Project on Predicting Manner of Exercise

R Markdown

Synopsis

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, 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: [link] http://web.archive.org/web/20161224072740/ 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:

[link] https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

[link] https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: [link]http://web.archive.org/web/20161224072740/http: /groupware.les.inf.puc-rio.br/har.

Load the Required Packages

```
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
library(randomForest)
```

```
## randomForest 4.6-14

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

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

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

```
library(e1071)
library(rattle)
```

```
## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'

## The following object is masked from 'package:randomForest':
##
##     importance
```

Load the Data

```
pml_training = read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!",""))
pml_testing = read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
```

Cleaning the Data

Remove the columns that are mostly (over 20%) NA's

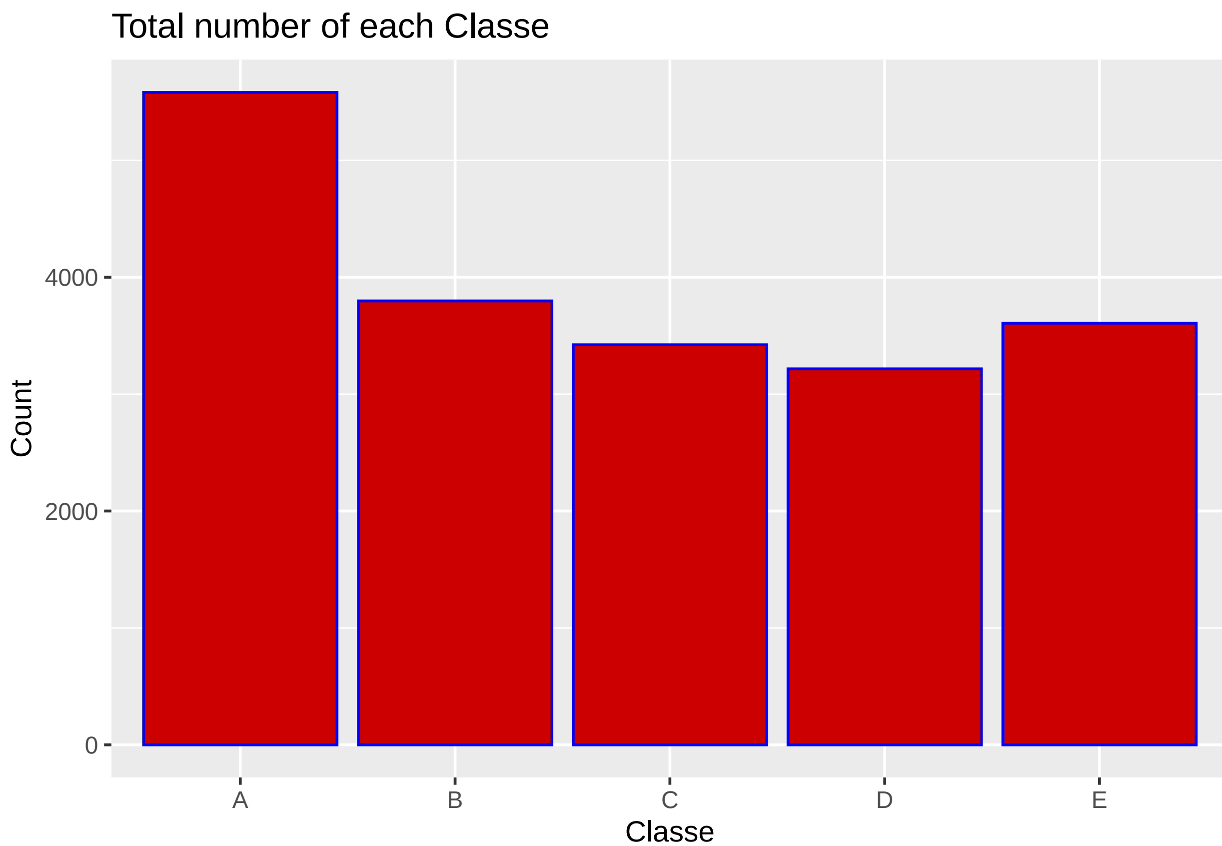
```
cleantrain <- pml_training[,colSums(is.na(pml_training)) <= .2*nrow(pml_training)]
cleantest <-pml_testing[,colSums(is.na(pml_testing)) <= .2*nrow(pml_testing)]
```

Remove the columns that do not pertain to our study

```
cleantrain <- cleantrain[,-(1:7)]
cleantest <- cleantest[,-(1:7)]
```

Graph the classe variable

```r
library(ggplot2)
g <- ggplot(cleantrain, aes(x = factor(classe))) + geom_bar(stat = "count", fill="red3", color = "blue"
g <- g + ggtitle("Total number of each Classe")
g <- g + xlab("Classe")
g <- g + ylab("Count")
g
```



Create Training and Testing sets for our models

Cross Validation

We will use 70% of cleantrain set data to built a model (training), and use the rest to test the model (training)

```r
set.seed(1234)
train <- createDataPartition(y=cleantrain$classe,p=.70,list=F)
training <- cleantrain[train,]
testing <- cleantrain[-train,]
head(cleantrain)
```

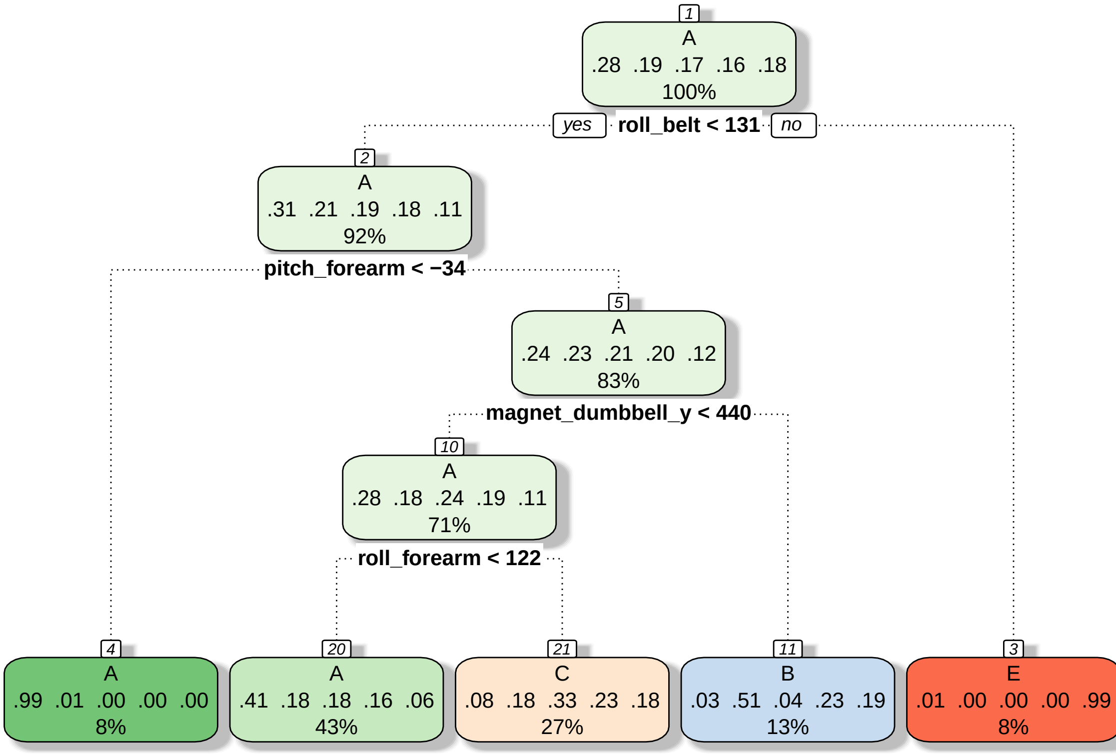
```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1      1.41       8.07    -94.4                3         0.00         0.00
## 2      1.41       8.07    -94.4                3         0.02         0.00
```

```
## 3       1.42       8.07    -94.4                3         0.00         0.00
## 4       1.48       8.05    -94.4                3         0.02         0.00
## 5       1.48       8.07    -94.4                3         0.02         0.02
## 6       1.45       8.06    -94.4                3         0.02         0.00
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.02          -21            4           22            -3
## 2        -0.02          -22            4           22            -7
## 3        -0.02          -20            5           23            -2
## 4        -0.03          -22            3           21            -6
## 5        -0.02          -21            2           24            -6
## 6        -0.02          -21            4           21             0
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           599          -313     -128      22.5    -161              34
## 2           608          -311     -128      22.5    -161              34
## 3           600          -305     -128      22.5    -161              34
## 4           604          -310     -128      22.1    -161              34
## 5           600          -302     -128      22.1    -161              34
## 6           603          -312     -128      22.0    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1        0.00        0.00       -0.02        -288         109        -123
## 2        0.02       -0.02       -0.02        -290         110        -125
## 3        0.02       -0.02       -0.02        -289         110        -126
## 4        0.02       -0.03        0.02        -289         111        -123
## 5        0.00       -0.03        0.00        -289         111        -123
## 6        0.02       -0.03        0.00        -289         111        -122
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -368          337          516      13.05217      -70.49400
## 2         -369          337          513      13.13074      -70.63751
## 3         -368          344          513      12.85075      -70.27812
## 4         -372          344          512      13.43120      -70.39379
## 5         -374          337          506      13.37872      -70.42856
## 6         -369          342          513      13.38246      -70.81759
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1    -84.87394                   37                0            -0.02
## 2    -84.71065                   37                0            -0.02
## 3    -85.14078                   37                0            -0.02
## 4    -84.87363                   37                0            -0.02
## 5    -84.85306                   37                0            -0.02
## 6    -84.46500                   37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1             0.00             -234               47             -271
## 2             0.00             -233               47             -269
## 3             0.00             -232               46             -270
## 4            -0.02             -232               48             -269
## 5             0.00             -233               48             -270
## 6             0.00             -234               48             -269
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1              -559               293               -65         28.4
## 2              -555               296               -64         28.3
## 3              -561               298               -63         28.3
## 4              -552               303               -60         28.1
## 5              -554               292               -68         28.0
## 6              -558               294               -66         27.9
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x gyros_forearm_y
```

```
## 1          -63.9       -153                    36             0.03              0.00
## 2          -63.9       -153                    36             0.02              0.00
## 3          -63.9       -152                    36             0.03             -0.02
## 4          -63.9       -152                    36             0.02             -0.02
## 5          -63.9       -152                    36             0.02              0.00
## 6          -63.9       -152                    36             0.02             -0.02
##   gyros_forearm_z accel_forearm_x accel_forearm_y accel_forearm_z
## 1           -0.02             192             203            -215
## 2           -0.02             192             203            -216
## 3            0.00             196             204            -213
## 4            0.00             189             206            -214
## 5           -0.02             189             206            -214
## 6           -0.03             193             203            -215
##   magnet_forearm_x magnet_forearm_y magnet_forearm_z classe
## 1              -17              654              476      A
## 2              -18              661              473      A
## 3              -18              658              469      A
## 4              -16              658              469      A
## 5              -17              655              473      A
## 6               -9              660              478      A
```

Create a Decision Tree for Prediction and Classification

```r
modFit <- train(classe ~ .,method="rpart",data=training)
fancyRpartPlot(modFit$finalModel)
```



Rattle 2020−Aug−07 08:42:16 Ken

Now, we will examine 3 methods for doing our prediction. Random Forests (rf), Support vector machine(svm) and Linear discriminant analysis (lda).

- We will:
  - Fit the Model
  - Use the Model to Predict on the Test set
  - Creat the Confusion Matrix
  - Find the Accuracy of the Model from the Confusion Matrix

```
fitrf <- train(classe ~ ., data=training, method="rf", trControl=trainControl(method="none"), tuneGrid=

fitsvm <- svm(as.factor(classe) ~. , data=training)
fitlda <- train(as.factor(classe) ~ .,method="lda",data= training)
predrf <- predict(fitrf, testing)
predsvm <- predict(fitsvm, testing)
predlda <- predict(fitlda, testing)
confMrf <- confusionMatrix(predrf, as.factor(testing$classe))$overall[1]
confsvm <- confusionMatrix(predsvm, as.factor(testing$classe))$overall[1]
conflda <- confusionMatrix(predlda, as.factor(testing$classe))$overall[1]
confMrf
```

```
##  Accuracy
```

```
## 0.9957519
```

```
confsvm
```

```
##  Accuracy
## 0.9420561
```

```
conflda
```

```
##  Accuracy
## 0.6960068
```

We see Random Forest has the best accuracy but, we will use each model to make predictions on the cleantrain dataset.

Using Random Forest

```
Predictionrf <- predict(fitrf, newdata = cleantest)
Predictionrf
```

```
##  [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
```

Using SVM

```
Predictionsvm <- predict(fitsvm, newdata = cleantest)
Predictionsvm
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  A  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

Using lda

```
Predictionlda <- predict(fitlda, newdata = cleantest)
Predictionlda
```

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

We use the Random Forest outcome for our predictions!