Prediction Assignment - Practical Machine Learning Course

Francesco Golfieri

December, 31 2016

Goals of the project

The goal of this project is to predict the manner in which users of devices such as Jawbone Up, Nike FuelBand, and Fitbit exercise.
Given a dataset of training routines, the “classe” variable is the one we aim to predict.
We may use any of the other variables to predict with, and we will create a report describing how we built our model, how we used cross validation, what we think the expected out of sample error is, and why we made the choices we did.

Loading Libraries

For this analysis we will use the following libraries:

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

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

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## Loading required package: MASS
```

Set seed

We need to set a seed for the random values generation

```
set.seed(999)
```

Loading Data

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/redmachlearn/pml-testing.csv
(https://d396qusza40orc.cloudfront.net/redmachlearn/pml-testing.csv)

```
train_Url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
train_dataset_raw <- read.csv(url(train_Url), na.strings=c("NA","#DIV/0!",""))
test_Url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
test_dataset_raw <- read.csv(url(test_Url), na.strings=c("NA","#DIV/0!",""))
```

We now want to clean the variables with a high number of not available values (70%). There are not suitable to be used in the prediction model.

```
high_na <- sapply(colnames(train_dataset_raw), function(x)
if(sum(is.na(train_dataset_raw[, x])) > 0.70*nrow(train_dataset_raw))
  {return(TRUE)}
else
  {return(FALSE)}
)
train_dataset_raw <- train_dataset_raw[, !high_na]
```

As we can see, only 60 variables out of 160 remain in the model. The first 7 columns of our train dataset are system or user informations, not suitable to be used in the prediction model.

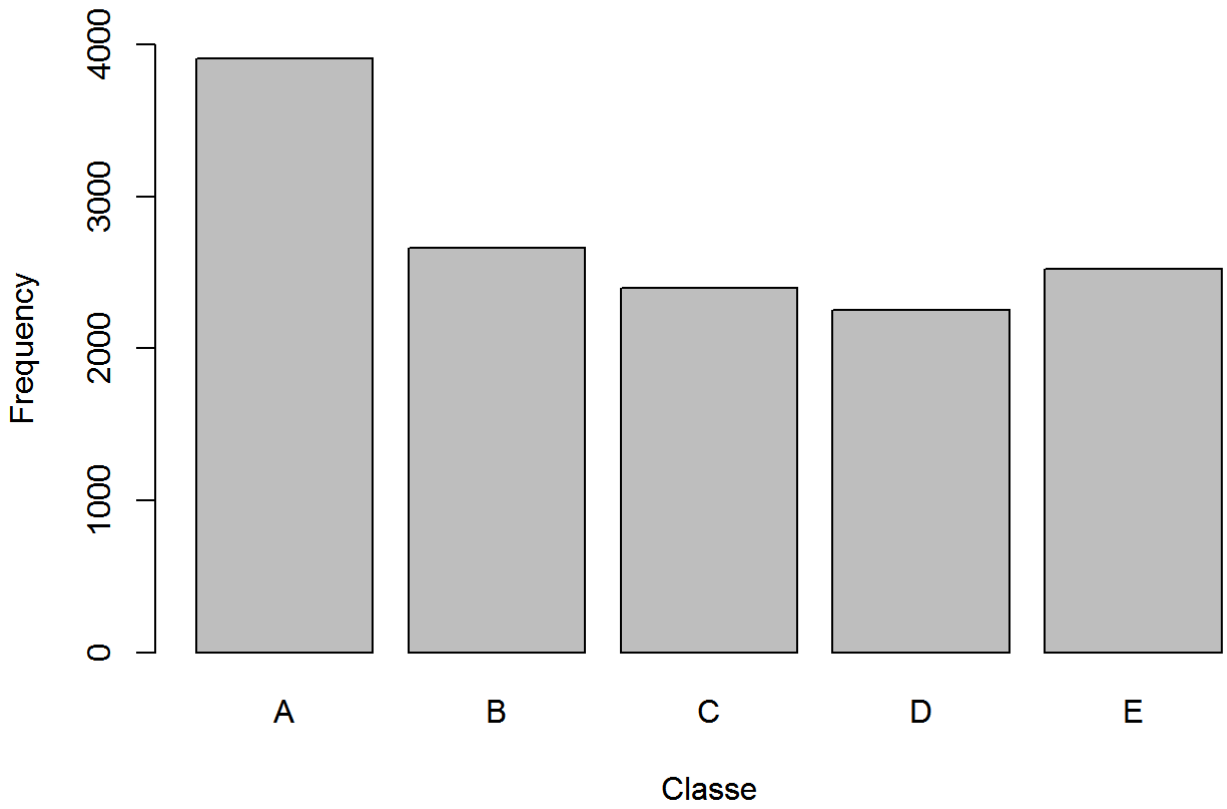
```
train_dataset <- train_dataset_raw[, -(1:7)]

#We apply the same modifications to the test dataset
clean_columns <- c(colnames(train_dataset))
test_dataset <- test_dataset_raw[,names(test_dataset_raw) %in% clean_columns]
```

It’s now necessary to partition the training set in two subsets: one will be used for actual training the model and one for actual testing it. To partition it we, of course, use the “Classe” variable, the one we’ll aim to predict.

```
train_dataset_wip <- createDataPartition(y=train_dataset$classe, p=0.7, list=FALSE)
#Working version training set
wtr <- train_dataset[train_dataset_wip, ]
#Working version testing set
wte <- train_dataset[-train_dataset_wip, ]
```

Let’s take a look to the “classe” variable of our clean training set

```
plot(wtr$classe, xlab = "Classe", ylab = "Frequency", ylim = c(0,4000))
```



As we can see, it has 5 values with the same order of size. We can proceed with the modeling.

Predictive Models

You can now try many different methods to train our set. Before that, we setup a trControl parameters set to correctly control cross validation.

```
tc <- trainControl(method = "cv", number = 5, verboseIter=FALSE , preProcOptions="pca", allow
Parallel=TRUE)
```

We can now apply several predictive methods to obtain each corresponding model and to apply them to our testing set.

```
## Model01 = Random Forest (this can take a while)
Model01 <- train(classe ~ ., data = wtr, method = "rf", trControl = tc)
Prediction01 <- predict(Model01, wte)
CM01 <- confusionMatrix(Prediction01, wte$classe)

## Model02 = Decision tree
Model02 <- rpart(classe ~ ., data = wtr, method = "class")
Prediction02 <- predict(Model02, wte, type = "class")
CM02 <- confusionMatrix(Prediction02, wte$classe)

## Model03 = Linear Discriminant Analysis
Model03 <- train(classe ~ ., data = wtr, method = "lda", trControl = tc)
Prediction03 <- predict(Model03, wte)
CM03 <- confusionMatrix(Prediction03, wte$classe)

## Model04 = Bagged CART
Model04 <- train(classe ~ ., data = wtr, method = "treebag", trControl = tc)
Prediction04 <- predict(Model04, wte)
CM04 <- confusionMatrix(Prediction04, wte$classe)

## Model05 = Neural Network with Feature Extraction - Standard Neural Net proved in previous
 test to be very ineffective
Model05 <- train(classe ~ ., data = wtr, method = "pcaNNet", trControl= tc, verbose=FALSE)
```

```
## Loading required package: nnet
```

```
Prediction05 <- predict(Model05, wte)
CM05 <- confusionMatrix(Prediction05, wte$classe)
```

Now we can compare the accuracy of the 5 models:

```
mod <- c("Random Forest", "Decision tree","Linear Discriminant Analysis","Bagged CART","Neura
l Network w. Feature Extraction")
acc <- c(CM01$overall['Accuracy'],
                CM02$overall['Accuracy'],
                CM03$overall['Accuracy'],
                CM04$overall['Accuracy'],
                CM05$overall['Accuracy']
                )
cbind(mod,acc)
```

```
##          mod                                    acc
## Accuracy "Random Forest"                        "0.994222599830076"
## Accuracy "Decision tree"                        "0.750212404418012"
## Accuracy "Linear Discriminant Analysis"         "0.698895497026338"
## Accuracy "Bagged CART"                          "0.985556499575191"
## Accuracy "Neural Network w. Feature Extraction" "0.572812234494477"
```

Conclusions

As we can see, the best performing model is the first one (Random Forest) with an accuracy of over 99%. Bagged CART performed very well (98,66%). Let's use it to predict the 20 testing cases.

```
Prediction_Final <- predict(Model01, test_dataset)
```