810 Team Project

Bo Li U24425931

2/24/2021

```
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(ipred)
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(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>
```

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

```
# 1-1. Build linear regression model
linreg = lm(price_range ~., data = data_train)

summary(linreg)
```

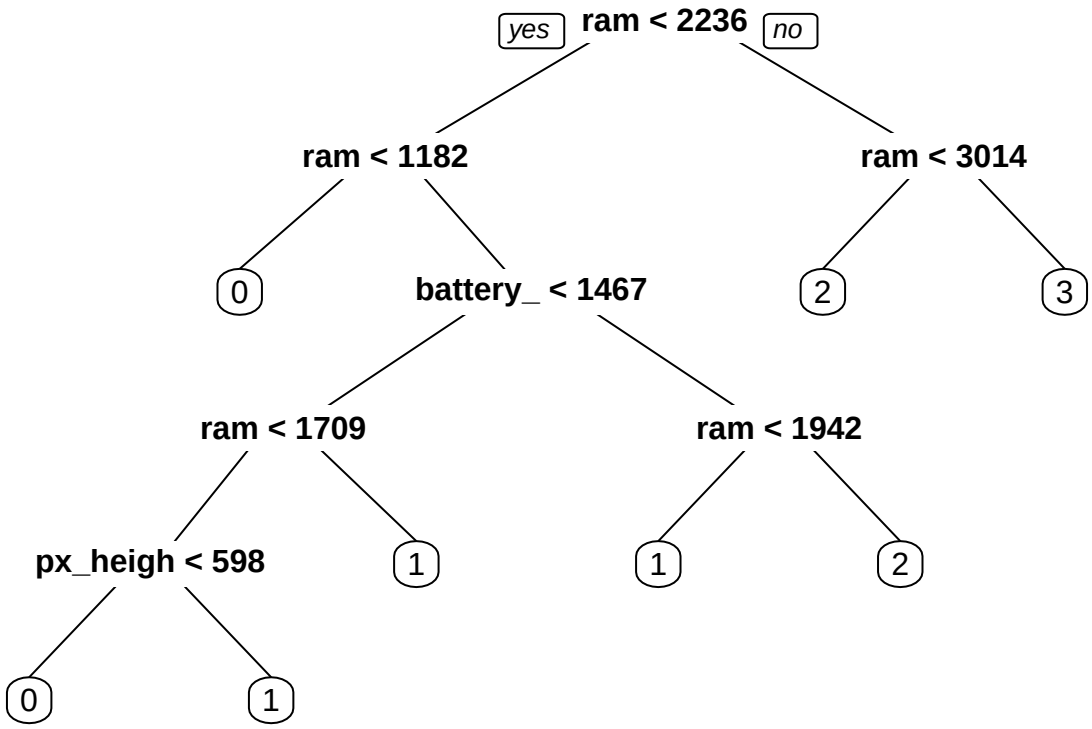
```
##
## Call:
## lm(formula = price_range ~ ., data = data_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.01082 -0.24804  0.00676  0.24155  0.84015
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.532e+00  7.328e-02 -20.901  < 2e-16 ***
## battery_power  5.100e-04  1.930e-05  26.426  < 2e-16 ***
## blue          -6.726e-03  1.723e-02  -0.390  0.69637
## clock_speed   -2.043e-02  1.054e-02  -1.937  0.05289 .
## dual_sim      -2.722e-02  1.726e-02  -1.577  0.11503
## fc             1.262e-03  2.587e-03   0.488  0.62587
## four_g        -3.044e-02  2.114e-02  -1.439  0.15027
## int_memory     8.259e-04  4.743e-04   1.741  0.08184 .
## m_dep         -3.580e-02  2.991e-02  -1.197  0.23152
```

```
## mobile_wt     -7.184e-04  2.430e-04  -2.957  0.00316 **
## n_cores        1.126e-03  3.769e-03   0.299  0.76517
## pc            -3.519e-04  1.809e-03  -0.195  0.84575
## px_height      2.498e-04  2.236e-05  11.171  < 2e-16 ***
## px_width       2.866e-04  2.294e-05  12.493  < 2e-16 ***
## ram            9.477e-04  7.942e-06 119.326  < 2e-16 ***
## sc_h           7.565e-04  2.380e-03   0.318  0.75065
## sc_w          -1.125e-03  2.285e-03  -0.493  0.62243
## talk_time     -1.214e-04  1.575e-03  -0.077  0.93854
## three_g        3.325e-02  2.496e-02   1.332  0.18314
## touch_screen   7.650e-03  1.714e-02   0.446  0.65536
## wifi          -1.956e-02  1.719e-02  -1.138  0.25538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3196 on 1379 degrees of freedom
## Multiple R-squared:  0.9195, Adjusted R-squared:  0.9183
## F-statistic: 787.5 on 20 and 1379 DF,  p-value: < 2.2e-16
```

```
# 1-2. linear regression sse is 63.55421
linreg.pred = predict(linreg, newdata = data_test)
linreg.sse = sum((linreg.pred - data_test$price_range)^2)

linreg.sse
```

```
## [1] 63.55421
```

```
# 2-1. Build single decision tree classification

tree = rpart(price_range ~ .,method = "class", data = data_train,control = rpart.control(minsplit = 1)

prp(tree)
```




```
print(tree)
```

```
## n= 1400
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1400 1050 0 (0.250000000 0.250000000 0.250000000 0.250000000)
##    2) ram< 2235.5 727  377 0 (0.481430536 0.416781293 0.101788171 0.000000000)
##      4) ram< 1182 337   41 0 (0.878338279 0.121661721 0.000000000 0.000000000) *
##      5) ram>=1182 390  128 1 (0.138461538 0.671794872 0.189743590 0.000000000)
##       10) battery_power< 1466.5 249   68 1 (0.216867470 0.726907631 0.056224900 0.000000000)
##         20) ram< 1708.5 123   53 1 (0.430894309 0.569105691 0.000000000 0.000000000)
##           40) px_height< 598 61   17 0 (0.721311475 0.278688525 0.000000000 0.000000000) *
##           41) px_height>=598 62    9 1 (0.145161290 0.854838710 0.000000000 0.000000000) *
##         21) ram>=1708.5 126   15 1 (0.007936508 0.880952381 0.111111111 0.000000000) *
##       11) battery_power>=1466.5 141   60 1 (0.000000000 0.574468085 0.425531915 0.000000000)
##         22) ram< 1941.5 110   30 1 (0.000000000 0.727272727 0.272727273 0.000000000) *
##         23) ram>=1941.5 31    1 2 (0.000000000 0.032258065 0.967741935 0.000000000) *
##    3) ram>=2235.5 673  323 3 (0.000000000 0.069836553 0.410104012 0.520059435)
##      6) ram< 3013.5 318   98 2 (0.000000000 0.147798742 0.691823899 0.160377358) *
##      7) ram>=3013.5 355   56 3 (0.000000000 0.000000000 0.157746479 0.842253521) *
```

```
summary(tree)
```

```
## Call:
## rpart(formula = price_range ~ ., data = data_train, method = "class",
##     parms = list(split = "information"), control = rpart.control(minsplit = 1))
##   n= 1400
##
##           CP nsplit rel error    xerror       xstd
## 1 0.33333333      0 1.0000000 1.0495238 0.01458632
## 2 0.19809524      1 0.6666667 0.6676190 0.01781740
## 3 0.16095238      2 0.4685714 0.4771429 0.01708226
## 4 0.01380952      3 0.3076190 0.3257143 0.01531097
## 5 0.01285714      5 0.2800000 0.3038095 0.01494703
## 6 0.01000000      7 0.2542857 0.2752381 0.01442290
##
## Variable importance
##           ram battery_power     px_height      px_width          sc_w
##            78             7             5             2             2
##    int_memory     mobile_wt            fc            pc      dual_sim
##             2             1             1             1             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=650.760600, (0 missing)
##       battery_power < 1332.5 to the left,  improve= 37.065280, (0 missing)
##       px_width      < 1630.5 to the left,  improve= 25.439830, (0 missing)
##       px_height     < 1212   to the left,  improve= 18.147350, (0 missing)
##       mobile_wt     < 104.5  to the left,  improve=  9.566532, (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=231.36100, (0 missing)
##       battery_power < 1455   to the left,  improve= 47.93258, (0 missing)
##       px_height     < 639.5  to the left,  improve= 33.61836, (0 missing)
##       px_width      < 1144.5 to the left,  improve= 29.33494, (0 missing)
##       mobile_wt     < 186.5  to the left,  improve=  4.38501, (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=180.93670, (0 missing)
##       battery_power < 1352.5 to the left,  improve= 46.75949, (0 missing)
##       px_width      < 1283   to the left,  improve= 31.47901, (0 missing)
##       px_height     < 955    to the left,  improve= 24.86811, (0 missing)
##       int_memory    < 10.5   to the left,  improve=  7.02468, (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=0.01380952
##   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=57.255060, (0 missing)
##       ram           < 1508.5 to the left,  improve=47.743900, (0 missing)
##       px_height     < 674.5  to the left,  improve=31.980610, (0 missing)
##       px_width      < 1113.5 to the left,  improve=29.405230, (0 missing)
##       n_cores       < 4.5    to the left,  improve= 3.540274, (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=0.01285714
##   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 (123 obs) right son=21 (126 obs)
##   Primary splits:
##       ram           < 1708.5 to the left,  improve=46.820460, (0 missing)
##       px_width      < 1479.5 to the left,  improve=24.350920, (0 missing)
##       px_height     < 736    to the left,  improve=20.883800, (0 missing)
##       battery_power < 1027.5 to the left,  improve=15.672300, (0 missing)
##       sc_h          < 11.5   to the right, improve= 6.621616, (0 missing)
##   Surrogate splits:
##       sc_w          < 4.5    to the left,  agree=0.574, adj=0.138, (0 split)
##       px_width      < 1779   to the right, agree=0.570, adj=0.130, (0 split)
##       battery_power < 558    to the left,  agree=0.558, adj=0.106, (0 split)
##       blue          < 0.5    to the left,  agree=0.554, adj=0.098, (0 split)
##       mobile_wt     < 161.5  to the right, agree=0.554, adj=0.098, (0 split)
##
## Node number 11: 141 observations,    complexity param=0.01380952
##   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=27.291620, (0 missing)
##       px_height     < 696    to the left,  improve=13.931310, (0 missing)
##       px_width      < 1240   to the left,  improve=12.786660, (0 missing)
##       battery_power < 1990   to the left,  improve= 2.607385, (0 missing)
##       int_memory    < 47.5   to the left,  improve= 2.351360, (0 missing)
##
## Node number 20: 123 observations,    complexity param=0.01285714
##   predicted class=1  expected loss=0.4308943  P(node) =0.08785714
##     class counts:    53    70     0     0
##    probabilities: 0.431 0.569 0.000 0.000
##   left son=40 (61 obs) right son=41 (62 obs)
##   Primary splits:
##       px_height     < 598    to the left,  improve=22.302360, (0 missing)
##       px_width      < 994.5  to the left,  improve=19.697840, (0 missing)
##       battery_power < 1027.5 to the left,  improve=19.114670, (0 missing)
##       ram           < 1515.5 to the left,  improve= 5.609121, (0 missing)
##       pc            < 3.5    to the left,  improve= 4.698548, (0 missing)
##   Surrogate splits:
##       px_width      < 1085   to the left,  agree=0.667, adj=0.328, (0 split)
##       battery_power < 919    to the left,  agree=0.626, adj=0.246, (0 split)
##       clock_speed   < 1.65   to the right, agree=0.610, adj=0.213, (0 split)
##       dual_sim      < 0.5    to the right, agree=0.593, adj=0.180, (0 split)
##       four_g        < 0.5    to the right, agree=0.585, adj=0.164, (0 split)
##
## Node number 21: 126 observations
##   predicted class=1  expected loss=0.1190476  P(node) =0.09
##     class counts:     1   111    14     0
##    probabilities: 0.008 0.881 0.111 0.000
##
## Node number 22: 110 observations
##   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
```

```
##
## 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: 61 observations
##   predicted class=0  expected loss=0.2786885  P(node) =0.04357143
##     class counts:    44    17     0     0
##    probabilities: 0.721 0.279 0.000 0.000
##
## Node number 41: 62 observations
##   predicted class=1  expected loss=0.1451613  P(node) =0.04428571
##     class counts:     9    53     0     0
##    probabilities: 0.145 0.855 0.000 0.000
```

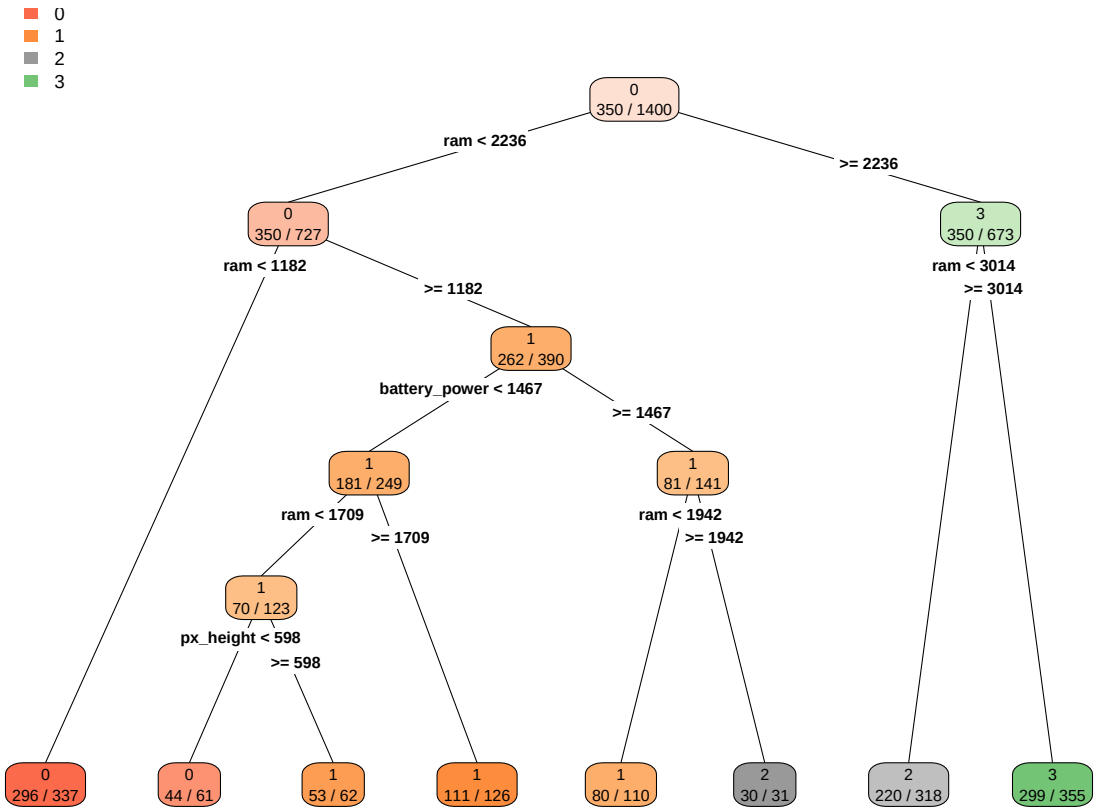
2-2. Single decision tree classification sse is 7021.066

```
tree.pred = predict(tree, newdata = data_test)

tree.sse = sum ((tree.pred - data_test$price_range)^2)

tree.sse
```

```
## [1] 7021.066
```

```
# 2-3. Build decision tree while cp = 0.01000000
# E1 is 0.1907143.
```

```
tree2 <- prune(tree, cp = 0.01000000)
rpart.plot(tree2, type = 4, branch = 0, extra = 2)
```




```
CFit1 <- predict(tree2, data_train, type = "class")
ConfM1 <- table(data_train$price_range, CFit1)
(E1 <- (sum(ConfM1) - sum(diag(ConfM1)))/sum(ConfM1))
```

```
## [1] 0.1907143
```

```
# 2-4. Utiltize bagging from ipred package to build combined decision tree 1
# Bagging classification trees with 25 bootstrap replications, E2 is 0.75.
BagM1 <- bagging(price_range ~., data = data_train, nbagg = 25, coob = TRUE, control = rpart.control(mi

CFit2 <- predict(BagM1, data_train, type = "class")

ConfM2 <- table(data_train$price_range, CFit2)

(E2 <- (sum(ConfM2) - sum(diag(ConfM2)))/sum(ConfM2))
```

```
## [1] 0.75
```

```
# 3-1. Random forest
# E3 is 0.9992.
rFM <- randomForest(price_range ~., data = data_train, importance = TRUE, proximity = TRUE)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

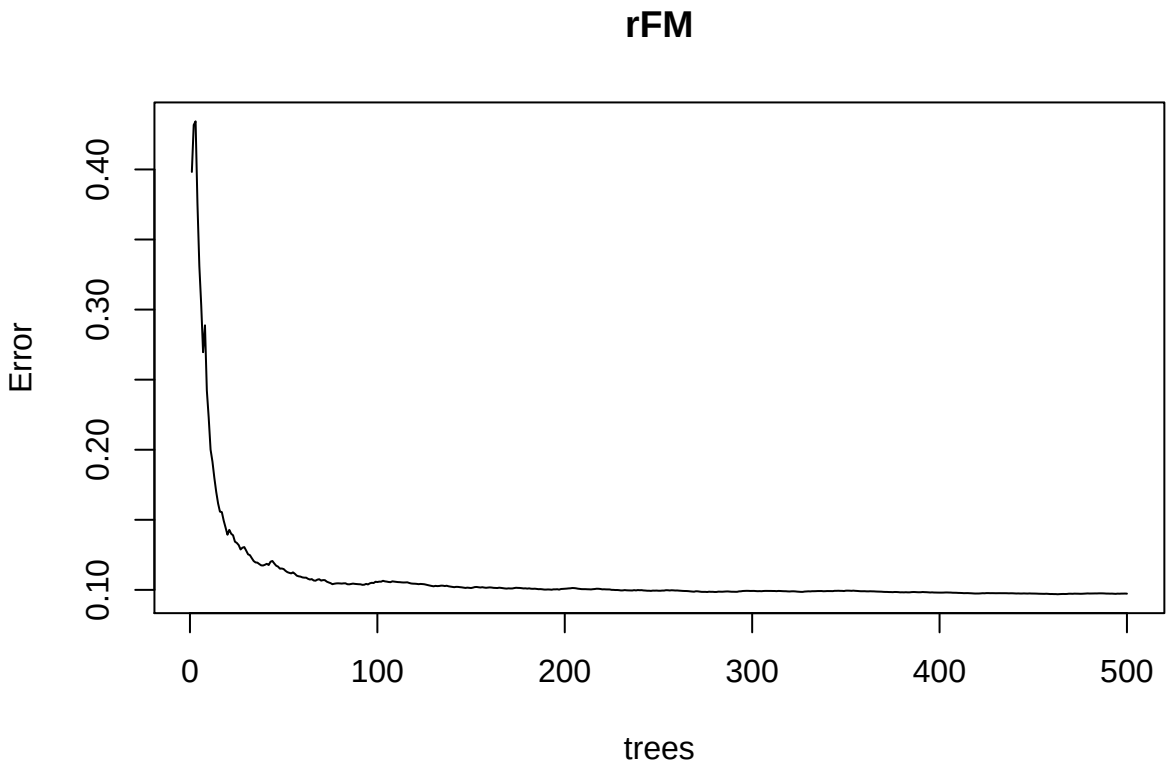
```
print(rFM)
```

```
##
## Call:
##  randomForest(formula = price_range ~ ., data = data_train, importance = TRUE,      proximity = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##           Mean of squared residuals: 0.09728984
##                     % Var explained: 92.22
```

```
summary(rFM)
```

```
##                 Length  Class  Mode
## call                 5 -none- call
## type                 1 -none- character
## predicted         1400 -none- numeric
## mse                500 -none- numeric
## rsq                500 -none- numeric
## oob.times         1400 -none- numeric
## importance          40 -none- numeric
## importanceSD        20 -none- numeric
## localImportance      0 -none- NULL
## proximity       1960000 -none- numeric
## ntree                1 -none- numeric
## mtry                 1 -none- numeric
## forest              11 -none- list
## coefs                0 -none- NULL
## y                 1400 -none- numeric
## test                 0 -none- NULL
## inbag                0 -none- NULL
## terms                3 terms  call
```

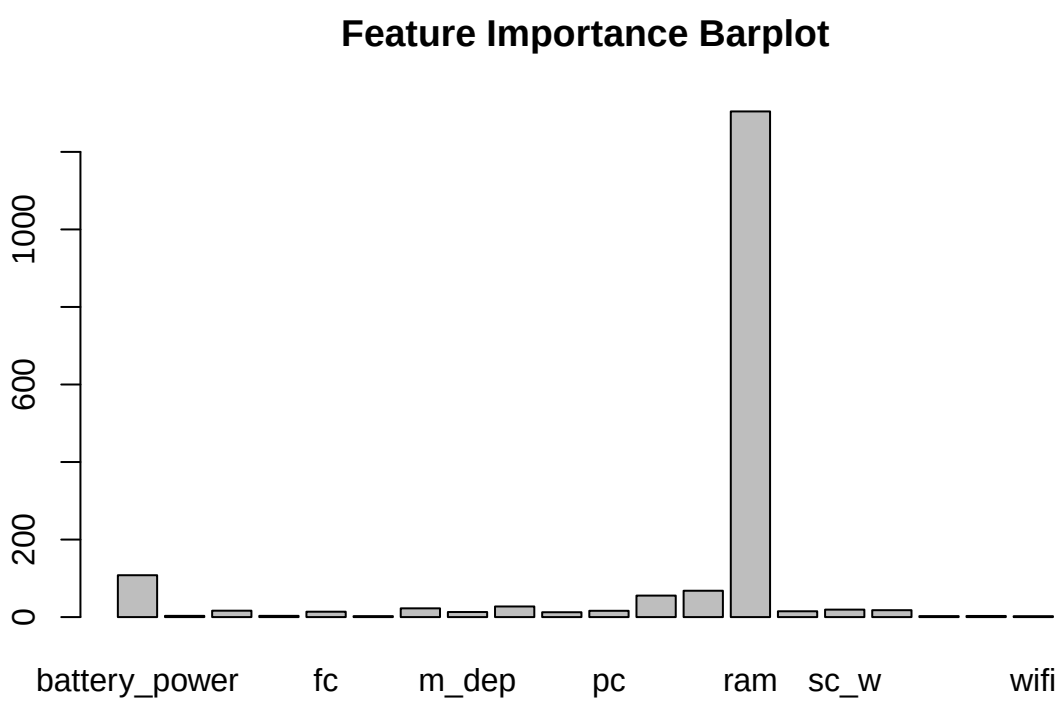
```
plot(rFM)
```




```
Fit3 <- predict(rFM, data_train)

ConfM3 <- table(data_train$price_range, Fit3)

(E3 <- (sum(ConfM3) - sum(diag(ConfM3)))/sum(ConfM3))
```

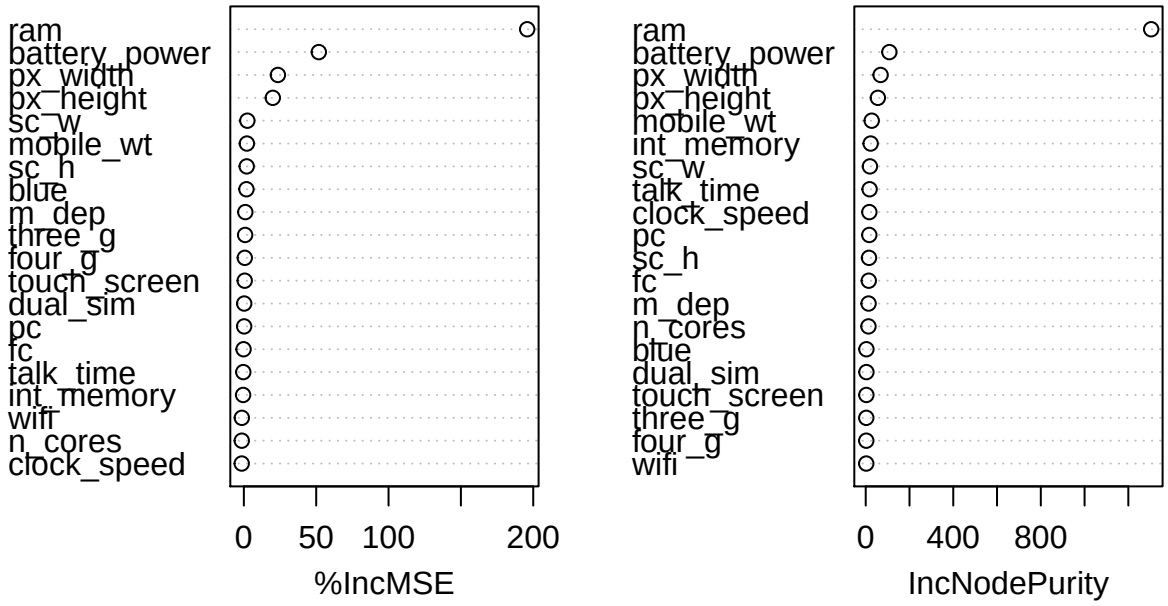
```
## [1] 0.9992857
```

```
# 3-2. Random Forest Feature Importance Barplot
barplot(rFM$importance[,2], main = "Feature Importance Barplot")
```




```r
# 3-3. Random Forest Feature Importance ScatterPlot
varImpPlot(x = rFM, sort = TRUE, n.var = nrow(rFM$importance), main = "Feature Importance ScatterPlot")
```

Feature Importance ScatterPlot

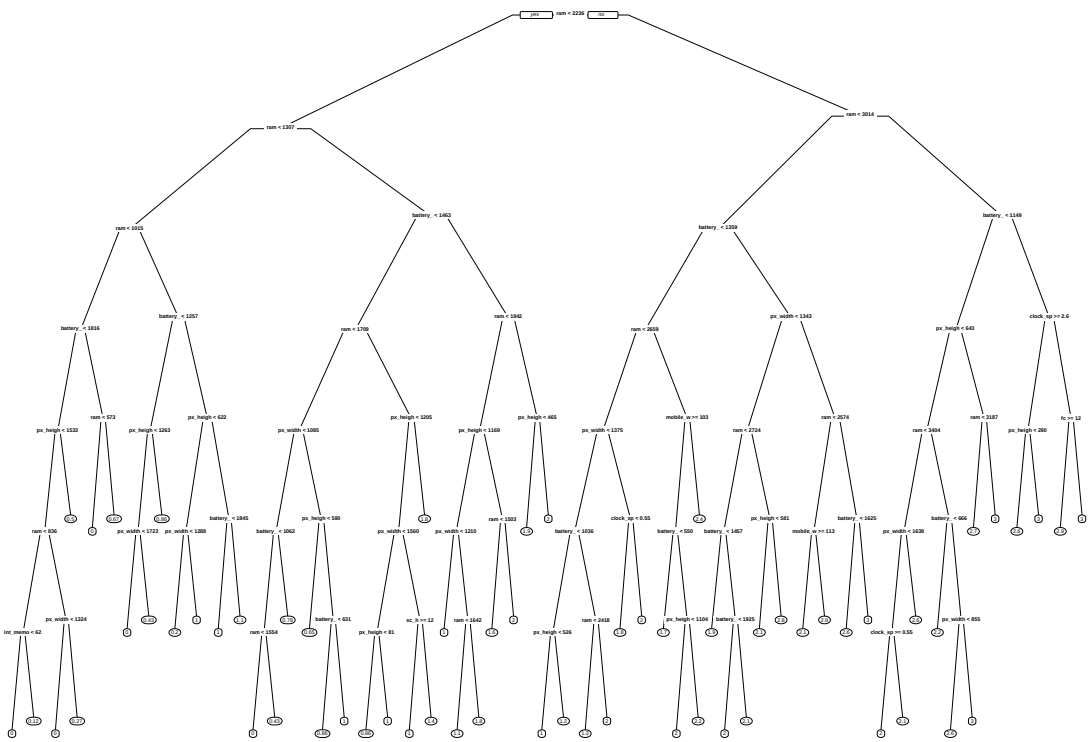



```
# 4-1. regression trees cross-validation
tr.control = trainControl(method = "cv", number = 10)
cp.grid = expand.grid(.cp = (0:10)*0.001)
tr = train(price_range ~., data = data_train, method = "rpart", trControl = tr.control, tuneGrid = cp.g
tr
```

```
## CART
##
## 1400 samples
##   20 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1260, 1260, 1260, 1260, 1260, 1260, ...
## Resampling results across tuning parameters:
##
##   cp     RMSE       Rsquared   MAE
##   0.000  0.3564032  0.8994784  0.1825574
##   0.001  0.3654180  0.8936930  0.1933365
##   0.002  0.3805913  0.8849092  0.2138772
##   0.003  0.3859357  0.8814569  0.2305825
##   0.004  0.3940466  0.8769803  0.2473934
##   0.005  0.4141426  0.8638818  0.2867955
##   0.006  0.4267100  0.8553362  0.3193465
##   0.007  0.4392260  0.8466482  0.3399474
##   0.008  0.4398649  0.8464382  0.3379351
```

```
##   0.009  0.4410306  0.8456274  0.3381416
##   0.010  0.4410306  0.8456274  0.3381416
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was cp = 0.
```

```r
# 4-2. plot best tree
best.tree = tr$finalModel
prp(best.tree)
```



```r
# 4-3. best tree sse is 80.34064
best.tree.pred = predict(best.tree, newdata = data_test)
best.tree.sse = sum((best.tree.pred - data_test$price_range)^2)
best.tree.sse
```

```
## [1] 80.34064
```