**Machine Learning Project from Coursera**

*Bigyan Khanal*

**Introduction to the project**

This project is about the predicting the manner in which they did the exercise based on the historical data obtained from the different accelerometers.

The human activity recognition research is focused on identifying different activities, i.e. to predict "which" activity was performed at a specific point in time. The approach is done in order to identify how well a Weight Lifting Exercises is performed by the wearer based on the data collected [1].

In this work, six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions:

Class A: Exactly according to the specification

Class B: Throwing the elbows to the front

Class C: Lifting the dumbbell only halfway

Class D: Lowering the dumbbell only halfway

Class E: Throwing the hips to the front.

For the prediction of the test set, we need to import the data, process it, clean the data, split the data, train the predictor algorithm, validate it and use the prediction on the testing data. For prediction algorithm, I have followed the random forest algorithm since it uses the multiple decision trees to get the decision and are accurate.

**Importing of the data**

The training and test data are available at the website as:

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

Some required library R package are loaded:

## Packages required

library (caret) # classification and regression training

library (randomForest) # for use of random forest

library(AppliedPredictiveModeling) # Functions and Data Sets for 'Applied Predictive Modeling'

After downloading the data in .csv format they were read to R as training\_data and testing\_data and given the required column names and tested for the similarity in column names of this two data set which was found to be true.

#Data Import

setwd("D:/BIGYAN/Coursera/Machine\_learning")

training\_data<-read.csv("pml-training.csv", na.strings=c("NA",""), header=TRUE)

colnames\_training <- colnames(training\_data)

testing\_data<-read.csv("pml-testing.csv", na.strings=c("NA",""), header=TRUE)

colnames\_testing <- colnames(testing\_data)

all\_training\_colnames<-colnames\_training[1:length(colnames\_training)-1]

all\_testing\_colnames<-colnames\_testing[1:length(colnames\_testing)-1]

all.equal(all\_training\_colnames, all\_testing\_colnames)

[[TRUE]] # TRUE indicate the train data and test data are in same format

**Cleaning the data**

The data seems to have too many missing values replaced with NAs from the above code so, the columns which NAs are removed and the first some variables are least significant to the prediction of classe so they are delated. This way data is cleaned from its raw structure.

# Cleaning the data

training\_data <- training\_data[, colSums(is.na(training\_data)) == 0]

testing\_data <- testing\_data[, colSums(is.na(testing\_data)) == 0]

# cleaning the data for those columns which are least significant to the model

Final\_train\_Data<- training\_data[, -c(1:7)]

dim(Final\_train\_Data)

Final\_test\_Data <- testing\_data[, -c(1:7)]

dim(Final\_test\_Data)

Finally, the number of columns (i.e. 23) were same in training and test data.

**Cutting the data**

It becomes necessary to cut the data into two parts one to build the model based on the training set and the remaining part is used for the validation of the model. For my case, I have used ¾th of the data for building model and 1/4th for the validation purpose.

#Splitting the data

set.seed(33433)

inTrain <- createDataPartition(Final\_train\_Data$classe, p = 0.75, list = FALSE)

Train\_data <- Final\_train\_Data[inTrain, ]

Validation\_data <- Final\_train\_Data[-inTrain, ]

**Model Building**

For model building, the random forest algorithm was chosen. Prior to applying random forest, we define the fit control as the 5 fold cross validation.

# Model building using Random Forest

fitControl <- trainControl(method="cv", number=5, verboseIter=F)

model\_fit <- train(classe ~ ., data=Train\_data, method="rf", trControl=fitControl)

print(model\_fit)

model\_fit$finalModel

With these code, the output as the model fit is obtained as:

Random Forest

14718 samples

52 predictor

5 classes: 'A', 'B', 'C', 'D', 'E'

No pre-processing

Resampling: Cross-Validated (5 fold)

Summary of sample sizes: 11773, 11774, 11775, 11775, 11775

Resampling results across tuning parameters:

mtry Accuracy Kappa

2 0.9917789 0.9895994

27 0.9921184 0.9900296

52 0.9872261 0.9838403

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 27.

Observing the final model of the model\_fit we get the output as:

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: 0.59%

Confusion matrix:

A B C D E class.error

A 4182 2 0 0 1 0.0007168459

B 17 2827 4 0 0 0.0073735955

C 0 12 2546 9 0 0.0081807557

D 0 1 27 2383 1 0.0120232172

E 0 2 4 7 2693 0.0048041390

**Model Evaluation:**

We then evaluate the model based on the prediction function from the package and use the fitted model to predict the label (“classe”) in Train\_data, and show the confusion matrix to compare the predicted versus the actual labels. The code for the prediction along with the confusion matrix is shown as:

#Model Evaluation

preds <- predict(model\_fit, newdata=Validation\_data)

confusionMatrix(Validation\_data$classe, preds)

And the confusion matrix obtained for the classe of the validation data is obtained as:

Confusion Matrix and Statistics

Reference

Prediction A B C D E

A 1394 1 0 0 0

B 8 941 0 0 0

C 0 9 844 2 0

D 0 0 7 797 0

E 0 3 0 3 895

Overall Statistics

Accuracy : 0.9933

95% CI : (0.9906, 0.9954)

No Information Rate : 0.2859

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9915

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: A Class: B Class: C Class: D Class: E

Sensitivity 0.9943 0.9864 0.9918 0.9938 1.0000

Specificity 0.9997 0.9980 0.9973 0.9983 0.9985

Pos Pred Value 0.9993 0.9916 0.9871 0.9913 0.9933

Neg Pred Value 0.9977 0.9967 0.9983 0.9988 1.0000

Prevalence 0.2859 0.1945 0.1735 0.1635 0.1825

Detection Rate 0.2843 0.1919 0.1721 0.1625 0.1825

Detection Prevalence 0.2845 0.1935 0.1743 0.1639 0.1837

Balanced Accuracy 0.9970 0.9922 0.9945 0.9960 0.9993

**Accuracy of the Model**

The accuracy of the model was tested with the first element of the confusion matrix with the help of following code:

#Accuracy of the model

accuracy\_rf <- confusion\_rf$overall[1]

accuracy\_rf

And the accuracy result was obtained as 0.9932708. The model was accurate because the random forests chooses a subset of predictors at each split and de-correlate the trees.

Accuracy

0.9932708

**Prediction on the final set**

The model developed was tested on the final test data using the following code:

#Prediction on test Set

predict(model\_fit, Final\_test\_Data)

And the result was found as :

[1] B A B A A E D B A A B C B A E E A B B B

**References**

[1] [Velloso, E.](http://groupware.les.inf.puc-rio.br/collaborator.jsf?p1=evelloso); Bulling, A.; Gellersen, H.; [Ugulino, W.](http://groupware.les.inf.puc-rio.br/collaborator.jsf?p1=ugulino); [Fuks, H.](http://groupware.les.inf.puc-rio.br/collaborator.jsf?p1=hugo) [Qualitative Activity Recognition of Weight Lifting Exercises](http://groupware.les.inf.puc-rio.br/work.jsf?p1=11201). Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013