Statistical Learning Methods Exercise 9

Tobias Famos

Preliminaries Task 1 - 3

Open and take a look at the `Low Weight` Dataset

```
low_weight.raw <- read.table("/home/tobias/unibe/statistical methods in R/Exercise9/LowWeight.txt", hea
summary(low_weight.raw)
```

```
##        id            low_bw            age        mother_weight
##  Min.   :  4.0   Min.   :0.0000   Min.   :14.00   Min.   : 80.0
##  1st Qu.: 68.0   1st Qu.:0.0000   1st Qu.:19.00   1st Qu.:110.0
##  Median :123.0   Median :0.0000   Median :23.00   Median :121.0
##  Mean   :121.1   Mean   :0.3122   Mean   :23.24   Mean   :129.8
##  3rd Qu.:176.0   3rd Qu.:1.0000   3rd Qu.:26.00   3rd Qu.:140.0
##  Max.   :226.0   Max.   :1.0000   Max.   :45.00   Max.   :250.0
##       race       smoking_status   premat_labour     hypertension
##  Min.   :1.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :1.000   Median :0.0000   Median :0.0000   Median :0.00000
##  Mean   :1.847   Mean   :0.3915   Mean   :0.1958   Mean   :0.06349
##  3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.00000
##  Max.   :3.000   Max.   :1.0000   Max.   :3.0000   Max.   :1.00000
##  uterine_irrit        visits        birth_weight
##  Min.   :0.0000   Min.   :0.0000   Min.   : 709
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:2414
##  Median :0.0000   Median :0.0000   Median :2977
##  Mean   :0.1481   Mean   :0.7937   Mean   :2945
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:3475
##  Max.   :1.0000   Max.   :6.0000   Max.   :4990
```

```
sum(is.na(low_weight.raw))
```

```
## [1] 0
```

Remove index

```
low_weight <- subset(low_weight.raw, select=-id)
```

Task 1 Building a tree

Import the library

```
library(tree)
```

Split the dataset into train and test (split in half)

```
set.seed(1)
n <- dim(low_weight)[1]
low_weight.train.indices <- sample(1:n, n/2)
low_weight.test <- low_weight[-low_weight.train.indices ]
```

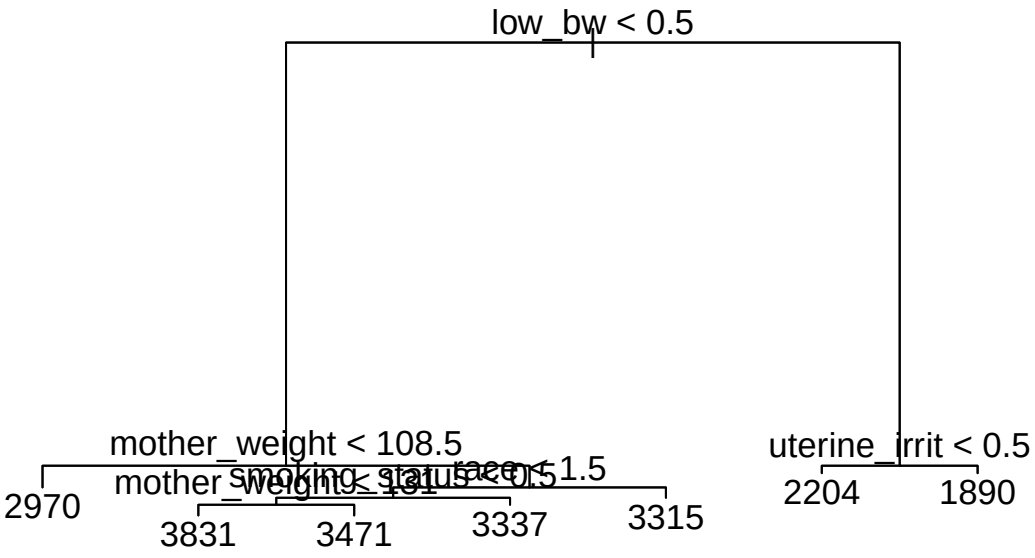
Build the regression tree

```
low_weight.tree <- tree(birth_weight ~ ., low_weight, split="deviance", subset=low_weight.train.indices)
summary(low_weight.tree)
```

```
##
## Regression tree:
## tree(formula = birth_weight ~ ., data = low_weight, subset = low_weight.train.indices,
##     split = "deviance")
## Variables actually used in tree construction:
## [1] "low_bw"         "mother_weight"  "race"           "smoking_status"
## [5] "uterine_irrit"
## Number of terminal nodes:  7
## Residual mean deviance:  154300 = 13420000 / 87
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1181.00  -240.90    28.61     0.00   206.10  1027.00
```

Plot the initial Tree

```
plot(low_weight.tree)
text(low_weight.tree, pretty=0, cex=1.1)
```



Task 2

Calculate the train MSE

```
train_labels <- low_weight[low_weight.train.indices,]$birth_weight
low_weight.tree.predictions <- predict(low_weight.tree, low_weight[low_weight.train.indices, ], type="v
mean((low_weight.tree.predictions - train_labels)^2)
```

```
## [1] 142767.3
```

Calculate the Test MSE

```
test_labels <- low_weight[-low_weight.train.indices,]$birth_weight
test_data <- low_weight[-low_weight.train.indices,]
low_weight.tree.predictions <- predict(low_weight.tree, test_data, type="vector")
mean((low_weight.tree.predictions - test_labels)^2)
```

```
## [1] 179828.6
```

As expected, the MSE in the test **179828.6** dataset is higher than the MSE on the train dataset **142767.3**

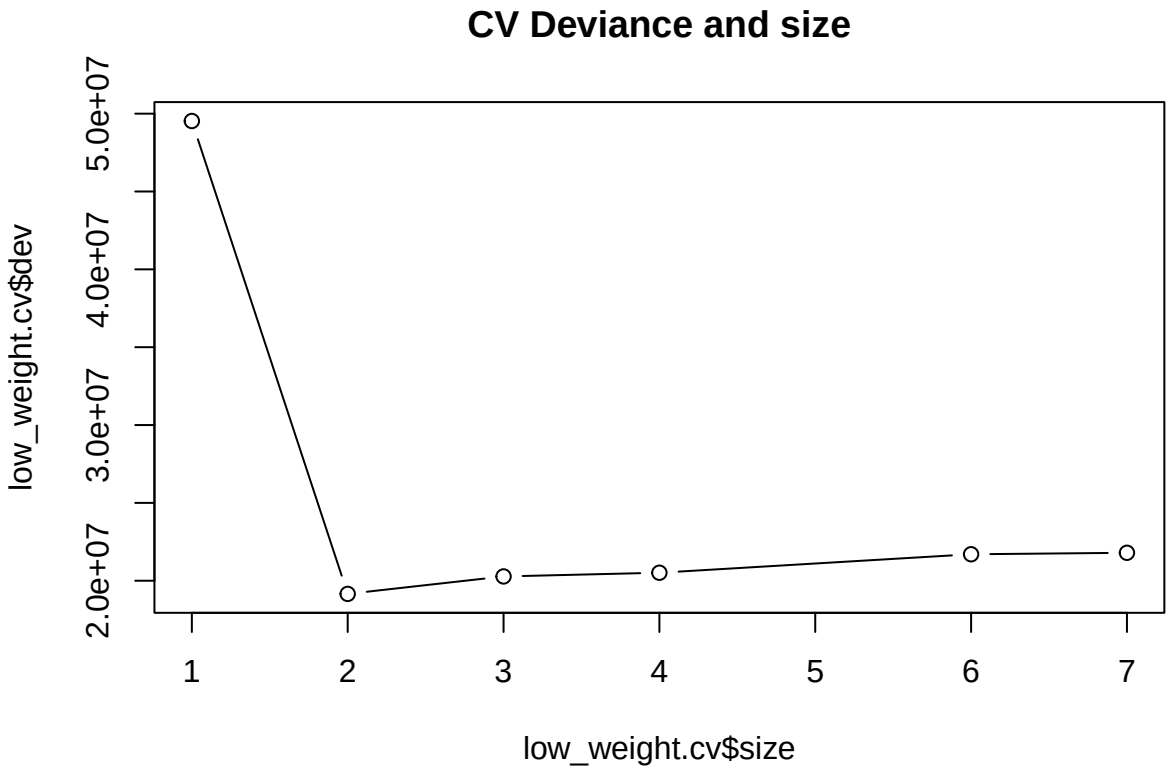
Task 3 Pruning

As the tree has a higher MSE on the test Dataset than on the train dataset it might still have some overfitting. Thus pruning can be a valid approach to reduce the test MSE Let's use a 10-fold Cross validation approach for the pruning. It is my understanding, that the cv.tree package tries different cost-complexity parameters and extrapolates the best fit

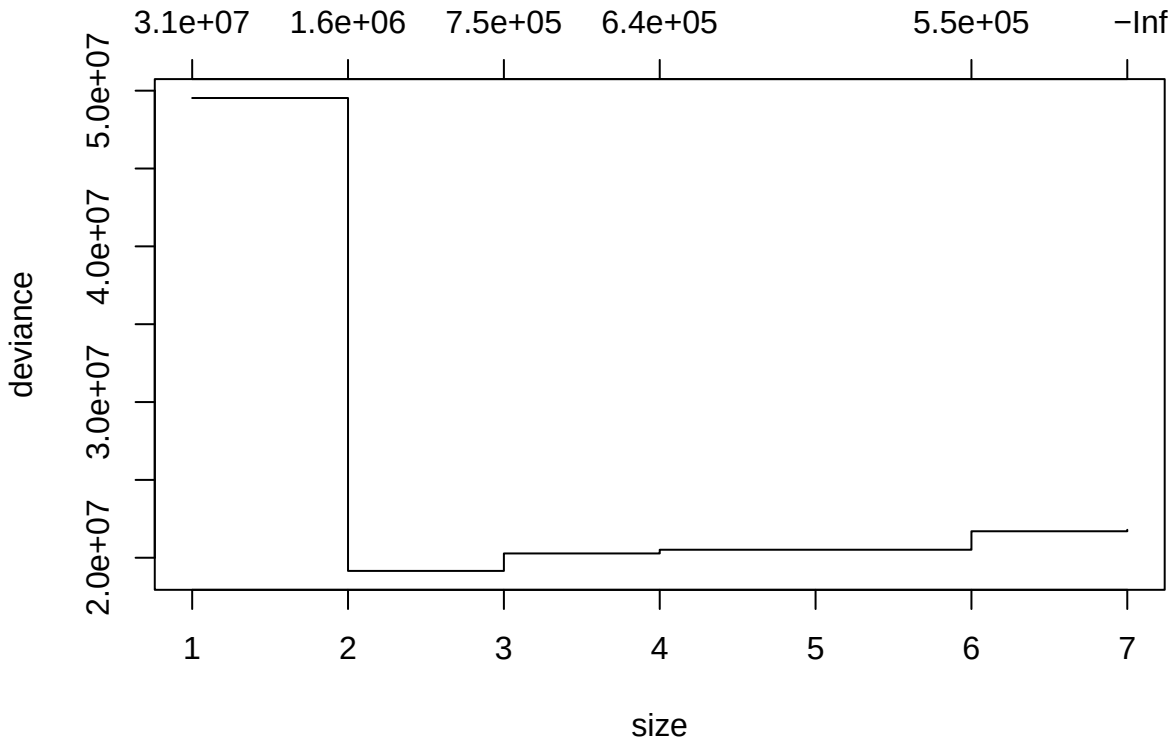
```
low_weight.cv <- cv.tree(low_weight.tree, K=10)
```

Now lets plot the tree size and deviance to get a better picture

```
plot(low_weight.cv$size, low_weight.cv$dev, main="CV Deviance and size", type="b")
```

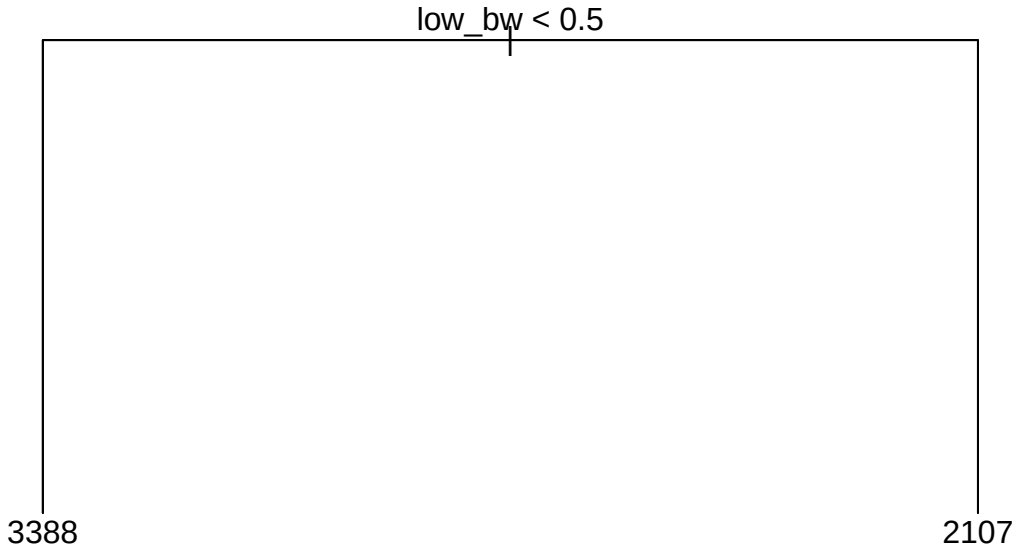



```
plot(low_weight.cv)
```



As we can clearly see, the smallest deviance is with the size 2

```
low_weight.pruned <- prune.tree(low_weight.tree, best=2)
plot(low_weight.pruned)
text(low_weight.pruned, pretty=0)
```




```
pruned.predictions <- predict(low_weight.pruned , test_data, type="vector")
mean((pruned.predictions - test_labels)^2)
```

```
## [1] 222689.6
```

Somehow, the pruning did not bring any benefit, as the MSE is higher now on based on the pruned decision tree on the test set as with the unpruned deicision tree on the test set.

Preliminaries Task 4 - 6

Read the heart data

```
heart.raw <- read.table("/home/tobias/unibe/statistical methods in R/Exercise9/Heart.txt", header = T, a
heart.raw$disease <- factor(heart.raw$disease, levels=c(1,2), labels=c('Absent', 'Present'))
str(heart.raw)
```

```
## 'data.frame':    272 obs. of  18 variables:
##  $ ID         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ age        : int  70 67 57 64 74 65 56 59 60 63 ...
##  $ sex        : int  1 0 1 1 0 1 1 1 1 0 ...
##  $ pain       : int  4 3 2 4 2 4 3 4 4 4 ...
##  $ pres       : int  130 115 124 128 120 120 130 110 140 150 ...
##  $ cholesterol: int  322 564 261 263 269 177 256 239 293 407 ...
##  $ sugar      : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ electro    : int  2 2 0 0 2 0 2 2 2 2 ...
##  $ gramstein  : num  10.4 9.6 10.9 9.5 7.3 10.1 9.7 10.2 11 8.8 ...
##  $ rate       : int  109 160 141 105 121 140 142 142 170 154 ...
##  $ angina     : int  0 0 0 1 1 0 1 1 0 0 ...
##  $ fiss       : int  24 30 26 27 27 26 19 24 31 38 ...
##  $ peak       : num  2.4 1.6 0.3 0.2 0.2 0.4 0.6 1.2 1.2 4 ...
##  $ slope      : int  2 2 1 2 1 1 2 2 2 2 ...
##  $ vessels    : int  3 0 0 1 1 0 1 1 2 3 ...
##  $ thal       : int  3 7 7 7 3 7 6 7 7 7 ...
##  $ blst       : num  73.7 57.6 69.5 52.4 71.4 ...
##  $ disease    : Factor w/ 2 levels "Absent","Present": 2 1 2 1 1 1 2 2 2 2 ...
```

```
summary(heart.raw)
```

```
##        ID              age              sex              pain       
##  Min.   :  1.00   Min.   : 3.00   Min.   :0.0000   Min.   :1.000  
##  1st Qu.: 68.75   1st Qu.:47.75   1st Qu.:0.0000   1st Qu.:3.000  
##  Median :136.50   Median :55.00   Median :1.0000   Median :3.000  
##  Mean   :136.50   Mean   :54.24   Mean   :0.6765   Mean   :3.173  
##  3rd Qu.:204.25   3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:4.000  
##  Max.   :272.00   Max.   :77.00   Max.   :1.0000   Max.   :4.000  
##       pres        cholesterol        sugar           electro      
##  Min.   : 94.0   Min.   :125.0   Min.   :0.0000   Min.   :0.000  
##  1st Qu.:120.0   1st Qu.:212.8   1st Qu.:0.0000   1st Qu.:0.000  
##  Median :130.0   Median :245.0   Median :0.0000   Median :2.000  
##  Mean   :131.3   Mean   :249.3   Mean   :0.1471   Mean   :1.029  
##  3rd Qu.:140.0   3rd Qu.:278.0   3rd Qu.:0.0000   3rd Qu.:2.000  
##  Max.   :200.0   Max.   :564.0   Max.   :1.0000   Max.   :2.000  
##    gramstein           rate           angina           fiss      
##  Min.   :-4.500   Min.   : 71.0   Min.   :0.0000   Min.   :11.00  
##  1st Qu.: 9.300   1st Qu.:132.8   1st Qu.:0.0000   1st Qu.:22.00  
##  Median :10.100   Median :153.5   Median :0.0000   Median :25.00  
##  Mean   : 9.975   Mean   :149.6   Mean   :0.3346   Mean   :24.94  
##  3rd Qu.:10.700   3rd Qu.:166.0   3rd Qu.:1.0000   3rd Qu.:28.00  
##  Max.   :13.300   Max.   :202.0   Max.   :1.0000   Max.   :39.00  
##       peak          slope          vessels            thal       
##  Min.   :0.00   Min.   :1.000   Min.   :0.0000   Min.   :3.000  
##  1st Qu.:0.00   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:3.000  
##  Median :0.80   Median :2.000   Median :0.0000   Median :3.000  
##  Mean   :1.05   Mean   :1.588   Mean   :0.6765   Mean   :4.713  
```

```
##  3rd Qu.:1.65   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:7.000
##  Max.   :6.20   Max.   :3.000   Max.   :3.0000   Max.   :7.000
##       blst          disease
##  Min.   :50.14   Absent :150
##  1st Qu.:57.50   Present:122
##  Median :66.01
##  Mean   :65.28
##  3rd Qu.:71.88
##  Max.   :79.77
```

```
sum(is.na(heart.raw))
```

```
## [1] 0
```

```
heart <- subset(heart.raw, select=-ID)
```

Task 3 Build a decision tree for classification

Splitting into test and train dataset

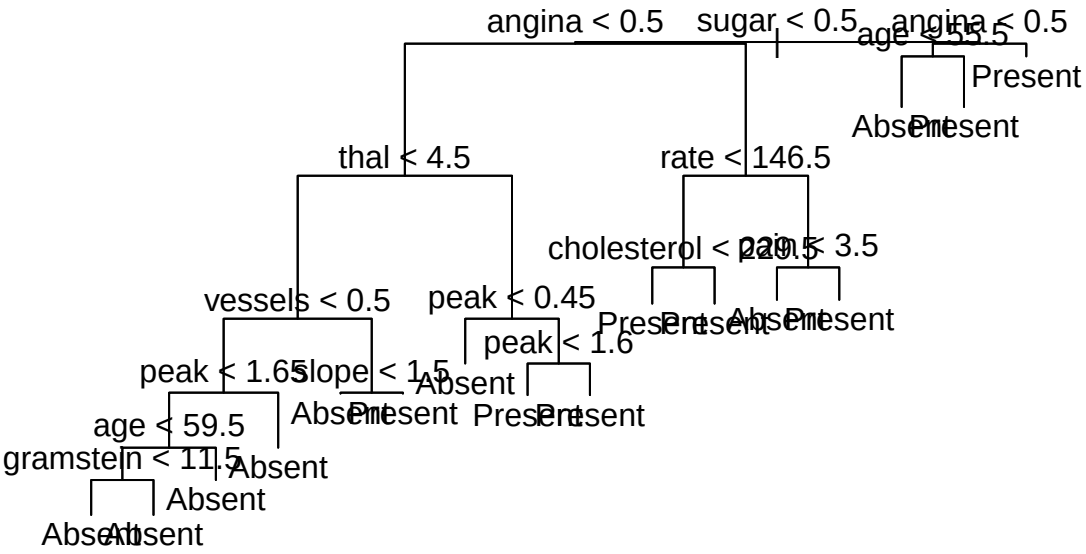
```
set.seed(1)
n <- dim(heart)[1]
heart.train.indices <- sample(1:n, n/2)
heart.test <- heart[-heart.train.indices, ]
heart.train <- heart[heart.train.indices, ]
heart.train.labels <- heart.train$disease
heart.test.labels <- heart.test$disease
```

Build the decision tree on the train dataset

```
heart.tree <- tree(disease ~ ., heart, split="gini", method="deviance", subset=heart.train.indices)
summary(heart.tree)
```

```
##
## Classification tree:
## tree(formula = disease ~ ., data = heart, subset = heart.train.indices,
##     method = "deviance", split = "gini")
## Variables actually used in tree construction:
##  [1] "sugar"       "angina"      "thal"        "vessels"     "peak"
##  [6] "age"         "gramstein"   "slope"       "rate"        "cholesterol"
## [11] "pain"
## Number of terminal nodes:  16
## Residual mean deviance:  0.7619 = 91.43 / 120
## Misclassification error rate: 0.1838 = 25 / 136
```

```
plot(heart.tree)
text(heart.tree, pretty=0)
```



Task 4 Confustion matrix

```
pred.train <- predict(heart.tree ,heart.train, type="class")
pred.test <- predict(heart.tree ,heart.test, type="class")

train.table <- table(pred.train, heart.train.labels, dnn = c('Predicted Disease','Actual Disease'))
test.table <- table(pred.test, heart.test.labels, dnn = c('Predicted Disease','Actual Disease'))
test.table
```

```
##                  Actual Disease
## Predicted Disease Absent Present
##           Absent      40       9
##           Present     32      55
```

```
train.table
```

```
##                  Actual Disease
## Predicted Disease Absent Present
##           Absent      62      12
##           Present     16      46
```

Calculating the Accuracy, sensitivity and specivity First extract the base values for the train and test datasets for TP, TN, FP, FN

```
train.tp <- train.table[2,2]
test.tp <- test.table[2,2]

train.tn <- train.table[1,1]
test.tn <- test.table[1,1]

train.fn <- train.table[1,2]
test.fn <- test.table[1,2]

train.fp <- train.table[2,1]
test.fp <- test.table[2,1]
```

Accuracy

```
Accuracy = (TN + TP)/N
```

```
train.accuracy <- (train.tn + train.tp) / (train.tn + train.tp + train.fn + train.fp)
train.accuracy
```

```
## [1] 0.7941176
```

```
test.accuracy <- (test.tn + test.tp) / (test.tn + test.tp + test.fn + test.fp)
test.accuracy
```

```
## [1] 0.6985294
```

Sensitivity

```
Sensitivity = TP/(TP + FN)
```

```
train.sensitivity <- train.tp / (train.tp + train.fn)
train.sensitivity
```

```
## [1] 0.7931034
```

```
test.sensitivity <- test.tp / (test.tp + test.fn)
test.sensitivity
```

```
## [1] 0.859375
```

Specificity

```
Specificity = TN/(TN + FP)
```

```
train.specificity <- train.tn / (train.tn + train.fp)
train.specificity
```

```
## [1] 0.7948718
```

```
test.specificity <- test.tn / (test.tn + test.fp)
test.specificity
```

```
## [1] 0.5555556
```

Analysis

The test Accuracy is quite high and also the train Accuracy of 70% is not too bad for new data. Quite remarkable is that the sensitivity on the test data is higher than on the train data. This implies that the model might be better on classifying the disease (if it exists) on new data than on already seen. But, the specificity is quite low on the test dataset. This means that in many cases, when the disease is not present, a patient is still marked as having the disease.

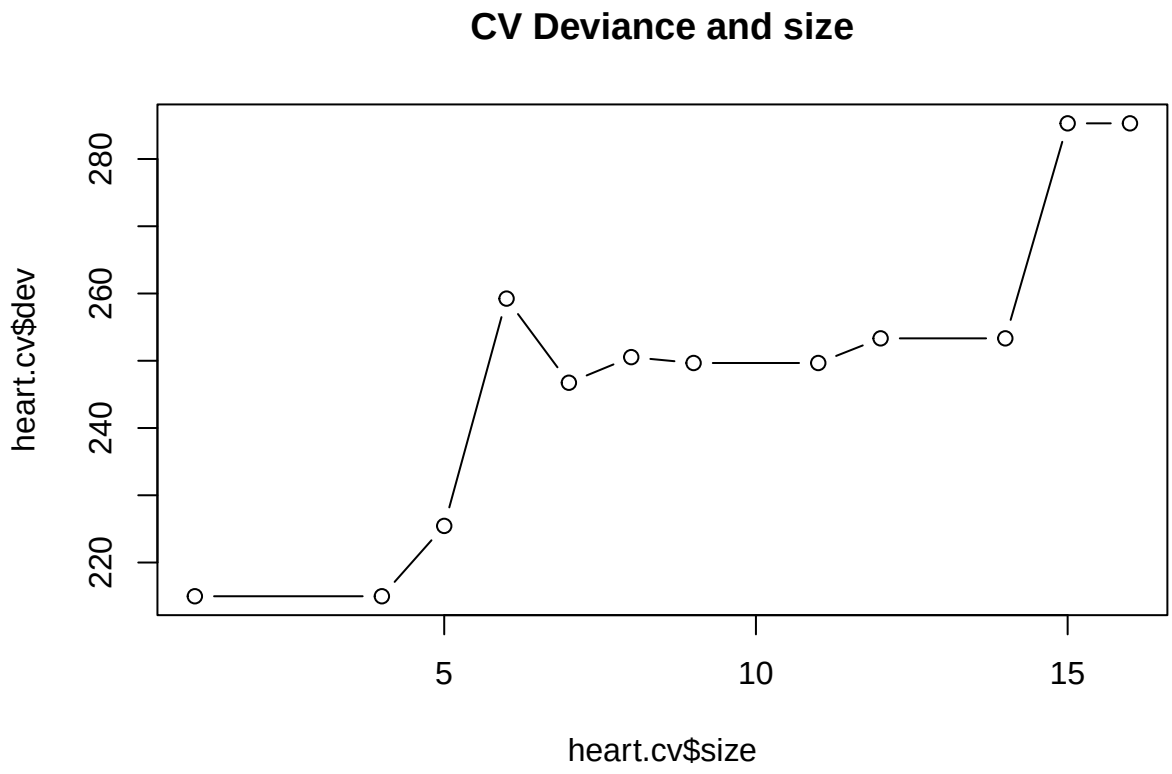
Task 6 Pruning

A little pruning might be a good thing. If one only looks at the plot of the graph we see a lot of potential for simplification as many decisions on internal node of the last level lead to terminal nodes of the same class

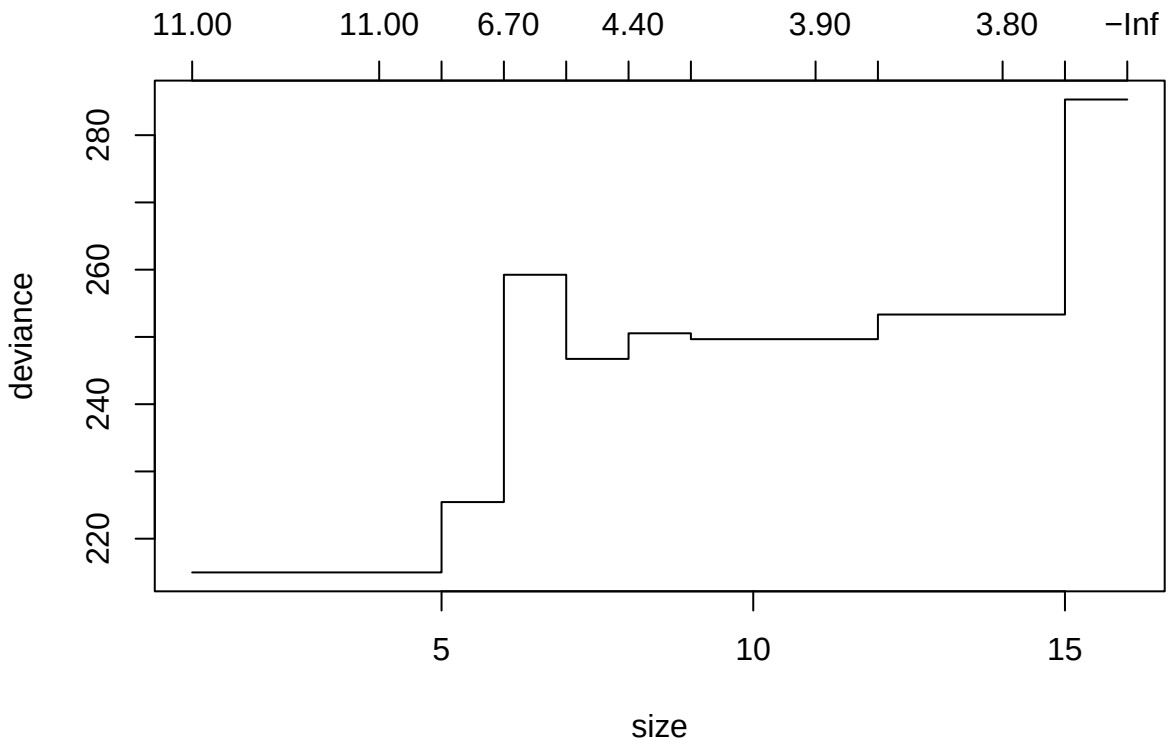
```
heart.cv <- cv.tree(heart.tree, K=10)
```

Now lets plot the tree size and deviance to get a better picture

```
plot(heart.cv$size, heart.cv$dev, main="CV Deviance and size", type="b")
```

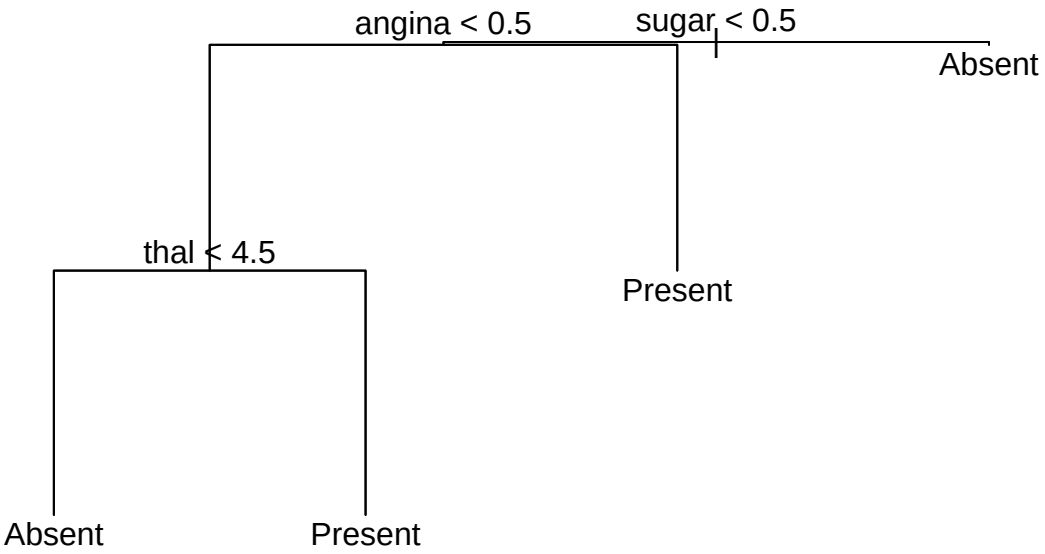



```
plot(heart.cv)
```



The best size would be 4.

```
heart.pruned <- prune.tree(heart.tree, best=4)
plot(heart.pruned)
text(heart.pruned, pretty=0)
```




```
prune.pred.train <- predict(heart.pruned ,heart.train, type="class")
prune.pred.test <- predict(heart.pruned ,heart.test, type="class")

prune.train.table <- table(prune.pred.train, heart.train.labels, dnn = c('Predicted Disease','Actual Di
prune.test.table <- table(prune.pred.test, heart.test.labels, dnn = c('Predicted Disease','Actual Disea
prune.test.table
```

```
##                  Actual Disease
## Predicted Disease Absent Present
##           Absent      51      15
##           Present     21      49
```

```
prune.train.table
```

```
##                  Actual Disease
## Predicted Disease Absent Present
##           Absent      61      18
##           Present     17      40
```

Comparison

Calculating the Accuracy, sensitivity and specivity First extract the base values for the train and test datasets for TP, TN, FP, FN

```
train.tp.prune <- prune.train.table[2,2]
test.tp.prune <- prune.test.table[2,2]

train.tn.prune <- prune.train.table[1,1]
test.tn.prune <- prune.test.table[1,1]

train.fn.prune <- prune.train.table[1,2]
test.fn.prune <- prune.test.table[1,2]

train.fp.prune <- prune.train.table[2,1]
test.fp.prune <- prune.test.table[2,1]
```

Accuracy

`Accuracy = (TN + TP)/N`

```
train.accuracy.prune <- (train.tn.prune + train.tp.prune) / (train.tn.prune + train.tp.prune + train.fn
train.accuracy.prune
```

```
## [1] 0.7426471
```

```
test.accuracy.prune <- (test.tn.prune + test.tp.prune) / (test.tn.prune + test.tp.prune + test.fn.prune
test.accuracy.prune
```

```
## [1] 0.7352941
```

Sensitivity

`Sensitivity = TP/(TP + FN)`

```
train.sensitivity.prune <- train.tp.prune / (train.tp.prune + train.fn.prune)
train.sensitivity.prune
```

```
## [1] 0.6896552
```

```
test.sensitivity.prune <- test.tp.prune / (test.tp.prune + test.fn.prune)
test.sensitivity.prune
```

```
## [1] 0.765625
```

Specificity

`Specificity = TN/(TN + FP)`

```
train.specificity.prune <- train.tn.prune / (train.tn.prune + train.fp.prune)
train.specificity.prune
```

```
## [1] 0.7820513
```

```
test.specificity.prune <- test.tn.prune / (test.tn.prune + test.fp.prune)
test.specificity.prune
```

```
## [1] 0.7083333
```

It is worth noting, that the values have all decreased for the train dataset and increased for the test dataset. Thus we have reached a higher level of generalization (or a lower level of overfitting)