Machine Learning Project - Cousera Week 4

Ridzuan Mohamad

5/11/2017

1. Overview

The objectives of this project is to use the data from accelerometers on the belt, forearm, arm, and dumbbell obtain from 6 participants ("Pedro", "Jeremy", "Adelmo", "Eurico", "Carlitos" and "Charles") and to "predict" the manner in which they did the exercise. The 6 participants were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The outcome of the exercises performed by the 6 participants, was recorded into 2 types of data sets, namely, "training" and "test" ("pml-training.csv" and "pml-testing.csv") from the following URL link source:

1. Training Data: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
2. Test Data: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

2. Data Exploratory Analysis

2.1 Data Loading

The training dan test dataset is provided by the data provider and we going to download both dataset from the URL link described above.

```
# set the URL for the download
url_train_data <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
url_test_test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

file_training_set = file.path("dataset","pml-training.csv")
file_testing_set = file.path("dataset","pml-testing.csv")

# download the datasets
if (!file.exists(file_training_set) || !file.exists(file_testing_set)){
    if(!dir.exists(file.path("dataset"))){
        dir.create("dataset")
    }

    download.file(url_train_data, file_training_set, method="curl")
    download.file(url_test_test, file_testing_set, method="curl")
}

# load into data.frame
train.dataset.raw <- read.csv(file_training_set, na.strings = c("NA", ""))
test.dataset.raw <- read.csv(file_testing_set, na.strings = c("NA", ""))
```

2.2 Exploring and Cleansing

The downloaded dataset have the same variable which is 160 and difference numbers of observation data. The training dataset have 19622 observations data and the test dataset 20 have observations data. It is normal practices to have training dataset is larger than the test dataset.

There are (160) variable (predictors) but we only need a few which will contribute to the accuracy of our prediction model later on. Refering to ***Appendix A - Dataset Structure for Raw Dataset***, we found out there are a lot of variable contain only **NA**, **#DIV/0!** which is should be remove from the dataset. There are 7 variables is used for data identificatoin such as `X`, `user_name`, `raw_timestamp_part_1`, `raw_timestamp_part_2`, `cvtd_timestamp`, `new_window` and `num_window` need to be remove as well.

```
##Remove NA column & column 1-7 as identifier column
train.dataset.clean <- train.dataset.raw[, colSums(is.na(train.dataset.raw)) == 0]
train.dataset.clean <- train.dataset.clean[, -c(1:7)]

##Remove NA column & column 1-7 as identifier column
test.dataset.clean <- test.dataset.raw[, colSums(is.na(test.dataset.raw)) == 0]
test.dataset.clean <- test.dataset.clean[, -c(1:7)]
```

Now, we only have 53 variable for both datasets and store it in the new data frame known as `train.dataset.clean` & `test.dataset.clean`. Detail dataset structure please refer to ***Appendix B - List of Variable for Clean Dataset***

Exploring the classes data it shows that the *classe A* have more records compare to other classes. Based on the ***Appendix C - Histrogram*** and the table below; *classe A* have more than 1000 records compared to other classes.

```
## display records into table by classe
table(train.dataset.clean$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

3. Building Prediction Model

The clean training dataset (*train.dataset.clean*) is split into two (2) partition :-

1. Training Set (*trainset*) - 70%
2. Cross Validation Set (*crossvalidationset*) - 30%

The Training set mention above will be fit into **Random Forest Algorithm** to create a prediction model which will be used to make a prediction. The prediction model information can be refered on ***Appendix D - Prediction Model Summary*** The other sub-training set (*Cross Validation*) will be used to evaluate the prediction model accuracy.

3.1 Data Partioning

```
inTrain <- createDataPartition(y=train.dataset.clean$classe, p=0.7, list = FALSE)
trainset <- train.dataset.clean[inTrain,]
crossvalidationset <- train.dataset.clean[-inTrain,]
```

3.2 Random Forest

```
set.seed(1235)

## create proediction model using Random Forest Algorithm
model.fit <- randomForest(classe ~ ., data=trainset, method="class", importance=TRUE,
                          proximity=TRUE, ntree=30)
```

```
rf.predict <- predict(model.fit, crossvalidationset)

## evaluate prediction model
confusion.matrix <- confusionMatrix(rf.predict, crossvalidationset$classe)
```

The cross validation check shows that the accuracy is 99%. The detail can be refered at ***Appendix E - Confusion Matrix Output***.

4. Applying Model to Test Data

Fit the clean test dataset into the prediction model and print the output.

```
final.predict <- predict(model.fit, test.dataset.clean)
final.predict
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

References

Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.

Appendixies

Appendix A - Dataset Structure for Raw Dataset

```
str(train.dataset.raw)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 1323084231 1323084232 1323084232 1323084232 1
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 304277 368296 440390 484323 484
##  $ 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 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
##  $ 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 ...
##  $ kurtosis_roll_belt      : Factor w/ 396 levels "-0.016850","-0.021024",..: NA NA NA NA NA NA NA NA
##  $ kurtosis_picth_belt     : Factor w/ 316 levels "-0.021887","-0.060755",..: NA NA NA NA NA NA NA NA
##  $ kurtosis_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : Factor w/ 394 levels "-0.003095","-0.010002",..: NA NA NA NA NA NA NA NA
##  $ skewness_roll_belt.1    : Factor w/ 337 levels "-0.005928","-0.005960",..: NA NA NA NA NA NA NA NA
##  $ skewness_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : Factor w/ 3 levels "#DIV/0!","0.00",..: NA NA NA NA NA NA NA NA NA NA .
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x            : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...
##  $ gyros_belt_y            : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z            : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x            : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y            : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z            : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x           : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
```

```
##  $ magnet_belt_y           : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z           : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
##  $ roll_arm                : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm               : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...
##  $ 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 ...
##  $ var_accel_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x             : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y             : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z             : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x             : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y             : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z             : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x            : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y            : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z            : int  516 513 513 512 506 513 509 510 518 516 ...
##  $ kurtosis_roll_arm       : Factor w/ 329 levels "-0.02438","-0.04190",..: NA NA NA NA NA NA NA NA N
##  $ kurtosis_picth_arm      : Factor w/ 327 levels "-0.00484","-0.01311",..: NA NA NA NA NA NA NA NA N
##  $ kurtosis_yaw_arm        : Factor w/ 394 levels "-0.01548","-0.01749",..: NA NA NA NA NA NA NA NA N
##  $ skewness_roll_arm       : Factor w/ 330 levels "-0.00051","-0.00696",..: NA NA NA NA NA NA NA NA N
##  $ skewness_pitch_arm      : Factor w/ 327 levels "-0.00184","-0.01185",..: NA NA NA NA NA NA NA NA N
##  $ skewness_yaw_arm        : Factor w/ 394 levels "-0.00311","-0.00562",..: NA NA NA NA NA NA NA NA N
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : Factor w/ 397 levels "-0.0035","-0.0073",..: NA NA NA NA NA NA NA NA NA
##  $ kurtosis_picth_dumbbell : Factor w/ 400 levels "-0.0163","-0.0233",..: NA NA NA NA NA NA NA NA NA
##  $ kurtosis_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : Factor w/ 400 levels "-0.0082","-0.0096",..: NA NA NA NA NA NA NA NA NA
##  $ skewness_pitch_dumbbell : Factor w/ 401 levels "-0.0053","-0.0084",..: NA NA NA NA NA NA NA NA NA
##  $ skewness_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ min_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

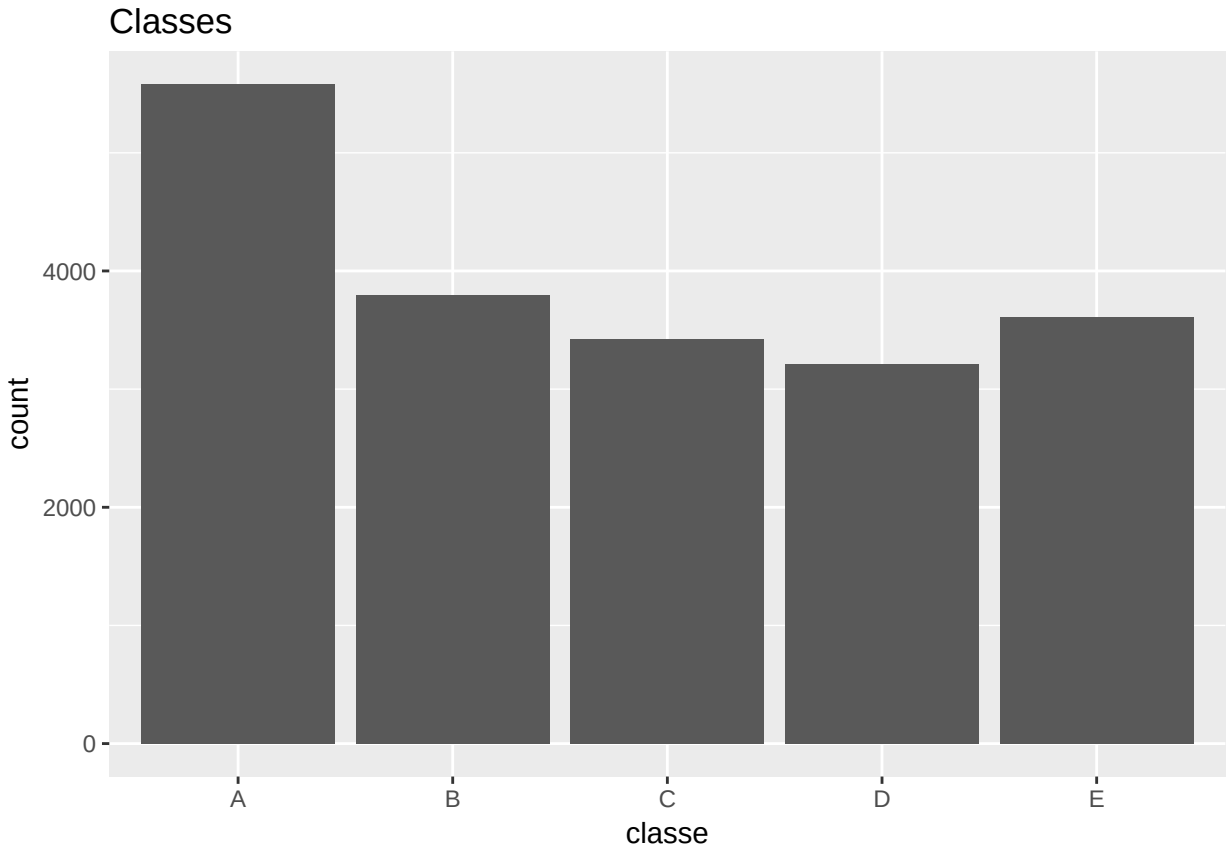
Appendix B - List of Variable for Clean Dataset

```
names(train.dataset.clean)
```

```
##  [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"
```

Appendix C - Histogram

```
ggplot(data = train.dataset.clean, aes(x=classe)) +
    geom_histogram(stat = "count") +
    scale_x_discrete(name = "classe") +
    scale_y_continuous(name = "count") +
    ggtitle("Classes")
```



Appendix D - Prediction Model Summary

```
model.fit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainset, method = "class",      importance = TRUE, proxi
##                Type of random forest: classification
##                      Number of trees: 30
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 1.21%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3893    8    2    2    1 0.003328213
## B   29 2606   17    2    4 0.019563582
## C    0   28 2356   10    2 0.016694491
## D    1    0   37 2209    5 0.019094139
## E    1    4    4    9 2507 0.007128713
```

Appendix E - Confusion Matrix Output

```
confusion.matrix
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    5    0    0    0
##          B    0 1131   14    0    0
##          C    0    3 1012   14    0
##          D    1    0    0  949    1
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.9934
##                  95% CI : (0.991, 0.9953)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9916
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9930   0.9864   0.9844   0.9991
## Specificity            0.9988   0.9971   0.9965   0.9996   0.9998
## Pos Pred Value         0.9970   0.9878   0.9835   0.9979   0.9991
## Neg Pred Value         0.9998   0.9983   0.9971   0.9970   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1922   0.1720   0.1613   0.1837
## Detection Prevalence   0.2851   0.1946   0.1749   0.1616   0.1839
## Balanced Accuracy      0.9991   0.9950   0.9914   0.9920   0.9994
```