Machine Learning Project

Sofian Hamiti

Summary

The objective of the project is to provide 20 submissions that predict the exercises being performed in the test set.

We use the Random Forests method to fit our model.

The data for this project come from the paper: 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 enabling multi-core processing

```
library(caret)
library(randomForest)
library(doParallel)
set.seed(1337)
cl <- makeCluster(detectCores())
registerDoParallel(cl)
```

Downloading and reading data

```
training.file <- "pml-training.csv"
test.file     <- "pml-testing.csv"
training.url  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test.url      <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(training.url, training.file, method="auto")
download.file(test.url, test.file, method="auto")
training       <- read.csv(training.file, na.strings = c("", "NA", "#DIV/0!") )
test           <- read.csv(test.file, na.strings = c("", "NA", "#DIV/0!") )
training       <- training[,-c(1,5,6)]
test           <- test[,-c(1,5,6)]
```

Tidying data

We remove NA values on both data frames.

```
training <- training[,sapply(training, function(x) !any(is.na(x)))]
training <- training[ , colSums(is.na(training)) < nrow(training)]
test <- test[,sapply(test, function(x) !any(is.na(x)))]
test <- test[ , colSums(is.na(test)) < nrow(test)]
```

Partition training data into training.train and training.test sets (50% in size)

```
partition  <- createDataPartition(training$classe, p=.50, list=FALSE)
training.train <- training[partition,]
training.test  <- training[-partition,]
```

Fitting a Random Forests model

```
model <- train(training.train[,-57],
                       training.train$classe,
                       tuneGrid=data.frame(mtry=3),
                       trControl=trainControl(method="none")
                       )
```

Confusion Matrix for testing set

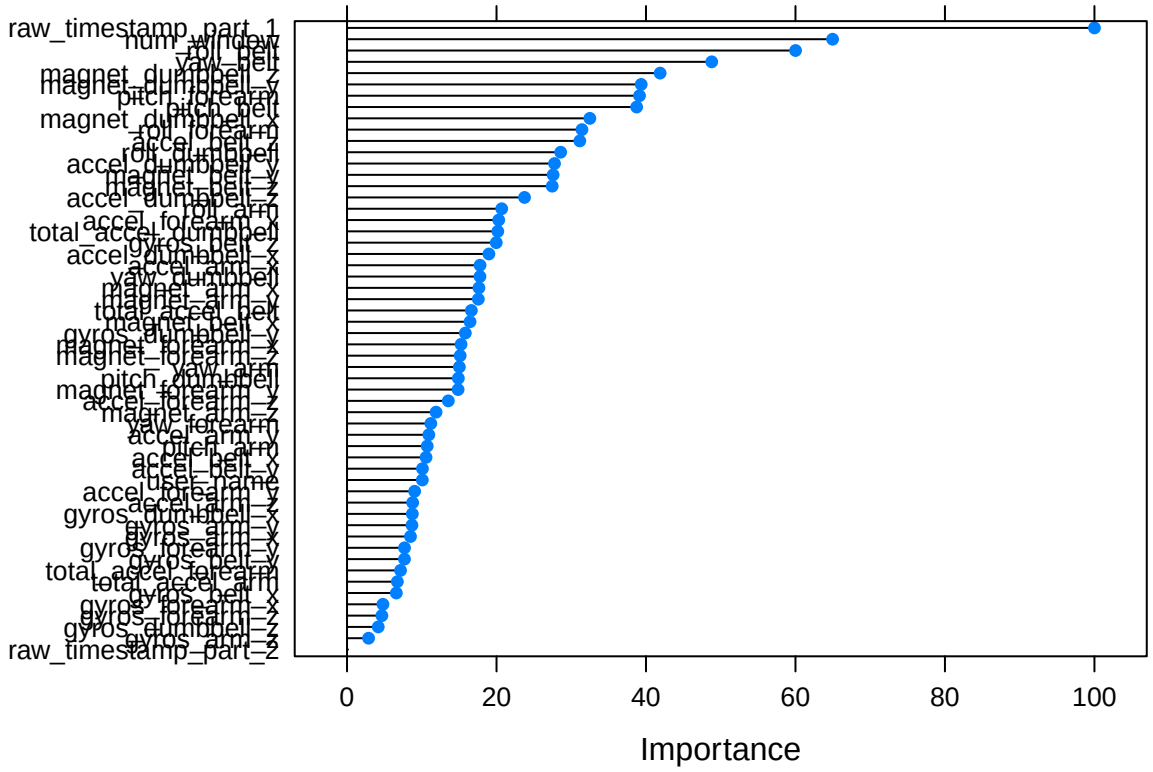
```
confusionMatrix(predict(model,
                        newdata=training.test[,-57]),
                training.test$classe
                )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2790    5    0    0    0
##          B    0 1892   12    0    0
##          C    0    1 1699   13    0
##          D    0    0    0 1595    4
##          E    0    0    0    0 1799
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.995, 0.9975)
##     No Information Rate : 0.2844
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9955
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9968   0.9930   0.9919   0.9978
## Specificity            0.9993   0.9985   0.9983   0.9995   1.0000
## Pos Pred Value         0.9982   0.9937   0.9918   0.9975   1.0000
## Neg Pred Value         1.0000   0.9992   0.9985   0.9984   0.9995
```

```
## Prevalence             0.2844   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2844   0.1929   0.1732   0.1626   0.1834
## Detection Prevalence   0.2849   0.1941   0.1746   0.1630   0.1834
## Balanced Accuracy      0.9996   0.9977   0.9956   0.9957   0.9989
```

Accuracy: 0.9964, Kappa statistic: 0.9955. We can consider the model as really good.

```
plot(varImp(model))
```




```
stopCluster(cl)
```