Predicting with trees

Jeffrey Leek

May 18, 2016

Key ideas

- Iteratively split variables into groups
- Evaluate "homogeneity" within each group
- Split again if necessary

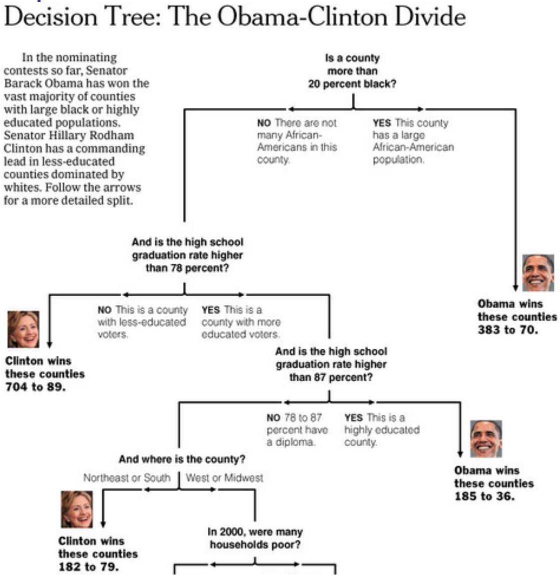
Pros:

- Easy to interpret
- Better performance in nonlinear settings

Cons:

- Without pruning/cross-validation can lead to overfitting
- Harder to estimate uncertainty
- Results may be variable

Example Tree



http://graphics8.nytimes.com/images/2008/04/16/us/0416-nat-subOBAMA.jpg

Basic algorithm

1. Start with all variables in one group
2. Find the variable/split that best separates the outcomes
3. Divide the data into two groups ("leaves") on that split ("node")
4. Within each split, find the best variable/split that separates the outcomes
5. Continue until the groups are too small or sufficiently "pure"

Measures of impurity

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \ in \ Leaf \ m} \not{1}(y_i = k)$$

Misclassification Error:

$$1 - \hat{p}_{mk(m)}; k(m) = \mathrm{most; common; k}$$

* 0 = perfect purity * 0.5 = no purity

Gini index:

$$\sum_{k \neq k'} \hat{p}_{mk} \times \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) = 1 - \sum_{k=1}^{K} p_{mk}^2$$

- 0 = perfect purity
- 0.5 = no purity

http://en.wikipedia.org/wiki/Decision_tree_learning

Measures of impurity

Deviance/information gain:

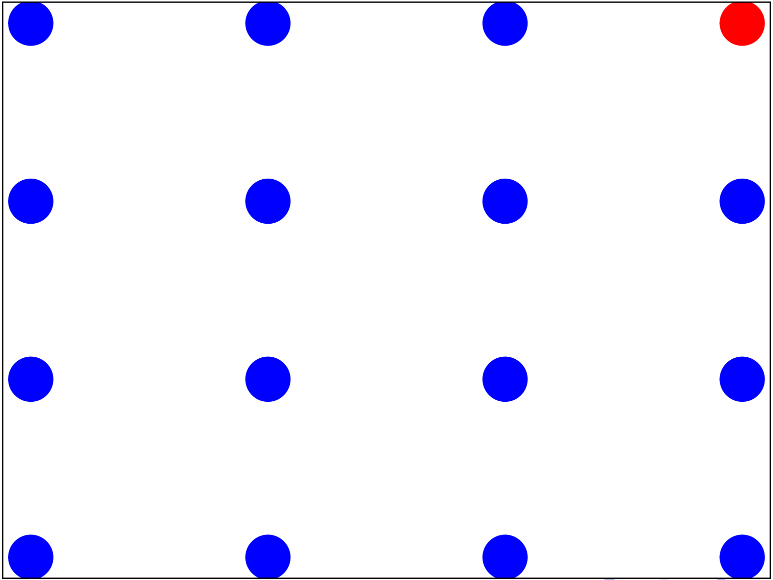
$$-\sum_{k=1}^{K} \hat{p}_{mk} \log_2 \hat{p}_{mk}$$

* 0 = perfect purity * 1 = no purity

http://en.wikipedia.org/wiki/Decision_tree_learning

Measures of impurity

*** =left



Example: Iris Data

```
data(iris); library(ggplot2)
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.
## [5] "Species"
```

```
table(iris$Species)
```

```
##
##     setosa versicolor  virginica
##         50         50         50
```

Create training and test sets

```r
library(caret)
```

```
## Loading required package: lattice
```

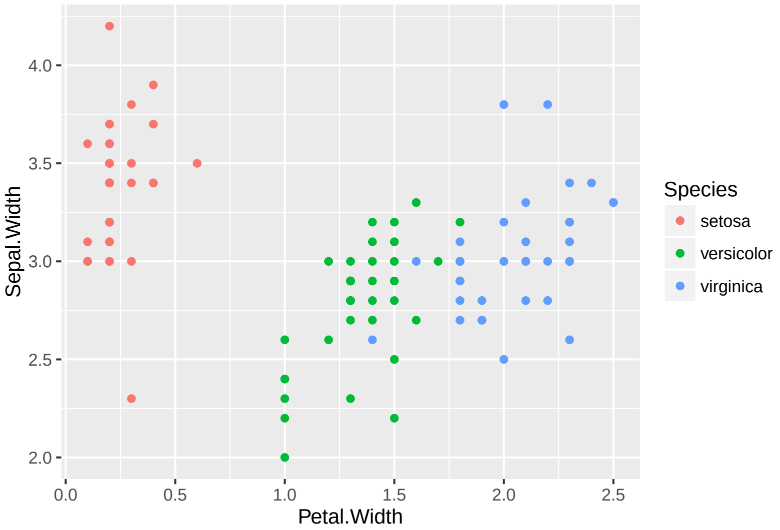
```r
inTrain <- createDataPartition(y=iris$Species,
                               p=0.7, list=FALSE)
training <- iris[inTrain,]
testing <- iris[-inTrain,]
dim(training); dim(testing)
```

```
## [1] 105   5
```

```
## [1] 45  5
```

Iris petal widths/sepal width

```
library(ggplot2)
qplot(Petal.Width,Sepal.Width,colour=Species,data=training)
```



Iris petal widths/sepal width

```
library(caret)
modFit <- train(Species ~ .,method="rpart",data=training)
```

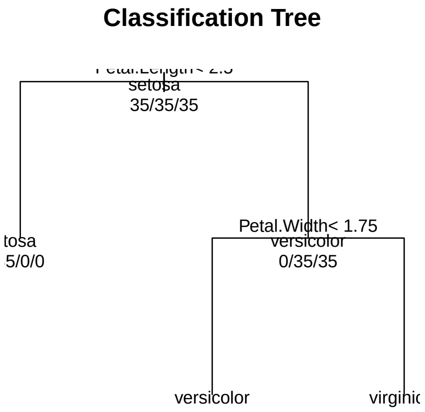
```
## Loading required package: rpart
```

```
print(modFit$finalModel)
```

```
## n= 105
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 105 70 setosa (0.33333333 0.33333333 0.33333333)
##   2) Petal.Length< 2.5 35  0 setosa (1.00000000 0.000000
##   3) Petal.Length>=2.5 70 35 versicolor (0.00000000 0.50
##     6) Petal.Width< 1.75 36  2 versicolor (0.00000000 0.
##     7) Petal.Width>=1.75 34  1 virginica (0.00000000 0.0
```

Plot tree

```
plot(modFit$finalModel, uniform=TRUE,
      main="Classification Tree")
text(modFit$finalModel, use.n=TRUE, all=TRUE, cex=.8)
```

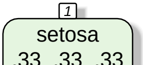


Prettier plots

```
library(rattle)
```

```
## Warning: Failed to load RGtk2 dynamic library, attemptin

## Please install GTK+ from http://r.research.att.com/libs/

## If the package still does not load, please ensure that G

## IN ANY CASE, RESTART R BEFORE TRYING TO LOAD THE PACKAGE

## Rattle: A free graphical interface for data mining with
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
fancyRpartPlot(modFit$finalModel)
```



Predicting new values

```r
predict(modFit,newdata=testing)
```

```
##  [1] setosa     setosa     setosa     setosa     setosa
##  [7] setosa     setosa     setosa     setosa     setosa
## [13] setosa     setosa     setosa     versicolor versico
## [19] versicolor versicolor versicolor versicolor versico
## [25] versicolor versicolor versicolor versicolor versico
## [31] virginica  versicolor virginica  virginica  virgini
## [37] virginica  versicolor virginica  virginica  virgini
## [43] virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

Notes and further resources

- Classification trees are non-linear models
- They use interactions between variables
- Data transformations may be less important (monotone transformations)
- Trees can also be used for regression problems (continuous outcome)
- Note that there are multiple tree building options in R both in the caret package - party, rpart and out of the caret package - tree
- Introduction to statistical learning
- Elements of Statistical Learning
- Classification and regression trees