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)
```

Load Dataset

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

```
head(m, 5)
```

```
##    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
## 2:          1021    1         0.5        1  0      1         53   0.7       136
## 3:           563    1         0.5        1  2      1         41   0.9       145
## 4:           615    1         2.5        0  0      0         10   0.8       131
## 5:          1821    1         1.2        0 13      1         44   0.6       141
##    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
## 2:       3  6       905     1988 2631   17    3         7       1            1
## 3:       5  6      1263     1716 2603   11    2         9       1            1
## 4:       6  9      1216     1786 2769   16    8        11       1            0
## 5:       2 14      1208     1212 1411    8    2        15       1            1
##    wifi price_range price_binary p0 p1 p2 p3
## 1:    1           1            0  0  1  0  0
## 2:    0           2            1  0  0  1  0
## 3:    0           2            1  0  0  1  0
## 4:    0           2            1  0  0  1  0
## 5:    0           1            0  0  1  0  0
```

```
dim(m)
```

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

Binary

Logistic regression

```
m_binary <- m[, !c("price_range", "p0", "p1", "p2", "p3")]
head(m_binary, 3)
```

```
##    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
## 2:          1021    1         0.5        1  0      1         53   0.7       136
## 3:           563    1         0.5        1  2      1         41   0.9       145
##    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
## 2:       3  6       905     1988 2631   17    3         7       1            1
## 3:       5  6      1263     1716 2603   11    2         9       1            1
##    wifi price_binary
## 1:    1            0
## 2:    0            1
## 3:    0            1
```

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)
```

```
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, 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.

Random Forest

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

```
probabs <- predict(rf_model_binary, testSet)
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.

Multi

Logistic regression

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

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

pr1 <- predict(log_model_1, testSet_multi[,!c("price_range","p0", "p2", "p3")],
               type='response')

pr2 <- predict(log_model_2, testSet_multi[,!c("price_range","p1", "p0", "p3")],
               type='response')

pr3 <- predict(log_model_3, testSet_multi[,!c("price_range","p1", "p2", "p0")],
               type='response')

res <- cbind(pr0,pr1,pr2,pr3)
label <- apply(res,1,which.max)-1
```

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

```
##
## label   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 <- sum(label==testSet_multi$price_range)/length(label)
accuracy_multi
```

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

Overall, the accuracy of our logistic regression model is 86.33%.