CART for Continuous Y

Rscript: mtcars CART.R

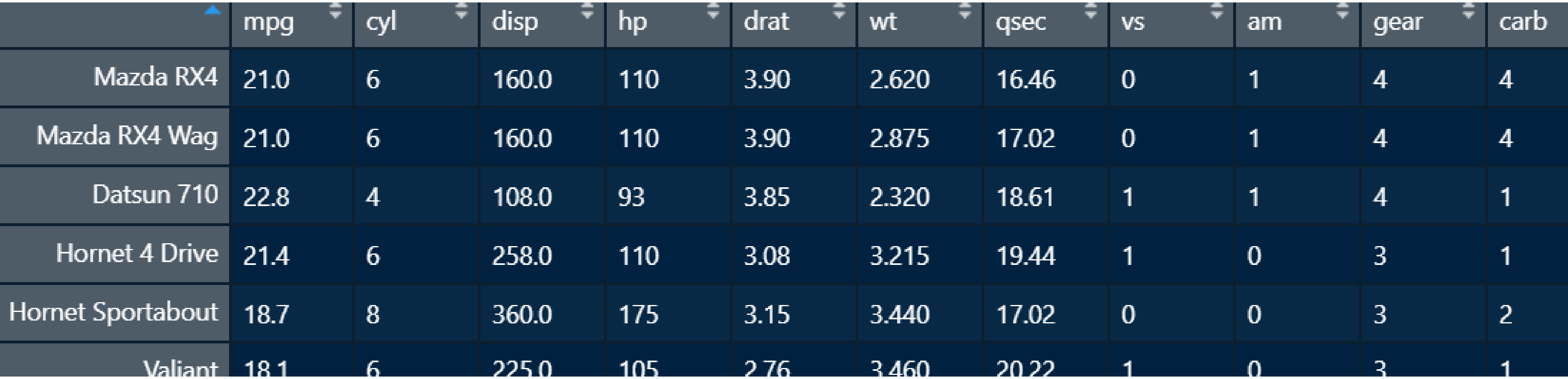
CART

Based on Chew C. H. (2020) textbook: AI, Analytics and Data Science. Vol 1., Chap 8.

Base R dataset: mtcars

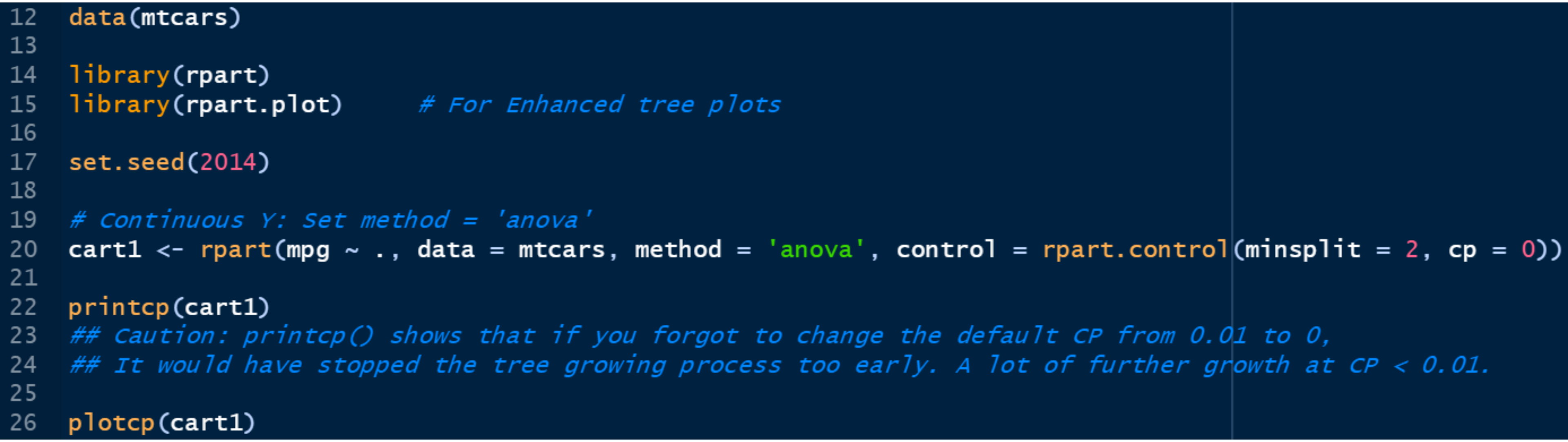
32 cases, 11 columns.

Continuous Outcome variable Y: mpg



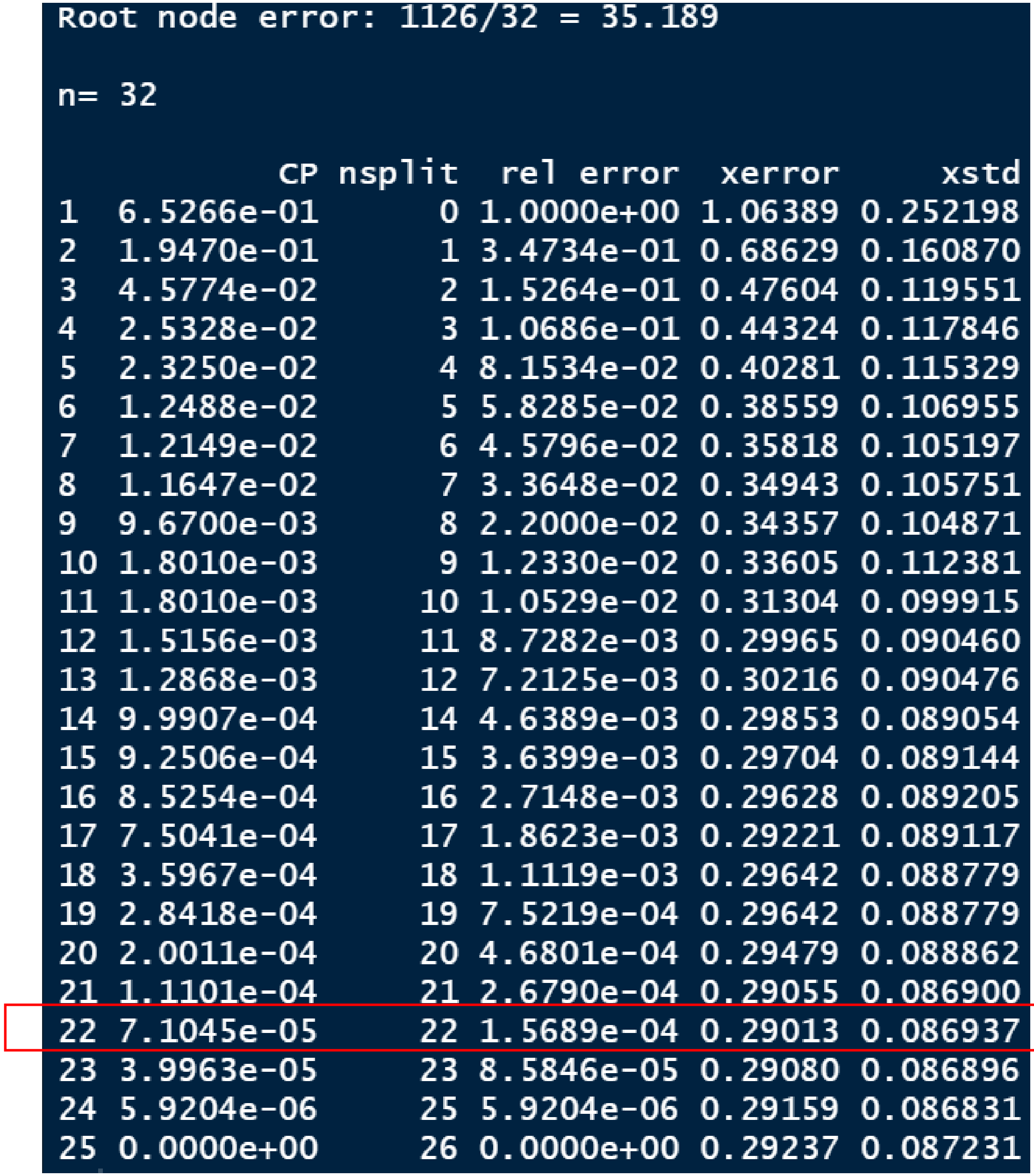
	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1

Continuous Y: method = ‘anova’



```r
data(mtcars)

library(rpart)
library(rpart.plot)      # For Enhanced tree plots

set.seed(2014)

# Continuous Y: Set method = 'anova'
cart1 <- rpart(mpg ~ ., data = mtcars, method = 'anova', control = rpart.control(minsplit = 2, cp = 0))

printcp(cart1)
## Caution: printcp() shows that if you forgot to change the default CP from 0.01 to 0,
## It would have stopped the tree growing process too early. A lot of further growth at CP < 0.01.

plotcp(cart1)
```

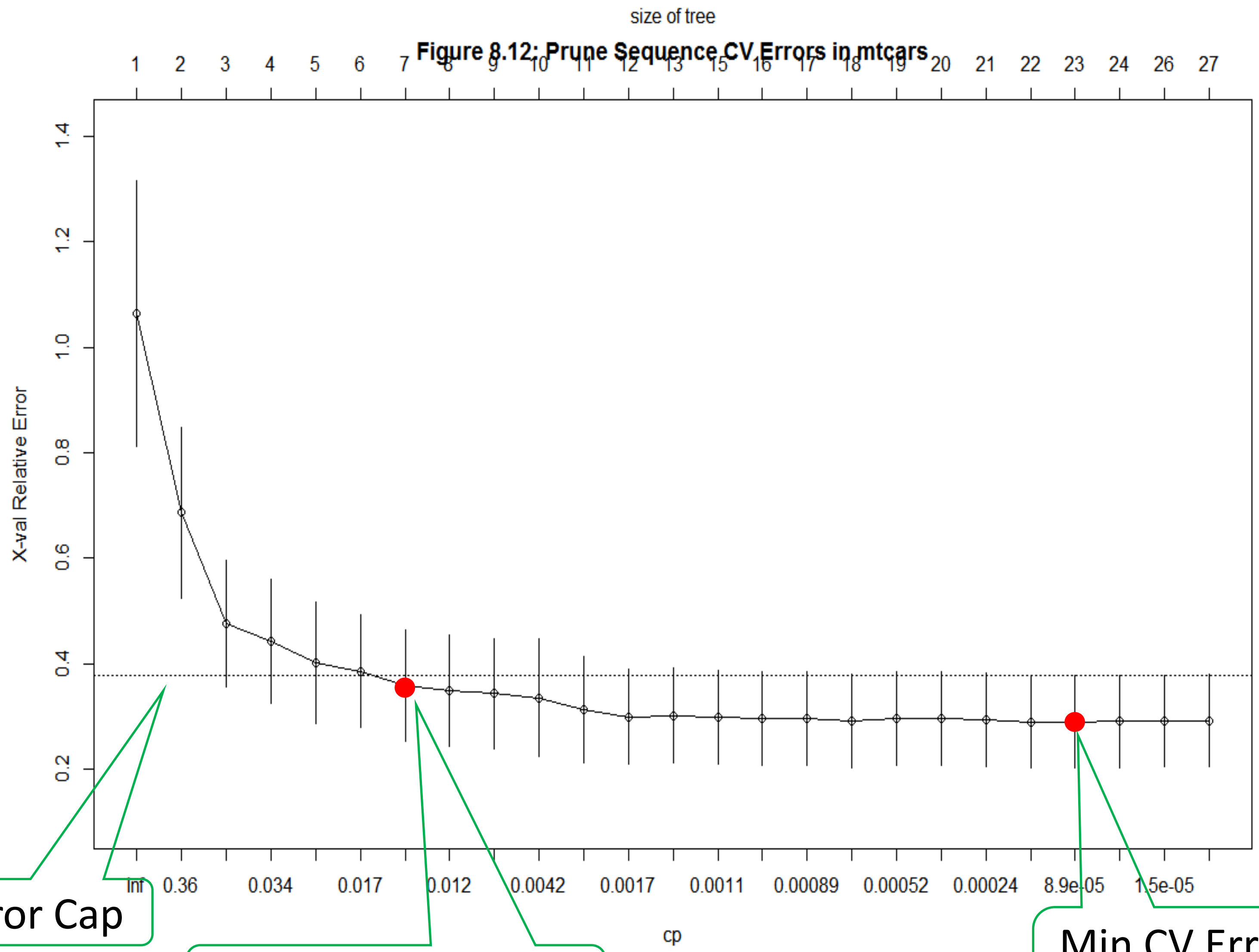
CP Table shows many trees with cp < 0.01



```
Root node error: 1126/32 = 35.189

n= 32

           CP nsplit  rel error  xerror     xstd
1  6.5266e-01      0 1.0000e+00 1.06389 0.252198
2  1.9470e-01      1 3.4734e-01 0.68629 0.160870
3  4.5774e-02      2 1.5264e-01 0.47604 0.119551
4  2.5328e-02      3 1.0686e-01 0.44324 0.117846
5  2.3250e-02      4 8.1534e-02 0.40281 0.115329
6  1.2488e-02      5 5.8285e-02 0.38559 0.106955
7  1.2149e-02      6 4.5796e-02 0.35818 0.105197
8  1.1647e-02      7 3.3648e-02 0.34943 0.105751
9  9.6700e-03      8 2.2000e-02 0.34357 0.104871
10 1.8010e-03      9 1.2330e-02 0.33605 0.112381
11 1.8010e-03     10 1.0529e-02 0.31304 0.099915
12 1.5156e-03     11 8.7282e-03 0.29965 0.090460
13 1.2868e-03     12 7.2125e-03 0.30216 0.090476
14 9.9907e-04     14 4.6389e-03 0.29853 0.089054
15 9.2506e-04     15 3.6399e-03 0.29704 0.089144
16 8.5254e-04     16 2.7148e-03 0.29628 0.089205
17 7.5041e-04     17 1.8623e-03 0.29221 0.089117
18 3.5967e-04     18 1.1119e-03 0.29642 0.088779
19 2.8418e-04     19 7.5219e-04 0.29642 0.088779
20 2.0011e-04     20 4.6801e-04 0.29479 0.088862
21 1.1101e-04     21 2.6790e-04 0.29055 0.086900
22 7.1045e-05     22 1.5689e-04 0.29013 0.086937
23 3.9963e-05     23 8.5846e-05 0.29080 0.086896
24 5.9204e-06     25 5.9204e-06 0.29159 0.086831
25 0.0000e+00     26 0.0000e+00 0.29237 0.087231
```

Min CV Error Tree

plotcp() on mtcars dataset shows CV Error Cap

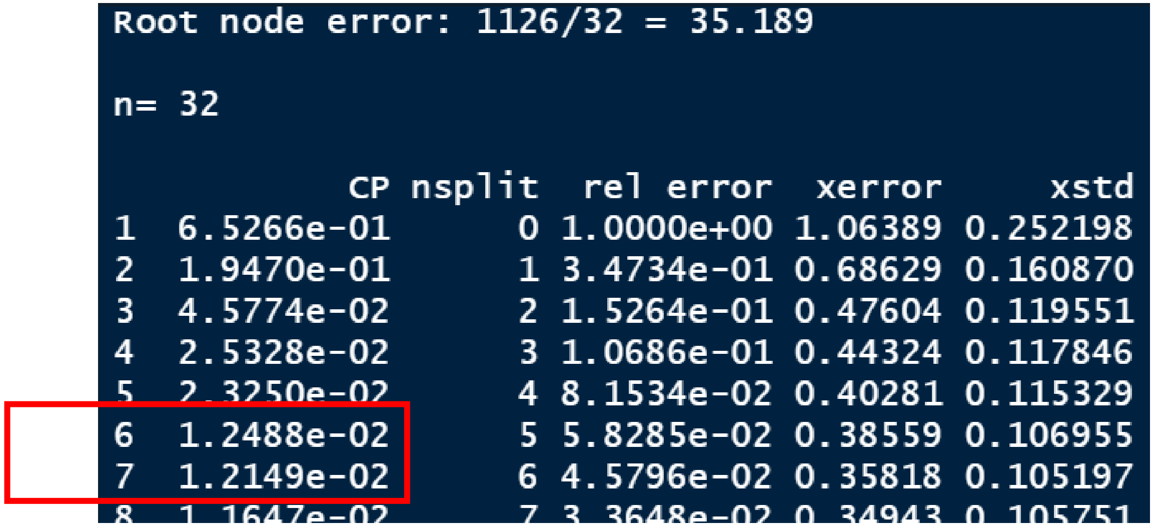


CV Error Cap

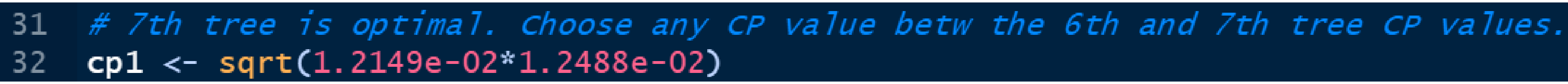
Optimal Tree #7

Min CV Error Tree #22

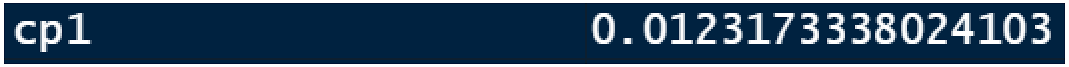
To get the optimal tree from the list of 25 trees, use prune() with a specific value of cp.



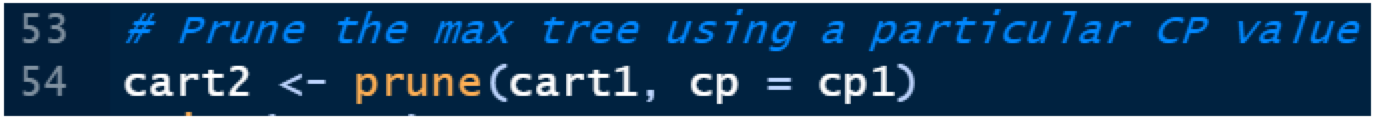
```
Root node error: 1126/32 = 35.189

n= 32

           CP nsplit  rel error  xerror     xstd
1  6.5266e-01      0 1.0000e+00 1.06389 0.252198
2  1.9470e-01      1 3.4734e-01 0.68629 0.160870
3  4.5774e-02      2 1.5264e-01 0.47604 0.119551
4  2.5328e-02      3 1.0686e-01 0.44324 0.117846
5  2.3250e-02      4 8.1534e-02 0.40281 0.115329
6  1.2488e-02      5 5.8285e-02 0.38559 0.106955
7  1.2149e-02      6 4.5796e-02 0.35818 0.105197
8  1.1647e-02      7 3.3648e-02 0.34943 0.105751
```



```r
# 7th tree is optimal. Choose any CP value betw the 6th and 7th tree CP values.
cp1 <- sqrt(1.2149e-02*1.2488e-02)
```

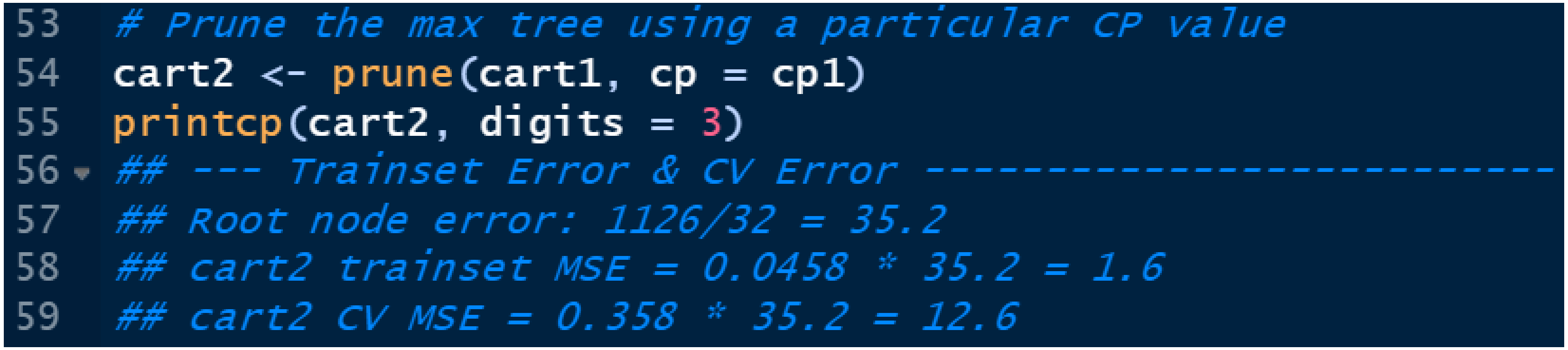


```
cp1                    0.0123173338024103
```

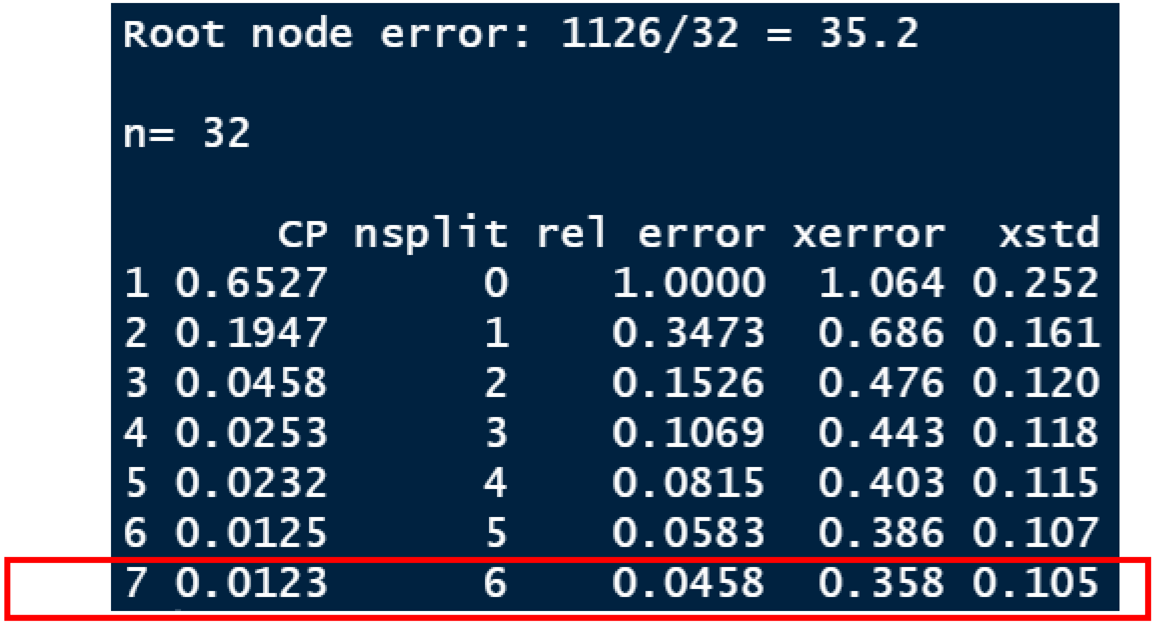


```r
# Prune the max tree using a particular CP value
cart2 <- prune(cart1, cp = cp1)
```

Get optimal tree (based on 1 SE rule) CV error

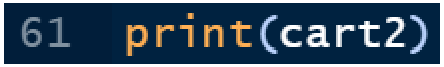


```
# Prune the max tree using a particular CP value
cart2 <- prune(cart1, cp = cp1)
printcp(cart2, digits = 3)
## --- Trainset Error & CV Error --------------------------
## Root node error: 1126/32 = 35.2
## cart2 trainset MSE = 0.0458 * 35.2 = 1.6
## cart2 CV MSE = 0.358 * 35.2 = 12.6
```

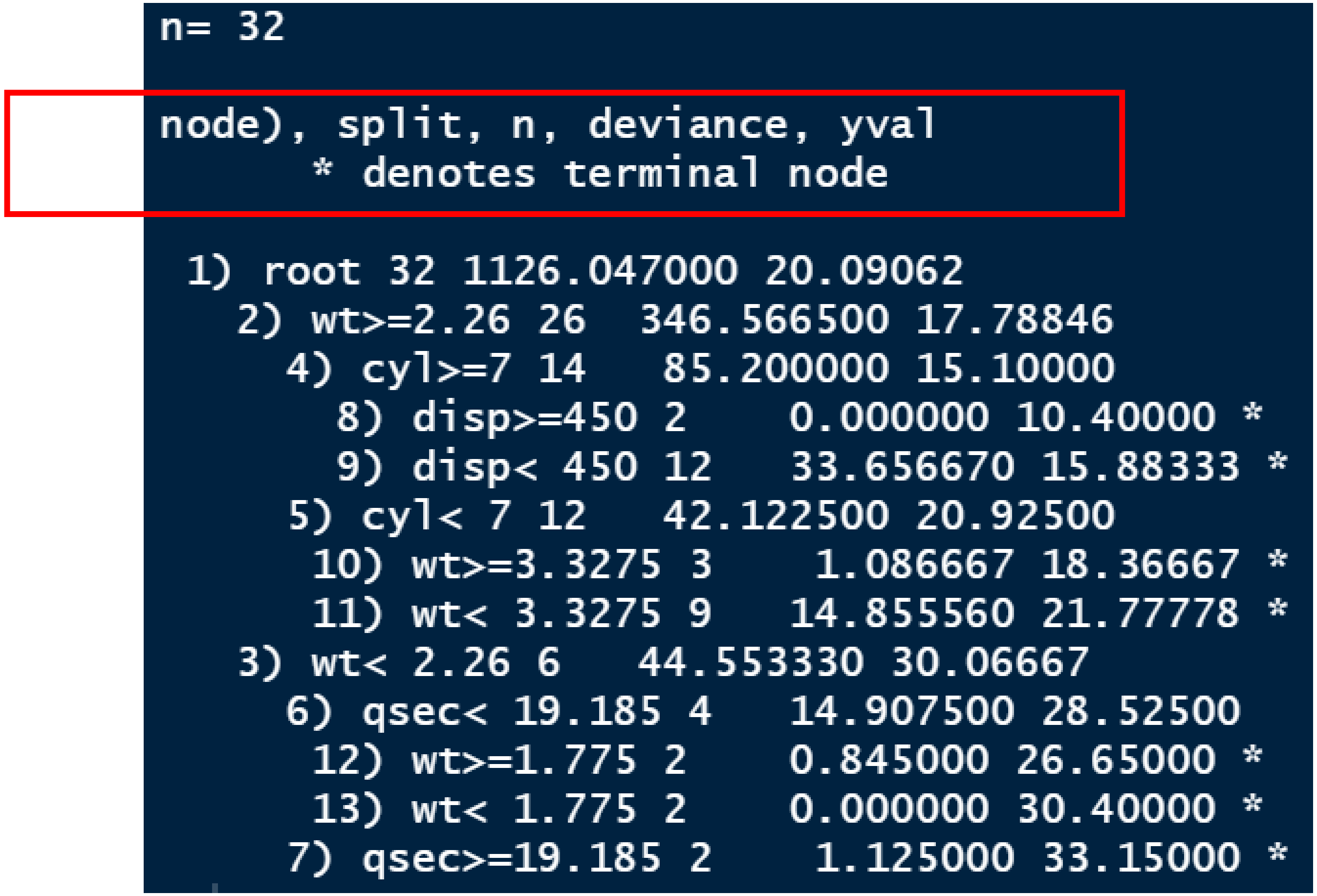


```
Root node error: 1126/32 = 35.2

n= 32

      CP nsplit rel error xerror  xstd
1 0.6527      0    1.0000  1.064 0.252
2 0.1947      1    0.3473  0.686 0.161
3 0.0458      2    0.1526  0.476 0.120
4 0.0253      3    0.1069  0.443 0.118
5 0.0232      4    0.0815  0.403 0.115
6 0.0125      5    0.0583  0.386 0.107
7 0.0123      6    0.0458  0.358 0.105
```

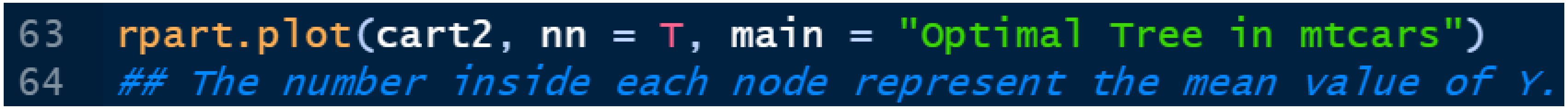
View tree structure on console with print()



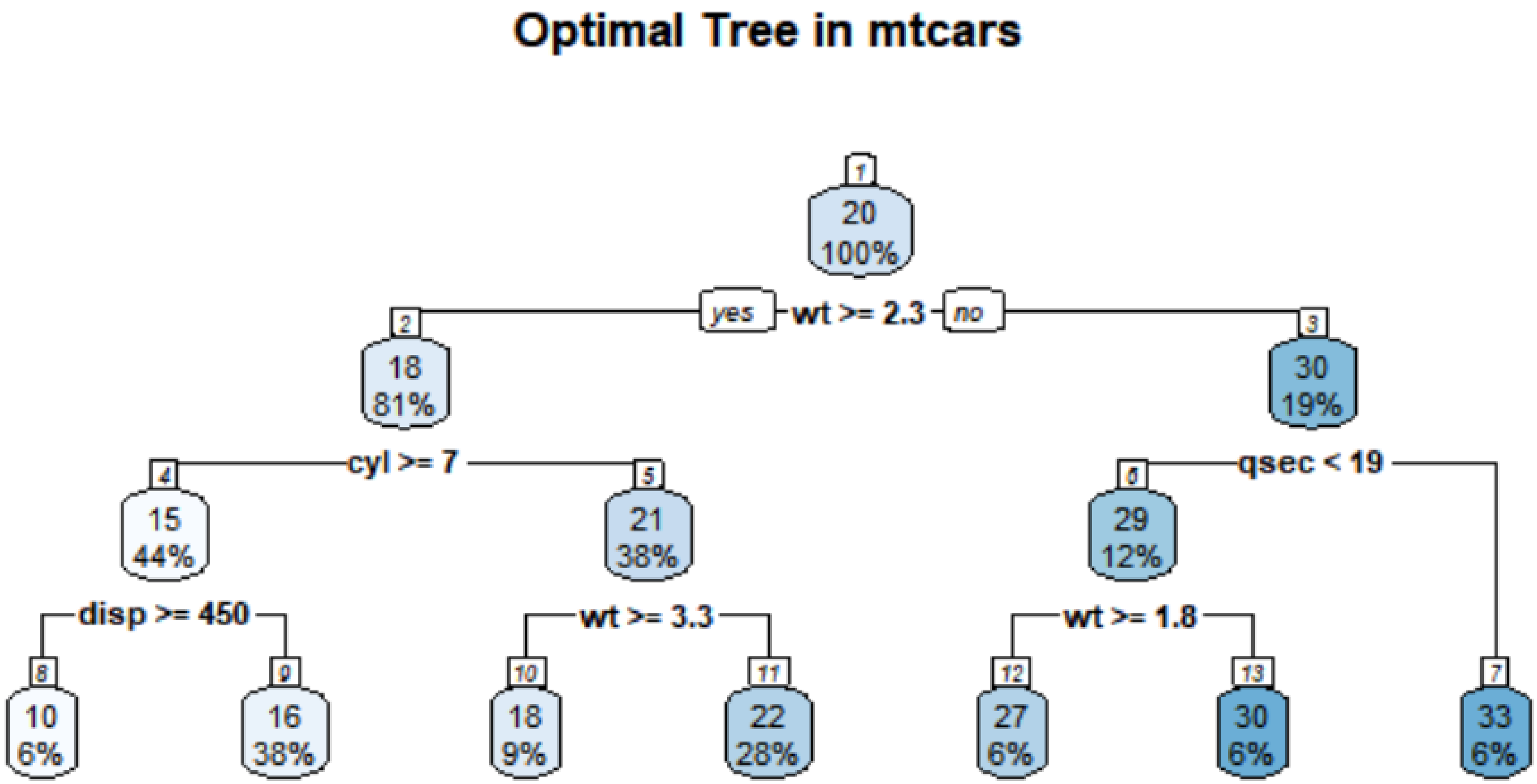
```
print(cart2)
```



```
n= 32

node), split, n, deviance, yval
      * denotes terminal node

 1) root 32 1126.047000 20.09062
   2) wt>=2.26 26  346.566500 17.78846
     4) cyl>=7 14   85.200000 15.10000
       8) disp>=450 2    0.000000 10.40000 *
       9) disp< 450 12   33.656670 15.88333 *
     5) cyl< 7 12   42.122500 20.92500
      10) wt>=3.3275 3    1.086667 18.36667 *
      11) wt< 3.3275 9   14.855560 21.77778 *
   3) wt< 2.26 6   44.553330 30.06667
     6) qsec< 19.185 4   14.907500 28.52500
      12) wt>=1.775 2    0.845000 26.65000 *
      13) wt< 1.775 2    0.000000 30.40000 *
     7) qsec>=19.185 2    1.125000 33.15000 *
```

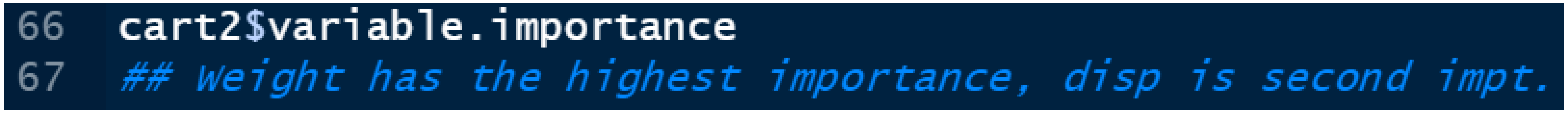


```r
rpart.plot(cart2, nn = T, main = "Optimal Tree in mtcars")
## The number inside each node represent the mean value of Y.
```

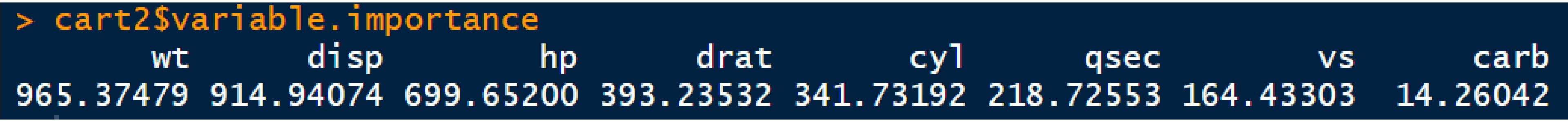


Optimal Tree in mtcars

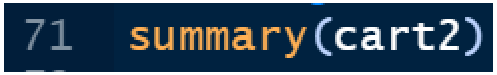
Variable Importance



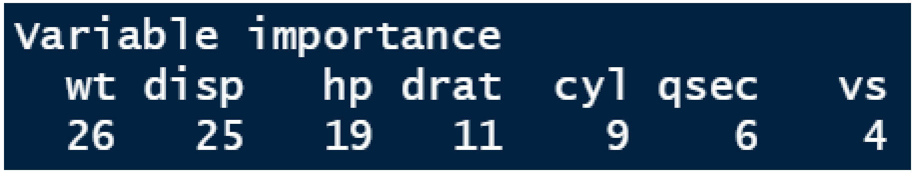
```
cart2$variable.importance
## Weight has the highest importance, disp is second impt.
```



```
> cart2$variable.importance
       wt      disp        hp      drat       cyl      qsec        vs      carb 
965.37479 914.94074 699.65200 393.23532 341.73192 218.72553 164.43303  14.26042 
```



```
summary(cart2)
```



```
Variable importance
  wt disp   hp drat  cyl qsec   vs 
  26   25   19   11    9    6    4 
```

Next Video: Surrogates

- How CART handles missing values automatically.