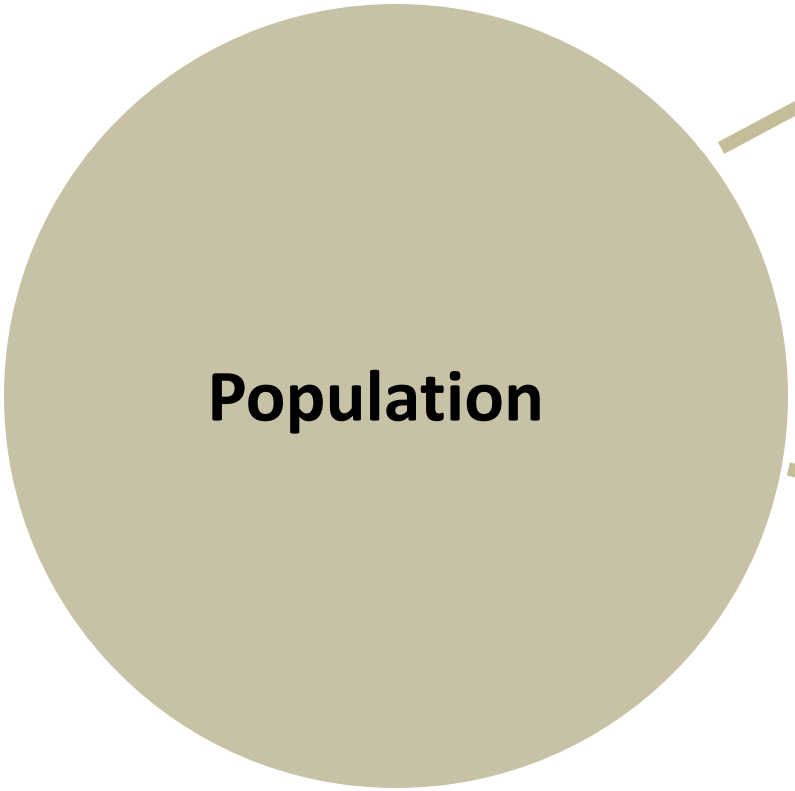
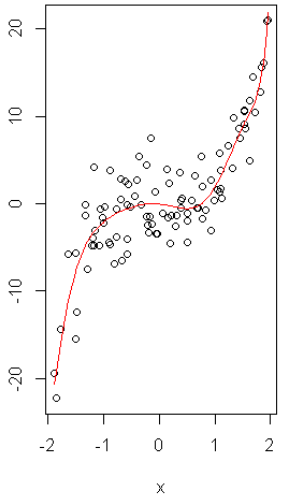
In the example last time, we compared the models by their performance on an independent test set from the same population as the data

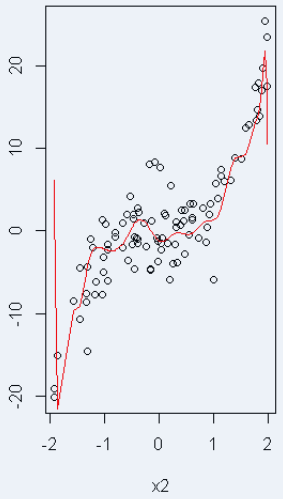


Data



Fit (aka train)

Test



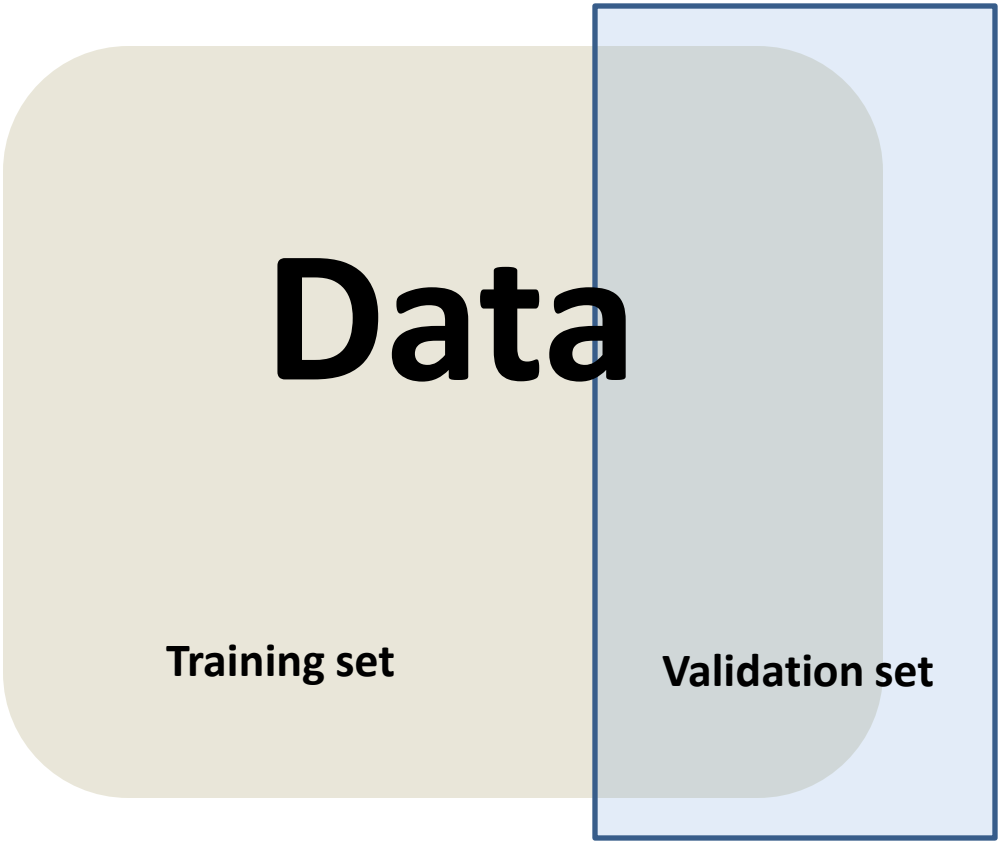
Oops!

We can’t always do this because

- We don’t know the population from which the data came, so we can’t sample from it.
- We don’t know the process which generated the data.

Idea: set aside some of the data. Fit the model to the rest. Use the set-aside data for validation.

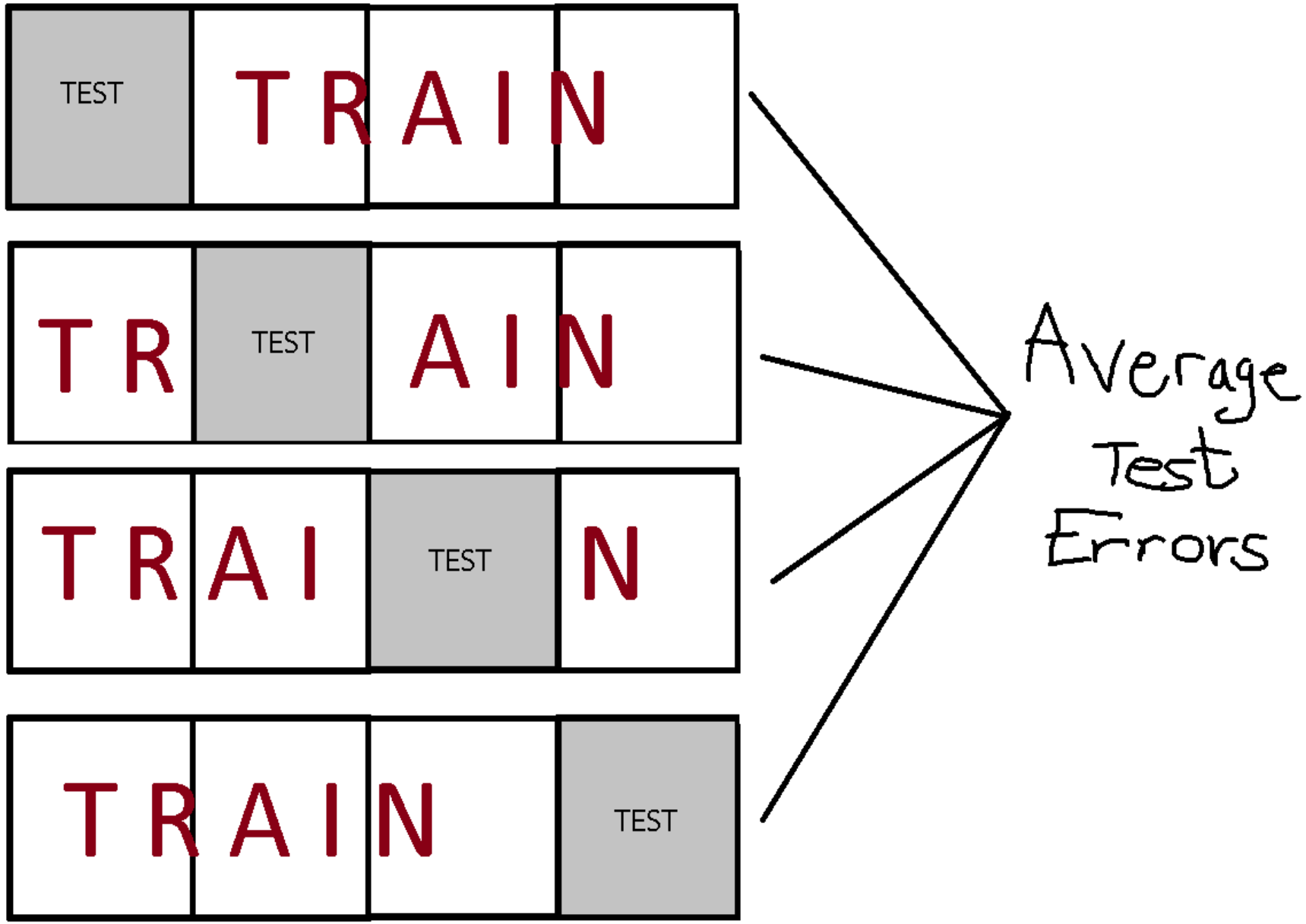
Validation set



A better way of getting an estimate of the MSE on a test set is to hold out many different test sets and then average them.

Cross-validation

is an efficient way of doing this.

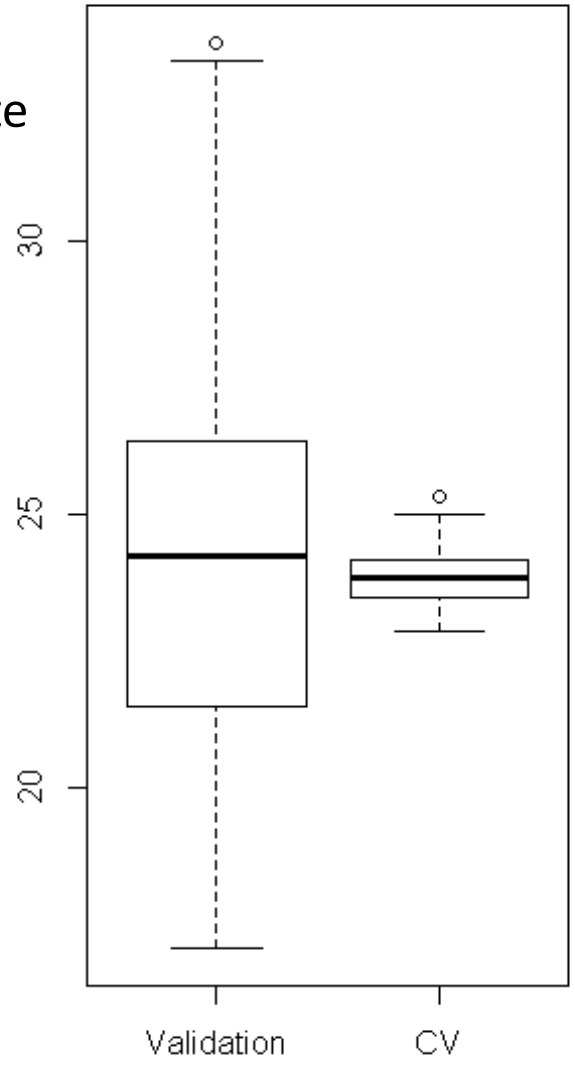

TEST
TRAIN
TR
TEST
AIN
TRAI
TEST
N
TRAIN
TEST
Average
Test
Errors

cv_k = test error on kth fold

$\overline{cv} = \frac{1}{k}\sum cv_k$ = cv estimate of test error

$\sqrt{\frac{1}{k(k-1)}\sum(cv_k - \overline{cv})^2}$ = standard error of cv estimate

One of the prices paid for more precise estimates from CV (decreased variance) is that CV estimates tend to be biased. Hastie and Tibshirani recommend 5- or 10-fold cross-validation. Witten and Frank recommend 10-fold cross-validation repeated ten times (on different random splits of the data set) with the standard error = std dev of the 10 replications.

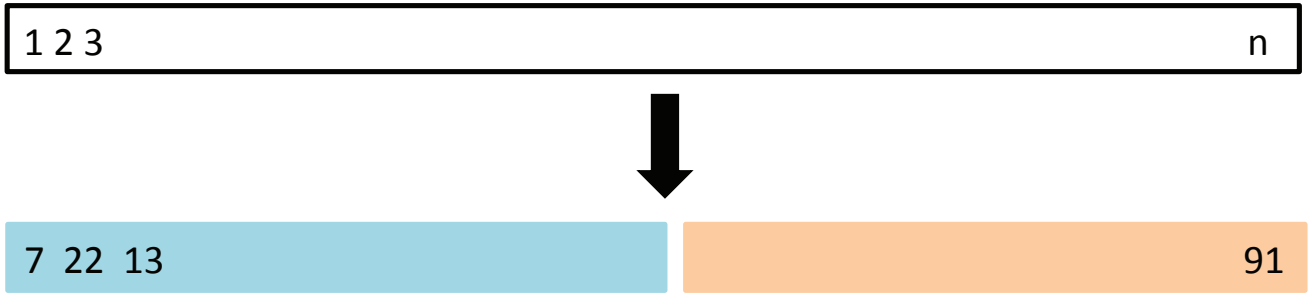


Validation set approach

Goal: Estimate the test error for a supervised learning method.

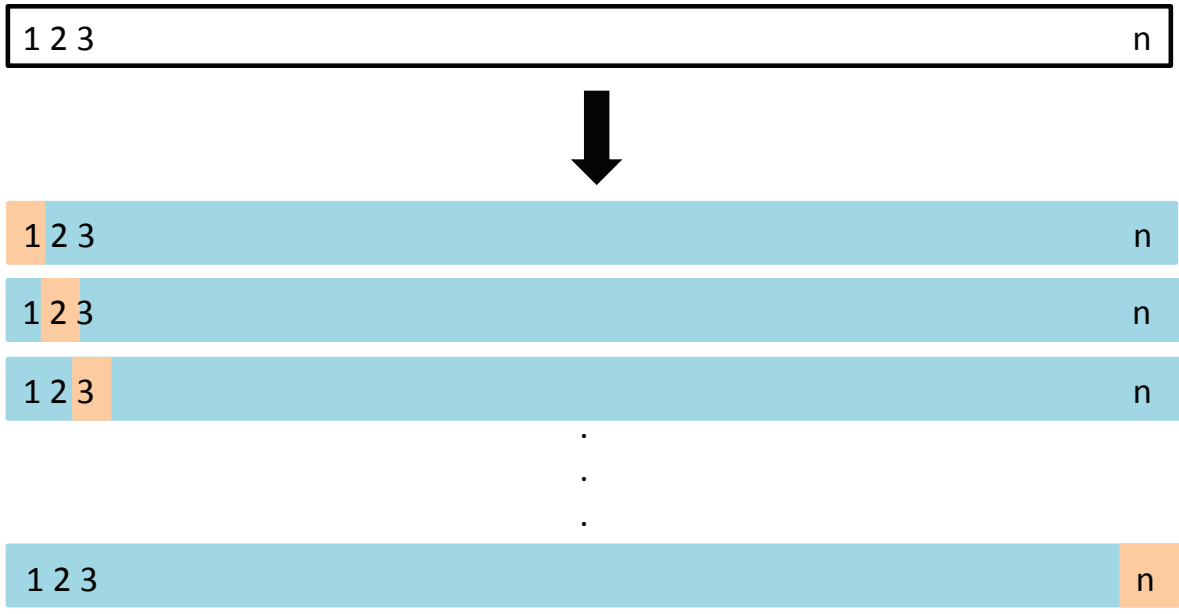
Strategy:

- Split the data in two parts.
- Train the method in the first part.
- Compute the error on the second part.



Leave one out cross-validation

- For every $i = 1, \ldots, n$:
  - train the model on every point except i,
  - compute the test error on the held out point.
- Average the test errors.



Leave one out cross-validation

- For every $i = 1, \ldots, n$:
  - train the model on every point except i,
  - compute the test error on the held out point.
- Average the test errors.

$$\mathsf{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i^{(-i)})^2$$

Prediction for the i sample without using the ith sample.

Leave one out cross-validation

- For every $i = 1, \ldots, n$:
  - train the model on every point except i,
  - compute the test error on the held out point.
- Average the test errors.

$$\mathsf{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq \hat{y}_i^{(-i)})$$

... for a classification problem.

Leave one out cross-validation

Computing $\text{CV}_{(n)}$ can be computationally expensive, since it involves fitting the model n times.

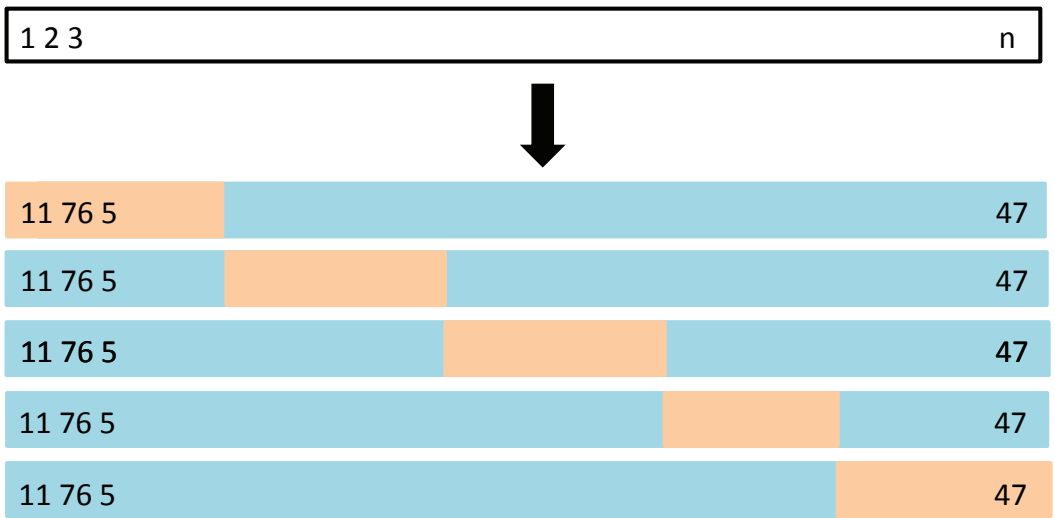
For linear regression, there is a shortcut:

$$\text{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_{ii}}\right)^2$$

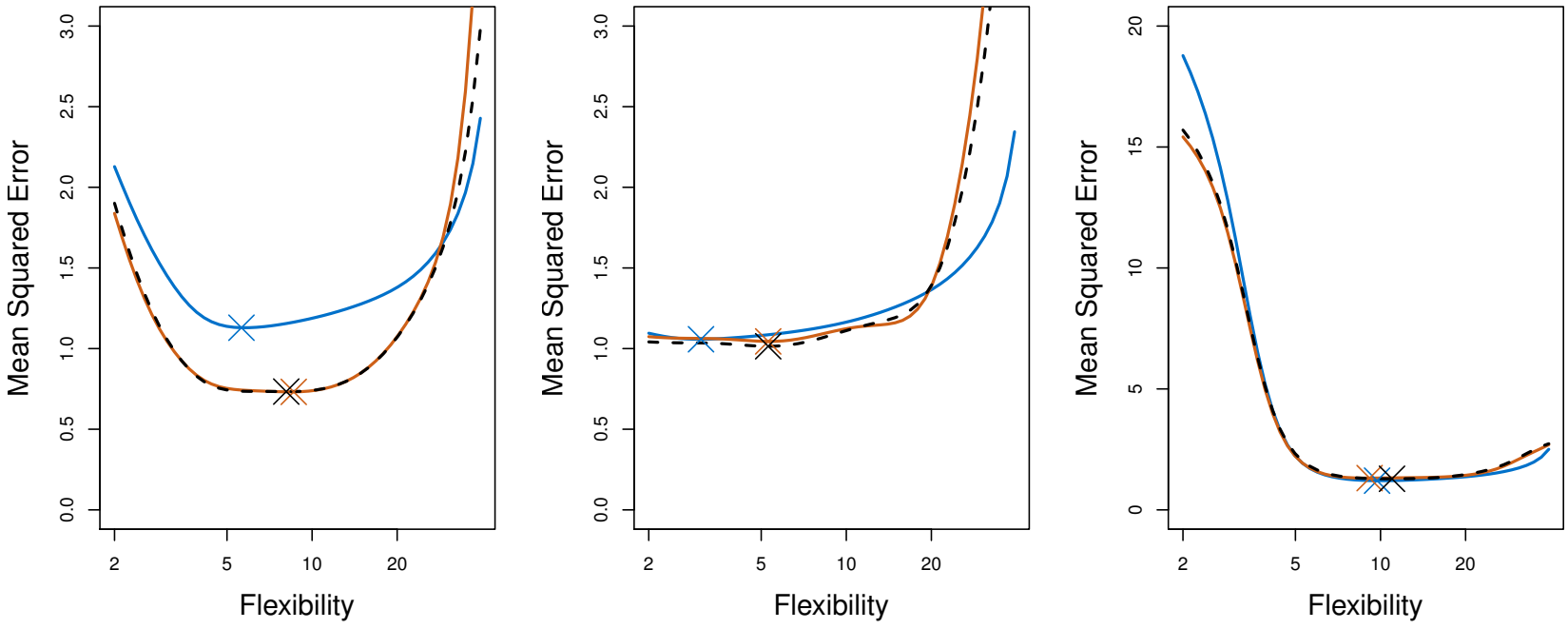
where h_{ii} is the leverage statistic.

k-fold cross-validation

- Split the data into k subsets or *folds*.
- For every $i = 1, \ldots, k$:
  - train the model on every fold except the ith fold,
  - compute the test error on the ith fold.
- Average the test errors.



Choosing an optimal model

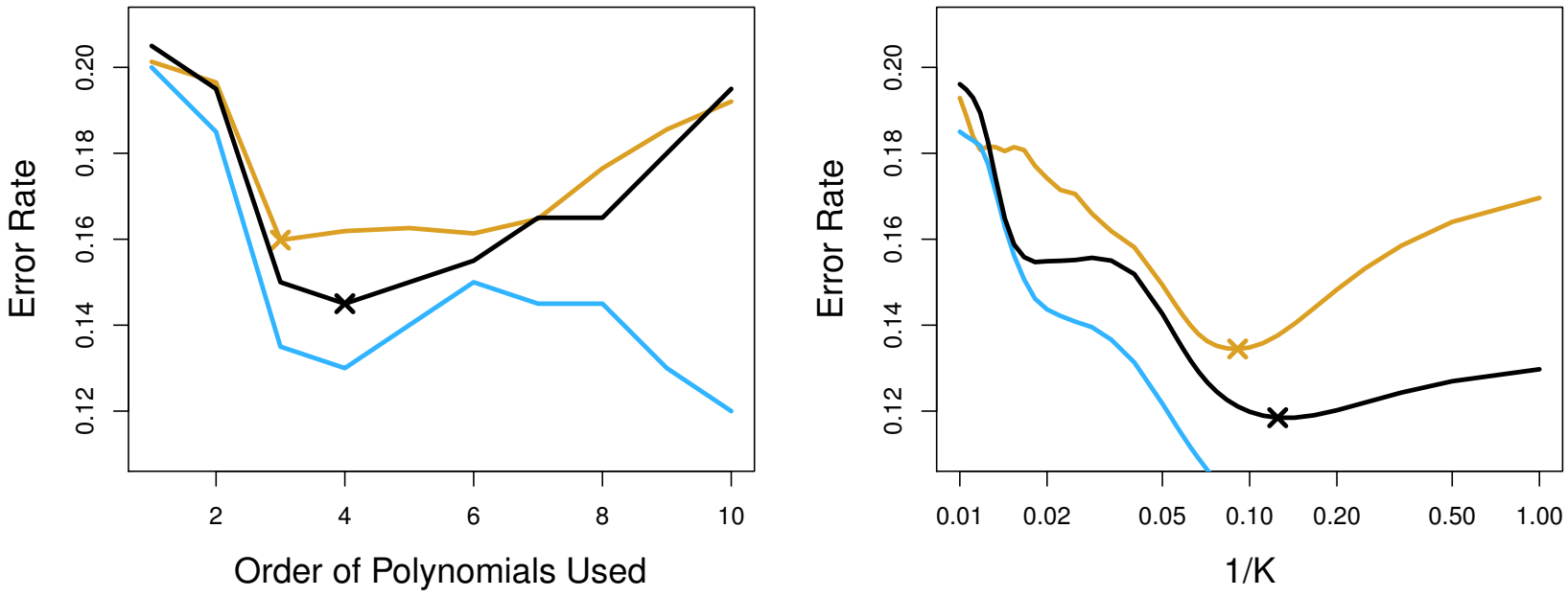


Even if the CV error estimates differ significantly from the true test error, the model with the minimum cross validation error often has relatively small test error.

Choosing an optimal model

In a classification problem, things look similar.

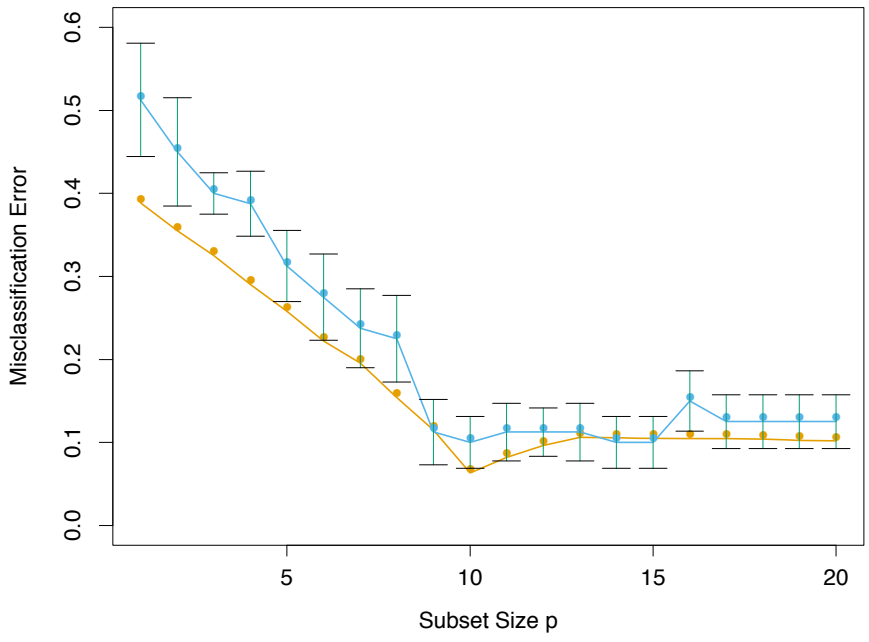
— Training error, — 10-fold CV, — Test error.



Note: Training error rate sometimes increases as logistic regression does not directly minimize 0-1 error rate but maximizes likelihood.

The one standard error rule

Forward stepwise selection



Blue: 10-fold cross validation
Yellow: True test error

- Curves minimized at $p = 10$.
- Models with $9 \leq p \leq 15$ have very similar CV error.
- The vertical bars represent 1 standard error in the test error from the 10 folds.
- **Rule of thumb:** Choose the simplest model whose CV error is no more than one standard error above the model with the lowest CV error.

Three-way data splits