Machine Learning Assignment

kd88

2016-02-24

Note

In order to view this document with all variables rendered correctly, please download and view the .html in-browser, or alternatively view the automatically generated .pdf.

Introduction

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.

The aim of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell of a group of participants in order to predict the manner in which they 6 other participants performed the exercise. The particpants were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The variable that will be used to denote the 5 ways of performing the exercise will be referred to as "classe". More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Settings

The following libraries and settings have been applied to this analysis.

```
#---- Libraries
library(ggplot2)
library(lattice)
library(caret)
library(randomForest)
```

```
## 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
```

```
#---- Parameters
set.seed(1988)
varName   <- c("classe")
corThresh <- c(0.1)
partition <- c(0.7)
nCrossVal <- c(5)
nTree     <- c(100)
```

The dataset

The training and test data are read from the corresponding *.csv files:

```
training = read.csv("~/Downloads/pml-training.csv")
testing  = read.csv("~/Downloads/pml-testing.csv")
```

The "training"" dataset will later be split into a "main" training dataset and a "validation" dataset. The "testing" dataset contains the data of the 6 participants to be predicted.

There are 160 variables in the dataset, which have the following names:

```
names(training)
```

```
##   [1] "X"                       "user_name"
##   [3] "raw_timestamp_part_1"    "raw_timestamp_part_2"
##   [5] "cvtd_timestamp"          "new_window"
##   [7] "num_window"              "roll_belt"
##   [9] "pitch_belt"              "yaw_belt"
##  [11] "total_accel_belt"        "kurtosis_roll_belt"
##  [13] "kurtosis_picth_belt"     "kurtosis_yaw_belt"
##  [15] "skewness_roll_belt"      "skewness_roll_belt.1"
##  [17] "skewness_yaw_belt"       "max_roll_belt"
##  [19] "max_picth_belt"          "max_yaw_belt"
##  [21] "min_roll_belt"           "min_pitch_belt"
##  [23] "min_yaw_belt"            "amplitude_roll_belt"
##  [25] "amplitude_pitch_belt"    "amplitude_yaw_belt"
##  [27] "var_total_accel_belt"    "avg_roll_belt"
##  [29] "stddev_roll_belt"        "var_roll_belt"
##  [31] "avg_pitch_belt"          "stddev_pitch_belt"
##  [33] "var_pitch_belt"          "avg_yaw_belt"
##  [35] "stddev_yaw_belt"         "var_yaw_belt"
##  [37] "gyros_belt_x"            "gyros_belt_y"
##  [39] "gyros_belt_z"            "accel_belt_x"
##  [41] "accel_belt_y"            "accel_belt_z"
##  [43] "magnet_belt_x"           "magnet_belt_y"
##  [45] "magnet_belt_z"           "roll_arm"
##  [47] "pitch_arm"               "yaw_arm"
##  [49] "total_accel_arm"         "var_accel_arm"
##  [51] "avg_roll_arm"            "stddev_roll_arm"
##  [53] "var_roll_arm"            "avg_pitch_arm"
##  [55] "stddev_pitch_arm"        "var_pitch_arm"
##  [57] "avg_yaw_arm"             "stddev_yaw_arm"
##  [59] "var_yaw_arm"             "gyros_arm_x"
```

```
##  [61] "gyros_arm_y"              "gyros_arm_z"
##  [63] "accel_arm_x"              "accel_arm_y"
##  [65] "accel_arm_z"              "magnet_arm_x"
##  [67] "magnet_arm_y"             "magnet_arm_z"
##  [69] "kurtosis_roll_arm"        "kurtosis_picth_arm"
##  [71] "kurtosis_yaw_arm"         "skewness_roll_arm"
##  [73] "skewness_pitch_arm"       "skewness_yaw_arm"
##  [75] "max_roll_arm"             "max_picth_arm"
##  [77] "max_yaw_arm"              "min_roll_arm"
##  [79] "min_pitch_arm"            "min_yaw_arm"
##  [81] "amplitude_roll_arm"       "amplitude_pitch_arm"
##  [83] "amplitude_yaw_arm"        "roll_dumbbell"
##  [85] "pitch_dumbbell"           "yaw_dumbbell"
##  [87] "kurtosis_roll_dumbbell"   "kurtosis_picth_dumbbell"
##  [89] "kurtosis_yaw_dumbbell"    "skewness_roll_dumbbell"
##  [91] "skewness_pitch_dumbbell"  "skewness_yaw_dumbbell"
##  [93] "max_roll_dumbbell"        "max_picth_dumbbell"
##  [95] "max_yaw_dumbbell"         "min_roll_dumbbell"
##  [97] "min_pitch_dumbbell"       "min_yaw_dumbbell"
##  [99] "amplitude_roll_dumbbell"  "amplitude_pitch_dumbbell"
## [101] "amplitude_yaw_dumbbell"   "total_accel_dumbbell"
## [103] "var_accel_dumbbell"       "avg_roll_dumbbell"
## [105] "stddev_roll_dumbbell"     "var_roll_dumbbell"
## [107] "avg_pitch_dumbbell"       "stddev_pitch_dumbbell"
## [109] "var_pitch_dumbbell"       "avg_yaw_dumbbell"
## [111] "stddev_yaw_dumbbell"      "var_yaw_dumbbell"
## [113] "gyros_dumbbell_x"         "gyros_dumbbell_y"
## [115] "gyros_dumbbell_z"         "accel_dumbbell_x"
## [117] "accel_dumbbell_y"         "accel_dumbbell_z"
## [119] "magnet_dumbbell_x"        "magnet_dumbbell_y"
## [121] "magnet_dumbbell_z"        "roll_forearm"
## [123] "pitch_forearm"            "yaw_forearm"
## [125] "kurtosis_roll_forearm"    "kurtosis_picth_forearm"
## [127] "kurtosis_yaw_forearm"     "skewness_roll_forearm"
## [129] "skewness_pitch_forearm"   "skewness_yaw_forearm"
## [131] "max_roll_forearm"         "max_picth_forearm"
## [133] "max_yaw_forearm"          "min_roll_forearm"
## [135] "min_pitch_forearm"        "min_yaw_forearm"
## [137] "amplitude_roll_forearm"   "amplitude_pitch_forearm"
## [139] "amplitude_yaw_forearm"    "total_accel_forearm"
## [141] "var_accel_forearm"        "avg_roll_forearm"
## [143] "stddev_roll_forearm"      "var_roll_forearm"
## [145] "avg_pitch_forearm"        "stddev_pitch_forearm"
## [147] "var_pitch_forearm"        "avg_yaw_forearm"
## [149] "stddev_yaw_forearm"       "var_yaw_forearm"
## [151] "gyros_forearm_x"          "gyros_forearm_y"
## [153] "gyros_forearm_z"          "accel_forearm_x"
## [155] "accel_forearm_y"          "accel_forearm_z"
## [157] "magnet_forearm_x"         "magnet_forearm_y"
## [159] "magnet_forearm_z"         "classe"
```

The variable "X" simply corresponds to the row number in the file, so this is removed.

```
training <- training[-which(names(training) %in% c("X"))]
```

There are 67 columns containing missing values, so these are removed.

```
training <- training[, colSums(is.na(training)) == 0]
```

This leaves 92 variables. There are also a further 1 variables which has non-numeric values. For simplicity, we remove these.

```
numNames <- c("classe")
for(i in names(training)){
    if(is.numeric(unlist(training[i]))){
        numNames <- c(numNames,i)
    }
}
training <- subset(x=training,select=numNames)
```

There are now 56 variables. To further reduce the number of variables, bi remove variables which are not particularly correlated with the "classe" variable. The threshold "0.1" has been optimised for reducing the algorithm running time whilst maintaining high accuracy.

```
corrNames <- varName
corrs <- c()
for(i in names(training)){
    if(i != varName){
        correlation <- cor(as.numeric(unlist(training[varName])),training[i])
        if(abs(correlation) > corThresh){
            corrNames <- c(corrNames,i)
            corrs[i] <- abs(correlation)
        }
    }
}
training <- subset(x=training,select=corrNames)
```

There are now only 16 variables.

Random forest training

A "validation" dataset is then split from the main "training" dataset, with a partition of 0.7.

```
inTrain <- createDataPartition(training$classe, p=partition, list=FALSE)
validation <- training[-inTrain, ]
training   <- training[inTrain, ]
```

A random forest ("rf") is trained using a 5-fold cross validation ("cv") with `nTree` trees. The parameters have been optimised for algorithmic speed and accuracy.

```
modFit <- train(classe ~ .,
                data=training,
                method="rf",
                trControl=trainControl(method="cv", nCrossVal),
                ntree=nTree)
```

The details of the model fit are given below.

```
print(modFit)
```

```
## Random Forest
##
## 13737 samples
##    15 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10991, 10989, 10991, 10988, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9501356  0.9369020  0.006791299  0.008586731
##    8    0.9495534  0.9361724  0.005723515  0.007227955
##   15    0.9429297  0.9277855  0.006800118  0.008597854
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

Prediction

A prediction is applied to the "validation" dataset.

```
pred <- predict(modFit,validation)
#---- A function to test the accuracy of the prediction
myaccuracy = function(values,prediction){sum(prediction==values)/length(values)}
```

The prediction has an accuracy of 96.346644%, which is sufficient for this study. The prediction for the testing dataset is therefore: B, A, B, A, A, E, D, B, A, A, B, C, B, A, E, E, A, B, B, B.