Prostate Cancer Prediction

Gary Seamans April 19, 2018

- This example predicts tumor spread in this dataset of 97 men who had undergone a biopsy.
- The measures to be used for prediction are BPH, PSA, Gleason Score, CP, and size of prostate.

Tumor spread is indicated by the presence of cancer outside of the prostrate. In this dataset that is indicated by capsular penetration (lcp).

Data Preparation

The initial step was to load the Prostate Cancer dataset, examine the data, and determine if any adjustments will be necessary.

Load the data

```r
library('lasso2')
data('Prostate')
```

Examine the data

First a table of the definitions of the *datadefinitions.csv*.

```r
## Load the data
definitions <- read.csv(file = "datadefinitions.csv")
## Load the xtable library for LaTex tables
library(xtable)

## Source the code to create pretty str() tables
source('strtable.R')

## Remove all but the assignment variables from the dataset
prostateSubset <- Prostate[c(2,4,6,7,9)]
print(xtable(strtable(prostateSubset), caption = "Data Types"))
print(xtable(definitions, caption = "Data Descriptions"),
      include.rownames = FALSE)
```

Table 1: Data Descriptions

Name	Description
lcavol	log(cancer volume)
* lweight	log(prostate weight)
age	age
* lbph	log(benign prostatic hyperplasia amount)
svi	seminal vesicle invasion
* lcp, log(capsular penetration)	
* gleason	Gleason score
pgg45	percentage Gleason scores 4 or 5
* lpsa	log(prostate specific antigen)

The datatypes for the variables that will be used in creating the decision tree are shown in table 2. The dependent variable is highlighted in red. Only those variables that will be used in the analysis are shown in table 2.

Lets print a summary of the data to get a better idea of how our variables are distributed.

```r
## Print a summary of the data
print(xtable(summary(prostateSubset), caption = "Prostate Subset Summary"),
      include.rownames = FALSE)
```

Table 2: Data Types

	variable	class	levels	examples
2	lweight	numeric		2.769 3.319, 2.691 ...
4	lbph	numeric		-1.386, -1.386, -1.386 ...
6	lcp	numeric		-1.386 -1.386, -1.386, ...
7	gleason	numeric		6, 6, 7, 6, ...
9	lpsa	numeric		-0.430, -0.162, -0.162 ...

Table 3: Prostate Subset Summary

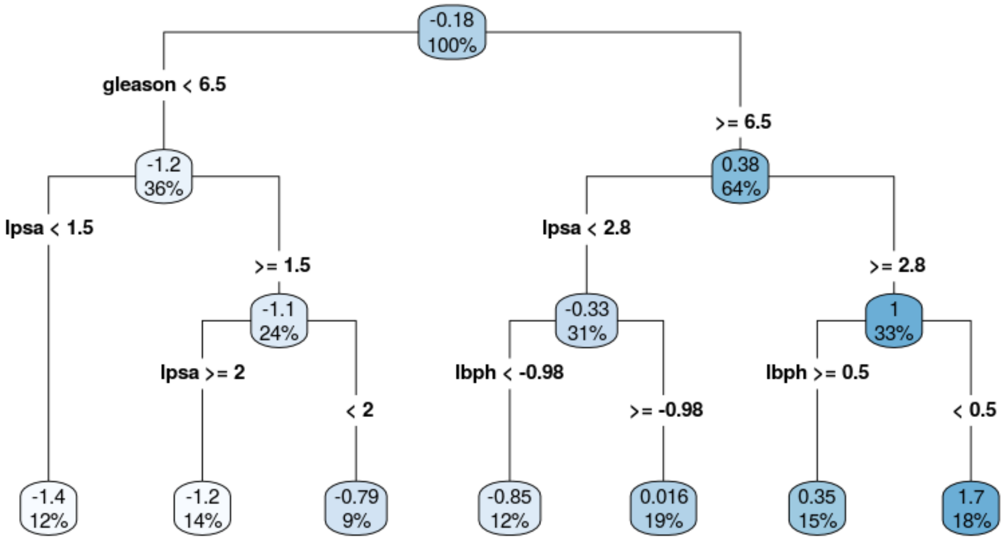
lweight	lbph	lcp	gleason	lpsa
Min. :2.375	Min. :-1.3863	Min. :-1.3863	Min. :6.000	Min. :-0.4308
1st Qu.:3.376	1st Qu.:-1.3863	1st Qu.:-1.3863	1st Qu.:6.000	1st Qu.: 1.7317
Median :3.623	Median : 0.3001	Median :-0.7985	Median :7.000	Median : 2.5915
Mean :3.653	Mean : 0.1004	Mean :-0.1794	Mean :6.753	Mean : 2.4784
3rd Qu.:3.878	3rd Qu.: 1.5581	3rd Qu.: 1.1787	3rd Qu.:7.000	3rd Qu.: 3.0564
Max. :6.108	Max. : 2.3263	Max. : 2.9042	Max. :9.000	Max. : 5.5829

Create a decision tree

To create a decision tree for predicting whether or not the cancer has spread outside the prostate (lcp) the following *R* code was used:

```
library(rpart)
library(rpart.plot)
## Now create/grow the tree
## Since lcp is continuous ranging from -1.3863 to 2.9042
## we'll use a control point of 0.0
dt <- rpart(lcp ~ lweight + lbph + gleason + lpsa, data = prostateSubset,
            control = rpart.control(cp = 0.0))
## Now print the decision tree
rpart.plot::rpart.plot(dt, type = 4)
```

Figure 1: Decision Tree



```
Call:
rpart(formula = lcp ~ lweight + lbph + gleason + lpsa, data = prostateSubset,
    control = rpart.control(cp = 0))
  n= 97

           CP nsplit rel error    xerror      xstd
```

```
1 0.287761884      0 1.0000000 1.0100242 0.1049737
2 0.157270357      1 0.7122381 0.9311028 0.1171615
3 0.074627049      2 0.5549678 0.7996757 0.1158883
4 0.028844280      3 0.4803407 0.7021377 0.1053865
5 0.005079222      4 0.4514964 0.6759693 0.1043335
6 0.000000000      6 0.4413380 0.6730510 0.1029787

Variable importance
   lpsa gleason    lbph lweight
     41      36      14      10

Node number 1: 97 observations,    complexity param=0.2877619
  mean=-0.1793656, MSE=1.934946
  left son=2 (35 obs) right son=3 (62 obs)
  Primary splits:
      gleason < 6.5        to the left,  improve=0.28776190, (0 missing)
      lpsa    < 2.847795   to the left,  improve=0.27998210, (0 missing)
      lweight < 3.355056   to the left,  improve=0.05829831, (0 missing)
      lbph    < 2.040693   to the right, improve=0.02812798, (0 missing)
  Surrogate splits:
      lpsa < 2.066682   to the left,  agree=0.794, adj=0.429, (0 split)

Node number 2: 35 observations,    complexity param=0.005079222
  mean=-1.172511, MSE=0.3564017
  left son=4 (12 obs) right son=5 (23 obs)
  Primary splits:
      lpsa    < 1.469912   to the left,  improve=0.06690534, (0 missing)
      lweight < 3.347753   to the left,  improve=0.03306504, (0 missing)
      lbph    < -1.092401  to the right, improve=0.01534593, (0 missing)
  Surrogate splits:
      lweight < 3.613572   to the left,  agree=0.714, adj=0.167, (0 split)

Node number 3: 62 observations,    complexity param=0.1572704
  mean=0.3812811, MSE=1.954932
  left son=6 (30 obs) right son=7 (32 obs)
  Primary splits:
      lpsa    < 2.833001   to the left,  improve=0.24353660, (0 missing)
      lweight < 3.189355   to the left,  improve=0.08617635, (0 missing)
      lbph    < 1.94384    to the right, improve=0.03940827, (0 missing)
  Surrogate splits:
      lweight < 3.517497   to the left,  agree=0.645, adj=0.267, (0 split)
      lbph    < -0.5537256 to the left,  agree=0.532, adj=0.033, (0 split)

Node number 4: 12 observations
  mean=-1.386294, MSE=0

Node number 5: 23 observations,    complexity param=0.005079222
  mean=-1.060972, MSE=0.5060643
  left son=10 (14 obs) right son=11 (9 obs)
  Primary splits:
      lpsa    < 1.966231   to the right, improve=0.09210508, (0 missing)
      lbph    < -1.092401  to the right, improve=0.06691219, (0 missing)
      lweight < 3.65838    to the right, improve=0.06200897, (0 missing)
  Surrogate splits:
      lweight < 3.458693   to the right, agree=0.696, adj=0.222, (0 split)

Node number 6: 30 observations,    complexity param=0.02884428
  mean=-0.3313459, MSE=1.433503
  left son=12 (12 obs) right son=13 (18 obs)
  Primary splits:
      lbph    < -0.9830564 to the left,  improve=0.12588690, (0 missing)
      lweight < 3.401163   to the left,  improve=0.06387888, (0 missing)
      lpsa    < 2.612773   to the right, improve=0.02068067, (0 missing)
  Surrogate splits:
      lweight < 3.005655   to the left,  agree=0.733, adj=0.333, (0 split)
      lpsa    < 1.434446   to the left,  agree=0.733, adj=0.333, (0 split)

Node number 7: 32 observations,    complexity param=0.07462705
  mean=1.049369, MSE=1.521332
  left son=14 (15 obs) right son=15 (17 obs)
  Primary splits:
      lbph    < 0.4989354  to the right, improve=0.28771530, (0 missing)
      lpsa    < 3.523388   to the left,  improve=0.08441155, (0 missing)
      lweight < 3.63538    to the right, improve=0.04055550, (0 missing)
  Surrogate splits:
      lweight < 3.63538    to the right, agree=0.688, adj=0.333, (0 split)
      lpsa    < 2.993028   to the left,  agree=0.688, adj=0.333, (0 split)

Node number 10: 14 observations
  mean=-1.234074, MSE=0.09139957

Node number 11: 9 observations
  mean=-0.7917025, MSE=1.031981

Node number 12: 12 observations
  mean=-0.8516236, MSE=0.5698761

Node number 13: 18 observations
  mean=0.01550589, MSE=1.708489

Node number 14: 15 observations
  mean=0.3450452, MSE=1.462536

Node number 15: 17 observations
  mean=1.670831, MSE=0.7492848
```

Note that *lweight* is the least important predictive variable and was not shown in the graphic.

Next I'll validate the model and see if we should do some pruning.

```
print(xtable(printcp(dt)),include.rownames = FALSE)
```

Table 4: printcp()

CP	nsplit	rel error	xerror	xstd
0.29	0.00	1.00	1.01	0.10
0.16	1.00	0.71	0.93	0.12
0.07	2.00	0.55	0.80	0.12
0.03	3.00	0.48	0.70	0.11
0.01	4.00	0.45	0.68	0.10
0.00	6.00	0.44	0.67	0.10

Now I should use the CP which generates the least *xerror*. It is easy to see that the **CP** value to use is *0.0*, lucky guess on my part.

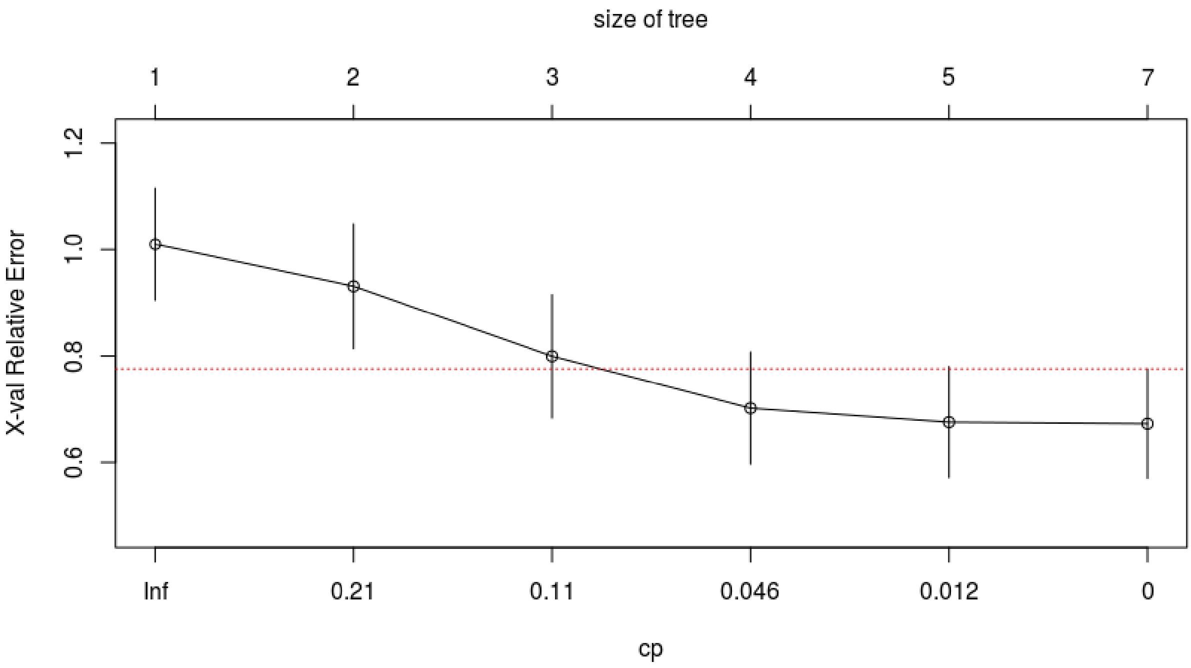
If the **CP** table is very large I could use the below to find the value of **CP** to use to prune the tree.

```
dt$cptable[which.min(dt$cptable[,"xerror"]),"CP"]
[1] 0
```

Which returns the same value that we found by visually examining the **CP** table and the value that was originally used, so no further pruning is required. I can also plot the **CP** table to visualize the deviation until the minimum error is calculated.

```
plotcp(dt, col = "red")
```

Figure 2: Plot Cutoff Point



Just for fun

Since we're really not concerned with *how invasive* the cancer is, just that it has spread outside of the prostate, I'll change the *lcp* variable to a factor and redo the calculations. We are trying to predict the probability of whether or not the cancer has spread outside the prostate indicated by an *lcp* score greater 0. So I'll make then *lcp* score in the dataset *Invasive* for scores greater than 0, and *Non-Invasive* for scores less than, or equalto, 0.

I'll also do cross-validation and print the new decision tree.

```
pS1 <- prostateSubset
pS1$lcp <- sapply(pS1$lcp, function(x) if( x > 0) {"Invasive"} else{ if(x <=0)"Non-Invasive"})
pS1$lcp <- as.factor(pS1$lcp)
set.seed(2016)
dtTrain <- sample(1:nrow(pS1), 0.8 * nrow(pS1))
trainDT <- rpart(lcp ~ ., data = pS1[dtTrain,], method = "class")
rpart.plot::rpart.plot(trainDT)
```

The next code segment preforms the prediction, runs the test, and calculates the accuracy.

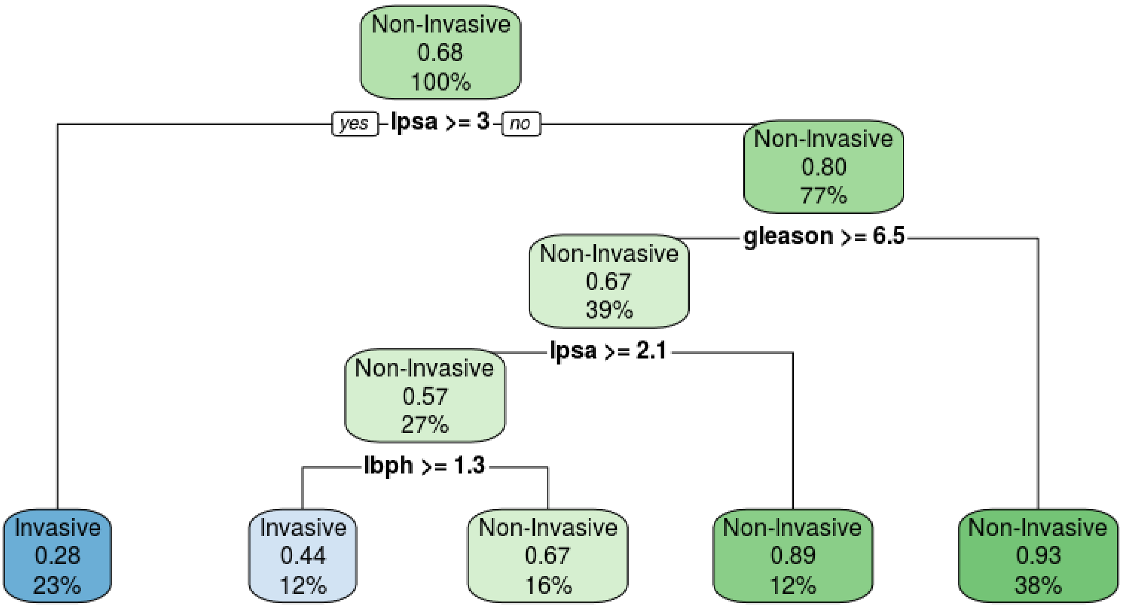
```
prostatePredict <- predict(trainDT, pS1[-dtTrain,], type = "class")
pTable <- table(prostatePredict, pS1[-dtTrain,]$lcp)
print(xtable(pTable, caption = "Cross Table"))
sum(diag(pTable))/sum(pTable)
[1] 0.85
```

Table 5: Cross Table

	Invasive	Non-Invasive
Invasive	11	1
Non-Invasive	2	6

From the cross table, and the cross table calculations, we can estimate the accuracy of the model at 85%.

Figure 3: Decision Tree Modified



And here is the graphical representation of the model. I find that this version is much easier for me to read than the previous version.