Weight Lifting Exercise Dataset - Prediction Assignment

TJ

26 September 2015

Executive Summary

This report is created in scope of the Practical Machine Learning Course Project Writeup excercise in Coursera to predict the manner in which the exercise was done, using the **Weight Lifting Exercises Dataset**. The approach researcher proposed for the Weight Lifting Exercises dataset is to investigate "how (well)" an activity was performed by the wearer. This analysis describe why and how the random forest approach was selected for model building and predicting the test classes of performed activity.

The Weight Lifting Exercises Dataset

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

The following sources are used for training and test data for this course project:

- Data source
- Training dataset
- Test dataset

Data Processing

The following steps were executed in order to create the tidy dataset ready for prediction model building:

```
# Load required packages
library(caret); library(AppliedPredictiveModeling); library(corrplot)
setwd("C:/Users/HP/Documents/Judit/Data_Science/Course_Project/PracMachLearn")
```

The training and test data csv files were downloaded into a "data" subirectory within the working directory, then read into R for processing.

```
# read the data into trainRaw and testRaw dataframes
trainRaw <- read.csv("./data/trainData.csv", header = TRUE, sep = ",",
                     na.strings = c("NA", "#DIV/0!", ""))
testRaw <- read.csv("./data/testData.csv", header = TRUE, sep = ",",
                     na.strings = c("NA", "#DIV/0!", ""))
```

The two dataset structures are almost identical, and the following steps were executed to clean the datasets:

1. Columns with more than 97% "NA" values were removed.
2. Columns with non numeric or integer values also removed as they are non-relevant for this analysis.
3. The last column in train set is called "classe", storing factor information on the performed activity.
4. The last column in test set is called "problem_id" and that information is used for assignment submittion.

```
## [1] "dim(train), dim(test), max(naTrainCount), max(naTestCount)"
```

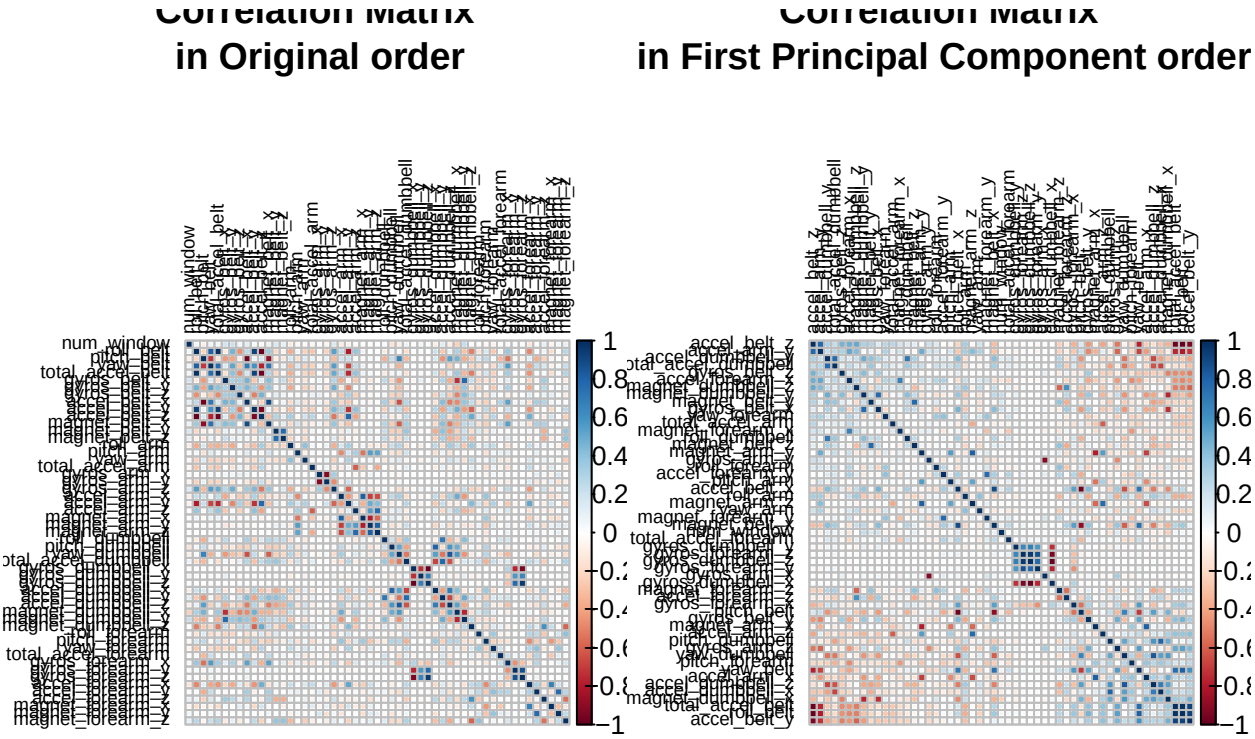
```
## [1] 19622    60    20    60     0     0
```

```
## [1] "' classe ' column in train dataset is diferent"
## [1] "Names are checked, all other are identical"
```

Model Based Prediction

In order to build a model on a training set of data and be able to test on a validation set, which has valid information in "classe" variable, the test data set is partitioned into training (70% of the data in train data set) and testing (30%) data sets.

First start with graphical exploration of the dataset and especially checking if the variables are correlated:



The above plots show the correlation between the variables, which suggests that Principal component analysis would be a good selection to use for model building. This hypotesis will be true in case the model fit accuracy is not lower then 98%, providing a small max. 2% out-of-sample error. The Hypotesis will be tested after running the model creattion.

Principal component analysis (PCA)

The reasion for this model selection also explained by the definition of the model:

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. PCA is the simplest of the true eigenvector-based multivariate analyses.

Often, its operation can be thought of as revealing the internal structure of the data in a way that best explains the variance in the data.

The first step in the model bilding is to preprocess the datasets using PCA method and 99% tresh. The next step is to build a model and the selected method is the random forest algorithm. To test the model a reference test values were predicted and the confusion matrix printed:

```
# preprocess tha dataset for better model fitting
preObj <- preProcess(training[, -y], method="pca", thresh = 0.99)
trainPC <- predict(preObj, training[, -y])
testPC <- predict(preObj, testing[, -y])

# build the model
modelFit <- train(training[, y] ~ ., method="rf", data = trainPC,
                  trControl = trainControl(method = "cv", number = 4),
                  importance = TRUE)

# predict reference test values and check the confusionMatrix table
confusionM <- confusionMatrix(testing[, y], predict(modelFit, testPC))
confusionM$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1669    2    1    2    0
##          B   18 1109   10    0    2
##          C    0    9 1011    5    1
##          D    2    0   30  932    0
##          E    0    7    7    2 1066
```

The confusion matrix shows a pretty good prediction for the validation set, the next chapter will test and see the accuracy of this model.

Cross validation and Out-of-sample error rate

The out-of-sample error is equal with the error rate on the validation data set. Sometimes called generalization error.

```
## [1] 0.9831776
```

This high model accuracy results in the following out-of-sample error rate:

```
## [1] 0.01682243
```

The out-of-sample error rate is not higher than 2%, thus the null hypotesis is valid, and this prediction model is accurate enough for predicting the test observations.

Note

The avaible time and resources for this course project limited the possibility to build the most ideal model for this practical machine learning excercise, which would contains some more steps as per below approach. As the out-of-sample error is minimal the analysis is stopped and used for predict the 20 test observations for assignment submission.

Approach: 1. Use the training set 2. Split into training/test sets 3. Build a model on the training set 4. evaluate on the test set 5. Repeat and average the estimated errors

Used for: 1. Picking variables to include in a model 2. Picking the type of prediction function to use 3. Picking the parameters in the prediction function 4. Comparing different predictors

Reference

The source Rmd file avaible on GitHub using this link

Weight Lifting Exercises 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.

Principal component analysis From Wikipedia, the free encyclopedia