Trees

A simple example of a decision tree in R using the package Caret.

```
# Define columnn class for dataset
colCls <- c("integer",         # row id
            "character",       # analysis year
            "numeric",         # exposure
            "character",       # new business / renewal business
            "numeric",         # driver age (continuous)
            "character",       # driver age (categorical)
            "character",       # driver gender
            "character",       # marital status
            "numeric",         # years licensed (continuous)
            "character",       # years licensed (categorical)
            "character",       # ncd level
            "character",       # region
            "character",       # body code
            "numeric",         # vehicle age (continuous)
            "character",       # vehicle age (categorical)
            "numeric",         # vehicle value
            "character",       # seats
            rep("numeric", 6), # ccm, hp, weight, length, width, height (all continuous)
            "character",       # fuel type
            rep("numeric", 3)  # prior claims, claim count, claim incurred (all continuous)
)
```

```
# Define the data path and filename
data.path <- "C:\\Users\\William.Tiritilli\\Documents\\Project P\\Frees\\Tome 2 - Chapter 1\\"
data.fn <- "sim-modeling-dataset2.csv"

# Read in the data with the appropriate column classes
dta <- read.csv(paste(data.path, data.fn, sep = "/"),
                colClasses = colCls)
str(dta)
```

```
## 'data.frame':    40760 obs. of  27 variables:
##  $ row.id        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ year          : chr  "2010" "2010" "2010" "2010" ...
##  $ exposure      : num  1 1 1 0.08 1 0.08 1 1 0.08 1 ...
##  $ nb.rb         : chr  "RB" "NB" "RB" "RB" ...
##  $ driver.age    : num  63 33 68 68 68 68 53 68 68 65 ...
##  $ drv.age       : chr  "63" "33" "68" "68" ...
##  $ driver.gender : chr  "Male" "Male" "Male" "Male" ...
##  $ marital.status: chr  "Married" "Married" "Married" "Married" ...
##  $ yrs.licensed  : num  5 1 2 2 2 2 5 2 2 2 ...
##  $ yrs.lic       : chr  "5" "1" "2" "2" ...
```

```
##  $ ncd.level     : chr  "6" "5" "4" "4" ...
##  $ region        : chr  "3" "38" "33" "33" ...
##  $ body.code     : chr  "A" "B" "C" "C" ...
##  $ vehicle.age   : num  3 3 2 2 1 1 3 1 1 5 ...
##  $ veh.age       : chr  "3" "3" "2" "2" ...
##  $ vehicle.value : num  21.4 17.1 17.3 17.3 25 ...
##  $ seats         : chr  "5" "3" "5" "5" ...
##  $ ccm           : num  1248 2476 1948 1948 1461 ...
##  $ hp            : num  70 94 90 90 85 85 70 85 85 65 ...
##  $ weight        : num  1285 1670 1760 1760 1130 ...
##  $ length        : num  4.32 4.79 4.91 4.91 4.04 ...
##  $ width         : num  1.68 1.74 1.81 1.81 1.67 ...
##  $ height        : num  1.8 1.97 1.75 1.75 1.82 ...
##  $ fuel.type     : chr  "Diesel" "Diesel" "Diesel" "Diesel" ...
##  $ prior.claims  : num  0 0 0 0 0 0 4 0 0 0 ...
##  $ clm.count     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ clm.incurred  : num  0 0 0 0 0 0 0 0 0 0 ...
```

```r
set.seed(54321) # reproducubility
# Create a stratified data partition
train_id <- caret::createDataPartition(
  y = dta$clm.count/dta$exposure,
  p = 0.8,
  groups = 100
)[[1]]
```

```r
# Divide the data in training and test set
dta_trn <- dta[train_id,]
dta_tst <- dta[-train_id,]
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Proportions of the number of claims in train data
dta_trn$clm.count %>% table %>% prop.table %>% round(5)
```

```
## .
##       0       1       2       3       4       5
## 0.92257 0.07163 0.00537 0.00037 0.00003 0.00003
```

Proportions of the number of claims in test data
```r
# Proportions of the number of claims in test data
dta_tst$clm.count %>% table %>% prop.table %>% round(5)
```

```
## .
##       0       1       2       3
## 0.92098 0.07252 0.00613 0.00037
```

Proportions in train and test set are welle balanced.

```r
with ( dta , table ( driver.gender, clm.count) )
```

```
##              clm.count
## driver.gender     0     1     2     3     4     5
##        Female  4110   365    39     4     0     1
##        Male   33481  2562   186    11     1     0
```

Fitting a simple tree to the Car data

```r
library(rpart)       # direct engine for decision tree application
library(caret)       # meta engine for decision tree application
```

```
## Warning: package 'caret' was built under R version 4.1.1

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.1.2
```

```r
fit <- rpart(formula =
               cbind(exposure,clm.count) ~
               driver.age  + hp
               + fuel.type + driver.gender + body.code + yrs.licensed,
             data = dta_trn,
             method = 'poisson',
             control = rpart.control(
               maxdepth = 3,
               cp = 0)
)

print(fit)
```

```
## n= 32610
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 32610 13257.680000 0.16469930
##    2) yrs.licensed>=4.5 8167  2491.723000 0.10910610
##      4) body.code=B,C,D,H 2509   640.461400 0.08026114
```

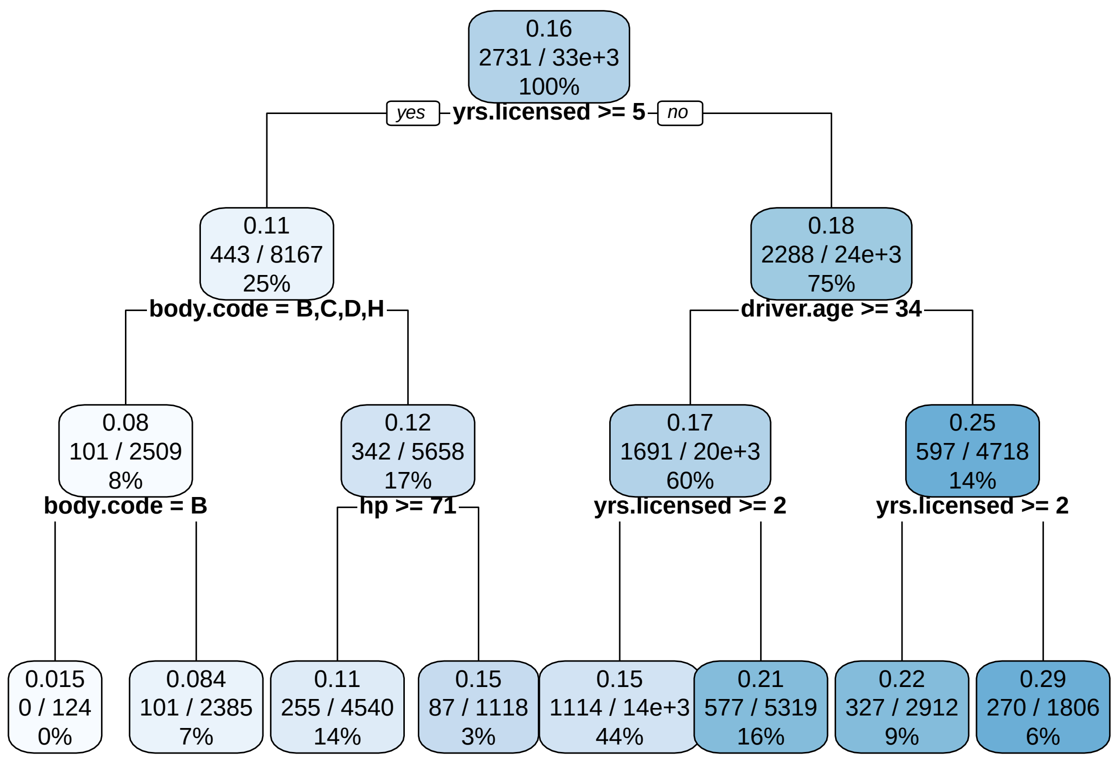
```
##        8) body.code=B 124     1.821845 0.01467100 *
##        9) body.code=C,D,H 2385   630.293000 0.08438388 *
##      5) body.code=A,E,F,G 5658  1836.083000 0.12229680
##       10) hp>=70.5 4540  1400.678000 0.11434330 *
##       11) hp< 70.5 1118   429.949900 0.15388610 *
##    3) yrs.licensed< 4.5 24443 10655.050000 0.18276260
##      6) driver.age>=33.5 19725  8136.286000 0.16741700
##       12) yrs.licensed>=1.5 14406  5570.771000 0.15211440 *
##       13) yrs.licensed< 1.5 5319  2529.809000 0.20772230 *
##      7) driver.age< 33.5 4718  2456.388000 0.24669850
##       14) yrs.licensed>=1.5 2912  1415.248000 0.22165730 *
##       15) yrs.licensed< 1.5 1806  1031.732000 0.28516660 *
```

A graph is a better way to read the informatin

```r
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.1
```

```r
rpart.plot(fit, cex = 0.75)
```



To check if the tree gives us a similar prediction: It works!

```r
dta_trn %>%
  dplyr::filter(yrs.licensed < 5,
                driver.age < 34, yrs.licensed< 2) %>%
  dplyr::summarise(claim_freq =
                     sum(clm.count)/sum(exposure))
```

```
##   claim_freq
## 1  0.2859412
```

```r
k <- 1
alpha <- 1/k^2
mu <- dta_trn %>%
  with(sum(clm.count)/sum(exposure))
beta <- alpha/mu
dta_trn %>%
  dplyr::filter(yrs.licensed < 5,
                driver.age < 34, yrs.licensed< 2) %>%
  dplyr::summarise(prediction =
                     (alpha + sum(clm.count))/ (beta + sum(exposure)))
```

```
##   prediction
## 1  0.2851666
```

```r
fit2 <- rpart(formula =
               cbind(exposure,clm.count) ~
               driver.age  + hp
               + fuel.type + driver.gender + body.code + yrs.licensed,
             data = dta_trn,
             method = 'poisson',
             control = rpart.control(
               maxdepth = 3,
               cp = 0),
             parms = list(shrink = 10^5)
)
print(fit2)
```

```
## n= 32610
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 32610 1.325768e+04 1.646993e-01
##    2) yrs.licensed>=4.5 8167 2.491723e+03 1.090231e-01
##      4) body.code=B,C,D,H 2509 6.404588e+02 7.985579e-02
##        8) body.code=B 124 2.000000e-10 1.610565e-12 *
##        9) body.code=C,D,H 2385 6.302906e+02 8.397842e-02 *
##      5) body.code=A,E,F,G 5658 1.836083e+03 1.222048e-01
##       10) hp>=70.5 4540 1.400678e+03 1.142064e-01 *
##       11) hp< 70.5 1118 4.299499e+02 1.537700e-01 *
##    3) yrs.licensed< 4.5 24443 1.065505e+04 1.827714e-01
##      6) driver.age>=33.5 19725 8.136286e+03 1.674186e-01
##       12) yrs.licensed>=1.5 14406 5.570771e+03 1.521039e-01 *
```

```
##       13) yrs.licensed< 1.5 5319 2.529809e+03 2.078163e-01 *
##      7) driver.age< 33.5 4718 2.456388e+03 2.469044e-01
##       14) yrs.licensed>=1.5 2912 1.415248e+03 2.218920e-01 *
##       15) yrs.licensed< 1.5 1806 1.031730e+03 2.859412e-01 *
```

Pruning the tree

Step 1: we start with a complex tree

str(dta)

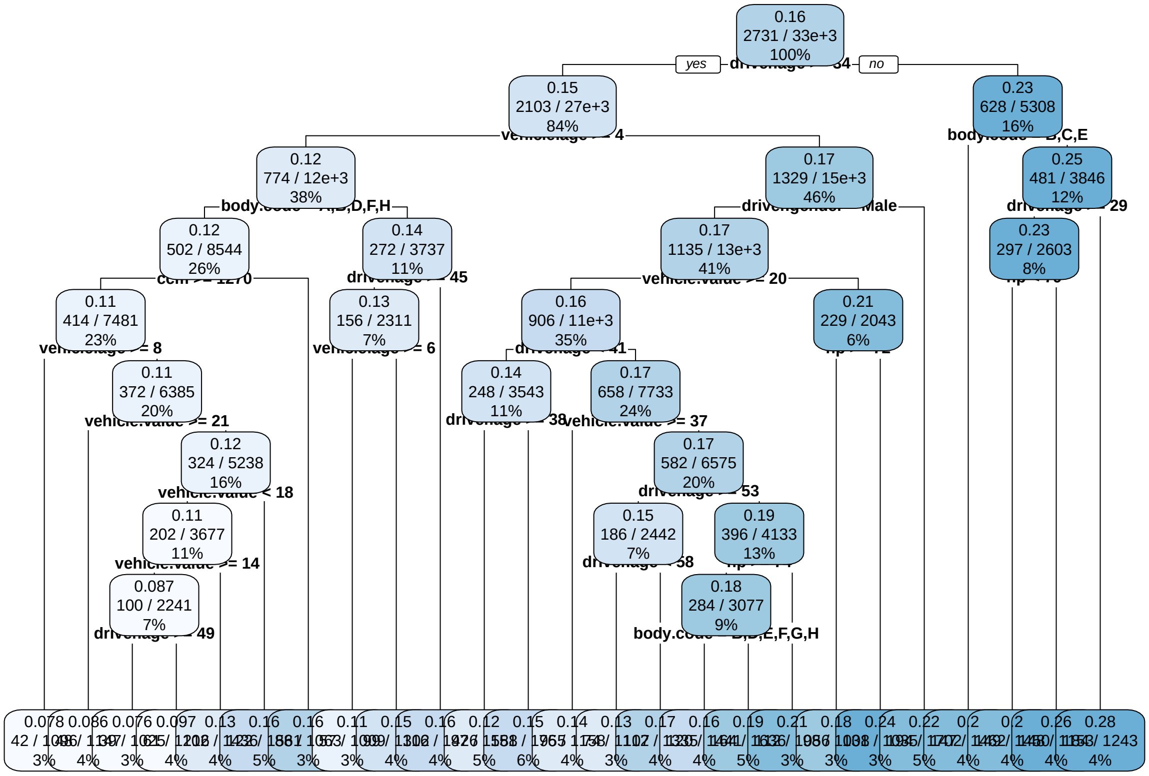
```r
set.seed(9753) # reproducibilty
fit <- rpart(formula =
               cbind(exposure,clm.count) ~
               driver.age + vehicle.age + vehicle.value + hp +
                fuel.type  + ccm + body.code + driver.gender,
             data = dta_trn,
             method = 'poisson',
             control = rpart.control(
               maxdepth = 20,
               minsplit = 2000,
               minbucket = 1000,
               cp = 0,
               xval = 5
             )
)
```

```r
print(fit)
```

```
## n= 32610
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##   1) root 32610 13257.6800 0.16469930
##     2) driver.age>=33.5 27302 10518.2200 0.15159810
##       4) vehicle.age>=3.5 12281  4135.4210 0.12424850
##         8) body.code=A,B,D,F,H 8544  2731.5010 0.11562580
##          16) ccm>=1270 7481  2311.5720 0.10888840
##            32) vehicle.age>=7.5 1096   262.8892 0.07757369 *
##            33) vehicle.age< 7.5 6385  2041.9950 0.11431190
##              66) vehicle.value>=21.075 1147   293.5474 0.08603862 *
##              67) vehicle.value< 21.075 5238  1743.0360 0.12038990
##               134) vehicle.value< 17.5935 3677  1144.3190 0.10574060
##                 268) vehicle.value>=14.225 2241   613.9341 0.08691075
##                   536) driver.age>=48.5 1025   255.6467 0.07563844 *
##                   537) driver.age< 48.5 1216   356.7168 0.09697324 *
##                 269) vehicle.value< 14.225 1436   520.5739 0.13486060 *
##               135) vehicle.value>=17.5935 1561   587.3745 0.15652010 *
##          17) ccm< 1270 1063   408.9968 0.16328130 *
##         9) body.code=C,E,G 3737  1395.5120 0.14419030
##          18) driver.age>=44.5 2311   828.2863 0.13286260
##            36) vehicle.age>=5.5 1009   324.1642 0.11303580 *
##            37) vehicle.age< 5.5 1302   501.3168 0.14822900 *
```

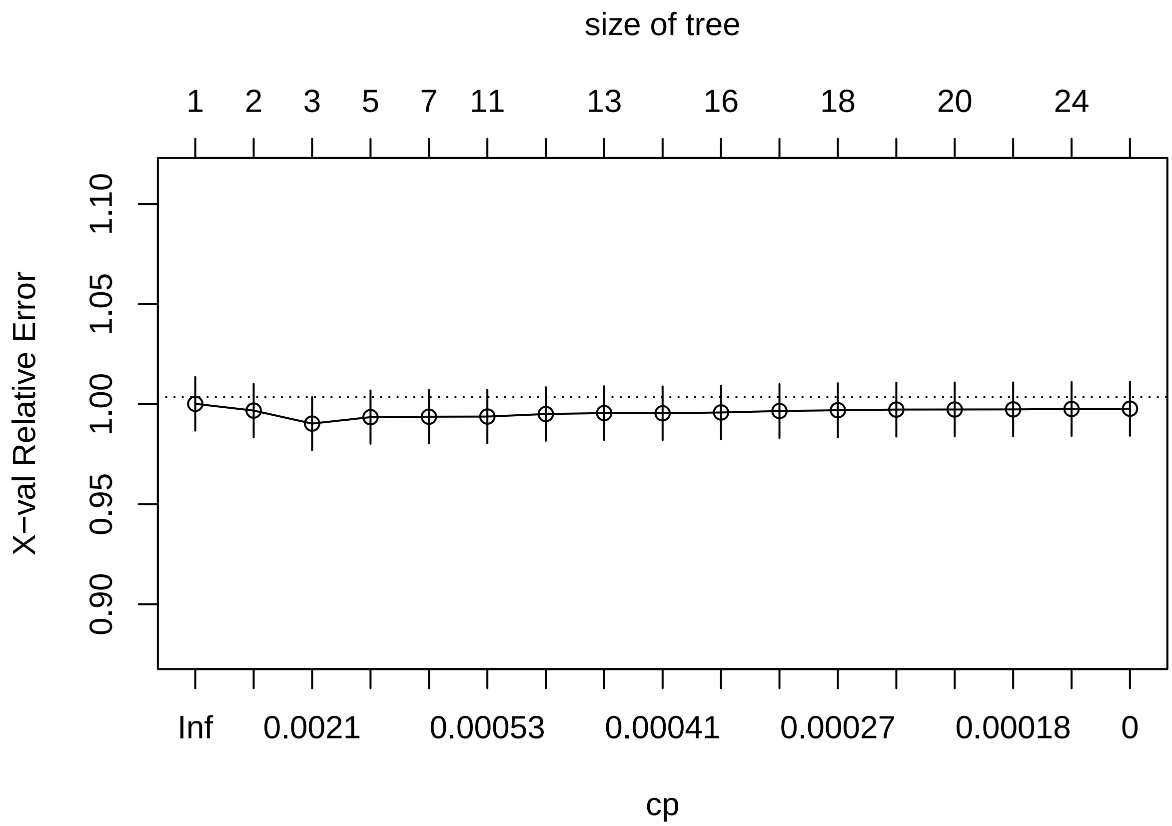
```
##          19) driver.age< 44.5 1426   564.4441 0.16301360 *
##       5) vehicle.age< 3.5 15021  6325.9490 0.17391600
##        10) driver.gender=Male 13319  5454.5660 0.16745230
##          20) vehicle.value>=20.3775 11276  4447.7560 0.15896770
##            40) driver.age< 40.5 3543  1279.6230 0.13940350
##              80) driver.age>=37.5 1588   525.7628 0.12194030 *
##              81) driver.age< 37.5 1955   750.5764 0.15375560 *
##            41) driver.age>=40.5 7733  3161.6950 0.16787880
##              82) vehicle.value>=36.806 1158   394.7900 0.13503990 *
##              83) vehicle.value< 36.806 6575  2762.3240 0.17344370
##               166) driver.age>=52.5 2442   931.6891 0.15109140
##                 332) driver.age< 57.5 1107   386.4593 0.13075960 *
##                 333) driver.age>=57.5 1335   542.2371 0.16861600 *
##               167) driver.age< 52.5 4133  1824.8740 0.18640850
##                 334) hp>=74 3077  1334.9270 0.17958170
##                   668) body.code=B,D,E,F,G,H 1441   590.4866 0.16205900 *
##                   669) body.code=A,C 1636   742.0253 0.19493140 *
##                 335) hp< 74 1056   488.4328 0.20591070 *
##          21) vehicle.value< 20.3775 2043   992.3633 0.21207310
##            42) hp>=72 1008   441.4296 0.18437410 *
##            43) hp< 72 1035   547.1302 0.23841710 *
##        11) driver.gender=Female 1702   858.0214 0.22428700 *
##     3) driver.age< 33.5 5308  2658.8920 0.23166980
##       6) body.code=B,C,E 1462   664.2686 0.19522960 *
##       7) body.code=A,D,F,G,H 3846  1988.4820 0.24553480
##        14) driver.age>=28.5 2603  1275.9330 0.22661740
##          28) hp< 78.5 1450   657.1034 0.20194090 *
##          29) hp>=78.5 1153   614.3514 0.25766440 *
##        15) driver.age< 28.5 1243   706.9522 0.28281340 *
```

```r
rpart.plot(fit, cex = 0.5)
```



Step 2: Inspect the cross-validation results

```
plotcp(fit)
```



Breiman (1984) suggested that in actual practice, it's common to instead use the smallest tree within 1 standard error (SE) of the minimum CV error (this is called the 1-SE rule). Here it looks like taking a tree with 3 terminal nodes will give similar results within a small margin of error.

Step 3: chose the cp value (cp: complexity parameter)

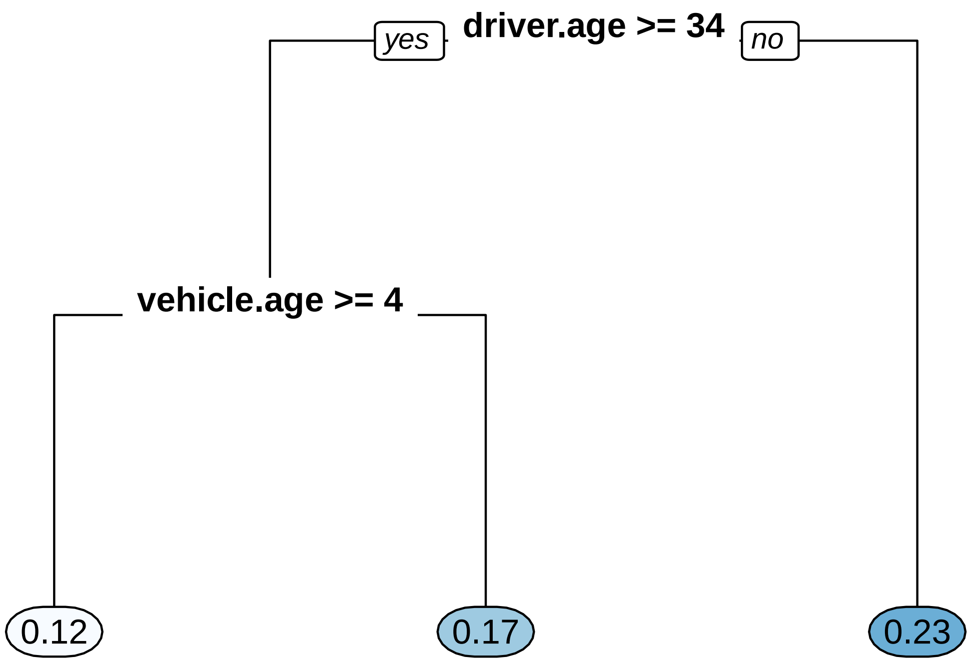
```
# Get the cross-validation results
cpt <- fit$cptable

# Look for the minimal xerror
min_xerr <- which.min(cpt[,'xerror'])
cpt[min_xerr,]
```

```
##          CP      nsplit   rel error      xerror        xstd
## 0.001048774 2.000000000 0.989635131 0.990310664 0.013240434
```

```
# Prune the tree
fit_srt <- prune(fit,
                 cp = cpt[min_xerr, 'CP'])
```

Step 4: plot the final tree

```
# Plot the tree
rpart.plot(fit_srt, type = 0, extra = 0, cex = 1.1)
```



The tree has been pruned.

#Making sense of a tree model

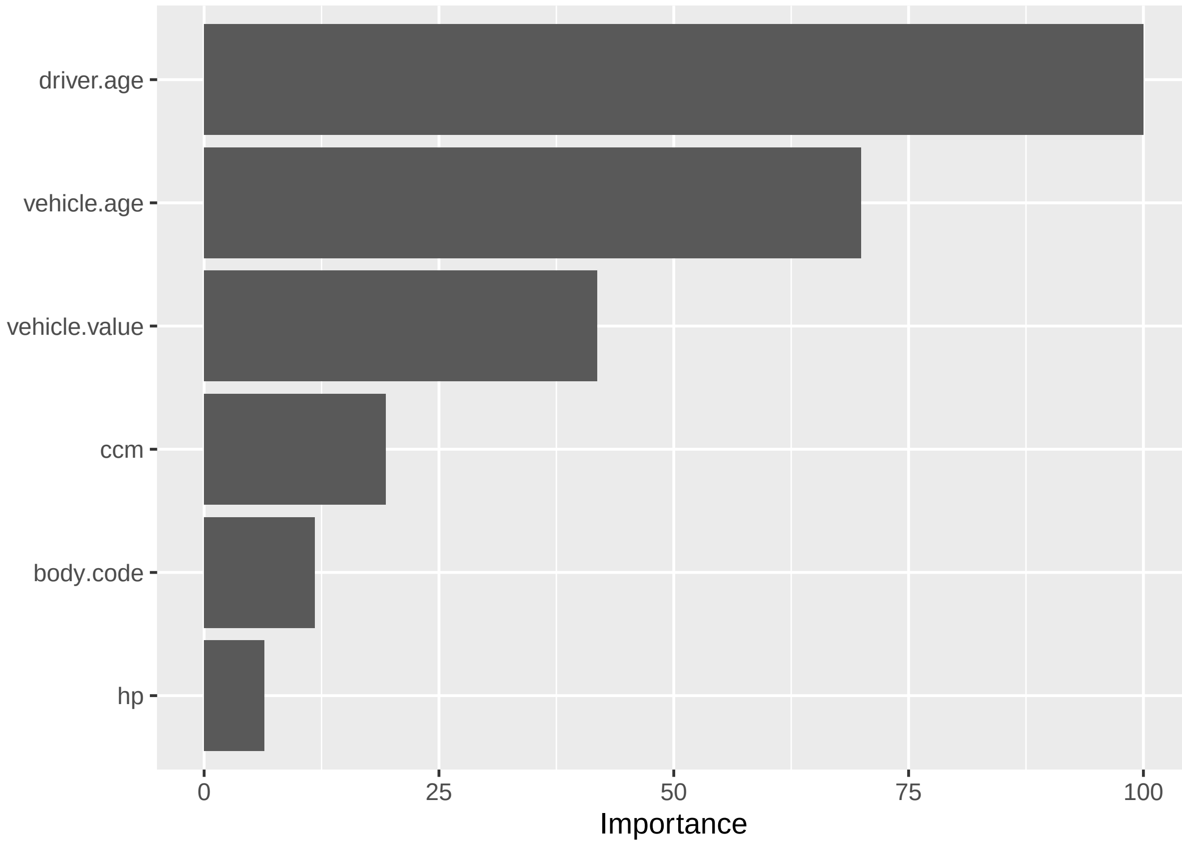
1. Feature importance

Function vi gives you the data

```
# Use of the package vip
var_imp <- vip::vi(fit_srt)
print(var_imp)
```

```
## # A tibble: 6 x 2
##   Variable      Importance
##   <chr>              <dbl>
## 1 driver.age         81.3
## 2 vehicle.age        56.9
## 3 vehicle.value      34.0
## 4 ccm                15.7
## 5 body.code           9.57
## 6 hp                  5.23
```

```r
# Function vip makes the plot
vip::vip(fit_srt, scale = TRUE)
```



Driver age has a non-linar relationship such that it has increasingly stronger effect on the frequency of claims