Prediction Assignment Writeup

Harsha vardhan reddy

12/23/2020

Prediction Assignment Writeup

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.(predict the manner in which they did the exercise)

Background

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, 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 (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv The test data are available here: 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. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Goal of the project

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with.

You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

loading packages

```
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.

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)
library(rpart.plot)
library(corrplot)

## corrplot 0.84 loaded

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

## The following object is masked from 'package:rattle':
##
##     importance

library(RColorBrewer)
set.seed(333)
```

Read Data

```
trainRaw <- read.csv("pml-training.csv", na.strings = c("NA", ""))
testRaw <- read.csv("pml-testing.csv", na.strings = c("NA", "" ))
dim(trainRaw)

## [1] 19622   160
```

```
dim(testRaw)

## [1]  20 160

rm(trainFile)

## Warning in rm(trainFile): object 'trainFile' not found

rm(testFile)

## Warning in rm(testFile): object 'testFile' not found
```

cleaning data

removing meaningless variables and outliers, like near zero variance variables removing columns of dataset that do not contribute much to accelerometer measurements.

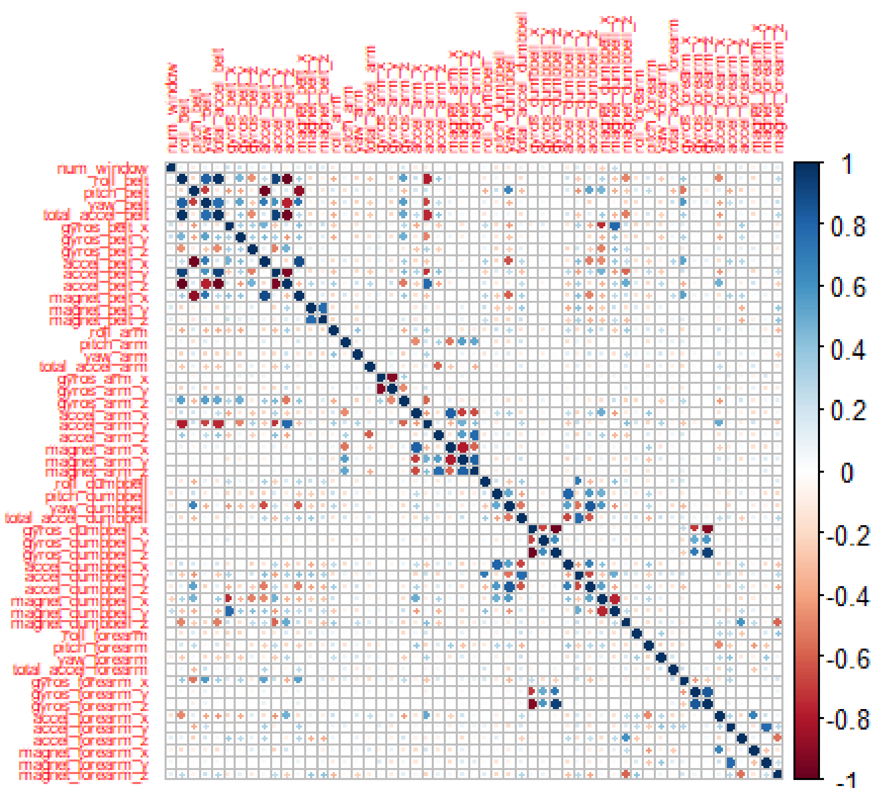
```
NZV <- nearZeroVar(trainRaw, saveMetrics = TRUE)
head(NZV, 20)

##                      freqRatio percentUnique zeroVar   nzv
## X                     1.000000  100.00000000   FALSE FALSE
## user_name             1.100679    0.03057792   FALSE FALSE
## raw_timestamp_part_1  1.000000    4.26562022   FALSE FALSE
## raw_timestamp_part_2  1.000000   85.53154622   FALSE FALSE
## cvtd_timestamp        1.000668    0.10192641   FALSE FALSE
## new_window           47.330049    0.01019264   FALSE  TRUE
## num_window            1.000000    4.37264295   FALSE FALSE
## roll_belt             1.101904    6.77810621   FALSE FALSE
## pitch_belt            1.036082    9.37722964   FALSE FALSE
## yaw_belt              1.058480    9.97349913   FALSE FALSE
## total_accel_belt      1.063160    0.14779329   FALSE FALSE
## kurtosis_roll_belt    5.000000    2.01814290   FALSE FALSE
## kurtosis_picth_belt   8.000000    1.61043726   FALSE FALSE
## kurtosis_yaw_belt     0.000000    0.00509632    TRUE  TRUE
## skewness_roll_belt    2.250000    2.00795026   FALSE FALSE
## skewness_roll_belt.1  8.000000    1.71745999   FALSE FALSE
## skewness_yaw_belt     0.000000    0.00509632    TRUE  TRUE
## max_roll_belt         1.000000    0.99378249   FALSE FALSE
## max_picth_belt        1.538462    0.11211905   FALSE FALSE
## max_yaw_belt          1.034483    0.34145347   FALSE FALSE

training01 <- trainRaw[, !NZV$nzv]
testing01 <- testRaw[, !NZV$nzv]
dim(training01)

## [1] 19622   117

dim(testing01)

## [1]  20 117
```

```
rm(trainRaw)
rm(testRaw)
rm(NZV)
regex <- grepl("^X|timestamp|user_name", names(training01))
training <- training01[, !regex]
testing <- testing01[, !regex]
rm(regex)
rm(training01)
rm(testing01)
dim(training)

## [1] 19622   112

cond <- (colSums(is.na(training)) == 0)
training <- training[, cond]
testing <- testing[, cond]
rm(cond)
```

After cleaning training data set contain 19622 observations and 54 variables and testing data set contain 20 observations and 54 variables.

correlation Matrix in training Data set.

```
corrplot(cor(training[, -length(names(training))]), method = "circle", tl.cex
= 0.5)
```



splitting training set

```
set.seed(333)
inTrain <- createDataPartition(training$classe, p = 0.80, list = FALSE)
validation <- training[-inTrain, ]
training <- training[inTrain, ]
rm(inTrain)
```

training data set 80%, validation data set 20%

Data Modelling

Random Forest

training

```
modelRF <- train(classe ~ ., data = training, method = "rf", trControl =
trainControl(method = "cv", 5), ntree = 250)
modelRF
```

```
## Random Forest
##
## 15699 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 12560, 12559, 12559, 12558, 12560
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9943306  0.9928279
##   27    0.9970697  0.9962935
##   53    0.9941394  0.9925861
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

validation data set(20%)

```
predictRF <- predict(modelRF, validation)
confusionMatrix(as.factor(validation$classe), predictRF)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    0    0    0    0
##          B    0  759    0    0    0
```

```
##          C    0    1  683    0    0
##          D    0    0    2  641    0
##          E    0    0    0    0  721
##
## Overall Statistics
##
##                Accuracy : 0.9992
##                  95% CI : (0.9978, 0.9998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.999
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9987   0.9971   1.0000   1.0000
## Specificity            1.0000   1.0000   0.9997   0.9994   1.0000
## Pos Pred Value         1.0000   1.0000   0.9985   0.9969   1.0000
## Neg Pred Value         1.0000   0.9997   0.9994   1.0000   1.0000
## Prevalence             0.2845   0.1937   0.1746   0.1634   0.1838
## Detection Rate         0.2845   0.1935   0.1741   0.1634   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   0.9993   0.9984   0.9997   1.0000
```

Accuracy of the Random Forest Model is 99.8810535% and the Estimated Out-of-Sample Error is 0.1189465%

Decision Tree

```r
modelTree <- rpart(classe ~ ., data = training, method = "class")
predictTree <- predict(modelTree, validation, type = "class")
confusionMatrix(as.factor(validation$classe), predictTree)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 970  36  17  62  31
##          B 121 446  49 106  37
##          C  24  26 544  48  42
##          D  45  38 105 410  45
##          E  38  64  67  71 481
##
## Overall Statistics
##
##                Accuracy : 0.7267
##                  95% CI : (0.7125, 0.7406)
##     No Information Rate : 0.3054
```

```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6538
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8097   0.7311   0.6957   0.5882   0.7563
## Specificity            0.9464   0.9055   0.9554   0.9278   0.9270
## Pos Pred Value         0.8692   0.5876   0.7953   0.6376   0.6671
## Neg Pred Value         0.9188   0.9482   0.9265   0.9125   0.9516
## Prevalence             0.3054   0.1555   0.1993   0.1777   0.1621
## Detection Rate         0.2473   0.1137   0.1387   0.1045   0.1226
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.8781   0.8183   0.8255   0.7580   0.8416
```

Accuracy of the Random Forest Model is 74.4774851% and the Estimated Out-of-Sample Error is 25.5225149%.

so random forest is the better model.

Important variables

```
vi <- varImp(modelRF)$importance
vi[head(order(unlist(vi), decreasing = TRUE), 5L), , drop = FALSE]
```

```
##                    Overall
## num_window       100.00000
## roll_belt         60.95798
## pitch_forearm     38.93027
## yaw_belt          31.56578
## magnet_dumbbell_z 29.22849
```

random forest on test data set

```
rm(accuracy)
```

```
## Warning in rm(accuracy): object 'accuracy' not found
```

```
rm(ose)
```

```
## Warning in rm(ose): object 'ose' not found
```

```
predict(modelRF, testing[, -length(names(testing))])
```

```
##  [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
```