Lecture 11: Cross validation

Reading: Chapter 5

STATS 202: Data mining and analysis

Jonathan Taylor, 10/17
Slide credits: Sergio Bacallado

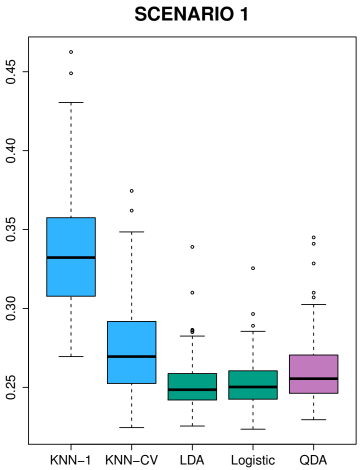
Comparing classification methods through simulation

1. Simulate data from several different known distributions with 2 predictors and a binary response variable.

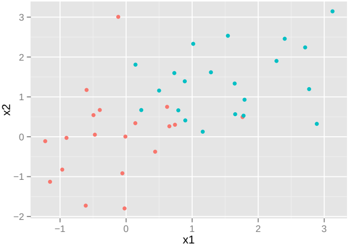
Comparing classification methods through simulation

1. Simulate data from several different known distributions with 2 predictors and a binary response variable.
2. Compare the test error (0-1 loss) for the following methods:
   - KNN-1
   - KNN-CV (“optimal” KNN)
   - Logistic regression
   - Linear discriminant analysis (LDA)
   - Quadratic discriminant analysis (QDA)

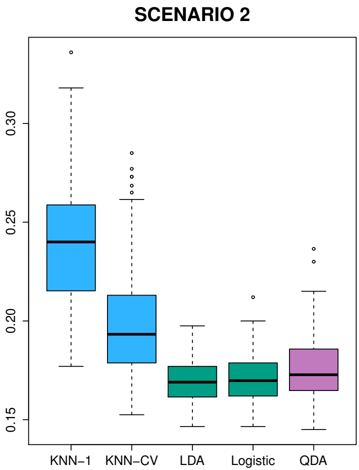
Scenario 1



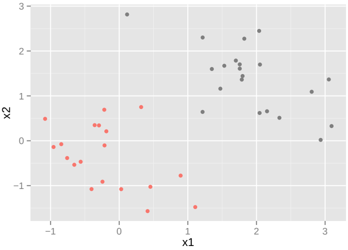
- X_1, X_2 standard normal.
- No correlation in either class.



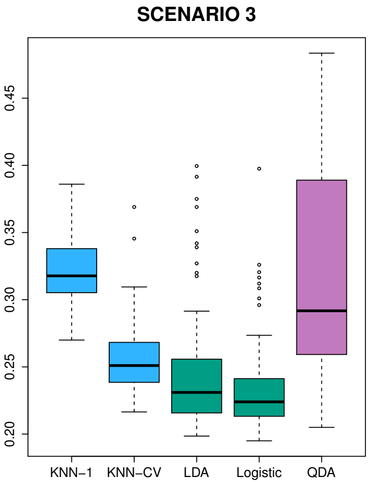
Scenario 2



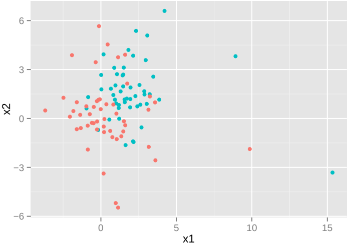
- X_1, X_2 standard normal.
- Correlation is -0.5 in both classes.



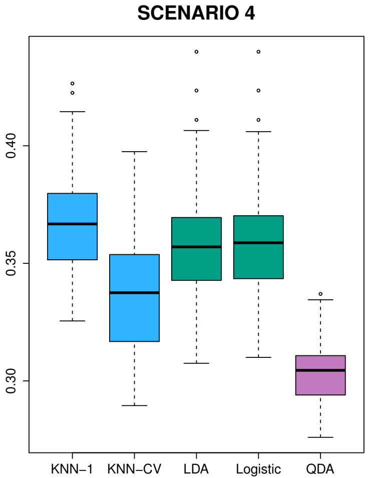
Scenario 3



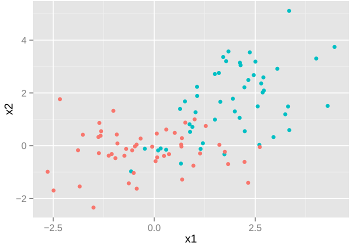
- X_1, X_2 Student t random variables.
- No correlation in either class.



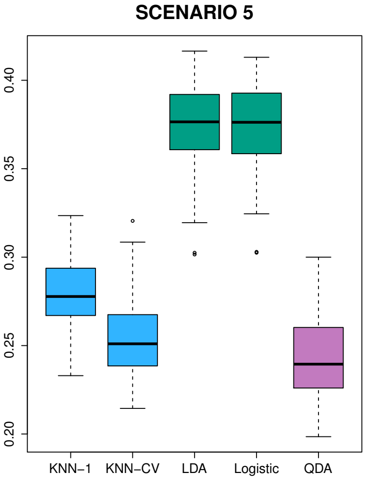
Scenario 4



- X_1, X_2 standard normal.
- First class has correlation 0.5, second class has correlation -0.5.



Scenario 5

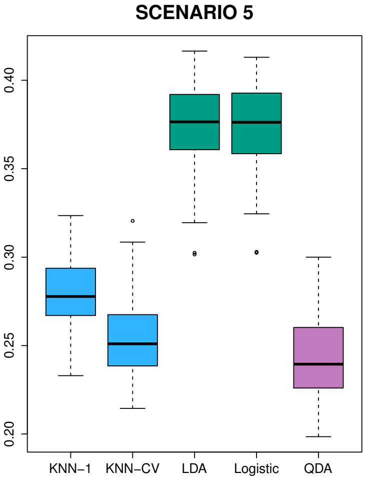


- X_1, X_2 uncorrelated, standard normal.
- Response Y was sampled from:

$$P(Y=1|X) = \frac{e^{\beta_0+\beta_1(X_1^2)+\beta_2(X_2^2)+\beta_3(X_1X_2)}}{1+e^{\beta_0+\beta_1(X_1^2)+\beta_2(X_2^2)+\beta_3(X_1X_2)}}.$$

Nonlinear model

Scenario 5

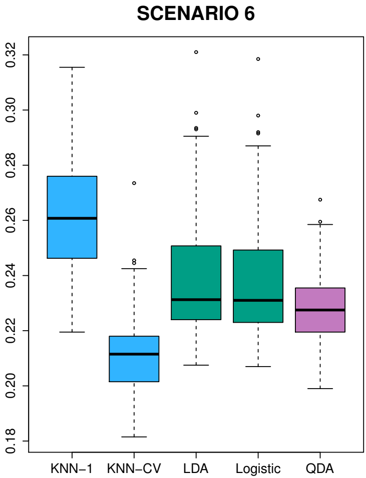


- X_1, X_2 uncorrelated, standard normal.
- Response Y was sampled from:

$$P(Y = 1|X) = \frac{e^{\beta_0+\beta_1(X_1^2)+\beta_2(X_2^2)+\beta_3(X_1X_2)}}{1 + e^{\beta_0+\beta_1(X_1^2)+\beta_2(X_2^2)+\beta_3(X_1X_2)}}.$$

- The true decision boundary is quadratic.

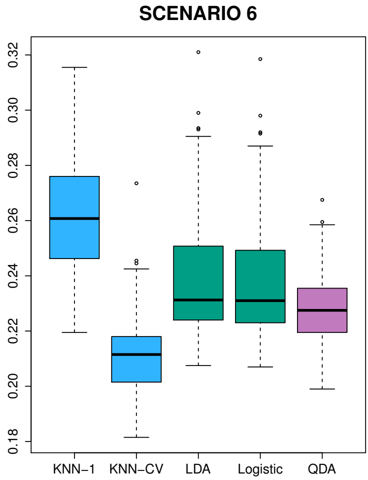
Scenario 6



- X_1, X_2 uncorrelated, standard normal.
- Response Y was sampled from:

$$P(Y = 1|X) = \frac{e^{f_{\text{nonlinear}}(X_1,X_2)}}{1 + e^{f_{\text{nonlinear}}(X_1,X_2)}}.$$

Scenario 6



- X_1, X_2 uncorrelated, standard normal.
- Response Y was sampled from:

$$P(Y = 1|X) = \frac{e^{f_{\text{nonlinear}}(X_1,X_2)}}{1 + e^{f_{\text{nonlinear}}(X_1,X_2)}}.$$

- The true decision boundary is very rough.

Thinking about the loss function is important

Most of the **regression** methods we've studied aim to minimize the RSS, while **classification** methods aim to minimize the 0-1 loss.

Thinking about the loss function is important

Most of the **regression** methods we've studied aim to minimize the RSS, while **classification** methods aim to minimize the 0-1 loss.

In classification, we often care about certain kinds of error more than others; i.e. the natural loss function is not the 0-1 loss.

Thinking about the loss function is important

Most of the **regression** methods we've studied aim to minimize the RSS, while **classification** methods aim to minimize the 0-1 loss.

In classification, we often care about certain kinds of error more than others; i.e. the natural loss function is not the 0-1 loss.

Even if we use a method which minimizes a certain kind of training error, we can *tune* it to optimize our true loss function.

Thinking about the loss function is important

Most of the **regression** methods we've studied aim to minimize the RSS, while **classification** methods aim to minimize the 0-1 loss.

In classification, we often care about certain kinds of error more than others; i.e. the natural loss function is not the 0-1 loss.

Even if we use a method which minimizes a certain kind of training error, we can *tune* it to optimize our true loss function.

- e.g. Find the threshold that brings the False negative rate below an acceptable level.

Thinking about the loss function is important

Most of the **regression** methods we've studied aim to minimize the RSS, while **classification** methods aim to minimize the 0-1 loss.

In classification, we often care about certain kinds of error more than others; i.e. the natural loss function is not the 0-1 loss.

Even if we use a method which minimizes a certain kind of training error, we can *tune* it to optimize our true loss function.

- e.g. Find the threshold that brings the False negative rate below an acceptable level.

In the Kaggle competition, what is our loss function?

Validation

Problem: Choose a supervised method that minimizes the test error.

Validation

Problem: Choose a supervised method that minimizes the test error. In addition, *tune* the parameters of each method:

- k in k-nearest neighbors.

Validation

Problem: Choose a supervised method that minimizes the test error. In addition, *tune* the parameters of each method:

- k in k-nearest neighbors.
- The number of variables to include in forward or backward selection.

Validation

Problem: Choose a supervised method that minimizes the test error. In addition, *tune* the parameters of each method:

- k in k-nearest neighbors.
- The number of variables to include in forward or backward selection.
- The order of a polynomial in polynomial regression.

Validation

Problem: Choose a supervised method that minimizes the test error. In addition, *tune* the parameters of each method:

- k in k-nearest neighbors.
- The number of variables to include in forward or backward selection.
- The order of a polynomial in polynomial regression.

Use of a **validation set** is one way to approximate the test error:

- Divide the data into two parts.
- Train each model with one part.
- Compute the error on the other.

Validation set approach

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

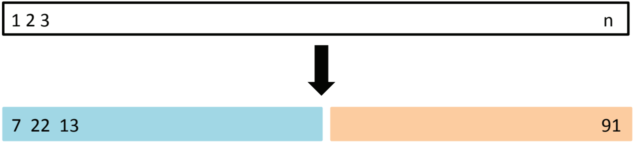
Strategy:

Validation set approach

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

Strategy:

- Split the data in two parts.

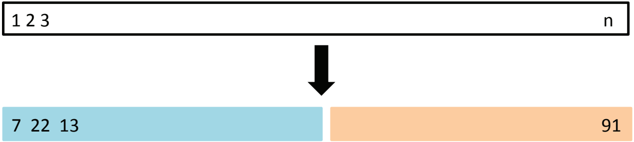


Validation set approach

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

Strategy:

- Split the data in two parts.
- Train the method in the first part.

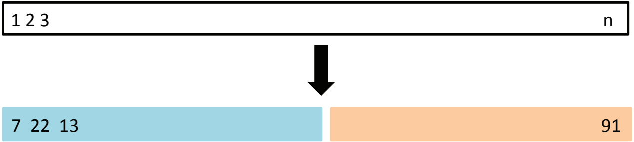


Validation set approach

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

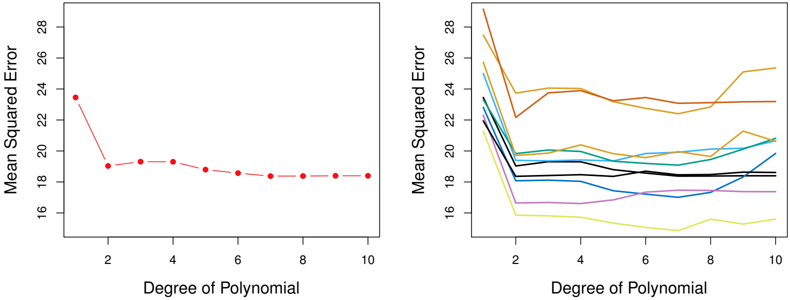
Strategy:

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



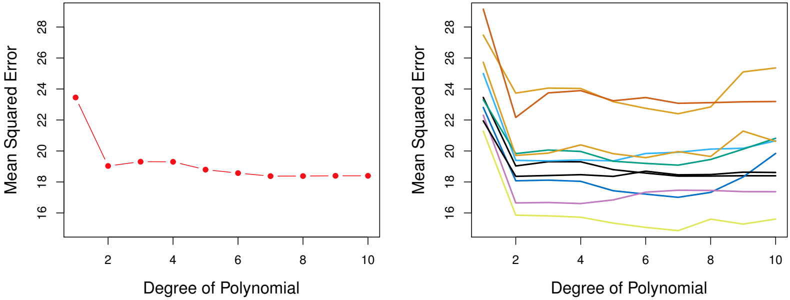
Validation set approach

Polynomial regression to estimate `mpg` from `horsepower` in the Auto data.



Validation set approach

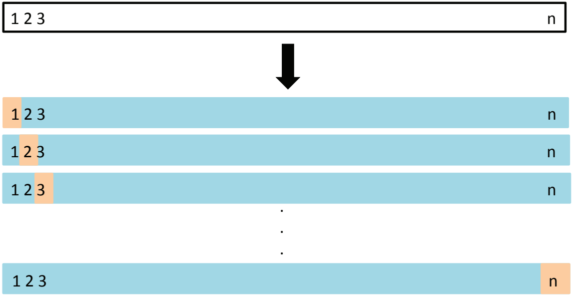
Polynomial regression to estimate `mpg` from `horsepower` in the Auto data.



Problem: Every split yields a different estimate of the error.

Leave one out cross-validation

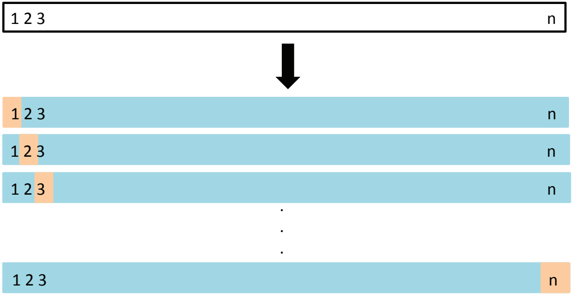
- For every $i = 1, \ldots, n$:
  - train the model on every point except i,
  - compute the test error on the held out point.



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.

偏差较小，用到了很多数据训练模型
但是方差较大，用较少的数据进行测量，过拟合，没有足够数据来接近总体的分布



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.

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

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.

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

... for a classification problem.

Leave one out cross-validation

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

Leave one out cross-validation

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

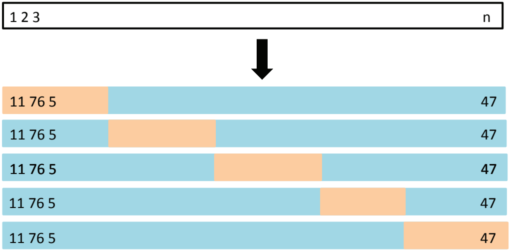
For linear regression, there is a shortcut:

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

where h_{ii} is the leverage statistic.

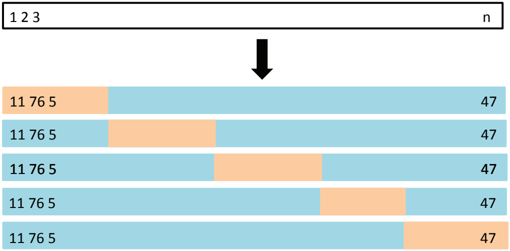
k-fold cross-validation

- Split the data into k subsets or *folds*.



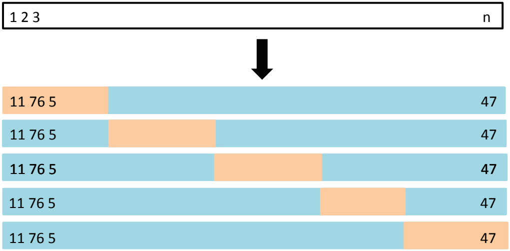
k-fold cross-validation

- 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.

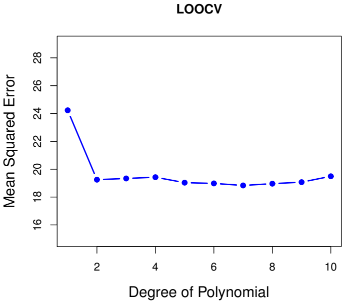


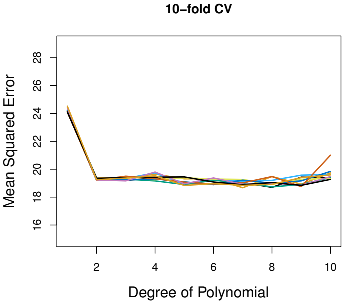
k-fold cross-validation

- 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.

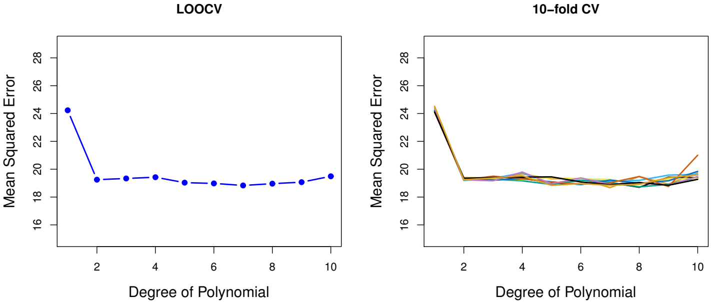


LOOCV vs. k-fold cross-validation





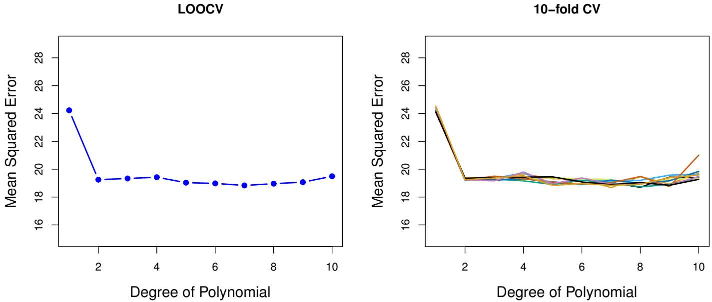
LOOCV vs. k-fold cross-validation



- k-fold CV depends on the chosen split (somewhat).

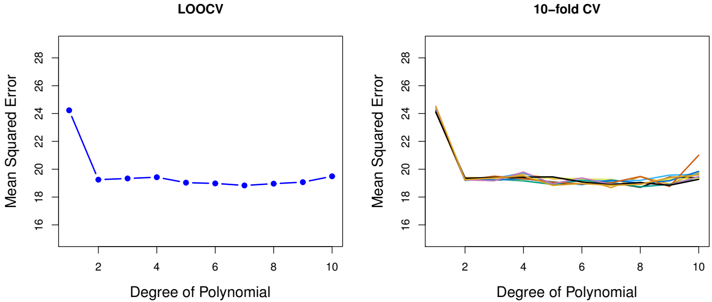
n fold无论算多少次，就只有一条线，而n-fold有很多条线

LOOCV vs. k-fold cross-validation



- k-fold CV depends on the chosen split (somewhat).
- In k-fold CV, we train the model on less data than what is available. This introduces **bias** into the estimates of test error.

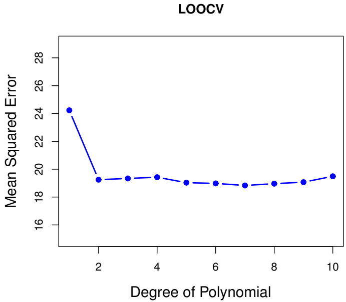
LOOCV vs. k-fold cross-validation



- k-fold CV depends on the chosen split (somewhat).
- In k-fold CV, we train the model on less data than what is available. This introduces **bias** into the estimates of test error.
- In LOOCV, the training samples highly resemble each other. This increases the **variance** of the test error estimate.

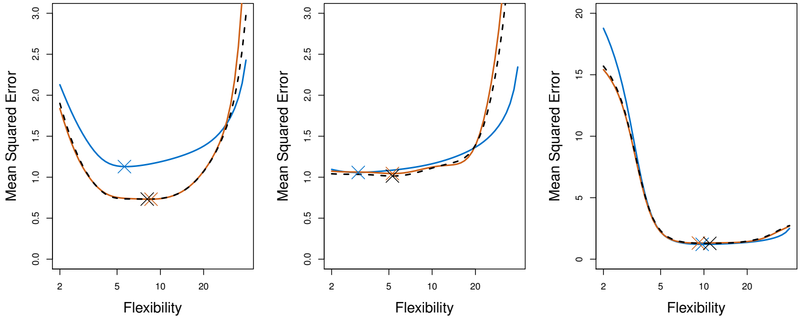
一个引进偏差，因为训练数据集的原因，而另一个增加方差变动

LOOCV vs. k-fold cross-validation



- k-fold CV depends on the chosen split (somewhat).
- In k-fold CV, we train the model on less data than what is available. This introduces **bias** into the estimates of test error.
- In LOOCV, the training samples highly resemble each other. This increases the **variance** of the test error estimate.
- n-fold CV is equivalent LOOCV.

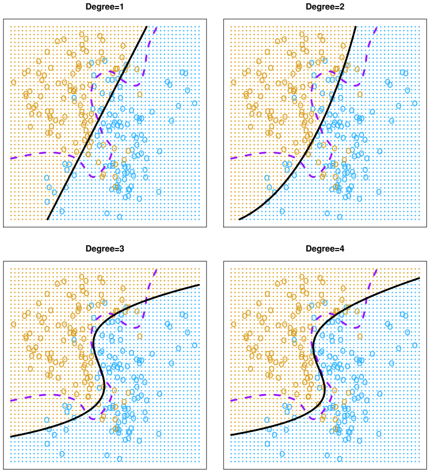
Choosing an optimal model



Even if the error estimates are off, choosing the model with the minimum cross validation error often leads to a method with near minimum test error.

Choosing an optimal model

In a classification problem, things look similar.

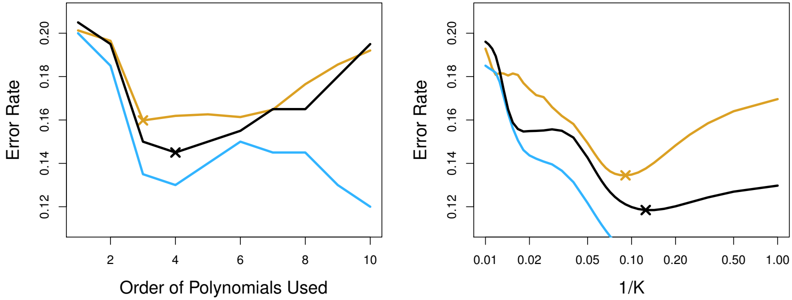


- - - Bayes boundary

—— Logistic regression with polynomial predictors of increasing degree.

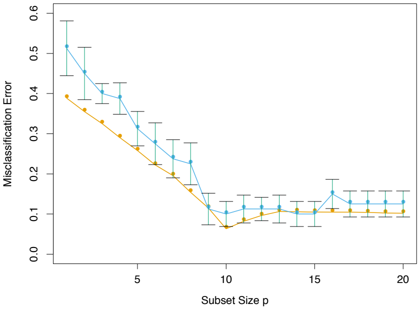
Choosing an optimal model

In a classification problem, things look similar.



The one standard error rule

Forward stepwise selection

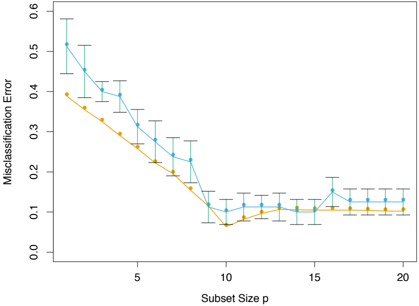


Blue: 10-fold cross validation

Yellow: True test error

The one standard error rule

Forward stepwise selection

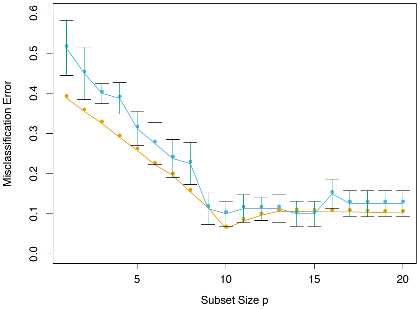


Blue: 10-fold cross validation
Yellow: True test error

- A number of models with $10 \leq p \leq 15$ have almost the same CV error.

The one standard error rule

Forward stepwise selection

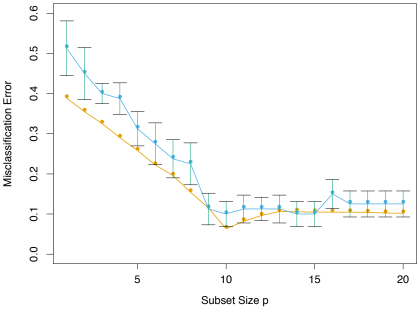


Blue: 10-fold cross validation
Yellow: True test error

- A number of models with $10 \leq p \leq 15$ have almost the same CV error.
- The vertical bars represent 1 standard error in the test error from the 10 folds.

The one standard error rule

Forward stepwise selection



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

- A number of models with $10 \leq p \leq 15$ have almost the same 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.

The wrong way to do cross validation

Reading: Section 7.10.2 of The Elements of Statistical Learning.

We want to classify 200 individuals according to whether they have cancer or not.

The wrong way to do cross validation

Reading: Section 7.10.2 of The Elements of Statistical Learning.

We want to classify 200 individuals according to whether they have cancer or not. We use logistic regression onto 1000 measurements of gene expression.

Proposed strategy:

The wrong way to do cross validation

Reading: Section 7.10.2 of The Elements of Statistical Learning.

We want to classify 200 individuals according to whether they have cancer or not. We use logistic regression onto 1000 measurements of gene expression.

Proposed strategy:

- Using all the data, select the 20 most significant genes using z-tests.

The wrong way to do cross validation

Reading: Section 7.10.2 of The Elements of Statistical Learning.

We want to classify 200 individuals according to whether they have cancer or not. We use logistic regression onto 1000 measurements of gene expression.

Proposed strategy:

- Using all the data, select the 20 most significant genes using z-tests.
- Estimate the test error of logistic regression with these 20 predictors via 10-fold cross validation.

The wrong way to do cross validation

To see how that works, let's use the following simulated data:

The wrong way to do cross validation

To see how that works, let's use the following simulated data:

- Each gene expression is standard normal and independent of all others.

The wrong way to do cross validation

To see how that works, let's use the following simulated data:

- Each gene expression is standard normal and independent of all others.
- The response (cancer or not) is sampled from a coin flip — no correlation to any of the "genes".

The wrong way to do cross validation

To see how that works, let's use the following simulated data:

- Each gene expression is standard normal and independent of all others.
- The response (cancer or not) is sampled from a coin flip — no correlation to any of the "genes".

What should the misclassification rate be for any classification method using these predictors?

The wrong way to do cross validation

To see how that works, let's use the following simulated data:

- Each gene expression is standard normal and independent of all others.
- The response (cancer or not) is sampled from a coin flip — no correlation to any of the "genes".

What should the misclassification rate be for any classification method using these predictors?

Roughly 50%.

The wrong way to do cross validation

We run this simulation, and obtain a CV error rate of 3%!

Why is this?

The wrong way to do cross validation

We run this simulation, and obtain a CV error rate of 3%!

Why is this?

- Since we only have 200 individuals in total, among 1000 variables, at least some will be correlated with the response.

The wrong way to do cross validation

We run this simulation, and obtain a CV error rate of 3%!

Why is this?

- Since we only have 200 individuals in total, among 1000 variables, at least some will be correlated with the response.
- We do variable selection using *all the data*, so the variables we select have some correlation with the response in every subset or fold in the cross validation.

The **right** way to do cross validation

- Divide the data into 10 folds.

The **right** way to do cross validation

- Divide the data into 10 folds.
- For $i = 1, \ldots, 10$:
  - Using every fold except i, perform the variable selection and fit the model with the selected variables.
  - Compute the error on fold i.

The **right** way to do cross validation

- Divide the data into 10 folds.
- For $i = 1, \ldots, 10$:
  - Using every fold except i, perform the variable selection and fit the model with the selected variables.
  - Compute the error on fold i.
- Average the 10 test errors obtained.

The **right** way to do cross validation

- Divide the data into 10 folds.
- For $i = 1, \ldots, 10$:
  - Using every fold except i, perform the variable selection and fit the model with the selected variables.
  - Compute the error on fold i.
- Average the 10 test errors obtained.

In our simulation, this produces an error estimate of close to 50%.

The **right** way to do cross validation

- Divide the data into 10 folds.
- For $i = 1, \ldots, 10$:
  - Using every fold except i, perform the variable selection and fit the model with the selected variables.
  - Compute the error on fold i.
- Average the 10 test errors obtained.

In our simulation, this produces an error estimate of close to 50%.

Moral of the story: Every aspect of the learning method that involves using the data — variable selection, for example — must be cross-validated.