Project Practical Machine Learning

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 the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

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. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Data Analysis and Predictions

We have to download some packages, such as caret, knitr, doMC (to run parallel). This last one is very important due to the excesive time the model takes to run.

```
library(caret)
```

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

```
library(knitr)
library(doMC)
```

```
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: parallel
```

```
registerDoMC(cores = 4)
set.seed(12345)
```

Now we read the data and perform some filtering. We remove missing values or empty strings:

```
dat.train <- read.csv("pml-training.csv", stringsAsFactors=FALSE)
dat.test <- read.csv("pml-testing.csv", stringsAsFactors=FALSE)

#This function removes features with missing data:
filterData <- function(idf) {
  # Since we have lots of variables, remove any with NA's
  # or have empty strings
  idx.keep <- !sapply(idf, function(x) any(is.na(x)))
  idf <- idf[, idx.keep]
  idx.keep <- !sapply(idf, function(x) any(x==""))
  idf <- idf[, idx.keep]
  col.rm <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2",
              "cvtd_timestamp", "new_window", "num_window")
  idx.rm <- which(colnames(idf) %in% col.rm)
  idf <- idf[, -idx.rm]
  return(idf)
}

#Finally we have to convert classe into a factor:
dat.train <- filterData(dat.train)
dat.train$classe <- factor(dat.train$classe)
dat.test <- filterData(dat.test)
```

This process give us a set of 52 columns, the rest were eliminated. Now we are going to build a prediction model using random forest and 5 fold cross validation to tune the parameters (due to excesive time in running we don't use 10)

```
cvCtrl <- trainControl(method = "cv", number = 5, allowParallel = TRUE, verboseIter = TRUE)
m1 <- train(classe ~ ., data = dat.train, method = "rf", trControl = cvCtrl)
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.

## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=27
## - Fold1: mtry=27
## + Fold1: mtry=52
## - Fold1: mtry=52
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=27
## - Fold2: mtry=27
## + Fold2: mtry=52
## - Fold2: mtry=52
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=27
## - Fold3: mtry=27
## + Fold3: mtry=52
## - Fold3: mtry=52
## + Fold4: mtry= 2
```

```
## - Fold4: mtry= 2
## + Fold4: mtry=27
## - Fold4: mtry=27
## + Fold4: mtry=52
## - Fold4: mtry=52
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry=27
## - Fold5: mtry=27
## + Fold5: mtry=52
## - Fold5: mtry=52
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 2 on full training set
```

```
Accuracy=round(max(head(m1$results)$Accuracy), 3)
```

```
Accuracy
```

```
## [1] 0.995
```

As we can see, Random forest give us good accuracy ,now let's do the prediction using the test data set.

```
test.pred.1 <- predict(m1, dat.test)
```

Predictions are:

```
test.pred.1
```

```
##  [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
```

Finally we have to generate the test results files to be uploaded for automated grading, we use the code given from the coursera website:

```
answers<-test.pred.1
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
pml_write_files(answers)
```

Final comment, all predictions were submitted and all of them are correct.