Machine_Learning_Assignment

roei fredo

July 19, 2019

Overview

Final report for the peer review practical machine learning assignment is presented as a part of the data science specialization. The purpose of this work was to to quantify how well an excersize of barbell lifts is practiced. For this reason, measurements about barbell lifts performed by 6 participants were taken, using body sensors. training and testing data was downloaded, read and cleaned. The. 4 machine learning models were tested for predicting barber lifts quality. The most accurate model was performed by the random forest method. Finally, the model was used to predict 20 long test data for excersize quality.

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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data Source

The training data (pmlTraining) for this project are available here:

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

The test data (pmlTesting) are available here:

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

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

Open Libraries

First, I opened the needed libraries for this analysis:

```
library(caret)
library(rattle)
library(parallel)
library(doParallel)
library(xtable)
library(corrplot)
```

Loading the Data

Than I loaded and read the data into R, data is divided into training set and out of sample set

```
url1 <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
training <- basename(url1);
if (!file.exists(training)) {
    download.file(url1, training, method='curl')
}

url2 <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
testing <- basename(url2);
if (!file.exists(testing)) {
    download.file(url2, testing, method='curl')
}

#read data

pmlTraining <- read.csv(training, na.strings = c("", "NA"))
pmlTesting <- read.csv(testing, na.strings = c("", "NA"))
```

Splitting the data

Than, the data was partitioned into training set and test set:

```
## [1] "training set dimensions:  Rows:13737, Cols:160"
```

```
## [1] "testing set dimensions:  Rows:5885, Cols:160"
```

After building the data structure, it is time to select the right features for the model:

Feature Selection

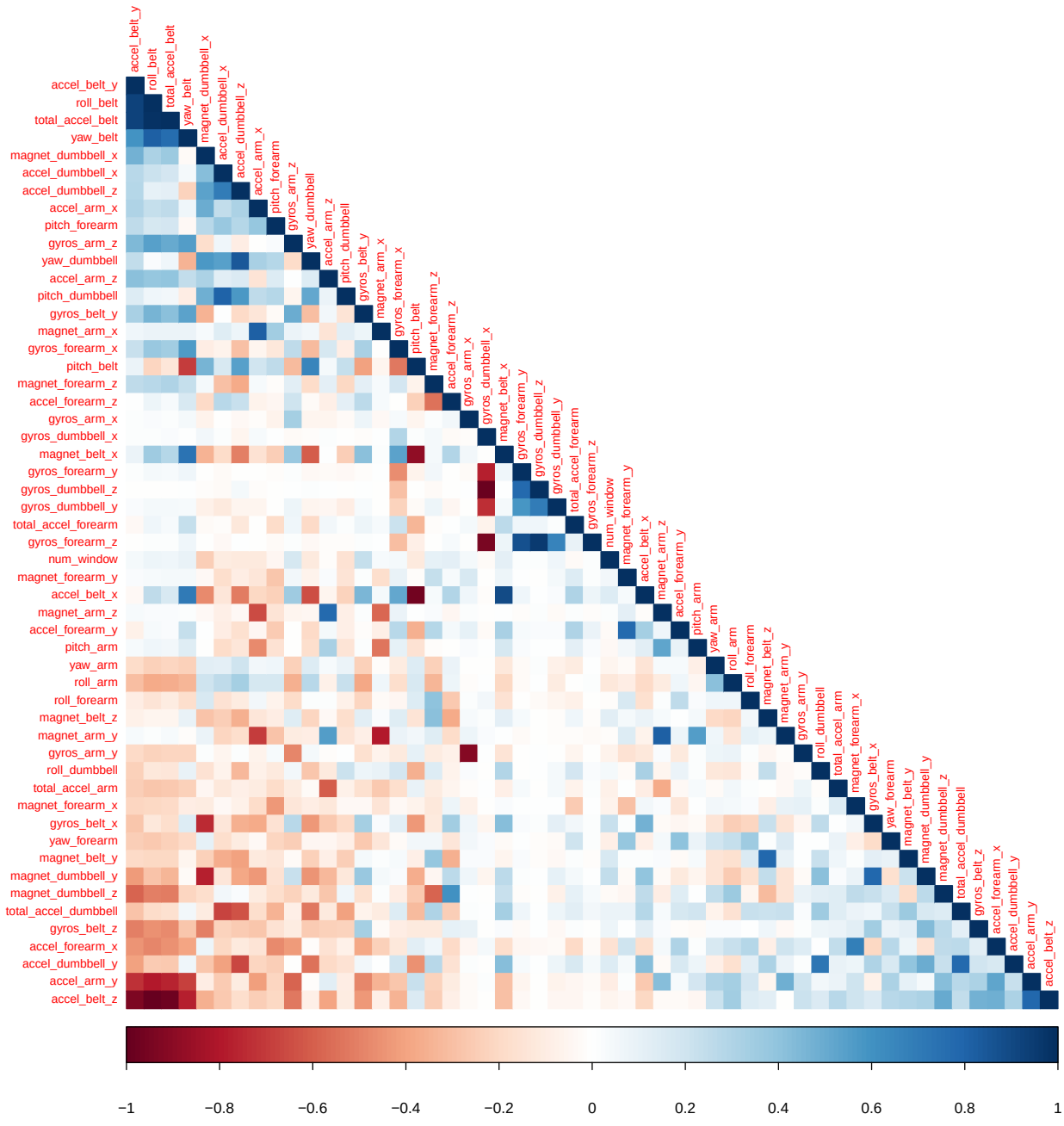
Cleaning unnecessary Features

Cleaning the data was done in 4 stages. First, columns with more than 20% NAs was eliminated from data frame. Than,features with near zero variance, that do not affect the outcome values were removed. Third, the first 5 descriptive columns were removed. Finally, i removed rows that gave summaries of the raw data. I needed only the raw data for this prediction, and in addition summary values were not a part of the test set

```
#Establish the NAs % for every columns for feature selection
pmlNA <- sapply(training, FUN = function(x) mean(is.na(x)))
training <- training[,pmlNA<0.2]
testing <- testing[,pmlNA<0.2]

#eliminating average observations
training <- training[training$new_window == "no",]
testing <-testing[testing$new_window == "no",]

#remove feature with zero variance
nzv <- nearZeroVar(training)
training <- training[, -nzv]
testing <- testing[, -nzv]

#removing non-numeric first 5 columns

training <- training[,-c(1:5)]
testing <- testing[,-c(1:5)]
```

Correlations Between Features

Before proceeding to the modeling procedures, a correlation among predictor variables was performed:



The above graph shows correlations across 53 features, ordered by first principal component order. The dark shades of blue and red show high correlated pairs. As can be seen, correlations are quite negligible and it was assumed that they had a little influence on model accuracy and variance.

Model Selection

For model selection 4 models were formed for test set prediction: 1. Classification Trees 2. Random Forest 3. Gradient Boosting Machine 4. Linear Discriminant Analysis

Models were fitted to the training dataset, every model were cross-validated using 5-fold CV. Then testset outcome predictions were established and compared to the real observational data for accuracy calculations:

Defining cross validation method and parallel processing

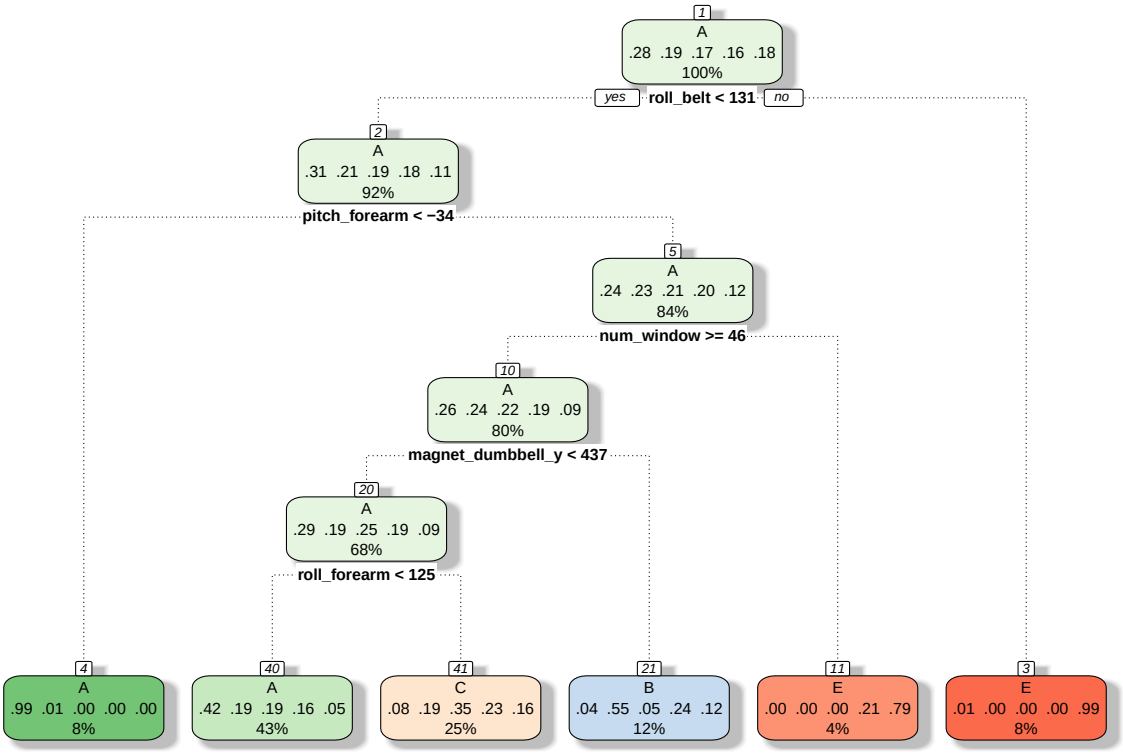
At first, large computation times made it difficult to model with random forest and GBM techniques. For this reason, I parallel computing to minimize processing time.

```
set.seed(323)
#Start Parallel Computing
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)

#5-fold Cross Validation Train Control
data_ctrl <- trainControl(method="cv", number = 5, allowParallel = T)
```

Here we can see the confusion matrices final tables and accuracy values for each model:

Predicting with Classification Trees

```
set.seed(323)
modFitrpart <- train(classe ~ ., method = "rpart", data = training, trControl = data_ctrl)
#print(modFitrpart$finalModel)
fancyRpartPlot(modFitrpart$finalModel)
```



Rattle 2019−Jul−28 01:26:37 roeifredo

```
prpart <- predict(modFitrpart, testing)
confrpart <- confusionMatrix(prpart, testing$classe)
Accurrpart <- confrpart$overall[1]
confrpart$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1496  461  467  394  103
##          B   23  389   34  180   82
##          C  123  271  502  334  214
##          D    0    0    0    0    0
##          E    5    0    0   38  658
```

Predicting with Random Forest

```
set.seed(323)
modFitrf <- train(classe ~ ., method = "rf", data = training, trControl = data_ctrl)
#print(modFitrf$finalModel)
prf <- predict(modFitrf, testing)
confrf <- confusionMatrix(prf, testing$classe)
Accurrf <- confrf$overall[1]
confrf$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1647    2    0    0    0
```

```
##          B    0 1119    2    0    0
##          C    0    0 1001    3    0
##          D    0    0    0  942    0
##          E    0    0    0    1 1057
```

Predicting with GBM

```
set.seed(323)
modFitgbm <- train(classe ~ ., method = "gbm", data = training, verbose = F, trControl = data_ctrl)
#print(modFitgbm$finalModel)
pgbm <- predict(modFitgbm, testing)
confgbm <- confusionMatrix(pgbm, testing$classe)
Accurgbm <- confgbm$overall[1]
confgbm$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1644   17    0    2    0
##          B    2 1089   13    6    3
##          C    0   13  987    8    3
##          D    1    2    3  927    7
##          E    0    0    0    3 1044
```

Predicting with Linear Discriminant Analysis

```
set.seed(323)
modFitlda <- train(classe ~ ., method = "lda", data = training, trControl = data_ctrl)
#print(modFitlda$finalModel)
plda <- predict(modFitlda, testing)
conflda <- confusionMatrix(plda, testing$classe)
Accurlda <- conflda$overall[1]
conflda$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1385  144   97   55   42
##          B   49  732  104   51  160
##          C   99  146  649  106   93
##          D  106   37  117  699   91
##          E    8   62   36   35  671
```

Compare Accuracies across Model

```
Accuracycomp <- data.frame(Accuracy = c(Accurrpart, Accurrf, Accurgbm, Accurlda), row.names = c("rpart"
xt <- xtable(Accuracycomp, digits = 4, align = "cc")
print(xt, type = "html")
```

Accuracy

rpart

0.5274

rf

0.9986

gbm

0.9856

lda

0.7163

As can be seen, Random Forest gave the most accurate predictions with an accuracy value of 0.9986145. For this reason this model was chosen to predict the 20 observation long test data:

The out of sample error was estimated by subtracting the test set accuracy percentage (99.88%) from 100%, getting an out of sample error of 0.12%.

Predicting TestSet

Applying the random forest model to the test data gave the following predictions:

```
##  [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
```

Since every row of the tested 20 rows is a different test case, the projected probability of making 20 right predictions equals (model accuracy)ˆ20 = 99.63% ˆ 20 = 92.85%