ML_FinalPorject

elobo

August 30th 2016

Machine Learning applied to Human Activity Recognition (HAR) classification problem

Synopsis

This document wants to show the concepts learned on Machine Learning field and thus apply those techiques to a specicif data set in order to predict the performance of some physical exercises done by six individuals during a monitored workout sessions. Raw data used to populate dataset comes from accelerometers used on the belt, forearm, arm, and dumbell during workout. The purpose of the analysis is to build a model which can be used to predict how well some specific exercises will be performed after a short session of repetitions. As our putcome reperesents some quilitative response the nature of the issue focuses on resolving a classification problem.

Data collection

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://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

Model Building

The model built will consist on selecting the best fetures out of the total provided by the original data set. Then a revision of nature of predictors and deep cleaning will end to a polish reduced data set. According to the source, the caegorical outcome will consist on:

“Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).”

For comprison purposes there will be sused a simple classification tree and and random forest algorithm and the best with higher model accuracy will be taken as the chosen for a particular cases for prediction. Caret packege will be mainly used to test several ML algorithm.

Read more: http://groupware.les.inf.puc-rio.br/har#literature#ixzz4IrzHF3TY (http://groupware.les.inf.puc-rio.br/har#literature#ixzz4IrzHF3TY)

Loading data

```
library(readr); library(caret); library(ggplot2);  library(dplyr); library(rpart)
setwd("/Users/elobo/Documents/COURSERA/DATA_SCIENCE/CURSO8")
training_set <- read.csv("pml-training.csv", stringsAsFactors = TRUE)
testing_set <- read.csv("pml-testing.csv",stringsAsFactors = TRUE)
```

Base cleaning on training_set

```
# Cleaning original training set
# check zero variance predictor, nzv = TRUE, those variables should be thrown out fro
m final model
        nzv.check_tr <-  nearZeroVar(training_set, saveMetrics=TRUE)
        nzv.check_tr$varnames <- rownames(nzv.check_tr)
        names.to.remove <- nzv.check_tr %>% filter(nzv %in% "TRUE") %>% select(varnam
es)
        names.to.remove <- sapply(names.to.remove, as.character)[,1]

        training_set_clnd <- training_set[,!(names(training_set) %in% names.to.remove)]

        dim(training_set_clnd)
```

```
## [1] 19622   100
```

```
        # removing variables that are not valuable for final model
        training_set_clnd <- training_set_clnd[,-c(1:6)]
        # checking NAs of the current data set
        # function to check NA > 90 %
        calc_NA <- function(x){sum(is.na(x))/length(x)*100}
        check_NA <- as.data.frame(apply(training_set_clnd,2,calc_NA))
        table(check_NA[,1] > 90.0)
```

```
##
## FALSE  TRUE
##    53    41
```

```
# subsetting for final model (without NA-Variables)
        tr2 <- training_set_clnd[,!check_NA[,1] > 90.0]
```

general check for cleaned data

```
# general check
        str(tr2); summary(tr2)
```

```
# general check
        dim(tr2)
```

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

```
        names(tr2)
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

Cross- Validation

From original training set, once cleaned, we split data for correct cross-validation. The model will be trained with data of the subtrainining portion and the fitting performance will be done over the subtesting portion of the original train set.

```
#Creating the subtraining/subtest sets
        set.seed(1234)          # seed for reproducibility
        inTrain <- createDataPartition(tr2$classe, p = 3/4)[[1]]
        subtraining <- tr2[ inTrain,]
        subtesting <- tr2[-inTrain,]
        dim(subtraining); dim(subtesting)
```

```
## [1] 14718    53
```

```
## [1] 4904   53
```

final data set dimention:

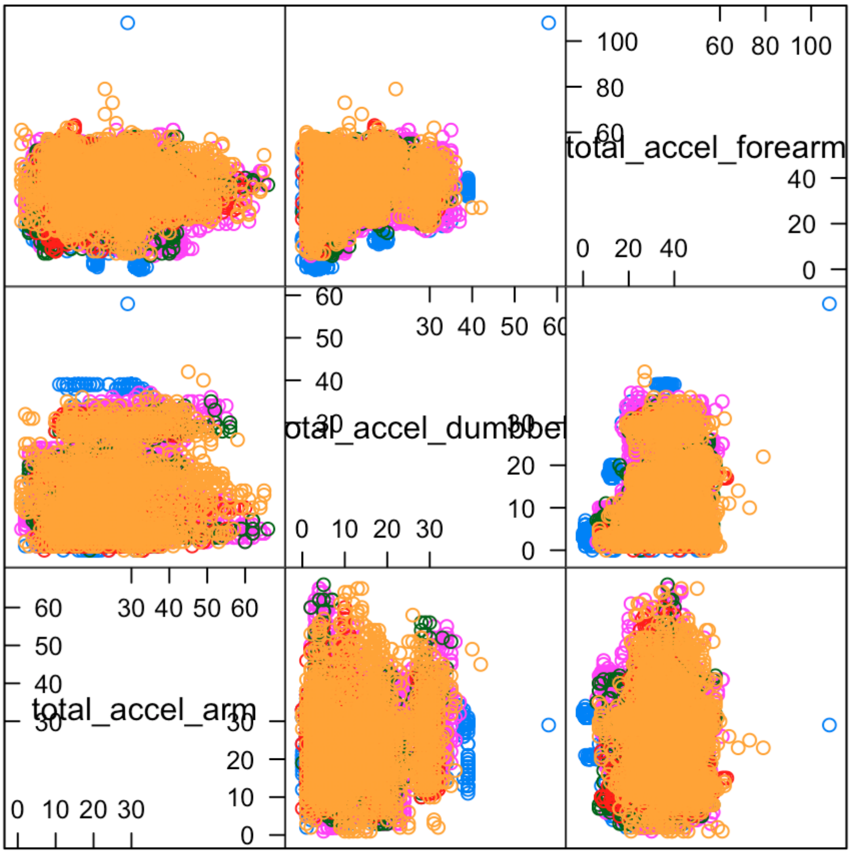
```
## [1] 14718    53
```

```
## [1] 4904   53
```

Exploratory Anlysis

We check some total-based predictors behavior

```
        featurePlot(x=subtraining[,c("total_accel_arm", "total_accel_dumbbell" ,"tota
l_accel_forearm")],
                    y=subtraining$classe,
                    plot="pairs")
```



Scatter Plot Matrix

Checking Models

```
        #Classification tree evaluation
        rpart.fit <- train(classe ~ ., method="rpart", data = subtraining, preProcess=c
("center", "scale"))
        rpart.predi <- predict(rpart.fit, subtesting)
        # model check
        rpart.fit$finalModel
```

```
## n= 14718
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 14718 10533 A (0.28 0.19 0.17 0.16 0.18)
##    2) roll_belt< 1.050356 13483  9308 A (0.31 0.21 0.19 0.18 0.11)
##      4) pitch_forearm< -1.609042 1163     4 A (1 0.0034 0 0 0) *
##      5) pitch_forearm>=-1.609042 12320  9304 A (0.24 0.23 0.21 0.2 0.12)
##       10) magnet_dumbbell_y< 0.6692937 10462  7499 A (0.28 0.18 0.24 0.19 0.11)
##         20) roll_forearm< 0.8245366 6480  3837 A (0.41 0.18 0.18 0.17 0.06) *
##         21) roll_forearm>=0.8245366 3982  2663 C (0.08 0.18 0.33 0.22 0.18) *
##       11) magnet_dumbbell_y>=0.6692937 1858   912 B (0.029 0.51 0.041 0.23 0.19) *
##    3) roll_belt>=1.050356 1235    10 E (0.0081 0 0 0 0.99) *
```

```
# prediction check
rpart.cmx <- confusionMatrix(rpart.predi, subtesting$classe)
rpart.cmx$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   4.953100e-01   3.398572e-01   4.812209e-01   5.094046e-01   2.844617e-01
## AccuracyPValue  McnemarPValue
##  3.016071e-212            NaN
```

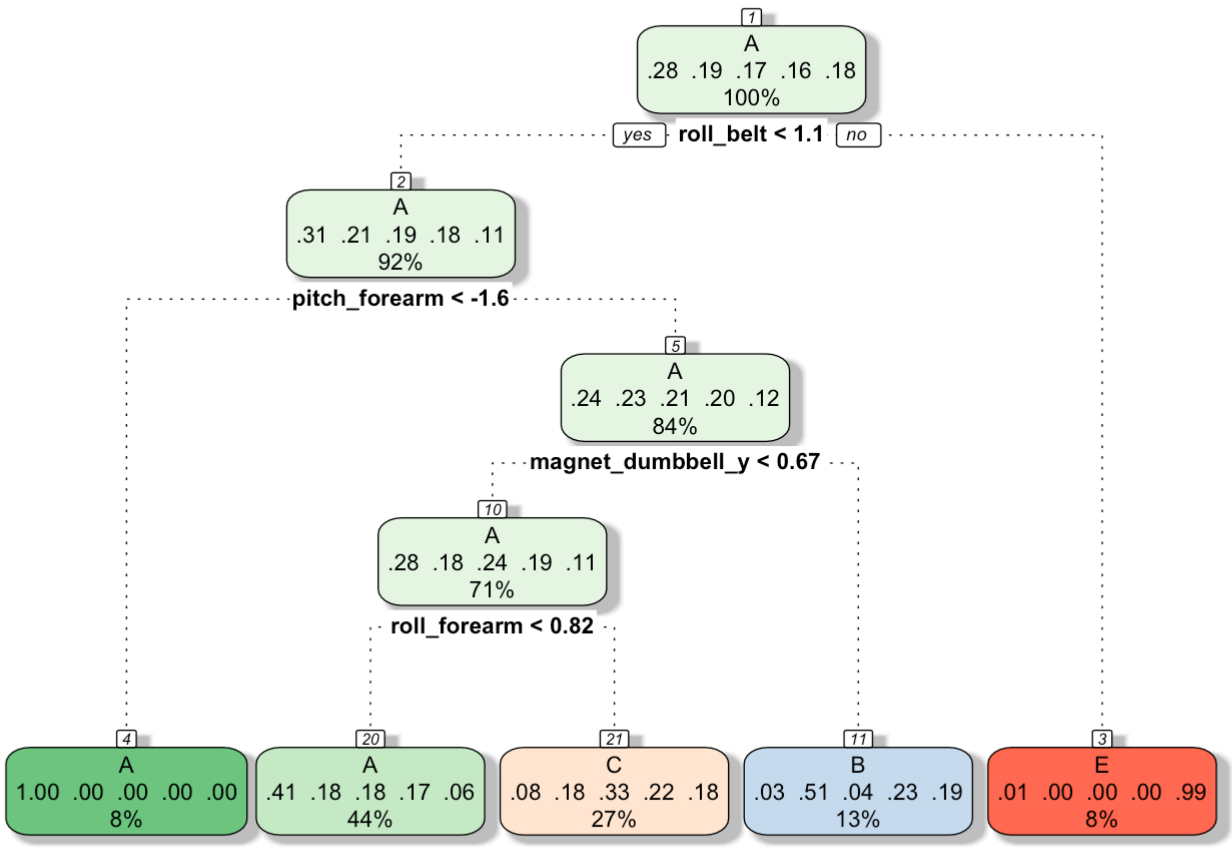
```
rattle::fancyRpartPlot(rpart.fit$finalModel)
```

```
## Warning: Failed to load RGtk2 dynamic library, attempting to install it.
```

```
## Please install GTK+ from http://r.research.att.com/libs/GTK_2.24.17-X11.pkg
```

```
## If the package still does not load, please ensure that GTK+ is installed and that
 it is on your PATH environment variable
```

```
## IN ANY CASE, RESTART R BEFORE TRYING TO LOAD THE PACKAGE AGAIN
```



Rattle 2016-Aug-30 20:00:04 elobo

Random Forest

Using a Random Forest method regressin variable classe on all predictors.

```
# Random Forest model evaluation
        fitControl <- trainControl(method = "cv", number = 5)
        rf.fit <- train(classe  ~ ., method="rf", data=subtraining, trControl = fitCo
ntrol)
# model check
        rf.fit$finalModel$confusion
# prediction check
        rf.predi <- predict(rf.fit, subtesting)
        cf.cmx <- confusionMatrix(rf.predi, subtesting)
```

Model Performance:

Comparing both model it is seen that rf algorithm shows better accuracy that classification tree.

```
cf.cmx <- confusionMatrix(rf.predi, subtesting$classe)
cf.cmx$overall
      Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull AccuracyPV
alue  McnemarPValue 
     0.9936786      0.9920026      0.9910392      0.9957010      0.2844617      0.000
0000            NaN 

cf.cmx$table
          Reference
Prediction    A    B    C    D    E
         A 1395    7    0    0    0
         B    0  939    8    1    0
         C    0    3  845    8    1
         D    0    0    2  794    0
         E    0    0    0    1  900
```

Assignment for Project Quiz.

```
#filtering variables used in cross validation
# formatting testing_data set to model fit. Leaving relevant variables (columns)
        comm_names <- names(testing_set) %in% names(subtraining)
        TSET <- testing_set[,comm_names]
        str(TSET)
```

```
# Prediction test
        predict(rf.fit, TSET)
 [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
```

CONCLUSION

For the model choosed, looking at the confusion matrix, the out of sample error is very low showing that the chosen model is not overfitting the testing data set. Then an acurracy of .099 will give us a good chance of correctly presume a right result when predicting specific data for a new testing set.