# Practical Machine Learning Project

*R. Jennings*

*January 30, 2016*

## Introduction

This project uses machine learning to classify human activity from data recorded about movements in weightlifting. The data is the HAR (Human Activity Recognition) dataset (reference: http://groupware.les.inf.puc-rio.br/har) and includes data from accelerometers on the belt, arm, forearm and dumbbell. The goal was to build a classifier that can predict the class for each data entry. The classes are defined as A, B, C, D, E.

## Preparing the Datasets

The training and test datasets were donwloaded into local files for processing. The test dataset is the dataset for evaluation. Before splittling the training dataset into training and testing for algorithm testing, the two datasets were cleaned and processed to leave the desired features.

Processing steps:

1. Read csv files
2. Make a copy for processing
3. Change all empty entries in cells of csv to NA for processing
4. Set a threshold of 90% for NAs. Only keep columns that have at least 10% values that are not NAs.
5. Drop other columns that don't have value (X, user_name, time-related, etc.) for prediction
6. This leaves cleaned datasets for training and test

```
suppressWarnings(suppressMessages(library(caret))) #to keep messages from html
pml_train <- read.csv("./pml-training.csv")
pml_test <- read.csv("./pml-testing.csv")
pml_train_cleaned <- pml_train
pml_test_cleaned <- pml_test
pml_train_cleaned[pml_train_cleaned==""] <- NA
pml_test_cleaned[pml_test_cleaned==""] <- NA
na_threshold <- .9
na_empty_max <- dim(pml_train_cleaned)[1] * na_threshold
goodcol <- !colSums(is.na(pml_train_cleaned)) > na_empty_max
pml_train_cleaned <- pml_train_cleaned[, goodcol]
pml_test_cleaned <- pml_test_cleaned[, goodcol]
drop_columns <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp", "
                  "num_window")
pml_train_cleaned <- pml_train_cleaned[,-which(names(pml_train_cleaned) %in% drop_columns)]
pml_test_cleaned <- pml_test_cleaned[,-which(names(pml_test_cleaned) %in% drop_columns)]
dim(pml_train_cleaned)
```

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

```
dim(pml_test_cleaned)
```

```
## [1] 20 53
```

```
summary(pml_train_cleaned)
```

```
##    roll_belt        pitch_belt          yaw_belt       total_accel_belt
##  Min.   :-28.90   Min.   :-55.8000   Min.   :-180.00   Min.   : 0.00
##  1st Qu.:  1.10   1st Qu.:  1.7600   1st Qu.: -88.30   1st Qu.: 3.00
##  Median :113.00   Median :  5.2800   Median : -13.00   Median :17.00
##  Mean   : 64.41   Mean   :  0.3053   Mean   : -11.21   Mean   :11.31
##  3rd Qu.:123.00   3rd Qu.: 14.9000   3rd Qu.:  12.90   3rd Qu.:18.00
##  Max.   :162.00   Max.   : 60.3000   Max.   : 179.00   Max.   :29.00
##   gyros_belt_x        gyros_belt_y       gyros_belt_z
##  Min.   :-1.040000   Min.   :-0.64000   Min.   :-1.4600
##  1st Qu.:-0.030000   1st Qu.: 0.00000   1st Qu.:-0.2000
##  Median : 0.030000   Median : 0.02000   Median :-0.1000
##  Mean   :-0.005592   Mean   : 0.03959   Mean   :-0.1305
##  3rd Qu.: 0.110000   3rd Qu.: 0.11000   3rd Qu.:-0.0200
##  Max.   : 2.220000   Max.   : 0.64000   Max.   : 1.6200
##   accel_belt_x       accel_belt_y     accel_belt_z     magnet_belt_x
##  Min.   :-120.000   Min.   :-69.00   Min.   :-275.00   Min.   :-52.0
##  1st Qu.: -21.000   1st Qu.:  3.00   1st Qu.:-162.00   1st Qu.:  9.0
##  Median : -15.000   Median : 35.00   Median :-152.00   Median : 35.0
##  Mean   :  -5.595   Mean   : 30.15   Mean   : -72.59   Mean   : 55.6
##  3rd Qu.:  -5.000   3rd Qu.: 61.00   3rd Qu.:  27.00   3rd Qu.: 59.0
##  Max.   :  85.000   Max.   :164.00   Max.   : 105.00   Max.   :485.0
##  magnet_belt_y   magnet_belt_z       roll_arm         pitch_arm
##  Min.   :354.0   Min.   :-623.0   Min.   :-180.00   Min.   :-88.800
##  1st Qu.:581.0   1st Qu.:-375.0   1st Qu.: -31.77   1st Qu.:-25.900
##  Median :601.0   Median :-320.0   Median :   0.00   Median :  0.000
##  Mean   :593.7   Mean   :-345.5   Mean   :  17.83   Mean   : -4.612
##  3rd Qu.:610.0   3rd Qu.:-306.0   3rd Qu.:  77.30   3rd Qu.: 11.200
##  Max.   :673.0   Max.   : 293.0   Max.   : 180.00   Max.   : 88.500
##     yaw_arm          total_accel_arm  gyros_arm_x        gyros_arm_y
##  Min.   :-180.0000   Min.   : 1.00   Min.   :-6.37000   Min.   :-3.4400
##  1st Qu.: -43.1000   1st Qu.:17.00   1st Qu.:-1.33000   1st Qu.:-0.8000
##  Median :   0.0000   Median :27.00   Median : 0.08000   Median :-0.2400
##  Mean   :  -0.6188   Mean   :25.51   Mean   : 0.04277   Mean   :-0.2571
##  3rd Qu.:  45.8750   3rd Qu.:33.00   3rd Qu.: 1.57000   3rd Qu.: 0.1400
##  Max.   : 180.0000   Max.   :66.00   Max.   : 4.87000   Max.   : 2.8400
##   gyros_arm_z       accel_arm_x       accel_arm_y      accel_arm_z
##  Min.   :-2.3300   Min.   :-404.00   Min.   :-318.0   Min.   :-636.00
##  1st Qu.:-0.0700   1st Qu.:-242.00   1st Qu.: -54.0   1st Qu.:-143.00
##  Median : 0.2300   Median : -44.00   Median :  14.0   Median : -47.00
##  Mean   : 0.2695   Mean   : -60.24   Mean   :  32.6   Mean   : -71.25
##  3rd Qu.: 0.7200   3rd Qu.:  84.00   3rd Qu.: 139.0   3rd Qu.:  23.00
##  Max.   : 3.0200   Max.   : 437.00   Max.   : 308.0   Max.   : 292.00
##   magnet_arm_x     magnet_arm_y     magnet_arm_z    roll_dumbbell
##  Min.   :-584.0   Min.   :-392.0   Min.   :-597.0   Min.   :-153.71
##  1st Qu.:-300.0   1st Qu.:  -9.0   1st Qu.: 131.2   1st Qu.: -18.49
##  Median : 289.0   Median : 202.0   Median : 444.0   Median :  48.17
```

```
##  Mean   : 191.7   Mean   : 156.6   Mean   : 306.5   Mean   :  23.84
##  3rd Qu.: 637.0   3rd Qu.: 323.0   3rd Qu.: 545.0   3rd Qu.:  67.61
##  Max.   : 782.0   Max.   : 583.0   Max.   : 694.0   Max.   : 153.55
##  pitch_dumbbell     yaw_dumbbell      total_accel_dumbbell
##  Min.   :-149.59   Min.   :-150.871   Min.   : 0.00
##  1st Qu.: -40.89   1st Qu.: -77.644   1st Qu.: 4.00
##  Median : -20.96   Median :  -3.324   Median :10.00
##  Mean   : -10.78   Mean   :   1.674   Mean   :13.72
##  3rd Qu.:  17.50   3rd Qu.:  79.643   3rd Qu.:19.00
##  Max.   : 149.40   Max.   : 154.952   Max.   :58.00
##  gyros_dumbbell_x    gyros_dumbbell_y   gyros_dumbbell_z
##  Min.   :-204.0000   Min.   :-2.10000   Min.   : -2.380
##  1st Qu.:  -0.0300   1st Qu.:-0.14000   1st Qu.: -0.310
##  Median :   0.1300   Median : 0.03000   Median : -0.130
##  Mean   :   0.1611   Mean   : 0.04606   Mean   : -0.129
##  3rd Qu.:   0.3500   3rd Qu.: 0.21000   3rd Qu.:  0.030
##  Max.   :   2.2200   Max.   :52.00000   Max.   :317.000
##  accel_dumbbell_x  accel_dumbbell_y  accel_dumbbell_z  magnet_dumbbell_x
##  Min.   :-419.00   Min.   :-189.00   Min.   :-334.00   Min.   :-643.0
##  1st Qu.: -50.00   1st Qu.:  -8.00   1st Qu.:-142.00   1st Qu.:-535.0
##  Median :  -8.00   Median :  41.50   Median :  -1.00   Median :-479.0
##  Mean   : -28.62   Mean   :  52.63   Mean   : -38.32   Mean   :-328.5
##  3rd Qu.:  11.00   3rd Qu.: 111.00   3rd Qu.:  38.00   3rd Qu.:-304.0
##  Max.   : 235.00   Max.   : 315.00   Max.   : 318.00   Max.   : 592.0
##  magnet_dumbbell_y magnet_dumbbell_z  roll_forearm       pitch_forearm
##  Min.   :-3600     Min.   :-262.00   Min.   :-180.0000   Min.   :-72.50
##  1st Qu.:  231     1st Qu.: -45.00   1st Qu.:  -0.7375   1st Qu.:  0.00
##  Median :  311     Median :  13.00   Median :  21.7000   Median :  9.24
##  Mean   :  221     Mean   :  46.05   Mean   :  33.8265   Mean   : 10.71
##  3rd Qu.:  390     3rd Qu.:  95.00   3rd Qu.: 140.0000   3rd Qu.: 28.40
##  Max.   :  633     Max.   : 452.00   Max.   : 180.0000   Max.   : 89.80
##   yaw_forearm      total_accel_forearm gyros_forearm_x
##  Min.   :-180.00   Min.   :  0.00      Min.   :-22.000
##  1st Qu.: -68.60   1st Qu.: 29.00      1st Qu.: -0.220
##  Median :   0.00   Median : 36.00      Median :  0.050
##  Mean   :  19.21   Mean   : 34.72      Mean   :  0.158
##  3rd Qu.: 110.00   3rd Qu.: 41.00      3rd Qu.:  0.560
##  Max.   : 180.00   Max.   :108.00      Max.   :  3.970
##  gyros_forearm_y     gyros_forearm_z    accel_forearm_x   accel_forearm_y
##  Min.   : -7.02000   Min.   : -8.0900   Min.   :-498.00   Min.   :-632.0
##  1st Qu.: -1.46000   1st Qu.: -0.1800   1st Qu.:-178.00   1st Qu.:  57.0
##  Median :  0.03000   Median :  0.0800   Median : -57.00   Median : 201.0
##  Mean   :  0.07517   Mean   :  0.1512   Mean   : -61.65   Mean   : 163.7
##  3rd Qu.:  1.62000   3rd Qu.:  0.4900   3rd Qu.:  76.00   3rd Qu.: 312.0
##  Max.   :311.00000   Max.   :231.0000   Max.   : 477.00   Max.   : 923.0
##  accel_forearm_z   magnet_forearm_x  magnet_forearm_y magnet_forearm_z
##  Min.   :-446.00   Min.   :-1280.0   Min.   :-896.0   Min.   :-973.0
##  1st Qu.:-182.00   1st Qu.: -616.0   1st Qu.:   2.0   1st Qu.: 191.0
##  Median : -39.00   Median : -378.0   Median : 591.0   Median : 511.0
##  Mean   : -55.29   Mean   : -312.6   Mean   : 380.1   Mean   : 393.6
##  3rd Qu.:  26.00   3rd Qu.:  -73.0   3rd Qu.: 737.0   3rd Qu.: 653.0
##  Max.   : 291.00   Max.   :  672.0   Max.   :1480.0   Max.   :1090.0
##  classe
##  A:5580
```

```
##  B:3797
##  C:3422
##  D:3216
##  E:3607
##
```

The dimensions of the two dataframes show there are 52 features remaining (52 features plus one predictor). Their summaries show that there are no NAs left that need to be addressed. For report brevity, only the results for pml_train_cleaned is included

## Split training dataset for training and testing models

The training dataset was split into 70% training and 30% testing for building and testing models.

```
set.seed(54321)
inTrain <- createDataPartition(pml_train_cleaned$classe, p=0.70, list=F)
training <- pml_train_cleaned[inTrain, ]
testing <- pml_train_cleaned[-inTrain, ]
```

## Cross Validation

Repeated cross-validation was used with 10 folds and 3 repetitions. Varying the number of folds and repetitions only produced slight changes in model accuracy. Verbosity was turned off to limit messages. These settings were used to build all models.

```
trainsettings <- trainControl(method="repeatedcv", number=10, repeats=3, verboseIter=FALSE)
```

## Build and test models

Numerous model types were built and compared, including random forest, naive bayes, decision tree, LDA, GBM, and SVM. In addition, bagged decision trees and random forest with changes in the number of features at each split were implemented. Model accuracy was used as the method to select the best model. Below is a summary of accuracies for each type on the test set.

Model (Accuracy):
Random Forest - default settings (.9949023)
Decision Tree (.4926083)
Naive Bayes (.7294817)
LDA (.697706)
GBM (.9624469)
svmRadial (.9235344)
Bagged Decision Tree (.9899745)
Random Forest Grid Selection by best mtry (8 was best) (.9964316)

## Out of Sample Errors

Based on the model accuracies from the confusion matrices, the out of sample error for each model is predicted to be:

Model (Predicted Out of Sample Error):
Random Forest - default settings (.0050977)
Decision Tree (.5073917)
Naive Bayes (.2705183)
LDA (.302294)
GBM (.0375531)
svmRadial (.0764656)
Bagged Decision Tree (.0100255)
Random Forest Grid Selection by best mtry (8 was best) (.0035684)

Since it very time consuming to run all models, this report includes runs for decision tree, random forest (default), and the best model, random forest using a grid to select by best mtry.

To speed up the runs, parallel processing was enabled as shown below.

```
suppressWarnings(suppressMessages(library(randomForest)))
#Set up parallel processes on all CPU cores
suppressWarnings(suppressMessages(library(doParallel)))
cl <- makeCluster(detectCores(), type='PSOCK')
registerDoParallel(cl)

#Base decision tree
suppressWarnings(suppressMessages(library(rpart)))
mod_tr_tree <- train(classe ~ ., data=training, method="rpart", trControl=trainsettings)
pred_tst_tree <- predict(mod_tr_tree, testing)
cm_tree <- confusionMatrix(pred_tst_tree, testing$classe)
cat("Decision Tree accuracy is: ", cm_tree$overall['Accuracy'])
```

```
## Decision Tree accuracy is:  0.4926083
```

```
#Default random forest
mod_tr_rf <- train(classe ~ ., data=training, method="rf", trControl=trainsettings)
pred_tst_rf <- predict(mod_tr_rf, testing)
cm_rf <- confusionMatrix(pred_tst_rf, testing$classe)
cat("Random Forest (default) accuracy is: ", cm_rf$overall['Accuracy'])
```

```
## Random Forest (default) accuracy is:  0.9949023
```

```
#Random Forest Controlling Number of Features at Each Split
#Total features is 52, default algorithm select sqrt(52)
#This control varies the number of features at each split = 4, 8, 16, 32, 40, 52
#Will take a long time to run
grid_rf <- expand.grid(.mtry = c(4, 8, 16, 32, 40, 52))
mod_tr_rf_grid <- train(classe ~ ., data=training, method = "rf", trControl=trainsettings, tuneGrid = gr
pred_tst_rf_grid <- predict(mod_tr_rf_grid, testing)
cm_rf_grid <- confusionMatrix(pred_tst_rf_grid, testing$classe)
cat("RF with Grid accuracy is: ", cm_rf_grid$overall['Accuracy'])
```

```
## RF with Grid accuracy is:  0.9964316
```

## Get predictions for 20 test cases

Using the three models built above, the predictions for the quiz test cases are as follows.

```
pred_quiz_test_set_tree <- predict(mod_tr_tree, pml_test_cleaned)
pred_quiz_test_set_tree_pr <- paste0(pred_quiz_test_set_tree, collapse = " ")
print(paste0("Predictions for test set - base decision tree: ", pred_quiz_test_set_tree_pr, collapse =
```

```
## [1] "Predictions for test set - base decision tree: C A C A A C C A A A C C C A C A A A A C"
```

```
pred_quiz_test_set_rf <- predict(mod_tr_rf, pml_test_cleaned)
pred_quiz_test_set_rf_pr <- paste0(pred_quiz_test_set_rf, collapse = " ")
print(paste0("Predictions for test set - base random forest: ", pred_quiz_test_set_rf_pr))
```

```
## [1] "Predictions for test set - base random forest: B A B A A E D B A A B C B A E E A B B B"
```

```
pred_quiz_test_set_rf_grid <- predict(mod_tr_rf_grid, pml_test_cleaned)
pred_quiz_test_set_rf_grid_pr <- paste0(pred_quiz_test_set_rf_grid, collapse = " ")
print(paste0("Predictions for test set - random forest with grid for mtry: ", pred_quiz_test_set_rf_gri
```

```
## [1] "Predictions for test set - random forest with grid for mtry: B A B A A E D B A A B C B A E E A
```

Since random forest with grid was most accurate, these predictions were used for the quiz and were 100% accurate. The predictions for base random forest were the same however.