Assessing Performance

- The process thus far:

```
    - Model + algorithm -> fitted function.
    - Predictions -> decisions -> outcomes;
```

- But is the above fitted function on the dataset the best result? How much can one be losing compared to perfection?

```
    - Perfect predictions : Loss = 0;
    - The preictions from the model : ??
```

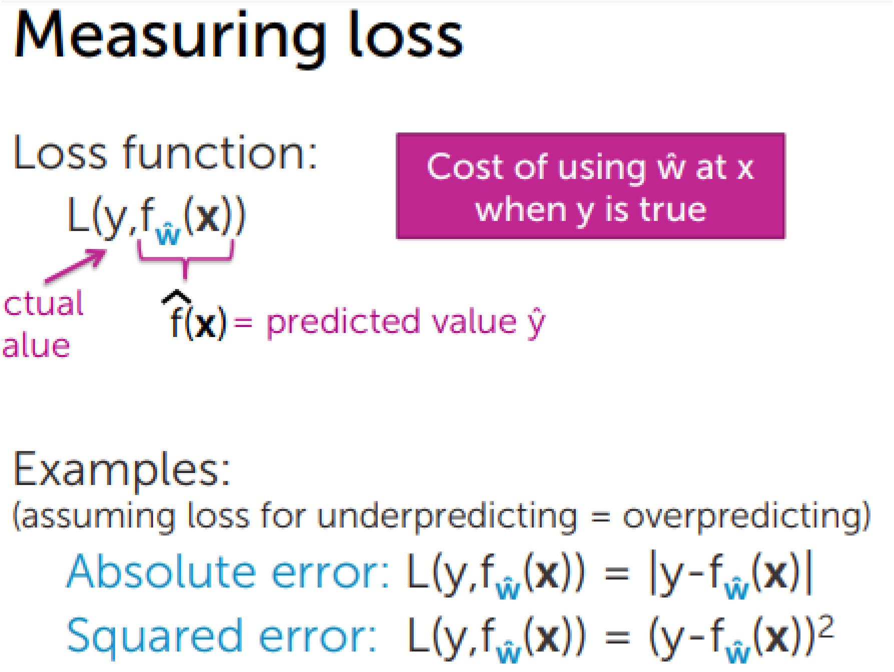
Measuring the Loss:

- In machine learning the loss incured by a fitted function is termed Loss function.
- Loss function : Cost of using w(hat) at x when y is true.
- Loss function : L(y, f(w(hat))(X));

```
    - where y -> actual value;
    - f(hat)(x) = f(w(hat))(X) -> predicted value of y(hat)
```

- Examples : Symmetric loss function (Assuming loss of underpredicting = overpedicting)

```
    - Absolute Error : L(y, f(w - hat)(X)) = |y - f(w-hat)(X)|
    - Squared Error : L(y, f(w - hat)(X)) = (y - f(w-hat)(X))^2
```



Measuring loss

Loss function:

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

Cost of using ŵ at x when y is true

ctual alue

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

Examples:

(assuming loss for underpredicting = overpredicting)

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

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

Assessing the Loss:

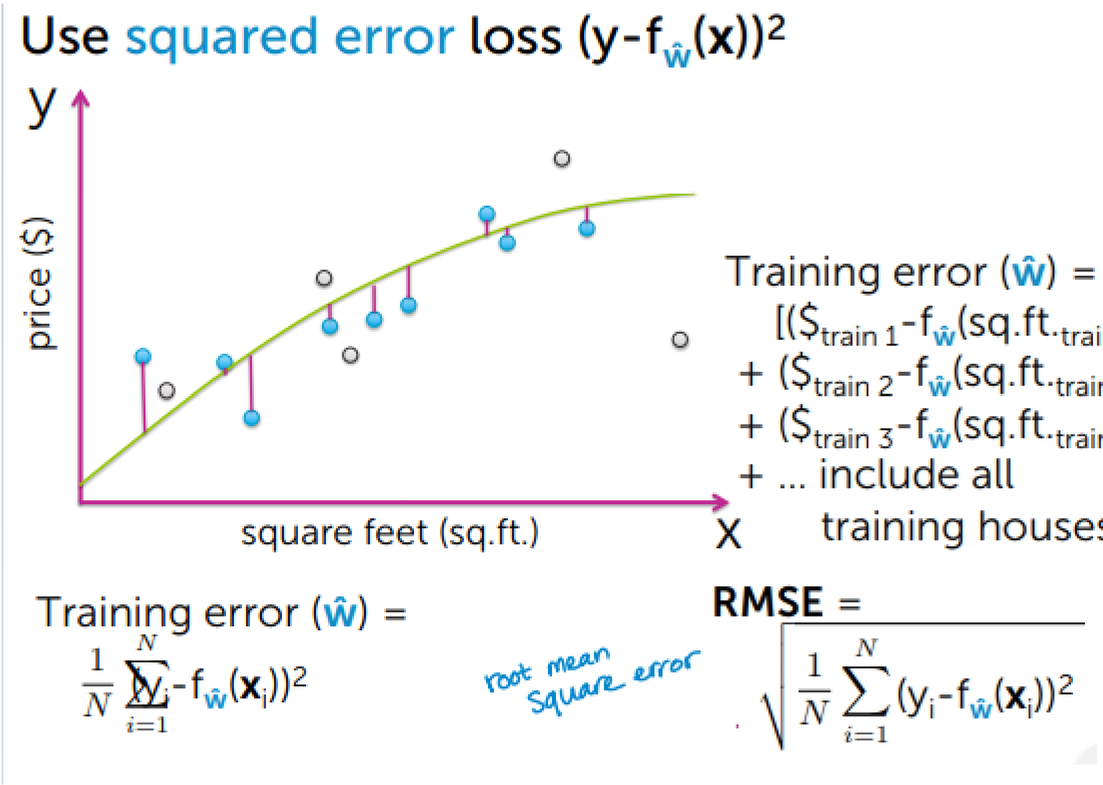
Part 1: Training Error:

- Training Dataset -> It is a subset of all the dataset.

- Fit a model through the dataset -> linear function, quadratic, nth polynomial, etc; Then estimate the **model parameters (w-hat)**, and minimize the **RSS of the training data using (w-hat)**.
- Compute training error:

```
- 1. Define a loss function L(y, f(w-hat)(X))
- 2. Training Error
  - Avg. loss on houses in training set.
  - (1/N) Σ(i = 1...N) * L(y(i),f(w-hat)(x(i))) - Training set
```

Using Squared Error Loss:



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

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

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

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

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

+ ... include all training houses]

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

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

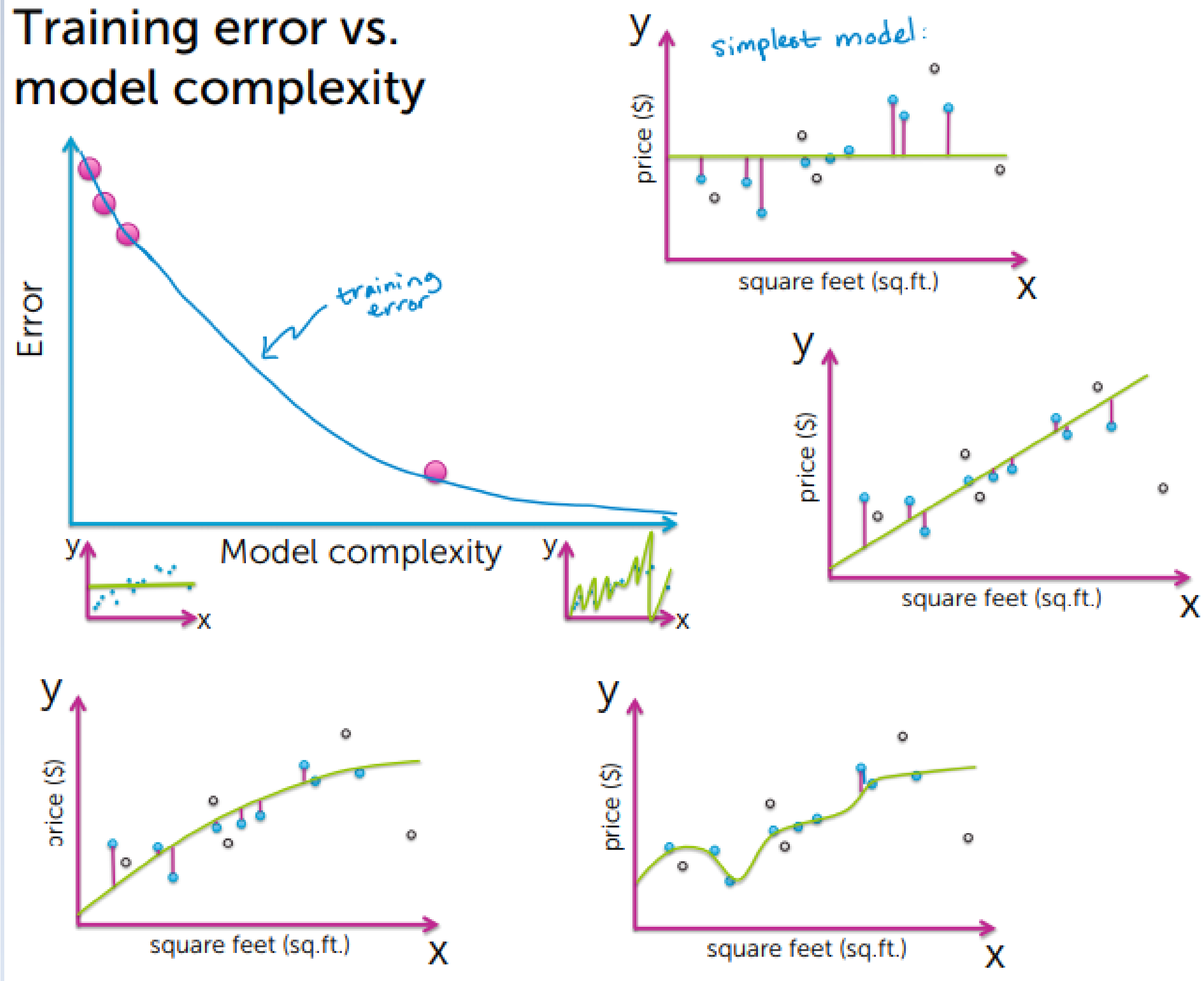
root mean square error

RMSE =

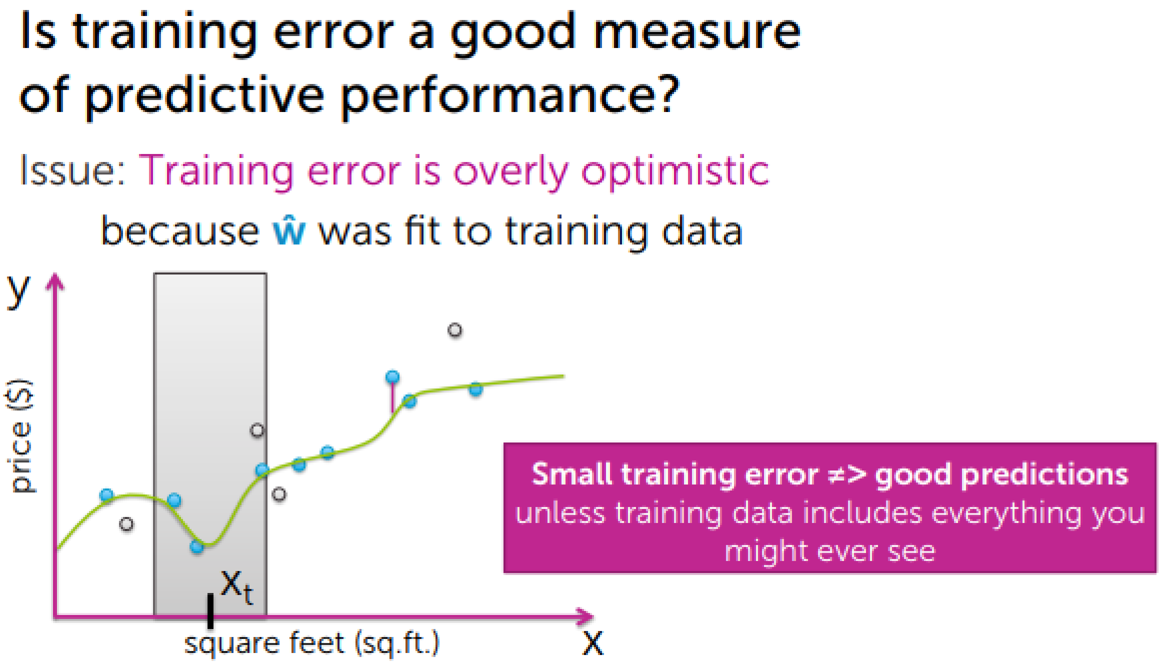
$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-f_{\hat{w}}(\mathbf{x}_i))^2}$$

Training Error vs. Model Complexity:

- Error ∝ 1/Model complexity;
- Model's with low complexity -> high training error;
- Model's with high complexity -> low training error;



Training error vs. model complexity



Is training error a good measure of predictive performance?

Issue: Training error is overly optimistic because ŵ was fit to training data

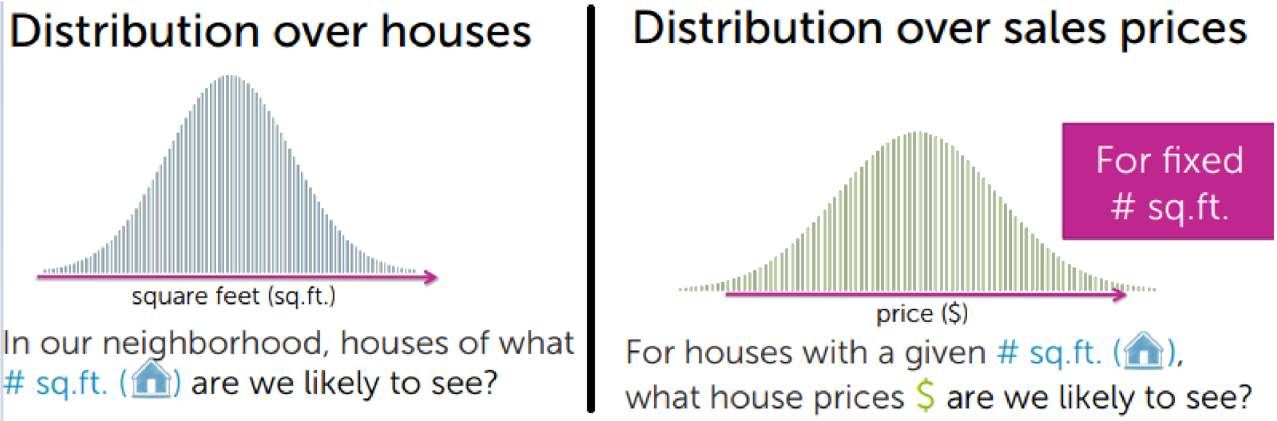
Small training error ≠> good predictions unless training data includes everything you might ever see

Part 2: Generalization (true) Error:

- Really want to estimate the loss over all posible dataset pairs, but not enough in the dataset.
- For input -> sqft; target -> price; Inorder to effectively predict and compute minimum loss:

```
1. Create a distribution/bell curve for the range of sqft in the neighbouro
od.
2. Create a distribution over the sales prices.
    - For houses with a given #sqft of house what if the price.
```

- Generalization Error - (avg of all possible(x,y) pairs weighted on how likely each is) X (training error)
- Generalization error = E(x,y) * [L(y, f(w-hat)(X))];



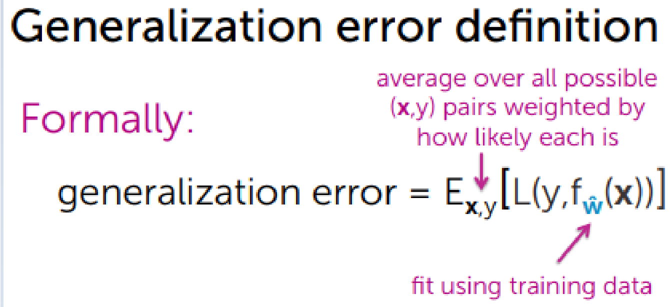
Distribution over houses

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

Distribution over sales prices

For fixed # sq.ft.

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



Generalization error definition

Formally:

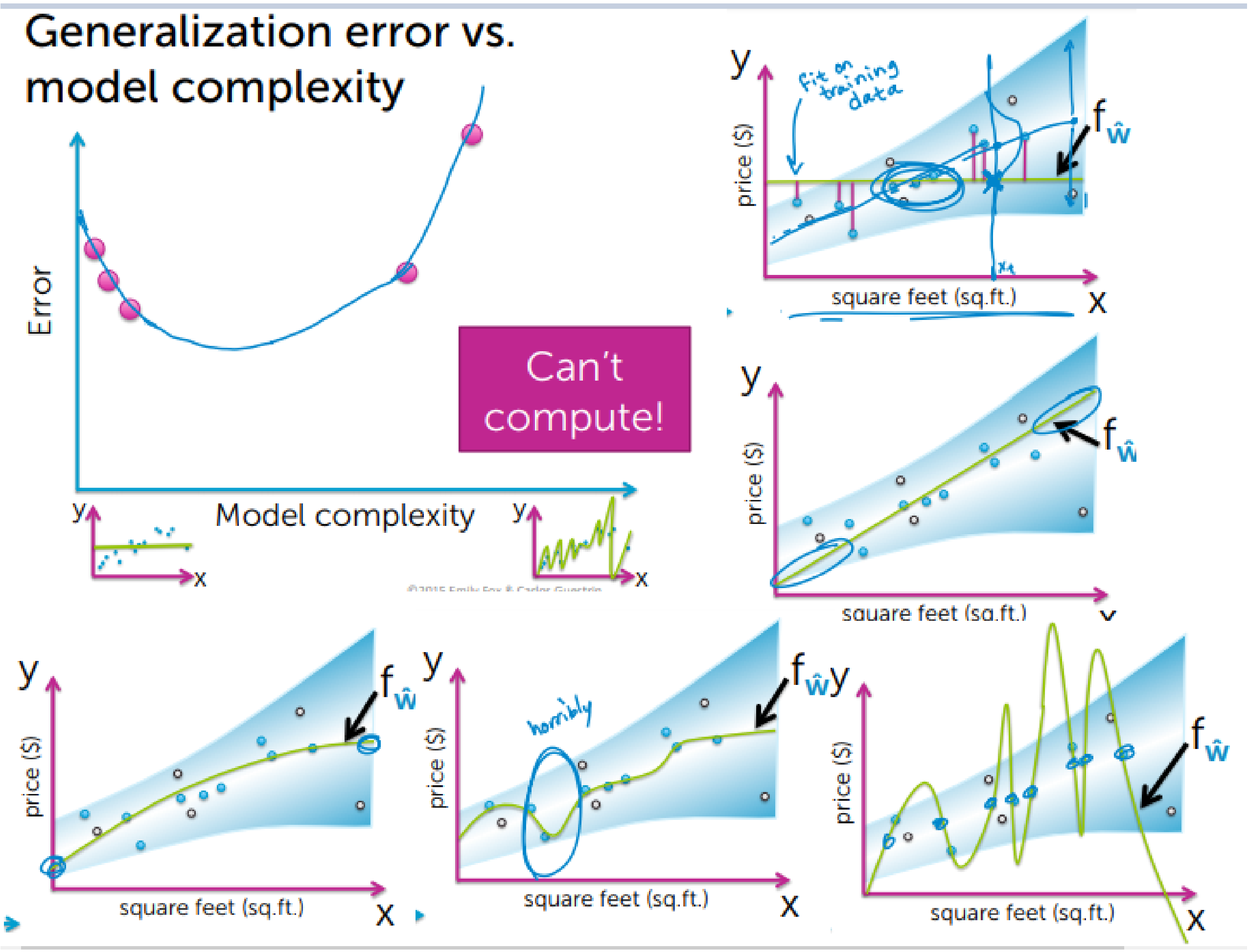
average over all possible (**x**,y) pairs weighted by how likely each is

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

fit using training data

Generalization Error vs. Model Complexity:

- The white region is where the expected predictions occurs, and as it moves away from the white to the blue region the error increases as the predictions are in general not expected.
- In general - generalization error doesn't reduce with increase in model complexity.



Generalization error vs. model complexity

Can't compute!

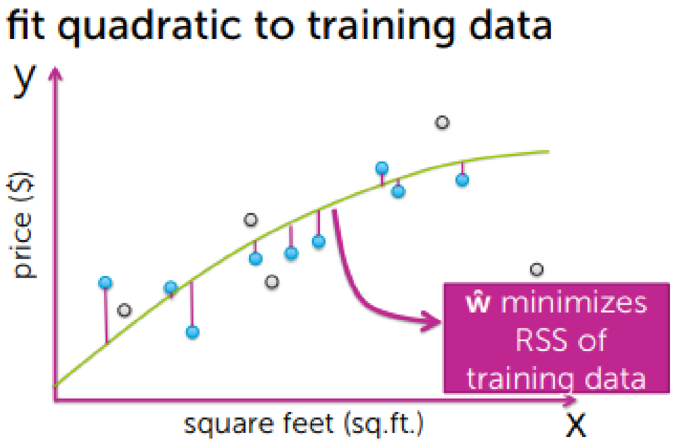
Part 3: Test Error:

- The Test Error -> Approximates the **Generalization Error**.
- Generalization Error -> It attempts to estimate the loss of all possible dataset pairs. (house, price);
- In order to caluclate the test error:

```
    - The dataset is divided into the training error and test error.
```

- **TEST ERROR** : avg. loss on houses in the test set.
- **TEST ERROR** = (1/N(test)) Σ(i in test set) * L(y(i),f(w-hat)(X(i)))

```
   - Where f((w-hat)(X(i))) -> fit using the training data.
   - The training data is different from the test data.
```

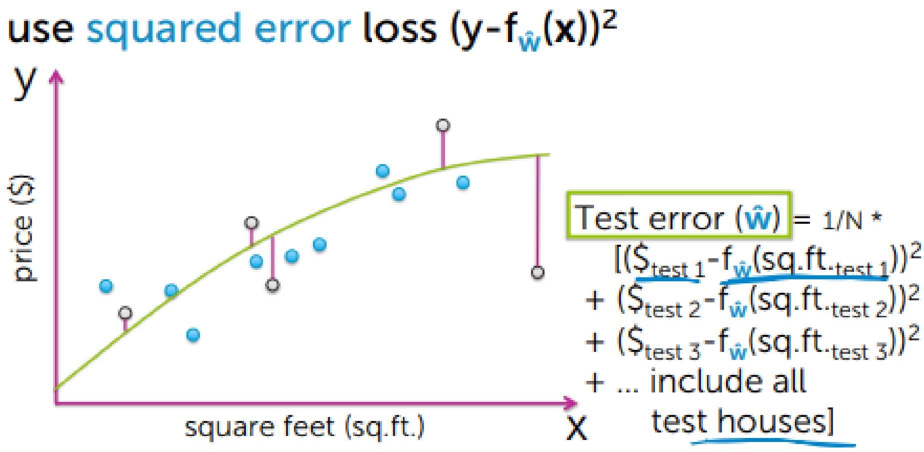

fit quadratic to training data

ŵ minimizes RSS of training data

```
blue dots -> training dataset
grey dots -> test dataset

The model was fit for the training
dataset.

The test error -> difference between
actual - predcition based on the fitted
model.
```


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

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

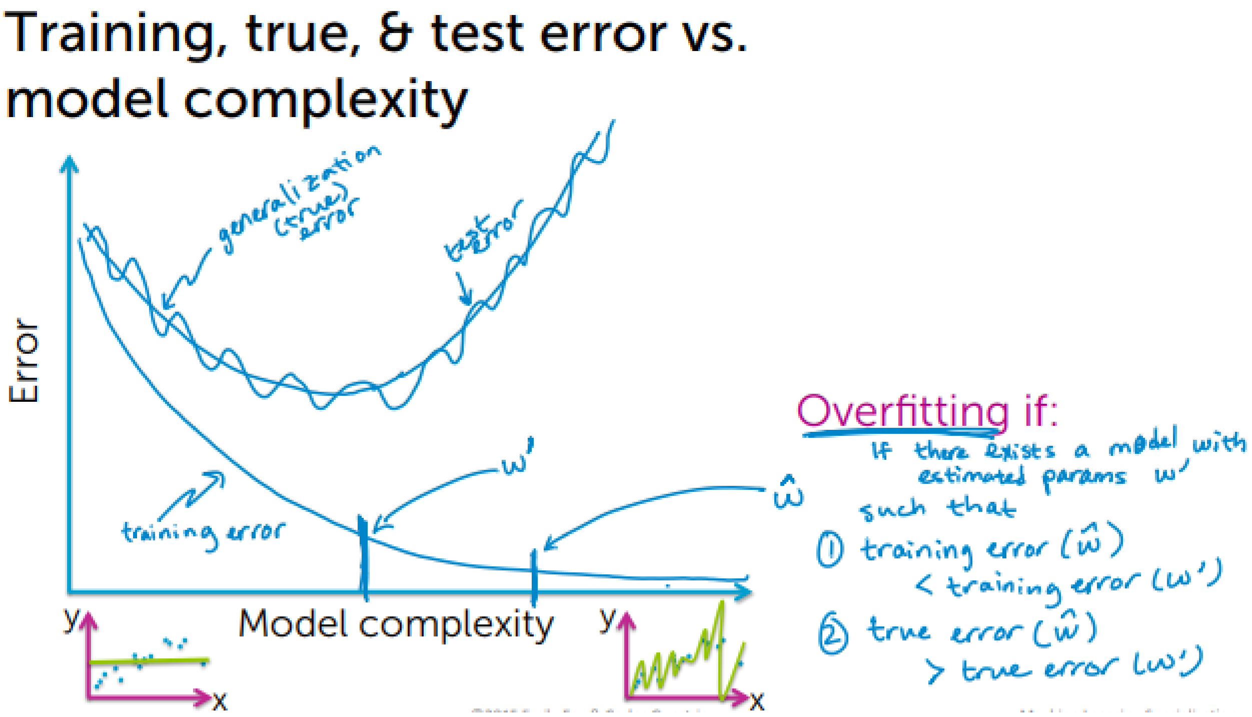
Errors:

- 1. Training error : Decreases with increase in model complexity.'
- 1. Generalization : Can't compute, could increase with low and high model complexity.
- 1. Test error : noisy approximation of the generalization error. Could increase with low and high model complexity.

Overfitting:

- The model is developed specific for the dataset.
- It can be computed based on: consider for a model with w-hat estimated parameters and another set of estimated parameters w-prime;

```
- training error(w-hat) < training error(w-prime);
- true error (w-hat) > true error (w-prime);
```



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 Splits:

- Training set -> few : the estimated parameters (w-hat) - poorly estimated.
- Test set -> few : the test error can be a bad approximation of the generalization error. Since it might not represent a wide array of thing.
- Typically - require enough test points to form a reasonable estimate of the generalization error.
- In case dof less dataset -> "Cross Validation"

3 Source of Errors:

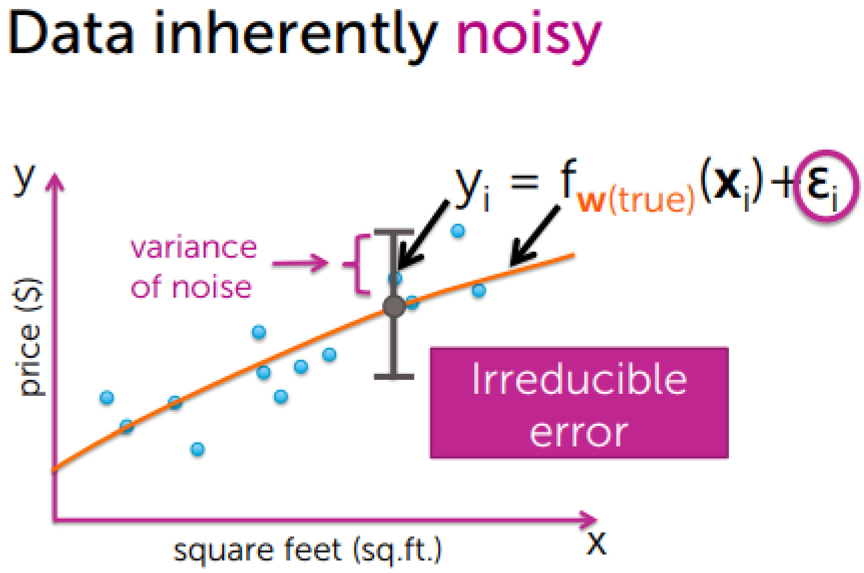
- When forming predictions, there are 3 sources of error:
- Noise
- Bias
- Variance

3 measures of errors:

1. Training Error
2. Generalization Error / true error
3. Test Error

1. Noise

- Data is inherently noisy.
- y(i) = f(w-true)(Xi) + ε(i);
- ε -> **error term** since the output y is not exactly predicted from the input features, there is an error term, there exist an error value.
- It was assumed the the noise has 0 mean.
- Spread of the noise - at a given square feet - the variation of the house price possible.
- **Variance of noise - of ε** - spread of the noise.This is a property of the data and has nothing to do with the model, fit, etc.
- Noise - Irreducible error. We have no control on this error.

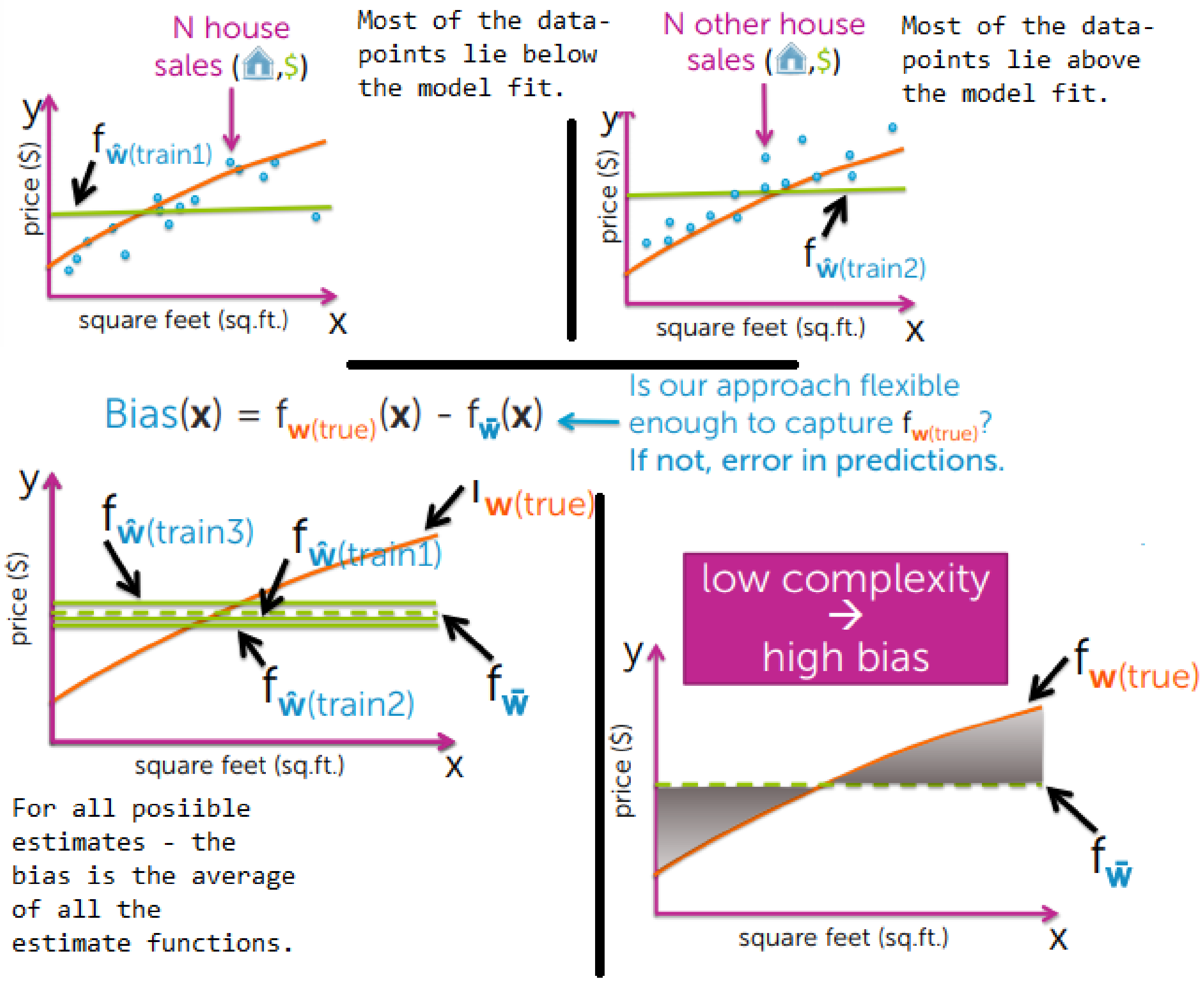


Data inherently noisy

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

2. Bias contribution

- Bias -> Measure of how well a model can fit the true relationship between x and y.



Most of the data-points lie below the model fit.

Most of the data-points lie above the model fit.

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

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

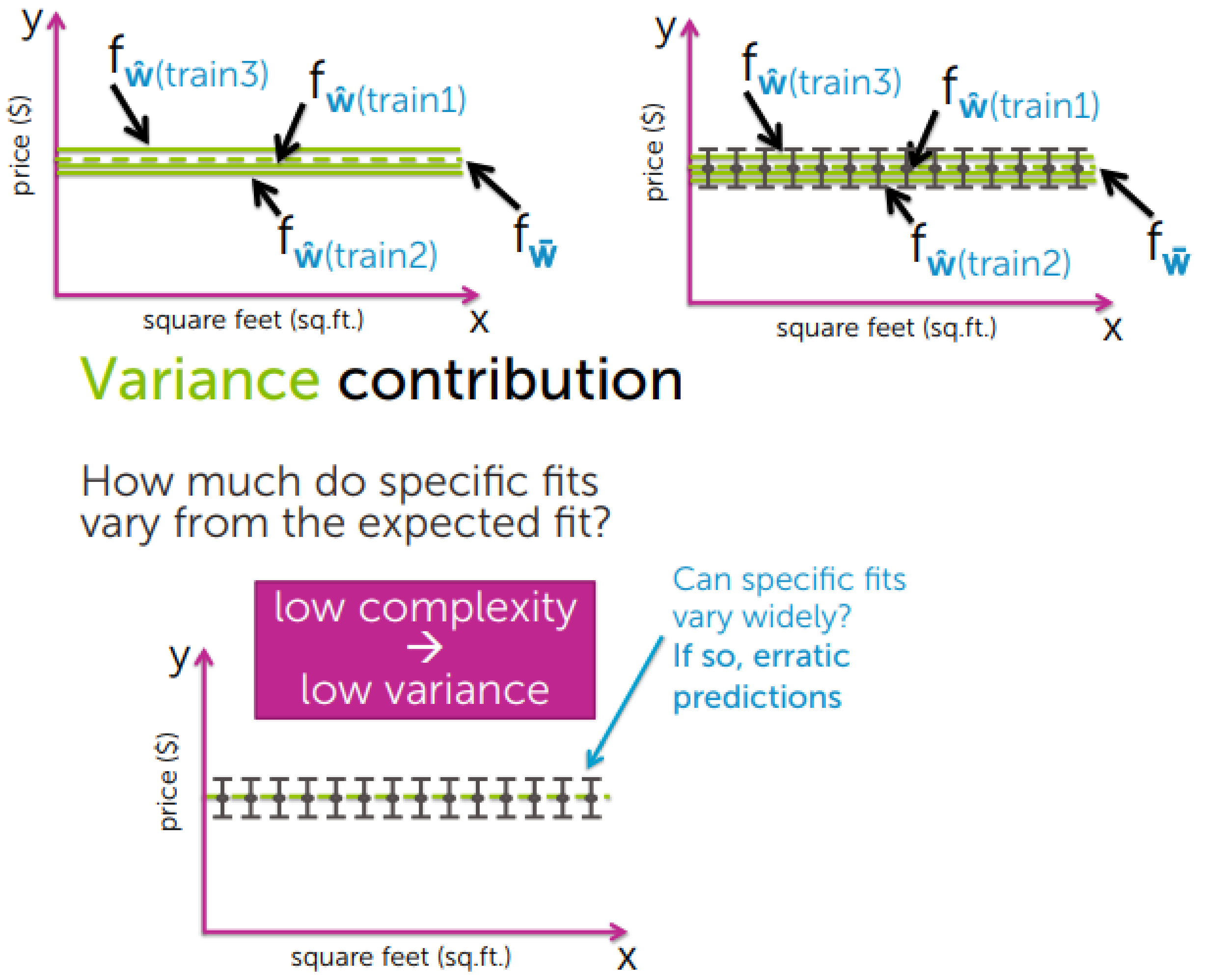
For all posible estimates - the bias is the average of all the estimate functions.

- Bias is the difference between the average fitted line (----) and true function (this case - quadratic fit);
- The grey shaded region is the **Difference between the true function and average fitted line (----)**
- Therefore Bias states if the model is flexible enough to capture f(w-true), else error in predictions.
- The model shown has :

```
- low complexity
- high bias
- not flexible to have a good approximation to a true relationship.
- this bias -> errors in predictions.
```

3. Variance contribution

- Variance - It is the measure of how much a specific fit can vary from the expected fit.
- Usually **Low complexity models** have **Low variance**.



Variance contribution

How much do specific fits vary from the expected fit?

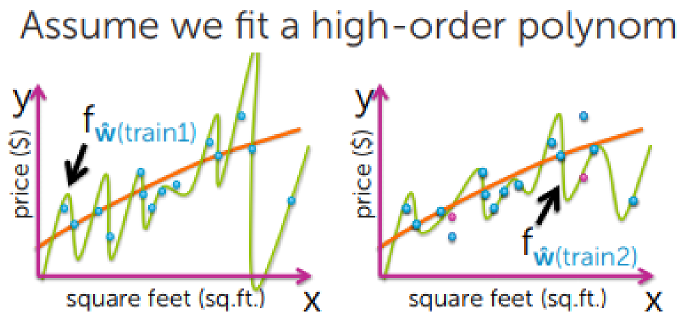
low complexity → low variance

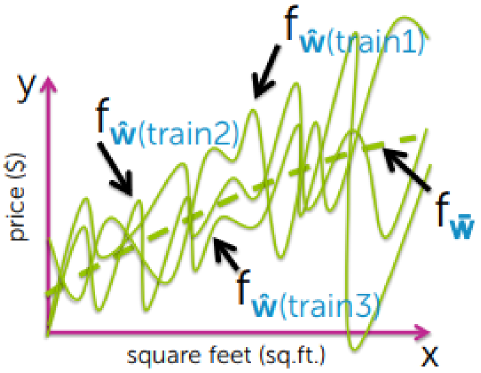
Can specific fits vary widely? If so, erratic predictions

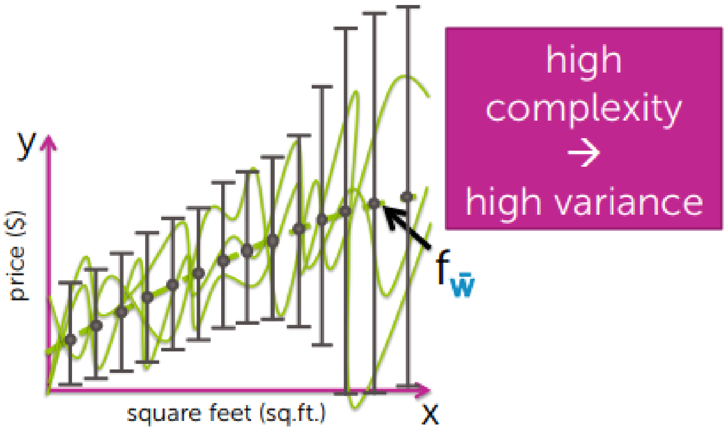
- Variance of high-complexity model:

```
- In case the dataset is fit with a high-order polynomial.
- The model fit changes even for the slightest change in the dataset.
```

- For a **high complexity model** the **variance is high**.

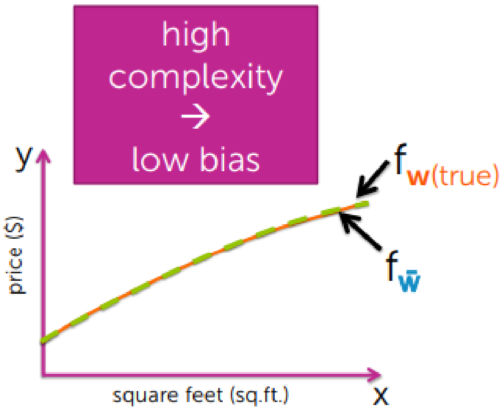

Assume we fit a high-order polynomial

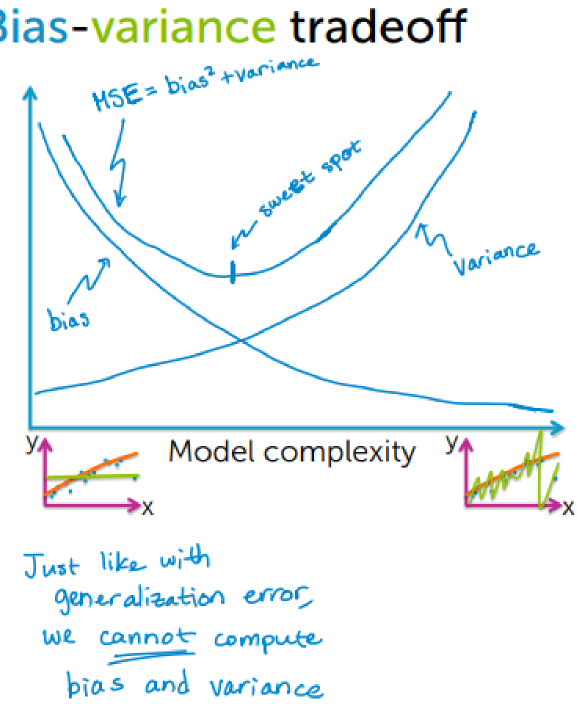



high complexity → high variance

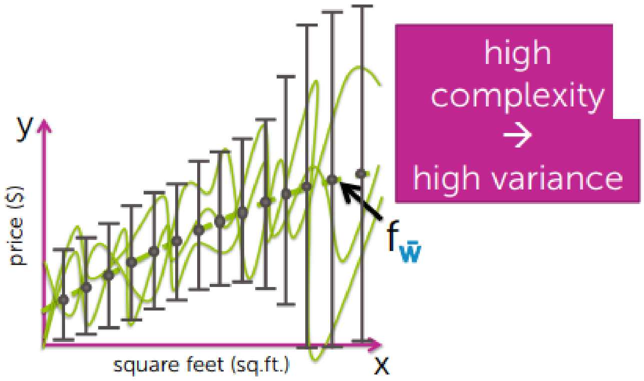
Bias-Variance Trade-off

- Model complexity increases -> Bias decrease.
- Model complexity increases -> Variance increases.
- Mean squared error -> sum(bias^2 + variance);
- Goal : Find between the bias-variance.


high complexity → low bias


Bias-variance tradeoff

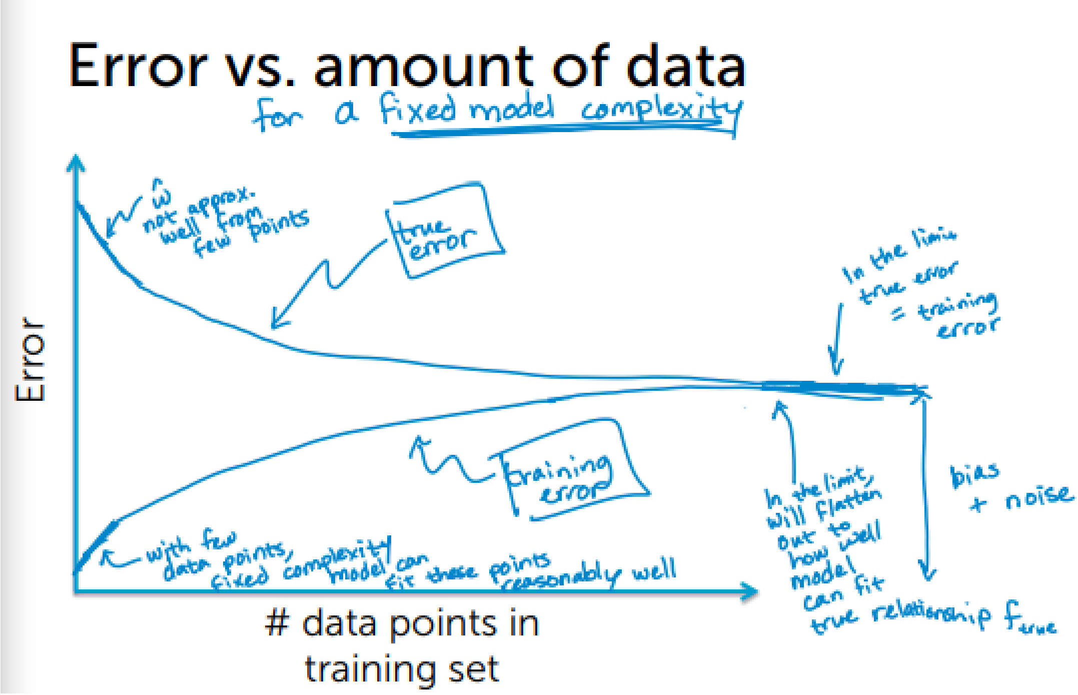
Just like with generalization error, we cannot compute bias and variance


high complexity → high variance

Error vs. amount of data

- For fixed model complexity.

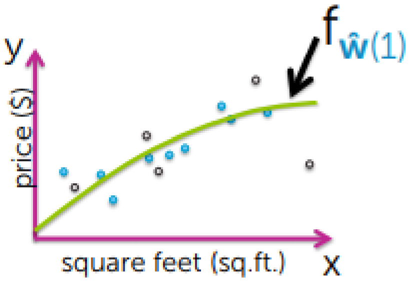
- True error -> it **decreases** with **increase in data-points in the data-set**. Thereafter it remains constant. The constant limit -> **bias + noise**
- Training error -> It **increases** with **increase in the data-points in the data-set**. Thereafter it remains constant. The constant limiy -> **bias + noise**
- The constant limit reaches when the model is fit to all the possible dataset available. The training and true error converge at that point.
- In thelimit, the curve will flatten out to how well the moel can fit true relationship. f(true);

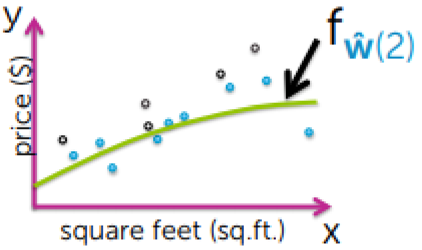


Formally defining and deriving 3 sources of errors

Accounting for training set randomness

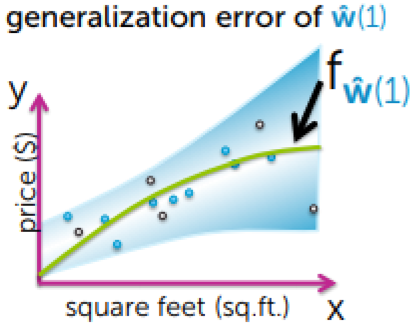
- Training set was just a random sample of N houses sold.If N other houses had been sold and recorded, then the model fit will change.

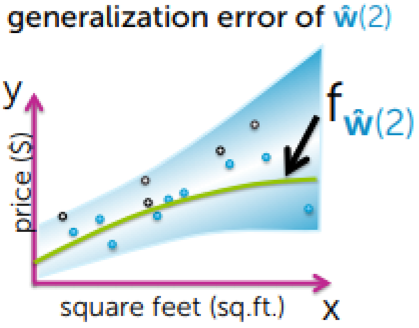




- The above fits on the model change based on the training points choosen.

The performance of each of these fits can be assessed using the Generalization Error.

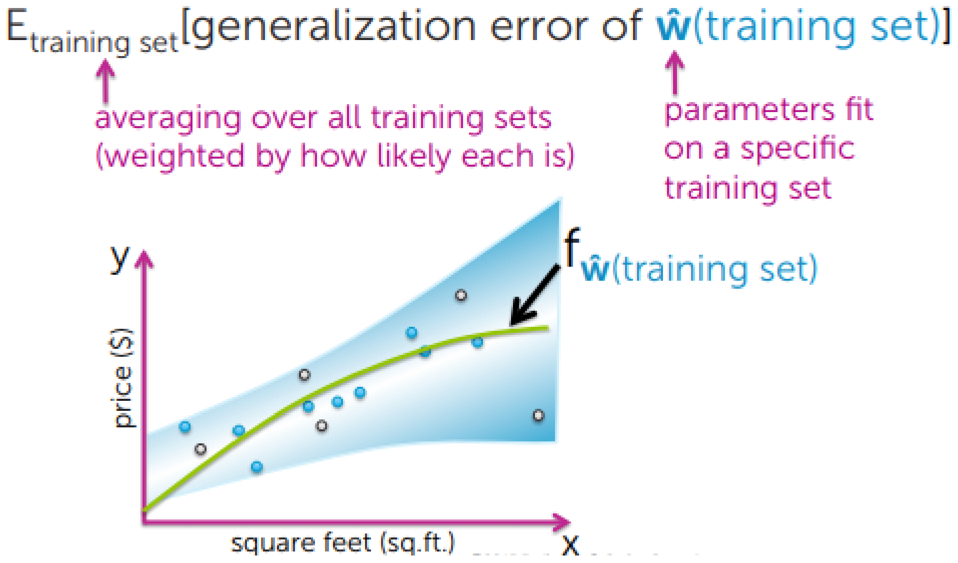




Ideally want performance averaged over all possible training sets of size N.

Expected Prediction Error :

- E(training set) [generalization error of w(hat)(training set)]


E_training set[generalization error of ŵ(training set)]
averaging over all training sets (weighted by how likely each is)
parameters fit on a specific training set
f_ŵ(training set)
y
price ($)
square feet (sq.ft.)
x

Prediction error at target input

- Consider:

```
1. Loss at target X(t) (e.g 2640 sq.ft)
2. Squared error loss L(y, f(w-hat)(X)) = (y-f(w-hat)(X))^2
```

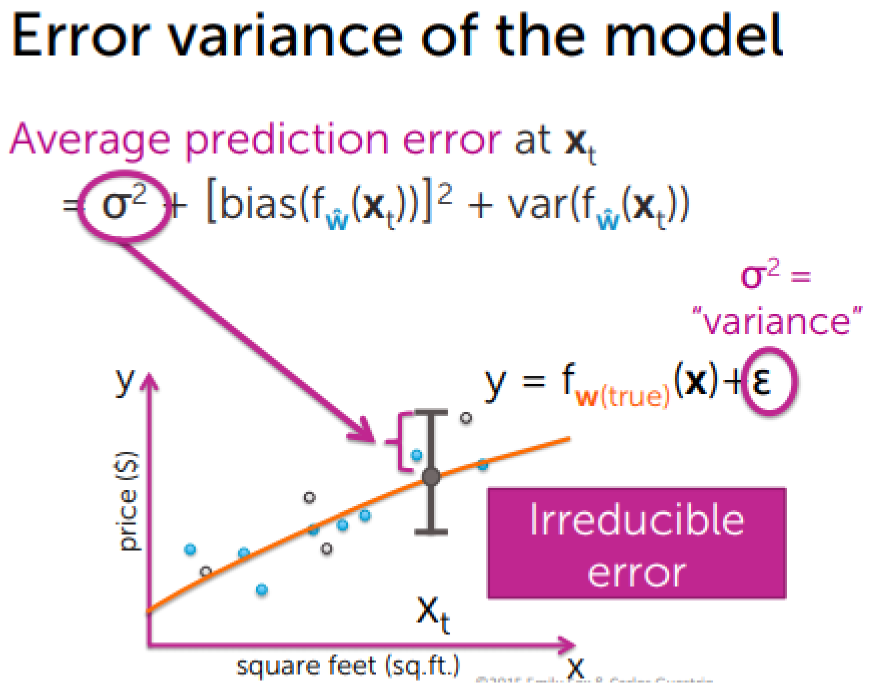
Sum of 3 sources of errors

- Average prediction error at X(t)

```
= σ^2 + [bias(f(w-hat)(X(t)))]^2 + var(f(w-hat)(X(t)))
```

Term 1 : Error variance of the model

- **σ^2** -> variance of the (noise - irreducible error) It is the spread of noise.
- There is some true relationship between sqft(x) and price of the house(y);But there are other factors that influence the price - and all these factors are captured by the additive term **ε**.
- This noise term **ε** - has 0 mean.
- The spread of noise at any point in the input space, this spread is termed as variance - denoted by - **σ^2**;



Error variance of the model

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

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

σ^2 = "variance"

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

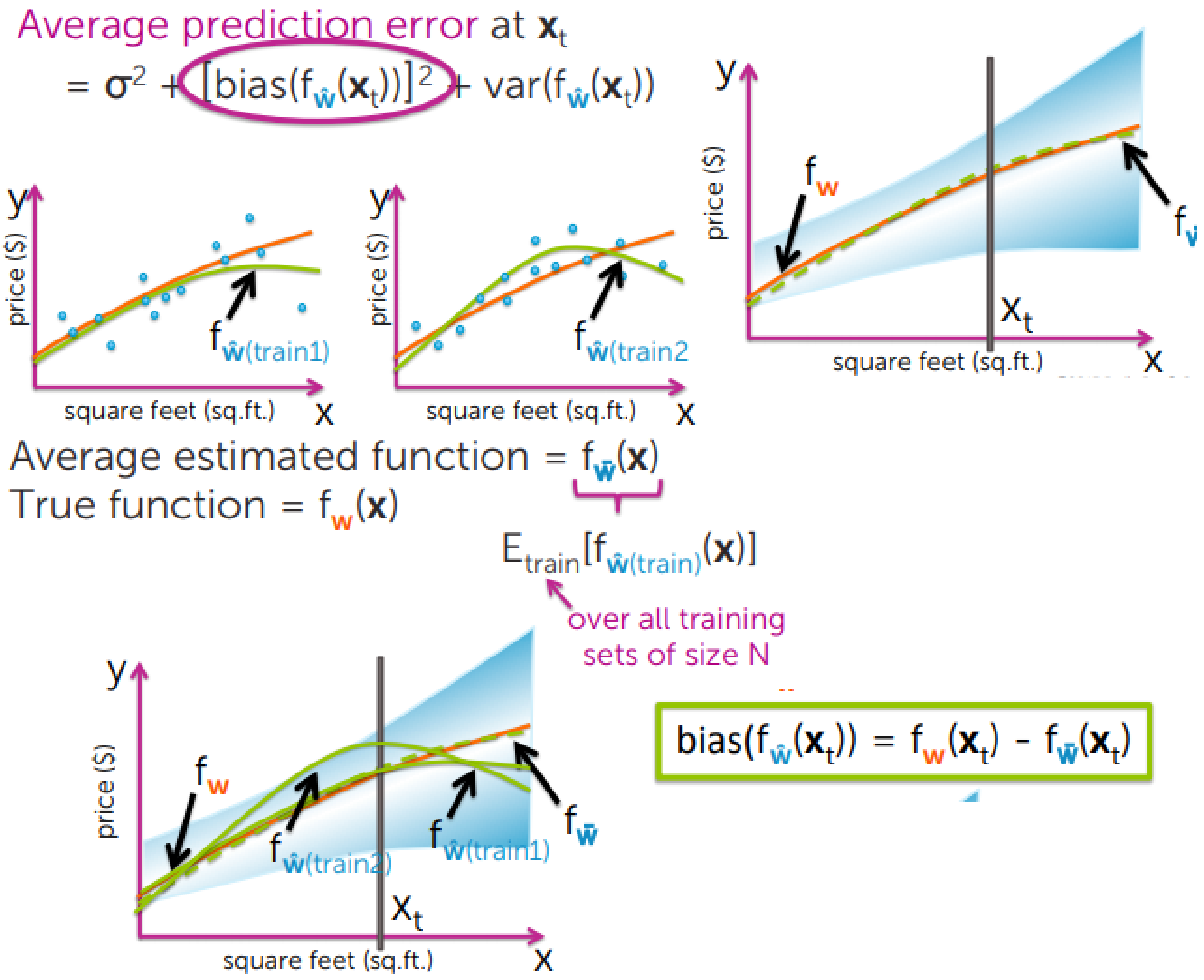
Irreducible error

Term 2 : Bias function estimator

- Bias - how well the model could on an average fit the true relationship between x and y.
- The a generic model is fit on the dataset -> f-w(X)
- Different N set of training dataset -> different model fit -> f-w-hat(train1), f-w-hat(train2), etc;
- Therefore and average estimated function is constructed

```
          = f(w-bar)(X) = E(train)[f(w-hat-train)(X)]
```

- bias(f(w-hat)(Xt)) = f-w(Xt) - f-w-bar(Xt)



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

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

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

True function = $f_w(\mathbf{x})$

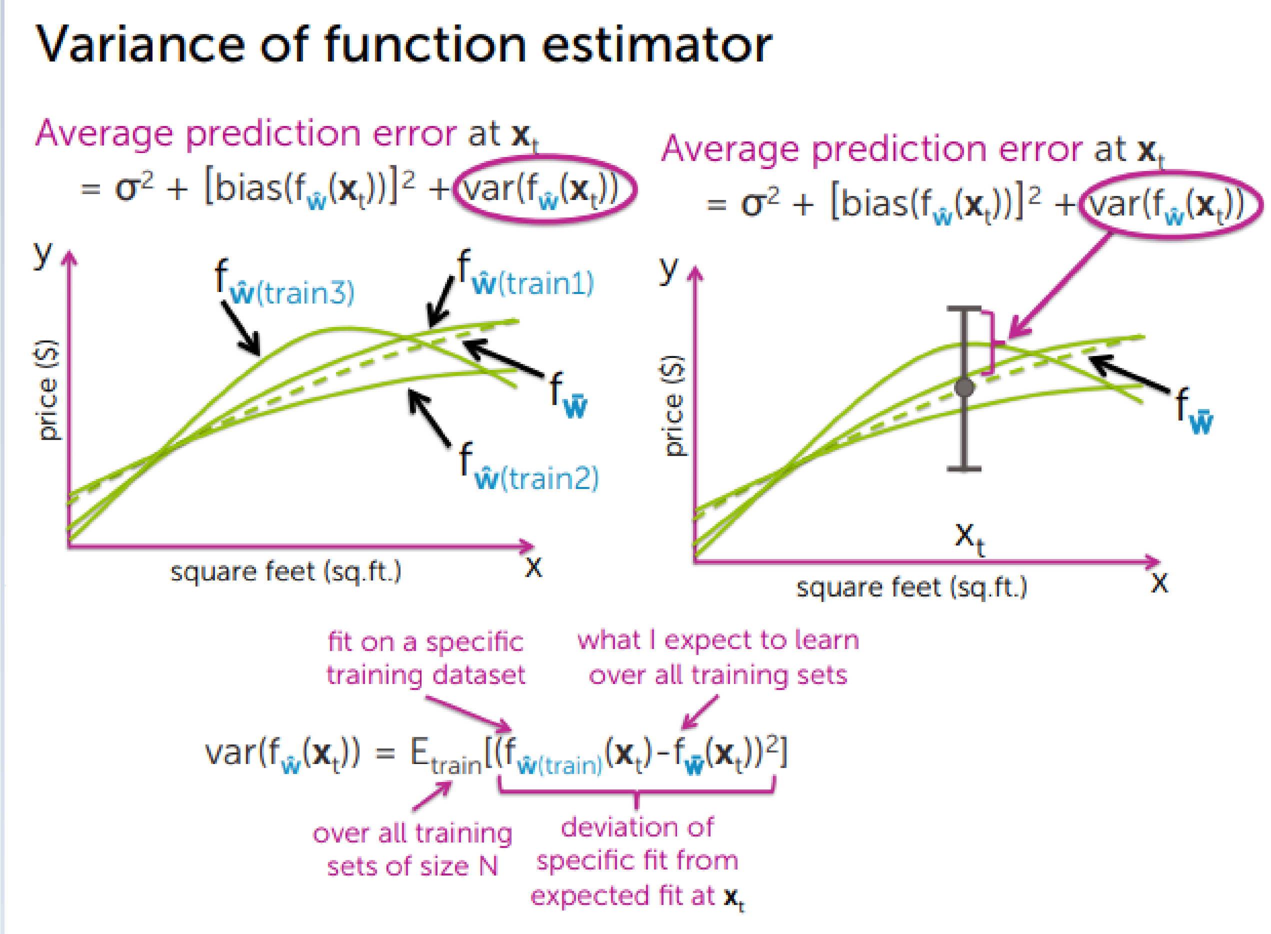
over all training sets of size N

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

y
price ($)
square feet (sq.ft.)
x
$f_{\hat{w}(\text{train1})}$
$f_{\hat{w}(\text{train2}}$
f_w
$f_{\bar{w}}$
x_t
$f_{\hat{w}(\text{train2})}$
$f_{\hat{w}(\text{train1})}$

Term 3 : Variance of the function estimator

- The average fit (green dashed line) is compared w.r.t to other fits.
- Over all possible fits , how much does the fit deviate from the mean fitted line(green dashed line);
- Variance - some random variable is the difference of the expected random value and the mean(dashed line ----), and then squared.



Variance of function estimator

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

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

fit on a specific training dataset

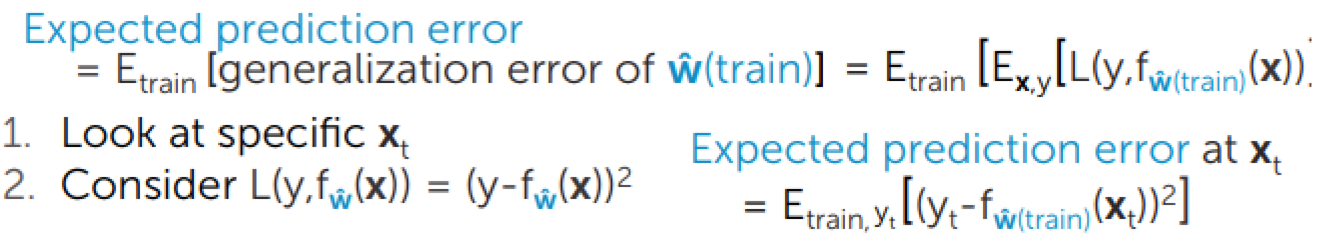
what I expect to learn over all training sets

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

over all training sets of size N

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

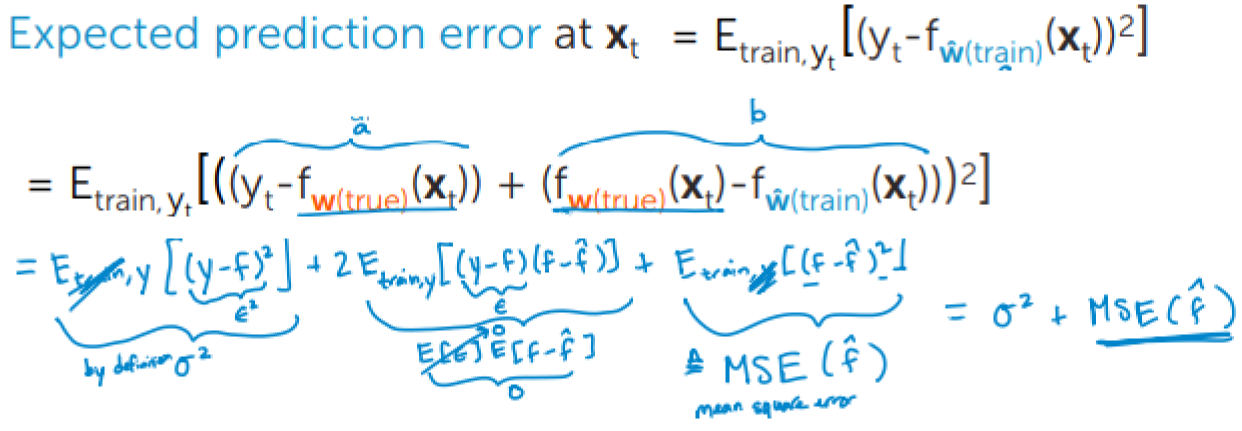
Deriving Expected prediction error



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

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

Expected prediction error at $\mathbf{x}_t$
$= E_{\text{train}, y_t}[(y_t - f_{\hat{w}(\text{train})}(\mathbf{x}_t))^2]$



Expected prediction error at $\mathbf{x}_t$ $= E_{\text{train}, y_t}[(y_t - f_{\hat{w}(\text{train})}(\mathbf{x}_t))^2]$

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

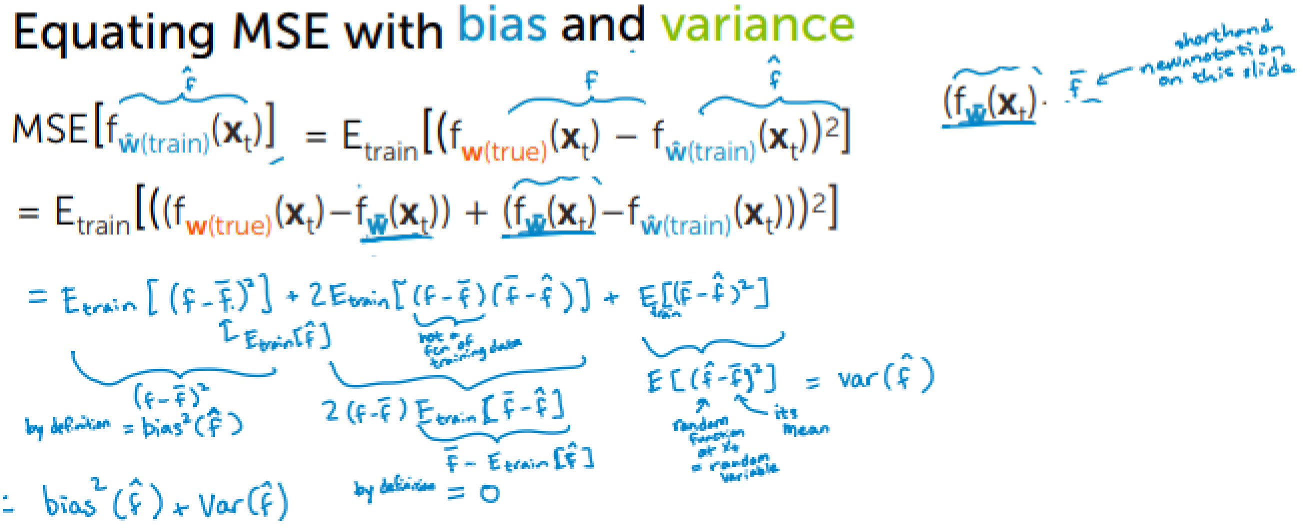
$$= \underbrace{E_{\cancel{\text{train}}, y}[\underbrace{(y-f)^2}_{\epsilon^2}]}_{\text{by definition } \sigma^2} + 2\underbrace{E_{\text{train},y}[\underbrace{(y-f)}_{\epsilon}(f-\hat{f})]}_{\underbrace{E[\epsilon] E[f-\hat{f}]}_{0}} + \underbrace{E_{\text{train}, \cancel{y}}[(f-\hat{f})^2]}_{\triangleq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

mean square error

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

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

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



Equating MSE with bias and variance

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

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

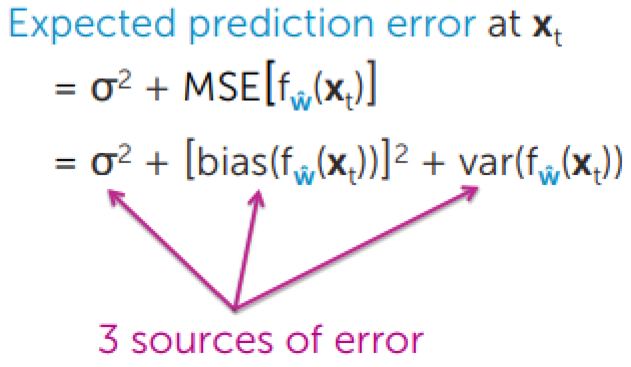
$(f_{\bar{w}}(\mathbf{x}_t))$. $\bar{f}$ ← shorthand new notation on this slide

$$= \underbrace{E_{\text{train}}[(f - \bar{f})^2]}_{\substack{(f-\bar{f})^2 \\ \text{by definition } = \text{bias}^2(\hat{f})}} + \underbrace{2E_{\text{train}}[\underbrace{(f-\bar{f})}_{\text{not a fcn of training data}}(\bar{f}-\hat{f})]}_{2(f-\bar{f})\underbrace{E_{\text{train}}[\bar{f}-\hat{f}]}_{\bar{f} - E_{\text{train}}[\hat{f}] \text{ by definition } = 0}} + \underbrace{E_{\text{train}}[(\bar{f}-\hat{f})^2]}_{E[(\hat{f}-\bar{f})^2] = \text{var}(\hat{f})}$$

$\bar{f}$ ↑ $E_{\text{train}}[\hat{f}]$

$\hat{f}$ ← random function at x_t = random variable; $\bar{f}$ ← its mean

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



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

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

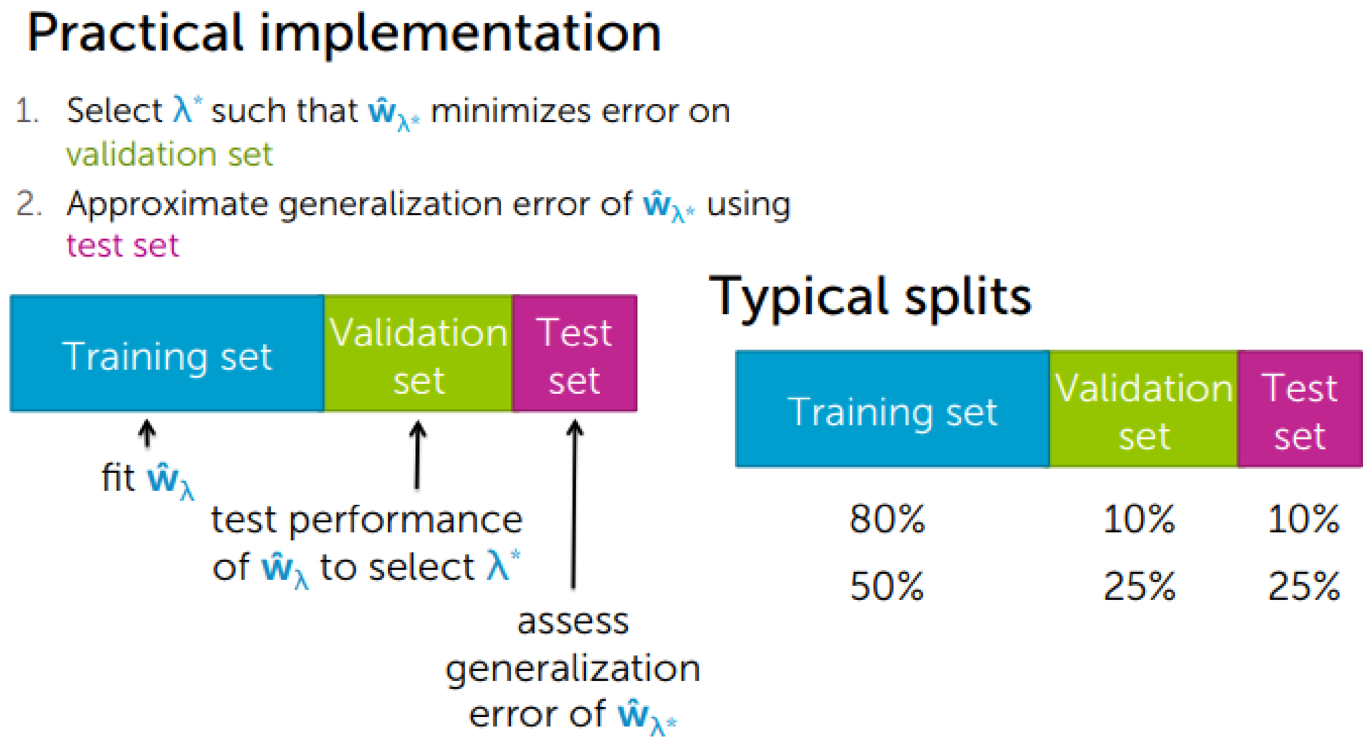
$$= \sigma^2 + [\text{bias}(f_{\hat{w}}(\mathbf{x}_t))]^2 + \text{var}(f_{\hat{w}}(\mathbf{x}_t))$$

3 sources of error

Training/Validation/Test split - Model Selection, Fitting, and Assessment

The regression ML workflow

1. **Model Selection** : Often need to choose tuning parameter (λ) - to control the model complexity. (e.g. degree of polynomial.)
2. **Model Assessment** : Having selected a model, assess the generalization error.



Practical implementation

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

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

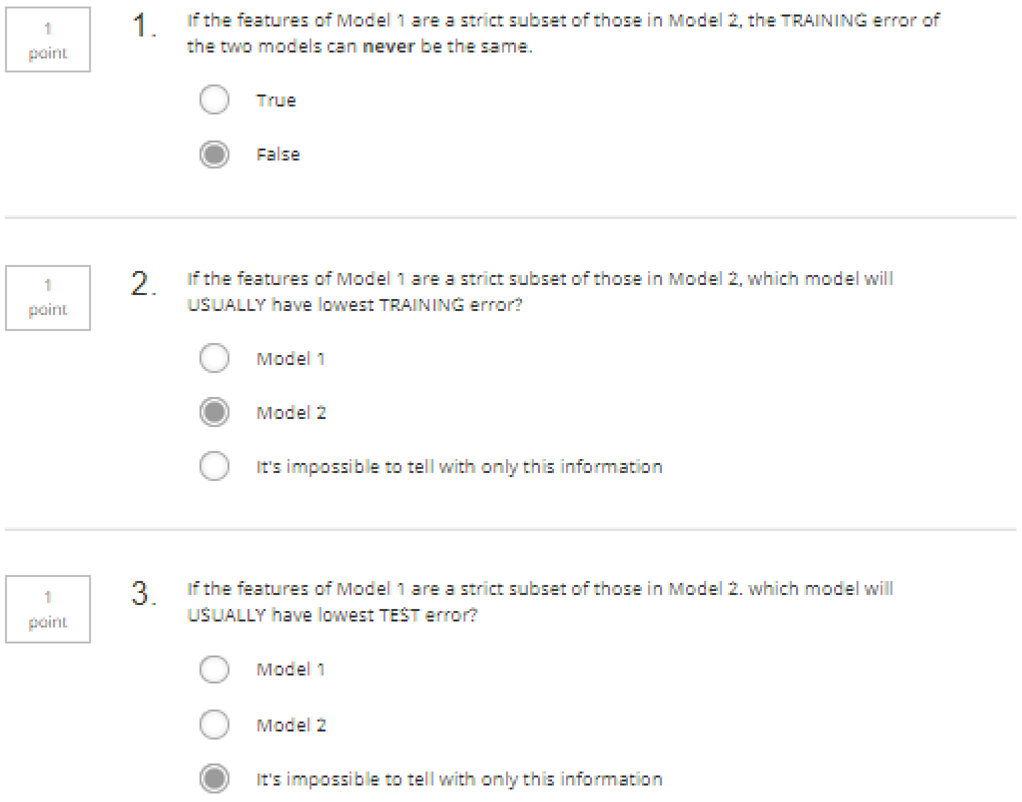
Training set	Validation set	Test set
80%	10%	10%
50%	25%	25%

$\hat{\mathbf{w}}_\lambda$ = estimate parameters on training data

λ^* = tuning parameter to control te model complexity with lowest test error

$\hat{\mathbf{w}}_{\lambda^*}$ = Fitted model for selected complexity λ*

Quiz



1 point

1. If the features of Model 1 are a strict subset of those in Model 2, the TRAINING error of the two models can **never** be the same.

○ True

● False

1 point

2. If the features of Model 1 are a strict subset of those in Model 2, which model will USUALLY have lowest TRAINING error?

○ Model 1

● Model 2

○ It's impossible to tell with only this information

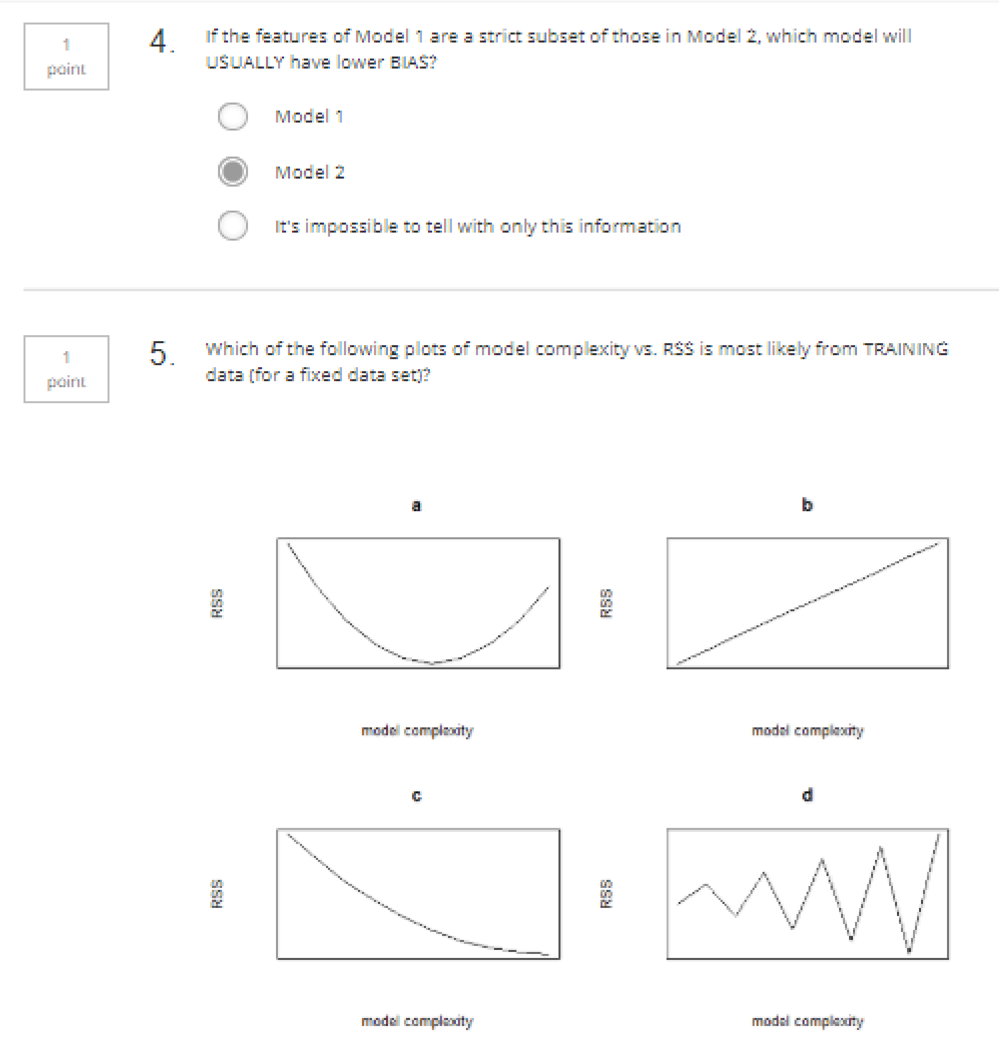
1 point

3. If the features of Model 1 are a strict subset of those in Model 2. which model will USUALLY have lowest TEST error?

○ Model 1

○ Model 2

● It's impossible to tell with only this information



1 point

4. If the features of Model 1 are a strict subset of those in Model 2, which model will USUALLY have lower BIAS?

○ Model 1

● Model 2

○ It's impossible to tell with only this information

1 point

5. Which of the following plots of model complexity vs. RSS is most likely from TRAINING data (for a fixed data set)?

○ a

○ b

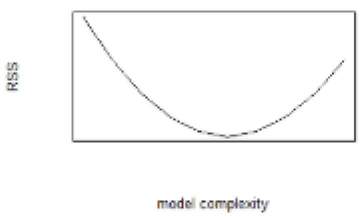
● c

○ d

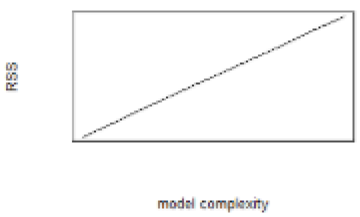
1 point

6. Which of the following plots of model complexity vs. RSS is most likely from TEST data (for a fixed data set)?

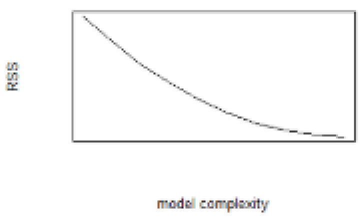
a



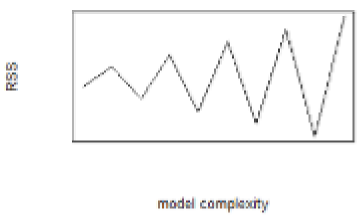
b



c



d



● a

○ b

○ c

○ d

1 point

7. It is **always** optimal to add more features to a regression model.

○ True

● False

1 point

8. A simple model with few parameters is most likely to suffer from:

● High Bias

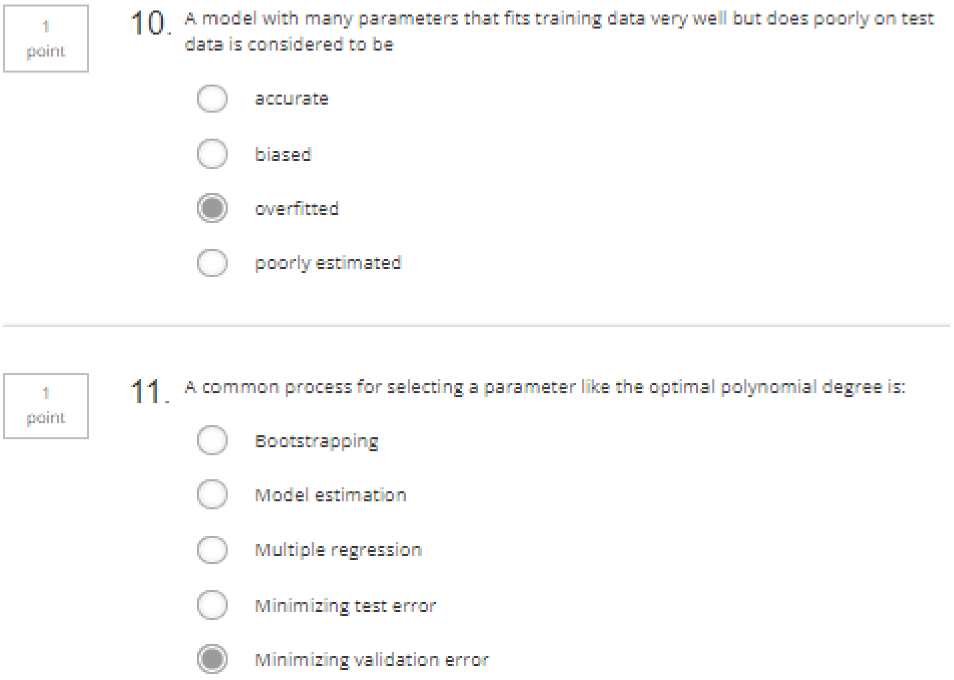
○ High Variance

1 point

9. A complex model with many parameters is most likely to suffer from:

○ High Bias

● High Variance



1 point

10. A model with many parameters that fits training data very well but does poorly on test data is considered to be

○ accurate

○ biased

● overfitted

○ poorly estimated

1 point

11. A common process for selecting a parameter like the optimal polynomial degree is:

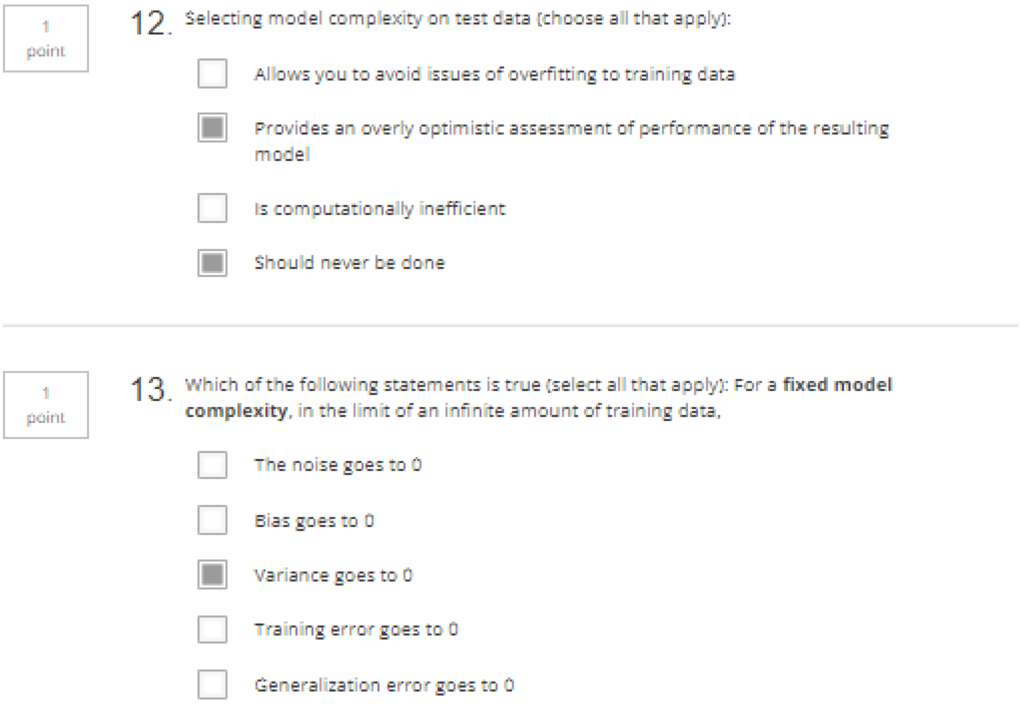
○ Bootstrapping

○ Model estimation

○ Multiple regression

○ Minimizing test error

● Minimizing validation error



1 point

12. Selecting model complexity on test data (choose all that apply):

☐ Allows you to avoid issues of overfitting to training data

☑ Provides an overly optimistic assessment of performance of the resulting model

☐ Is computationally inefficient

☑ Should never be done

1 point

13. Which of the following statements is true (select all that apply): For a **fixed model complexity**, in the limit of an infinite amount of training data,

☐ The noise goes to 0

☐ Bias goes to 0

☑ Variance goes to 0

☐ Training error goes to 0

☐ Generalization error goes to 0

- Model 1 is a strict subset of model 2 -> all elements in model 1 are a part of Model 2. While Model 2 has more parameters.
- Model 2 is more complex than Model 1:

```
    High Bias - Low complexity -> Model 1
    High Variance - High Complexity -> Model 2 (more features)
```

- Training error -> decreases with increase in model complexity.
- True and test error are not proportional to the model complexity.
- With infinite amount of data -> variance = 0;

```
In [ ]:
```