Machine Learning Project

Leonardo Simioni

2022-09-27

Synopsis

This project will analyse and perform some predictions using the data from accelerometers on the belt, forearm, arm, and dumbbell of 6 participants. These participants were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

The goal of this project is to predict the manner in which they did the exercise, which is the "classe" variable in the training set. The prediction model will also be used in the end to predict 20 different test cases.

The training data for this project are available here.

The test data are available here.

The data for this project come from this source.

Downloading and preprocessing the data

```
library(caret)
```

Loading the required packages

```
## Loading required package: ggplot2

## Loading required package: lattice
```

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

```
## 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
```

```r
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```r
trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trainFile <- "./data/pml-training.csv"
testFile  <- "./data/pml-testing.csv"
if (!file.exists("./data")) {
  dir.create("./data")
}
if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="curl")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="curl")
}
```

Downloading the data sets

```r
trainRaw <- read.csv("./data/pml-training.csv")
testRaw <- read.csv("./data/pml-testing.csv")
dim(trainRaw)
```

Reading the data sets

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

```r
dim(testRaw)
```

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

```r
sum(complete.cases(trainRaw))
```

Cleaning the data

```
## [1] 406
```

```r
trainRaw <- trainRaw[, colSums(is.na(trainRaw)) == 0]
testRaw <- testRaw[, colSums(is.na(testRaw)) == 0]
```

```r
classe <- trainRaw$classe
trainRemove <- grepl("^X|timestamp|window", names(trainRaw))
trainRaw <- trainRaw[, !trainRemove]
trainCleaned <- trainRaw[, sapply(trainRaw, is.numeric)]
trainCleaned$classe <- classe
testRemove <- grepl("^X|timestamp|window", names(testRaw))
testRaw <- testRaw[, !testRemove]
testCleaned <- testRaw[, sapply(testRaw, is.numeric)]
```

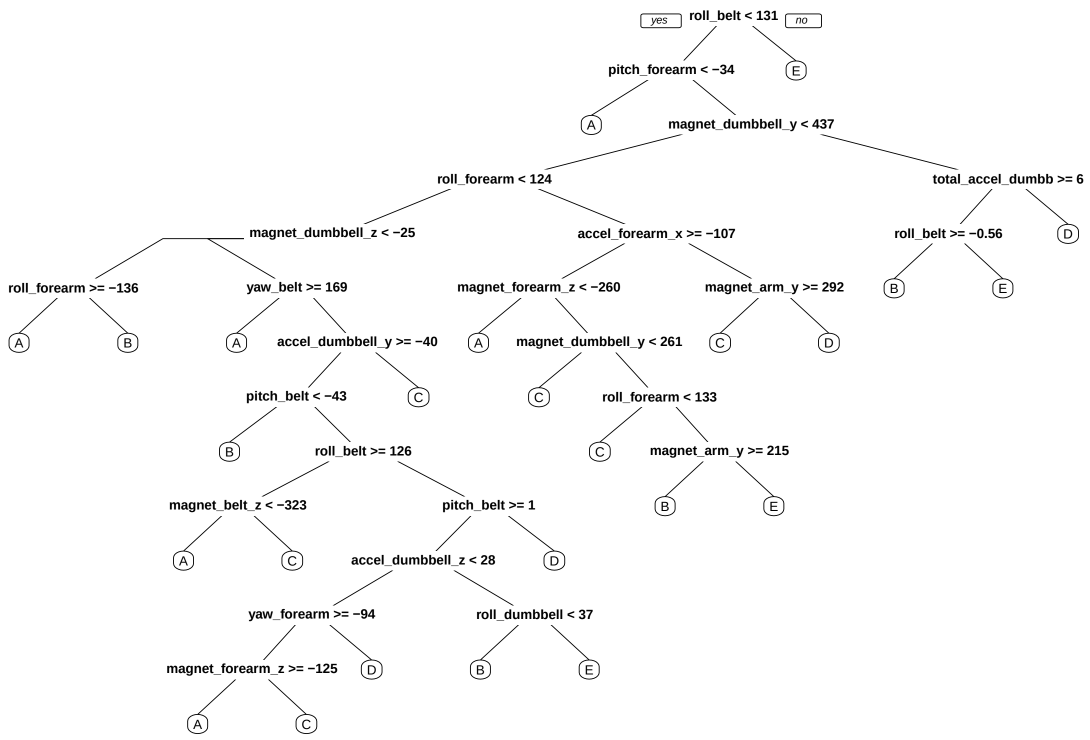
Removing the missing (NA) values and irrelevant columns

```r
set.seed(22519)
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=F)
trainData <- trainCleaned[inTrain, ]
testData <- trainCleaned[-inTrain, ]
```

Splitting the training set: Training data set (70%) and validation data set (30%)

Data Modeling

```r
treeModel <- rpart(classe ~ ., data=trainData, method="class")
prp(treeModel)
```



Decision Tree

```r
controlRf <- trainControl(method="cv", 5)
modelRf <- train(classe ~ ., data=trainData, method="rf", trControl=controlRf, ntree=250)
modelRf
```

Using Random Forest and 5-fold cross validation

```
## 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: 10990, 10991, 10988, 10990, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9904632  0.9879354
##   27    0.9913368  0.9890413
##   52    0.9827463  0.9781726
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

```
predictRf <- predict(modelRf, testData)
confusionMatrix(table(testData$classe, predictRf))
```

Estimating the performance on the validation data set

```
## Confusion Matrix and Statistics
##
##    predictRf
##        A    B    C    D    E
##   A 1671    1    2    0    0
##   B    5 1129    4    1    0
##   C    0    4 1019    3    0
##   D    0    0    9  955    0
##   E    0    0    4    2 1076
##
## Overall Statistics
##
##                Accuracy : 0.9941
##                  95% CI : (0.9917, 0.9959)
##     No Information Rate : 0.2848
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9925
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9956   0.9817   0.9938   1.0000
## Specificity            0.9993   0.9979   0.9986   0.9982   0.9988
## Pos Pred Value         0.9982   0.9912   0.9932   0.9907   0.9945
## Neg Pred Value         0.9988   0.9989   0.9961   0.9988   1.0000
## Prevalence             0.2848   0.1927   0.1764   0.1633   0.1828
## Detection Rate         0.2839   0.1918   0.1732   0.1623   0.1828
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9982   0.9967   0.9901   0.9960   0.9994
```

```
oose <- 1 - as.numeric(confusionMatrix(table(testData$classe, predictRf))$overall[1])
oose
```

```
## [1] 0.005947324
```

The estimated accuracy of the model is 99.37% and the estimated out-of-sample error is 0.62%.

Predicting 20 different test cases

```
result <- predict(modelRf, testCleaned[, -length(names(testCleaned))])
result
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```