Economics 217 - "Modern" Data Science

- Topics covered in this lecture
  - K-nearest neighbors
  - Decision Trees
- There is no reading for these lectures. Just notes. However, I have copied a number of online websites that may help to the course schedule.
- I hired a PhD student who works in this area to prepare some extra notes on the website, which provide more examples for those who are interested.

Modern data science

- In 216 and 217, we have (mostly) evaluated empirical relationships using parametric models
  - Parametric models almost surely have some form of model mis-specification, but are helpful in that the techniques to analyze the models are well sussed-out and interpretations of the model are fairly straightforward (ie. take a derivative)
  - In new data science lingo, we are "supervising" the data with a model
- In this last few lectures, we have been more flexible with our modeling choices
  - More flexible models that are non-parametric
  - Resampling procedures to conduct inference and choose smoothing parameters
- Practically, much of the new data science literature, learning and otherwise, isn't all that different from what we're doing already.
  - The main difference is the choice of non-parametric model, and the goal is to improve prediction.

Modern data science (cont.)

- Whether you adopt new techniques or old techniques is usually a function of your research objective
- In economics, we often wish to understand the mechanisms behind behaviors, as opposed to the collection of attributes that lead to behaviors.
  - Example: Knowing that graduates from Harvard are more likely to own a new house than graduates of Cabrillo college might be interesting from a marketing perspective, but it tells us nothing of *why* this is the case
  - If we are constructing policy, we want to know why. That is the big difference between modern data science and econometrics as I see it (even though in principle the techniques are very similar)
  - To be sure, the techniques can be complementary.
- In this lecture, we will study two techniques:
  - **K-Nearest Neighbors**: Similar people do similar things.
  - **Decision Trees**: Individuals adopt a heuristic to make choices.

K-Nearest Neighbors

- K-Nearest Neighbors is extremely similar to the Nadaraya-Watson binned estimator
  - In NW, we take a bandwith of h on either side of a given x, and average the behaviour within the region to generate a prediction for y.
  - This technique can be extended to more than one dimension of x by using a measure of "Euclidean Distance".
- K-Nearest Neighbors (KNN) also measures average (or modal) behavior around a particular point.
  - Instead of a fixed distance of h around a particular x, KNN, uses k nearest neighboring observations to measure behavior.
- The key inputs to a basic KNN model
  - The choice of k (obviously)
  - The distance function
  - The outcome variable (eg. unemployment)
  - The input variables (which will be used to determine who is nearest)

K-Nearest Neighbors - Distance

- Euclidean Distance is a common measure of distance.
- In P dimensions, Euclidean distance of two observations, $\mathbf{x}_i = (x_{i1}, x_{i2}, \cdots, x_{ip})$, and $\mathbf{x}_j = (x_{j1}, x_{j2}, \cdots, x_{jp})$, is:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{l=1}^{p} (x_{il} - x_{jl})^2}$$

- In one dimension, it is just absolute distance
- In two dimensions, this is basically the Pythagorean theorem.
- Other distance functions exist, but we'll just use Euclidean distance

K-Nearest Neighbors - Outcomes

- In the NW estimator, we averaged outcomes within the bandwidth.
  - Eg. Average real wage
  - Averages might be weighted by a kernel function
- In data science jargon, outcomes can also be "classifications"
  - Unemployed, part-time, employed, out of workforce
  - Classifications are hard to average
- For KNN, the prediction is:
  - Average value if outcome is numeric
  - The modal value if outcome is a classification (this is called "majority rule" in data science lingo)
- Similar to h being chosen by cross-validation in NW, k can be chosen by a similar technique for KNN.

R example: K-Nearest Neighbors

- Load the necessary libraries

```
library(caret)
library(foreign)
```

- Load and clean data

```
d<-read.dta("/Users/acspearot/Data/CPSDWS/org_example.dta")
d<-subset(d,is.na(nilf)==FALSE)
d<-subset(d,is.na(educ)==FALSE)
d<-subset(d,is.na(age)==FALSE)
d<-subset(d,is.na(female)==FALSE)
```

- Construct the "training" and "testing" samples:

```
subtrain<-subset(d,year==2013&state=="CA")
subtest<-subset(d,year==2013&state!="CA")
```

- Run your model:

```
model.knn <- train(nilf ~age+educ+female, data = subtrain, method =
"knn")
```

R example: K-Nearest Neighbors

- After the regression, check accuracy using the training sample

```
val.pred <- predict(model.knn, subtrain)
```

- Calculate the share of predictions that match the actual values in the training sample

```
val.acc <- sum(val.pred ==
subtrain$nilf,na.rm=TRUE)/length(subtrain$nilf)

print(val.acc)
```

- Now to the same with the testing sample

```
pred <- predict(model.knn2, subtest)

accuracy <- sum(pred == subtest$nilf2,na.rm=TRUE)/length(subtest$nilf2)

print(accuracy)
```

- By comparing "acc" and "accuracy", we can compare how well the model does within sample and out of sample.

R example: K-Nearest Neighbors (cont.)

- The results look pretty poor. So, let's redefine our outcome variable as non-numeric

```
d$nilf2<-ifelse(d$nilf==1,"Out of Labor Force", "In Labor Force")
```

- Re-construct the "training" and "testing" samples:

```
subtrain<-subset(d,year==2013&state=="CA")
subtest<-subset(d,year==2013&state!="CA")
```

- Run the model:

```
model.knn <- train(nilf2 ~age+educ+female, data = subtrain, method =
"knn")
```

- And compare accuracy:

```
obsr<-subtrain$nilf2
val.pred <- predict(model.knn2, subtrain)
val.acc <- sum(val.pred == obsr,na.rm=TRUE)/length(obsr)
pred <- predict(model.knn2, subtest)
accuracy <- sum(pred == obsr,na.rm=TRUE)/length(obsr)
print(val.acc)
print(accuracy)
```

Decision Trees

- Decision Trees are a form of classification, and map nicely into a "heuristic" approach of decision making by individuals.
- An example: Buying a car
    - Car or Truck
        - Domestic or Foreign
- Decision Trees can also be used to categorize outcomes by defining thresholds
- Suppose the outcome is "employed"
    - White or Non-White
        - Education greater than X, or less than X
- These are very complex models, but they general require (1) an order of "sub-trees", (2) splitting variables and (3) splitting points.
    - All three components can be chosen by cross-validation.
- The technique that is used for estimation is called recursive partitioning.

R example: Decision Trees

- Let's evaluate employment outcomes as a function of education and demographics.
- Load the required libraries

```
library(rpart)
library(foreign)
library(rattle)
```

- Reload and prepare outcome variable

```
d<-read.dta("/Users/acspearot/Data/CPSDWS/org_example.dta")
d<-subset(d,is.na(educ)==FALSE)
d<-subset(d,is.na(age)==FALSE)
d<-subset(d,is.na(female)==FALSE)
```

- Take "lfstat", which is labor force status, and create a dichotomous variable for whether or not the respondent is employed

```
d$lfstat2<-ifelse(d$lfstat=="Employed","Employed","Not Employed")
```

R example: Decision Trees (cont)

- Just like with the KNN, create the training and testing samples

```
subtrain<-subset(d,year==2013&state=="CA")
```

```
subtest<-subset(d,year==2013&state!="CA")
```

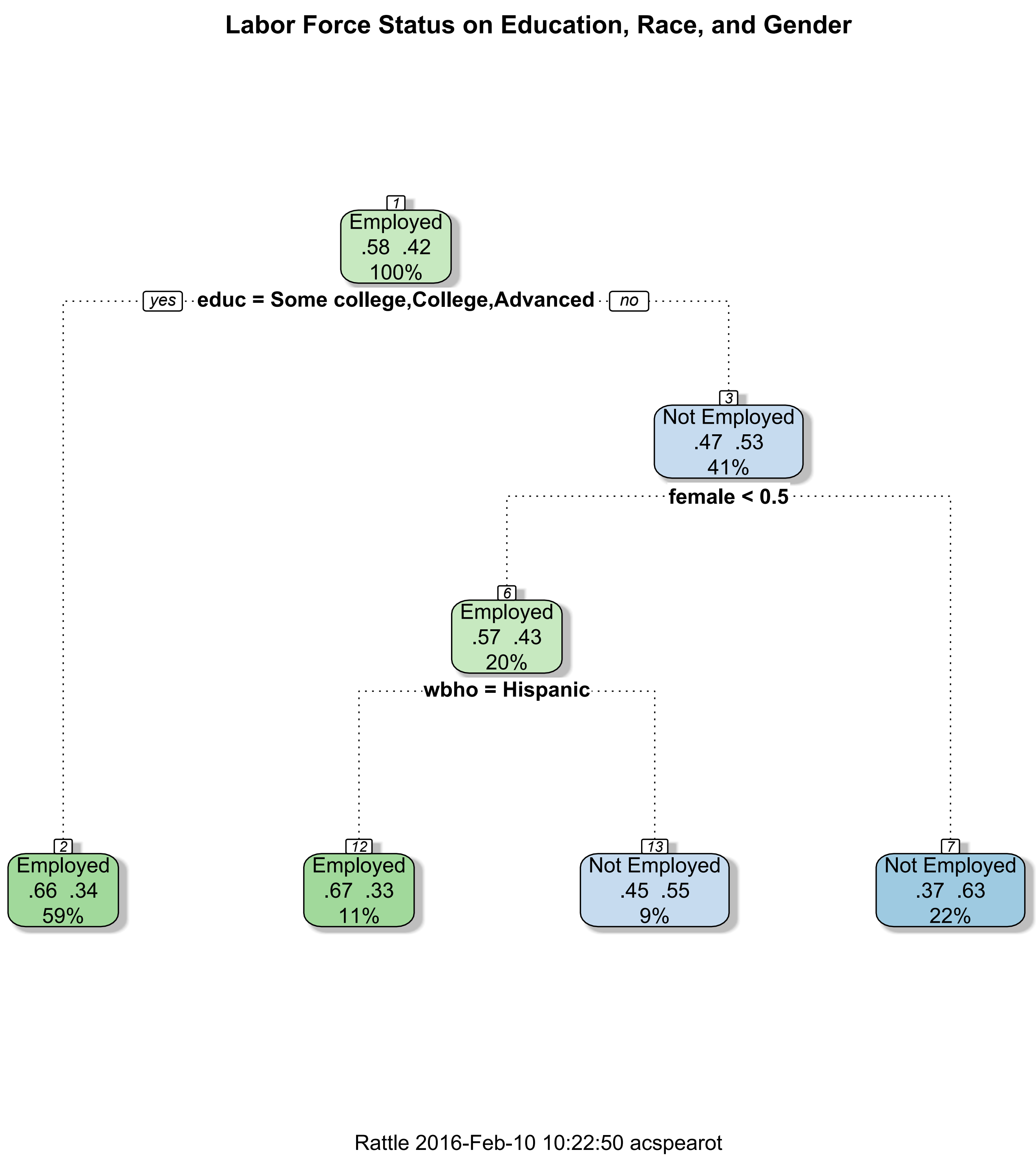
- Run the classification tree

```
tree <- rpart(lfstat2 ~educ+wbho+female,data = subtrain, method =
"class")
```

- Use "fancyRpartPlot" from the library "rattle" to visualize the results

```
fancyRpartPlot(tree, main = "Labor Force Status on Education, Race, and
Gender")
```

- There are a host of other plotting functions


Labor Force Status on Education, Race, and Gender
1
Employed
.58 .42
100%
yes
educ = Some college,College,Advanced
no
3
Not Employed
.47 .53
41%
female < 0.5
6
Employed
.57 .43
20%
wbho = Hispanic
2
Employed
.66 .34
59%
12
Employed
.67 .33
11%
13
Not Employed
.45 .55
9%
7
Not Employed
.37 .63
22%
Rattle 2016-Feb-10 10:22:50 acspearot

R example: Decision Trees (cont)

- Calculate accuracy for the training model

```
obsr<-subtest$lfstat2
outcomes <- predict(tree, subtest)
preds <- ifelse(outcomes[,1] >= .5, "Employed", "Not Employed")
accuracy <- round(sum(preds == obsr,na.rm=TRUE)/length(preds))
print(accuracy)
```

- And compare with the testing model

```
obsr<-subtrain$lfstat2
outcomes <- predict(tree, subtrain)
preds <- ifelse(outcomes[,1] >= .5, "Employed", "Not Employed")
accuracy <- round(sum(preds == obsr,na.rm=TRUE)/length(preds), digits =
4)
print(accuracy)
```