Prediction Assignment Write Up

John Mastapeter

11/18/2020

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

Prelimary Steps

In order to preform any kind of predictive analysis, the necessary R libraies must be loaded and the online data downloaded and scraped of uncessary data prior to the making the calculations.

Libraries

```
#Please download the following libraries to complete the code;
library(RCurl)
library(knitr)
library(caret)
```

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

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

```
library(rattle)
```

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(rpart)
library(rpart.plot)
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:rattle':
##
##     importance
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

Set Working Directory and Download Data

Set pre-established working directory to save data.

```
#Set working directory
setwd("C:/Users/mastapeterj/Documents/Coursera_DataScience/PredictionAssignmentWriteUp")
#download files
#pml testing
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile = "pmltra
#pml training
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfile = "pmltes
#Read data into R
trainingpml <- read.csv("pmltraining.csv")
testingpml <- read.csv("pmltesting.csv")
#Review Data
training_head <-head(trainingpml)
```

```
testing_head <-head(testingpml)
#If interested or necessary, print review of data
#training_head
#testing_head
```

##Cleaning the Data

```
#set seed
set.seed(12345)
#Set up initial training partition to use during analysis
inTrain <- createDataPartition(trainingpml$classe, p = 0.7, list = FALSE)
train_set <- trainingpml[inTrain, ]
test_set <- trainingpml[-inTrain, ]
dim(train_set)
```

```
## [1] 13737   160
```

```
dim(test_set)
```

```
## [1] 5885  160
```

```
#Remove all Near Zero Values whicch will not affect the analysis
nzvs <- nearZeroVar(train_set)
train_set <- train_set[, -nzvs]
test_set <- test_set[, -nzvs]
dim(train_set)
```

```
## [1] 13737   104
```

```
dim(test_set)
```

```
## [1] 5885  104
```

```
#Remoove all NA values that will not affect the analysis
allna <- sapply(train_set, function(x) mean(is.na(x))) > 0.95
train_set <- train_set[, allna==FALSE]
test_set <- test_set[, allna==FALSE]
dim(train_set)
```

```
## [1] 13737    59
```

```
dim(test_set)
```

```
## [1] 5885   59
```

```
#Remove the first through fifth columns, as they add no valuable information for analysis
train_set <- train_set[, -(1:5)]
test_set <- test_set[, -(1:5)]
dim(train_set)
```

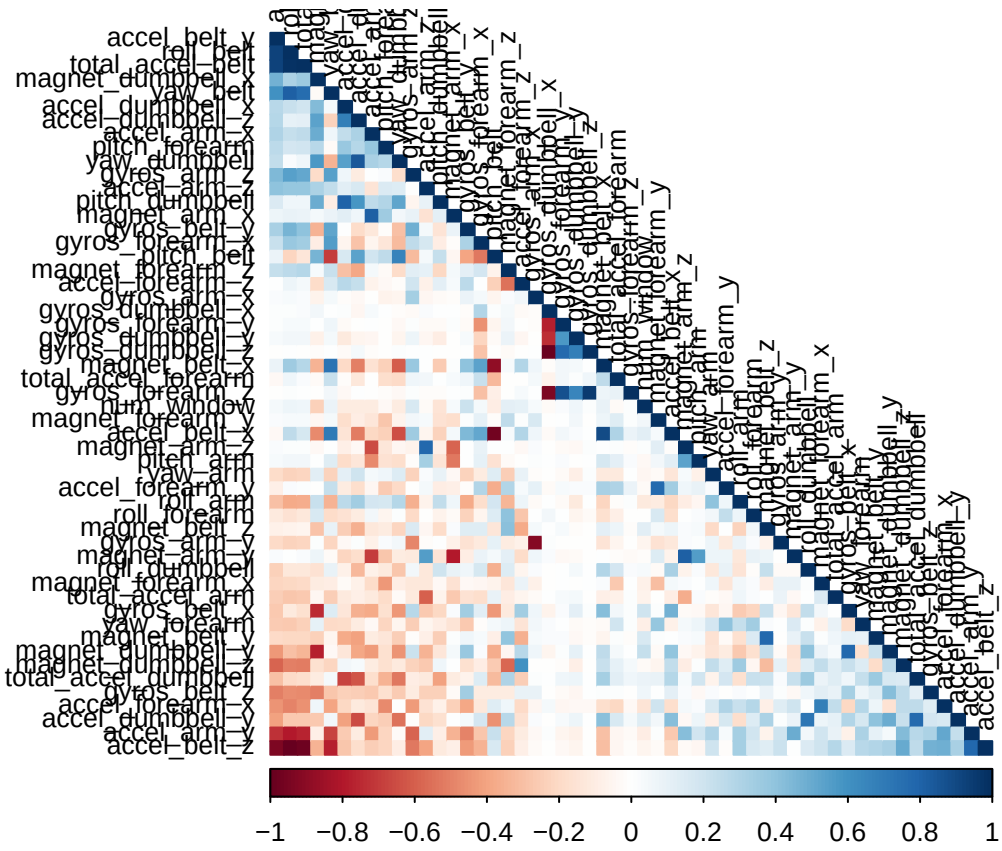
```
## [1] 13737    54
```

```
dim(test_set)
```

```
## [1] 5885   54
```

##Determine Correlation through Fixed Point Clustering

```
cor_Matrix <- cor(train_set[,-54])
corrplot(cor_Matrix, order = "FPC", method = "color", type = "lower", tl.cex = 0.8, tl.col=rgb(0 ,0 ,0))
```



##Determine correlation through Random Forests

```
set.seed(12345)
control_rf <- trainControl(method = "cv", number = 3, verboseIter = FALSE)
modfit_randforest <- train(classe ~., data=train_set, method = "rf", trControl = control_rf)
modfit_randforest$finalModel
```

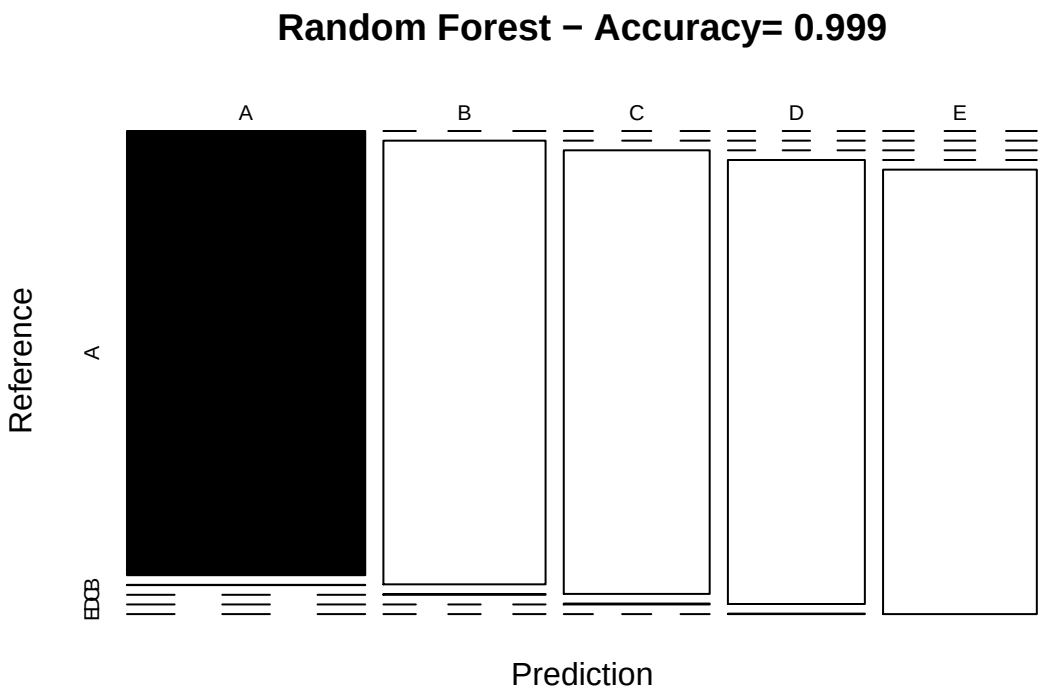
```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.23%
```

```
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    2    0    0    0 0.0005120328
## B    6 2647    4    1    0 0.0041384500
## C    0    5 2391    0    0 0.0020868114
## D    0    0    9 2243    0 0.0039964476
## E    0    0    0    5 2520 0.0019801980
```

```
predict_randforest <- predict(modfit_randforest, newdata = test_set)
confmat_randforest <- confusionMatrix(predict_randforest, test_set$classe)
confmat_randforest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1138    2    0    0
##          C    0    0 1024    2    0
##          D    0    0    0  962    1
##          E    0    0    0    0 1081
##
## Overall Statistics
##
##                Accuracy : 0.999
##                  95% CI : (0.9978, 0.9996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9987
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9991   0.9981   0.9979   0.9991
## Specificity            0.9998   0.9996   0.9996   0.9998   1.0000
## Pos Pred Value         0.9994   0.9982   0.9981   0.9990   1.0000
## Neg Pred Value         1.0000   0.9998   0.9996   0.9996   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1934   0.1740   0.1635   0.1837
## Detection Prevalence   0.2846   0.1937   0.1743   0.1636   0.1837
## Balanced Accuracy      0.9999   0.9994   0.9988   0.9989   0.9995
```

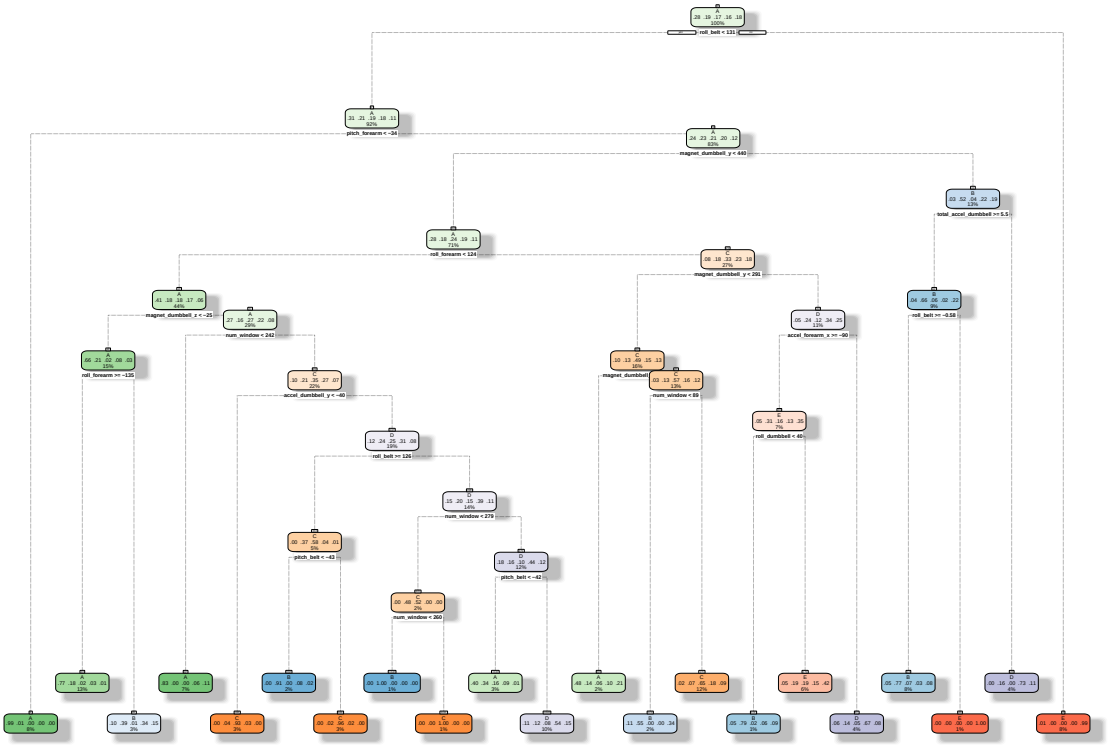
```
plot(confmat_randforest$table, col= confmat_randforest$byClass, main = paste("Random Forest - Accuracy=
```



##Determine correlation through Decision Tree

```
set.seed(12345)
modfit_dectree <- rpart(classe~., data = train_set, method = "class")
fancyRpartPlot((modfit_dectree))
```

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



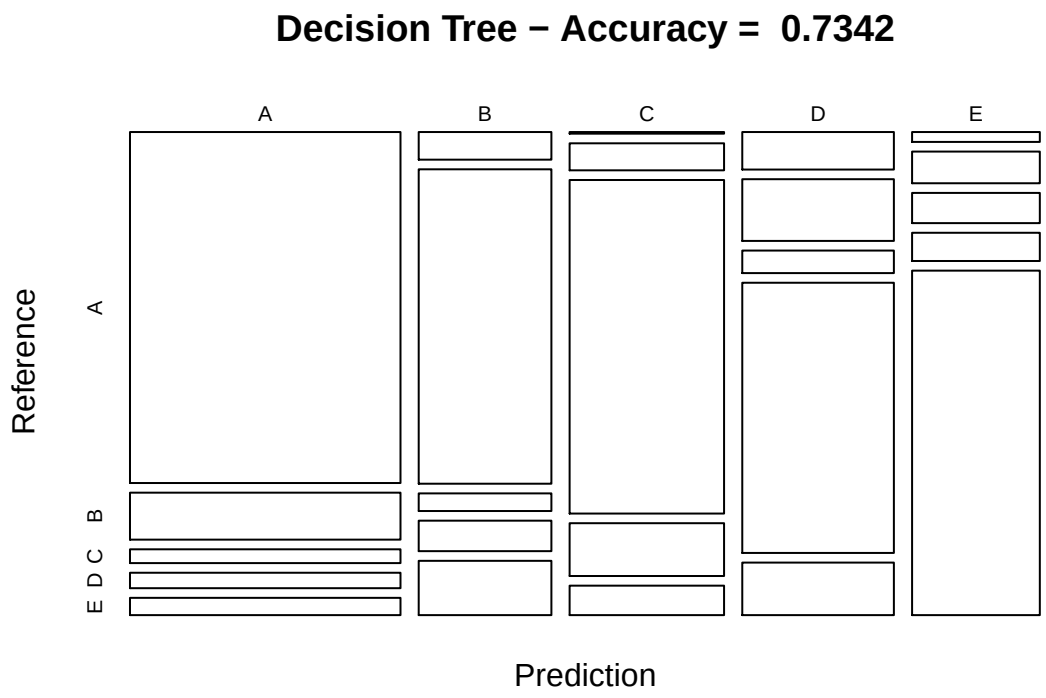
Rattle 2020−Nov−19 06:48:07 mastapeterj

```
predict_dectree <-predict(modfit_dectree, newdata = test_set, type = "class")
confmat_dectree <- confusionMatrix(predict_dectree, test_set$classe)
confmat_dectree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1502  201   59   66   74
##          B   58  660   37   64  114
##          C    4   66  815  129   72
##          D   90  148   54  648  126
##          E   20   64   61   57  696
##
## Overall Statistics
##
##                Accuracy : 0.7342
##                  95% CI : (0.7228, 0.7455)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6625
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8973   0.5795   0.7943   0.6722   0.6433
## Specificity            0.9050   0.9425   0.9442   0.9151   0.9579
## Pos Pred Value         0.7897   0.7074   0.7505   0.6079   0.7751
## Neg Pred Value         0.9568   0.9033   0.9560   0.9344   0.9226
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2552   0.1121   0.1385   0.1101   0.1183
## Detection Prevalence   0.3232   0.1585   0.1845   0.1811   0.1526
## Balanced Accuracy      0.9011   0.7610   0.8693   0.7936   0.8006
```

```
plot(confmat_dectree$table, col = confmat_dectree$byClass, main = paste("Decision Tree - Accuracy = ", r
```



##Determine correlation through Generalilzed Boosted Model

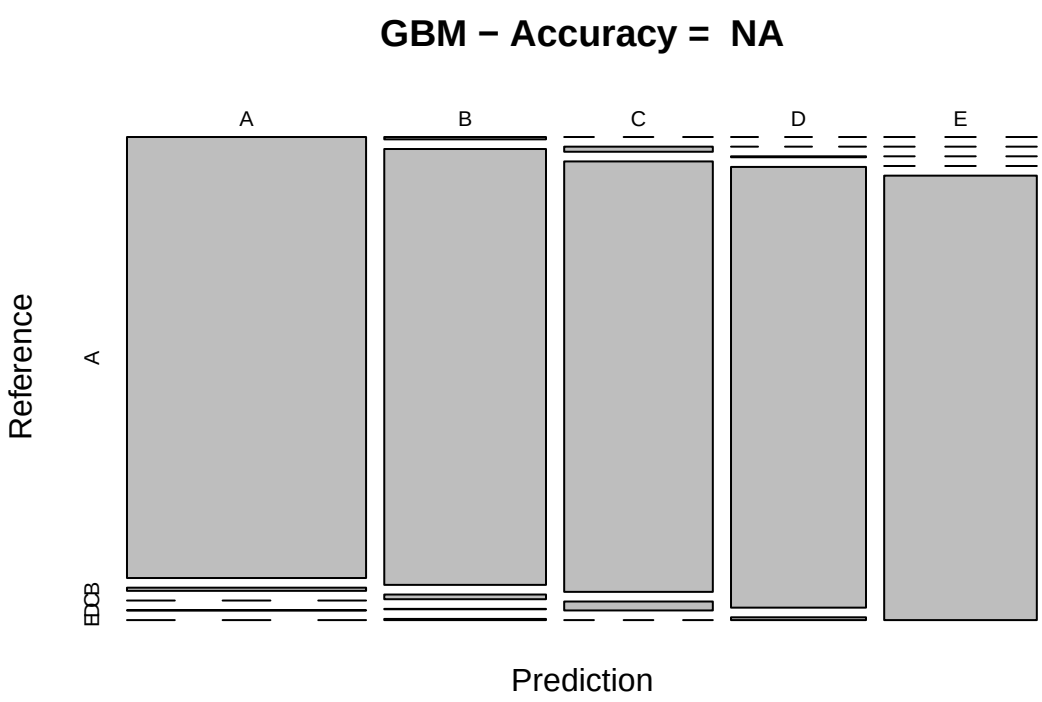
```
set.seed(12345)
control_gbm <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modfit_gbm <- train(classe~., data = train_set, method = "gbm", trControl = control_gbm, verbose = FALSE
modfit_gbm$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 53 had non-zero influence.
```

```r
predict_gbm <- predict(modfit_gbm, newdata = test_set)
confmat_gbm <- confusionMatrix(predict_gbm, test_set$classe)
confmat_gbm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1668   12    0    1    0
##          B    6 1115   12    1    3
##          C    0   12 1012   21    0
##          D    0    0    2  941    6
##          E    0    0    0    0 1073
##
## Overall Statistics
##
##                Accuracy : 0.9871
##                  95% CI : (0.9839, 0.9898)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9837
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9964   0.9789   0.9864   0.9761   0.9917
## Specificity            0.9969   0.9954   0.9932   0.9984   1.0000
## Pos Pred Value         0.9923   0.9807   0.9684   0.9916   1.0000
## Neg Pred Value         0.9986   0.9949   0.9971   0.9953   0.9981
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2834   0.1895   0.1720   0.1599   0.1823
## Detection Prevalence   0.2856   0.1932   0.1776   0.1613   0.1823
## Balanced Accuracy      0.9967   0.9871   0.9898   0.9873   0.9958
```

```r
plot(confmat_gbm$table, confmat_gbm$byClass, main = paste("GBM - Accuracy = ", round(confmat_gbm$overall
```


GBM − Accuracy = NA
A
B
C
D
E
Reference
A
B
C
D
E
Prediction