MACHINE LEARNING PREDICTION

Tamara Rueda

23 de julio de 2018

```
library(knitr)
options(width=87)
knitr::opts_chunk$set(
    echo = TRUE,
    fig.height = 9,
    fig.width = 9,
    message = TRUE,
    warning = FALSE,
    background = "#ffffff",
    collapse = FALSE,
    comment = "#"
)
```

The following data analysis is focused on to predict how in which 6 participants: Adelmo, carlitos, Eurico, Jeremy, Pedro, and Charles, self measured on how they performed certain excercise. There are 160 variables on the training set and 19622 observations.“Classe” is the one related to the excercise. Different machine learning algorithms are applied to the 20 test cases available in the test data.

About the data

The exercise data and personal activity is measured Using devices such as Jawbone Up, Nike FuelBand, and Fitbit. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, we are asked to use data from accelerometers on the belt, forearm, arm, and dumbell of the participants. All 6 participants performed barbell lifts correctly and incorrectly in 5 different ways.

On decision tree model we look at class A, as the dominant class on how well the participant perform on the exercises across the different measurement devices ##READING AND IMPORTING DATA Both training and testing sets are imported and created as objects at R global environment.

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
library(readr)
training <- read.csv("pml-training.csv", na.strings = c("NA", ""))
testing  <- read.csv("pml-testing.csv", na.strings = c("NA", ""))
dim(training)
```

```
# [1] 19622   160
```

```
dim(testing)
```

```
# [1]  20 160
```

```
#look at variables class
#sapply(training, class)
```

DATA CLEANING

(1)The data is already separated into training and testing sets. (2)First Column X can be excluded from the model. (3)Second column, Names, may remain in the model as it holds information about the subjects participants. (4) Columns 3-4 are raw time in seconds elapsed. I see no reason to remove them from the model. They hold information about the duration of excercise activities (5) Training data as well as Testing had problems with NAs and zero values. (6)The near zero variance variables (NZV). This is important, you dont need to consider in your model explanatory variables wich are not really explaining the manner in which the participants did the exercise. We will do this after removing the NAs

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
###(2)Remove first column
Training <- training[, -(1:1)]
Testing  <- testing[, -(1:1)]
dim(Training)
```

```
# [1] 19622   159
```

```
dim(Testing)
```

```
# [1]  20 159
```

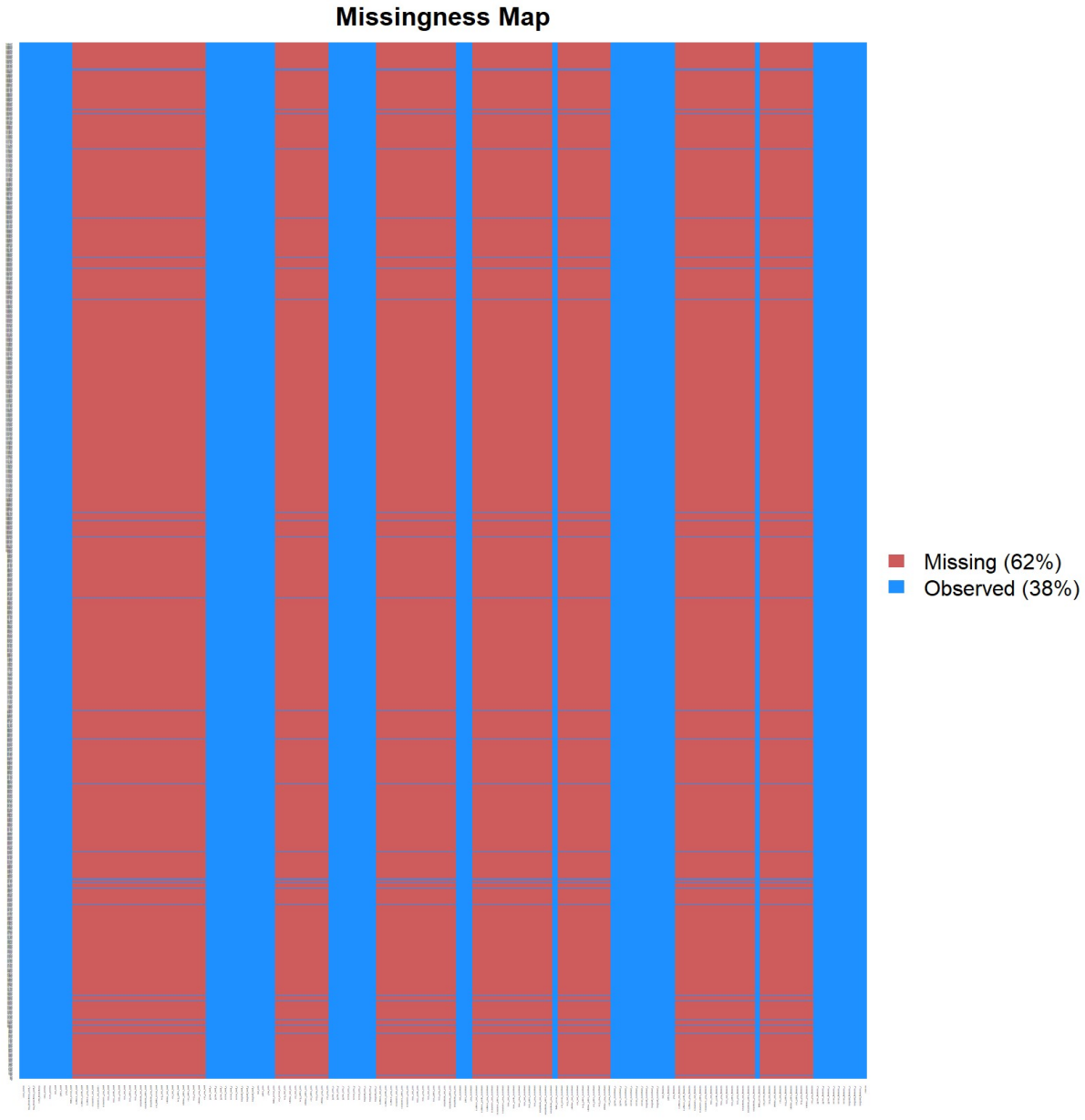
Removing NAS by MULTIPLE IMPUTATION with Amelia R function Check the Training Dataset missing values

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
library(Amelia)#handling NAs from data
```

```
# Loading required package: Rcpp
```

```
# ##
# ## Amelia II: Multiple Imputation
# ## (Version 1.7.5, built: 2018-05-07)
# ## Copyright (C) 2005-2018 James Honaker, Gary King and Matthew Blackwell
# ## Refer to http://gking.harvard.edu/amelia/ for more information
# ##
```

```
missmap(Training, legend=TRUE,col = c("indianred", "dodgerblue"), rank.order=FA
LSE, margins = c(3,1), y.cex = 0.1, x.cex = 0.1)
```



Amelia can be called using the Ameliaview function too which is faster. Please be patient with the map takes time to show. missing values map shows we have a PATTERN of missing values in the dataset. Over a 45% of the values are missing. The pattern is presented by variable or column. Since NAs are not random, is not possible run amelia or mice package for multiple imputation. Also because of collinearity and non normally distributed variables. Its best then to work only with complete cases column vectors.

```
###Now remove NAS by column
Traindata <-Training[,complete.cases(t(Training))];
Testdata <-Testing[,complete.cases(t(Testing))]; dim(Traindata);
```

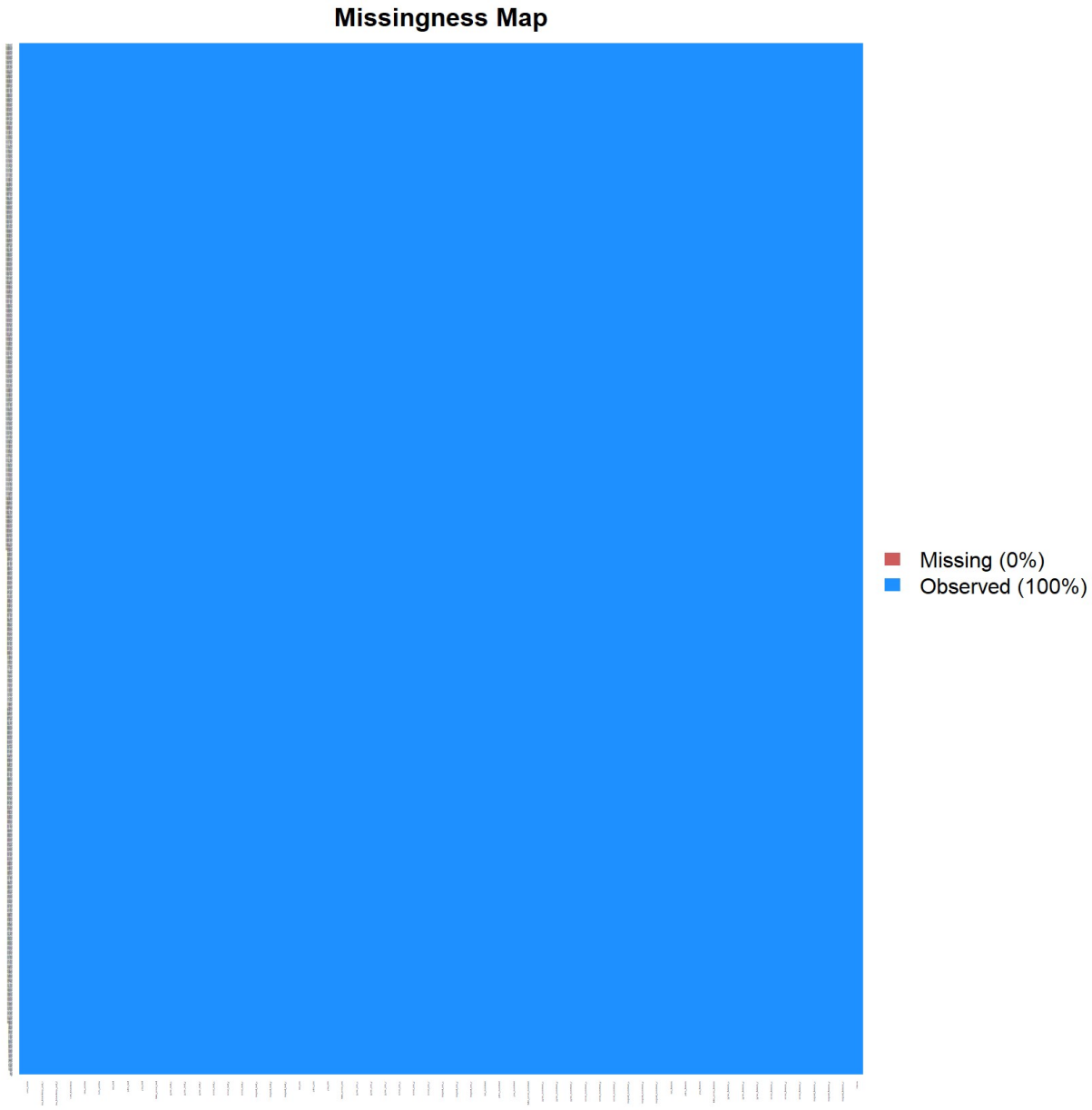
```
# [1] 19622    59
```

```
dim(Testdata)
```

```
# [1] 20 59
```

Check again for missing values: No more NAs

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
missmap(Traindata, legend=TRUE,col = c("indianred", "dodgerblue"), rank.order=F
ALSE, margins = c(3,1), y.cex = 0.1, x.cex = 0.1)
```



After removal of NAs, we have left 59 variables. Both Traindata and Testdata are now the same length.

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
library(caret)
```

```
# Loading required package: lattice
```

```
# Loading required package: ggplot2
```

```
NZV <- nearZeroVar(Traindata)
Traindata1 <- Traindata[, -NZV]
Testdata1  <- Testdata[, -NZV]
dim(Traindata1)
```

```
# [1] 19622    58
```

```
dim(Testdata1)
```

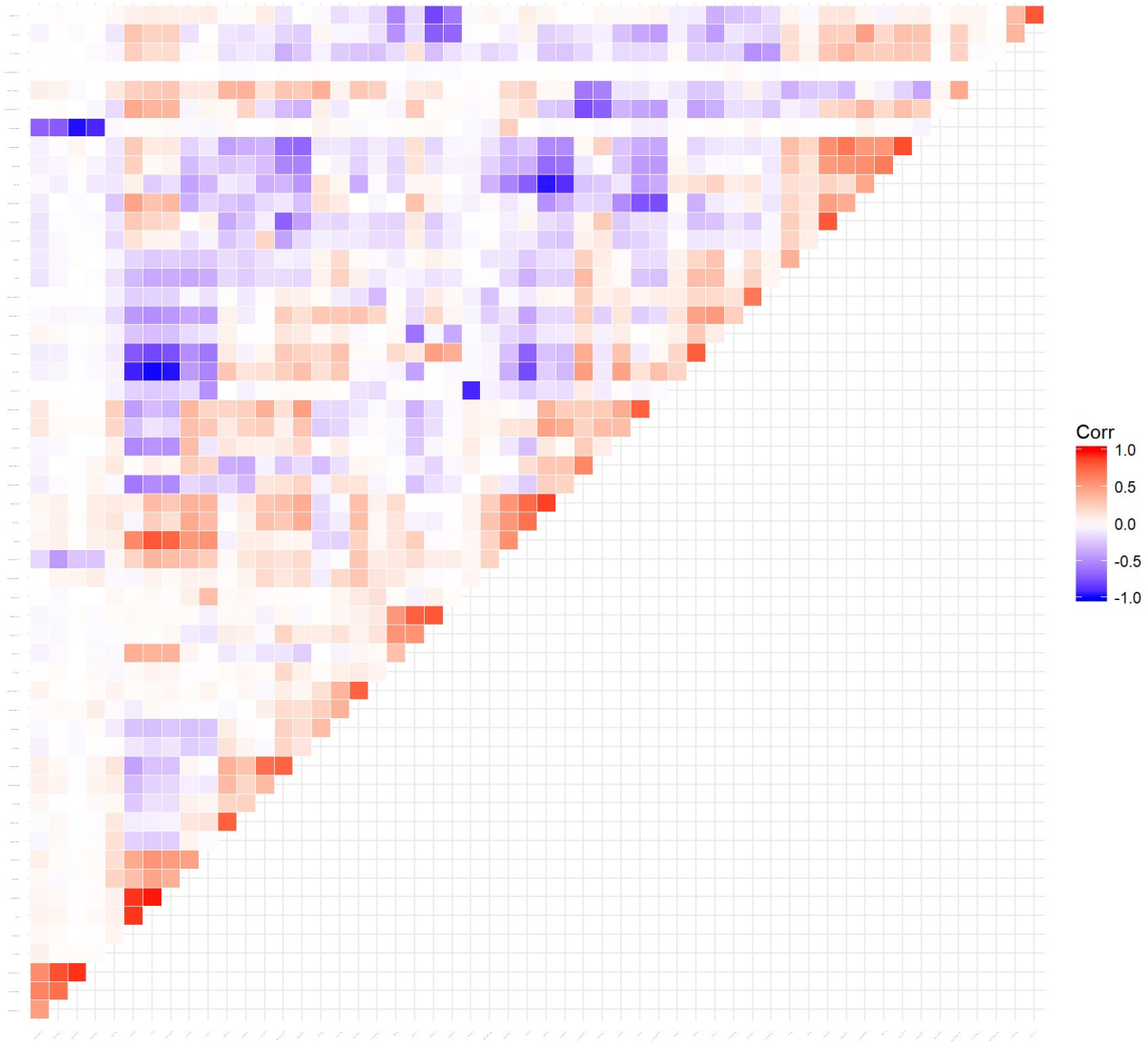
```
# [1] 20 58
```

1 additional variable was removed on Traindata and Testdata because the predictor have very few unique values relative to number of samples.

DATA ANALYSIS AND EXPLORATION

Correlation

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
library(ggcorrplot)
Traindata_cor <- Traindata1[,-c(1,4,58)]
#correlation matrix
cormatrix <-cor(Traindata_cor)
ggcorrplot(cormatrix, method = "square",type = "upper",tl.cex = 0.5, insig = "b
lank",outline.color = "white",hc.order = TRUE)
```



I explored further the correlations between variables. For instance I did a principal component analysis both on observations and variables. The result wasnt that surprising: on observations we had 1 component and 2 for variables. We have only 6 participants in the study, observations are very clustered. Anyhow p NIR] : < 2.2e-16

#

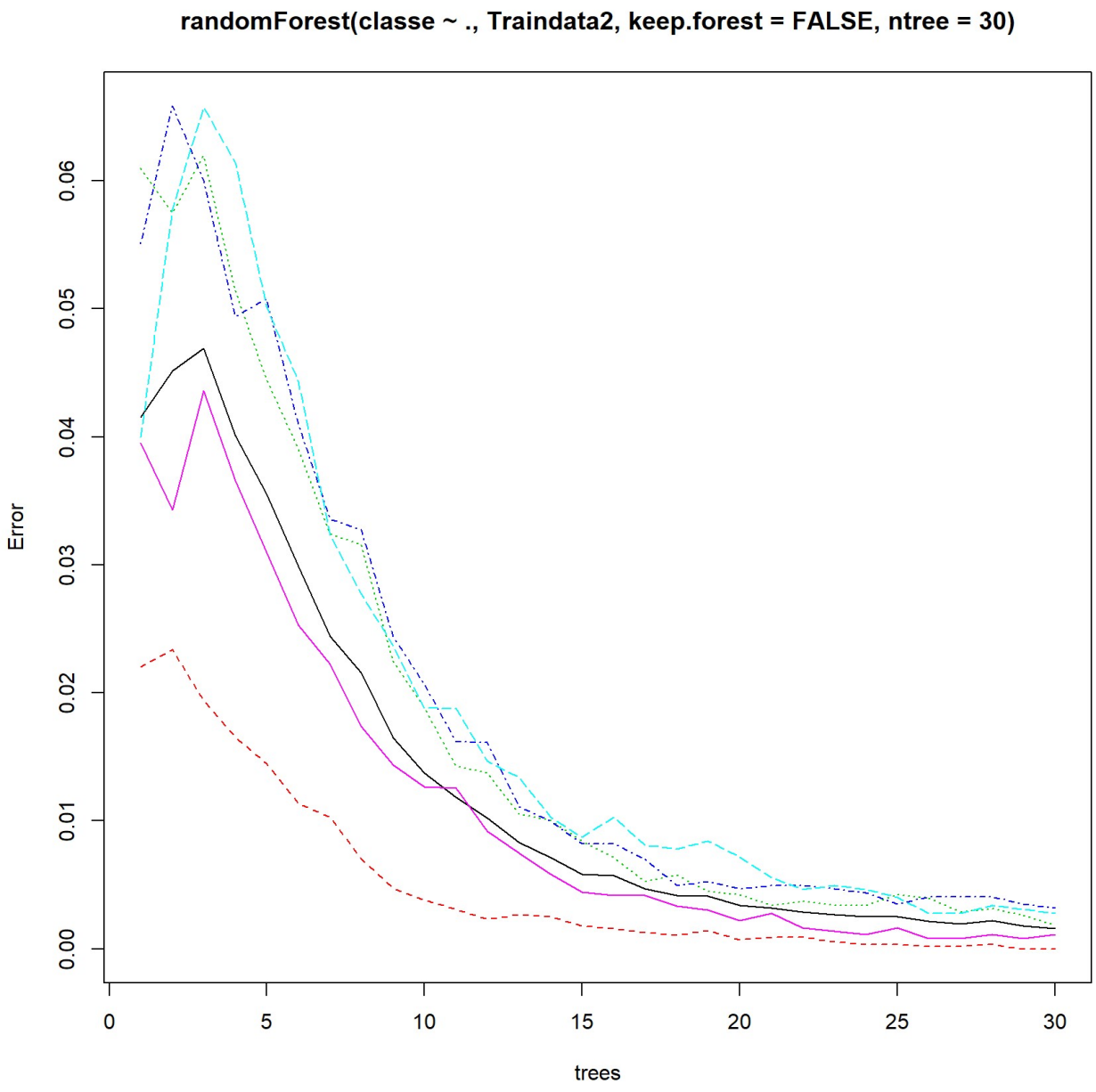
Kappa : 1

Mcnemar's Test P-Value : NA

Statistics by Class: # # Class: A Class: B Class: C Class: D Class: E # Sensitivity 1.0000 1.0000 1.0000 1.0000 1.0000 # Specificity 1.0000 1.0000 1.0000 1.0000 1.0000 # Pos Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000 # Neg Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000 # Prevalence 0.2844 0.1935 0.1744 0.1639 0.1838 # Detection Rate 0.2844 0.1935 0.1744 0.1639 0.1838 # Detection Prevalence 0.2844 0.1935 0.1744 0.1639 0.1838 # Balanced Accuracy 1.0000 1.0000 1.0000 1.0000 1.0000 ```

PLOT THE ERROR RATES OF A RANDOMFOREST

```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
plot(randomForest(classe ~ ., Traindata2, keep.forest=FALSE, ntree=30))
```



FIT A DECISION TREE MODEL

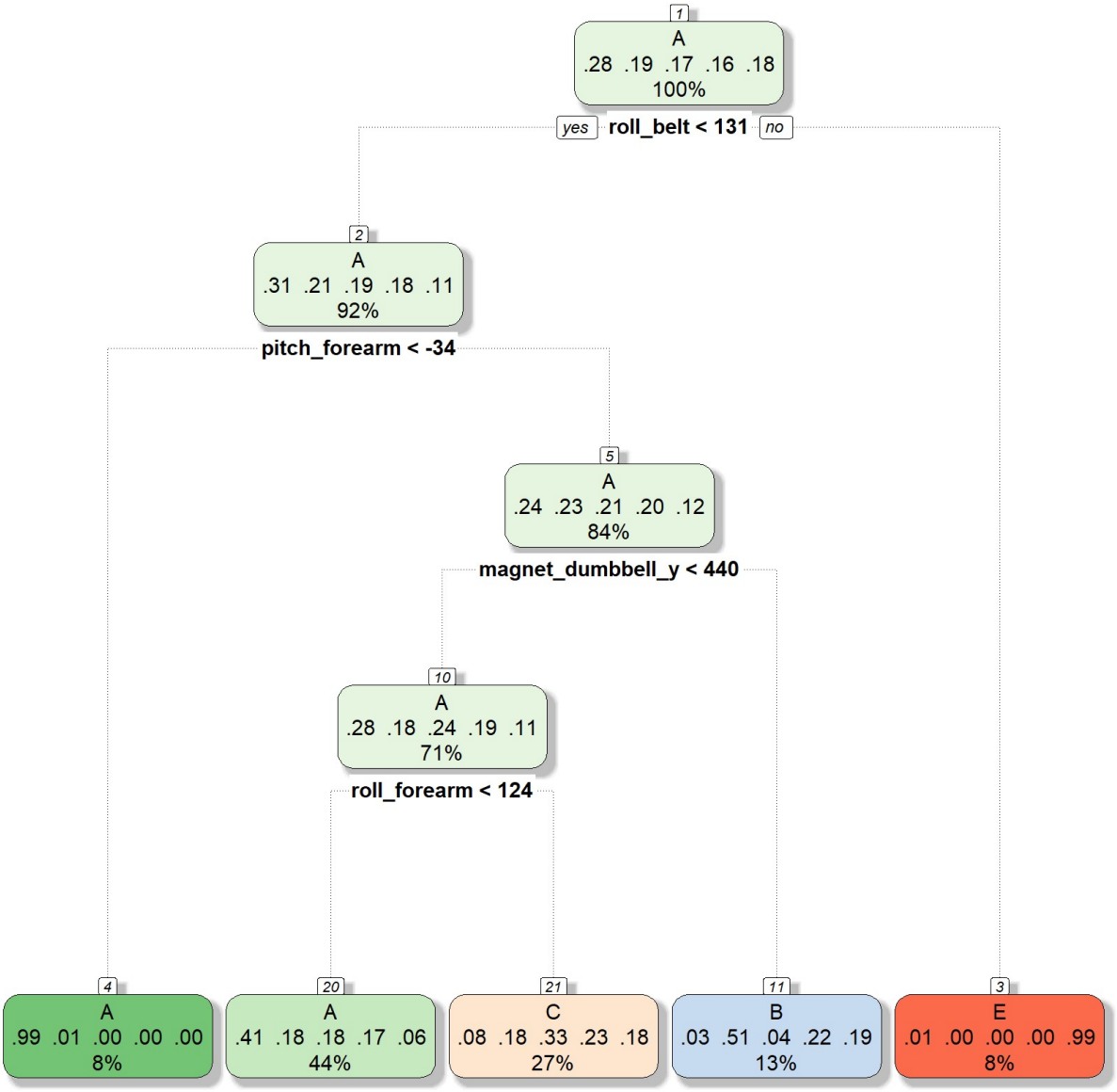
```
knitr::opts_chunk$set(message = TRUE, echo = TRUE)
library(rpart)
set.seed(3545)
modeldt <- rpart(classe~., data = Traindata2, method = "class")
#Prune the big tree object
library(rattle)
```

```
# Rattle: A free graphical interface for data science with R.
# Versión 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
# Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

```
#
# Attaching package: 'rattle'
```

```
# The following object is masked from 'package:randomForest':
#
#     importance
```

```
Prunedt <- prune(modeldt, cp = 0.045)
fancyRpartPlot(Prunedt) #now plot smaller tree
```



Rattle 2018-jul.-24 17:51:24 TRUEDA

At the top of the tree plot,we follow all measurements from the participants,¿How well they did it?. Notice class A is dominant class across all the different devices such as roll belt, patch forearm, etc.

FIT Linear discriminant analysis

```
mod_lda<-train(classe ~., data=Traindata2, method="lda")
```

FIT A BOOSTED PREDICTOR WITH THE “gbm” model.

This will take too long to compute, is the standard function we used. #modelgbm<-train(classe~., data=Traindata2, method="gbm") #pred_gbm<-predict(modelgbm,Traindata2) thats the reason I used h2o package instead.

```
#library(h2o)
#h2o.init()
#H2o.init()#initiate conection with H2o
#Create the H2o frame
#Traindata2.hex <- as.h2o(Traindata2)
##fit the algorithm now
#gbm1 <- h2o.gbm(training_frame = Traindata2.hex,
#x=1:56,y=57, model_id = "gbm",seed = 3545)
#gbm1
```

Now predict using gbm, this connect with a cloud and rmd cannot retrieve info from there. This is why im leaving the code deactivated in green. But the code is fast and works properly

```
#predict with gbm
#gbm_predict <-h2o.predict(gbm1, Traindata2.hex)
#Confusion Matrix
#gbm_confmatrix <-h2o.confusionMatrix(gbm1)
#gbm_confmatrix
```

DATA VALIDATION PREDICTION ON TEST DATASET

```
predict_Test_rf <- predict(modelrf, newdata=Testdata2)
predict_Test_dt <- predict(modeldt, newdata=Testdata2)
#create h2o object
#Testdata2.hex <- as.h2o(Testdata2)
#Test_gbm_predict <-h2o.predict(gbm1, Testdata2.hex, type="response")
#predict with linear discriminant analysis
pred_lda<-predict(mod_lda,Testdata2)
```

ACCURACY fot TESTdata

```
library(caret)
#Random forest
Testdata2$classe <- predict_Test_rf
c <-confusionMatrix(predict_Test_rf, Testdata2$classe)
c$overall
```

```
#       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
#   1.000000e+00   1.000000e+00   8.315665e-01   1.000000e+00   4.000000e-01
# AccuracyPValue  McnemarPValue
#   1.099512e-08            NaN
```

```
#linear discrimination analysis
Testdata2$classe <- pred_lda
c <-confusionMatrix(pred_lda, Testdata2$classe)
c$overall
```

```
#       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
#   1.000000e+00   1.000000e+00   8.315665e-01   1.000000e+00   4.500000e-01
# AccuracyPValue  McnemarPValue
#   1.159445e-07            NaN
```

Overall, both lda and random forest provide the response prediction with similar performance.