Practical Machine Learning - Week 4 Peer Assignment

Evezmher Ovwvtr er

<3563645<

Analysis of Weight lifting exercises and classification into classes

This Report is to publish the analysis and results of prediction of weight lifting exercises into one of five classes - A,B,C,D and E based on the sensor measurements from the fitness devices. There are 19622 observations in the training set with 160 predictors. The test set contains 20 observations with 160 predictors. Objective is to correctly classify each of the test observations into A,B,C,D or E classes.

A quick look at the training set shows that there are lots of columns which have either blanks, NA or #DIV/0! values. While reading the data into R, we will mark all of these values as NA so that we can decide to discard or impute NA values in later stages. The training and test sets have been read into R directly from the links provided in the assignment page.

```
pml.training1 <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-tr
aining.csv",na.strings = c("#DIV/0!","","NA"))
pml.testing <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-test
ing.csv",na.strings = c("#DIV/0!","","NA"))
```

Now check the columns which had blanks, NA or #DIV/0! values, these should have only NA substituted values.

Check how many columns have NA values and how many NA values in each of these columns. This lets us know the requirement for Data Imputation.

```
ind <- sapply(pml.training1,function(x) sum(is.na(x)))
ind
```

```
##                        X                user_name     raw_timestamp_part_1
##                        0                        0                        0
##     raw_timestamp_part_2          cvtd_timestamp               new_window
##                        0                        0                        0
##               num_window              roll_belt             pitch_belt
##                        0                        0                        0
##                yaw_belt        total_accel_belt       kurtosis_roll_belt
##                        0                        0                    19226
##      kurtosis_picth_belt       kurtosis_yaw_belt       skewness_roll_belt
##                    19248                    19622                    19225
##     skewness_roll_belt.1       skewness_yaw_belt            max_roll_belt
##                    19248                    19622                    19216
##           max_picth_belt             max_yaw_belt            min_roll_belt
##                    19216                    19226                    19216
##           min_pitch_belt             min_yaw_belt       amplitude_roll_belt
##                    19216                    19226                    19216
##     amplitude_pitch_belt       amplitude_yaw_belt      var_total_accel_belt
##                    19216                    19226                    19216
##            avg_roll_belt         stddev_roll_belt            var_roll_belt
##                    19216                    19216                    19216
##           avg_pitch_belt         stddev_pitch_belt           var_pitch_belt
##                    19216                    19216                    19216
##             avg_yaw_belt           stddev_yaw_belt             var_yaw_belt
##                    19216                    19216                    19216
##             gyros_belt_x             gyros_belt_y             gyros_belt_z
##                        0                        0                        0
##             accel_belt_x             accel_belt_y             accel_belt_z
##                        0                        0                        0
##            magnet_belt_x            magnet_belt_y            magnet_belt_z
##                        0                        0                        0
##                roll_arm                 pitch_arm                   yaw_arm
##                        0                        0                        0
##           total_accel_arm            var_accel_arm            avg_roll_arm
##                        0                    19216                    19216
##           stddev_roll_arm            var_roll_arm            avg_pitch_arm
##                    19216                    19216                    19216
##          stddev_pitch_arm           var_pitch_arm             avg_yaw_arm
##                    19216                    19216                    19216
##           stddev_yaw_arm             var_yaw_arm             gyros_arm_x
##                    19216                    19216                        0
##              gyros_arm_y              gyros_arm_z             accel_arm_x
##                        0                        0                        0
##              accel_arm_y              accel_arm_z             magnet_arm_x
##                        0                        0                        0
##             magnet_arm_y             magnet_arm_z        kurtosis_roll_arm
##                        0                        0                    19294
##        kurtosis_picth_arm         kurtosis_yaw_arm        skewness_roll_arm
##                    19296                    19227                    19293
##        skewness_pitch_arm         skewness_yaw_arm             max_roll_arm
##                    19296                    19227                    19216
##             max_picth_arm              max_yaw_arm             min_roll_arm
##                    19216                    19216                    19216
##             min_pitch_arm              min_yaw_arm        amplitude_roll_arm
##                    19216                    19216                    19216
##       amplitude_pitch_arm        amplitude_yaw_arm             roll_dumbbell
##                    19216                    19216                        0
##            pitch_dumbbell             yaw_dumbbell    kurtosis_roll_dumbbell
```

```
##                         0                         0                     19221
##  kurtosis_picth_dumbbell    kurtosis_yaw_dumbbell   skewness_roll_dumbbell
##                     19218                     19622                     19220
##  skewness_pitch_dumbbell    skewness_yaw_dumbbell        max_roll_dumbbell
##                     19217                     19622                     19216
##       max_picth_dumbbell         max_yaw_dumbbell        min_roll_dumbbell
##                     19216                     19221                     19216
##       min_pitch_dumbbell         min_yaw_dumbbell  amplitude_roll_dumbbell
##                     19216                     19221                     19216
## amplitude_pitch_dumbbell   amplitude_yaw_dumbbell     total_accel_dumbbell
##                     19216                     19221                         0
##       var_accel_dumbbell         avg_roll_dumbbell     stddev_roll_dumbbell
##                     19216                     19216                     19216
##        var_roll_dumbbell        avg_pitch_dumbbell    stddev_pitch_dumbbell
##                     19216                     19216                     19216
##       var_pitch_dumbbell         avg_yaw_dumbbell      stddev_yaw_dumbbell
##                     19216                     19216                     19216
##         var_yaw_dumbbell         gyros_dumbbell_x         gyros_dumbbell_y
##                     19216                         0                         0
##         gyros_dumbbell_z         accel_dumbbell_x         accel_dumbbell_y
##                         0                         0                         0
##         accel_dumbbell_z        magnet_dumbbell_x        magnet_dumbbell_y
##                         0                         0                         0
##        magnet_dumbbell_z             roll_forearm            pitch_forearm
##                         0                         0                         0
##              yaw_forearm    kurtosis_roll_forearm   kurtosis_picth_forearm
##                         0                     19300                     19301
##     kurtosis_yaw_forearm    skewness_roll_forearm   skewness_pitch_forearm
##                     19622                     19299                     19301
##     skewness_yaw_forearm         max_roll_forearm        max_picth_forearm
##                     19622                     19216                     19216
##          max_yaw_forearm         min_roll_forearm        min_pitch_forearm
##                     19300                     19216                     19216
##          min_yaw_forearm   amplitude_roll_forearm  amplitude_pitch_forearm
##                     19300                     19216                     19216
##    amplitude_yaw_forearm      total_accel_forearm        var_accel_forearm
##                     19300                         0                     19216
##         avg_roll_forearm      stddev_roll_forearm         var_roll_forearm
##                     19216                     19216                     19216
##        avg_pitch_forearm     stddev_pitch_forearm        var_pitch_forearm
##                     19216                     19216                     19216
##          avg_yaw_forearm       stddev_yaw_forearm          var_yaw_forearm
##                     19216                     19216                     19216
##          gyros_forearm_x         gyros_forearm_y          gyros_forearm_z
##                         0                         0                         0
##          accel_forearm_x         accel_forearm_y          accel_forearm_z
##                         0                         0                         0
##         magnet_forearm_x        magnet_forearm_y         magnet_forearm_z
##                         0                         0                         0
##                   classe
##                         0
```

```
indNA <- ind[ind>19000]
length(which(ind>19000))
```

```
## [1] 100
```

We see that there are about 100 predictors which have NA values greater than 19000 with total dataset having 19622 observations. Since the % of missing values is really high, it does not make sense to perform data imputation to populate these NA values. It is probably better to remove these predictors altogether from the training set. Check the structure of the reduced training set.

```
pml.training2 <- pml.training1[-which(ind>19000)]
str(pml.training2)
```

```
## 'data.frame':    19622 obs. of  60 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 13230842
32 1323084232 1323084232 1323084232 1323084232 1323084232 ...
##  $ raw_timestamp_part_2: int  788290 808298 820366 120339 196328 304277 368296 440
390 484323 484434 ...
##  $ 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 ...
##  $ 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 -0.0
2 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ gyros_dumbbell_y     : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.0
2 -0.02 ...
##  $ gyros_dumbbell_z     : num  0 0 0 -0.02 0 0 0 0 0 0 ...
```

```
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -234 -232 -234 -232 -235
...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270
...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558
...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.
8 -63.8 ...
##  $ yaw_forearm         : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -152
...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.03 0.02
...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215
...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 478 470 474 476 473 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1
1 ...
```

```
pml2class <- sapply(pml.training2,class)
table(pml2class)
```

```
## pml2class
##  factor integer numeric
##       4      29      27
```

We can see that the first 4 columns have no significance for our prediction. Hence we can remove them from the training set further. This leaves us with only 56 predictors from the original 160.

We can further check for correlated predictors and remove them from the training set in order to reduce the predictor size further. For this we need to remove the 4 factor predictors in the training set.

```
pml.trainingnum <- pml.training2[pml2class != "factor"]
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

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

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

```
nearZeroVar(pml.trainingnum)
```

```
## integer(0)
```

```
pml.trainingnumfilt <- pml.trainingnum[-c(1:4)]
high.cor.pml1 <- findCorrelation(cor(pml.trainingnumfilt),cutoff = 0.75)
table(high.cor.pml1)
```

```
## high.cor.pml1
##  1  2  4  8  9 10 13 18 21 22 23 25 31 33 34 35 36 37 38 45 48
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

We see that none of the 56 predictors have near zero variance. so no further reduction from this method. We also check for correlated predictors with correlation > 0.75. The indices for high correlated predictors is stored to be used later to filter the training dataset further. We observe that there are 21 correlated predictors to be removed.

```
pml.training2filt <- pml.training2[-c(1:4)]
pml.training.finalset <- pml.training2filt[-high.cor.pml1]
dim(pml.training.finalset)
```

```
## [1] 19622    35
```

Now we have the final training set which has only 35 predictors along with 19622 samples. We are now ready to run a random forest algorithm. We are using this as this method is resistant to data skews without any need for preprocessing such as centering, scaling or other transformations.the subset of random predictors to be taken is roughly square root (p) for classification problems. hence we use a value of 6. The default number of trees is 500. We simulate with 25,50 and 100 to get some quick results to ascertain the accuracy levels. Finally we settle for 100 trees as this gives good enough and fast prediction. We now review the final training set prediction accuracy.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
pml.rf.fit <- randomForest(classe~.,pml.training.finalset,mtry=6,ntree=100,importance
=TRUE)
pml.rf.fit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = pml.training.finalset,      mtry = 6, n
tree = 100, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.2%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5578    1    0    0    1 0.0003584229
## B    5 3791    1    0    0 0.0015801949
## C    0   10 3410    2    0 0.0035067212
## D    0    0   13 3202    1 0.0043532338
## E    0    0    1    4 3602 0.0013861935
```

The out of bag error rate is only 0.2% as show below.This seems to be a pretty good model. We can now predict the testing set values with this model.

Now we prepare the test set to have same predictors as training set and predict teh test set values.

```
pml.testing2 <- pml.testing[-which(ind>19000)]
pml.testing2filt <- pml.testing2[-c(1:4)]
pml.testing.finalset <- pml.testing2filt[-high.cor.pml1]
pml.test.pred <- predict(pml.rf.fit,pml.testing.finalset[-35])
pml.test.pred
```

```
##  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
```

We can see the prediction for the 20 test observations. The same were submitted in the course prediction quiz with 100% score further confirming the robustness of teh fitted random forest model.