```
# A. Practical Machine Learning Project : Prediction Assignment Writeup

# B. Background - https://www.coursera.org/learn/practical-machine-learning/peer/R43St/prediction-assignment-writeup

# C.--> Data Loading and Exploratory Analysis
  # a) Dataset Overview
    # 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 http://groupware.les.inf.puc-rio.br/har.

  #b) Environment Preparation - Load the necessary and relevant R libraries

     rm(list=ls())     # free up memory for the download of the data sets
     getwd()
     setwd("C:/R/MyProjects/CourseraML")
     library(knitr)
     library(caret)
     library(rpart)
     library(rpart.plot)
     library(rattle)
     library(randomForest)
     library(corrplot)
     library(RColorBrewer)
     set.seed(1234)
```

```
#c) Data Loading and Cleaning -

 #The next step is loading the dataset from the URL provided.

 #The training dataset is then partinioned in 2 to create a Training set

 #(70% of the data) for the modeling process and a Test set (with the remaining 30%) for the validations.

 #The testing dataset is not changed and will only be used for the quiz results generation.


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

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


 training <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))

 testing <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))


 # create a partition within the training dataset (2 parts of training set - 70:30 ratio)

 inTrain <- createDataPartition(training$classe, p=0.7, list=FALSE)

 myTraining <- training[inTrain, ]

 myTesting <- training[-inTrain, ]


 dim(myTraining)      #160 variables loaded from csv

 dim(myTesting)       #160 variables loaded from csv


 #Both created datasets have 160 variables. Those variables have plenty of NA,

 #that can be removed with the cleaning procedures below.

 #The Near Zero variance (NZV) variables are also removed and

 #the ID variables as well.


 # Remove variables with Nearly Zero Variance

 NZV <- nzv(myTraining)

 myTraining <- myTraining [ , -NZV]
```

```
        myTesting <- myTesting [ , -NZV]


        dim(myTraining)     #128 variables left

        dim(myTesting)      #128 variables left


        # Also remove variables that are mostly NA i.e remove variables with more than 95% NA

        MostlyNA    <- sapply(myTraining, function(x) mean(is.na(x) ) ) > 0.95

        myTraining <- myTraining[, MostlyNA==FALSE]

        myTesting  <- myTesting[, MostlyNA==FALSE]


        dim(myTraining)     #59 variables now

        dim(myTesting)      #59 variables now


        # remove identification only variables (columns 1 to 5)

        myTraining <- myTraining[, -(1:5)]

        myTesting  <- myTesting[, -(1:5)]


        dim(myTraining)    #Finally, 54 variables left for modelling

        dim(myTesting)     #Finally, 54 variables left for modelling



# D. Prediction Model Building

   #Three methods will be applied to model the regressions (in the Train dataset)

        #and the best one (with higher accuracy when applied to the Test dataset) will be used for the quiz predictions.

        #The methods are: (1)Decision Tree,(2) Generalized Boosted Model and (3)Random Forests


        set.seed(1234)
```

```
#Cross validation - Cross validation is done for each model with K = 3.

fitControl <- trainControl(method='cv', number = 3)

#1. Prediction with Decision Trees (using caret's train function)
model_cart <- train(classe ~ .,
                data= myTraining,
                trControl= fitControl,
                method= 'rpart'
                )
save(model_cart, file='./ModelFitCART.RData')

#2. Prediction with Generalized Boosted Regression
model_gbm <- train(classe ~ .,
                data=myTraining,
                trControl=fitControl,
                method='gbm',
                verbose = FALSE
               )
save(model_gbm, file='./ModelFitGBM.RData')

#3. Prediction with Random Forests
model_rf <- train(classe ~ .,
  data=myTraining,
  trControl=fitControl,
  method='rf',
  ntree=100
)
save(model_rf, file='./ModelFitRF.RData')
```

```
# Model Assessment (in-sample error)

  predCART <- predict(model_cart, newdata=myTesting)
  cmCART <- confusionMatrix(predCART, myTesting$classe)

  predGBM <- predict(model_gbm, newdata=myTesting)
  cmGBM <- confusionMatrix(predGBM, myTesting$classe)

  predRF <- predict(model_rf, newdata=myTesting)
  cmRF <- confusionMatrix(predRF, myTesting$classe)

  AccuracyResults <- data.frame(Model = c('CART', 'GBM', 'RF'),
                      Accuracy = rbind(cmCART$overall[1], cmGBM$overall[1], cmRF$overall[1])
                      )
  print(AccuracyResults)

  #  Model  Accuracy
  #1  CART 0.4890399
  #2   GBM 0.9877655
  #3    RF 0.9984707

  #Based on an assessment of these 3 model fits and out-of-sample results,
  #it looks like both gradient boosting and random forests outperform the CART model,
  #with random forests being slightly more accurate.
  # The confusion matrix for the random forest model is below.

  #         Reference
```

```
#Prediction    A    B    C    D    E
#A          1674    2    0    0    0
#B             0 1136    1    0    0
#C             0    1 1025    1    0
#D             0    0    0  963    4
#E             0    0    0    0 1078
```

#E. Applying the Selected Model to the Test Data (‘pml-testing.csv’)

#Predicting Results on the Test Data (‘pml-testing.csv’)

#Random Forests gave an Accuracy in the myTesting dataset of 99.84%,

#which was more accurate that what I got from the Decision Trees or GBM.

#The Random Forest model will be selected and applied to predict the 20 quiz results (testing dataset) as shown below.

#The expected out-of-sample error is 100-99.84 = 0.16%.

```
predictTEST <- predict(model_rf, newdata=testing)
predictTEST
# [1] 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

# output in better readable format below
TESTPredictionResults <- data.frame(
                              problem_id=testing$problem_id,
                              predicted=predictTEST
                            )
print(TESTPredictionResults)
```

```
  #   problem_id predicted

  #1        1       B

  #2        2       A

  #3        3       B

  #4        4       A

  #5        5       A

  #6        6       E

  #7        7       D

  #8        8       B

  #9        9       A

  #10       10       A

  #11       11       B

  #12       12       C

  #13       13       B

  #14       14       A

  #15       15       E

  #16       16       E

  #17       17       A

  #18       18       B

  #19       19       B

  #20       20       B
```

```
#Function to generate files with predictions 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(predictTEST)
```