Fitting_Documents

Milton Osiel Candela Leal

7/18/2021

Packages used

The following packages were used:

```
library(R.matlab)
library(dplyr)
library(caret)
library(MASS)
library(earth)
library(lattice)
```

Generalized testing

As stated on the previous milestone, Linear Discriminant Analysis (LDA) performed poorly when the number of features was reduced, and so only Random Forest (rf) and Gradient Boosting Method (gbm) would be employed from now on. Previous analysis use only the document that collected data from 1 minute (WFM.1), the current milestone presents analysis on the other documents (WFM.1.2, WFM.1.2, WFM.1.4, WFM.2, WFM.4) in order to perceive similarities and differences across documents.

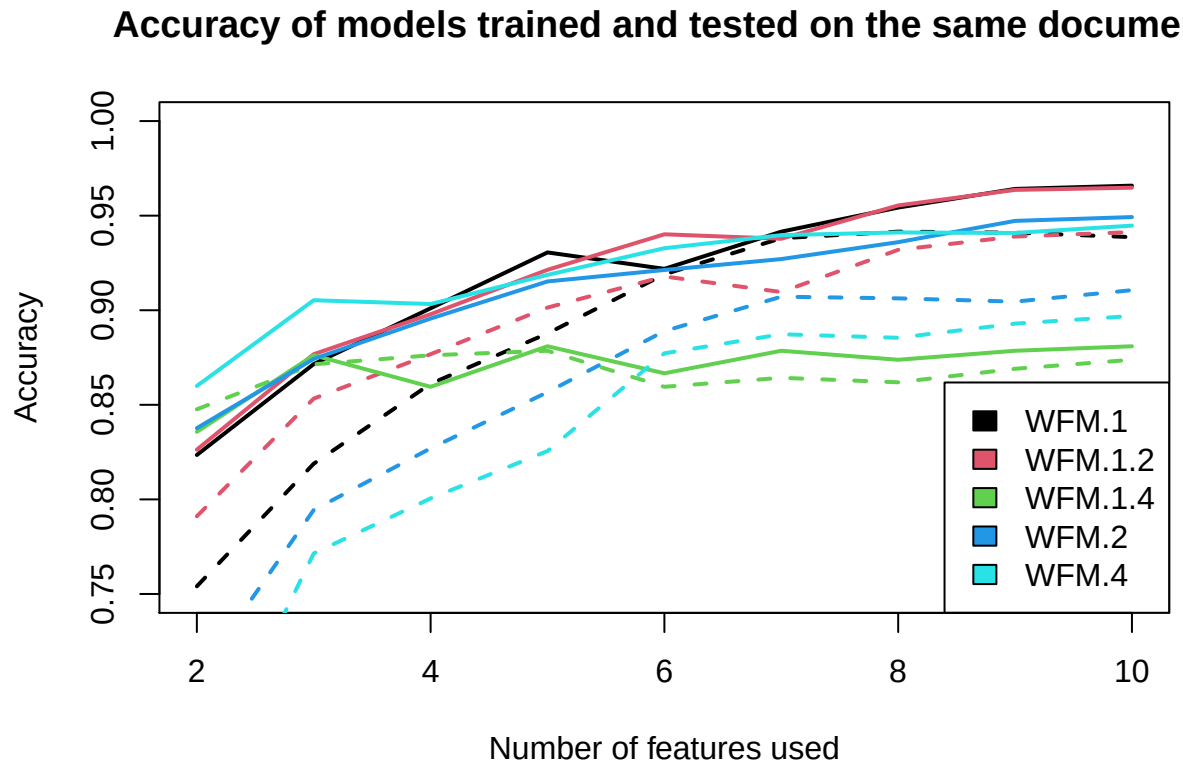
The next chunk of code is based on the chunks of the scrips of the previous milestones, it uses generalized testing for the two algorithms (rf, gbm), an incremental number of cross-validations (2 to 5) from which the mean accuracy is obtained, and an incremental number of features (2 to 10). The features that are used are the ones obtained by the previous hybrid selection techniques employed on the previous milestone (RFE, MARS).

As for the last plot, the colors of the lines represents the document on which the model was trained and tested on, while the type of line represent the model being used. The solid line represents the accuracy of the rf model, while the dashed line represents the accuracy of the gbm model.

```
geneTesting <- function(df){
     set.seed(1002)
     trainIndex <- createDataPartition(df$Clas, p = 0.70, list = FALSE)
     training <- df[trainIndex,]
     testing <- df[-trainIndex,]

     prob_modelo <- function(training, met, testing, control){

          if(met == 'gbm'){
               modelo <- train(Clas ~ ., data = training, method = met,
                               trControl = control, verbose = FALSE)
          } else{
```

```
                modelo <- train(Clas ~ ., data = training, method = met,
                                trControl = control, tuneLength = length(train))
          }

          Pred <- predict(modelo, testing)
          acc <- round(confusionMatrix(testing$Clas, Pred)$overall[1], 5)
          return(acc)
     }

     n <- 5
     metodos <- c('rf', 'gbm')
     df_modelos <- data.frame(rf = NA, gbm = NA)
     for(metodo in metodos){
          accs <- c()
          for(num in 2:n){
               control <- trainControl(method = 'cv', number = num)
               acc <- prob_modelo(training, metodo, testing, control)
               accs <- c(accs, acc)
          }
          df_modelos[,metodo] <- mean(accs)
     }
     return(df_modelos)
}

rfgbmTest <- function(df){
     n <- length(df) - 1
     dfEval <- data.frame(n_vars = 2:n, rf = NA, gbm = NA)
     for(i in 2:n){
          dfEval[dfEval$n_vars == i,c('rf', 'gbm')] <- geneTesting(df[,c(1:i, length(df))])
     }
     return(dfEval)
}

getDF <- function(doc){
     data <- readMat('Data/Raw/FeatureMatrices.mat')
     unmin <- as.data.frame(data[doc])
     top10 <- c("a-g_P8", "Gamma_P8", "Alpha_P8", "d-g_P8",
                "t-g_O1", "Theta_O1", "Gamma_O1", "t-g_P7",
                "Beta_FP1", "t-g_FP1", "Clas")

     unmin <- as.data.frame(data[doc])

     feature_names <- unlist(data$FMinfo)
     feature_names <- gsub(' \\(', '_', feature_names)
     feature_names <- gsub(')', '', feature_names)
     feature_names <- gsub('`', '', feature_names)
     feature_names <- gsub('\\/', '-', feature_names)

     colnames(unmin) <- feature_names

     df <- unmin[,top10]
     df <- filter_all(df, all_vars(. < 10))
     df <- filter_all(df, all_vars(. > -10))
```

```
     df <- mutate(df, Clas = as.factor(Clas))
     return(df)
}

pruebasets <- function(doc){
     df <- getDF(doc)
     set.seed(2031)
     df <- upSample(df[,-length(df)], df$Clas, yname = 'Clas')
     dfEval <- rfgbmTest(df)
     return(dfEval)
}

docs <- c('WFM.1', 'WFM.1.2', 'WFM.1.4', 'WFM.2', 'WFM.4')
plot(0,0, xlim = c(2,10), ylim = c(0.75,1), ylab = 'Accuracy',
     xlab = 'Number of features used',
     main = 'Accuracy of models trained and tested on the same document')
for(doc in docs){
     df <- pruebasets(doc)
     lines(df$rf ~ df$n_vars, col = which(doc == docs), lwd = 2, lty = 1)
     lines(df$gbm ~ df$n_vars, col = which(doc == docs), lwd = 2, lty = 2)
}
legend('bottomright', legend = docs, fill = 1:5)
```



As it can be seen on the plot, it seems that the rf model outperforms the gbm model on all documents. Although the gbm model catches up the rf model when the number of features increased, the main objective is to create a reliable model with the least amount of features, and so rf seems to be the candidate to perform

this task.

Get a model

A function to get a predictive model is then created, it uses the 1 minute document and five features, an Up-Sampling technique in order to solve the class imbalance and a 70:30 partition on training, testing data.

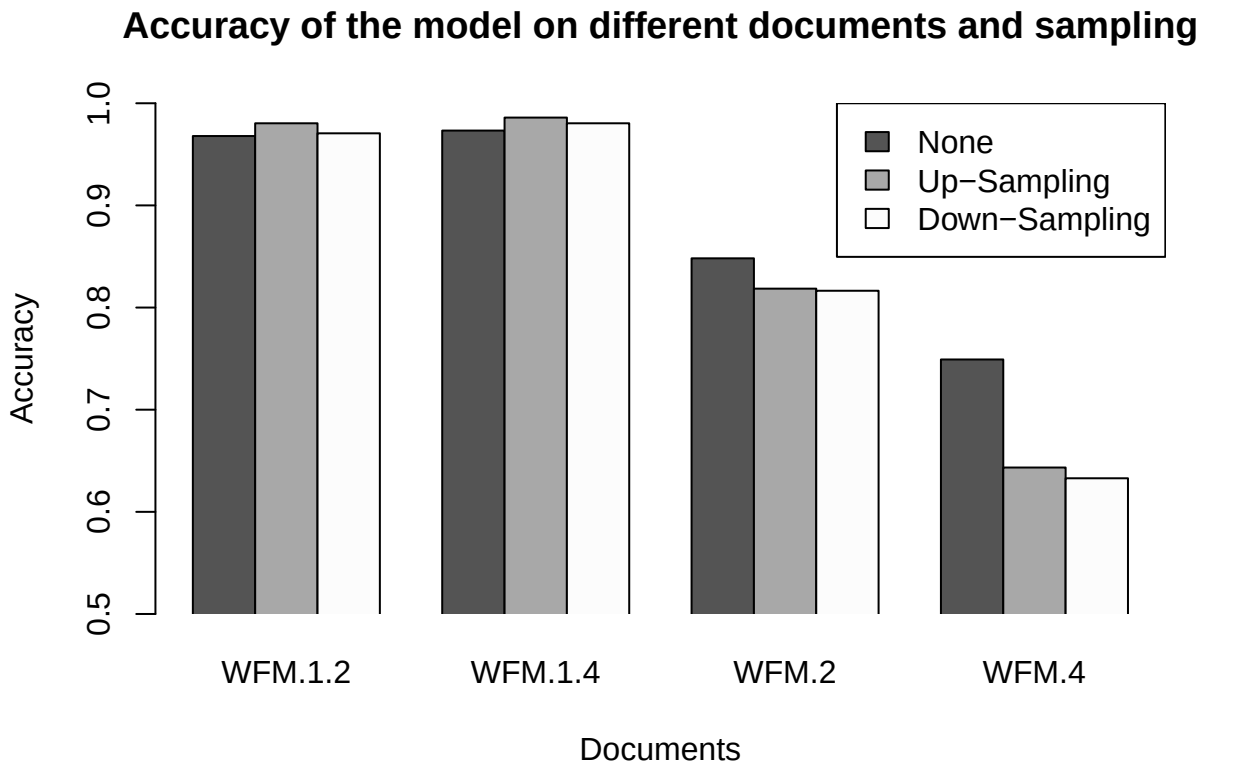
```
getModel <- function(doc, n){
     df <- getDF(doc)

     set.seed(2031)
     df <- upSample(df[,-length(df)], df$Clas, yname = 'Clas')
     df <- df[,c(1:n, length(df))]

     set.seed(1002)
     trainIndex <- createDataPartition(df$Clas, p = 0.70, list = FALSE)
     training <- df[trainIndex,]

     control <- trainControl(method = 'cv', number = 5)
     modelo <- train(Clas ~ ., data = training, method = 'rf',
                     trControl = control, ntree = 50,
                     tuneLength = length(train))
     return(modelo)
}

modelo <- getModel('WFM.1', 5)
```

Testing WFM.1 model in various documents

Using the previously created model, then a generalized testing, using the whole information of the other documents, is done. The results are displayed on a bar plot on which each set of bars is a document, and each bar represents a sampling technique that reduces class imbalance.

```
perfDocs <- function(modelo, doc, n, samp = 'none'){
     df <- getDF(doc)
     df <- df[,c(1:n, length(df))]

     if(samp == 'up'){
          set.seed(2031)
          df <- upSample(df[,-length(df)], df$Clas, yname = 'Clas')
     } else if(samp == 'down'){
          set.seed(2031)
          df <- downSample(df[,-length(df)], df$Clas, yname = 'Clas')
     }
     Pred <- predict(modelo, df)
     acc <- round(confusionMatrix(df$Clas, Pred)$overall[1], 5)
     return(acc)
}

n <- 5
modelo <- getModel('WFM.1', n)
dfSampling <- data.frame(doc = c('WFM.1.2', 'WFM.1.4', 'WFM.2', 'WFM.4'),
```

```
                          none = NA, up = NA, down = NA)
for(doc in dfSampling$doc){
     for(boot in colnames(dfSampling)[2:length(dfSampling)]){
          dfSampling[dfSampling$doc == doc, boot] <- perfDocs(modelo, doc,
                                                              n, samp = boot)
     }
}
row.names(dfSampling) <- dfSampling$doc
dfSampling$doc <- NULL

barplot(t(as.matrix.data.frame(dfSampling)), beside = TRUE, ylim = c(0.5,1),
        main = 'Accuracy of the model on different documents and sampling',
        col = c(gray(0.33), gray(0.66), gray(0.99)), xpd = FALSE,
        xlab = 'Documents', ylab = 'Accuracy')
legend('topright', legend = c('None', 'Up-Sampling', 'Down-Sampling'),
       fill = c(gray(0.33), gray(0.66), gray(0.99)))
```

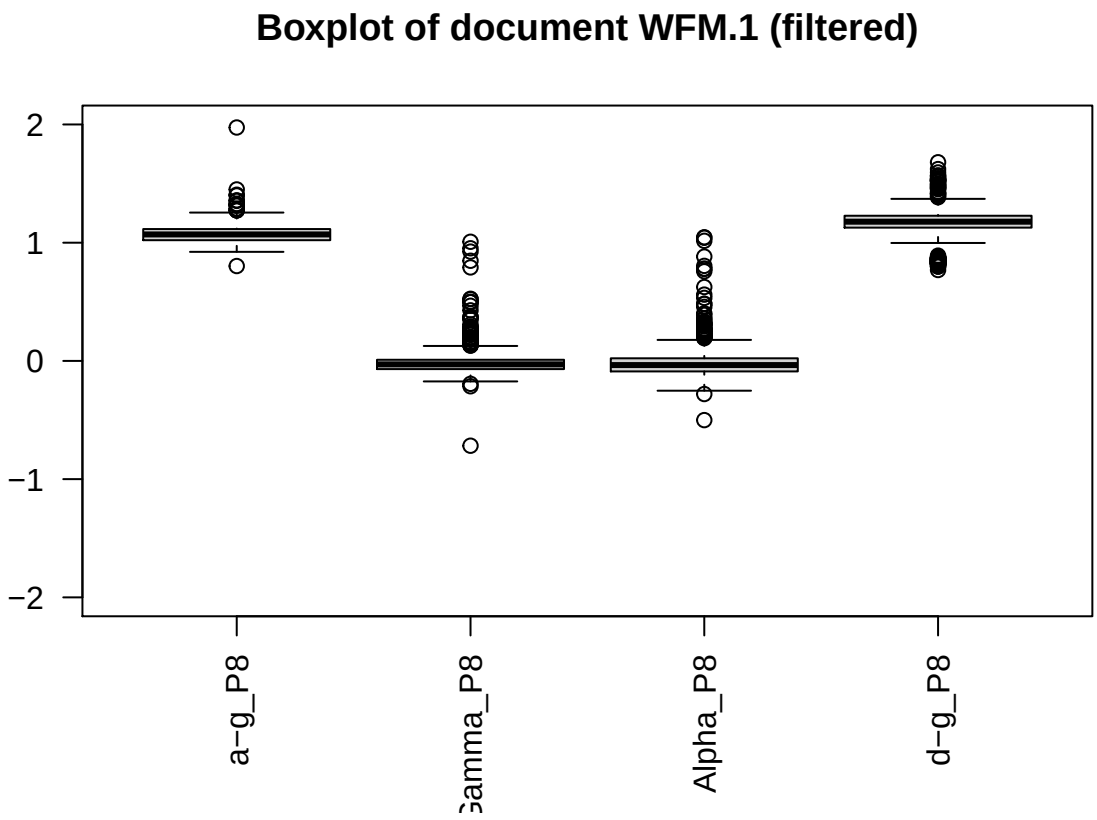


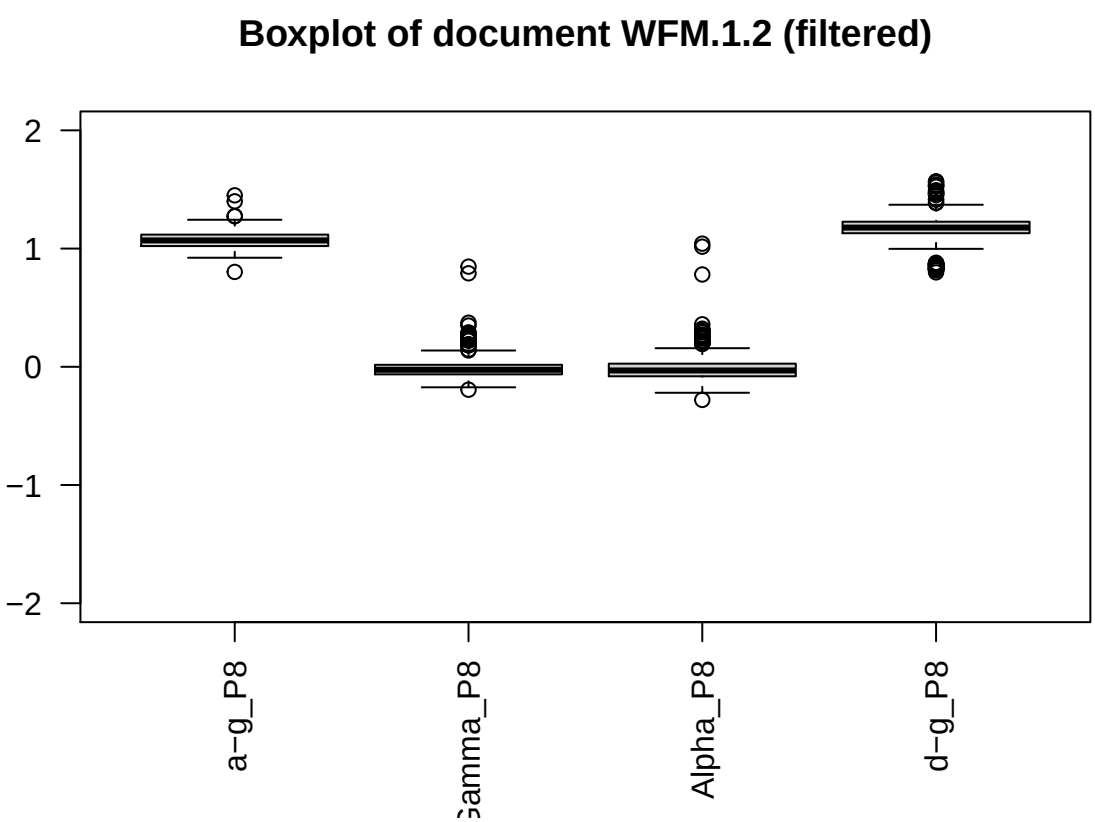
Based on the previous bar plot, it seems that data from the WFM.1 document is present on the other documents, as these is nearly perfect performance on WFM.1.2 and WFM.1.4 documents. Although there is some information that the tree-based model do not know, and thus has low performance on WFM.2 and WFM.4 documents, which is information from 2 and 4 minutes.

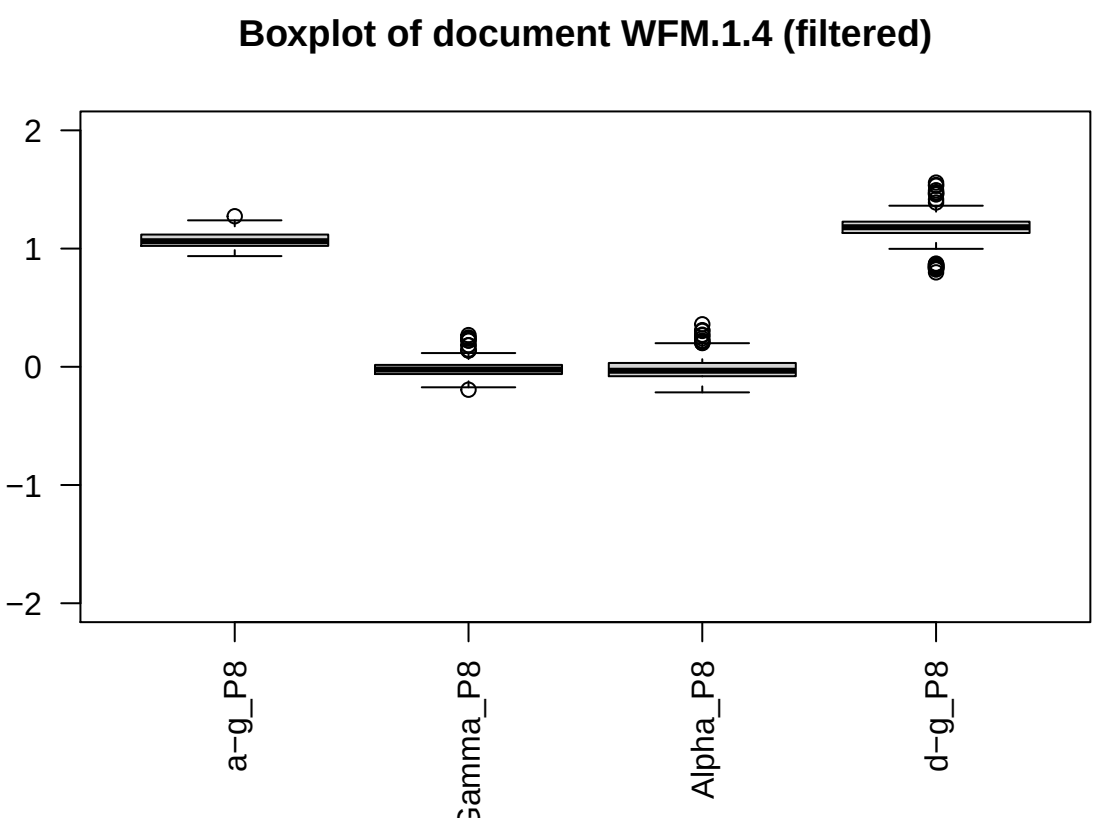
Box plot of documents

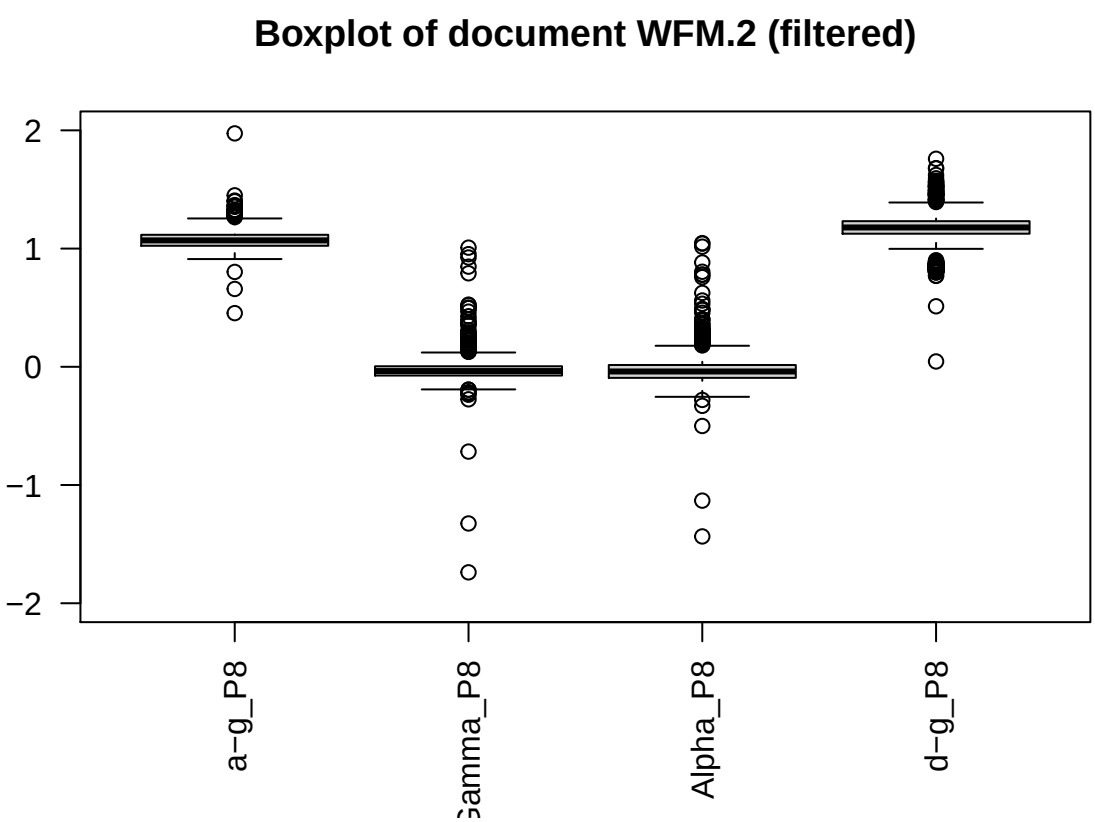
In order to visualize if there is a difference in the distribution of the used features, a set of box plots are being produced. The box plots contains filtered information with a threshold value of 10 and -10, and only five of the ten features are being displayed. Although, it seems that WFM.4 document has more distribution of values that the WFM.1, that might be a reason why the tree-based model performed poorly on those documents, because it had not the information that those values could have existed.

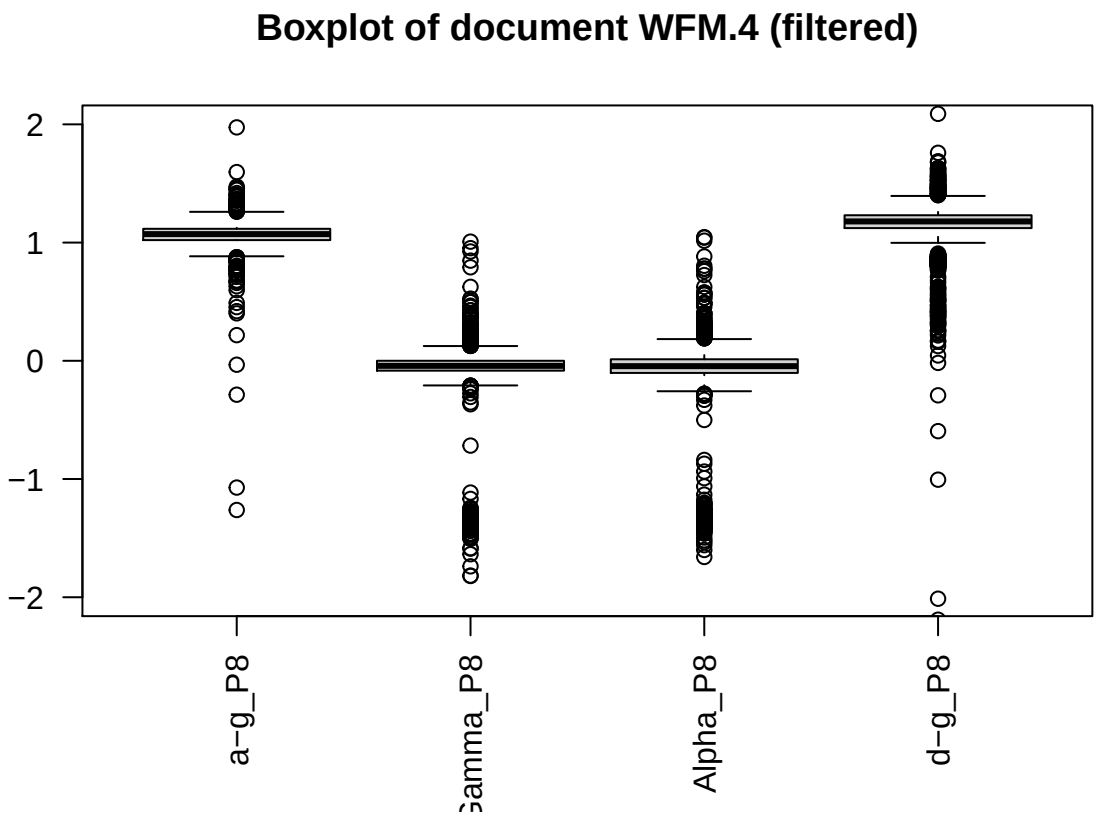
```
boxPlots <- function(doc, n){
     df <- getDF(doc)
     df <- df[,1:n]

     boxplot(df[,-length(df)], las = 2, ylim = c(-2,2),
     main = paste('Boxplot of document', doc, '(filtered)'))
}
docs <- c('WFM.1', 'WFM.1.2', 'WFM.1.4', 'WFM.2', 'WFM.4')
for(doc in docs){
     boxPlots(doc, 5)}
```




Boxplot of document WFM.1.2 (filtered)
2
1
0
−1
−2
a−g_P8
amma_P8
Alpha_P8
d−g_P8


Boxplot of document WFM.1.4 (filtered)
2
1
0
−1
−2
a−g_P8
amma_P8
Alpha_P8
d−g_P8


Boxplot of document WFM.2 (filtered)
2
1
0
−1
−2
a−g_P8
amma_P8
Alpha_P8
d−g_P8



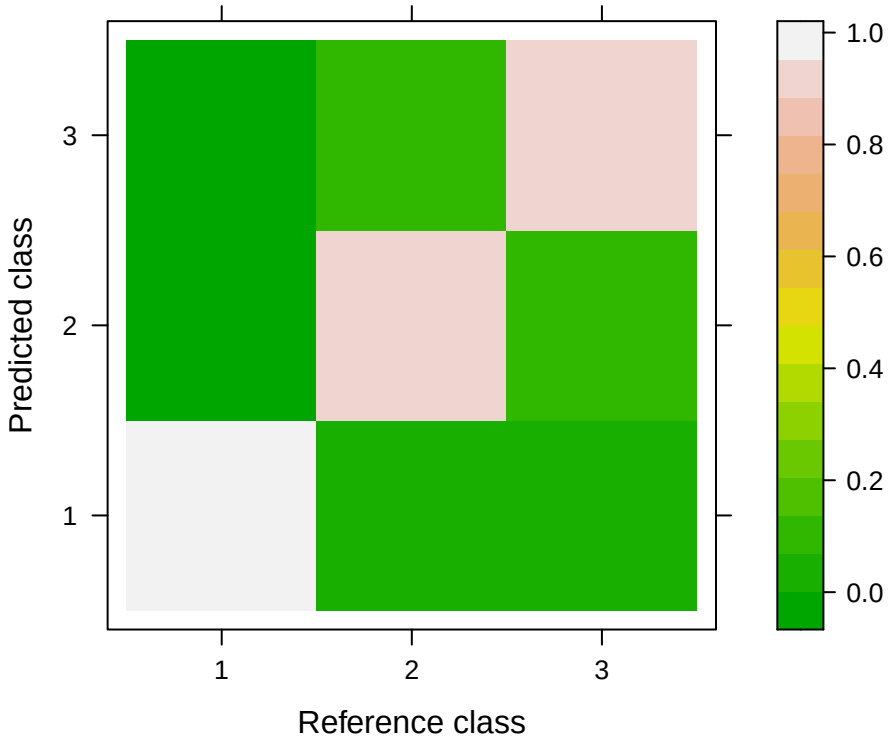
WFM.1 model's confusion matrix

The insights found on the milestone suggest that WFM.1 document might not be the best document to train the data on, because it has a low distribution of values. Thus, WFM.4 might be used on the following milestones, although, a final heat map that represents the model's confusion matrix would be displayed, in order to show the excellent performance of the rf model using five features, five cross-validations and a 70:30 split of data.

```
plotConfMat <- function(doc, n){
     df <- getDF(doc)

     set.seed(2031)
     df <- upSample(df[,-length(df)], df$Clas, yname = 'Clas')
     df <- df[,c(1:n, length(df))]

     set.seed(1002)
     trainIndex <- createDataPartition(df$Clas, p = 0.70, list = FALSE)
     training <- df[trainIndex,]
     testing <- df[-trainIndex,]
     control <- trainControl(method = 'cv', number = 5)

     modelo <- train(Clas ~ ., data = training, method = 'rf',
                     trControl = control, ntree = 50,
                     tuneLength = length(train))
     Pred <- predict(modelo, testing)
```

```
    print(confusionMatrix(testing$Clas, Pred))
    confm <- confusionMatrix(testing$Clas, Pred)$table

    sum_row <- apply(confm, FUN = sum, MARGIN = 2)
    conf_centage <- sweep(confm, 2, sum_row, FUN = '/')
    levelplot(conf_centage, col.regions = terrain.colors(100),
              xlab = 'Reference class', ylab = 'Predicted class',
              main = "Confusion matrix's final model (WFM.1, 5 features)")
}
plotConfMat('WFM.1', 5)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3
##          1 144   0   0
##          2   6 125  13
##          3   1  10 133
##
## Overall Statistics
##
##                Accuracy : 0.9306
##                  95% CI : (0.9023, 0.9527)
##     No Information Rate : 0.3495
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8958
##
##  Mcnemar's Test P-Value : 0.06042
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity            0.9536   0.9259   0.9110
## Specificity            1.0000   0.9360   0.9615
## Pos Pred Value         1.0000   0.8681   0.9236
## Neg Pred Value         0.9757   0.9653   0.9549
## Prevalence             0.3495   0.3125   0.3380
## Detection Rate         0.3333   0.2894   0.3079
## Detection Prevalence   0.3333   0.3333   0.3333
## Balanced Accuracy      0.9768   0.9310   0.9362
```


Confusion matrix's final model (WFM.1, 5 features)
Predicted class
Reference class
1
2
3
1.0
0.8
0.6
0.4
0.2
0.0