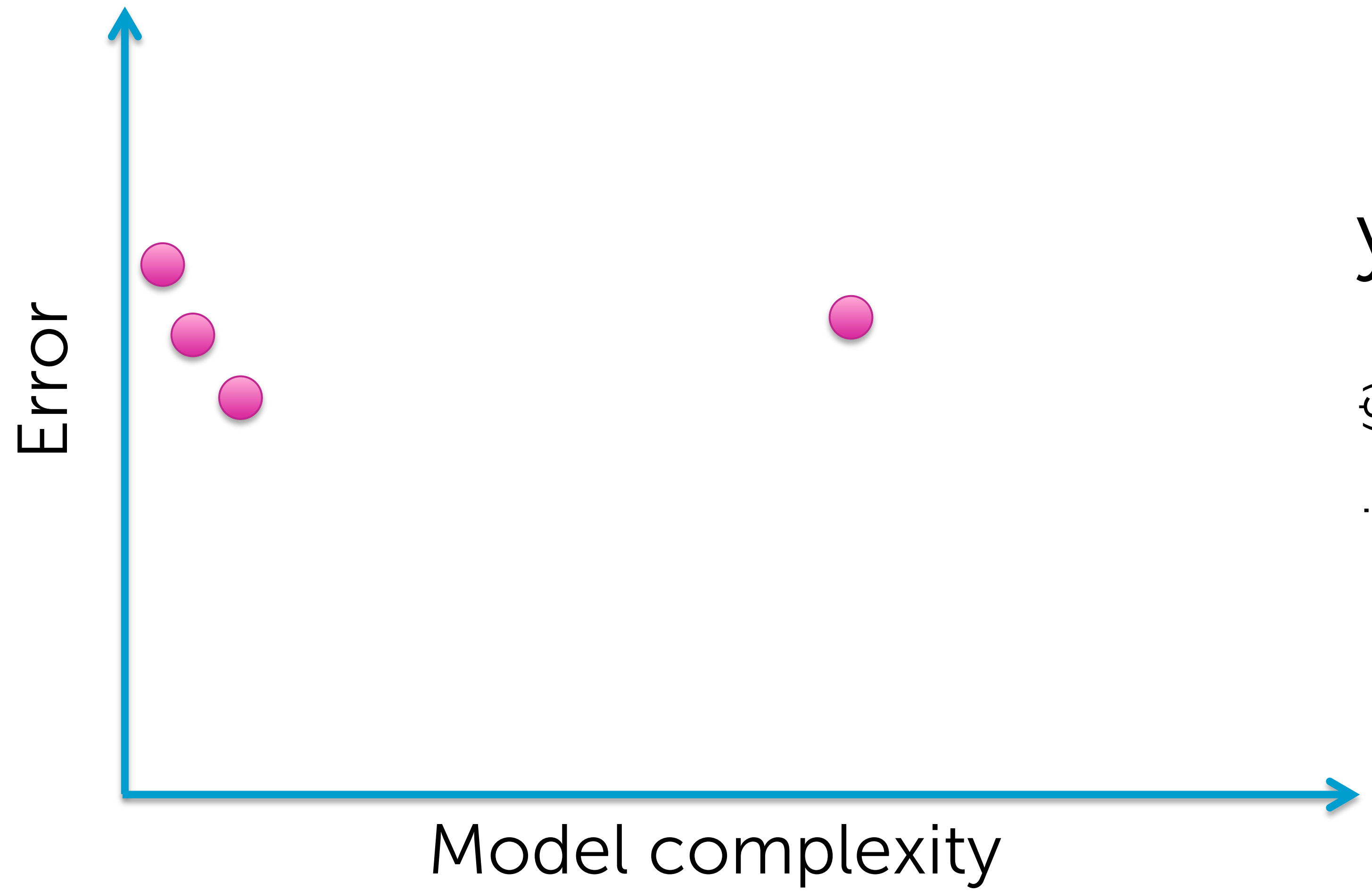


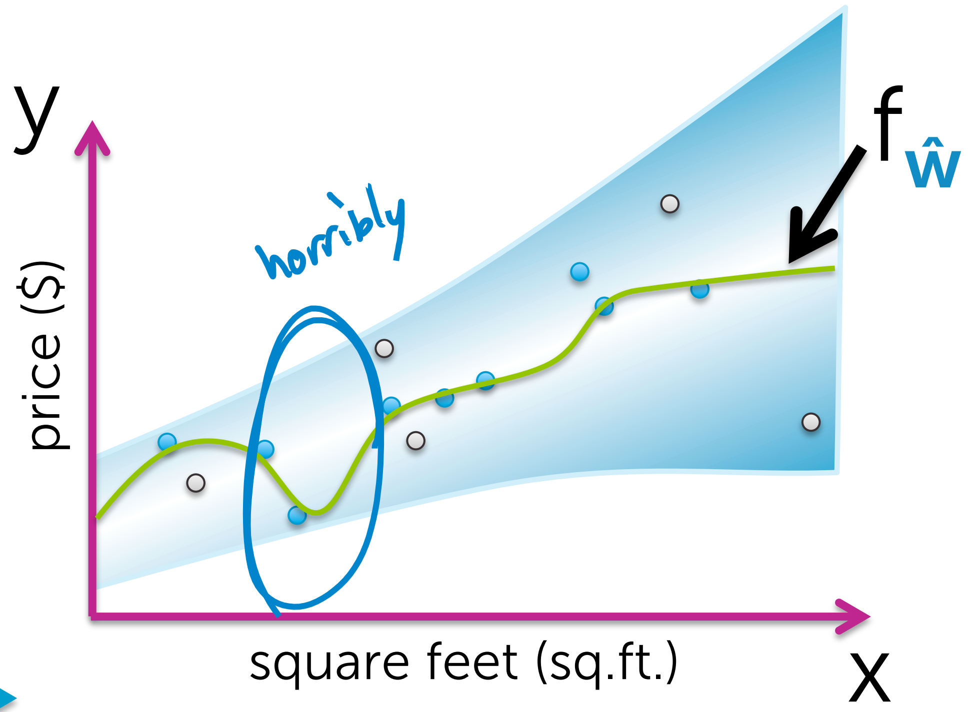
Generalization error vs. model complexity



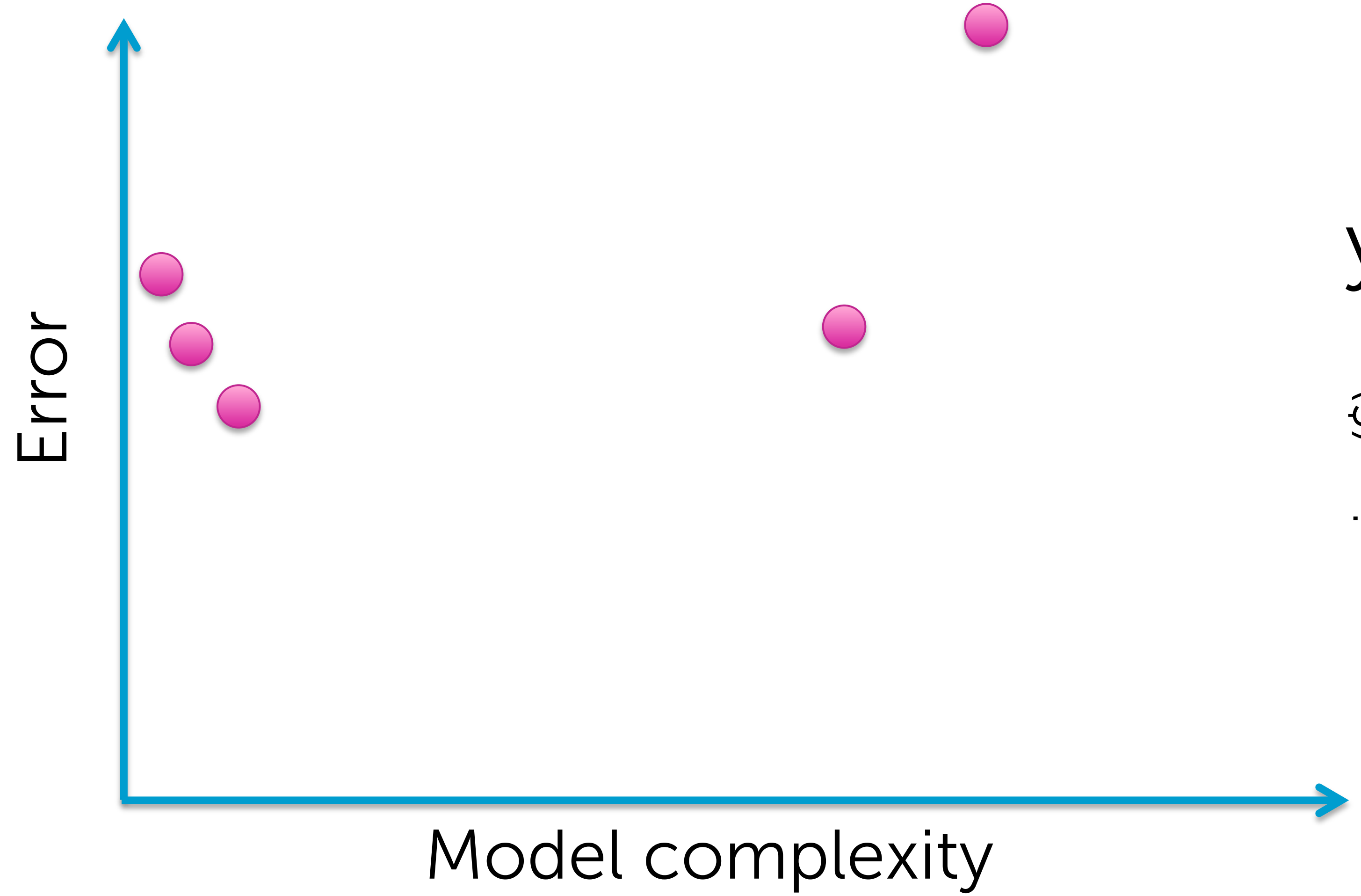


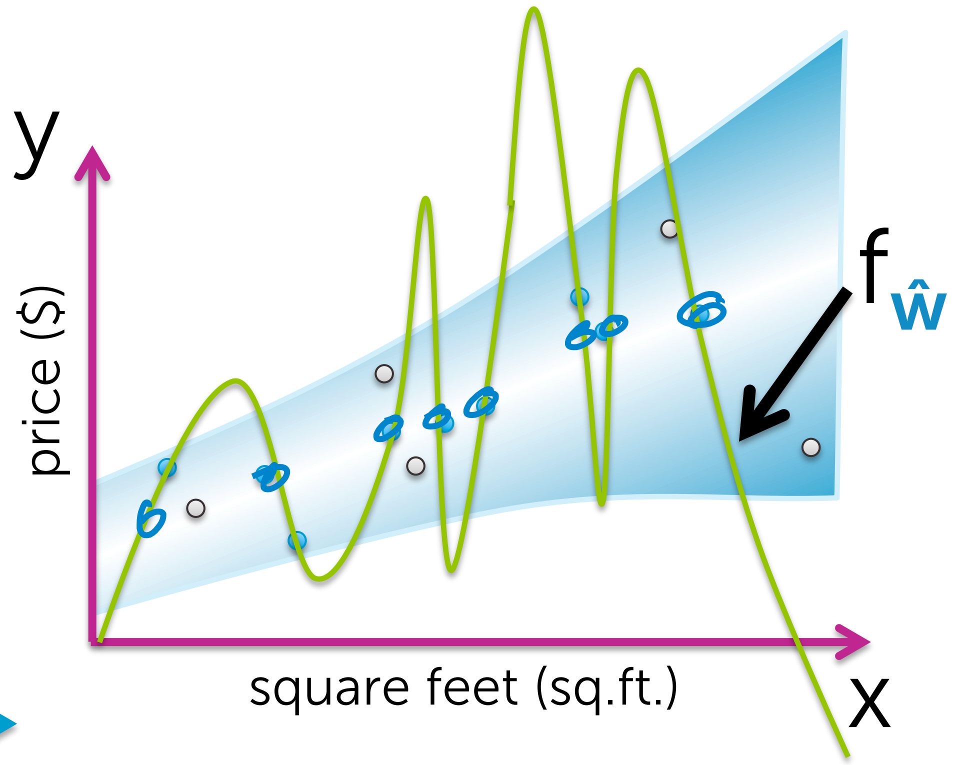
Generalization error vs. model complexity



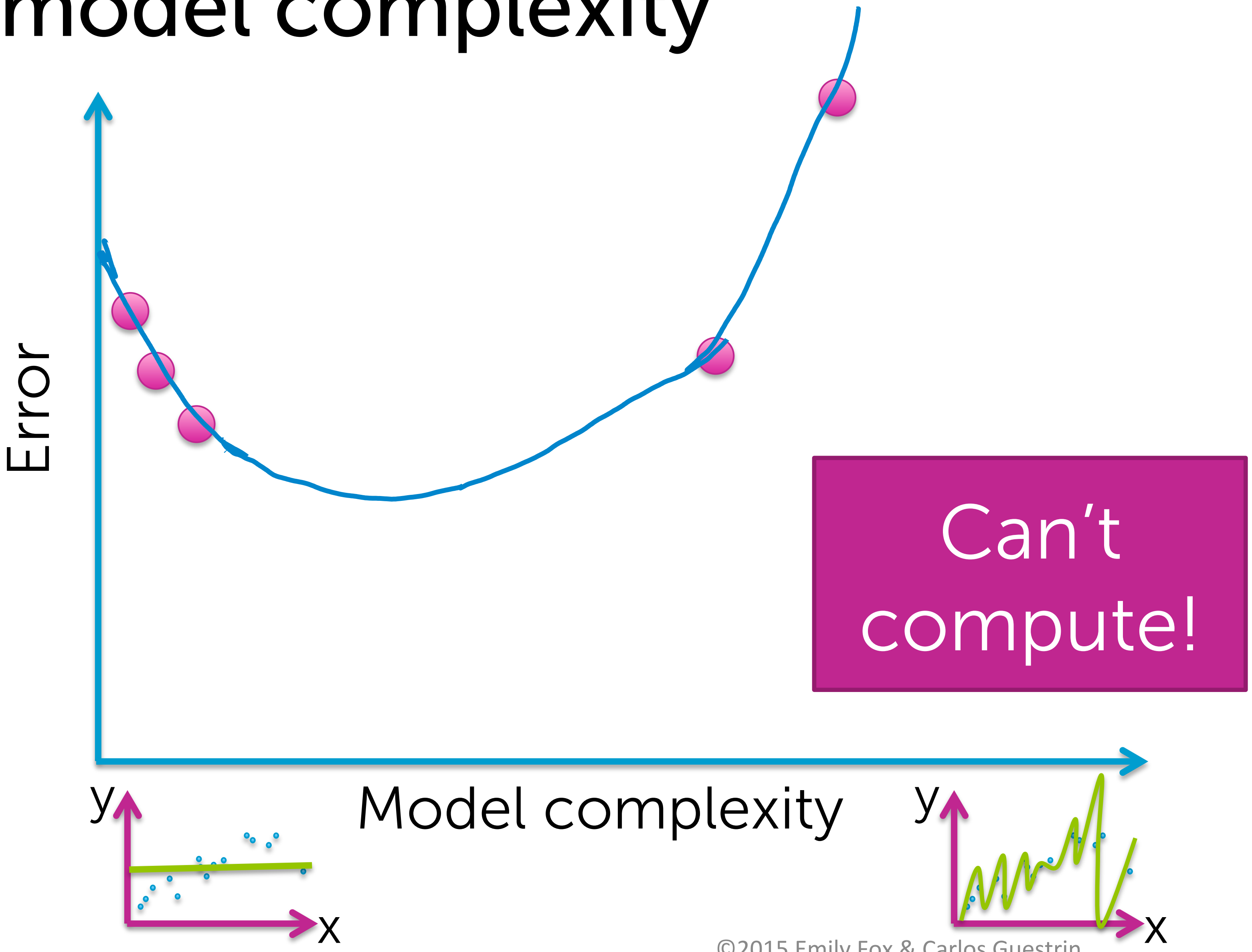


Generalization error vs. model complexity





Generalization error vs. model complexity



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

Test error

= avg. loss on houses in test set

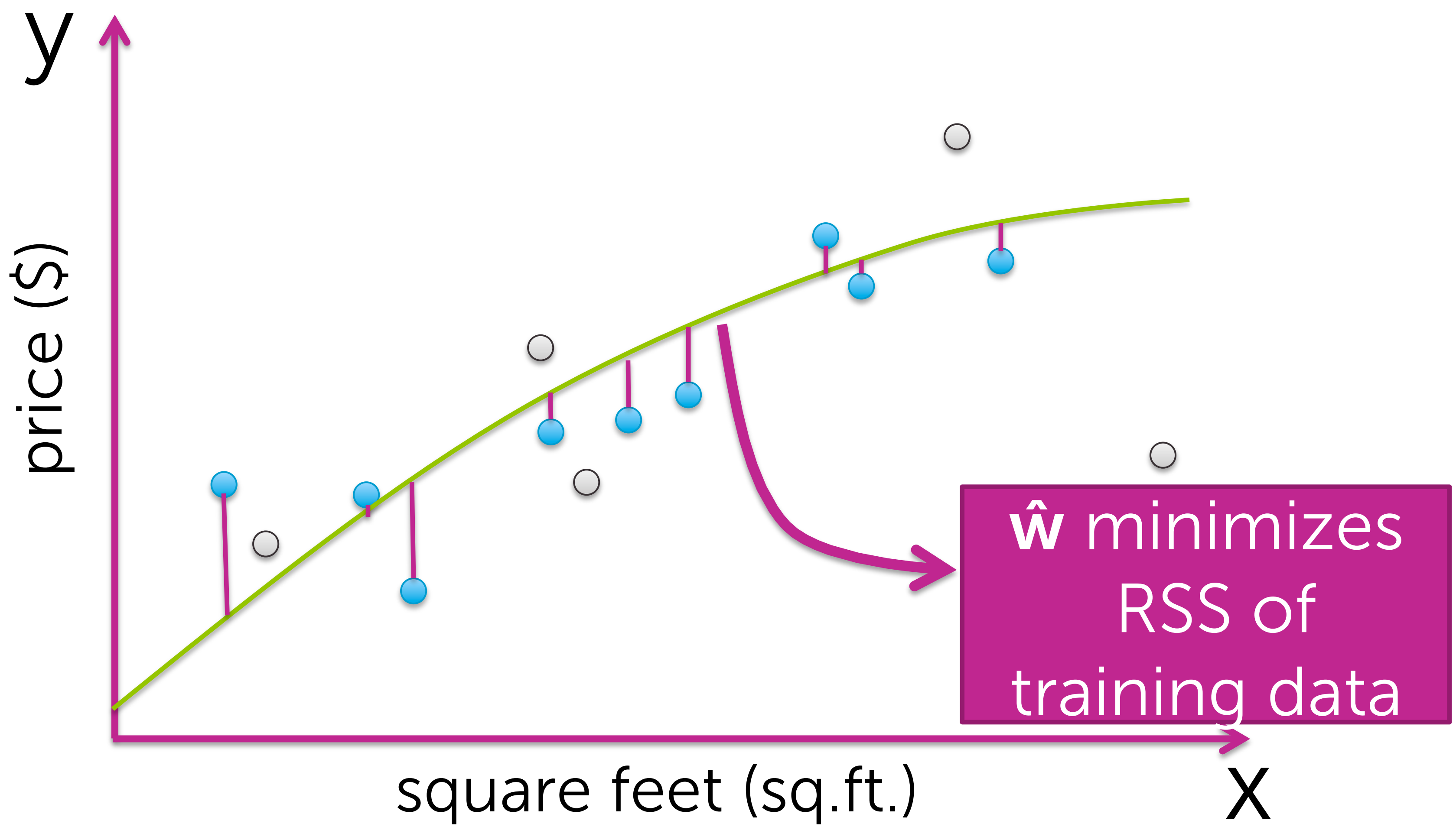
$$= \frac{1}{N_{test}} \sum_{i \text{ in test set}} L(y_i, f_{\hat{\mathbf{w}}}(\mathbf{x}_i))$$

test points

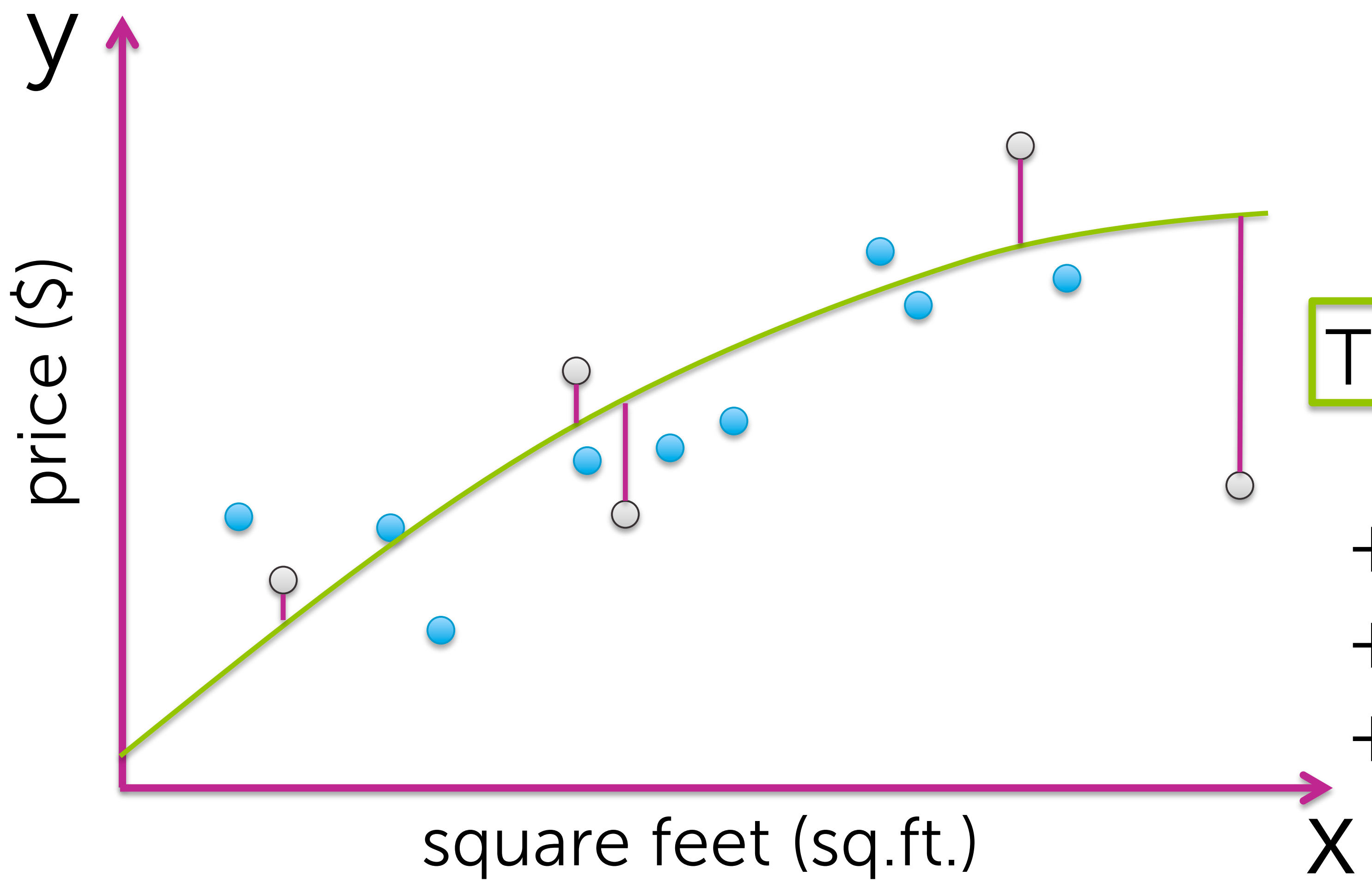
fit using training data

has never seen test data!

Example: As before, fit quadratic to training data



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



$\boxed{\text{Test error } (\hat{\mathbf{w}})} = 1/N \; *$

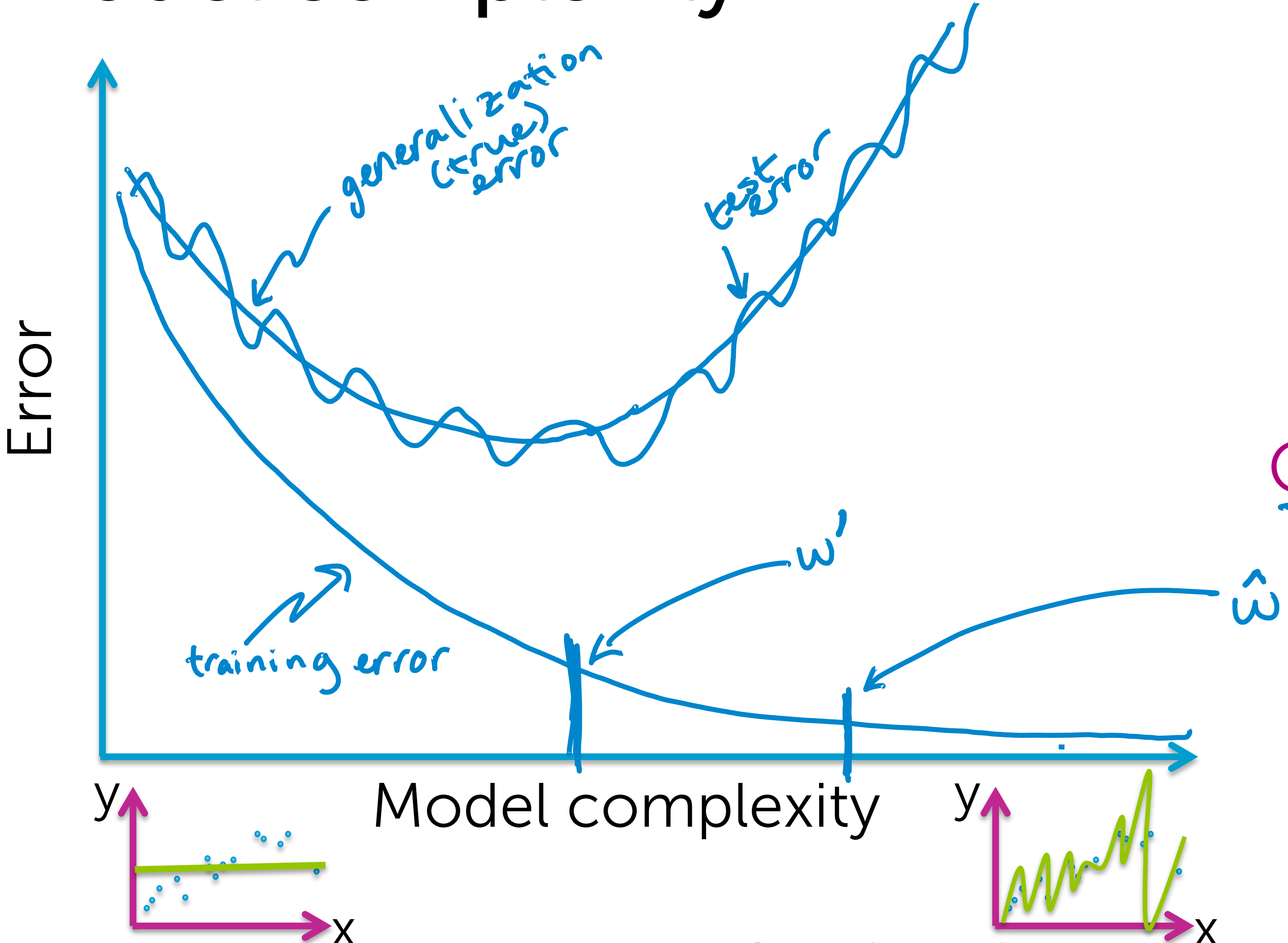
$[(\$_{\text{test 1}} - f_{\hat{\mathbf{w}}}(\text{sq.ft.}_{\text{test 1}}))^2$

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

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

+ ... include all test houses]

Training, true, & test error vs. model complexity



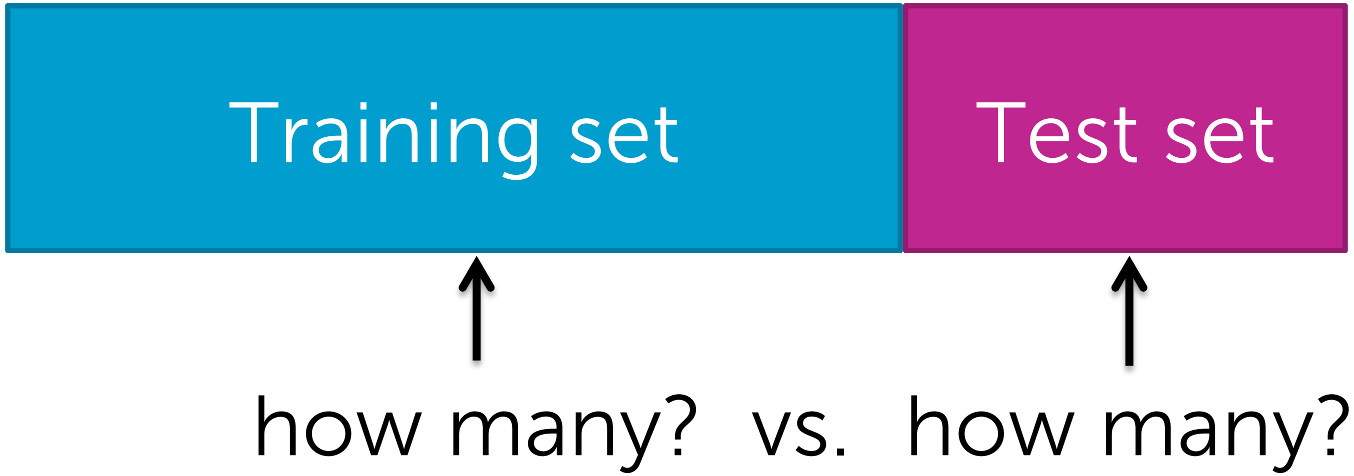
Overfitting if:

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

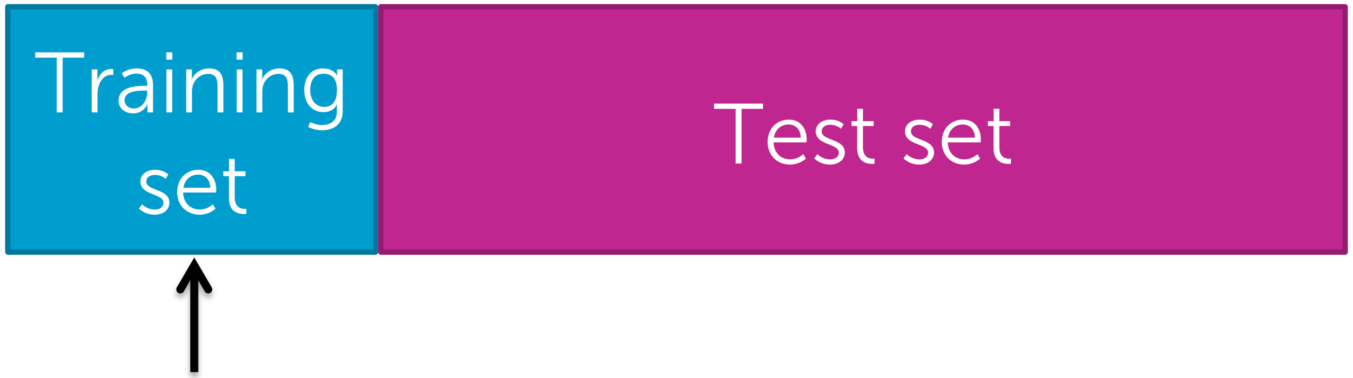
1. training error $(\hat{w})$ < training error (w')
2. true error $(\hat{w})$ > true error (w')

Training/test split

Training/test splits



Training/test splits



Too few → $\hat{\mathbf{w}}$ 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** (will see later…)

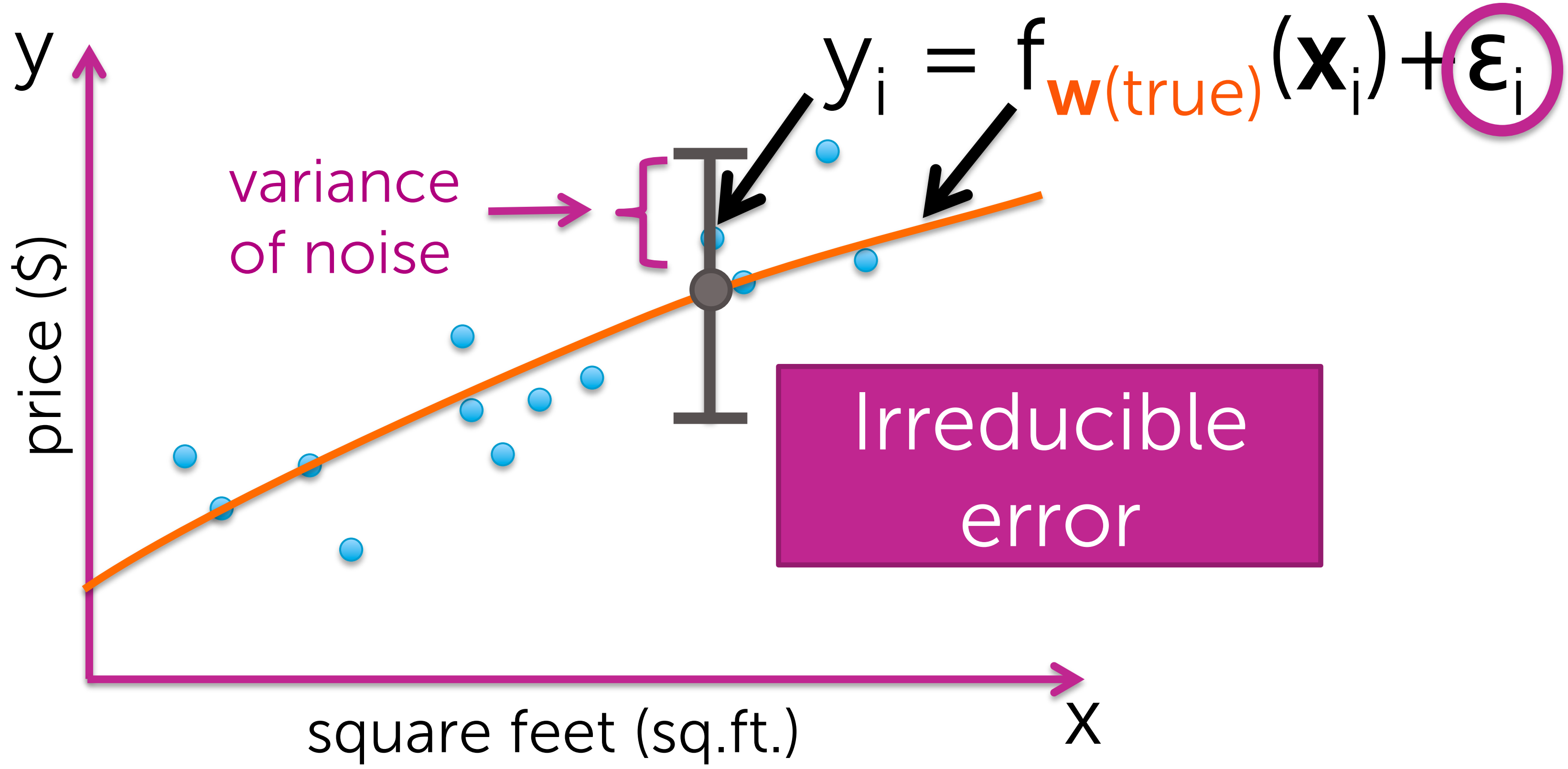
3 sources of error + the bias-variance tradeoff

3 sources of error

In forming predictions, there are 3 sources of error:

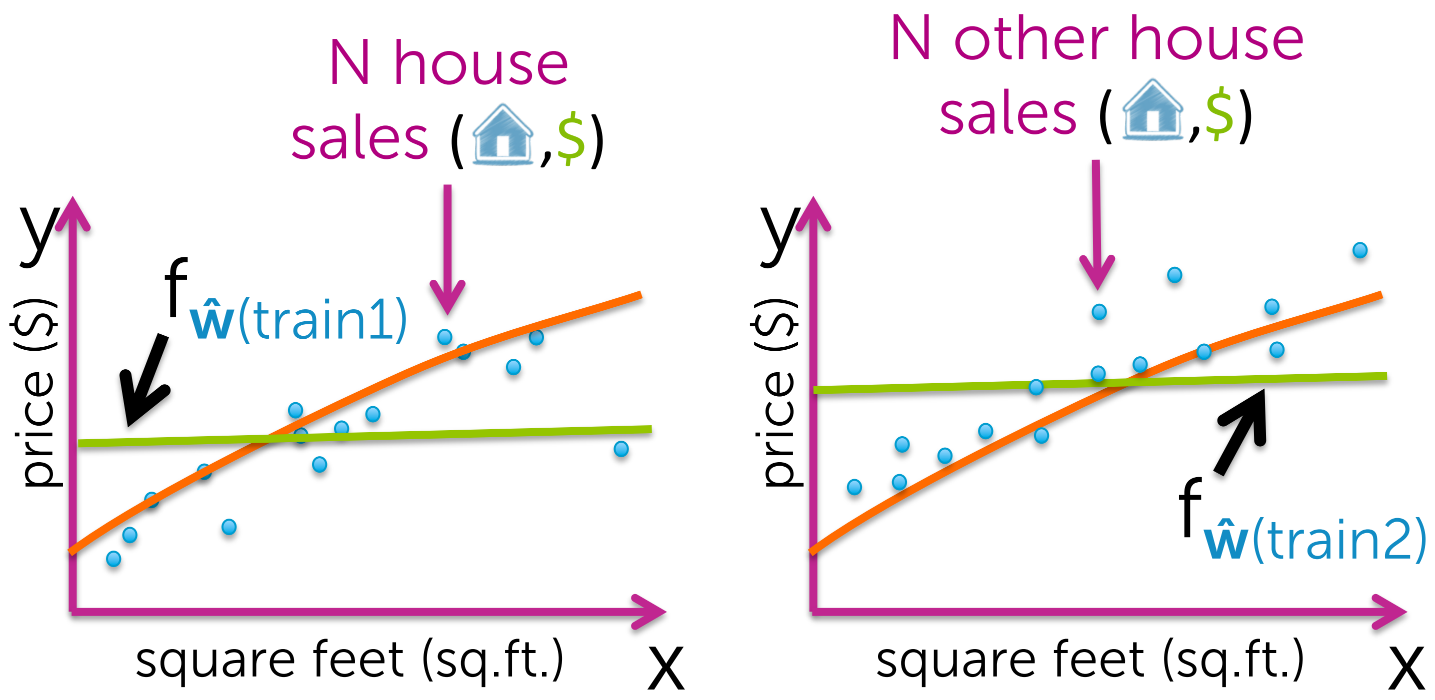
1. Noise
2. Bias
3. Variance

Data inherently noisy



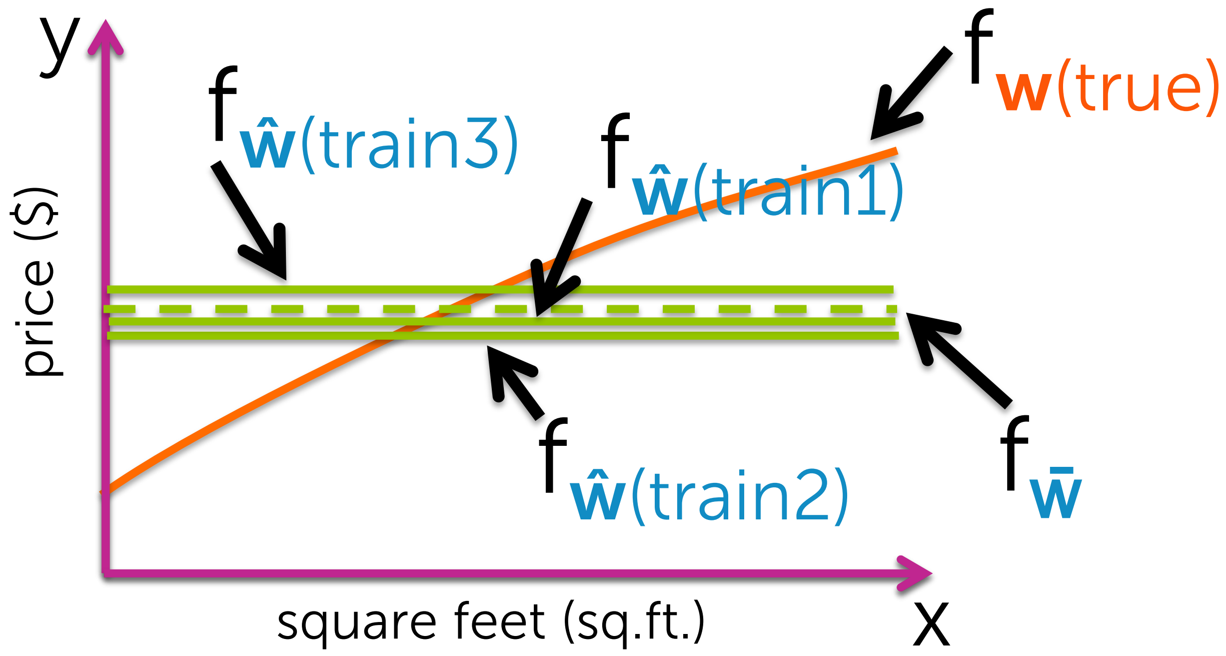
Bias contribution

Assume we fit a constant function



Bias contribution

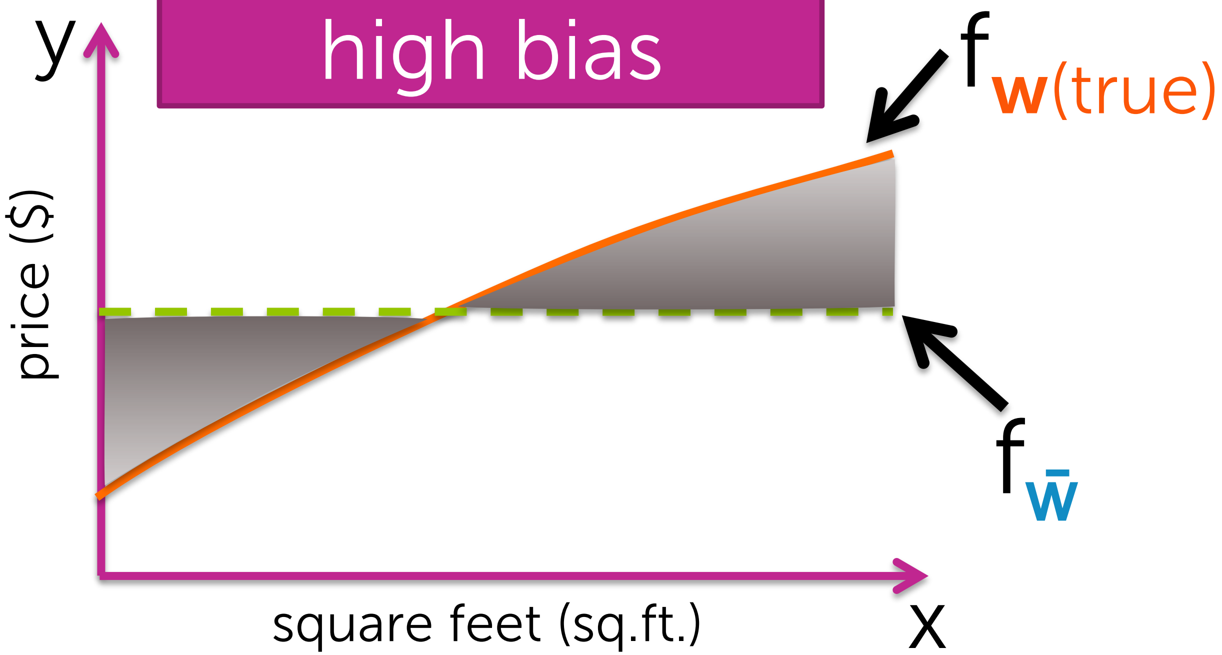
Over all possible size N training sets, what do I expect my fit to be?



Bias contribution

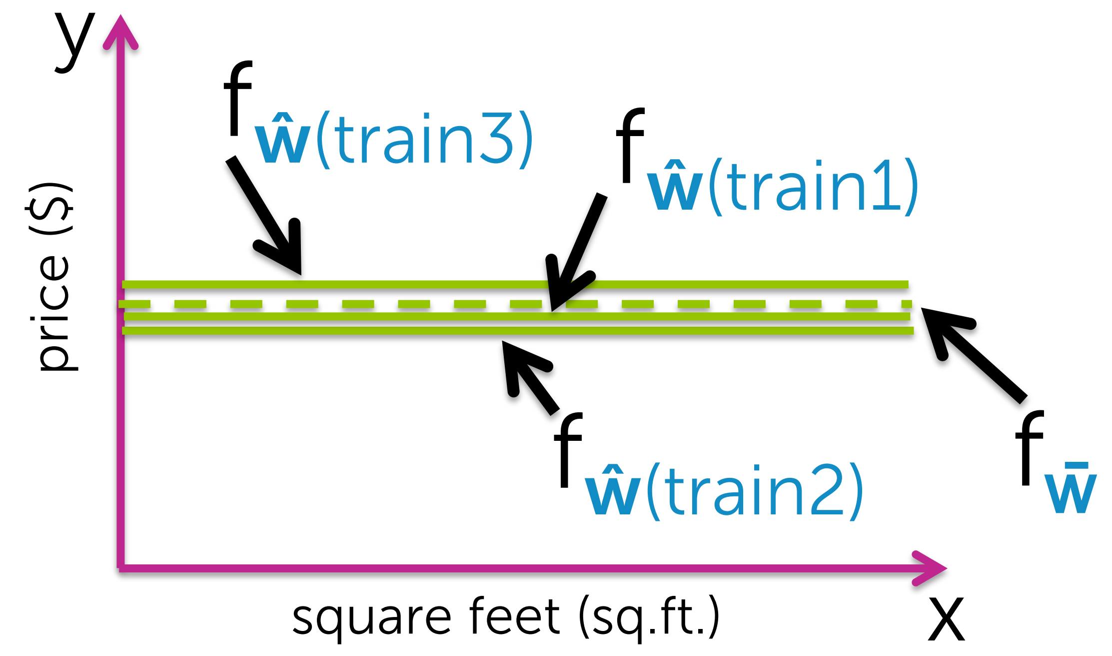
$\text{Bias}(\mathbf{x}) = f_{\mathbf{w}(\text{true})}(\mathbf{x}) - f_{\bar{\mathbf{w}}}(\mathbf{x})$ ← Is our approach flexible enough to capture $f_{\mathbf{w}(\text{true})}$?
If not, error in predictions.

low complexity
→
high bias



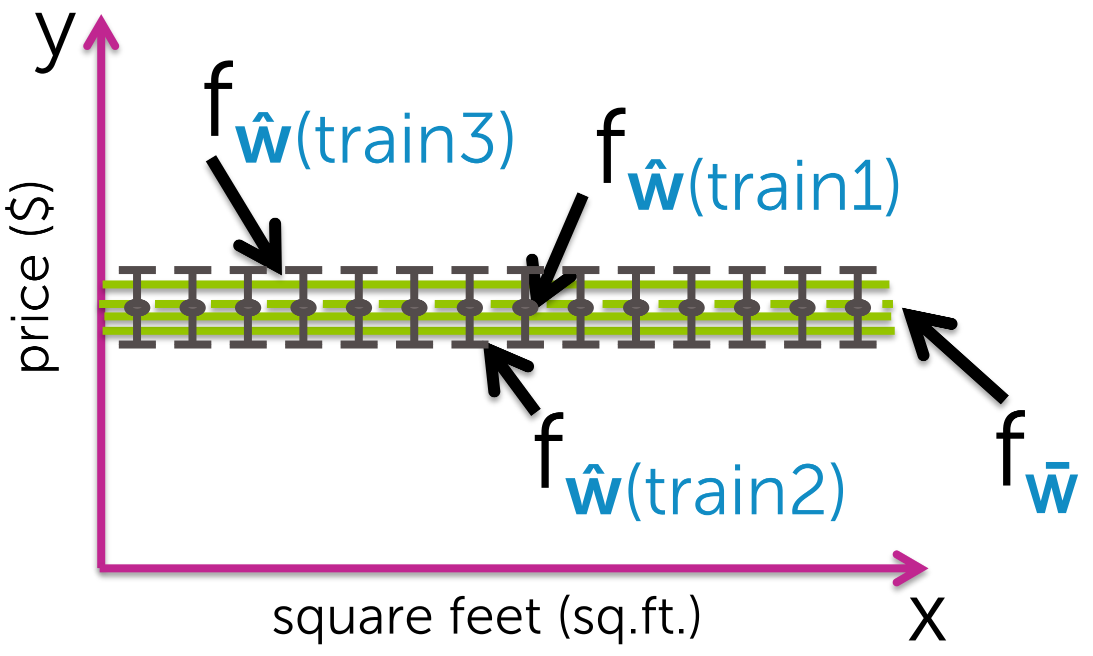
Variance contribution

How much do specific fits vary from the expected fit?



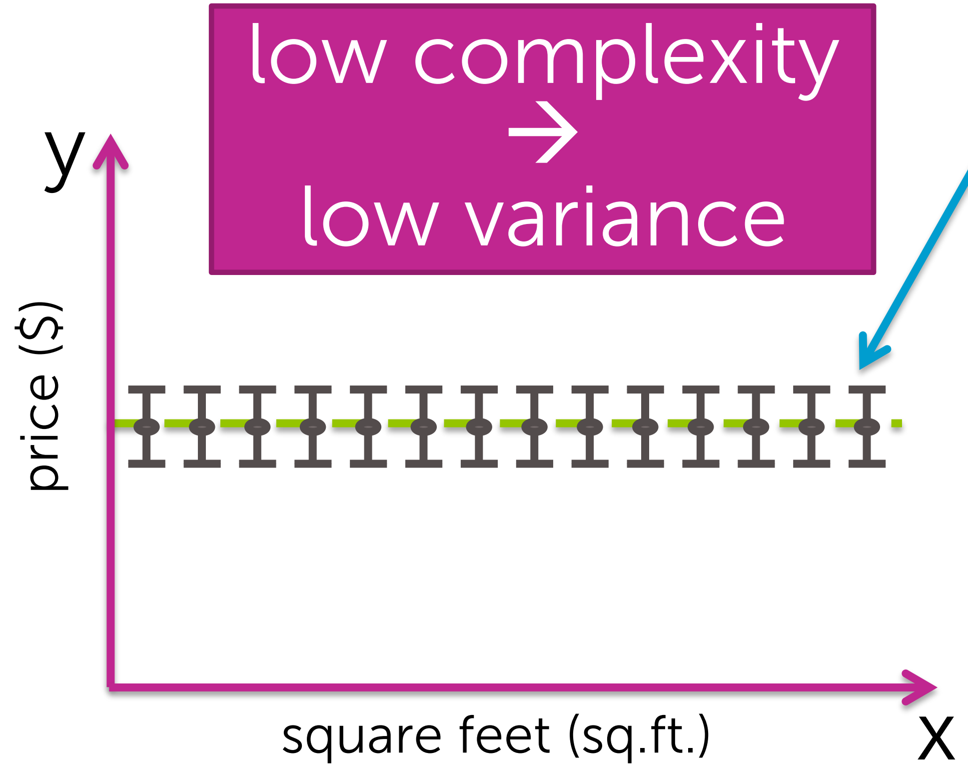
Variance contribution

How much do specific fits vary from the expected fit?



Variance contribution

How much do specific fits vary from the expected fit?

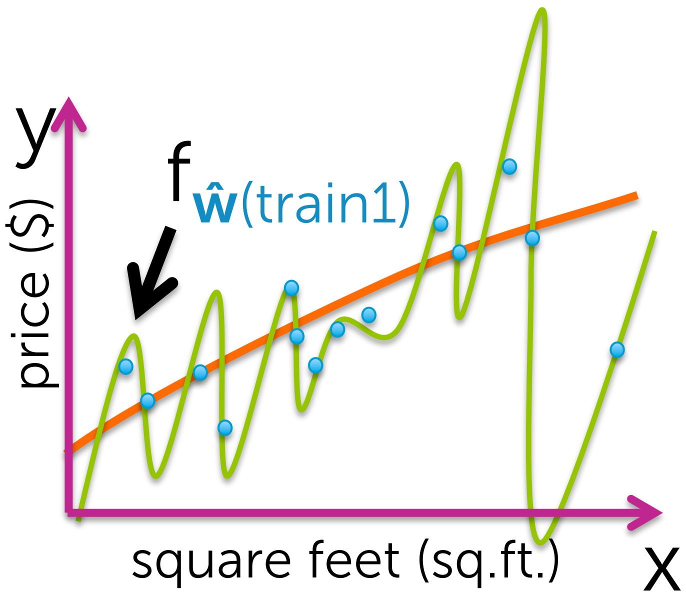


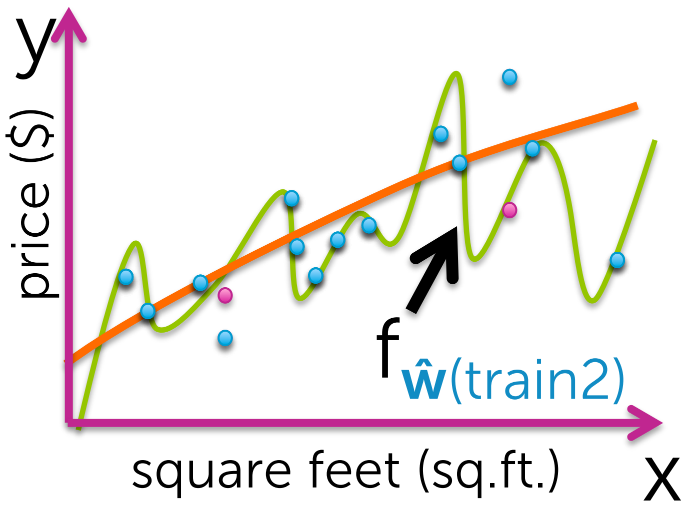
Can specific fits vary widely?

If so, erratic predictions

Variance of high-complexity models

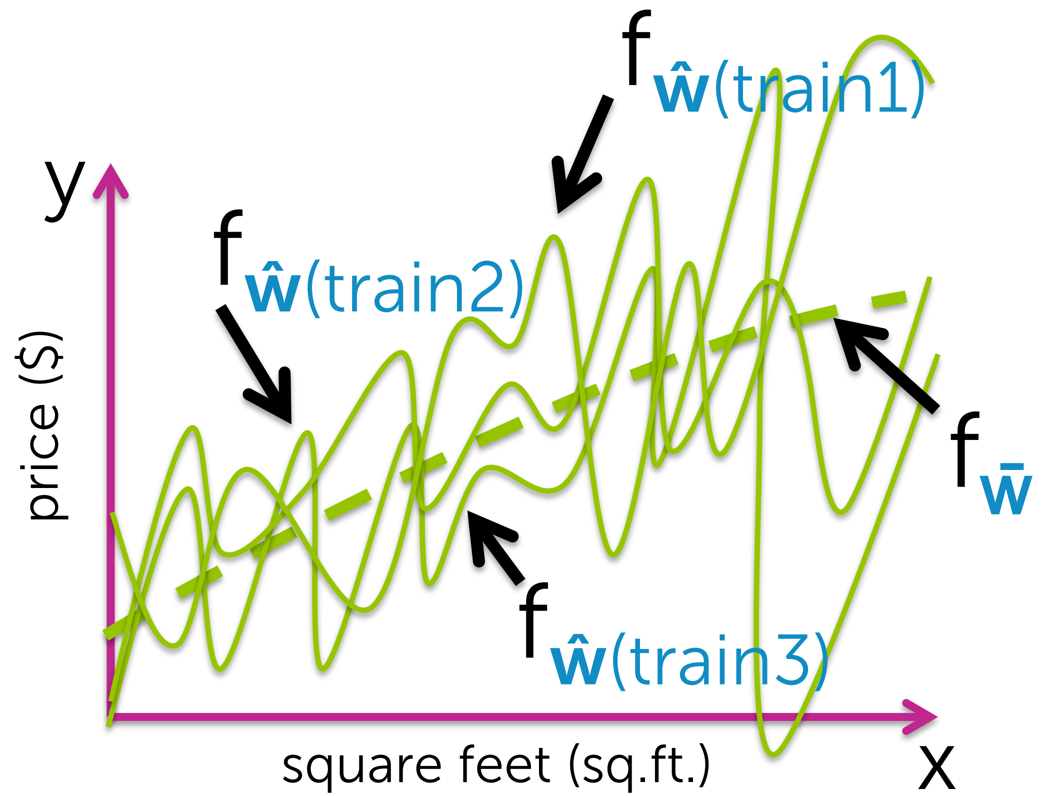
Assume we fit a high-order polynomial



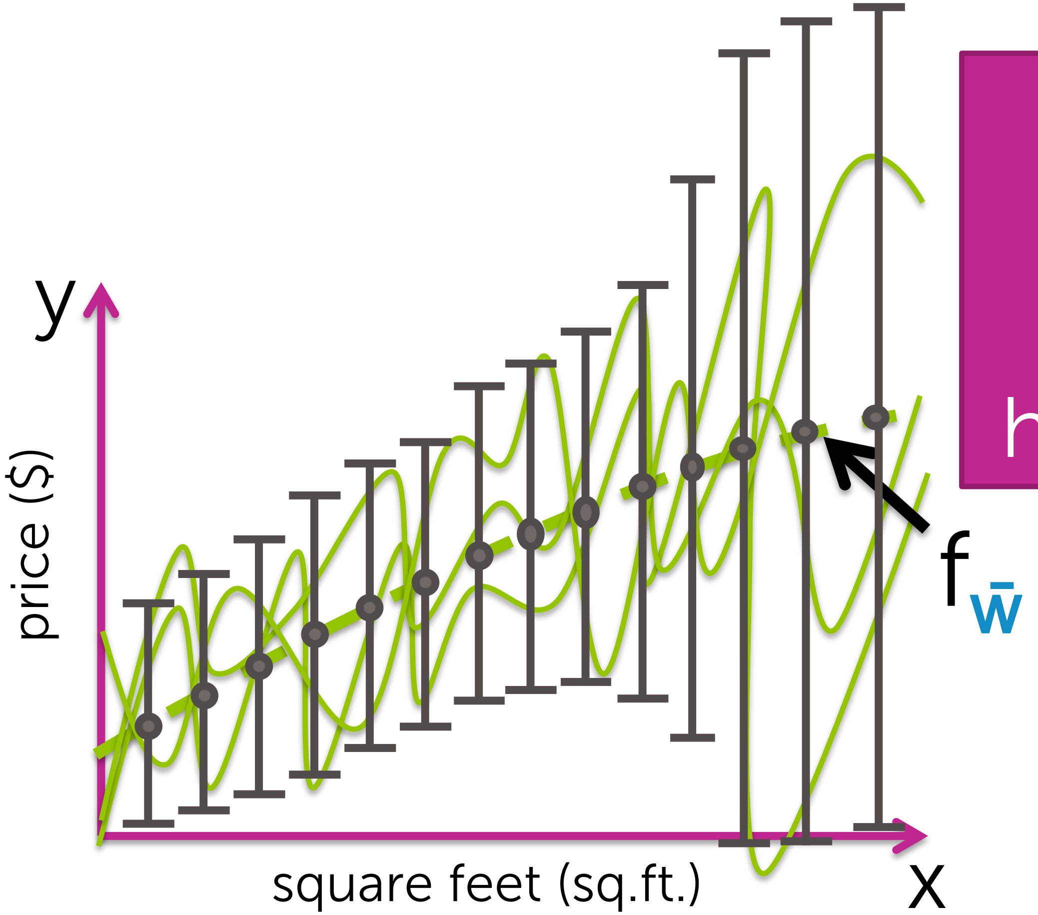


Variance of high-complexity models

Assume we fit a high-order polynomial



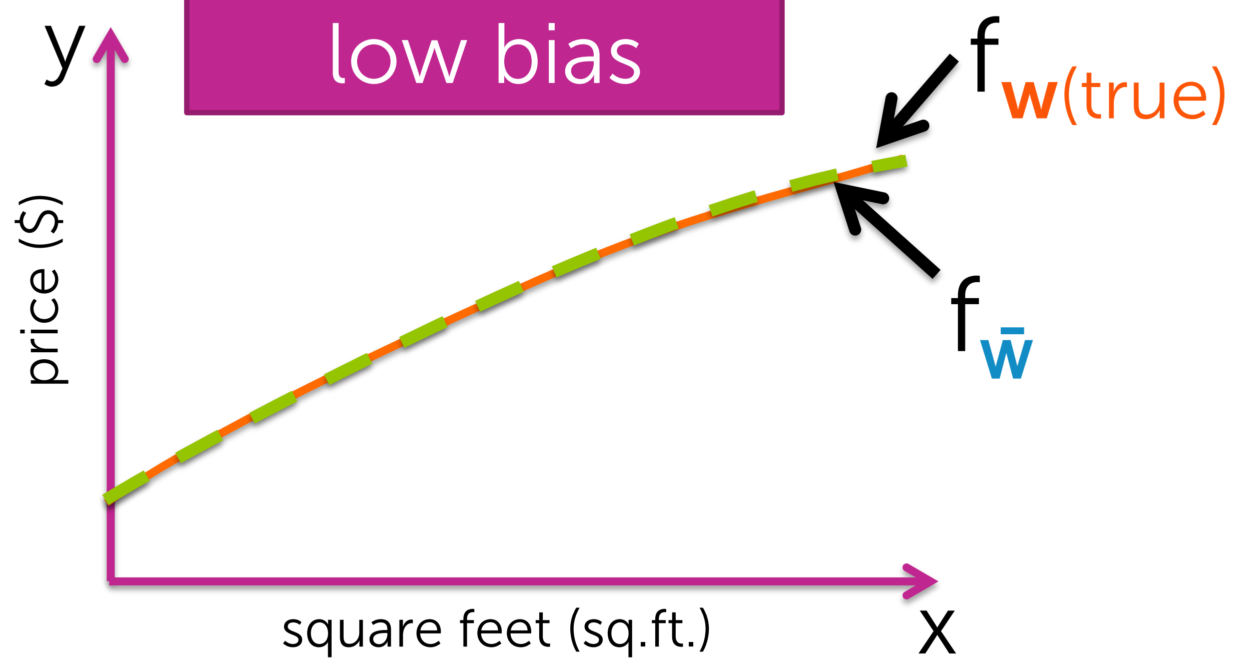
Variance of high-complexity models



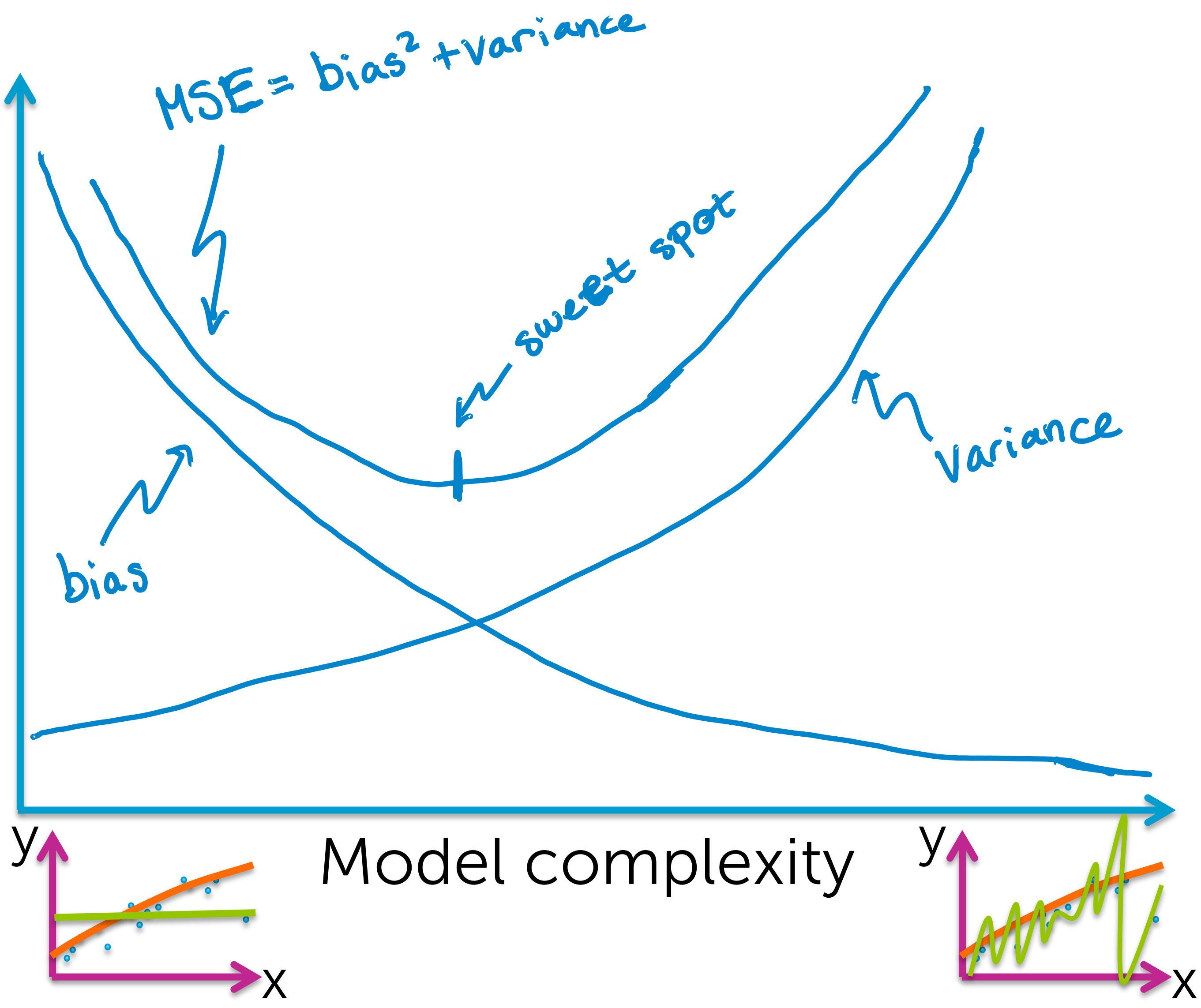
high complexity
→
high variance

Bias of high-complexity models

high complexity → low bias



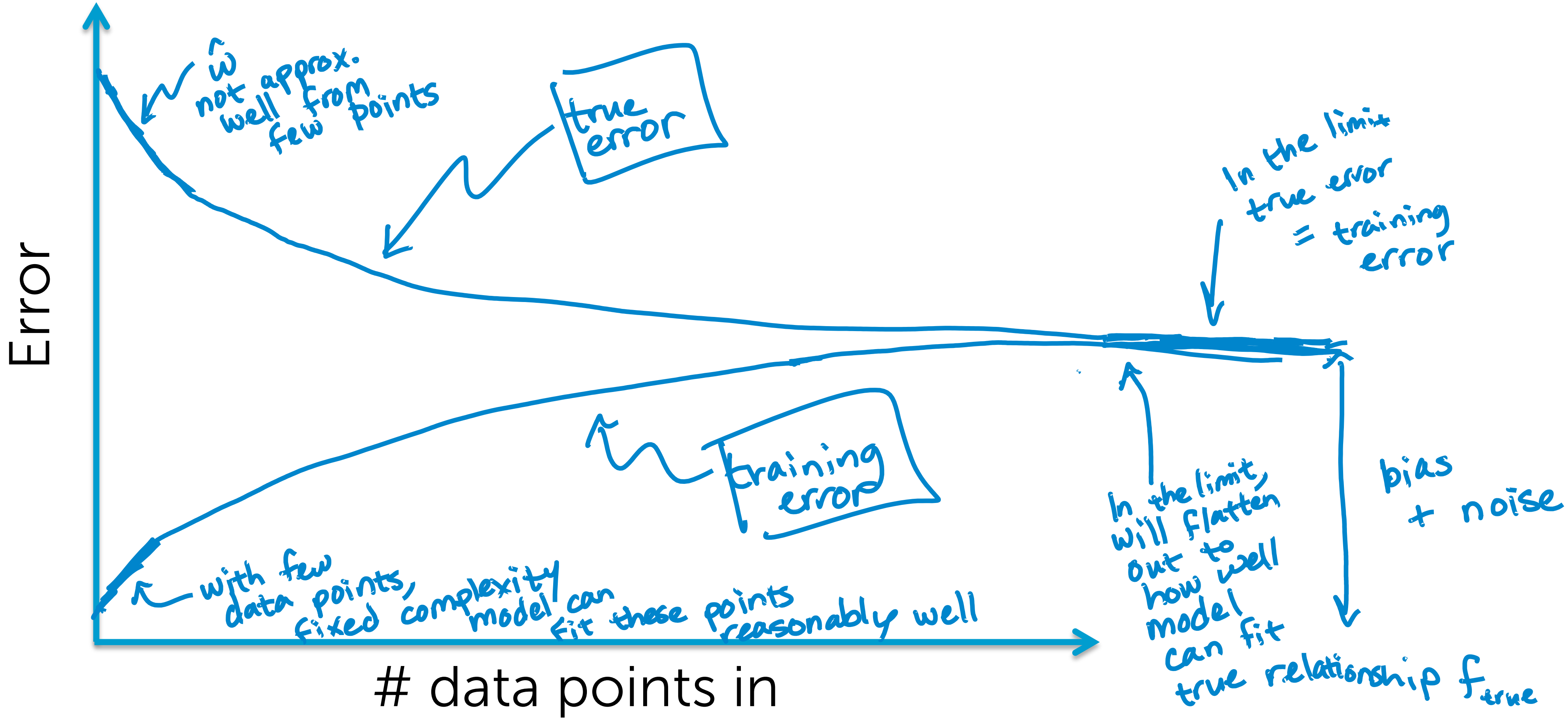
Bias-variance tradeoff



Just like with generalization error, we cannot compute bias and variance

Error vs. amount of data

for a fixed model complexity



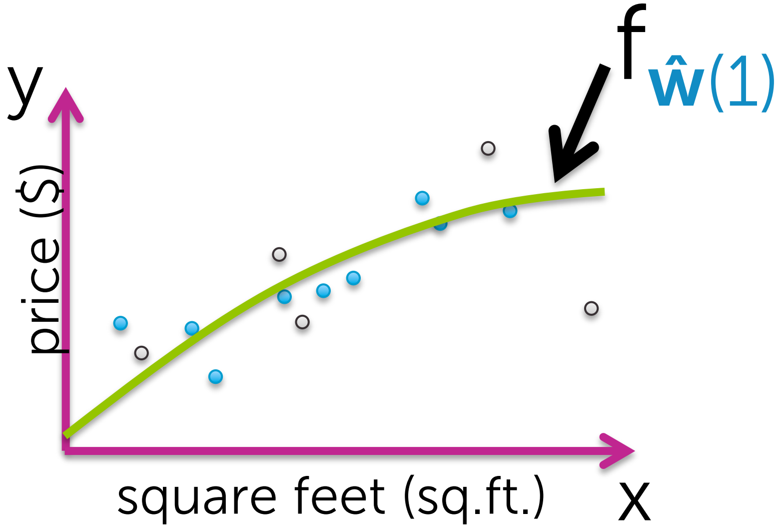
More in depth on the 3 sources of errors…

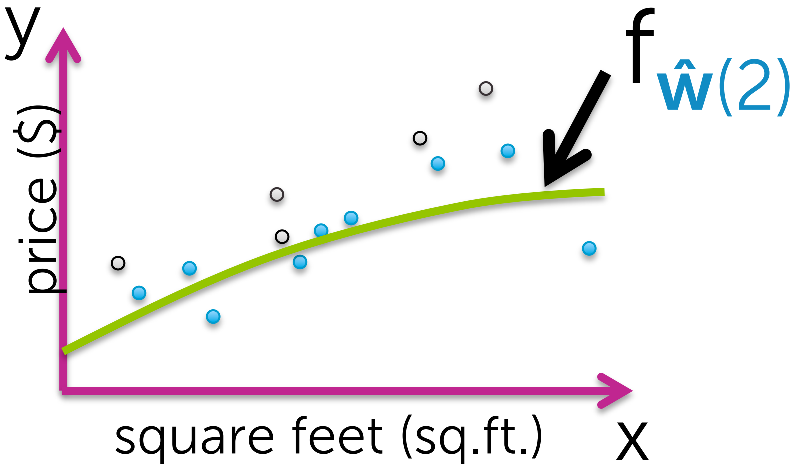
OPTIONAL

Accounting for training set randomness

Training set was just a random sample of N houses sold

What if N other houses had been sold and recorded?



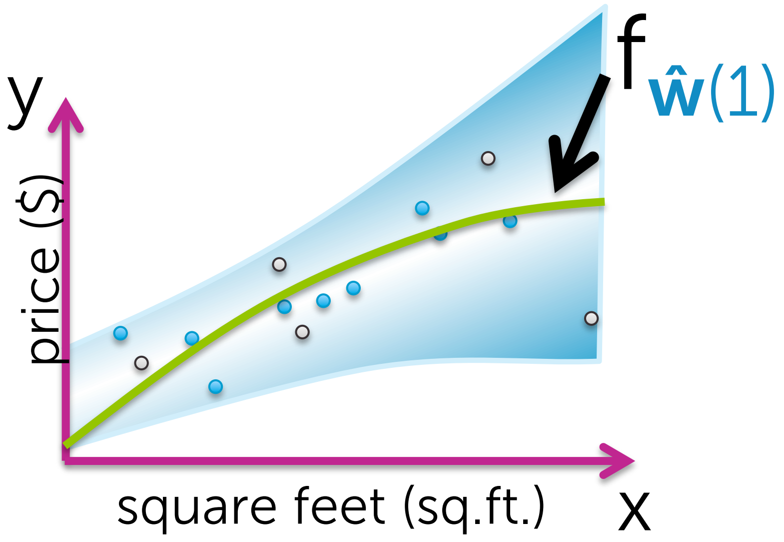


Accounting for training set randomness

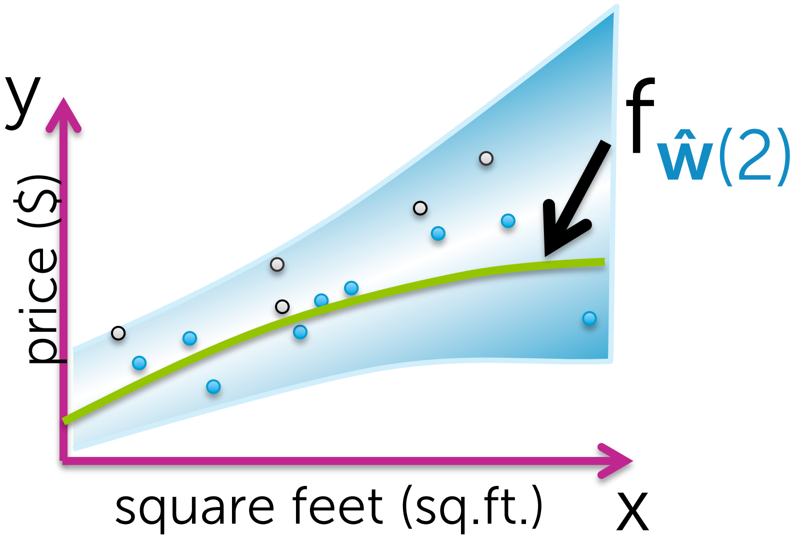
Training set was just a random sample of N houses sold

What if N other houses had been sold and recorded?

generalization error of $\hat{\mathbf{w}}(1)$



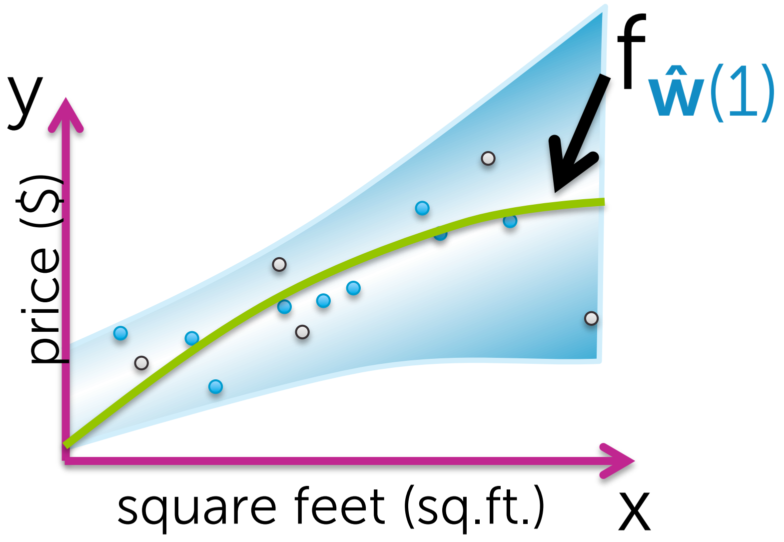
generalization error of $\hat{\mathbf{w}}(2)$



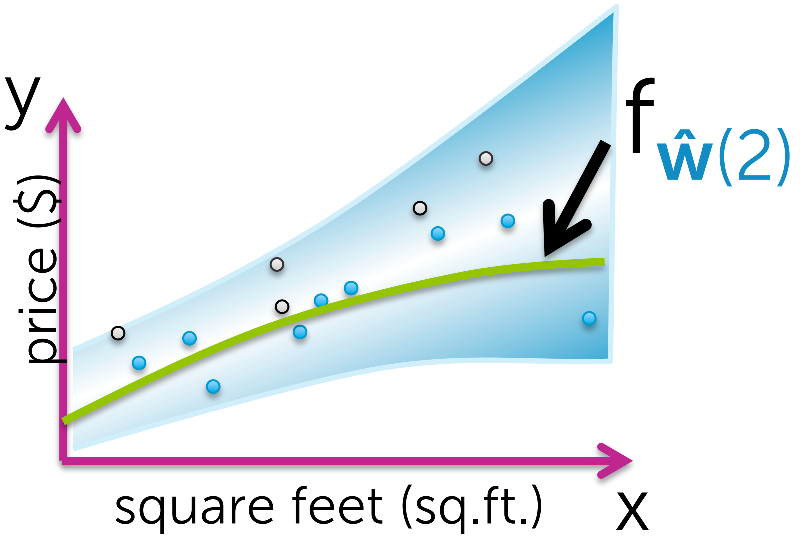
Accounting for training set randomness

Ideally, want performance averaged over all possible training sets of size N

generalization error of $\hat{\mathbf{w}}(1)$



generalization error of $\hat{\mathbf{w}}(2)$

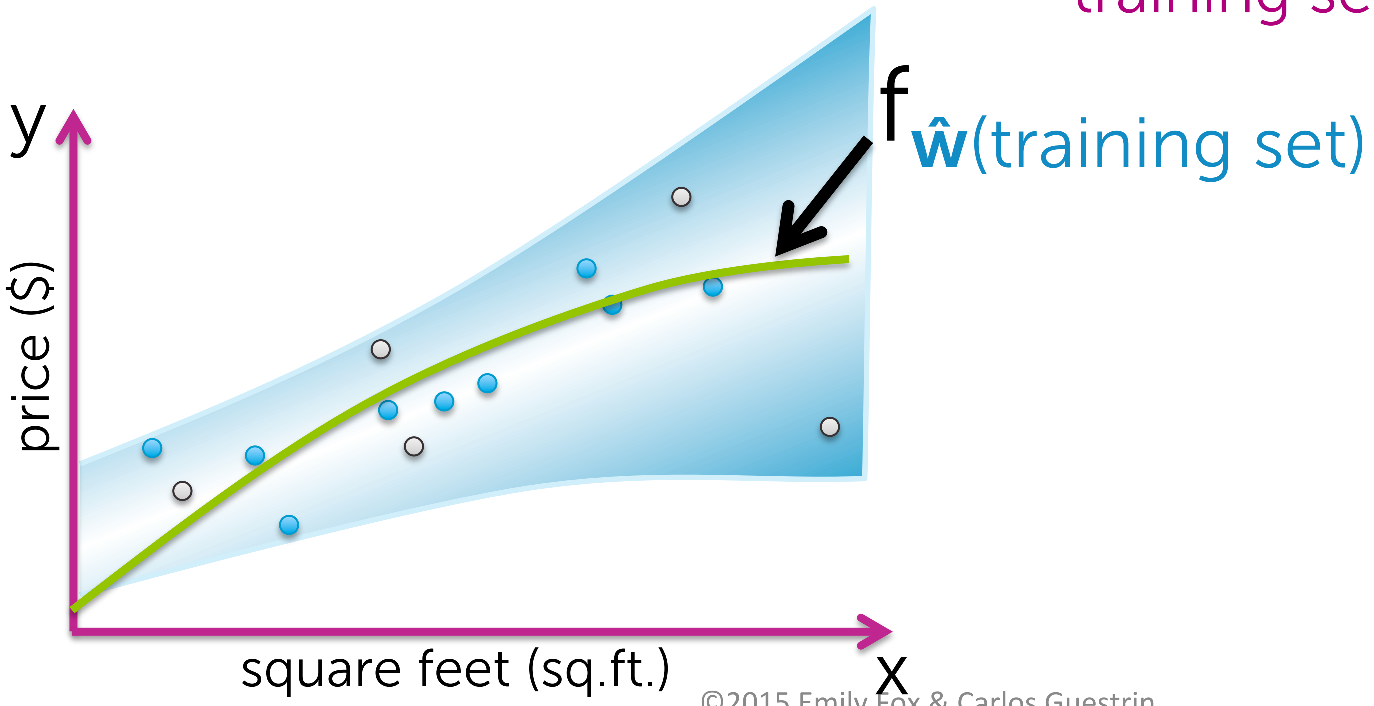


Expected prediction error

$E_{\text{training set}}$[generalization error of $\hat{\mathbf{w}}$(training set)]

averaging over all training sets (weighted by how likely each is)

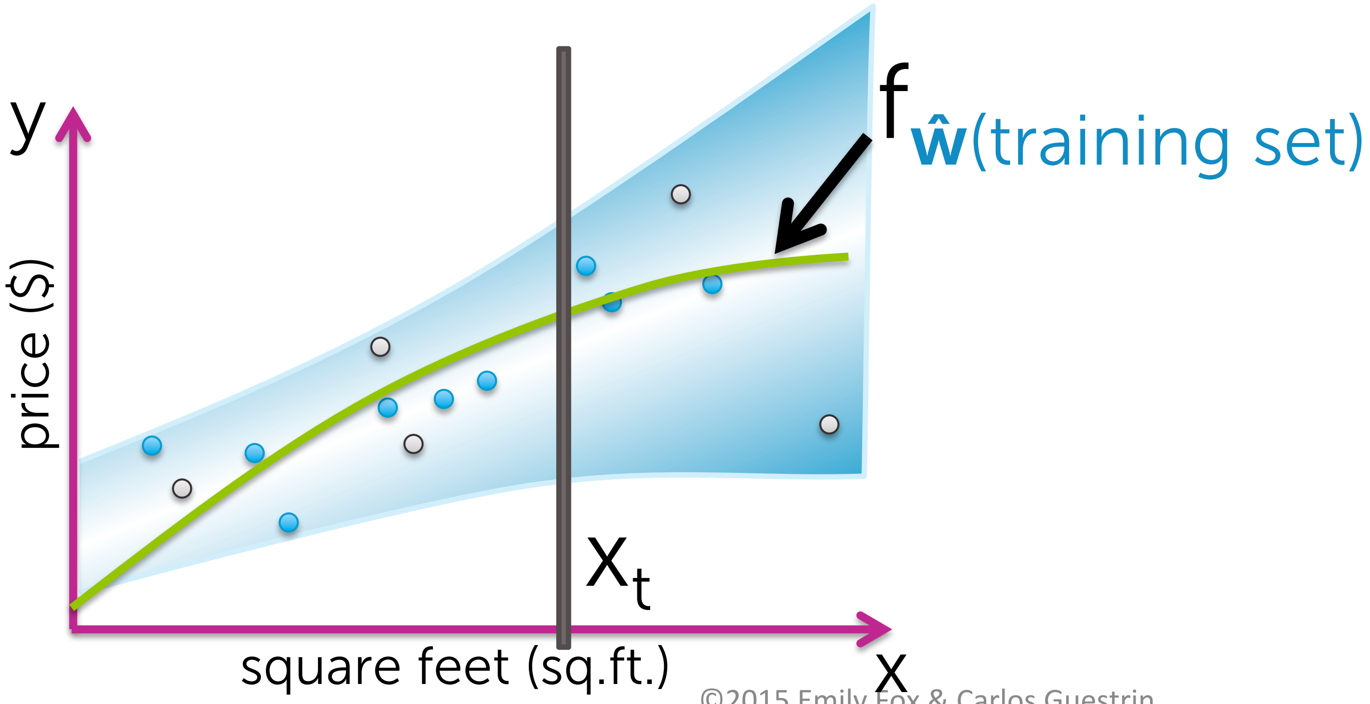
parameters fit on a specific training set



Prediction error at target input

Start by considering:

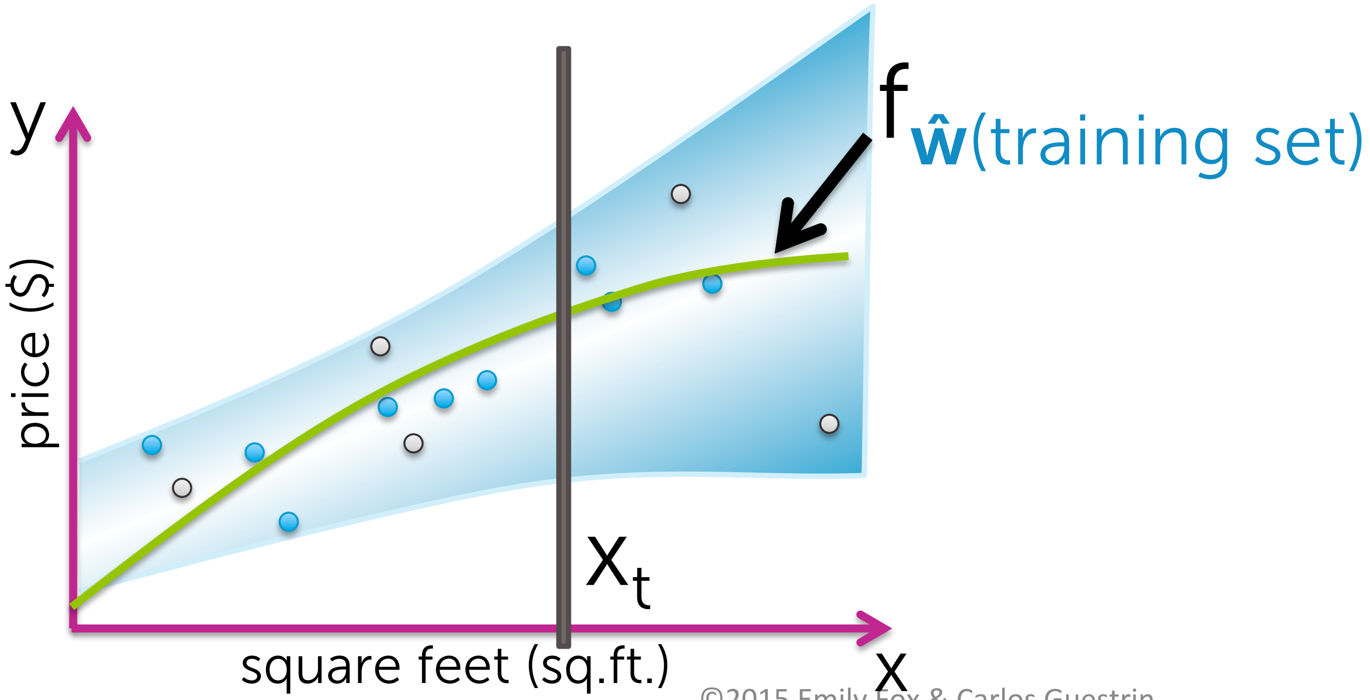
1. Loss at target $\mathbf{x}_t$ (e.g. 2640 sq.ft.)
2. Squared error loss $L(y,f_{\hat{\mathbf{w}}}(\mathbf{x})) = (y-f_{\hat{\mathbf{w}}}(\mathbf{x}))^2$



Sum of 3 sources of error

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

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



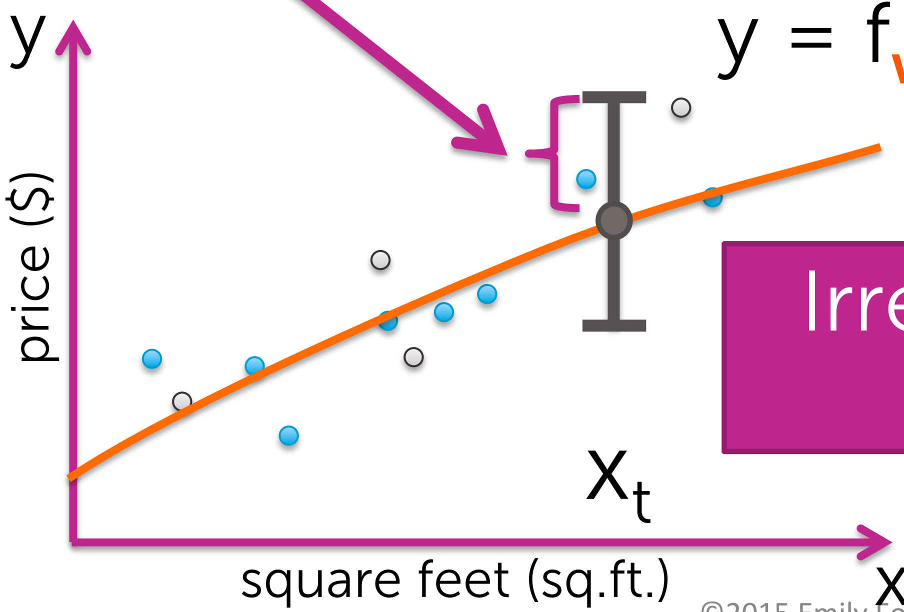
Error variance of the model

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

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

$\sigma^2 =$ "variance"

$y = f_{\mathbf{w}(\text{true})}(\mathbf{x}) + \varepsilon$

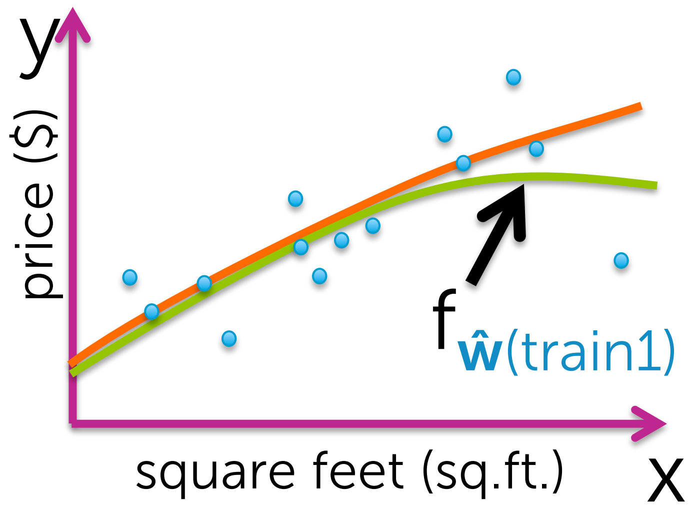


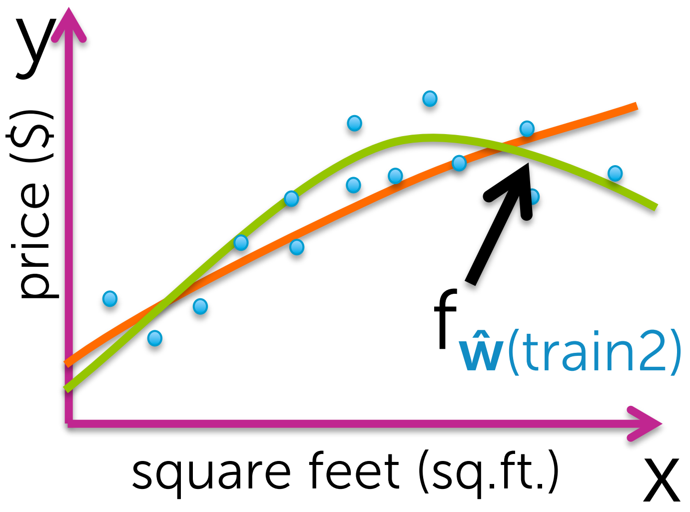
Irreducible error

Bias of function estimator

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

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





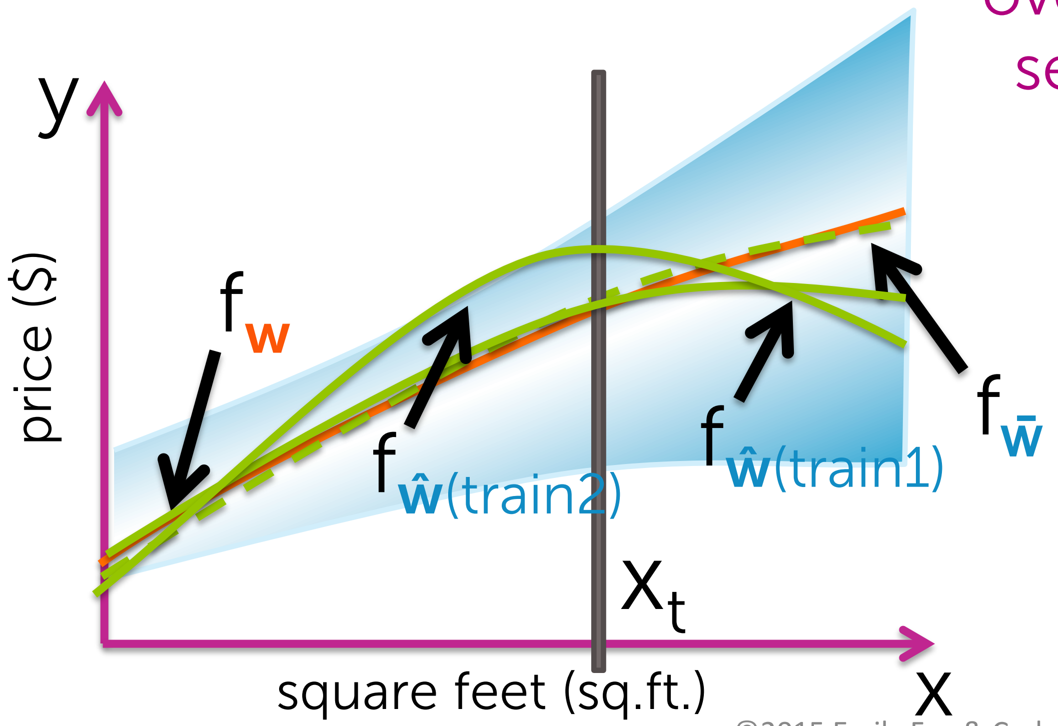
Bias of function estimator

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

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

$$\underbrace{f_{\bar{\mathbf{w}}}(\mathbf{x})}_{E_{\text{train}}[f_{\hat{\mathbf{w}}(\text{train})}(\mathbf{x})]}$$

over all training sets of size N

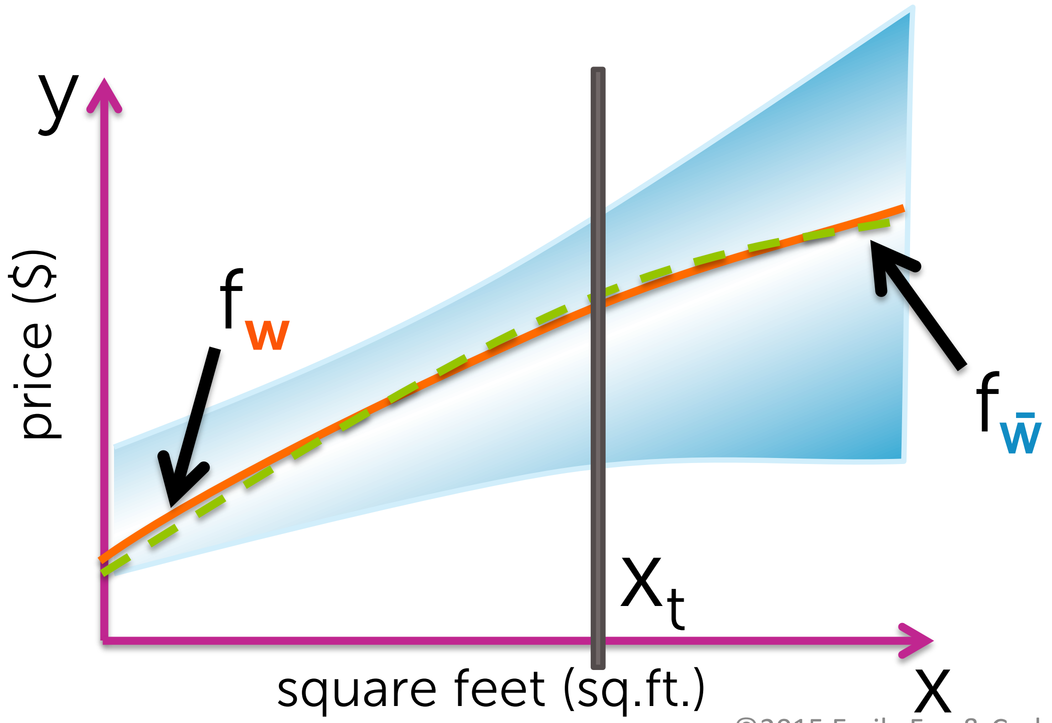


Bias of function estimator

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

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

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



Bias of function estimator

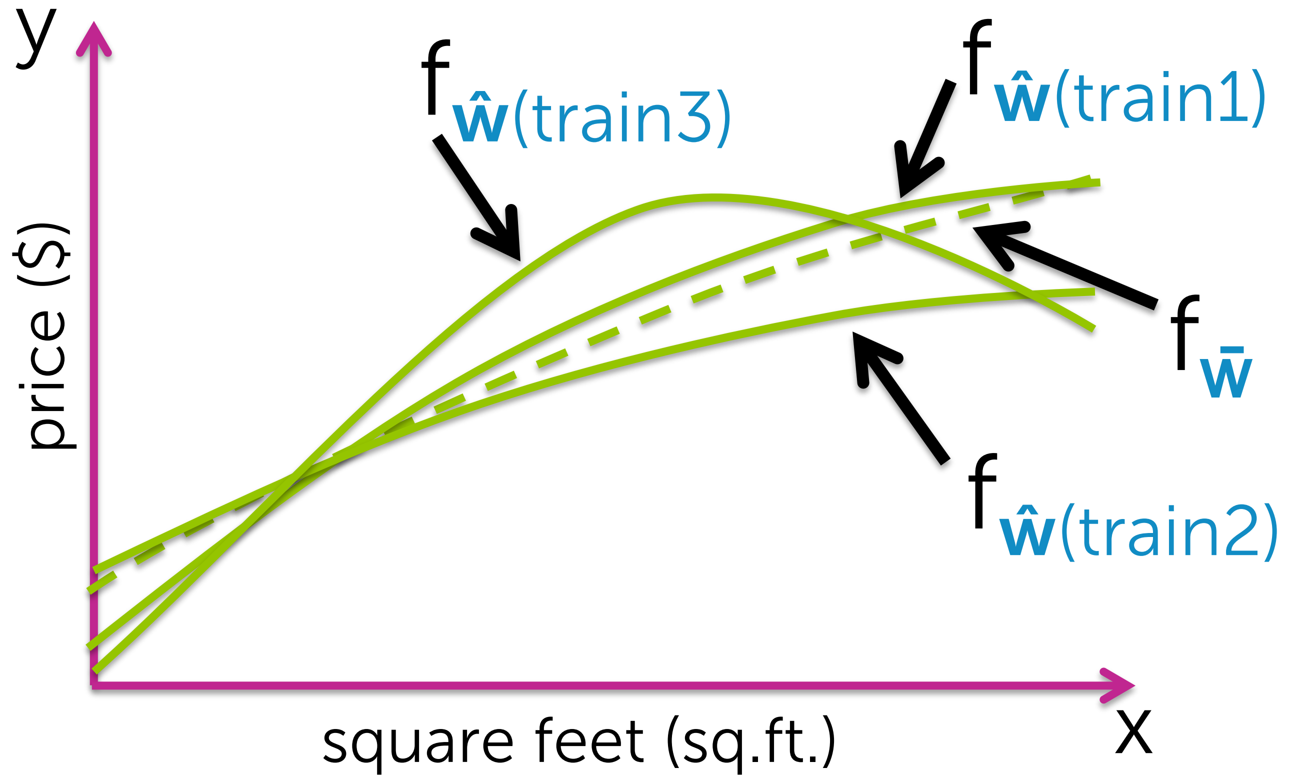
Average prediction error at $\mathbf{x}_t$

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

Variance of function estimator

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

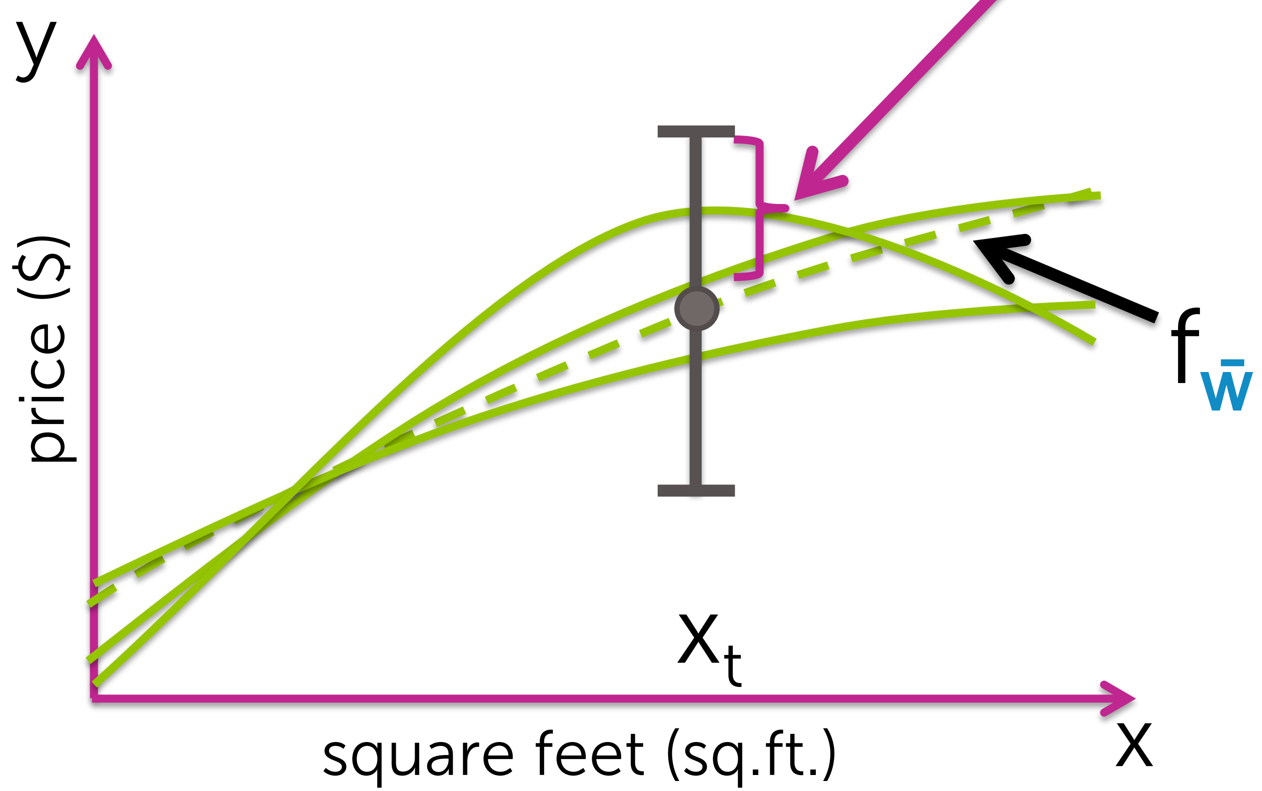
$$= \sigma^2 + [\text{bias}(f_{\hat{\mathbf{w}}}(\mathbf{x}_t))]^2 + \text{var}(f_{\hat{\mathbf{w}}}(\mathbf{x}_t))$$



Variance of function estimator

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

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



Variance of function estimator

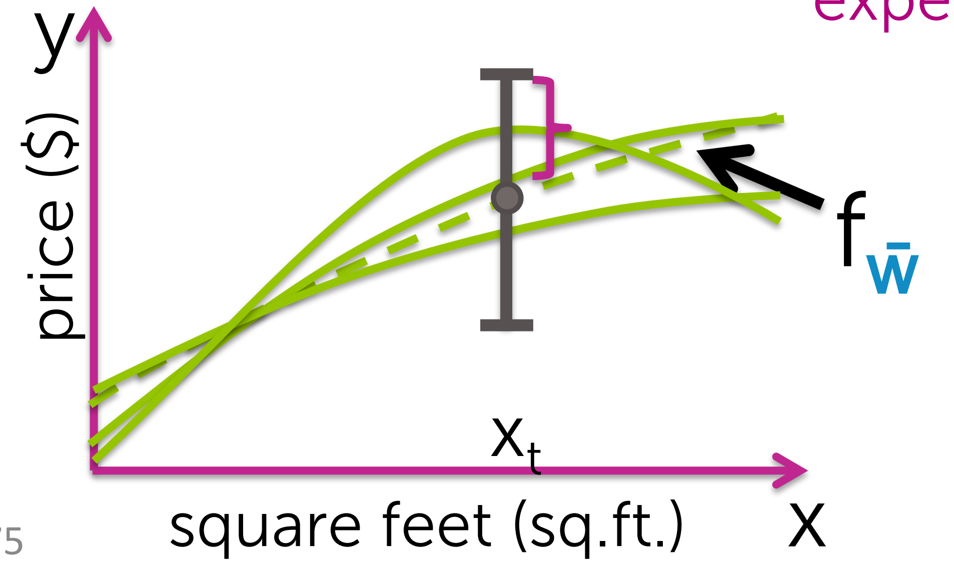
fit on a specific training dataset

what I expect to learn over all training sets

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

over all training sets of size N

deviation of specific fit from expected fit at $\mathbf{x}_t$



Why 3 sources of error? A formal derivation

OPTIONAL

Deriving expected prediction error

Expected prediction error

$= E_{\text{train}}$ [generalization error of $\hat{\mathbf{w}}$(train)]

$= E_{\text{train}}\left[E_{\mathbf{x},y}\left[L(y, f_{\hat{\mathbf{w}}(\text{train})}(\mathbf{x}))\right]\right]$

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

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

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

Deriving expected prediction error

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

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

$$= E_{\text{train},y_t}\left[(\overbrace{(y_t - f_{\mathbf{w}(\text{true})}(\mathbf{x}_t))}^{a} + \overbrace{(f_{\mathbf{w}(\text{true})}(\mathbf{x}_t) - f_{\hat{\mathbf{w}}(\text{train})}(\mathbf{x}_t))}^{b})^2\right]$$

$$= \underbrace{E_{\cancel{\text{train}},y}[\underbrace{(y-f)^2}_{\epsilon^2}]}_{\text{by definition } \sigma^2} + 2\underbrace{E_{\text{train},y}[\underbrace{(y-f)}_{\epsilon}(f-\hat{f})]}_{\underbrace{\cancel{E[\epsilon]}}_{0}\,E[f-\hat{f}]} + \underbrace{E_{\text{train},\cancel{y}}[(f-\hat{f})^2]}_{\triangleq \text{MSE}(\hat{f})}$$

mean square error

$$= \sigma^2 + \underline{\text{MSE}(\hat{f})}$$

ind. r.v.'s
$E[ab] = E[a]E[b]$

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

Shorthand:
$y_t \to y$
$f_{w(\text{true})} \to f$
$f_{\hat{w}(\text{train})} \to \hat{f}$

Equating MSE with bias and variance

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

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

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

$\bar{f} \leftarrow$ shorthand new notation on this slide

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

$\bar{f} = E_{\mathrm{train}}[\hat{f}]$

$(f - \bar{f})$: not a fcn of training data

by definition $(f-\bar{f})^2 = \mathrm{bias}^2(\hat{f})$

by definition $\bar{f} - E_{\mathrm{train}}[\hat{f}] = 0$

$E[(\hat{f}-\bar{f})^2] = \mathrm{var}(\hat{f})$

$\hat{f}$: random function at x_t = random variable

$\bar{f}$: its mean

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

Putting it all together

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

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

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

3 sources of error

Summary of tasks

The regression/ML workflow

1. **Model selection**
   Often, need to choose tuning parameters λ controlling model complexity (e.g. degree of polynomial)

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

Hypothetical implementation

1. **Model selection**

For each considered model complexity 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

2. **Model assessment**

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

Hypothetical implementation

Training set	Test set

1. **Model selection**

For each considered model complexity 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 complexity 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

Training set	Validation set	Test set
fit $\hat{\mathbf{w}}_\lambda$	test performance of $\hat{\mathbf{w}}_\lambda$ to select λ^*	assess generalization error of $\hat{\mathbf{w}}_{\lambda^*}$

Typical splits

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

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
- Describe tradeoffs in forming training/test splits
- List and interpret the 3 sources of avg. prediction error
  - Irreducible error, bias, and variance
- Discuss issue of selecting model complexity on test data and then using test error to assess generalization error
- Motivate use of a validation set for selecting tuning parameters (e.g., model complexity)
- Describe overall regression workflow