Post02

Anbo Cao

12/3/2017

Introduction

In this post I will build a neural network for the purpose of introduce how machine learning is achieved in R. It will be very interesting to see how everyone is easily access to the powerful tool. There are few packages that is very useful when implementing a neural network for machine learning. In this post, I will introduce some useful machine learning packages and use them to illustrate R is as important as other languages in the area of machine learning.

Motivation

Machine learning is one of the most popular topics in today's computer science. However, the name of "Machine Learning" does not only belongs to students who study computer science. In fact, before the computer is invented, the concept of machine learning already exist in statistics, called "statistical learning".

Preparation

We need to install the package 'neuralnet' and 'maps'

```
install.packages("neuralnet")
install.packages("maps")
```

NN explanation

Like human brain, a neural network is made up by each "neuron". The neuro is usually called an node and they are connected just like our brain, layer by layer.

```
library("neuralnet")
library("maps")
XOR <- c(0,1,1,0)
xor.data <- data.frame(expand.grid(c(0,1), c(0,1)), XOR)
print(net.xor <- neuralnet( XOR~Var1+Var2, xor.data, hidden=c(2), rep=5))
```

```
## $call
## neuralnet(formula = XOR ~ Var1 + Var2, data = xor.data, hidden = c(2),
##     rep = 5)
##
## $response
##   XOR
## 1   0
## 2   1
## 3   1
## 4   0
##
## $covariate
##      [,1] [,2]
## [1,]    0    0
## [2,]    1    0
## [3,]    0    1
## [4,]    1    1
##
## $model.list
## $model.list$response
## [1] "XOR"
##
## $model.list$variables
## [1] "Var1" "Var2"
##
##
## $err.fct
## function (x, y)
## {
##     1/2 * (y - x)^2
## }
## <environment: 0x7fe1bc041078>
## attr(,"type")
## [1] "sse"
##
## $act.fct
## function (x)
## {
##     1/(1 + exp(-x))
## }
## <environment: 0x7fe1bc041078>
## attr(,"type")
## [1] "logistic"
##
```

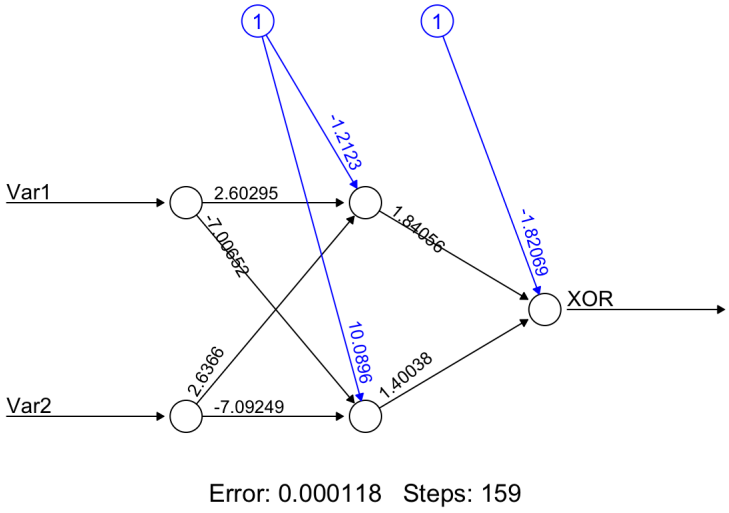
```
## $linear.output
## [1] TRUE
##
## $data
##   Var1 Var2 XOR
## 1    0    0   0
## 2    1    0   1
## 3    0    1   1
## 4    1    1   0
##
## $net.result
## $net.result[[1]]
##             [,1]
## 1 0.027449055524
## 2 0.995548383701
## 3 0.981631230570
## 4 0.001774637076
##
## $net.result[[2]]
##            [,1]
## 1 0.02194157041
## 2 0.97879413178
## 3 0.49519399499
## 4 0.51201651218
##
## $net.result[[3]]
##             [,1]
## 1 0.002671006546
## 2 0.996405219064
## 3 0.990911233075
## 4 0.012698986215
##
## $net.result[[4]]
##             [,1]
## 1 0.008221419974
## 2 1.001822189560
## 3 0.983219747208
## 4 0.001905332014
##
## $net.result[[5]]
##             [,1]
## 1 0.001653315593
## 2 0.992056797509
## 3 0.996594525537
## 4 0.012589649288
##
##
## $weights
## $weights[[1]]
## $weights[[1]][[1]]
##               [,1]         [,2]
## [1,] -0.8598942245 -3.294346742
## [2,] -2.5289455253  4.110777619
## [3,]  2.6112011086 -5.456598655
##
## $weights[[1]][[2]]
##               [,1]
## [1,] -0.6058712109
## [2,]  1.8625960386
## [3,]  2.2215740377
##
##
## $weights[[2]]
## $weights[[2]][[1]]
##              [,1]          [,2]
## [1,] -1.333560270 -0.9792836865
## [2,]  1.379886310  5.6230663769
## [3,] -4.505668181  5.6396154497
##
## $weights[[2]][[2]]
##               [,1]
## [1,] -0.4308913589
## [2,]  0.9509860694
## [3,]  0.9320627076
##
##
## $weights[[3]]
## $weights[[3]][[1]]
##                [,1]         [,2]
## [1,] -0.08878218756  3.124463407
## [2,] -8.47878785897 -2.470394348
## [3,] -8.88380476467 -2.483739325
##
## $weights[[3]][[2]]
##               [,1]
```

```
## [1,] -0.2492284832
## [2,] -3.2701758361
## [3,]  1.8942159395
##
##
## $weights[[4]]
## $weights[[4]][[1]]
##              [,1]         [,2]
## [1,] -2.012864559  1.272435483
## [2,]  2.878858308  4.287534317
## [3,] -2.856134764 -4.245845802
##
## $weights[[4]][[2]]
##              [,1]
## [1,]  1.047263926
## [2,]  2.311338228
## [3,] -1.678856020
##
##
## $weights[[5]]
## $weights[[5]][[1]]
##              [,1]         [,2]
## [1,] -1.212302279 10.089604333
## [2,]  2.602945553 -7.006520182
## [3,]  2.636595220 -7.092493076
##
## $weights[[5]][[2]]
##              [,1]
## [1,] -1.820693088
## [2,]  1.840557638
## [3,]  1.400375811
##
##
##
## $startweights
## $startweights[[1]]
## $startweights[[1]][[1]]
##                [,1]          [,2]
## [1,]  0.59017284127 -0.6829585444
## [2,] -0.54201146229  0.3541873244
## [3,]  0.05791999643  0.2470574541
##
## $startweights[[1]][[2]]
##              [,1]
## [1,] 0.2026000167
## [2,] 1.5353789349
## [3,] 0.3306122763
##
##
## $startweights[[2]]
## $startweights[[2]][[1]]
##                [,1]          [,2]
## [1,] -0.46423315124 -0.6815703774
## [2,] -0.04027369011  1.4546663769
## [3,] -2.42226818224  1.4712154497
##
## $startweights[[2]][[2]]
##                [,1]
## [1,] -1.17825810603
## [2,] -0.72898331071
## [3,] -0.02145678982
##
##
## $startweights[[3]]
## $startweights[[3]][[1]]
##              [,1]          [,2]
## [1,] -1.268216204 -0.8071449326
## [2,] -1.513550739  0.2425929097
## [3,] -1.047927645 -0.3525411191
##
## $startweights[[3]][[2]]
##                [,1]
## [1,] -0.28678521088
## [2,] -0.27082229898
## [3,]  0.06278055498
##
##
## $startweights[[4]]
## $startweights[[4]][[1]]
##               [,1]          [,2]
## [1,] -0.9172638039  0.2905971618
## [2,]  1.4306729438  0.4010077574
## [3,]  0.4346598835 -0.1881771691
##
## $startweights[[4]][[2]]
```

```
##              [,1]
## [1,] 1.5174148783
## [2,] 2.1943846036
## [3,] 0.2520993973
##
##
## $startweights[[5]]
## $startweights[[5]][[1]]
##               [,1]          [,2]
## [1,] -1.3508913094  2.2918628401
## [2,] -0.2765762761 -1.7278529618
## [3,]  0.7393007755 -0.3742651676
##
## $startweights[[5]][[2]]
##               [,1]
## [1,] -0.3959847084
## [2,]  1.7398889164
## [3,] -0.3009613487
##
##
##
## $generalized.weights
## $generalized.weights[[1]]
##            [,1]          [,2]
## 1  -25.06916575   22.40588381
## 2  404.45936706 -546.77193178
## 3  -32.83878438   33.88325224
## 4 -524.74810012  527.41726589
##
## $generalized.weights[[2]]
##             [,1]           [,2]
## 1 58.56857717645  15.6590752597
## 2 18.18042229600 -49.1910333021
## 3  0.20989990485   0.1456703900
## 4  0.06010117456  -0.1931837163
##
## $generalized.weights[[3]]
##            [,1]          [,2]
## 1 2526.16791718 2649.84046064
## 2 -292.55321985 -294.07123614
## 3 -117.03601691 -117.65170128
## 4  -44.47292295  -44.71316155
##
## $generalized.weights[[4]]
##            [,1]         [,2]
## 1  -66.06889759  65.27123694
## 2 -744.63463822 738.78461181
## 3  -17.13868539  16.96645995
## 4 -261.74865121 258.52629241
##
## $generalized.weights[[5]]
##           [,1]         [,2]
## 1 512.68400435 519.31190639
## 2  44.86159943  45.47582464
## 3  89.76929113  91.01583572
## 4  -7.18449752  -7.26829718
##
##
## $result.matrix
##                                     1                2                  3
## error                  0.000556914282  0.258960566395   0.0001319633305
## reached.threshold      0.007076680726  0.007946209356   0.0040029749376
## steps                 97.000000000000 46.000000000000 108.0000000000000
## Intercept.to.1layhid1 -0.859894224464 -1.333560270305  -0.0887821875606
## Var1.to.1layhid1      -2.528945525274  1.379886309895  -8.4787878589719
## Var2.to.1layhid1       2.611201108629 -4.505668181064  -8.8838047646668
## Intercept.to.1layhid2 -3.294346741827 -0.979283686488   3.1244634074240
## Var1.to.1layhid2       4.110777619155  5.623066376869  -2.4703943476266
## Var2.to.1layhid2      -5.456598654974  5.639615449746  -2.4837393253597
## Intercept.to.XOR      -0.605871210880 -0.430891358936  -0.2492284831835
## 1layhid.1.to.XOR       1.862596038564  0.950986069376  -3.2701758361314
## 1layhid.2.to.XOR       2.221574037671  0.932062707579   1.8942159395499
##                                     4                5
## error                  0.0001780596475   0.0001179622219
## reached.threshold      0.0089230102368   0.0089611249206
## steps                 96.0000000000000 159.0000000000000
## Intercept.to.1layhid1 -2.0128645588702  -1.2123022793666
## Var1.to.1layhid1       2.8788583075218   2.6029455529488
## Var2.to.1layhid1      -2.8561347642390   2.6365952201150
## Intercept.to.1layhid2  1.2724354832125  10.0896043326653
## Var1.to.1layhid2       4.2875343174261  -7.0065201820761
## Var2.to.1layhid2      -4.2458458019465  -7.0924930759777
## Intercept.to.XOR       1.0472639263810  -1.8206930875574
## 1layhid.1.to.XOR       2.3113382277714   1.8405576378795
## 1layhid.2.to.XOR      -1.6788560195108   1.4003758111994
##
```

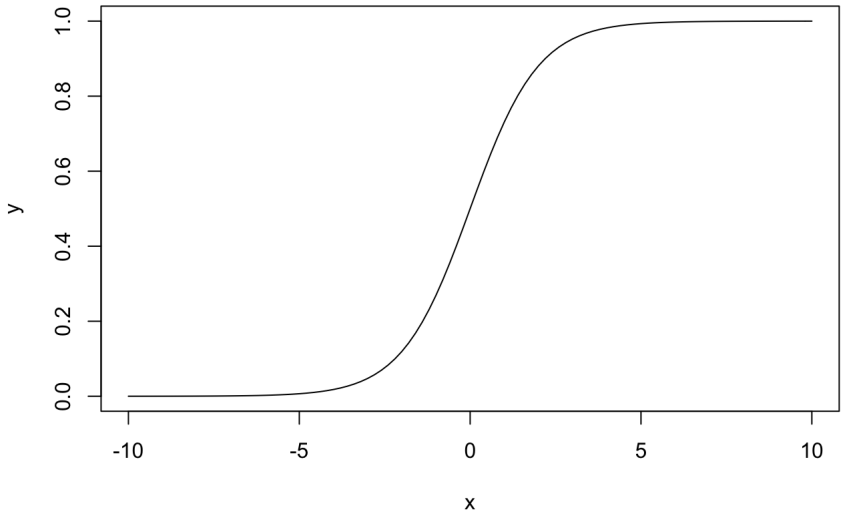
```
##
## attr(,"class")
## [1] "nn"
```

```
plot(net.xor, rep="best")
```



As we can observe above, this is a very simple NN network representing a NOR gate (not or gate). Let's focus on the nodes and the arrows first. I will explain the numbers and other things later. As you can see from the graph, we observe there are two inputs: Var1 and Var2, they are being passed into two nodes, and each node will calculate the output by a function called activation function (ie 1/exp(-x)) that map our input into a value between [0, 1]:

```
curve(1/ (1 + exp(-x)), from=-10, to=10, , xlab="x", ylab="y")
```



Finally, we all the computations we operate above, we have the output that is generated by all the nodes above represent the boolean value of the XOR gate. In the next example, I will build a more complex NN network to do something even more useful.

Training a more complex neural network(NN)

In this case, we are going to build a NN for predicting a function that map a real number to a real number (ie. sin(x)):

```
sins <- vector(length = 11)
for (i in 1:length(sins)) {
  sins[i] = sin(0.01 * i)
}
sins.data <- data.frame(x = seq(0, 1, 0.1), values = sins)
print(net.sin <- neuralnet(values~x, sins.data, hidden=c(4, 2)))
```

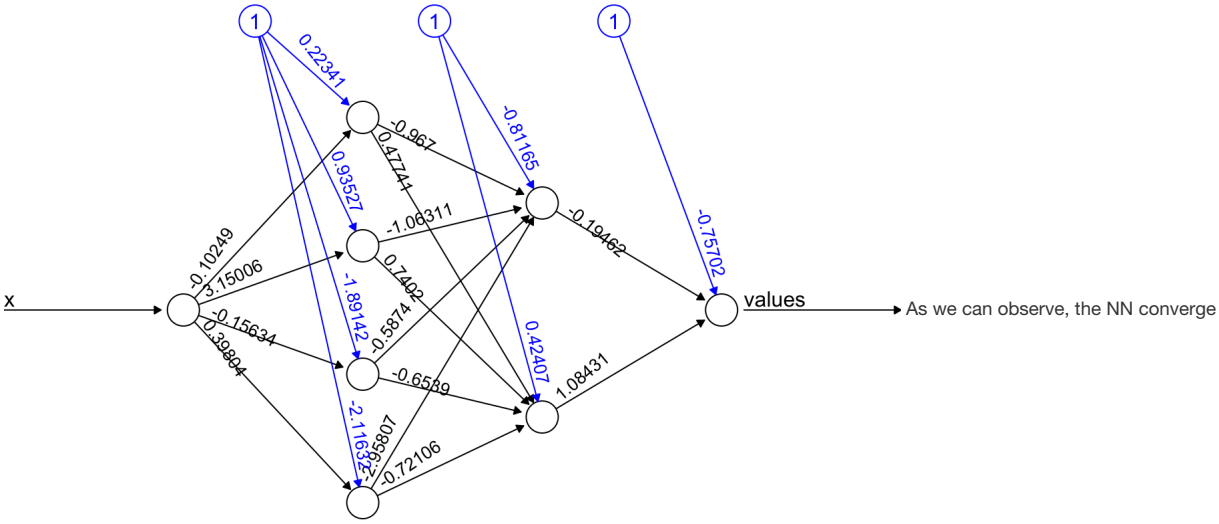
```
## $call
## neuralnet(formula = values ~ x, data = sins.data, hidden = c(4,
##     2))
##
## $
```

```
## $response
##            values
## 1  0.009999833334
## 2  0.019998666693
## 3  0.029995500202
## 4  0.039989334187
## 5  0.049979169271
## 6  0.059964006479
## 7  0.069942847338
## 8  0.079914693969
## 9  0.089878549198
## 10 0.099833416647
## 11 0.109778300837
##
## $covariate
##       [,1]
##  [1,]  0.0
##  [2,]  0.1
##  [3,]  0.2
##  [4,]  0.3
##  [5,]  0.4
##  [6,]  0.5
##  [7,]  0.6
##  [8,]  0.7
##  [9,]  0.8
## [10,]  0.9
## [11,]  1.0
##
## $model.list
## $model.list$response
## [1] "values"
##
## $model.list$variables
## [1] "x"
##
##
## $err.fct
## function (x, y)
## {
##     1/2 * (y - x)^2
## }
## <environment: 0x7fe1bb219e28>
## attr(,"type")
## [1] "sse"
##
## $act.fct
## function (x)
## {
##     1/(1 + exp(-x))
## }
## <environment: 0x7fe1bb219e28>
## attr(,"type")
## [1] "logistic"
##
## $linear.output
## [1] TRUE
##
## $data
##      x         values
## 1  0.0 0.009999833334
## 2  0.1 0.019998666693
## 3  0.2 0.029995500202
## 4  0.3 0.039989334187
## 5  0.4 0.049979169271
## 6  0.5 0.059964006479
## 7  0.6 0.069942847338
## 8  0.7 0.079914693969
## 9  0.8 0.089878549198
## 10 0.9 0.099833416647
## 11 1.0 0.109778300837
##
## $net.result
## $net.result[[1]]
##             [,1]
## 1  0.03314499338
## 2  0.04249355017
## 3  0.05008513283
## 4  0.05604680133
## 5  0.06058844878
## 6  0.06394783864
## 7  0.06635464749
## 8  0.06801160591
## 9  0.06908768362
## 10 0.06971837456
## 11 0.07000944761
##
```

```
""
##
## $weights
## $weights[[1]]
## $weights[[1]][[1]]
##               [,1]         [,2]         [,3]          [,4]
## [1,]  0.2234117501 0.9352688923 -1.891421078 -2.1163172301
## [2,] -0.1024900084 3.1500559366 -0.156339062  0.3980440665
##
## $weights[[1]][[2]]
##               [,1]          [,2]
## [1,] -0.8116475750  0.4240716929
## [2,] -0.9669974918  0.4774133800
## [3,] -1.0631080197  0.7402018387
## [4,] -0.5874000370 -0.6538994298
## [5,] -2.9580730512 -0.7210593816
##
## $weights[[1]][[3]]
##               [,1]
## [1,] -0.7570180457
## [2,] -0.1946190271
## [3,]  1.0843147909
##
##
##
## $startweights
## $startweights[[1]]
## $startweights[[1]][[1]]
##               [,1]         [,2]          [,3]          [,4]
## [1,] -0.2669382499 0.5295147739 -1.4030960785 -1.9351610701
## [2,] -0.6612358900 2.6597059366  0.4024068196  0.6223411012
##
## $startweights[[1]][[2]]
##                [,1]           [,2]
## [1,] -0.32332257496 -0.06627830705
## [2,] -0.47867249180 -0.01293662000
## [3,] -0.57275801968  0.24985183875
## [4,] -0.09907503695 -1.14222442979
## [5,] -2.46772305117 -1.21988738159
##
## $startweights[[1]][[3]]
##               [,1]
## [1,] -1.2473680457
## [2,] -0.6829440271
## [3,]  0.5939647909
##
##
##
## $generalized.weights
## $generalized.weights[[1]]
##             [,1]
## 1  3.19388312017
## 2  2.07902222068
## 3  1.41801138661
## 4  0.98530874641
## 5  0.68707945857
## 6  0.47556752143
## 7  0.32296625065
## 8  0.21159773705
## 9  0.12958742456
## 10 0.06868705737
## 11 0.02305000301
##
##
## $result.matrix
##                                    1
## error                  0.002453438879
## reached.threshold      0.009785793841
## steps                 25.000000000000
## Intercept.to.1layhid1  0.223411750051
## x.to.1layhid1         -0.102490008446
## Intercept.to.1layhid2  0.935268892252
## x.to.1layhid2          3.150055936637
## Intercept.to.1layhid3 -1.891421078456
## x.to.1layhid3         -0.156339062010
## Intercept.to.1layhid4 -2.116317230075
## x.to.1layhid4          0.398044066471
## Intercept.to.2layhid1 -0.811647574960
## 1layhid.1.to.2layhid1 -0.966997491802
## 1layhid.2.to.2layhid1 -1.063108019679
## 1layhid.3.to.2layhid1 -0.587400036952
## 1layhid.4.to.2layhid1 -2.958073051173
## Intercept.to.2layhid2  0.424071692946
## 1layhid.1.to.2layhid2  0.477413379999
## 1layhid.2.to.2layhid2  0.740201838747
## 1layhid.3.to.2layhid2 -0.653899429788
```

```
## 1layhid.4.to.2layhid2 -0.721059381593
## Intercept.to.values   -0.757018045712
## 2layhid.1.to.values   -0.194619027107
## 2layhid.2.to.values    1.084314790917
##
## attr(,"class")
## [1] "nn"
```

```
plot(net.sin, rep="best")
```



in 19 steps (it is able to predict the correct value)

How can we improve its performance? There is generally two ways: 1. increase its layer or give mroe node (sometime will decrease the performance which is called “overfitting”)

```
sins <- vector(length = 11)
for (i in 1:length(sins)) {
  sins[i] = sin(0.01 * i)
}
sins.data <- data.frame(x = seq(0, 1, 0.1), values = sins)
print(net.sin <- neuralnet(values~x, sins.data, hidden=c(10, 5)))
```

```
## $call
## neuralnet(formula = values ~ x, data = sins.data, hidden = c(10,
##     5))
##
## $response
##             values
## 1   0.009999833334
## 2   0.019998666693
## 3   0.029995500202
## 4   0.039989334187
## 5   0.049979169271
## 6   0.059964006479
## 7   0.069942847338
## 8   0.079914693969
## 9   0.089878549198
## 10 0.099833416647
## 11 0.109778300837
##
## $covariate
##       [,1]
##  [1,]  0.0
##  [2,]  0.1
##  [3,]  0.2
##  [4,]  0.3
##  [5,]  0.4
##  [6,]  0.5
##  [7,]  0.6
##  [8,]  0.7
##  [9,]  0.8
## [10,]  0.9
## [11,]  1.0
##
## $model.list
## $model.list$response
## [1] "values"
##
```

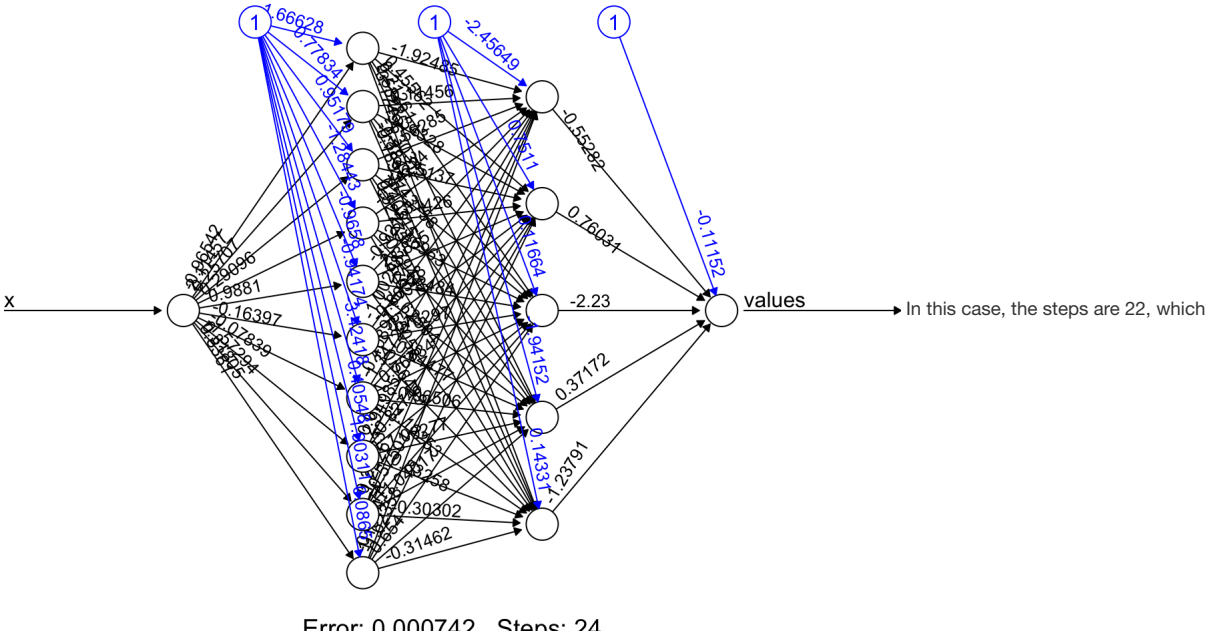
```
""
## $model.list$variables
## [1] "x"
##
##
## $err.fct
## function (x, y)
## {
##     1/2 * (y - x)^2
## }
## <environment: 0x7fe1ba7bdce0>
## attr(,"type")
## [1] "sse"
##
## $act.fct
## function (x)
## {
##     1/(1 + exp(-x))
## }
## <environment: 0x7fe1ba7bdce0>
## attr(,"type")
## [1] "logistic"
##
## $linear.output
## [1] TRUE
##
## $data
##      x         values
## 1  0.0 0.009999833334
## 2  0.1 0.019998666693
## 3  0.2 0.029995500202
## 4  0.3 0.039989334187
## 5  0.4 0.049979169271
## 6  0.5 0.059964006479
## 7  0.6 0.069942847338
## 8  0.7 0.079914693969
## 9  0.8 0.089878549198
## 10 0.9 0.099833416647
## 11 1.0 0.109778300837
##
## $net.result
## $net.result[[1]]
##               [,1]
## 1  -0.006145297272
## 2   0.006215486010
## 3   0.019188848905
## 4   0.032636748047
## 5   0.046418860966
## 6   0.060394914798
## 7   0.074428023089
## 8   0.088388412355
## 9   0.102156902842
## 10  0.115627648698
## 11  0.128709864927
##
##
## $weights
## $weights[[1]]
## $weights[[1]][[1]]
##               [,1]         [,2]         [,3]          [,4]          [,5]
## [1,] -1.6662815032 0.7783389572 0.9517919145 -1.2844268622 -0.9657970006
## [2,] -0.9654217561 2.1145705964 0.2570710910  0.2909624472  0.9881007828
##               [,6]           [,7]         [,8]         [,9]         [,10]
## [1,] -0.9417377128 -3.42418350916  0.405478725 1.9031073098 0.08650054589
## [2,] -0.1639713231 -0.07839014899 -1.572941199 0.3180506272 0.83609871109
##
## $weights[[1]][[2]]
##                [,1]          [,2]          [,3]           [,4]
##  [1,] -2.45648808474  0.7510999141  0.1166410163 -1.94152417226
##  [2,] -1.92484868282  0.4556219716  0.2126139286  0.53285046074
##  [3,]  0.11456341301  1.3432822601 -1.3031243420 -0.88490067829
##  [4,] -1.56284613950  0.2513694698 -1.4375757566  0.40314721164
##  [5,] -0.54223602780  1.3142587621  0.4696295956 -0.79442083060
##  [6,] -0.92150791157  2.3834986681  0.4248375346 -0.62504492407
##  [7,] -1.00298063303 -0.8032772108  0.1629715590 -0.64476753651
##  [8,] -0.40896982146  1.3472349923  0.7673436564 -0.06505559427
##  [9,]  0.07409860535  2.9498519706 -0.8274690301  1.09276614279
## [10,]  0.24123143281  1.2426012774 -0.5750085999 -1.04016593504
## [11,] -0.47474291102  1.1614660571 -1.6235826252  0.65400417553
##                [,5]
##  [1,]  0.1433096977
##  [2,] -3.1749520928
##  [3,] -0.1893117781
##  [4,]  0.5277461742
##  [5,]  1.2370168066
##  [6,] -1.5505186167
```

```
##  [7,]  0.5649933905
##  [8,] -1.1225267358
##  [9,]  0.8325761563
## [10,] -0.3030220119
## [11,] -0.3146249309
##
## $weights[[1]][[3]]
##               [,1]
## [1,] -0.1115248675
## [2,] -0.5528217024
## [3,]  0.7603106816
## [4,] -2.2299961537
## [5,]  0.3717217501
## [6,] -1.2379074178
##
##
##
## $startweights
## $startweights[[1]]
## $startweights[[1]][[1]]
##               [,1]           [,2]          [,3]          [,4]
## [1,] -1.6058915032 -0.07044824277  0.4585106645 -0.6433018622
## [2,] -0.8934217561  1.48957059640 -0.2618819090  0.8935224472
##               [,5]          [,6]         [,7]          [,8]         [,9]
## [1,] -0.7404720006 -0.3006127128 -3.044746009  1.0466037250 1.7855753098
## [2,]  1.2503931028  0.4385886769  0.320329851 -0.9703811992 0.2338387872
##              [,10]
## [1,] -0.5546244541
## [2,]  0.2335387111
##
## $startweights[[1]][[2]]
##                [,1]          [,2]           [,3]            [,4]
##  [1,] -1.8153630847  0.1099749141  0.75776601633 -1.881134172263
##  [2,] -1.0760614828 -0.3931652284  1.06140112856  0.443723676745
##  [3,]  0.7556884130  0.7021572601 -0.66199934200 -0.824510678289
##  [4,] -0.9217211395 -0.3897555302 -0.79645075657  0.463537211644
##  [5,]  0.0988889722  0.6731337621  1.11075459562 -0.734030830595
##  [6,] -0.2803829116  1.7423736681  1.06596253463 -0.564654924071
##  [7,] -0.3618556330 -1.4444022108  0.80409655903 -0.584377536507
##  [8,]  0.2321551785  0.7061099923  1.40846865640 -0.004665594274
##  [9,]  0.9228858054  2.1010647706  0.02131816989  1.003639358794
## [10,]  0.8823564328  0.6014762774  0.06611640013 -0.979775935037
## [11,]  0.1663820890  0.5203410571 -0.98245762520  0.714394175533
##                [,5]
##  [1,]  0.7844346977
##  [2,] -2.3261648928
##  [3,]  0.4518132219
##  [4,]  1.1688711742
##  [5,]  1.8781418066
##  [6,] -0.9093936167
##  [7,]  1.2061183905
##  [8,] -0.4814017358
##  [9,]  1.6813633563
## [10,]  0.3381029881
## [11,]  0.3265000691
##
## $startweights[[1]][[3]]
##               [,1]
## [1,] -0.7526498675
## [2,] -1.1939467024
## [3,]  0.1191856816
## [4,] -2.8711211537
## [5,] -0.2694032499
## [6,] -1.8790324178
##
##
##
## $generalized.weights
## $generalized.weights[[1]]
##             [,1]
## 1  -19.422873745
## 2   20.544147021
## 3    7.031581352
## 4    4.319860689
## 5    3.140758919
## 6    2.471819806
## 7    2.034824811
## 8    1.723004943
## 9    1.486689296
## 10   1.299685120
## 11   1.146888218
##
##
## $result.matrix
##
```

```
## error                   0.0007424437912
## reached.threshold       0.0087460952087
## steps                  24.0000000000000
## Intercept.to.1layhid1  -1.6662815032236
## x.to.1layhid1          -0.9654217560574
## Intercept.to.1layhid2   0.7783389572323
## x.to.1layhid2           2.1145705963972
## Intercept.to.1layhid3   0.9517919144640
## x.to.1layhid3           0.2570710909818
## Intercept.to.1layhid4  -1.2844268622363
## x.to.1layhid4           0.2909624472361
## Intercept.to.1layhid5  -0.9657970005518
## x.to.1layhid5           0.9881007827819
## Intercept.to.1layhid6  -0.9417377127914
## x.to.1layhid6          -0.1639713230752
## Intercept.to.1layhid7  -3.4241835091602
## x.to.1layhid7          -0.0783901489861
## Intercept.to.1layhid8   0.4054787249649
## x.to.1layhid8          -1.5729411992442
## Intercept.to.1layhid9   1.9031073097642
## x.to.1layhid9           0.3180506271960
## Intercept.to.1layhid10  0.0865005458885
## x.to.1layhid10          0.8360987110854
## Intercept.to.2layhid1  -2.4564880847423
## 1layhid.1.to.2layhid1  -1.9248486828222
## 1layhid.2.to.2layhid1   0.1145634130070
## 1layhid.3.to.2layhid1  -1.5628461394956
## 1layhid.4.to.2layhid1  -0.5422360277972
## 1layhid.5.to.2layhid1  -0.9215079115697
## 1layhid.6.to.2layhid1  -1.0029806330347
## 1layhid.7.to.2layhid1  -0.4089698214626
## 1layhid.8.to.2layhid1   0.0740986053503
## 1layhid.9.to.2layhid1   0.2412314328104
## 1layhid.10.to.2layhid1 -0.4747429110228
## Intercept.to.2layhid2   0.7510999140874
## 1layhid.1.to.2layhid2   0.4556219715839
## 1layhid.2.to.2layhid2   1.3432822600565
## 1layhid.3.to.2layhid2   0.2513694698463
## 1layhid.4.to.2layhid2   1.3142587620807
## 1layhid.5.to.2layhid2   2.3834986681191
## 1layhid.6.to.2layhid2  -0.8032772107590
## 1layhid.7.to.2layhid2   1.3472349923393
## 1layhid.8.to.2layhid2   2.9498519706356
## 1layhid.9.to.2layhid2   1.2426012774287
## 1layhid.10.to.2layhid2  1.1614660571235
## Intercept.to.2layhid3   0.1166410163263
## 1layhid.1.to.2layhid3   0.2126139285641
## 1layhid.2.to.2layhid3  -1.3031243420008
## 1layhid.3.to.2layhid3  -1.4375757565698
## 1layhid.4.to.2layhid3   0.4696295956215
## 1layhid.5.to.2layhid3   0.4248375346316
## 1layhid.6.to.2layhid3   0.1629715590285
## 1layhid.7.to.2layhid3   0.7673436564027
## 1layhid.8.to.2layhid3  -0.8274690301076
## 1layhid.9.to.2layhid3  -0.5750085998684
## 1layhid.10.to.2layhid3 -1.6235826251973
## Intercept.to.2layhid4  -1.9415241722629
## 1layhid.1.to.2layhid4   0.5328504607448
## 1layhid.2.to.2layhid4  -0.8849006782889
## 1layhid.3.to.2layhid4   0.4031472116436
## 1layhid.4.to.2layhid4  -0.7944208305952
## 1layhid.5.to.2layhid4  -0.6250449240715
## 1layhid.6.to.2layhid4  -0.6447675365065
## 1layhid.7.to.2layhid4  -0.0650555942739
## 1layhid.8.to.2layhid4   1.0927661427943
## 1layhid.9.to.2layhid4  -1.0401659350370
## 1layhid.10.to.2layhid4  0.6540041755331
## Intercept.to.2layhid5   0.1433096976734
## 1layhid.1.to.2layhid5  -3.1749520928450
## 1layhid.2.to.2layhid5  -0.1893117781404
## 1layhid.3.to.2layhid5   0.5277461742387
## 1layhid.4.to.2layhid5   1.2370168065801
## 1layhid.5.to.2layhid5  -1.5505186167444
## 1layhid.6.to.2layhid5   0.5649933904585
## 1layhid.7.to.2layhid5  -1.1225267357899
## 1layhid.8.to.2layhid5   0.8325761562521
## 1layhid.9.to.2layhid5  -0.3030220119450
## 1layhid.10.to.2layhid5 -0.3146249308813
## Intercept.to.values    -0.1115248675409
## 2layhid.1.to.values    -0.5528217024333
## 2layhid.2.to.values     0.7603106815762
## 2layhid.3.to.values    -2.2299961536732
## 2layhid.4.to.values     0.3717217500719
## 2layhid.5.to.values    -1.2379074178356
##
```

```
## attr(,"class")
## [1] "nn"
```

```
plot(net.sin, rep="best")
```



In this case, the steps are 22, which is a little increase

2. Give more examples (This is always good)

```
sins <- vector(length = 101)
for (i in 1:length(sins)) {
  sins[i] = sin(0.01 * i)
}
sins.data <- data.frame(x = seq(0, 1, 0.01), values = sins)
print(net.sin <- neuralnet(values~x, sins.data, hidden=c(4, 2)))
```

```
## $call
## neuralnet(formula = values ~ x, data = sins.data, hidden = c(4,
##     2))
##
## $response
##              values
## 1   0.009999833334
## 2   0.019998666693
## 3   0.029995500202
## 4   0.039989334187
## 5   0.049979169271
## 6   0.059964006479
## 7   0.069942847338
## 8   0.079914693969
## 9   0.089878549198
## 10  0.099833416647
## 11  0.109778300837
## 12  0.119712207289
## 13  0.129634142620
## 14  0.139543114644
## 15  0.149438132474
## 16  0.159318206614
## 17  0.169182349067
## 18  0.179029573426
## 19  0.188858894977
## 20  0.198669330795
## 21  0.208459899846
## 22  0.218229623081
## 23  0.227977523535
## 24  0.237702626427
## 25  0.247403959255
## 26  0.257080551892
## 27  0.266731436689
## 28  0.276355648564
## 29  0.285952225105
## 30  0.295520206661
## 31  0.305058636443
## 32  0.314566560616
## 33  0.324043028395
## 34  0.333487092141
## 35  0.342897807455
## 36  0.352274233275
```

```
## 37  0.361615431965
## 38  0.370920469413
## 39  0.380188415123
## 40  0.389418342309
## 41  0.398609327984
## 42  0.407760453060
## 43  0.416870802429
## 44  0.425939465066
## 45  0.434965534111
## 46  0.443948106966
## 47  0.452886285379
## 48  0.461779175541
## 49  0.470625888171
## 50  0.479425538604
## 51  0.488177246883
## 52  0.496880137844
## 53  0.505533341205
## 54  0.514135991653
## 55  0.522687228931
## 56  0.531186197921
## 57  0.539632048734
## 58  0.548023936792
## 59  0.556361022913
## 60  0.564642473395
## 61  0.572867460100
## 62  0.581035160537
## 63  0.589144757942
## 64  0.597195441362
## 65  0.605186405736
## 66  0.613116851973
## 67  0.620985987037
## 68  0.628793024018
## 69  0.636537182222
## 70  0.644217687238
## 71  0.651833771022
## 72  0.659384671971
## 73  0.666869635004
## 74  0.674287911628
## 75  0.681638760023
## 76  0.688921445111
## 77  0.696135238627
## 78  0.703279419200
## 79  0.710353272418
## 80  0.717356090900
## 81  0.724287174370
## 82  0.731145829727
## 83  0.737931371110
## 84  0.744643119971
## 85  0.751280405140
## 86  0.757842562895
## 87  0.764328937026
## 88  0.770738878899
## 89  0.777071747527
## 90  0.783326909627
## 91  0.789503739690
## 92  0.795601620036
## 93  0.801619940884
## 94  0.807558100405
## 95  0.813415504789
## 96  0.819191568301
## 97  0.824885713338
## 98  0.830497370492
## 99  0.836025978601
## 100 0.841470984808
## 101 0.846831844618
##
## $covariate
##        [,1]
##   [1,] 0.00
##   [2,] 0.01
##   [3,] 0.02
##   [4,] 0.03
##   [5,] 0.04
##   [6,] 0.05
##   [7,] 0.06
##   [8,] 0.07
##   [9,] 0.08
##  [10,] 0.09
##  [11,] 0.10
##  [12,] 0.11
##  [13,] 0.12
##  [14,] 0.13
##  [15,] 0.14
##  [16,] 0.15
##  [17,] 0.16
##  [18,] 0.17
```

```
##  [18,] 0.17
##  [19,] 0.18
##  [20,] 0.19
##  [21,] 0.20
##  [22,] 0.21
##  [23,] 0.22
##  [24,] 0.23
##  [25,] 0.24
##  [26,] 0.25
##  [27,] 0.26
##  [28,] 0.27
##  [29,] 0.28
##  [30,] 0.29
##  [31,] 0.30
##  [32,] 0.31
##  [33,] 0.32
##  [34,] 0.33
##  [35,] 0.34
##  [36,] 0.35
##  [37,] 0.36
##  [38,] 0.37
##  [39,] 0.38
##  [40,] 0.39
##  [41,] 0.40
##  [42,] 0.41
##  [43,] 0.42
##  [44,] 0.43
##  [45,] 0.44
##  [46,] 0.45
##  [47,] 0.46
##  [48,] 0.47
##  [49,] 0.48
##  [50,] 0.49
##  [51,] 0.50
##  [52,] 0.51
##  [53,] 0.52
##  [54,] 0.53
##  [55,] 0.54
##  [56,] 0.55
##  [57,] 0.56
##  [58,] 0.57
##  [59,] 0.58
##  [60,] 0.59
##  [61,] 0.60
##  [62,] 0.61
##  [63,] 0.62
##  [64,] 0.63
##  [65,] 0.64
##  [66,] 0.65
##  [67,] 0.66
##  [68,] 0.67
##  [69,] 0.68
##  [70,] 0.69
##  [71,] 0.70
##  [72,] 0.71
##  [73,] 0.72
##  [74,] 0.73
##  [75,] 0.74
##  [76,] 0.75
##  [77,] 0.76
##  [78,] 0.77
##  [79,] 0.78
##  [80,] 0.79
##  [81,] 0.80
##  [82,] 0.81
##  [83,] 0.82
##  [84,] 0.83
##  [85,] 0.84
##  [86,] 0.85
##  [87,] 0.86
##  [88,] 0.87
##  [89,] 0.88
##  [90,] 0.89
##  [91,] 0.90
##  [92,] 0.91
##  [93,] 0.92
##  [94,] 0.93
##  [95,] 0.94
##  [96,] 0.95
##  [97,] 0.96
##  [98,] 0.97
##  [99,] 0.98
## [100,] 0.99
## [101,] 1.00
##
## $model.list
```

```
## $model.list$response
## [1] "values"
##
## $model.list$variables
## [1] "x"
##
##
## $err.fct
## function (x, y)
## {
##     1/2 * (y - x)^2
## }
## <environment: 0x7fe1bc476ed8>
## attr(,"type")
## [1] "sse"
##
## $act.fct
## function (x)
## {
##     1/(1 + exp(-x))
## }
## <environment: 0x7fe1bc476ed8>
## attr(,"type")
## [1] "logistic"
##
## $linear.output
## [1] TRUE
##
## $data
##        x         values
## 1   0.00 0.009999833334
## 2   0.01 0.019998666693
## 3   0.02 0.029995500202
## 4   0.03 0.039989334187
## 5   0.04 0.049979169271
## 6   0.05 0.059964006479
## 7   0.06 0.069942847338
## 8   0.07 0.079914693969
## 9   0.08 0.089878549198
## 10  0.09 0.099833416647
## 11  0.10 0.109778300837
## 12  0.11 0.119712207289
## 13  0.12 0.129634142620
## 14  0.13 0.139543114644
## 15  0.14 0.149438132474
## 16  0.15 0.159318206614
## 17  0.16 0.169182349067
## 18  0.17 0.179029573426
## 19  0.18 0.188858894977
## 20  0.19 0.198669330795
## 21  0.20 0.208459899846
## 22  0.21 0.218229623081
## 23  0.22 0.227977523535
## 24  0.23 0.237702626427
## 25  0.24 0.247403959255
## 26  0.25 0.257080551892
## 27  0.26 0.266731436689
## 28  0.27 0.276355648564
## 29  0.28 0.285952225105
## 30  0.29 0.295520206661
## 31  0.30 0.305058636443
## 32  0.31 0.314566560616
## 33  0.32 0.324043028395
## 34  0.33 0.333487092141
## 35  0.34 0.342897807455
## 36  0.35 0.352274233275
## 37  0.36 0.361615431965
## 38  0.37 0.370920469413
## 39  0.38 0.380188415123
## 40  0.39 0.389418342309
## 41  0.40 0.398609327984
## 42  0.41 0.407760453060
## 43  0.42 0.416870802429
## 44  0.43 0.425939465066
## 45  0.44 0.434965534111
## 46  0.45 0.443948106966
## 47  0.46 0.452886285379
## 48  0.47 0.461779175541
## 49  0.48 0.470625888171
## 50  0.49 0.479425538604
## 51  0.50 0.488177246883
## 52  0.51 0.496880137844
## 53  0.52 0.505533341205
## 54  0.53 0.514135991653
## 55  0.54 0.522687228931
```

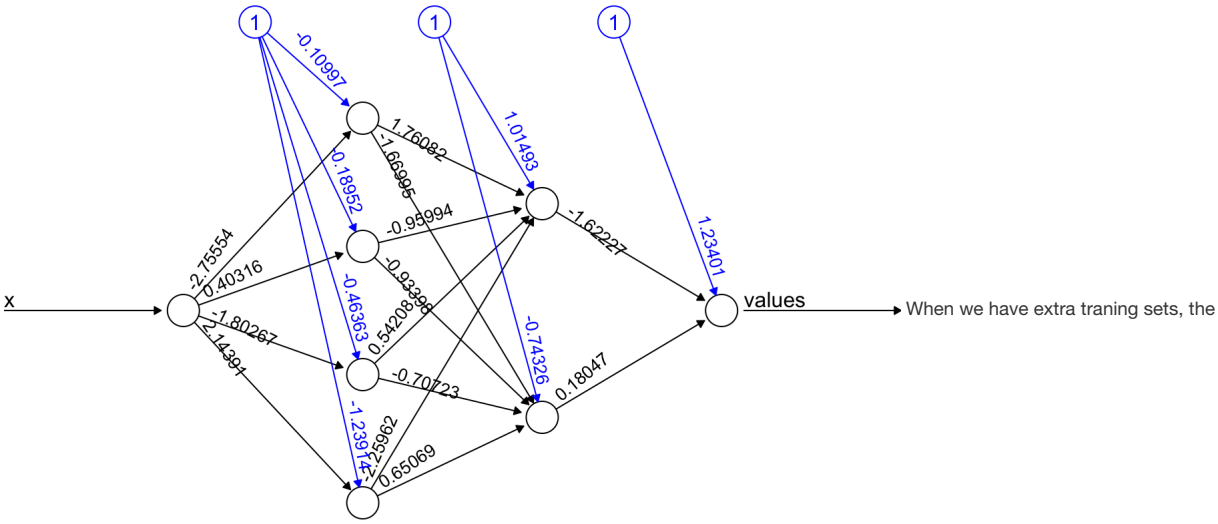
```
## 56  0.55 0.531186197921
## 57  0.56 0.539632048734
## 58  0.57 0.548023936792
## 59  0.58 0.556361022913
## 60  0.59 0.564642473395
## 61  0.60 0.572867460100
## 62  0.61 0.581035160537
## 63  0.62 0.589144757942
## 64  0.63 0.597195441362
## 65  0.64 0.605186405736
## 66  0.65 0.613116851973
## 67  0.66 0.620985987037
## 68  0.67 0.628793024018
## 69  0.68 0.636537182222
## 70  0.69 0.644217687238
## 71  0.70 0.651833771022
## 72  0.71 0.659384671971
## 73  0.72 0.666869635004
## 74  0.73 0.674287911628
## 75  0.74 0.681638760023
## 76  0.75 0.688921445111
## 77  0.76 0.696135238627
## 78  0.77 0.703279419200
## 79  0.78 0.710353272418
## 80  0.79 0.717356090900
## 81  0.80 0.724287174370
## 82  0.81 0.731145829727
## 83  0.82 0.737931371110
## 84  0.83 0.744643119971
## 85  0.84 0.751280405140
## 86  0.85 0.757842562895
## 87  0.86 0.764328937026
## 88  0.87 0.770738878899
## 89  0.88 0.777071747527
## 90  0.89 0.783326909627
## 91  0.90 0.789503739690
## 92  0.91 0.795601620036
## 93  0.92 0.801619940884
## 94  0.93 0.807558100405
## 95  0.94 0.813415504789
## 96  0.95 0.819191568301
## 97  0.96 0.824885713338
## 98  0.97 0.830497370492
## 99  0.98 0.836025978601
## 100 0.99 0.841470984808
## 101 1.00 0.846831844618
##
## $net.result
## $net.result[[1]]
##              [,1]
## 1   0.03259390156
## 2   0.04011833769
## 3   0.04775322475
## 4   0.05549707209
## 5   0.06334823908
## 6   0.07130493532
## 7   0.07936522130
## 8   0.08752700958
## 9   0.09578806634
## 10  0.10414601349
## 11  0.11259833127
## 12  0.12114236120
## 13  0.12977530964
## 14  0.13849425167
## 15  0.14729613555
## 16  0.15617778743
## 17  0.16513591662
## 18  0.17416712115
## 19  0.18326789376
## 20  0.19243462813
## 21  0.20166362552
## 22  0.21095110163
## 23  0.22029319375
## 24  0.22968596807
## 25  0.23912542726
## 26  0.24860751813
## 27  0.25812813945
## 28  0.26768314986
## 29  0.27726837582
## 30  0.28687961959
## 31  0.29651266721
## 32  0.30616329637
## 33  0.31582728435
## 34  0.32550041568
## 35  0.33517848981
```

```
## 36  0.34485732855
## 37  0.35453278334
## 38  0.36420074229
## 39  0.37385713707
## 40  0.38349794944
## 41  0.39311921759
## 42  0.40271704216
## 43  0.41228759195
## 44  0.42182710940
## 45  0.43133191563
## 46  0.44079841523
## 47  0.45022310068
## 48  0.45960255646
## 49  0.46893346282
## 50  0.47821259911
## 51  0.48743684693
## 52  0.49660319278
## 53  0.50570873050
## 54  0.51475066326
## 55  0.52372630536
## 56  0.53263308354
## 57  0.54146853817
## 58  0.55023032395
## 59  0.55891621049
## 60  0.56752408247
## 61  0.57605193962
## 62  0.58449789639
## 63  0.59286018142
## 64  0.60113713671
## 65  0.60932721667
## 66  0.61742898686
## 67  0.62544112259
## 68  0.63336240734
## 69  0.64119173100
## 70  0.64892808794
## 71  0.65657057498
## 72  0.66411838920
## 73  0.67157082562
## 74  0.67892727482
## 75  0.68618722044
## 76  0.69335023659
## 77  0.70041598518
## 78  0.70738421328
## 79  0.71425475027
## 80  0.72102750514
## 81  0.72770246357
## 82  0.73427968514
## 83  0.74075930046
## 84  0.74714150829
## 85  0.75342657269
## 86  0.75961482017
## 87  0.76570663683
## 88  0.77170246558
## 89  0.77760280334
## 90  0.78340819829
## 91  0.78911924717
## 92  0.79473659260
## 93  0.80026092047
## 94  0.80569295736
## 95  0.81103346805
## 96  0.81628325302
## 97  0.82144314610
## 98  0.82651401211
## 99  0.83149674459
## 100 0.83639226359
## 101 0.84120151357
##
##
## $weights
## $weights[[1]]
## $weights[[1]][[1]]
##              [,1]          [,2]          [,3]         [,4]
## [1,] -0.1099651408 -0.1895207070 -0.4636258916 -1.239140252
## [2,] -2.7555392920  0.4031569714 -1.8026739094  2.143912555
##
## $weights[[1]][[2]]
##              [,1]          [,2]
## [1,]  1.0149300638 -0.7432607342
## [2,]  1.7608175294 -1.6699526413
## [3,] -0.9599411097 -0.9339768496
## [4,]  0.5420822303 -0.7072305816
## [5,] -2.2596164615  0.6506938787
##
## $weights[[1]][[3]]
##              [,1]
```

```
## [1,]  1.234012818
## [2,] -1.622269970
## [3,]  0.180473419
##
##
##
## $startweights
## $startweights[[1]]
## $startweights[[1]][[1]]
##               [,1]          [,2]          [,3]          [,4]
## [1,] -0.4608935735 -0.1287265838 -0.5759274307 -1.3326321284
## [2,] -1.4957791000 -0.8566032206 -0.5429137174  0.8903109547
##
## $startweights[[1]][[2]]
##               [,1]          [,2]
## [1,]  0.9985952988 -0.7566059217
## [2,]  0.4044165694 -0.7694745613
## [3,] -1.0032392329 -0.9407595371
## [4,]  0.1877684658 -0.5937535800
## [5,] -1.4961809415 -0.1469848413
##
## $startweights[[1]][[3]]
##                [,1]
## [1,]  1.29507987857
## [2,] -1.50879296850
## [3,]  0.04041509584
##
##
##
## $generalized.weights
## $generalized.weights[[1]]
##             [,1]
## 1   23.686605284
## 2   19.683517513
## 3   16.910370521
## 4   14.876416464
## 5   13.321328956
## 6   12.094172823
## 7   11.101408023
## 8   10.281976824
## 9    9.594330842
## 10   9.009215658
## 11   8.505437173
## 12   8.067264767
## 13   7.682777670
## 14   7.342777663
## 15   7.040054055
## 16   6.768874598
## 17   6.524625258
## 18   6.303550423
## 19   6.102562334
## 20   5.919099127
## 21   5.751017609
## 22   5.596511250
## 23   5.454046721
## 24   5.322314271
## 25   5.200188543
## 26   5.086697354
## 27   4.980996608
## 28   4.882349990
## 29   4.790112400
## 30   4.703716359
## 31   4.622660768
## 32   4.546501576
## 33   4.474843973
## 34   4.407335837
## 35   4.343662201
## 36   4.283540553
## 37   4.226716848
## 38   4.172962078
## 39   4.122069345
## 40   4.073851322
## 41   4.028138069
## 42   3.984775132
## 43   3.943621889
## 44   3.904550110
## 45   3.867442685
## 46   3.832192526
## 47   3.798701580
## 48   3.766879976
## 49   3.736645259
## 50   3.707921719
## 51   3.680639788
## 52   3.654735516
## 53   3.630150089
## 54   3.606829413
```

```
## 54   3.606829415
## 55   3.584723734
## 56   3.563787302
## 57   3.543978070
## 58   3.525257426
## 59   3.507589949
## 60   3.490943193
## 61   3.475287499
## 62   3.460595815
## 63   3.446843546
## 64   3.434008417
## 65   3.422070347
## 66   3.411011346
## 67   3.400815413
## 68   3.391468459
## 69   3.382958226
## 70   3.375274234
## 71   3.368407718
## 72   3.362351593
## 73   3.357100410
## 74   3.352650334
## 75   3.348999124
## 76   3.346146121
## 77   3.344092241
## 78   3.342839984
## 79   3.342393439
## 80   3.342758311
## 81   3.343941938
## 82   3.345953334
## 83   3.348803227
## 84   3.352504115
## 85   3.357070323
## 86   3.362518077
## 87   3.368865586
## 88   3.376133133
## 89   3.384343180
## 90   3.393520491
## 91   3.403692259
## 92   3.414888258
## 93   3.427141012
## 94   3.440485976
## 95   3.454961747
## 96   3.470610297
## 97   3.487477225
## 98   3.505612052
## 99   3.525068539
## 100  3.545905050
## 101  3.568184947
##
##
## $result.matrix
##                                      1
## error                    0.002016850185
## reached.threshold        0.008798712905
## steps                   39.000000000000
## Intercept.to.1layhid1   -0.109965140800
## x.to.1layhid1           -2.755539292023
## Intercept.to.1layhid2   -0.189520706973
## x.to.1layhid2            0.403156971418
## Intercept.to.1layhid3   -0.463625891565
## x.to.1layhid3           -1.802673909392
## Intercept.to.1layhid4   -1.239140251574
## x.to.1layhid4            2.143912554705
## Intercept.to.2layhid1    1.014930063823
## 1layhid.1.to.2layhid1    1.760817529369
## 1layhid.2.to.2layhid1   -0.959941109674
## 1layhid.3.to.2layhid1    0.542082230350
## 1layhid.4.to.2layhid1   -2.259616461482
## Intercept.to.2layhid2   -0.743260734194
## 1layhid.1.to.2layhid2   -1.669952641283
## 1layhid.2.to.2layhid2   -0.933976849597
## 1layhid.3.to.2layhid2   -0.707230581584
## 1layhid.4.to.2layhid2    0.650693878655
## Intercept.to.values      1.234012817770
## 2layhid.1.to.values     -1.622269970081
## 2layhid.2.to.values      0.180473419037
##
## attr(,"class")
## [1] "nn"
```

```
plot(net.sin, rep="best")
```



Error: 0.002017 Steps: 39

steps also increases!

Message

This post is only an introductory to the NN network, there are much more mathematical terms to learn to understand how the NN network is built behind the packages such as back-propagation, gradient descent, and other things. If you are courious about the above examples, you can click on the references to view more examples from real life dataset. Such as recognizing hand written digits. I use functions regression since using real datas to predict is copying the code from the reference pages.

Reference

Fitting a Neural Network in R; neuralnet package https://datascienceplus.com/fitting-neural-network-in-r/

R Code Example for Neural Networks https://www.r-bloggers.com/r-code-example-for-neural-networks/

Artificial Neural Networks in R https://rpubs.com/julianhatwell/annr

Neural Networks with R – A Simple Example http://gekkoquant.com/2012/05/26/neural-networks-with-r-simple-example/

Using neural network for regression https://www.r-bloggers.com/using-neural-network-for-regression/

Using neural networks for regression https://www.packtpub.com/mapt/book/big_data_and_business_intelligence/9781783989065/4/ch04lvl1sec54/using-neural-networks-for-regression

Neural Networks https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-s06/www/nn.pdf