# Prediction Assignment

Ranjit Subudhi

6/23/2019

## Background

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

# Data Loading

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (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 (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.

```
setwd('C:/Users/h193736/Desktop/Online Course/Data Science/7. Practical Machine Learning/Week 4')
```

# Download Packages

```
library(caret)
```

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

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

```
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## randomForest 4.6-14
```

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

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

```
## The following object is masked from 'package:rattle':
##
##     importance
```

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

## Data Loading and Cleaning

```
Train <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
Test  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

```
training <- read.csv(url(Train))
testing  <- read.csv(url(Test))
```

# create a partition with the training dataset

```
inTrain  <- createDataPartition(training$classe, p=0.8, list=FALSE)
TrainingSet <- training[inTrain, ]
TestSet  <- training[-inTrain, ]
dim(TrainingSet)
```

```
## [1] 15699   160
```

```
dim(TestSet)
```

```
## [1] 3923  160
```

# Remove Zero Variance variables

```
NZV <- nearZeroVar(TrainingSet)
TrainingSet <- TrainingSet[, -NZV]
TestSet  <- TestSet[, -NZV]
dim(TrainingSet)
```

```
## [1] 15699   104
```

```
dim(TestSet)
```

```
## [1] 3923  104
```

# Remove NA variables

```
AllNA    <- sapply(TrainingSet, function(x) mean(is.na(x))) > 0.95
TrainingSet <- TrainingSet[, AllNA==FALSE]
TestSet  <- TestSet[, AllNA==FALSE]
dim(TrainingSet)
```

```
## [1] 15699    59
```

```
dim(TestSet)
```

```
## [1] 3923   59
```

# remove identification only variables (columns 1 to 5)

```
TrainingSet <- TrainingSet[, -(1:5)]
TestSet  <- TestSet[, -(1:5)]
dim(TrainingSet)
```

```
## [1] 15699    54
```

```
dim(TestSet)
```

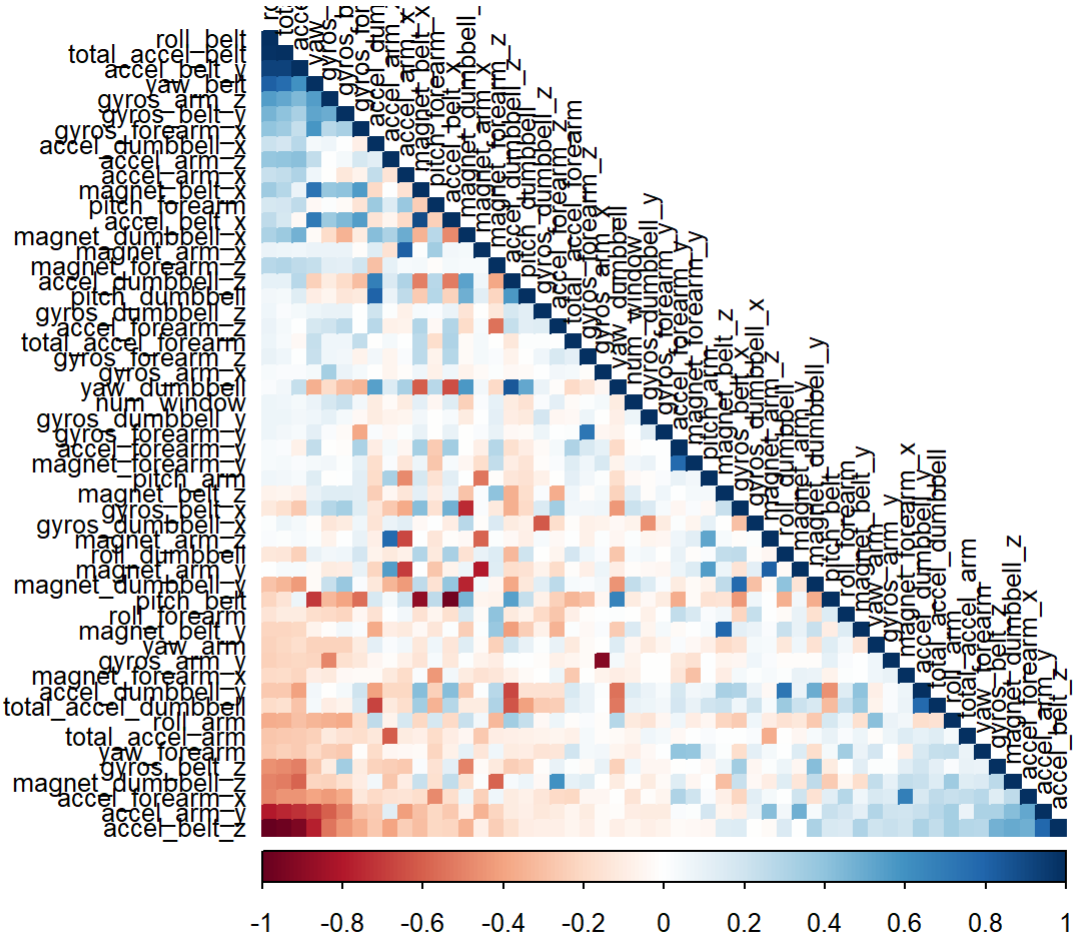
```
## [1] 3923   54
```

# Correlation Analysis

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

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




###Prediction Model Building

# Random Forest

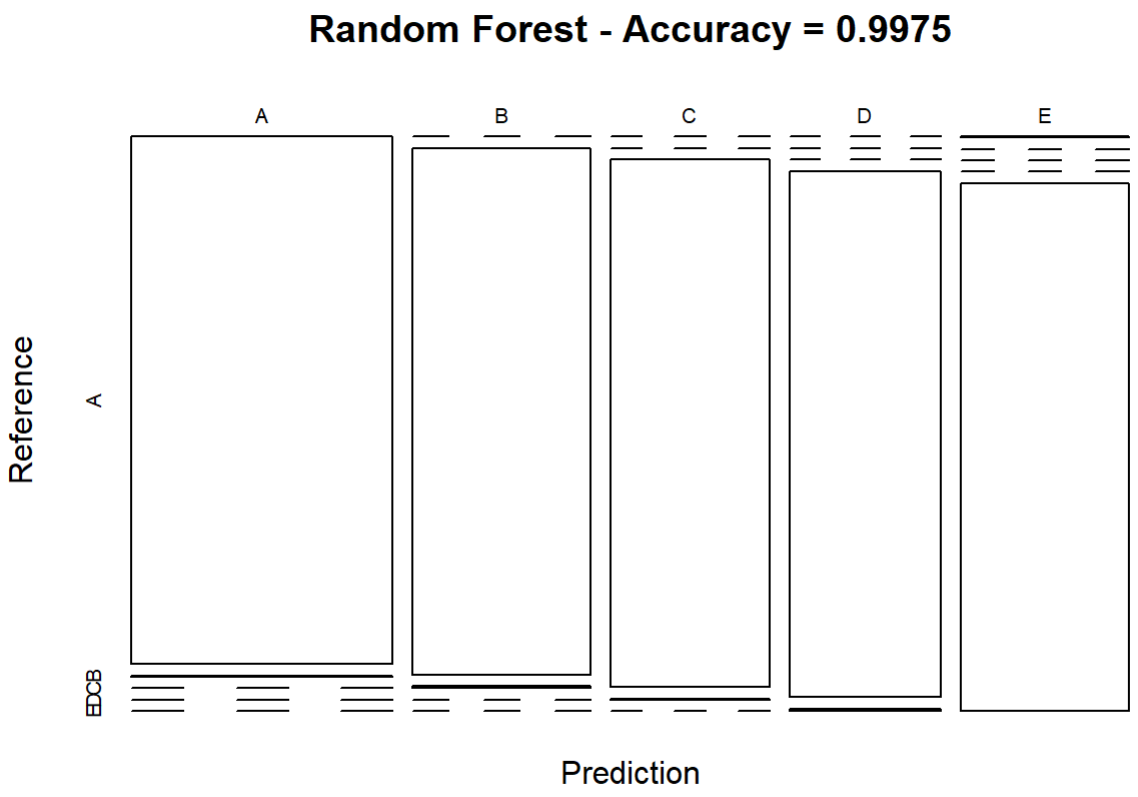
```
set.seed(12345)
controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
modFitRandForest <- train(classe ~ ., data=TrainingSet, method="rf",
                          trControl=controlRF)
modFitRandForest$finalModel
```

```
##
## 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: 0.18%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4463    1    0    0    0 0.0002240143
## B    5 3031    2    0    0 0.0023041475
## C    0    5 2732    1    0 0.0021913806
## D    0    0    8 2564    1 0.0034978624
## E    0    1    0    5 2880 0.0020790021
```

```
predictRandForest <- predict(modFitRandForest, newdata=TestSet)
confMatRandForest <- confusionMatrix(predictRandForest, TestSet$classe)
confMatRandForest
```

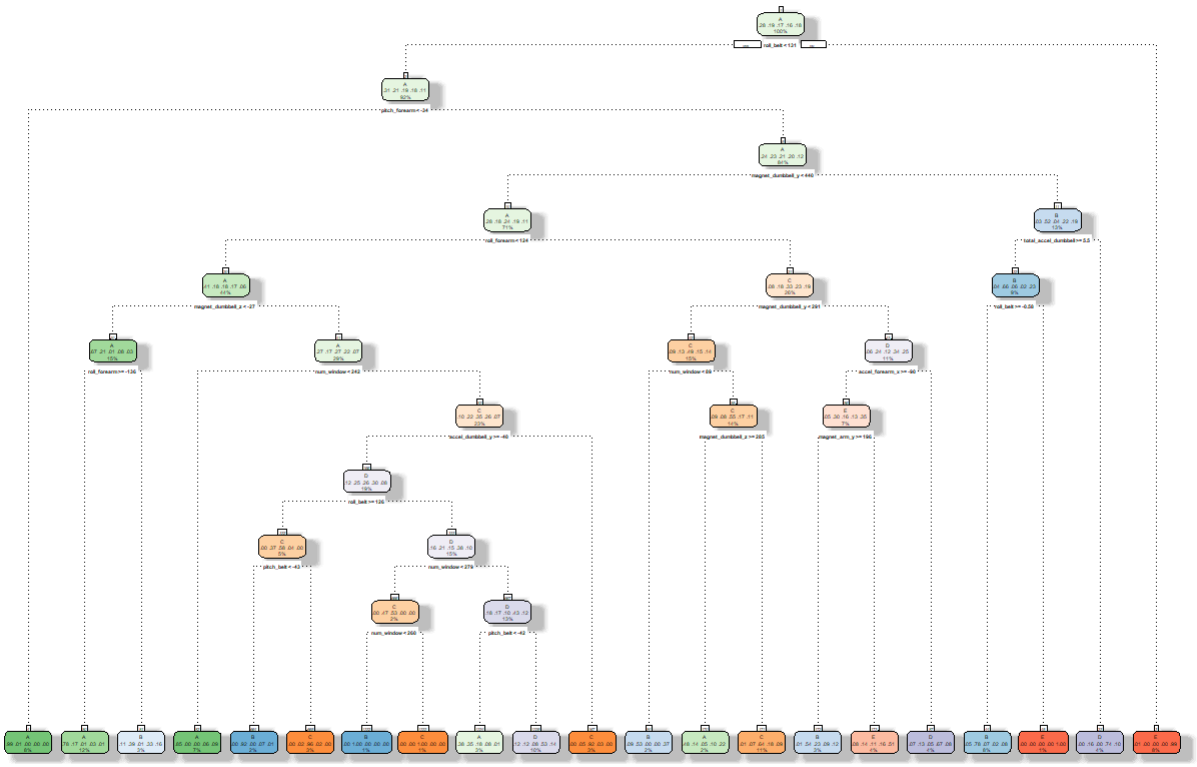
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    2    0    0    0
##          B    0  757    3    0    0
##          C    0    0  681    1    0
##          D    0    0    0  642    3
##          E    1    0    0    0  718
##
## Overall Statistics
##
##                Accuracy : 0.9975
##                  95% CI : (0.9953, 0.9988)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9968
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9974   0.9956   0.9984   0.9958
## Specificity            0.9993   0.9991   0.9997   0.9991   0.9997
## Pos Pred Value         0.9982   0.9961   0.9985   0.9953   0.9986
## Neg Pred Value         0.9996   0.9994   0.9991   0.9997   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1930   0.1736   0.1637   0.1830
## Detection Prevalence   0.2847   0.1937   0.1738   0.1644   0.1833
## Balanced Accuracy      0.9992   0.9982   0.9977   0.9988   0.9978
```

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




# Decision Tree

```
set.seed(12345)
modFitDecTree <- rpart(classe ~ ., data=TrainingSet, method="class")
fancyRpartPlot(modFitDecTree)
```

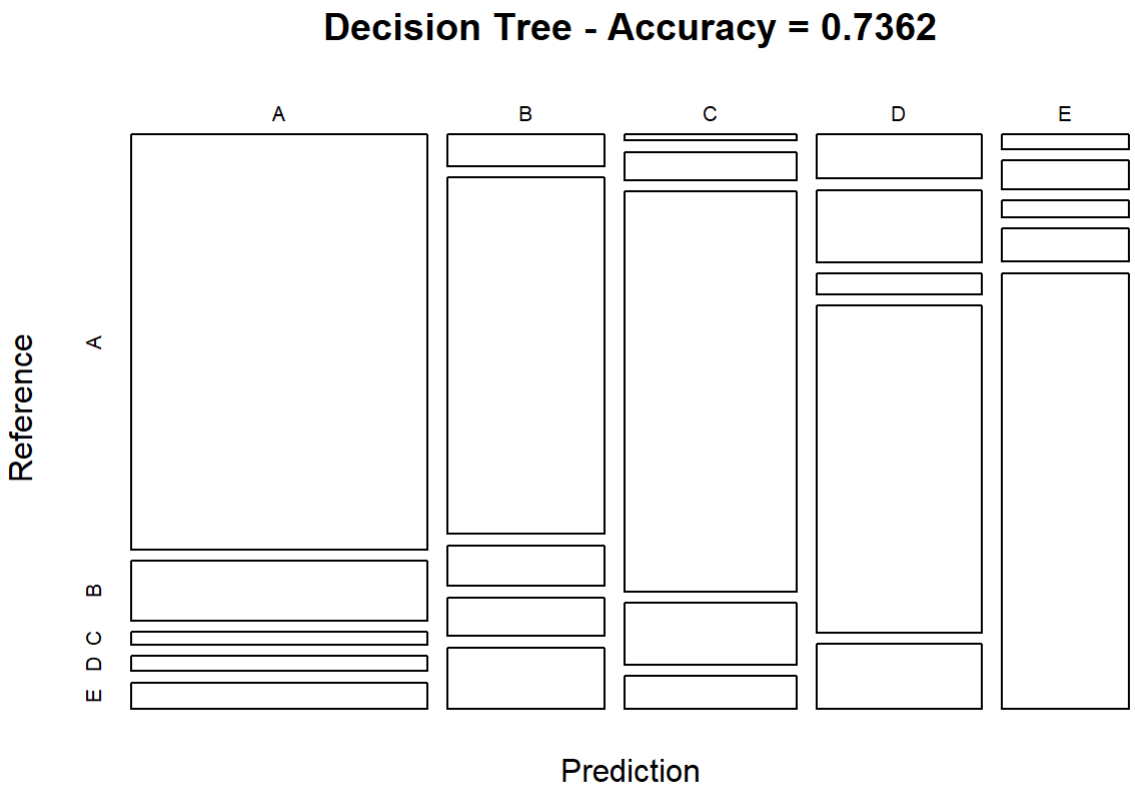


Rattle 2019-Jun-23 13:28:07 H193736

```
predictDecTree <- predict(modFitDecTree, newdata=TestSet, type="class")
confMatDecTree <- confusionMatrix(predictDecTree, TestSet$classe)
confMatDecTree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 994 142  30  37  63
##          B  40 453  52  49  78
##          C   8  39 557  86  46
##          D  59  96  28 437  87
##          E  15  29  17  34 447
##
## Overall Statistics
##
##                Accuracy : 0.7362
##                  95% CI : (0.7221, 0.7499)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6649
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8907   0.5968   0.8143   0.6796   0.6200
## Specificity            0.9031   0.9308   0.9447   0.9177   0.9703
## Pos Pred Value         0.7852   0.6741   0.7568   0.6181   0.8247
## Neg Pred Value         0.9541   0.9059   0.9602   0.9359   0.9190
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2534   0.1155   0.1420   0.1114   0.1139
## Detection Prevalence   0.3227   0.1713   0.1876   0.1802   0.1382
## Balanced Accuracy      0.8969   0.7638   0.8795   0.7987   0.7952
```

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




# Generalized Boosted Model

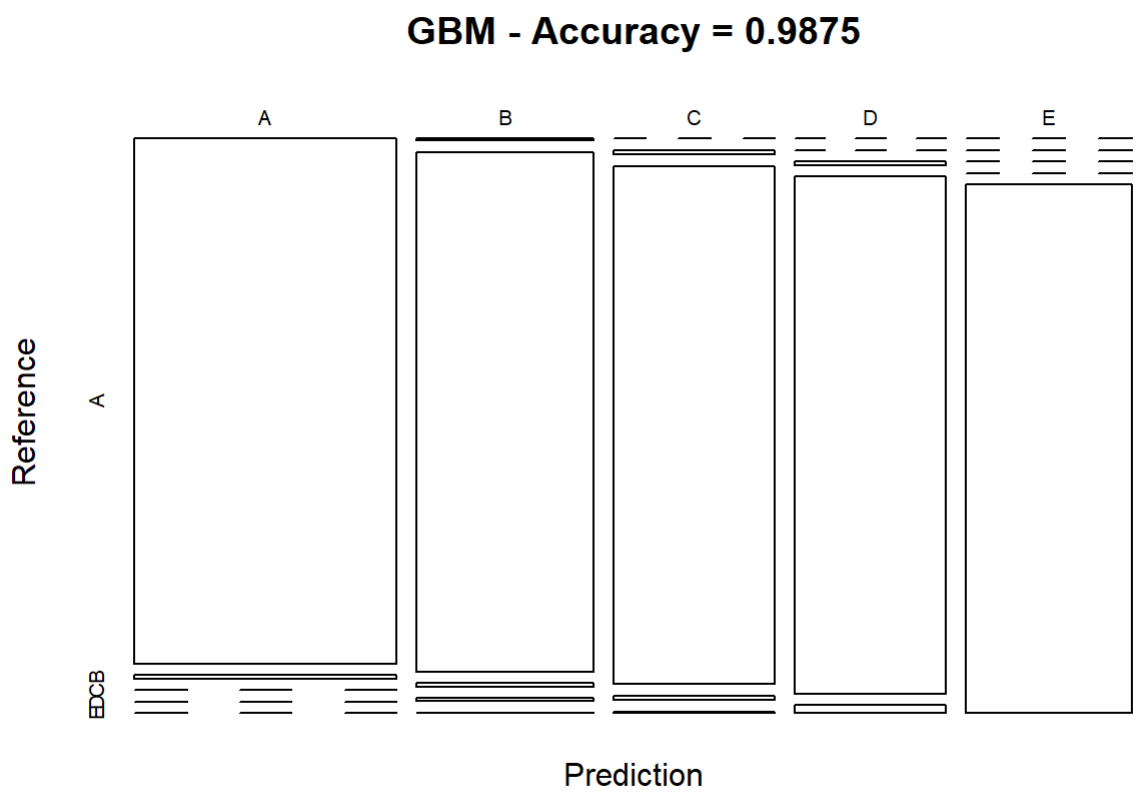
```
set.seed(12345)
controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modFitGBM  <- train(classe ~ ., data=TrainingSet, method = "gbm",
                    trControl = controlGBM, verbose = FALSE)
modFitGBM$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 53 had non-zero influence.
```

```
predictGBM <- predict(modFitGBM, newdata=TestSet)
confMatGBM <- confusionMatrix(predictGBM, TestSet$classe)
confMatGBM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1113    8    0    0    0
##          B    3  745    5    4    1
##          C    0    6  675    6    2
##          D    0    0    4  633   10
##          E    0    0    0    0  708
##
## Overall Statistics
##
##                Accuracy : 0.9875
##                  95% CI : (0.9835, 0.9907)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9842
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9973   0.9816   0.9868   0.9844   0.9820
## Specificity            0.9971   0.9959   0.9957   0.9957   1.0000
## Pos Pred Value         0.9929   0.9828   0.9797   0.9784   1.0000
## Neg Pred Value         0.9989   0.9956   0.9972   0.9969   0.9960
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2837   0.1899   0.1721   0.1614   0.1805
## Detection Prevalence   0.2858   0.1932   0.1756   0.1649   0.1805
## Balanced Accuracy      0.9972   0.9887   0.9913   0.9901   0.9910
```

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




# Apply the Select Model to the Test Data

```
predictTEST <- predict(modFitRandForest, 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
```