prediction writeup assignment finale

Gebremeskel Mamu Werid

February 22, 2019

```
pandoc <- file.path(Sys.getenv("RSTUDIO_PANDOC"), "pandoc")
command <- paste(shQuote(c(pandoc, args)), collapse = " ")
```

1. Introduction

This is a machine learning assignment on prediction. 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. The assignment will be done based on the following instructions. ### goal of the assignment

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.

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 (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 (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). 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.

Instructions for submitting the assignment

Peer Review Portion

Your submission for the Peer Review portion should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).

Course Project Prediction Quiz Portion

Apply your machine learning algorithm to the 20 test cases available in the test data above and submit your predictions in appropriate format to the Course Project Prediction Quiz for automated grading.

Reproducibility

Due to security concerns with the exchange of R code, your code will not be run during the evaluation by your classmates. Please be sure that if they download the repo, they will be able to view the compiled HTML version of your analysis.

2. getting and cleaning data

Before any thing else, we need to get (import or download) the data and make sure that it is clean and ready for analysis.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.2
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.5.2
```

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

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 3.5.2
```

```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.5.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.2
```

```
## 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(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.5.2
```

```
## corrplot 0.84 loaded
```

Get the data by downloading

```
trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trainFile <- "./data/pml-training.csv"
testFile  <- "./data/pml-testing.csv"
if (!file.exists("./data")) {
  dir.create("./data")
}
if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="curl")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="curl")
}
```

See how the data looklike and clean it

After downloading the data from the data source, read the two csv files into two data frames and remove all observations with N.A values and some meaningless variables

```
trainRaw <- read.csv ("./data/pml-training.csv", na.strings=c("NA","#DIV/0!",""))
testRaw <- read.csv ("./data/pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
dim(trainRaw)
```

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

```
dim(testRaw)
```

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

The training data set contains 19622 observations and 160 variables, while the testing data set contains 20 observations and 160 variables. The “classe” variable in the training set is the outcome to predict.

Remove columns that do not contribute much to the accelerometer measurements.

```
classe <- trainRaw$classe
trainRemove <- grepl("^X|timestamp|window", names(trainRaw))
trainRaw <- trainRaw[, !trainRemove]
trainCleaned <- trainRaw[, sapply(trainRaw, is.numeric)]
trainCleaned$classe <- classe
testRemove <- grepl("^X|timestamp|window", names(testRaw))
testRaw <- testRaw[, !testRemove]
testCleaned <- testRaw[, sapply(testRaw, is.numeric)]
```

Slice the data

Split the cleaned training set into a training data set (60%) and a validation data set (40%). We will use the validation data set to conduct cross validation in future steps.

```
set.seed(22519) # For reproducibile purpose
inTrain <- createDataPartition(trainCleaned$classe, p=0.60, list=F)
trainData <- trainCleaned[inTrain, ]
testData <- trainCleaned[-inTrain, ]
```

final cleaning and structuring

Then, we need to clean and structure the data using only valid indicators/features. A certain selection of feature will be removed. They are in the following criterias: 1.Variables with variances that are close to zero(considered constant) 2.Variables with missing readings over 75% 3.Variables that are not fit to be predictors

```
mytrain_SUB <- trainData #creating another subset to iterate in loop
for (i in 1:length(trainData)) { #for every column in the training dataset
if (sum(is.na(trainData[ , i])) / nrow(trainData) >= .75) { #if n?? NAs > 60% of total observati
ons
for (j in 1:length(mytrain_SUB)) {
if (length(grep(names(trainData[i]), names(mytrain_SUB)[j]))==1) { #if the columns are the same:
mytrain_SUB <- mytrain_SUB[ , -j] #Remove that column
    }
  }
 }
}
#To check the new N?? of observations
dim(mytrain_SUB)
```

```
## [1] 11776    53
```

```
mytrain_SUB2 <- mytrain_SUB[,8:length(mytrain_SUB)]
NZV <- nearZeroVar(mytrain_SUB2, saveMetrics = TRUE)
keep <- names(mytrain_SUB2)
```

Based on the graph above, we can see that each level frequency is within the same order of magnitude of each other. Level A is the most frequent while level D is the least frequent.

3. building a model

Using **Random Forest** algorithm, predictive model was fitted. when applying the algorithm, **5-fold cross validation** was used.

```
controlRf <- trainControl(method="cv", 5)
modelRf <- train(classe ~ ., data = mytrain_SUB2, method="rf", trControl=controlRf, ntree=250)
modelRf
```

```
## Random Forest
##
## 11776 samples
##    45 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 9421, 9421, 9421, 9421, 9420
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9791955  0.9736727
##   23    0.9813183  0.9763618
##   45    0.9722323  0.9648673
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 23.
```

Estimate the performance of the model on the validation data set.

```
predictRf <- predict(modelRf, testData)
confusionMatrix(testData$classe, predictRf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    2    0    0    1
##          B   15 1494    6    2    1
##          C    0   20 1348    0    0
##          D    0    6   43 1230    7
##          E    2    2    4    6 1428
##
## Overall Statistics
##
##                Accuracy : 0.9851
##                  95% CI : (0.9822, 0.9877)
##     No Information Rate : 0.2863
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9811
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9924   0.9803   0.9622   0.9935   0.9937
## Specificity            0.9995   0.9962   0.9969   0.9915   0.9978
## Pos Pred Value         0.9987   0.9842   0.9854   0.9565   0.9903
## Neg Pred Value         0.9970   0.9953   0.9918   0.9988   0.9986
## Prevalence             0.2863   0.1942   0.1786   0.1578   0.1832
## Detection Rate         0.2841   0.1904   0.1718   0.1568   0.1820
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9959   0.9883   0.9795   0.9925   0.9958
```

```
accuracy <- postResample(predictRf, testData$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9850879 0.9811318
```

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

```
## [1] 0.01491206
```

summary of acuracy and error

The estimated accuracy of the model is 98.5% and the estimated out-of-sample error is 1.5%.

Predicting for Test Data Set

We apply the model to the original testing data set downloaded from the data source. We remove the `problem_id` column first.

```
result <- predict(modelRf, testCleaned[, -length(names(testCleaned))])
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
```

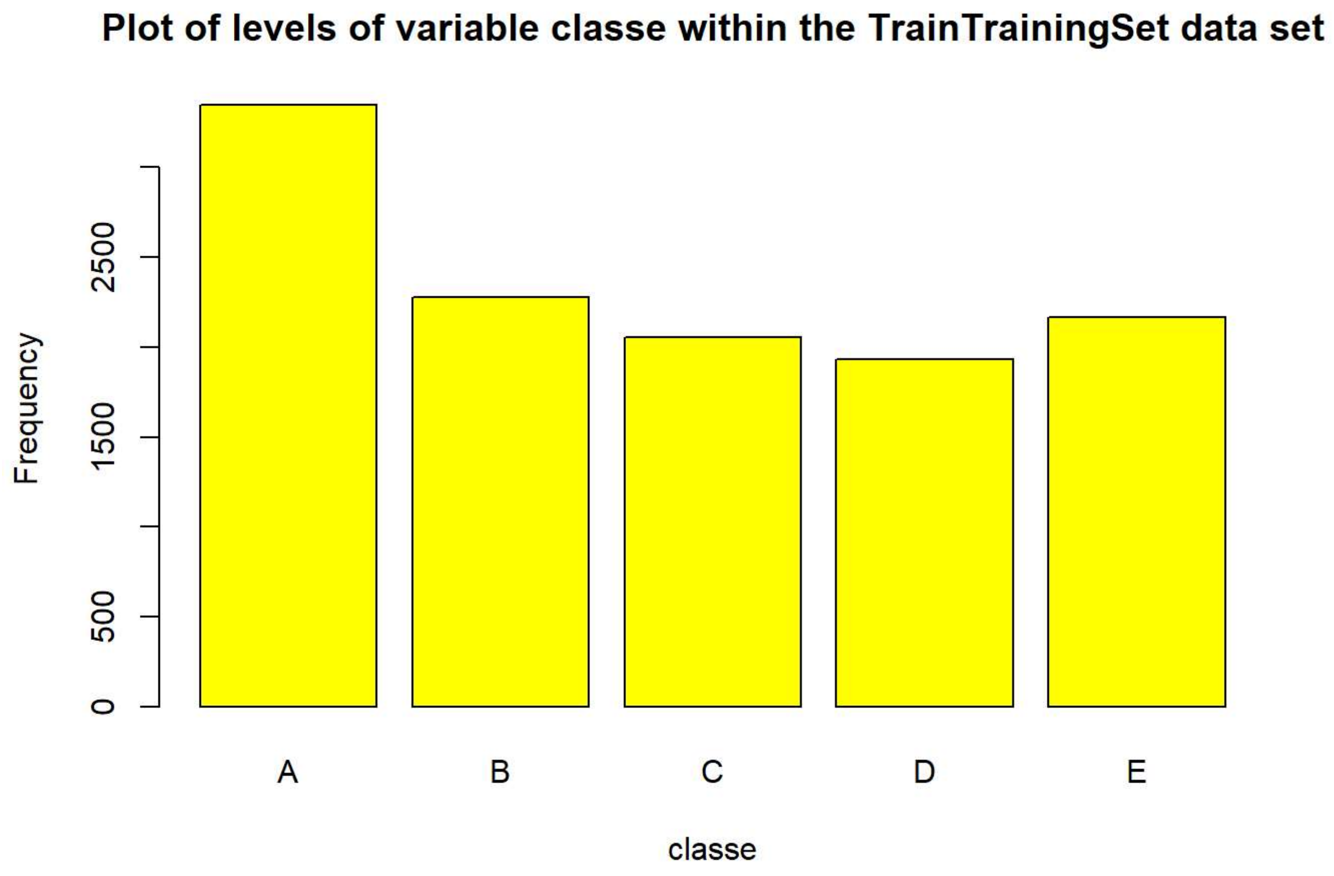
summary

The model statistics showed that the built model had the overall accuracy (with 95% CI) of 98.5% for the testing set, which is not overlapping with observations used to built the model. The sensitivity was in between 96%-99% and the specificity was over 99% for all classes (for all the 5 classes). Overall, the model is well developed to predict the exercise classes during weight lifting.

Annex: figures

1. For further understanding the data

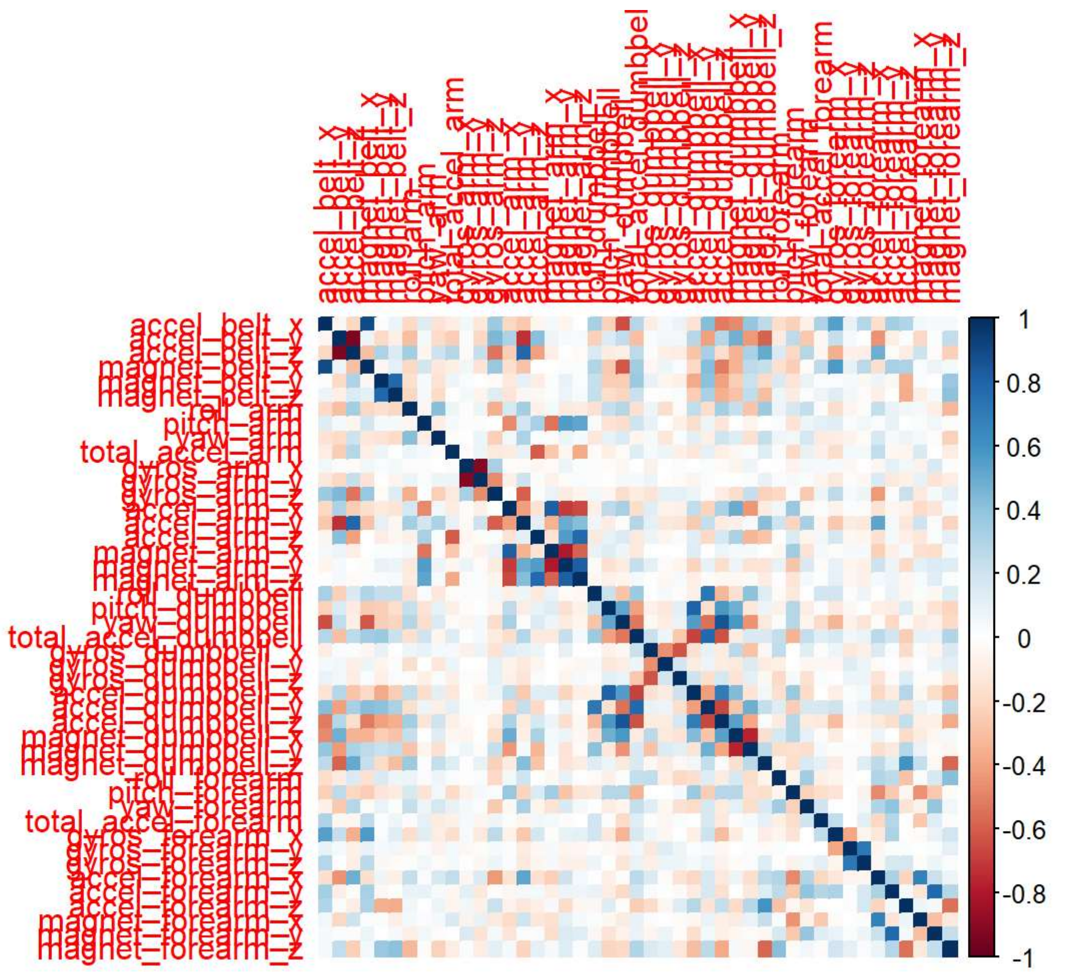
```
plot(mytrain_SUB2$classe, col="yellow", main="Plot of levels of variable classe within the Train
TrainingSet data set", xlab="classe", ylab="Frequency")
```



2.

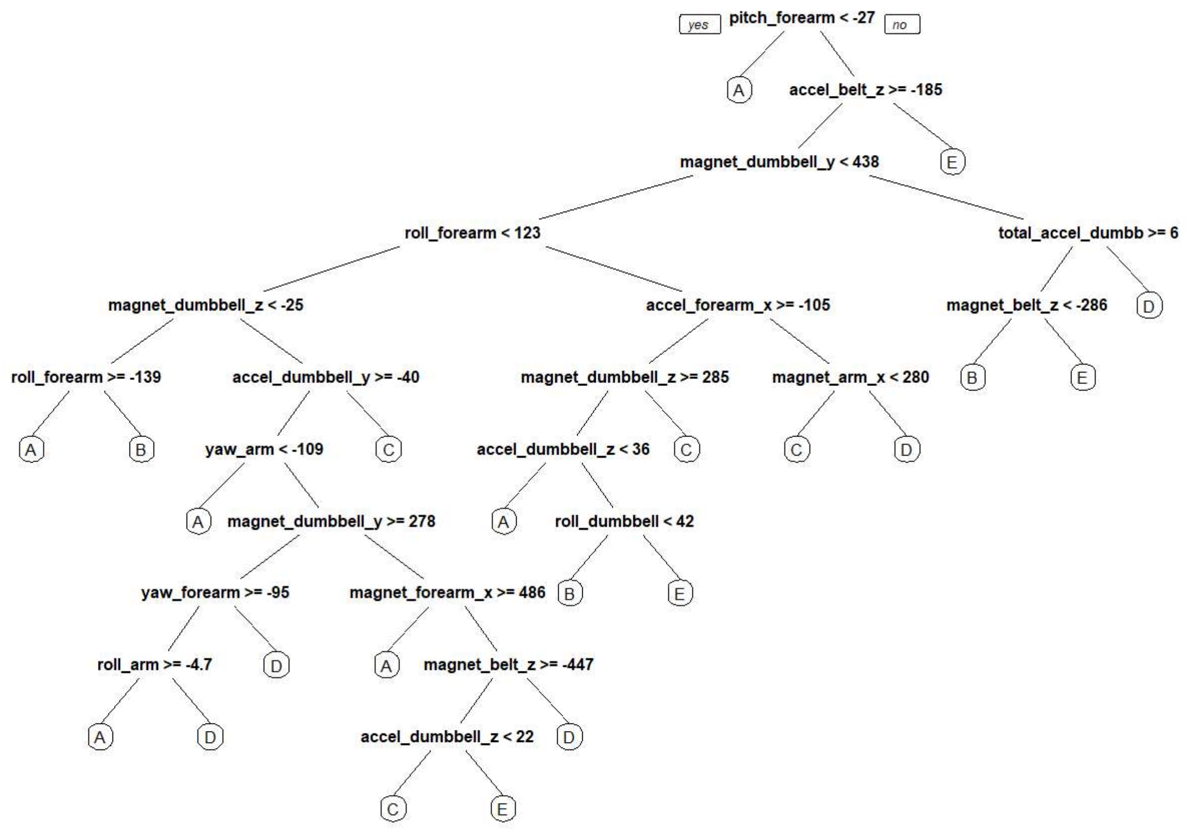
Correlation Matrix Visualization

```
corrPlot <- cor(mytrain_SUB2[, -length(names(mytrain_SUB2))])
corrplot(corrPlot, method="color")
```



3. Decision Tree Visualization

```
treeModel <- rpart(classe ~ ., data = mytrain_SUB2, method="class")
prp(treeModel) # fast plot
```


yes
pitch_forearm < -27
no
A
accel_belt_z >= -185
magnet_dumbbell_y < 438
E
roll_forearm < 123
total_accel_dumbb >= 6
magnet_dumbbell_z < -25
accel_forearm_x >= -105
magnet_belt_z < -286
D
roll_forearm >= -139
accel_dumbbell_y >= -40
magnet_dumbbell_z >= 285
magnet_arm_x < 280
B
E
A
B
yaw_arm < -109
C
accel_dumbbell_z < 36
C
C
D
A
magnet_dumbbell_y >= 278
A
roll_dumbbell < 42
yaw_forearm >= -95
magnet_forearm_x >= 486
B
E
roll_arm >= -4.7
D
A
magnet_belt_z >= -447
A
D
accel_dumbbell_z < 22
D
C
E