Regression Trees in Car Insurance: Estimation of Claims Frequency

In this markdown, we will estimate the claims frequency of car drivers using a Regression Tree with the R package {caret}. The data in use come from the chapter one of the book "Predictive Modelling Applications in Actuarial Science, Vol.2", Edited by E. Frees et al.. The website is at the following address: https://instruction.bus.wisc.edu/jfrees/jfreesbooks/PredictiveModelingVol1/glm/v2-chapter-1.html

Data have been already explored in a previous study (cf. EDA for Insurance) stored in another repository. A description of the fields is also available.

Introduction

Decision tree learning is one among many other predictive modelling approaches. They have the advantage to be easily interpreted and work for both classification and regression tasks. However, they typically lack in predictive performance comparing to aggregation methods like Random Forest or GBM that we will explore in another markdown.

Classification And Regression Tree algorithm, aka CART, developed by Breiman et al. in 1984 works by by partitioning the feature space into a number of smaller (non-overlapping) regions with similar response values using a set of splitting rules. Predictions are obtained by fitting a simpler model (e.g., a constant like the average response value) in each region.

We will add more explanation along the way while fitting the model.

1. Loading the data

```
# 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)
)
```

```r
# 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) # reproducibility
# 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
```

```
# 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
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 sets are well balanced.

We start by fitting a simple tree using our train set, taking the predictors that seem to be good candidate.

We calculate a frequency, but how to deal with a claim count in a decision tree? We use a Poisson Deviance as loss function ('method' parameter), keeping the exposure in a two-column matrix.

'maxdepth' represents the maxium depth of the tree 'cp' is the complexity parameter, that specify the proportion by which the overall error should improve for a split to be attempter.

```
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
```

```
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)
```

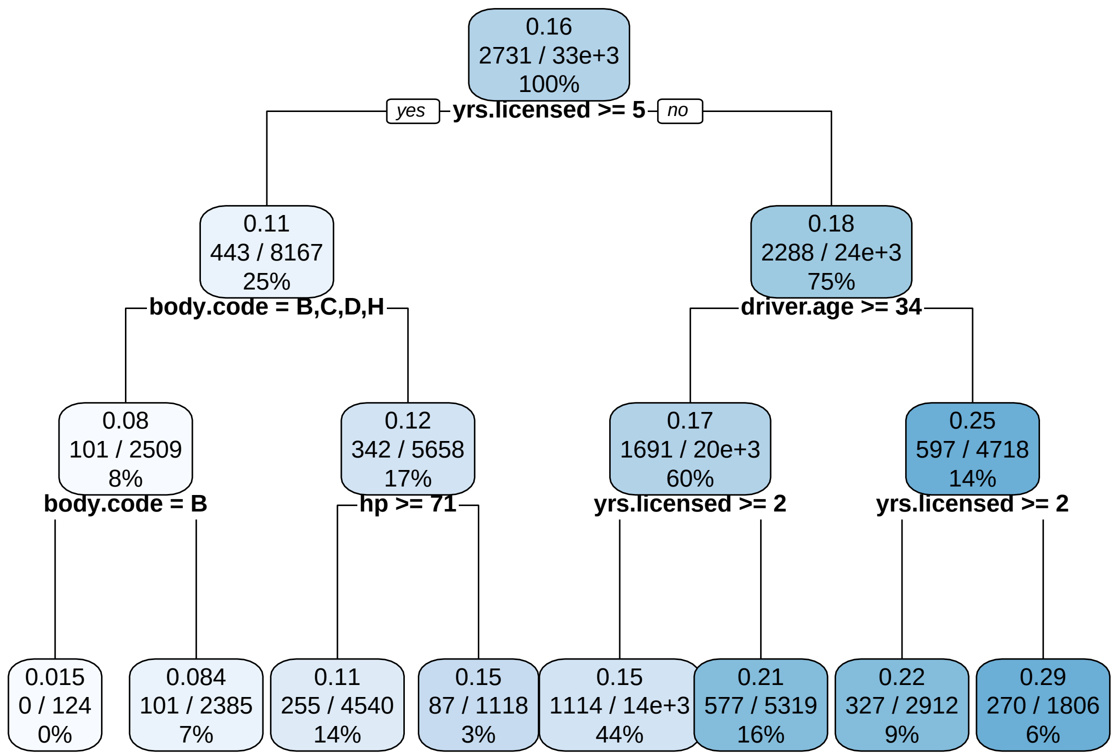
```
## n= 32610
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 32610 13257.680000 0.16469930
##    2) yrs.licensed>=4.5 8167  2491.723000 0.10910610
##      4) body.code=B,C,D,H 2509   640.461400 0.08026114
##        8) body.code=B 124     1.821845 0.01467100 *
##        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 *
```

We get the information on the nodes. To get a better idea, we print a graph using the package {rpartplot}:

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

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

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



To check if the tree gives us a similar prediction, we select the criterai of the last branch of the tree, and there is a slight difference.

```
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
```

We apply a correction here:

```
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 ),
             parms = list(shrink = 10^5)
```

```
)
print(fit)
```

```
## 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 *
```

Now we have the same value: 2.85941

Pruning the tree

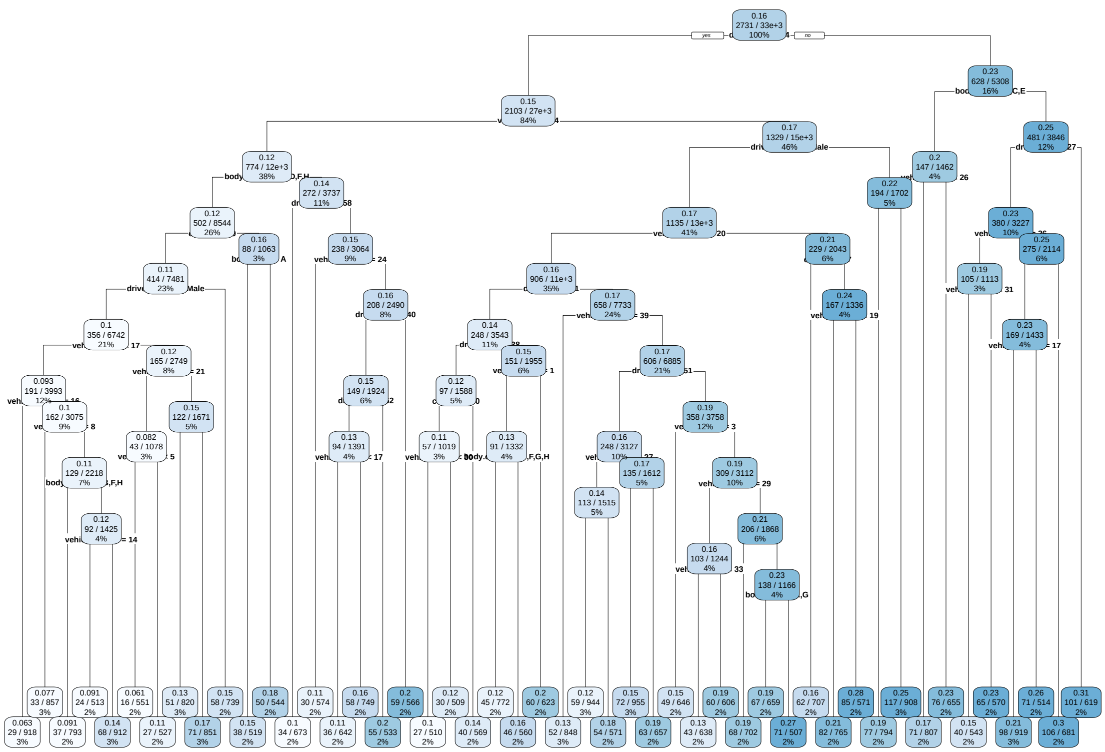
Now we want to follow a pruning strategy to develop a proper model.

We want to built a complex tree and prune it back to find an optimal subtree. To do this, we use the the complexity parameter that penalizes the loss function.

```
set.seed(9753) # reproducibilty
fit2 <- 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 = 1000,
               minbucket = 500,
               cp = 0,
               xval = 5
             )
)
```

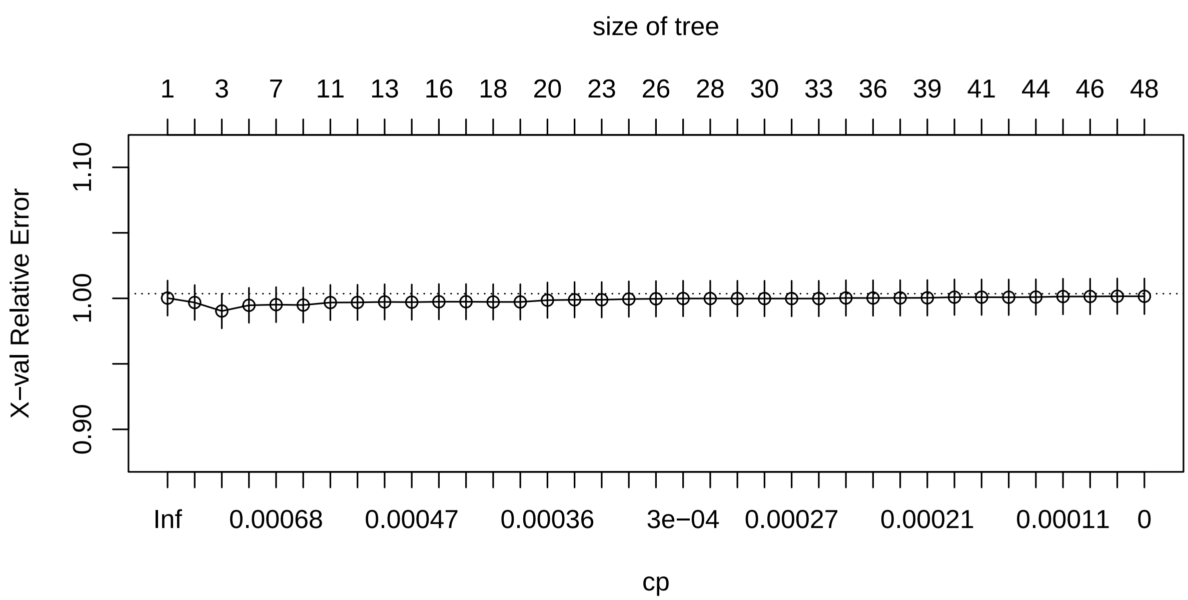
Visualization

```r
rpart.plot(fit2, cex = 0.25)
```



We inspect the cross-validation results

```r
plotcp(fit2)
```



The pruning cp plot shows the relative corss validation error (y-axis) for various cp value (x-axis).

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.

Now we chose the value for the complexity parameter that minimizes the error for pruning.

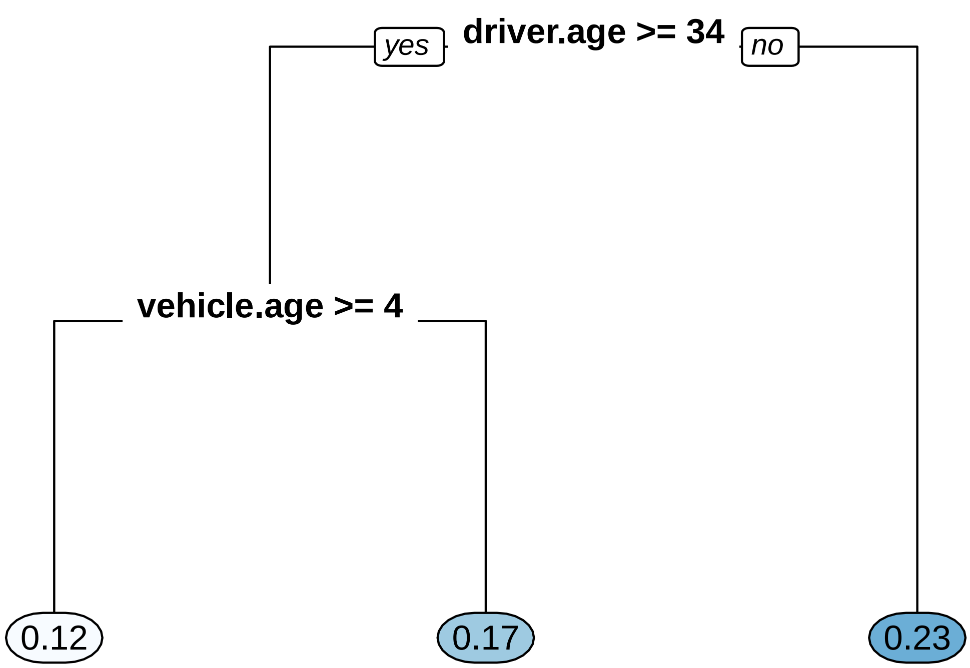
```
# Get the cross-validation results
cpt <- fit2$cptable

# 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(fit2,
                 cp = cpt[min_xerr, 'CP'])
```

We can have a look at our new tree.

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



The tree has been pruned pretty drastically. Two predictors have been retained. But how can we make sense of that.

Interpretability

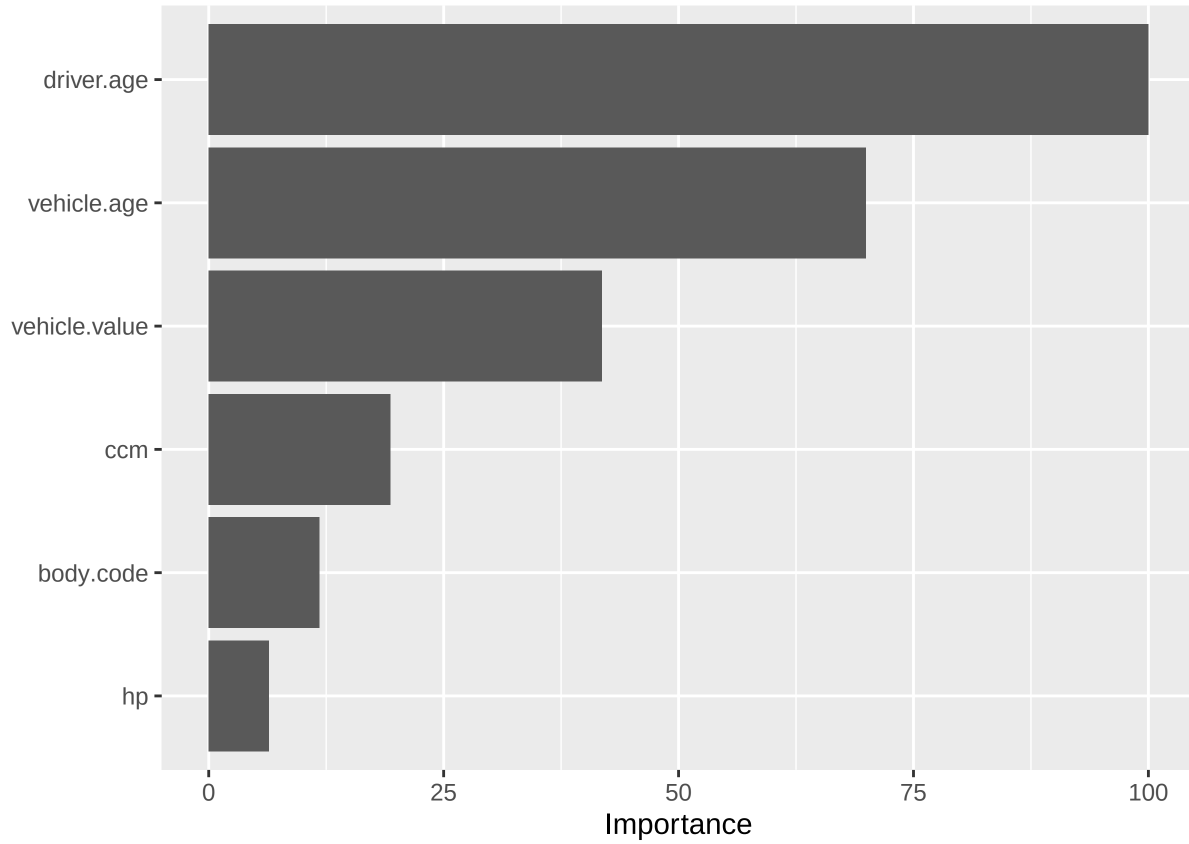
Feature importance is represented by the reduction in the loss function attributed to each variable at each split.

The function vi from the package vip is helpful here.

```r
# 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-linear relationship such that it has increasingly stronger effect on the frequency of claims

Partial dependence plot

```
# Need to define this helper function for Poisson
pred.fun <- function(object,newdata){
  mean(predict(object, newdata))
}
```

```
pred.fun(fit_srt,dta_trn)
```

```
## [1] 0.1646118
```