C8Proj1 Machine Learning

CLAM0905

January 14, 2020

This assignment is designed to perform prediction analysis on a large dataset, in order to predict the class of 20 test cases. Details can be found on the website, included below, and can be summed up by the following. Six participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D), and throwing the hips to the front (Class E). My algorithm uses the "KNN: K nearest neighbor" approach to classify the existing 19,000+ cases into their appropriate class, where I will validate the efficiveness by comparing the predicted class to the actual class. From there, I will use my algorithm to classify the test cases into their predicted class, verifying the accuracy using the follow up quiz.

HAR website with background and more details:
http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har

Data exploration begins below

Read in training data:
```
Train <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
Trainset <- Train
Trainsetsub <- subset(Train, select = -c(X,user_name,raw_timestamp_part_1,raw_timestamp_p
art_2,cvtd_timestamp,new_window,num_window))
```

Read in test data:
```
Test <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
Testset <- Test
Testsetsub <- subset(Testset, select = -c(X,user_name,raw_timestamp_part_1,raw_timestamp_
part_2,cvtd_timestamp,new_window,num_window))
```

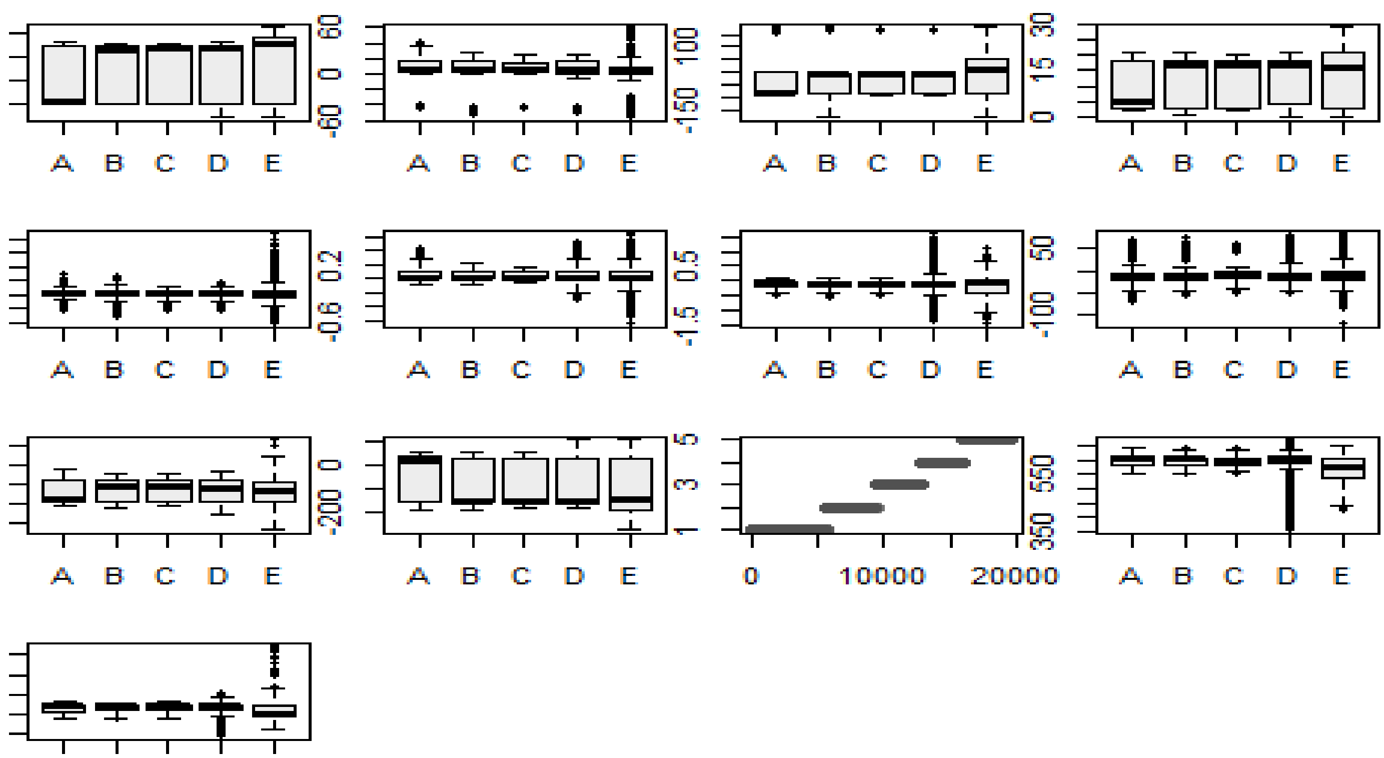
Summary of different classes in training data:
```
summary(Train$classe)

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

Preprocessing and exploratory data analysis:
```
Trainbelt <- subset(Trainsetsub[,grepl("belt", names(Trainsetsub))])
Trainbelt <- cbind(Trainbelt, userID = Trainsetsub$classe)
#remove columns with little to no data
Trainbelt <- Trainbelt[,c(1:4,30:39)]

Trainarm <- subset(Trainsetsub[,grepl("_arm", names(Trainsetsub))])
Trainarm <- cbind(Trainarm, userID = Trainsetsub$classe)
#remove columns with little to no data
Trainarm <- Trainarm[,c(1:4,15:23,39)]

Traindumbell <- subset(Trainsetsub[,grepl("dumbbell", names(Trainsetsub))])
Traindumbell <- cbind(Traindumbell, userID = Trainsetsub$classe)
```

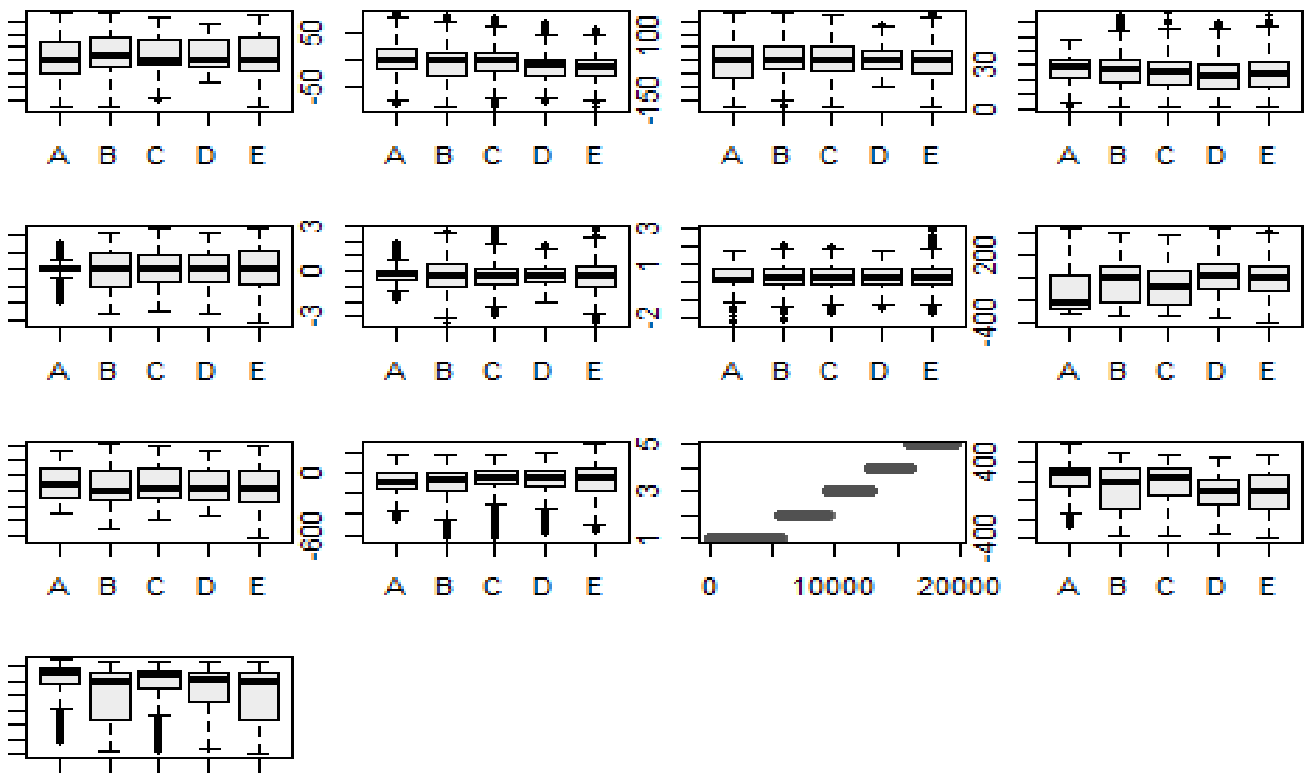
```
#remove columns with little to no data
Traindumbell <- Traindumbell[,c(1:3,19,30:39)]

Trainforearm <- subset(Trainsetsub[,grepl("forearm", names(Trainsetsub))])
Trainforearm <- cbind(Trainforearm, userID = Trainsetsub$classe)
#remove columns with little to no data
Trainforearm <- Trainforearm[,c(1:3,19,30:39)]
```

Exploratory data analysis: The below graphs break up activity up by classe for the 4 different categories: trainbelt, trainarm, traindumbbell, trainforearm. This gives us an idea of what variables are correlated with the different classes. The graphs highlight the class breakdown by the following exercises, in order: roll, pitch, yaw, total accel, gyros belt x, gyros belt y, gyros belt z, accel belt x, accell belt y, accell belt z, magnet belt x, magnet belt y, and magnet belt z.

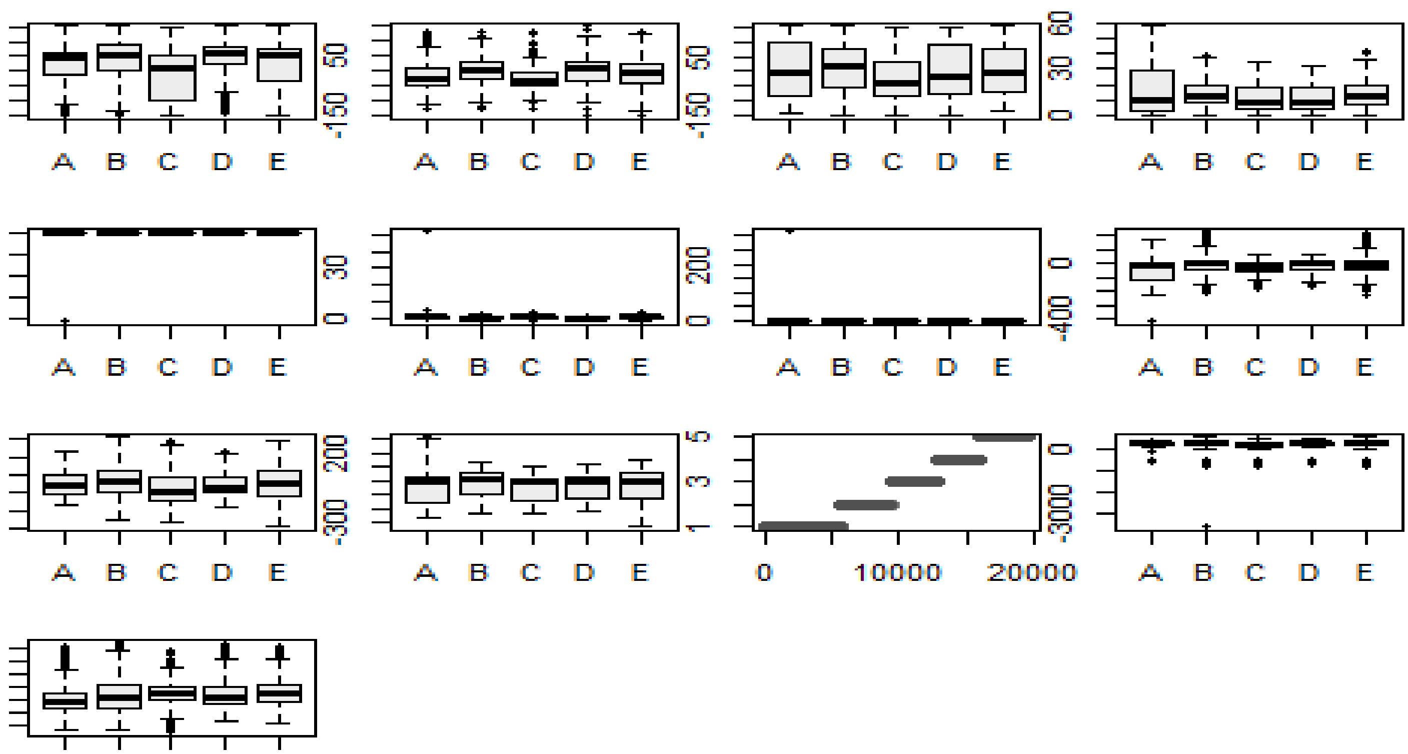
Train belt:



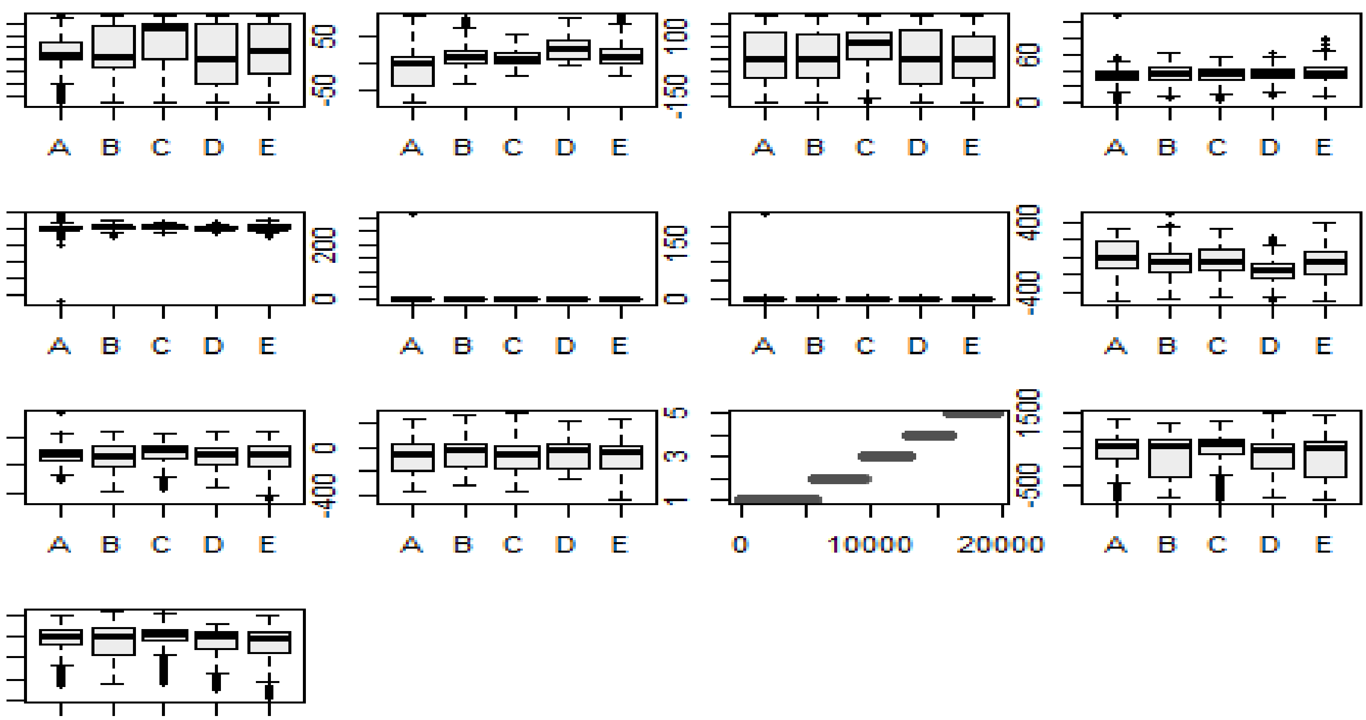
Train arm:



Train dumbbell:



Train forearm:



Prediction begins: We can now begin formatting and setting up the data for prediction analysis.

Begin cleanup for prediction

```
Trainsetsub1 <- Trainsetsub[,c(1:4,30:42,53:61,77:79,95,106:117,133,144:152,153)]

Trainsetsub1$ID <- seq.int(nrow(Trainsetsub1))

View(colnames(Trainsetsub1))
```

Generate a random number that is 90% of the total number of rows in dataset

```
set.seed(100)
subset <- sample(1:nrow(Trainsetsub1), 0.9 * nrow(Trainsetsub1))
```

Create normalization function

```
normalize <-function(x) {(x -min(x))/(max(x)-min(x))}
```

Run nomalization on first 52 columns of dataset because they are the predictors, note this drops classe column and shows the normalized values between 0-1

```
Training1<- as.data.frame(lapply(Trainsetsub1[,c(1:52)], normalize))
summary(Training1[1:5])

##    roll_belt        pitch_belt        yaw_belt      total_accel_belt
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1572   1st Qu.:0.4958   1st Qu.:0.2554   1st Qu.:0.1034
##  Median :0.7433   Median :0.5261   Median :0.4652   Median :0.5862
##  Mean   :0.4888   Mean   :0.4832   Mean   :0.4702   Mean   :0.3901
##  3rd Qu.:0.7957   3rd Qu.:0.6090   3rd Qu.:0.5373   3rd Qu.:0.6207
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##   gyros_belt_x
##  Min.   :0.0000
```

```
##  1st Qu.:0.3098
##  Median :0.3282
##  Mean   :0.3173
##  3rd Qu.:0.3528
##  Max.   :1.0000
```

Add classe back in

```
Training1$classe <- Trainsetsub1$classe
```

Extract training and test set

```
Training1_train <- Training1[subset,]
Training1_test <- Training1[-subset,]
```

Extract 53rd column of train dataset because it will be used as 'class' argument in knn function.

```
Training1_target_category <- Training1_train[c(53)]
Training1_train <- Training1_train[-c(53)]
```

Extract 53rd column of test dataset to measure the accuracy

```
Training1_test_category <- Training1_test[c(53)]
Training1_test<- Training1_test[-c(53)]

install.packages("class")
library(class)
install.packages("gmodels")
library(gmodels)
```

Format the datasets to data frames

```
Training1_train <- as.data.frame(Training1_train)
Training1_test <- as.data.frame(Training1_test)
Training1_target_category <- as.data.frame(Training1_target_category)
```

Run knn function using train, test and category datasets

```
knnpred <- knn(Training1_train,Training1_test,cl=Training1_target_category$classe,k=3)
```

Create a confusion matrix

```
confm <- table(knnpred,Training1_test_category$classe)
confm

## 
## knnpred   A   B   C   D   E
##       A 543   4   0   0   0
##       B   2 386   1   1   1
##       C   0   6 330   7   0
##       D   1   3   0 294   3
##       E   0   1   1   0 379
```

This function divides the correct predictions by total number of predictions to tell us how accurate the model is

```
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(confm)

## [1] 98.42078
```

Validation Table

```
crosstable <- CrossTable(x = knnpred, y = Training1_test_category$classe, prop.chisq = FA
LSE)
```

```
## Total Observations in Table:  1963
##
##              | Training1_test_category$classe
##      knnpred |         A |         B |         C |         D |         E | Row Total |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            A |       543 |         4 |         0 |         0 |         0 |       547 |
##              |     0.993 |     0.007 |     0.000 |     0.000 |     0.000 |     0.279 |
##              |     0.995 |     0.010 |     0.000 |     0.000 |     0.000 |           |
##              |     0.277 |     0.002 |     0.000 |     0.000 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            B |         2 |       386 |         1 |         1 |         1 |       391 |
##              |     0.005 |     0.987 |     0.003 |     0.003 |     0.003 |     0.199 |
##              |     0.004 |     0.965 |     0.003 |     0.003 |     0.003 |           |
##              |     0.001 |     0.197 |     0.001 |     0.001 |     0.001 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            C |         0 |         6 |       330 |         7 |         0 |       343 |
##              |     0.000 |     0.017 |     0.962 |     0.020 |     0.000 |     0.175 |
##              |     0.000 |     0.015 |     0.994 |     0.023 |     0.000 |           |
##              |     0.000 |     0.003 |     0.168 |     0.004 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            D |         1 |         3 |         0 |       294 |         3 |       301 |
##              |     0.003 |     0.010 |     0.000 |     0.977 |     0.010 |     0.153 |
##              |     0.002 |     0.007 |     0.000 |     0.974 |     0.008 |           |
##              |     0.001 |     0.002 |     0.000 |     0.150 |     0.002 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            E |         0 |         1 |         1 |         0 |       379 |       381 |
##              |     0.000 |     0.003 |     0.003 |     0.000 |     0.995 |     0.194 |
##              |     0.000 |     0.002 |     0.003 |     0.000 |     0.990 |           |
##              |     0.000 |     0.001 |     0.001 |     0.000 |     0.193 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
## Column Total |       546 |       400 |       332 |       302 |       383 |      1963 |
##              |     0.278 |     0.204 |     0.169 |     0.154 |     0.195 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
```

Predicting test set data into appropriate classe, output suppressed to honor course code

```
#Subset same columns as test set
Testset1 <- Testsetsub[,c(1:4,30:42,53:61,77:79,95,106:117,133,144:152,153)]
#normalize
Testset12<- as.data.frame(lapply(Testset1[,c(1:52)], normalize))
#predict
knntest <- knn(Training1_train,Testset12,cl=Training1_target_category$classe,k=3)
```

Conclusion: My knn algorithm was able to predict the class of the 20 test cases with 75% accuracy. The accuracy on the train dataset was 98%. This leads me to the conclusion that my knn algorithm was accurate, but overfit the train model and was not able to accurately predict on the test dataset to the same accuracy. More work needs to be done on my knn algorithm to be more accurate on test datasets, but it is a good start to learning and practicing machine learning techniques. Thank you for reading!