Decision Tree cross-validation

Bo Li U24425931

3/1/2021

```
library(data.table)
library(ggplot2)
library(ggthemes)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1
```

```
theme_set(theme_bw())
library(MASS)
library(rpart)
library(rpart.plot)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(e1071)
library(tree)
library(ISLR)
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich
```

```
library(tidymodels)
```

```
## Registered S3 method overwritten by 'cli':
##   method     from
##   print.tree tree

## -- Attaching packages --------------------------------------- tidymodels 0.1.2 --

## v broom     0.7.5      v recipes   0.1.15
## v dials     0.0.9      v rsample   0.0.9
## v dplyr     1.0.4      v tibble    3.0.6
## v infer     0.5.4      v tidyr     1.1.2
## v modeldata 0.1.0      v tune      0.1.2
## v parsnip   0.1.5      v workflows 0.2.1
## v purrr     0.3.4      v yardstick 0.0.7

## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x dplyr::between()         masks data.table::between()
## x dplyr::combine()         masks randomForest::combine()
## x purrr::discard()         masks scales::discard()
## x tidyr::expand()          masks Matrix::expand()
## x dplyr::filter()          masks stats::filter()
## x dplyr::first()           masks data.table::first()
## x parsnip::fit()           masks party::fit(), modeltools::fit()
## x dplyr::lag()             masks stats::lag()
## x dplyr::last()            masks data.table::last()
## x purrr::lift()            masks caret::lift()
## x randomForest::margin()   masks ggplot2::margin()
## x tidyr::pack()            masks Matrix::pack()
## x tune::parameters()       masks dials::parameters(), modeltools::parameters()
## x rsample::permutations()  masks e1071::permutations()
## x yardstick::precision()   masks caret::precision()
## x dials::prune()           masks rpart::prune()
## x yardstick::recall()      masks caret::recall()
```

```
## x dplyr::select()          masks MASS::select()
## x yardstick::sensitivity() masks caret::sensitivity()
## x yardstick::specificity() masks caret::specificity()
## x recipes::step()          masks stats::step()
## x purrr::transpose()       masks data.table::transpose()
## x tune::tune()             masks e1071::tune()
## x tidyr::unpack()          masks Matrix::unpack()
## x recipes::update()        masks stats4::update(), Matrix::update(), stats::update()
```

```
library(caTools)
```

```
data <- fread("C:/Users/boli0/Downloads/train.csv")
str(data)
```

```
## Classes 'data.table' and 'data.frame':   2000 obs. of  21 variables:
##  $ battery_power: int  842 1021 563 615 1821 1859 1821 1954 1445 509 ...
##  $ blue         : int  0 1 1 1 1 0 0 0 1 1 ...
##  $ clock_speed  : num  2.2 0.5 0.5 2.5 1.2 0.5 1.7 0.5 0.5 0.6 ...
##  $ dual_sim     : int  0 1 1 0 0 1 0 1 0 1 ...
##  $ fc           : int  1 0 2 0 13 3 4 0 0 2 ...
##  $ four_g       : int  0 1 1 0 1 0 1 0 0 1 ...
##  $ int_memory   : int  7 53 41 10 44 22 10 24 53 9 ...
##  $ m_dep        : num  0.6 0.7 0.9 0.8 0.6 0.7 0.8 0.8 0.7 0.1 ...
##  $ mobile_wt    : int  188 136 145 131 141 164 139 187 174 93 ...
##  $ n_cores      : int  2 3 5 6 2 1 8 4 7 5 ...
##  $ pc           : int  2 6 6 9 14 7 10 0 14 15 ...
##  $ px_height    : int  20 905 1263 1216 1208 1004 381 512 386 1137 ...
##  $ px_width     : int  756 1988 1716 1786 1212 1654 1018 1149 836 1224 ...
##  $ ram          : int  2549 2631 2603 2769 1411 1067 3220 700 1099 513 ...
##  $ sc_h         : int  9 17 11 16 8 17 13 16 17 19 ...
##  $ sc_w         : int  7 3 2 8 2 1 8 3 1 10 ...
##  $ talk_time    : int  19 7 9 11 15 10 18 5 20 12 ...
##  $ three_g      : int  0 1 1 1 1 1 1 1 1 1 ...
##  $ touch_screen : int  0 1 1 0 1 0 0 1 0 0 ...
##  $ wifi         : int  1 0 0 0 0 0 1 1 0 0 ...
##  $ price_range  : int  1 2 2 2 1 1 3 0 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
data$price_range <- as.factor(data$price_range)
```

```
set.seed(810)
split = sample.split(data$price_range, SplitRatio = 0.7)
data_train = subset(data, split == TRUE)
data_test = subset(data, split == FALSE)
y_test <- data_test[,price_range]
```

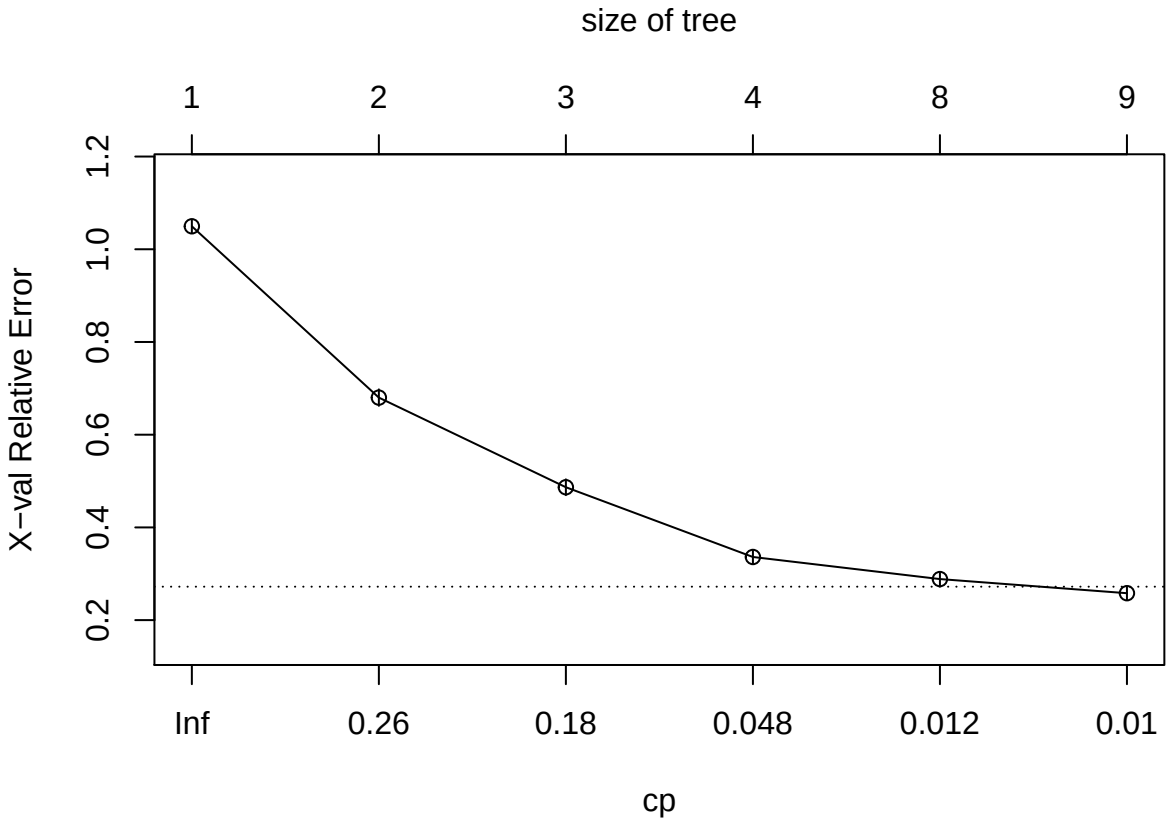
```
# grow tree
fit <- rpart(price_range ~., method = "class", data = data_train)
```

```
printcp(fit)
```

```
##
```

```
## Classification tree:
## rpart(formula = price_range ~ ., data = data_train, method = "class")
##
## Variables actually used in tree construction:
## [1] battery_power px_height     ram
##
## Root node error: 1050/1400 = 0.75
##
## n= 1400
##
##         CP nsplit rel error  xerror     xstd
## 1 0.333333      0   1.00000 1.04952 0.014586
## 2 0.198095      1   0.66667 0.68000 0.017814
## 3 0.160952      2   0.46857 0.48667 0.017156
## 4 0.014286      3   0.30762 0.33619 0.015474
## 5 0.010476      7   0.25048 0.28857 0.014675
## 6 0.010000      8   0.24000 0.25810 0.014079
```

```
plotcp(fit) #visualize cross-validation results
```




```
summary(fit)
```

```
## Call:
## rpart(formula = price_range ~ ., data = data_train, method = "class")
##   n= 1400
```

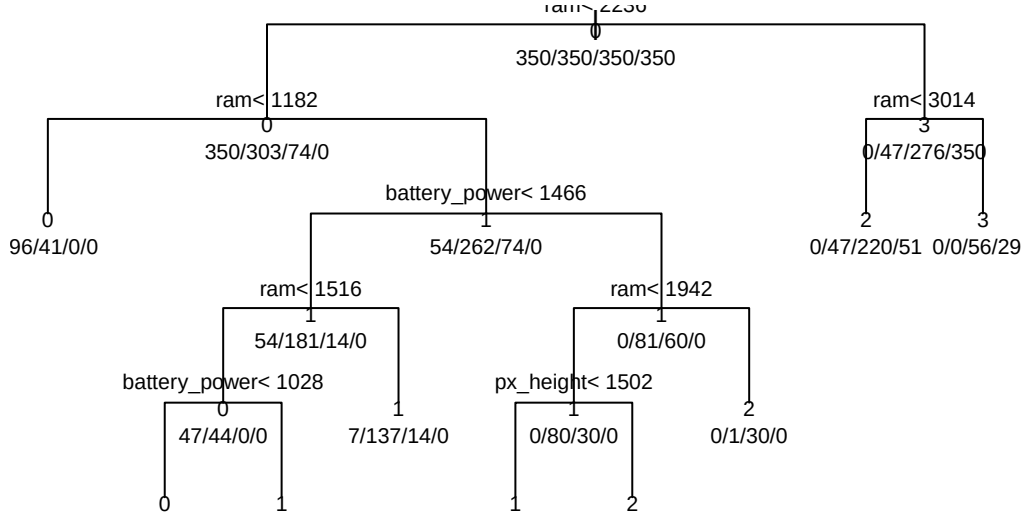
```
##
##           CP nsplit rel error    xerror       xstd
## 1 0.33333333      0 1.0000000 1.0495238 0.01458632
## 2 0.19809524      1 0.6666667 0.6800000 0.01781385
## 3 0.16095238      2 0.4685714 0.4866667 0.01715568
## 4 0.01428571      3 0.3076190 0.3361905 0.01547417
## 5 0.01047619      7 0.2504762 0.2885714 0.01467477
## 6 0.01000000      8 0.2400000 0.2580952 0.01407921
##
## Variable importance
##           ram battery_power     px_height      px_width    int_memory
##            77             7             5             2             2
##          sc_w            pc     mobile_wt          sc_h     talk_time
##             1             1             1             1             1
##            fc
##             1
##
## Node number 1: 1400 observations,    complexity param=0.3333333
##   predicted class=0  expected loss=0.75  P(node) =1
##     class counts:   350   350   350   350
##    probabilities: 0.250 0.250 0.250 0.250
##   left son=2 (727 obs) right son=3 (673 obs)
##   Primary splits:
##       ram           < 2235.5 to the left,  improve=250.809600, (0 missing)
##       battery_power < 1332.5 to the left,  improve= 18.337240, (0 missing)
##       px_width      < 1629.5 to the left,  improve= 12.652890, (0 missing)
##       px_height     < 1212   to the left,  improve=  8.656541, (0 missing)
##       mobile_wt     < 104.5  to the left,  improve=  4.703647, (0 missing)
##   Surrogate splits:
##       px_height     < 280.5  to the right, agree=0.549, adj=0.062, (0 split)
##       battery_power < 1721.5 to the left,  agree=0.534, adj=0.031, (0 split)
##       sc_w          < 10.5   to the left,  agree=0.534, adj=0.031, (0 split)
##       fc            < 13.5   to the left,  agree=0.528, adj=0.018, (0 split)
##       int_memory    < 42.5   to the left,  agree=0.528, adj=0.018, (0 split)
##
## Node number 2: 727 observations,    complexity param=0.1980952
##   predicted class=0  expected loss=0.5185695  P(node) =0.5192857
##     class counts:   350   303    74     0
##    probabilities: 0.481 0.417 0.102 0.000
##   left son=4 (337 obs) right son=5 (390 obs)
##   Primary splits:
##       ram           < 1182   to the left,  improve=160.186700, (0 missing)
##       battery_power < 1448.5 to the left,  improve= 19.954000, (0 missing)
##       px_height     < 639.5  to the left,  improve= 17.872100, (0 missing)
##       px_width      < 1481.5 to the left,  improve= 14.753150, (0 missing)
##       mobile_wt     < 186.5  to the left,  improve=  3.494098, (0 missing)
##   Surrogate splits:
##       px_width   < 684.5  to the left,  agree=0.567, adj=0.065, (0 split)
##       pc         < 1.5    to the left,  agree=0.557, adj=0.045, (0 split)
##       mobile_wt  < 100.5  to the left,  agree=0.556, adj=0.042, (0 split)
##       px_height  < 286.5  to the left,  agree=0.554, adj=0.039, (0 split)
##       int_memory < 6.5    to the left,  agree=0.550, adj=0.030, (0 split)
##
## Node number 3: 673 observations,    complexity param=0.1609524
```

```
##   predicted class=3  expected loss=0.4799406  P(node) =0.4807143
##     class counts:     0    47   276   350
##    probabilities: 0.000 0.070 0.410 0.520
##   left son=6 (318 obs) right son=7 (355 obs)
##   Primary splits:
##       ram           < 3013.5 to the left,  improve=129.502800, (0 missing)
##       battery_power < 1352.5 to the left,  improve= 25.470260, (0 missing)
##       px_width      < 1074   to the left,  improve= 20.660770, (0 missing)
##       px_height     < 672.5  to the left,  improve= 16.689370, (0 missing)
##       mobile_wt     < 121.5  to the right, improve=  6.052946, (0 missing)
##   Surrogate splits:
##       battery_power < 589    to the left,  agree=0.548, adj=0.044, (0 split)
##       sc_h          < 18.5   to the right, agree=0.544, adj=0.035, (0 split)
##       int_memory    < 4.5    to the left,  agree=0.541, adj=0.028, (0 split)
##       px_width      < 1074   to the left,  agree=0.541, adj=0.028, (0 split)
##       dual_sim      < 0.5    to the left,  agree=0.536, adj=0.019, (0 split)
##
## Node number 4: 337 observations
##   predicted class=0  expected loss=0.1216617  P(node) =0.2407143
##     class counts:   296    41     0     0
##    probabilities: 0.878 0.122 0.000 0.000
##
## Node number 5: 390 observations,    complexity param=0.01428571
##   predicted class=1  expected loss=0.3282051  P(node) =0.2785714
##     class counts:    54   262    74     0
##    probabilities: 0.138 0.672 0.190 0.000
##   left son=10 (249 obs) right son=11 (141 obs)
##   Primary splits:
##       battery_power < 1466.5 to the left,  improve=18.603900, (0 missing)
##       ram           < 1508.5 to the left,  improve=13.678960, (0 missing)
##       px_height     < 1169   to the left,  improve=13.131310, (0 missing)
##       px_width      < 1351   to the left,  improve= 9.417544, (0 missing)
##       n_cores       < 4.5    to the left,  improve= 2.217842, (0 missing)
##   Surrogate splits:
##       px_height < 1639.5 to the left,  agree=0.649, adj=0.028, (0 split)
##       talk_time < 3.5    to the right, agree=0.646, adj=0.021, (0 split)
##       px_width  < 530.5  to the right, agree=0.644, adj=0.014, (0 split)
##       ram       < 1203.5 to the right, agree=0.641, adj=0.007, (0 split)
##
## Node number 6: 318 observations
##   predicted class=2  expected loss=0.3081761  P(node) =0.2271429
##     class counts:     0    47   220    51
##    probabilities: 0.000 0.148 0.692 0.160
##
## Node number 7: 355 observations
##   predicted class=3  expected loss=0.1577465  P(node) =0.2535714
##     class counts:     0     0    56   299
##    probabilities: 0.000 0.000 0.158 0.842
##
## Node number 10: 249 observations,    complexity param=0.01428571
##   predicted class=1  expected loss=0.2730924  P(node) =0.1778571
##     class counts:    54   181    14     0
##    probabilities: 0.217 0.727 0.056 0.000
##   left son=20 (91 obs) right son=21 (158 obs)
```

```
##   Primary splits:
##       ram           < 1515.5 to the left,  improve=21.822950, (0 missing)
##       battery_power < 740.5  to the left,  improve= 8.768181, (0 missing)
##       px_width      < 980    to the left,  improve= 8.672876, (0 missing)
##       px_height     < 592    to the left,  improve= 7.562294, (0 missing)
##       pc            < 2.5    to the left,  improve= 2.146907, (0 missing)
##   Surrogate splits:
##       battery_power < 1456.5 to the right, agree=0.643, adj=0.022, (0 split)
##       px_height     < 1572   to the right, agree=0.643, adj=0.022, (0 split)
##       px_width      < 1907.5 to the right, agree=0.643, adj=0.022, (0 split)
##       sc_w          < 16.5   to the right, agree=0.643, adj=0.022, (0 split)
##
## Node number 11: 141 observations,    complexity param=0.01428571
##   predicted class=1  expected loss=0.4255319  P(node) =0.1007143
##     class counts:     0    81    60     0
##    probabilities: 0.000 0.574 0.426 0.000
##   left son=22 (110 obs) right son=23 (31 obs)
##   Primary splits:
##       ram        < 1941.5 to the left,  improve=23.364320, (0 missing)
##       px_height  < 696    to the left,  improve=13.131340, (0 missing)
##       px_width   < 1240   to the left,  improve=11.910450, (0 missing)
##       int_memory < 47.5   to the left,  improve= 2.319954, (0 missing)
##       n_cores    < 5.5    to the left,  improve= 2.030236, (0 missing)
##
## Node number 20: 91 observations,    complexity param=0.01428571
##   predicted class=0  expected loss=0.4835165  P(node) =0.065
##     class counts:    47    44     0     0
##    probabilities: 0.516 0.484 0.000 0.000
##   left son=40 (53 obs) right son=41 (38 obs)
##   Primary splits:
##       battery_power < 1027.5 to the left,  improve=19.332380, (0 missing)
##       px_height     < 671    to the left,  improve=19.332380, (0 missing)
##       px_width      < 1004   to the left,  improve=15.553280, (0 missing)
##       pc            < 2.5    to the left,  improve= 4.990549, (0 missing)
##       clock_speed   < 1.95   to the right, improve= 4.038473, (0 missing)
##   Surrogate splits:
##       px_height   < 671    to the left,  agree=0.692, adj=0.263, (0 split)
##       talk_time   < 14.5   to the left,  agree=0.692, adj=0.263, (0 split)
##       px_width    < 1416.5 to the left,  agree=0.659, adj=0.184, (0 split)
##       clock_speed < 2.75   to the left,  agree=0.626, adj=0.105, (0 split)
##       sc_h        < 5.5    to the right, agree=0.615, adj=0.079, (0 split)
##
## Node number 21: 158 observations
##   predicted class=1  expected loss=0.1329114  P(node) =0.1128571
##     class counts:     7   137    14     0
##    probabilities: 0.044 0.867 0.089 0.000
##
## Node number 22: 110 observations,    complexity param=0.01047619
##   predicted class=1  expected loss=0.2727273  P(node) =0.07857143
##     class counts:     0    80    30     0
##    probabilities: 0.000 0.727 0.273 0.000
##   left son=44 (99 obs) right son=45 (11 obs)
##   Primary splits:
##       px_height < 1502.5 to the left,  improve=12.929290, (0 missing)
```

```
##       px_width  < 1379   to the left,  improve=11.985170, (0 missing)
##       ram       < 1504.5 to the left,  improve= 7.272727, (0 missing)
##       n_cores   < 5.5    to the left,  improve= 3.217440, (0 missing)
##       pc        < 18.5   to the right, improve= 1.636364, (0 missing)
##   Surrogate splits:
##       battery_power < 1479.5 to the right, agree=0.918, adj=0.182, (0 split)
##       sc_w          < 14.5   to the left,  agree=0.909, adj=0.091, (0 split)
##
## Node number 23: 31 observations
##   predicted class=2  expected loss=0.03225806  P(node) =0.02214286
##     class counts:     0     1    30     0
##    probabilities: 0.000 0.032 0.968 0.000
##
## Node number 40: 53 observations
##   predicted class=0  expected loss=0.2075472  P(node) =0.03785714
##     class counts:    42    11     0     0
##    probabilities: 0.792 0.208 0.000 0.000
##
## Node number 41: 38 observations
##   predicted class=1  expected loss=0.1315789  P(node) =0.02714286
##     class counts:     5    33     0     0
##    probabilities: 0.132 0.868 0.000 0.000
##
## Node number 44: 99 observations
##   predicted class=1  expected loss=0.1919192  P(node) =0.07071429
##     class counts:     0    80    19     0
##    probabilities: 0.000 0.808 0.192 0.000
##
## Node number 45: 11 observations
##   predicted class=2  expected loss=0  P(node) =0.007857143
##     class counts:     0     0    11     0
##    probabilities: 0.000 0.000 1.000 0.000
```

```r
# plot tree
plot(fit, uniform = TRUE, main = "Classification Tree for price_range")
text(fit, use.n=TRUE, all=TRUE, cex=.7)
```

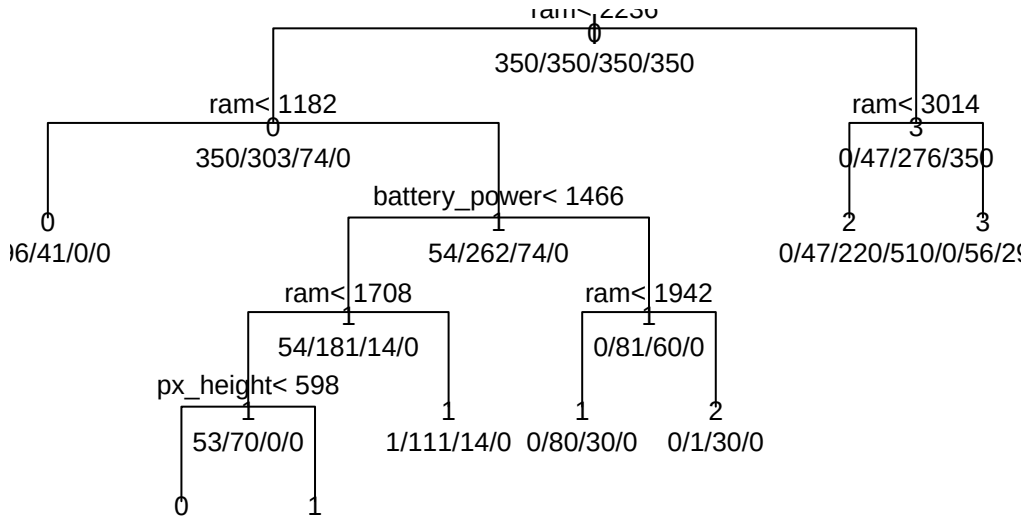



```
# orginal tree accuracy
fit.pred = predict(fit, newdata = data_test, type = "class")

test_accuary <- mean(fit.pred == y_test)

test_accuary
```

```
## [1] 0.7766667
```

```
# prune the tree
fit_cp = rpart(price_range ~ .,method = "class", data = data_train,control = rpart.control(minsplit = 1
```

```
# plot the pruned tree
plot(fit_cp, uniform=TRUE,
   main="Pruned Classification Tree for price_range")
text(fit_cp, use.n=TRUE, all=TRUE, cex=.8)
```




```
post(fit_cp, title = "Pruned Classification Tree for price_range")
```

```
# pruned tree accuracy
fit_cp.pred = predict(fit_cp, newdata = data_test, type = "class")

test_accuary_cp <- mean(fit_cp.pred == y_test)

test_accuary_cp
```

```
## [1] 0.775
```