CAS Spring Meeting 2012
Phoenix, AZ

Decision Trees in R

Linda Brobeck
Linda.Brobeck@comcast.net

Classical and Regression Trees

- Recursive partitioning is a fundamental tool in data mining.
- It helps us explore the structure of a set of data, while developing easy to visualize decision rules for predicting a categorical (classification tree) or continuous (regression tree) outcome.
- R package - rpart

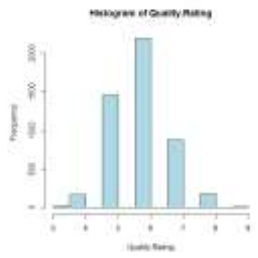
rpart(*formula*,data=,method=,control=)

formula	*outcome~predictor1+predictor2+predictor3+etc.*
data=	Specifies the data
method=	**“class”** for classification tree **“anova”** for regression tree
control=	Optional parameters for controlling tree growth

printcp()	display complexity parameter table
plotcp()	Plot cross-validation results
rsq.rpart()	Plot relative error for different splits
summary()	Detailed results including surrogate splits
plot()	Plot decision tree
text()	Label the decision tree plot

The Data

- **Wine Quality** (score 0-10) ~
  - fixed acidity
  - volatile acidity
  - citric acid
  - residual sugar
  - chlorides
  - free sulfur dioxide
  - total sulfur dioxide
  - density
  - pH
  - sulphates
  - alcohol

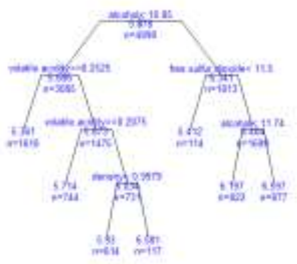


- 4898 observations
- Source: Paulo Cortez, University of Minho, Guimarzes, Portugal
  - A. Cerdeira, F. Almeida, T. Matos and J. Reis, Viticulture Commission of the Vinho Verde Region(CVRVV), Porto, Portugal @2009
  - http://archive.ics.uci.edu/ml/datasets/Wine+Quality

Data Sample

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6
6.3	0.3	0.34	1.6	0.049	14	132	0.994	3.3	0.49	9.5	6
8.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6
7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6
8.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
6.2	0.32	0.16	7	0.045	30	136	0.9949	3.18	0.47	9.6	6
7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6
6.3	0.3	0.34	1.6	0.049	14	132	0.994	3.3	0.49	9.5	6
8.1	0.22	0.43	1.5	0.044	28	129	0.9938	3.22	0.45	11	6
8.1	0.27	0.41	1.45	0.033	11	63	0.9908	2.99	0.56	12	5
8.6	0.23	0.4	4.2	0.035	17	109	0.9947	3.14	0.53	9.7	5
7.9	0.18	0.37	1.2	0.04	16	75	0.992	3.18	0.63	10.8	5
6.6	0.16	0.4	1.5	0.044	48	143	0.9912	3.54	0.52	12.4	7
8.3	0.42	0.62	19.25	0.04	41	172	1.0002	2.98	0.67	9.7	5
6.6	0.17	0.38	1.5	0.032	28	112	0.9914	3.25	0.55	11.4	7
6.3	0.48	0.04	1.1	0.046	30	99	0.9928	3.24	0.36	9.6	6
6.2	0.66	0.48	1.2	0.029	29	75	0.9892	3.33	0.39	12.8	8
7.4	0.34	0.42	1.1	0.033	17	171	0.9917	3.12	0.53	11.3	6
6.5	0.31	0.14	7.5	0.044	34	133	0.9955	3.22	0.5	9.5	5
6.2	0.66	0.48	1.2	0.029	29	75	0.9892	3.33	0.39	12.8	8

Classification Tree for White Wine Quality



>tree<-rpart(quality~fixed.acidity + citric.acid + volatile.acidity + residual.sugar + free.sulfur.dioxide + total.sulfur.dioxide + chlorides + alcohol + sulphates + pH + density)

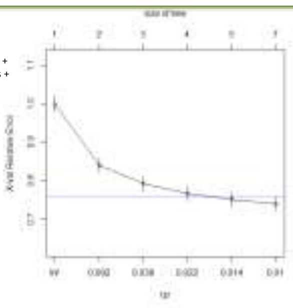
printcp and plotcp

```
Regression tree:
rpart(formula = quality ~ fixed.acidity + citric.acid +
volatile.acidity + residual.sugar + free.sulfur.dioxide +
total.sulfur.dioxide + chlorides + alcohol + sulphates +
pH + density)

Variables actually used in tree construction:
alcohol          volatile.acidity
density          free.sulfur.dioxide

n= 4898

        CP    nsplit  rel error   xerror     xstd
1 0.161007     0     1.00000   1.00013   0.021270
2 0.052469     1     0.83899   0.83971   0.020031
3 0.027342     2     0.78652   0.79292   0.019500
4 0.017711     3     0.75918   0.76757   0.018464
5 0.010355     4     0.74147   0.75181   0.017963
6 0.010000     6     0.72076   0.74062   0.017798
```



dotted line denotes the upper limit of the one standard deviation rule

Pruning the Tree

- Prune back the tree to avoid over-fitting the data.
- Typically, you will want to select a tree size that minimizes the cross-validated error, the **xerror** column printed by **printcp( )**
- Prune the tree to the desired size using **prune(***tree*, **cp=**)
- Specifically, use **printcp( )** to examine the cross-validated error results, select the complexity parameter associated with minimum error, and place it into the **prune( )** function.

- For this example, the pruned tree is identical to the original.

Appendix

```
library(rpart) #makes sure rpart is loaded

#load data
wine <-read.csv( "c:/Temp/wine_white_v01.csv",  header = TRUE, sep=",", na.strings = "NaN")

attach(wine) #allow use of variable names without wine$

#create tree
tree<-rpart(quality ~ fixed.acidity + citric.acid + volatile.acidity + residual.sugar + free.sulfur.dioxide +
total.sulfur.dioxide + chlorides + alcohol + sulphates + pH + density)

printcp(tree) # display the results
plotcp(tree) # visualize cross-validation results
summary(tree) # detailed summary of splits

# plot tree
plot(tree, uniform=TRUE,main="Classification Tree for White Wine Quality",branch=.5,margin=.1)
text(tree, use.n=T, all=TRUE, cex=.8, col="blue")

#prune tree
pruned.tree<-prune(tree, cp=.01)
```