#----

#Final Project Report - Practical Machine Learning Course

#These are the files produced during a homework assignment of Coursera's MOOC Practical Machine Learning from Johns Hopkins University.

#For more information about the several MOOCs comprised in this Specialization,

#please visit: https://www.coursera.org/specialization/jhudatascience/

#Student Developer: Fawzia Zehra Kara-Isitt (FuzzyLogic9)

#GitHub repo: https://github.com/FuzzyLogic9/Coursera_JHU_PML_Final

#Background Introduction

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

#Full source: Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. “Qualitative Activity Recognition of Weight Lifting Exercises.

#Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human ’13)”. Stuttgart, Germany: ACM SIGCHI, 2013.

#A special thanks to the above mentioned authors for being so generous in allowing their data to be used for my project assignment.

#Data Sources

#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 comes from this original source: 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.

#Project Intended Results

#The goal of your project is to predict the manner in which they did the exercise.

#This is the "classe" variable in the training set.

#You may use any of the other variables to predict with.

#You should create a report describing how you built your model, how you used cross validation,

#what you think the expected out of sample error is,

#and why you made the choices you did.

#You will also use your prediction model to predict 20 different test cases.

#Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis.

#Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5.

#It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online

#(and you always want to make it easy on graders :-))

#You should also apply your machine learning algorithm to the 20 test cases available in the test data above.

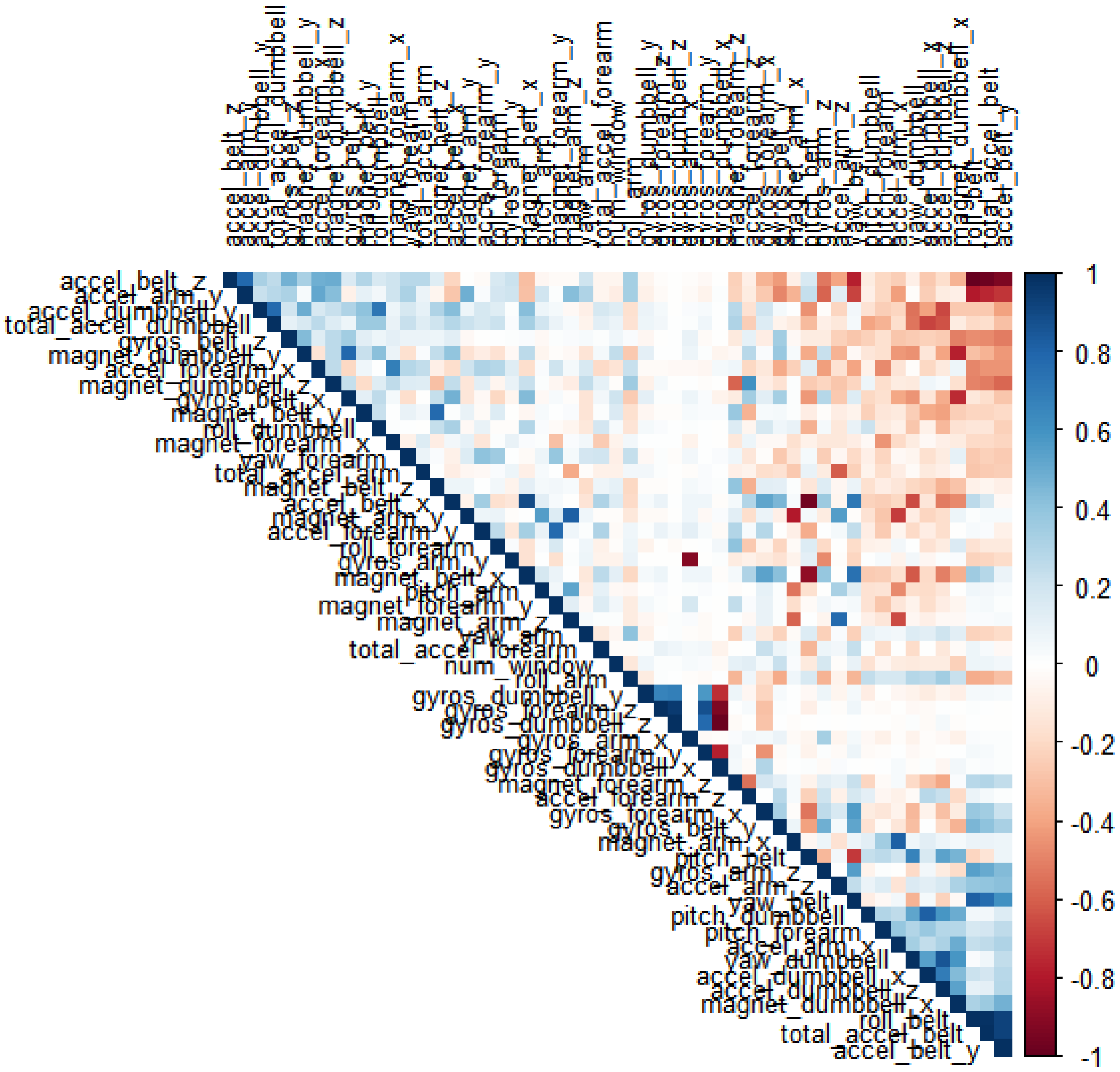
#Please submit your predictions in appropriate format to the programming assignment for automated grading.

#See the programming assignment for additional details.

```
#Libraries included for analysis


library(caret)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(corrplot)
library(RColorBrewer)
library(rattle)
library(randomForest)
library(gbm)
library(doParallel)
library(dplyr)
library(e1071)
library(AppliedPredictiveModeling)

# Load the same seed with the following line of code:
set.seed(12345)

#Dowloaded data and direct internet access, just because..

trainingset <- read.csv("C:/Users/fzkar/Desktop/Career2017-18/JH_PMLCoursera/pml-training.csv",
head=TRUE, sep=",", na.strings=c("NA","#DIV/0!",""))
testingset <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"),
head=TRUE, sep=",", na.strings=c("NA","#DIV/0!",""))

#Checked structure
str(trainingset)
str(testingset)
```

```
# Data Partitioning and Cleanup

#Data obtained in the testing and traning set need to be cleaned up, columns contaning a majority of NA values need to be removed,

#columns with low variance also need to be removed. In the end we should be left with only those columns that influence the prediction.

#Partioning Training data set into two data sets, 70% for myTraining, 30% for myTesting:


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

myTrainingset <- trainingset[inTrain, ]

myTestingset <- trainingset[-inTrain, ]

dim(myTrainingset)

dim(myTestingset)


# Starting Number of variables here is: 160!

#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.



#1.Viewing NearZeroVariance variables

myDataNZV <- nearZeroVar(myTrainingset)

goodTrainData <- myTrainingset[,-myDataNZV]

goodTestData <- myTestingset[,-myDataNZV]

dim(goodTrainData)

dim(goodTestData)


#Variables reduced to 130 at this point!
```

```
# 2. Remove variables that are mostly NA

RemNAData   <- sapply(goodTrainData, function(x) mean(is.na(x))) > 0.95

goodTrainData <- goodTrainData[, RemNAData==FALSE]

goodTestData  <- goodTestData[, RemNAData==FALSE]

dim(goodTrainData)

dim(goodTestData)


#Variables reduced to 59 at this point!


#Transformation 3: Killing first 5 columns of Dataset - ID  variable so  no interference

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

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


#With the cleaning process above, the number of variables for the analysis has been reduced.

#Final number of variables is: 54!!!

dim(goodTrainData)

dim(goodTestData)

goodTrainData

#A correlation among variables is analysed before proceeding to the modeling procedures.

corrMatrix <- cor(goodTrainData[,-54])

corrplot(corrMatrix, order = "FPC", method = "color", type = "upper", tl.cex = 0.8, tl.col = rgb(0, 0, 0))

#Figure.1 shows highly correlated variables in dark colors in the graph below
```



#Now Three methods will be applied to model the regressions (in the Train dataset) and the best one

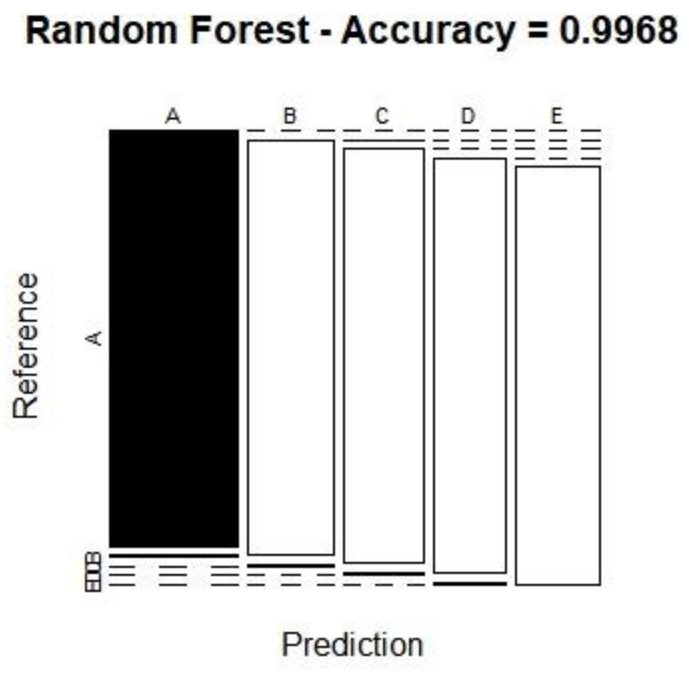
#(with higher accuracy when applied to the Test dataset) will be used for the quiz predictions.

#The methods are: Random Forests, Decision Tree and Generalized Boosted Model, as described below.

#A Confusion Matrix is plotted at the end of each analysis to better visualize the accuracy of the

#models.

```
#a) Method: Random Forest

# Make Random Forest model fit

set.seed(12345)

modDataRF <- trainControl(method="cv", number=3, verboseIter=FALSE)

modFitRandForest <- train(classe ~ ., data=goodTrainData, method="rf",

                     trControl=modDataRF)

modFitRandForest$finalModel


# Prediction on Test dataset

predictRandForest <- predict(modFitRandForest, newdata=goodTestData)

confMatRandForest <- confusionMatrix(predictRandForest, goodTestData$classe)

confMatRandForest


# Plot the matrix results

plot(confMatRandForest$table, col = confMatRandForest$byClass,

  main = paste("Random Forest - Accuracy =",

            round(confMatRandForest$overall['Accuracy'], 4)))



#Random Forest Overall Statistics

#Accuracy : 0.9968

#95% CI : (0.995, 0.9981)

#No Information Rate : 0.2845

#P-Value [Acc > NIR] : < 2.2e-16

#Kappa : 0.9959

#Mcnemar's Test P-Value : NA

#Figure.2 shows the Random Forest Accuracy =0.9968 in the Prediction bar table plot on the Test Data
```



```
# b) Method: Decision Trees

# Make Decision Tree model fit

set.seed(12345)

modFitDecTree <- rpart(classe ~ ., data=goodTrainData, method="class")

fancyRpartPlot(modFitDecTree)


# Prediction on Test dataset

predictDecTree <- predict(modFitDecTree, newdata=goodTestData, type="class")

confMatDecTree <- confusionMatrix(predictDecTree, goodTestData$classe)

confMatDecTree


# Plot the matrix results

plot(confMatDecTree$table, col = confMatDecTree$byClass,

    main = paste("Decision Tree - Accuracy =",

            round(confMatDecTree$overall['Accuracy'], 4)))
```

#Decision Tree Overall Statistics

#Accuracy : 0.7368

#95% CI : (0.7253, 0.748)

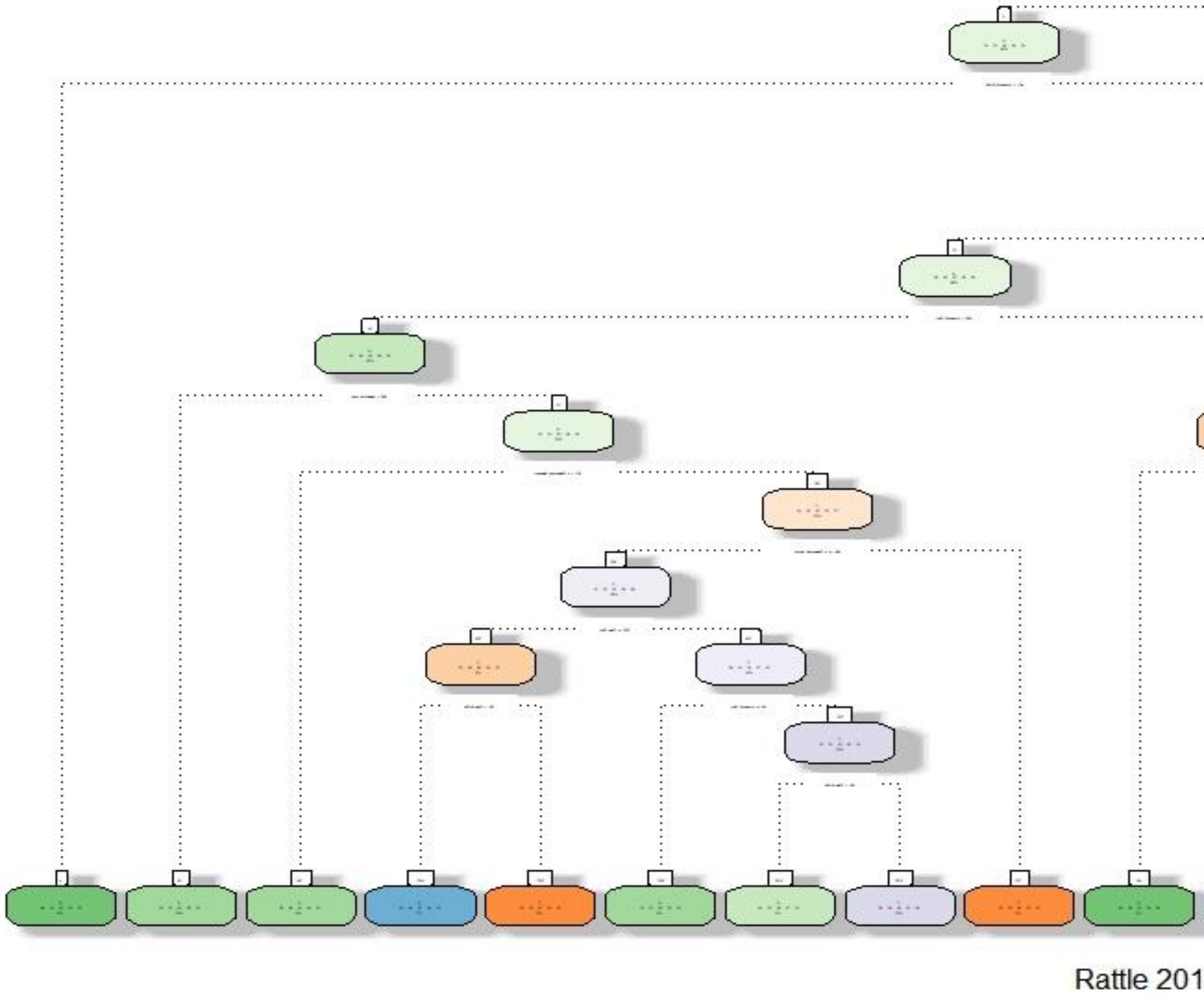
#No Information Rate : 0.2845

#P-Value [Acc > NIR] : < 2.2e-16

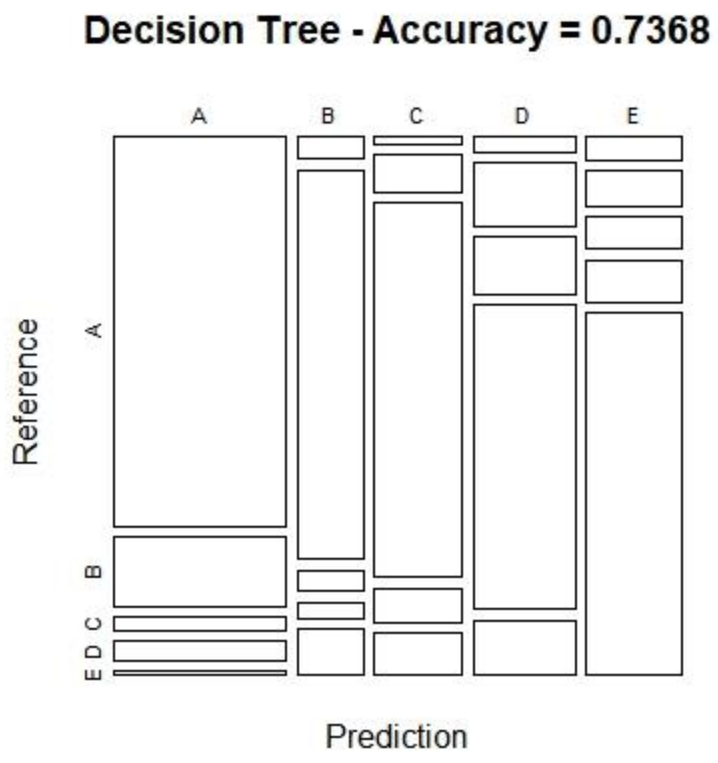
#Kappa : 0.6656

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

#Figure.3 shows the pretty Decision tree also saved more legibly as a PDF named PML_JHU_Final_DT.pdf, included in Github resource

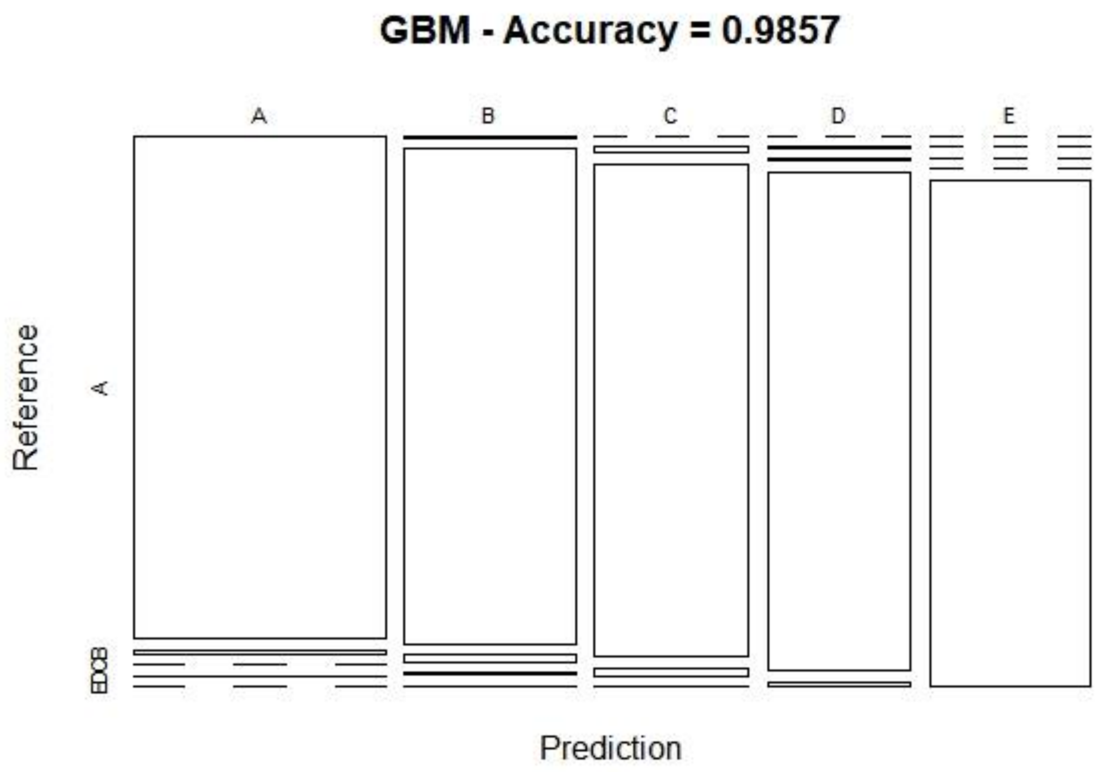


#Figure.4 shows the Decision Tree Accuracy = 0.7368 in the Prediction bar table plot on the Test Data



```
# c) Method: Generalized Boosted Model

# Making the GBM model fit

set.seed(12345)

controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)

modFitGBM  <- train(classe ~ ., data=goodTrainData, 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.
```

```
# Prediction on Test dataset
predictGBM <- predict(modFitGBM, newdata=goodTestData)
confMatGBM <- confusionMatrix(predictGBM, goodTestData$classe)
confMatGBM

# Plot the matrix results
plot(confMatGBM$table, col = confMatGBM$byClass,
    main = paste("GBM - Accuracy =", round(confMatGBM$overall['Accuracy'], 4)))

#GBM 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

#Figure.5 shows the GBM Accuracy = 0.9857 in the Prediction bar table plot on the Test Data
```



#Finally, applying the Selected Model to the Test Data

#The accuracy of the 3 regression modeling methods above are:

#Random Forest : 0.9968

#Decision Tree : 0.7368

#GBM : 0.9857

#In that case, the Random Forest model will be applied to predict the 20 quiz results (testing dataset) as shown below.

```
predictTEST <- predict(modFitRandForest, newdata=testingset)

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
```