Practical Machine Learning Notes

Coursera Course by John Hopkins University

INSTRUCTORS: Dr. Jeff Leek, Dr. Brian Caffo, Dr. Roger D. Peng

Contents

Intro

- This course covers the basic ideas behind machine learning/prediction
  + Study Design - training vs. test sets
  + Conceptual issues - out of sample error, overfitting, ROC curves
  + Practical Implementation - the caret package

- What this course depends on:
  + The Data Scientist's Toolbox
  + R Programming

- What would be useful
  + Exploratory Analysis
  + Reproducible Research
  + Regression Models
  + (Notes on these 5 courses are all in my GitHub repoes)

GitHub Link for Lectures

Practical Machine Learning lectures on GitHub

Course Book

The book for this course is available on this site

Instructor's Note

"*Welcome to Practical Machine Learning! This course will focus on developing the tools and techniques for understanding, building, and testing prediction functions.*

These tools are at the center of the Data Science revolution. Many researchers, companies, and governmental organizations would like to use the cheap and abundant data they are collecting to predict what customers will like, what services to offer, or how to improve people's lives.

Jeff Leek and the Data Science Track Team"

Prediction, Errors, and Cross Validation

Prediction

Prediction Motivation

- Who predicts things?
  + Local governments -> pension payments
  + Google -> whether you will click on an ad
  + Amazon -> what movies you will watch
  + Insurance companies -> what your risk of death is
  + Johns Hopkins -> who will succeed in their programs

- Why predict things
  + Glory (Nerd cred for accomplishing certain feats)
  - A lot of competitions are hosted on **Kaggle**
  + Riches (Completing some competition that offers a reward)
  + Save lives
  - **On cotype DX** reveals the underlying biology that changes treatment decisions 37% of the time.

More Resources

- **A course on more advanced material about ML**

- **List of machine learning resources on Quora**

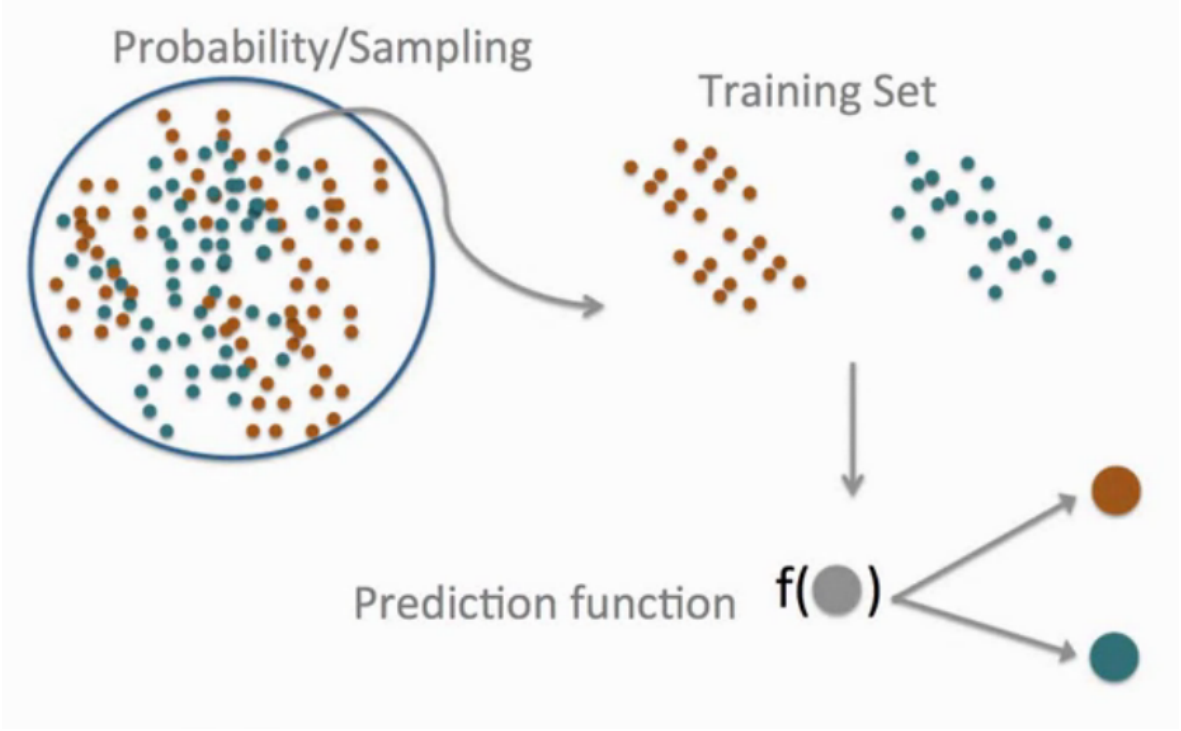
- **List of machine learning resources from Science**
- **Advanced notes from MIT open courseware**
- **Advanced notes from CMU**
- **Kaggle - machine learning competitions**

What is Prediction?

Main Idea

- One focus of ML is on what algorithms are the best for extracting information and using it to predict.

- Although the method used for producing a training set is also quite important



- One starts off with a dataset

1) One uses Probability/Sampling to select a Training Set
2) One measures characteristics of this training set to create a Predicition Function
3) One then uses the Prediction Function to take an uncolored dot and predict if it's red or blue
4) One would then go on to test how well their Prediction Function works

What Can Go Wrong

- An example is **Google Flu trends** (**A free overview of the issue witht he accuracy**)
  - Google tried to predict rate of flu using what people would search
  - Originally the algorithm was able to represent how many cases would appear in a region within a certain time
  - Although they didn't account for the fact that the terms would change over time
  - The way the terms were being used wasn't well understood so when terms changed they weren't able to accurately account for the change.
  - It also overestimated as it the search terms it looked at were often cofactors with other illnesses

Componets of a Predictor

1) Question

- Any problem in data science starts with a question, "What are you trying to predict and what are you trying to predict it with?"

2) Input Data

- Collect best input data you can to use to predict

3) Features

- From that data one builds features that they will use to predict

4) Algorithm

- One uses ML algorithms to develop a function

5) Parameters

- Estimate parameters of the algorithm

6) Evaluation

- Apply algorithm to a data set to evaluate how well the algorithm works

Example

- Start with a general question, "Can I automatically detect emails that are SPAM and those that are not?"

- Make the question more concrete, "Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?"

- Find input data
  + In this instance there is data avaliable in R via the `kernlab` package
  + Note that this data set won't necessarily be the perfect data as it doesn't contain all the emails ever sent, or the emails sent to you personally

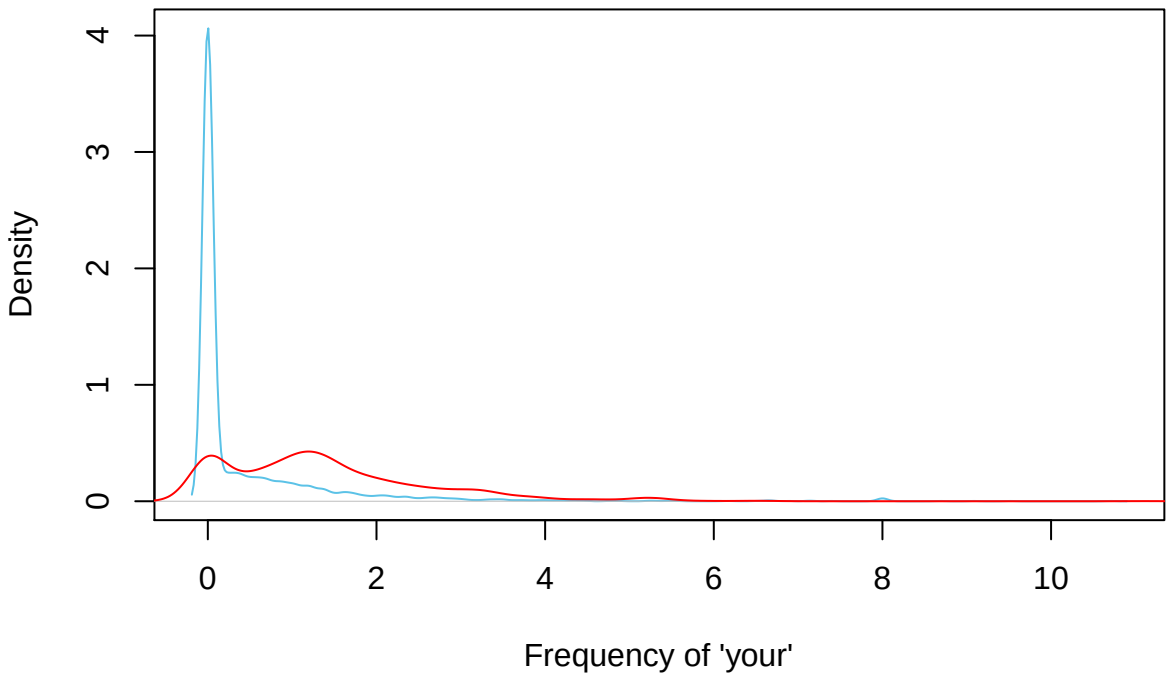
- Quantify features, such as the frequency of certain words or typeface. The `spam` dataset from `kernlab` contains these types of frequency.

```
library(kernlab)
data(spam)
str(spam)
```

```
## 'data.frame':    4601 obs. of  58 variables:
##  $ make             : num  0 0.21 0.06 0 0 0 0 0 0.15 0.06 ...
##  $ address          : num  0.64 0.28 0 0 0 0 0 0 0 0.12 ...
##  $ all              : num  0.64 0.5 0.71 0 0 0 0 0 0.46 0.77 ...
##  $ num3d            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ our              : num  0.32 0.14 1.23 0.63 0.63 1.85 1.92 1.88 0.61 0.19 ...
##  $ over             : num  0 0.28 0.19 0 0 0 0 0 0 0.32 ...
##  $ remove           : num  0 0.21 0.19 0.31 0.31 0 0 0 0.3 0.38 ...
##  $ internet         : num  0 0.07 0.12 0.63 0.63 1.85 0 1.88 0 0 ...
##  $ order            : num  0 0 0.64 0.31 0.31 0 0 0 0.92 0.06 ...
##  $ mail             : num  0 0.94 0.25 0.63 0.63 0 0.64 0 0.76 0 ...
##  $ receive          : num  0 0.21 0.38 0.31 0.31 0 0.96 0 0.76 0 ...
##  $ will             : num  0.64 0.79 0.45 0.31 0.31 0 1.28 0 0.92 0.64 ...
##  $ people           : num  0 0.65 0.12 0.31 0.31 0 0 0 0 0.25 ...
##  $ report           : num  0 0.21 0 0 0 0 0 0 0 0 ...
##  $ addresses        : num  0 0.14 1.75 0 0 0 0 0 0 0.12 ...
##  $ free             : num  0.32 0.14 0.06 0.31 0.31 0 0.96 0 0 0 ...
##  $ business         : num  0 0.07 0.06 0 0 0 0 0 0 0 ...
##  $ email            : num  1.29 0.28 1.03 0 0 0 0.32 0 0.15 0.12 ...
##  $ you              : num  1.93 3.47 1.36 3.18 3.18 0 3.85 0 1.23 1.67 ...
##  $ credit           : num  0 0 0.32 0 0 0 0 0 3.53 0.06 ...
##  $ your             : num  0.96 1.59 0.51 0.31 0.31 0 0.64 0 2 0.71 ...
##  $ font             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num000           : num  0 0.43 1.16 0 0 0 0 0 0 0.19 ...
##  $ money            : num  0 0.43 0.06 0 0 0 0 0 0.15 0 ...
##  $ hp               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hpl              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ george           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num650           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ lab              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ labs             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ telnet           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num857           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ data             : num  0 0 0 0 0 0 0 0 0.15 0 ...
##  $ num415           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num85            : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ technology       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num1999          : num  0 0.07 0 0 0 0 0 0 0 0 ...
##  $ parts            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ pm               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ direct           : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ cs               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ meeting          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ original         : num  0 0 0.12 0 0 0 0 0 0.3 0 ...
##  $ project          : num  0 0 0 0 0 0 0 0 0 0.06 ...
##  $ re               : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ edu              : num  0 0 0.06 0 0 0 0 0 0 0 ...
##  $ table            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ conference       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ charSemicolon    : num  0 0 0.01 0 0 0 0 0 0 0.04 ...
##  $ charRoundbracket : num  0 0.132 0.143 0.137 0.135 0.223 0.054 0.206 0.271 0.03 ...
##  $ charSquarebracket: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ charExclamation  : num  0.778 0.372 0.276 0.137 0.135 0 0.164 0 0.181 0.244 ...
##  $ charDollar       : num  0 0.18 0.184 0 0 0 0.054 0 0.203 0.081 ...
##  $ charHash         : num  0 0.048 0.01 0 0 0 0 0 0.022 0 ...
##  $ capitalAve       : num  3.76 5.11 9.82 3.54 3.54 ...
##  $ capitalLong      : num  61 101 485 40 40 15 4 11 445 43 ...
##  $ capitalTotal     : num  278 1028 2259 191 191 ...
##  $ type             : Factor w/ 2 levels "nonspam","spam": 2 2 2 2 2 2 2 2 2 2 ...
```

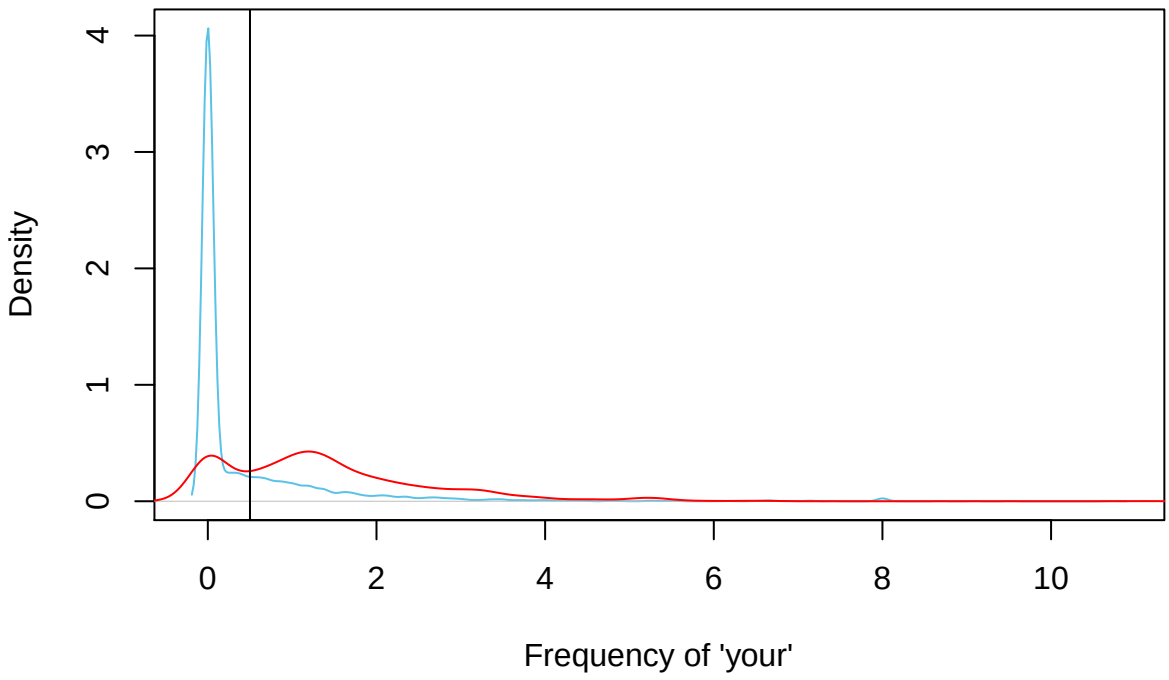
```
plot(density(spam$your[spam$type=="nonspam"]),
     col = "#5BC2E7", main = "", xlab = "Frequency of 'your'")
lines(density(spam$your[spam$type=="spam"]), col = "#FF0000")
```



- It can be seen here "your" appears more often in SPAM emails than it does in HAM
  + One could use this idea to create a cut-off point for predicting if a message is SPAM

- The proposed algorithm
  + Find a value of C
  + If the frequency of $'your' > C$ predict the message is SPAM

```
plot(density(spam$your[spam$type=="nonspam"]),
     col = "#5BC2E7", main = "", xlab = "Frequency of 'your'")
lines(density(spam$your[spam$type=="spam"]), col = "#FF0000")
abline(v = 0.5, col = "#000000")
```



- Choosing 0.5 would contain most spam messages and avoid the second spike of HAM emails

- We then evaluate this predictor

```
prediction <- ifelse(spam$your > 0.5, "spam", "nonspam")
res <- table(prediction, spam$type)/length(spam$type)
res
```

```
##
## prediction   nonspam      spam
##    nonspam 0.4590306 0.1017170
##    spam    0.1469246 0.2923278
```

- In this case our accuracy is $0.459 + 0.2923 = 0.7514$, or an accuracy of approximately 75.14%, although this is an opptamistic measure of the overall error, which will be discussed further later.

Relative Importance of Steps

question > data > features/variables > algorithms
"*The commbinaiton of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data.*" -John Tukey

- In other words, an important component of prediction is knowing when to give up, that is

that the data is not sufficient

Input Data: Garbage in = Garbage out

1. May be easy (movie ratings -> new movie ratings)
2. May be harder (gene expression data -> disease)
3. Depends on what is a "good prediction".
4. Often more **data > better models**
5. The most important step is collecting the right data

Features: They matter!

- Properties of good features
  + Lead to data compression
  + Retain relevant information
  + Are created based on expert application knowledge

- Common mistakes
  + Trying to automate feature selection (Although they may be automated with care)
  + Not paying attention to data-specific quirks
  + Throwing away information unnecessarily

Algorithm: They Matter Less Than You'd Think

- The above table shows that the Linear Discrimenate Analysis (Lindisc) error often was not that far off from the best method

- Using the best approach doesn't always largely improve the error

Issues to Consider

- The "Best" machine learning method would be:
  + Interpretable
  - If predictor is to be presented to an uninformed audience you'd want to to be understandable by them
  + Simple
  - Helps with interpretablity
  + Accurate
  - Getting a model to be interpretable can sometimes hurt the accuracy
  + Fast
  - Quick build the model, train, and test
  + Scalable
  - Easy to apply to a large dataset (either fast or parallelizable)

TABLE 1
Performance of linear discriminant analysis and the best result we found on ten randomly selected data sets

Data set	Best method e.r.	Lindisc e.r.	Default rule	Prop linear
Segmentation	0.0140	0.083	0.760	0.907
Pima	0.1979	0.221	0.350	0.848
House-votes16	0.0270	0.046	0.386	0.948
Vehicle	0.1450	0.216	0.750	0.883
Satimage	0.0850	0.160	0.758	0.889
Heart Cleveland	0.1410	0.141	0.560	1.000
Splice	0.0330	0.057	0.475	0.945
Waveform21	0.0035	0.004	0.667	0.999
Led7	0.2650	0.265	0.900	1.000
Breast Wisconsin	0.0260	0.038	0.345	0.963

Figure 1: Linear vs Model

Prediction is About Accuracy Tradeoffs

- Tradeoffs are made for interpretability, speed, simplicity, or scalability.
- Interpretability matters, decision tree-like results are more interpretable + "**if** total cholesterol $\geq$ 160 **and** they smoke **then** *10 year CHD risk* $\geq$ *5%* **else if** they smoke **and** systolic blood pressure $\geq$ 140 **then** *10 year CHD risk* $\geq$ 5% **else** *10 year CHD risk* $<$ 5%"
- Scalability matter
  + in "The Netflix $1 Million Challenge" Netflix never implemented the solution itself because the algorithm wasn't scalable and took way too long on the big data sets that Neflix was working with, so they went with something that was less accurate but more scalable.

Reminder to Commit (02), Delete this line *AFTER* Committing

Errors

In and Out of Sample Errors

Prediction Study Design

Types of Errors

Receiver Operating Characteristics

Reminder to Commit (03), Delete this line *AFTER* Committing

Cross Validation

Cross Validation

What Data Should You Use?

Reminder to Commit (04), Delete this line *AFTER* Committing

Quiz 1

Reminder to Commit (Q1), Delete this line *AFTER* Committing

The Caret Package

Caret Package

Caret Package

Training Options

Plotting Predictors

Reminder to Commit (05), Delete this line *AFTER* Committing

Preprocessing

Basic Preprocessing

Covariate Creation

Preprocessing with Principal Components Analysis (PCA)

Reminder to Commit (06), Delete this line *AFTER* Committing

Predicting

Predicting with Regression

Predicting with Regression Multiple Covariates

Reminder to Commit (07), Delete this line *AFTER* Committing

Quiz 2

Reminder to Commit (Q2), Delete this line *AFTER* Committing

Predicting with Trees, Random Forests, & Model Based Predictions

Trees

Predicting with Trees

Bagging

Reminder to Commit (08), Delete this line *AFTER* Committing

Random Forests

Random Forests

Boosting

Reminder to Commit (09), Delete this line *AFTER* Committing

Model Baded Predictions

Model Based Predictions

Reminder to Commit (10), Delete this line *AFTER* Committing

Quiz 3

Reminder to Commit (Q3), Delete this line *AFTER* Committing

Regularized Regression and Combining Predictors

Regularized Regression

Combining Predictors

Reminder to Commit (11), Delete this line *AFTER* Committing

Forecasting

Unsupervised Prediction

Reminder to Commit (12), Delete this line *AFTER* Committing

Quiz 4

Reminder to Commit (Q4), Delete this line *AFTER* Committing

Course Project

Reminder to Commit (P1), Delete this line *BEFORE* Committing