Practical Machine Learning

Eric BAUDON

In this report we are going to explain how we build our prediction model using the Train data available for the activity.

1.Exploratory Analysis

First we loaded the Train data and split in Training and Testing set.

```
Training_data=read.csv("D:/Coursera/Specialisation - Data Science/8.Machine Learning/Project/pml-training.csv")
library("caret")
```

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

```
set.seed(12345)
sample = createDataPartition(Training_data$classe, p = 3/4)[[1]]
Training = Training_data[sample,]
Testing = Training_data[-sample,]
```

Then we looked at the structure of Train data with following code

```
dim(Training)
```

```
## [1] 14718   160
```

```
str(Training[1:15])
```

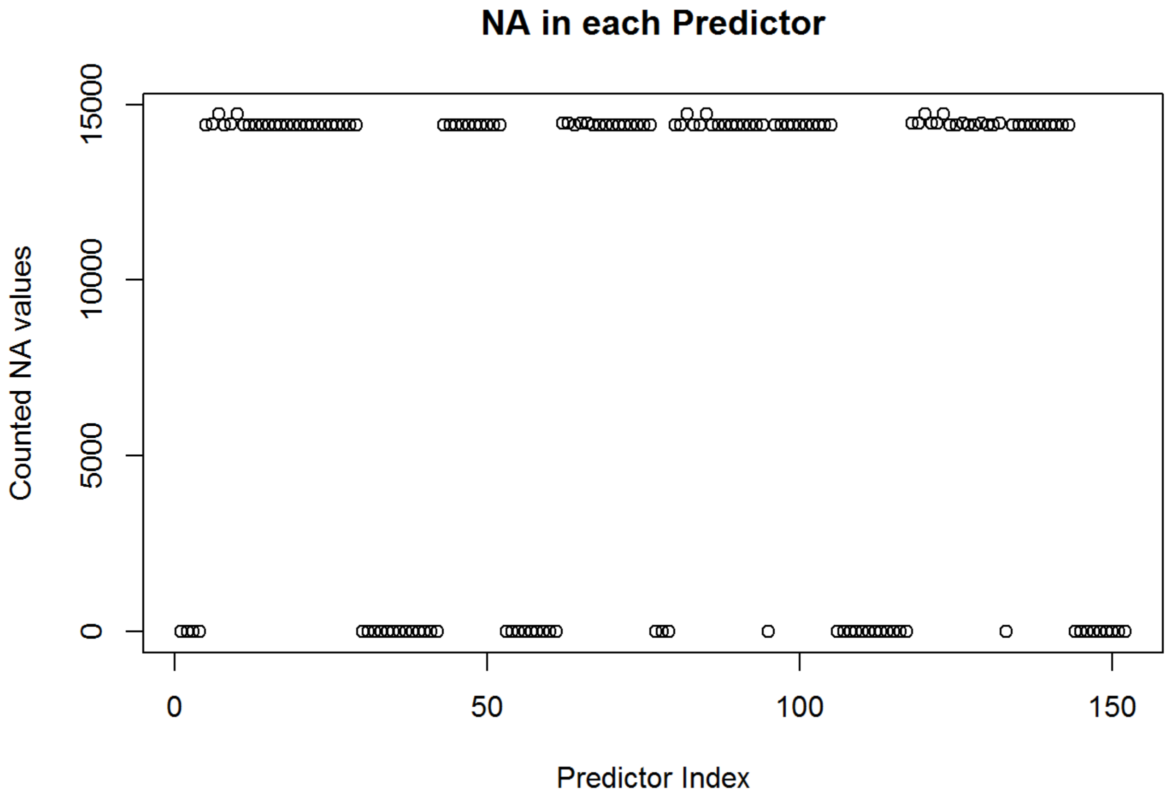
```
## 'data.frame':    14718 obs. of  15 variables:
##  $ X                   : int  2 3 4 5 6 7 8 11 12 13 ...
##  $ user_name           : Factor w/ 6 levels "adelmo","carlitos",..: 2 2
2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1: int  1323084231 1323084231 1323084232 132308423
2 1323084232 1323084232 1323084232 1323084232 1323084232 1323084232 ...
##  $ raw_timestamp_part_2: int  808298 820366 120339 196328 304277 368296 4
40390 500302 528316 560359 ...
##  $ cvtd_timestamp      : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9
9 9 9 9 9 9 9 9 ...
##  $ new_window          : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1
1 1 ...
##  $ num_window          : int  11 11 12 12 12 12 12 12 12 12 ...
##  $ roll_belt           : num  1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.45 1.4
3 1.42 ...
##  $ pitch_belt          : num  8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.18 8.1
8 8.2 ...
##  $ 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  : Factor w/ 397 levels "","-0.016850",..: 1 1 1 1
1 1 1 1 1 1 ...
##  $ kurtosis_picth_belt : Factor w/ 317 levels "","-0.021887",..: 1 1 1 1
1 1 1 1 1 1 ...
##  $ kurtosis_yaw_belt   : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1
1 1 1 ...
##  $ skewness_roll_belt  : Factor w/ 395 levels "","-0.003095",..: 1 1 1 1
1 1 1 1 1 1 ...
```

We can see that:

- The first 7 columns referres to user and timing. Not very usefull for prediction.
- The last column [160] is the one with the classe we want to predict.
- Then from column 8 to 159 are the predictor (some are factors or integers instead of numercial).

```
##Select only predictors
library("dplyr")
Training.clean<-select(Training,8:159)
##Convert all predictors to numerical
asNumeric <- function(x) as.numeric(as.character(x))
factorsNumeric <- function(d) modifyList(d, lapply(d[, sapply(d, is.facto
r)],asNumeric))
integerNumeric <- function(d) modifyList(d, lapply(d[, sapply(d, is.intege
r)],asNumeric))
Training.clean<-integerNumeric(Training.clean)
Training.clean<-factorsNumeric(Training.clean)
```

Finally, we can observe that most of predictors in Train set are almost all the time equal to N/A becasue no value was registered. This kind of predictor can’t be very usefull and then were eliminated.




```
Training.clean<-Training.clean[,colSums(is.na(Training.clean))==0]
Variables<-colnames(Training.clean)## Represent our set of predictors.
dim(Variables)
```

```
## NULL
```

We are now remaining with only 52 predictors which seem to be the important ones for classification.

2.Models creation

We are going to build 3 prediction models using the training set: “Random Forest”“,”Bagging" and “Linear Discrepancy Analysis”:

```
library("caret")
model_rf<-train(Training$classe~., data=Training.clean, method="rf")
model_gbm<-train(Training$classe~., data=Training.clean, method="gbm")
model_lda<-train(Training$classe~., data=Training.clean, method="lda")
```

3 Model Selection

We are then going to apply the models on the testing set. First we need to clean the Testing set such as we did for the Training set:

```
Testing.clean<-Testing[,Variables] #Select only the predictors defined abov
e.
Testing.clean<-integerNumeric(Testing.clean)
Testing.clean<-factorsNumeric(Testing.clean)
```

Now we can apply our models to the Testing Set and calculate for each one the prediction accuracy.

```
pred_rf<-predict(model_rf,Testing.clean)
pred_gbm<-predict(model_gbm,Testing.clean)
pred_lda<-predict(model_lda,Testing.clean)

accuracy_rf = sum(pred_rf == Testing$classe) / length(pred_rf)
accuracy_gbm = sum(pred_gbm == Testing$classe) / length(pred_gbm)
accuracy_lda = sum(pred_lda == Testing$classe) / length(pred_lda)

results<-data.frame("lda"=accuracy_lda,"gbm"=accuracy_gbm,"rf"=accuracy_rf)
results
```

```
##         lda       gbm        rf
## 1 0.6980016 0.9606444 0.9924551
```

Clearly **Random Forest** is our best model with an accuracy around **99%**.