Prediction Assignment Writeup

Danilo Correa

9/26/2019

1 - 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 this website (see the section on the Weight Lifting Exercise Dataset).

2 - Data

The training data for this project are available here, and the test data here.

3 - Data processing

Installing and/or loading the required R packages:

```
if (!"data.table" %in% rownames(installed.packages())) {install.packages("data.table")}
library(data.table)

if (!"corrplot" %in% rownames(installed.packages())) {install.packages("corrplot")}
library(corrplot)

if (!"Hmisc" %in% rownames(installed.packages())) {install.packages("Hmisc")}
library(Hmisc)

if (!"caret" %in% rownames(installed.packages())) {install.packages("caret")}
library(caret)

if (!"gbm" %in% rownames(installed.packages())) {install.packages("gbm")}
library(gbm)

if (!"randomForest" %in% rownames(installed.packages())) {install.packages("randomForest")}
library(randomForest)

if (!"adabag" %in% rownames(installed.packages())) {install.packages("adabag")}
library(adabag)

if (!"nnet" %in% rownames(installed.packages())) {install.packages("nnet")}
library(nnet)

if (!"parallel" %in% rownames(installed.packages())) {install.packages("parallel")}
library(parallel)

if (!"doParallel" %in% rownames(installed.packages())) {install.packages("doParallel")}
library(doParallel)
```

Setting seed for reproducibility:

```
set.seed(19)
```

Direcly download and read the datasets:

```
train.WLE <- fread("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
test.WLE <- fread("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
```

Removing near zero variance predictors:

```
nzv <- nearZeroVar(train.WLE)
train.WLE <- train.WLE[, -nzv]
test.WLE  <- test.WLE[, -nzv]
dim(train.WLE)
```

```
## [1] 19622   124
```

Removing features with more than 95% NA values:

```
train.WLE <- train.WLE[, which(lapply(train.WLE, function(x) mean(is.na(x))) < 0.95), with=F]
test.WLE <- test.WLE[, which(lapply(test.WLE, function(x) mean(is.na(x))) < 0.95), with=F]
dim(train.WLE)
```

```
## [1] 19622    59
```

Removing ID's and timestamp columns:

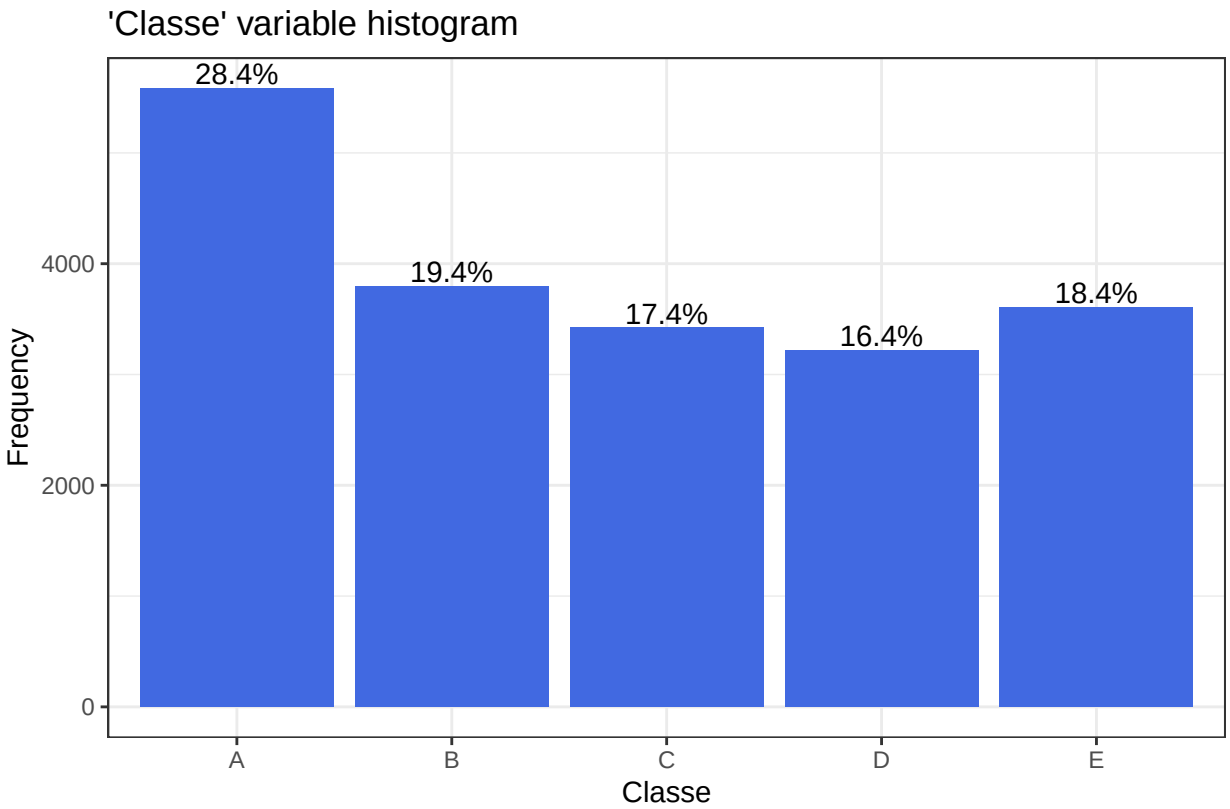
```
train.WLE <- train.WLE[, -(1:6)]
test.WLE <- test.WLE[, -(1:6)]
dim(train.WLE)
```

```
## [1] 19622    53
```

4 - Exploratory analysis

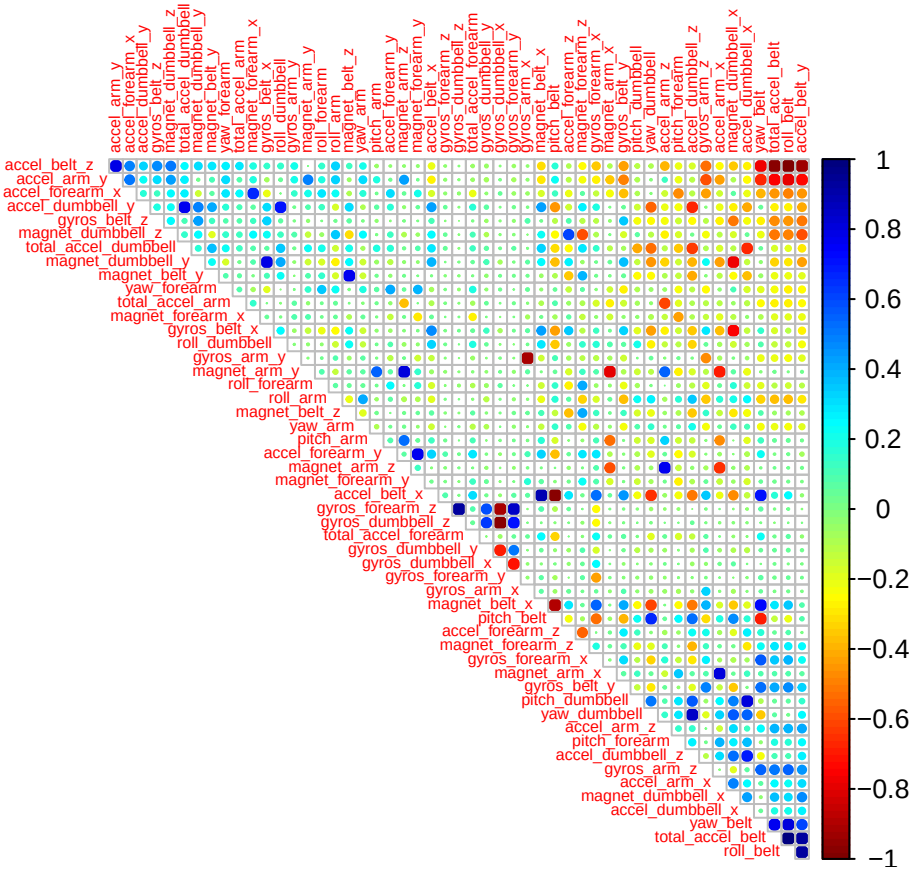
Making a histogram from the "classe" variable to check it's frequency distribution among levels:

```
ggplot(data.frame(train.WLE),aes(x = train.WLE$classe, y = stat(count))) +
    geom_bar(fill = "royalblue") +
    geom_text(aes(label=scales::percent(..count../sum(..count..))),
              stat='count',vjust= -0.2) +
    theme_bw() +
    labs(x = "Classe", y = "Frequency") +
    labs(title = "'Classe' variable histogram",
         caption = "Source: Weight Lifting Exercises dataset")
```



Ploting a correlation matrix:

```
col4 <- colorRampPalette(c("#7F0000", "red", "#FF7F00", "yellow", "#7FFF7F",
                           "cyan", "#007FFF", "blue", "#00007F"))
corrplot(cor(train.WLE[,-53]), diag = FALSE, order = "FPC",
         tl.pos = "td", tl.cex = 0.5, addrect = 2, col = col4(100), type = "upper")
```



Removing highly correlated features (correlation > 90%):

```
hc <- findCorrelation(cor(train.WLE[,-53]), cutoff = 0.9)
train.WLE <- train.WLE[,-..hc]
test.WLE <- test.WLE[,-..hc]
dim(train.WLE)
```

```
## [1] 19622    46
```

5 - Building the prediction model

Training data partition:

```
in.train.WLE <- createDataPartition(train.WLE$classe, p=0.70, list=F)
training.partition <- train.WLE[in.train.WLE,]
testing.partition <- train.WLE[-in.train.WLE,]
```

Applying the Repeated k-fold Cross Validation to estimate the further test accuracy with the training set:

```
fitControl <- trainControl(method = "repeatedcv", number = 5, repeats = 2, classProbs = TRUE)
```

Fitting models with 4 different algorithms:

```
# Set seed for reproducibility:
set.seed(19)

# Create a Parallel Socket Cluster:
cl <- makeCluster(detectCores())
```

```r
# register the parallel backend with the foreach package:
registerDoParallel(cl)

# Using the "Stochastic Gradient Boosting" method (gbm):
model.1 <- train(classe ~ ., data = training.partition, method = "gbm", na.action=na.omit, trControl = 

# Using the "Bagged AdaBoost" method (AdaBag):
model.2 <- train(classe ~ ., data = training.partition, method = "AdaBag", na.action=na.omit, trControl

# Using the "Neural Networks with Feature Extraction" method (pcaNNEt):
model.3 <- train(classe ~ ., data = training.partition, method = "pcaNNet", na.action=na.omit, maxit = 

# Using Random Forest method (rf):
model.4 <- train(classe ~ ., data = training.partition, method = "rf", na.action=na.omit, trControl = f

# Stop parallel cluster:
stopCluster(cl)
```

Comparing models resamples:

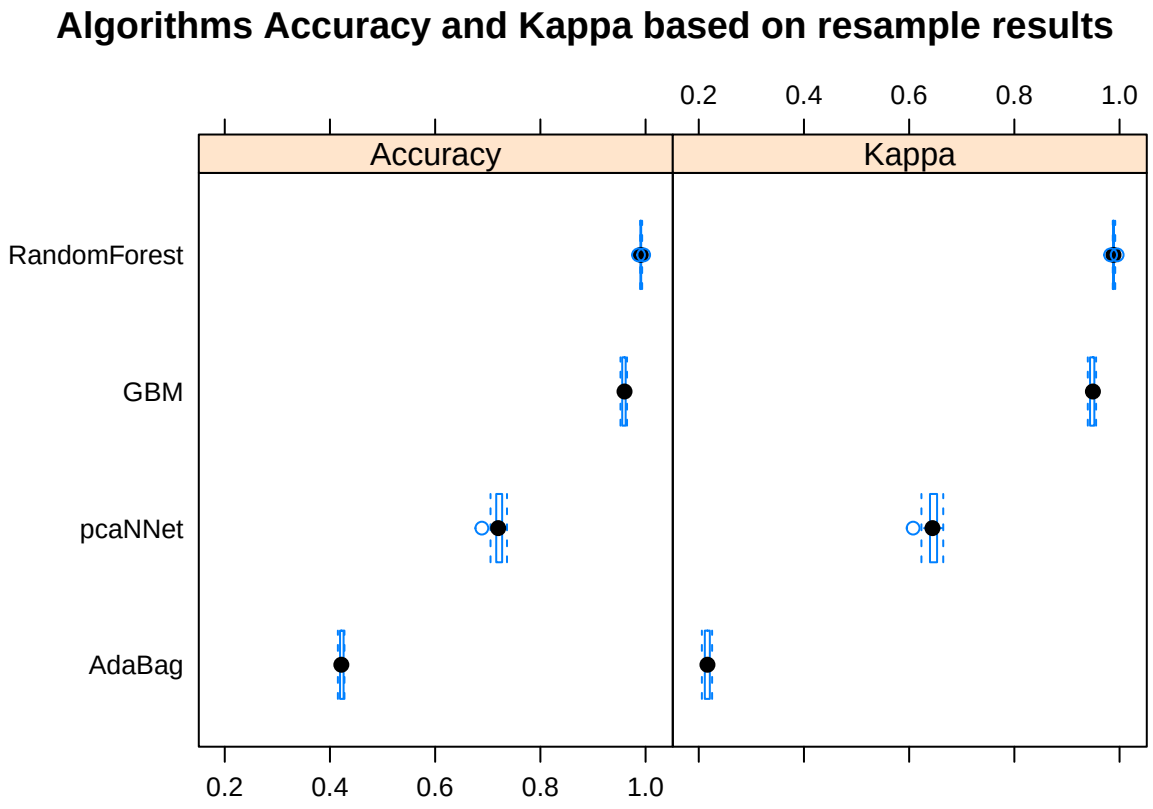
```r
# collect resamples
results <- resamples(list(GBM = model.1, AdaBag = model.2, pcaNNet=model.3, RandomForest=model.4))

# summarize the distributions
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: GBM, AdaBag, pcaNNet, RandomForest
## Number of resamples: 10
##
## Accuracy
##                   Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## GBM          0.9522942 0.9557032 0.9599708 0.9590152 0.9620426 0.9647016
## AdaBag       0.4150600 0.4194283 0.4216562 0.4218904 0.4253346 0.4275310
## pcaNNet      0.6887514 0.7164436 0.7198252 0.7184973 0.7255697 0.7365357
## RandomForest 0.9865259 0.9901756 0.9908992 0.9911548 0.9916272 0.9963623
##              NA's
## GBM             0
## AdaBag          0
## pcaNNet         0
## RandomForest    0
##
## Kappa
##                   Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## GBM          0.9396197 0.9439747 0.9493495 0.9481464 0.9519712 0.9553519
## AdaBag       0.2059485 0.2122387 0.2165352 0.2163291 0.2205974 0.2254418
## pcaNNet      0.6075409 0.6403447 0.6442807 0.6428197 0.6512034 0.6647963
## RandomForest 0.9829577 0.9875719 0.9884864 0.9888105 0.9894090 0.9953984
##              NA's
## GBM             0
## AdaBag          0
## pcaNNet         0
```

```
## RandomForest    0
```

Making a boxplot with the results to easier the algorithm selection:

```
bwplot(results, metric=c("Accuracy", "Kappa"), main = "Algorithms Accuracy and Kappa based on resample r
```



According to the boxplot above, the **Random Forest algorithm performed better among those evaluated based on accuracy and kappa evaluation metrics (Accuracy: 99,11%, Kappa: 98,88%).**

6 - Apply the best machine learning algorithm to the 20 test cases

Predicting the test data:

```
# Predicting classes with Random Forest algorithm (model.4):
predict.test.WLE <- predict(model.4, test.WLE)

# Checking results:
predict.test.WLE
```

```
##  [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
```

7 - Conclusions

- the **Random Forest algorithm** proved to be the best algorithm among those evaluated, with and **accuracy of 99,11%**;
- the 20 test classes predicted by the selected model are (in order): **B A B A A E D B A A B C B A E E A B B B**.

8 - References

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.