

# Predicting the manner and quality of Barbell Lifts using data from Fitness Devices

Rohin Bhagavatula

24/10/20

```
if(!file.exists("pml-training.csv"))
{
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "pml-training.csv")
}
dataset <- read.csv("pml-training.csv", na.strings = c("NA", ""))
if(!file.exists("pml-testing.csv"))
{
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "pml-testing.csv")
}
validation <- read.csv("pml-testing.csv")
```

importing files:

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
## randomForest 4.6-14
```

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

```
##
```

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

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

```
##
```

```
## margin
```

```
library(ggplot2)
```

```
set.seed(17)
```

```
inTrain = createDataPartition(y=dataset$classe, p=0.7, list=FALSE)
training_data = dataset[inTrain,]
testing_data = dataset[-inTrain,]
```

elimination of unwanted columns:

```
na = sapply(training_data, function(x) {sum(is.na(x))}) #Make a vector of all the columns and the number of NA
na
```

##	X	user_name	raw_timestamp_part_1
##	0	0	0
##	raw_timestamp_part_2	cvtd_timestamp	new_window
##	0	0	0
##	num_window	roll_belt	pitch_belt
##	0	0	0
##	yaw_belt	total_accel_belt	kurtosis_roll_belt
##	0	0	13443
##	kurtosis_pitch_belt	kurtosis_yaw_belt	skewness_roll_belt
##	13443	13443	13443
##	skewness_roll_belt.1	skewness_yaw_belt	max_roll_belt
##	13443	13443	13443
##	max_pitch_belt	max_yaw_belt	min_roll_belt
##	13443	13443	13443
##	min_pitch_belt	min_yaw_belt	amplitude_roll_belt
##	13443	13443	13443
##	amplitude_pitch_belt	amplitude_yaw_belt	var_total_accel_belt
##	13443	13443	13443
##	avg_roll_belt	stddev_roll_belt	var_roll_belt
##	13443	13443	13443
##	avg_pitch_belt	stddev_pitch_belt	var_pitch_belt
##	13443	13443	13443
##	avg_yaw_belt	stddev_yaw_belt	var_yaw_belt
##	13443	13443	13443
##	gyros_belt_x	gyros_belt_y	gyros_belt_z
##	0	0	0
##	accel_belt_x	accel_belt_y	accel_belt_z
##	0	0	0
##	magnet_belt_x	magnet_belt_y	magnet_belt_z
##	0	0	0
##	roll_arm	pitch_arm	yaw_arm
##	0	0	0
##	total_accel_arm	var_accel_arm	avg_roll_arm
##	0	13443	13443
##	stddev_roll_arm	var_roll_arm	avg_pitch_arm
##	13443	13443	13443
##	stddev_pitch_arm	var_pitch_arm	avg_yaw_arm
##	13443	13443	13443
##	stddev_yaw_arm	var_yaw_arm	gyros_arm_x

##	13443	13443	0
##	gyros_arm_y	gyros_arm_z	accel_arm_x
##	0	0	0
##	accel_arm_y	accel_arm_z	magnet_arm_x
##	0	0	0
##	magnet_arm_y	magnet_arm_z	kurtosis_roll_arm
##	0	0	13443
##	kurtosis_picth_arm	kurtosis_yaw_arm	skewness_roll_arm
##	13443	13443	13443
##	skewness_pitch_arm	skewness_yaw_arm	max_roll_arm
##	13443	13443	13443
##	max_picth_arm	max_yaw_arm	min_roll_arm
##	13443	13443	13443
##	min_pitch_arm	min_yaw_arm	amplitude_roll_arm
##	13443	13443	13443
##	amplitude_pitch_arm	amplitude_yaw_arm	roll_dumbbell
##	13443	13443	0
##	pitch_dumbbell	yaw_dumbbell	kurtosis_roll_dumbbell
##	0	0	13443
##	kurtosis_picth_dumbbell	kurtosis_yaw_dumbbell	skewness_roll_dumbbell
##	13443	13443	13443
##	skewness_pitch_dumbbell	skewness_yaw_dumbbell	max_roll_dumbbell
##	13443	13443	13443
##	max_picth_dumbbell	max_yaw_dumbbell	min_roll_dumbbell
##	13443	13443	13443
##	min_pitch_dumbbell	min_yaw_dumbbell	amplitude_roll_dumbbell
##	13443	13443	13443
##	amplitude_pitch_dumbbell	amplitude_yaw_dumbbell	total_accel_dumbbell
##	13443	13443	0
##	var_accel_dumbbell	avg_roll_dumbbell	stddev_roll_dumbbell
##	13443	13443	13443
##	var_roll_dumbbell	avg_pitch_dumbbell	stddev_pitch_dumbbell
##	13443	13443	13443
##	var_pitch_dumbbell	avg_yaw_dumbbell	stddev_yaw_dumbbell
##	13443	13443	13443
##	var_yaw_dumbbell	gyros_dumbbell_x	gyros_dumbbell_y
##	13443	0	0
##	gyros_dumbbell_z	accel_dumbbell_x	accel_dumbbell_y
##	0	0	0
##	accel_dumbbell_z	magnet_dumbbell_x	magnet_dumbbell_y
##	0	0	0
##	magnet_dumbbell_z	roll_forearm	pitch_forearm
##	0	0	0
##	yaw_forearm	kurtosis_roll_forearm	kurtosis_picth_forearm
##	0	13443	13443
##	kurtosis_yaw_forearm	skewness_roll_forearm	skewness_pitch_forearm
##	13443	13443	13443
##	skewness_yaw_forearm	max_roll_forearm	max_picth_forearm
##	13443	13443	13443
##	max_yaw_forearm	min_roll_forearm	min_pitch_forearm
##	13443	13443	13443
##	min_yaw_forearm	amplitude_roll_forearm	amplitude_pitch_forearm
##	13443	13443	13443
##	amplitude_yaw_forearm	total_accel_forearm	var_accel_forearm

```
##          13443          0          13443
##      avg_roll_forearm      stddev_roll_forearm      var_roll_forearm
##          13443          13443          13443
##      avg_pitch_forearm      stddev_pitch_forearm      var_pitch_forearm
##          13443          13443          13443
##      avg_yaw_forearm      stddev_yaw_forearm      var_yaw_forearm
##          13443          13443          13443
##      gyros_forearm_x      gyros_forearm_y      gyros_forearm_z
##          0          0          0
##      accel_forearm_x      accel_forearm_y      accel_forearm_z
##          0          0          0
##      magnet_forearm_x      magnet_forearm_y      magnet_forearm_z
##          0          0          0
##      classe
##          0
```

```
NAcolumn = names(na[na > 0]) #Vector with all the columns that has NA values
training_data = training_data[, !names(training_data) %in% NAcolumn]
names(training_data)
```

```
## [1] "X"          "user_name"      "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvt_d_timestamp" "new_window"
## [7] "num_window"      "roll_belt"      "pitch_belt"
## [10] "yaw_belt"        "total_accel_belt" "gyros_belt_x"
## [13] "gyros_belt_y"     "gyros_belt_z"    "accel_belt_x"
## [16] "accel_belt_y"     "accel_belt_z"    "magnet_belt_x"
## [19] "magnet_belt_y"    "magnet_belt_z"   "roll_arm"
## [22] "pitch_arm"       "yaw_arm"        "total_accel_arm"
## [25] "gyros_arm_x"     "gyros_arm_y"    "gyros_arm_z"
## [28] "accel_arm_x"     "accel_arm_y"    "accel_arm_z"
## [31] "magnet_arm_x"    "magnet_arm_y"   "magnet_arm_z"
## [34] "roll_dumbbell"   "pitch_dumbbell" "yaw_dumbbell"
## [37] "total_accel_dumbbell" "gyros_dumbbell_x" "gyros_dumbbell_y"
## [40] "gyros_dumbbell_z" "accel_dumbbell_x" "accel_dumbbell_y"
## [43] "accel_dumbbell_z" "magnet_dumbbell_x" "magnet_dumbbell_y"
## [46] "magnet_dumbbell_z" "roll_forearm"    "pitch_forearm"
## [49] "yaw_forearm"     "total_accel_forearm" "gyros_forearm_x"
## [52] "gyros_forearm_y" "gyros_forearm_z" "accel_forearm_x"
## [55] "accel_forearm_y" "accel_forearm_z" "magnet_forearm_x"
## [58] "magnet_forearm_y" "magnet_forearm_z" "classe"
```

```
training_data <- training_data[, !names(training_data) %in% c("X", "user_name", "raw_timestamp_part_1",
```

For validation set:

```
na = sapply(validation, function(x) {sum(is.na(x))}) #Make a vector of all the columns and the number o
NAcolumn = names(na[na > 0]) #Vector with all the columns that has NA values
validation = validation[, !names(validation) %in% NAcolumn]
validation <- validation[, !names(validation) %in% c("X", "user_name", "raw_timestamp_part_1", "raw_tim
```

For testing set:

```
na = sapply(testing_data, function(x) {sum(is.na(x))}) #Make a vector of all the columns and the number
NAcolumn = names(na[na > 0]) #Vector with all the columns that has NA values
testing_data = testing_data[, !names(testing_data) %in% NAcolumn]
testing_data <- testing_data[, !names(testing_data) %in% c("X", "user_name", "raw_timestamp_part_1", "r
```

```
model <- randomForest(classe ~ ., data=training_data, ntree = 50)
predictions <- predict(model, testing_data)
confusionMatrix(predictions, testing_data$classe)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction  A    B    C    D    E
##           A 1671   10    0    0    0
##           B    3 1124    2    0    0
##           C    0    5 1024   12    0
##           D    0    0    0  950    2
##           E    0    0    0    2 1080
```

```
## Overall Statistics
```

```
##
##           Accuracy : 0.9939
##           95% CI : (0.9915, 0.9957)
##           No Information Rate : 0.2845
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.9923
```

```
## McNemar's Test P-Value : NA
```

```
## Statistics by Class:
```

```
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity      0.9982  0.9868  0.9981  0.9855  0.9982
## Specificity      0.9976  0.9989  0.9965  0.9996  0.9996
## Pos Pred Value   0.9941  0.9956  0.9837  0.9979  0.9982
## Neg Pred Value   0.9993  0.9968  0.9996  0.9972  0.9996
## Prevalence       0.2845  0.1935  0.1743  0.1638  0.1839
## Detection Rate   0.2839  0.1910  0.1740  0.1614  0.1835
## Detection Prevalence 0.2856  0.1918  0.1769  0.1618  0.1839
## Balanced Accuracy 0.9979  0.9929  0.9973  0.9925  0.9989
```

```
modelAcc <- confusionMatrix(predictions, testing_data$classe)$overall[[1]]
```

The model is 0.9938828 accurate.

prediction:

```
predictions <- predict(model, validation)
predictions
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

```
## 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
```