Course Project Machine Learning

Lei

23 August 2017

Background

In this project, a machine learning model was generated using data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants, in order to predict the class of movements of the test dataset. This dataset is freely available under the “Creative Commons license (CC BY-SA) license from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (Weight Lifting Exercise Dataset).

Loading datasets

First, the data was downloaded from the course links, and loaded into the Rstudio workspace.

```
#change wd:
setwd("~/Rstudio")
#download the two files to wd:
traindataUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.c
sv"
#download.file(url=traindataUrl, destfile="~/Rstudio/training.csv")

testdataUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
"
#download.file(url=testdataUrl, destfile="test.csv")

#load datasets:
training <- read.csv(url(traindataUrl), na.strings=c("NA","#DIV/0!",""))
testing <- read.csv(url(testdataUrl), na.strings=c("NA","#DIV/0!",""))
```

For model building using machine learning, we first ignore the test dataset and make some exploratory analysis on the training dataset. Note that I out-commanded the summary function as the output is much too long for this report.

```
dim(training)
```

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

```
#summary(training)
#names(training)
unique(training$classe)
```

```
## [1] A B C D E
## Levels: A B C D E
```

From the output above, we see that there are six participants (User names as level) who provided these data. The last variable “Classe” seems to be the value to be predicted. It has five levels (A, B, C, D and E).

For the sake of cross validation of the model that will be generated, we first partitioned the training dataset into two parts: inTrain, and the rest:

```
library(caret) # this command load also the packages lattice and ggplot2.
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

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

```
## Warning: package 'lattice' was built under R version 3.3.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
inTrain <- createDataPartition(training$classe, p=0.6, list = F)
myTrain <- training[inTrain, ]
myTest <- training[-inTrain, ]
dim(myTrain)
```

```
## [1] 11776   160
```

```
dim(myTest)
```

```
## [1] 7846  160
```

```
percent <- paste(round(100*dim(myTrain)[[1]]/(dim(myTrain)[[1]] + dim(myTest)[[1]])
, 2), "%", sep="")
```

As shown above, a short check of the dataset dimension shows that 60.01% of the data points were assigned to the current training set.

Cleaning data

From the dataset summary above, it seems that a lot of variables contain only marginal, or unsignificant data. These variables need to be cleaned out before building the model. Note that all the cleaning steps need to be applied on myTrain, myTest and the ultimate testing datasets in the same way.

Step 1: Remove variables with nearly zero variance (both mean and sd are zero)

```
zeros <- nearZeroVar(myTrain)
myTrain <- myTrain[, -zeros]
myTest <- myTest[, -zeros]
testing <- testing[, -zeros]
```

By this means, 30 variables were removed.

Step 2: Remove variables that are NA in more than 95% of the cases:

```
mostlyNA <- sapply(myTrain, function(x) mean(is.na(x))) > 0.95
myTrain <- myTrain[, mostlyNA==F]
myTest <- myTest[, mostlyNA==F]
testing <- testing[, mostlyNA==F]
```

Step 3: Remove the first columns

Moreover, we remove the first five variables that contain meta data of the participants, while do not have predictive values.

```
myTrain <- myTrain[, -(1:5)]
myTest <- myTest[, -(1:5)]
testing <- testing[, -(1:5)]
```

By now, only 59 variables remained these cleaning steps.

Model building 1 - Decision Tree

The principle of Decision tree appears intuitive to me due to its step-by-step manor. Therefore I first try out the Decision tree machine learning approach:

```
library(rpart)

Mod_1 <- rpart(classe ~ ., data=myTrain, method="class")
#fancyRpartPlot(modFitA1)
predict_1 <- predict(Mod_1, myTest, type = "class")
tree <- confusionMatrix(predict_1, myTest$classe)
tree
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2009  272   48   81   62
##          B  145 1012  218  191  193
##          C   24   91  964  112   63
##          D   45   88   81  822  153
##          E    9   55   57   80  971
## 
## Overall Statistics
## 
##                Accuracy : 0.7364
##                  95% CI : (0.7265, 0.7461)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.665
##  Mcnemar's Test P-Value : < 2.2e-16
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9001   0.6667   0.7047   0.6392   0.6734
## Specificity            0.9175   0.8820   0.9552   0.9441   0.9686
## Pos Pred Value         0.8127   0.5753   0.7687   0.6913   0.8285
## Neg Pred Value         0.9585   0.9169   0.9387   0.9303   0.9294
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2561   0.1290   0.1229   0.1048   0.1238
## Detection Prevalence   0.3151   0.2242   0.1598   0.1515   0.1494
## Balanced Accuracy      0.9088   0.7743   0.8300   0.7916   0.8210
```

```
#plot(Mod_1)
```

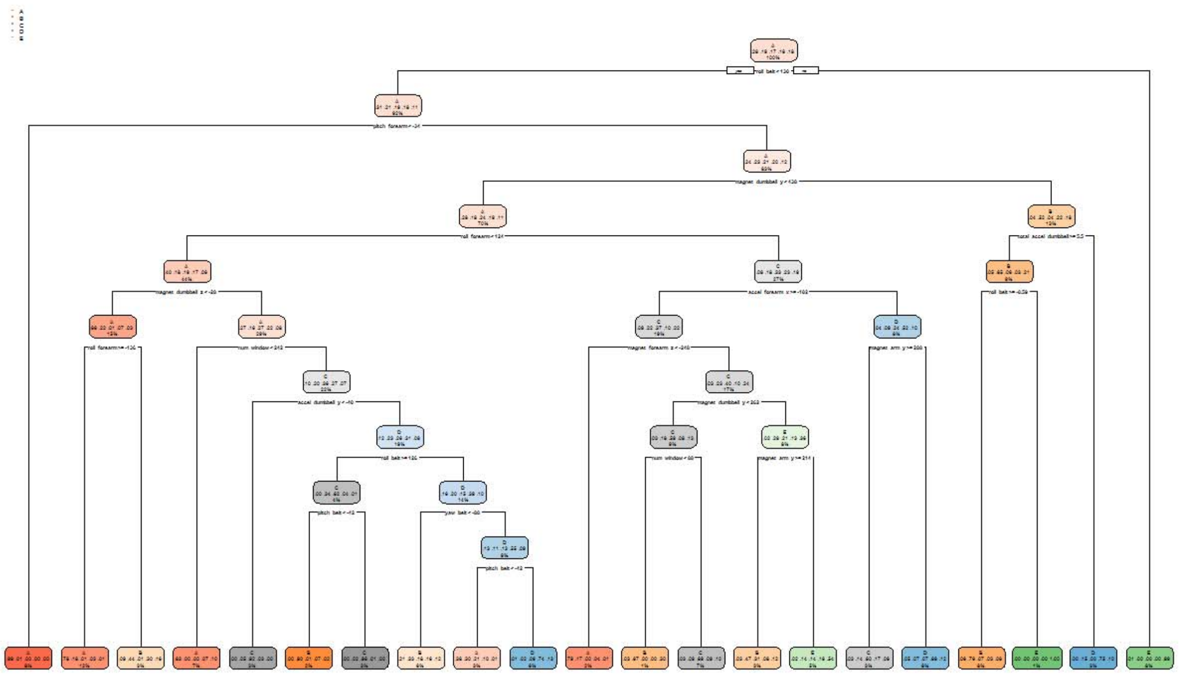
Now let's predict the myTest data using this tree model, while making some decision tree figure:

```
#install.packages("rpart.plot")
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.3.3
```

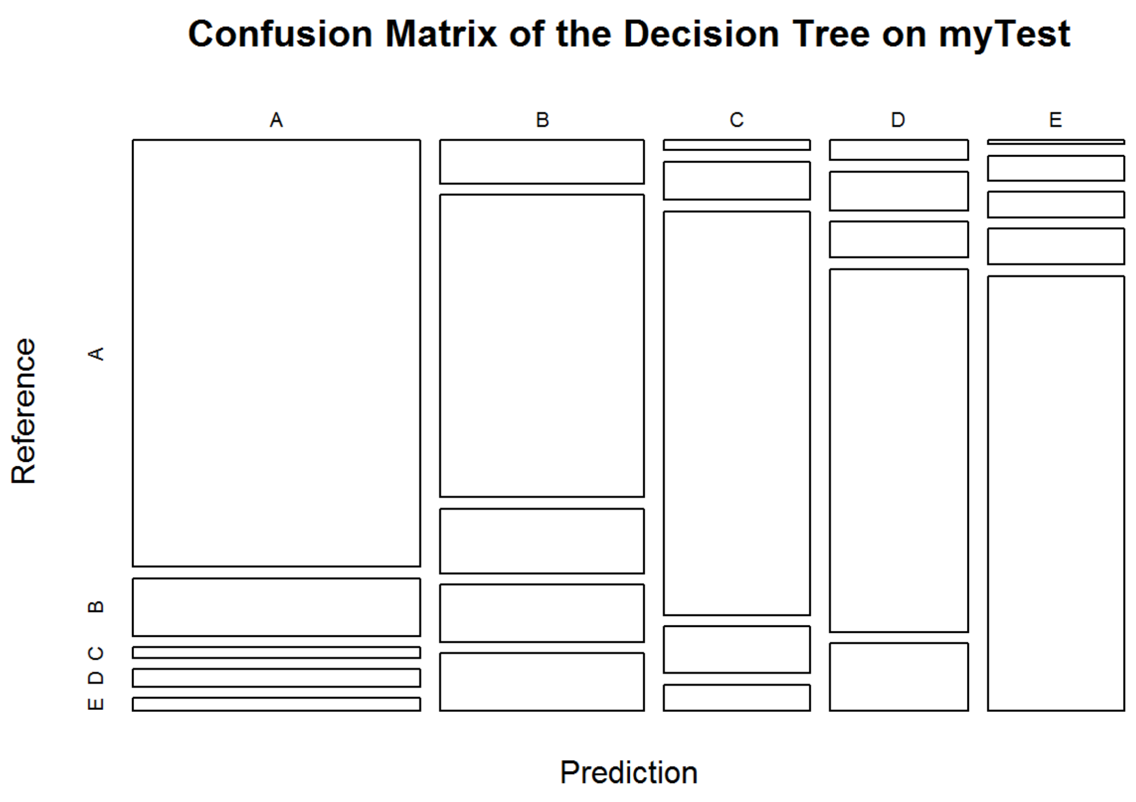
```
#library(RColorBrewer)
#library(rattle)

rpart.plot(Mod_1)
```



In the following we check the result of the cross-validation on myTest:

```
plot(tree$table, col=tree$byClass, main="Confusion Matrix of the Decision Tree on m
yTest")
```



The prediction worked in part, as can be seen by the biggest diagonal rectangulars in the confusion matrix plot above. However, the sensitivity of the model is not that high.

Model building 2 - Random Forest

Before proceeding to the final prediction on the test dataset, let’s try out the random forest model:

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

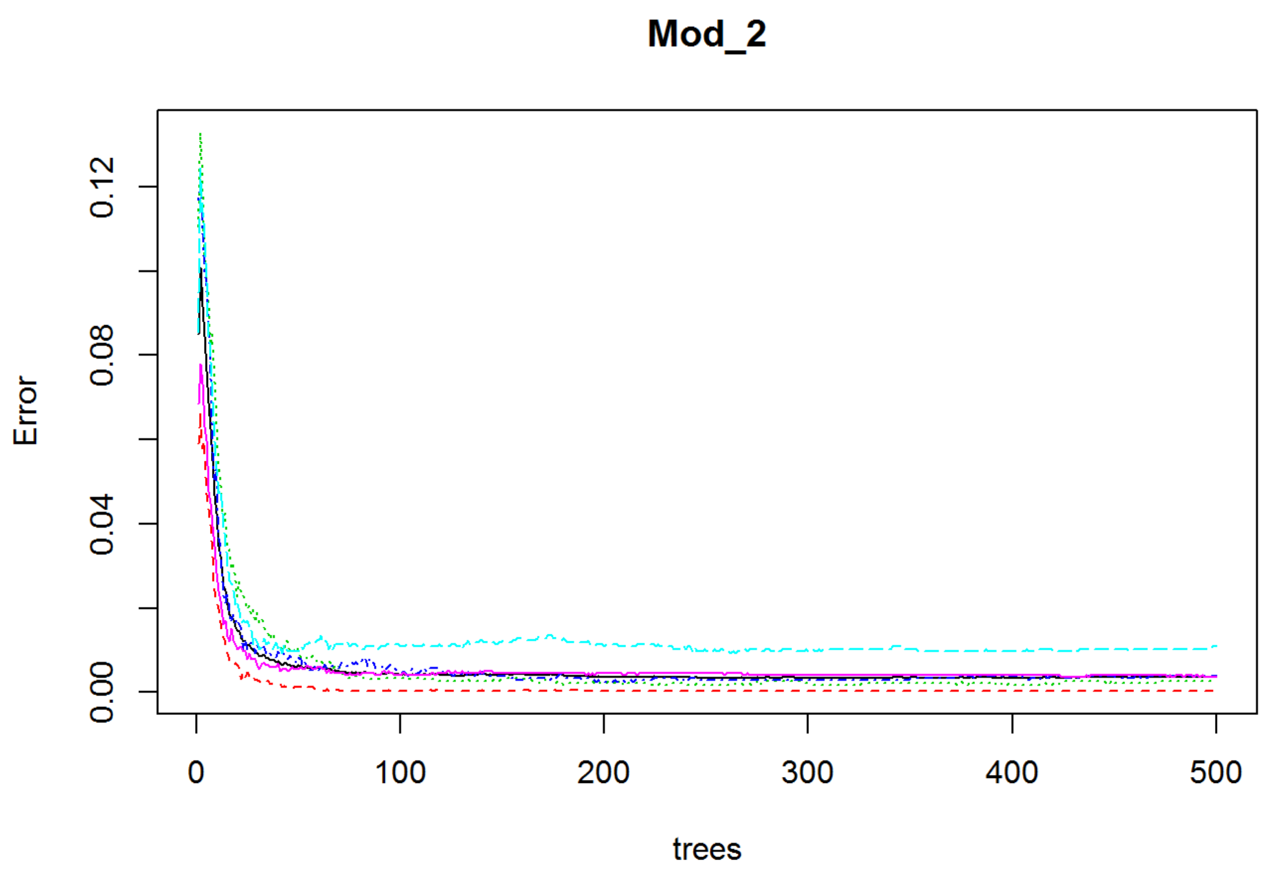
```
##
## Attaching package: 'randomForest'
```

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

```
Mod_2 <- randomForest(classe ~ ., data=myTrain)
predict_2 <- predict(Mod_2, myTest, type = "class")
forest <- confusionMatrix(predict_2, myTest$classe)
forest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2231    7    0    0    0
##          B    1 1511   11    0    0
##          C    0    0 1353   11    0
##          D    0    0    4 1275    6
##          E    0    0    0    0 1436
##
## Overall Statistics
##
##                Accuracy : 0.9949
##                  95% CI : (0.9931, 0.9964)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9954   0.9890   0.9914   0.9958
## Specificity            0.9988   0.9981   0.9983   0.9985   1.0000
## Pos Pred Value         0.9969   0.9921   0.9919   0.9922   1.0000
## Neg Pred Value         0.9998   0.9989   0.9977   0.9983   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1926   0.1724   0.1625   0.1830
## Detection Prevalence   0.2852   0.1941   0.1738   0.1638   0.1830
## Balanced Accuracy      0.9992   0.9967   0.9937   0.9950   0.9979
```

```
plot(Mod_2)
```



This time, both the sensitivity and specificity are really high, even 100% when the cross validation dataset is concerned. Therefore, we will conduct our final prediction using the random forest model:

```
#dim(myTrain)
#dim(myTest)
#dim(testing)
#names(myTest)
#names(testing)

predict_final <- predict(Mod_2, testing, type = "class")
predict_final
```

```
##  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
```

Conclusive notes

Amazingly, all 20 cases have been predicted correctly according to my quiz results! This supports a high prediction power of machine learning approaches.

Finally, I would like to acknowledge the following authors that provided the data: 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.