Practical Machine Learining Class Project

Nick Orka

3/3/2020

Overview

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. A goal of the project is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants and to predict the manner in which they did the exercise. This is the “classe” variable in the training set.

Data preprocessing

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).

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

Data summary

The initial data have NA data that required to be cleaned:

```
dim(training)
```

```
## [1] 19622   160
```

```
dim(testing)
```

```
## [1]  20 160
```

Cleaning data

Cleaned data:

```r
var_names <- names(training)
naNames <- names(training[, colSums(is.na(training)) > 0])
var_names <- var_names[!var_names %in% naNames]
var_names <- var_names[!grepl("X|timestamp|window", var_names)]
training <- training[, var_names]
classe <- training$classe
training <- training[, sapply(training, is.numeric)]
var_names <- names(training)
training$classe <- classe
testing <- testing[, var_names]
dim(training)
```

```
## [1] 19622    53
```

```r
dim(testing)
```

```
## [1] 20 52
```

Create validation dataset

```r
set.seed(34562)
isTrain <- createDataPartition(training$classe, p = .7, list = F)
validating <- training[-isTrain, ]
training <- training[isTrain,]
dim(validating)
```

```
## [1] 5885   53
```

```r
dim(training)
```

```
## [1] 13737    53
```

Prediction model

```r
control <- caret::trainControl(method = "cv", 5)
model <- caret::train(classe ~ ., data = training, method = "rf", trControl = control, ntree = 2
50, localImp = TRUE)
model
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10989, 10990, 10989, 10991, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9900269  0.9873832
##   27    0.9909734  0.9885804
##   52    0.9842034  0.9800127
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Then, we estimate the performance of the model on the validation data set.

```
p <- predict(model, validating)
confusionMatrix(validating$classe, p)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    1    0    0    0
##          B   11 1128    0    0    0
##          C    0    9 1015    2    0
##          D    0    0   15  948    1
##          E    0    1    3    6 1072
##
## Overall Statistics
##
##                Accuracy : 0.9917
##                  95% CI : (0.989, 0.9938)
##     No Information Rate : 0.2862
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9895
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9935   0.9903   0.9826   0.9916   0.9991
## Specificity            0.9998   0.9977   0.9977   0.9968   0.9979
## Pos Pred Value         0.9994   0.9903   0.9893   0.9834   0.9908
## Neg Pred Value         0.9974   0.9977   0.9963   0.9984   0.9998
## Prevalence             0.2862   0.1935   0.1755   0.1624   0.1823
## Detection Rate         0.2843   0.1917   0.1725   0.1611   0.1822
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9966   0.9940   0.9902   0.9942   0.9985
```

```
accuracy <- postResample(p, validating$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9916737 0.9894659
```

```
oose <- 1 - as.numeric(confusionMatrix(validating$classe, p)$overall[1])
oose
```

```
## [1] 0.008326253
```

So, the estimated accuracy of the model is 99.17% and the estimated out-of-sample error is 0.83%.

Predicting for Test Data Set

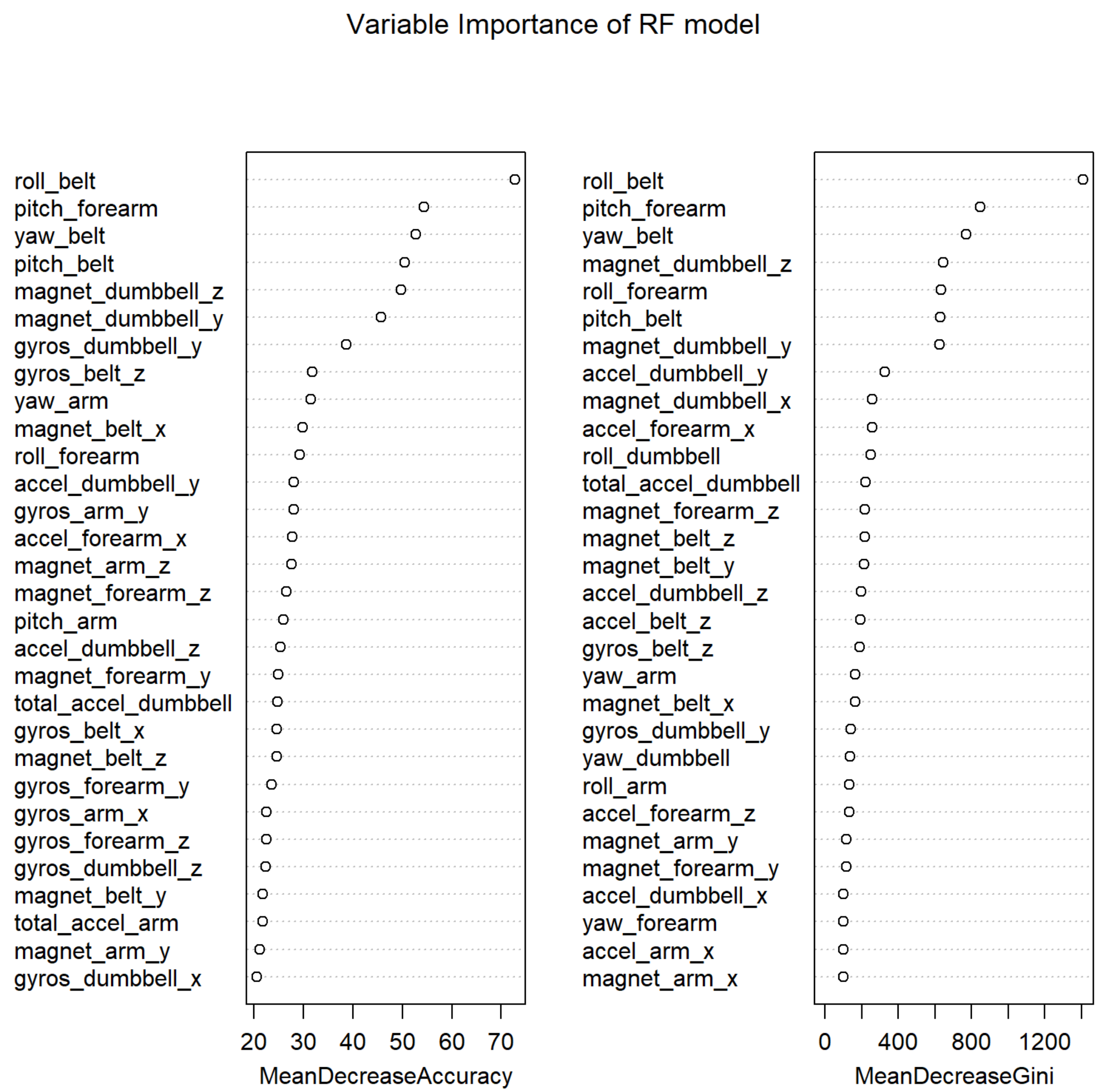
Now, we apply the model to the original testing data set downloaded from the data source.

```r
result <- predict(model, testing)
result
```

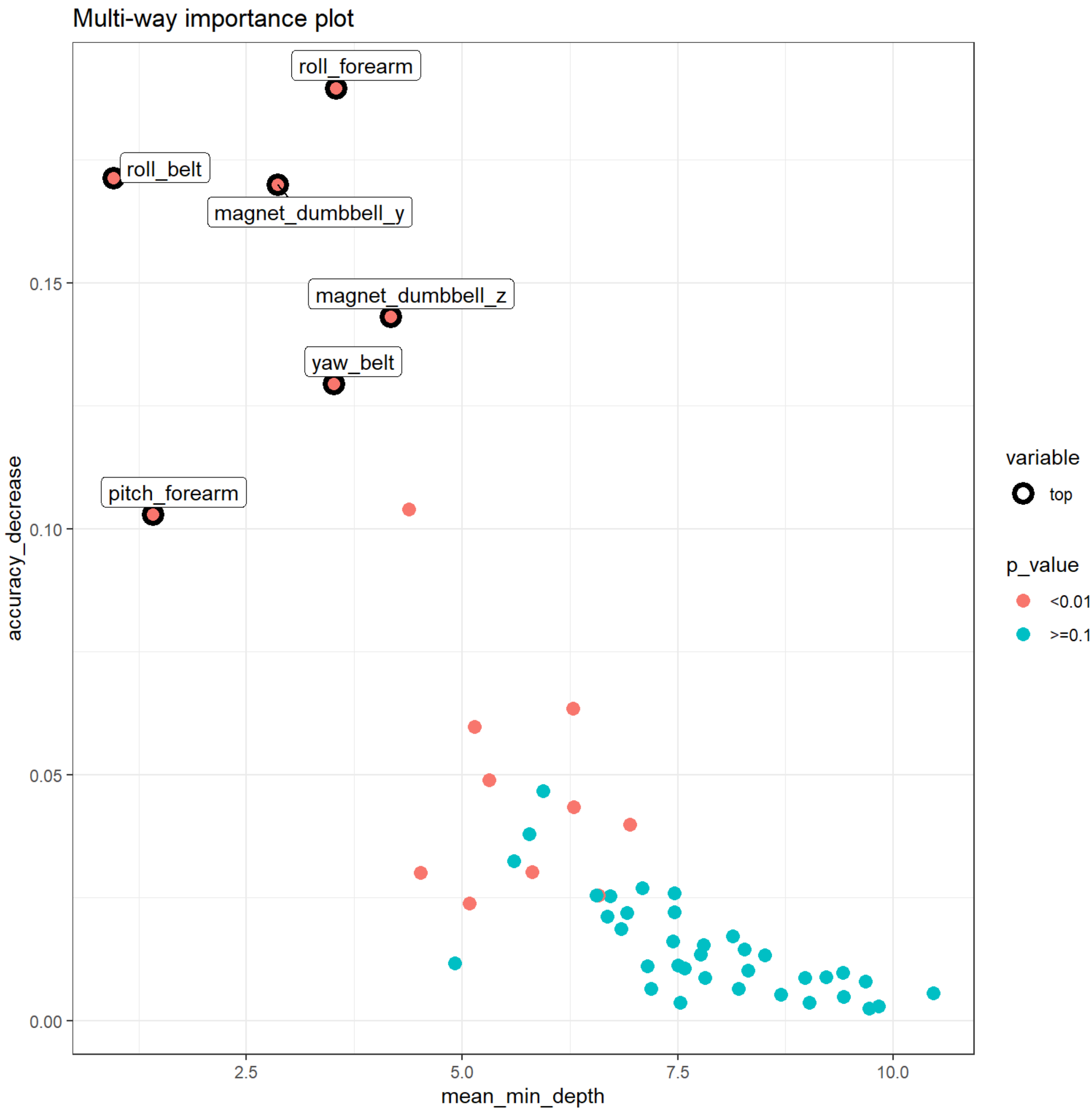
```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

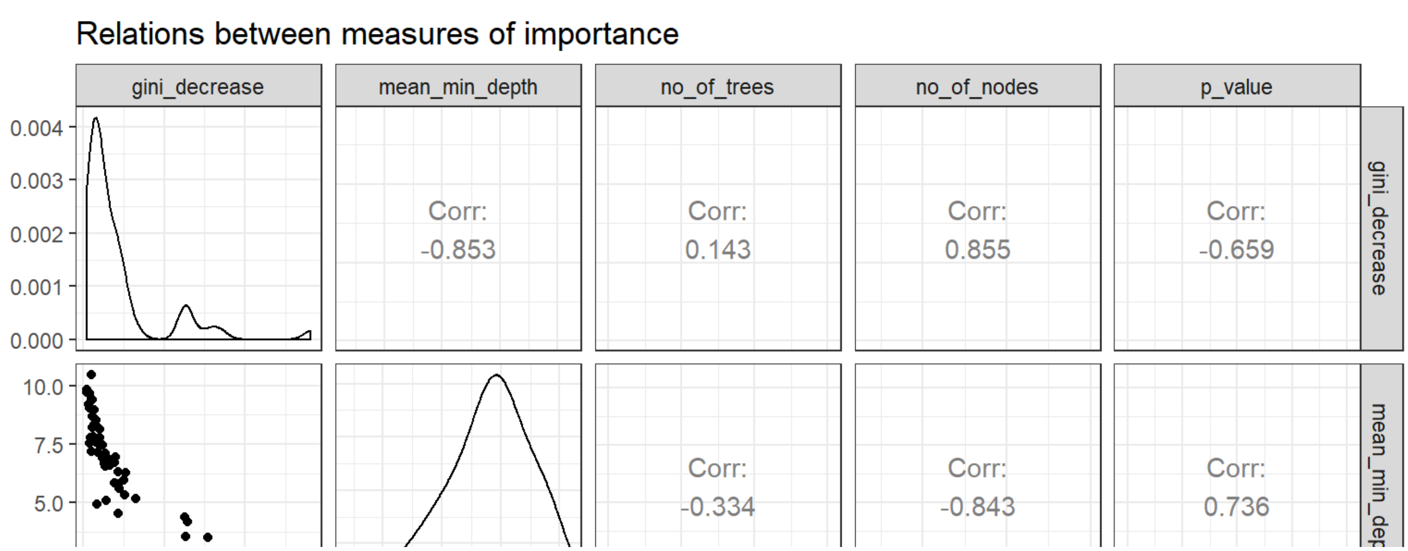
Appendix: Figures

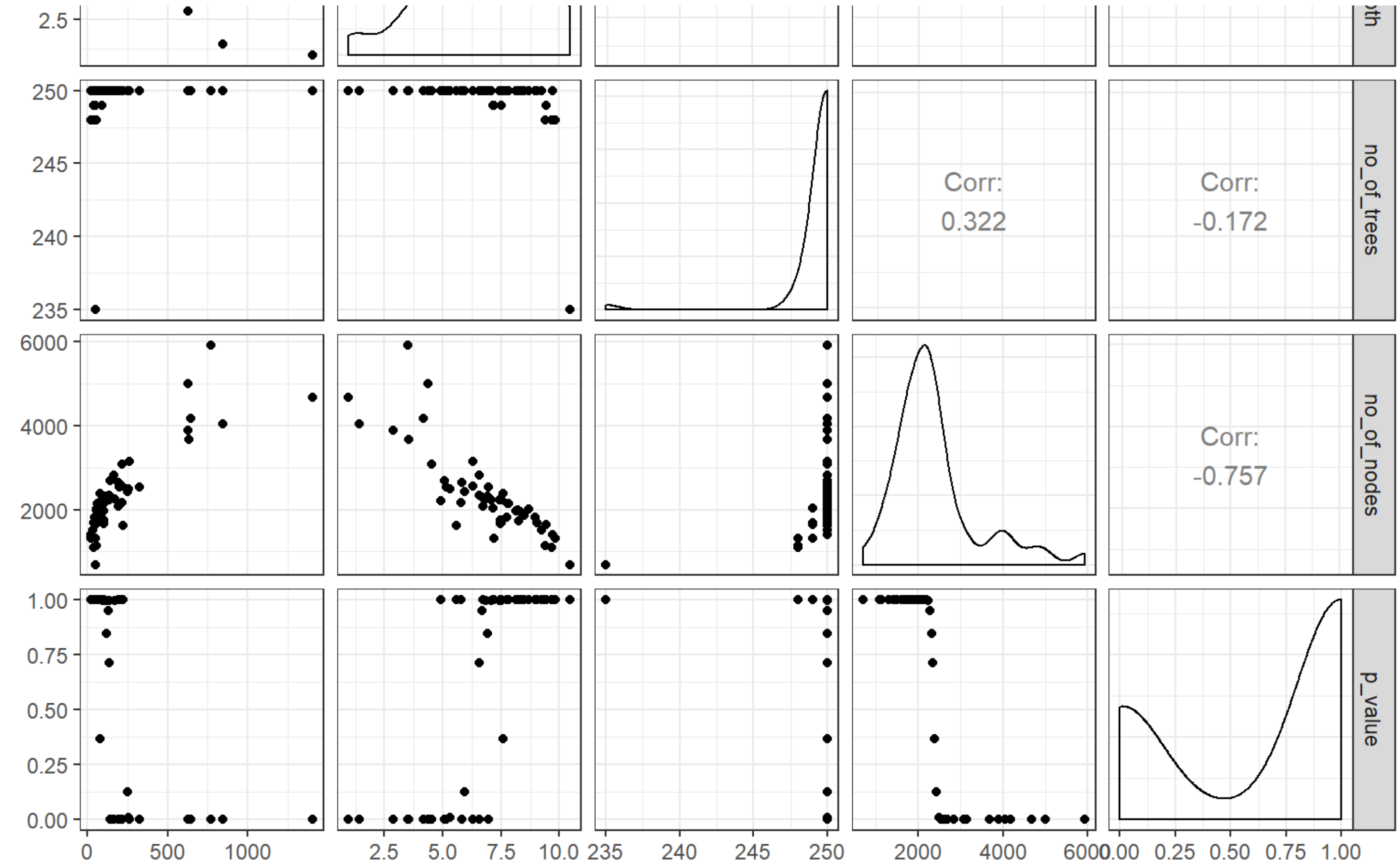
1. Variable importance



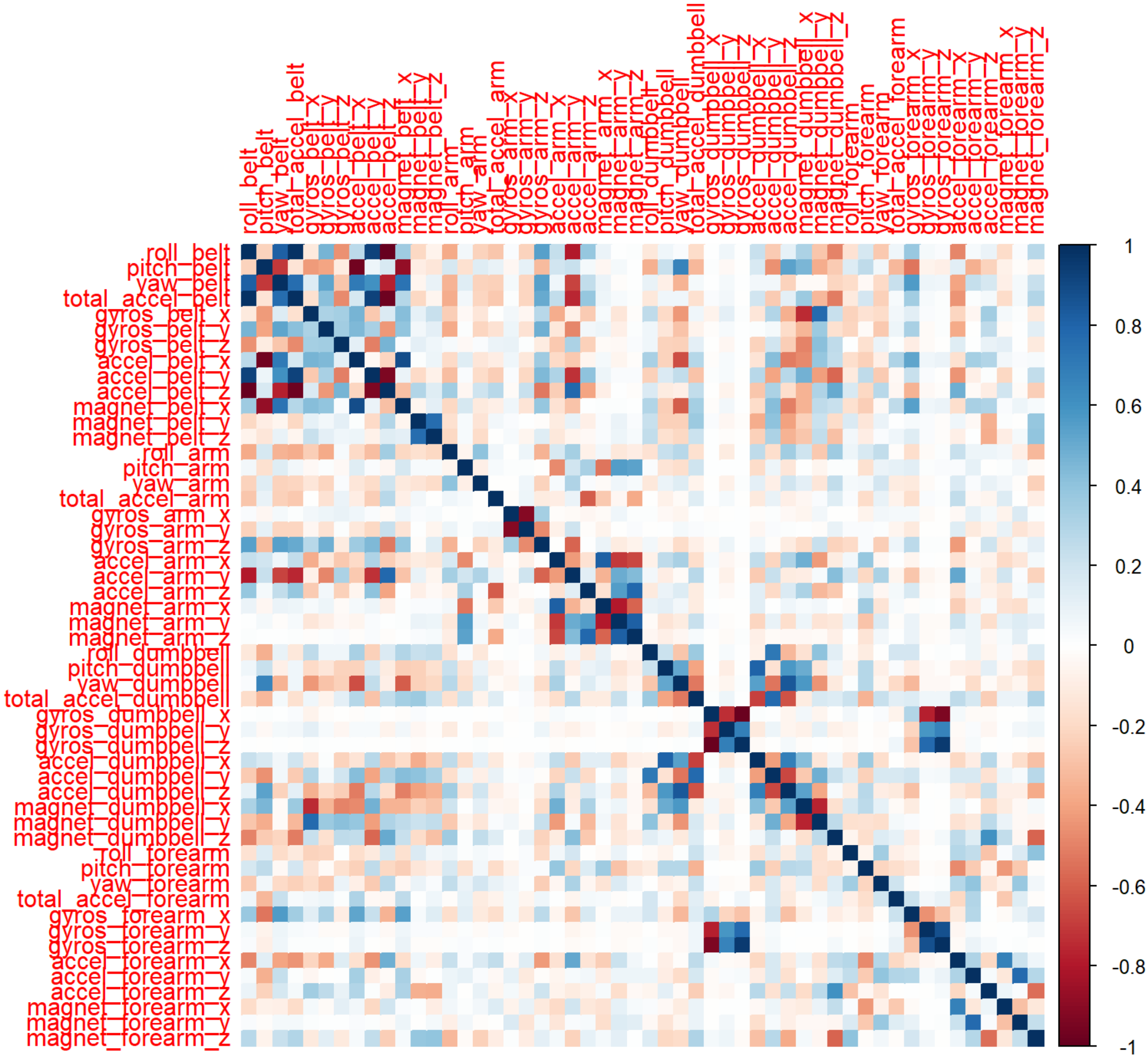
```
## Warning: Using alpha for a discrete variable is not advised.
```


Multi-way importance plot
roll_forearm
roll_belt
magnet_dumbbell_y
magnet_dumbbell_z
yaw_belt
pitch_forearm
accuracy_decrease
0.15
0.10
0.05
0.00
2.5
5.0
7.5
10.0
mean_min_depth
variable
top
p_value
<0.01
>=0.1

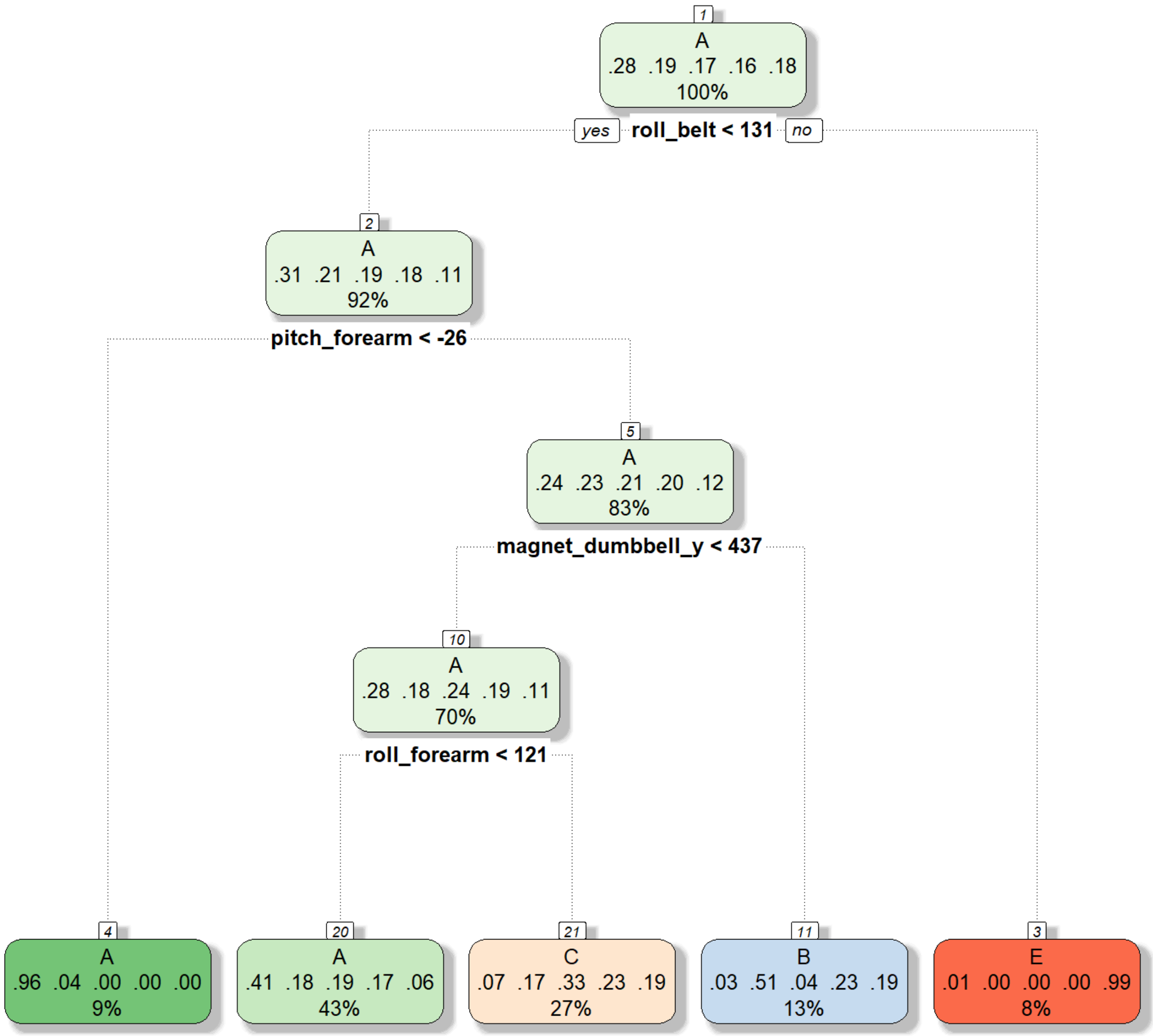

Relations between measures of importance
gini_decrease
mean_min_depth
no_of_trees
no_of_nodes
p_value
0.004
0.003
0.002
0.001
0.000
Corr:
-0.853
Corr:
0.143
Corr:
0.855
Corr:
-0.659
gini_decrease
10.0
7.5
5.0
Corr:
-0.334
Corr:
-0.843
Corr:
0.736



2. Correlation Matrix Visualization

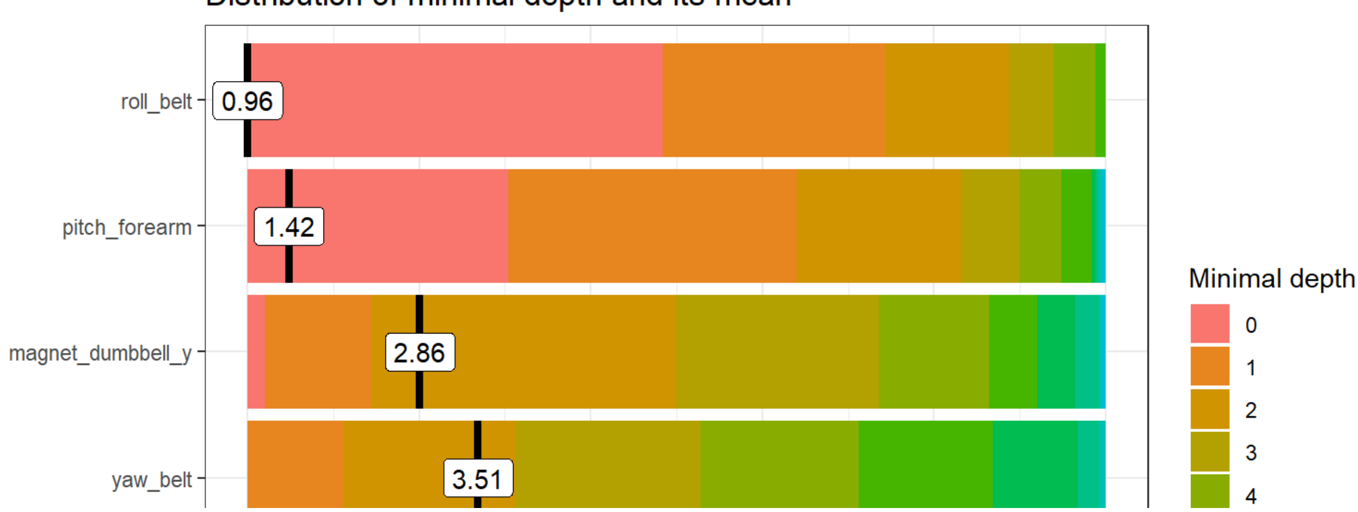


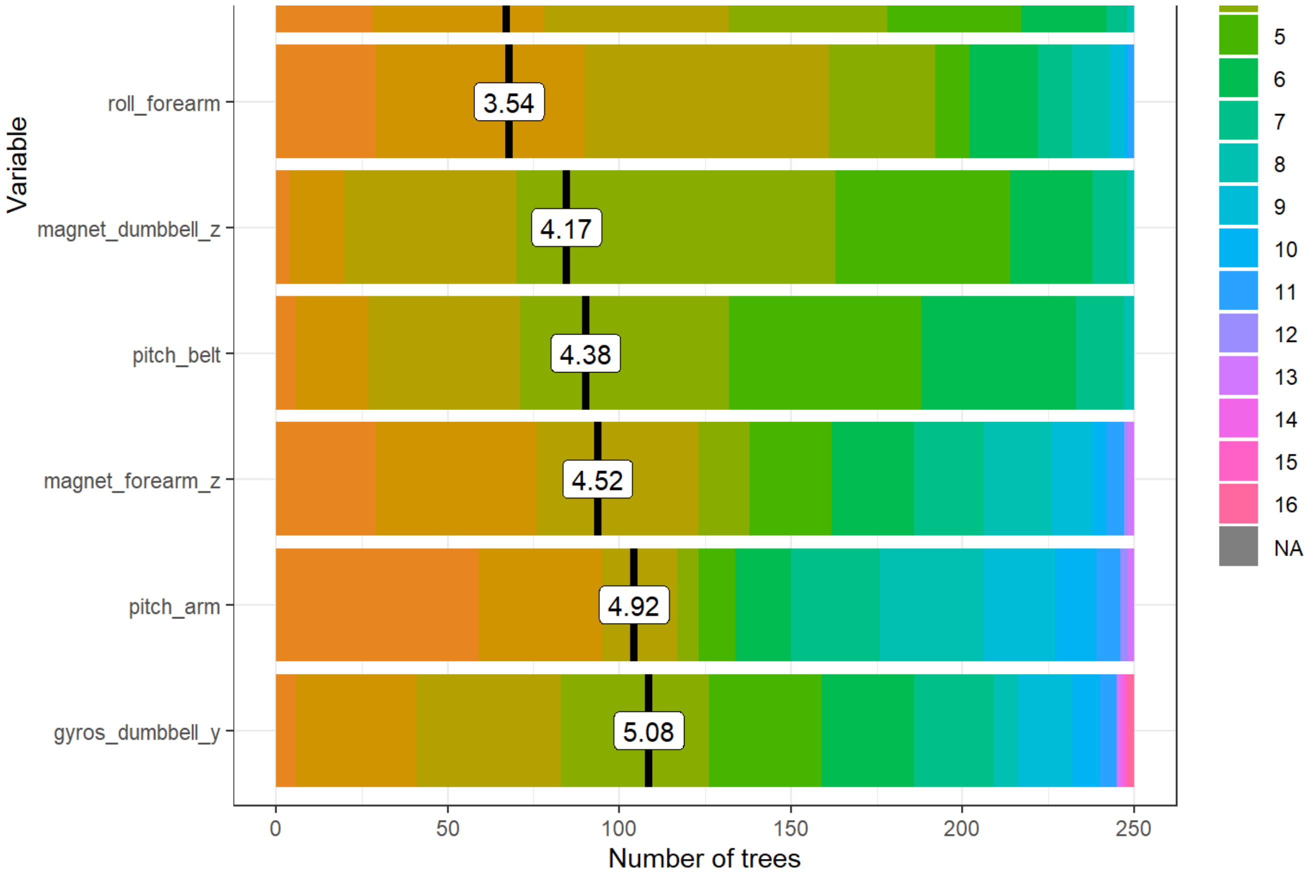
3. Decision Tree Visualization



Rattle 2020-Mar-04 17:00:48 Fox

Distribution of minimal depth and its mean





Variable
roll_forearm
magnet_dumbbell_z
pitch_belt
magnet_forearm_z
pitch_arm
gyros_dumbbell_y
3.54
4.17
4.38
4.52
4.92
5.08
0
50
100
150
200
250
Number of trees
5
6
7
8
9
10
11
12
13
14
15
16
NA