Machine learning course project

Alejandro Carrascosa

2024-05-03

Introduction

The aim of this work is to develop a model to help predict how people execute certain weightlifting exercises using their body performance monitoring devices. The data we have to build the model are the position of the sensors in an XYZ coordinate space, and their trajectory measured as Euler angles (roll, pitch and yaw). The authors of the original study also calculated some indices describing the distribution of the latter measures over time (skewness, kurtosis, mean…), using moving windows of 0.5 to 2.5 seconds.

Raw data preparation

Once I open the raw dataset, I notice some problems. The data provided to train and validate the model are full of NAs. Some of these NAs arise from the fact that the indices extracted with moving windows (skewness, variance, mean…) only have values in the last rows of each time window. One option to develop the model could be to use only the rows that have values for all the variables (those with "yes" values in the "new_window" column) but, as the dataset provided to test the model and make predictions does not have values for those columns either, I decided to simply remove those columns, and develop the model using only the characteristics of the positions and angles of the devices at each time.

```
#dataset for model developing
data <- read.csv2("pml-training.csv", header = T, dec = ".", sep = ",")
#dataset for predictions
data_test <- read.csv2("pml-testing.csv", header = T, dec = ".", sep = ",")

#The datasets do not have some missing values set as NA, so we transform them into NAs.

data[data == '#DIV/0!'] <- NA
data[data == ''] <- NA

data_test[data_test == '#DIV/0!'] <- NA
data_test[data_test == ''] <- NA

#Now I remove the columns that have several NAs, those related with  distribution indices in moving windows. I al
so remove the first column, that is just the number of the row, but is highly correlated with some data

data_clean <- data[c(2,8:11,37:49,113:124,140,151:159,160)]
data_test_clean <- data_test[c(2,8:11,37:49,113:124,140,151:159,160)]
```

Model development

The factors in the data are mostly continuous, but there is also a very important categorical characteristic, the "user name". It is easy to imagine that, depending on the individual, the relationship between the positions and angles of the devices and the way the exercise is performed may vary. For this reason, I decided to use the random forest to build the predictive model, since the random forest can handle both continuous and categorical values and no assumptions about the distribution of the variables need to be met. I build the model using the "caret" package. To test the model I used the random sampling cross-validation approach, using 70% of the database to train the model and the remaining 30% to test it, before using it to predict the test data provided by the instructors.

```
library(caret)

#Spliting the data in train and test datasets
data_train <- createDataPartition(data_clean$classe, p = 0.7, list = F)
training <- data_clean[data_train,]
testing <- data_clean[-data_train,]

#Chaging the response variable "classe" from character to factor, for better functioning of random forest functio
n.
set.seed(125)
training$classe <- as.factor(training$classe)
testing$classe <- as.factor(testing$classe)

#Random forest model development
model_rf <- train(classe ~., method = "rf", data = training)
```

This model “model_rf” was used to predict the values of the variable “classe” in the test data set. With the predicted values and the original test values, a confusion matrix is constructed to check the performance of the model.

```
predictions <- predict(model_rf, testing)

confusionMatrix(predictions,testing$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1673   14    0    0    0
##          B    1 1122    9    0    0
##          C    0    3 1012   16    0
##          D    0    0    5  948    4
##          E    0    0    0    0 1078
## 
## Overall Statistics
## 
##                Accuracy : 0.9912
##                  95% CI : (0.9884, 0.9934)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9888
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9851   0.9864   0.9834   0.9963
## Specificity            0.9967   0.9979   0.9961   0.9982   1.0000
## Pos Pred Value         0.9917   0.9912   0.9816   0.9906   1.0000
## Neg Pred Value         0.9998   0.9964   0.9971   0.9968   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1907   0.1720   0.1611   0.1832
## Detection Prevalence   0.2867   0.1924   0.1752   0.1626   0.1832
## Balanced Accuracy      0.9980   0.9915   0.9912   0.9908   0.9982
```

Results

The accuracy of this model is extremely high, more than 99% of the values are predicted correctly. Therefore, it can be concluded that the model is suitable for the proposals of this work.

Checking the importance of the variables in the model (Fig. 1) it can be seen that “roll belt” is the most important variable defining the way exercises are performed. On the other hand, the “user name” variables (which the model transforms into dummy variables) are not too important, which is actually good news, as the predictive model will depend less on the people wearing the devices and more on their movements, which makes the model more suitable for making predictions for any individual.

```
plot(varImp(model_rf))
```

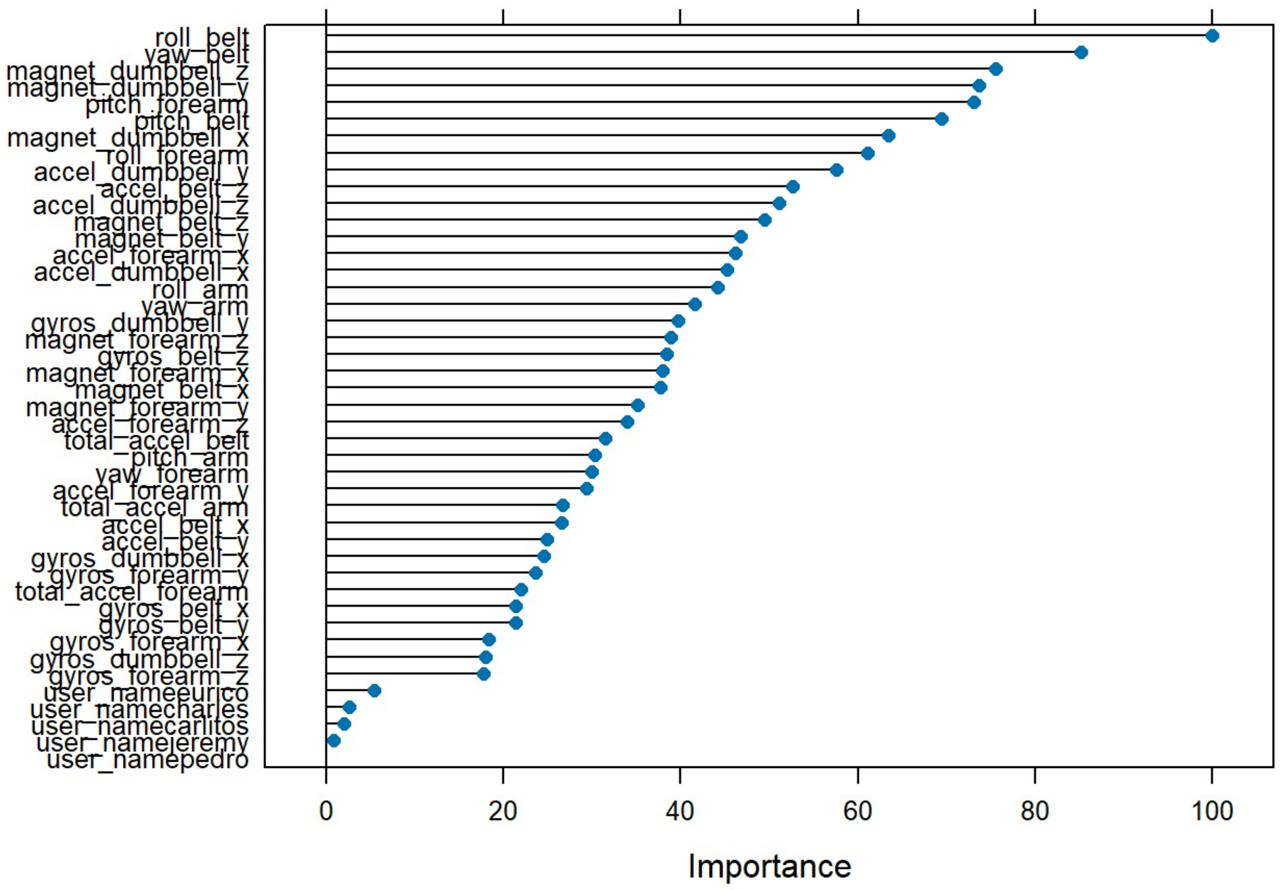

Figure 1. Importance of the variables in the random forest model.

Predictions for the testing data provided by the instructors

We have been asked to predict “classe” values from a set of test data. The predicted values are presented before.

```
library(flextable)
library(dplyr)

testing_predictions <- predict(model_rf, data_test_clean)

table_data <- data.frame(predicted_classe = testing_predictions, problem_id = data_test$problem_id)

text.t <- flextable(table_data, col_keys = c("predicted_classe", "problem_id")) %>%
    align(align = "center", part = "all") %>%
    set_table_properties(layout = "autofit", width = 1)

text.t
```

predicted_classe	problem_id
B	1
A	2
B	3
A	4
A	5
E	6
D	7
B	8
A	9
A	10

B	11
C	12
B	13
A	14
E	15
E	16
A	17
B	18
B	19
B	20