Which exercise did they do?

By Ling Tian

The datasets are from a study with 5 classes (sitting-down, standing-up, standing, walking, and sitting) collected on 8 hours of activities of 4 healthy subjects. The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set.

Load training data set

```
setwd("~/Desktop/Coursera/ML_predict-exerciseManner")
training_raw<-read.table("pml-training.csv", sep=",", na.string=c("NA",""), header=T)
```

Cleaning data

It's been noticed that in the training data set, many variables contain lots of NAs (19216 NAs out of 19622 observations). The number of observations for each class are 5580, 3797 ,3422 ,3216, and 3607 each. So it is unlikely that weather those variables are NAs or not is related to which exercise they did. As a result, all columns containing NAs are excluded from the data set for further analysis. Columns having unique values such as: "X", "user_name", "raw_timestamp_part_1","raw_timestamp_part_2","cvtd_timestamp", "new_window" ,"num_window" are also unrelated to exercise class, thus are removed from the data set as well. Zero covariates check was also performed and the results were negative for all variables so no changes applied. Then, the cleaned data set was split into two data sets: "mytrain" data set (75%) for training model and "mytest" data set (25%) for testing out of sample errors.

```
library(ggplot2)
library(caret)
```

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

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
training_noNA<-training_raw[,sapply(training_raw, Negate(anyNA)), drop=F]
training<-training_noNA[,8:ncol(training_noNA)]
nzv<-nearZeroVar(training, saveMetrics=T)
set.seed(123) #set seed 123 in order to repeat this analysis
intrain<-createDataPartition(y=training$classe, p=.75, list=F)
mytrain<-training[intrain,]
mytest<-training[-intrain,]
dim(mytrain); dim(mytest)
```

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

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

Model building

I decided to perform random forest model fitting on "mytrain" data set since it usually works well when data sets has lots of variables and gives a high accuracy. When train() function was at first applied for model fitting, it was running for hours. Then the approach was optimized by setting the train control method as "cv" and with 3 folds instead of 10 folds (default). This time, excursion took only 30 min with 500 trees generated. Then, I futher investigated the time efficiency issue by reducing the number of trees to 50. As a result, the excursion time went down to only 5 mins. The error rate of my final approach (In sample error) is 1.3%. So the out of error rate is expected to be around 1%. Mac 1.86 GHz Intel Core 2 Duo was used for all analysis for this study.

```
modrf<-train(mytrain$classe~., method="rf", data=mytrain, ntree=50, trControl = trainControl(method="cv
modrf$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 50, mtry = param$mtry, number = 3)
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 1.3%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4171    8    3    2    1 0.003345281
## B   34 2794   16    2    2 0.018960674
## C    5   36 2515    9    2 0.020257109
## D    4    0   50 2356    2 0.023217247
## E    0    3    6    7 2690 0.005912786
```

Cross validation

Prediction on "mytest"" data set using the model we've trained. The out of error rate is 0.92%, as the accuracy of the trained model is 99.08%.

```
prediction<-predict(modrf, newdata=mytest)
confusionMatrix(prediction, mytest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    5    0    0    0
##          B    0  941    9    0    0
##          C    0    3  845   23    1
##          D    2    0    1  780    0
##          E    0    0    0    1  900
##
## Overall Statistics
##
##                Accuracy : 0.9908
##                  95% CI : (0.9877, 0.9933)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9884
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9986   0.9916   0.9883   0.9701   0.9989
## Specificity            0.9986   0.9977   0.9933   0.9993   0.9998
## Pos Pred Value         0.9964   0.9905   0.9690   0.9962   0.9989
## Neg Pred Value         0.9994   0.9980   0.9975   0.9942   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2841   0.1919   0.1723   0.1591   0.1835
## Detection Prevalence   0.2851   0.1937   0.1778   0.1597   0.1837
## Balanced Accuracy      0.9986   0.9946   0.9908   0.9847   0.9993
```

Predict of the test cases

```
#Load the test data set and predict the classe for each cases. Submit the results in respective files.
testing<-read.table("pml-testing.csv", sep=",", na.string=c("NA",""), header=T)
prediction<-predict(modrf, newdata=testing)
prediction<-as.character(prediction)
cat("The prediction of the 20 test cases are:", prediction)
```

```
## The prediction of the 20 test cases are: B A B A A E D B A A B C B A E E A B B B
```

```r
#To generat files for submission
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(prediction)
```