Consumer Brand Preference Prediction

Jennifer Brosnahan

7/11/2020

The goal of this quick project is to predict which computer brand customers prefer (Acer (0) or Sony (1)) on a new dataset.

Dataframe = 'complete' (train/test), 'incomplete' (new data)

y Value = brand

```
# Loading packages
library(tidyverse)
library(caret)
library(ggplot2)
library(corrplot)
library(openxlsx)
library(h2o)
```

```
# Importing datasets
complete <- read.csv(file.path('C:/Users/jlbro/OneDrive/C3T2/C3T2', 'complete.csv'), stringsAsFactors =
incomplete <- read.csv(file.path('C:/Users/jlbro/OneDrive/C3T2/C3T2', 'incomplete.csv'), stringsAsFacto

# checking structure
str(complete)
```

```
## 'data.frame':    9898 obs. of  7 variables:
##  $ salary : num  119807 106880 78021 63690 50874 ...
##  $ age    : int  45 63 23 51 20 56 24 62 29 41 ...
##  $ elevel : int  0 1 0 3 3 3 4 3 4 1 ...
##  $ car    : int  14 11 15 6 14 14 8 3 17 5 ...
##  $ zipcode: int  4 6 2 5 4 3 5 0 0 4 ...
##  $ credit : num  442038 45007 48795 40889 352951 ...
##  $ brand  : int  0 1 0 1 0 1 1 1 0 1 ...
```

```
# checking descriptive stats
summary(complete)
```

```
##      salary            age            elevel           car
##  Min.   : 20000   Min.   :20.00   Min.   :0.000   Min.   : 1.00
```

```
##  1st Qu.: 52082   1st Qu.:35.00   1st Qu.:1.000   1st Qu.: 6.00
##  Median : 84950   Median :50.00   Median :2.000   Median :11.00
##  Mean   : 84871   Mean   :49.78   Mean   :1.983   Mean   :10.52
##  3rd Qu.:117162   3rd Qu.:65.00   3rd Qu.:3.000   3rd Qu.:15.75
##  Max.   :150000   Max.   :80.00   Max.   :4.000   Max.   :20.00
##     zipcode          credit            brand
##  Min.   :0.000   Min.   :     0   Min.   :0.0000
##  1st Qu.:2.000   1st Qu.:120807   1st Qu.:0.0000
##  Median :4.000   Median :250607   Median :1.0000
##  Mean   :4.041   Mean   :249176   Mean   :0.6217
##  3rd Qu.:6.000   3rd Qu.:374640   3rd Qu.:1.0000
##  Max.   :8.000   Max.   :500000   Max.   :1.0000
```

```
summary(complete$brand)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  1.0000  0.6217  1.0000  1.0000
```

```
# set seed
set.seed(123)

# preprocessing, no NAs
sum(is.na(complete))
```

```
## [1] 0
```

```
# converting variables
complete$brand <- as.factor(complete$brand)
complete$age <- as.integer(complete$age)
complete$elevel <- as.integer(complete$elevel)
complete$car <- as.integer(complete$car)
complete$zipcode <- as.integer(complete$zipcode)

str(complete)
```

```
## 'data.frame':    9898 obs. of  7 variables:
##  $ salary : num  119807 106880 78021 63690 50874 ...
##  $ age    : int  45 63 23 51 20 56 24 62 29 41 ...
##  $ elevel : int  0 1 0 3 3 3 4 3 4 1 ...
##  $ car    : int  14 11 15 6 14 14 8 3 17 5 ...
##  $ zipcode: int  4 6 2 5 4 3 5 0 0 4 ...
##  $ credit : num  442038 45007 48795 40889 352951 ...
##  $ brand  : Factor w/ 2 levels "0","1": 1 2 1 2 1 2 2 2 1 2 ...
```

Modeling - Classification

```
# createDataPartition() 75% and 25%
index <- createDataPartition(complete$brand, p=0.75, list = FALSE)
trainSet <- complete[ index,]
```

```r
testSet <- complete[-index,]

# Check structure of trainSet
str(trainSet)
```

```
## 'data.frame':    7424 obs. of  7 variables:
##  $ salary : num  119807 106880 78021 63690 130813 ...
##  $ age    : int  45 63 23 51 56 24 62 29 48 52 ...
##  $ elevel : int  0 1 0 3 3 4 3 4 4 1 ...
##  $ car    : int  14 11 15 6 14 8 3 17 16 6 ...
##  $ zipcode: int  4 6 2 5 3 5 0 0 5 0 ...
##  $ credit : num  442038 45007 48795 40889 135943 ...
##  $ brand  : Factor w/ 2 levels "0","1": 1 2 1 2 2 2 2 1 2 1 ...
```

```r
# setting cross validation
control <- trainControl(method = 'repeatedcv',
                        number=10,
                        repeats = 1)
```

```r
# train and automatic tuning
rfFit3 <- train(brand~.,
                data = trainSet,
                method = 'rf',
                trControl=control,
                tuneLength = 1)

rfFit3
```

```
## Random Forest
##
## 7424 samples
##    6 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 6682, 6682, 6682, 6681, 6682, 6682, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9193152  0.8290663
##
## Tuning parameter 'mtry' was held constant at a value of 2
```

```r
# train and manual tuning
rfFit4 <- train(brand~.,
                data = trainSet,
                method = 'rf',
                trControl=control,
                tuneLength = 5)

rfFit4
```

```
## Random Forest
##
## 7424 samples
##    6 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 6682, 6682, 6681, 6681, 6683, 6681, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9172956  0.8246651
##   3     0.9174297  0.8247653
##   4     0.9172944  0.8243331
##   5     0.9158117  0.8211470
##   6     0.9136570  0.8164884
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 3.
```

rfFit3 mtry = 1 (BEST MODEL):

Accuracy Kappa

0.9193152 0.8290663

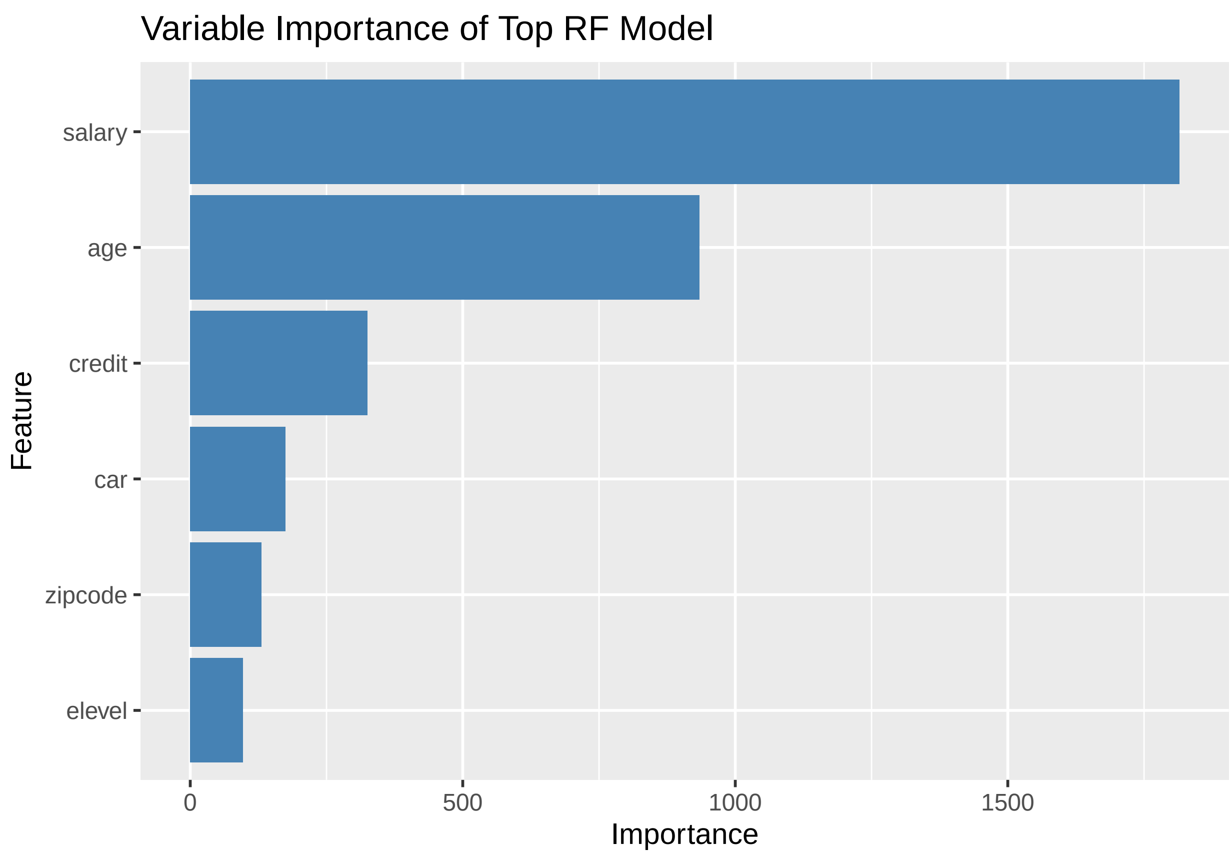
Tuning parameter 'mtry' was held constant at a value of 2

rfFit4 mtry = 2:

Accuracy Kappa

0.9174297 0.8247653

```r
# Variable importance using ggplot
ggplot(varImp(rfFit3, scale=FALSE)) +
  geom_bar(stat = 'identity', fill = 'steelblue') +
  ggtitle('Variable Importance of Top RF Model')
```




```r
# predicting on testSet using optimal rfFit3 model
rfPreds <- predict(rfFit3, newdata = testSet)

# predicting using type = 'prob' helps see prediction for each observation
rfProbs <- predict(rfFit3, newdata = testSet, type = 'prob')

# confusion matrix to see where it's right and where it's wrong
confusionMatrix(data = rfPreds, testSet$brand)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  861  107
##          1   75 1431
##
##                Accuracy : 0.9264
##                  95% CI : (0.9154, 0.9364)
##     No Information Rate : 0.6217
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8446
##
##  Mcnemar's Test P-Value : 0.02157
##
##             Sensitivity : 0.9199
```

```
##          Specificity : 0.9304
##       Pos Pred Value : 0.8895
##       Neg Pred Value : 0.9502
##           Prevalence : 0.3783
##       Detection Rate : 0.3480
## Detection Prevalence : 0.3913
##    Balanced Accuracy : 0.9252
##
##     'Positive' Class : 0
##
```

The confusion matrix shows 92.6% accuracy, 91.9% sensitivity (% of positives we are catching), and 93.1% specificty (% of negatives we are catching). This is a good model!

```r
# postResample is the only way to test if it will do well in real world OR if overfitting
# rfFit3 is not overfitting, accuracy and kappas are both near .92 and .84
postResample(rfPreds, testSet$brand)
```

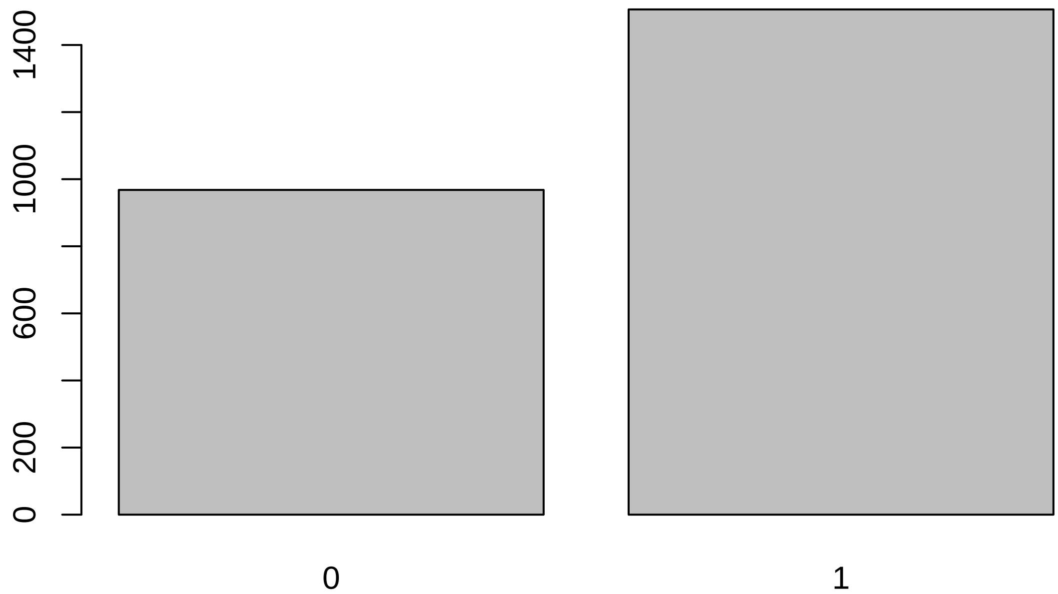
```
##  Accuracy     Kappa
## 0.9264349 0.8446495
```

```r
# Comparison of predictions to actual within same dataframe
compare_rf <- data.frame(testSet,rfPreds)
view(compare_rf)

# Summary gives count of predictions and plot gives distribution
summary(rfPreds)
```

```
##    0    1
##  968 1506
```

```r
plot(rfPreds)
```



Now we will make predictions using top model (rfFitr3) on new dataset = incomplete

```r
# import dataset and checking structure
str(incomplete)
```

```
## 'data.frame':    5000 obs. of  7 variables:
##  $ salary : num  150000 82524 115647 141443 149211 ...
##  $ age    : int  76 51 34 22 56 26 64 50 26 46 ...
##  $ elevel : int  1 1 0 3 0 4 3 3 2 3 ...
##  $ car    : int  3 8 10 18 5 12 1 9 3 18 ...
##  $ zipcode: int  3 3 2 2 3 1 2 0 4 6 ...
##  $ credit : num  377980 141658 360980 282736 215667 ...
##  $ brand  : int  1 0 1 1 1 1 1 1 1 0 ...
```

```r
# set seed
set.seed(123)

# preprocessing, no NAs
sum(is.na(incomplete))
```

```
## [1] 0
```

```r
# converting variables to integers and factors
incomplete$brand <- as.factor(incomplete$brand)
incomplete$age <- as.integer(incomplete$age)
incomplete$elevel <- as.integer(incomplete$elevel)
incomplete$car <- as.integer(incomplete$car)
incomplete$zipcode <- as.integer(incomplete$zipcode)

# check structure and summary of processed dataframe
str(incomplete)
```

```
## 'data.frame':    5000 obs. of  7 variables:
##  $ salary : num  150000 82524 115647 141443 149211 ...
##  $ age    : int  76 51 34 22 56 26 64 50 26 46 ...
##  $ elevel : int  1 1 0 3 0 4 3 3 2 3 ...
##  $ car    : int  3 8 10 18 5 12 1 9 3 18 ...
##  $ zipcode: int  3 3 2 2 3 1 2 0 4 6 ...
##  $ credit : num  377980 141658 360980 282736 215667 ...
##  $ brand  : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2 2 1 ...
```

```r
summary(incomplete)
```

```
##      salary            age            elevel           car
##  Min.   : 20000   Min.   :20.00   Min.   :0.000   Min.   : 1.0
##  1st Qu.: 52590   1st Qu.:35.00   1st Qu.:1.000   1st Qu.: 6.0
##  Median : 86221   Median :50.00   Median :2.000   Median :11.0
##  Mean   : 85794   Mean   :49.94   Mean   :2.009   Mean   :10.6
##  3rd Qu.:118535   3rd Qu.:65.00   3rd Qu.:3.000   3rd Qu.:16.0
##  Max.   :150000   Max.   :80.00   Max.   :4.000   Max.   :20.0
##     zipcode          credit         brand
##  Min.   :0.000   Min.   :     0   0:4937
##  1st Qu.:2.000   1st Qu.:122311   1:  63
##  Median :4.000   Median :250974
##  Mean   :4.038   Mean   :249546
##  3rd Qu.:6.000   3rd Qu.:375653
##  Max.   :8.000   Max.   :500000
```

Summary reveals the first 102 rows of brand have been filled in, the rest are unanswered

```r
# predicting on new data 'incomplete'
incompletePreds <- predict(rfFit3, newdata = incomplete)
str(incompletePreds)
```

```
##  Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2 2 1 ...
```

```r
# postResample on first 102 observations to determine how well model doing on test df
subset_incomplete <- incomplete %>% slice(1:103)
postResample(incompletePreds,subset_incomplete$brand)
```

```
##  Accuracy     Kappa
## 0.8640777 0.7085691
```

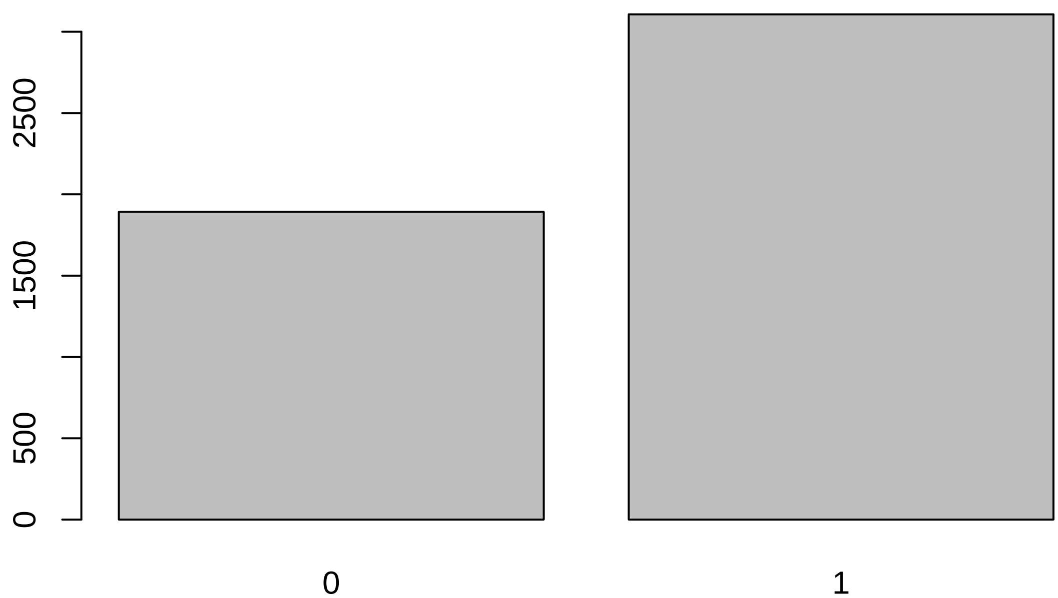
```r
# comparison of predictions to actual within the df
compare_incomplete <- data.frame(incomplete,incompletePreds)
view(compare_incomplete)
```

```r
# exporting to excel so we can see our predictions
library(openxlsx)
write.xlsx(compare_incomplete,"IncompleteComparison.xlsx")

# summary gives count of predictions and plot gives distribution
summary(incompletePreds)
```

```
##    0    1
## 1893 3107
```

```r
plot(incompletePreds)
```




```r
compare_incomplete %>%
  group_by(brand, incompletePreds) %>%
  summarise(count=n())
```

```
## `summarise()` regrouping output by 'brand' (override with `.groups` argument)
```

```
## # A tibble: 4 x 3
```

```
## # Groups:   brand [2]
##   brand incompletePreds count
##   <fct> <fct>           <int>
## 1 0     0                1888
## 2 0     1                3049
## 3 1     0                   5
## 4 1     1                  58
```