STEP1: First we will set the working directory and load the files from the working directory
getwd()
setwd("C:/Raj - Personal/CourseEra/Course 8- Practical Machine Learning/Assignment")
getwd()

STEP2: Will prepare the environment with all the required packages and libraries
Preparing the overall environment
library(caret);library(ggplot2);library(knitr);library(randomForest);library(rattle)
set.seed(1234)

STEP3: Read the csv files and save it to variables
training<- read.csv("./pml-training.csv")
testing<- read.csv("./pml-testing.csv")
dim(training);dim(testing)

STEP4: create the training and test sets
# create a partition with the training dataset
intrain <- createDataPartition(training$classe, p=0.7, list=FALSE)
training_set <- training[intrain,]
testing_set <- training[-intrain,]
dim(training_set);dim(testing_set)

STEP5: Remove the variables with zero variance and those which are with NA
# remove variables with Nearly Zero Variance
NZV <- nearZeroVar(training_set)
training_set <- training_set[, -NZV]
testing_set <- testing_set[, -NZV]
dim(training_set);dim(testing_set)

remove the variables that are NA
AllNA   <- sapply(training_set, function(x) mean(is.na(x))) > 0.95
training_set <- training_set[, AllNA==FALSE]
testing_set <- testing_set[, AllNA==FALSE]
dim(training_set);dim(testing_set)
str(training_set)

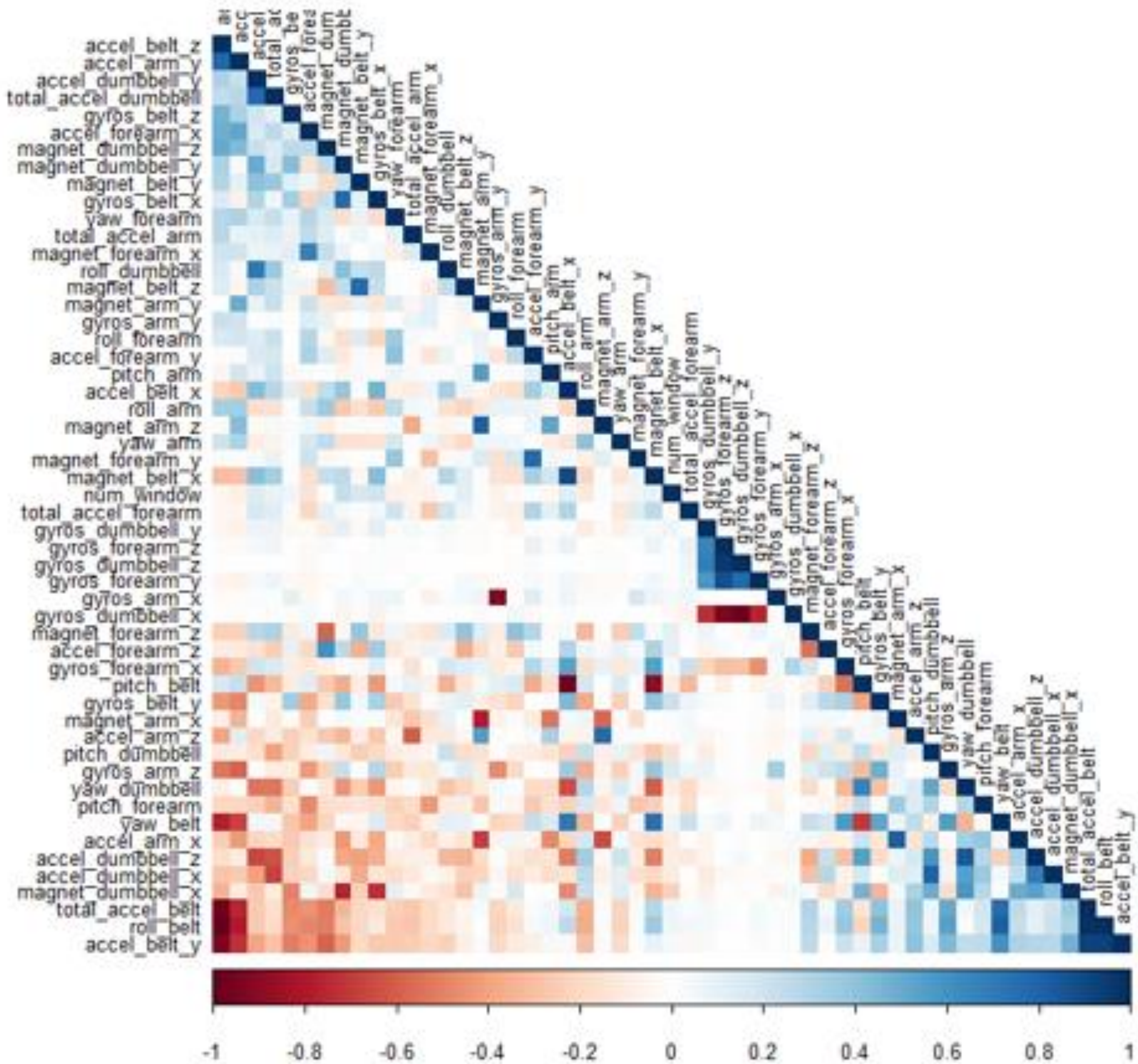
STEP6: Analyze the correlation between the various variables in the data set
training_set <- training_set[, -(1:5)]
testing_set <- testing_set[, -(1:5)]
dim(training_set);dim(testing_set)

Analyze the corelation between the various variables in the data set
my_num_data <- mydata[, sapply(mydata, is.numeric)]

```
library(corrplot)
corMatrix <- cor(training_set[, -54])
corrplot(corMatrix, order = "FPC", method = "color", type = "lower",
    tl.cex = 0.8, tl.col = rgb(0, 0, 0))
```

Outcome of the correlation chart picture below:

The highly corelated variables are in the dark color in the graph below



STEP7: Building a prediction model using various models such as Random forests, Decision trees, and Generalized Boosted Models.

A confusion matrix is plotted for each of these models to depict the accuracy of these models

Random Forest Model:

```
cv <- trainControl(method="cv", number=3, verboseIter=FALSE)
mod_rf <- train(classe ~ ., data=training_set, method="rf",
                trControl=cv)
```

```
> mod_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: 0.21%
Confusion matrix:
     A    B    C    D    E  class.error
A 3904    2    0    0    0 0.0005120328
B    3 2651    3    1    0 0.0026335591
C    0    6 2390    0    0 0.0025041736
D    0    0    8 2244    0 0.0035523979
E    0    1    0    5 2519 0.0023762376
```

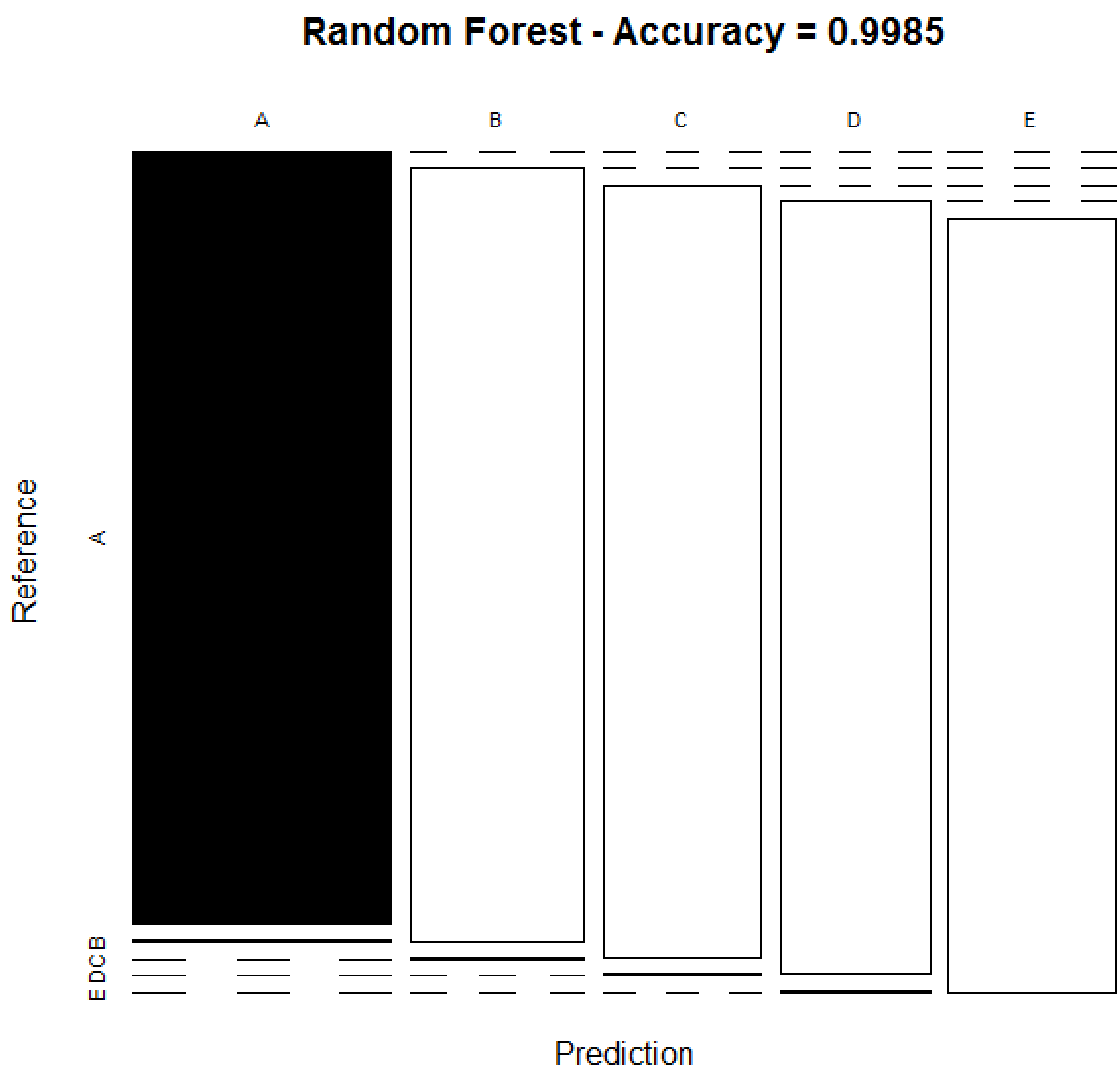
```
# predict the same on the test data
predictrf <- predict(mod_rf, newdata=testing_set)
cmrf <- confusionMatrix(predictrf, testing_set$classe)
cmrf
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1674    3    0    0    0
         B    0 1136    1    0    0
         C    0    0 1025    2    0
         D    0    0    0  962    3
         E    0    0    0    0 1079

Overall Statistics
                                          
               Accuracy : 0.9985          
                 95% CI : (0.9971, 0.9993)
    No Information Rate : 0.2845          
    P-Value [Acc > NIR] : < 2.2e-16       
                                          
                  Kappa : 0.9981          
 Mcnemar's Test P-Value : NA              

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9974   0.9990   0.9979   0.9972
Specificity            0.9993   0.9998   0.9996   0.9994   1.0000
```

Pos Pred Value	0.9982	0.9991	0.9981	0.9969	1.0000
Neg Pred Value	1.0000	0.9994	0.9998	0.9996	0.9994
Prevalence	0.2845	0.1935	0.1743	0.1638	0.1839
Detection Rate	0.2845	0.1930	0.1742	0.1635	0.1833
Detection Prevalence	0.2850	0.1932	0.1745	0.1640	0.1833
Balanced Accuracy	0.9996	0.9986	0.9993	0.9987	0.9986

Next is to plot the accuracy of the Random Forest Model

```
plot(cmrf$table, col = cmrf$byClass,
    main = paste("Random Forest - Accuracy =",
            round(cmrf$overall['Accuracy'], 4)))
```

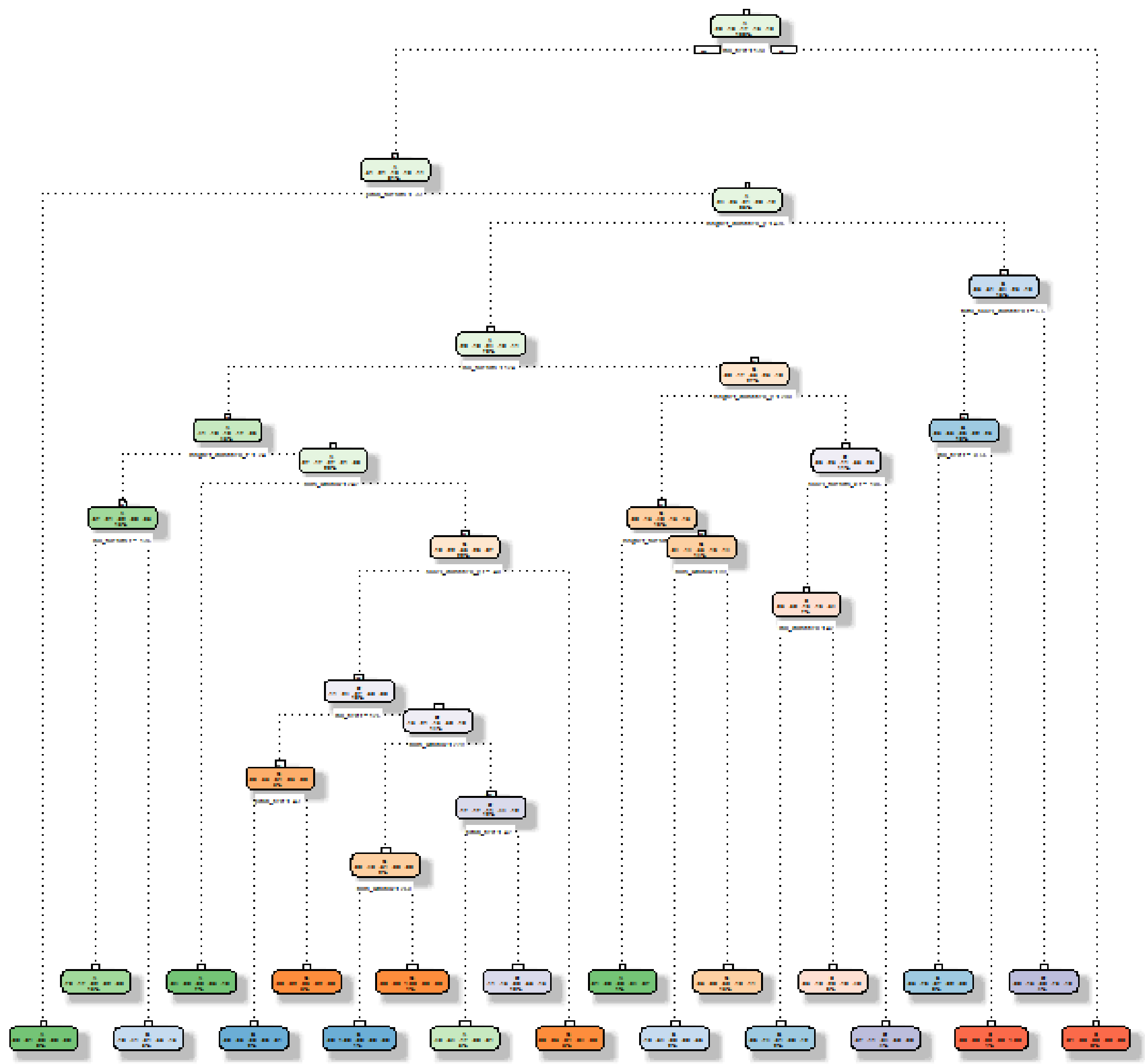


<u>Modeling using Decision Trees:</u>

We will use decision tree model and plot the outcome, below is the outcome of the model

```
library(rpart)
```

```
library(rpart.plot)
mod_dt <- rpart(classe ~ ., data=training_set, method="class")
fancyRpartPlot(mod_dt)
```



Appling the model on to the testing data set and check the outcome of the confusion matrix

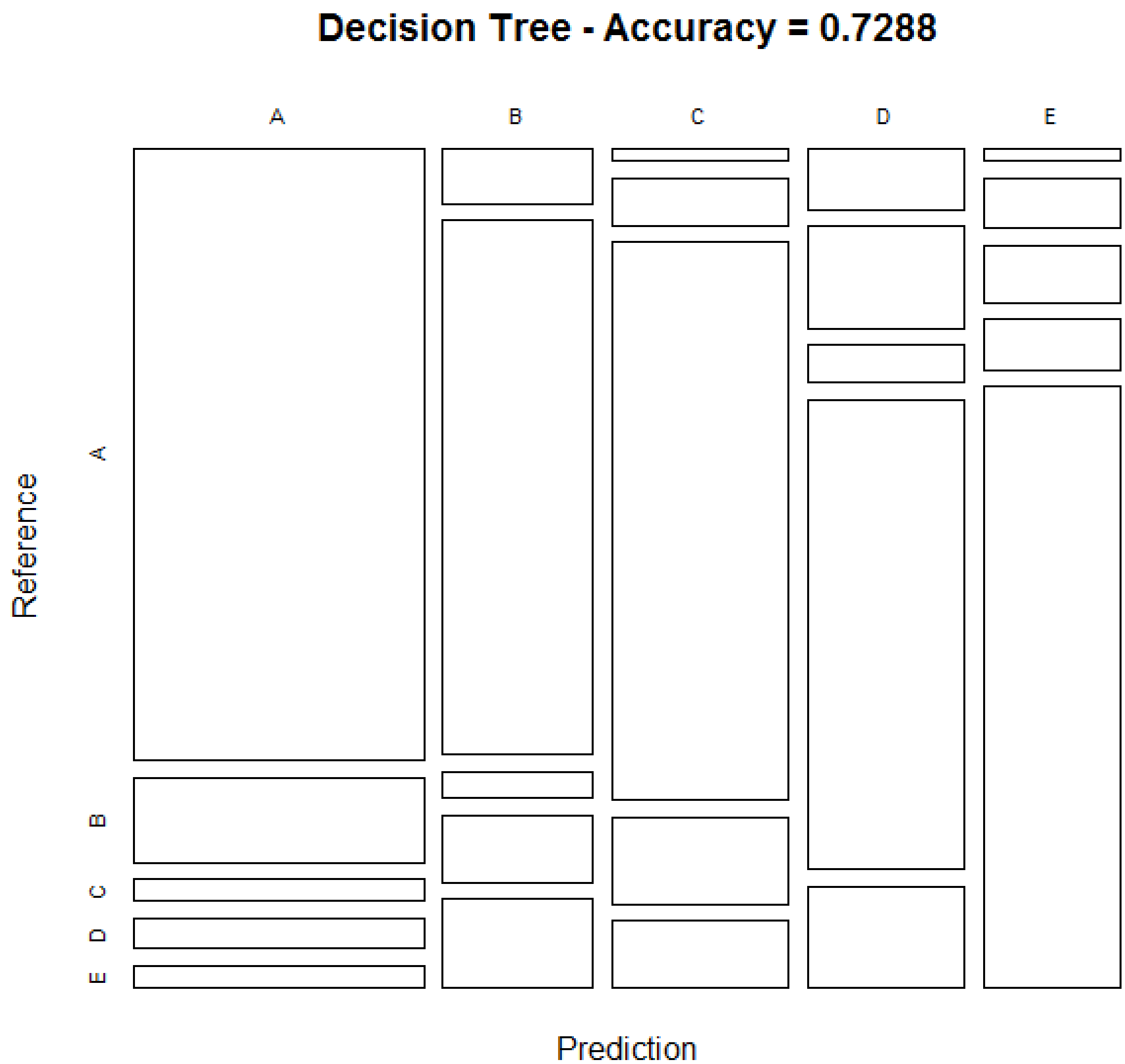
```
predictdt <- predict(mod_dt, newdata=testing_set, type="class")
cfdt <- confusionMatrix(predictdt, testing_set$classe)
cfdt
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1489  206   54   76   52
         B   70  671   34   84  111
```

```
         C   19   70  823  128   98
         D   81  134   49  618  133
         E   15   58   66   58  688

Overall Statistics

               Accuracy : 0.7288
                 95% CI : (0.7172, 0.7401)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6557
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.8895   0.5891   0.8021   0.6411   0.6359
Specificity            0.9079   0.9370   0.9352   0.9193   0.9590
Pos Pred Value         0.7933   0.6918   0.7232   0.6089   0.7774
Neg Pred Value         0.9538   0.9048   0.9572   0.9290   0.9212
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2530   0.1140   0.1398   0.1050   0.1169
Detection Prevalence   0.3189   0.1648   0.1934   0.1725   0.1504
Balanced Accuracy      0.8987   0.7631   0.8687   0.7802   0.7974
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1489  206   54   76   52
         B   70  671   34   84  111
         C   19   70  823  128   98
         D   81  134   49  618  133
         E   15   58   66   58  688

Overall Statistics

               Accuracy : 0.7288
                 95% CI : (0.7172, 0.7401)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6557
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.8895   0.5891   0.8021   0.6411   0.6359
Specificity            0.9079   0.9370   0.9352   0.9193   0.9590
Pos Pred Value         0.7933   0.6918   0.7232   0.6089   0.7774
Neg Pred Value         0.9538   0.9048   0.9572   0.9290   0.9212
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2530   0.1140   0.1398   0.1050   0.1169
Detection Prevalence   0.3189   0.1648   0.1934   0.1725   0.1504
Balanced Accuracy      0.8987   0.7631   0.8687   0.7802   0.7974
```

We will plot the matrix results

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



Generalized Boosting Model:
Lets apply the GBM model and see the outcome

Apply the model on the testing data set and see the outcome
predictGBM <- predict(mod_GBM, newdata=testing_set)
cmGBM <- confusionMatrix(predictGBM, testing_set$classe)
cmGBM

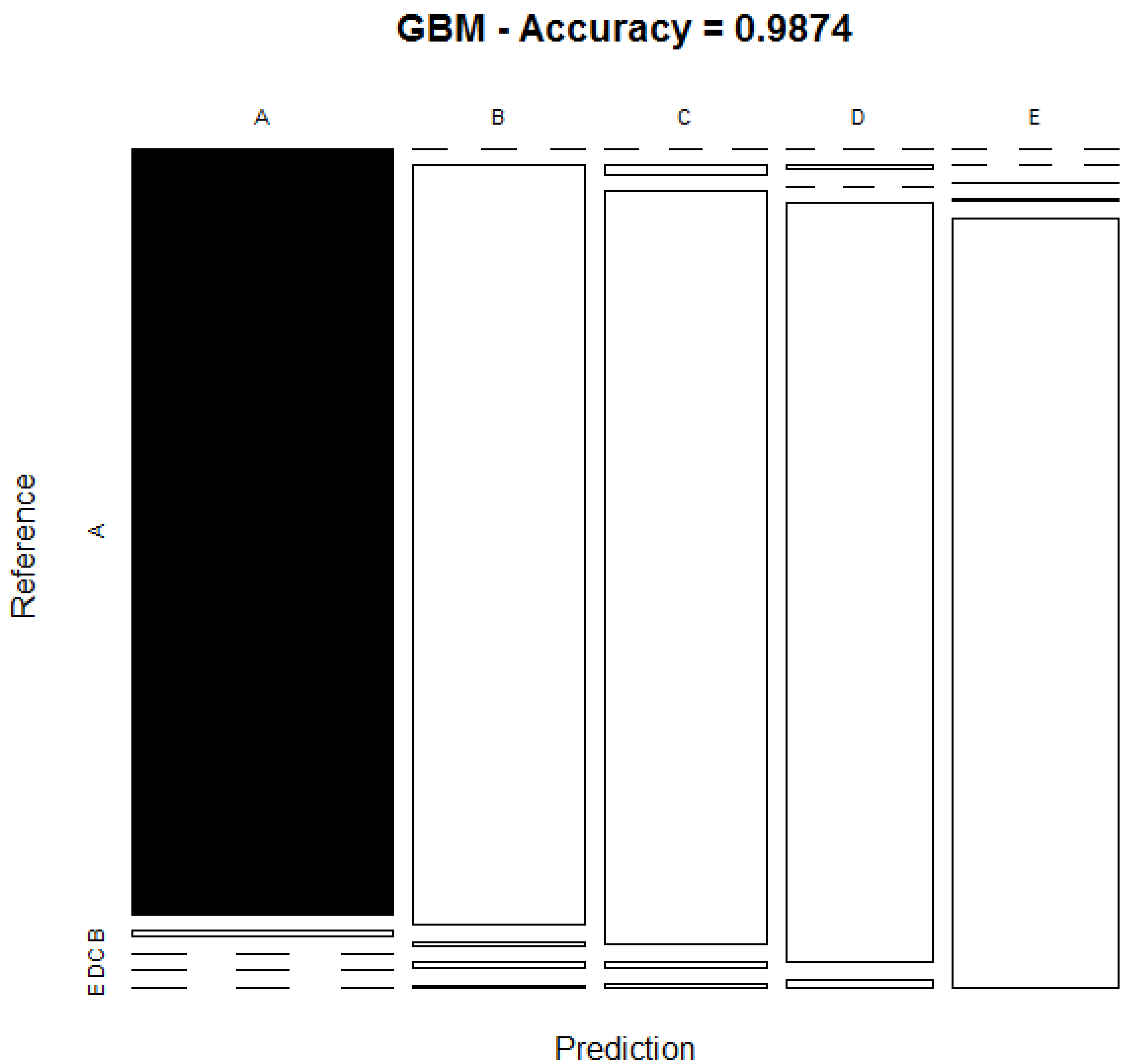
```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1674   14    0    0    0
         B    0 1107    6    8    3
         C    0   13 1019    8    4
         D    0    5    0  946   10
         E    0    0    1    2 1065
```

```
Overall Statistics

               Accuracy : 0.9874
                 95% CI : (0.9842, 0.9901)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9841
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9719   0.9932   0.9813   0.9843
Specificity            0.9967   0.9964   0.9949   0.9970   0.9994
Pos Pred Value         0.9917   0.9849   0.9761   0.9844   0.9972
Neg Pred Value         1.0000   0.9933   0.9986   0.9963   0.9965
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2845   0.1881   0.1732   0.1607   0.1810
Detection Prevalence   0.2868   0.1910   0.1774   0.1633   0.1815
Balanced Accuracy      0.9983   0.9842   0.9940   0.9891   0.9918
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1674   14    0    0    0
         B    0 1107    6    8    3
         C    0   13 1019    8    4
         D    0    5    0  946   10
         E    0    0    1    2 1065
```


```
Overall Statistics

               Accuracy : 0.9874
                 95% CI : (0.9842, 0.9901)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9841
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9719   0.9932   0.9813   0.9843
Specificity            0.9967   0.9964   0.9949   0.9970   0.9994
Pos Pred Value         0.9917   0.9849   0.9761   0.9844   0.9972
Neg Pred Value         1.0000   0.9933   0.9986   0.9963   0.9965
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2845   0.1881   0.1732   0.1607   0.1810
Detection Prevalence   0.2868   0.1910   0.1774   0.1633   0.1815
Balanced Accuracy      0.9983   0.9842   0.9940   0.9891   0.9918
```


Let us plot the results

```
plot(cmGBM$table, col = cmGBM$byClass,
    main = paste("GBM - Accuracy =", round(cmGBM$overall['Accuracy'], 4)))
```



Based on the outcome of the three test results from three different models, we note that the most accurate one turns out to be Random Forest model. Hence we will apply Random Forest model on our testing data set.

Random Forest: 0.99
Decision Tree: 0.72
Gradient Boosting Model: 0.98

```
predict_testing <- predict(mod_rf, newdata=testing)
predict_testing
```