Perform Neural Networks ----------

Step 1: Collecting data

- Read in data and examine structure

```
concrete <- read.csv("http://www.sci.csueastbay.edu/~esuess/classes/Statistics_6620/Presentations/ml11/concrete.csv")
str(concrete)
```

```
## 'data.frame':    1030 obs. of  9 variables:
##  $ cement      : num  141 169 250 266 155 ...
##  $ slag        : num  212 42.2 0 114 183.4 ...
##  $ ash         : num  0 124.3 95.7 0 0 ...
##  $ water       : num  204 158 187 228 193 ...
##  $ superplastic: num  0 10.8 5.5 0 9.1 0 0 6.4 0 9 ...
##  $ coarseagg   : num  972 1081 957 932 1047 ...
##  $ fineagg     : num  748 796 861 670 697 ...
##  $ age         : int  28 14 28 28 28 90 7 56 28 28 ...
##  $ strength    : num  29.9 23.5 29.2 45.9 18.3 ...
```

Step 2: Exploring and preparing the data ----

- Custom normalization function

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

- Apply normalization to entire data frame

```
concrete_norm <- as.data.frame(lapply(concrete, normalize))
```

- Confirm that the range is now between zero and one

```
summary(concrete_norm$strength)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.2664  0.4001  0.4172  0.5457  1.0000
```

- Compared to the original minimum and maximum

```
summary(concrete$strength)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.33   23.71   34.44   35.82   46.14   82.60
```

- Create training and test data

```
concrete_train <- concrete_norm[1:773, ]
concrete_test <- concrete_norm[774:1030, ]
```

Step 3: Training a model on the data ----

- train the neuralnet model

```
library(neuralnet)
```

- simple ANN with only a single hidden neuron

```
set.seed(12345) # to guarantee repeatable results
concrete_model <- neuralnet(formula = strength ~ cement + slag +
                                ash + water + superplastic +
                                coarseagg + fineagg + age,
                                data = concrete_train)
```

- visualize the network topology

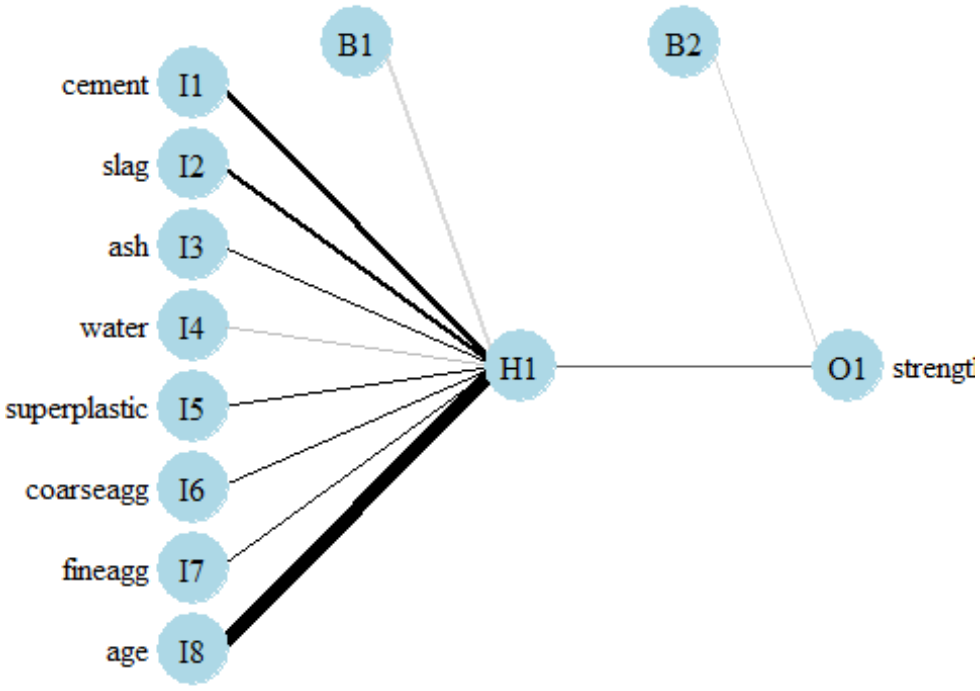
```
plot(concrete_model)
```

- alternative plot

```
library(NeuralNetTools)
```

- plotnet

```
par(mar = numeric(4), family = 'serif')
plotnet(concrete_model, alpha = 0.6)
```



Step 4:

Evaluating model performance ---- - obtain model results

```
model_results <- compute(concrete_model, concrete_test[1:8])
```

- obtain predicted strength values

```
predicted_strength <- model_results$net.result
```

- examine the correlation between predicted and actual values is 0.806 which is quite strong.

```
cor(predicted_strength, concrete_test$strength)
```

```
##              [,1]
## [1,] 0.8064655576
```

- produce actual predictions by

```
head(predicted_strength)

##             [,1]
## 774 0.3258991537
## 775 0.4677425372
## 776 0.2370268181
## 777 0.6718811029
## 778 0.4663428766
## 779 0.4685272270

concrete_train_original_strength <- concrete[1:773,"strength"]

strength_min <- min(concrete_train_original_strength)
strength_max <- max(concrete_train_original_strength)

head(concrete_train_original_strength)

## [1] 29.89 23.51 29.22 45.85 18.29 21.86
```

- custom normalization function
- we need to unormalize for the result to make sense

```
unnormalize <- function(x, min, max) {
  return( (max - min)*x + min )
}

strength_pred <- unnormalize(predicted_strength, strength_min, strength_max)
head(strength_pred)

##            [,1]
## 774 28.21291079
## 775 39.47811230
## 776 21.15466990
## 777 55.69079719
## 778 39.36695126
## 779 39.54043237
```

Step 5: Improving model performance ----

- a more complex neural network topology with 5 hidden neurons

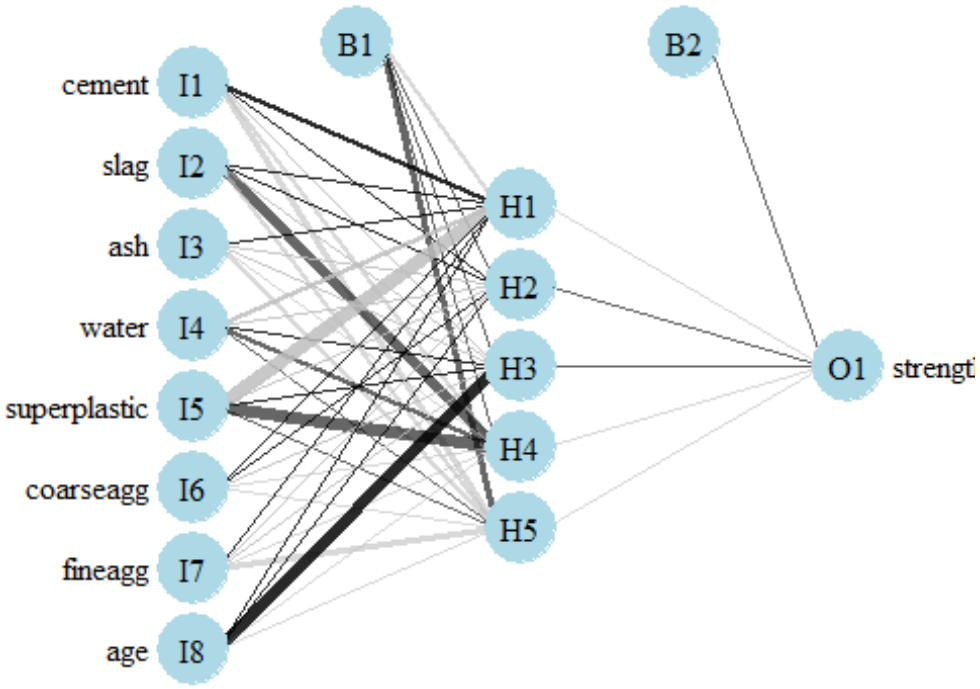
```
set.seed(12345) # to guarantee repeatable results
concrete_model2 <- neuralnet(strength ~ cement + slag +
                               ash + water + superplastic +
                               coarseagg + fineagg + age,
                               data = concrete_train, hidden = 5, act.fct = "
logistic")
```

- plot the network

```
plot(concrete_model2)
```

- plotnet

```
par(mar = numeric(4), family = 'serif')
plotnet(concrete_model2, alpha = 0.6)
```



- the correlation has improved to 0.92

```
model_results2 <- compute(concrete_model2, concrete_test[1:8])
predicted_strength2 <- model_results2$net.result
cor(predicted_strength2, concrete_test$strength)

##              [,1]
## [1,] 0.9244533426
```

- try different activation function
- a more complex neural network topology with 5 hidden neurons

```
set.seed(12345) # to guarantee repeatable results
concrete_model2 <- neuralnet(strength ~ cement + slag +
                               ash + water + superplastic +
                               coarseagg + fineagg + age,
                             data = concrete_train, hidden = 5, act.fct = "ta
nh")
```

- the correlation has gotten worse to 0.574

```
model_results2 <- compute(concrete_model2, concrete_test[1:8])
predicted_strength2 <- model_results2$net.result
cor(predicted_strength2, concrete_test$strength)
```

```
##              [,1]
## [1,] 0.5741729322
```

- using h2o deeplearning
- h2o produce `0.865` in correlation.

```
library(h2o)
```

```
##
## ----------------------------------------------------------------------
##
## Your next step is to start H2O:
##     > h2o.init()
##
## For H2O package documentation, ask for help:
##     > ??h2o
##
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit http://docs.h2o.ai
##
## ----------------------------------------------------------------------
```

```
##
## Attaching package: 'h2o'
```

```
## The following objects are masked from 'package:stats':
##
##     cor, sd, var
```

```
## The following objects are masked from 'package:base':
##
##     %*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,
##     colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##     log10, log1p, log2, round, signif, trunc
```

```
h2o.init()
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         3 minutes 1 seconds
##     H2O cluster version:        3.10.4.6
##     H2O cluster version age:    25 days
##     H2O cluster name:           H2O_started_from_R_tra_gfp373
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   3.30 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  2
```

```
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     R Version:                  R version 3.3.3 (2017-03-06)
```

```
concrete.hex <- h2o.importFile("http://www.sci.csueastbay.edu/~esuess/classes
/Statistics_6620/Presentations/ml11/concrete.csv")
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
summary(concrete.hex)
```

```
## Warning in summary.H2OFrame(concrete.hex): Approximated quantiles
## computed! If you are interested in exact quantiles, please pass the
## `exact_quantiles=TRUE` parameter.
```

```
##  cement          slag             ash              water
##  Min.   :102.0   Min.   :  0.00   Min.   :  0.00   Min.   :121.8
##  1st Qu.:192.1   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:164.9
##  Median :272.6   Median : 21.92   Median :  0.00   Median :184.9
##  Mean   :281.2   Mean   : 73.90   Mean   : 54.19   Mean   :181.6
##  3rd Qu.:349.9   3rd Qu.:142.68   3rd Qu.:118.26   3rd Qu.:191.9
##  Max.   :540.0   Max.   :359.40   Max.   :200.10   Max.   :247.0
##  superplastic     coarseagg        fineagg          age
##  Min.   : 0.000   Min.   : 801.0   Min.   :594.0   Min.   :  1.00
##  1st Qu.: 0.000   1st Qu.: 931.7   1st Qu.:730.8   1st Qu.:  7.00
##  Median : 6.376   Median : 967.8   Median :779.1   Median : 28.00
##  Mean   : 6.205   Mean   : 972.9   Mean   :773.6   Mean   : 45.66
##  3rd Qu.:10.175   3rd Qu.:1029.1   3rd Qu.:824.0   3rd Qu.: 56.00
##  Max.   :32.200   Max.   :1145.0   Max.   :992.6   Max.   :365.00
##  strength
##  Min.   : 2.33
##  1st Qu.:23.68
##  Median :34.40
##  Mean   :35.82
##  3rd Qu.:46.10
##  Max.   :82.60
```

```
splits <- h2o.splitFrame(concrete.hex, 0.75, seed=1234)
```

```
dl <- h2o.deeplearning(x=1:8,y="strength",training_frame=splits[[1]],activati
on = "Tanh",
                       hidden = 200, distribution = "gaussian")
```

```
##
  |
  |                                                                      |   0%
  |
  |======                                                                |  10%
  |
  |======================================================================| 100%
```

```
dl.predict <- h2o.predict (dl, splits[[2]])
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
cor(as.vector(dl.predict), as.vector(splits[[2]]$strength))
```

```
## [1] 0.8652384433
```

```
dl@parameters
```

```
## $model_id
## [1] "DeepLearning_model_R_1495502863840_2"
##
## $training_frame
## [1] "RTMP_sid_b093_2"
##
## $activation
## [1] "Tanh"
##
## $hidden
## [1] 200
##
## $seed
## [1] -4281458590304587264
##
## $distribution
## [1] "gaussian"
##
## $x
## [1] "cement"       "slag"         "ash"          "water"
## [5] "superplastic" "coarseagg"    "fineagg"      "age"
##
## $y
## [1] "strength"
```

```
h2o.performance(dl)
```

```
## H2ORegressionMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on full training frame **
```

```
##
## MSE:  58.88580991
## RMSE:  7.673709006
## MAE:  5.957876254
## RMSLE:  0.2482161599
## Mean Residual Deviance :  58.88580991
```

```
h2o.shutdown()
```

```
## Are you sure you want to shutdown the H2O instance running at http://local
host:54321/ (Y/N)?
```