Machine Learning Peer-graded Assignment

A Steenekamp

10 November 2016

Executive Summary

Background

By 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, 6 participants produce data from accelerometers on the belt, forearm, arm, and dumbell.They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

Note: Classe A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes.

Read more on HAR and the Weight-lifting dataset at http://groupware.les.inf.puc-rio.br/har#ixzz4PagQx474

Objective

The goal of this project is to predict the manner in which the participants did the exercise. This is the "classe" variable in the training set.A report describing how the model was built, how cross validation was used, what the expected out of sample error is, and why certain choices were made. The prediction model will be used to predict 20 different test cases.

Setup environments

Load libraries

```
suppressMessages(library(randomForest))
suppressMessages(library(knitr))
suppressMessages(library(ggplot2))
suppressMessages(library(datasets))
suppressMessages(library(corrplot))
suppressMessages(library(caret))
suppressMessages(library(rpart.plot))
suppressMessages(library(rattle))
suppressMessages(library(rpart))
suppressMessages(library(gbm))
```

Getting and Cleaning the data

Read the data

```
cache=TRUE
Work_Dir <- "C:/Users/Ada/Documents/Coursera/Mod8MachineLearning"
setwd(Work_Dir)
if (!file.exists("pml-training.csv")) {
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                  destfile = paste(Work_Dir, "pml-training.csv", sep="/"))
}
if (!file.exists("pml-testing.csv")) {
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
                  destfile = paste(Work_Dir, "pml-testing.csv", sep="/"))
}
traindata <- data.frame(read.csv("pml-training.csv",na.strings=c("", "NA", "NULL")))
testdata <- data.frame(read.csv("pml-testing.csv",na.strings=c("", "NA", "NULL")))
```

Explore the original data

```
dim(traindata)
```

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

```
dim(testdata)
```

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

There is a very large number of variables and we need to determine which are the most important ones and try and reduce the dataset so that only these are left.

Create a new dataset with only the most important variables

1) Manual inspection

By manually looking at the data it is clear that we can remove the first 7 columns.They are identifying data and will not contribute to the correctness of the exercise

```
traindata<-traindata[-c(1:7)]
dim(traindata)
```

```
## [1] 19622   153
```

2) Remove columns with too many NA's

```
set.seed(12345)
trainNA <- traindata[ , colSums(is.na(traindata)) == 0]
dim(trainNA)
```

```
## [1] 19622    53
```

3) Determine highly correlated values

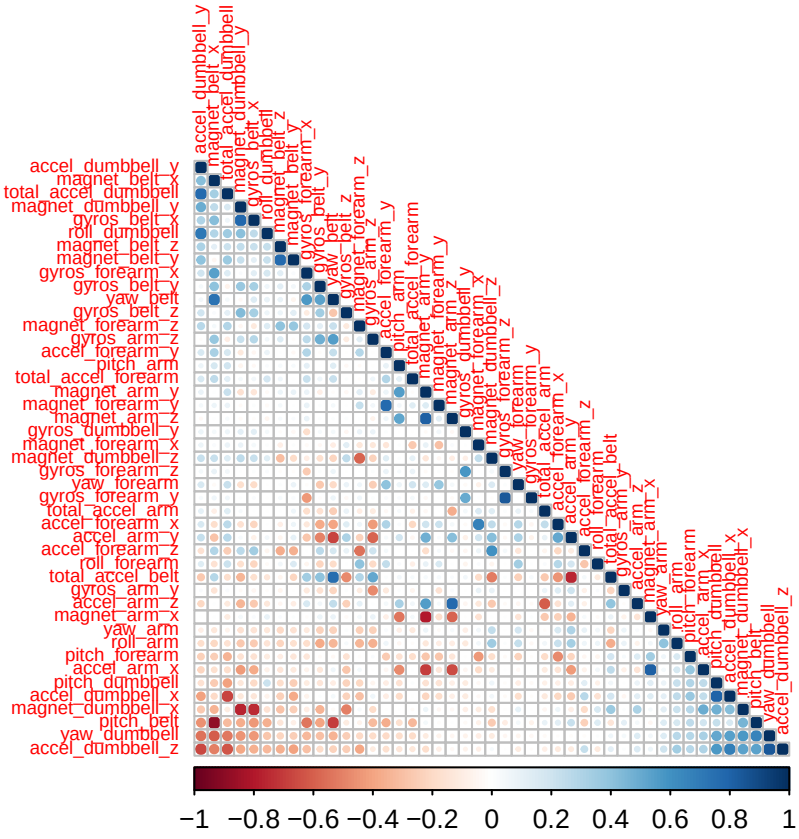
```
# first we need to remove the "classe" variable, since the cor function can only be performed on numeri
trainnum <- trainNA[,-c(53)]
cortrain<-cor(trainnum)
```

4) Remove highly correlated values

```
removecor = findCorrelation(cortrain, cutoff = .90, verbose = FALSE)
trainrem = trainnum[,-removecor]
dim(trainrem)
```

```
## [1] 19622    45
```

```
cortrain<-cor(trainrem)
corrplot(cortrain,method="circle",order = "FPC", type = "lower", tl.cex = 0.6)
```



5) Add classe variable back into the dataset

```
trainfin=trainNA[,-removecor]
dim(trainfin)
```

```
## [1] 19622    46
```

6) Create final training and testing datasets

```
inTrain = createDataPartition(y=trainfin$classe,p = 3/4,list=FALSE)
training=trainfin[inTrain,]
testing=trainfin[-inTrain,]
dim(training)
```

```
## [1] 14718    46
```

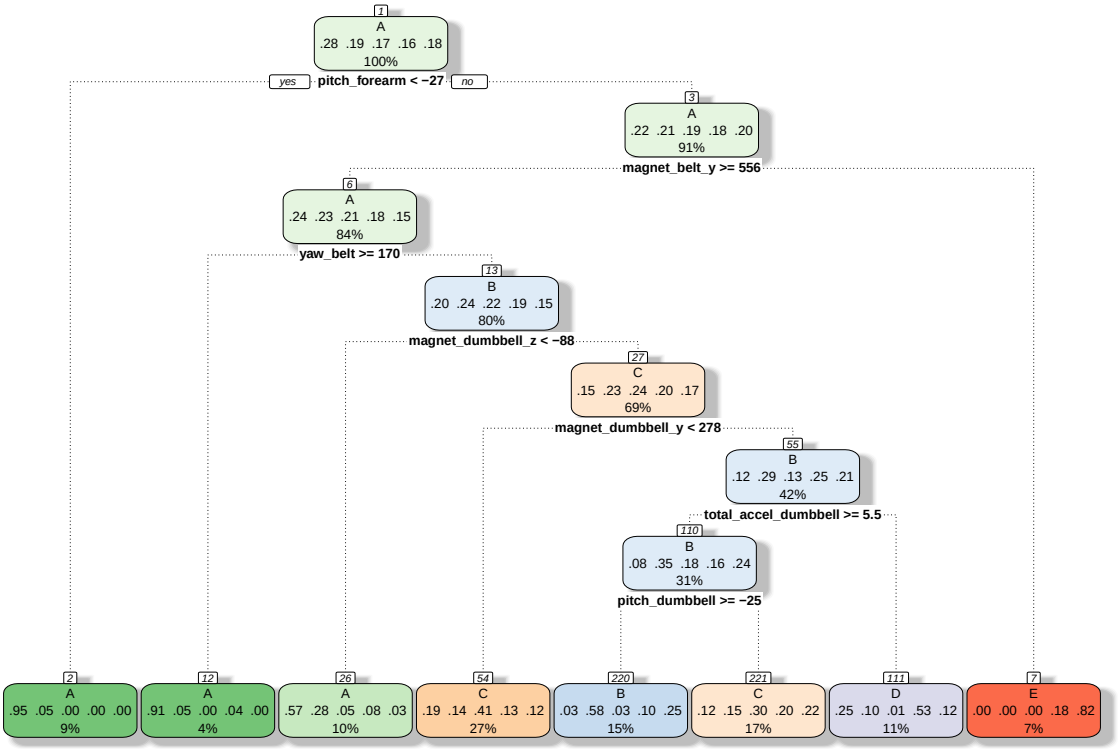
```
dim(testing)
```

```
## [1] 4904   46
```

Apply machine learning algorithms

1) Decision Tree (method="rpart")

```
set.seed(12345)
fit1<-train(classe~.,method="rpart",data=training)
fancyRpartPlot(fit1$finalModel)
```



Rattle 2016−Nov−10 16:13:56 Ada

```
predrpart<-predict(fit1,testing)
confusionMatrix(predrpart,testing$classe)$overall[1]
```

```
##  Accuracy
## 0.5358891
```

2) Random Forest (method="rf")

```
set.seed(12345)
fit2<-randomForest(classe~.,data=training,ntree=100, importance=TRUE)
predrf<-predict(fit2,testing)
confusionMatrix(predrf,testing$classe)$overall[1]
```

```
##  Accuracy
## 0.9940865
```

3) Generalized Boosted Regression Models (method = "gbm")

```
set.seed(12345)
controlgbm <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
fit3<-train(classe~.,method="gbm",data=training,verbose=FALSE,trControl = controlgbm)
```

```
## Loading required package: plyr
```

```
predgbm<-predict(fit3,testing)
confusionMatrix(predgbm,testing$classe)$overall[1]
```

```
##  Accuracy
## 0.9598287
```

Predict the test cases

The accuracy rate for the three methods are: 1) Decision Tree 54%, 51% expected out of error rate 2) Random Forest 99%, 1% expected out of error rate 3) Boosting 96%, 4% expected out of error rate The most accurate method is **Random Forest** and this will be used to predict the test cases

```
predtest <- predict(fit2, newdata=testdata)
predtest
```

```
##  1  2  3  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
## Levels: A B C D E
```