BA 810 Team Project

Yixuan Wang

2/19/2021

Mobile price classification

How to best predict the price range of a mobile phone based on technical features

Load useful libraries

```
library(data.table)
library(dplyr)
library(stringr)
library(caTools)
library(caret)
library(randomForest)
library(rpart)
library(rpart.plot)
```

Load Dataset

```
m <- fread("/Users/wangyixuan/Desktop/BA810 Supervised machine learning/data/processe
d_train.csv")
```

```
head(m, 1)
```

```
##    battery_power blue clock_speed dual_sim fc four_g int_memory m_dep mobile_wt
## 1:           842    0         2.2        0  1      0          7   0.6       188
##    n_cores pc px_height px_width  ram sc_h sc_w talk_time three_g touch_screen
## 1:       2  2        20      756 2549    9    7        19       0            0
##    wifi price_range price_binary p0 p1 p2 p3
## 1:    1           1            0  0  1  0  0
```

```
dim(m)
```

```
## [1] 2000   26
```

1. Binary

```
m_binary <- m[, !c("price_range", "p0", "p1", "p2", "p3")]
```

Split our data set on training and testing subset.

```
set.seed(810)
sampleSplit <- sample.split(Y=m_binary$price_binary, SplitRatio=0.7)
trainSet <- subset(x=m_binary, sampleSplit==TRUE)
testSet <- subset(x=m_binary, sampleSplit==FALSE)
```

1.1 Logistic regression

```
log_model_binary <- glm(price_binary ~ ., family=binomial(link='logit'), data=trainSe
t)
```

```
summary(log_model_binary)
```

```
##
## Call:
## glm(formula = price_binary ~ ., family = binomial(link = "logit"),
##     data = trainSet)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.592   0.000   0.000   0.000   2.969
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.107e+02  1.030e+02  -3.018 0.002548 **
## battery_power  5.582e-02  1.890e-02   2.953 0.003149 **
## blue          -4.575e-01  2.111e+00  -0.217 0.828441
## clock_speed    9.472e-01  9.297e-01   1.019 0.308269
## dual_sim      -8.063e-01  1.425e+00  -0.566 0.571616
## fc            -7.492e-02  2.968e-01  -0.252 0.800702
## four_g        -2.352e+00  1.830e+00  -1.285 0.198899
## int_memory     1.369e-01  6.131e-02   2.234 0.025515 *
## m_dep         -4.170e+00  2.467e+00  -1.690 0.090976 .
## mobile_wt     -9.513e-02  3.353e-02  -2.838 0.004547 **
## n_cores        3.221e-01  3.063e-01   1.052 0.292982
## pc             2.103e-01  1.553e-01   1.354 0.175625
## px_height      3.019e-02  9.071e-03   3.328 0.000873 ***
## px_width       3.391e-02  1.219e-02   2.781 0.005422 **
## ram            8.798e-02  2.893e-02   3.041 0.002358 **
## sc_h          -1.976e-02  2.029e-01  -0.097 0.922421
## sc_w           2.505e-01  2.108e-01   1.188 0.234763
## talk_time      6.372e-02  1.186e-01   0.537 0.591108
## three_g        2.473e+00  1.814e+00   1.363 0.172915
## touch_screen  -8.278e-01  1.540e+00  -0.538 0.590831
## wifi          -3.303e+00  1.882e+00  -1.755 0.079244 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1940.812  on 1399  degrees of freedom
## Residual deviance:   32.228  on 1379  degrees of freedom
## AIC: 74.228
##
## Number of Fisher Scoring iterations: 15
```

```
probabs <- predict(log_model_binary, testSet[,!c("price_binary")],type='response')
preds <- ifelse(probabs > 0.5, 1, 0)
```

```
confusionMatrix(factor(preds), factor(testSet$price_binary))
```

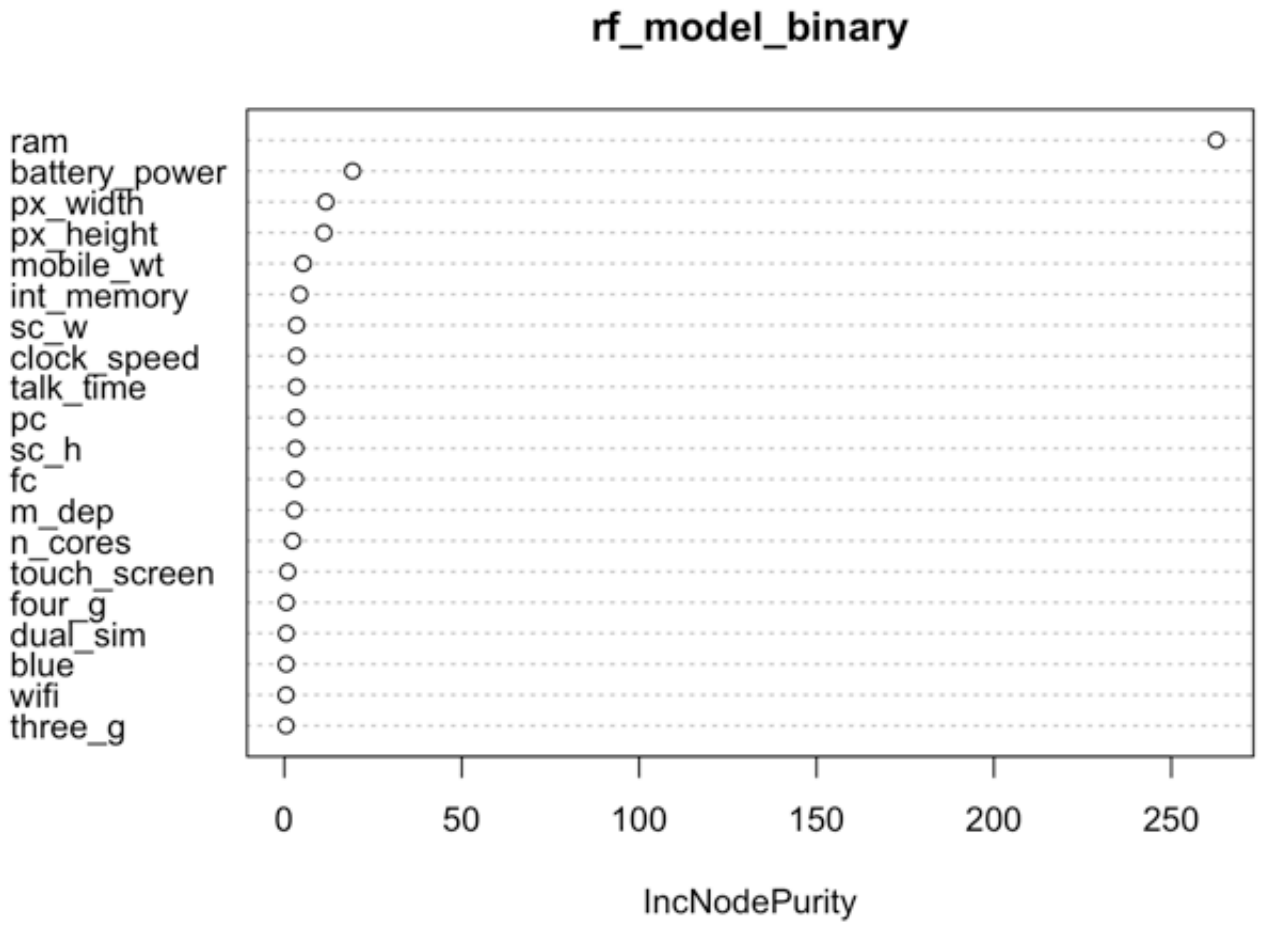
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 297   2
##          1   3 298
##
##                Accuracy : 0.9917
##                  95% CI : (0.9807, 0.9973)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9833
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9900
##             Specificity : 0.9933
##          Pos Pred Value : 0.9933
##          Neg Pred Value : 0.9900
##              Prevalence : 0.5000
##          Detection Rate : 0.4950
##    Detection Prevalence : 0.4983
##       Balanced Accuracy : 0.9917
##
##        'Positive' Class : 0
##
```

Overall, our logistic regression model is correct in roughly 99.17% of the test cases.

1.2 Random Forest

```
rf_model_binary <- randomForest(
  price_binary ~ .,
  data=trainSet
)
```

```
varImpPlot(rf_model_binary)
```




```
probabs <- predict(rf_model_binary, testSet[,!c("price_binary")])
preds <- ifelse(probabs > 0.5, 1, 0)
```

```
confusionMatrix(factor(preds), factor(testSet$price_binary))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 279  12
##          1  21 288
##
##                Accuracy : 0.945
##                  95% CI : (0.9236, 0.9618)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.89
##
##  Mcnemar's Test P-Value : 0.1637
##
##             Sensitivity : 0.9300
##             Specificity : 0.9600
##          Pos Pred Value : 0.9588
##          Neg Pred Value : 0.9320
##              Prevalence : 0.5000
##          Detection Rate : 0.4650
##    Detection Prevalence : 0.4850
##       Balanced Accuracy : 0.9450
##
##        'Positive' Class : 0
##
```

Overall, our random forest model is correct in roughly 94.5% of the test cases.

1.3 Decision Tree

```
dt_model_binary <- rpart(
  price_binary ~ .,
  data=trainSet,
  control = rpart.control(cp = 0.001)
)
```

```
summary(dt_model_binary)
```

```
## Call:
## rpart(formula = price_binary ~ ., data = trainSet, control = rpart.control(cp = 0.
001))
##   n= 1400
##
```

```
##             CP nsplit  rel error    xerror         xstd
## 1  0.712292103      0 1.00000000 1.0014266 0.0004618705
## 2  0.040447915      1 0.28770790 0.3033506 0.0246951258
## 3  0.014897602      3 0.20681207 0.2273332 0.0213406550
## 4  0.014470878      6 0.16211926 0.2025358 0.0209735991
## 5  0.013293301      7 0.14764838 0.2044743 0.0211132382
## 6  0.010886359      8 0.13435508 0.1947813 0.0206726764
## 7  0.009012297     10 0.11258236 0.1829114 0.0201548729
## 8  0.008266573     11 0.10357007 0.1757567 0.0198066886
## 9  0.004085576     14 0.07751079 0.1672664 0.0192980114
## 10 0.001563161     15 0.07342521 0.1599929 0.0183030364
## 11 0.001088435     16 0.07186205 0.1588639 0.0179122533
## 12 0.001000000     17 0.07077361 0.1612110 0.0180007744
##
## Variable importance
##          ram battery_power     px_height      px_width    int_memory
##           70             9             8             3             3
##         sc_w            pc     talk_time     mobile_wt
##            2             2             1             1
##
## Node number 1: 1400 observations,    complexity param=0.7122921
##   mean=0.5, MSE=0.25
##   left son=2 (736 obs) right son=3 (664 obs)
##   Primary splits:
##       ram           < 2231.5 to the left,  improve=0.712292100, (0 missing)
##       battery_power < 1474   to the left,  improve=0.030292390, (0 missing)
##       px_width      < 1558.5 to the left,  improve=0.019302290, (0 missing)
##       px_height     < 1154.5 to the left,  improve=0.011284800, (0 missing)
##       touch_screen  < 0.5    to the right, improve=0.005308434, (0 missing)
##   Surrogate splits:
##       px_height     < 275.5  to the right, agree=0.551, adj=0.054, (0 split)
##       int_memory    < 42.5   to the left,  agree=0.539, adj=0.027, (0 split)
##       battery_power < 1707.5 to the left,  agree=0.537, adj=0.024, (0 split)
##       sc_w          < 10.5   to the left,  agree=0.537, adj=0.024, (0 split)
##       pc            < 14.5   to the left,  agree=0.536, adj=0.023, (0 split)
##
## Node number 2: 736 observations,    complexity param=0.04044791
##   mean=0.09918478, MSE=0.08934716
##   left son=4 (572 obs) right son=5 (164 obs)
##   Primary splits:
##       ram           < 1740   to the left,  improve=0.18836370, (0 missing)
##       battery_power < 1466.5 to the left,  improve=0.09868195, (0 missing)
##       px_height     < 698.5  to the left,  improve=0.07308115, (0 missing)
##       px_width      < 1489   to the left,  improve=0.05335825, (0 missing)
##       mobile_wt     < 81.5   to the right, improve=0.01131750, (0 missing)
##   Surrogate splits:
##       mobile_wt < 81.5   to the right, agree=0.78, adj=0.012, (0 split)
```

```
##       px_height < 5.5    to the right, agree=0.78, adj=0.012, (0 split)
##
## Node number 3: 664 observations,    complexity param=0.0148976
##   mean=0.9442771, MSE=0.05261785
##   left son=6 (61 obs) right son=7 (603 obs)
##   Primary splits:
##       ram           < 2369.5 to the left,  improve=0.14239120, (0 missing)
##       battery_power < 572.5  to the left,  improve=0.05122474, (0 missing)
##       px_width      < 939    to the left,  improve=0.04614000, (0 missing)
##       px_height     < 548.5  to the left,  improve=0.04327061, (0 missing)
##       clock_speed   < 0.65   to the left,  improve=0.01087310, (0 missing)
##
## Node number 4: 572 observations,    complexity param=0.008266573
##   mean=0.02972028, MSE=0.02883698
##   left son=8 (491 obs) right son=9 (81 obs)
##   Primary splits:
##       px_height     < 1191   to the left,  improve=0.09783483, (0 missing)
##       ram           < 1481   to the left,  improve=0.08397591, (0 missing)
##       px_width      < 1618   to the left,  improve=0.04748299, (0 missing)
##       battery_power < 1466.5 to the left,  improve=0.04105284, (0 missing)
##       sc_w          < 15.5   to the left,  improve=0.01393621, (0 missing)
##
## Node number 5: 164 observations,    complexity param=0.04044791
##   mean=0.3414634, MSE=0.2248662
##   left son=10 (111 obs) right son=11 (53 obs)
##   Primary splits:
##       battery_power < 1485   to the left,  improve=0.43187840, (0 missing)
##       px_height     < 695    to the left,  improve=0.22531990, (0 missing)
##       px_width      < 1124.5 to the left,  improve=0.16184420, (0 missing)
##       sc_h          < 12.5   to the right, improve=0.03212933, (0 missing)
##       int_memory    < 56.5   to the right, improve=0.02909248, (0 missing)
##   Surrogate splits:
##       talk_time   < 18.5   to the left,  agree=0.713, adj=0.113, (0 split)
##       px_width    < 550.5  to the right, agree=0.701, adj=0.075, (0 split)
##       clock_speed < 2.95   to the left,  agree=0.689, adj=0.038, (0 split)
##       mobile_wt   < 196.5  to the left,  agree=0.689, adj=0.038, (0 split)
##
## Node number 6: 61 observations,    complexity param=0.0148976
##   mean=0.6721311, MSE=0.2203709
##   left son=12 (32 obs) right son=13 (29 obs)
##   Primary splits:
##       px_width      < 1175   to the left,  improve=0.3539752, (0 missing)
##       battery_power < 1467.5 to the left,  improve=0.3164140, (0 missing)
##       px_height     < 551.5  to the left,  improve=0.3047997, (0 missing)
##       int_memory    < 19.5   to the right, improve=0.1408600, (0 missing)
##       ram           < 2346   to the right, improve=0.1220548, (0 missing)
##   Surrogate splits:
```

```
##       px_height     < 602    to the left,  agree=0.803, adj=0.586, (0 split)
##       mobile_wt     < 125    to the right, agree=0.656, adj=0.276, (0 split)
##       battery_power < 800    to the right, agree=0.639, adj=0.241, (0 split)
##       int_memory    < 17     to the right, agree=0.639, adj=0.241, (0 split)
##       sc_w          < 6.5    to the left,  agree=0.639, adj=0.241, (0 split)
##
## Node number 7: 603 observations,    complexity param=0.01088636
##   mean=0.9718076, MSE=0.02739756
##   left son=14 (112 obs) right son=15 (491 obs)
##   Primary splits:
##       battery_power < 768    to the left,  improve=0.09308335, (0 missing)
##       ram           < 2649.5 to the left,  improve=0.08991420, (0 missing)
##       px_height     < 544.5  to the left,  improve=0.02183305, (0 missing)
##       px_width      < 1004.5 to the left,  improve=0.01917885, (0 missing)
##       mobile_wt     < 181.5  to the right, improve=0.01475055, (0 missing)
##
## Node number 8: 491 observations,    complexity param=0.001563161
##   mean=0.00814664, MSE=0.008080272
##   left son=16 (484 obs) right son=17 (7 obs)
##   Primary splits:
##       battery_power < 1986.5 to the left,  improve=0.137900000, (0 missing)
##       ram           < 1610.5 to the left,  improve=0.075804240, (0 missing)
##       px_height     < 864    to the left,  improve=0.016812010, (0 missing)
##       px_width      < 1496.5 to the left,  improve=0.012182720, (0 missing)
##       talk_time     < 11.5   to the left,  improve=0.008520283, (0 missing)
##
## Node number 9: 81 observations,    complexity param=0.008266573
##   mean=0.1604938, MSE=0.1347356
##   left son=18 (54 obs) right son=19 (27 obs)
##   Primary splits:
##       ram           < 1313.5 to the left,  improve=0.38235290, (0 missing)
##       battery_power < 1459.5 to the left,  improve=0.16277540, (0 missing)
##       m_dep         < 0.15   to the right, improve=0.11145310, (0 missing)
##       pc            < 9.5    to the right, improve=0.08096388, (0 missing)
##       fc            < 0.5    to the right, improve=0.06148236, (0 missing)
##   Surrogate splits:
##       px_height < 1209.5 to the right, agree=0.704, adj=0.111, (0 split)
##       m_dep     < 0.15   to the right, agree=0.691, adj=0.074, (0 split)
##       px_width  < 1220.5 to the right, agree=0.691, adj=0.074, (0 split)
##       sc_h      < 6.5    to the right, agree=0.679, adj=0.037, (0 split)
##       sc_w      < 15.5   to the left,  agree=0.679, adj=0.037, (0 split)
##
## Node number 10: 111 observations,    complexity param=0.0132933
##   mean=0.1261261, MSE=0.1102183
##   left son=20 (95 obs) right son=21 (16 obs)
##   Primary splits:
##       px_height     < 1077.5 to the left,  improve=0.38029800, (0 missing)
```

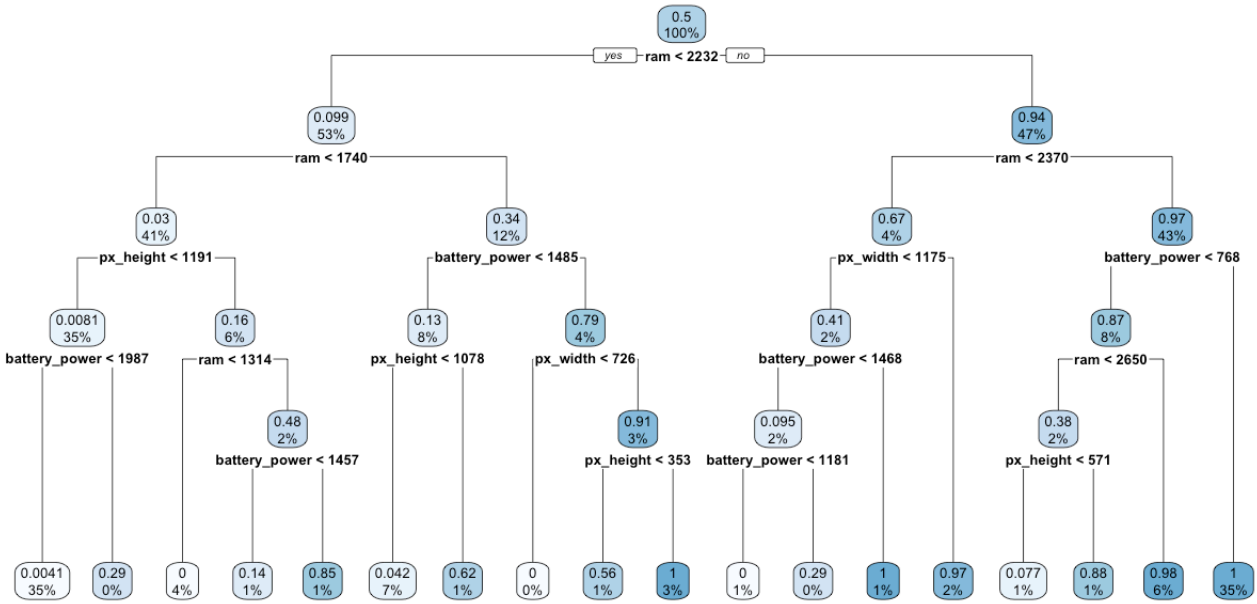
```
##       px_width      < 1559   to the left,  improve=0.20568550, (0 missing)
##       sc_h          < 9.5    to the right, improve=0.11948430, (0 missing)
##       battery_power < 917    to the left,  improve=0.07179772, (0 missing)
##       ram           < 2108.5 to the left,  improve=0.06120124, (0 missing)
##
## Node number 11: 53 observations,    complexity param=0.01447088
##   mean=0.7924528, MSE=0.1644713
##   left son=22 (7 obs) right son=23 (46 obs)
##   Primary splits:
##       px_width  < 726    to the left,  improve=0.58102770, (0 missing)
##       px_height < 604.5  to the left,  improve=0.36904760, (0 missing)
##       ram       < 1928.5 to the left,  improve=0.15740010, (0 missing)
##       m_dep     < 0.15   to the left,  improve=0.12747670, (0 missing)
##       sc_w      < 3.5    to the right, improve=0.09146333, (0 missing)
##   Surrogate splits:
##       talk_time < 19.5   to the right, agree=0.906, adj=0.286, (0 split)
##       fc        < 12.5   to the right, agree=0.887, adj=0.143, (0 split)
##
## Node number 12: 32 observations,    complexity param=0.0148976
##   mean=0.40625, MSE=0.2412109
##   left son=24 (21 obs) right son=25 (11 obs)
##   Primary splits:
##       battery_power < 1467.5 to the left,  improve=0.7655678, (0 missing)
##       int_memory    < 30.5   to the right, improve=0.2237938, (0 missing)
##       px_height     < 144.5  to the left,  improve=0.1915789, (0 missing)
##       fc            < 4.5    to the left,  improve=0.1805154, (0 missing)
##       ram           < 2297.5 to the left,  improve=0.1427800, (0 missing)
##   Surrogate splits:
##       int_memory < 30.5   to the right, agree=0.812, adj=0.455, (0 split)
##       px_height  < 551.5  to the left,  agree=0.781, adj=0.364, (0 split)
##       fc         < 4.5    to the left,  agree=0.719, adj=0.182, (0 split)
##       m_dep      < 0.15   to the right, agree=0.688, adj=0.091, (0 split)
##       n_cores    < 2.5    to the right, agree=0.688, adj=0.091, (0 split)
##
## Node number 13: 29 observations
##   mean=0.9655172, MSE=0.0332937
##
## Node number 14: 112 observations,    complexity param=0.01088636
##   mean=0.8660714, MSE=0.1159917
##   left son=28 (21 obs) right son=29 (91 obs)
##   Primary splits:
##       ram         < 2649.5 to the left,  improve=0.46821750, (0 missing)
##       px_height   < 490.5  to the left,  improve=0.10548560, (0 missing)
##       px_width    < 1004.5 to the left,  improve=0.08450393, (0 missing)
##       m_dep       < 0.65   to the left,  improve=0.04915946, (0 missing)
##       clock_speed < 0.65   to the left,  improve=0.04301627, (0 missing)
##   Surrogate splits:
```

```
##       sc_w < 15.5   to the right, agree=0.830, adj=0.095, (0 split)
##       pc   < 0.5    to the left,  agree=0.821, adj=0.048, (0 split)
##
## Node number 15: 491 observations
##   mean=0.9959267, MSE=0.004056728
##
## Node number 16: 484 observations
##   mean=0.004132231, MSE=0.004115156
##
## Node number 17: 7 observations
##   mean=0.2857143, MSE=0.2040816
##
## Node number 18: 54 observations
##   mean=0, MSE=0
##
## Node number 19: 27 observations,    complexity param=0.008266573
##   mean=0.4814815, MSE=0.2496571
##   left son=38 (14 obs) right son=39 (13 obs)
##   Primary splits:
##       battery_power < 1457   to the left,  improve=0.4946263, (0 missing)
##       pc            < 9.5    to the right, improve=0.2747253, (0 missing)
##       fc            < 0.5    to the right, improve=0.2390433, (0 missing)
##       sc_w          < 4.5    to the left,  improve=0.1663649, (0 missing)
##       clock_speed   < 2.05   to the right, improve=0.1607535, (0 missing)
##   Surrogate splits:
##       pc          < 5.5    to the right, agree=0.667, adj=0.308, (0 split)
##       px_height   < 1228.5 to the right, agree=0.667, adj=0.308, (0 split)
##       talk_time   < 5.5    to the right, agree=0.667, adj=0.308, (0 split)
##       clock_speed < 2.35   to the right, agree=0.630, adj=0.231, (0 split)
##       int_memory  < 26     to the right, agree=0.630, adj=0.231, (0 split)
##
## Node number 20: 95 observations
##   mean=0.04210526, MSE=0.04033241
##
## Node number 21: 16 observations
##   mean=0.625, MSE=0.234375
##
## Node number 22: 7 observations
##   mean=0, MSE=0
##
## Node number 23: 46 observations,    complexity param=0.004085576
##   mean=0.9130435, MSE=0.07939509
##   left son=46 (9 obs) right son=47 (37 obs)
##   Primary splits:
##       px_height < 352.5  to the left,  improve=0.3915344, (0 missing)
##       px_width  < 1230.5 to the left,  improve=0.2417582, (0 missing)
##       ram       < 1917   to the left,  improve=0.1624650, (0 missing)
```

```
##       pc        < 5.5    to the left,  improve=0.1366110, (0 missing)
##       fc        < 4.5    to the left,  improve=0.1038961, (0 missing)
##   Surrogate splits:
##       battery_power < 1927   to the right, agree=0.848, adj=0.222, (0 split)
##       n_cores       < 1.5    to the left,  agree=0.826, adj=0.111, (0 split)
##
## Node number 24: 21 observations,    complexity param=0.001088435
##   mean=0.0952381, MSE=0.0861678
##   left son=48 (14 obs) right son=49 (7 obs)
##   Primary splits:
##       battery_power < 1180.5 to the left,  improve=0.2105263, (0 missing)
##       four_g        < 0.5    to the right, improve=0.2105263, (0 missing)
##       px_height     < 393    to the left,  improve=0.1710526, (0 missing)
##       sc_w          < 7.5    to the left,  improve=0.1710526, (0 missing)
##       touch_screen  < 0.5    to the right, improve=0.1710526, (0 missing)
##   Surrogate splits:
##       n_cores     < 7.5    to the left,  agree=0.810, adj=0.429, (0 split)
##       mobile_wt   < 178    to the left,  agree=0.762, adj=0.286, (0 split)
##       px_width    < 683    to the right, agree=0.762, adj=0.286, (0 split)
##       sc_w        < 7.5    to the left,  agree=0.762, adj=0.286, (0 split)
##       clock_speed < 1.85   to the left,  agree=0.714, adj=0.143, (0 split)
##
## Node number 25: 11 observations
##   mean=1, MSE=0
##
## Node number 28: 21 observations,    complexity param=0.009012297
##   mean=0.3809524, MSE=0.2358277
##   left son=56 (13 obs) right son=57 (8 obs)
##   Primary splits:
##       px_height   < 571    to the left,  improve=0.63692680, (0 missing)
##       px_width    < 1290.5 to the left,  improve=0.50080130, (0 missing)
##       m_dep       < 0.65   to the left,  improve=0.35539940, (0 missing)
##       sc_w        < 4.5    to the right, improve=0.09695513, (0 missing)
##       clock_speed < 0.95   to the right, improve=0.08012821, (0 missing)
##   Surrogate splits:
##       px_width  < 1290.5 to the left,  agree=0.762, adj=0.375, (0 split)
##       sc_h      < 17.5   to the left,  agree=0.762, adj=0.375, (0 split)
##       m_dep     < 0.65   to the left,  agree=0.714, adj=0.250, (0 split)
##       mobile_wt < 134.5  to the left,  agree=0.714, adj=0.250, (0 split)
##       ram       < 2600.5 to the left,  agree=0.667, adj=0.125, (0 split)
##
## Node number 29: 91 observations
##   mean=0.978022, MSE=0.02149499
##
## Node number 38: 14 observations
##   mean=0.1428571, MSE=0.122449
##
```

```
## Node number 39: 13 observations
##   mean=0.8461538, MSE=0.1301775
##
## Node number 46: 9 observations
##   mean=0.5555556, MSE=0.2469136
##
## Node number 47: 37 observations
##   mean=1, MSE=0
##
## Node number 48: 14 observations
##   mean=0, MSE=0
##
## Node number 49: 7 observations
##   mean=0.2857143, MSE=0.2040816
##
## Node number 56: 13 observations
##   mean=0.07692308, MSE=0.07100592
##
## Node number 57: 8 observations
##   mean=0.875, MSE=0.109375
```

```
rpart.plot(dt_model_binary)
```




```
probabs <- predict(dt_model_binary, testSet[,!c("price_binary")])
preds <- ifelse(probabs > 0.5, 1, 0)
```

```
confusionMatrix(table(testSet$price_binary, preds))
```

```
## Confusion Matrix and Statistics
##
##    preds
##       0   1
##   0 278  22
##   1  16 284
##
##                Accuracy : 0.9367
##                  95% CI : (0.9141, 0.9548)
##     No Information Rate : 0.51
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8733
##
##  Mcnemar's Test P-Value : 0.4173
##
##             Sensitivity : 0.9456
##             Specificity : 0.9281
##          Pos Pred Value : 0.9267
##          Neg Pred Value : 0.9467
##              Prevalence : 0.4900
##          Detection Rate : 0.4633
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.9368
##
##        'Positive' Class : 0
##
```

Overall, our logistic regression model is correct in roughly 93.67% of the test cases.

2. Multiple Binary

```
m_multi <- m[, !c("price_binary")]
```

2.1 Logistic regression

```
set.seed(810)

sampleSplit_multi <- sample.split(Y=m_multi$price_range, SplitRatio=0.7)
trainSet_multi <- subset(x=m_multi, sampleSplit_multi==TRUE)
testSet_multi <- subset(x=m_multi, sampleSplit_multi==FALSE)
```

```
log_model_0 <- glm(p0 ~ ., family=binomial(link='logit'),
                   data=trainSet_multi[,!c("price_range", "p1", "p2", "p3")])
log_model_1 <- glm(p1 ~ ., family=binomial(link='logit'),
                   data=trainSet_multi[,!c("price_range","p0", "p2", "p3")])
log_model_2 <- glm(p2 ~ ., family=binomial(link='logit'),
                   data=trainSet_multi[,!c("price_range", "p0", "p1", "p3")])
log_model_3 <- glm(p3 ~ ., family=binomial(link='logit'),
                   data=trainSet_multi[,!c("price_range", "p0", "p1", "p2")])
```

```
pr0_log <- predict(log_model_0, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3
")],
               type='response')

pr1_log <- predict(log_model_1, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3
")],
               type='response')

pr2_log <- predict(log_model_2, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3
")],
               type='response')

pr3_log <- predict(log_model_3, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3
")],
               type='response')

res_log <- cbind(pr0_log,pr1_log,pr2_log,pr3_log)
label_log <- apply(res_log,1,which.max)-1
```

```
table(label_log,testSet_multi$price_range)
```

```
##
## label_log   0   1   2   3
##         0 142   4   0   0
##         1   5 113  31   0
##         2   3  33 116   3
##         3   0   0   3 147
```

```
accuracy_multi_log <- sum(label_log==testSet_multi$price_range)/length(label_log)
accuracy_multi_log
```

```
## [1] 0.8633333
```

2.2 Random forest

```
set.seed(810)

sampleSplit_multi <- sample.split(Y=m_multi$price_range, SplitRatio=0.7)
trainSet_multi <- subset(x=m_multi, sampleSplit_multi==TRUE)
testSet_multi <- subset(x=m_multi, sampleSplit_multi==FALSE)
```

```
rf_model_0 <- randomForest(p0 ~ .,
                           data=trainSet_multi[,!c("price_range","p1", "p2", "p3")])
rf_model_1 <- randomForest(p1 ~ .,
                           data=trainSet_multi[,!c("price_range","p0", "p2", "p3")])
rf_model_2 <- randomForest(p2 ~ .,
                           data=trainSet_multi[,!c("price_range","p0", "p1", "p3")])
rf_model_3 <- randomForest(p3 ~ .,
                           data=trainSet_multi[,!c("price_range","p0", "p1", "p2")])
```

```
pr0_rf <- predict(rf_model_0, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr1_rf <- predict(rf_model_1, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr2_rf <- predict(rf_model_2, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr3_rf <- predict(rf_model_3, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

res_rf <- cbind(pr0_rf,pr1_rf,pr2_rf,pr3_rf)
label_rf <- apply(res_rf,1,which.max)-1
```

```
table(label_rf,testSet_multi$price_range)
```

```
##
## label_rf   0   1   2   3
##        0 138  16   0   0
##        1  12 121  18   0
##        2   0  13 116  15
##        3   0   0  16 135
```

```
accuracy_multi_rf <- sum(label_rf==testSet_multi$price_range)/length(label_rf)
accuracy_multi_rf
```

```
## [1] 0.85
```

2.3 Decision Tree

```
set.seed(810)

sampleSplit_multi <- sample.split(Y=m_multi$price_range, SplitRatio=0.7)
trainSet_multi <- subset(x=m_multi, sampleSplit_multi==TRUE)
testSet_multi <- subset(x=m_multi, sampleSplit_multi==FALSE)
```

```
dt_model_0 <- rpart(p0 ~ ., control = rpart.control(cp = 0.001),
                    data=trainSet_multi[,!c("price_range", "p1", "p2", "p3")])
dt_model_1 <- rpart(p1 ~ ., control = rpart.control(cp = 0.001),
                    data=trainSet_multi[,!c("price_range","p0", "p2", "p3")])
dt_model_2 <- rpart(p2 ~ ., control = rpart.control(cp = 0.001),
                    data=trainSet_multi[,!c("price_range", "p0", "p1", "p3")])
dt_model_3 <- rpart(p3 ~ ., control = rpart.control(cp = 0.001),
                    data=trainSet_multi[,!c("price_range", "p0", "p1", "p2")])
```

```
pr0_dt <- predict(dt_model_0, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr1_dt <- predict(dt_model_1, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr2_dt <- predict(dt_model_2, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

pr3_dt <- predict(dt_model_3, testSet_multi[,!c("price_range","p0", "p1", "p2", "p3")
])

res_dt <- cbind(pr0_dt,pr1_dt,pr2_dt,pr3_dt)
label_dt <- apply(res_dt,1,which.max)-1
```

```
table(label_dt,testSet_multi$price_range)
```

```
##
## label_dt   0   1   2   3
##        0 132   9   1   0
##        1  18 126  17   1
##        2   0  13 113  15
##        3   0   2  19 134
```

```
accuracy_multi_dt <- sum(label_dt==testSet_multi$price_range)/length(label_dt)
accuracy_multi_dt
```

```
## [1] 0.8416667
```

3.Summary

```
accuracy_binary_summary <- matrix(c(0.9917, 0.945, 0.9367),ncol=1,byrow=TRUE)
colnames(accuracy_binary_summary) <- c("accuracy_binary")
rownames(accuracy_binary_summary) <- c("logistic regreesion","random forest","decisio
n tree")
accuracy_binary_summary <- as.table(accuracy_binary_summary)
accuracy_binary_summary
```

```
##                     accuracy_binary
## logistic regreesion          0.9917
## random forest                0.9450
## decision tree                0.9367
```

```
accuracy_multi_summary <- matrix(c(0.8633, 0.85, 0.8417),ncol=1,byrow=TRUE)
colnames(accuracy_multi_summary) <- c("accuracy_multi_binary")
rownames(accuracy_multi_summary) <- c("logistic regreesion","random forest","decision
tree")
accuracy_multi_summary <- as.table(accuracy_multi_summary)
accuracy_multi_summary
```

```
##                     accuracy_multi_binary
## logistic regreesion                0.8633
## random forest                      0.8500
## decision tree                      0.8417
```