FINAL PROJECT: PRACTICAL MACHINE LEARNING

```
## Warning: package 'knitr' was built under R version 3.3.3
```

```
## Warning: package 'UsingR' was built under R version 3.3.3
```

```
## Loading required package: MASS
```

```
## Warning: package 'MASS' was built under R version 3.3.3
```

```
## Loading required package: HistData
```

```
## Warning: package 'HistData' was built under R version 3.3.3
```

```
## Loading required package: Hmisc
```

```
## Warning: package 'Hmisc' was built under R version 3.3.3
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.3.3
```

```
## Loading required package: survival
```

```
## Warning: package 'survival' was built under R version 3.3.3
```

```
## Loading required package: Formula
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.3.3
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
```

```
##
## Attaching package: 'UsingR'
```

```
## The following object is masked from 'package:survival':
##
##     cancer
```

1.0 EXECUTIVE SUMMARY

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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

2.0 GETTING AND CLEANING DATA

2.1 GETTING THE DATA

```
setwd("C:/Soumik/Datasets/machinelearning")
training<-read.csv("pml-training.csv")
testing<-read.csv("pml-testing.csv")
dim(training)
```

```
## [1] 19622   160
```

```
dim(testing)
```

```
## [1]  20 160
```

The data is partitioned into training, validation and test sets as below -

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:survival':
##
```

```
##     cluster
```

```
inTrain<-createDataPartition(training$classe,p=0.7,list=FALSE)
trainset<-training[inTrain,]
testset<-training[-inTrain,]
dim(trainset)
```

```
## [1] 13737   160
```

```
dim(testset)
```

```
## [1] 5885  160
```

2.2 CLEANING THE DATA

All the 3 datasets are cleaned based on the following -

- All predictors with near zero variance are removed
- All predictors with high percentage of NAs are removed

```
nzv_trg<-nearZeroVar(trainset)
trainset<-trainset[,-nzv_trg]
testset<-testset[,-nzv_trg]
AllNA<-sapply(trainset,function(x) mean(is.na(x)))>0.95
trainset<-trainset[,AllNA==FALSE]
testset<-testset[,AllNA==FALSE]

dim(trainset)
```

```
## [1] 13737    59
```

```
dim(testset)
```

```
## [1] 5885   59
```

```
trainset<-trainset[,-c(1:5)]
testset<-testset[,-c(1:5)]

dim(trainset)
```

```
## [1] 13737    54
```

```
dim(testset)
```

```
## [1] 5885   54
```

```
dim(testing)
```

```
## [1]  20 160
```

```
testing<-testing[,-nzv_trg]
testing<-testing[,AllNA==FALSE]
testing<-testing[,-c(1:5)]
dim(testing)
```

```
## [1] 20 54
```

3.0 APPLYING PREDICTION MOEDLLING

3.1 PREDICTION WITH RANDOM FORREST

We first predict using Random Forrest -

```
set.seed(12345)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:Hmisc':
##
##     combine
```

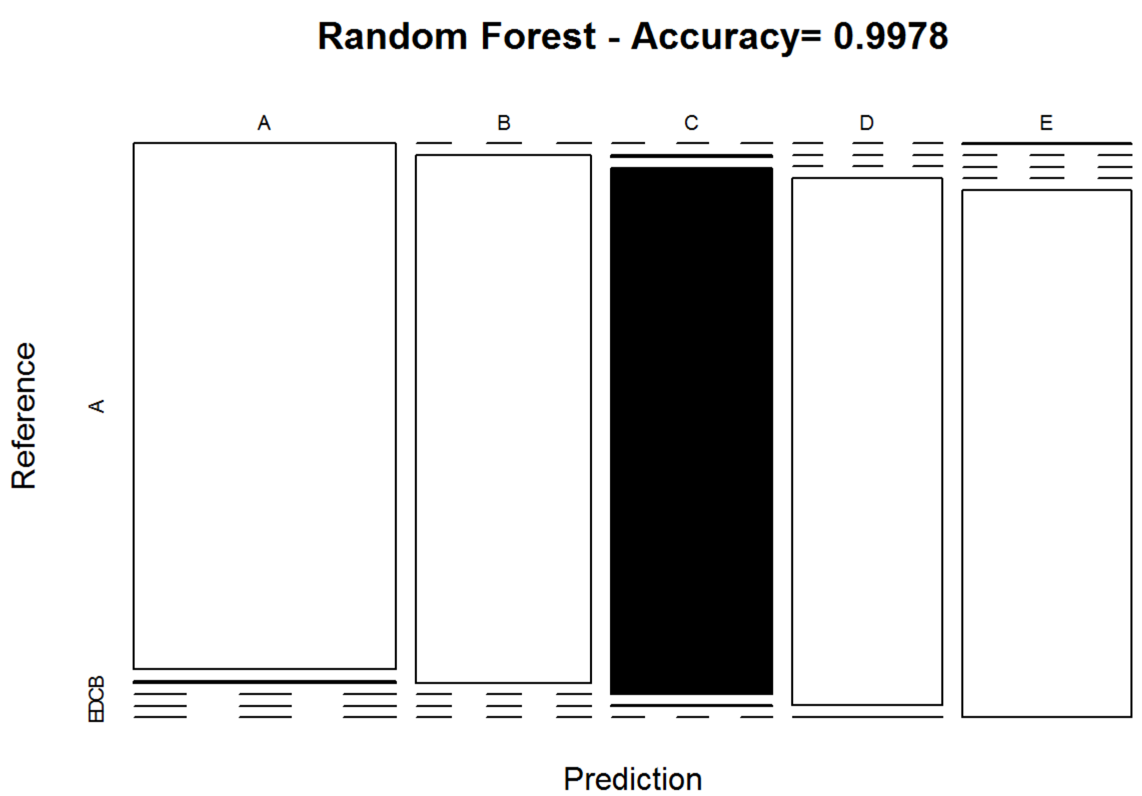
```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
controlRF<-trainControl(method = "cv",number = 3,verboseIter = FALSE)
modfit_RF<-train(classe~.,data=trainset,method="rf",trControl=controlRF)
modfit_RF$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: 27
##
##          OOB estimate of  error rate: 0.23%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3905    1    0    0    0 0.0002560164
## B    8 2647    3    0    0 0.0041384500
## C    0    6 2390    0    0 0.0025041736
## D    0    0    8 2243    1 0.0039964476
## E    0    1    0    4 2520 0.0019801980
```

```
predict_RF<-predict(modfit_RF,newdata = testset)
confM<-confusionMatrix(predict_RF,testset$classe)
plot(confM$table,col=confM$byClass,main=paste("Random Forest - Accuracy=",round(confM$overall
["Accuracy"],4)))
```



The accuracy of Random Forrest model is =**0.9978**

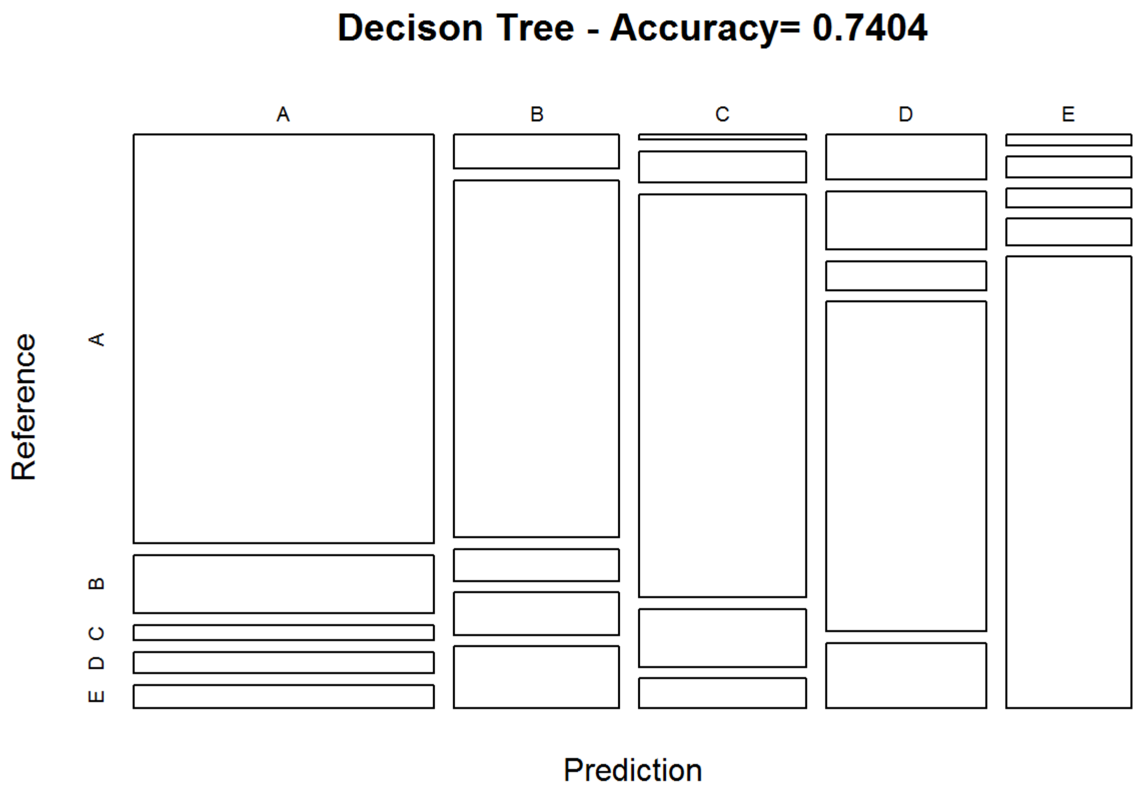
3.2 PREDICTION WITH DECISION TREE

```
set.seed(12345)
library(rpart)
modfit_tree<-rpart(classe~.,data=trainset,method="class")
```

```
predict_tree<-predict(modfit_tree,newdata=testset,type="class")
confM_tree<-confusionMatrix(predict_tree,testset$classe)
confM_tree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1493  213   57   79   85
##          B   68  717   64   86  124
##          C   10   64  821  118   61
##          D   87  114   56  641  127
##          E   16   31   28   40  685
##
## Overall Statistics
##
##                Accuracy : 0.7404
##                  95% CI : (0.729, 0.7515)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.67
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8919   0.6295   0.8002   0.6649   0.6331
## Specificity            0.8969   0.9279   0.9479   0.9220   0.9761
## Pos Pred Value         0.7748   0.6771   0.7644   0.6254   0.8563
## Neg Pred Value         0.9543   0.9126   0.9574   0.9335   0.9219
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2537   0.1218   0.1395   0.1089   0.1164
## Detection Prevalence   0.3274   0.1799   0.1825   0.1742   0.1359
## Balanced Accuracy      0.8944   0.7787   0.8741   0.7935   0.8046
```

```
plot(confM_tree$table,col=confM_tree$byClass,main=paste("Decison Tree - Accuracy=",round(conf
M_tree$overall["Accuracy"],4)))
```



The accuracy of Decision Tree model is =**0.7404**

3.3 PREDICTION WITH BOSSTING

```
set.seed(12345)
controlGBM<-trainControl(method="repeatedcv",number = 5,repeats = 1)
modfit_gbm<-train(classe~.,data=trainset,method="gbm",trControl=controlGBM,verbose=FALSE)
```

```
## Loading required package: gbm
```

```
## Warning: package 'gbm' was built under R version 3.3.3
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.3
```

```
## Loading required package: plyr
```

```
## Warning: package 'plyr' was built under R version 3.3.3
```

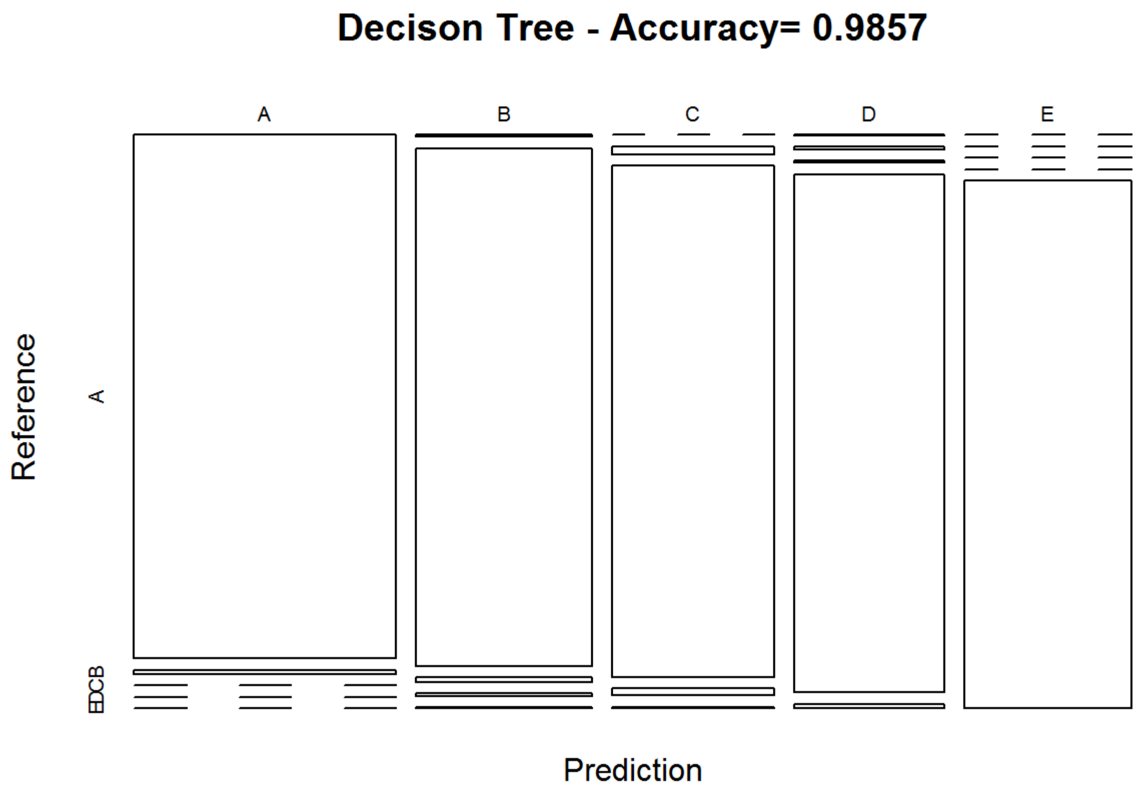
```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize
```

```
predict_gbm<-predict(modfit_gbm,newdata = testset)
confM_gbm<-confusionMatrix(predict_gbm,testset$classe)
confM_gbm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1669   13    0    0    0
##          B    4 1106    9    6    2
##          C    0   16 1013   14    3
##          D    1    4    4  944    8
##          E    0    0    0    0 1069
##
## Overall Statistics
##
##                Accuracy : 0.9857
##                  95% CI : (0.9824, 0.9886)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9819
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9710   0.9873   0.9793   0.9880
## Specificity            0.9969   0.9956   0.9932   0.9965   1.0000
## Pos Pred Value         0.9923   0.9814   0.9685   0.9823   1.0000
## Neg Pred Value         0.9988   0.9931   0.9973   0.9959   0.9973
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2836   0.1879   0.1721   0.1604   0.1816
## Detection Prevalence   0.2858   0.1915   0.1777   0.1633   0.1816
## Balanced Accuracy      0.9970   0.9833   0.9903   0.9879   0.9940
```

```
plot(confM_gbm$table,col=confM_gbm$byClass,main=paste("Decison Tree - Accuracy=",round(confM_gbm$overall["Accuracy"],4)))
```



The accuracy of Boosting is =**0.9857**

4.0 CONCLUSION

It is observed that out of these three models, Random Forrest provides more accuracy and hence the Random Forrest model will be used for prediction using the test data set.

4.1 PREDICTION OF TEST DATA USING RANDOM FORREST

```
predict(modfit_RF,newdata = testing)
```

```
##  [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
```