Project: Practical Machine Learning

Elsa

20 July 2015

This work was developed as the final project for the Practical Machine Learning Course - Coursera, from Johns Hopkins University (for more details visit https://www.coursera.org/course/predmachlearn (https://www.coursera.org/course/predmachlearn)).

Introduction

(Note: In this section the info provided for the assignment is reproduced)

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.

GOALS

In this project, the goals are to:

1. Use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants, that 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) (see the section on the Weight Lifting Exercise Dataset).
2. Predict the manner in which the participants did the exercise. This is the “classe” variable in the training set (any other variables may be used to predict with).
3. Create a report describing how the model was built, how cross validation was used and what the expected out of sample error is, and why the different choices were made.
4. Use prediction model to predict 20 different test cases.

Data

The TRAINING data for this project are available here:
https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The TEST data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

Code and Results

Set working directory, load libraries and set seed

Set working directory:

```
setwd("/Volumes/BIG_backup/DATA_All/Science_stuff/COURSES/10-Data_Science_Specialization/08_PracticalMachineLearning/Quizzes/Project")
```

Load required libraries:

```
library(AppliedPredictiveModeling)
library(caret)
```

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

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(rpart)
```

Set seed for reproducibility

```
set.seed(1111)
```

Load Data

Data was downloaded from the provided source (see Introduction section) to the working directory and was then loaded using the read.csv() function. The first column was not loaded since

```
training = read.csv("pml-training.csv")[-1]
testing = read.csv("pml-testing.csv")[-1]
```

Data checks and cleaning

Data was initially checked using the function summary(). The code used is shown below, but the output was omited for simplicity.

```
summary(training)
summary(testing)
```

Analysis of the summary data reveals several variables where there is missing data. The table below shows that there are 100 columns/variables with missing data and 59 without.

```
missingData = sapply(training, function (x) any(is.na(x) | x == ""))
table(missingData)
```

```
## missingData
## FALSE  TRUE
##    59   100
```

Only the 59 complete variables will be kept (i.e. variables containing missing data will be removed):

```
variables = names(subset(missingData, missingData == "FALSE"))
training_clean = training[,variables]
```

The summary() and class() functions were then used to check that the predictor variable to use - “classe” - was a factor variable.

```
summary(training_clean$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

I next checked for the presence of variables that have almost no variability, using the nearZeroVar() function:

```
nzv = nearZeroVar(training_clean, saveMetrics=TRUE)
table(nzv$nzv)
```

```
##
## FALSE  TRUE
##    58     1
```

This analysis identifies the variable “new_window” as having almost no variability, and, therefore, this variable was removed.

```
drop <- c("new_window")
training_clean = training_clean[,!names(training_clean) %in% drop]
```

Finally, I checked for variables related with data acquisition and that, therefore, are not suitable to be used in prediction. The identified variables were: user_name, raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timestamp, and num_window. Since these are the first 5 variables, they were removed using the

following code:

```
training_clean = training_clean[, -(1:5)]
```

So, the training_clean set has now 53 variables, including the predictor “classe”, shown below:

```
names(training_clean)
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

Partioning training set

In order to be able to do cross-validation, the cleaned training set was partioned into two subsets: trainingS (75%) and testingS (25%)

```
subsetTraining <- createDataPartition(y=training_clean$classe, p=0.75, list=FALSE)
trainingS <- training_clean[subsetTraining, ]
testingS <- training_clean[-subsetTraining, ]
```

PREDICTION MODELS

I will test two different prediction models - Decision Tree (DT) and Random Forest (RF) - and chose the most accurate to predict the 20 different test cases provided.

Model Specification

Models

```
modelDT = rpart(classe ~ ., data=trainingS, method="class")
modelRF = randomForest(classe ~. , data=trainingS, method="class")
```

Predictions

```
predictionDT = predict(modelDT, testingS, type = "class")
predictionRF = predict(modelRF, testingS, type = "class")
```

Confusion Matrix (Model cross-validation)

```
confMatrixDT = confusionMatrix(predictionDT, testingS$classe)
confMatrixRF = confusionMatrix(predictionRF, testingS$classe)
```

Print accuracy values for both models:

```
confMatrixDT[[3]]["Accuracy"]
```

```
##  Accuracy
## 0.7398042
```

```
confMatrixRF[[3]]["Accuracy"]
```

```
##  Accuracy
## 0.9959217
```

Decision

The Random Forest algorithm performed better than the Decision Trees (0.996 vs 0.740 accuracy, respectively); therefore, the Random Forest model was chosen.

For the chosen model, the expected out-of-sample error (= 1 - accuracy) is estimated at 0.004, or 0.4%. Since the test data set comprises only 20 cases, with the obtained accuracy of above 99% on the cross-validation data indicated that very few, or none, of the test samples should be missclassified.

Model predictions using the Random Forest model

Finally, I predicted the outcome levels using the Random Forest algorithm and the provided TEST dataset.

```
predictionRF_final = predict(modelRF, testing, type = "class")
predictionRF_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
```

Generate files for submission

```
pm_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)
  }
}
pm_files(predictionRF_final)
```

Session Info

```
sessionInfo()
```

```
## R version 3.2.1 (2015-06-18)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.9.5 (Mavericks)
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] rpart_4.1-10                    randomForest_4.6-10
## [3] caret_6.0-47                    ggplot2_1.0.1
## [5] lattice_0.20-33                 AppliedPredictiveModeling_1.1-6
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.11.6          formatR_1.2          nloptr_1.0.4
##  [4] plyr_1.8.3           class_7.3-13         iterators_1.0.7
##  [7] tools_3.2.1          digest_0.6.8         lme4_1.1-8
## [10] evaluate_0.7         gtable_0.1.2         nlme_3.1-121
## [13] mgcv_1.8-6           Matrix_1.2-2         foreach_1.4.2
## [16] parallel_3.2.1       yaml_2.1.13          SparseM_1.6
## [19] brglm_0.5-9          proto_0.3-10         e1071_1.6-4
## [22] BradleyTerry2_1.0-6 stringr_1.0.0        cluster_2.0.2
## [25] knitr_1.10.5         gtools_3.5.0         nnet_7.3-10
## [28] grid_3.2.1           rmarkdown_0.7        minqa_1.2.4
## [31] car_2.0-25           reshape2_1.4.1       magrittr_1.5
## [34] scales_0.2.5         codetools_0.2-14     htmltools_0.2.6
## [37] MASS_7.3-43          splines_3.2.1        pbkrtest_0.4-2
## [40] colorspace_1.2-6     quantreg_5.11        stringi_0.5-5
## [43] munsell_0.4.2        COREcarn_0.9.46
```