Lecture 5: Evaluation and Training

STATS 202: Data mining and analysis

Rajan Patel

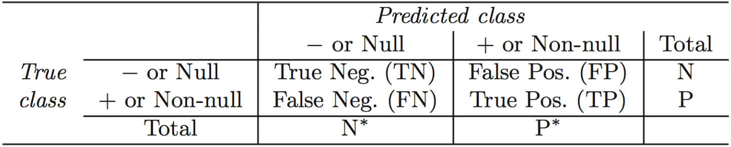
Evaluating a classification method

We have talked about the 0-1 loss:

$$\frac{1}{m}\sum_{i=1}^{m}\mathbf{1}(y_i \neq \hat{y}_i).$$

It is possible to make the wrong prediction for some classes more often than others. The 0-1 loss doesn't tell you anything about this.

A much more informative summary of the error is a **confusion matrix**:

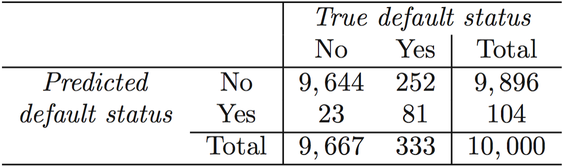


		Predicted class		
		− or Null	+ or Non-null	Total
True	− or Null	True Neg. (TN)	False Pos. (FP)	N
class	+ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N*	P*	

Example. Predicting `default`

Use a classifier to predict credit card default in a dataset of 10K people.

Predicted "yes" if $P(\texttt{default} = \text{yes}|X) > 0.5$.



		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

Example. Predicting `default`

Use a classifier to predict credit card default in a dataset of 10K people.

Predicted "yes" if $P(\texttt{default} = \text{yes}|X) > 0.5$.

		True default status		
		No	Yes	Total
Predicted	No	9,644	252	9,896
default status	Yes	23	81	104
	Total	9,667	333	10,000

- The error rate among people who do **not** default (false positive rate) is very low.

Example. Predicting `default`

Use a classifier to predict credit card default in a dataset of 10K people.

Predicted “yes” if $P(\texttt{default} = \text{yes}|X) > 0.5$.

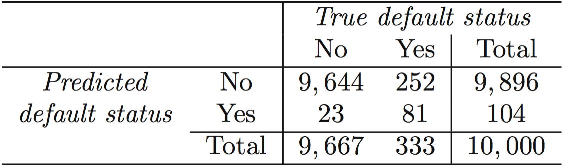
		True default status		
		No	Yes	Total
Predicted default status	No	$9,644$	252	$9,896$
	Yes	23	81	104
	Total	$9,667$	333	$10,000$

- The error rate among people who do **not** default (false positive rate) is very low.
- However, the rate of false negatives is 76%.

Example. Predicting `default`

Use a classifier to predict credit card default in a dataset of 10K people.

Predicted “yes” if $P(\texttt{default} = \text{yes}|X) > 0.5$.



		True default status		
		No	Yes	Total
Predicted default status	No	$9,644$	252	$9,896$
	Yes	23	81	104
	Total	$9,667$	333	$10,000$

- The error rate among people who do **not** default (false positive rate) is very low.
- However, the rate of false negatives is 76%.
- It is possible that false negatives are a bigger source of concern!

Example. Predicting `default`

Use a classifier to predict credit card default in a dataset of 10K people.

Predicted "yes" if $P(\texttt{default} = \text{yes}|X) > 0.5$.

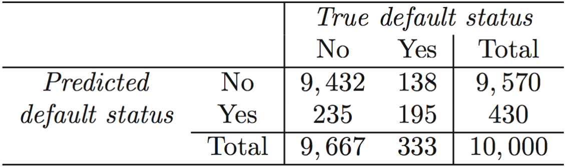
		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

- The error rate among people who do **not** default (false positive rate) is very low.
- However, the rate of false negatives is 76%.
- It is possible that false negatives are a bigger source of concern!
- One possible solution: Change the threshold.

Example. Predicting `default`

Changing the threshold to 0.2 makes it easier to classify to “yes”.

Predicted “yes” if $P(\texttt{default} = \text{yes}|X) > 0.2$.

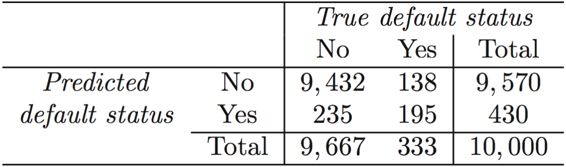


		True default status		
		No	Yes	Total
Predicted default status	No	9, 432	138	9, 570
	Yes	235	195	430
	Total	9, 667	333	10, 000

Example. Predicting `default`

Changing the threshold to 0.2 makes it easier to classify to “yes”.

Predicted “yes” if $P(\texttt{default} = \text{yes}|X) > 0.2$.

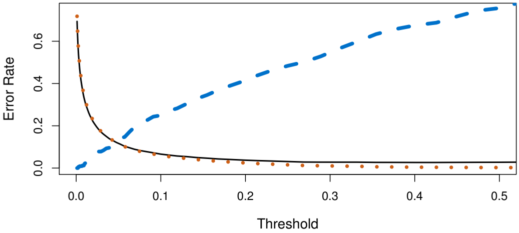


		True default status		
		No	Yes	Total
Predicted	No	$9,432$	138	$9,570$
default status	Yes	235	195	430
	Total	$9,667$	333	$10,000$

Note that the rate of false positives became higher! That is the price to pay for fewer false negatives.

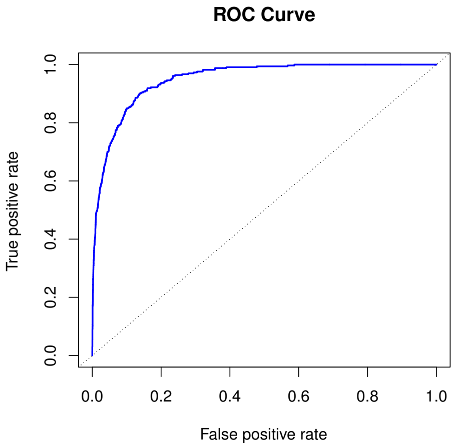
Example. Predicting `default`

Let's visualize the dependence of the error on the threshold:



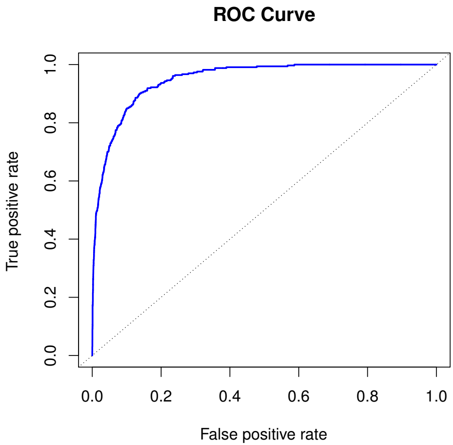
- – – – False negative rate (error for defaulting customers)
- · · · · · False positive rate (error for non-defaulting customers)
- —— 0-1 loss or total error rate.

Example. The ROC curve



- Displays the performance of the method for any choice of threshold.

Example. The ROC curve



- Displays the performance of the method for any choice of threshold.
- The area under the curve (AUC) measures the quality of the classifier:
    - 0.5 is the AUC for a random classifier
    - The closer AUC is to 1, the better.

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.

Cross-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.

Cross-validation is one way to approximate the test error:

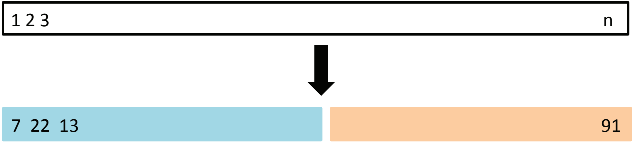
- Divide the data into two parts.
- Train each model with one part.
- Compute the error on the other.

Validation set approach

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

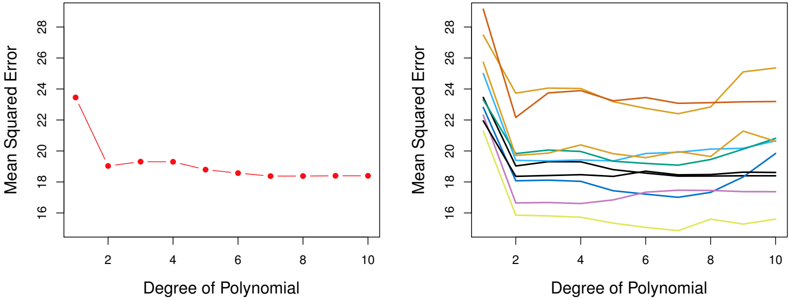
Strategy:

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



Validation set approach

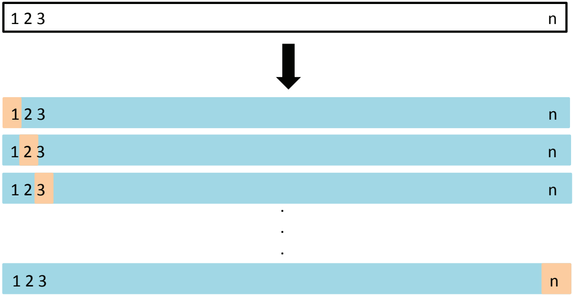
Polynomial regression to estimate `mpg` from `horsepower` in the Auto data.



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

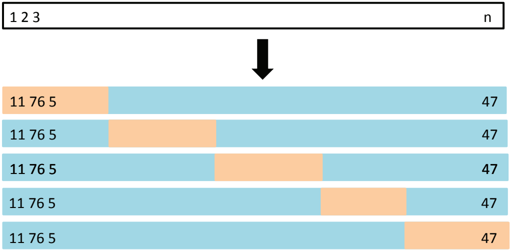
For linear regression, there is a shortcut:

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

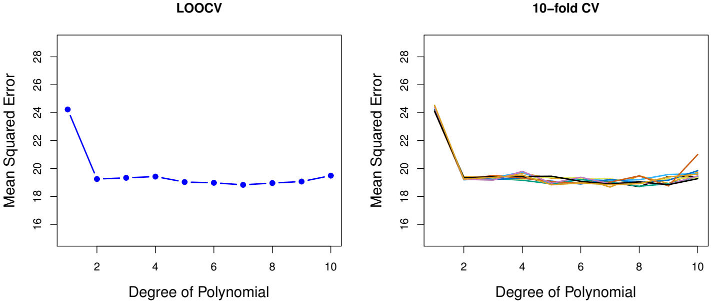
where h_{ii} is the leverage statistic.

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.

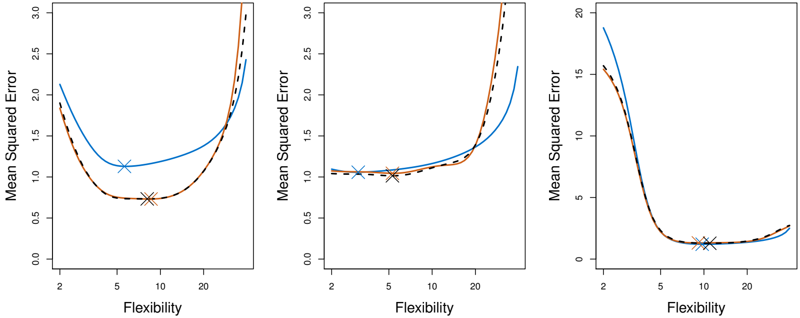


LOOCV vs. k-fold cross-validation



- k-fold CV depends on the chosen split.
- 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.

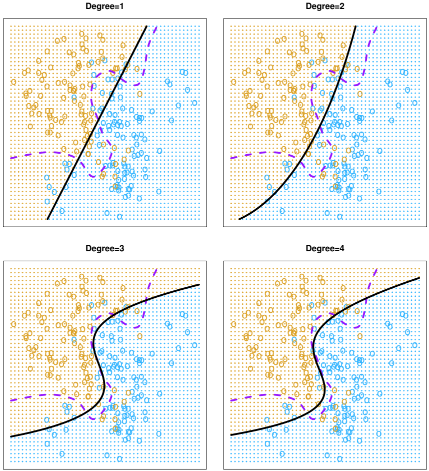
Choosing an optimal model



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

Choosing an optimal model

In a classification problem, things look similar.



- - - Bayes boundary

—— Logistic regression with polynomial predictors of increasing degree.

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:

- 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?

- 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.
- 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.