Practical Machine Learning

Yadder Aceituno

August 19, 2019

Libraries

```
library(caret)
```

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

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

Overview

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, the 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.

The goal is to predict the manner they did the different excercises. There is a variable called classe which stores 5 different classes:

- Class A - exactly according to the specification.
- Class B - throwing the elbows to the front.
- Class C - lifting the dumbbell only halfway.
- Class D - lowering the dumbbell only halfway.
- Class E - throwing the hips to the front.

We'll build different prediction models to compare them and decide which model is the best to predict the variable classe.

Getting Data

The data comes from

Getting the testing and training data

```
training <- read.csv("pml-training.csv")
testing <- read.csv("pml-testing.csv")

# Checking dimensions
dim(training)
```

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

```
dim(testing)
```

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

We can see that training data frame has 19622 rows. We will use this data frame to build our models. We also can see that testing data frame has 20 rows. We will use this data frame to check each model builded.

Partition Data

First, we need to create a partition for our training data. We will create a variable called myPartition which will hold 70% of the initial training data. Also we will create a variable called myTesting whicho will hold 30% of the initial training data.

```
# Setting seed to reproduce the results
set.seed(12345)

inTrain <- createDataPartition(training$classe, p = 0.7, list = FALSE)

myTraining <- training[inTrain,]
myTesting <- training[-inTrain,]


# Checking dimensions
dim(myTraining)
```

```
## [1] 13737   160
```

```
dim(myTesting)
```

```
## [1] 5885  160
```

Cleaning Training Data

Identifying Covariates

First we need to take a first look to identify those variables which will help to build our model. Those are:

```
# Identifying covariates
covariates <- c("classe", "roll_belt","pitch_belt", "yaw_belt", "total_accel_belt", "gyros_belt_x", "gyr
"accel_belt_x", "accel_belt_y", "accel_belt_z", "magnet_belt_x", "magnet_belt_y", "magnet_belt_z", "roll
"total_accel_arm", "gyros_arm_x", "gyros_arm_y", "gyros_arm_z", "accel_arm_x", "accel_arm_y", "accel_arm
"magnet_arm_z", "roll_dumbbell", "pitch_dumbbell", "yaw_dumbbell", "gyros_dumbbell_x", "gyros_dumbbell_y
"accel_dumbbell_y", "accel_dumbbell_z", "magnet_dumbbell_x", "magnet_dumbbell_y", "magnet_dumbbell_z",
"yaw_forearm", "total_accel_forearm", "gyros_forearm_x", "gyros_forearm_y", "gyros_forearm_z", "accel_fo
"accel_forearm_z","magnet_forearm_x", "magnet_forearm_y", "magnet_forearm_z")

# Reducing covariates
myTraining <- myTraining[, covariates]
myTesting <- myTesting[, covariates]
```

Removing Zero Covariates

We need to identify covariates that will not help to build our model. Using nearZeroVar function we can identify those variables which have low variabilty.

```
nzv <- nearZeroVar(myTraining, saveMetrics = TRUE)
```

```r
# Looking covariates with near zero variabilty
nzv[nzv$nzv == TRUE,]
```

```
## [1] freqRatio     percentUnique zeroVar       nzv
## <0 rows> (or 0-length row.names)
```

We can see that all variables have variabity. Using nearZeroVar function we didn't remove any variable.

Building Models