Peer-graded Assignment: Prediction Assignment Writeup

Bing Hu

March 17, 2021

Overview

This is the report generated for course project of Practical Machine Learning course from John Hopkins University on Coursera as part of Data Science specification. The goal of your project is to predict the manner in which they did the exercise, which is the "classe" variable in the training set. Report will include description of the problem, dataset description, variables used to build model for prediction and prediction results of applying the model.

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 (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Data Source

The training and test data for this project are collected using the link below:
https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)
https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
(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 (http://groupware.les.inf.puc-rio.br/har). The full reference of this data is as follows: Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. "Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13)". Stuttgart, Germany: ACM SIGCHI, 2013.

Setup environment

```
setwd("C:/Users/bingh/Documents/Coursera/JHU_Course8_Practical Machine Learning/CourseProject")
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.4
```

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

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

```
## Warning: package 'ggplot2' was built under R version 4.0.4
```

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

```
## Warning: package 'rpart.plot' was built under R version 4.0.4
```

```
set.seed(1234)
```

Download in dataset

```
url_train <- "pml-training.csv"
rawtraining <- read.csv(url_train, na.strings = c("", "NA"))
url_test <- "pml-testing.csv"
rawtesting <- read.csv(url_test, na.strings = c("", "NA"))
dim(rawtraining)
```

```
## [1] 19622   160
```

```
dim(rawtesting)
```

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

Clean Data

To avoid error, data cleaning process to remove NA and near-zero-variance (NZV) variables is needed.

```
nzv_var <- nearZeroVar(rawtraining)
rawtraining <- rawtraining[ , -nzv_var]
dim(rawtraining)
```

```
## [1] 19622   117
```

```
#Remove NA cols
colname <- colnames(rawtraining)[!colSums(is.na(rawtraining)) > 0]
colname
```

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

```
#Slice data relatd with exercise
colname <- colname[8: length(colname)]
training0 <- rawtraining[colname]
```

Split data

create a partition using caret with the training dataset on 70,30 ratio

```
inTrain  <- createDataPartition(training0$classe, p=0.7, list=FALSE)
training <- training0[inTrain, ]
testing  <- training0[-inTrain, ]
dim(training)
```

```
## [1] 13737    52
```

```
dim(testing)
```

```
## [1] 5885   52
```

Apply different Prediction model

create a partition using caret with the training dataset on 70,30 ratio

```
inTrain  <- createDataPartition(training0$classe, p=0.7, list=FALSE)
training <- training0[inTrain, ]
testing  <- training0[-inTrain, ]
dim(training)
```

```
## [1] 13737    52
```

```
dim(testing)
```

```
## [1] 5885   52
```

```
colnames(training)
```

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

Model 1: Linear Discriminant Analysis

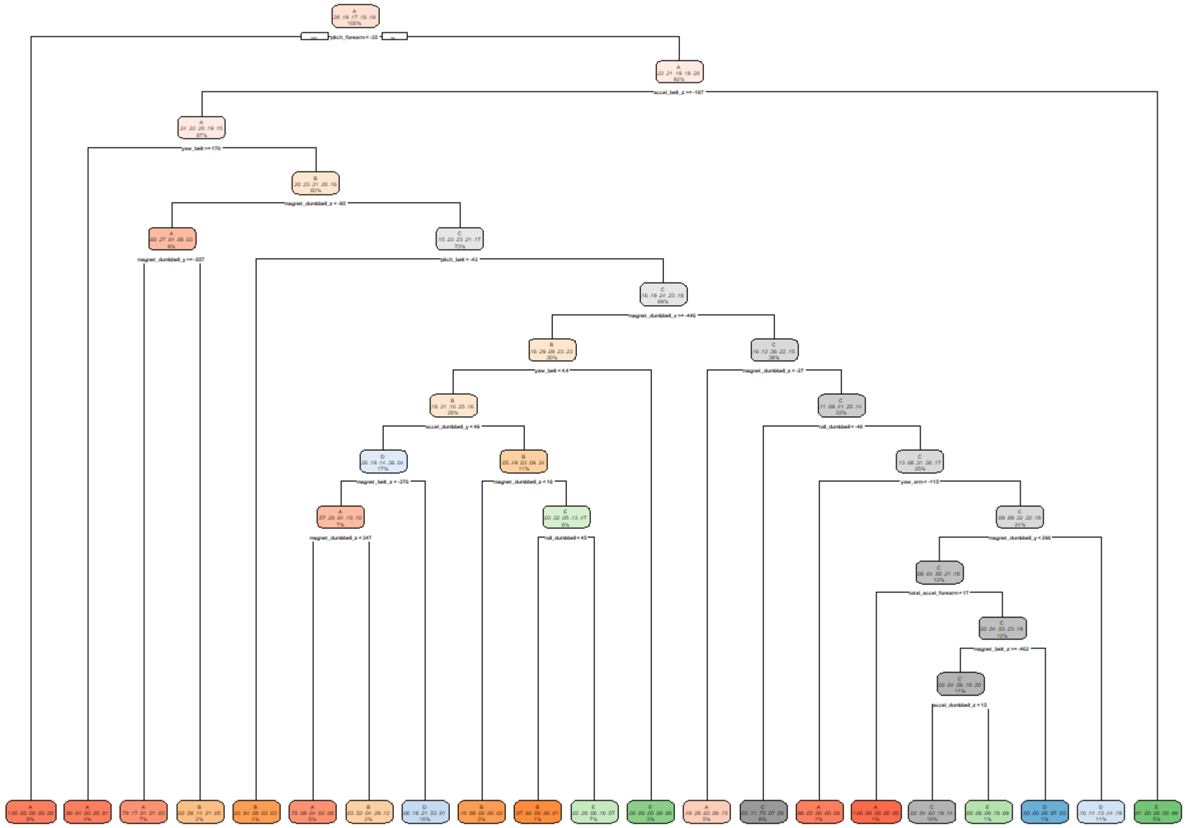
```
model_lda <- train(classe ~ ., data = training, method = "lda")
pred_lda <- predict(model_lda, testing)
confusionMatrix_lda <- confusionMatrix(pred_lda, factor(testing$classe))
confusionMatrix_lda
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1349  172  100   38   52
##          B   36  732  108   57  185
##          C  137  137  668  118  110
##          D  148   46  128  694  143
##          E    4   52   22   57  592
## 
## Overall Statistics
## 
##                Accuracy : 0.6856
##                  95% CI : (0.6736, 0.6975)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.6025
## 
##  Mcnemar's Test P-Value : < 2.2e-16
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8059   0.6427   0.6511   0.7199   0.5471
## Specificity            0.9140   0.9187   0.8967   0.9055   0.9719
## Pos Pred Value         0.7884   0.6547   0.5709   0.5988   0.8143
## Neg Pred Value         0.9221   0.9146   0.9241   0.9429   0.9050
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2292   0.1244   0.1135   0.1179   0.1006
## Detection Prevalence   0.2907   0.1900   0.1988   0.1969   0.1235
## Balanced Accuracy      0.8599   0.7807   0.7739   0.8127   0.7595
```

The The predictive accuracy of the linear discriminant model is 0.6856

Model 2: Dicision Tree Model

```
set.seed(1234)
pred_dtree <- rpart(classe ~ ., data = training, method="class")
rpart.plot(pred_dtree)
```



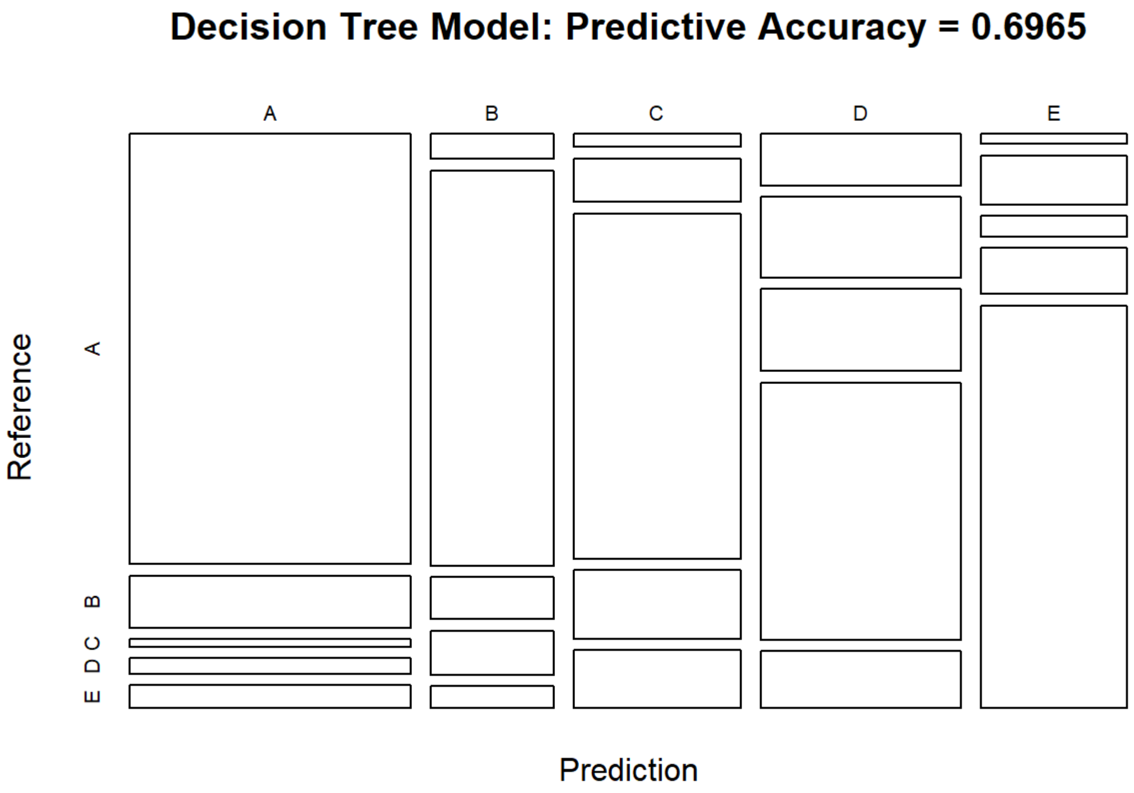
Predictions of the decision tree model on test data

```
pred_decision_tree <- predict(pred_dtree, newdata = testing, type="class")
confmatrix_dt <- confusionMatrix(pred_decision_tree, factor(testing$classe))
confmatrix_dt
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1468  177   25   53   79
##          B   37  591   63   66   33
##          C   26   89  702  139  118
##          D  125  196  199  625  139
##          E   18   86   37   81  713
##
## Overall Statistics
##
##                Accuracy : 0.6965
##                  95% CI : (0.6846, 0.7082)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6159
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8769   0.5189   0.6842   0.6483   0.6590
## Specificity            0.9207   0.9581   0.9234   0.8661   0.9538
## Pos Pred Value         0.8147   0.7481   0.6536   0.4868   0.7626
## Neg Pred Value         0.9495   0.8924   0.9327   0.9263   0.9255
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2494   0.1004   0.1193   0.1062   0.1212
## Detection Prevalence   0.3062   0.1342   0.1825   0.2182   0.1589
## Balanced Accuracy      0.8988   0.7385   0.8038   0.7572   0.8064
```

Plot the predictive accuracy of the decision tree model.

```
plot(confmatrix_dt$table, col = confmatrix_dt$byClass,
     main = paste("Decision Tree Model: Predictive Accuracy =",
                  round(confmatrix_dt$overall['Accuracy'], 4)))
```


Decision Tree Model: Predictive Accuracy = 0.6965

The The predictive accuracy of the linear discriminant model is 0.6965

Model 3: Radom Tree Model

```
model_rf <- train(classe ~ ., data = training, method = "rf")
model_rf$finalModel
```

```
## 
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry) 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 26
## 
##         OOB estimate of  error rate: 0.74%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3900    3    1    0    2 0.001536098
## B   22 2628    7    0    1 0.011286682
## C    0   16 2372    7    1 0.010016694
## D    0    0   29 2220    3 0.014209591
## E    0    0    3    6 2516 0.003564356
```

```
pred_rf <- predict(model_rf, newdata=testing)
confmatrix_rf <- confusionMatrix(pred_rf, factor(testing$classe))
confmatrix_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1671   12    0    0    0
##          B    3 1126   10    0    0
##          C    0    1 1014   10    2
##          D    0    0    2  953    1
##          E    0    0    0    1 1079
##
## Overall Statistics
##
##                Accuracy : 0.9929
##                  95% CI : (0.9904, 0.9949)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.991
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9886   0.9883   0.9886   0.9972
## Specificity            0.9972   0.9973   0.9973   0.9994   0.9998
## Pos Pred Value         0.9929   0.9886   0.9873   0.9969   0.9991
## Neg Pred Value         0.9993   0.9973   0.9975   0.9978   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2839   0.1913   0.1723   0.1619   0.1833
## Detection Prevalence   0.2860   0.1935   0.1745   0.1624   0.1835
## Balanced Accuracy      0.9977   0.9929   0.9928   0.9940   0.9985
```

The Random Forest model is selected and applied to make predictions on the 20 data points from the original testing dataset

```
predict_test <- as.data.frame(predict(model_rf, newdata = rawtesting))
predict_test
```

```
##    predict(model_rf, newdata = rawtesting)
## 1                                         B
## 2                                         A
## 3                                         B
## 4                                         A
## 5                                         A
## 6                                         E
## 7                                         D
## 8                                         B
## 9                                         A
## 10                                        A
## 11                                        B
## 12                                        C
## 13                                        B
## 14                                        A
## 15                                        E
## 16                                        E
## 17                                        A
## 18                                        B
## 19                                        B
## 20                                        B
```