Practical Machine Learning Course Project

Amitabh Mishra

April 25, 2020

1. Introduction

Human Activity Recognition (HAR) is an active area of research for monotoring human exercise activities facilitated by wearable and interconnected sensors (e.g. Accelerometers) on the body as well as on exercise equipment. In this project students are asked to analyse the HAR dataset using machine learning algorithms. There are many potential applications of HAR, e.g. elderley monitoring, life log monitoring for energy expenditure and supporting weight loss program, and digital assistants for weight lifting exercises and so on so forth.

The dataset for this project consists of five classes of activities such as sitting down, standing up, standing, walking ad sitting collected on 8 hour of activities of four healthy suspects. The approach proposed in weight lifting exercises dataset is to investigate "how well" an activity was performed by the wearer. For this experiment six young health participants were asked to perform one set of 10 repetitions of the unilaterla Dumbbell Biceps Curl in five different fashions exactly according to the specifications. If subjects performed

1. Exactly according to the specification - (Class A) - correct
2. Throwing the elbows to the front (Class B) - mistake
3. Lifting the dumbbell only halfway (Class C) - mistake
4. Lowering the dumbbell only halfway (Class D) - mistake
5. Throwing the hips to the front (Class E) - mistake

Accelerometers were located on (1) belt, (2) forearm, and (3) arm which took the measurements.

For this assignment we are asked to create a report describing:

1. How the model was built
2. how you used cross validation
3. what you think the expected out of sample error is
4. why you made the choices you did

2. Plan

The model building used in this documnet follws the suggested model in the lectures. We begin by (1) identifying the questions that we would like model to provide answers for, (2) prepare the date for the analysis from the HAR dataset, (3) Choose appropriate algorithms which are capable of answering the questions, (4) Predict results, (5) Validate results, (6)

Evaluate further if necessary. We use cross Validation for the trainControl function with 4 folds. The out of sample error was found to be 0.0037% when the model was applied to the test data derived from the training set.

Model choices made at every step are described in this rmd file.

3. Download the Data and Libraries

Libraries that are needed for the project are caret, e1071, randomForest, and doParallel. Also Suppressing the warning signs so that they do not appear in the project file that will be submitted.

```
# training data
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "training.csv", method = "curl")
#test data
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "testing.csv", method = "curl")
train <- read.csv("training.csv")
test <- read.csv("testing.csv")
```

4. Load Libraries

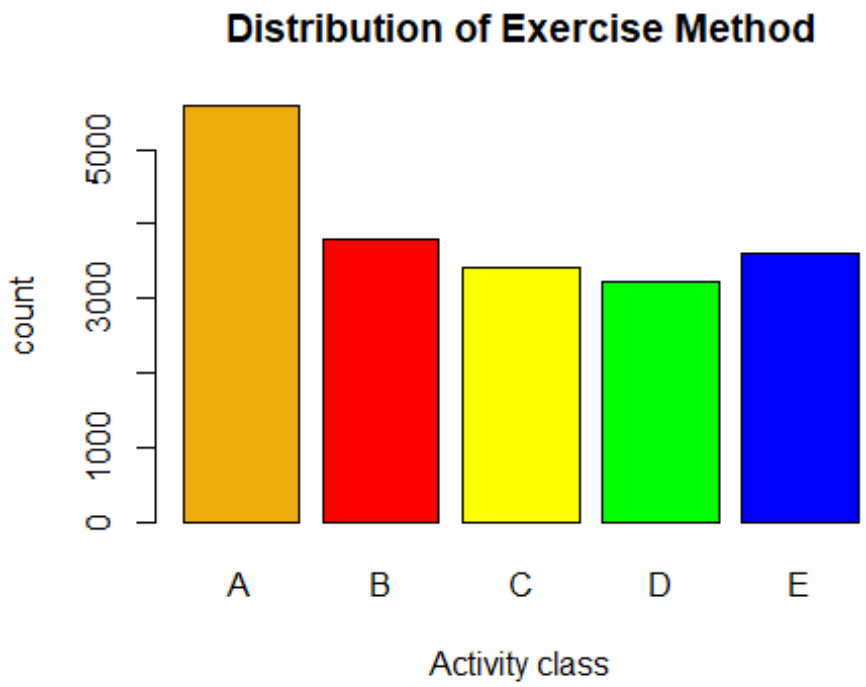
```
defaultW <- getOption("warn")
options(warn = -1)
library(doParallel)
library(randomForest)
library(e1071)
library(caret)
options(warn = defaultW)
```

On inspection in the input Excel files, found NA,#DIV/0! and blank values in the data. These are not valid observed values, so removing them with na.strings parameter andlisting the data size.

```
#download.file(trainingUrl, trainingFilename)
#download.file(quizUrl,quizFilename)
trainingFilename  <- 'pml-training.csv'
quizFilename      <- 'pml-testing.csv'
```

5. Data Cleansing for Trainin Data

```
defaultW <- getOption("warn")
options(warn = -1)
set.seed(1603)
training.df <- read.csv(trainingFilename, na.strings=c("NA","","#DIV/0!"))
training.df <-training.df[,colSums(is.na(training.df)) == 0]
plot(train$classe, xlab="Activity class", ylab="count", main="Distribution of Exercise Classes", col=c("darkgoldenrod2","red","yellow","green","blue"))
```




```
options(warn = defaultW)

dim(training.df)

## [1] 19622    60
```

We see that both the test and training data sets have the same column dimensions, with only the last column differing in name. For our training data set the last column is the "classe" variable, which is the variable that predicts the manner in which the participants do excercise. From the dataset documentation, we get that five different fashions of activity are: 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). For our testing set the last column is a problem id.

In the plot, we can see that most activities are classified in class "A", which is performing the activity exactly as specified.

6. Data Cleansing for Test Data

```
quiz.df         <-read.csv(quizFilename , na.strings=c("NA", "", "#DIV/0!"))
quiz.df         <-quiz.df[,colSums(is.na(quiz.df)) == 0]
```

7. Feature Reorganization

Removing non-essentials predictors from the dataset. If you check the data files you will find the index, subject name, time and window variables as predictors. Also checking and then removing near zero values in training data

```
Training.df   <-training.df[,-c(1:7)]
Quiz.df <-quiz.df[,-c(1:7)]
Training.nzv<-nzv(Training.df[,-ncol(Training.df)],saveMetrics=TRUE)
dim(Training.df)

## [1] 19622    53

dim(Quiz.df)

## [1] 20 53

dim(Training.nzv) [1]

## [1] 52
```

We find that there are seven such predictors which are now removed from the Training and Test Data files

8. Model Development - Selection Of Algorithm

Partition the training data into a training set and a testing/validation set.

```
inTrain     <- createDataPartition(Training.df$classe, p = 0.6, list = FALSE)
inTraining  <- Training.df[inTrain,]
inTest      <- Training.df[-inTrain,]
dim(inTraining);dim(inTest)

## [1] 11776    53

## [1] 7846   53

myModelFilename <- "myModel.RData"
if (!file.exists(myModelFilename)) {

 # Using Parallel cores and Using doParallel Library for Faster Execution

    library(doParallel)
    ncores <- makeCluster(detectCores() - 1)
    registerDoParallel(cores=ncores)
    getDoParWorkers() # 3

 # Use Random Forest method with Cross Validation, 4 folds
    myModel <- train(classe ~ .
```

```
                , data = inTraining
                , method = "rf"

#Categorical outcome variable so choosing accuracy
                , metric = "Accuracy"

                , preProcess=c("center", "scale")

# attempt to improve accuracy by normalising
                , trControl=trainControl(method = "cv"
                                         , number = 4

# folds of the training data
                                         , p= 0.60
                                         , allowParallel = TRUE
#                                        , seeds=NA

# don't let workers set seed
                                         )
                )

    save(myModel, file = "myModel.RData")

    stopCluster(ncores)
} else {
# Use cached model
    load(file = myModelFilename, verbose = TRUE)
}

## Loading objects:
##   myModel

print(myModel, digits=5)

## Random Forest
##
## 11776 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (52), scaled (52)
## Resampling: Cross-Validated (4 fold)
## Summary of sample sizes: 8831, 8831, 8834, 8832
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    0.98718   0.98378
##   27    0.98820   0.98507
##   52    0.98166   0.97679
##
```

```
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

9. Prediction

Predicting the activity performed using the training file derived as test data.

```
predTest <- predict(myModel, newdata=inTest)
```

10. Final Model Evaluation and Selection

```
confusionMatrix(predTest, inTest$classe)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    7    0    1    0
##          B    2 1509    6    1    0
##          C    0    2 1359    8    5
##          D    0    0    3 1276    6
##          E    1    0    0    0 1431
##
## Overall Statistics
##
##                Accuracy : 0.9946
##                  95% CI : (0.9928, 0.9961)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9932
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9941   0.9934   0.9922   0.9924
## Specificity            0.9986   0.9986   0.9977   0.9986   0.9998
## Pos Pred Value         0.9964   0.9941   0.9891   0.9930   0.9993
## Neg Pred Value         0.9995   0.9986   0.9986   0.9985   0.9983
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1923   0.1732   0.1626   0.1824
## Detection Prevalence   0.2851   0.1935   0.1751   0.1638   0.1825
## Balanced Accuracy      0.9986   0.9963   0.9956   0.9954   0.9961
```

Out of Sample Error for the model is 1 - Acuracy = 1 - 0.9946 = 0.0054 which is very small or equal to 0.54%. This accuracy level is within the 95% confidence interval <

Accuracy is very high, at 0.9963, and this figure lies within the 95% confidence interval (0.9928, 0.9961).

11. Summary of Final Model

```
myModel$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: 27
##
##         OOB estimate of  error rate: 0.86%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3344    2    1    0    1 0.001194743
## B   20 2251    6    2    0 0.012286090
## C    0   14 2031    9    0 0.011197663
## D    0    1   29 1897    3 0.017098446
## E    0    2    4    7 2152 0.006004619
```

12. Choosing the Most Important Variables

```
varImp(myModel)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 52)
##
##                      Overall
## roll_belt            100.000
## pitch_forearm         61.314
## yaw_belt              54.425
## pitch_belt            44.937
## magnet_dumbbell_z     42.705
## magnet_dumbbell_y     42.677
## roll_forearm          40.036
## accel_dumbbell_y      23.081
## magnet_dumbbell_x     18.778
## roll_dumbbell         18.585
## accel_forearm_x       16.913
## magnet_belt_z         16.052
## accel_dumbbell_z      14.119
## magnet_forearm_z      13.891
## magnet_belt_y         13.496
## total_accel_dumbbell  12.884
```

```
## accel_belt_z          12.413
## gyros_belt_z          11.382
## yaw_arm               10.311
## magnet_belt_x          9.237
```

13. Validation of Model Through Quiz

```
print(predict(myModel, newdata=Quiz.df))
```

```
##  [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
```

These answers are entered in the quiz for this assigment on the course website which accepted these values as correct to pass.

14. References

1. Velloso, E., Butling, A., Vgolino, W. Fuks,H., " Quantitative Activity/Recognition of Weight Lifting Exercises", Proc. Of 4th International Conference in Cooperation with SIGCHI, (Augmented Human'13) Stuttgart, Germany: ACM SIGCHI, 2013.