Olympics Project

Marley Akonnor

9/14/2021

Question: Can we predict the Medal won based on Age, Height, and Weight

```
library(neuralnet)
```

Read in the dataset

```
oly_data <- read.csv("/Users/m/Documents/M.S Syracuse Data Science/Courses/IST 687 - Intro to Data Scie
```

#Clean the dataset - rid of NAs

```
oly_data <- na.omit(oly_data)
```

Check to see if any remain

```
sum(is.na(oly_data))
```

```
## [1] 0
```

Inspect the structure

```
str(oly_data)
```

```
## 'data.frame':    30181 obs. of  15 variables:
##  $ ID    : int  16 17 17 17 17 17 20 20 20 20 ...
##  $ Name  : chr  "Juhamatti Tapio Aaltonen" "Paavo Johannes Aaltonen" "Paavo Johannes Aaltonen" "Paavo
##  $ Sex   : chr  "M" "M" "M" "M" ...
##  $ Age   : int  28 28 28 28 28 32 20 20 22 22 ...
##  $ Height: int  184 175 175 175 175 175 176 176 176 176 ...
##  $ Weight: num  85 64 64 64 64 64 85 85 85 85 ...
##  $ Team  : chr  "Finland" "Finland" "Finland" "Finland" ...
##  $ NOC   : chr  "FIN" "FIN" "FIN" "FIN" ...
##  $ Games : chr  "2014 Winter" "1948 Summer" "1948 Summer" "1948 Summer" ...
##  $ Year  : int  2014 1948 1948 1948 1948 1952 1992 1992 1994 1994 ...
##  $ Season: chr  "Winter" "Summer" "Summer" "Summer" ...
##  $ City  : chr  "Sochi" "London" "London" "London" ...
##  $ Sport : chr  "Ice Hockey" "Gymnastics" "Gymnastics" "Gymnastics" ...
##  $ Event : chr  "Ice Hockey Men's Ice Hockey" "Gymnastics Men's Individual All-Around" "Gymnastics M
##  $ Medal : chr  "Bronze" "Bronze" "Gold" "Gold" ...
##  - attr(*, "na.action")= 'omit' Named int [1:240935] 1 2 3 4 5 6 7 8 9 10 ...
##   ..- attr(*, "names")= chr [1:240935] "1" "2" "3" "4" ...
```

Look at a summmary of our data

```
summary(oly_data)
```

```
##        ID             Name               Sex                 Age
##  Min.   :    16   Length:30181       Length:30181       Min.   :13.00
##  1st Qu.: 37494   Class :character   Class :character   1st Qu.:22.00
```

```
##  Median : 69771   Mode  :character   Mode  :character   Median :25.00
##  Mean   : 70226                                        Mean   :25.43
##  3rd Qu.:104111                                        3rd Qu.:28.00
##  Max.   :135563                                        Max.   :66.00
##      Height          Weight           Team               NOC
##  Min.   :136.0   Min.   : 28.00   Length:30181       Length:30181
##  1st Qu.:170.0   1st Qu.: 63.00   Class :character   Class :character
##  Median :178.0   Median : 73.00   Mode  :character   Mode  :character
##  Mean   :177.6   Mean   : 73.75
##  3rd Qu.:185.0   3rd Qu.: 83.00
##  Max.   :223.0   Max.   :182.00
##     Games                Year         Season              City
##  Length:30181       Min.   :1896   Length:30181       Length:30181
##  Class :character   1st Qu.:1976   Class :character   Class :character
##  Mode  :character   Median :1992   Mode  :character   Mode  :character
##                     Mean   :1988
##                     3rd Qu.:2006
##                     Max.   :2016
##     Sport              Event              Medal
##  Length:30181       Length:30181       Length:30181
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
```

Convert the "Sport" column from characters to factors

```
oly_data$Sport <- as.factor(oly_data$Sport)
```

What are the unique values of medal?

```
unique(oly_data$Medal)
```

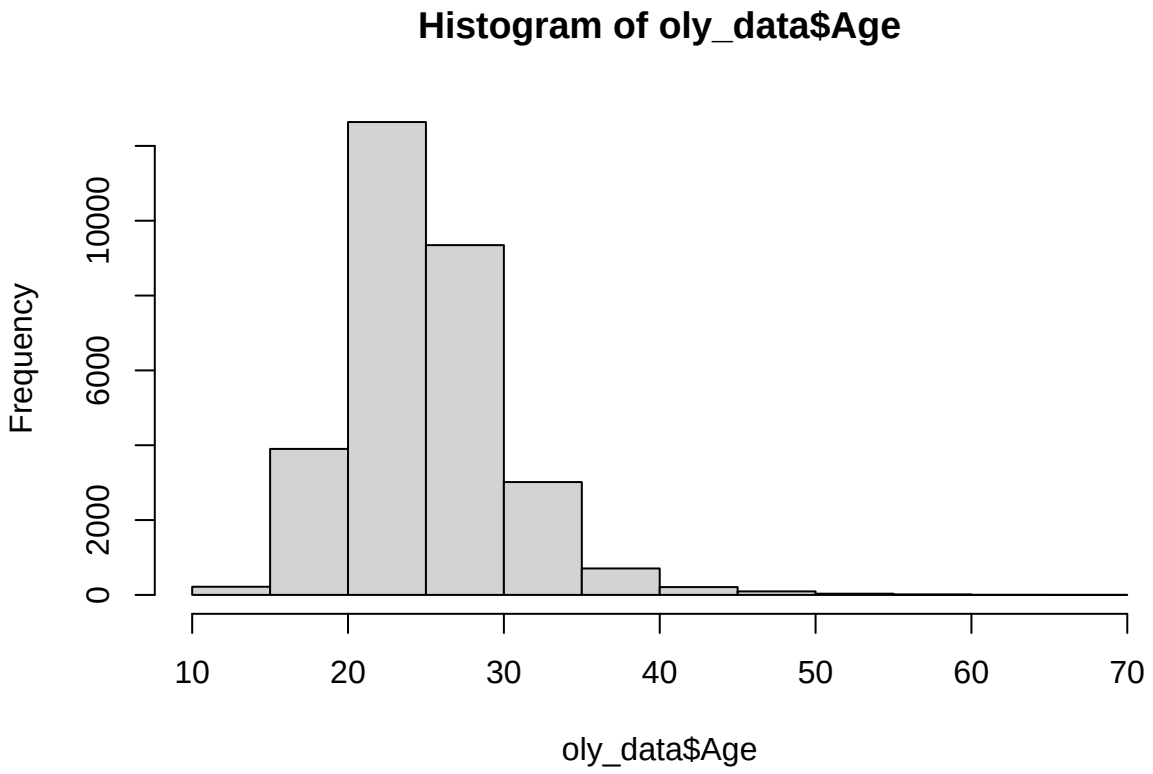
```
## [1] "Bronze" "Gold"   "Silver"
```

Make "Medal" column easier to read and convert to int

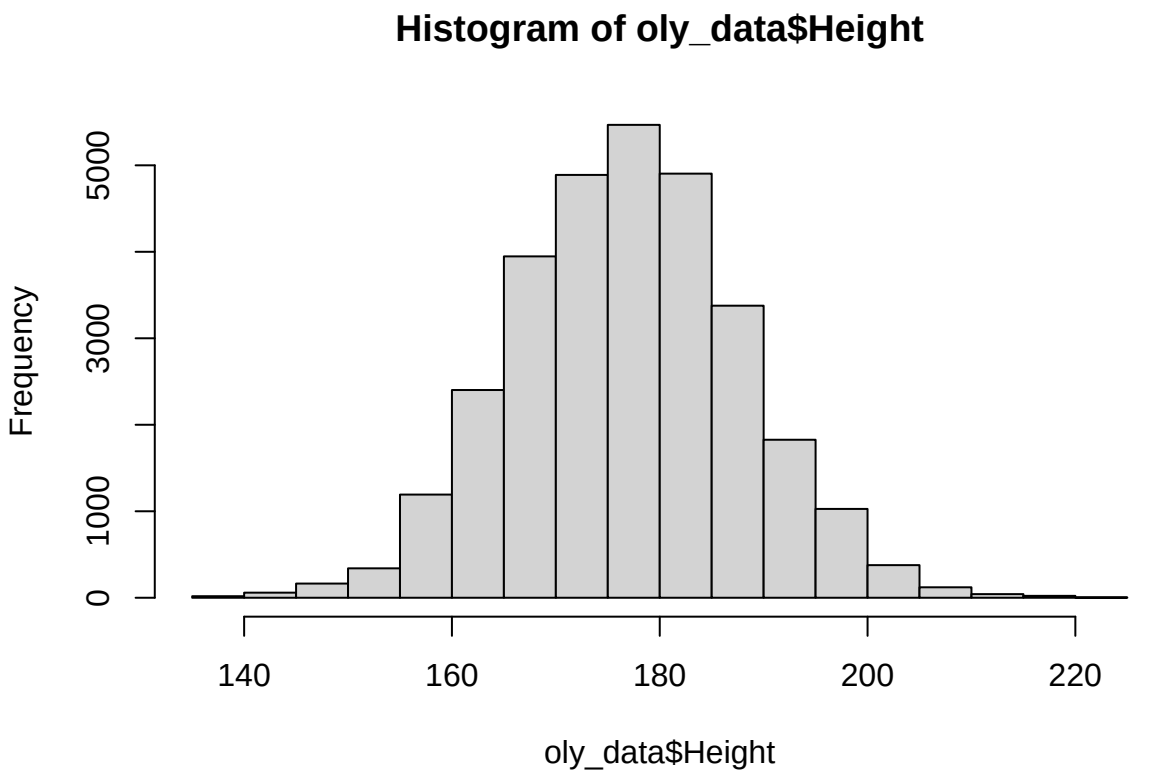
```
oly_data[oly_data$Medal == "Gold", ]$Medal <- 0
oly_data[oly_data$Medal == "Silver", ]$Medal <- 1
oly_data[oly_data$Medal == "Bronze", ]$Medal <- 2
oly_data$Medal <- as.numeric(oly_data$Medal)
```

Examine distributions with histograms

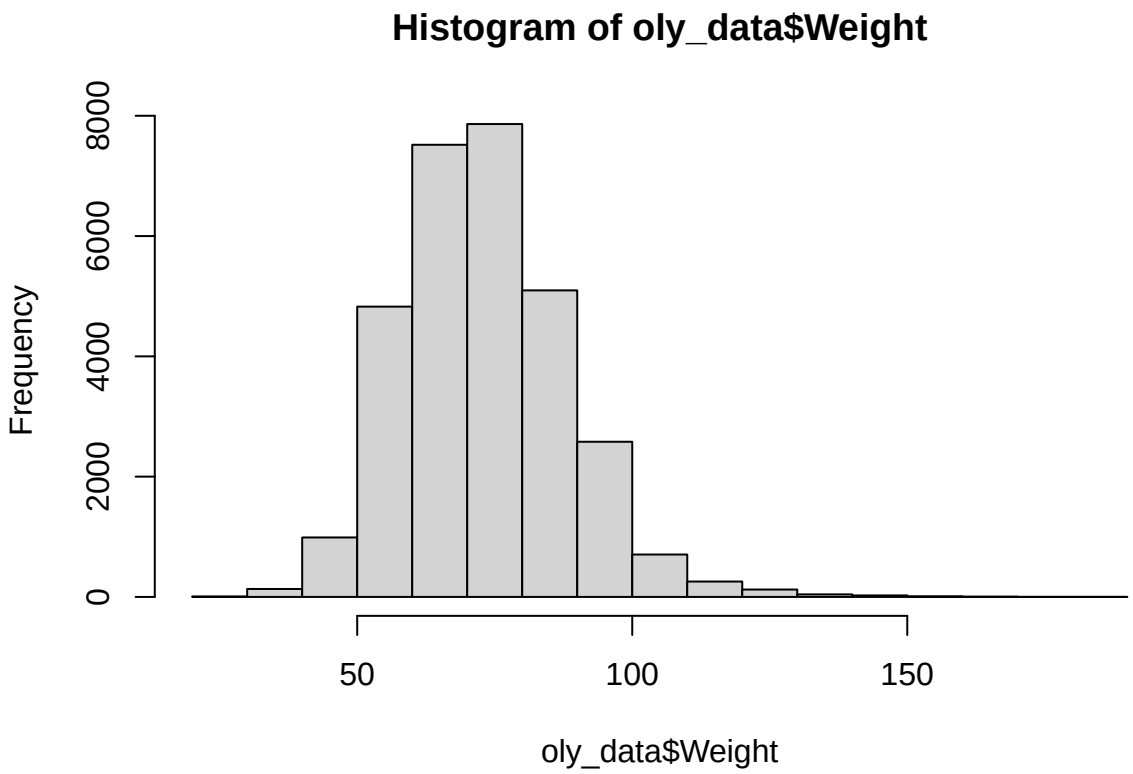
```
hist(oly_data$Age)
```




```
hist(oly_data$Height)
```




```
hist(oly_data$Weight)
```

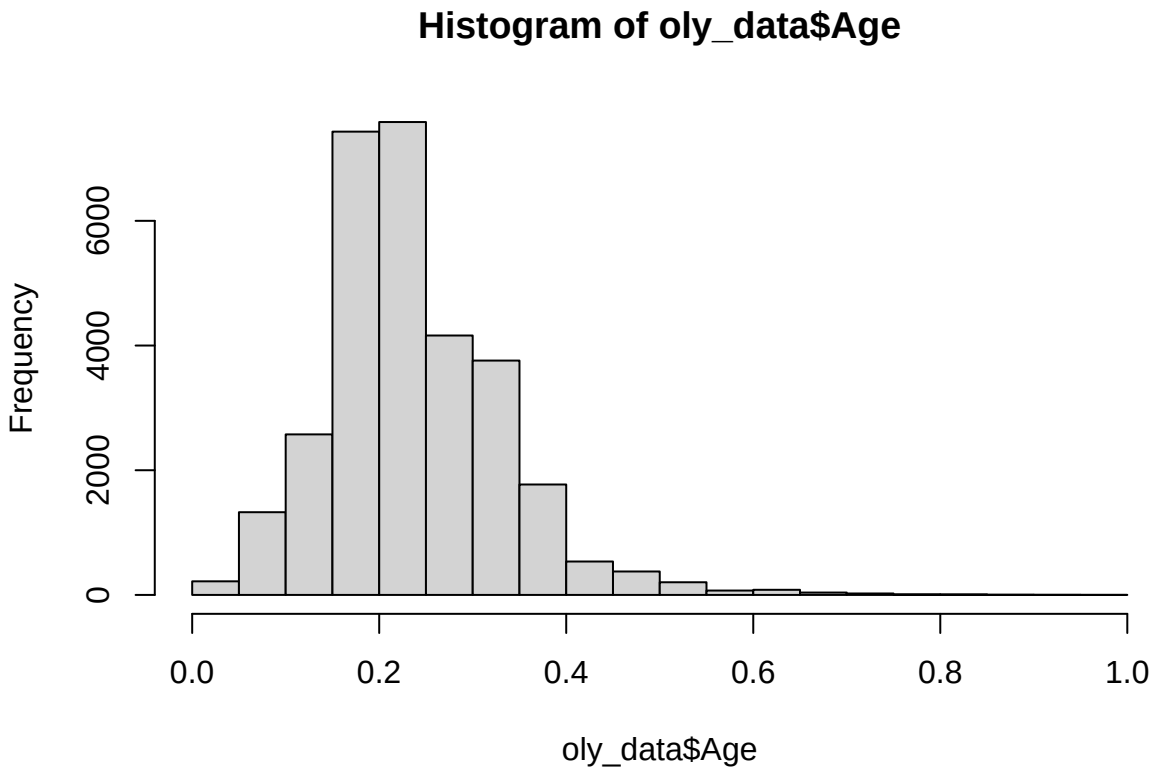


Min-Max normalization create Normalization function (https://datasharkie.com/how-to-normalize-data-in-r/)

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

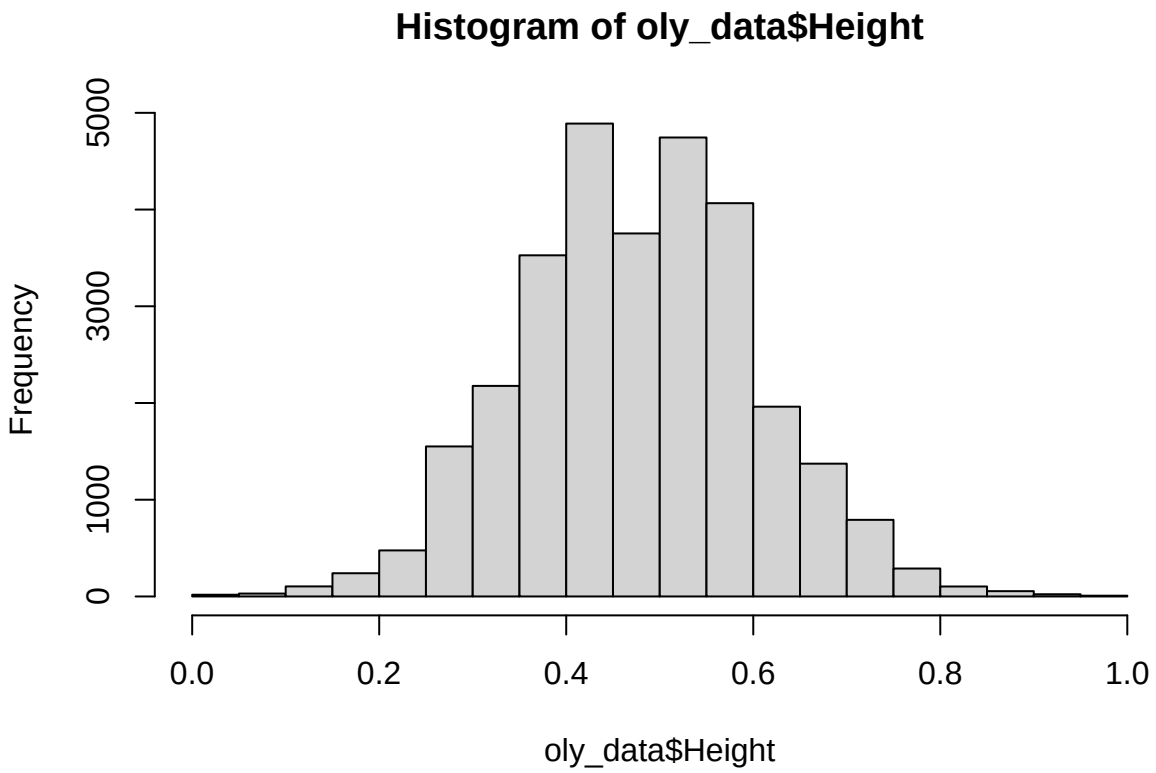
Normalize Age and look at histogram

```
oly_data$Age <- normalize(oly_data$Age)
hist(oly_data$Age)
```



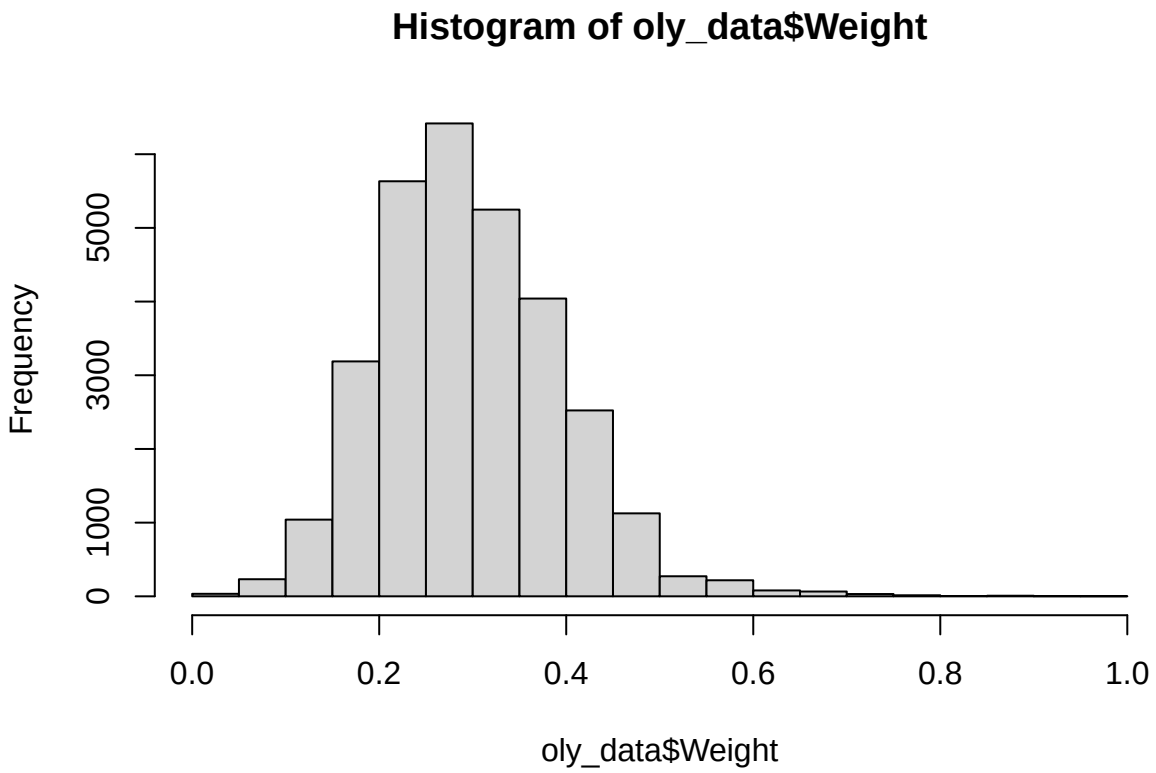
Normalize Height and look at histogram

```
oly_data$Height <- normalize(oly_data$Height)
hist(oly_data$Height)
```



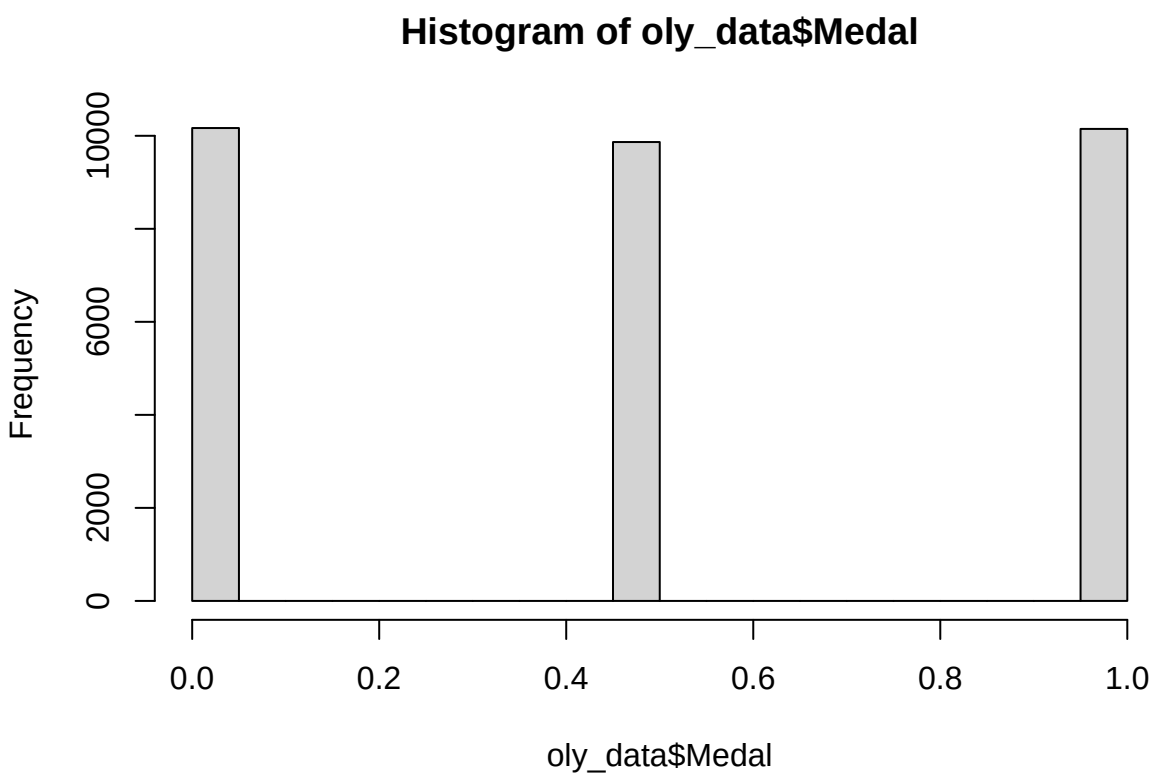
Normalize Weight and look at histogram

```
oly_data$Weight <- normalize(oly_data$Weight)
hist(oly_data$Weight)
```



Normalize Medal and look at histogram

```
oly_data$Medal <- normalize(oly_data$Medal)
hist(oly_data$Medal)
```



Create a new Olympic dataframe with relevant variables

```
new_oly_data <- data.frame(oly_data$Medal, oly_data$Age, oly_data$Height, oly_data$Weight)
```

Rename the columns

```
col_names <- c("Medal", "Age", "Height", "Weight")
colnames(new_oly_data) <- col_names
```

Data partition and random indexing

```
ran_ind <- sample(1:dim(new_oly_data)[1])
prorat_data <- floor(2 * dim(new_oly_data)[1]/3)
```

Create training data

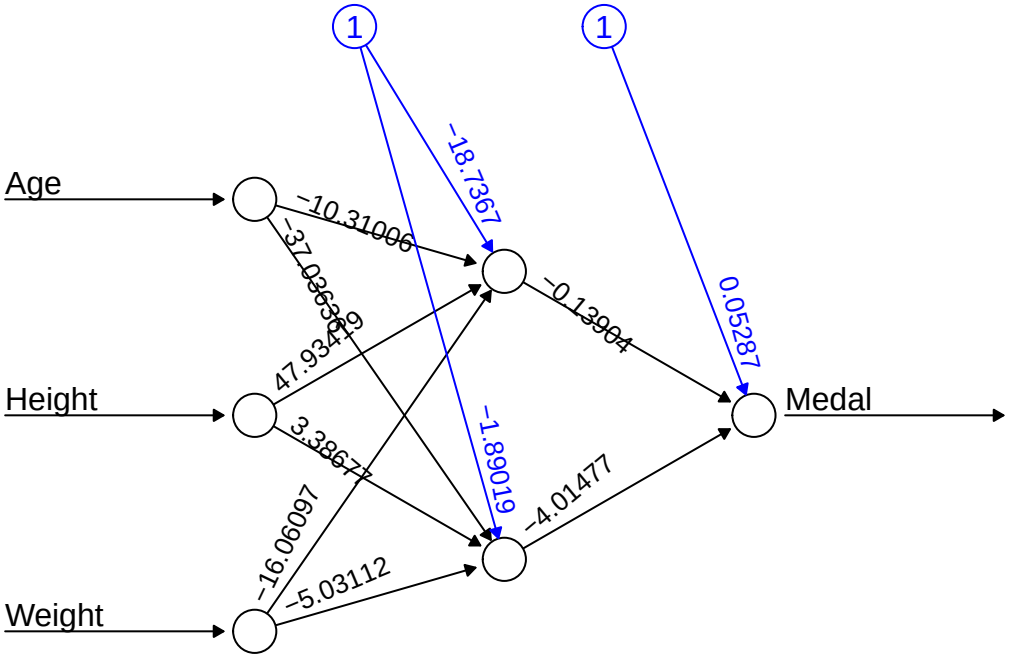
```
training_data <- new_oly_data[ran_ind[1:prorat_data], ]
```

Create test data

```
testing_data <- new_oly_data[ran_ind[(prorat_data+1):dim(new_oly_data)[1]],]
```

Neural Network model

```
oly_nn <- neuralnet(Medal ~ Age + Height + Weight, data = training_data, hidden = 2, err.fct = "sse", li
plot(oly_nn, rep = "best")
```



Error: 1689.78555 Steps: 8876

Prediction

```
nn_output <- compute(oly_nn, training_data[, -1])
head(nn_output$net.result)
```

```
##            [,1]
## 22711 0.4919877
## 7961  0.4775706
## 10998 0.5128599
## 19823 0.4979021
## 3281  0.5102859
## 12298 0.5128011
```

```
head(training_data[1:6, ])
```

```
##       Medal        Age    Height    Weight
## 22711   0.0 0.22641509 0.5402299 0.2727273
## 7961    0.5 0.15094340 0.6896552 0.3636364
## 10998   0.0 0.16981132 0.3333333 0.1883117
## 19823   0.0 0.09433962 0.4712644 0.2467532
## 3281    0.5 0.11320755 0.2758621 0.1363636
## 12298   0.5 0.16981132 0.4482759 0.3766234
```

Create a subset of the testing data

```
temp_test <- subset(testing_data, select = c("Age", "Height", "Weight"))
head(temp_test)
```

```
##              Age    Height    Weight
## 2118  0.16981132 0.5747126 0.2857143
## 6777  0.18867925 0.5402299 0.3571429
## 8314  0.22641509 0.5862069 0.3181818
## 15663 0.35849057 0.6896552 0.3961039
```

```
## 5958  0.05660377 0.2988506 0.2532468
## 19246 0.11320755 0.5402299 0.3376623
```

Run Neural Net on test data

```
olynn.results <- compute(oly_nn, temp_test)
results <- data.frame(actual=testing_data$Medal, prediction=olynn.results$net.result)
results[1:20,]
```

```
##       actual prediction
## 2118     0.0  0.4807068
## 6777     0.0  0.5001350
## 8314     0.0  0.4828676
## 15663    0.0  0.4789540
## 5958     0.0  0.4990791
## 19246    0.0  0.4884275
## 4489     1.0  0.3529856
## 29878    0.5  0.4886681
## 2766     0.0  0.4998082
## 15240    0.0  0.4708438
## 22957    1.0  0.5130093
## 1265     1.0  0.5064021
## 8503     1.0  0.5127882
## 20460    0.0  0.5129187
## 14775    0.5  0.5113103
## 20574    0.5  0.5130989
## 28641    0.0  0.5130130
## 20231    0.5  0.5129660
## 22203    0.5  0.5124880
## 5231     0.5  0.4883407
```