#Libraries

library(tidyverse)

library(dplyr)

library(randomForest)

library(tree)

library(MASS)

library(psych)

library(naivebayes)

library(ggplot2)

library(gbm)

require(lattice)

library(caret)

library(knitr)

library(rmarkdown)

library(XLConnect)

library(readxl)

library(e1071)

#Importing Training file

# #importing training file.

# Train <- read.csv(file="/Users/karunakarsastry/Downloads/bbdc\_2019\_Bewegungsdaten/train.csv", header=TRUE, sep=",")

#

# Train <- data.frame(Train)

# Temp<-data.frame(Train[,-2])

# Lables <- Temp[-1]

# uniquelable<-data.frame(unique(Lables))

# uniquelables<-count(unique(Lables))

# paste("no.of unique lables are:",uniquelables)

# #Adding new variable Lablecode to Train dataframe.

# Train$Labelcode<-NA

# #Populating Labelcode

# Train$Labelcode <- ifelse(Train$Label == "curve-left-step",1,ifelse(Train$Label == "stand-to-sit",2,ifelse(Train$Label == "curve-right-spin-Rfirst",3,ifelse(Train$Label == "jump-one-leg",4,ifelse(Train$Label == "lateral-shuffle-right",5,ifelse(Train$Label == "curve-right-spin-Lfirst",6,ifelse(Train$Label == "v-cut-right-Lfirst",7,ifelse(Train$Label == "stair-down",8,ifelse(Train$Label == "v-cut-left-Rfirst",9,ifelse(Train$Label == "v-cut-right-Rfirst",10,ifelse(Train$Label == "jump-two-leg",11, ifelse(Train$Label == "sit",12,ifelse(Train$Label == "stair-up",13,ifelse(Train$Label == "curve-right-step",14,ifelse(Train$Label == "sit-to-stand",15,ifelse(Train$Label == "run",16, ifelse(Train$Label == "v-cut-left-Lfirst",17,ifelse(Train$Label == "stand",18,ifelse(Train$Label == "curve-left-spin-Lfirst",19,ifelse(Train$Label == "walk",20,ifelse(Train$Label == "curve-left-spin-Rfirst",21,ifelse(Train$Label == "lateral-shuffle-left",22,23))))))))))))))))))))))

#

# #Coverting shortened lable to factor type.

#

# Train$Labelcode <- as.factor(Train$Labelcode )

# str(Train)

#importing test data file

# #importing test data file

# Test <- read.csv(file="/Users/karunakarsastry/Downloads/bbdc\_2019\_Bewegungsdaten/challenge.csv", header=TRUE, sep=",")

#

# Test <- data.frame(Test)

#

# ```

# #combining training file and subject folder for training data. Normalising the data

# ```{r}

# combinevec <- c(colsd,colmin,colmax,colmean)

# #columnew<- c("Labelcode","EMG1","EMG2","EMG3","EMG4","Airborne","ACC\_upper\_X","ACC\_upper\_Y","ACC\_upper\_Z" ,"Goniometer\_X","ACC\_lower\_X","ACC\_lower\_Y","ACC\_lower\_Z","Goniometer\_Y","Gyro\_upper\_X","Gyro\_upper\_Y","Gyro\_upper\_Z","Gyro\_lower\_X","Gyro\_lower\_Y","Gyro\_lower\_Z")

# #colum<- c("EMG1","EMG2","EMG3","EMG4","Airborne","ACC\_upper\_X","ACC\_upper\_Y","ACC\_upper\_Z" ,"Goniometer\_X","ACC\_lower\_X","ACC\_lower\_Y","ACC\_lower\_Z","Goniometer\_Y","Gyro\_upper\_X","Gyro\_upper\_Y","Gyro\_upper\_Z","Gyro\_lower\_X","Gyro\_lower\_Y","Gyro\_lower\_Z")

#

# columnew <- c("Labelcode",combinevec)

# colum <- combinevec

#

# forcbind<-rbind(S02,S03,S04,S05,S06,S07,S08,S09,S011,S012,S013,S016,S017,S018,S019)

# #forcbind<-normalize(forcbind)

#

# forcbindtest<-rbind(S01,S010,S014,S015)

# #forcbindtest <- normalize((forcbindtest))

#

# #TrainTestscale <- scale(rbind(forcbind,forcbindtest))

# #forcbind<-head(TrainTestscale,6401)

# #forcbindtest <- tail(TrainTestscale,1738)

#combining training file and subject folder for training data.

# Trainsample <- Train

# #Trainsample <- head(Train,1318)

#

# #forcbind <- scale(rbind(S02,S03,S04,S05,S06,S07,S08,S09,S011,S012,S013,S016,S017,S018,S019))

# #forcbind

#

# library(XLConnect)

# library("readxl")

#

# Trainsample<-data.frame(cbind(Trainsample,forcbind))

#

# tirf <- Sys.time()

#

# Trainsam <- write.csv(Trainsample,file = paste('/Users/karunakarsastry/Desktop/Bremenchallengeresults/Trainingdatabbdc.csv',"\_",tirf),quote=FALSE)

#

Trainsampl1 <- read\_csv("/Users/karunakarsastry/Desktop/Bremenchallengeresults/Trainingdatabbdc.csv \_ 2019-03-30 20:47:15")

Trainsampl1$Labelcode <- ifelse(Trainsampl1$Label == "curve-left-step",1,ifelse(Trainsampl1$Label == "stand-to-sit",2,ifelse(Trainsampl1$Label == "curve-right-spin-Rfirst",3,ifelse(Trainsampl1$Label == "jump-one-leg",4,ifelse(Trainsampl1$Label == "lateral-shuffle-right",5,ifelse(Trainsampl1$Label == "curve-right-spin-Lfirst",6,ifelse(Trainsampl1$Label == "v-cut-right-Lfirst",7,ifelse(Trainsampl1$Label == "stair-down",8,ifelse(Trainsampl1$Label == "v-cut-left-Rfirst",9,ifelse(Trainsampl1$Label == "v-cut-right-Rfirst",10,ifelse(Trainsampl1$Label == "jump-two-leg",11, ifelse(Trainsampl1$Label == "sit",12,ifelse(Trainsampl1$Label == "stair-up",13,ifelse(Trainsampl1$Label == "curve-right-step",14,ifelse(Trainsampl1$Label == "sit-to-stand",15,ifelse(Trainsampl1$Label == "run",16, ifelse(Trainsampl1$Label == "v-cut-left-Lfirst",17,ifelse(Trainsampl1$Label == "stand",18,ifelse(Trainsampl1$Label == "curve-left-spin-Lfirst",19,ifelse(Trainsampl1$Label == "walk",20,ifelse(Trainsampl1$Label == "curve-left-spin-Rfirst",21,ifelse(Trainsampl1$Label == "lateral-shuffle-left",22,23))))))))))))))))))))))

Trainsampl1$Labelcode <- as.factor(Trainsampl1$Labelcode )

#

TrainFinal <- data.frame(Trainsampl1[5:80])

TrainFinal$Labelcode <- as.factor(TrainFinal$Labelcode )

# #names(TrainFinal) <- columnew

#

# #TrainFinal

#

# str(TrainFinal)

#

# TrainTestscale <- TrainFinal[,-1]

# descr\_cor=cor(TrainTestscale)

# descr\_cor

#

# #setting cutoff value of correlation coeff=0.70

# ax=findCorrelation(descr\_cor, cutoff = 0.70)

#

# #displaying highly correlated variables

# highly\_cor\_var=colnames(TrainTestscale[ax])

# highly\_cor\_var

#

# #removing the highly correlated variables

# train1=TrainFinal[,-ax]

# str(train1)

#

# require(lattice)

# #train2 <- log(train1)

# Labelcode <- TrainFinal$Labelcode

# TrainFinal <- cbind(Labelcode,train1)

#

#

# #combining test file and subject folder for test data.

# Testsample <- Test

# #Testsample <- head(Test,440)

# #forcbindtest<-scale(rbind(S01,S010,S014,S015))

#

#

#

# Testsample<-data.frame(cbind(Testsample,forcbindtest))

#

# Testsam <- write.csv(Testsample,file = paste('/Users/karunakarsastry/Desktop/Bremenchallengeresults/Testingdatabbdc.csv',"\_",tirf),quote=FALSE)

# All above process was done to compress the sensor data into 76 predictors matrix as per this below script.

Testsampl1 <- read\_csv("/Users/karunakarsastry/Desktop/Bremenchallengeresults/Testingdatabbdc.csv \_ 2019-03-30 20:47:15")

FinalTest<-data.frame(Testsampl1[5:79])

#Data Visualisation

# Bar plot of Train Classes.

barplot(prop.table(table(TrainFinal$Labelcode)),

col = rainbow(20),

ylim = c(0, 0.10),

main = "Class Distribution")

table(TrainFinal$Labelcode)

prop.table(table(TrainFinal$Labelcode))

#pairs plot of label means and standard devations to check co-relation.

#pairs.panels(TrainFinal[-1])

#Box plot of label mean.

TrainFinal %>% ggplot(aes(x=Labelcode,y=EMG1sd,fill=Labelcode))+

geom\_boxplot()+ggtitle("Box Plot")

#Box plot of label standard deviation.

TrainFinal %>% ggplot(aes(x=Labelcode,y=EMG1min,fill=Labelcode))+

geom\_boxplot()+ggtitle("Box Plot")

#Density Plot(pdf) of a particular sensor.

TrainFinal %>% ggplot(aes(x=EMG1mean,fill=Labelcode))+geom\_density(alpha=0.8,color='black')+

ggtitle("Density Plot")

#Density Plot(pdf) of a particular sensor.

TrainFinal %>% ggplot(aes(x=Gyro\_upper\_Zsd ,fill=Labelcode))+geom\_density(alpha=0.8,color='black')+

ggtitle("Density Plot")

pairs.panels(TrainFinal[2:4])

pairs.panels(TrainFinal[5:6])

plot(TrainFinal[6:9])

#plotting sensor data points vs Frequency ( at 1000 Hz )

#mean of sub01

S01000 <- read.csv(file="/Users/karunakarsastry/Downloads/bbdc\_2019\_Bewegungsdaten/Subject01/Subject01\_Aufnahme000.csv", header=FALSE, sep=",")

names(S01000) <- colum

S0100<-apply(S01000, 2, sd, na.rm = TRUE)

ncol(S01000)

par("mar")

par(mar=c(1.5,2,1.5,2))

i=1

j=1

par(mfrow=c(5,4))

cl <- rainbow(ncol(S01000))

while (i <= ncol(S01000) && j <= ncol(S01000))

{

S01000TEMP <- S01000[i]

FREQS01000EMG1 = seq(1,nrow(S01000),1)

NS01000TEMP = cbind(FREQS01000EMG1,S01000TEMP)

plot(NS01000TEMP ,type = "l",main=colum[j],pch= 1,col=cl[j], xlab = "Freq", ylab = colum[j])

#legend("bottomright", colum[j], pch=0.03)

i=i+1

j = j+1

}

#Data Partition

set.seed(1234)

ind <-sample(2,nrow(TrainFinal),replace =T,prob =c(0.80,0.20))

train <-TrainFinal[ind == 1,]

test <- TrainFinal[ind == 2,]

#Random Forests

rndftrain <- randomForest(train$Labelcode~., data=train, ntree = 300,mtry = 4, importance = TRUE,proximity = TRUE,do.trace=100)

#Tuning Random forests

# change model after above run. It appears that OOB error is min at mtry = 2. Trees = 300.

s = train[,-1]

w = train[,1]

t<- tuneRF(s,w, stepFactor = 0.5,plot = TRUE,ntreeTry = 300 ,trace = TRUE,improve = 0.01)

print(t)

rndftrain <- randomForest(train$Labelcode~., data=train, ntree = 300,mtry = 4, importance = TRUE,proximity = TRUE,do.trace=100)

rndftrain

plot(rndftrain)

legend("topright", colnames(rndftrain$err.rate),col=1:4,cex=0.8,fill=1:4,horiz=T)

predicttrain <- predict(rndftrain, train)

confmatrixtrain<-confusionMatrix(predicttrain,train$Labelcode)

confmatrixtrain

rndftabletrain<-table(predicttrain,train$Labelcode)

rndmisclassification <- (1-sum(diag(rndftabletrain)/sum(rndftabletrain)))\*100

rndmisclassification

rndtrainaccuracy <- 100-rndmisclassification

#Labels

labelscorrect <- sum(diag(rndftabletrain))

labelscorrect

labelsoverall <- sum(rndftabletrain)

labelsoverall

Accuracylabels <- labelscorrect/labelsoverall\*100

Accuracylabels

#Random Forests Training Accuracy

rndtrainaccuracy

originaltr<-train$Labelcode

prdtionstr<-t(t(predicttrain))

resultstr<-data.frame(cbind(prdtionstr,originaltr))

names(resultstr) <- c("prdtionstr","originalstr")

str(resultstr)

resultstr$desc <- NA

resultstr$desc <- ifelse(resultstr$prdtionstr == 1,"curve-left-step",ifelse(resultstr$prdtionstr == 2,"stand-to-sit",ifelse(resultstr$prdtionstr == "curve-right-spin-Rfirst",3,ifelse(resultstr$prdtionstr == 4,"jump-one-leg",ifelse(resultstr$prdtionstr == 5,"lateral-shuffle-right",ifelse(resultstr$prdtionstr == 6,"curve-right-spin-Lfirst",ifelse(resultstr$prdtionstr == 7,"v-cut-right-Lfirst",ifelse(resultstr$prdtionstr == 8,"stair-down",ifelse(resultstr$prdtionstr == 9,"v-cut-left-Rfirst",ifelse(resultstr$prdtionstr == 10,"v-cut-right-Rfirst",ifelse(resultstr$prdtionstr == 11,"jump-two-leg", ifelse(resultstr$prdtionstr == 12,"sit",ifelse(resultstr$prdtionstr == 13,"stair-up",ifelse(resultstr$prdtionstr == 14,"curve-right-step",ifelse(resultstr$prdtionstr == 15,"sit-to-stand",ifelse(resultstr$prdtionstr == 16,"run", ifelse(resultstr$prdtionstr == 17,"v-cut-left-Lfirst",ifelse(resultstr$prdtionstr == 18,"stand",ifelse(resultstr$prdtionstr == 19,"curve-left-spin-Lfirst",ifelse(resultstr$prdtionstr == 20,"walk",ifelse(resultstr$prdtionstr == 21,"curve-left-spin-Rfirst",ifelse(resultstr$prdtionstr == 22,"lateral-shuffle-left","lay"))))))))))))))))))))))

#Trainig Results

resultstr

#Testing data

predicttest <- predict(rndftrain,test)

confmatrixtst<-confusionMatrix(predicttest,test$Labelcode)

confmatrixtst

originalts<-test$Labelcode

prdtionsts<-t(t(predicttest))

resultsts<-data.frame(cbind(prdtionsts,originalts))

names(resultsts) <- c("prdtionsts","originalsts")

str(resultsts)

resultsts$desc <- NA

resultsts$desc <- ifelse(resultsts$prdtionsts == 1,"curve-left-step",ifelse(resultsts$prdtionsts == 2,"stand-to-sit",ifelse(resultsts$prdtionsts == "curve-right-spin-Rfirst",3,ifelse(resultsts$prdtionsts == 4,"jump-one-leg",ifelse(resultsts$prdtionsts == 5,"lateral-shuffle-right",ifelse(resultsts$prdtionsts == 6,"curve-right-spin-Lfirst",ifelse(resultsts$prdtionsts == 7,"v-cut-right-Lfirst",ifelse(resultsts$prdtionsts == 8,"stair-down",ifelse(resultsts$prdtionsts == 9,"v-cut-left-Rfirst",ifelse(resultsts$prdtionsts == 10,"v-cut-right-Rfirst",ifelse(resultsts$prdtionsts == 11,"jump-two-leg", ifelse(resultsts$prdtionsts == 12,"sit",ifelse(resultsts$prdtionsts == 13,"stair-up",ifelse(resultsts$prdtionsts == 14,"curve-right-step",ifelse(resultsts$prdtionsts == 15,"sit-to-stand",ifelse(resultsts$prdtionsts == 16,"run", ifelse(resultsts$prdtionsts == 17,"v-cut-left-Lfirst",ifelse(resultsts$prdtionsts == 18,"stand",ifelse(resultsts$prdtionsts == 19,"curve-left-spin-Lfirst",ifelse(resultsts$prdtionsts == 20,"walk",ifelse(resultsts$prdtionsts == 21,"curve-left-spin-Rfirst",ifelse(resultsts$prdtionsts == 22,"lateral-shuffle-left","lay"))))))))))))))))))))))

resultsts

originalts<-test$Labelcode

prdtionsts<-t(t(predicttest))

resultsts1<-data.frame(cbind(prdtionsts,originalts))

names(resultsts1) <- c("prdtionsts","originalsts")

resultsts1

#Final Testing Folder

predicttestfinal <- predict(rndftrain,FinalTest)

predicttestfinal

Testfinaltablerf<-data.frame(predicttestfinal)

Testfinaltablerf$Label <-NA

names(Testfinaltablerf)<-c("Finalpredtst","Label")

Testfinaltablerf$Label <- ifelse(Testfinaltablerf$Finalpredtst == 1,"curve-left-step",ifelse(Testfinaltablerf$Finalpredtst == 2,"stand-to-sit",ifelse(Testfinaltablerf$Finalpredtst == "curve-right-spin-Rfirst",3,ifelse(Testfinaltablerf$Finalpredtst == 4,"jump-one-leg",ifelse(Testfinaltablerf$Finalpredtst == 5,"lateral-shuffle-right",ifelse(Testfinaltablerf$Finalpredtst == 6,"curve-right-spin-Lfirst",ifelse(Testfinaltablerf$Finalpredtst == 7,"v-cut-right-Lfirst",ifelse(Testfinaltablerf$Finalpredtst == 8,"stair-down",ifelse(Testfinaltablerf$Finalpredtst == 9,"v-cut-left-Rfirst",ifelse(Testfinaltablerf$Finalpredtst == 10,"v-cut-right-Rfirst",ifelse(Testfinaltablerf$Finalpredtst == 11,"jump-two-leg", ifelse(Testfinaltablerf$Finalpredtst == 12,"sit",ifelse(Testfinaltablerf$Finalpredtst == 13,"stair-up",ifelse(Testfinaltablerf$Finalpredtst == 14,"curve-right-step",ifelse(Testfinaltablerf$Finalpredtst == 15,"sit-to-stand",ifelse(Testfinaltablerf$Finalpredtst == 16,"run", ifelse(Testfinaltablerf$Finalpredtst == 17,"v-cut-left-Lfirst",ifelse(Testfinaltablerf$Finalpredtst == 18,"stand",ifelse(Testfinaltablerf$Finalpredtst == 19,"curve-left-spin-Lfirst",ifelse(Testfinaltablerf$Finalpredtst == 20,"walk",ifelse(Testfinaltablerf$Finalpredtst == 21,"curve-left-spin-Rfirst",ifelse(Testfinaltablerf$Finalpredtst == 22,"lateral-shuffle-left","lay"))))))))))))))))))))))

Testfinaltablerf

dtrf <- Sys.Date()

tirf <- Sys.time()

#Exporting results to Csv and formating

#Define the file name that will be deleted

#fn <- "finaltestresults.csv"

#Check its existence

#if (file.exists(fn))

#Delete file if it exists

# file.remove(fn)

challenge<-data.frame(Testsampl1[2:3])

Testfinalrf <- data.frame(Testfinaltablerf)

finalfileresults<-cbind(challenge,Testfinalrf[-1])

finallables <- write.csv(finalfileresults,file = paste('/Users/karunakarsastry/Desktop/Bremenchallengeresults/finaltestresultsrf.csv',"\_",tirf),quote=FALSE)

#importing file and convering into CSV

library("readxl")

Finalresulformating <- read.csv(file="/Users/karunakarsastry/Desktop/Bremenchallengeresults/finaltestresults22mar2019.csv", header=TRUE, sep=",")

Finalresulformating

Finalchallenge <- write.csv(Finalresulformating,file = '/Users/karunakarsastry/Desktop/Bremenchallengeresults/456.csv',quote=FALSE)

Finalchallenge

#Exporting to Excel

library(XLConnect)

library("readxl")

finalreslts<-writeWorksheetToFile("finalres.xlsx",

data = Testfinaltablerf,

sheet = "final",

header = TRUE,

clearSheets = TRUE)

#variable importance

varimp<-varImpPlot(rndftrain,sort=T,n.var=18,main="Variable Importance", col="blue")

importance(rndftrain)

varUsed(rndftrain)

#partialPlot(rndftrain,train,Gyro\_lower\_Z,"2")

#partialPlot(rndftrain,train,ACC\_upper\_Z,"2")

getTree(rndftrain,1,labelVar=TRUE)

#Multidimensional Scaling plot

MDSplot(rndftrain,train$Labelcode)

#Tree plot

library(tree)

tree.tr <- tree(train$Labelcode~.,data = train[-1])

summary(tree.tr)

plot(tree.tr)