Weight Lifting Predictive Model

Samantha_Baron

January 26, 2019

Executive Summary

An experiement was designed where subjects wore sensors and did bicep curls in 5 different manners, one using correct form and the others being incorrect form. The idea was to see if the quality of the exercise could be assessed without the need for a live trainer to be present.

Data was taken from all of the sensors and other variables were derived fromthe direct sensor data. This report will attempt to determine a prediction model such that one could classify in which of the 5 manners the exercise was done from the sensor data and thereby provide feedback to the user about the quality of their exercising.

```
#Data URLs - for Reference
training.url    <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
test.cases.url  <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

#read downloaded data from working directory
train <- read.csv("./pml-training.csv", na.strings=c("NA","#DIV/0!",""))
test <- read.csv("./pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
```

Preprocessing the data

The data first needed to be cleaned before a model could be buit on it. The steps below outline how this was accomplished: 1. Remove all the columns where the % of NA values is greater than 0 2. Remove the demographic fields like user name and the fields that give information about how the test was performed like window and time _stamps. These are columns 1:7 3. Do the same processing on the training and testing set 4. Partition the training set to get a validation set to test on for purposes of determining which model is the most accurate.

```
#Useful class forum discussion on cleaning/ pruning variables

na_count <- data.frame(x = colSums(is.na(train))/(nrow(train)))
na_remove <- which(na_count$x>0)
c.to.remove <- c(1:7, na_remove)
train <- train[,-c.to.remove] #with non-variable data removed
test <- test[,-c.to.remove]

#create validation set from training set
inTrain <- createDataPartition(y = train$classe, p=0.75, list=FALSE)
training <- train[inTrain,]
validation <- train[-inTrain,]
```

Model 1 - Random Forest

The first model tested will use the random forest technique.

```
set.seed(777)

#Configure parallel processing (per article: #https://github.com/lgreski/datasciencectacontent/blob/master/markdown/pml-rand
omForestPerformance.md)
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)

#create model
fitControlRF <- trainControl(method = "cv", number = 5, allowParallel = TRUE)

modFitRF <- train(classe ~ ., method="rf", data = training, trControl=fitControlRF)

#De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()

#predict on validation set
predModRF <- predict(modFitRF, validation[,-53])
RF <- confusionMatrix(validation$classe, predModRF)
```

Model 2 - Boosting

The second model tested will use the boosting technique.

```
set.seed(777)

#Configure parallel processing
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)

#create model
fitControlGBM <- trainControl(method = "cv", number = 5, allowParallel = TRUE)

modFitGBM <- train(classe ~ ., method="gbm", data = training, trControl=fitControlGBM, verbose = FALSE)

#De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()

#predict on validation set
predModGBM <- predict(modFitGBM, validation[,-53])
GBM <- confusionMatrix(validation$classe, predModGBM)
```

Model 3 - Linear Discriminant Analysis

The third model tested will use the linear discriminant analysis technique.

```
set.seed(777)

#Configure parallel processing
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)

#create model
fitControlLDA <- trainControl(method = "cv", number = 5, allowParallel = TRUE)

modFitLDA <- train(classe ~ ., method="lda", data = training, trControl=fitControlLDA)

#De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()

#predict on validation set
predModLDA <- predict(modFitLDA, validation[,-53])
LDA <- confusionMatrix(validation$classe, predModLDA)
```

Results and Analysis

As seen in Table 1, which shows the accuracy and the out of sample error for the three modeling methods tried, random forest appears to have the best predictive results with the lowest out of sample error.

Out of sample error was calculated for the validation set as [1-accuracy].

```
Acc <- rbind("LDA" = LDA$overall["Accuracy"], "GBM" = GBM$overall["Accuracy"], "RF" = RF$overall["Accuracy"])

OutSamError <- 100*(1-Acc)
colnames(OutSamError) <- c("OutSampleError")

Results <- round(cbind(100*Acc, OutSamError), 2)

kable(Results, caption = "Table 1")
```

Table 1

	Accuracy	OutSampleError
LDA	70.35	29.65
GBM	95.96	4.04
RF	99.37	0.63

Data Exploration

It is interesting to note that the three different modeling methods each chose slightly different variables to be the most important, although each weighted magnet_dumbell_z and y and the pitch_forearm heavily.

```
varLDA <- as.matrix(varImp(modFitLDA)$importance)
varGBM <- as.matrix(varImp(modFitGBM)$importance)
varRF <- as.matrix(varImp(modFitRF)$importance)

varimp <- varimp <- merge(varRF, varGBM, by = "row.names", all = TRUE)

rownames(varimp) <- varimp[,1]
varimp <- varimp[,-1]
varimp <- merge(varimp, varLDA, by="row.names", all=TRUE)
colnames(varimp) <- c("variable", "RF", "GBM", "LDA-A", "LDA-B","LDA-C","LDA-D", "LDA-E")
varimp <- tbl_df(varimp)
varimp <- arrange(varimp, desc(RF), desc(GBM))

varimp[,2:8] <- round(varimp[,2:8], 1)
kable(varimp, caption = "Table 2")
```

Table 2

variable	RF	GBM	LDA-A	LDA-B	LDA-C	LDA-D	LDA-E
roll_belt	100.0	100.0	13.1	7.6	44.4	7.6	13.1
pitch_forearm	60.0	49.5	61.3	100.0	67.1	61.3	100.0
yaw_belt	57.4	58.4	12.0	12.0	16.2	18.3	5.4
pitch_belt	45.8	27.6	8.4	5.1	9.7	5.1	8.4
magnet_dumbbell_y	45.1	29.6	46.7	46.7	46.7	66.6	45.6
magnet_dumbbell_z	44.5	40.6	54.7	35.9	53.5	24.3	54.7
roll_forearm	42.6	30.8	34.0	6.9	13.6	23.6	34.0
accel_dumbbell_y	22.9	11.6	37.2	16.3	16.3	42.7	37.2
accel_forearm_x	17.9	15.7	39.4	78.5	39.4	39.4	78.5
roll_dumbbell	17.5	15.0	43.0	50.3	30.6	63.8	50.3
magnet_dumbbell_x	16.5	7.9	65.2	65.2	65.2	65.2	51.5
magnet_belt_z	15.0	23.0	3.3	1.8	51.6	2.9	3.3
accel_dumbbell_z	14.6	6.2	43.2	43.2	43.2	43.2	18.5
total_accel_dumbbell	14.3	5.1	8.6	17.7	8.6	20.6	17.7
accel_belt_z	14.0	6.6	19.2	15.9	39.0	15.9	19.2
magnet_forearm_z	13.8	14.0	17.6	15.8	12.2	14.0	17.6
magnet_belt_y	13.3	5.9	17.3	9.8	67.5	11.3	17.3
yaw_arm	11.5	10.5	16.1	16.1	16.1	16.1	13.5
gyros_belt_z	11.0	14.9	7.9	18.1	8.9	7.5	18.1
magnet_belt_x	10.4	7.4	26.3	26.3	26.3	26.3	20.8

variable	RF	GBM	LDA-A	LDA-B	LDA-C	LDA-D	LDA-E
roll_arm	9.5	4.7	34.1	34.1	34.1	34.1	24.4
yaw_dumbbell	8.8	1.9	20.5	20.5	20.5	40.0	11.8
accel_forearm_z	8.1	10.5	7.1	7.1	7.1	9.9	6.5
gyros_dumbbell_y	7.5	9.4	8.4	13.5	8.9	12.1	13.5
magnet_forearm_y	6.7	2.1	18.6	36.6	27.6	24.6	36.6
accel_arm_x	6.6	3.4	49.8	73.5	61.2	49.8	73.5
magnet_forearm_x	5.9	4.8	37.5	69.5	32.2	32.2	69.5
magnet_arm_y	5.9	5.9	38.2	76.2	67.5	38.2	76.2
accel_dumbbell_x	5.7	6.9	57.4	57.4	57.4	57.4	48.1
yaw_forearm	5.7	0.9	9.3	12.8	10.7	21.8	12.8
magnet_arm_x	5.6	4.6	51.9	78.2	64.5	51.9	78.2
magnet_arm_z	5.3	8.9	51.9	51.9	51.9	51.9	39.9
pitch_arm	4.8	1.5	27.1	43.0	49.2	27.1	43.0
gyros_arm_y	3.9	4.0	7.4	7.4	8.2	7.4	7.2
pitch_dumbbell	3.8	2.8	52.7	52.7	52.7	70.0	43.6
accel_forearm_y	3.6	2.2	29.9	2.8	2.8	25.1	29.9
accel_arm_y	3.0	1.0	11.9	16.4	23.7	11.9	16.4
gyros_dumbbell_x	2.7	2.5	3.0	2.3	2.8	7.5	3.0
gyros_arm_x	2.5	1.7	6.9	4.8	6.3	4.8	6.9
gyros_forearm_y	2.4	0.5	4.1	4.1	4.1	4.1	0.4
accel_arm_z	2.3	1.5	31.0	28.6	11.9	25.6	31.0
total_accel_belt	1.9	1.2	6.4	8.7	20.3	8.7	8.7
total_accel_arm	1.8	0.9	31.7	38.8	31.3	18.3	38.8
gyros_belt_y	1.8	3.3	3.5	7.0	1.0	4.9	7.0
total_accel_forearm	1.7	2.3	23.4	26.2	31.8	23.4	26.2
gyros_belt_x	1.5	1.2	6.2	6.2	6.2	6.2	0.9
gyros_dumbbell_z	1.1	1.1	5.1	5.1	5.1	5.1	2.9
accel_belt_y	1.0	0.0	7.0	7.0	7.0	7.0	5.6
gyros_forearm_z	1.0	1.3	7.2	7.2	7.2	7.2	3.9
accel_belt_x	0.8	0.6	11.1	10.7	15.4	8.8	11.1
gyros_forearm_x	0.3	0.6	2.5	6.9	3.3	6.2	6.9
gyros_arm_z	0.0	0.1	2.3	0.0	3.4	0.0	2.3

Results of applying the model to the test data

The random forest model was applied to the test data. The results can be found in Table 3.

```
PredTestRF <- predict(modFitRF, test)
scenario <- c(1:20)
TestResults <- cbind(scenario, results = as.character(PredTestRF))
kable(TestResults, caption = "Table 3")
```

Table 3

scenario	results
1	B
2	A
3	B
4	A
5	A
6	E
7	D
8	B
9	A
10	A
11	B
12	C
13	B
14	A
15	E
16	E
17	A
18	B
19	B
20	B