Practical Machine Learning Coursera Peer graded Assignment

### Data Description

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement â a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website [here:](http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

### Data

The raw data is split between training and validation sets. All columns that contain NA and features that are not in the testing dataset are removed. The features containing NA are the variance, mean and standard devition (SD) within each window for each feature. Since the testing dataset has no time-dependence, these values are useless and can be disregarded. We will also remove the first 7 features since they are related to the time-series or are not numeric.

The training data for this project are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv>

The test data are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv>

### setting work environment and knitr options

rm(list=ls(all=TRUE)) #start with empty workspace  
startTime <- Sys.time()  
  
library(knitr)  
opts\_chunk$set(echo = TRUE, cache= TRUE, results = 'hold')

### Load Libraries

Load all libraries used, and setting seed for reproducibility.

library(ElemStatLearn)  
library(caret)  
library(rpart)  
library(randomForest)  
library(RCurl)  
set.seed(1)

### Load and prepare the data and clean up the data

Load and prepare the data

trainingLink <- getURL("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")  
pml\_CSV <- read.csv(text = trainingLink, header=TRUE, sep=",", na.strings=c("NA",""))  
  
pml\_CSV <- pml\_CSV[,-1] # Remove the first column that represents a ID Row

### Data Sets Partitions Definitions

Create data partitions of training and validating data sets.The datais split into training data set (60% of the total cases) and testing data set (40% of the total cases). This allows to estimate the out of sample error of our predictor.

inTrain = createDataPartition(pml\_CSV$classe, p=0.60, list=FALSE)  
training = pml\_CSV[inTrain,]  
validating = pml\_CSV[-inTrain,]  
  
# number of rows and columns of data in the training set  
  
dim(training)  
  
# number of rows and columns of data in the validating set  
  
dim(validating)

## [1] 11776 159  
## [1] 7846 159

## Data Exploration and Cleaning

Since we choose a random forest model and we have a data set with too many columns, first we check if we have any problems with columns without data. Remove the columns having less than 60% of data entered.

# Number of cols with less than 60% of data  
sum((colSums(!is.na(training[,-ncol(training)])) < 0.6\*nrow(training)))

[1] 100

# apply our definition of remove columns that most doesn't have data, before its apply to the model.  
  
Keep <- c((colSums(!is.na(training[,-ncol(training)])) >= 0.6\*nrow(training)))  
training <- training[,Keep]  
validating <- validating[,Keep]  
  
# number of rows and columns of data in the final training set  
  
dim(training)

[1] 11776 59

# number of rows and columns of data in the final validating set  
  
dim(validating)

[1] 7846 59

## Modeling

In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the execution. So, we proceed with the training the model (Random Forest) with the training data set.

model <- randomForest(classe~.,data=training)  
print(model)

##   
## Call:  
## randomForest(formula = classe ~ ., data = training)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 7  
##   
## OOB estimate of error rate: 0.16%  
## Confusion matrix:  
## A B C D E class.error  
## A 3347 1 0 0 0 0.0002986858  
## B 1 2277 1 0 0 0.0008775779  
## C 0 3 2049 2 0 0.0024342746  
## D 0 0 6 1923 1 0.0036269430  
## E 0 0 0 4 2161 0.0018475751

### Model Evaluation

Proceeding with the verification of variable importance measures as produced by random Forest:

importance(model)

## MeanDecreaseGini  
## user\_name 103.1758200  
## raw\_timestamp\_part\_1 951.9215402  
## raw\_timestamp\_part\_2 10.0854476  
## cvtd\_timestamp 1442.9618905  
## new\_window 0.1746792  
## num\_window 556.0642674  
## roll\_belt 519.5984019  
## pitch\_belt 295.8388795  
## yaw\_belt 343.1031813  
## total\_accel\_belt 108.1028950  
## gyros\_belt\_x 37.3204300  
## gyros\_belt\_y 51.3852345  
## gyros\_belt\_z 126.1669798  
## accel\_belt\_x 60.3460091  
## accel\_belt\_y 65.5411224  
## accel\_belt\_z 197.7011375  
## magnet\_belt\_x 101.8091441  
## magnet\_belt\_y 188.7481735  
## magnet\_belt\_z 179.9940893  
## roll\_arm 118.3456397  
## pitch\_arm 56.4748726  
## yaw\_arm 84.1620730  
## total\_accel\_arm 27.6244838  
## gyros\_arm\_x 40.4711618  
## gyros\_arm\_y 43.1011593  
## gyros\_arm\_z 19.8475271  
## accel\_arm\_x 86.5018861  
## accel\_arm\_y 49.9395312  
## accel\_arm\_z 39.5980688  
## magnet\_arm\_x 96.0615670  
## magnet\_arm\_y 77.0549967  
## magnet\_arm\_z 56.6725706  
## roll\_dumbbell 188.9235619  
## pitch\_dumbbell 79.9014619  
## yaw\_dumbbell 108.4398245  
## total\_accel\_dumbbell 117.1885758  
## gyros\_dumbbell\_x 42.1855235  
## gyros\_dumbbell\_y 91.3055337  
## gyros\_dumbbell\_z 24.7340768  
## accel\_dumbbell\_x 128.3694807  
## accel\_dumbbell\_y 194.3282275  
## accel\_dumbbell\_z 131.7890004  
## magnet\_dumbbell\_x 230.9813626  
## magnet\_dumbbell\_y 313.3695030  
## magnet\_dumbbell\_z 298.3814591  
## roll\_forearm 235.4421018  
## pitch\_forearm 303.5713241  
## yaw\_forearm 55.0947271  
## total\_accel\_forearm 33.3861279  
## gyros\_forearm\_x 23.0332691  
## gyros\_forearm\_y 39.7280706  
## gyros\_forearm\_z 25.5679380  
## accel\_forearm\_x 140.7269530  
## accel\_forearm\_y 42.1179735  
## accel\_forearm\_z 89.0866390  
## magnet\_forearm\_x 71.1284614  
## magnet\_forearm\_y 73.8414658  
## magnet\_forearm\_z 91.4716222

Now we evaluate our model results through confusion Matrix.

confusionMatrix(predict(model,newdata=validating[,-ncol(validating)]),validating$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2232 1 0 0 0  
## B 0 1517 2 0 0  
## C 0 0 1366 0 0  
## D 0 0 0 1286 4  
## E 0 0 0 0 1438  
##   
## Overall Statistics  
##   
## Accuracy : 0.9991   
## 95% CI : (0.9982, 0.9996)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9989   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 0.9993 0.9985 1.0000 0.9972  
## Specificity 0.9998 0.9997 1.0000 0.9994 1.0000  
## Pos Pred Value 0.9996 0.9987 1.0000 0.9969 1.0000  
## Neg Pred Value 1.0000 0.9998 0.9997 1.0000 0.9994  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2845 0.1933 0.1741 0.1639 0.1833  
## Detection Prevalence 0.2846 0.1936 0.1741 0.1644 0.1833  
## Balanced Accuracy 0.9999 0.9995 0.9993 0.9997 0.9986

And confirmed the accuracy at validating data set by calculate it with the formula:

accuracy <-c(as.numeric(predict(model,newdata=validating[,-ncol(validating)])==validating$classe))  
  
accuracy <-sum(accuracy)\*100/nrow(validating)

Model Accuracy as tested over Validation set = **99.9%**.

### Model Test

Finally, we proceed with predicting the new values in the testing csv provided, first we apply the same data cleaning operations on it and coerce all columns of testing data set for the same class of previous data set.

#### Loading the Test Dataset

testingLink <- getURL("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")  
pml\_CSV <- read.csv(text = testingLink, header=TRUE, sep=",", na.strings=c("NA",""))  
  
pml\_CSV <- pml\_CSV[,-1] # Remove the first column that represents a ID Row  
pml\_CSV <- pml\_CSV[ , Keep] # Keep the same columns of testing dataset  
pml\_CSV <- pml\_CSV[,-ncol(pml\_CSV)] # Remove the problem ID  
  
# Apply the Same Transformations and Coerce Testing Dataset  
  
# Coerce testing dataset to same class and strucuture of training dataset   
testing <- rbind(training[100, -59] , pml\_CSV)   
# Apply the ID Row to row.names and 100 for dummy row from testing dataset   
row.names(testing) <- c(100, 1:20)

#### Predicting with Test Dataset

As a last step in the project, I’ll use the validation data sample (‘pml-testing.csv’) to predict a classe for each of the 20 observations based on the other information we know about these observations contained in the validation sample.

predictions <- predict(model,newdata=testing[-1,])  
print(predictions)

## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20   
## 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