Practical\_Machine\_Learning Project

## 1. Executive Summary

This document presents the results of the Practical Machine Learning Peer Assessments in a report using a single R markdown document.

Since I have a data set with to many columns(160) and it needs to make a class prediction, I select a random forests model as a learning method, that’s no need cross-validation or a separate test set to get an unbiased estimate of the test set error. Before apply the dataset to this prediction model, I decide remove all the columns that having less than 80% of data filled, instead try to filled it with some center measure. This model accuracy over validation dataset is equal to 99.9235%. This model promoted a excelente prediction results with testing dataset and generated the 20th files answers to submit for the Assignments.

## 2. Problem explanation

The goal of this project is to :

1. predict the manner in which they did the exercise.
2. This is the "classe" variable in the training set. All other variables can be use as predictor.
3. It needs to be described how to built prediction model, how to use cross validation, what sample error is expected out, and why the choices was maded.
4. It will be used also use prediction model to predict 20 different test cases.

#### Dataset - HAR

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>

The data for this project come from this source: <http://groupware.les.inf.puc-rio.br/har>. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

## 2. Data Analysis

#### 2.1 Exploratoty Data Analysis

##### Data Loading and Cleaning

library(caret)  
library(rpart)  
library(randomForest)  
library(Formula)  
# data loading  
training<-read.csv("pml-training.csv",na.strings=c("NA",""))  
training<-training[,-1] # Remove the first column that represents a ID Row

* This data have total 159 variables.
* But several vairabls do not have values(missing values). First we check if we have many problems with columns without data.

##### Number of columns with less than 90% of data

sum((colSums(!is.na(training[,-ncol(training)])) < 0.9\*nrow(training)))

## [1] 100

* So, we 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)])) >=0.9\*nrow(training)))  
training <- training[,Keep]

* New data have 59 variables.

##### Data Sets Partitions Definitions

* Create the data partitions of training and validating data sets from original training data sets.

set.seed(357)  
inTrain = createDataPartition(training$classe, p=0.60, list=FALSE)  
validating = training[-inTrain,]  
training = training[inTrain,]

#### 2.2 Data Modeling

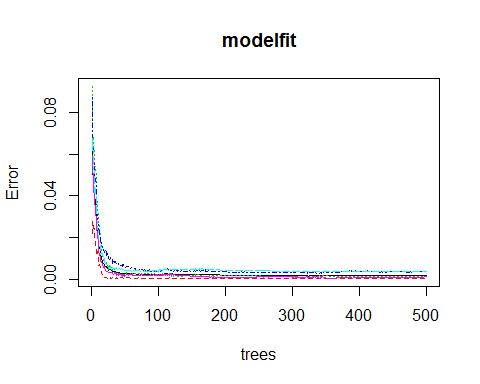
I select random forests as a modeling of this project. In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the execution. So, I proced with the training the model (Random Forest) with the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

Due to extremely long running time with caret’s train function(1 hour>~), randomForest of randomForest package are chosen(~1 min) for this project.

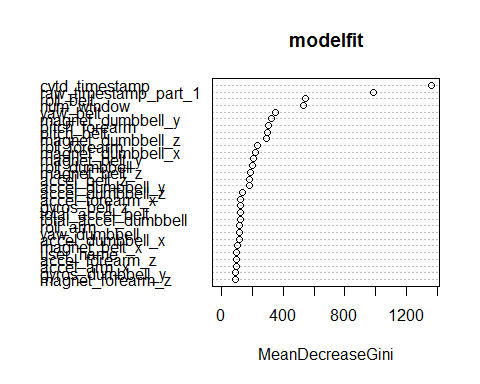
modelfit <- randomForest(classe~.,data=training)  
##modelfit <- train(classe~.,data=training,method="rf",prox=TRUE )  
print(modelfit)

##   
## 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: 0.18%  
## Confusion matrix:  
## A B C D E class.error  
## A 3347 1 0 0 0 0.0002987  
## B 2 2276 1 0 0 0.0013164  
## C 0 4 2047 3 0 0.0034080  
## D 0 0 6 1923 1 0.0036269  
## E 0 0 0 3 2162 0.0013857

plot(modelfit)



#importance(modelfit)  
varImpPlot(modelfit)



You can see figure with the verification of variable importance measures as produced by random Forest.

confusionMatrix(predict(modelfit,validating[,-ncol(validating)]),validating$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2232 0 0 0 0  
## B 0 1518 0 0 0  
## C 0 0 1367 5 0  
## D 0 0 1 1281 0  
## E 0 0 0 0 1442  
##   
## Overall Statistics  
##   
## Accuracy : 0.999   
## 95% CI : (0.998, 1)  
## No Information Rate : 0.284   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.999   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.000 1.000 0.999 0.996 1.000  
## Specificity 1.000 1.000 0.999 1.000 1.000  
## Pos Pred Value 1.000 1.000 0.996 0.999 1.000  
## Neg Pred Value 1.000 1.000 1.000 0.999 1.000  
## Prevalence 0.284 0.193 0.174 0.164 0.184  
## Detection Rate 0.284 0.193 0.174 0.163 0.184  
## Detection Prevalence 0.284 0.193 0.175 0.163 0.184  
## Balanced Accuracy 1.000 1.000 0.999 0.998 1.000

I confirmed the accurancy at validating data set by calculate it with the following formula:

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

Model Accuracy as tested over Validation set = 99.9235%.

## 3. Model Test

Finaly, I prossed with predicting the new values in the testing csv provided, first I apply the same data cleannig operations on it and coerce all columns of Testing data set for the same class of previuos data set.

#### 3.1 Apply the Same Transformations between testing and training data

testing<-read.csv("pml-testing.csv",na.strings=c("NA",""))  
testing<-testing[,-1]  
testing <- testing[,Keep]  
testing<- testing[,-ncol(testing)] #remove problem ID  
  
# Coerce testing dataset to same class and strucuture of training dataset   
class\_check <- (sapply(testing, class) == sapply(training, class))  
testing[, !class\_check] <- sapply(testing[, !class\_check], as.numeric)  
testing <- rbind(training[100, -59] ,testing)   
testing <-testing[-1,]

#### 3.2 Prediciting with Testing Dataset

predictions<-predict(modelfit,testing)  
print(predictions)

## 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21   
## 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

#### 3.3 Generating Answers Files to Submit for Assignment

pml\_write\_files = function(x){  
 n = length(x)  
 for(i in 1:n){  
 filename = paste0("problem\_id\_",i,".txt")  
 write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)  
 }  
}  
pml\_write\_files(predictions)

Since the greatest accuracy level of my model, as expect, all 20th files answers submitted were correct!