Predicting Manner of Barbell Lifts from Accelerometers

chandu dugyala

Project Information

Personnel tracking 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://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)

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)

Settingup environment for data analysis

Below R packages are used for this project,these packages must load before continuning on project

```
library(lattice)
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
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:rattle':
##
##     importance
```

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

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
set.seed(36154)
```

Loading data for analysis

```
UrlTrain <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
UrlTest  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"

training <- read.csv(url(UrlTrain))
testing  <- read.csv(url(UrlTest))

inTrain  <- createDataPartition(training$classe, p=0.6, list=FALSE)
TrainSet <- training[inTrain, ]
TestSet  <- training[-inTrain, ]
dim(TrainSet)
```

```
## [1] 11776   160
```

```
dim(TestSet)
```

```
## [1] 7846  160
```

Cleaning Data

Reducing the NZV, NA variables from data

```
NZV <- nearZeroVar(TrainSet)
TrainSet <- TrainSet[, -NZV]
TestSet  <- TestSet[, -NZV]
allNA    <- sapply(TrainSet, function(x) mean(is.na(x))) > 0.95
TrainSet <- TrainSet[, allNA==FALSE]
TestSet  <- TestSet[, allNA==FALSE]
```

removing variables which doesn’t make sense for prediction

```
TrainSet <- TrainSet[, -(1:5)]
TestSet  <- TestSet[, -(1:5)]
dim(TrainSet)
```

```
## [1] 11776    54
```

```
dim(TestSet)
```

```
## [1] 7846   54
```

Machine learning approach

The goal of this project is to be able to predict the manner in which a barbell lift was done. I will use a random forest and Decision Tree methods make this prediction, and use best results to answer the quiz.

Random Forests Methos

```
modelB1 <- randomForest(classe ~. , data=TrainSet)
```

prediction on Test dataset

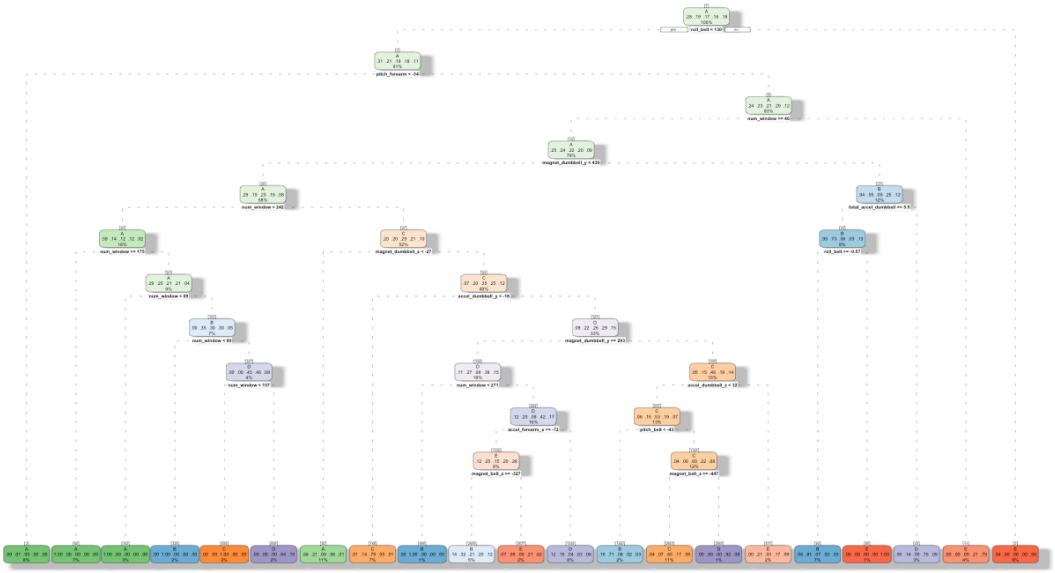
```
predictionsB1 <- predict(modelB1, TestSet, type = "class")
```

```
confusionMatrix(predictionsB1, TestSet$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    2    0    0    0
##          B    0 1516   11    0    0
##          C    0    0 1355   10    0
##          D    0    0    2 1276    3
##          E    0    0    0    0 1439
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.9948, 0.9976)
##     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   0.9987   0.9905   0.9922   0.9979
## Specificity            0.9996   0.9983   0.9985   0.9992   1.0000
## Pos Pred Value         0.9991   0.9928   0.9927   0.9961   1.0000
## Neg Pred Value         1.0000   0.9997   0.9980   0.9985   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1932   0.1727   0.1626   0.1834
## Detection Prevalence   0.2847   0.1946   0.1740   0.1633   0.1834
## Balanced Accuracy      0.9998   0.9985   0.9945   0.9957   0.9990
```

Decision Tree Methos

```
modelDecTree <- rpart(classe ~ ., data=TrainSet, method="class")
fancyRpartPlot(modelDecTree)
```



Rattle 2018-Dec-31 08:29:51 chandu

Prediction on Test dataset

```
predictDecTree <- predict(modelDecTree, newdata=TestSet, type="class
")
confMatDecTree <- confusionMatrix(predictDecTree, TestSet$classe)
confMatDecTree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1925  203   56   64   13
##          B  139 1000  185  103   82
##          C   45  121 1089  178   47
##          D   76  150   31  822   83
##          E   47   44    7  119 1217
##
## Overall Statistics
##
##                Accuracy : 0.7715
##                  95% CI : (0.762, 0.7807)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7107
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8625   0.6588   0.7961   0.6392   0.8440
## Specificity            0.9401   0.9196   0.9396   0.9482   0.9661
## Pos Pred Value         0.8514   0.6627   0.7358   0.7074   0.8487
## Neg Pred Value         0.9450   0.9183   0.9562   0.9306   0.9649
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2453   0.1275   0.1388   0.1048   0.1551
## Detection Prevalence   0.2882   0.1923   0.1886   0.1481   0.1828
## Balanced Accuracy      0.9013   0.7892   0.8678   0.7937   0.9050
```

Generating files to submit assignmnet

```
predictTEST <- predict(modelB1, newdata=testing)
predictTEST
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```