Representation

Milton Candela

8/7/2021

Packages used

The following packages were used:

```
library(R.matlab)
library(dplyr)
library(caret)
library(randomForest)
library(RColorBrewer)
library(lattice)
```

ROC curve

The first representation in order to validate that would be created, the multi-class Receiver Operating Characteristic (ROC) curve measures the accuracy of prediction on Area Under the Curve (AUC) as well as the sensitivity and specificity for each class.

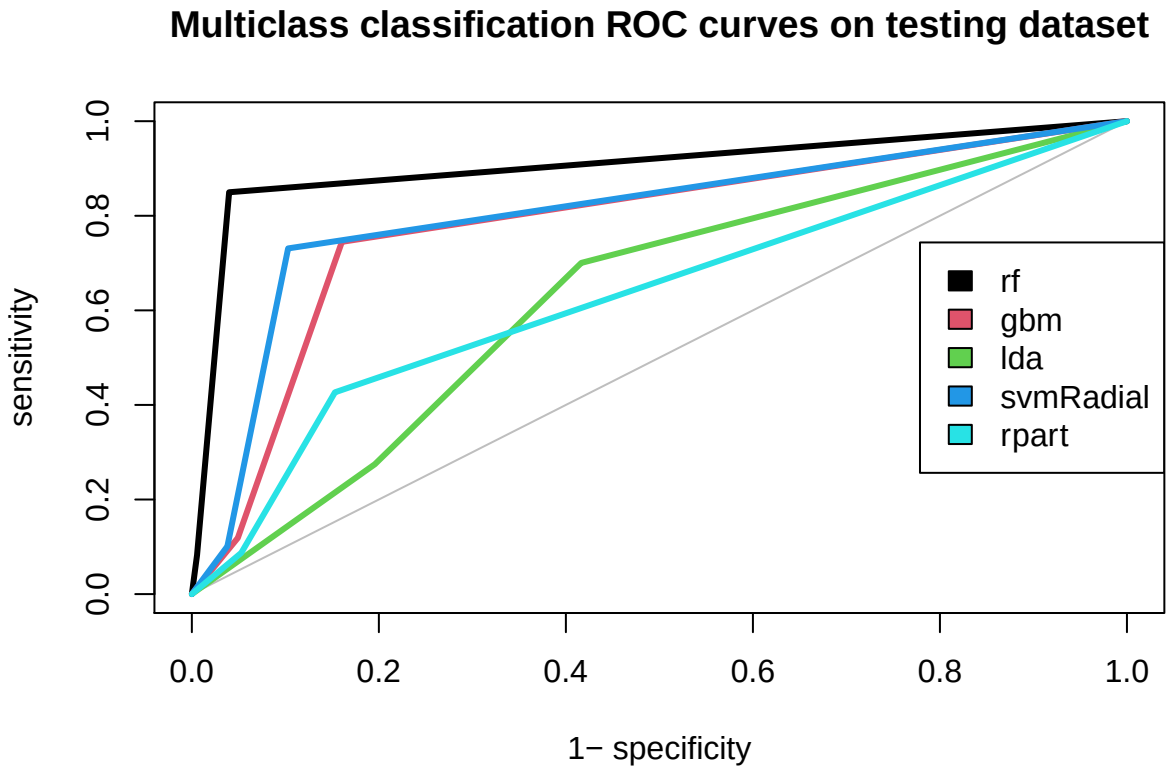
Five Machine Learning models would be used to contrast our final RF model's metrics:

- rf (Random Forest)
- gbm (Gradient Boosting Machine)
- lda (Linear Discriminant Analysis)
- svmRadial (Support Vector Machines with Radial Basis Function Kernel)
- rpart (Recursive Partitioning and Regression Trees)

The only model on which hyper-parameters would be tuned, is Random Forest (RF), as the best parameters were found on grid search performed on the previous milestone report.

```
models <- c('rf', 'gbm', 'lda', 'svmRadial', 'rpart')

df <- read.csv('Data/Processed/RF_MARS_Up.csv')
df <- mutate(df, Clas = as.factor(Clas))

set.seed(1002)
trainIndex <- createDataPartition(df$Clas, p = 0.8, list = FALSE)
training <- df[trainIndex,]
testing <- df[-trainIndex,]
control <- trainControl(method = 'cv', number = 5)

model1 <- randomForest(Clas ~ ., data = training, mtry = 9, ntree = 60,
```

```
                        maxnodes = 1300, nodesize = 8)
model2 <- train(Clas ~ ., data = training, method = 'gbm',
                trControl = control, verbose = FALSE)
model3 <- train(Clas ~ ., training, method = 'lda', trControl = control)
model4 <- train(Clas ~ ., training, method = 'svmRadial', trControl = control)
model5 <- train(Clas ~ ., training, method = 'rpart', trControl = control)

pred1 <- predict(model1, testing)
pred2 <- predict(model2, testing)
pred3 <- predict(model3, testing)
pred4 <- predict(model4, testing)
pred5 <- predict(model5, testing, type = 'raw')

lwd = 3
plot(AUC::roc(pred1, testing$Clas), col = 1, lwd = lwd,
     main = 'Multiclass classification ROC curves on testing dataset')
plot(AUC::roc(pred2, testing$Clas), col = 2, add = TRUE, lwd = lwd)
plot(AUC::roc(pred3, testing$Clas), col = 3, add = TRUE, lwd = lwd)
plot(AUC::roc(pred4, testing$Clas), col = 4, add = TRUE, lwd = lwd)
plot(AUC::roc(pred5, testing$Clas), col = 5, add = TRUE, lwd = lwd)
legend('right', legend = models, fill = 1:length(models))
```

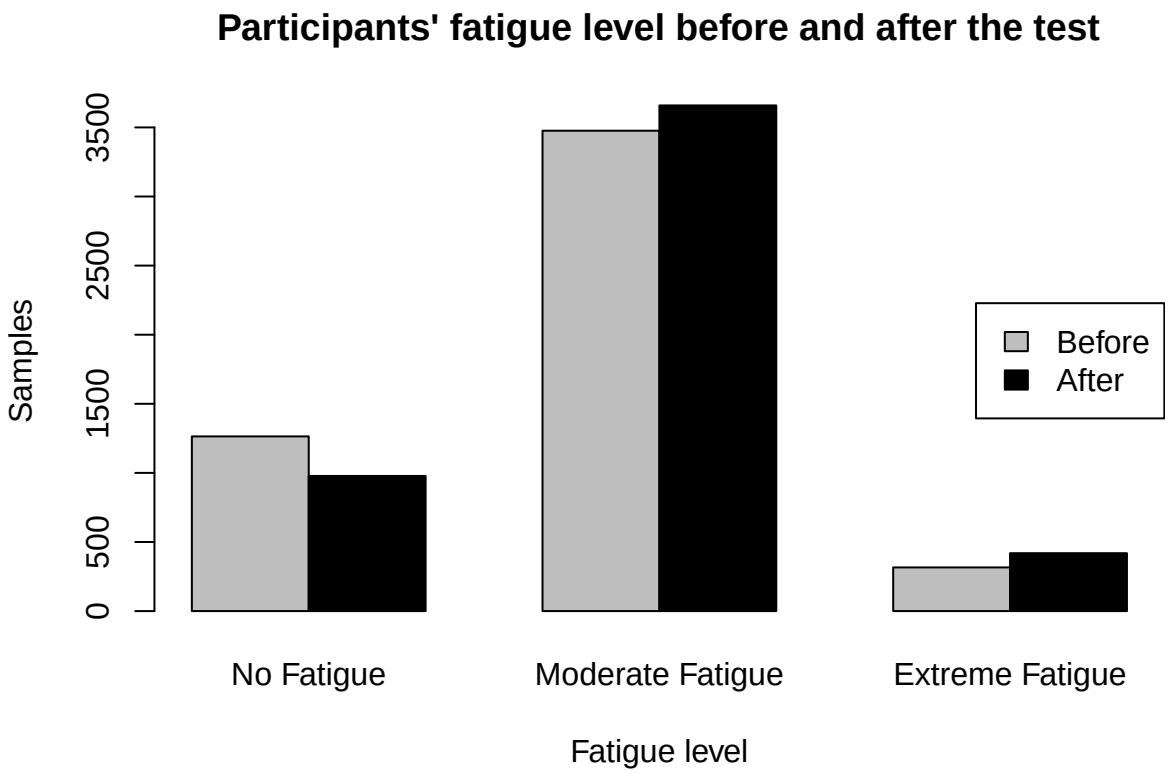


As it can be visualized, if we roughly calculate the AUC just by looking at the figure, Random Forest (rf) was superior, following Support Vector Machines with Radial Basis Function Kernel (svmRadial) and Gradient Boosting Machine (gbm), lastly, Recursive Partitioning and Regression Trees (rpart) and Linear Discriminant Analysis (lda).

Predicting post test values

Now, a bar plot on the post test recording would be made. Remember that the model was trained on biometrics recorded before and during the test, although a *.mat* file (*FeatureMatrices_post.mat*) is still remaining, which does not contain labels for each row. And so the current model would be used to predict a class on each row from the whole file, and thus the prevalence of classes would be compared with the known classes to assess whether it exists a difference on the overall fatigue.

```
getDF <- function(mat, doc, lim = 10){
     data <- R.matlab::readMat(mat)
     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[,-length(unmin)]
     df <- filter_all(df, all_vars(. < lim))
     df <- filter_all(df, all_vars(. > -lim))
     return(df)
}

postDF <- getDF('Data/Raw/FeatureMatrices_post.mat', 'WFM.4', lim = 10)

getClas <- function(mat, doc){
     data <- R.matlab::readMat(mat)
     unmin <- as.data.frame(data[doc])
     df <- unmin[,length(unmin)]
     return(df)
}

preClas <- getClas('Data/Raw/FeatureMatrices_pre.mat', 'WFM.4')
predClas <- predict(model1, postDF)

numbers <- data.frame(level1 = c(sum(preClas == 1), sum(predClas == 1)),
                      level2 = c(sum(preClas == 2), sum(predClas == 2)),
                      level3 = c(sum(preClas == 3), sum(predClas == 3)))

barplot(as.matrix(numbers), ylab = 'Samples', xlab = 'Fatigue level',
        names.arg = c('No Fatigue', 'Moderate Fatigue', 'Extreme Fatigue'),
        main = "Participants' fatigue level before and after the test",
        beside = TRUE, col = c('grey', 'black'))
legend('right', legend = c('Before', 'After'), fill = c('grey', 'black'))
```



It seems clear that there was a reduction on the number of samples that were categorized as "No Fatigue", which is then related to an increase of the number of samples that presented "Moderate Fatigue", while the third classification, "Extreme Fatigue" remained similar before and after the test.

Although this results refer to data obtained by a previous study, it gives a certain degree of validation to our model, as it was successfully capable of generalizing on a completely new dataset. This was due to the fact that the increase of overall fatigue is explained by students completing a task, and thus were expected to experience more mental fatigue as compared to their baseline.

Bar plot

Furthermore, to compare the accuracy of predictions using the testing dataset, a bar plot with error bars would be created. The error bars correspond to the accuracy of each model using 5-folds on the testing dataset. The plot is similar to the ROC's AUC, although it only takes into account the model's accuracy.

A set of 5 random seeds would be used to implement the cross-validation on the RF model, which was trained using the *randomForest* package instead of the *caret* package, where cross-validation is easily implemented.

```
df_acc <- data.frame(Acc = NA, Acc_Down = NA, Acc_Up = NA)
rf_cv <- data.frame(acc = NA, acc_Low = NA, acc_Up = NA)

for (seed in 1:5){
     set.seed(seed)
     trainIndex <- createDataPartition(df$Clas, p = 0.8, list = FALSE)
     training <- df[trainIndex,]
     testing <- df[-trainIndex,]
```

```
     model1 <- randomForest(Clas ~ ., data = training, mtry = 9, ntree = 60,
                            maxnodes = 1300, nodesize = 8)
     pred1 <- predict(model1, testing)
     rf_cv[seed,] <- confusionMatrix(testing$Clas, pred1)$overall[c(1, 3, 4)]
}

df_acc[1,] <- c(mean(rf_cv$acc), mean(rf_cv$acc_Low), mean(rf_cv$acc_Up))
df_acc[2,] <- confusionMatrix(testing$Clas, pred2)$overall[c(1, 3, 4)]
df_acc[3,] <- confusionMatrix(testing$Clas, pred3)$overall[c(1, 3, 4)]
df_acc[4,] <- confusionMatrix(testing$Clas, pred4)$overall[c(1, 3, 4)]
df_acc[5,] <- confusionMatrix(testing$Clas, pred5)$overall[c(1, 3, 4)]
rownames(df_acc) <- models
df_acc <- as.matrix(t(df_acc)) * 100

barras <- barplot(df_acc[1,], ylim = c(50,100), ylab = 'Accuracy %',
                  main = 'Accuracy of each model using 5-fold cross-validation',
                  xpd = FALSE, col = c('grey', 2:5))
error_bar <- function(x, y, upper, lower = upper, length = 0.1,...){
     arrows(x, y+upper, x, y - lower, angle = 90, code = 3, length = length, ...)
}
text(barras, df_acc[1,] + 5, round(df_acc[1,]), cex = 1)
error_bar(barras, df_acc[1,], upper = df_acc[3,] - df_acc[1,], lower = df_acc[1,] - df_acc[2,])
```