# Brand Prediction

*Marc Soler*

*17/2/2019*

```
#call relevant libraries
library(readxl)
library(caret)
library(corrplot)
library(ggplot2)
library(RColorBrewer)
library(kernlab)
library(randomForest)
library(gbm)
library(plyr)
library(dplyr)
library(reshape2)
library(party)
library(plot3D)
library(hexbin)
#import dataset
Survey <- read_excel('datasets/Survey_Key_and_Complete_Responses_excel.xlsx',
                           sheet = 2)
```

### Introoduction

We were asked by Danielle Sherman, CTO of Blackwell Electronics (an eletronic retailer) to predict the customers' brand preferences that were missing from an incomplete surveys.

These predictions were carried out following the next steps: * An intensive and descriptive exploration of the variables * A selection of the most relevant variables * Building KNN, SVM, RF & GBM Models * Building Decision Tree Models * Checking convergence for RF

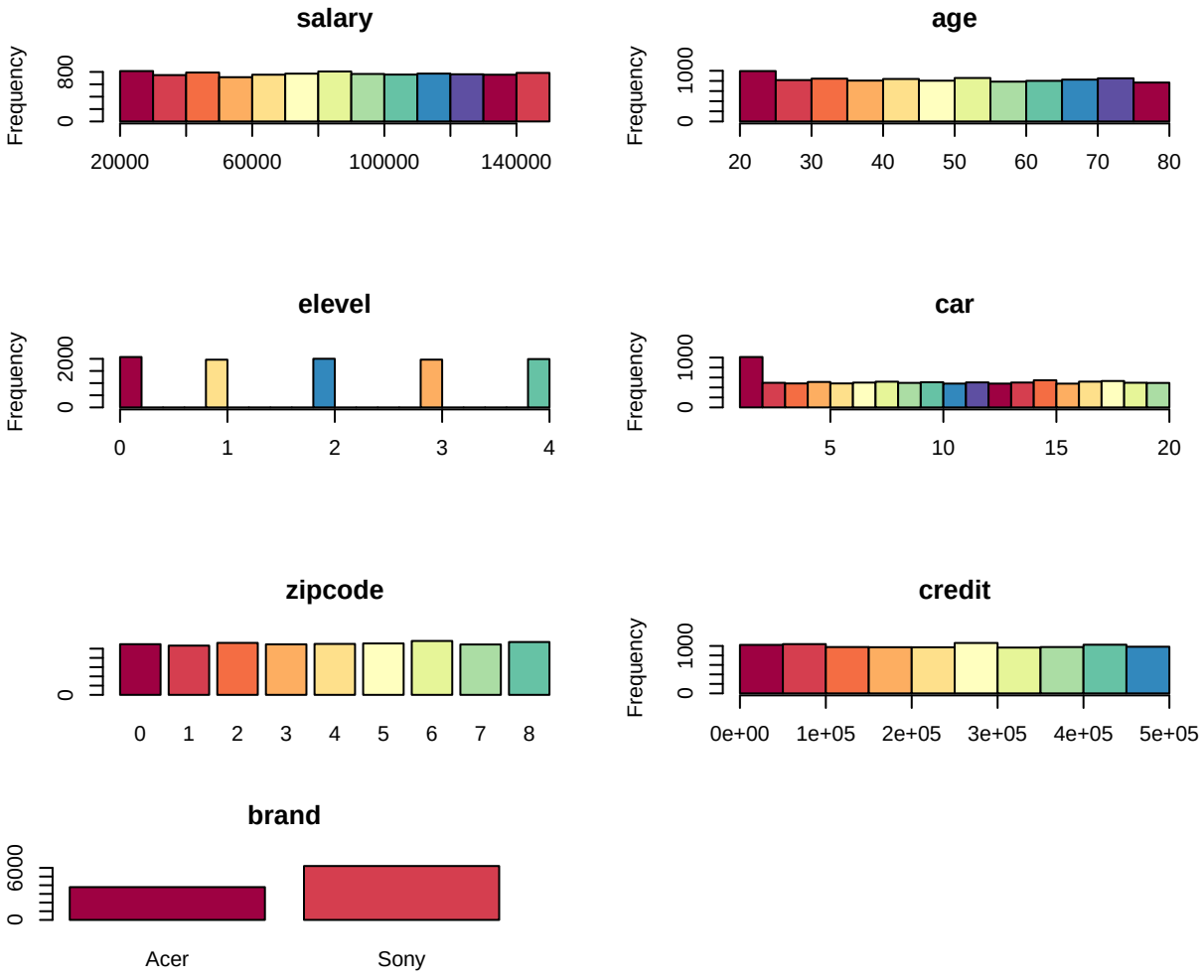
### Understanding the dataset

```
#Investigate how the dataset looks like
glimpse(Survey)
```

```
## Observations: 10,000
## Variables: 7
## $ salary  <dbl> 119806.54, 106880.48, 78020.75, 63689.94, 50873.62, 13...
## $ age     <dbl> 45, 63, 23, 51, 20, 56, 24, 62, 29, 41, 48, 52, 52, 33...
## $ elevel  <dbl> 0, 1, 0, 3, 3, 3, 4, 3, 4, 1, 4, 1, 3, 4, 2, 1, 2, 1, ...
## $ car     <dbl> 14, 11, 15, 6, 14, 14, 8, 3, 17, 5, 16, 6, 20, 13, 6, ...
## $ zipcode <dbl> 4, 6, 2, 5, 4, 3, 5, 0, 0, 4, 5, 0, 4, 3, 3, 4, 7, 2, ...
## $ credit  <dbl> 442037.71, 45007.18, 48795.32, 40888.88, 352951.50, 13...
## $ brand   <chr> "Acer", "Sony", "Acer", "Sony", "Acer", "Sony", "Sony"...
```

### How does each variable relate to brand?

```
#Relation brand-variables
p = list()
par(mfrow=c(3,2))
for(i in (1:(ncol(Survey)-1))) {
     if (names(Survey[i]) == "brand"){
       p[[i]]<-barplot(table(Survey$brand), main = colnames(Survey[i]),  col = brewer.pal(11,'Spectral'))
       }
    else if (is.numeric(Survey[[i]]) == "TRUE"){
    p[[i]]<-hist(Survey[[i]], col = brewer.pal(11,'Spectral'), xlab = ' ',
                 main = colnames(Survey[i]))
    }
    else{
    p[[i]] <- barplot(table(Survey[i]), main = colnames(Survey[i]), col = brewer.pal(11,'Spectral'))
    }
}
```




The dataset show an unusual distribution. Instead of having a normal or Gaussian bell shape, they are mostly rectangular. This happens with salary, car and age distributions.

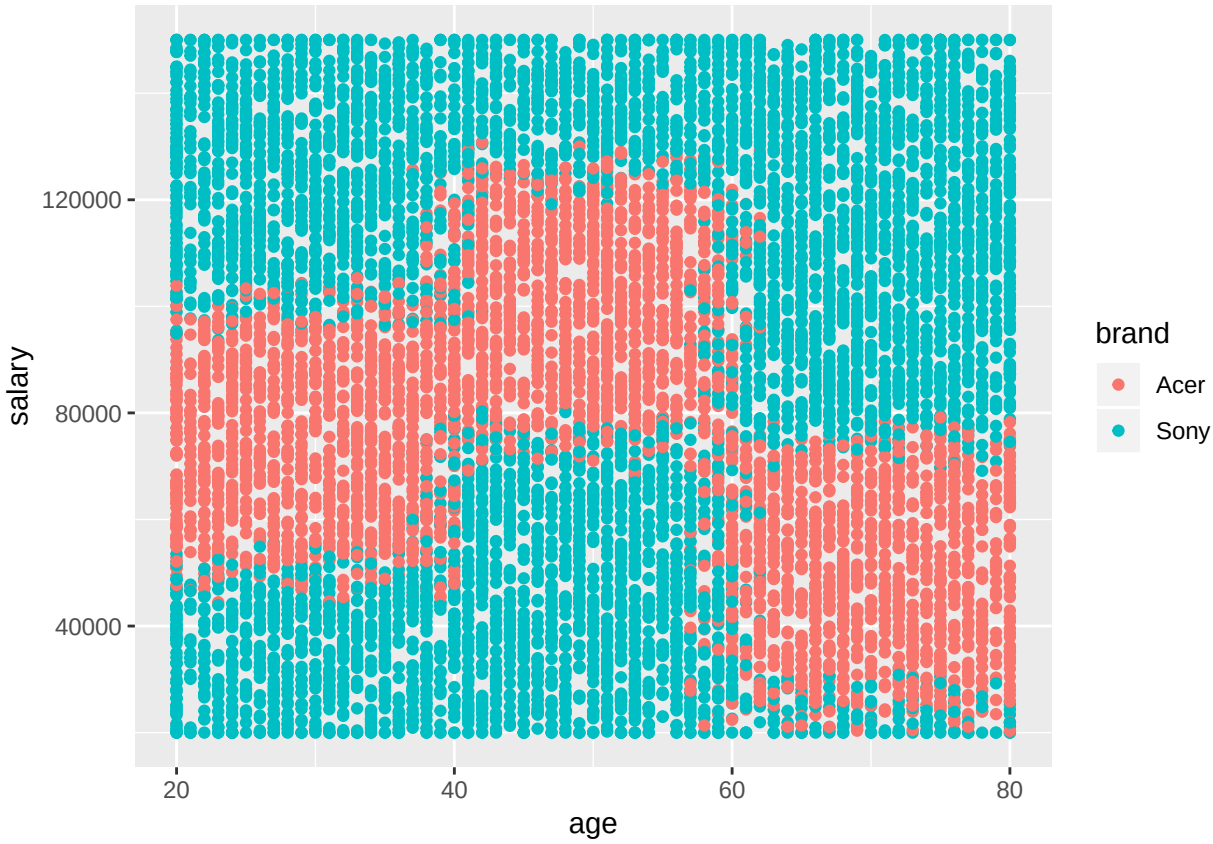
The rectangular shape is a clear indicator that the data has not been collected randomly. Instead, my guess is that data has been tried to be collected in equal amounts for all ranges of values, avoiding randomness at all cost.

It is tempting to think that since the data has not been randomly selected, it has no predicting value. This

is, however, false. Both datasets have had it's data collected the same way and, have the same distribution shape. Hence, predicting capabilities will be somehow true for the testing dataset as an isolated exercise, and results will be highly accurate, but these predictions would not be useful with the general public or real world scenario.

**How does age and salary affect buying patterns?**

```
ggplot(Survey, aes(x = age, y = salary, color = brand)) + geom_point()
```




This is a very interesting graph which deserves our upmost attention:

If we recall the previous graph, it was stated that salary and age had nothing to do with each other. Here, it is clear that the brand of choice (Acer or Sony), is clearly dependent on age & salary groups altogether.

There are some hard borders, marking the age & salary ranges with the brand choice. See appendix for more. There are 3 age ranges: [20,40),(40,60),(60,80], And 3 salary ranges: (50k, 100k), (80k,130k), [20k,80k).

Wealthier people (above 120k income), regardless of age, all prefer Sony. This can be seen as the status brand or the professional brand.

It is interesting to see that working age people but with the lowest income range, also choose Sony. This might be a psychological reason, ie, they buy 'good' products to resemble rich people because they want to improve their social status.

Middle class, regardless of the age, are money sensible, and prefer the Acer, since it does the same thing as the Sony but for less. They prefer functionality above status or cache.

### Statistical Analysis

For choosing the most relevant variable por the predictive models, we carried out three statistical analyses: * Correlation: when both variables are quantitative. (high values, high correlation) * Chi Squared: when both variables are qualitative ($p<0.05$, significative differences ) * ANOVA: when one variable is quantitative and the other is qualitative ($p<0.05$, significative differences) Only the salary seems to create significative differences in the Brand ($p=0$). Although the age doesn't have a $p<0.05$, it's close, so we're also going to consider this variable for the model.

```
# Removing redundance
MatrixTest<-matrix(ncol=7,nrow=7)
rownames(MatrixTest)<-c("Salary", "Age", "EducationalLevel", "Car", "ZipCode", "Credit", "Brand")
colnames(MatrixTest)<-c("Salary", "Age", "EducationalLevel", "Car", "ZipCode", "Credit", "Brand")

for (i in 1:(ncol(Survey)-1)){
  for (j in 1:(ncol(Survey)-1)){
    if (is.numeric(Survey[[i]]) == "TRUE" & is.numeric(Survey[[j]]) == "TRUE"){
      MatrixTest[i,j]<-(cor(Survey[[i]], Survey[[j]]))

    } else if (is.numeric(Survey[[i]]) == "TRUE" & is.numeric(Survey[[j]]) == "FALSE"){
      MatrixTest[i,j]<-(summary(aov((Survey[[i]]~Survey[[j]]), data=Survey))[[1]][[5]][1])

    } else if (is.numeric(Survey[[i]]) == "FALSE" & is.numeric(Survey[[j]]) == "TRUE"){
      MatrixTest[i,j]<-(summary(aov((Survey[[j]]~Survey[[i]]), data=Survey))[[1]][[5]][1])

    } else {
      MatrixTest[i,j]<-(chisq.test(Survey[[i]], Survey[[j]])$p.value)
    }
  }
}

MatrixTest<-round(MatrixTest, digits = 5)
library(knitr)
kable(MatrixTest)
```

| | Salary | Age | EducationalLevel | Car | ZipCode | Credit | Brand |
|---|---|---|---|---|---|---|---|
| Salary | 1.00000 | 0.00703 | -0.00821 | -0.00705 | 0.30284 | -0.02405 | 0.00000 |
| Age | 0.00703 | 1.00000 | -0.00473 | 0.01062 | 0.02954 | -0.00496 | 0.15252 |
| EducationalLevel | -0.00821 | -0.00473 | 1.00000 | 0.00041 | 0.44908 | 0.00064 | 0.67863 |
| Car | -0.00705 | 0.01062 | 0.00041 | 1.00000 | 0.38592 | -0.00951 | 0.65642 |
| ZipCode | 0.30284 | 0.02954 | 0.44908 | 0.38592 | 0.00000 | 0.29748 | 0.49532 |
| Credit | -0.02405 | -0.00496 | 0.00064 | -0.00951 | 0.29748 | 1.00000 | 0.56212 |
| Brand | 0.00000 | 0.15252 | 0.67863 | 0.65642 | 0.49532 | 0.56212 | 0.00000 |

### Check varialble Importance

```
#function controlling the settings of the model
fitControl <- trainControl(method = "repeatedcv", number = 10, repeats = 3,classProbs = FALSE)

set.seed(123)

#generate indices for splitting
```

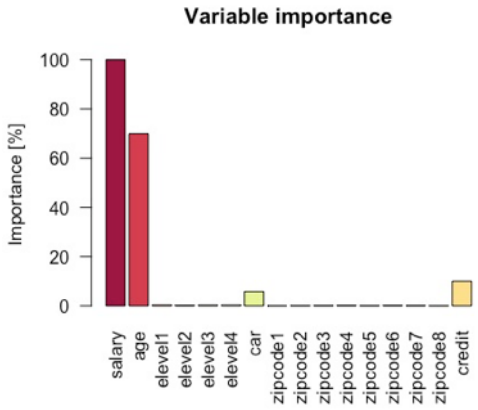



Figure 1: Variable Importance

```
set.seed(123)

indices <- createDataPartition( y = Survey$brand,
                                p = .75,
                                list = FALSE)

#split the data in two!
training <- Survey[indices,]
testing <- Survey[-indices,]

rffit_1 <- train(brand ~ .-brand_bi, data = training,method = "rf",trControl = fitControl,preProcess = c
```

```
#checking and plotting how important are variables in the random forest
importance_variables <- data.frame(c(varImp(rffit_1)))
importance_variables$labels <- rownames(importance_variables)
barplot(importance_variables$Overall, names.arg = importance_variables$labels, las = 2,
        col = brewer.pal(11, 'Spectral'), main = 'Variable importance', #xlab = 'Variables',
        ylab = 'Importance [%]')
```

```
formula <- brand ~ age + salary
dec_tree<-ctree(formula, data=training, controls = ctree_control(maxdepth=3))
plot(dec_tree, tp_args = list(beside = TRUE ))
```

Here, the decision tree shows similar results as the scatter plot (age & salary with brand). Salary and age are the no.1 factors on which computer brands customers purchase. On the right, varImp() has been called and plotted for the random forest. Again, it yields to the same results. It is not surprising that we get these results. Before, I talked about the 'hard boundaries or hard borders ' of the scatter plot. These borders actually represent the decisions or logical rules on which the random forest or decision tree splits the data.

### Running Machine Learning Algorithms: Train and Predict

```
training$brand <- as.factor(training$brand)
methods <- c('rf', 'gbm', 'knn', 'svmRadial')
formulae <- c("2var" = 'brand_bi ~ age + salary', "allvar" = 'brand_bi ~.')
error_df <- c()
```

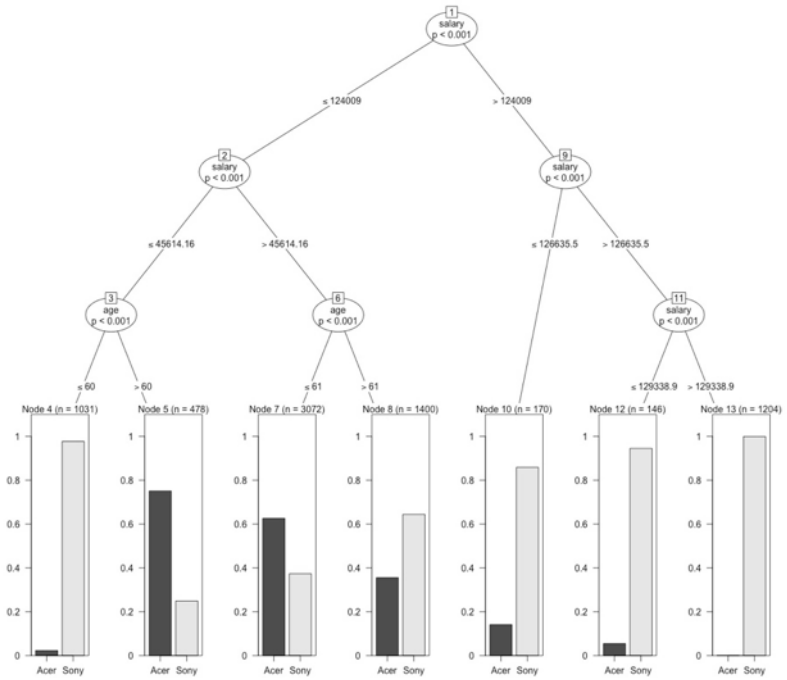



Figure 2: Decision Tree

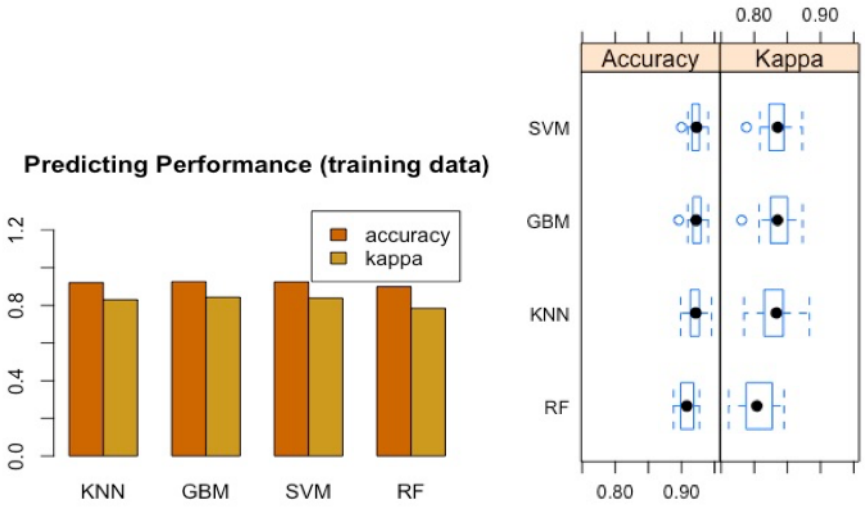
```
models <- c()
for(f in names(formulae)){
    submodels <- c()
    theformula <- as.formula(formulae[[f]])
    for(m in methods){
        model <- train(theformula,
                       data = training,
                       method = m,
                       trControl = fitControl,
                       verbose = F)
        prediction <- predict(model, newdata = testing)
        errors <- postResample(testing$brand_bi, prediction)
        error_df <- cbind(error_df,errors)
        submodels[[m]] <- model
    }
    models[[f]] <- submodels
}
```

```
x <- as.vector(outer(formulae, methods, paste, sep="."))
colnames(error_df) <- x
error_df

error_dfmelt <- melt(error_df, varnames = c("metric", "model"))
error_dfmelt <- as.data.frame(error_dfmelt)
error_dfmelt
```

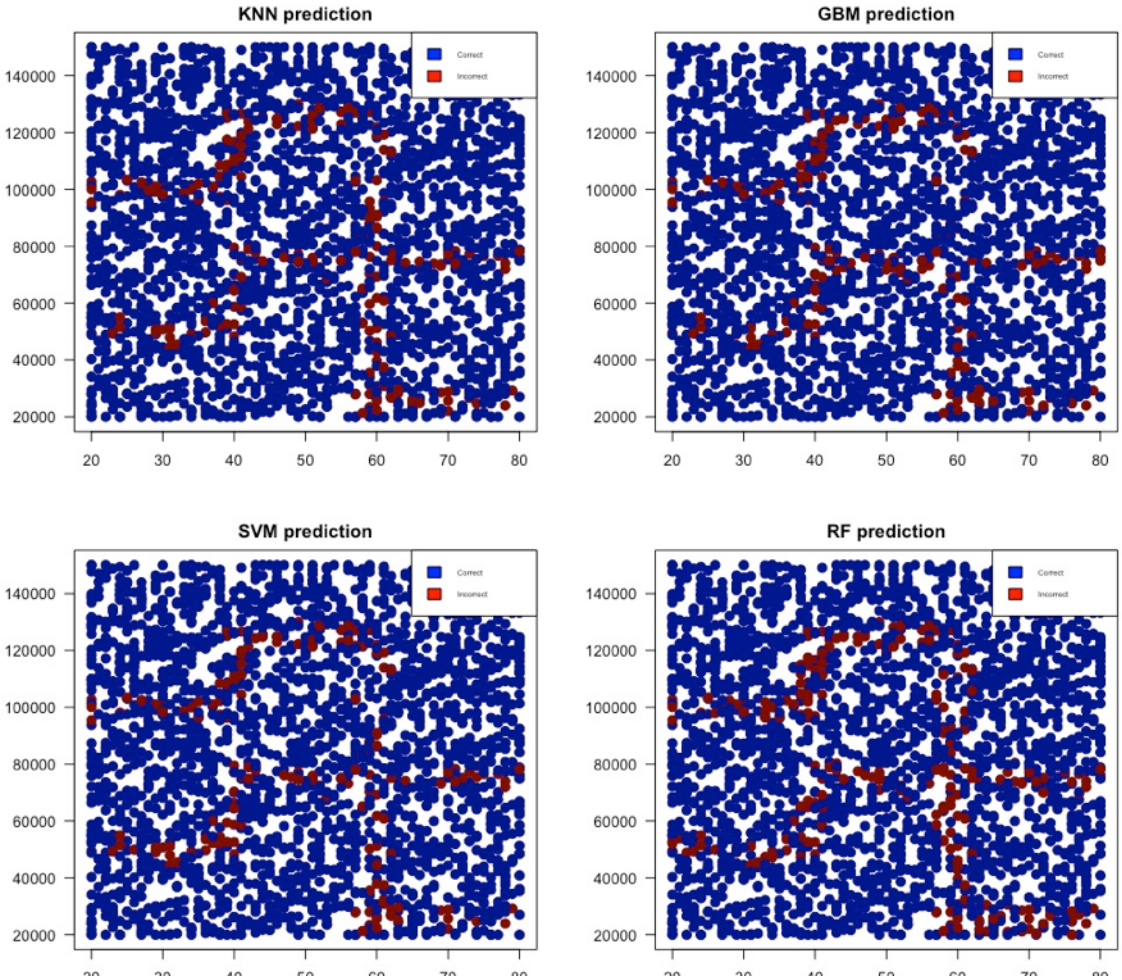
```
ggplot(error_dfmelt, aes(x=model, y=value))+
```

```
geom_col()+
facet_grid(metric~., scales="free") + theme(axis.text.x = element_text(angle = 35, hjust = 1))
```




### Testing prediction errors

This chart shows the hard borders in with the tree based models make a split in the tree. Each red line indicates the model to split the data to create another branch. The graphs actually show in red, the incorrect guesses of the 4 trained models with the testing data from the complete dataset. This can be read as a visual representation of how a tree based model makes its decisions and, why it yields such results (salary and age are very important when choosing computer brand, even if correlation matrix says it has zero correlation between the




two.)

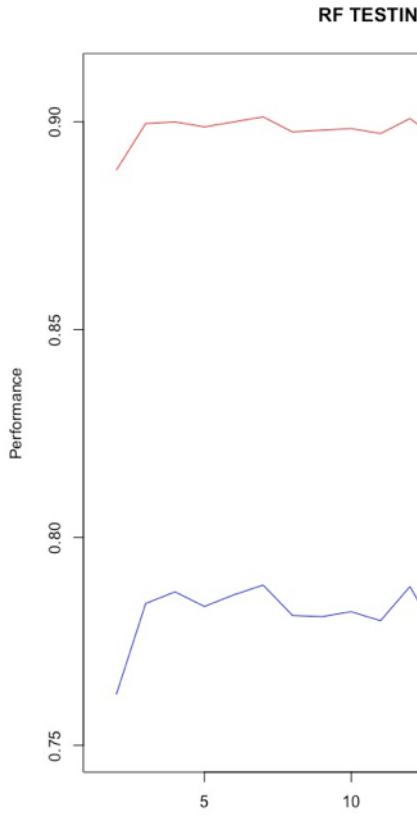
**Finding Random Forest performance convergence**

```
#write a for loop to iterate over the number of trees and find a performance convergence
results_train <- c()
results_test  <- c()
for(i in c(2:30)){
    cat(i)
    rf_it <- train(formula,
                data = training,
                method = "rf",
                ntree = i,
                trControl = fitControl,
                preProcess = c("range"),
                verbose = TRUE)
    rf_it_pred_train <- predict(rf_it, newdata = training)
    rf_it_pred_test <- predict(rf_it, newdata = testing)
    cm_rf_it_pred_train <- confusionMatrix(data = rf_it_pred_train, reference = training$brand)
    cm_rf_it_pred_test <- confusionMatrix(data = rf_it_pred_test, reference = testing$brand)
    accuracy_p_train <- cm_rf_it_pred_train$overall[1]
```

```
    accuracy_p_test <-cm_rf_it_pred_test$overall[1]
    kappa_p_train <- cm_rf_it_pred_train$overall[2]
    kappa_p_test <- cm_rf_it_pred_test$overall[2]
    results_train <- rbind(results_train, c(accuracy_p_train, kappa_p_train,i))
    results_test <- rbind(results_test, c(accuracy_p_test, kappa_p_test,i))
}
results_train <- as.data.frame(results_train)
results_test  <- as.data.frame(results_test)
colnames(results_train) <- c("Accuracy","kappa","ntree")
colnames(results_test) <- c("Accuracy","kappa","ntree")
```

```
#plot the results for the training set
plot(x = results_train$ntree, y = results_train$Accuracy, type = 'l', col = 'red',
     main = 'RF TRAINING performance on number of trees', xlab = 'Number of trees', ylab = 'Performance
lines(x = results_train$ntree, y = results_train$kappa, type = 'l', col = 'blue')
legend("bottomright", c("Kappa","Accuracy"), fill=c("blue","red"))

#plot the resutls for the testing set
plot(x = results_test$ntree, y = results_test$Accuracy, type = 'l', col = 'red',
     main = 'RF TESTING performance on number of trees', xlab = 'Number of trees', ylab = 'Performance'
     ylim = c(0.75, 0.91))
lines(x = results_test$ntree, y = results_test$kappa, type = 'l', col = 'blue')
legend("right", c("Kappa","Accuracy"), fill=c("blue","red"))
```

## Apply the selected trained ML algorithm to new unseen data

### Checking how the new data looks like

```
#import new set (incomplete) for predicting brand
testSet <- read.csv('datasets/SurveyIncomplete.csv')

#Check for missing values
sum(is.na(testSet$salary))
sum(is.na(testSet$age))
sum(is.na(testSet$elevel))
sum(is.na(testSet$zipcode))
sum(is.na(testSet$credit))

#change datatypes
testSet$brand <- factor(testSet$brand,levels = c(0,1),labels = c("Acer","Sony"))
testSet$zipcode <- as.factor(testSet$zipcode)
testSet$elevel <- as.numeric(testSet$elevel)
testSet$car <- as.numeric(testSet$car)

testSet[, c("brand", "brand_bi")] <- NA
```

### Predictions

```
predictions <- c()
# drops <- c("brand","brand_bi")
# testSet <- testSet[ , !(names(testSet) %in% drops)]
for(f in names(formulae)){
    subpredictions <- list()
    for(m in methods){
        model <- models[[f]][[m]]
        predicted <- predict(model, testSet)
        subpredictions[[m]] <- predicted
    }
    predictions[[f]] <- subpredictions
}

for(f in names(formulae)){
    for(m in methods){
       titulo <- paste(m,f)
       scatter2D(testSet$age, testSet$salary, colvar = as.numeric(predictions[[f]][[m]]),
                 xlab = 'Age',ylab = 'Salary', main = titulo, pch = 20, cex =2, colkey = FALSE,
                 las = 1)
    }
}
```

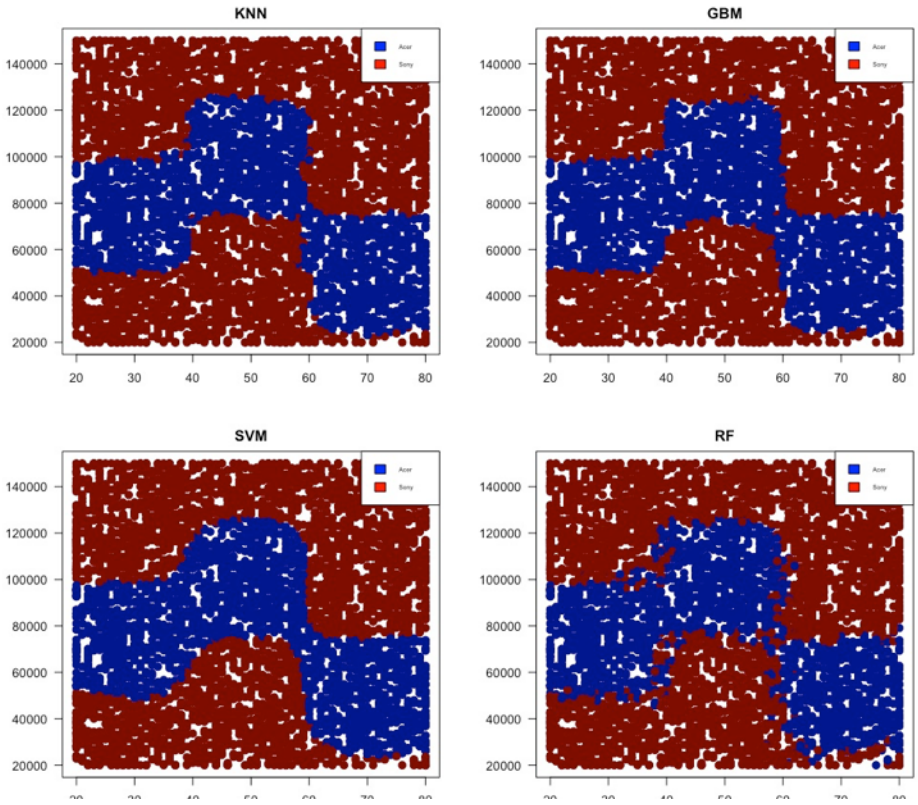



Figure 3: Algorithm prediction behaviour