Practical Machine Learning Course Project

Katherine Tansey

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. 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).

Load libraries and data

Load the R libraries needed for the analysis.

```r
library(AppliedPredictiveModeling)
library(caret)
```

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

```r
library(ElemStatLearn)
library(pgmm)
library(rpart)
library(e1071)
library(randomForest)
```

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

```r
library(rpart.plot)
library(RColorBrewer)
library(party)
```

```
## Loading required package: grid
## Loading required package: mvtnorm
## Loading required package: modeltools
## Loading required package: stats4
## Loading required package: strucchange
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: sandwich
```

```r
library(partykit)
```

```
##
## Attaching package: 'partykit'
##
## The following objects are masked from 'package:party':
##
##     cforest, ctree, ctree_control, edge_simple, mob, mob_control,
##     node_barplot, node_bivplot, node_boxplot, node_inner,
##     node_surv, node_terminal
```

Set the seed for the analysis so it can be reproduced.

```r
set.seed(12345)
```

Load in the data from the web. Check the size of the two datasets.

```r
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
validationUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))
validation <- read.csv(url(validationUrl), na.strings=c("NA","#DIV/0!",""))
dim(training)
```

```
## [1] 19622   160
```

```r
dim(validation)
```

```
## [1]  20 160
```

Split training dataset into two

The testing set will be the final set that we predict into, and so we will use it as a validation set (called validation). For this reason, we will split the training dataset into two, for training and testing the model we built. This will allow us to investigate the out of sample error rate of the model before we do the final prediction into the validation sets. The training data is split into 70% training and 30% testing.

```r
inTrain <- createDataPartition(y=training$classe, p=0.7, list=FALSE)
training1 <- training[inTrain, ]
testing <- training[-inTrain, ]
dim(training1)
```

```
## [1] 13737   160
```

```r
dim(testing)
```

```
## [1] 5885  160
```

Clean the data

Assess the data for the amount of missing (NA) values.

```
na_test = sapply(training1, function(x) {sum(is.na(x))})
table(na_test)
```

```
## na_test
##     0 13464 13465 13466 13471 13472 13488 13516 13518 13522 13523 13737
##    60    67     2     5     5     2     2     2     2     5     2     6
```

There are numerous variables without and missing, and then a lot of variables with almost all the data missing. Remove all variables with missing data, and just use the variables we have complete data on to build the model.

```
bad_columns = names(na_test[na_test!=0])
training1 = training1[, !names(training1) %in% bad_columns]
str(training1)
```

```
## 'data.frame':    13737 obs. of  60 variables:
##  $ X                   : int  2 3 4 5 6 7 8 12 13 14 ...
##  $ 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 1323084232 1323084232 1323084232 13230
##  $ raw_timestamp_part_2: int  808298 820366 120339 196328 304277 368296 440390 528316 560359 576390
##  $ 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 1.43 1.42 1.42 ...
##  $ pitch_belt          : num  8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.18 8.2 8.21 ...
##  $ 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 ...
##  $ gyros_belt_x        : num  0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 ...
##  $ gyros_belt_y        : num  0 0 0 0.02 0 0 0 0 0 0 ...
##  $ gyros_belt_z        : num  -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 -0.02 ...
##  $ accel_belt_x        : int  -22 -20 -22 -21 -21 -22 -22 -22 -22 -22 ...
##  $ accel_belt_y        : int  4 5 3 2 4 3 4 2 4 4 ...
##  $ accel_belt_z        : int  22 23 21 24 21 21 21 23 21 21 ...
##  $ magnet_belt_x       : int  -7 -2 -6 -6 0 -4 -2 -2 -3 -8 ...
##  $ magnet_belt_y       : int  608 600 604 600 603 599 603 602 606 598 ...
##  $ magnet_belt_z       : int  -311 -305 -310 -302 -312 -311 -313 -319 -309 -310 ...
##  $ roll_arm            : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm           : num  22.5 22.5 22.1 22.1 22 21.9 21.8 21.5 21.4 21.4 ...
##  $ yaw_arm             : num  -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...
##  $ total_accel_arm     : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ gyros_arm_x         : num  0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 0.02 ...
##  $ gyros_arm_y         : num  -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.02 0 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 0.02 0 0 0 0 0 -0.02 -0.03 ...
##  $ accel_arm_x         : int  -290 -289 -289 -289 -289 -289 -289 -288 -287 -288 ...
##  $ accel_arm_y         : int  110 110 111 111 111 111 111 111 111 111 ...
##  $ accel_arm_z         : int  -125 -126 -123 -123 -122 -125 -124 -123 -124 -124 ...
##  $ magnet_arm_x        : int  -369 -368 -372 -374 -369 -373 -372 -363 -372 -371 ...
##  $ magnet_arm_y        : int  337 344 344 337 342 336 338 343 338 331 ...
```

```
##  $ magnet_arm_z        : int  513 513 512 506 513 509 510 520 509 523 ...
##  $ roll_dumbbell       : num  13.1 12.9 13.4 13.4 13.4 ...
##  $ pitch_dumbbell      : num  -70.6 -70.3 -70.4 -70.4 -70.8 ...
##  $ yaw_dumbbell        : num  -84.7 -85.1 -84.9 -84.9 -84.5 ...
##  $ total_accel_dumbbell: int  37 37 37 37 37 37 37 37 37 37 ...
##  $ gyros_dumbbell_x    : num  0 0 0 0 0 0 0 0 0 0.02 ...
##  $ gyros_dumbbell_y    : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 ...
##  $ gyros_dumbbell_z    : num  0 0 -0.02 0 0 0 0 0 -0.02 -0.02 ...
##  $ accel_dumbbell_x    : int  -233 -232 -232 -233 -234 -232 -234 -233 -234 -234 ...
##  $ accel_dumbbell_y    : int  47 46 48 48 48 47 46 47 48 48 ...
##  $ accel_dumbbell_z    : int  -269 -270 -269 -270 -269 -270 -272 -270 -269 -268 ...
##  $ magnet_dumbbell_x   : int  -555 -561 -552 -554 -558 -551 -555 -554 -552 -554 ...
##  $ magnet_dumbbell_y   : int  296 298 303 292 294 295 300 291 302 295 ...
##  $ magnet_dumbbell_z   : num  -64 -63 -60 -68 -66 -70 -74 -65 -69 -68 ...
##  $ roll_forearm        : num  28.3 28.3 28.1 28 27.9 27.9 27.8 27.5 27.2 27.2 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.9 -63.9 ...
##  $ yaw_forearm         : num  -153 -152 -152 -152 -152 -152 -152 -152 -151 -151 ...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.02 0 0 ...
##  $ gyros_forearm_y     : num  0 -0.02 -0.02 0 -0.02 0 -0.02 0.02 0 -0.02 ...
##  $ gyros_forearm_z     : num  -0.02 0 0 -0.02 -0.03 -0.02 0 -0.03 -0.03 -0.03 ...
##  $ accel_forearm_x     : int  192 196 189 189 193 195 193 191 193 193 ...
##  $ accel_forearm_y     : int  203 204 206 206 203 205 205 203 205 202 ...
##  $ accel_forearm_z     : int  -216 -213 -214 -214 -215 -215 -213 -215 -215 -214 ...
##  $ magnet_forearm_x    : int  -18 -18 -16 -17 -9 -18 -9 -11 -15 -14 ...
##  $ magnet_forearm_y    : num  661 658 658 655 660 659 660 657 655 659 ...
##  $ magnet_forearm_z    : num  473 469 469 473 478 470 474 478 472 478 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
dim(training1)
```

```
## [1] 13737    60
```

Remove the first seven columns of data. This information is about the person, time and other information that is not related to the movement. So remove these columns as they are not going to be used in the model.

```
training1 = training1[,-c(1:7)]
```

Check the data for variables that may have near zero variance.

```
nzv_data <- nearZeroVar(training1, saveMetrics = TRUE)
dim(nzv_data)
```

```
## [1] 53  4
```

```
nzv_data
```

```
##                      freqRatio percentUnique zeroVar   nzv
## roll_belt             1.080123    8.13132416   FALSE FALSE
## pitch_belt            1.006849   12.17150761   FALSE FALSE
## yaw_belt              1.022161   12.92130742   FALSE FALSE
```

```
## total_accel_belt      1.054462    0.19654946   FALSE FALSE
## gyros_belt_x          1.033024    0.96090850   FALSE FALSE
## gyros_belt_y          1.129367    0.48773386   FALSE FALSE
## gyros_belt_z          1.085784    1.18657640   FALSE FALSE
## accel_belt_x          1.061338    1.15017835   FALSE FALSE
## accel_belt_y          1.117151    0.99730654   FALSE FALSE
## accel_belt_z          1.103110    2.13292568   FALSE FALSE
## magnet_belt_x         1.069231    2.18388294   FALSE FALSE
## magnet_belt_y         1.123894    2.08196841   FALSE FALSE
## magnet_belt_z         1.024845    3.18118949   FALSE FALSE
## roll_arm             55.488372   17.76224794   FALSE FALSE
## pitch_arm            91.807692   20.13540074   FALSE FALSE
## yaw_arm              33.138889   19.29096600   FALSE FALSE
## total_accel_arm       1.032206    0.47317464   FALSE FALSE
## gyros_arm_x           1.008065    4.58615418   FALSE FALSE
## gyros_arm_y           1.470588    2.66433719   FALSE FALSE
## gyros_arm_z           1.203390    1.70342870   FALSE FALSE
## accel_arm_x           1.008547    5.57618112   FALSE FALSE
## accel_arm_y           1.163265    3.77083788   FALSE FALSE
## accel_arm_z           1.030612    5.58346073   FALSE FALSE
## magnet_arm_x          1.016129    9.55812768   FALSE FALSE
## magnet_arm_y          1.063492    6.23134600   FALSE FALSE
## magnet_arm_z          1.061728    9.13591032   FALSE FALSE
## roll_dumbbell         1.155556   87.00589648   FALSE FALSE
## pitch_dumbbell        2.048077   84.92392808   FALSE FALSE
## yaw_dumbbell          1.155556   86.36529082   FALSE FALSE
## total_accel_dumbbell  1.080559    0.31302322   FALSE FALSE
## gyros_dumbbell_x      1.004684    1.70342870   FALSE FALSE
## gyros_dumbbell_y      1.207229    1.94365582   FALSE FALSE
## gyros_dumbbell_z      1.061576    1.42680352   FALSE FALSE
## accel_dumbbell_x      1.030043    2.98464002   FALSE FALSE
## accel_dumbbell_y      1.023121    3.28310403   FALSE FALSE
## accel_dumbbell_z      1.126582    2.91912353   FALSE FALSE
## magnet_dumbbell_x     1.181818    7.84741938   FALSE FALSE
## magnet_dumbbell_y     1.333333    5.98383927   FALSE FALSE
## magnet_dumbbell_z     1.084034    4.78998326   FALSE FALSE
## roll_forearm         12.013100   13.59831113   FALSE FALSE
## pitch_forearm        62.477273   18.83235059   FALSE FALSE
## yaw_forearm          15.531073   12.88490937   FALSE FALSE
## total_accel_forearm   1.146991    0.50229308   FALSE FALSE
## gyros_forearm_x       1.016173    2.02373153   FALSE FALSE
## gyros_forearm_y       1.034483    5.24131907   FALSE FALSE
## gyros_forearm_z       1.154762    2.15476450   FALSE FALSE
## accel_forearm_x       1.111111    5.67081604   FALSE FALSE
## accel_forearm_y       1.029412    7.11945840   FALSE FALSE
## accel_forearm_z       1.009174    4.08386111   FALSE FALSE
## magnet_forearm_x      1.075472   10.50447696   FALSE FALSE
## magnet_forearm_y      1.000000   13.27800830   FALSE FALSE
## magnet_forearm_z      1.095238   11.71289219   FALSE FALSE
## classe                1.469526    0.03639805   FALSE FALSE
```

None of the remaining variables have near zero variance, so there is no need to remove variables for this reason. Our training dataset is now clean and ready to be used in model buliding.

Build model using RPART

Builiding a prediction model using recursive partitioning for classification algorithm (rpart).

```
modelRPART <- rpart(classe ~ ., data=training1, method="class")
```

Predict into the testing dataset and see how well we are classifying movements in the new dataset.

```
predictions <- predict(modelRPART, testing, type = "class")
confusionMatrix(predictions, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1498  196   69  106   25
##          B   42  669   85   86   92
##          C   43  136  739  129  131
##          D   33   85   98  553   44
##          E   58   53   35   90  790
##
## Overall Statistics
##
##                Accuracy : 0.722
##                  95% CI : (0.7104, 0.7334)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6467
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8949   0.5874   0.7203  0.57365   0.7301
## Specificity            0.9060   0.9357   0.9097  0.94717   0.9509
## Pos Pred Value         0.7909   0.6869   0.6273  0.68020   0.7700
## Neg Pred Value         0.9559   0.9043   0.9390  0.91897   0.9399
## Prevalence             0.2845   0.1935   0.1743  0.16381   0.1839
## Detection Rate         0.2545   0.1137   0.1256  0.09397   0.1342
## Detection Prevalence   0.3218   0.1655   0.2002  0.13815   0.1743
## Balanced Accuracy      0.9004   0.7615   0.8150  0.76041   0.8405
```

The out-sample accuracy of the model is 71%, making the out-sample error rate 29%, which is high, and could be lower. Let’s try out a different algorithm and see if the out-sample error rate can be reduced.

Build model using RandomForest

Second attempt will use the random forest algortihm to build the model.

```r
modelRF <- randomForest(classe ~. , data=training1)
```

Predict into the testing dataset and see how well we are classifying movements in the new dataset.

```r
predictionsRF <- predict(modelRF, testing, type = "class")
confusionMatrix(predictionsRF, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    9    0    0    0
##          B    1 1127   13    0    0
##          C    0    3 1011   14    0
##          D    0    0    2  949    5
##          E    0    0    0    1 1077
##
## Overall Statistics
##
##                Accuracy : 0.9918
##                  95% CI : (0.9892, 0.994)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9897
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9895   0.9854   0.9844   0.9954
## Specificity            0.9979   0.9971   0.9965   0.9986   0.9998
## Pos Pred Value         0.9946   0.9877   0.9835   0.9927   0.9991
## Neg Pred Value         0.9998   0.9975   0.9969   0.9970   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1915   0.1718   0.1613   0.1830
## Detection Prevalence   0.2858   0.1939   0.1747   0.1624   0.1832
## Balanced Accuracy      0.9986   0.9933   0.9909   0.9915   0.9976
```

The out-sample accuracy of the model is 99.52%, making the out-sample error rate 0.48%, which implies the sample is doing really well in classifying the data. This model is performing much better than the previous one. We will use this model for predicting into the validation set.

Validation Prediction

In the final step, the validation data and the randomforest model will be used to predict movements.

```r
predictions_final <- predict(modelRF, validation, type = "class")
```

The code below uploads the information to Coursera

```
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(predictions_final)
```