Machine Learning Course Project

ETSulato

10/11/2020

The dataset refers to data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. “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 needed packages

```
library(caret)
library(ggplot2)
library(dplyr)
library(randomForest)
library(e1071)
```

Downloading and loading the data

```
if (!file.exists("pml-training.csv")) {
  URLtrain <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
  download.file(URLtrain, destfile = "pml-training.csv", method = "curl")
}
if (!file.exists("pml-testing.csv")) {
  URLtest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"
  download.file(URLtest, destfile = "pml-testing.csv", method = "curl")
}

training_raw <- read.csv("pml-training.csv",
na.strings=c("#DIV/0!","NA"), row.names=1)
testing_raw <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!","NA"),
row.names=1)
```

Exploring and cleaning the data

Fist, the str() function was used in order to obtain the classes of each column and the dimensions of the dataset.

```
str(training_raw)

## 'data.frame':    19622 obs. of  159 variables:
##  $ user_name               : chr  "carlitos" "carlitos" "carlitos"
"carlitos" ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084231 1323084231
1323084232 1323084232 1323084232 1323084232 1323084232 1323084232
1323084232 ...
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328
304277 368296 440390 484323 484434 ...
##  $ cvtd_timestamp          : chr  "05/12/2011 11:23" "05/12/2011
11:23" "05/12/2011 11:23" "05/12/2011 11:23" ...
##  $ new_window              : chr  "no" "no" "no" "no" ...
##  $ num_window              : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42
1.42 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09
8.13 8.16 8.17 ...
##  $ yaw_belt                : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4
-94.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ kurtosis_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt.1    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x            : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02
0.02 0.03 ...
##  $ gyros_belt_y            : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z            : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02
```

```
-0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x           : int  -21 -22 -20 -22 -21 -21 -22 -22 -20
-21 ...
##  $ accel_belt_y           : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z           : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x          : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y          : int  599 608 600 604 600 603 599 603 602
609 ...
##  $ magnet_belt_z          : int  -313 -311 -305 -310 -302 -312 -311 -
313 -312 -308 ...
##  $ roll_arm               : num  -128 -128 -128 -128 -128 -128 -128 -
128 -128 -128 ...
##  $ pitch_arm              : num  22.5 22.5 22.5 22.1 22.1 22 21.9
21.8 21.7 21.6 ...
##  $ yaw_arm                : num  -161 -161 -161 -161 -161 -161 -161 -
161 -161 -161 ...
##  $ total_accel_arm        : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ var_accel_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x            : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02
0.02 ...
##  $ gyros_arm_y            : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -
0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z            : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02
-0.02 ...
##  $ accel_arm_x            : int  -288 -290 -289 -289 -289 -289 -289 -
289 -288 -288 ...
##  $ accel_arm_y            : int  109 110 110 111 111 111 111 111 109
110 ...
##  $ accel_arm_z            : int  -123 -125 -126 -123 -123 -122 -125 -
124 -122 -124 ...
##  $ magnet_arm_x           : int  -368 -369 -368 -372 -374 -369 -373 -
372 -369 -376 ...
##  $ magnet_arm_y           : int  337 337 344 344 337 342 336 338 341
334 ...
##  $ magnet_arm_z           : int  516 513 513 512 506 513 509 510 518
516 ...
##  $ kurtosis_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ skewness_yaw_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_dumbbell: num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

By the analysis of the results from above, it can be observed that many columns are wrongly set as factors when they present quantitative data. Therefore, the whole dataset was coerced to the numeric class.The user_name and classe (columns 1 and 159 in the training_raw dataset) variables were not coerced into numeric and the function was applied to all the other columns execept these two. By sapply() function, it was possible to notice that all the variables, apart from user_name and classe, were successfully coerced. Missing values were also removed. Any variable that presented 500 or more missing values were promptly removed and the ones bellow this cutoff value were considered for imputation of NAs by the column mean. However, by the sum of the results of is.na() function, no variable with remaning NAs were identified.

```
# Coercing into numeric
training_raw[-c(1,159)] <- lapply(training_raw[-c(1,159)], function(x) {
    if(is.factor(x)|is.integer(x)|is.logical(x))
as.numeric(as.character(x)) else x
})

# Dealing with NAs
training<-training_raw[apply(is.na(training_raw),2, sum)<500]
```

Variable reduction

Considering that variables with low variance would not contribute to the final response of each classe, those variables were checked by nearZeroVar() function of caret package in order to be previously removed if still necessary.

```
Low_variance_columns<- nearZeroVar(training, saveMetrics = TRUE)
training <- training[,!Low_variance_columns$nzv]
```

Performing the analysis of the data

Crossvalidation

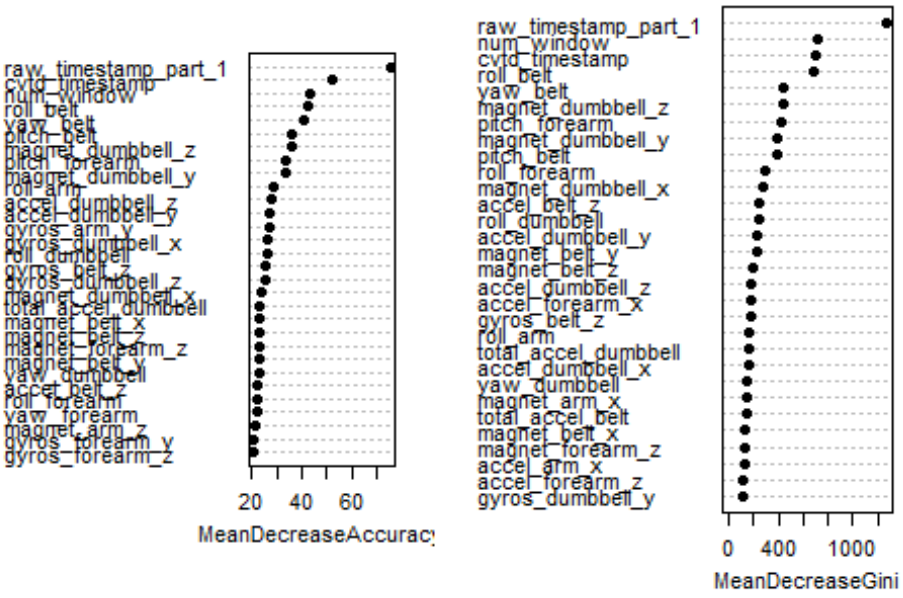
In order to perform the crossvalidation step rows were partioned into training and crossvalidation groups.

```
set.seed(333)
inTrain <- createDataPartition(training$classe, p = 0.70, list = FALSE)
training <- training[inTrain,]
crossvalidation <- training[-inTrain,]
```

Model training and validation

The modelling was performed by the Random Forest method. This method was chosen because it is good to handle large datasets with unknown interactions between variables. The model was obtained by the randomForest() function from the randomForest package. Then, the performance of the model was evaluated in the crossvalidation dataset by predict() and the results were assessed by confusionMatrix(). This was perfomed to allow us to spot overfitting.

```
model_RF<- randomForest(factor(classe) ~., data = training, importance =
TRUE, ntrees = 10)
varImpPlot(model_RF, pch=16, cex=0.7)
```




```
prediction_RF <- predict(model_RF, crossvalidation)
confusionMatrix(table(prediction_RF, crossvalidation$classe))
```

```
## Confusion Matrix and Statistics
##
##
## prediction_RF    A    B    C    D    E
##             A 1184    0    0    0    0
##             B    0  789    0    0    0
##             C    0    0  721    0    0
##             D    0    0    0  669    0
##             E    0    0    0    0  741
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9991, 1)
##     No Information Rate : 0.2885
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
```

```
## Sensitivity            1.0000   1.0000   1.0000    1.000   1.0000
## Specificity            1.0000   1.0000   1.0000    1.000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000    1.000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000    1.000   1.0000
## Prevalence             0.2885   0.1923   0.1757    0.163   0.1806
## Detection Rate         0.2885   0.1923   0.1757    0.163   0.1806
## Detection Prevalence   0.2885   0.1923   0.1757    0.163   0.1806
## Balanced Accuracy      1.0000   1.0000   1.0000    1.000   1.0000
```

An accuracy of 100% was obtained for the model.

Test dataset prediction

The prediction was performed in the test dataset.

```
test <- predict(model_RF, testing_raw)
print(test)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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

The Random Forest method successfully modelled the dataset with high accuraccy.