Machine Learning: Course Project

Jeremy Peters

February 6, 2018

Executive Summary

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. In a study, Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).Only Class A corresponds to correct performance. The objective of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to build a machine learning algorithm to predict the manner/class type in which an exerise was completed. More information about the study and data set can be found in the section on the Weight Lifting Exercise Dataset at the following URL: http://groupware.les.inf.puc-rio.br/har. The training data for this project was download from the following URL: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv. The test data for this project was download from the following URL: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Exploratory Data Analysis

- Read the Training and Testing CSV files in table format, specify types of missing values (NA, empty strings and div0), and create data frames
- Display the internal structure of an R object and generate summary statistics of the training dataset
- The Training dataset contains 160 variables and 19,622 records
- The Testing dataset contains 160 variables and 20 records

```
# Load the required r packages
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
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
```

```
dfTrain <- read.csv("pml-training.csv", header = TRUE, na.strings=c("NA","#DIV/0!",""))
dfTest <- read.csv("pml-testing.csv", header = TRUE, na.strings=c("NA","#DIV/0!",""))

# Get variable names
names(dfTrain)
```

```
##   [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"
```

```r
str(dfTrain)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ user_name               : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 304277 368296 440390 484323 4844
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ num_window              : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
##  $ yaw_belt                : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ kurtosis_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt.1    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ avg_roll_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x           : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...
##  $ gyros_belt_y           : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z           : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x           : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y           : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z           : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x          : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y          : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z          : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
##  $ roll_arm               : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm              : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...
##  $ yaw_arm                : num  -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...
##  $ total_accel_arm        : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ var_accel_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x            : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y            : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z            : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x            : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y            : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z            : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x           : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y           : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z           : int  516 513 513 512 506 513 509 510 518 516 ...
##  $ kurtosis_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm            : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm            : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

```r
dim(dfTest)
```

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

```r
#summary(dfTrain)
summary(dfTrain$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Data Processing: Cleaning and Preparation

- Remove the first seven descriptive variables/fields (X/Id, user_name,raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timestamp, new_window, num_window) from both data sets that will not help predict the manner in which an exercise was completed.
- Remove the variables/fields from the data set that contain missing values
- Remove Near Zero Variance Variables
- The resulting Training and Testing datasets both have 53 variables/fields the last of which is the classe variable/field
- Split the cleaned training data set into a training set (75%) that will be used for prediction and a testing/validation set (25%) that will be used to determine out-of-sample errors

```r
dfTrain <- dfTrain[, -c(1:7)]
dfTest <- dfTest[, -c(1:7)]


dfTrain <- dfTrain[, colSums(is.na(dfTrain)) == 0]
dfTest <- dfTest[, colSums(is.na(dfTest)) == 0]

#Check if their are Near Zero Variance Variables to remove
nzVar <- nearZeroVar(dfTrain, saveMetrics = TRUE)
nzVar
```

```
##                      freqRatio percentUnique zeroVar   nzv
## roll_belt             1.101904     6.7781062   FALSE FALSE
## pitch_belt            1.036082     9.3772296   FALSE FALSE
```

```
## yaw_belt               1.058480     9.9734991   FALSE FALSE
## total_accel_belt       1.063160     0.1477933   FALSE FALSE
## gyros_belt_x           1.058651     0.7134849   FALSE FALSE
## gyros_belt_y           1.144000     0.3516461   FALSE FALSE
## gyros_belt_z           1.066214     0.8612782   FALSE FALSE
## accel_belt_x           1.055412     0.8357966   FALSE FALSE
## accel_belt_y           1.113725     0.7287738   FALSE FALSE
## accel_belt_z           1.078767     1.5237998   FALSE FALSE
## magnet_belt_x          1.090141     1.6664968   FALSE FALSE
## magnet_belt_y          1.099688     1.5187035   FALSE FALSE
## magnet_belt_z          1.006369     2.3290184   FALSE FALSE
## roll_arm              52.338462    13.5256345   FALSE FALSE
## pitch_arm             87.256410    15.7323412   FALSE FALSE
## yaw_arm               33.029126    14.6570176   FALSE FALSE
## total_accel_arm        1.024526     0.3363572   FALSE FALSE
## gyros_arm_x            1.015504     3.2769341   FALSE FALSE
## gyros_arm_y            1.454369     1.9162165   FALSE FALSE
## gyros_arm_z            1.110687     1.2638875   FALSE FALSE
## accel_arm_x            1.017341     3.9598410   FALSE FALSE
## accel_arm_y            1.140187     2.7367241   FALSE FALSE
## accel_arm_z            1.128000     4.0362858   FALSE FALSE
## magnet_arm_x           1.000000     6.8239731   FALSE FALSE
## magnet_arm_y           1.056818     4.4439914   FALSE FALSE
## magnet_arm_z           1.036364     6.4468454   FALSE FALSE
## roll_dumbbell          1.022388    84.2065029   FALSE FALSE
## pitch_dumbbell         2.277372    81.7449801   FALSE FALSE
## yaw_dumbbell           1.132231    83.4828254   FALSE FALSE
## total_accel_dumbbell   1.072634     0.2191418   FALSE FALSE
## gyros_dumbbell_x       1.003268     1.2282132   FALSE FALSE
## gyros_dumbbell_y       1.264957     1.4167771   FALSE FALSE
## gyros_dumbbell_z       1.060100     1.0498420   FALSE FALSE
## accel_dumbbell_x       1.018018     2.1659362   FALSE FALSE
## accel_dumbbell_y       1.053061     2.3748853   FALSE FALSE
## accel_dumbbell_z       1.133333     2.0894914   FALSE FALSE
## magnet_dumbbell_x      1.098266     5.7486495   FALSE FALSE
## magnet_dumbbell_y      1.197740     4.3012945   FALSE FALSE
## magnet_dumbbell_z      1.020833     3.4451126   FALSE FALSE
## roll_forearm          11.589286    11.0895933   FALSE FALSE
## pitch_forearm         65.983051    14.8557741   FALSE FALSE
## yaw_forearm           15.322835    10.1467740   FALSE FALSE
## total_accel_forearm    1.128928     0.3567424   FALSE FALSE
## gyros_forearm_x        1.059273     1.5187035   FALSE FALSE
## gyros_forearm_y        1.036554     3.7763735   FALSE FALSE
## gyros_forearm_z        1.122917     1.5645704   FALSE FALSE
## accel_forearm_x        1.126437     4.0464784   FALSE FALSE
## accel_forearm_y        1.059406     5.1116094   FALSE FALSE
## accel_forearm_z        1.006250     2.9558659   FALSE FALSE
## magnet_forearm_x       1.012346     7.7667924   FALSE FALSE
## magnet_forearm_y       1.246914     9.5403119   FALSE FALSE
## magnet_forearm_z       1.000000     8.5771073   FALSE FALSE
## classe                 1.469581     0.0254816   FALSE FALSE
```

```r
#dim(nzVar)
#head(nzVar, 60)
```

```
dfTrain <- dfTrain[, !nzVar$nzv]
dfTest <- dfTest[, !nzVar$nzv]
dim(dfTrain)
```

```
## [1] 19622    53
```

```
dfInTrain <- createDataPartition(dfTrain$classe, p = 0.75, list = FALSE)
dfPredict <- dfTrain[dfInTrain, ]
dfValidate <- dfTrain[-dfInTrain, ]
```

Model Fitting

- set.seed for pseudo-random number generation and ensure reproducible results
- A predictive model is fitted to predict the manner/class type in which an exerise was completed using Random Forest algorithm
- Random Forest algorithm is selected here because it is one of the most accurate learning algorithms available and produces highly accurate classifier for many datasets. It provides estimates of what variables are important in the classification and handles correlated covariates & outliers.
- A 5-fold cross validation (cv) resampling method is applied to the algorithm

- The results are predicted using the validation data set
- The results are compared using a confusionMatrix: a cross-tabulation of observed and predicted classes with associated statistics.
- The accuracy/overall agreement rate and Kappa are computed

```
set.seed(25)
fitControl <- trainControl(method='cv', number = 5)
modFitRf<- train(classe ~ ., data = dfPredict, method = "rf", trControl = fitControl)
#print(modFitRf)
modFitRf
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 11775, 11773, 11775, 11775, 11774
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9910315  0.9886536
##   27    0.9915748  0.9893420
##   52    0.9843724  0.9802286
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

```
predictRf <- predict(modFitRf, dfValidate)

confusionMatrix(dfValidate$classe, predictRf)
```

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

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    0    0    0    1
##          B   10  937    1    1    0
##          C    0    7  843    5    0
##          D    0    2   13  789    0
##          E    0    0    0    4  897
##
## Overall Statistics
##
##                Accuracy : 0.991
##                  95% CI : (0.988, 0.9935)
##     No Information Rate : 0.2863
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9886
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9929   0.9905   0.9837   0.9875   0.9989
## Specificity            0.9997   0.9970   0.9970   0.9963   0.9990
## Pos Pred Value         0.9993   0.9874   0.9860   0.9813   0.9956
## Neg Pred Value         0.9972   0.9977   0.9965   0.9976   0.9998
## Prevalence             0.2863   0.1929   0.1748   0.1629   0.1831
## Detection Rate         0.2843   0.1911   0.1719   0.1609   0.1829
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9963   0.9937   0.9903   0.9919   0.9989
```

```r
accuracy <- postResample(predictRf, dfValidate$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9910277 0.9886485
```

Conclusions & Test Data Set Prediction

- The Random Forest algoithm performed well with an accuracy of 0.995. The expected out-of-sample error rate is estimated at 0.005 (1 - accuracy).
- Therefore, the Random Forest predictive model is applied to the 20 test cases available in the test data set. We can expected that few of the test samples will be misclassified based on the accurate shown on the cross-validation data set.
- 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

```r
predictRf <- predict(modFitRf, dfTest)
predictRf
```

```
##  [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
```