Prediction Project Report of Practical Machine Learning

C. O.

Sunday, June 21, 2015

Executive Summary

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, the main goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict how well they were doing the exercise using a relatively simple prediction model.

Loading and Preprocessing the Data

First of all, it is necessary to set up working directory before R programming.

```
setwd("/Users/Michael/Desktop/Codes/PracticalMachineLearning")
```

```
library (lattice)
library(ggplot2)
library(knitr)
library(caret)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.2.1
```

```
library(randomForest)
```

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

```
opts_chunk$set(echo = TRUE, cache = FALSE,eval = TRUE, results = 'hold')
options(scipen = 1)  # Turn off scientific notations for numbers
```

Download the training and testing data sets.

```
# check if a data directory exists; if not then create a new one
if (!file.exists("data"))
{
  dir.create("data")
}

# training and testing url and dest files
url_training <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
dest_training <- "./data/training.csv"
```

```r
url_testing <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
dest_testing <- "./data/testing.csv"

# Download the training and testing files and note the downloaded time
download.file(url_training, destfile = dest_training, method="libcurl")
download.file(url_testing, destfile = dest_testing, method="libcurl")
dateDownloaded <- date()
```

The training data was then loaded into R.

```r
# Read the training data file
data_training <- read.csv("./data/training.csv", na.strings= c("NA",""," "))
data_testing <- read.csv("./data/testing.csv", na.strings= c("NA",""," "))

#str(data_training)   ##To be commmented in the report
#str(data_testing)     ##To be commmented in the report
#View(data_training) ##To be commmented in the report
#View(data_testing) ##To be commmented in the report
```

There are loads of NA values in the data. We need to clean and remove these columns were removed from the data set. The first eight columns that acted as identifiers for the experiment were also removed.

```r
# clean the data by removing columns with NAs
training_NAs <- apply(data_training, 2, function(x) {sum(is.na(x))})
training_new <- data_training[,which(training_NAs == 0)]

testing_NAs <- apply(data_testing, 2, function(x) {sum(is.na(x))})
testing_new <- data_testing[,which(testing_NAs == 0)]

# remove redundant columns
training_final <- training_new[8:length(training_new)]
testing_final  <- testing_new[8:length(testing_new)]

#str(training_final)     ##To be commmented in the report
#str(testing_final)       ##To be commmented in the report
#dim(training_final)  #19622    53   ##To be commmented in the report
#dim(testing_final)  #20 53     ##To be commmented in the report
```

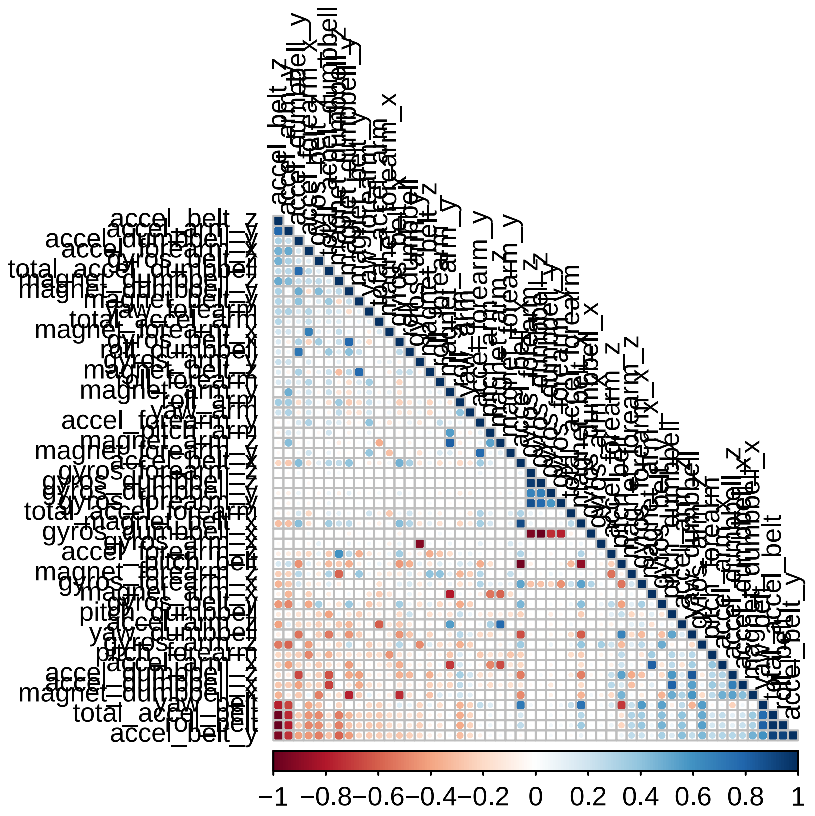
Training a Model

The test data set was split up into training and cross validation sets in a 70:30 ratio in order to train the model and then test it against data it was not specifically fitted to.

```r
### split the cleaned training data into training and cross validation data set
inTrain <-  createDataPartition(y = training_final$classe, p = 0.7, list = FALSE)
training <- training_final[inTrain, ]
crossval <- training_final[-inTrain, ]
```

A random forest model was selected to predict the classification. The correllation plot was used to see how strong the variables relationships are with each other.

```
# plot a correlation matrix using corrplot
corMatrix <- cor(training[, -length(training)])
corrplot(corMatrix, order = "FPC", method = "circle", type = "lower", tl.cex = 0.8,  tl.col = rgb(0, 0,
```



In the graph, the dark red and blue colours indicate a highly negative and positive relationship respectively between the variables. There was not much concern for highly correlated predictors which meant that all of them can be contained in the model.

And then, a model was fitted with the outcome set to the training class and all the other variables used to predict.

```
# fit a model to predict the classe using everything else as a predictor
model <- randomForest(classe ~ ., data = training)
model
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.48%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3902    2    0    0    2 0.001024066
## B    9 2644    5    0    0 0.005267118
## C    0   11 2380    5    0 0.006677796
## D    0    0   23 2227    2 0.011101243
## E    0    0    1    6 2518 0.002772277
```

The model produced a very small OOB estimate of error rate of 0.5%. This was deemed good enough to progress the testing.

Model Cross Validation

The model was further used to classify the remaining 30% of cross validation data.

```
## Model cross validation using the rest 30% of training data
predictCross <- predict(model, crossval)
confusionMatrix(crossval$classe, predictCross) #a confusion matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    8 1129    2    0    0
##          C    0    8 1014    4    0
##          D    0    0   13  951    0
##          E    0    0    0    2 1080
##
## Overall Statistics
##
##                Accuracy : 0.9937
##                  95% CI : (0.9913, 0.9956)
##     No Information Rate : 0.2858
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.992
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9952   0.9930   0.9854   0.9937   1.0000
## Specificity            1.0000   0.9979   0.9975   0.9974   0.9996
## Pos Pred Value         1.0000   0.9912   0.9883   0.9865   0.9982
## Neg Pred Value         0.9981   0.9983   0.9969   0.9988   1.0000
## Prevalence             0.2858   0.1932   0.1749   0.1626   0.1835
## Detection Rate         0.2845   0.1918   0.1723   0.1616   0.1835
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9976   0.9954   0.9915   0.9955   0.9998
```

The confusion matrix and statistics shows that this model has a 99.51% prediction accuracy. Again, this model was proved good enough to predict new data.

Prediction of Testing Data

A separate data set was then loaded into R and cleaned in the same manner as before. The model was then used to predict the classifications of the 20 results of this new data.

```
# predict the classes using the testing data set
predTesting <- predict(model, testing_final)
predTesting
```

```
##  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
```

Conclusions

With the abundance of information given from multiple measuring instruments it's possible to accurately predict how well a person is preforming an excercise using a relatively simple prediction model.