Prediction Assignment - Practical Machine Learning

Pragya Kalia

11/19/2017

This document is part of an assignemnt from Coursera’s course on Practical Machine Learning module of Data Science Specialization. This is built up in RStudio using the knitr functions & is meant to be published in HTML format.

(A) Background & 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 ?V 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, our 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).

(B) Objective-

The goal of this project is to predict the manner in which the subjects did the exercise. This is the “classe” variable in the training set. We are free to use any of the other variables to predict with. The machine algorithm explained here is applied to the 20 test cases vaialble in the test data and the predictions are submitted in appropriate expected format.

(C) Data Description : Loading & Exploring-

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)

Credits :

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). Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012.

My sincere thanks to the above mentioned authors for being so generous in allowing their data to be used for this kind of assignment.

A short description of the datasets content from the authors?? website:

??Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg)."

D) Packages, Data Loading & Cleaning-

We first upload required packages & libraries for the anaysis

```r
list = ls()
setwd("~/Documents/PracticalMachineLearning")
library(knitr)
library(caret)
library(randomForest)
library(rpart)
library(rpart.plot)
library(ggplot2)
library(rattle.data)
library(corrplot)
set.seed(12345)
```

```r
URLTrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
URLTest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

# download
training <- read.csv(url(URLTrain))
testing <- read.csv(url(URLTest))

# createDataPartition
inTrain <- createDataPartition(training$classe, p=0.7, list = FALSE)
TrngSet <- training[inTrain, ]
TestSet <- training[-inTrain, ]
dim(TrngSet)
```

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

```
dim(TestSet)
```

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

```
#remove variables with Nearly Zero Variance
nzv <- nearZeroVar(TrngSet)
TrngSet <- TrngSet[, -nzv]
TestSet <- TestSet[, -nzv]
dim(TrngSet)
```

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

```
dim(TestSet)
```

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

```
# remove variables with mostly NAs
allNA <- sapply(TrngSet, function(x) mean(is.na(x))) > 0.95
TrngSet <- TrngSet[, allNA == FALSE]
TestSet <- TestSet[, allNA == FALSE]
dim(TrngSet)
```

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

```
dim(TestSet)
```

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

```
#remove variable with only identification
TrngSet <- TrngSet[, -(1:5)]
TestSet <- TestSet[, -(1:5)]
dim(TrngSet)
```

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

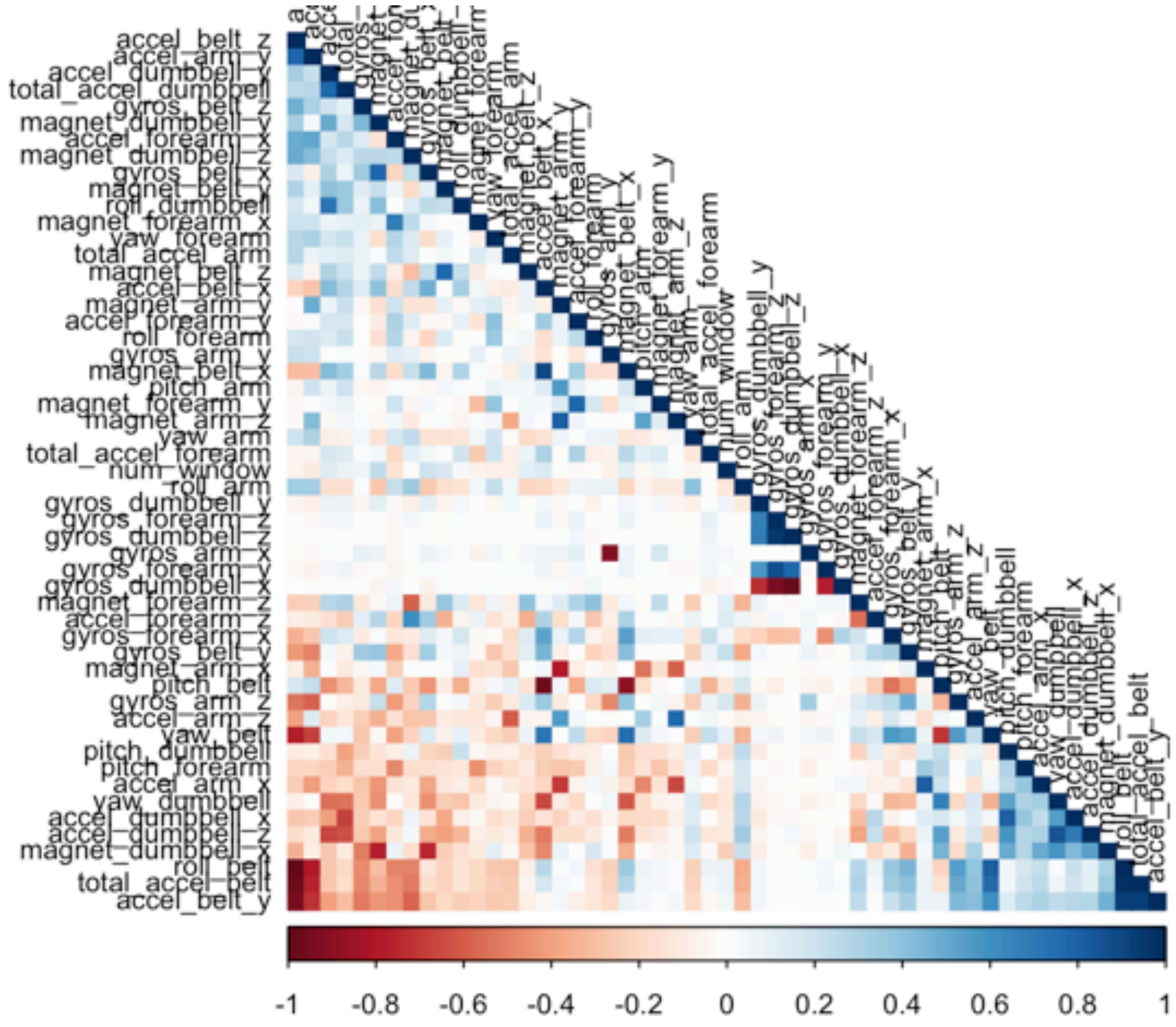
```
dim(TestSet)
```

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

```
# Only 54 variables left post data cleaning
```

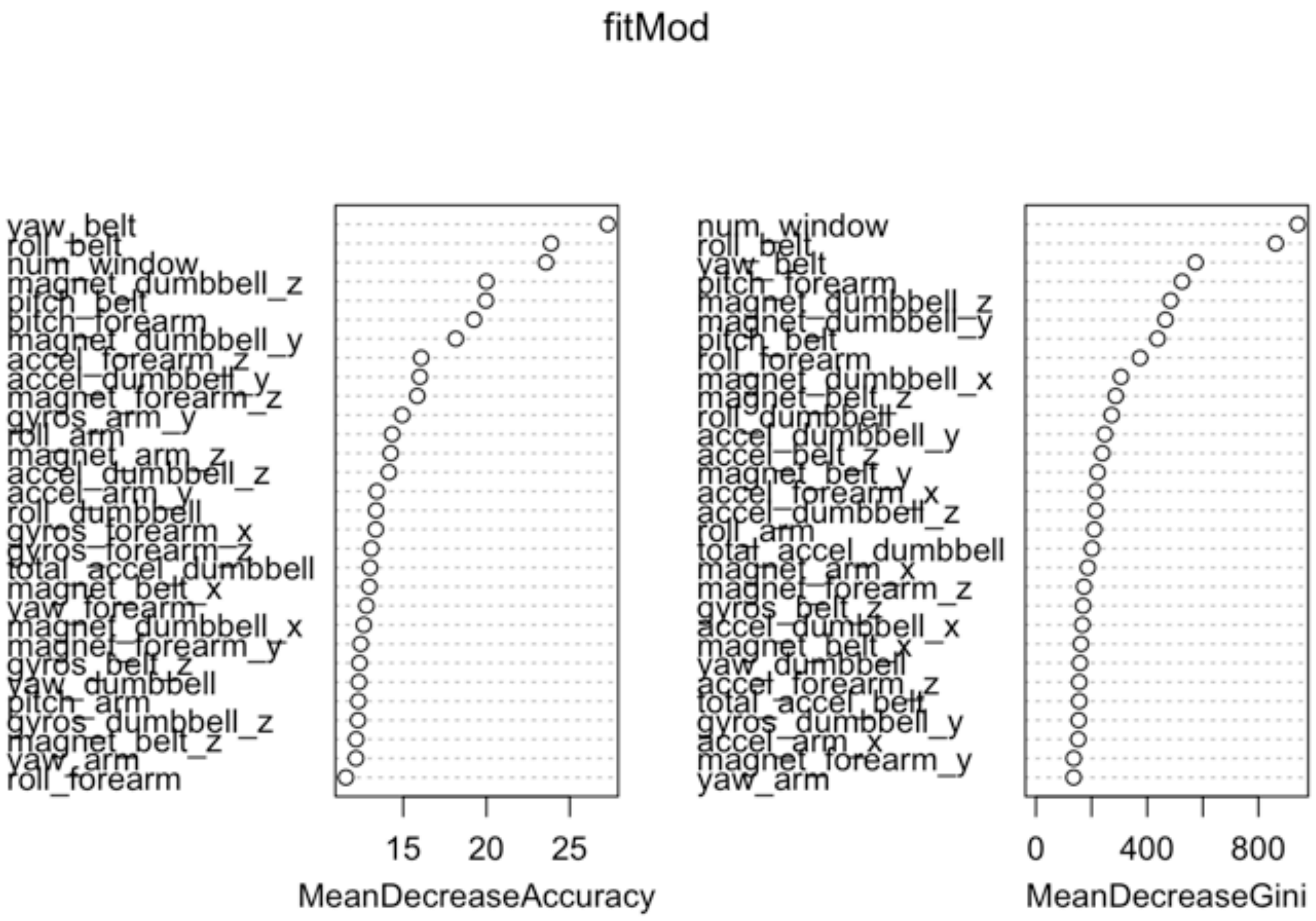
(E) Correlation Analysis-

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



```r
# Highly correlated variables shown in dark colours in the graph
```

```r
library(randomForest)
set.seed(12345)
fitMod<- randomForest(classe~. , data = TrngSet, importance = TRUE, ntree = 100)
varImpPlot(fitMod)
```



(F) Prediction Models-

Random Forests(RF), Generalized Boosted Model(GBM) & Decicion Tree models will be used to model the regression in the Training data. Method with the highest accuracy will be then used for quiz predictions. At the end of each analysis, Confusion Matrix is plotted for better comprehension & visualization purpose.

(i) RF Method

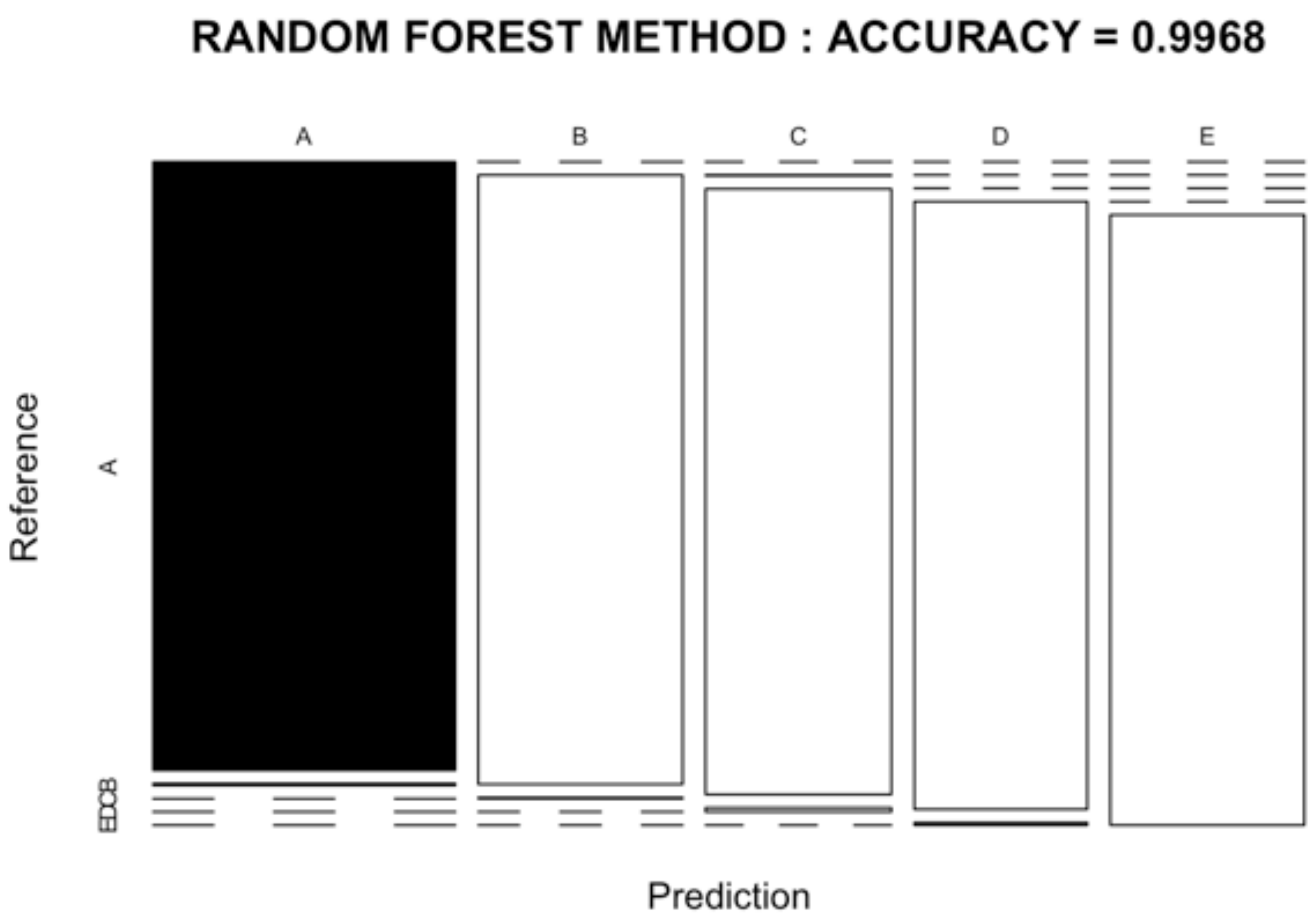
```
set.seed(12345)
ConRF <- trainControl(method = "cv", number = 3, verboseIter = FALSE)
modelfitRF <- train(classe ~., data = TrngSet, method = "rf", trControl = ConRF)
modelfitRF$finalModel
```

```
## 
## 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: 0.19%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    6 2651    1    0    0 0.0026335591
## C    0    6 2390    0    0 0.0025041736
## D    0    0    8 2244    0 0.0035523979
## E    0    0    0    3 2522 0.0011881188
```

```
# Prediction (Test Data Set)
PreRF <- predict(modelfitRF, newdata = TestSet)
ConMatRF <- confusionMatrix(PreRF, TestSet$classe)
ConMatRF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    2    0    0
##          C    0    1 1024    7    0
##          D    0    0    0  957    4
##          E    0    0    0    0 1078
##
## 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
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9981   0.9927   0.9963
## Specificity            0.9988   0.9996   0.9984   0.9992   1.0000
## Pos Pred Value         0.9970   0.9982   0.9922   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9996   0.9986   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1740   0.1626   0.1832
## Detection Prevalence   0.2853   0.1929   0.1754   0.1633   0.1832
## Balanced Accuracy      0.9994   0.9972   0.9982   0.9960   0.9982
```

```
# Plotting results
plot(ConMatRF$table, col = ConMatRF$byClass, main = paste("RANDOM FOREST METHOD :
ACCURACY =", round(ConMatRF$overall['Accuracy'], 4)))
```



(ii) Generalized Boosted Model (GBM)

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

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

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

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

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

```
## Loaded gbm 2.1.3
```

```
modelfitGBM$finalModel
```

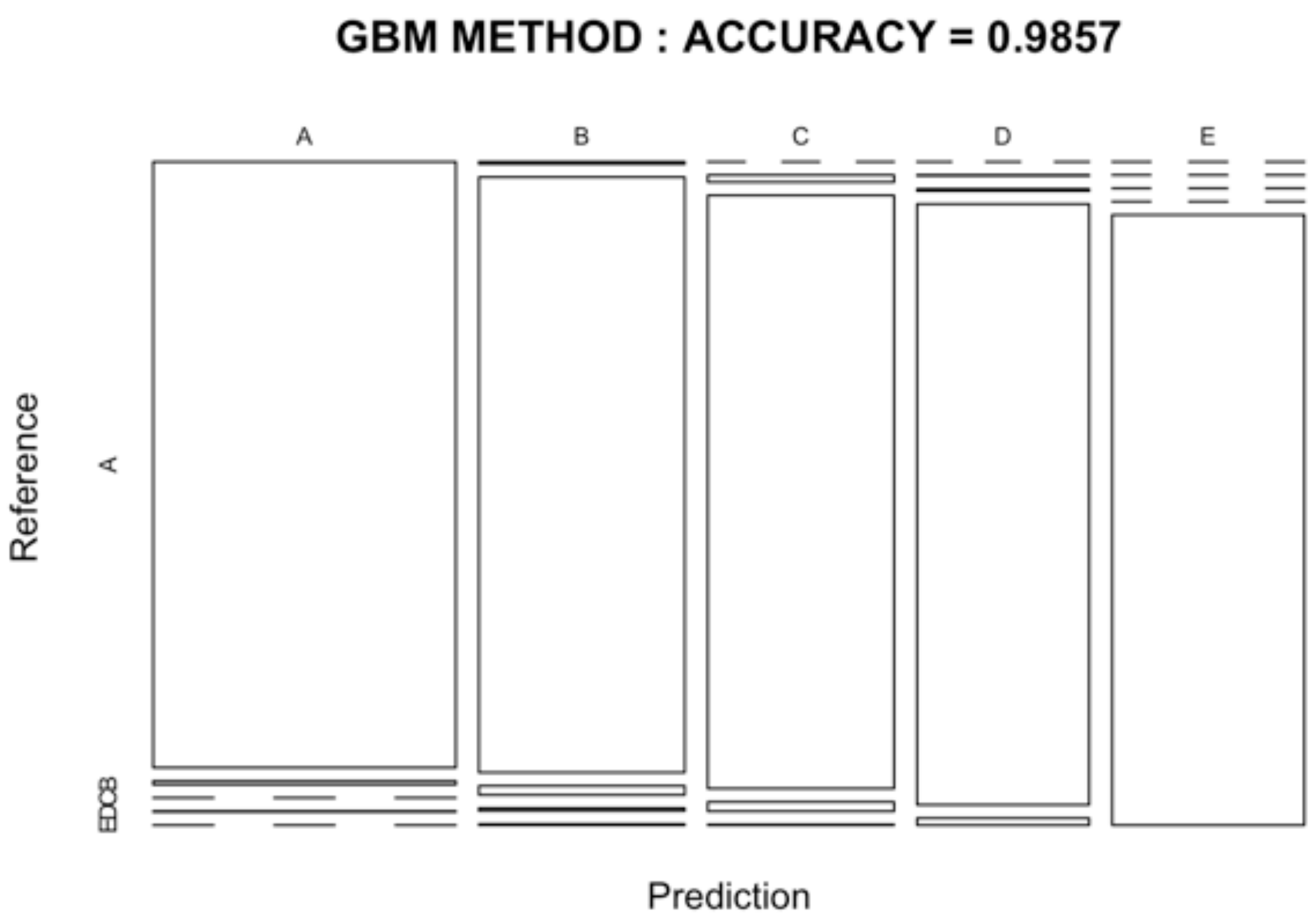
```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 41 had non-zero influence.
```

A gradient boosted model with multinomial loss function.150 iterations were performed.There were 53 predictors of which 41 had non-zero influence.

```
#Prediction (Test Data Set)
PreGBM <- predict(modelfitGBM, newdata = TestSet)
ConMatGBM <- confusionMatrix(PreGBM, TestSet$classe)
ConMatGBM
```

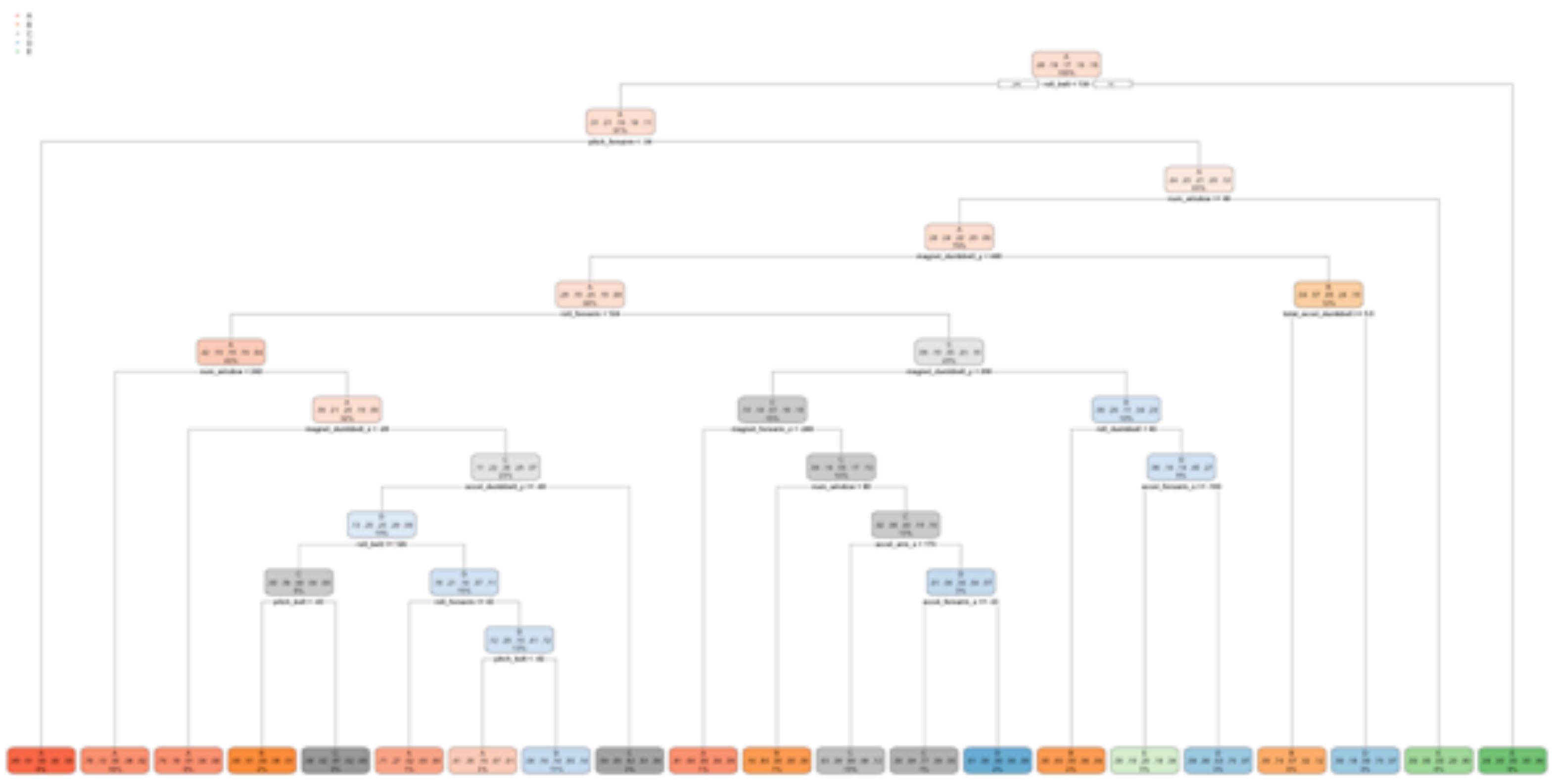
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670   10    0    2    0
##          B    4 1116   17    5    2
##          C    0   12 1006   16    1
##          D    0    1    3  941   11
##          E    0    0    0    0 1068
##
## 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            0.9976   0.9798   0.9805   0.9761   0.9871
## Specificity            0.9972   0.9941   0.9940   0.9970   1.0000
## Pos Pred Value         0.9929   0.9755   0.9720   0.9843   1.0000
## Neg Pred Value         0.9990   0.9951   0.9959   0.9953   0.9971
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1896   0.1709   0.1599   0.1815
## Detection Prevalence   0.2858   0.1944   0.1759   0.1624   0.1815
## Balanced Accuracy      0.9974   0.9870   0.9873   0.9865   0.9935
```

```
# Plotting results
plot(ConMatGBM$table, col = ConMatGBM$byClass, main = paste("GBM METHOD : ACCURACY
=", round(ConMatGBM$overall['Accuracy'], 4)))
```



(iii) Decision Tree Model(DT)

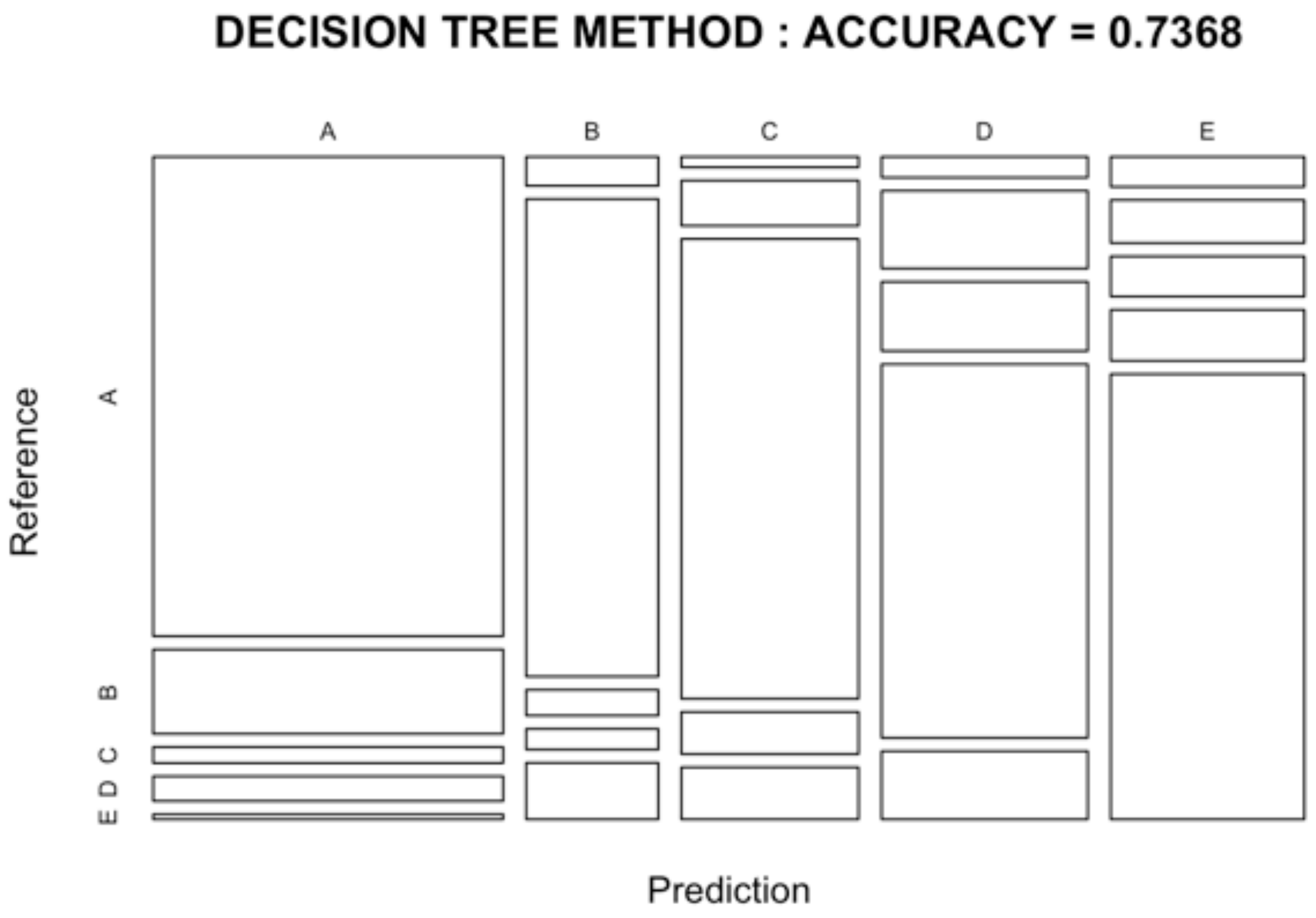
```
set.seed(12345)
modelfitDT <- rpart(classe ~., data = TrngSet, method = "class")
rpart.plot(modelfitDT)
```



```
#Prediction (Test Data Set)
PreDT <- predict(modelfitDT, newdata = TestSet, type = "class")
ConMatDT <- confusionMatrix(PreDT, TestSet$classe)
ConMatDT
```

```
## 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
```

```
# Plotting results
plot(ConMatDT$table, col = ConMatDT$byClass, main = paste("DECISION TREE METHOD :
ACCURACY =", round(ConMatDT$overall['Accuracy'], 4)))
```



(G) Choosen Model's application on the Test Data-

Given below are the accuracy of the three regression models: 1. RF: 0.9968 2. GBM: 0.9857 3. Decision Tree: 0.7368 Since RF has the best accuracy, RF model will be applied to predict the 20 quiz results.

```
PreTest <- predict(modelfitRF, newdata = testing)
PreTest
```

```
##  [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
```

(H) Conclusion-

Using RF method with 99.68% accuracy (0.23% of out of sample error), we can comfortably conclude that we could accurately predict the classification of 20 observations.