Project Machine Learning

Diego Valdes

March 20, 2019

```
# 005_project_ml.R
# @version: 1
# @author:  Diego Valdes
# @date:  Feb 14, 2019
# IST 687
# Clean openpowerlifting.csv - https://www.kaggle.com/open-powerlifting/powerlifting-database

library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.2
```

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

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

```
library(neuralnet)
```

```
## Warning: package 'neuralnet' was built under R version 3.5.2
```

```
rm(list=ls()) # clear work space
#dev.off(dev.list()["RStudioGD"]) # clear plots

#######################################################################################################
#                                            Prepare Data
#######################################################################################################

setwd("C:/Users/dvjr2/Google Drive/Documents/Syracuse/IST_687/Project/")
filePath = "clean_power_lifting.csv" # cleaned data - yay Katie!

# read data and take a look
powerLiftingOG <- read.csv(file = filePath, header=TRUE, sep=",", stringsAsFactors = FALSE)
str(powerLiftingOG)
```

```
## 'data.frame':    316076 obs. of  13 variables:
##  $ X             : int  1 2 4 5 6 7 9 10 11 12 ...
##  $ MeetID        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Name          : chr  "Angie Belk Terry" "Dawn Bogart" "Dawn Bogart" "Destiny Dula" ...
##  $ Sex           : chr  "F" "F" "F" "F" ...
##  $ Equipment     : chr  "Wraps" "Single-ply" "Raw" "Raw" ...
##  $ Age           : int  47 42 42 18 28 60 52 52 24 56 ...
##  $ BodyweightKg  : num  59.6 58.5 58.5 63.7 62.4 ...
##  $ WeightClassKg : chr  "60" "60" "60" "67.5" ...
##  $ BestSquatKg   : num  47.6 142.9 NA NA 170.1 ...
##  $ BestBenchKg   : num  20.4 95.2 95.2 31.8 77.1 ...
##  $ BestDeadliftKg: num  70.3 163.3 NA 90.7 145.2 ...
##  $ TotalKg       : num  138.3 401.4 95.2 122.5 392.4 ...
##  $ Place         : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
summary(powerLiftingOG)
```

```
##        X               MeetID         Name               Sex
##  Min.   :     1   Min.   :   0   Length:316076      Length:316076
##  1st Qu.: 90561   1st Qu.:2516   Class :character   Class :character
##  Median :179024   Median :5801   Mode  :character   Mode  :character
##  Mean   :183718   Mean   :4962
##  3rd Qu.:274312   3rd Qu.:7018
##  Max.   :386414   Max.   :8481
##
##   Equipment              Age          BodyweightKg    WeightClassKg
##  Length:316076      Min.   : 5.0     Min.   : 17.24   Length:316076
##  Class :character   1st Qu.:22.0     1st Qu.: 70.00   Class :character
##  Mode  :character   Median :28.0     Median : 82.80   Mode  :character
##                     Mean   :30.9     Mean   : 86.63
##                     3rd Qu.:36.0     3rd Qu.:100.00
##                     Max.   :95.0     Max.   :242.40
##                     NA's   :204920   NA's   :823
##   BestSquatKg      BestBenchKg      BestDeadliftKg      TotalKg
##  Min.   :-175.0   Min.   :-167.50   Min.   :  2.27   Min.   :  11.0
##  1st Qu.: 127.5   1st Qu.:  79.38   1st Qu.:149.69   1st Qu.: 287.5
##  Median : 174.6   Median : 115.00   Median :195.00   Median : 435.4
##  Mean   : 177.9   Mean   : 119.00   Mean   :195.57   Mean   : 433.9
##  3rd Qu.: 217.7   3rd Qu.: 150.00   3rd Qu.:237.50   3rd Qu.: 570.0
##  Max.   : 573.8   Max.   : 455.86   Max.   :460.40   Max.   :1365.3
##  NA's   :55095    NA's   :8299      NA's   :39720
##      Place
##  Min.   : 1.000
##  1st Qu.: 1.000
##  Median : 1.000
##  Mean   : 2.951
##  3rd Qu.: 3.000
##  Max.   :77.000
##
```

```r
powerLiftingOG$Sex = as.factor(powerLiftingOG$Sex) # Make Sex a factor for predicting

powerLifting = powerLiftingOG[ , c(-1,-2,-3, -5,-6, -8, -12, -13)] # get rid of unwanted cols
powerLifting = na.omit(powerLifting) # clear NAs to run model

powerLifting = powerLifting[sample(nrow(powerLifting)), ] # randomize
trngData = powerLifting[1:ceiling(nrow(powerLifting)*.7) , ] # trng data 70% of original data
testData = powerLifting[(ceiling(nrow(powerLifting)*.7)+1):nrow(powerLifting), ] # test data

#######################################################################################################
#                                             Random Forest
#######################################################################################################

modelRF = randomForest(trngData[ , -1], trngData[ , 1]) # independent, dependent
summary(modelRF)
```

```
##                 Length Class  Mode
## call                 3 -none- call
## type                 1 -none- character
## predicted       181170 factor numeric
## err.rate          1500 -none- numeric
## confusion            6 -none- numeric
## votes           362340 matrix numeric
## oob.times       181170 -none- numeric
## classes              2 -none- character
## importance           4 -none- numeric
## importanceSD         0 -none- NULL
## localImportance      0 -none- NULL
## proximity            0 -none- NULL
## ntree                1 -none- numeric
## mtry                 1 -none- numeric
## forest              14 -none- list
## y               181170 factor numeric
## test                 0 -none- NULL
## inbag                0 -none- NULL
```

```r
round(importance(modelRF), 2) # importance of variables, high is better
```

```
##                 MeanDecreaseGini
## BodyweightKg             9521.10
## BestSquatKg             10644.46
## BestBenchKg             28598.59
## BestDeadliftKg          15228.76
```

```r
# make predictions
results = predict(modelRF, testData[ ,  -1])

# look at results w/ confusion matrix
table(results)
```

```
## results
##     F     M
## 17789 59854
```

```r
table(testData$Sex)
```

```
##
##     F     M
## 18334 59309
```

```r
matrix = table(results, testData$Sex)
matrix
```

```
##
## results     F     M
##       F 15103  2686
##       M  3231 56623
```

```r
# male vs female accuracy
print(paste('Female Predict Accuracy:', matrix[1,1]/sum(testData$Sex == "F")))
```

```
## [1] "Female Predict Accuracy: 0.823770044725646"
```

```r
print(paste('Male Predict Accuracy:', matrix[2,2]/sum(testData$Sex == "M")))
```

```
## [1] "Male Predict Accuracy: 0.954711763813249"
```

```r
# Overall accuracy
print(paste('Overall Accuracy: ', (matrix[1,1] + matrix[2,2])/length(testData$Sex )))
```

```
## [1] "Overall Accuracy:  0.923792228533158"
```

```
#######################################################################################################
#                                           Linear Model
#######################################################################################################


# change F/M to 1/0, LM seems to work better this way
trngData$Sex = ifelse(trngData$Sex == 'F', 1, 0)
testData$Sex = ifelse(testData$Sex == 'F', 1, 0)

# Lm with same variables used in randomForest()
modelGLM = glm(Sex ~ BodyweightKg + BestSquatKg + BestBenchKg + BestDeadliftKg, data = trngData, family
summary(modelGLM) # all variables are significant
```

```
##
## Call:
## glm(formula = Sex ~ BodyweightKg + BestSquatKg + BestBenchKg +
##     BestDeadliftKg, family = "binomial", data = trngData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.3538  -0.3689  -0.1137  -0.0047   4.9663
##
## Coefficients:
##                  Estimate Std. Error  z value Pr(>|z|)
## (Intercept)     5.9136238  0.0433610  136.381  < 2e-16 ***
## BodyweightKg   -0.0024241  0.0004953   -4.894 9.88e-07 ***
## BestSquatKg     0.0076637  0.0004247   18.043  < 2e-16 ***
## BestBenchKg    -0.0686702  0.0006539 -105.016  < 2e-16 ***
## BestDeadliftKg -0.0113614  0.0004353  -26.102  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 197497  on 181169  degrees of freedom
## Residual deviance:  99006  on 181165  degrees of freedom
## AIC: 99016
##
## Number of Fisher Scoring iterations: 7
```

```
accuracy = .5 # accuracy variable, if prediction is above, assign predicition value
result = predict(modelGLM, testData[ , 2:5], type = "response") # we want the percentage of females
result = ifelse(result > accuracy, 1, 0) # assign male or female prediction based on accuracy
Error = mean(result != testData[ , 1]) # getting wrong values
print(paste('Overall Accuracy:', 1  - Error)) # total accuracy
```

```
## [1] "Overall Accuracy: 0.883917416895277"
```

```
# build a neural network based on linear model
modelNeural = neuralnet(Sex ~ BodyweightKg + BestSquatKg + BestBenchKg + BestDeadliftKg, data = trngDat
                        hidden = 2, lifesign = 'minimal', linear.output = FALSE, threshold = 0.1)
```

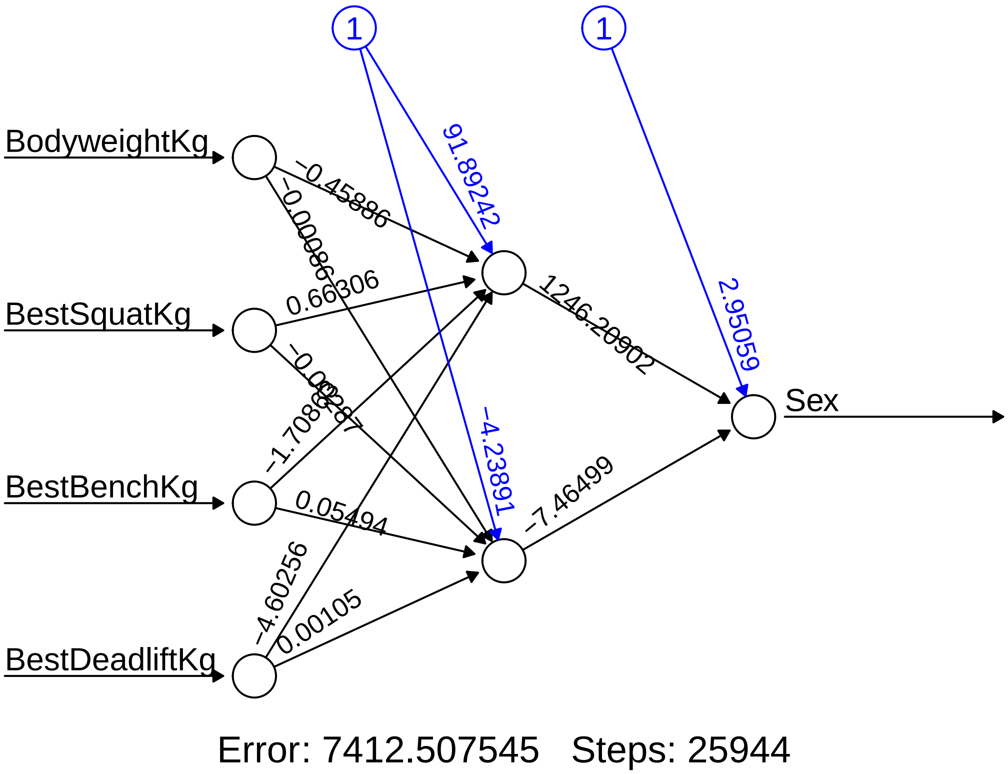
```
## hidden: 2 thresh: 0.1 rep: 1/1 steps:
```

```
##   25944  error: 7412.50755   time: 18.02 mins
```

```
summary(modelNeural)
```

```
##                       Length Class      Mode    
## call                       7 -none-     call    
## response              181170 -none-     numeric 
## covariate             724680 -none-     numeric 
## model.list                 2 -none-     list    
## err.fct                    1 -none-     function
## act.fct                    1 -none-     function
## linear.output              1 -none-     logical 
## data                       5 data.frame list    
## exclude                    0 -none-     NULL    
## net.result                 1 -none-     list    
## weights                    1 -none-     list    
## generalized.weights        1 -none-     list    
## startweights               1 -none-     list    
## result.matrix             16 -none-     numeric
```

```
plot(modelNeural, rep = 'best') #plot it
```



```
modelNeural$result.matrix # see the matrix
```

```
##                                    [,1]
## error                      7.412508e+03
## reached.threshold          8.854398e-02
## steps                      2.594400e+04
## Intercept.to.1layhid1      9.189242e+01
## BodyweightKg.to.1layhid1   -4.588611e-01
## BestSquatKg.to.1layhid1     6.630582e-01
## BestBenchKg.to.1layhid1    -1.708625e+00
## BestDeadliftKg.to.1layhid1 -4.602560e+00
## Intercept.to.1layhid2      -4.238908e+00
## BodyweightKg.to.1layhid2   -8.584729e-04
## BestSquatKg.to.1layhid2    -2.868649e-03
## BestBenchKg.to.1layhid2     5.494070e-02
## BestDeadliftKg.to.1layhid2  1.052004e-03
## Intercept.to.Sex            2.950593e+00
## 1layhid1.to.Sex             1.246209e+03
## 1layhid2.to.Sex            -7.464994e+00
```

```
# predict and accuracy
resultNeural = predict(modelNeural, testData[ , 2:5], type = "response")
resultNeural = ifelse(resultNeural > accuracy, 1, 0) # assign male or female prediction based on accura
Error = mean(resultNeural != testData[ , 1]) # getting wrong values
print(paste('Overall Accuracy:', 1  - Error)) # total accuracy
```

```
## [1] "Overall Accuracy: 0.885385675463339"
```