

# Machine Learning Peer Graded Assignment

Stephen Armah

August 9, 2016

## Objective

This assignment tasks me with predicting how well participants performed various fitness activities. To do so, I will load the training and testing data, perform exploratory analysis, build models and select the best one, and apply it to the testing data.

The first activity is to load appropriate libraries and the data sets.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.2.5
```

```
library(survival)
library(splines)
library(parallel)
library(plyr)
library(MASS)
set.seed(8675309)

## this is a large data set so clear memory
rm(list=ls())

## Load data
training <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.strings = "#DIV/0!")
testing  <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", na.strings = "#DIV/0!")
```

## Exploratory Analysis

Now we perform exploratory analysis on the training data set for insight into how to proceed.

```
summary(training$skewness_yaw_belt)
```

```
##      Mode      NA's  
## logical 19622
```

A summary of the training data set reveals that a great many variables have a prohibitive number of null or NA values. The `skewness_yaw_belt` variables (above) is an example. To improve model quality and expedite computations, I subset the data set and remove columns that have greater than 90% NA values.

## Data Clensing

```
## remove columns that are not measurements  
training <- training[, -(1:5)]  
testing  <- testing[, -(1:5)]  
  
## determine ratio of NAs to data in each column  
x <- 0  
for(i in 1:length(training)) {  
  x[i] <- print(sum(is.na(training[,i]))/nrow(training))  
}  
  
## remove columns with NAs > 90%  
training <- subset(training, select = x < 0.90)  
testing  <- subset(testing, select = x < 0.90)  
  
## remove columns with very low variances  
noVar <- nearZeroVar(training)  
training <- training[, -noVar]  
testing  <- testing[, -noVar]
```

## Cross Validation

Subsample training into `subTrain` and `subTest` data sets. Model on `subTrain`, evaluate on `subTest`. Find best model and apply to testing

```
## Sub-sample the training data set into a sub training and testing data set  
inTrain <- createDataPartition(y=training$classe, p=0.75, list=FALSE)  
subTrain <- training[inTrain,]  
subTest  <- training[-inTrain,]
```

## Building Models

I fit multiple models to the data to determine the one with the highest accuracy.

```
## random forest  
controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)  
modRF <- train(subTrain$classe~., method="rf", data=subTrain, trControl=controlRF)
```

```
## Loading required package: randomForest
```

```
## Warning: package 'randomForest' was built under R version 3.2.5
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##  
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':  
##  
##     margin
```

```
outRF  <- predict(modRF, subTest)  
conRF  <- confusionMatrix(subTest$classe, outRF)  
  
## decision tree  
modRP  <- train(subTrain$classe~., method="rpart", data=subTrain)
```

```
## Loading required package: rpart
```

```
outRP  <- predict(modRP, subTest)  
conRP  <- confusionMatrix(subTest$classe, outRP)  
  
## LDA  
modLDA <- train(subTrain$classe~., method="lda", data=subTrain)  
outLDA <- predict(modLDA, subTest)  
conLDA <- confusionMatrix(subTest$classe, outLDA)  
  
## generalized booster  
modGBM <- train(subTrain$classe~., method="gbm", data=subTrain, verbose=FALSE)  
outGBM <- predict(modGBM, subTest)  
conGBM <- confusionMatrix(subTest$classe, outGBM)
```

We now review the accuracy for each model tested.

```
conRF$overall[1]
```

```
## Accuracy  
## 0.9969413
```

```
conRP$overall[1]
```

```
## Accuracy  
## 0.487969
```

```
conLDA$overall[1]
```

```
## Accuracy  
## 0.7077896
```

```
conGBM$overall[1]
```

```
## Accuracy  
## 0.9883768
```

## Conclusion

The Random Forest model provided the best accuracy so I will apply RF method to the 20 observations in the testing data set.