Lecture 10: Classification examples

Reading: Chapter 4

STATS 202: Data mining and analysis

October 14, 2019

Recap: Predicting `default`

Used LDA 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".

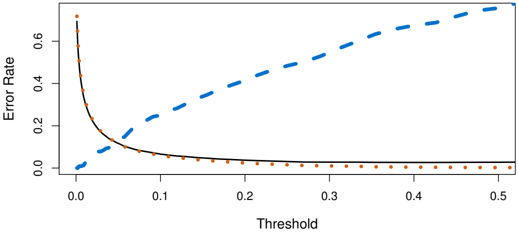
Predicted "yes" if $P(\texttt{default} = \text{yes}|X) > 0.2$.

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

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

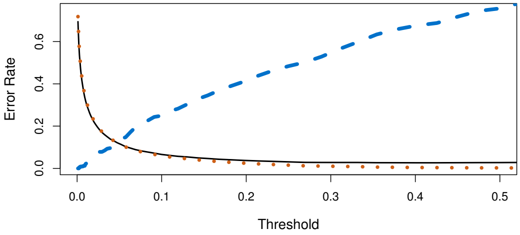
Example. Predicting `default`

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 $= \frac{\#\text{FP}+\#\text{FN}}{n}$

Example. Predicting `default`



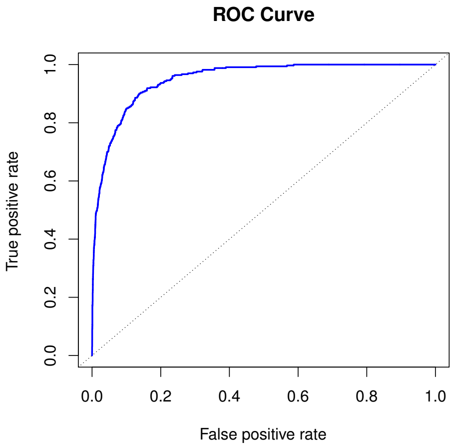
- – – – False negative rate (error for defaulting customers)
- · · · · · False positive rate (error for non-defaulting customers)
- —— 0-1 loss or total error rate

$$\text{0-1 error rate} = \frac{\#\text{FP} + \#\text{FN}}{n}$$

$$= (\text{FP rate}) \times \frac{n-m}{n} + (\text{FN rate}) \times \frac{m}{n}$$

where $m = \#$ of people who did default, $n =$ total sample size

Example. The ROC curve



- Displays the performance of the method for any choice of threshold.
- TP rate = 1-FN rate
- 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.

Changing the classification threshold to trade off between false positives and false negatives is an example of **tuning** a classifier to perform well with respect to a problem-specific objective.

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

Comparing classification methods through simulation

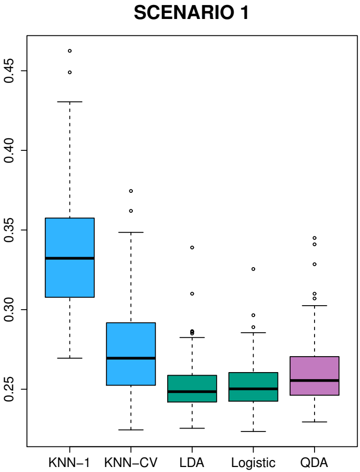
1. Simulate 100 random training sets from several different known distributions with 2 predictors and a binary response variable.

2. Compare the test error (0-1 loss) for the following methods:

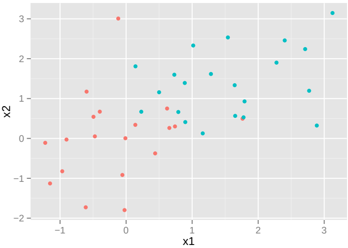
    - KNN-1
    - KNN-CV (parameter K selected using cross-validation)
    - Logistic regression
    - Linear discriminant analysis (LDA)
    - Quadratic discriminant analysis (QDA)

Check that the logistic regression boundary is linear!

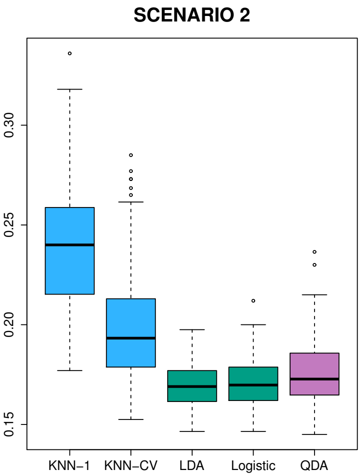
Scenario 1



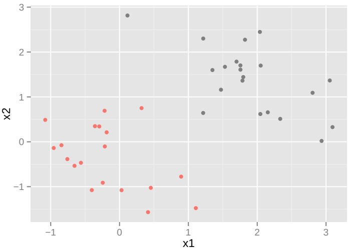
- In each class, (X_1, X_2) uncorrelated and bivariate normal, means differ between classes.



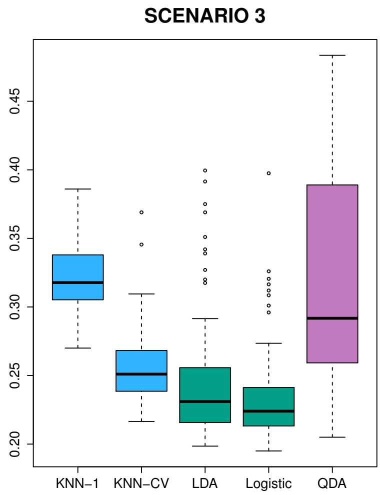
Scenario 2



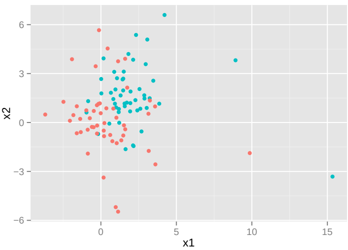
- In each class, (X_1, X_2) is bivariate normal with correlation -0.5, means differ between classes.



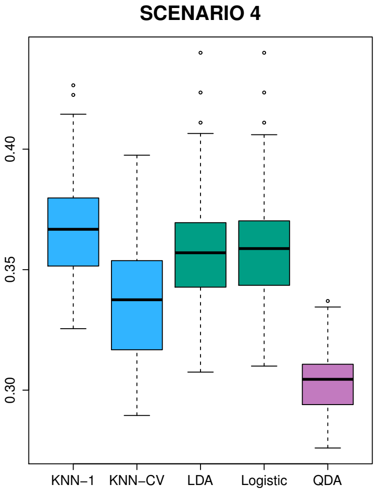
Scenario 3



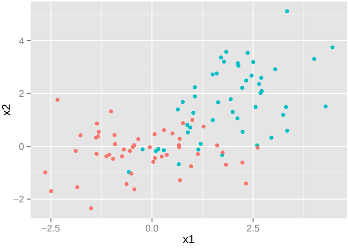
- In each class, (X_1, X_2) independent Student t random variables, means differ between classes.



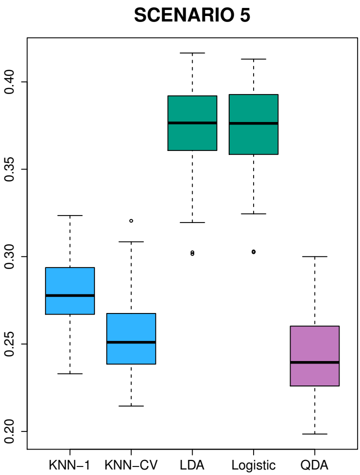
Scenario 4



- In each class, (X_1, X_2) bivariate normal, means differ between classes.
- First class has correlation 0.5, second class has correlation -0.5.



Scenario 5

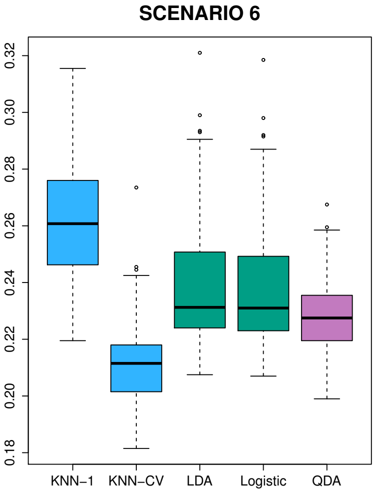


- X_1, X_2 uncorrelated, standard normal.
- Response Y was sampled from:

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

- The true decision boundary is quadratic.
- Note: We could also modify logistic regression to include quadratic terms, in the same way that we modified linear regression!

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.

Cross-validation

Key question: Given a single training set, how do we choose the supervised learning method with the best test error or select the tuning parameter for the method, e.g.

- k in k-nearest neighbors,
- the number of variables to include in forward or backward selection,
- the order of a polynomial in polynomial regression?

The **validation set** or **hold-out** approach is one way to approximate the test error:

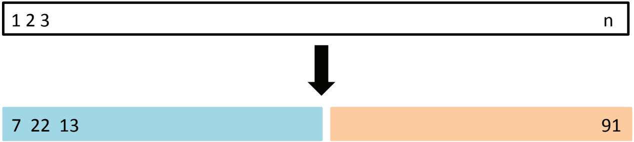
- Divide the data into two parts.
- Train each model on one part.
- Compute the error on the other.

Validation set approach

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

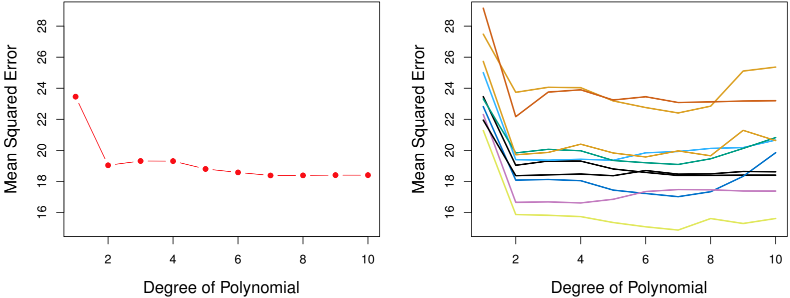
Strategy:

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



Validation set approach

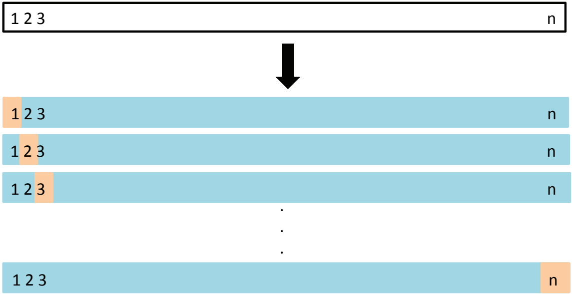
Polynomial regression to estimate `mpg` from `horsepower` in the Auto data.



Problems: 1. Every split yields a different estimate of the error. 2. Only a subset of points is used to evaluate the model.

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.

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

... for a classification problem.

Leave one out cross-validation

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

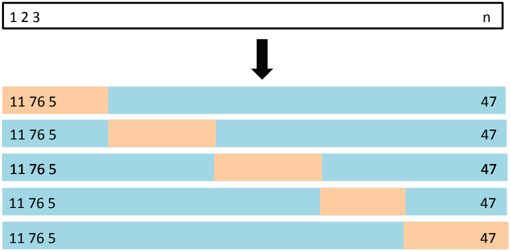
For linear regression, there is a shortcut:

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

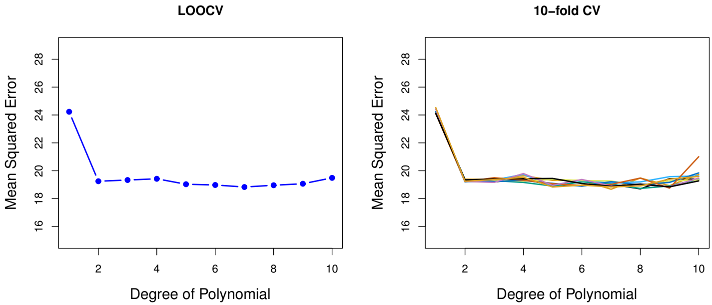
where h_{ii} is the leverage statistic.

k-fold cross-validation

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