Practical Machine Learning Final Assignment

Yann K

June 12, 2016

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 preparation

1. Load the data into dataframes (both training and testing sets)
2. Remove the first 7 columns that we will not use as predictor variables: `X`,`user_name`, `raw_timestamp_part_1`, `raw_timestamp_part_2`, `cvtd_timestamp`, `new_window`, `num_window`
3. Remove all the columns having their sums=NA, we will not use them as predictor variables
4. Remove near zero variances columns using the nearZeroVar from the caret package
5. Create a data partition using only 20% of the data for model fitting due to my laptop performance

```
set.seed(99)
library(caret)
library(plyr)
library(parallel)
library(doParallel)
#1. Download and create dataframes
teurl<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trurl<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
tedf<-read.csv(teurl)
trdf<-read.csv(trurl)
names(tedf)
```

```
##   [1] "X"                        "user_name"
##   [3] "raw_timestamp_part_1"     "raw_timestamp_part_2"
##   [5] "cvtd_timestamp"           "new_window"
##   [7] "num_window"               "roll_belt"
##   [9] "pitch_belt"               "yaw_belt"
##  [11] "total_accel_belt"         "kurtosis_roll_belt"
##  [13] "kurtosis_picth_belt"      "kurtosis_yaw_belt"
##  [15] "skewness_roll_belt"       "skewness_roll_belt.1"
##  [17] "skewness_yaw_belt"        "max_roll_belt"
##  [19] "max_picth_belt"           "max_yaw_belt"
##  [21] "min_roll_belt"            "min_pitch_belt"
##  [23] "min_yaw_belt"             "amplitude_roll_belt"
```

```
##  [25] "amplitude_pitch_belt"     "amplitude_yaw_belt"
##  [27] "var_total_accel_belt"     "avg_roll_belt"
##  [29] "stddev_roll_belt"         "var_roll_belt"
##  [31] "avg_pitch_belt"           "stddev_pitch_belt"
##  [33] "var_pitch_belt"           "avg_yaw_belt"
##  [35] "stddev_yaw_belt"          "var_yaw_belt"
##  [37] "gyros_belt_x"             "gyros_belt_y"
##  [39] "gyros_belt_z"             "accel_belt_x"
##  [41] "accel_belt_y"             "accel_belt_z"
##  [43] "magnet_belt_x"            "magnet_belt_y"
##  [45] "magnet_belt_z"            "roll_arm"
##  [47] "pitch_arm"                "yaw_arm"
##  [49] "total_accel_arm"          "var_accel_arm"
##  [51] "avg_roll_arm"             "stddev_roll_arm"
##  [53] "var_roll_arm"             "avg_pitch_arm"
##  [55] "stddev_pitch_arm"         "var_pitch_arm"
##  [57] "avg_yaw_arm"              "stddev_yaw_arm"
##  [59] "var_yaw_arm"              "gyros_arm_x"
##  [61] "gyros_arm_y"              "gyros_arm_z"
##  [63] "accel_arm_x"              "accel_arm_y"
##  [65] "accel_arm_z"              "magnet_arm_x"
##  [67] "magnet_arm_y"             "magnet_arm_z"
##  [69] "kurtosis_roll_arm"        "kurtosis_picth_arm"
##  [71] "kurtosis_yaw_arm"         "skewness_roll_arm"
##  [73] "skewness_pitch_arm"       "skewness_yaw_arm"
##  [75] "max_roll_arm"             "max_picth_arm"
##  [77] "max_yaw_arm"              "min_roll_arm"
##  [79] "min_pitch_arm"            "min_yaw_arm"
##  [81] "amplitude_roll_arm"       "amplitude_pitch_arm"
##  [83] "amplitude_yaw_arm"        "roll_dumbbell"
##  [85] "pitch_dumbbell"           "yaw_dumbbell"
##  [87] "kurtosis_roll_dumbbell"   "kurtosis_picth_dumbbell"
##  [89] "kurtosis_yaw_dumbbell"    "skewness_roll_dumbbell"
##  [91] "skewness_pitch_dumbbell"  "skewness_yaw_dumbbell"
##  [93] "max_roll_dumbbell"        "max_picth_dumbbell"
##  [95] "max_yaw_dumbbell"         "min_roll_dumbbell"
##  [97] "min_pitch_dumbbell"       "min_yaw_dumbbell"
##  [99] "amplitude_roll_dumbbell"  "amplitude_pitch_dumbbell"
## [101] "amplitude_yaw_dumbbell"   "total_accel_dumbbell"
## [103] "var_accel_dumbbell"       "avg_roll_dumbbell"
## [105] "stddev_roll_dumbbell"     "var_roll_dumbbell"
## [107] "avg_pitch_dumbbell"       "stddev_pitch_dumbbell"
## [109] "var_pitch_dumbbell"       "avg_yaw_dumbbell"
## [111] "stddev_yaw_dumbbell"      "var_yaw_dumbbell"
## [113] "gyros_dumbbell_x"         "gyros_dumbbell_y"
## [115] "gyros_dumbbell_z"         "accel_dumbbell_x"
## [117] "accel_dumbbell_y"         "accel_dumbbell_z"
## [119] "magnet_dumbbell_x"        "magnet_dumbbell_y"
## [121] "magnet_dumbbell_z"        "roll_forearm"
## [123] "pitch_forearm"            "yaw_forearm"
## [125] "kurtosis_roll_forearm"    "kurtosis_picth_forearm"
## [127] "kurtosis_yaw_forearm"     "skewness_roll_forearm"
## [129] "skewness_pitch_forearm"   "skewness_yaw_forearm"
## [131] "max_roll_forearm"         "max_picth_forearm"
```

```
## [133] "max_yaw_forearm"          "min_roll_forearm"
## [135] "min_pitch_forearm"        "min_yaw_forearm"
## [137] "amplitude_roll_forearm"   "amplitude_pitch_forearm"
## [139] "amplitude_yaw_forearm"    "total_accel_forearm"
## [141] "var_accel_forearm"        "avg_roll_forearm"
## [143] "stddev_roll_forearm"      "var_roll_forearm"
## [145] "avg_pitch_forearm"        "stddev_pitch_forearm"
## [147] "var_pitch_forearm"        "avg_yaw_forearm"
## [149] "stddev_yaw_forearm"       "var_yaw_forearm"
## [151] "gyros_forearm_x"          "gyros_forearm_y"
## [153] "gyros_forearm_z"          "accel_forearm_x"
## [155] "accel_forearm_y"          "accel_forearm_z"
## [157] "magnet_forearm_x"         "magnet_forearm_y"
## [159] "magnet_forearm_z"         "problem_id"
```

```
#2. Removing the first columns (X, username, timestamp, window) having non predictive data
tedf<-tedf[,8:160]
trdf<-trdf[,8:160]
dim(trdf)
```

```
## [1] 19622   153
```

```
#Checking for NAs in the training dataset
count(trdf[is.na(trdf)])
```

```
##      x    freq
## 1 <NA> 1287472
```

```
#3. Removing NA variables
trdfna<-trdf[,colSums(is.na(trdf))==0]

#4. Removing near zero variance columns
nz<-nearZeroVar(trdfna, saveMetrics=TRUE)
trdfna<-trdfna[, !as.logical(nz$nzv)]
dim(trdfna)
```

```
## [1] 19622    53
```

```
names(trdfna)
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
```

```
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

```
#5. Create training dataframe subset with only 20% of the data for performance on my laptop
it<-createDataPartition(trdfna$classe, p=.2, list=F)
training<-trdfna[it,]
testing<-trdfna[-it,]
```

Model Fitting

1. Setup parallel calculation cluster using all but one core of the laptop
2. We will use a random forest using the `train` function from the `caret` package with a `trainControl` parameter using the parallel cluster
3. Display the final model result

```
#1. Parallel calculation cluster
cluster<-makeCluster(detectCores()-1)
registerDoParallel(cluster)

#2. Random Forest model fitting
tparam<-trainControl(allowParallel=TRUE)
trmodel<-train(classe~., data=training, method="rf", importance=TRUE, trainControl=tparam)
stopCluster(cluster)
#3. Final model
trmodel
```

```
## Random Forest
##
## 3927 samples
##   52 predictor
##    5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 3927, 3927, 3927, 3927, 3927, 3927, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9527590  0.9402240
##   27    0.9561320  0.9445089
##   52    0.9414863  0.9259874
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
trmodel$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, importance = TRUE,      trainControl = ..2)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 2.85%
## Confusion matrix:
##      A   B   C   D   E class.error
## A 1108   5   1   2   0 0.007168459
## B   19 729  12   0   0 0.040789474
## C    1  20 657   6   1 0.040875912
## D    1   2  20 617   4 0.041925466
## E    0   4   6   8 704 0.024930748
```

Model Evaluation and out of sample error estimation

1. Use the model to predict the `classe` of the testing dataset (80% of the data)
2. Display the confusion matrix between the predicted values and the actual ones
3. Out sample error estimation
4. Plotting the top 20 indicators for information

```
#1. Evaluate training model on the testing dataset
pred<-predict(trmodel, testing)
#2. Confusion matrix
confusionMatrix(pred, testing$classe)
```

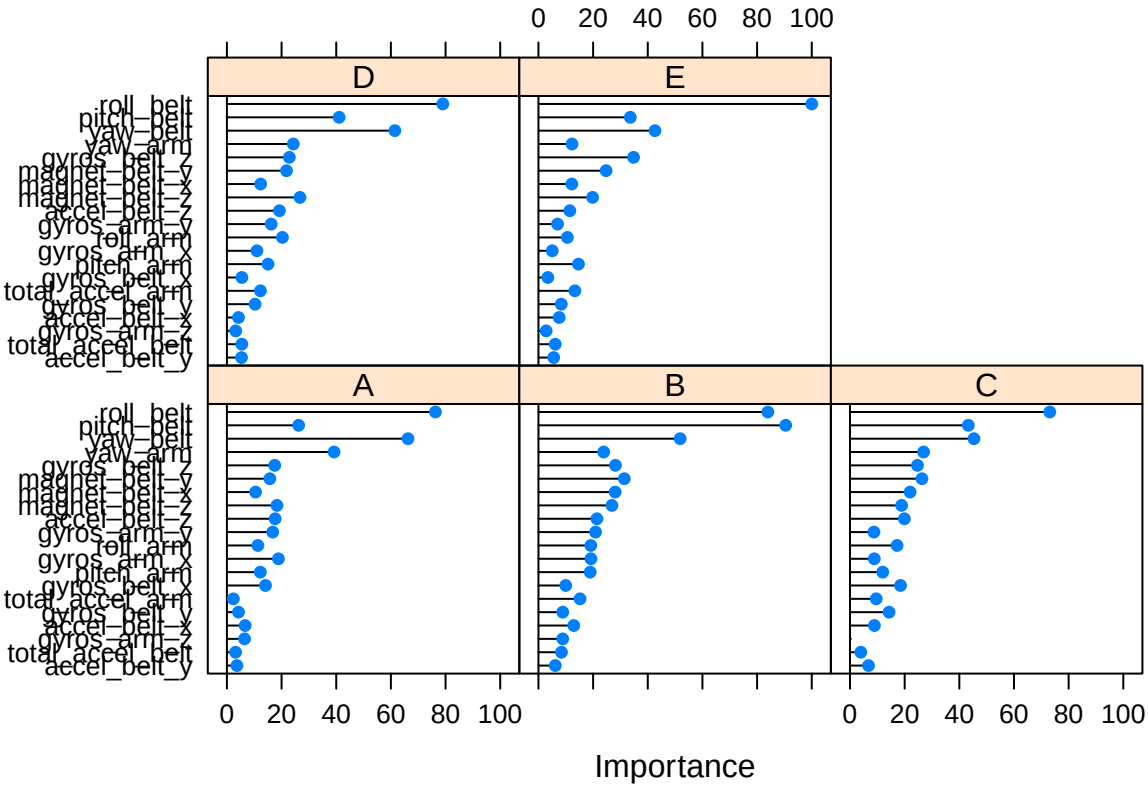
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4426   96    0    3    0
##          B   19 2900  104   17   16
##          C    3   30 2619   49   11
##          D   14    8   14 2494   12
##          E    2    3    0    9 2846
##
## Overall Statistics
##
##                Accuracy : 0.9739
##                  95% CI : (0.9713, 0.9763)
##     No Information Rate : 0.2844
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9669
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9915   0.9549   0.9569   0.9697   0.9865
## Specificity            0.9912   0.9877   0.9928   0.9963   0.9989
## Pos Pred Value         0.9781   0.9490   0.9657   0.9811   0.9951
## Neg Pred Value         0.9966   0.9892   0.9909   0.9941   0.9970
## Prevalence             0.2844   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2820   0.1848   0.1669   0.1589   0.1813
## Detection Prevalence   0.2883   0.1947   0.1728   0.1620   0.1822
## Balanced Accuracy      0.9913   0.9713   0.9749   0.9830   0.9927
```

```r
#3. Out sample error
outOfSampleErr<- (1 - (sum(pred==testing$classe)/length(pred)))*100
print(paste("Out of sample error is:", round(outOfSampleErr, digits=4)))
```

```
## [1] "Out of sample error is: 2.6123"
```

```r
#4. Plotting the top 20 most important predictors
#The most important predictors are roll_belt, pitch_belt, yaw_belt
vi<-varImp(trmodel)
vi[[1]]<-vi[[1]][1:20,]
plot(vi)
```



Applying model to the testing dataset

```
pred<-predict(trmodel, tedf)
pred
```

```
##  [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
```