Week 4

Cogs 109: Data Analysis and Modeling

Fall 2017

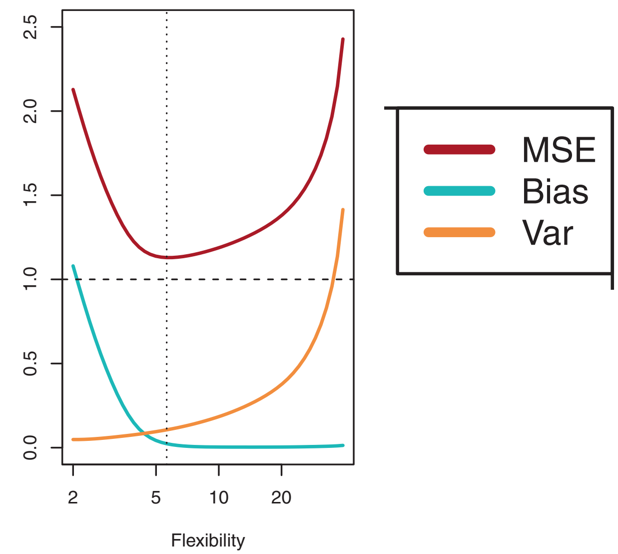
Prof. Eran Mukamel

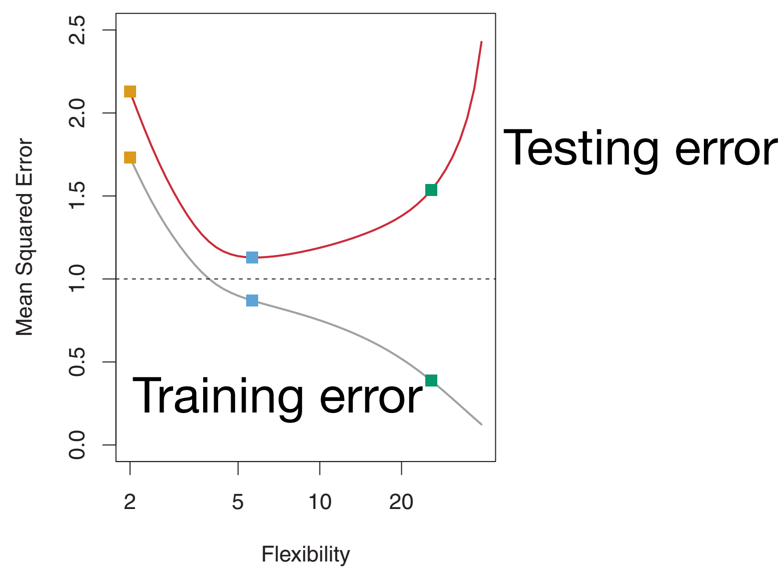
Recap of terminology for classification

- Logistic regression
- LDA: Linear discriminant analysis
- Odds ratio, log-odds, logistic function
- Bayesian classifier
- Prior probability, posterior probability, data likelihood
- Decision boundary, decision threshold
- Confusion matrix
  - Errors: False positive, False negatives
  - Sensitivity, specificity
- ROC analysis, ROC curve

Resampling for model selection and assessment

- Recall the key tradeoffs in data modeling:
  - Bias vs. Variance
  - Training vs. Testing error
  - Flexible vs. Simple models
- Resampling is an incredibly useful way to choose a model that balances these tradeoffs



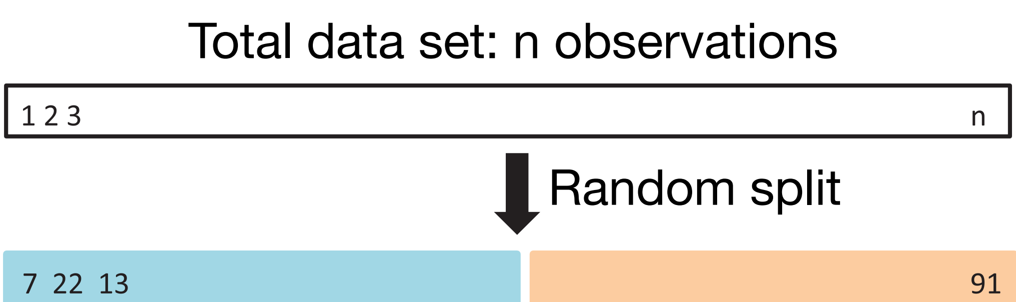


Two uses for resampling

- **Model assessment:** How good is our model?
    - Training error vs. Testing error
    - We really care about testing error, i.e. predictive performance for new data
- **Model selection:** Which model should I choose to achieve the lowest possible testing error?
    - Example: K-nearest neighbor classifier with K=1, 2, … or 100?
    - Example: Fit a line, quadratic, or higher-order polynomial?

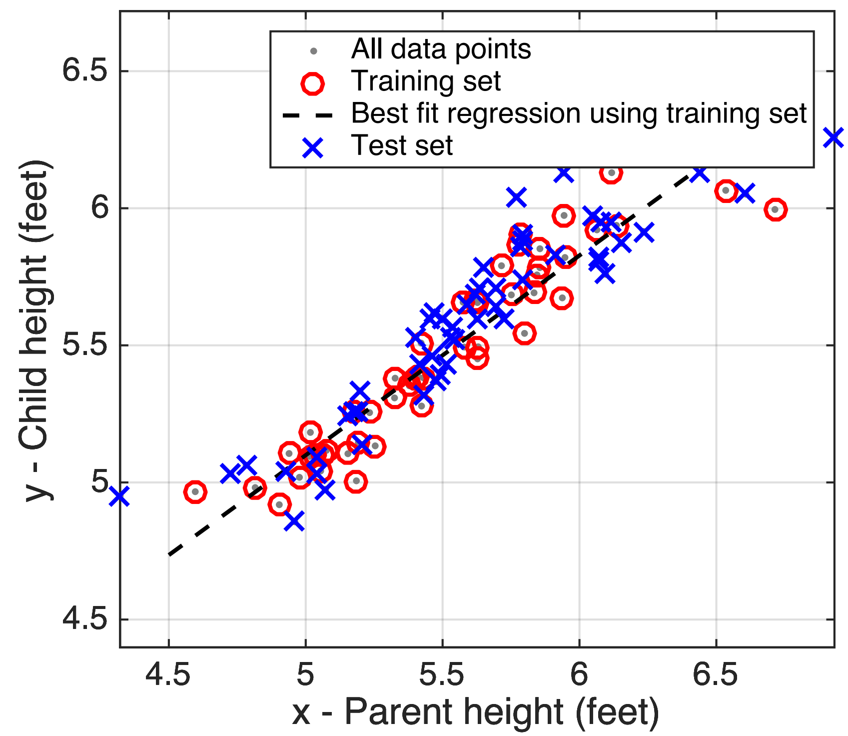
Validation set

- Given that we have *n* total observations, we can split these (randomly) into training and testing (validation) sets



<u>Training set:</u>
Use these data points to fit the model

<u>Test set:</u>
Use these data points to evaluate the testing error



Cross-validation workflow

Total data set: n observations

1 2 3 n

1. Random split

7 22 13 91

<u>Training set:</u>

$$X_{train} = \{x_7, x_{22}, x_{13}, \ldots\}$$

<u>Test set:</u>

$$X_{test} = X \setminus X_{train} = \{\ldots, x_{91}\}$$

2. Fit model

$$\{\beta_0, \beta_1, \ldots\}$$

3. Make predictions

$$\hat{y}_{train} = \{\hat{y}_7, \hat{y}_{22}, \hat{y}_{13}, \ldots\}$$

$$\hat{y}_{test} = \{\ldots, \hat{y}_{91}\}$$

4. Evaluate error (MSE)

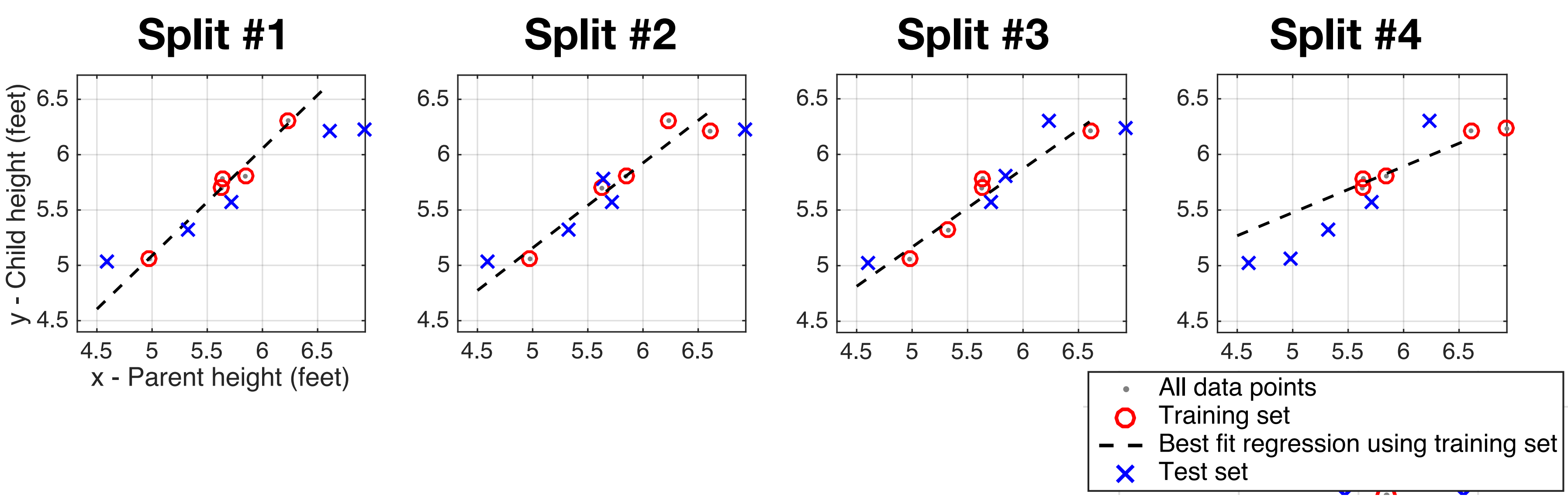
Resubstitution error:

$$MSE_{resub} = \frac{1}{N_{train}} \sum_{i \in train} (y_i - \hat{y}_i)^2$$

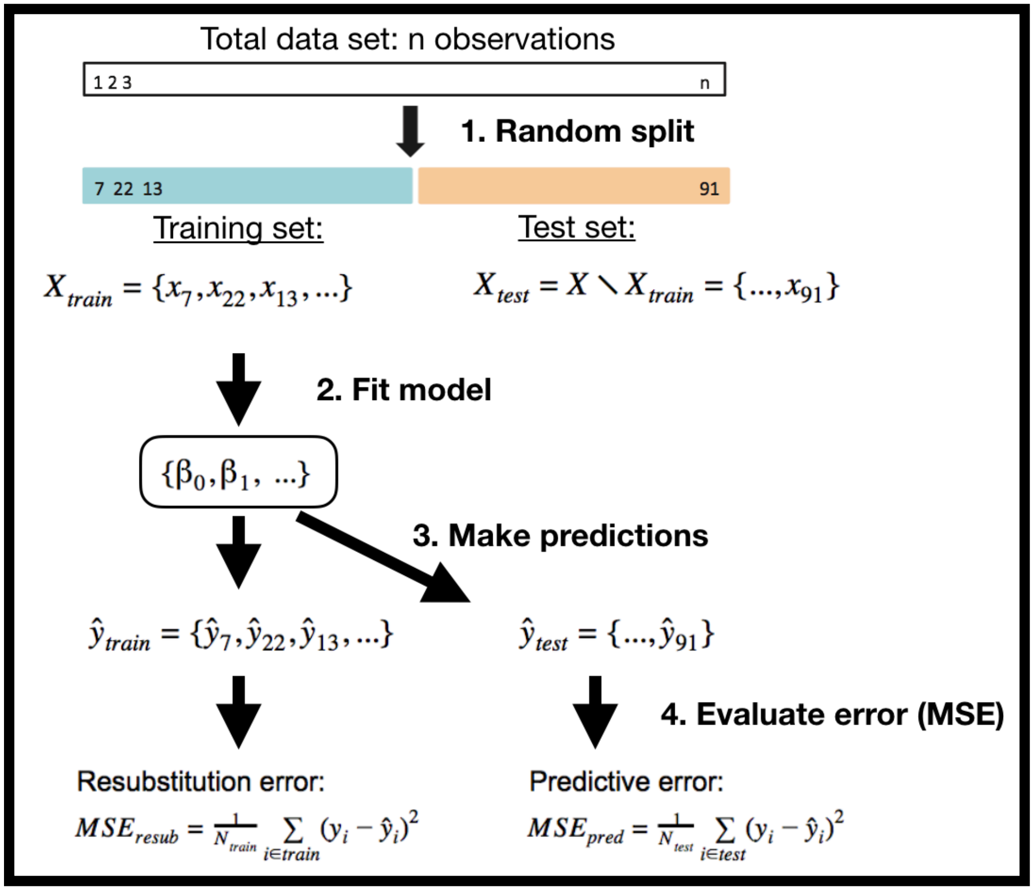
Predictive error:

$$MSE_{pred} = \frac{1}{N_{test}} \sum_{i \in test} (y_i - \hat{y}_i)^2$$

- Each random split of the data will give a different model fit and different mean squared error (MSE)
- What we really care about is the average MSE on validation sets



For i in 1…100 {



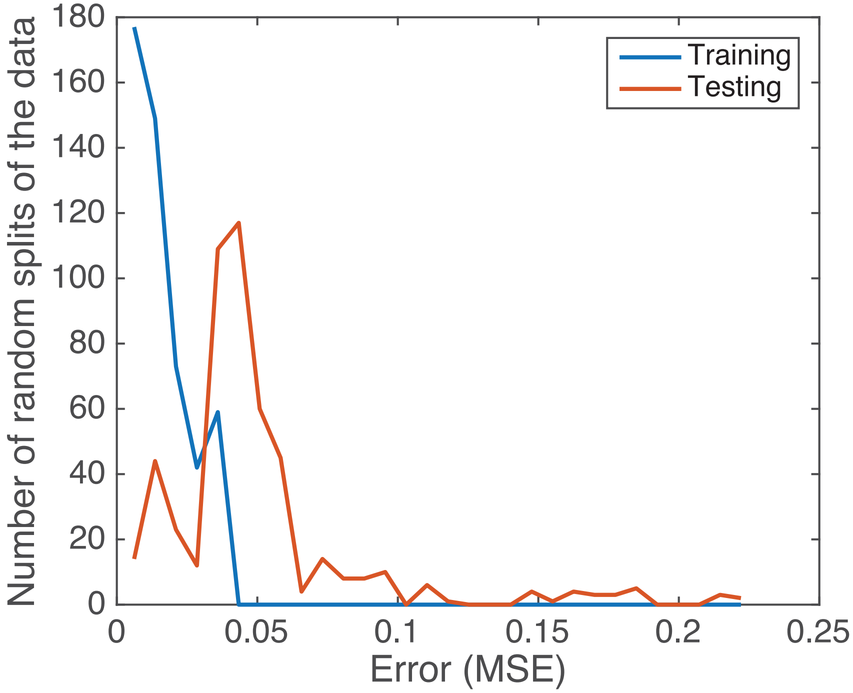
Store MSEpred[i]

}

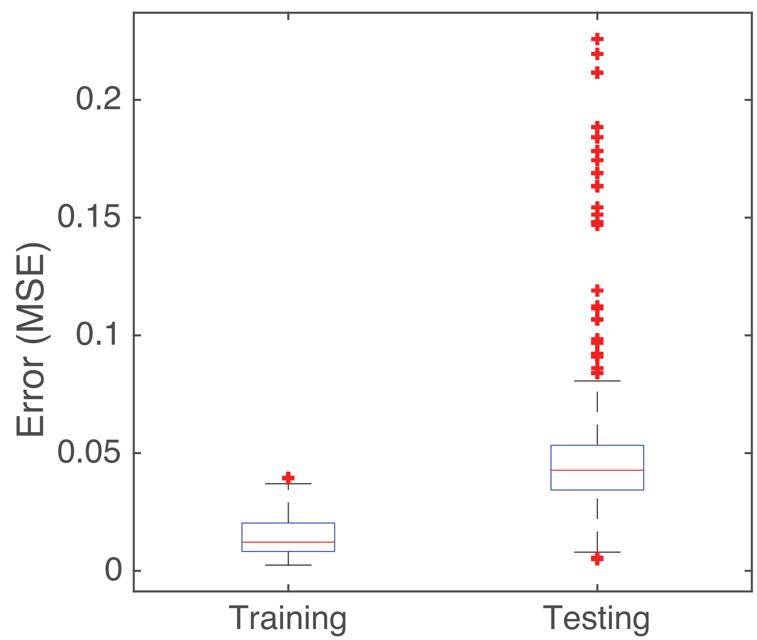
Evaluate mean MSEpred

Example code for cross-validation

```
train_error = []; test_error = [];
for j=1:500
    test_fraction = 0.5;

    % Create a logical vector (true/false) that splits the data into
    % training and testing sets
    % Method 1:
    test = false(N,1);
    test(randperm(N,N/2)) = true;

    % Method 2:
    test = rand(N,1)<test_fraction;

    train = ~test;
    fit = fitlm(x(train),y(train));

    yhat = predict(fit,x);
    train_error(j) = mean( (yhat(train)-y(train)).^2 );
    test_error(j) = mean( (yhat(test)-y(test)).^2 );
end
```



Training
Testing
Number of random splits of the data
Error (MSE)
0
20
40
60
80
100
120
140
160
180
0
0.05
0.1
0.15
0.2
0.25



Error (MSE)
0
0.05
0.1
0.15
0.2
Training
Testing

Using cross-validation to compare models

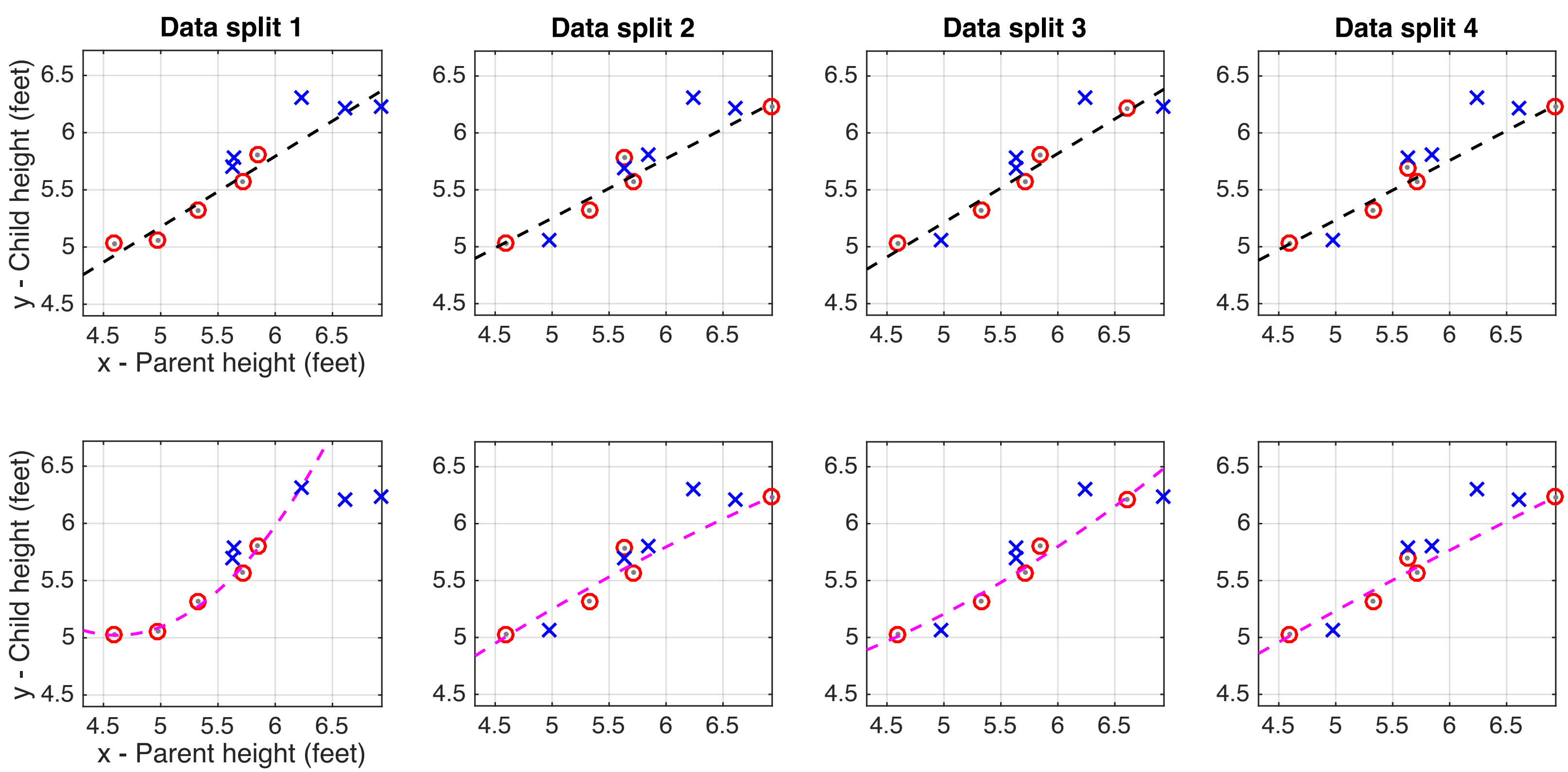
- Compare a linear regression vs. a quadratic regression (i.e. fitting a 2nd order polynomial)

Linear:
y ~ 1+x

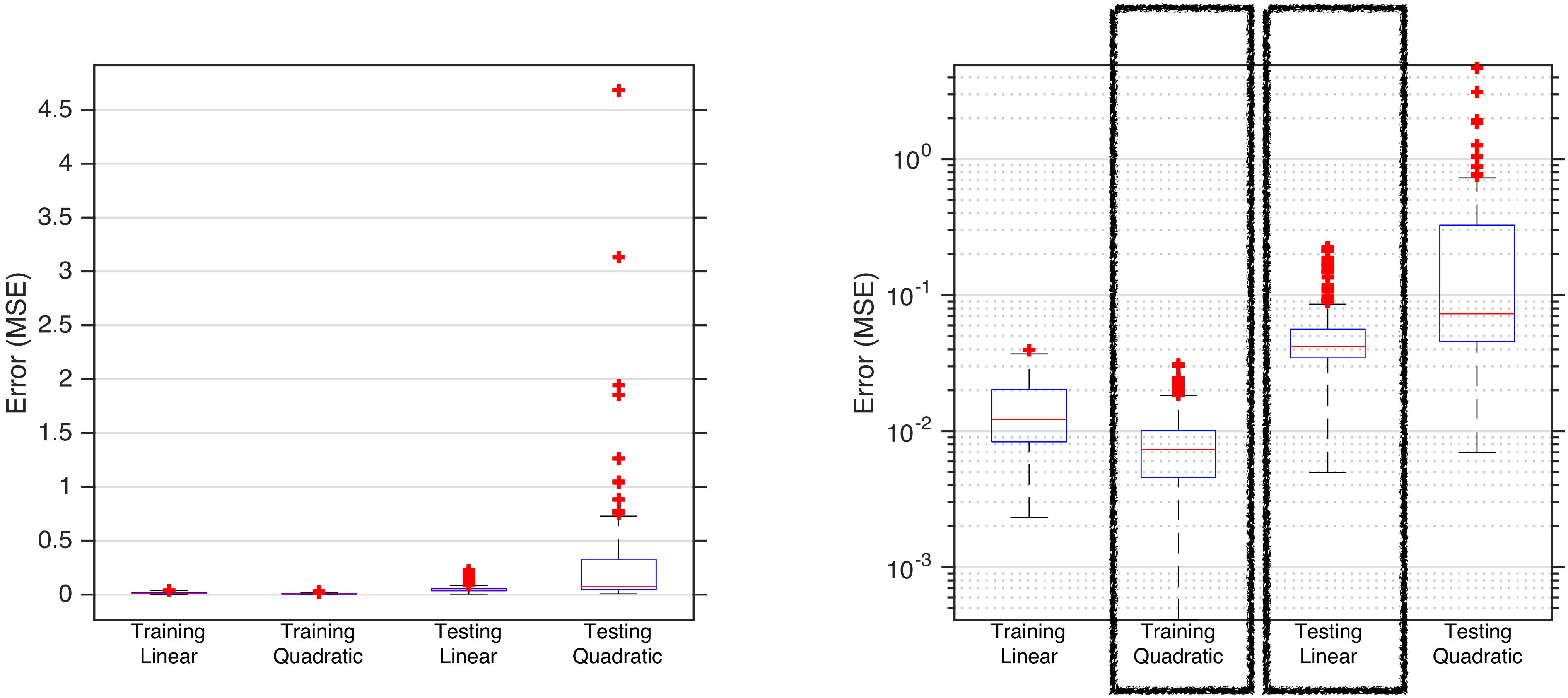
$$y = \beta_0 + \beta_1 x + \varepsilon$$

Quadratic:
y ~ 1+x+x^2

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

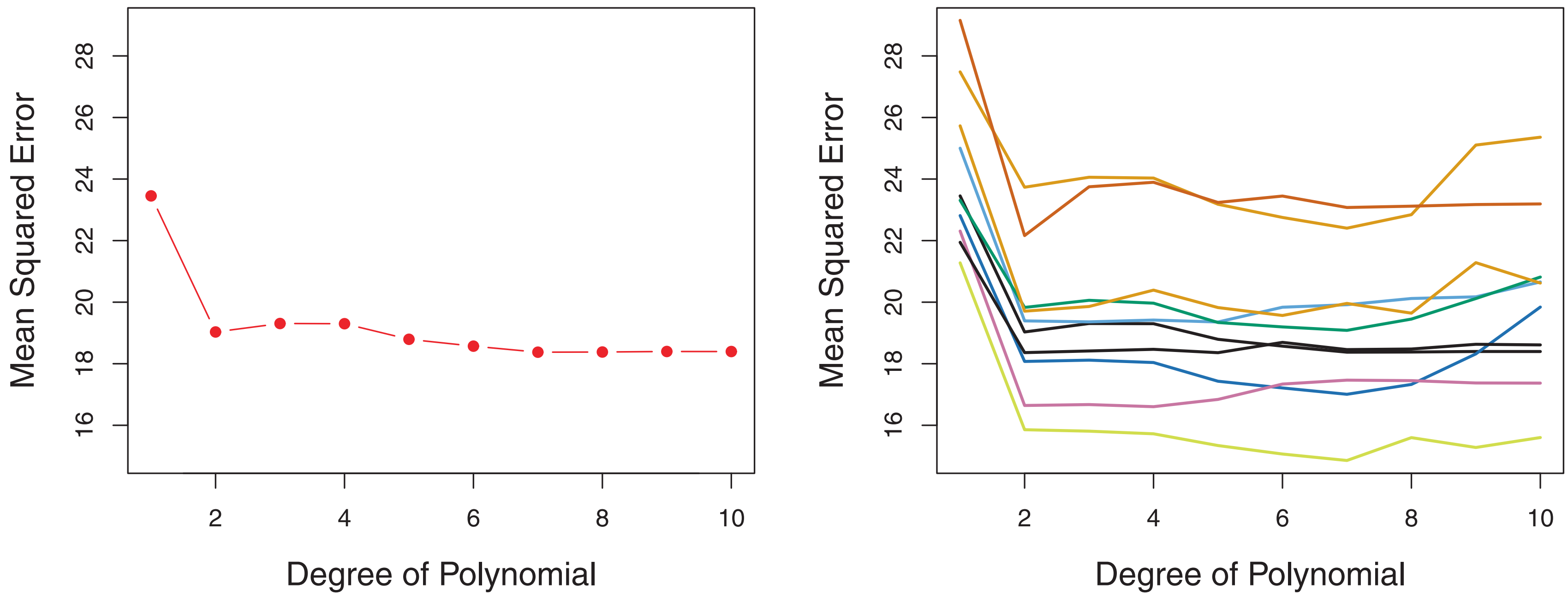


Using cross-validation we can select the model with lowest testing error



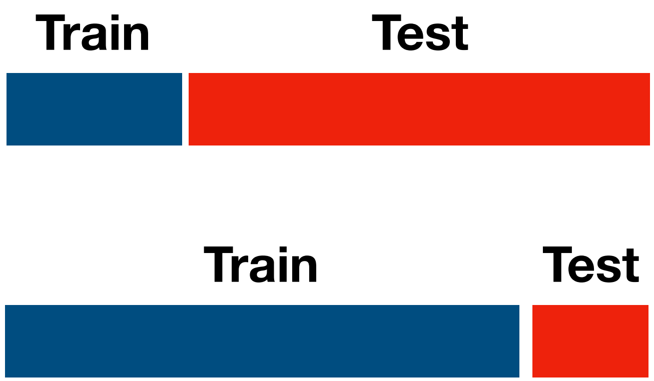
Best (lowest) training error

Best (lowest) testing error


Mean Squared Error
16
18
20
22
24
26
28
2
4
6
8
10
Degree of Polynomial
Mean Squared Error
16
18
20
22
24
26
28
2
4
6
8
10
Degree of Polynomial

How to choose the test set fraction

- The more data points we use for training, the better our parameter estimates will be
    - More precisely, more data means lower <u>variance</u>
- On the other hand, we will have fewer data points left over for testing
- Thus, our estimate of MSE_{pred} will be less accurate (higher variance)



Leave-one-out cross-validation (LOOCV)

- Choose 1 observation to use for validation/test; use the remaining (n-1) for training

- Repeat this procedure n times, using each observation as the test once

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i.$$

Advantages of LOOCV

- Since we use (n-1) observations for training, the model fit will be almost as good as possible

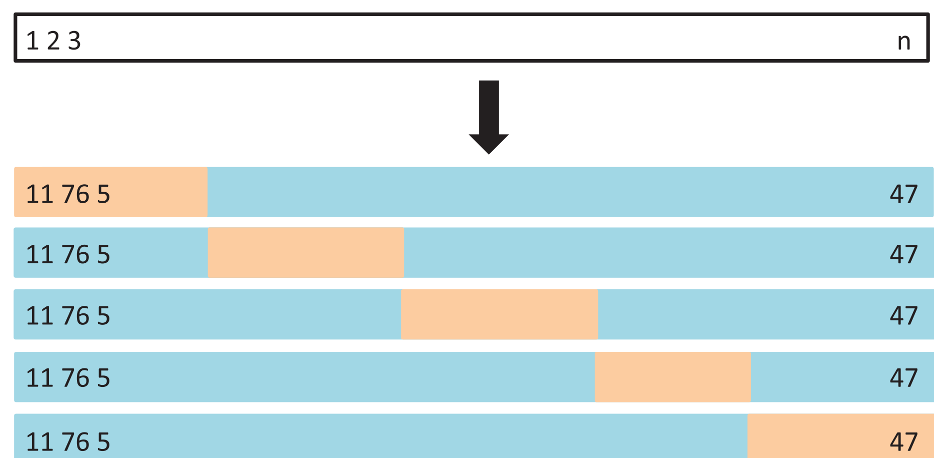
- There is no randomness; we use each observation for testing exactly once

- Disadvantage: If the dataset is large, it may require a lot of computation to fit the model n times

k-Fold cross-validation

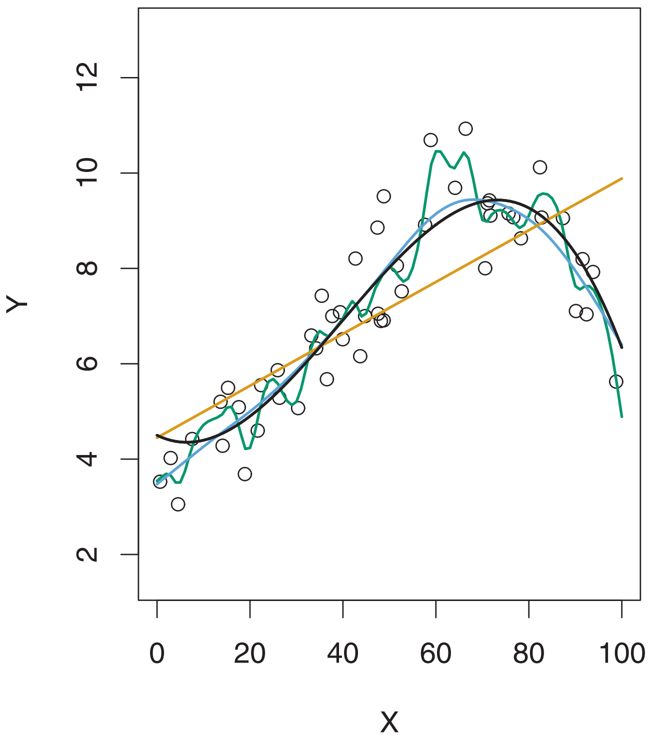
- Randomly assign each observation to one of k data "subsets" (or "folds")
- Choose one "fold" as the test set, and train using the remaining (k-1) folds
- Repeat this k times, using each fold for testing once
- Note: If k=n, then this is the same as LOOCV!

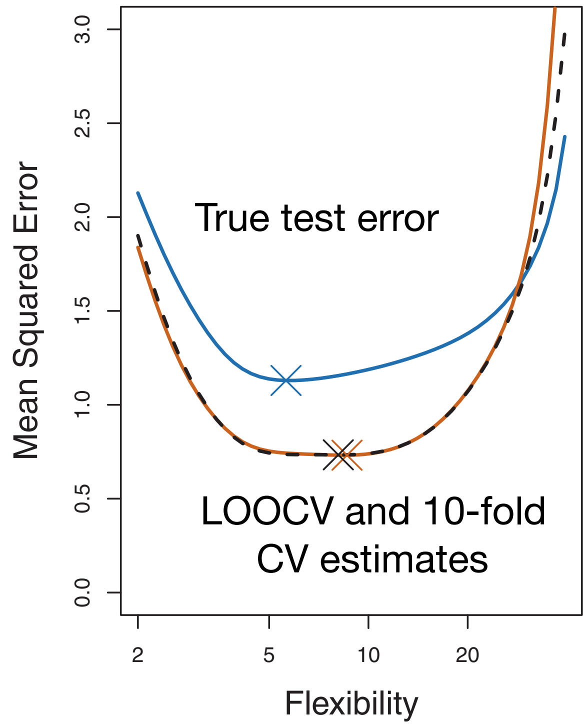
$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i.$$



k-fold CV vs. LOOCV

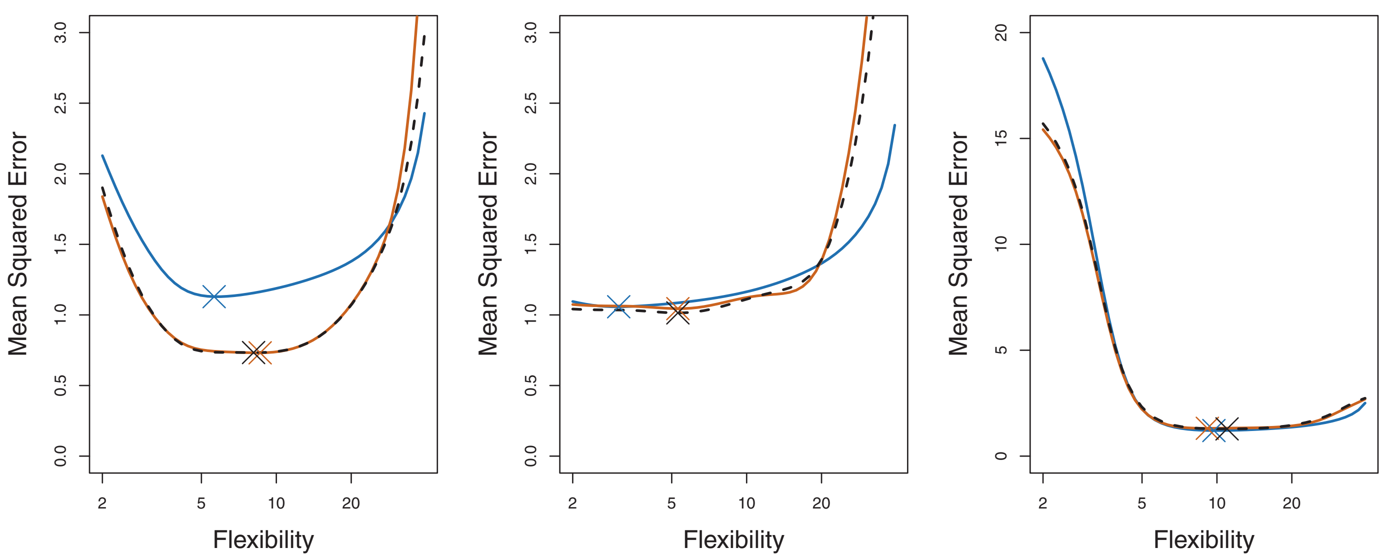
- LOOCV requires fitting n models, so it may be more computationally expensive than k-fold CV
- In addition, k-fold CV may give more accurate measures of test error than LOOCV
    - Even though LOOCV averages over n different samples, each of those samples is almost completely overlapping
- In practice, k-fold CV with k=5 or k=10 are common choices



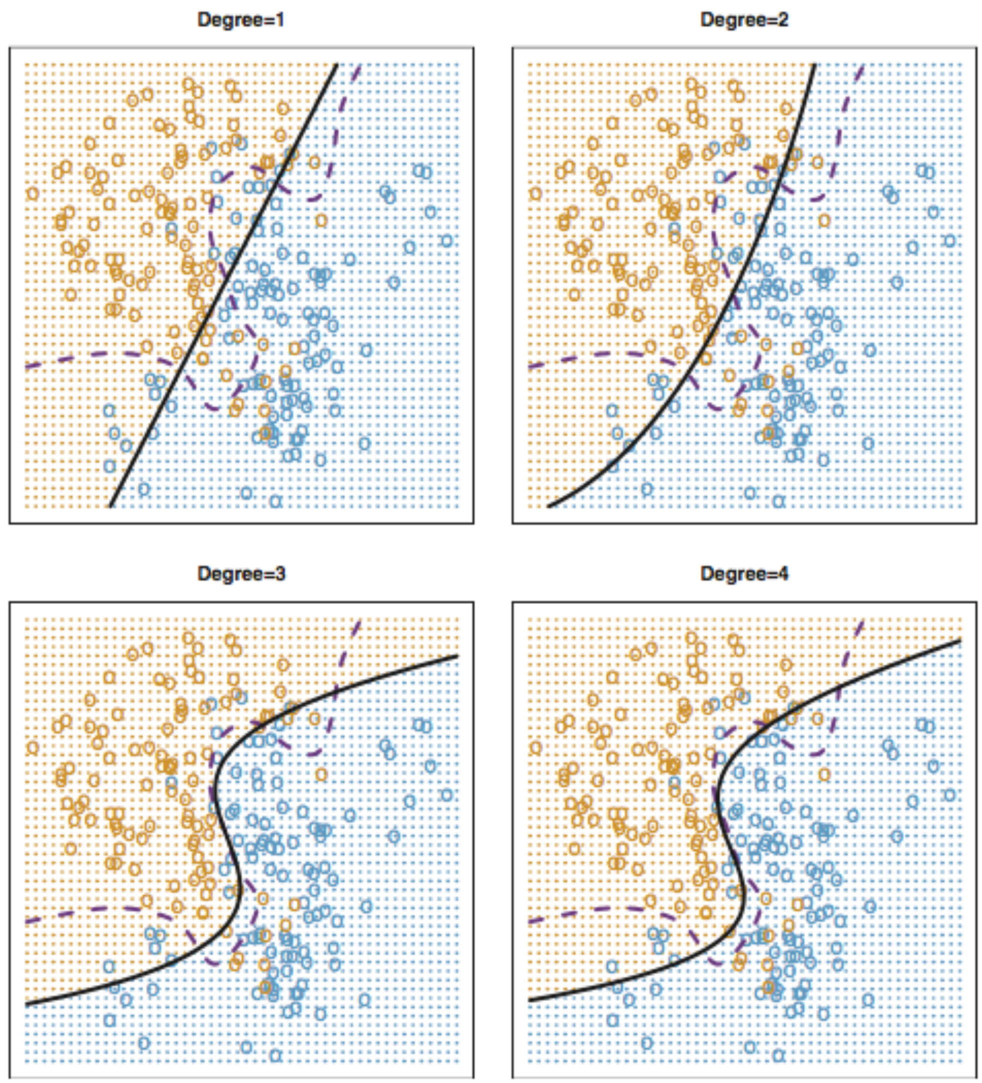


Note: the “true test error” is known in this case for simulated data.

In real data applications, the “true error” is unknown



Cross-validation for classification



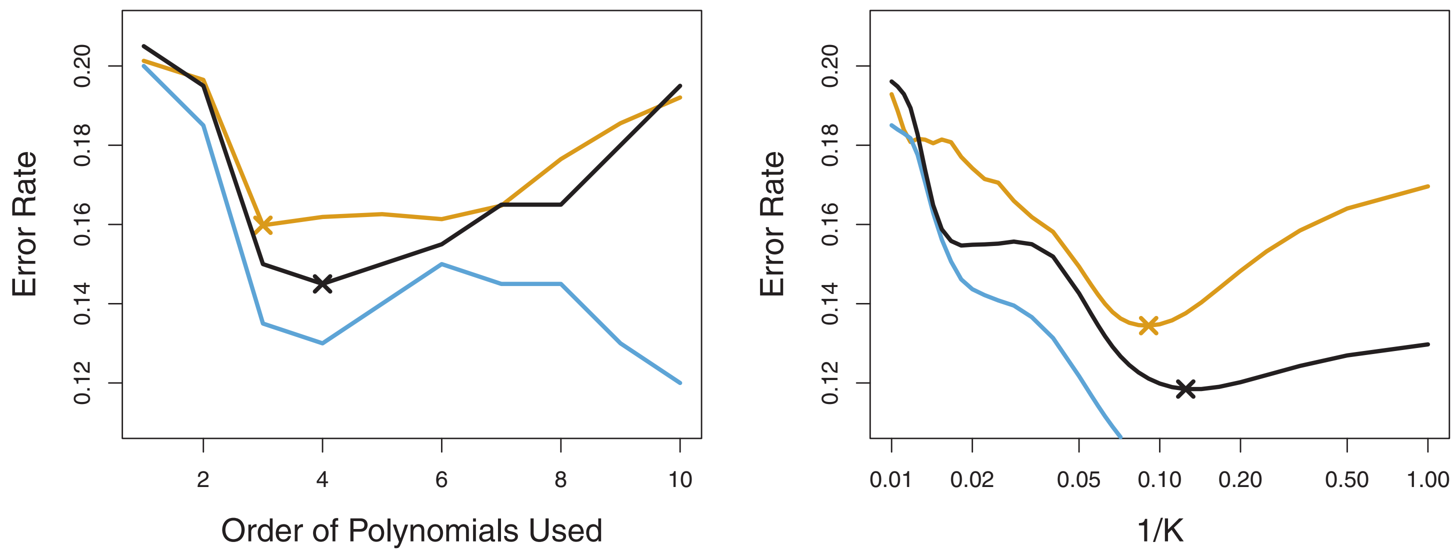


FIGURE 5.8. *Test error (brown), training error (blue), and* 10*-fold CV error (black) on the two-dimensional classification data displayed in Figure 5.7.* Left: *Logistic regression using polynomial functions of the predictors. The order of the polynomials used is displayed on the x-axis.* Right: *The KNN classifier with different values of K, the number of neighbors used in the KNN classifier.*

Bootstrap resampling

- Recall that in linear and logistic regression, we obtain estimates and SE (standard error) for each model parameter

	Coefficient	Std. error	t-statistic	p-value
Intercept	509.80	33.13	15.389	< 0.0001
gender[Female]	19.73	46.05	0.429	0.6690

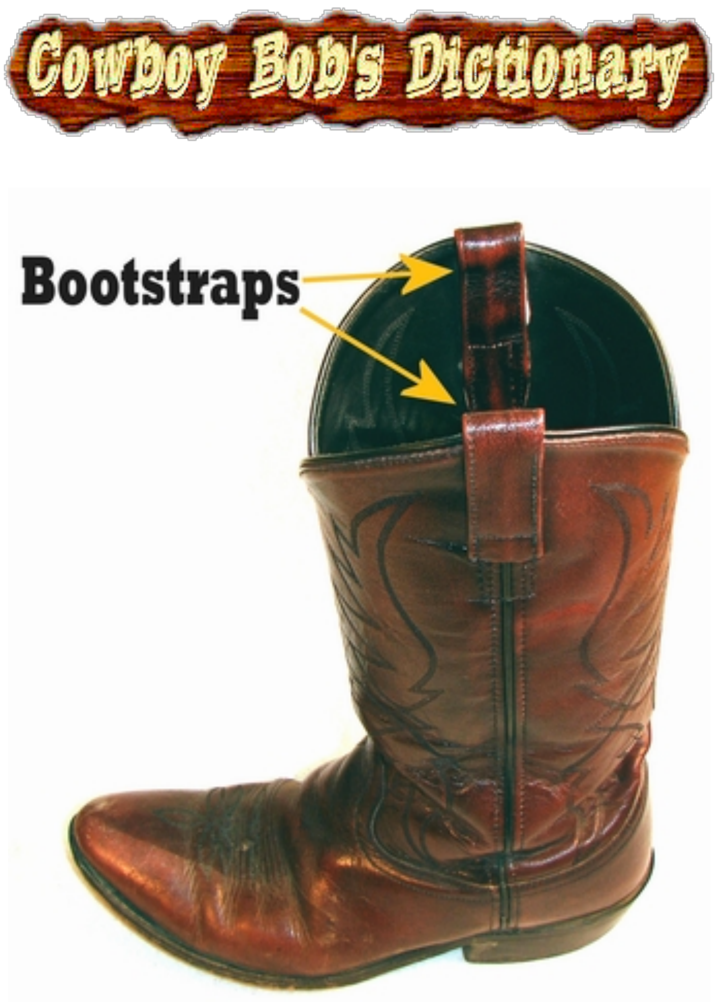
TABLE 3.7. *Least squares coefficient estimates associated with the regression of* balance *onto* gender *in the* Credit *data set. The linear model is given in (3.27). That is, gender is encoded as a dummy variable, as in (3.26).*

- The SE is calculated using assumptions, such as normality of the error.

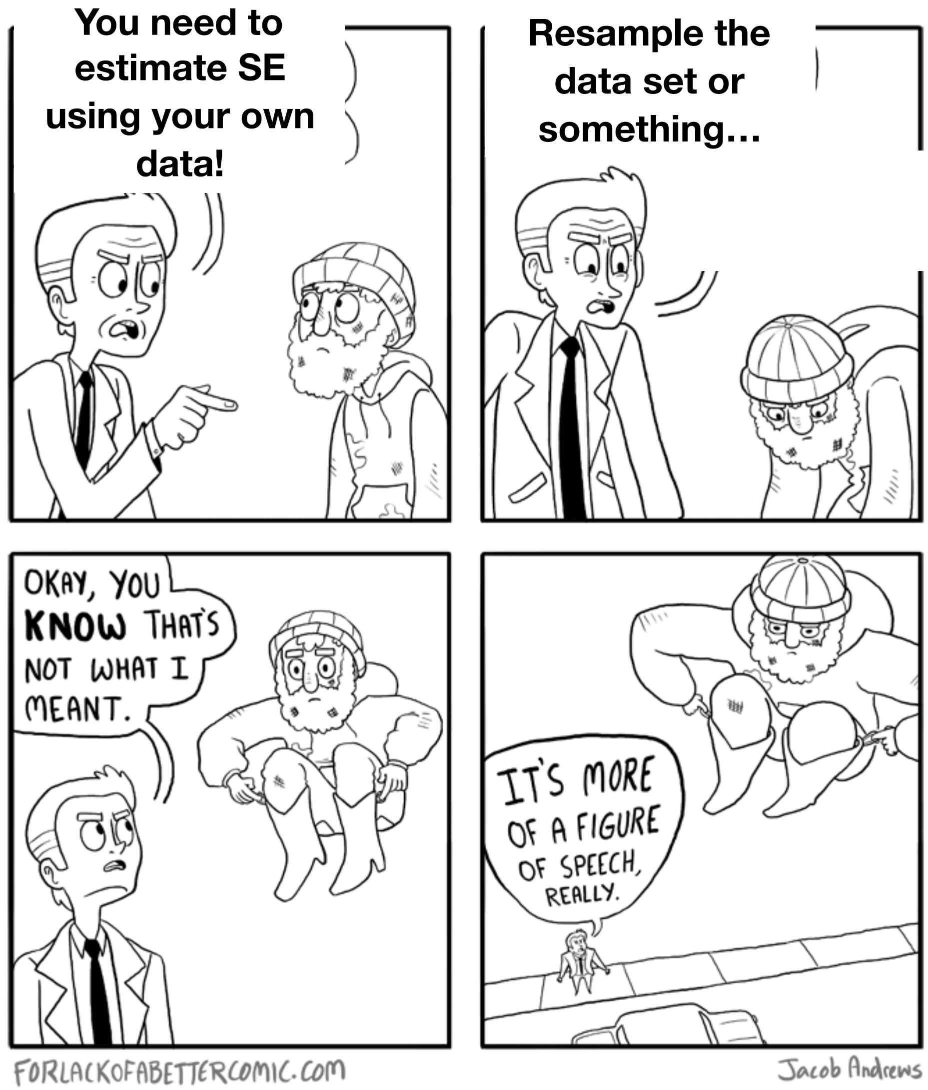
- When we have more complex, non-linear models, or when the data are strongly non-normal, it may be hard to estimate the SE

- We would like a general way of estimating the SE that makes very few assumptions and that works for any model

The bootstrap

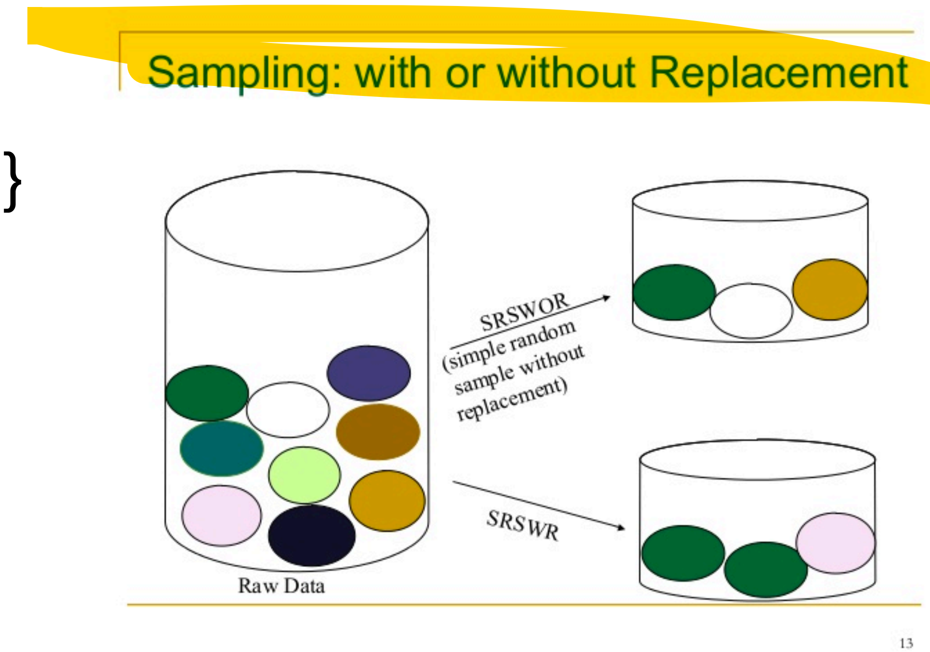


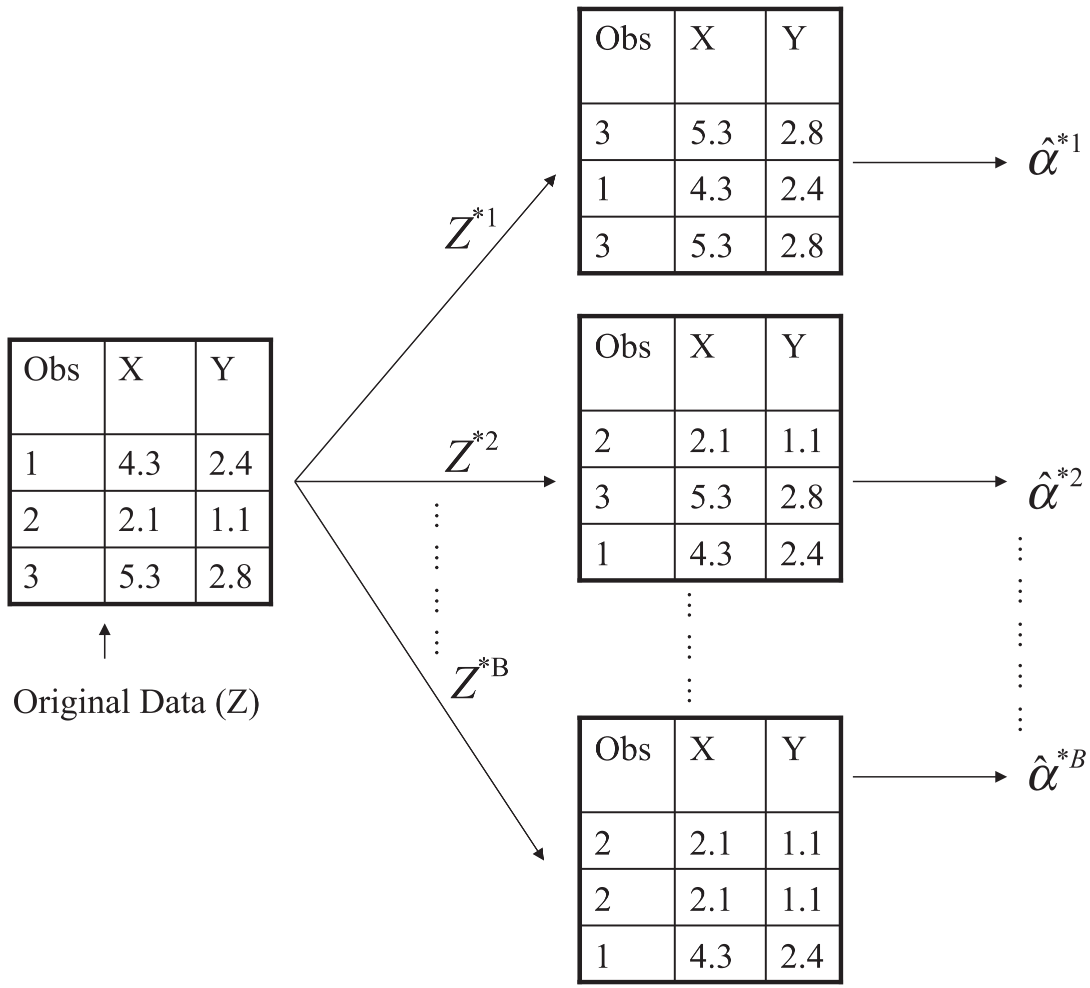
The bootstrap



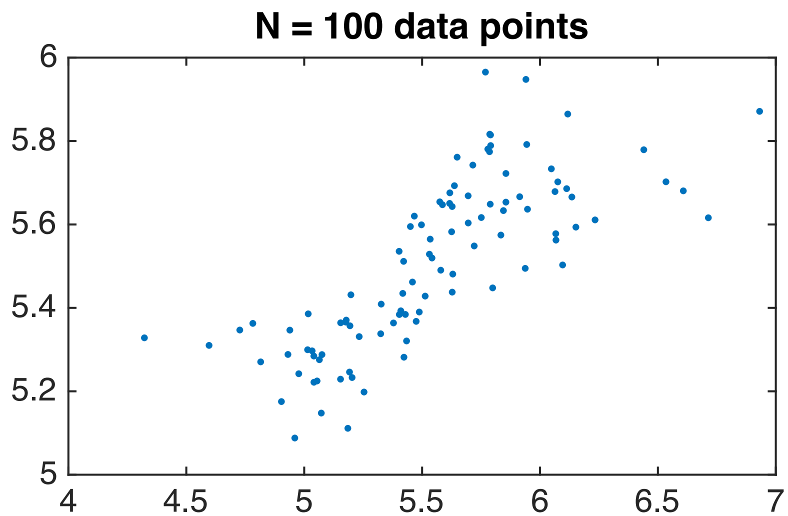
Bootstrap: Resample with replacement

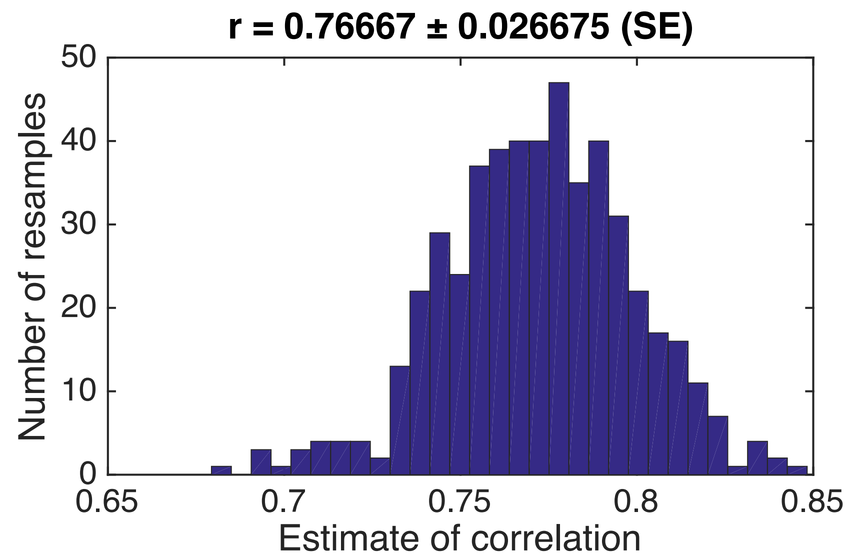
- Original data: {x1, x2, x3, x4, x5}
- Sample 1: {x1, x1, x5, x2, x5}
- Sample 2: {x1, x3, x4, x2, x4}




Obs X Y
1 4.3 2.4
2 2.1 1.1
3 5.3 2.8
Original Data (Z)
Z*1
Z*2
Z*B
Obs X Y
3 5.3 2.8
1 4.3 2.4
3 5.3 2.8
Obs X Y
2 2.1 1.1
3 5.3 2.8
1 4.3 2.4
Obs X Y
2 2.1 1.1
2 2.1 1.1
1 4.3 2.4
α̂*1
α̂*2
α̂*B

Bootstrap example:
How good is our estimate of correlation?



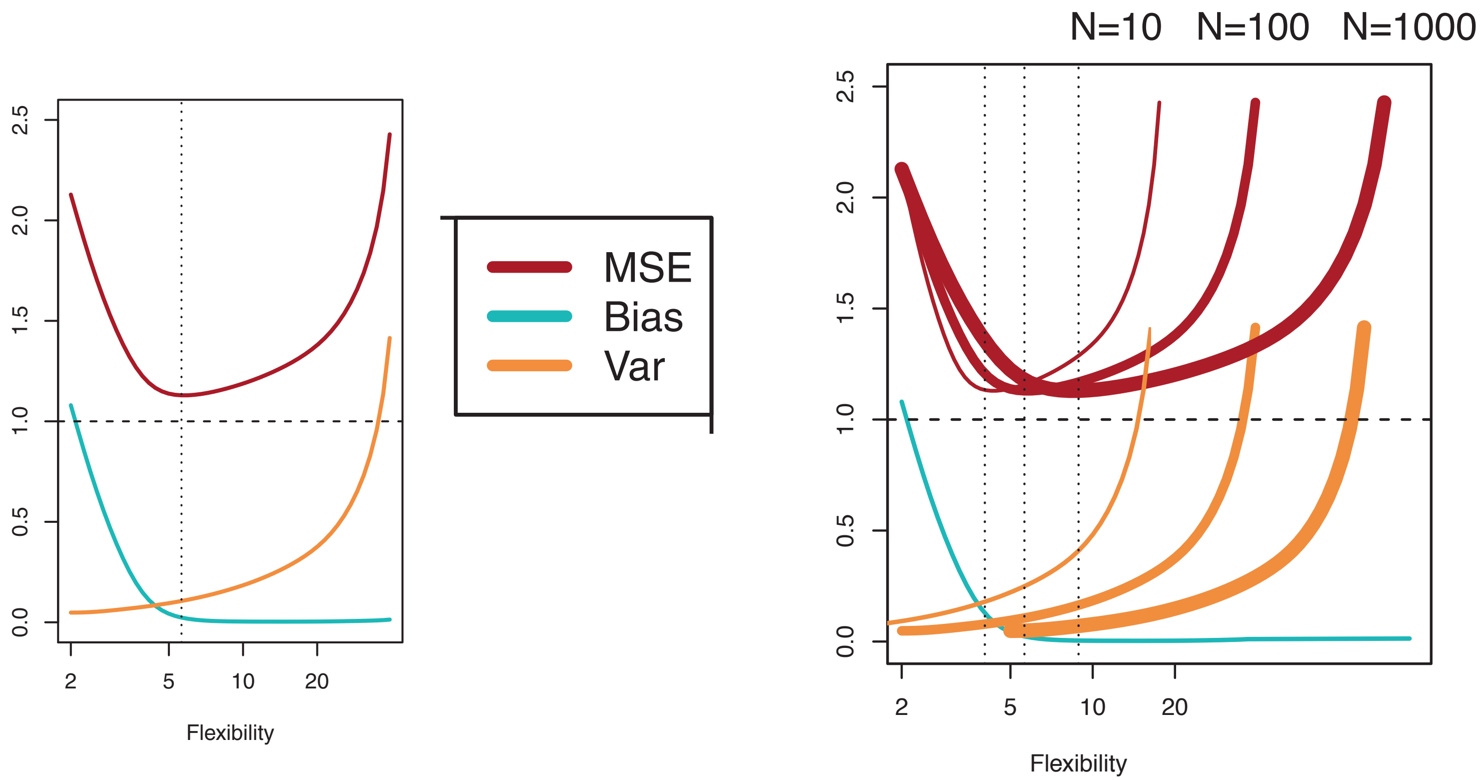



```
% Bootstrap resampling
nboot = 500;
rboot = [];
for j=1:nboot
    samples = randi(N,N,1);
    rboot(j) = corr(x(samples),y(samples));
end
```

Factors that affect model selection

- What is the true relationship, y = f(x)
    - If the relationship is complicated/non-linear, you may need a complex/flexible model
    - Then again, you can always approximate a complicated function with a simpler one (e.g. Taylor expansion)
- How much data do I have
    - If I have very few data points, I won't be able to accurately estimate the parameters of a complex model
    - Simpler models may perform better — even if they don't match the true relationship

Model variance depends on the size of (training) data

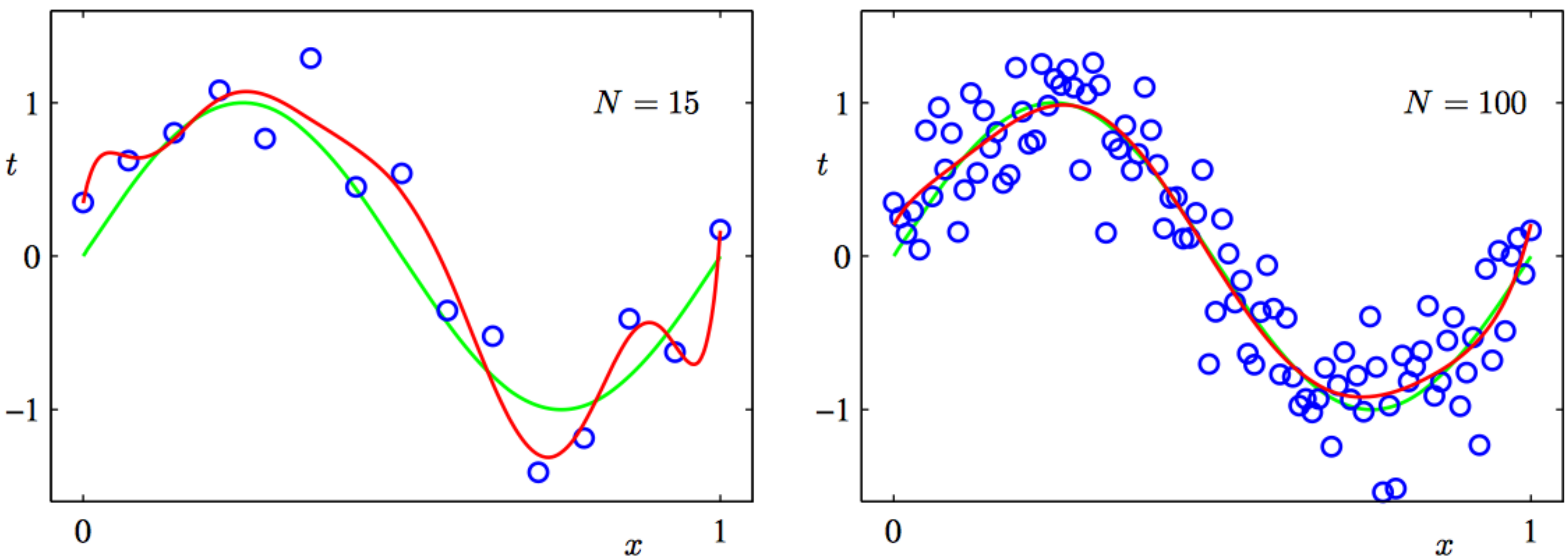


Bias measures mismatch between the true model and the best possible model fit.

Variance measures mismatch between the best possible model fit and the actual model fit (due to not enough training data!)

If you have more training data, you can reduce variance and thus increase the complexity of your model without overfitting

Example: Data set size determines the optimal model complexity



Some reminders

- Data analysis and modeling requires *both* solid statistics and practical coding skills
    - Figuring out how to import, plot, and process data is an essential part of data analysis, as important as statistics
    - You can only learn these skills through experience
    - Grappling with new statistical and programming concepts at the same time is **hard** and can be frustrating
- This is an advanced, upper-division class
- We (professor and TAs) are here to help!
    - This is our first time teaching this course, but our goal is to make the experience as efficient and rewarding as possible

How we can be (more) helpful

- We will go over coding examples related to the homework in class and section
- HW will be due after the relevant material has been covered in section
- We are available in class, section, office hours, and online (Piazza)
- Any suggestions? Please fill out the course survey (on the website)

Tips for coding assignments

- Work together! It's more fun, efficient, and educational
- Take the time to read the documentation — carefully. Every MATLAB/Python/R function has docs and examples you can try.
    - Feel free to read online resources (forums, stackexchange, etc.) — as long as you write and execute your code
- Get started early
- Take advantage of resources: TA and Professor's office hours; Section; Piazza discussion board; Ask questions in class
- As you gain experience, you will start to feel less lost — but you will never be done learning (and that's a good thing!)

perfcurve

Receiver operating characteristic (ROC) curve or other performance curve for classifier output

collapse all in page

Syntax

```
[X,Y] = perfcurve(labels,scores,posclass)
[X,Y,T] = perfcurve(labels,scores,posclass)
```

Plot ROC Curve for Classification by Logistic Regression

Load the sample data.

Open Script

```
load fisheriris
```

Use only the first two features as predictor variables. Define a binary classification problem by using only the measurements that correspond to the species versicolor and virginica.

```
pred = meas(51:end,1:2);
```

Define the binary response variable.

```
resp = (1:100)'>50;  % Versicolor = 0, virginica = 1
```

Fit a logistic regression model.

Course grades so far

HW1: Median = 87.5% [78.3% - 93.3% IQR]
HW2: Median = 107.8% [93.8% - 116% IQR]

A “model” is a whole <u>class</u> of possible predictions

- Linear, quadratic and cubic polynomials are different “models” or “model classes”

$$y = \beta_0 + \beta_1 x + \varepsilon$$ **Model 1: Linear**

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$ **Model 2: Quadratic**

- A particular “model fit” corresponds to a specific set of parameter values (and their corresponding SE, p-value etc.)

$$\{\beta_0 = 1, \beta_1 = 3\} : \ y = 1 + 3x + \varepsilon$$

$$\{\beta_0 = 0.3, \beta_1 = 5.1\} : \ y = 0.3 + 5.1x + \varepsilon$$

Predictive data modeling workflow

Data
(n observations x p predictors)

1. Model selection:
Which model will give the best predictions?

Use cross-validation to estimate Err_{test} for each model class. For this we need to separate training/testing data

Train **Test**

Linear, Quadratic, Cubic, …, 10-th order

Err_{test} Err_{test} $\boxed{Err_{test}}$ Err_{test}

2. Model fit/Parameter estimation:

After selecting the best model class, we fit the model parameters using the full data set (no cross-validation)

Best fit parameters:

$$\{\beta_0 = 0.5, \beta_1 = 3.1, \beta_2 = 0.2, \beta_3 = 0.8\}$$

3. Estimate parameter SE:
Use bootstrap resampling to determine error bars for the model parameters

$$\{\beta_0 = 0.5 \pm 0.2, \beta_1 = 3.1 \pm 1.1, \ \ldots \}$$

ROC plot concepts

		Predicted class		
		− or Null	+ or Non-null	Total
True class	− or Null	True Neg. (TN)	False Pos. (FP)	N
	+ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N*	P*	

ABLE 4.6. *Possible results when applying a classifier or diagnostic test to a pulation.*

Name	Definition	Synonyms
False Pos. rate	FP/N	Type I error, 1−Specificity
True Pos. rate	TP/P	1−Type II error, power, sensitivity, recall
Pos. Pred. value	TP/P*	Precision, 1−false discovery proportion
Neg. Pred. value	TN/N*	

TABLE 4.7. *Important measures for classification and diagnostic testing, derived from quantities in Table 4.6.*