Coursera\_PracticalMachineLearning\_Assignment

Sabitabrata Maity

May 13, 2018

## Background

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, your 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. More information is available from the website here: <http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har> (see the section on the Weight Lifting Exercise Dataset).

## Data

The training data for this project are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv>

The test data are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv>

The data for this project come from this source: <http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har>.

## Overview

From the HAR (Human Activity Recognition) project, we come to know that 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).

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg).

In this assignment, we were provided with the data from the subset of HAR data for building a predictive model and test the model.

## Loading the Libraries

We will be using the following libraries for this assignment.

library(caret)

## Warning: package 'caret' was built under R version 3.4.4

## Warning: package 'ggplot2' was built under R version 3.4.4

library(rpart)  
library(e1071)

## Warning: package 'e1071' was built under R version 3.4.4

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.4.4

## Loading the Data

For this assignment, we have a training data (pml-training.csv) to build our model and a validation data (pml-testing.csv) on which our model will be applied to predict the output. It has been observed that the data contains blank strings (“”) and some invalid numbers (“#DIV/0!”). We will consider those strins as NA. After loading the data, we will apply str function on the data to see the structures.

if (!file.exists("pml-training.csv")) {  
 download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv" ,"pml-training.csv",method="auto")  
}  
if (!file.exists("pml-testing.csv")) {  
 download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv" ,"pml-testing.csv",method="auto")  
}  
pml\_training <- read.csv("pml-training.csv", na.strings = c("#DIV/0!","","NA"))  
  
pml\_testing <- read.csv("pml-testing.csv", na.strings = c("#DIV/0!","","NA"))  
  
str(pml\_training)

## '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 1323084232 1323084232 1323084232 1323084232 ...  
## $ raw\_timestamp\_part\_2 : int 788290 808298 820366 120339 196328 304277 368296 440390 484323 484434 ...  
## $ 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]

## Cleaning the Data

From the structure, it seems that the serial number (X), user\_ \_name, timestanp (raw\_timestamp\_part\_1, raw\_timestamp\_part\_2, cvtd\_timestamp) and window (new\_window, num\_window) are the description fields of the data and they don’t influence the data.

data\_training <- subset(pml\_training, select = -c(X, user\_name, raw\_timestamp\_part\_1, raw\_timestamp\_part\_2, cvtd\_timestamp, new\_window, num\_window))

We can see from the data that few of the columns have NA value. Let us look into the data to find the proportion of NA values for the columns to the valid values. If the missing values for a column are completely at random or more than 95% of the data for a column is missing value, we will remove the coulmn.

pMiss <- function(x){sum(is.na(x))/length(x) > .95}  
missingVector <- apply(data\_training, 2, pMiss)  
missingVector

## roll\_belt pitch\_belt yaw\_belt   
## FALSE FALSE FALSE   
## total\_accel\_belt kurtosis\_roll\_belt kurtosis\_picth\_belt   
## FALSE TRUE TRUE   
## kurtosis\_yaw\_belt skewness\_roll\_belt skewness\_roll\_belt.1   
## TRUE TRUE TRUE   
## skewness\_yaw\_belt max\_roll\_belt max\_picth\_belt   
## TRUE TRUE TRUE   
## max\_yaw\_belt min\_roll\_belt min\_pitch\_belt   
## TRUE TRUE TRUE   
## min\_yaw\_belt amplitude\_roll\_belt amplitude\_pitch\_belt   
## TRUE TRUE TRUE   
## amplitude\_yaw\_belt var\_total\_accel\_belt avg\_roll\_belt   
## TRUE TRUE TRUE   
## stddev\_roll\_belt var\_roll\_belt avg\_pitch\_belt   
## TRUE TRUE TRUE   
## stddev\_pitch\_belt var\_pitch\_belt avg\_yaw\_belt   
## TRUE TRUE TRUE   
## stddev\_yaw\_belt var\_yaw\_belt gyros\_belt\_x   
## TRUE TRUE FALSE   
## gyros\_belt\_y gyros\_belt\_z accel\_belt\_x   
## FALSE FALSE FALSE   
## accel\_belt\_y accel\_belt\_z magnet\_belt\_x   
## FALSE FALSE FALSE   
## magnet\_belt\_y magnet\_belt\_z roll\_arm   
## FALSE FALSE FALSE   
## pitch\_arm yaw\_arm total\_accel\_arm   
## FALSE FALSE FALSE   
## var\_accel\_arm avg\_roll\_arm stddev\_roll\_arm   
## TRUE TRUE TRUE   
## var\_roll\_arm avg\_pitch\_arm stddev\_pitch\_arm   
## TRUE TRUE TRUE   
## var\_pitch\_arm avg\_yaw\_arm stddev\_yaw\_arm   
## TRUE TRUE TRUE   
## var\_yaw\_arm gyros\_arm\_x gyros\_arm\_y   
## TRUE FALSE FALSE   
## gyros\_arm\_z accel\_arm\_x accel\_arm\_y   
## FALSE FALSE FALSE   
## accel\_arm\_z magnet\_arm\_x magnet\_arm\_y   
## FALSE FALSE FALSE   
## magnet\_arm\_z kurtosis\_roll\_arm kurtosis\_picth\_arm   
## FALSE TRUE TRUE   
## kurtosis\_yaw\_arm skewness\_roll\_arm skewness\_pitch\_arm   
## TRUE TRUE TRUE   
## skewness\_yaw\_arm max\_roll\_arm max\_picth\_arm   
## TRUE TRUE TRUE   
## max\_yaw\_arm min\_roll\_arm min\_pitch\_arm   
## TRUE TRUE TRUE   
## min\_yaw\_arm amplitude\_roll\_arm amplitude\_pitch\_arm   
## TRUE TRUE TRUE   
## amplitude\_yaw\_arm roll\_dumbbell pitch\_dumbbell   
## TRUE FALSE FALSE   
## yaw\_dumbbell kurtosis\_roll\_dumbbell kurtosis\_picth\_dumbbell   
## FALSE TRUE TRUE   
## kurtosis\_yaw\_dumbbell skewness\_roll\_dumbbell skewness\_pitch\_dumbbell   
## TRUE TRUE TRUE   
## skewness\_yaw\_dumbbell max\_roll\_dumbbell max\_picth\_dumbbell   
## TRUE TRUE TRUE   
## max\_yaw\_dumbbell min\_roll\_dumbbell min\_pitch\_dumbbell   
## TRUE TRUE TRUE   
## min\_yaw\_dumbbell amplitude\_roll\_dumbbell amplitude\_pitch\_dumbbell   
## TRUE TRUE TRUE   
## amplitude\_yaw\_dumbbell total\_accel\_dumbbell var\_accel\_dumbbell   
## TRUE FALSE TRUE   
## avg\_roll\_dumbbell stddev\_roll\_dumbbell var\_roll\_dumbbell   
## TRUE TRUE TRUE   
## avg\_pitch\_dumbbell stddev\_pitch\_dumbbell var\_pitch\_dumbbell   
## TRUE TRUE TRUE   
## avg\_yaw\_dumbbell stddev\_yaw\_dumbbell var\_yaw\_dumbbell   
## TRUE TRUE TRUE   
## gyros\_dumbbell\_x gyros\_dumbbell\_y gyros\_dumbbell\_z   
## FALSE FALSE FALSE   
## accel\_dumbbell\_x accel\_dumbbell\_y accel\_dumbbell\_z   
## FALSE FALSE FALSE   
## magnet\_dumbbell\_x magnet\_dumbbell\_y magnet\_dumbbell\_z   
## FALSE FALSE FALSE   
## roll\_forearm pitch\_forearm yaw\_forearm   
## FALSE FALSE FALSE   
## kurtosis\_roll\_forearm kurtosis\_picth\_forearm kurtosis\_yaw\_forearm   
## TRUE TRUE TRUE   
## skewness\_roll\_forearm skewness\_pitch\_forearm skewness\_yaw\_forearm   
## TRUE TRUE TRUE   
## max\_roll\_forearm max\_picth\_forearm max\_yaw\_forearm   
## TRUE TRUE TRUE   
## min\_roll\_forearm min\_pitch\_forearm min\_yaw\_forearm   
## TRUE TRUE TRUE   
## amplitude\_roll\_forearm amplitude\_pitch\_forearm amplitude\_yaw\_forearm   
## TRUE TRUE TRUE   
## total\_accel\_forearm var\_accel\_forearm avg\_roll\_forearm   
## FALSE TRUE TRUE   
## stddev\_roll\_forearm var\_roll\_forearm avg\_pitch\_forearm   
## TRUE TRUE TRUE   
## stddev\_pitch\_forearm var\_pitch\_forearm avg\_yaw\_forearm   
## TRUE TRUE TRUE   
## stddev\_yaw\_forearm var\_yaw\_forearm gyros\_forearm\_x   
## TRUE TRUE FALSE   
## gyros\_forearm\_y gyros\_forearm\_z accel\_forearm\_x   
## FALSE FALSE FALSE   
## accel\_forearm\_y accel\_forearm\_z magnet\_forearm\_x   
## FALSE FALSE FALSE   
## magnet\_forearm\_y magnet\_forearm\_z classe   
## FALSE FALSE FALSE

Exclude the columns that contains more than 95% missing value.

data\_training <- data\_training[,names(missingVector[missingVector == FALSE])]

Now let us check, how many variables are there with near zero variance and if there is any we will investigate and exclude such columns.

sum(nearZeroVar(data\_training, saveMetrics=TRUE)$nzv)

## [1] 0

As the sum of all the near zero variables are zero, we don’t need to exclude any columns from the data set.

## Partitioning the Training Data

We will be pratitioning the training data into two parts. We will use 80% of the randomly partitioned data to build the model and rest of the data (20%) to test the model. We will set the seed 12345 to replicate the scenarios in future.

set.seed(12345)  
inTrain <- createDataPartition(data\_training$classe, p = 0.8, list = FALSE)  
dt\_train <- data\_training[inTrain,]  
dt\_test <- data\_training[-inTrain,]

## Decision Tree

Let us train the decision tree model and find out the accuracy of the model for the training data.

modelDecisionTree <- rpart(classe~., data=dt\_train)

Now we will test the model by applying it onto the testing data that was randomly partitioned from the original training data.

predictDecisionTree <- predict(modelDecisionTree, dt\_test, type="class")  
confusionMatrix(predictDecisionTree, dt\_test$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1003 165 9 77 12  
## B 21 390 72 23 50  
## C 36 64 542 101 92  
## D 33 58 45 408 36  
## E 23 82 16 34 531  
##   
## Overall Statistics  
##   
## Accuracy : 0.7326   
## 95% CI : (0.7185, 0.7464)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6604   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.8987 0.51383 0.7924 0.6345 0.7365  
## Specificity 0.9063 0.94753 0.9095 0.9476 0.9516  
## Pos Pred Value 0.7923 0.70144 0.6491 0.7034 0.7741  
## Neg Pred Value 0.9575 0.89041 0.9540 0.9297 0.9413  
## Prevalence 0.2845 0.19347 0.1744 0.1639 0.1838  
## Detection Rate 0.2557 0.09941 0.1382 0.1040 0.1354  
## Detection Prevalence 0.3227 0.14173 0.2128 0.1478 0.1749  
## Balanced Accuracy 0.9025 0.73068 0.8510 0.7910 0.8440

Accuracy of the decision tree model is given below.

confusionMatrix(predictDecisionTree, dt\_test$classe)$overall["Accuracy"]

## Accuracy   
## 0.7326026

The accuracy of the decision tree algorithm is very low. Let us try some other algorithm to fit the model.

## Random Forest

Let us train the decision tree model and find out the accuracy of the model for the training data.

modelRandomForest <- randomForest(classe~., data=dt\_train)

Now we will test the model by applying it onto the testing data that was randomly partitioned from the original training data.

predictRandomForest <- predict(modelRandomForest, dt\_test)  
confusionMatrix(predictRandomForest, dt\_test$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1116 3 0 0 0  
## B 0 756 10 0 0  
## C 0 0 674 10 0  
## D 0 0 0 633 1  
## E 0 0 0 0 720  
##   
## Overall Statistics  
##   
## Accuracy : 0.9939   
## 95% CI : (0.9909, 0.9961)  
## 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 1.0000 0.9960 0.9854 0.9844 0.9986  
## Specificity 0.9989 0.9968 0.9969 0.9997 1.0000  
## Pos Pred Value 0.9973 0.9869 0.9854 0.9984 1.0000  
## Neg Pred Value 1.0000 0.9990 0.9969 0.9970 0.9997  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2845 0.1927 0.1718 0.1614 0.1835  
## Detection Prevalence 0.2852 0.1953 0.1744 0.1616 0.1835  
## Balanced Accuracy 0.9995 0.9964 0.9911 0.9921 0.9993

Accuracy of the random forest model is given below.

confusionMatrix(predictRandomForest, dt\_test$classe)$overall["Accuracy"]

## Accuracy   
## 0.9938822

As the accuracy of the random forest model is very high, we will be using this model to predict the data from pml-testing.csv.

## Predict the data

Below is the predited output of the classe variable for pml-testing.csv.

predict(modelRandomForest, pml\_testing)

## 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