Practical Machine Learing: Peer review project

Dewald Olivier

12 February 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://groupware.les.inf.puc-rio.br/har (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 (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://groupware.les.inf.puc-rio.br/har (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.

```
set.seed(12345)
testdata<-read.csv("D:/Programs/R/WD/Assignments/Practical Machine Learning/pml-testing.csv")
traindata<-read.csv("D:/Programs/R/WD/Assignments/Practical Machine Learning/pml-training.cs
v")
inTrain<-createDataPartition(y=traindata$classe,p=0.7, list=FALSE)
training<-traindata[inTrain,]
testing<-traindata[-inTrain,]
dim(training);dim(testing)
```

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

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

The training data contains 160 variables. This data needs to be cleaned up by removing NA values, Near Zero Variance (NZV) variables and ID variables.

```
##remove NZV variables
NZV<-nearZeroVar(training)
training<- training[,-NZV]
testing<-testing[,-NZV]
dim(training)
```

```
## [1] 13737   106
```

```
dim(testing)
```

```
## [1] 5885  106
```

```
##remove NA variables
AllNA<-sapply(training,function(x) mean(is.na(x)))>0.95
training<-training[,AllNA==FALSE]
testing<-testing[,AllNA==FALSE]
dim(training)
```

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

```
##remove ID variables
training<-training[,-(1:5)]
testing<-testing[,-(1:5)]

dim(training)
```

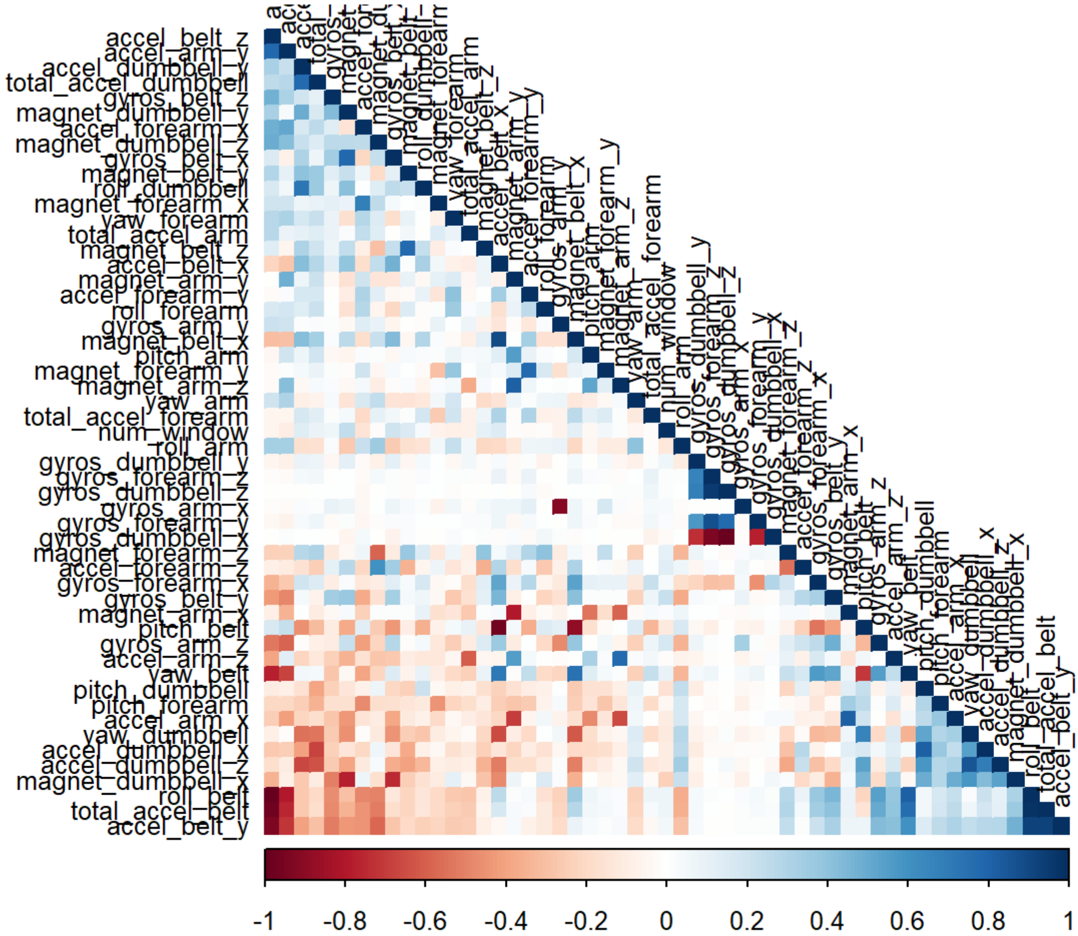
```
## [1] 13737    54
```

After cleaning the data, there are 54 variables remaining.

Correlation Analysis

We need to analyse the teh correlation before performing modeling procedures.

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



The higlhy correlated variables are dark colours in the graph. To make a more compact analysis a Principal Compoonent Analysis could be performed as pre-processing step to the datasets.

Prediction Models

Random Forest

```
set.seed(12345)

## use parallel processing

#step1:
cluster<-makeCluster(detectCores()-1)
registerDoParallel(cluster)
##step2:
controlRF<-trainControl(method="cv", number=5, allowParallel=TRUE)
##step3:
rf<-train(classe~., data=training, method="rf",trControl=controlRF )
##step4:
stopCluster(cluster)
registerDoSEQ()
rf$resample
```

```
##    Accuracy     Kappa Resample
## 1 0.9974518 0.9967767    Fold1
## 2 0.9967249 0.9958575    Fold3
## 3 0.9967225 0.9958537    Fold2
## 4 0.9981812 0.9976994    Fold5
## 5 0.9967237 0.9958562    Fold4
```

```
confusionMatrix.train(rf)
```

```
## Cross-Validated (5 fold) Confusion Matrix
## 
## (entries are percentual average cell counts across resamples)
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 28.4  0.1  0.0  0.0  0.0
##          B  0.0 19.3  0.0  0.0  0.0
##          C  0.0  0.0 17.4  0.1  0.0
##          D  0.0  0.0  0.0 16.3  0.1
##          E  0.0  0.0  0.0  0.0 18.3
## 
##  Accuracy (average) : 0.9972
```

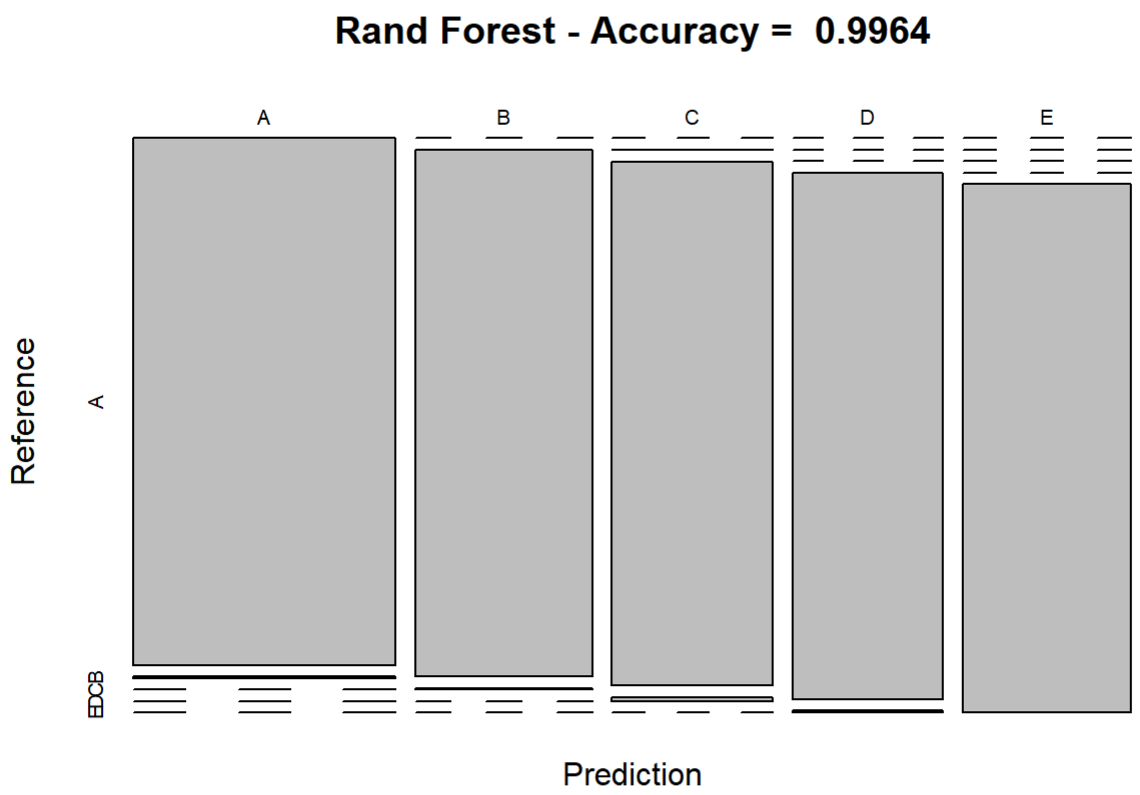
```
##Predicting new values

predrf<-predict(rf,newdata=testing);testing$predrfRight<-predrf==testing$classe
confMRF<-confusionMatrix(predrf,testing$classe)
confMRF
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    3    0    0
##          C    0    1 1023    8    0
##          D    0    0    0  956    4
##          E    0    0    0    0 1078
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9964          
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9955          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9971   0.9917   0.9963
## Specificity            0.9988   0.9994   0.9981   0.9992   1.0000
## Pos Pred Value         0.9970   0.9974   0.9913   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9994   0.9984   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1738   0.1624   0.1832
## Detection Prevalence   0.2853   0.1930   0.1754   0.1631   0.1832
## Balanced Accuracy      0.9994   0.9971   0.9976   0.9954   0.9982
```

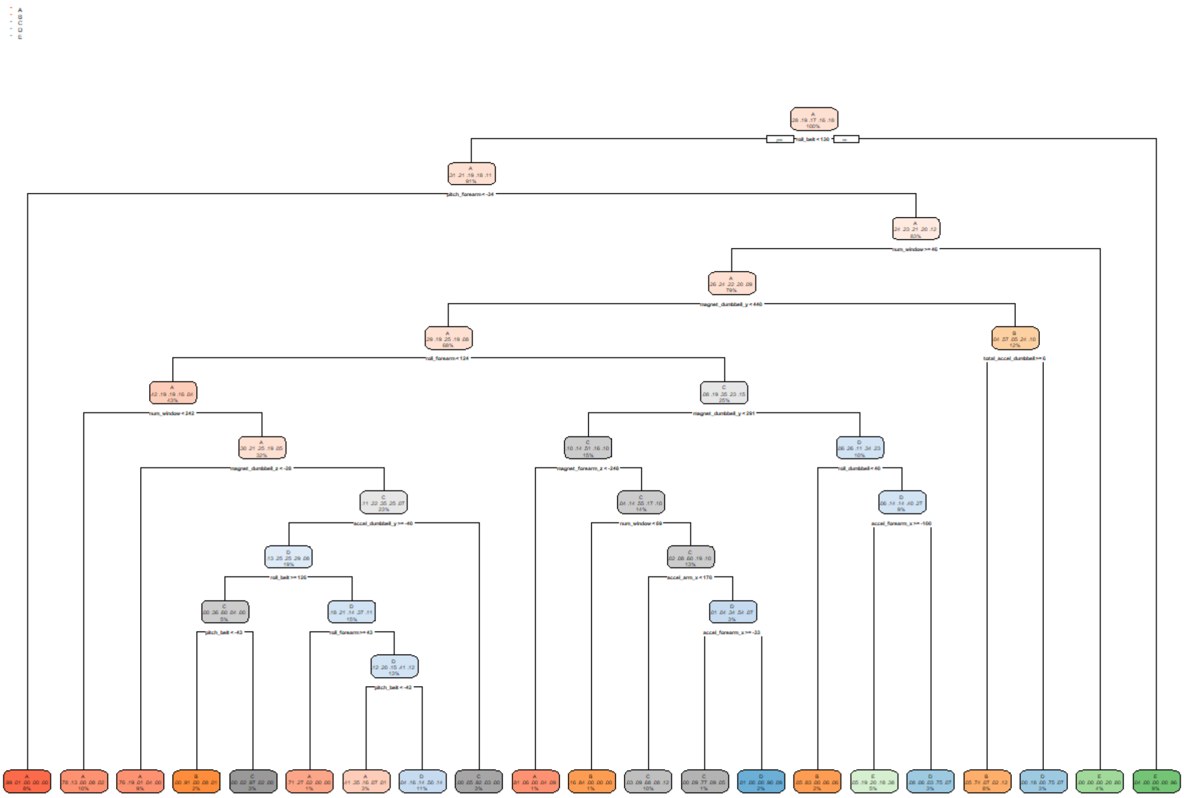
```
##plot matrix results
plot(confMRF$table,COL=confMRF$byClass, main=paste("Rand Forest - Accuracy = ", round(confMRF
$overall["Accuracy"],4)))
```

```
## Warning: In mosaicplot.default(x, xlab = xlab, ylab = ylab, ...) :
##  extra argument 'COL' will be disregarded
```



Decision Trees

```
set.seed(12345)
dt<-rpart(classe~., data=training, method="class")
rpart.plot(dt)
```

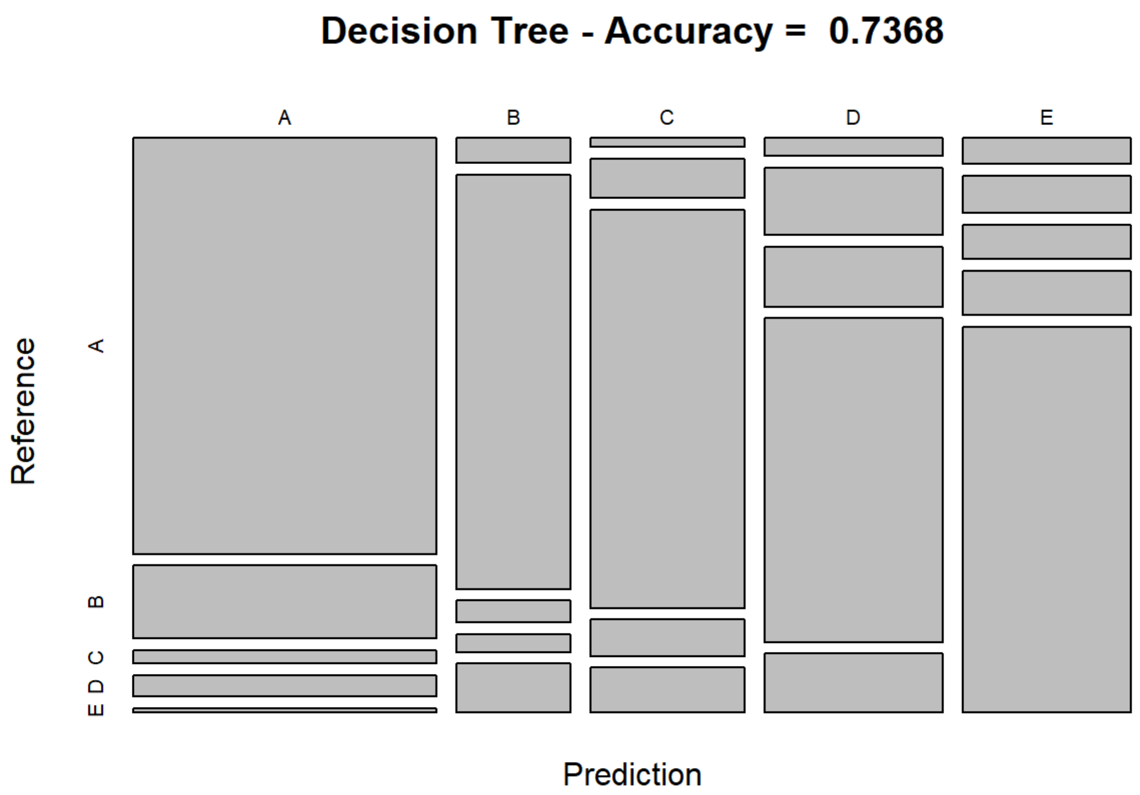


```
##Predicitng values
predDT<-predict(dt, newdata=testing, type="class")
confMDT<-confusionMatrix(predDT,testing$classe)
confMDT
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  269   51   79   16
##          B   35  575   31   25   68
##          C   17   73  743   68   84
##          D   39  146  130  702  128
##          E   53   76   71   90  786
##
## 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
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9140  0.50483   0.7242   0.7282   0.7264
## Specificity            0.9014  0.96650   0.9502   0.9100   0.9396
## Pos Pred Value         0.7866  0.78338   0.7543   0.6131   0.7305
## Neg Pred Value         0.9635  0.89051   0.9422   0.9447   0.9384
## Prevalence             0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate         0.2600  0.09771   0.1263   0.1193   0.1336
## Detection Prevalence   0.3305  0.12472   0.1674   0.1946   0.1828
## Balanced Accuracy      0.9077  0.73566   0.8372   0.8191   0.8330
```

```
##Plot Confusion Matrix Results
plot(confMDT$table,COL=confMDT$byClass, main=paste("Decision Tree - Accuracy = ", round(confM
DT$overall["Accuracy"],4)))
```

```
## Warning: In mosaicplot.default(x, xlab = xlab, ylab = ylab, ...) :
##  extra argument 'COL' will be disregarded
```



Generalized Boosted Model

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

gbm<-train(classe ~ ., data=training, method = "gbm",
                    trControl = controlGBM, verbose = FALSE)
gbm$finalModel
```

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

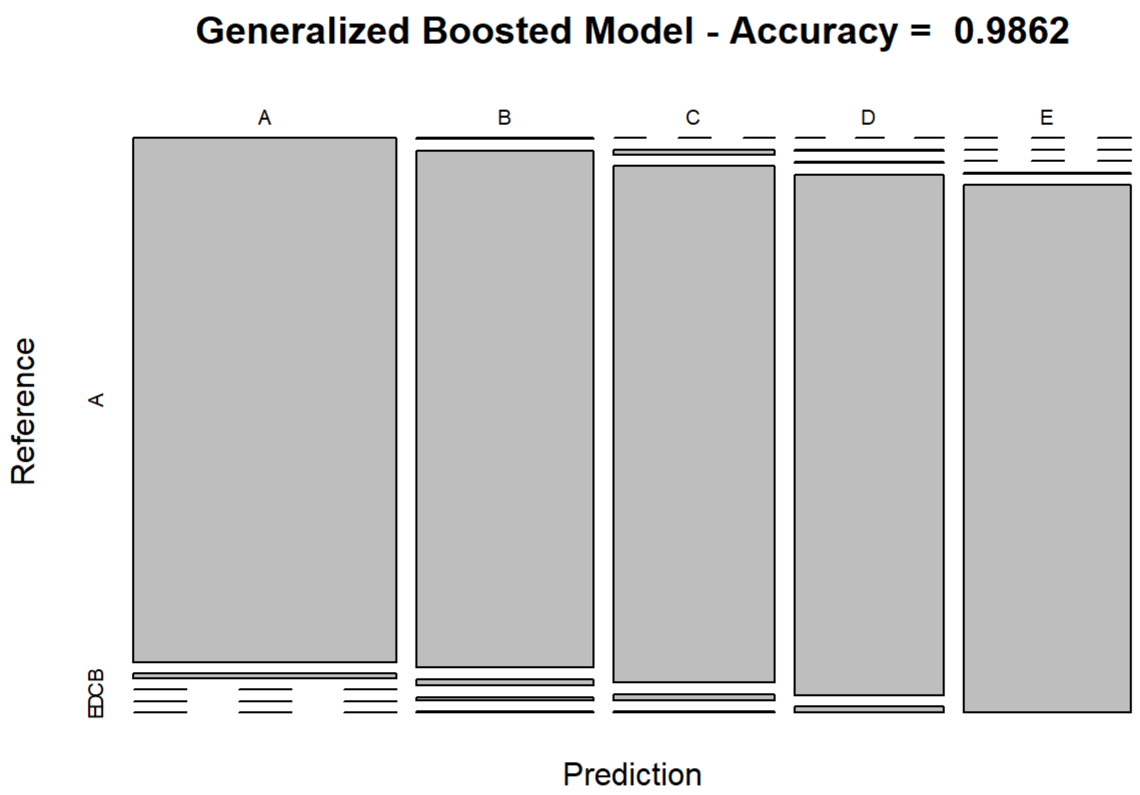
```
##Predicting Values
predGBM <- predict(gbm, newdata=testing)
confMGBM <- confusionMatrix(predGBM, testing$classe)
confMGBM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672   15    0    0    0
##          B    2 1112   14    7    2
##          C    0   10 1010   12    2
##          D    0    2    2  943   11
##          E    0    0    0    2 1067
##
## Overall Statistics
##
##                Accuracy : 0.9862
##                  95% CI : (0.9829, 0.9891)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9826
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9763   0.9844   0.9782   0.9861
## Specificity            0.9964   0.9947   0.9951   0.9970   0.9996
## Pos Pred Value         0.9911   0.9780   0.9768   0.9843   0.9981
## Neg Pred Value         0.9995   0.9943   0.9967   0.9957   0.9969
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1890   0.1716   0.1602   0.1813
## Detection Prevalence   0.2867   0.1932   0.1757   0.1628   0.1816
## Balanced Accuracy      0.9976   0.9855   0.9897   0.9876   0.9929
```

```
##Plotting the GBM

plot(confMGBM$table,COL=confMGBM$byClass, main=paste("Generalized Boosted Model - Accuracy =
 ", round(confMGBM$overall["Accuracy"],4)))
```

```
## Warning: In mosaicplot.default(x, xlab = xlab, ylab = ylab, ...) :
##  extra argument 'COL' will be disregarded
```

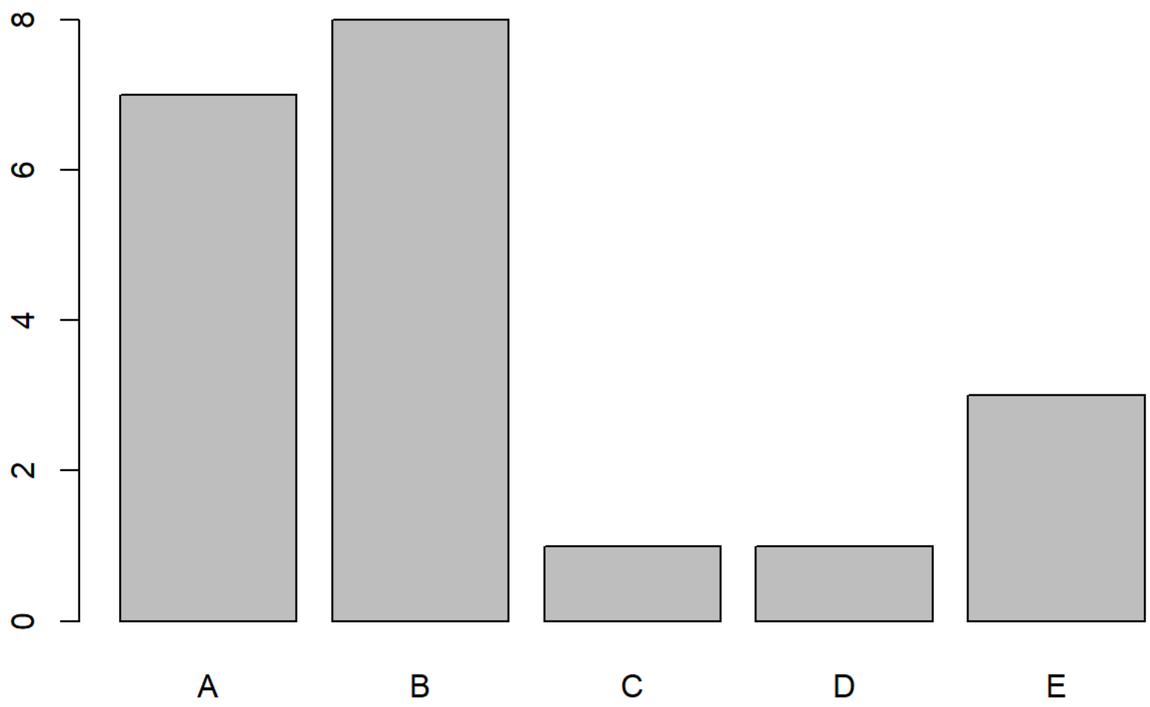


Apply The Selected Model to The Test Data

The accuracy of the 3 regression modeling methods are: a. Random Forest: b. Decision Trees: c. Generalized Boosted Model:

It appears that Random Forest is te most accurate model to predict the 20 quiz results.

```
predTEST<-predict(rf,newdata=testdata)
plot(predTEST)
```



chunk to prevent printing of the R code that generated the plot.