Predictions on corectness of exercise movements

Anamaria Ahuis

20 March 2017

Executive summary / Conclusions

This document summarizes the manner in which predictions regarding the corectness of 20 physical exercises were made. The predictions are based on data collected by monitoring activity of various muscles in the athletes' body.

The accuracy of the prediction is 25%, to be expected given that the provided testing dataset has 20 rows, while the outcome is a factor with 5 levels. Out of the 20 testing data records, 8 were predicted as correct exercise (A value for the outcome).

Data Analysis

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

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

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

```
## Warning: package 'tree' was built under R version 3.3.3
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

```
## randomForest 4.6-12
```

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

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
training <- read.csv('pml-training.csv',stringsAsFactors = FALSE)
testing <- read.csv('pml-testing.csv',stringsAsFactors = FALSE)
```

The data sets have features whose values are empty character strings or “DIV/0”. These features will be eliminated from the training set thus reducing the number of possible predictors to 144.

There are also (85) variables containing empty or NA values besides numeric values.

For these features, information will be imputed where missing.

```
## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt

## Warning: NAs durch Umwandlung erzeugt
```

```
preObj <- preProcess(training[,-145],'knnImpute')
#impute NA values
xxx <- predict(preObj,training[,-145])
yyy <- predict(preObj,testing[,-145])

for (j in low_info) training[,j]<- xxx[,j]
for (j in low_info) testing[,j]<- yyy[,j]
```

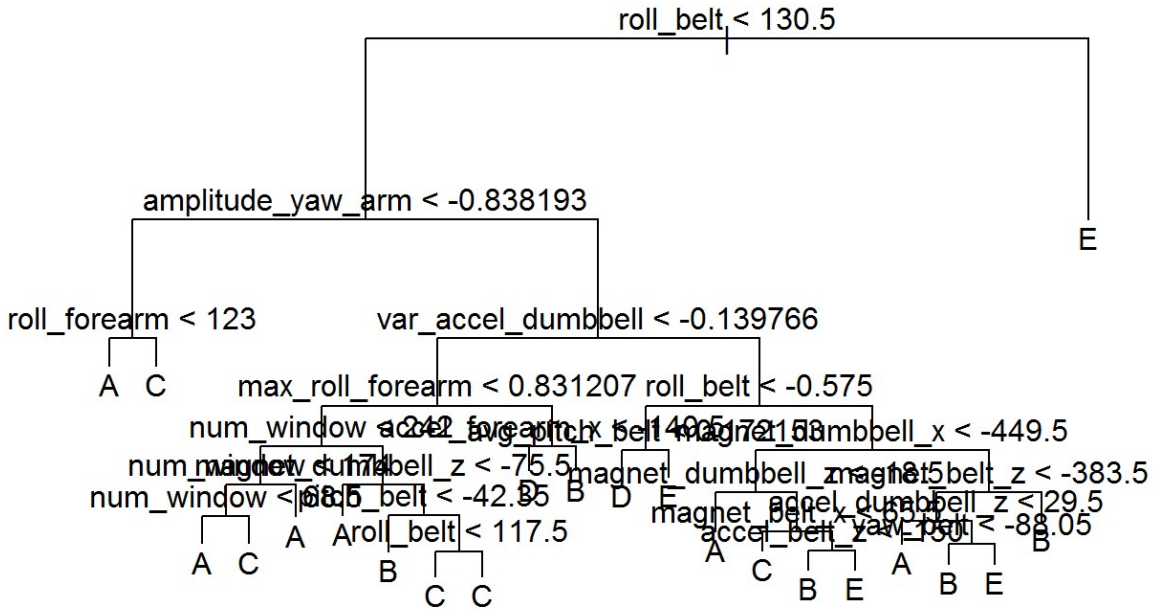
The training set has many observations, while in the provided testing set there is no outcome variable ‘classe’, as well as no other variable with 5 levels - and thus comparable to ‘classe’.For this reason, the provided training set is split into a training and testing data set, allowing fitting the algorithm as well as checking its accuracy. When a good accuracy is obtained, the predictions will be made on the provided test set of 20 observations.

Because the outcome - ‘classe’ - is a categorical variable with more than two values (and thus logistic regression is excluded from start), the best suitable models are classification ones, namely: KNN, decision trees and random forests. KNN can be slow when computing the distances on such a large dataset as the training one. Also, KNN is affected by a large amount of predictors, as in this case. For these reasons KNN is not pursued as the model of choice for this analysis, which leaves it with Decision Trees and respectively Random Forests.

```
bigTree <- tree(classe~.,data=train)
summary(bigTree)
```

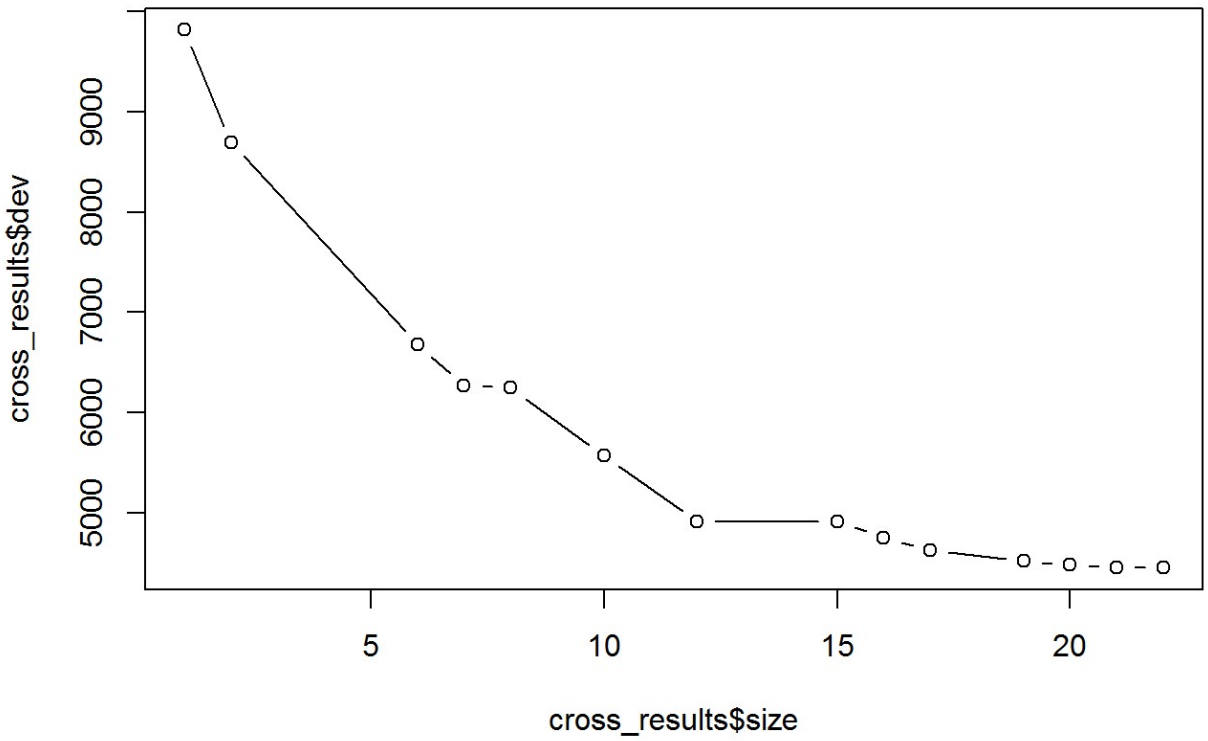
```
##
## Classification tree:
## tree(formula = classe ~ ., data = train)
## Variables actually used in tree construction:
##  [1] "roll_belt"          "amplitude_yaw_arm"  "roll_forearm"
##  [4] "var_accel_dumbbell" "max_roll_forearm"   "num_window"
##  [7] "magnet_dumbbell_z"  "pitch_belt"         "accel_forearm_x"
## [10] "avg_pitch_belt"     "magnet_dumbbell_x"  "magnet_belt_x"
## [13] "accel_belt_z"       "magnet_belt_z"      "accel_dumbbell_z"
## [16] "yaw_belt"
## Number of terminal nodes:  22
## Residual mean deviance:  1.596 = 21880 / 13710
## Misclassification error rate: 0.3155 = 4334 / 13735
```

```
plot(bigTree);text(bigTree)
```



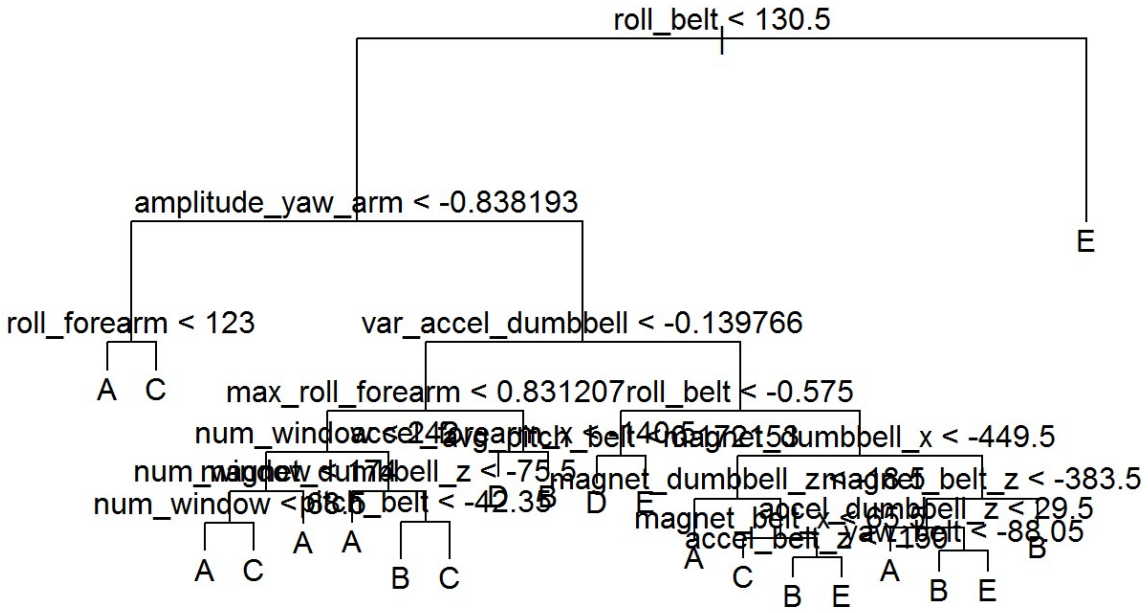
This tree achieves a 68% accuracy rate on the training data. The result can be optimized by checking the optimal tree size through cross-validation:

```
set.seed(1)
cross_results <- cv.tree(bigTree,FUN=prune.misclass)
plot(cross_results$size,cross_results$dev,type='b')
```



The optimal tree size seems to be 21, so the bigTree is pruned to this size:

```
prunedTree <- prune.misclass(bigTree,best=21)
plot(prunedTree);text(prunedTree)
```



And finally the accuracy of the model can be tested against the test dataset:

```
accuracy <- function(preds,answers) { sum((preds==answers)/(length(answers)))}
predictions <- predict(prunedTree,newdata=test,type="class")
accuracy(predict(prunedTree,newdata=train),train$classe)
```

```
## [1] 0
```

```
accuracy(predictions,test$classe)
```

```
## [1] 0.6869373
```

Because of the zero accuracy on the training data set, randomForest is used:

```
forest <- randomForest(classe ~ ., data=train,importance=TRUE,ntree=200,mtry=3)
accuracy(predict(forest),train$classe)
```

```
## [1] 0.9754641
```

```
predictions <- predict(forest,newdata=test)
accuracy(predictions,test$classe)
```

```
## [1] 0.9779174
```

Finally, using the fitted randomForest algorithm an attempt is made at predicting the corectness of the 20 exercise samples in the provided testing dataset.

```
predictions <- predict(forest,newdata=testing)
predictions
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  A  A  A  C  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

```
accuracy(predictions,testing$classe)
```

```
## [1] 0.25
```