Analysis of Exercise Habits

Chris Lorenzini

February 22, 2018

Introduction

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 (http://groupware.les.inf.puc-rio.br/har).

Loading and Cleaning the Data

First thing to do is to load in all of the appropriate packages. Once we do this, we pull the data from the appropriate sources into a test and trainging data set:

```
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(RGtk2)
library(rattle)
library(randomForest)
library(e1071)

trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

train <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))
test <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))

Train <- createDataPartition(y=train$classe, p=0.6, list=FALSE)
Train1 <- train[Train, ]; Test1 <- train[-Train, ]
```

Now that we have pulled in the data and split our traning data into two sets, we need to clean up the data a bit. First, we will remove the NearZeroVariance variables, and remove variables that are mostly filled with NA results:

```
myDataNZV <- nearZeroVar(Train1, saveMetrics=TRUE)
myNZVvars <- names(Train1)
myTrain <- Train1[!myNZVvars]
myTrain <- myTrain[c(-1)]
training1 <- myTrain #creating another subset to iterate in loop
for(i in 1:length(myTrain)) { #for every column in the training dataset
  if( sum( is.na( myTrain[, i] ) ) /nrow(myTrain) >= .6 ) { #if n?? NAs > 60% of tota
l observations
    for(j in 1:length(training1)) {
      if( length( grep(names(myTrain[i]), names(training1)[j]) ) ==1)  { #if the colum
ns are the same:
        training1 <- training1[ , -j] #Remove that column
      }
    }
  }
}
```

Next, we will transform the datasets and get the data into the same type:

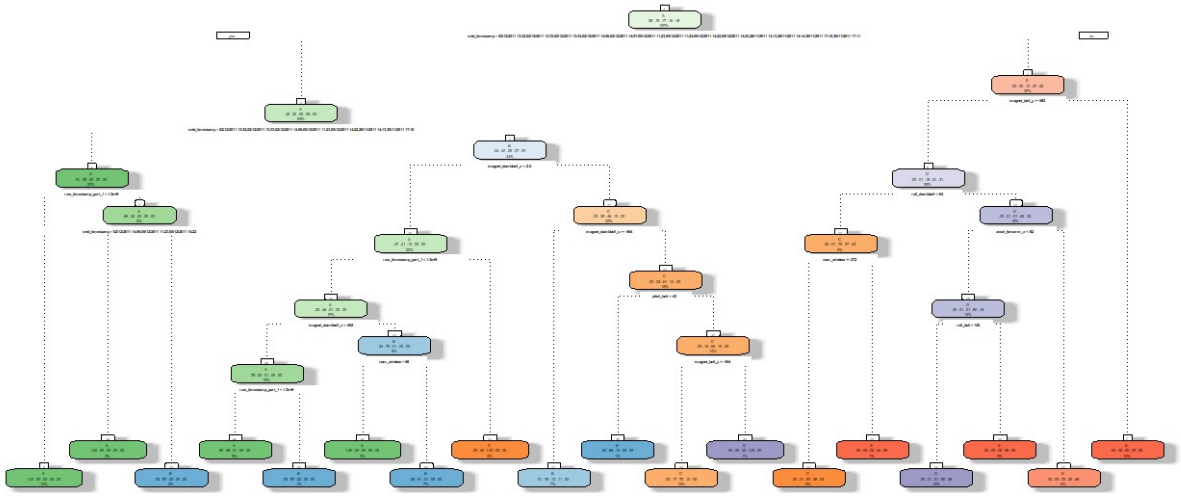
```
c1 <- colnames(myTrain)
c2 <- colnames(myTrain[, -59])
myTest <- Test1[c1]
test <- test[c2]

for (i in 1:length(test) ) {
  for(j in 1:length(myTrain)) {
    if( length( grep(names(myTrain[i]), names(test)[j]) ) ==1)  {
      class(test[j]) <- class(myTrain[i])
    }
  }
}
test <- rbind(myTrain[2, -59] , test)
test <- test[-1,]
```

Decision Tree Analysis

In order to see which exercise was used, a decision tree will be constructed.

```
Fit <- rpart(classe ~ ., data=myTrain, method="class")
fancyRpartPlot(Fit)
```



Rattle 2018-Feb-22 17:31:16 Christopher Lorenzin

Decision Tree

```
pred <- predict(Fit, myTest, type = "class")
confusionMatrix(pred, myTest$classe)

Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 2159   79    7    3    0
         B   56 1252   74   67    0
         C   17  177 1262  139   13
         D    0   10   14  838   81
         E    0    0   11  239 1348

Overall Statistics

               Accuracy : 0.8742
                 95% CI : (0.8667, 0.8815)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8407
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9673   0.8248   0.9225   0.6516   0.9348
Specificity            0.9841   0.9689   0.9466   0.9840   0.9610
Pos Pred Value         0.9604   0.8640   0.7848   0.8887   0.8436
Neg Pred Value         0.9870   0.9584   0.9830   0.9351   0.9850
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2752   0.1596   0.1608   0.1068   0.1718
Detection Prevalence   0.2865   0.1847   0.2049   0.1202   0.2037
Balanced Accuracy      0.9757   0.8968   0.9346   0.8178   0.9479
```

Random Forest Prediction

```
Fit1 <- randomForest(classe ~. , data=myTraining)
pred1 <- predict(Fit1, myTest, type = "class")
confusionMatrix(pred1, myTest$classe)

Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 2232    2    0    0    0
         B    0 1516    0    0    0
         C    0    0 1367    6    0
         D    0    0    1 1280    0
         E    0    0    0    0 1442

Overall Statistics

               Accuracy : 0.9989
                 95% CI : (0.9978, 0.9995)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9985
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9987   0.9993   0.9953   1.0000
Specificity            0.9996   1.0000   0.9991   0.9998   1.0000
Pos Pred Value         0.9991   1.0000   0.9956   0.9992   1.0000
Neg Pred Value         1.0000   0.9997   0.9998   0.9991   1.0000
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2845   0.1932   0.1742   0.1631   0.1838
Detection Prevalence   0.2847   0.1932   0.1750   0.1633   0.1838
Balanced Accuracy      0.9998   0.9993   0.9992   0.9976   1.0000
```

```
pred2 <- predict(Fit1, test, type = "class")
pred2

 1  2 31  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
```