Exercise Movement Prediction

Magdalene Ler

Overview

Human Activity Recognition (HAR) is a key research area in the past few years, which is garnering increasing attention with its pervasive computing research community. There are many potential applications for HAR, like elderly monitoring, life log systems for monitoring energy expenditure and for supporting weight-loss programs, and digital assistants for weight lifting exercises. Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity at a rate never seen before.

Six participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different ways:

- 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).

This report will describe how the captured data are used to identify the important predictors involved in movement prediction of the above classification. It is subsequently used to predict the movement of 20 test cases. The training data is subdivided into two groups: a training data and a validation data to be used for cross-validation.

The training model used for the prediction is Random Forest and was able to achieve >99% accuracy with 0.03% out-of-sample error. It was able to achieve 100% accuracy in predicting 20 test cases in the testing dataset.

Downloading Data and Package

Downloading Package

```
library(ggplot2)
library(lattice)
library(caret)
library(foreach)
library(parallel)
library(doParallel)
```

Downloading Data

Downloading Training Data

```
csvfile1 <- "pml_training.csv"
if (!file.exists(csvfile1)) {
    url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
    download.file(url, destfile = csvfile1)
}
training <- read.csv(csvfile1, na.strings = c("NA","#DIV/0!",""))
```

Downloading Testing Data

```
csvfile2 <- "pml_testing.csv"
if (!file.exists(csvfile2)) {
    url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
    download.file(url, destfile = csvfile2)
}
testing <- read.csv(csvfile2, na.strings = c("NA","#DIV/0!",""))
```

Data Cleansing

The data is cleansed to remove invalid predictors by:

- Removing columns with near zero values
- Removing columns with NA or is empty
- Removing columns that are non-predictors

Removing columns with near zero values

```
subtraining <-training[, names(training)[!(nzv(training, saveMetrics = T)[, 4])]]
```

Removing columns with NA or is empty

```
subtraining <-subtraining[, names(subtraining)[sapply(subtraining, function (x) ! (any(is.na(x) | x ==
```

Removing columns that are non-predictors

Columns such as serial number are removed as they are not needed in the prediction exercise.

```
subtraining <- subtraining[,-1]
subtraining <- subtraining[, c(1:3, 5:58)]
```

Split the dataset to be used for Cross Validation

A subset is separated out from the training dataset in order to be used for validation. The cross validation dataset is used to compare the model created through the training dataset.

```
partitions <- createDataPartition(subtraining$classe, p = 0.6, list = FALSE)
trainingDataSet <- subtraining[partitions,]
validationDataSet <- subtraining[-partitions,]
```

Build Prediction Model

The training dataset from the previous section is used to create the prediction model using Random Forest. The CPU Resources is set to run in parallel to maximize the usage of all CPU Cores.

```
# Check if model file exists
model <- "modelFit.RData"
if (!file.exists(model)) {

    # set up the parallel clusters
    cl <- makeCluster(detectCores() - 1)
    registerDoParallel(cl)

    modelFit <- train(trainingDataSet$classe ~ ., method = "rf", data = trainingDataSet)
    save(modelFit, file = "modelFit.RData")
    stopCluster(cl)
} else {
    # existing model exists from previous run, load it and use it.
    load(file = "modelFit.RData", verbose = FALSE)
}
```

Accuracy and Sample Error Measurement of the Prediction Model

Confusion Matrix is ran against the prediction results from various model fitting functions of both training data set and validation data set to determine its accuracy and sample error.

Confusion Matrix for Training Data Set

From the training dataset, the accuracy is very high at >99% with a sample error of 0.0003.

```
trainingPrediction <- predict(modelFit, trainingDataSet)
confusionMatrix(trainingPrediction, trainingDataSet$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3348    0    0    0    0
##          B    0 2279    3    0    0
```

```
##          C    0    0 2051    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 0.9997
##                  95% CI : (0.9993, 0.9999)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9997
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   0.9985   1.0000   1.0000
## Specificity            1.0000   0.9997   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   0.9987   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   0.9997   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1742   0.1639   0.1838
## Detection Prevalence   0.2843   0.1938   0.1742   0.1639   0.1838
## Balanced Accuracy      1.0000   0.9998   0.9993   1.0000   1.0000
```

Confusion Matrix for Validation Data Set

From the validation dataset, the accuracy is also very high at >99% with an out-of-sample error of 0.0003.

```
validationPrediction <- predict(modelFit, validationDataSet)
confusionMatrix(validationPrediction, validationDataSet$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    0    0    0    0
##          B    0 1517    1    0    0
##          C    0    1 1367    1    0
##          D    0    0    0 1285    0
##          E    0    0    0    0 1442
##
## Overall Statistics
##
##                Accuracy : 0.9996
##                  95% CI : (0.9989, 0.9999)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9995
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9993   0.9993   0.9992   1.0000
## Specificity            1.0000   0.9998   0.9997   1.0000   1.0000
## Pos Pred Value         1.0000   0.9993   0.9985   1.0000   1.0000
## Neg Pred Value         1.0000   0.9998   0.9998   0.9998   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1933   0.1742   0.1638   0.1838
## Detection Prevalence   0.2845   0.1935   0.1745   0.1638   0.1838
## Balanced Accuracy      1.0000   0.9996   0.9995   0.9996   1.0000
```

Out-of-Bag (OOB) Estimated Error

The OOB Estimated Error is 0.13%, however since we observe the following:

- a consistent accuracy of >99% observed for both the training dataset and validation dataset
- cross-validation out-of-sample error of 0.03% at confidence interval of (0.9993, 0.9999)

It is safe to apply the prediction model to the 20 test cases in the testing dataset to predict the classe.

```
modelFit$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 31
##
##         OOB estimate of  error rate: 0.13%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3347    1    0    0    0 0.0002986858
## B    2 2276    1    0    0 0.0013163668
## C    0    1 2052    1    0 0.0009737098
## D    0    0    6 1922    2 0.0041450777
## E    0    0    0    1 2164 0.0004618938
```

Important Predictors for the Prediction Model

Below is a list of the important predictors used in the Random Forest prediction model that facilitates it in achieving >99% accuracy.

```
varImp(modelFit)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 60)
##
```

```
##                      Overall
## raw_timestamp_part_1 100.000
## num_window            49.388
## roll_belt             47.281
## pitch_forearm         28.280
## magnet_dumbbell_z     21.941
## magnet_dumbbell_y     20.244
## yaw_belt              18.039
## roll_forearm          16.732
## pitch_belt            16.099
## accel_dumbbell_y       8.046
## accel_forearm_x        6.813
## magnet_dumbbell_x      6.700
## accel_belt_z           6.505
## roll_dumbbell          6.238
## total_accel_dumbbell   6.175
## magnet_belt_z          5.242
## magnet_belt_y          4.906
## accel_dumbbell_z       4.711
## magnet_forearm_z       3.185
## yaw_dumbbell           3.114
```

Applying Prediction Model to Testing Dataset

```
testingPrediction <- predict(modelFit, testing)
testingPrediction
```

```
##  [1] 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
```

Conclusion

The prediction model predicted the 20 test cases with 100% accuracy.