Practical Machine Learning - Course Project

VICTOR CAMPOS

2022-09-07

#Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. 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, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

#Data According to the information, the URL to download this information is as follows: The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

#Loading the libraries

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.1.3

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.1.3

## Loading required package: ggplot2

## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'pillar'

## Loading required package: lattice
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.1.3
```

```
## corrplot 0.92 loaded
```

#ownloading the data

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "C:/Users/vicct/Do
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "C:/Users/vicct/Dow
```

##Cleaning the data The data information comes with (N/A)s & empty spaces, so the following commands will clean the data.

```
trainingcsv = read.csv("C:/Users/vicct/Downloads/pml-training.csv", na.strings=c("NA","#DIV/0!",""))
testcsv = read.csv("C:/Users/vicct/Downloads/pml-training.csv", na.strings=c("NA","#DIV/0!",""))
trainingcsv <- trainingcsv[, colSums(is.na(trainingcsv)) == 0]
testcsv <- testcsv[, colSums(is.na(testcsv)) == 0]
```

##Removing columns with the X, timestamp & windows strigns. Those columns are unnecessary for the calculations. Using grepl to look for the unwised columns from te training and test data sets.

```
classe <- trainingcsv$classe

trainRemove <- grepl("^X|timestamp|window", names(trainingcsv))
trainingcsv <- trainingcsv[, !trainRemove]
trainingcsv_cleaned <- trainingcsv[, sapply(trainingcsv, is.numeric)]
trainingcsv_cleaned$classe <- classe
testRemove <- grepl("^X|timestamp|window", names(testcsv))
testcsv <- testcsv[, !testRemove]
testcsv_cleaned <- testcsv[, sapply(testcsv, is.numeric)]
```

#Partitioning the Data ##Using Caret to create a 70/30% split of the cleaned training data.

```
training_partition <- createDataPartition(y = trainingcsv_cleaned$classe, p = 0.7, list = F)
trainingdata_partition <- trainingcsv_cleaned[training_partition,] #training data frame
testingdata_partition <- trainingcsv_cleaned[-training_partition,] #test data frame
```

#Examining the proportions of the data

```
prop.table(table(trainingdata_partition$classe))
```

```
##
##         A         B         C         D         E
## 0.2843416 0.1934920 0.1744195 0.1639368 0.1838101
```

```
prop.table(table(testingdata_partition$classe))
```

```
##
##         A         B         C         D         E
## 0.2844520 0.1935429 0.1743415 0.1638063 0.1838573
```

##Portions are equivalent for the training data and the testing data.

##Building the Model ## Set up caret to perform 5-fold cross validation. Ramdom Forest is the method we will use to predict the activity recognition. This method will select the variables and correlate covariables.

```
train.control <- trainControl(method = "cv", 5)
model.rf <- train(classe ~ ., data = trainingdata_partition, method = "rf", trControl = train.control)
##Examining model.rf
model.rf
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10989, 10990, 10990, 10989, 10990
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9926477  0.9906991
##   27    0.9914830  0.9892252
##   52    0.9842765  0.9801103
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

##Validating the performance using the testing data set.

```r
predict_classe <- predict(model.rf, testingdata_partition)
#Checking the levels
table(testingdata_partition$classe) ##Predicted
```

```
##
##    A    B    C    D    E
## 1674 1139 1026  964 1082
```

```r
table(predict_classe) ##Actual
```

```
## predict_classe
##    A    B    C    D    E
## 1676 1137 1043  946 1083
```

```r
confusionMatrix(as.factor(testingdata_partition$classe), predict_classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    2    0    0    0
##          B    4 1131    4    0    0
##          C    0    4 1022    0    0
##          D    0    0   17  946    1
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.9946
##                  95% CI : (0.9923, 0.9963)
##     No Information Rate : 0.2848
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9931
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9947   0.9799   1.0000   0.9991
## Specificity            0.9995   0.9983   0.9992   0.9964   1.0000
## Pos Pred Value         0.9988   0.9930   0.9961   0.9813   1.0000
## Neg Pred Value         0.9991   0.9987   0.9957   1.0000   0.9998
## Prevalence             0.2848   0.1932   0.1772   0.1607   0.1840
## Detection Rate         0.2841   0.1922   0.1737   0.1607   0.1839
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9986   0.9965   0.9895   0.9982   0.9995
```

###Reviewing thw sensitivity and specifity of the information, then checking the Accuracy and the out of sample error of the results.

```
accuracy <- postResample(predict_classe, as.factor(testingdata_partition$classe))
accuracy
```

```
##  Accuracy     Kappa
## 0.9945624 0.9931217
```

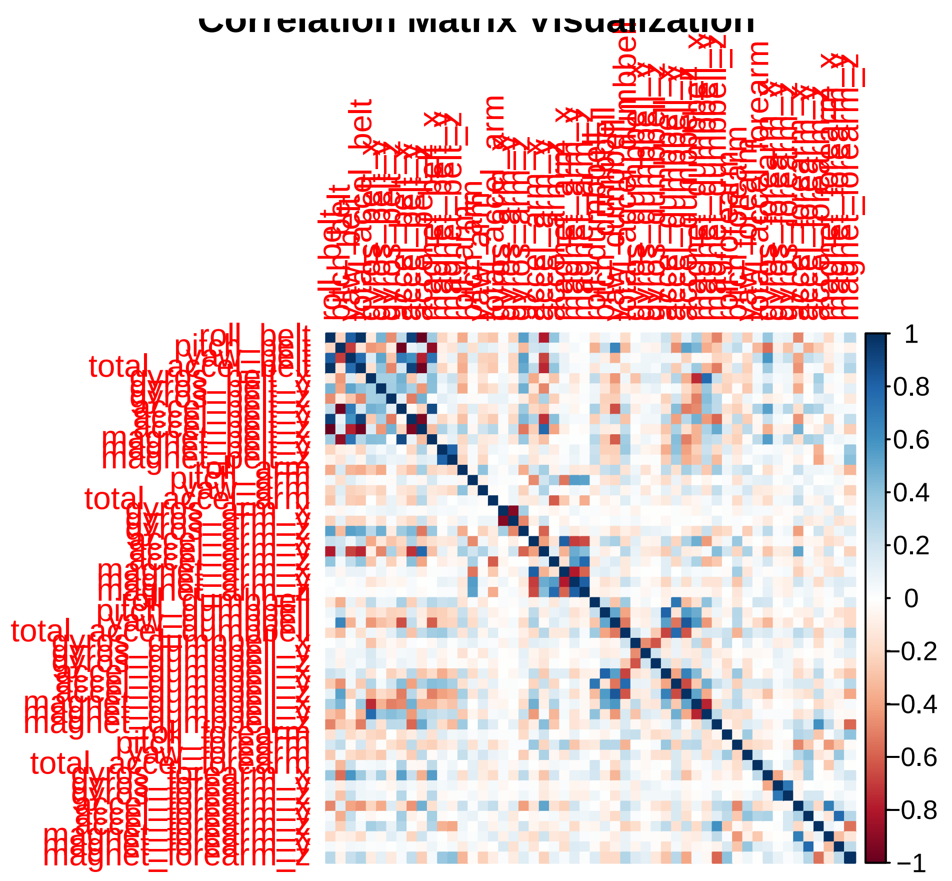
```
outofsampleerror <- 1 - as.numeric(confusionMatrix(as.factor(testingdata_partition$classe), predict_cla
#outofsampleerror
outofsampleerror
```

```
## [1] 0.005437553
```

the out of sample error result is 0.005 and the accuracy of 0.9949~, this indicate a good result for the prediction.

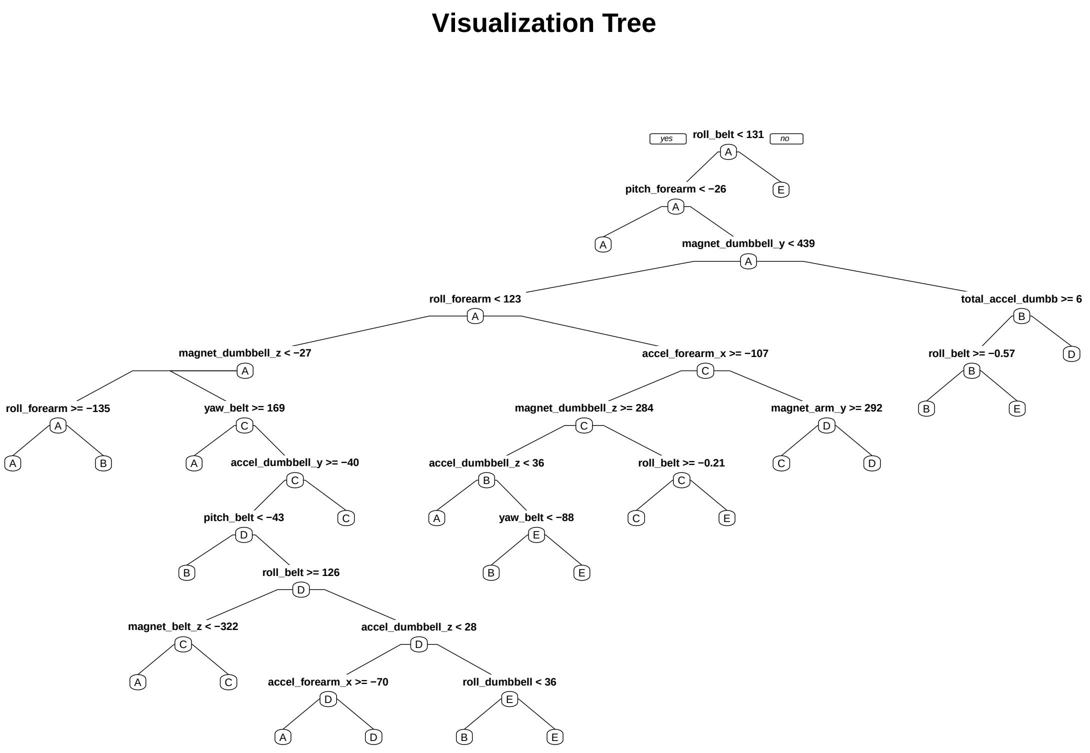
#Ploting the correlation of the training data. Correlation among the different variables.

```
plotdata <- cor(trainingdata_partition[,-length(names(trainingdata_partition))])
corrplot(plotdata, main = "Correlation Matrix Visualization", method="color")
```



#Tree model ##In this model it is possible to check the correct and incorrect barbell lifts executed by the participants.

```
tree_model <- rpart(classe ~ ., data=trainingdata_partition, method="class")
prp(tree_model, main = "Visualization Tree", type = 1 ) # fast plot
```


Visualization Tree
yes
roll_belt < 131
no
A
pitch_forearm < −26
E
A
A
magnet_dumbbell_y < 439
A
roll_forearm < 123
total_accel_dumbb >= 6
A
B
magnet_dumbbell_z < −27
accel_forearm_x >= −107
roll_belt >= −0.57
D
A
C
B
roll_forearm >= −135
yaw_belt >= 169
magnet_dumbbell_z >= 284
magnet_arm_y >= 292
B
E
A
C
C
D
A
B
A
accel_dumbbell_y >= −40
accel_dumbbell_z < 36
roll_belt >= −0.21
C
D
C
B
C
pitch_belt < −43
C
A
yaw_belt < −88
C
E
D
E
B
roll_belt >= 126
B
E
D
magnet_belt_z < −322
accel_dumbbell_z < 28
C
D
A
C
accel_forearm_x >= −70
roll_dumbbell < 36
D
E
A
D
B
E