---

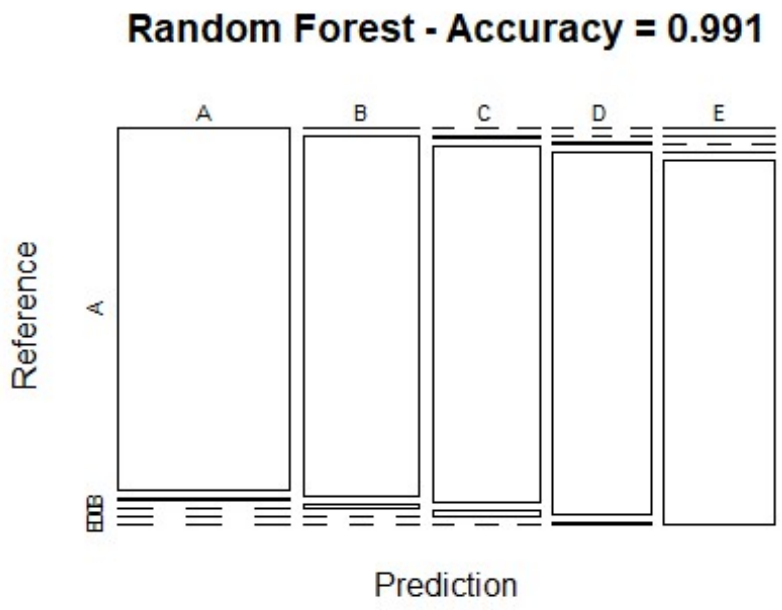
title: “Practical Machine Learning” author: “Avanindra Nath Thakur” date: “11/03/2021” output: html_document

```
#Introduction
# 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
– 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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-
rio.br/har (see the section on the Weight Lifting Exercise Dataset).

#Data

#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://web.archive.org/web/20161224072740/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.
#Executive Summary
#This project predicts the  “classe” variable in the training data set.
different methods are used in this report to explain the model and its cross
validation by using alternative methods as explained in the class.


library(caret);library(rpart)

## Warning: package 'caret' was built under R version 4.0.4

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 4.0.3

## Loading required package: ggplot2
```

```
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.0.4

library(randomForest)

## Warning: package 'randomForest' was built under R version 4.0.4

## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

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

# downloading data
urlTrain <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
urlTest  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"

trainData <- read.csv(url(urlTrain))
testData  <- read.csv(url(urlTest))

# creating a partition within the training data
inTrain  <- createDataPartition(trainData$classe, p=0.7, list=FALSE)
TrainSet <- trainData[inTrain, ]
TestSet  <- trainData[-inTrain, ]
dim(TrainSet)

## [1] 13737   160

dim(TestSet)

## [1] 5885  160

#cleaning data with the removal of almost zero variance
NearlyZeroVar<- nearZeroVar(TrainSet)
TrainSet <- TrainSet[, -NearlyZeroVar]
TestSet  <- TestSet[, -NearlyZeroVar]
dim(TrainSet)

## [1] 13737   106

dim(TestSet)

## [1] 5885  106
```

```r
#second round of cleaning by removing data with missing values
TotalNA    <- sapply(TrainSet, function(x) mean(is.na(x))) > 0.95
TrainSet <- TrainSet[, TotalNA==FALSE]
TestSet  <- TestSet[, TotalNA==FALSE]
dim(TrainSet)

## [1] 13737    59

dim(TestSet)

## [1] 5885   59

#Removing the first 7 non-numeric Variables
TrainSet<- TrainSet[, 8:59]
TestSet<- TestSet[, 8:59]
dim(TrainSet)

## [1] 13737    52

dim(TestSet)

## [1] 5885   52

##Step 2: Random Forest
set.seed(123)
contRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
modFitRF <- train(classe ~ ., data=TrainSet, method="rf",
                          trControl=contRF)
modFitRF$finalModel

##
## 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: 26
##
##         OOB estimate of  error rate: 0.71%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3901    3    1    0    1 0.001280082
## B   24 2624    8    0    2 0.012791573
## C    0   14 2375    7    0 0.008764608
## D    0    1   22 2228    1 0.010657194
## E    0    2    4    7 2512 0.005148515

# performing prediction on Test dataset
predictRF <- predict(modFitRF, newdata=TestSet)
confusMatRF<- confusionMatrix(predictRF, as.factor(TestSet$classe))
#ploting the result
# plot matrix results
plot(confusMatRF$table, col = confusMatRF$byClass,
```

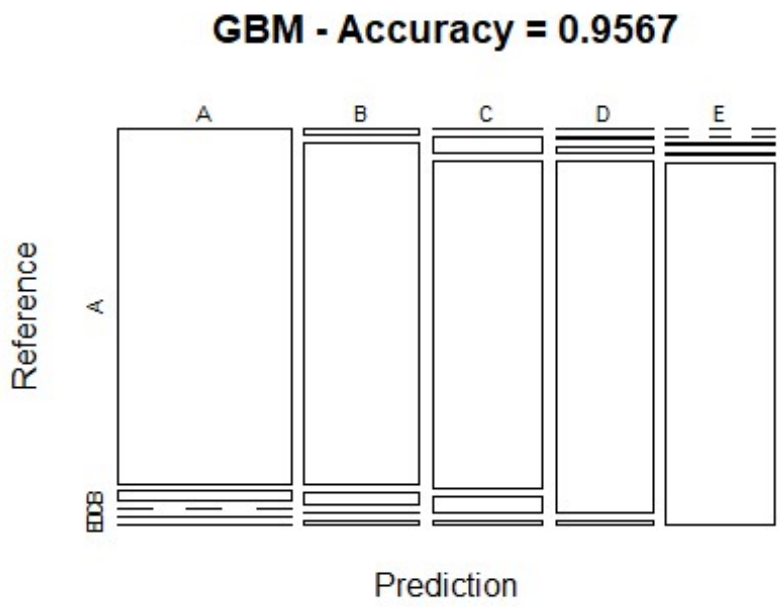
```
     main = paste("Random Forest - Accuracy =",
                  round(confusMatRF$overall['Accuracy'], 4)))
```




```
#step 3 Generalised Boosted Model and prediction
set.seed(123)
library(gbm)

## Warning: package 'gbm' was built under R version 4.0.4

## Loaded gbm 2.1.8

contGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modFitGBM  <- train(classe ~ ., data=TrainSet, method = "gbm",
                    trControl = contGBM, verbose = FALSE)
modFitGBM$finalModel

## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 51 predictors of which 51 had non-zero influence.

#predicting with GBM
predictGBM <- predict(modFitGBM, newdata=TestSet)
confusMatGBM <- confusionMatrix(predictGBM, as.factor(TestSet$classe))
confusMatGBM

## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    A    B    C    D    E
##          A 1645   39    0    2    1
##          B   20 1053   35    3    9
##          C    5   44  973   46   11
##          D    4    3   15  906    8
##          E    0    0    3    7 1053
##
## Overall Statistics
##
##                Accuracy : 0.9567
##                  95% CI : (0.9512, 0.9617)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9452
##
##  Mcnemar's Test P-Value : 4.574e-06
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9827   0.9245   0.9483   0.9398   0.9732
## Specificity            0.9900   0.9859   0.9782   0.9939   0.9979
## Pos Pred Value         0.9751   0.9402   0.9018   0.9679   0.9906
## Neg Pred Value         0.9931   0.9820   0.9890   0.9883   0.9940
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2795   0.1789   0.1653   0.1540   0.1789
## Detection Prevalence   0.2867   0.1903   0.1833   0.1590   0.1806
## Balanced Accuracy      0.9864   0.9552   0.9633   0.9669   0.9856
```

```r
# plotting the results
plot(confusMatGBM$table, col = confusMatGBM$byClass,
     main = paste("GBM - Accuracy =", round(confusMatGBM$overall['Accuracy'],
4)))
```




```
#final Step Applying the model on test data
predictTEST <- predict(modFitRF, newdata=testData)
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
```