Predictive Machine Learning - Course Project Writeup

Sunday, December 16, 2014

Wearable mobile devices a a trend today and especially devices that monitor the human health parameters like blood pressure, stress levels et al have become increasingly popular. Wearable fitness devices like Fit Bit, Nike Fuel Band collect a large amount of data and if measurable insights can be drawn from it, it is not only beneficial to the wearer, but can also predict ways in which certain exercises/activities can be done to improve overall health.

In this assignment, we start with the raw data (provided seprately as training and testing data) from accelerometers on the belt, forearm, arm & dumbell of 6 participants. It is required to analyse and exercise data and then be able to predict the manned in which the participants performed the exercise.

We start with setting up of the libraries required to perform the analysis

Initial steps - library loading and download data

```
library(knitr)
opts_chunk$set(cache=TRUE,echo=TRUE)
library(caret)
library(randomForest)
#setwd("E:/Personal/__Learning/__Coursera/Johns Hopkins - Data Science/08. Predictive Machine Learning/Project")
```

Download data

```
downloadDataset <- function(URL="", destFile="data.csv"){
  if(!file.exists(destFile)){
    download.file(URL, destFile, method="curl")
  }else{
    message("Dataset already downloaded.")
  }
}

trainURL<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testURL <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
downloadDataset(trainURL, "pml-training.csv")
```

```
## Dataset already downloaded.
```

```
downloadDataset(testURL, "pml-testing.csv")
```

```
## Dataset already downloaded.
```

Loading the data

```
training <- read.csv("pml-training.csv",na.strings=c("NA",""))
testing <-read.csv("pml-testing.csv",na.strings=c("NA",""))
```

We now check how many rows/columns are present in training and test sets

```
dim(training)
```

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

```
dim(testing)
```

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

Data processing

```
sum(is.na(training))  # Total NA values
```

```
## [1] 1921600
```

```
table(colSums(is.na(training)))
```

```
##
##     0 19216
##    60   100
```

```
table(colSums(is.na(testing)))
```

```
##
##   0  20
##  60 100
```

It is clear that 60 variables have 0 NA values while the rest have NA values for almost all the rows of the dataset, so we will ignore them

Clean up of training dataset

```
columnNACounts <- colSums(is.na(training))
delColumns <- columnNACounts >= 0.95* nrow(training)
cleanTrainingdata <- training[!delColumns]
sum(is.na(cleanTrainingdata))
```

```
## [1] 0
```

```
cleanTrainingdata <- cleanTrainingdata[, c(7:60)] # removing columns not needed
```

Clean up of testing dataset

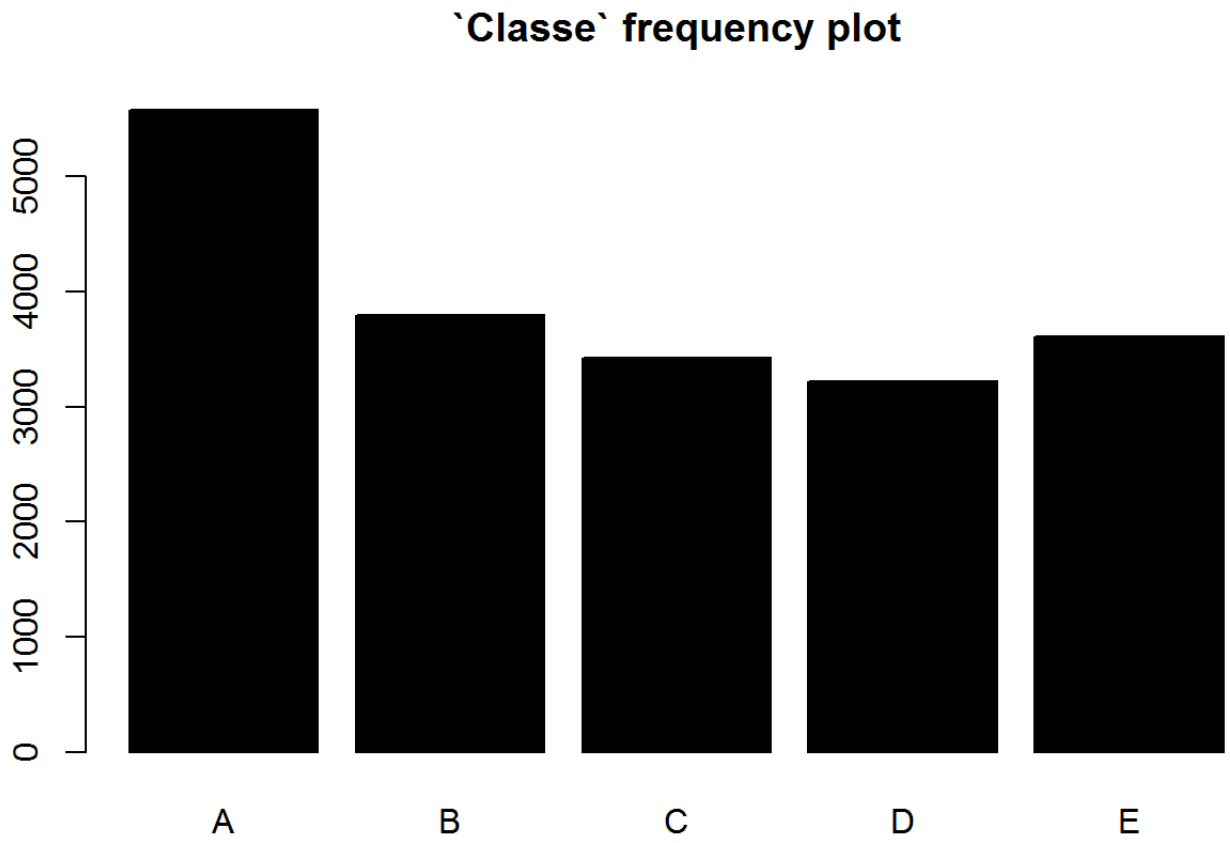
```
columnNACounts <- colSums(is.na(testing))
delColumns <- columnNACounts >= 0.95*nrow(testing)
cleanTestingdata <- testing[!delColumns]
sum(is.na(cleanTestingdata))
```

```
## [1] 0
```

```
cleanTestingdata <- cleanTestingdata[, c(7:60)] # removing columns not needed
```

Exploratory data anaylsis

```
s <- summary(cleanTrainingdata$classe)
plot(cleanTrainingdata$classe,col=cleanTrainingdata$classe,main = "`Classe` frequency plot")
```



Data partitioning & Model building

```
partition <- createDataPartition(y = cleanTrainingdata$classe, p = 0.6, list = FALSE)
trainingdata <- cleanTrainingdata[partition, ]
testdata <- cleanTrainingdata[-partition, ]
```

We set the seed before building the Random forest model. And we use a 4 fold cross-validation during the process.

In-sample accuracy

```
training_pred <- predict(model, trainingdata)
confusionMatrix(training_pred, trainingdata$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3348    0    0    0    0
##          B    0 2279    0    0    0
##          C    0    0 2054    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

out of sample accuracy

```
testing_pred <- predict(model, testdata)
confusionMatrix(testing_pred, testdata$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    3    0    0    0
##          B    0 1512    3    0    0
##          C    0    3 1362    2    0
##          D    0    0    3 1283    3
##          E    0    0    0    1 1439
##
## Overall Statistics
##
##                Accuracy : 0.9977
##                  95% CI : (0.9964, 0.9986)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9971
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9960   0.9956   0.9977   0.9979
## Specificity            0.9995   0.9995   0.9992   0.9991   0.9998
## Pos Pred Value         0.9987   0.9980   0.9963   0.9953   0.9993
## Neg Pred Value         1.0000   0.9991   0.9991   0.9995   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1927   0.1736   0.1635   0.1834
## Detection Prevalence   0.2849   0.1931   0.1742   0.1643   0.1835
## Balanced Accuracy      0.9997   0.9978   0.9974   0.9984   0.9989
```

From the above, we see that the in-sample and out-of-sampe accuracies are very high - 100% and 99.76% respectively for trainig and testing data sets.

2nd part of the project: Prediction on the testing set

In this section, we are required to predict the exercise types of 20 test cases based on the model we have created thus far.

```
predictions <- predict(model, cleanTestingdata)
predictions <- as.character(predictions)
predictions
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

These answers have to be written to individual files so that they can submitted for evaluation via the Coursera web site. The following code achieves this objective.

```
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)
  }
}

pml_write_files(predictions)
```