Weight-lifting Data Analysis - Practical Machine Learning

October 2015

Experiment: Weight-lifting

The motivation for the investigation was to analyse the cause of differing quality in weight-lifting techniques. With systematic grouping of observations of the methods used by the chosen weight-lifters, a dataset was collected and for each set of observation one of five quality grades was assigned. The resulting dataset was published in 2013 (acknowledged below).

Statistical Analysis:

Approach

We wish to find the best model for predicting classes of weight-lifting exercises based on the most significant factors (predictors). The statistical packets, developed for “machine learning”, such as caret, rpart and randomForest, can be used to formulate an optimal model.

Data analysis and conversion

The data provided included 19621 observations of 160 variables, the first 7 of which were identifiers and experimental design factors, and the last of which was the response to the remaining variables, the predictors. The data was quite “sparse” since for many variables either data was missing or not assigned. On removal of those factors for which values were missing, there remained 52 parameters to analyse as predictors. Of the two models developed using rpart and ransdomForest on 60% of the chosen data, randomForest produced the highest prediction accuracy and lowest errors on cross validation with the other 40% of the data. Therefore this model was used to operate on the assignment’s test data in calculating the response, that is, the classification of the weight-lifting exercises, to be submitted for grading.

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

Loading libraries and exploring the loaded data:

```
library(ggplot2)
library(lattice)
library(caret)
library(e1071)
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
projectTrain <-read.csv("pml-training.csv", stringsAsFactors=FALSE)
projectTest <-read.csv("pml-testing.csv", stringsAsFactors=FALSE)
dim(projectTrain)
```

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

```
dim(projectTest)
```

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

Interpreting blank entries as not assigned (“NA”) simply by re-loading the data in the uniform format:

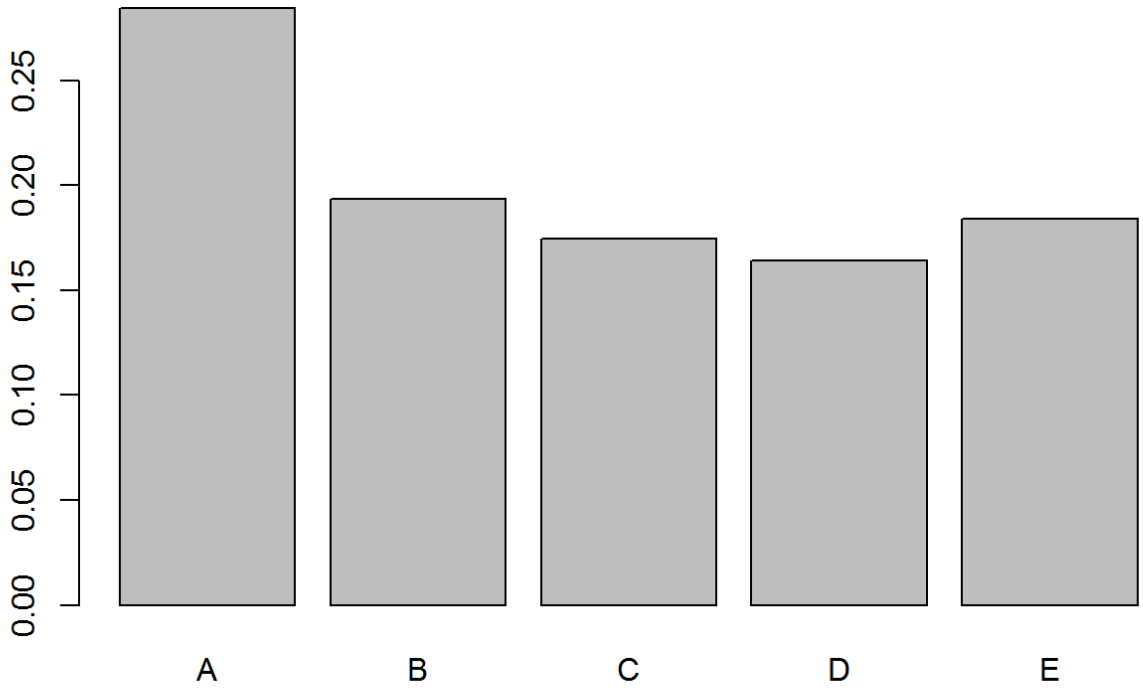
```
projectTrain <-read.csv("pml-training.csv", na.strings=c("NA",""), strip.white = T
, stringsAsFactors=FALSE)
projectTest <-read.csv("pml-testing.csv", na.strings=c("NA",""), strip.white = T,
stringsAsFactors=FALSE)
dim(projectTrain)
```

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

```
dim(projectTest)
```

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

```
barplot(prop.table(table(projectTrain$classe)))
```



The number of missing entries in both datasets is calculated in order to decide which variables may justifiably be ignored in the model.

```
sumNAtrain <- apply(projectTrain, 2, function(x) { sum(is.na(x)) })
sumNAtest <- apply(projectTest, 2, function(x) { sum(is.na(x)) })
Training <- subset(projectTrain[, which(sumNAtrain == 0)])
Testing  <- subset(projectTest[, which(sumNAtest == 0)])
```

It is obvious that the first 7 columns are identfiers and experimental design constants, rather than observations, are therefore eliminated from the two datasets.

```
trainSet <- Training[,8:60]
testSet <- Testing[,8:60]
dim(trainSet)
```

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

```
dim(testSet)
```

```
## [1] 20 53
```

The last (53rd. column) contains the response “classe” in this training dataset, trainSet, while it contains the identification, “problem_id”, in the assignment’s test dataset, testSet. In order to choose between models with the classification tree (rpart) and the Random Forest (rf), the training set will be divided into the partial training set and the model testing set on the recommended 60-40% basis using the slicing function, createDataPartition, in the caret package.

```
partition <- createDataPartition(trainSet$classe, p = 0.60, list=FALSE)
trainPart <- trainSet[partition,]
testPart  <- trainSet[-partition,]
```

The resulting training dataset will be used to measure the out of sample fit.

```
dim(trainPart)
```

```
## [1] 11776    53
```

```
dim(testPart)
```

```
## [1] 7846   53
```

```
trainPart$classe <- as.factor(trainPart$classe)
table(trainPart$classe)
```

```
##
##    A    B    C    D    E
## 3348 2279 2054 1930 2165
```

Creating the models

```
set.seed(2024)
modelRpart <- train(classe ~., method="rpart", data=trainPart)
modelRForest <-randomForest(classe~., data=trainPart, type="class")
```

Comparing the out of sample accuracey with each model

```
confusionMatrix(testPart$classe, predict(modelRpart, testPart))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1985   45  167    0   35
##          B  630  540  348    0    0
##          C  645   44  679    0    0
##          D  578  250  458    0    0
##          E  197  191  376    0  678
##
## Overall Statistics
##
##                Accuracy : 0.4948
##                  95% CI : (0.4837, 0.5059)
##     No Information Rate : 0.5143
##     P-Value [Acc > NIR] : 0.9997
##
##                   Kappa : 0.34
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.4919  0.50467  0.33481       NA  0.95091
## Specificity            0.9352  0.85567  0.88157   0.8361  0.89289
## Pos Pred Value         0.8893  0.35573  0.49635       NA  0.47018
## Neg Pred Value         0.6348  0.91625  0.79176       NA  0.99453
## Prevalence             0.5143  0.13638  0.25848   0.0000  0.09087
## Detection Rate         0.2530  0.06882  0.08654   0.0000  0.08641
## Detection Prevalence   0.2845  0.19347  0.17436   0.1639  0.18379
## Balanced Accuracy      0.7136  0.68017  0.60819       NA  0.92190
```

```
confusionMatrix(testPart$classe, predict(modelRForest, testPart))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2226    4    2    0    0
##          B   13 1501    4    0    0
##          C    0    5 1356    7    0
##          D    0    0    9 1277    0
##          E    0    0    1    8 1433
##
## Overall Statistics
##
##                Accuracy : 0.9932
##                  95% CI : (0.9912, 0.9949)
##     No Information Rate : 0.2854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9915
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9942   0.9940   0.9883   0.9884   1.0000
## Specificity            0.9989   0.9973   0.9981   0.9986   0.9986
## Pos Pred Value         0.9973   0.9888   0.9912   0.9930   0.9938
## Neg Pred Value         0.9977   0.9986   0.9975   0.9977   1.0000
## Prevalence             0.2854   0.1925   0.1749   0.1647   0.1826
## Detection Rate         0.2837   0.1913   0.1728   0.1628   0.1826
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9966   0.9957   0.9932   0.9935   0.9993
```

It can be seen how all fitting parameters in the Random Forest model are superior to those of the Classification Tree method. The accuracy of the former (99%) is in fact double that of the latter model (49%). Therefore the Random Forest model is chosen for the final model. The importance of the variable can be seen using the function impVar, as follows:

```
importPredictors <-varImp(modelRForest)
index <- order(importPredictors$Overall,decreasing=TRUE)
tablePred <-cbind(rownames(importPredictors),importPredictors)
decImportPredictors <-tablePred[index,]
rownames(decImportPredictors) <- c()
colnames(decImportPredictors) <- c("Exercise", "Overall")
head(decImportPredictors, 15)
```

```
##             Exercise  Overall
## 1          roll_belt 747.2706
## 2           yaw_belt 543.7211
## 3      pitch_forearm 483.8990
## 4  magnet_dumbbell_z 454.0379
## 5         pitch_belt 415.6360
## 6  magnet_dumbbell_y 397.7315
## 7       roll_forearm 358.4567
## 8  magnet_dumbbell_x 284.3688
## 9   accel_dumbbell_y 252.2286
## 10     magnet_belt_z 238.9570
## 11     roll_dumbbell 235.3683
## 12     magnet_belt_y 229.1628
## 13      accel_belt_z 228.4016
## 14  accel_dumbbell_z 201.6146
## 15   accel_forearm_x 194.6736
```

The final model is applied to the completely independent test data provided by the assignment.

Result:

```
testAnswer <-predict(modelRForest,newdata=testSet)

pml_write_files = function (x) {
          n = length(x)
          for (i in 1:n) {
                filename = paste0('problem_id_',i,'.txt')
                write.table(x[i], file=filename, quote=FALSE,
                                  row.names=FALSE, col.names=FALSE)
                }
          }
answers <- as.character(testAnswer)
pml_write_files(answers)
```

The files produced contain the prediction of the model developed for each of the 20 sets of predictors provided in the test. According to the submission, these were all acceptable predictions.