Peer-graded Assignment: Prediction Assignment Writeup

hadeer mahmoud

8 october 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://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

```
library(lattice);library(ggplot2)

library(caret);library(rpart)

library(randomForest)
```

```
## 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
```

Loading Data

CHeck the training and testing data, identifying the missing data, “NA” and “#DIV” as “NA” everywhere.

```
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!"))
```

Cleaning data

We should delete the column that contains NA to avoid the error. In addition, in order to make accurate predictions, columns that is not related exercise must also be deleted.

```
colname <- colnames(training)[!colSums(is.na(training)) > 0]
colname

##  [1] "X"                    "user_name"            "raw_timestamp_part_1"
##  [4] "raw_timestamp_part_2" "cvtd_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<-training[,colSums(is.na(training)) == 0]
testing <-testing[,colSums(is.na(testing)) == 0]
```

We need to define the same columns

```
sameColumsName <- colnames(training) == colnames(testing)
colnames(training)[sameColumsName==FALSE]

## [1] "classe"
```

Therefore,the “classe” is not included in the testing data.

Checking the column names of traning dataset

```
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 were deleted, because they are irrelevant to the prediction

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

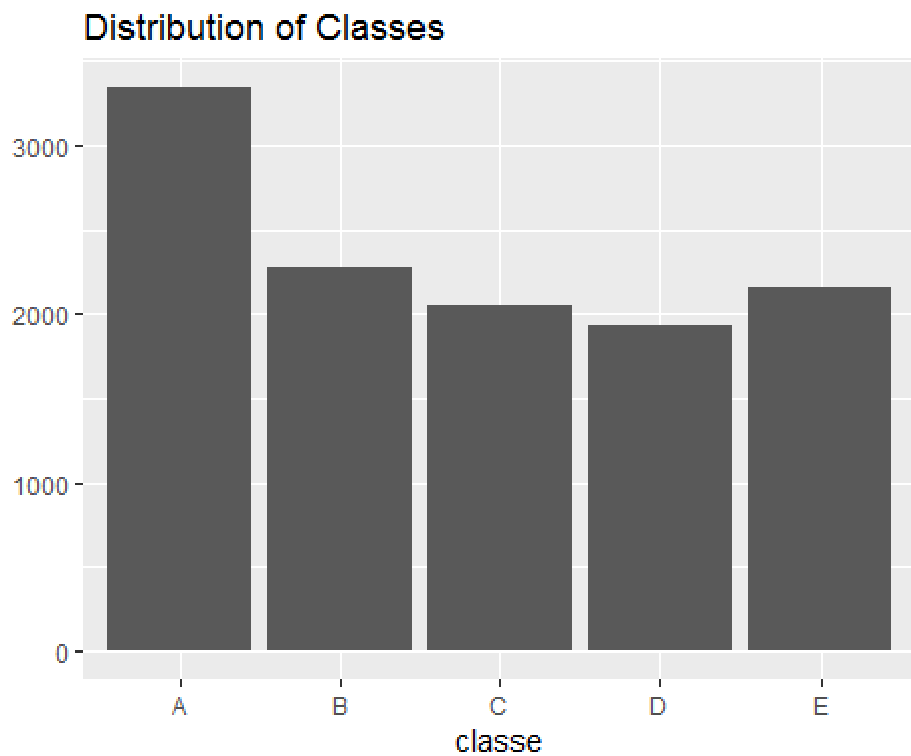
Training, testing & validation data

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

```
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,]
```

Data exploration

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



Findout the predictors

```
names(training_data2[,-50])
```

```
##  [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_y"     "magnet_forearm_z"    
## [52] "classe"
```

Prediction model

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

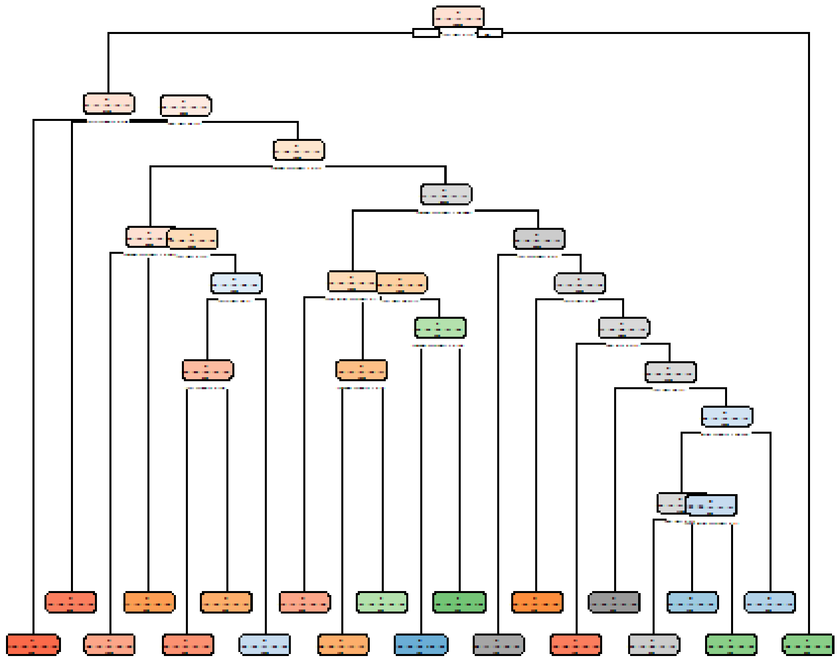
```r
class_tree <- confusionMatrix(prediction_tree, testing_data$classe)
class_tree

## 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
```

Checking the model_tree

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

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



##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   10    0
##          D    0    1    2  633    2
##          E    0    0    0    0  719
##
## Overall Statistics
##
##                Accuracy : 0.9936
##                  95% CI : (0.9906, 0.9959)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9919
##  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   0.9844   0.9972
## Specificity            0.9993   0.9984   0.9960   0.9985   1.0000
## Pos Pred Value         0.9982   0.9934   0.9812   0.9922   1.0000
## Neg Pred Value         1.0000   0.9981   0.9978   0.9970   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1919   0.1726   0.1614   0.1833
## Detection Prevalence   0.2850   0.1932   0.1759   0.1626   0.1833
## Balanced Accuracy      0.9996   0.9953   0.9929   0.9915   0.9986
```

Final prediction

Prediction Algorithm and Confusion Matrix

```
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   10    0
##          D    0    1    2  633    2
##          E    0    0    0    0  719
##
## Overall Statistics
##
##                Accuracy : 0.9936
##                  95% CI : (0.9906, 0.9959)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9919
##  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   0.9844   0.9972
## Specificity            0.9993   0.9984   0.9960   0.9985   1.0000
## Pos Pred Value         0.9982   0.9934   0.9812   0.9922   1.0000
## Neg Pred Value         1.0000   0.9981   0.9978   0.9970   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1919   0.1726   0.1614   0.1833
## Detection Prevalence   0.2850   0.1932   0.1759   0.1626   0.1833
## Balanced Accuracy      0.9996   0.9953   0.9929   0.9915   0.9986
```

The Random Forest is a much better predictive model than the Decision Tree, which has a larger accuracy (99.91%)

Conclusions

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%).