Peer-graded Assignment: Prediction Assignment Writeup

Amit Kumar

June 16, 2017

1. Overview:

This document is my submission report for the Peer Assessment project from Coursera course "Practical Machine Learning" as a part of the Specialization in Data Science. It is prepared in RStudio, using its knitr functions, meant to be published in .Rmd and and compiled HTML file.

The objective of the project is to predict the manner in which 6 participants performed the exercise. This is the "classe" variable in the training set. The machine learning algorithm (prediction model) is also used to predict 20 different test cases.

2. 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 have ake 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, my goal is 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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

3. Data

The training data for this project were available here:

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

The test data were available here:

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

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. They have been very generous in allowing their data to be used for this kind of assignments. Thanks.

3.1 Preparing Environment

I first upload the R libraries that are necessary for the complete analysis.

```
getwd()
```

```
## [1] "C:/Coursera/Machine Learning"
```

```r
setwd("C:/Coursera/Machine Learning")
library(caret);
```

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

```r
library(rpart);
library(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
```

```r
library(corrplot)
```

3.2 Loading Data

```r
url.train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
url.test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(url.train), na.strings = c("NA", "", "#DIV0!"))
testing <- read.csv(url(url.test), na.strings = c("NA", "", "#DIV0!"))
```

3.3 Cleaning Data

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

3.4 Checking the column names of traning dataset

```r
head(colnames(training))
```

```
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
```

The first 7 variables of the training data are irrelevant to the prediction and hence deleted.

```r
training <- training[,8:dim(training)[2]]
testing <- testing[,8:dim(testing)[2]]
```

3.5 Training, testing & validation data:

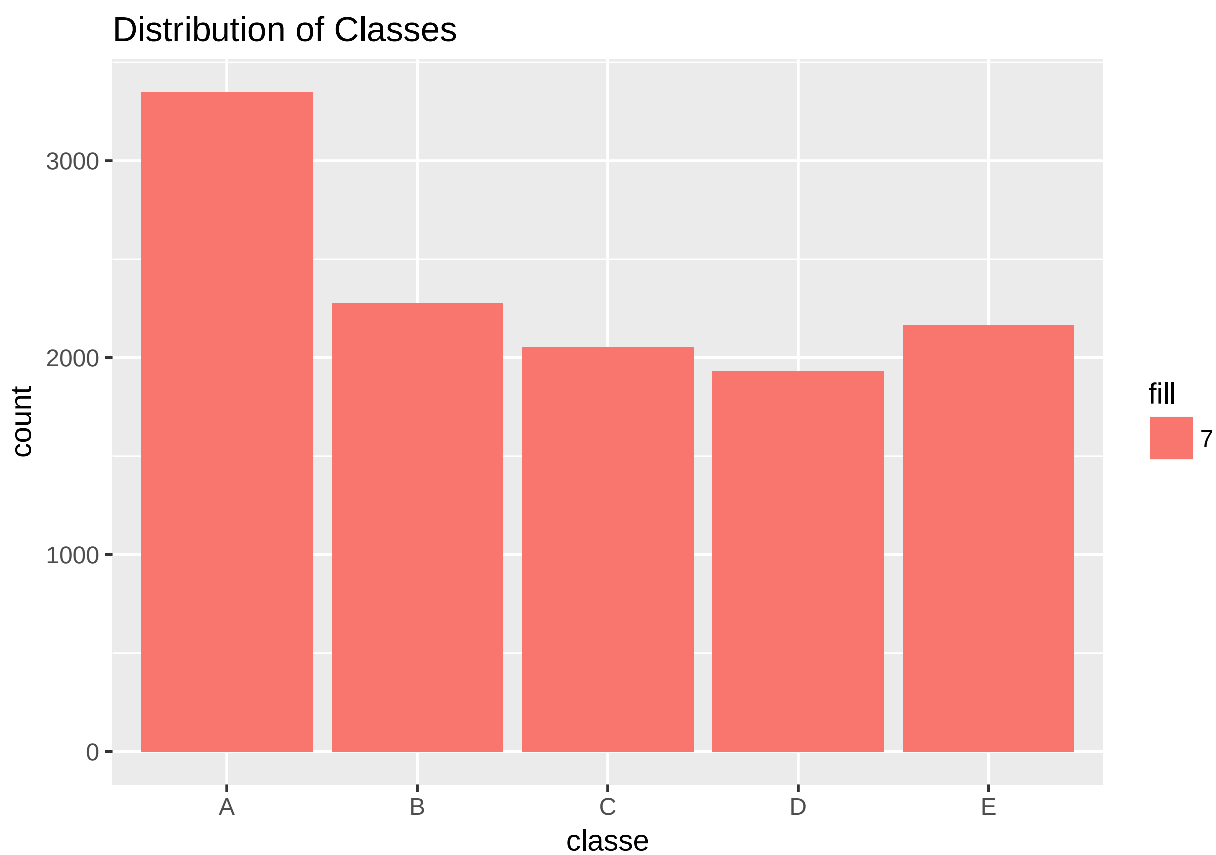
The training dataset is separated into three parts: tranining part (60%), testing part (20%), and validation part (20%)

```r
set.seed(123)
Seeddata1 <- createDataPartition(y = training$classe, p = 0.8, list = F)
Seeddata2 <- training[Seeddata1,]
validation <- training[-Seeddata1,]
Training_data1 <- createDataPartition(y = Seeddata2$classe, p = 0.75, list = F)
training_data2 <- Seeddata2[Training_data1,]
testing_data <- Seeddata2[-Training_data1,]
```

4.Exploratory Data Analysis:

4.1 Checking Distribution for Classes:

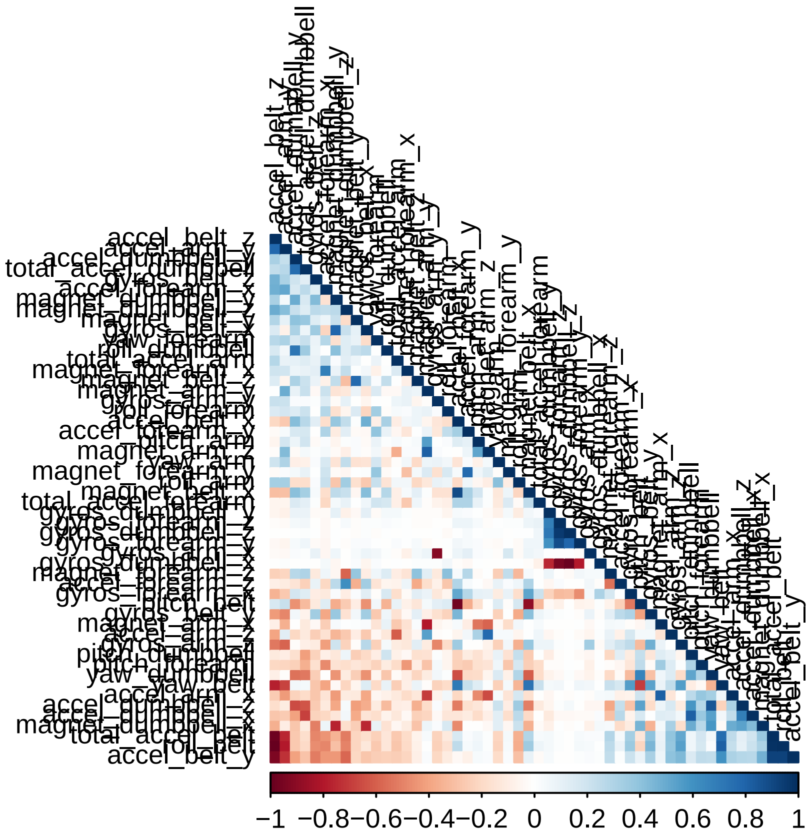
```r
qplot(classe, fill = "7", data=training_data2, main="Distribution of Classes")
```



4.2.Correlation Analysis:

A correlation among variables is analysed before proceeding to the modeling procedures.

```r
corMatrix <- cor(training_data2[, -53])
corrplot(corMatrix, order = "FPC", method = "color", type = "lower",
         tl.cex = 0.8, tl.col = rgb(0, 0, 0))
```



5. Model Building:

5.1 Finding out the predictors:

```
names(training_data2[,-53])
```

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

5.2 Prediction Model (Classification Tree model):

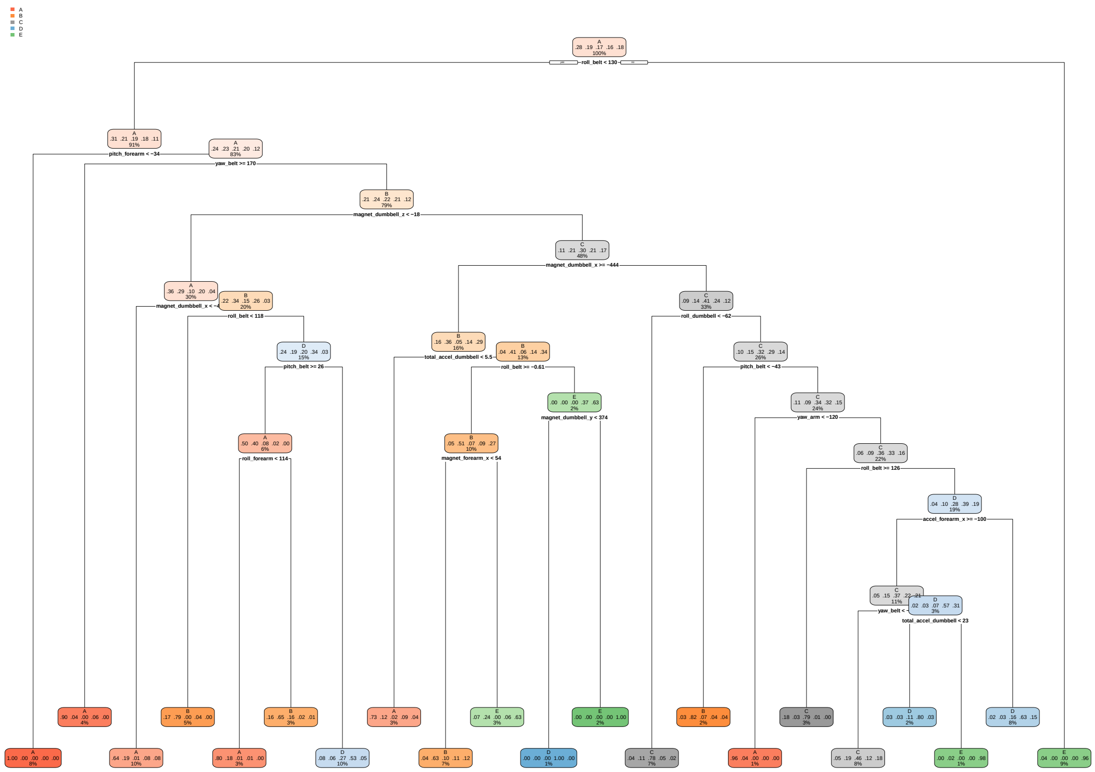
```
model_tree <- rpart(classe ~ ., data=training_data2, method="class")
prediction_tree <- predict(model_tree, testing_data, type="class")
class_tree <- confusionMatrix(prediction_tree, testing_data$classe)
class_tree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 944 111   7  53  42
##          B  53 492  51  59  53
##          C  49  98 447  54  51
##          D  47  33 178 468  70
##          E  23  25   1   9 505
##
## Overall Statistics
##
##                Accuracy : 0.728
##                  95% CI : (0.7138, 0.7419)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6559
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8459   0.6482   0.6535   0.7278   0.7004
## Specificity            0.9241   0.9317   0.9222   0.9000   0.9819
## Pos Pred Value         0.8159   0.6949   0.6395   0.5879   0.8970
## Neg Pred Value         0.9378   0.9170   0.9265   0.9440   0.9357
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2406   0.1254   0.1139   0.1193   0.1287
## Detection Prevalence   0.2949   0.1805   0.1782   0.2029   0.1435
## Balanced Accuracy      0.8850   0.7900   0.7879   0.8139   0.8412
```

5.3 Checking The Tree Model

```
library(rpart.plot)
rpart.plot(model_tree)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



5.4 Random Forest Model:

```
forest_model <- randomForest(classe ~ ., data=training_data2, method="class")
prediction_forest <- predict(forest_model, testing_data, type="class")
random_forest <- confusionMatrix(prediction_forest, testing_data$classe)
random_forest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    2    0    0    0
##          B    0  753    5    0    0
##          C    0    3  677    9    0
##          D    0    1    2  634    2
##          E    0    0    0    0  719
##
## 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   0.9921   0.9898   0.9860   0.9972
## Specificity            0.9993   0.9984   0.9963   0.9985   1.0000
## Pos Pred Value         0.9982   0.9934   0.9826   0.9922   1.0000
## Neg Pred Value         1.0000   0.9981   0.9978   0.9973   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1919   0.1726   0.1616   0.1833
## Detection Prevalence   0.2850   0.1932   0.1756   0.1629   0.1833
## Balanced Accuracy      0.9996   0.9953   0.9930   0.9922   0.9986
```

5.5 Final prediction

Prediction Algorithm and Confusion Matrix

```r
prediction1 <- predict(forest_model, newdata=testing_data)
confusionMatrix(prediction1, testing_data$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    2    0    0    0
##          B    0  753    5    0    0
##          C    0    3  677    9    0
##          D    0    1    2  634    2
##          E    0    0    0    0  719
##
## 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   0.9921   0.9898   0.9860   0.9972
## Specificity            0.9993   0.9984   0.9963   0.9985   1.0000
## Pos Pred Value         0.9982   0.9934   0.9826   0.9922   1.0000
## Neg Pred Value         1.0000   0.9981   0.9978   0.9973   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1919   0.1726   0.1616   0.1833
## Detection Prevalence   0.2850   0.1932   0.1756   0.1629   0.1833
## Balanced Accuracy      0.9996   0.9953   0.9930   0.9922   0.9986
```

The Random Forest is a much better predictive model than the Decision Tree, which has a larger accuracy

(99.91%).Therefore, we don't need to consider more important predictors for the Random Forest model.

6. Conclusions

In this study, the characteristics of predictors for both traning and testing datasets (train and test) are reduced. These characteristics are the percentage of NAs values, low variance, correlation and skewness. Therefore, the variables of the data sets are scaled. The training dataset is splitted into subtraining and validation parts to construct a predictive model and evaluate its accuracy. Decision Tree and Random Forest are applied.The Random Forest is a much better predictive model than the Decision Tree, which has a larger accuracy (99.91%).

7. Reproducibility:

This project is reproducible and was done with the following environment:

```
sessionInfo()
```

```
## R version 3.3.3 (2017-03-06)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] rpart.plot_2.1.2    corrplot_0.77       randomForest_4.6-12
## [4] rpart_4.1-11        caret_6.0-76        ggplot2_2.2.1
## [7] lattice_0.20-35
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.10       nloptr_1.0.4       plyr_1.8.4
##  [4] class_7.3-14       iterators_1.0.8    tools_3.3.3
##  [7] digest_0.6.12      lme4_1.1-13        evaluate_0.10
## [10] tibble_1.3.0       nlme_3.1-131       gtable_0.2.0
## [13] mgcv_1.8-17        Matrix_1.2-10      foreach_1.4.3
## [16] yaml_2.1.14        parallel_3.3.3     SparseM_1.77
## [19] e1071_1.6-8        stringr_1.2.0      knitr_1.15.1
## [22] MatrixModels_0.4-1 stats4_3.3.3       rprojroot_1.2
## [25] grid_3.3.3         nnet_7.3-12        rmarkdown_1.5
## [28] minqa_1.2.4        reshape2_1.4.2     car_2.1-4
## [31] magrittr_1.5       backports_1.0.5    scales_0.4.1
## [34] codetools_0.2-15   ModelMetrics_1.1.0 htmltools_0.3.6
## [37] MASS_7.3-47        splines_3.3.3      pbkrtest_0.4-7
## [40] colorspace_1.3-2   labeling_0.3       quantreg_5.33
## [43] stringi_1.1.5      lazyeval_0.2.0     munsell_0.4.3
```