Practical Machine Learning (Prediction) - Assignment

Nhlakanipho

21 December 2017

Synopsis

This paper aims at analyzing the training patterns for 6 diffent users doing Weight Lifting execises. The data set used in this paper was collected by Groupware@LES for their project in Human Activity Recognition which can be found, http://groupware.les.inf.puc-rio.br/har. After exploring, cleaning and applying prediction models on this dataset, the results produced the most likely prediction given specific datapoints.

Data

Data load from Groupware's project dataset as our source data. There is a Training set and a Testing set which we will use for our final predictions.

```
if(!file.exists("exerciseTrainingData.csv"))
{
  download.file(
     "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
     ,destfile="./exerciseTrainingData.csv"
     ,method="auto"
  )
}

if(!file.exists("exerciseTestData.csv"))
{
  download.file(
    "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
    ,destfile="./exerciseTestData.csv"
    ,method="auto"
  )
}

exerciseTrainingData <- read.csv(file = "./exerciseTrainingData.csv", sep = ",", header = TRUE, na.st

exerciseTestData <- read.csv(file = "./exerciseTestData.csv", sep = ",", header = TRUE, na.strings =
```

In preparation of the data, the Trainng had to be further split to a training set and a validation set which we will use to try verify which model/prediction works better for preditions. A 70% split was used on the training data to get these sets.

```
library(caret)
```

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

```
set.seed(1654)
inTrain <- createDataPartition(y=exerciseTrainingData$classe,p=0.70, list = FALSE)
```

```
myTrainData <- exerciseTrainingData[inTrain,]
myTestData <- exerciseTrainingData[-inTrain,]
```

Data Exploration and Cleaning

In exploration of the data there was a view on some columns having NA (Index - Diagram1) and some having a very low variance to be considered as they didn't change much and had a potential of misrepresenting the predictions. Looking at the first 6 values of the data, one can identify all descriptive columns that don't influence or models but just identify the users.

```
#head(myTrainData)
#summary(myTrainData)


myTrainDataDescriptive <- myTrainData[,-(1:5)]
myTestDataDescriptive <- myTestData[,-(1:5)]

myTrainDataDescriptive$new_window <- sapply(myTrainDataDescriptive$new_window, function(x){if(x=='no'){
myTestDataDescriptive$new_window <- sapply(myTestDataDescriptive$new_window, function(x){if(x=='no'){x=


dim(myTrainDataDescriptive)
```

```
## [1] 13737   155
```

```
nzvTraining <- nearZeroVar(myTrainDataDescriptive[,-155])
myTrainDataDescriptive <- myTrainDataDescriptive[,-nzvTraining]
myTestDataDescriptive <- myTestDataDescriptive[,-nzvTraining]

rowcount <- nrow(myTrainDataDescriptive)
lenghtDescription <- length(myTrainDataDescriptive)
naData <- colMeans(is.na(myTrainDataDescriptive))

exData <- c()
for(i in 1:lenghtDescription)
{
  if(naData[i] >= .7)
  {
    exData <- append(exData, i)
  }
}

myTrainDataDescriptive <- myTrainDataDescriptive[,-exData]
myTestDataDescriptive <- myTestDataDescriptive[,-exData]

rm(rowcount,lenghtDescription,naData,exData)
```

Predictions

This section applied model fits and runs predictions on the validation sets to try get the most closely aligned preditions.The models of choice used for identifying the model with the best out of sample error accuracy are

Classification Trees, Random Forest and Grdient Boosting Model. Thses are investigated below.

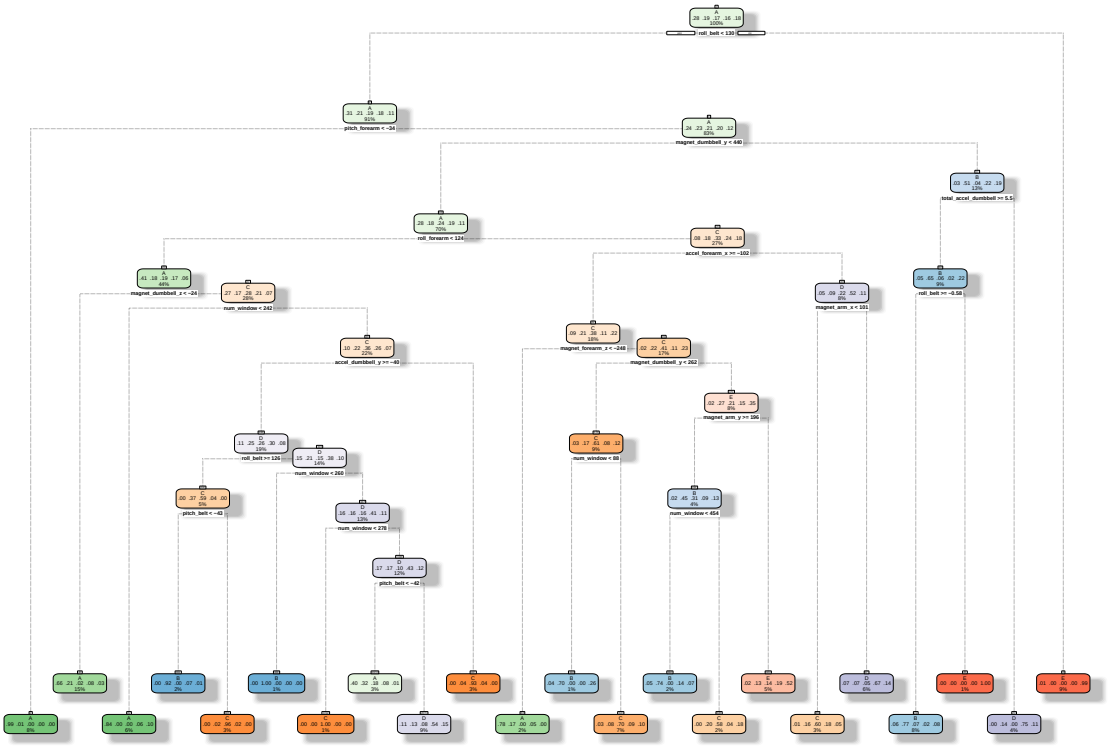
Classication Tree

In the first test, a classification tree in the form of a regression tree is investigated. By default Rpart using a 10 fold cross validation.

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
set.seed(4154)
rpartModelFit <- rpart(classe~., data = myTrainDataDescriptive, method = "class")
rpartPredict <- predict(rpartModelFit, myTestDataDescriptive, type = "class")
rpartConf <- confusionMatrix(rpartPredict, myTestDataDescriptive$classe)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



Rattle 2017−Dec−21 15:10:00 Nhlakanipho.Muholi1

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1521  268   49  131   77
##          B   29  580   31   31   80
##          C   16  109  826   82   67
##          D   98  138   85  673  154
##          E   10   44   35   47  704
```

```
##
## Overall Statistics
##
##                Accuracy : 0.7314
##                  95% CI : (0.7198, 0.7426)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.658
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9086  0.50922   0.8051   0.6981   0.6506
## Specificity            0.8753  0.96397   0.9436   0.9035   0.9717
## Pos Pred Value         0.7434  0.77230   0.7509   0.5862   0.8381
## Neg Pred Value         0.9601  0.89112   0.9582   0.9386   0.9251
## Prevalence             0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate         0.2585  0.09856   0.1404   0.1144   0.1196
## Detection Prevalence   0.3477  0.12761   0.1869   0.1951   0.1427
## Balanced Accuracy      0.8920  0.73659   0.8743   0.8008   0.8112
##
## Classification tree:
## rpart(formula = classe ~ ., data = myTrainDataDescriptive, method = "class")
##
## Variables actually used in tree construction:
##  [1] accel_dumbbell_y     accel_forearm_x      magnet_arm_x
##  [4] magnet_arm_y         magnet_dumbbell_y    magnet_dumbbell_z
##  [7] magnet_forearm_z     num_window           pitch_belt
## [10] pitch_forearm        roll_belt            roll_forearm
## [13] total_accel_dumbbell
##
## Root node error: 9831/13737 = 0.71566
##
## n= 13737
##
##          CP nsplit rel error  xerror      xstd
## 1  0.118198      0   1.00000 1.00000 0.0053780
## 2  0.060184      1   0.88180 0.88180 0.0057525
## 3  0.039670      4   0.70125 0.70156 0.0059609
## 4  0.034279      6   0.62191 0.62130 0.0059243
## 5  0.031024      7   0.58763 0.59038 0.0058890
## 6  0.022683      8   0.55661 0.56017 0.0058427
## 7  0.020140     11   0.48795 0.48754 0.0056823
## 8  0.016682     12   0.46781 0.47157 0.0056373
## 9  0.015766     13   0.45112 0.45570 0.0055890
## 10 0.015563     14   0.43536 0.45194 0.0055769
## 11 0.014851     15   0.41979 0.44756 0.0055627
## 12 0.012105     16   0.40494 0.42142 0.0054716
## 13 0.011799     17   0.39284 0.40423 0.0054058
## 14 0.011698     18   0.38104 0.39264 0.0053587
## 15 0.010070     20   0.35764 0.37046 0.0052623
```

```
## 16 0.010000     21   0.34757 0.36161 0.0052215
```

The results and output diagram show the resulting choices of each tree branch with the specific node criterias. The classification tree resulted in a 73% accuracy rate and a out of sample error rate of 17% on the validation dataset. Rpart used at most 13 variable for the tree classification, which is identified as the most relevant classifiers for the model.

Random Forest

Secondly an investiagtion into Random Forest Model is performed to see how it performs in terms of predictions. 3 folds cross validation was used for perfoamance purposes.

```
library("randomForest")
```

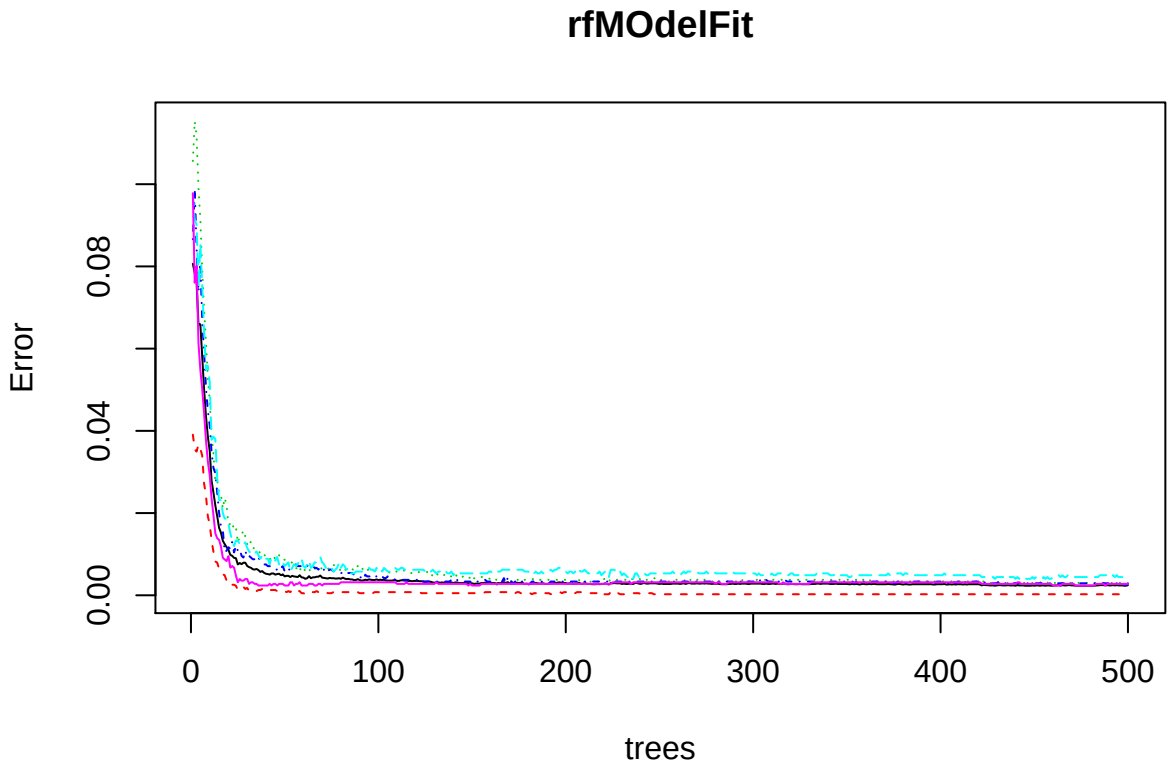
```
## randomForest 4.6-12
```

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

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

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

```
set.seed(3154)
rfMOdelFit <- randomForest(
                            classe~.
                            ,data= myTrainDataDescriptive
                            ,importance = T
                            ,trControl = trainControl(method = "cv", number = 3)
                           )
plot(rfMOdelFit)
```




```
rfPredict <- predict(rfMOdelFit,myTestDataDescriptive)

rfConf <- confusionMatrix(rfPredict, myTestDataDescriptive$classe)
rfConf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    6    0    0    0
##          B    0 1133    7    0    0
##          C    0    0 1019    9    0
##          D    0    0    0  955    2
##          E    0    0    0    0 1080
##
## Overall Statistics
##
##                Accuracy : 0.9959
##                  95% CI : (0.9939, 0.9974)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9948
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9932   0.9907   0.9982
## Specificity            0.9986   0.9985   0.9981   0.9996   1.0000
## Pos Pred Value         0.9964   0.9939   0.9912   0.9979   1.0000
## Neg Pred Value         1.0000   0.9987   0.9986   0.9982   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1732   0.1623   0.1835
## Detection Prevalence   0.2855   0.1937   0.1747   0.1626   0.1835
## Balanced Accuracy      0.9993   0.9966   0.9957   0.9951   0.9991
```

The results and diagram show how increasing the number of trees decreases the error rate for each classe predition. Random Forest predictions resulted in a 99.6% accuracy rate and a out of sample error rate of 0.4% on the validation dataset.The variable importances for the model is shown in (Index - Diagram3).

Gradient Boosting Model

Lastly an investiagtion into Gradient Boosting Model is performed to see how it performs in terms of predictions. 3 folds cross validation was used for performance purposes.

```
set.seed(1154)
gbmModelFit <- train(classe~.,method = 'gbm',data = myTrainDataDescriptive, verbose = FALSE,trControl =
```

```
## Loading required package: gbm
```

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

```
## Warning: package 'survival' was built under R version 3.3.3
```

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

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

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

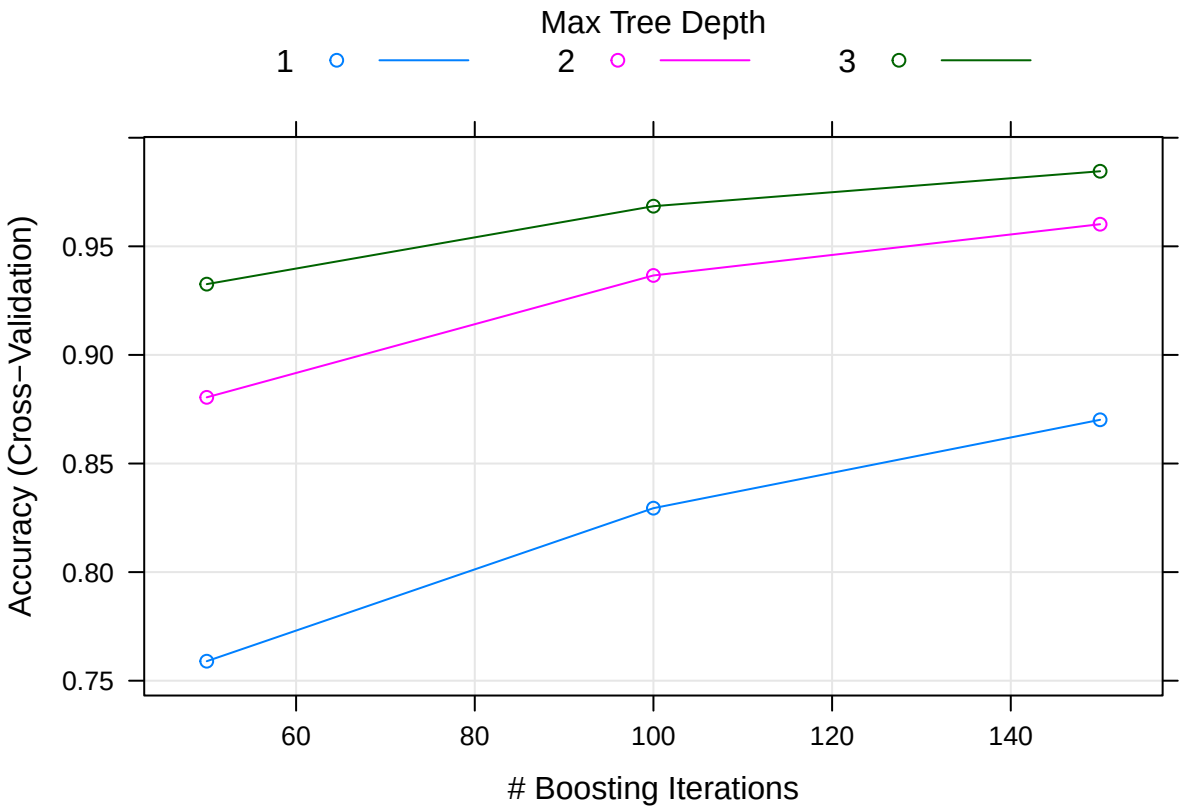
```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
## Loading required package: plyr
```

```
## Warning: package 'plyr' was built under R version 3.3.3
```

```
gbmPredict <- predict(gbmModelFit,myTestDataDescriptive)
```




```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1669   18    0    0    1
##          B    4 1107    6    3    4
##          C    0   13 1017    9    2
##          D    1    1    3  950   11
##          E    0    0    0    2 1064
##
## Overall Statistics
##
##                Accuracy : 0.9867
##                  95% CI : (0.9835, 0.9895)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9832
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9719   0.9912   0.9855   0.9834
## Specificity            0.9955   0.9964   0.9951   0.9967   0.9996
## Pos Pred Value         0.9887   0.9849   0.9769   0.9834   0.9981
```

```
## Neg Pred Value         0.9988   0.9933   0.9981   0.9972   0.9963
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2836   0.1881   0.1728   0.1614   0.1808
## Detection Prevalence   0.2868   0.1910   0.1769   0.1641   0.1811
## Balanced Accuracy      0.9963   0.9842   0.9931   0.9911   0.9915
```

The results and diagram show how increasing the number of trees and increasing the boosting increases overall accuracy in preditions. Gradient Boosting Model predictions resulted in a 98.7% accuracy rate and a out of sample error rate of 1.3% on the validation dataset. By default GBM uses the full dataset for predictions under it's cross validation methods.

Conclusion

Random Forest Show the highest accuracy and according to accuracy seen for each Confusion Matrix and further check (Index - Diagram 2) which proves there is a high specificy and high sensitivity overall for the Random Forestcompared to the rest of the data.

Index

Diagram1

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.4, built: 2015-12-05)
## ## Copyright (C) 2005-2017 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

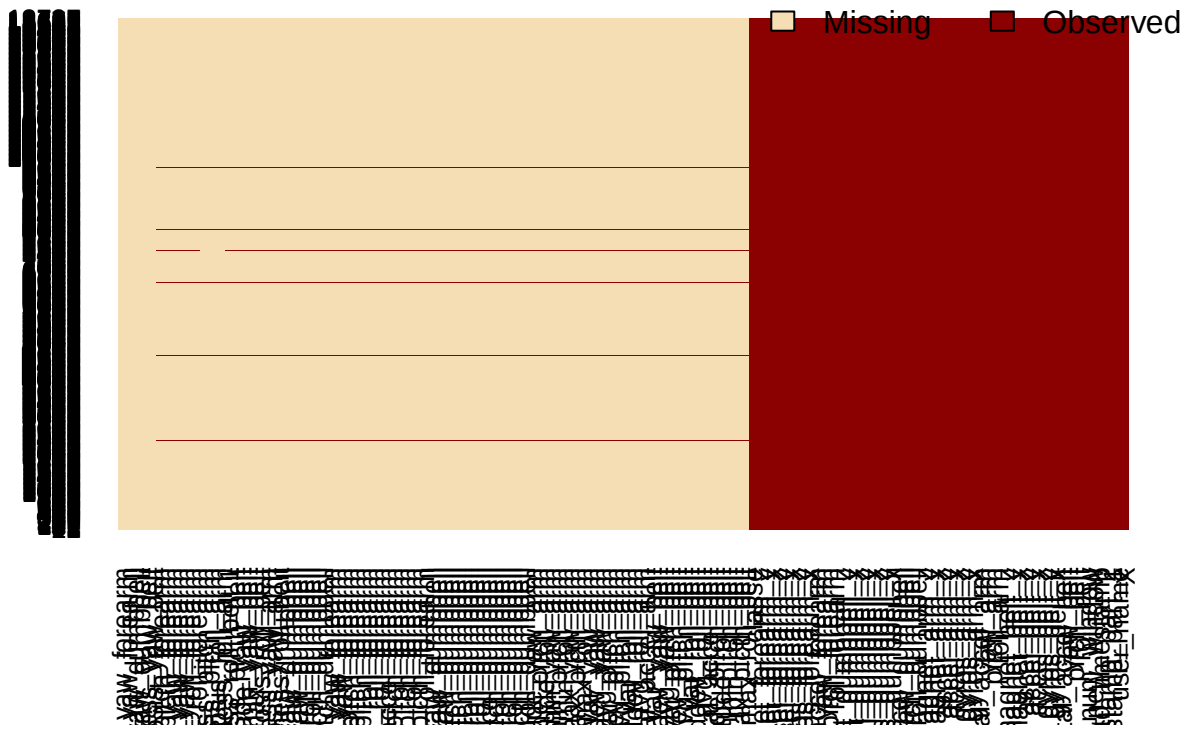
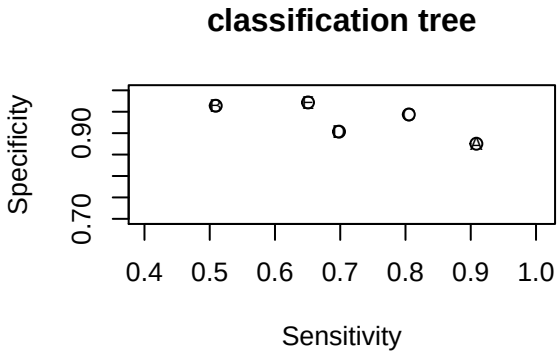
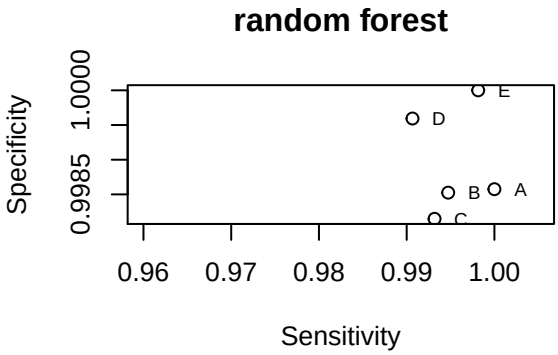

Evaluation of Missing Values
Missing
Observed

Diagram2


classification tree
Specificity
0.70
0.90
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Sensitivity


random forest
Specificity
0.9985
1.0000
E
D
B
A
C
0.96
0.97
0.98
0.99
1.00
Sensitivity

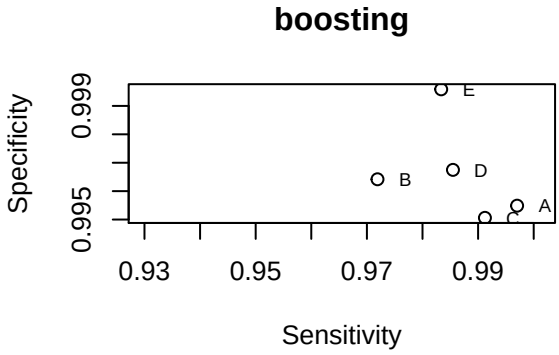
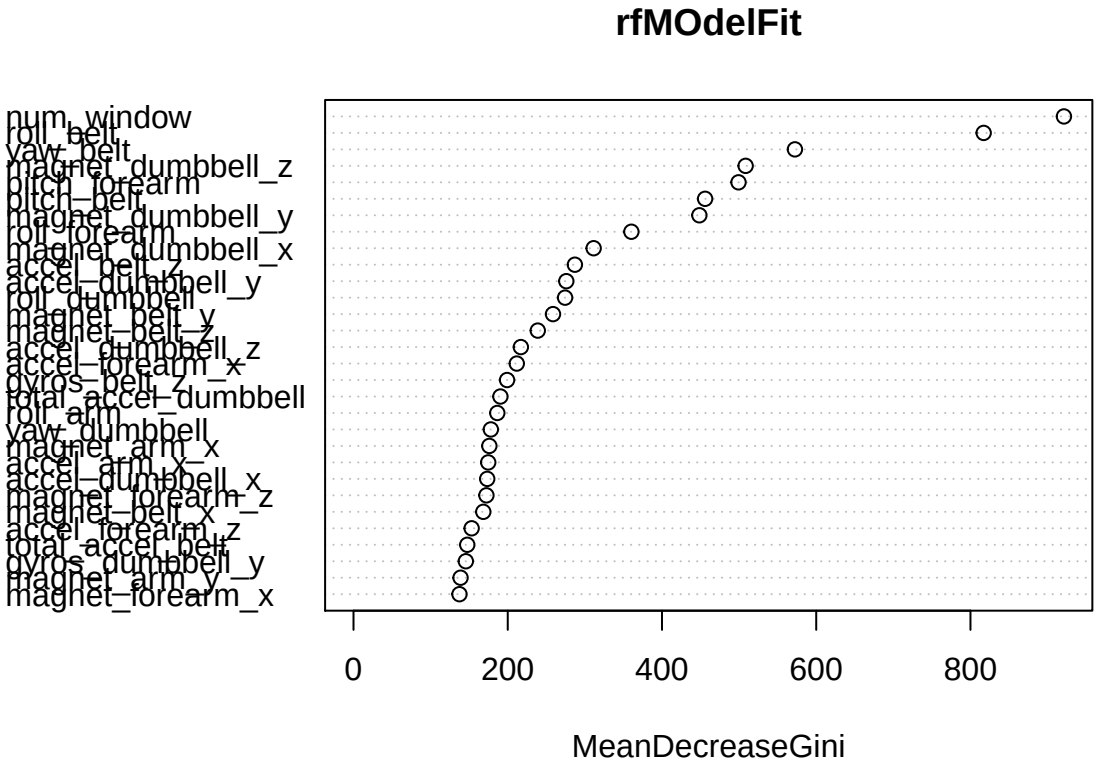

boosting
Specificity
0.995
0.999
E
B
D
A
C
0.93
0.95
0.97
0.99
Sensitivity

Diagram3


rfMOdelFit
num_window
roll_belt
yaw_belt
magnet_dumbbell_z
pitch_forearm
pitch_belt
magnet_dumbbell_y
roll_forearm
magnet_dumbbell_x
accel_belt_z
accel_dumbbell_y
roll_dumbbell
magnet_belt_y
magnet_belt_z
accel_dumbbell_z
accel_forearm_x
gyros_belt_z
total_accel_dumbbell
roll_arm
yaw_dumbbell
magnet_arm_x
accel_arm_x
accel_dumbbell_x
magnet_forearm_z
magnet_belt_x
accel_forearm_z
total_accel_belt
gyros_dumbbell_y
magnet_arm_y
magnet_forearm_x
0
200
400
600
800
MeanDecreaseGini