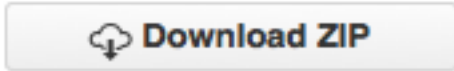


Boston Data Mining Intermediate R Workshop I

Paul Paczuski & Matthew Eaton

3/28/2015

Workshop Setup

- Download the REPO:
- <https://github.com/pavopax/intermediate-r>
- Hit the “Download ZIP” button on the lower right:

- Unzip, and then that's your working directory for R/R Studio.
- Install the packages listed in the readme.

Workshop Format

- By request: a follow-on to the R Beginner Boston Data Mining Workshop, but the Beginner workshop is not a prerequisite.
- Combining pragmatic data analysis and manipulation modules (plyr, excel import, etc) with selections from ISLR book:
<http://www-bcf.usc.edu/~gareth/ISL/>
 - Beginner workshop: Chapters 1 & 2
 - Intermediate I workshop: Chapters 3-5

Workshop Format Continued

- We will be working through various pragmatic examples of data manipulation and machine learning in R.
- Emphasis is example heavy, theory light.
- Following examples from ISLR book, but using different data sets.
 - We encourage you to work through the examples in the book after this class.

Workshop Agenda

- Basic ML introduction
- Data manipulation
 - Importing excel data
 - plyr / dplyr
- Methods
 - Linear & Logistic Regression
 - Cross-validation, AUC, bootstrap
 - Advanced classification methods: LDA, QDA, KNN

Basic ML Intro

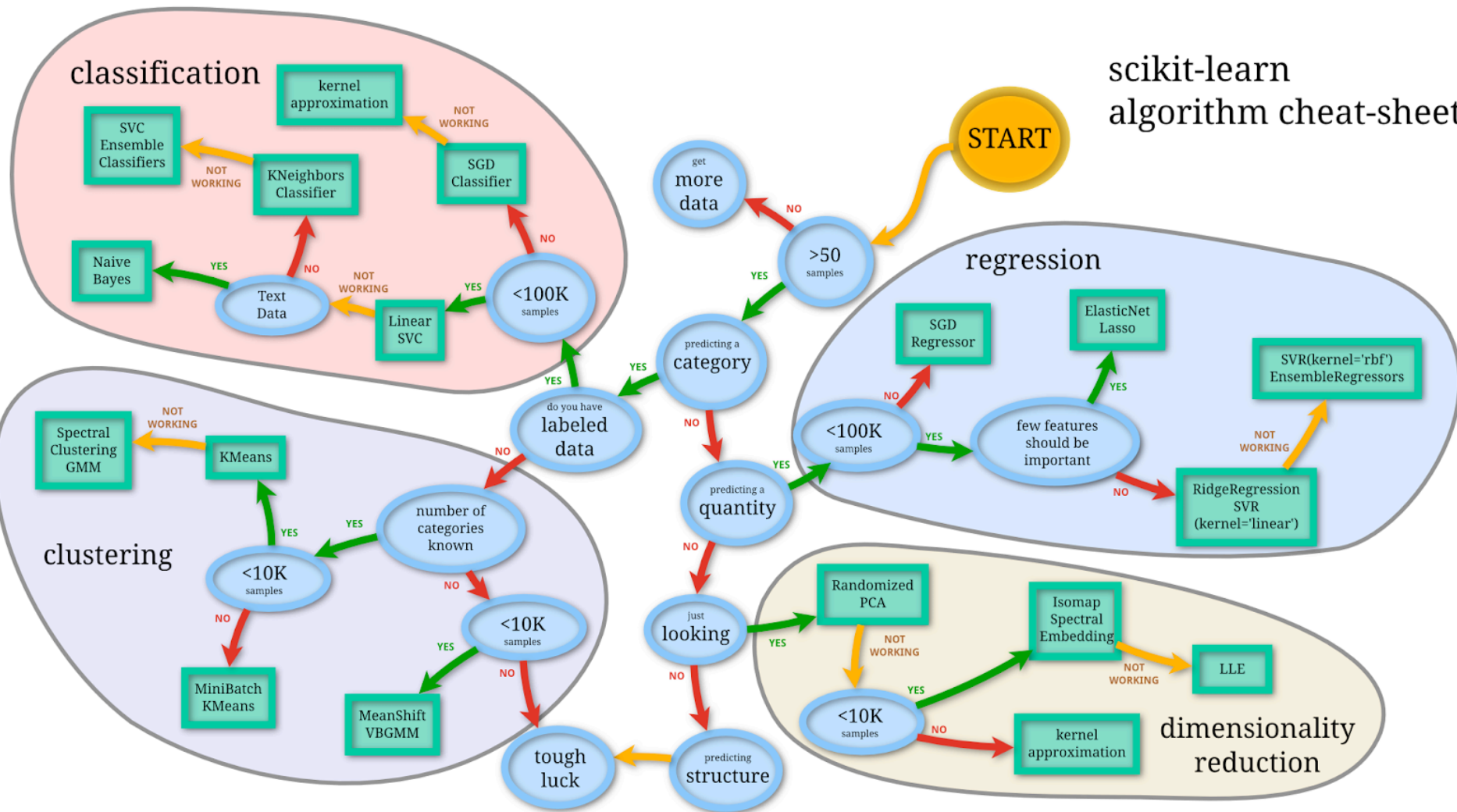
- *It's tough to make predictions, especially about the future.* – Yogi Berra

Basic ML Intro

- Statistical learning (as the book refers to it) covers a wide range of methods, with the common theme of understanding data through prediction.
- Supervised vs unsupervised.
- Predicting classes vs predicting real values.

Basic ML Intro

scikit-learn
algorithm cheat-sheet



Supervised vs Unsupervised

- Supervised: we have example data with some known outcome; class labels or real valued.
 - We learn a model that describes the relationship between the data and the outcome, and then hope that our model will accurately predict outcome based on new as-yet-unseen data.
 - Techniques: regression, classification, etc.
- Unsupervised: we have data with no class or outcome previously associated with it. We wish to discover relationships or patterns within the data.
 - Techniques: clustering, dimensionality reduction, etc.

Classes vs Real Valued

- Class-based learning:
 - We have example data associated with some labeled outcome.
 - Examples: buy vs did not buy; diabetic vs not diabetic; handwriting recognition; etc.
- Real valued learning:
 - We have example data associated with a numeric outcome.
 - Examples: predicting temperature; stock market values; presidential approval rating.

Modeling / Prediction (the ISLR treatment)

$$Y = f(X) + \varepsilon$$

Modeling / Prediction

$$Y = f(X) + \varepsilon$$

Y: Dependent Variable

f(): Some function

X: Input Variables

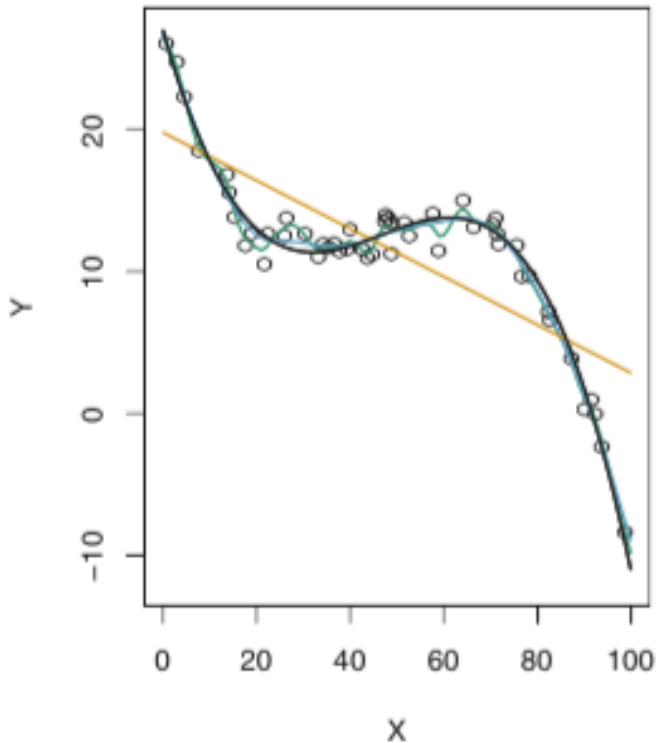
ε : Noise

- Our task is relatively straightforward: in general we have X, we may have some or all values of Y, ε we can do nothing about; we want to learn an approximation of f().
- Why do we want to learn f()? **Prediction** and **Inference**

Variance vs Bias Tradeoff

- ISLR takes an interesting approach to model selection: emphasis on variance vs bias tradeoff.
- Section 2.2 has a good treatment.

Variance vs Bias Tradeoff



- **Variance:** the amount our model would change if we estimated it using a different training data set.
 - All but guaranteed to learn a slightly different model given slightly different training data.
 - How sensitive is the model to slight variation in the training data?
- **Bias:** the error introduced by approximating a real-life problem, ..., by a much simpler model.
 - A single linear regression won't be able to exactly predict the high temperature for Boston year round.
 - Quantifying its error due to model simplicity is quantifying the bias.
- A model with low bias often has high variance and vice-versa, need to strike a balance.

Today's models

- All supervised problems.
- Mostly classification problems.
- Evaluation of the models.

Modeling / Prediction

$$Y = f(X) + \varepsilon$$

Y: Dependent Variable

f(): Some function

X: Input Variables

ε : Noise

- Our task is relatively straightforward: in general we have X, we may have some or all values of Y, ε we can do nothing about; we want to learn an approximation of f().
- Why do we want to learn f()? **Prediction** and **Inference**

Linear Model

- $f(X) = b_0 + b_1 * X_1$
- b_0 -> The intercept
- b_1 -> The slope of the dependent variable X_1 .
- $Y = f(X) + \varepsilon \rightarrow Y = b_0 + b_1 * X_1 + \varepsilon$
- We learn $f(X)$ in this case through linear regression with the `lm()` function.

The data

- Pima Native American Diabetes data set.
- Well studied in the data mining literature.
- We have 8 data points on ~700 women over 21, a little over half of whom will test positive for diabetes in five years.
- Our job is to predict which ones will eventually test positive.
- But first we'll see if we can predict Serum Insulin levels.

The data

```
options(menu.graphics=FALSE)
diabetes <- read.csv("data/pima-indians-diabetes.data.txt")
colnames(diabetes) <-
c("NumPreg","PlasmaGlucose","BP","SkinFoldThickness","SerumInsulin","BMI","Fam","
Age","Diabetes")

replace_0_with_NA <- function(x) {
  x[x==0] <- NA
  return(x)
}

for(cn in
c("PlasmaGlucose","BP","SkinFoldThickness","SerumInsulin","BMI","Fam","Age")) {
  diabetes[,cn] <- replace_0_with_NA(diabetes[,cn])
}
isna <- apply(diabetes,1,function(x) { any(is.na(x)) })
diabetes <- diabetes[!isna,]
diabetes$Diabetes <- diabetes$Diabetes==1

diabetes.raw <- diabetes
```

Data set description

- NumPreg: # times pregnant
- PlasmaGlucose: 2 hour test plasma glucose quantification
- BP: Blood pressure
- SkinFoldThickness: measure of body fat.
- SerumInsulin: quantification of insulin levels.
- BMI: Body Mass Index
- Fam: A quantification of family history for diabetes.
- Age: Current age
- Diabetes: Will they be diagnosed with diabetes in 5 years? (TRUE/FALSE)

Scale the data (makes regression coefficients more interpretable)

```
colmeans <- apply(diabetes,2,mean)
colstds <- apply(diabetes,2,sd)

## scale
for(i in 1:(ncol(diabetes)-1)) {
  diabetes[,i] <- scale(diabetes[,i])[,
1]
}

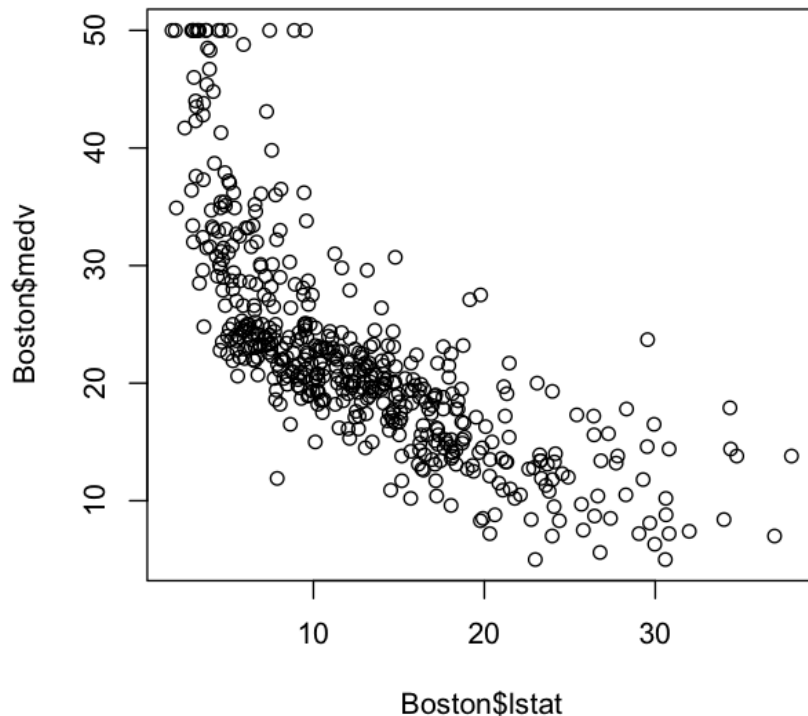
plot(density(diabetes$SerumInsulin))
```

Linear regression....

- Predict serum insulin from various factors and compare models with likelihood ratio test / ANOVA.
- Switch to code...

What if there was non-linearity?

- “Linear regression” as the name implies can be weaker (higher bias) when non-linearity.
- What is non-linearity?



Couldn't find a great non-linearity example in the Pima dataset.

Load up the Boston Housing data set (built into MASS package) and graph the % of families in a low income bracket vs the median house value in Boston Suburbs.

See how it's shaped with a bend?

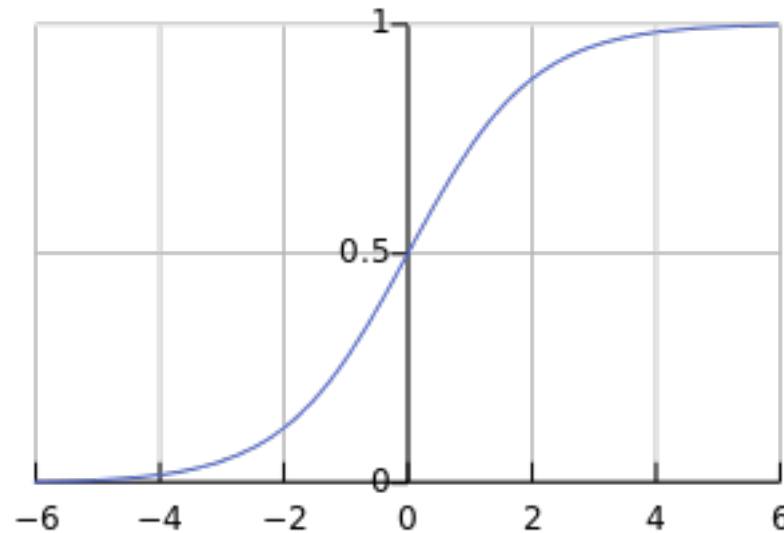
Switch to code...

Non-linearity continued...

- Other options for dealing with non-linearity:
 - Swap linear regression for a method better able to deal with non-linearity.
 - Examples:
 - QDA
 - KNN
 - Decision trees / random forests
 - Neural networks

Logistic Regression

- Goal: Make a true / false classifier.
- Replace $f(X)$ with a logistic function.



Predicting diabetes diagnosis 5 years in the future

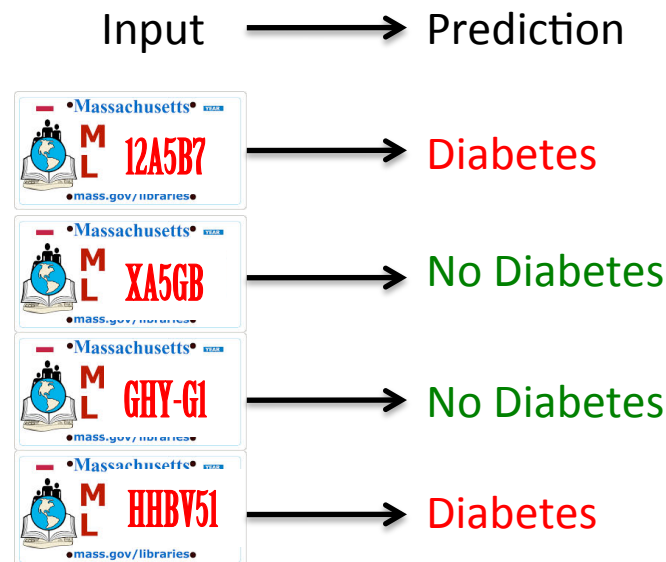
- Logistic regression
- Compare a model with age to a model that always guesses “yes”.
- Switch to code...

Predicting diabetes diagnosis 5 years in the future

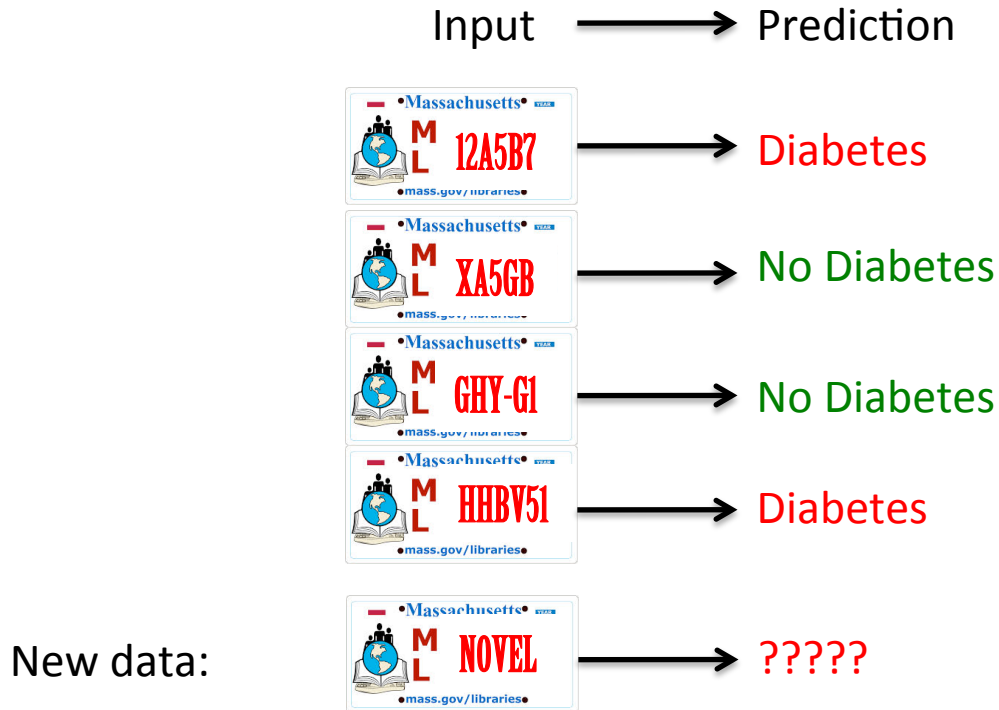
- Compare other models...
- How do we test which model is best?
- Currently we can look at accuracy of our classification, but that's significantly overfit.
 - Since we're evaluating our model's accuracy on the data we trained it on, we don't know the answer to the most fundamental question in machine learning:
 - **How well will my model do on data it's never seen before?**

Example taken to logical extreme:

- The license plate model.
- If the license plates of each person in the diabetes data set were included with the data, we could learn a model that simply maps license plates to disease status.



Example taken to logical extreme:



Our model would be 100% accurate on our training set but we would have no idea what to do when we saw a new, unlabeled license plate (no bias / infinite variance).

How do we test how our algorithm handles new data?

- Cross validation.
- Leave-one-out cross validation (code..)

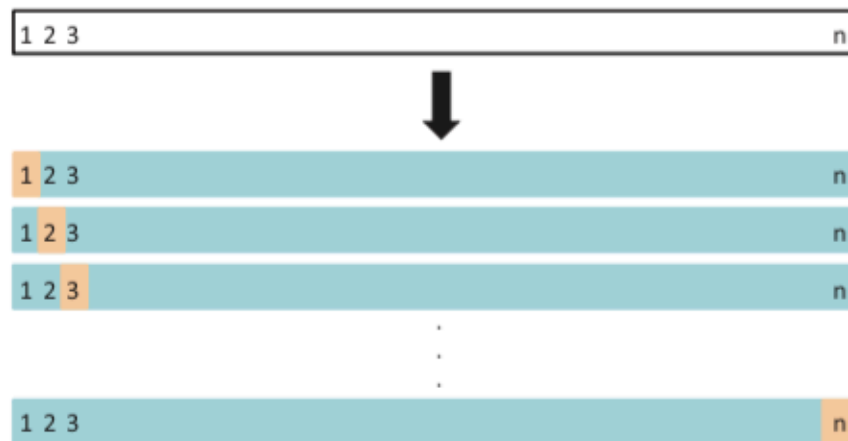


FIGURE 5.3. A schematic display of LOOCV. A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.

10-fold cross validation

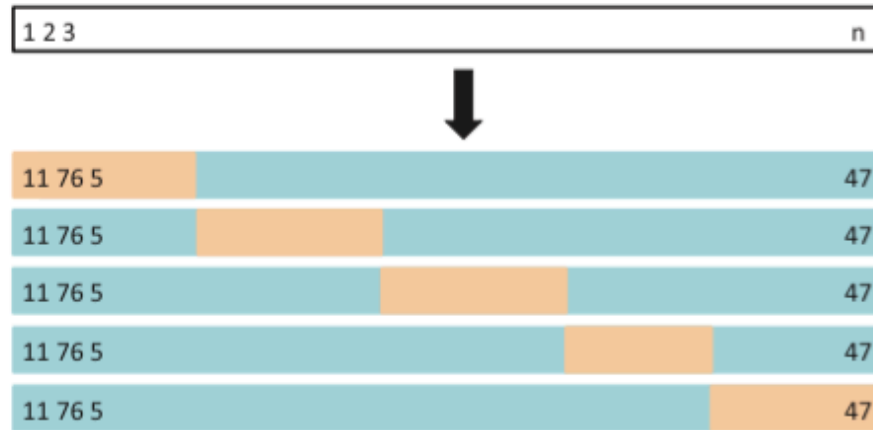


FIGURE 5.5. A schematic display of 5-fold CV. A set of n observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.

- Why might 10-fold cross validation be better than leave-one-out?

k-fold cross validation

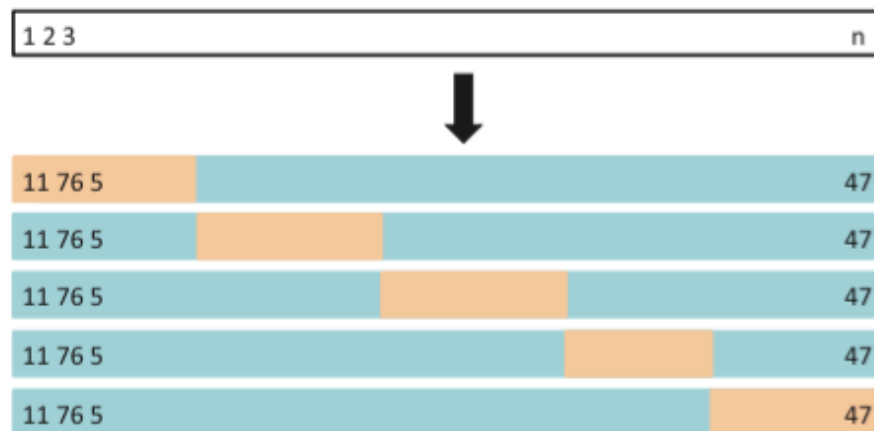
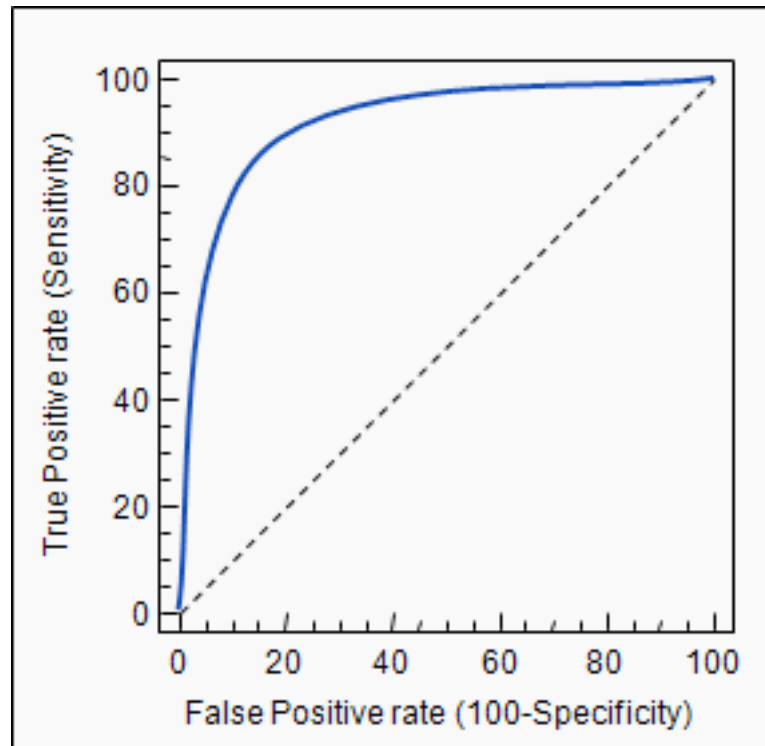


FIGURE 5.5. A schematic display of 5-fold CV. A set of n observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.

- Why might 10-fold cross validation be better than leave-one-out?
- Faster.
- Better bias vs variance tradeoff.

ROC Curve

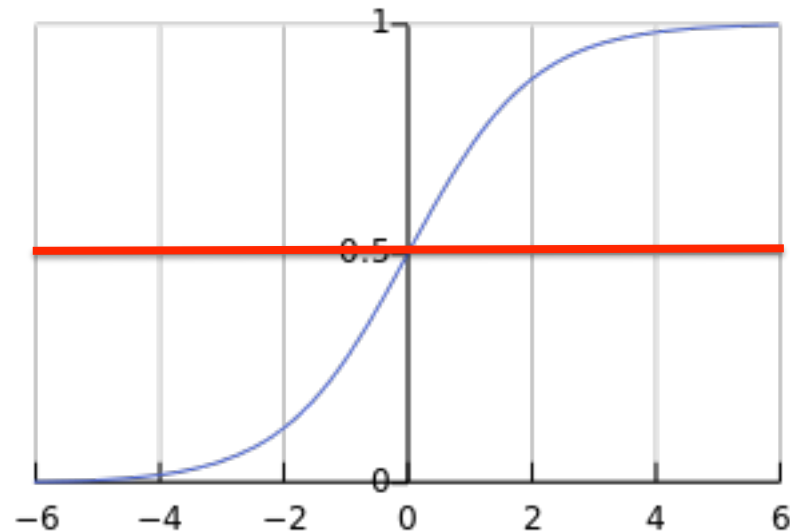


AUC: Area under the curve.

0.5 -> Random Model (coin flip)

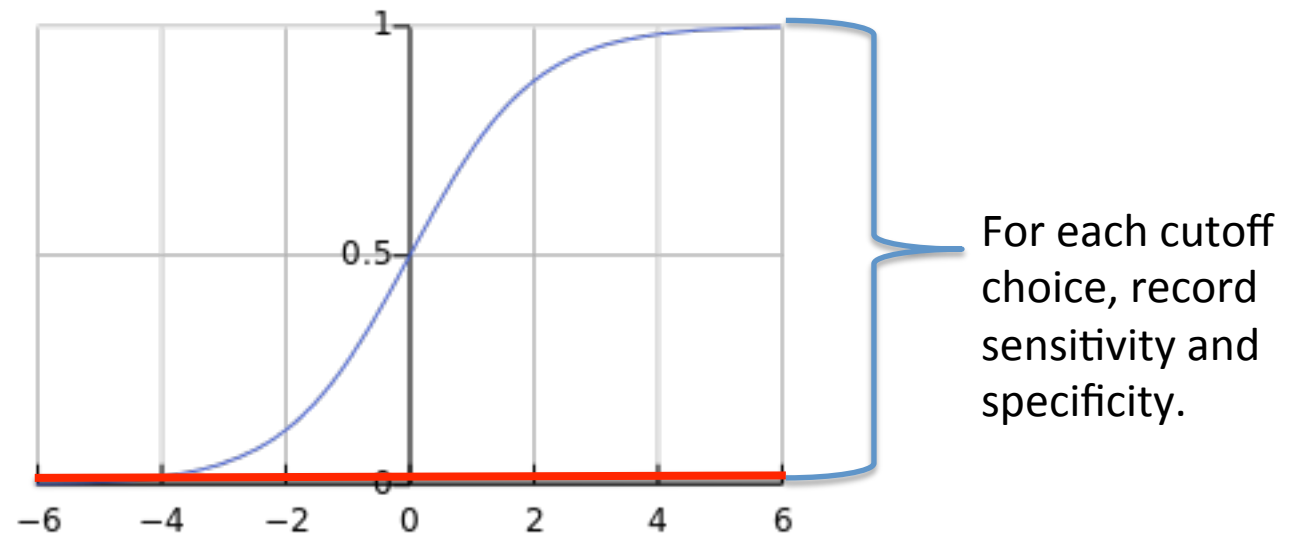
1.0 -> Perfect classifier

How do we vary sensitivity and specificity?



What if 0.5 wasn't the best place on the logistic function to put the cutoff?
What if there were things you could learn about how weighted errors were toward sensitivity or specificity by varying that cutoff?

How do we vary sensitivity and specificity?



What if 0.5 wasn't the best place on the logistic function to put the cutoff?
What if there were things you could learn about how weighted errors were toward sensitivity or specificity by varying that cutoff?

Bootstrap

- Estimating the variability of a statistic when we only have one sample set to draw from.
- Example: estimating the regression coefficients in our logistic regression models (boring)....

Bootstrap

- Estimating the variability of a statistic when we only have one sample set to draw from.
- Example: estimating the regression coefficients in our logistic regression models (boring)....
- Example: estimating the variability associated with the correlation of two variables and comparing it to a closed form estimate (mildly interesting).
- Comparing AUCs with lots of cross validation runs vs AUCs from bootstrapped version of the same cross validation scheme (very exciting!).

Advanced classification algorithms...

- LDA – Linear discriminant analysis.
- QDA – Quadratic discriminant analysis (for when straight lines will not do).
- KNN – Easy, straightforward, and usually pretty accurate.