6_NN

Park Ju ho

2022 6 9

사용 패키지

```
library(nnet)
library(devtools)
library(scales)
library(clusterGeneration)
library(reshape)
library(neuralnet)
library(MASS)
```

순전파 신경망

iris data에 대한 순전파 신경망 파라미터 설명 size = 벡터의 길이 = 층의 수, 벡터 안의 값 = 노드 수 decay = 수렴 판단 기준으로 추정의 변화량이 이 숫자보다 작아지면 종료 = early stopping maxit = epoch

```
nn.iris = nnet(Species~., data = iris, size = c(2,2), rang=.1, decay = 5e-4, maxit=200)
```

```
## # weights:  16
## initial  value 69.270455
## iter  10 value 33.114001
## iter  10 value 33.114001
## iter  10 value 33.114001
## final  value 33.114001
## converged
```

```
summary(nn.iris)
```

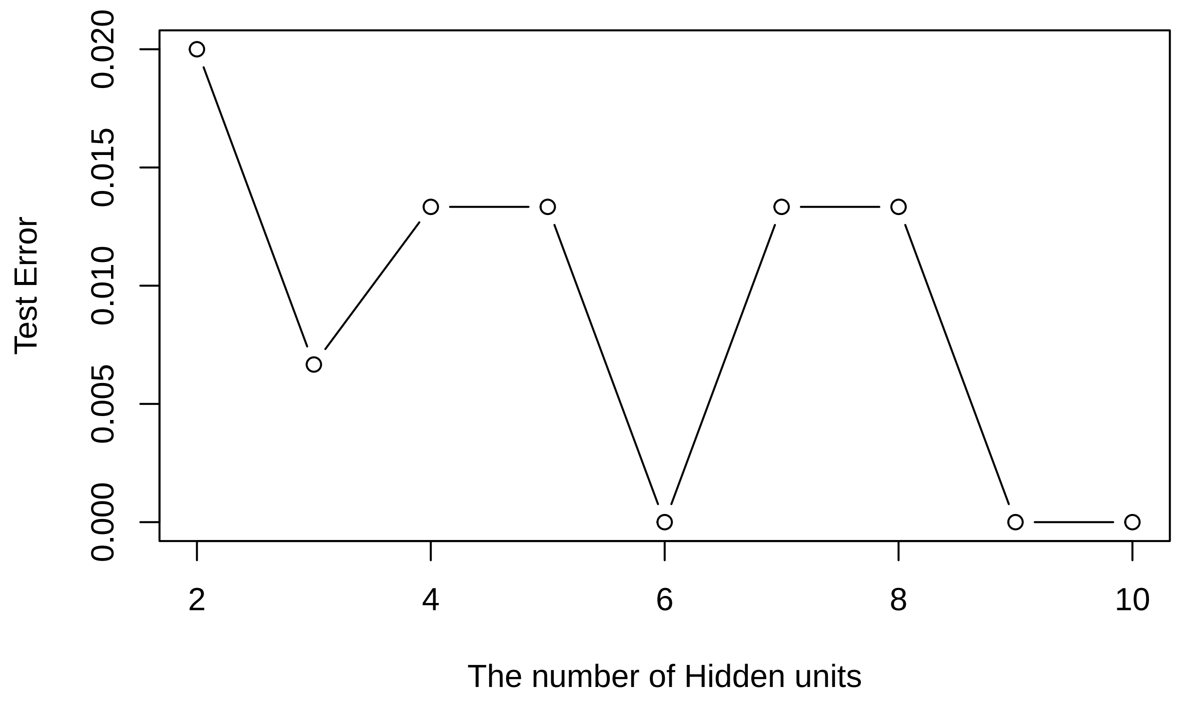
```
## a 4-2-2 network with 16 weights
## options were - softmax modelling  decay=5e-04
##  b->h1 i1->h1 i2->h1 i3->h1 i4->h1
##   3.45   6.51  21.86 -30.44 -13.07
##  b->h2 i1->h2 i2->h2 i3->h2 i4->h2
##  -3.39 -16.03 -11.64  -3.09  -0.08
##  b->o1 h1->o1 h2->o1
## -10.48  18.50 -17.50
##  b->o2 h1->o2 h2->o2
##  10.50 -18.50  17.39
```

즉 4 - 2 - 2의 신경망이 완성 절편을 포함하여 총 16개의 가중치를 계산함 i = input 즉 입력층 h = hidden 은닉층 o = 은닉층 b = 절편

은닉층의 수에 따라 오차 비교

```r
#          err
test.err = function(h.size){
  ir = nnet(Species~., data=iris, size = h.size,
            decay = 5e-4, trace=F)
  y = iris$Species
  p = predict(ir, iris, type = "class")
  err = mean(y != p)
  c(h.size, err)
}
```

```r
out = t(sapply(2:10,FUN = test.err))
plot(out,type="b",xlab = "The number of Hidden units", ylab = "Test Error")
```



은닉 노드수가 늘어 날 수록 오차가 줄어드는 것을 확인 할 수 있다 하지만 과적합이 일어날 수 있으므로 Drop out이 필요하다

역전파 인공신경망

```
net.iris = neuralnet(Species~.,hidden=c(2,2),data=iris,linear.output = F,stepmax=1e+10)
```

sample을 하나씩 넣었을 때 찾아낸 가중치의 결과

```
#net.iris$generalized.weights
```

결과값이 너무 길어서 출력은 하지 않음 NaN은 계산이 불가능한 경우, 가중치의 크기가 결국 표본별로 영향력의 척도라고 할 수 있으나 해석은 불가능

가중치의 초기치

```
net.iris$startweights
```

```
## [[1]]
## [[1]][[1]]
##             [,1]       [,2]
## [1,] -0.79083529 -0.1281320
## [2,]  0.57940358 -0.6253572
## [3,]  0.19851801  0.7017385
## [4,]  0.50915846  0.6521948
## [5,] -0.02994811 -0.9956029
##
## [[1]][[2]]
##           [,1]       [,2]
## [1,] 1.3474230  0.4061375
## [2,] 0.8615827 -0.5225046
## [3,] 0.7119399 -0.7683240
##
## [[1]][[3]]
##            [,1]        [,2]       [,3]
## [1,] -0.2106697  0.02548426 -0.1644927
## [2,] -1.5752531  1.59330403  0.3310968
## [3,]  0.5259995 -1.24597098 -0.1031725
```

인공신경망의 결과

```
net.iris$result.matrix
```

```
##                                  [,1]
## error                    1.000639e+00
## reached.threshold        9.872318e-03
## steps                    7.390500e+04
## Intercept.to.1layhid1    -5.986402e+00
## Sepal.Length.to.1layhid1 -3.026624e-01
## Sepal.Width.to.1layhid1  -1.419107e+00
## Petal.Length.to.1layhid1  1.177039e+00
## Petal.Width.to.1layhid1   3.291045e+00
## Intercept.to.1layhid2     5.691735e+00
## Sepal.Length.to.1layhid2  5.387764e-02
## Sepal.Width.to.1layhid2  -1.624370e-01
## Petal.Length.to.1layhid2 -6.806320e-01
## Petal.Width.to.1layhid2  -1.753263e+00
## Intercept.to.2layhid1     3.395701e+02
## 1layhid1.to.2layhid1      1.606538e+03
## 1layhid2.to.2layhid1     -4.470026e+02
## Intercept.to.2layhid2     3.964307e+00
## 1layhid1.to.2layhid2     -2.456248e+01
## 1layhid2.to.2layhid2      1.953489e+01
## Intercept.to.setosa       4.888838e+00
## 2layhid1.to.setosa       -3.439093e+02
## 2layhid2.to.setosa        3.184796e+01
## Intercept.to.versicolor  -2.995925e+02
## 2layhid1.to.versicolor    1.294816e+02
## 2layhid2.to.versicolor    2.921259e+02
## Intercept.to.virginica    1.361972e+02
## 2layhid1.to.virginica     1.424743e+02
## 2layhid2.to.virginica    -4.788436e+02
```

reach.threshold는 beta가 일정 값에 도달하였을 때 마지막과 그 직전의 변화량을 보여줌 -.to.- = 각 노드에서 노드별 가중치를 보여줌

신경망 그래프

```
plot(net.iris)
```

## 신경망을 이용한 예측

```
compute(net.iris,iris)$net.result
```

```
##                  [,1]         [,2]          [,3]
##   [1,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [2,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [3,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [4,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [5,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [6,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [7,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [8,]  1.000000e+00 5.715214e-04 1.550827e-149
##   [9,]  1.000000e+00 5.715214e-04 1.550827e-149
##  [10,]  1.000000e+00 5.715214e-04 1.550827e-149
##  [11,]  1.000000e+00 5.715214e-04 1.550827e-149
##  [12,]  1.000000e+00 5.715214e-04 1.550827e-149
##  [13,]  1.000000e+00 5.715214e-04 1.550827e-149
## [14,]  1.000000e+00 5.715214e-04 1.550827e-149
## [15,]  1.000000e+00 5.715214e-04 1.550827e-149
## [16,]  1.000000e+00 5.715214e-04 1.550827e-149
## [17,]  1.000000e+00 5.715214e-04 1.550827e-149
## [18,]  1.000000e+00 5.715214e-04 1.550827e-149
## [19,]  1.000000e+00 5.715214e-04 1.550827e-149
## [20,]  1.000000e+00 5.715214e-04 1.550827e-149
## [21,]  1.000000e+00 5.715214e-04 1.550827e-149
## [22,]  1.000000e+00 5.715214e-04 1.550827e-149
## [23,]  1.000000e+00 5.715214e-04 1.550827e-149
## [24,]  1.000000e+00 5.715214e-04 1.550827e-149
## [25,]  1.000000e+00 5.715214e-04 1.550827e-149
## [26,]  1.000000e+00 5.715214e-04 1.550827e-149
## [27,]  1.000000e+00 5.715214e-04 1.550827e-149
## [28,]  1.000000e+00 5.715214e-04 1.550827e-149
## [29,]  1.000000e+00 5.715214e-04 1.550827e-149
## [30,]  1.000000e+00 5.715214e-04 1.550827e-149
## [31,]  1.000000e+00 5.715214e-04 1.550827e-149
## [32,]  1.000000e+00 5.715214e-04 1.550827e-149
## [33,]  1.000000e+00 5.715214e-04 1.550827e-149
## [34,]  1.000000e+00 5.715214e-04 1.550827e-149
## [35,]  1.000000e+00 5.715214e-04 1.550827e-149
## [36,]  1.000000e+00 5.715214e-04 1.550827e-149
## [37,]  1.000000e+00 5.715214e-04 1.550827e-149
## [38,]  1.000000e+00 5.715214e-04 1.550827e-149
## [39,]  1.000000e+00 5.715214e-04 1.550827e-149
## [40,]  1.000000e+00 5.715214e-04 1.550827e-149
## [41,]  1.000000e+00 5.715214e-04 1.550827e-149
## [42,]  1.000000e+00 5.715214e-04 1.550827e-149
## [43,]  1.000000e+00 5.715214e-04 1.550827e-149
## [44,]  1.000000e+00 5.715214e-04 1.550827e-149
## [45,]  1.000000e+00 5.715214e-04 1.550827e-149
## [46,]  1.000000e+00 5.715214e-04 1.550827e-149
## [47,]  1.000000e+00 5.715214e-04 1.550827e-149
## [48,]  1.000000e+00 5.715214e-04 1.550827e-149
## [49,]  1.000000e+00 5.715214e-04 1.550827e-149
## [50,]  1.000000e+00 5.715214e-04 1.550827e-149
## [51,] 3.949278e-134 1.000000e+00  1.171558e-87
## [52,] 3.947241e-134 1.000000e+00  1.180684e-87
## [53,] 3.916507e-134 1.000000e+00  1.327924e-87
## [54,] 3.950560e-134 1.000000e+00  1.165858e-87
## [55,] 3.929353e-134 1.000000e+00  1.264130e-87
## [56,] 3.950000e-134 1.000000e+00  1.168342e-87
## [57,] 3.896864e-134 1.000000e+00  1.432205e-87
## [58,] 4.228463e-134 1.000000e+00  1.132772e-87
## [59,] 3.950265e-134 1.000000e+00  1.167164e-87
## [60,] 3.950521e-134 1.000000e+00  1.166030e-87
## [61,] 3.950728e-134 1.000000e+00  1.165109e-87
## [62,] 3.949222e-134 1.000000e+00  1.171807e-87
## [63,] 3.950727e-134 1.000000e+00  1.165117e-87
## [64,] 3.943506e-134 1.000000e+00  1.197606e-87
## [65,] 3.950721e-134 1.000000e+00  1.165142e-87
## [66,] 3.950289e-134 1.000000e+00  1.167059e-87
## [67,] 3.935934e-134 1.000000e+00  1.232722e-87
##  [68,] 3.950726e-134 1.000000e+00  1.165118e-87
##  [69,] 3.568118e-134 1.000000e+00  5.388866e-87
##  [70,] 3.950725e-134 1.000000e+00  1.165126e-87
##  [71,] 1.448364e-136 1.000000e+00  4.890273e-51
##  [72,] 3.950692e-134 1.000000e+00  1.165270e-87
##  [73,] 2.394657e-134 1.000000e+00  2.165460e-84
##  [74,] 3.950389e-134 1.000000e+00  1.166617e-87
##  [75,] 3.950616e-134 1.000000e+00  1.165607e-87
##  [76,] 3.950173e-134 1.000000e+00  1.167574e-87
##  [77,] 3.941794e-134 1.000000e+00  1.205454e-87
##  [78,] 2.085501e-135 1.000000e+00  1.876968e-68
##  [79,] 3.937554e-134 1.000000e+00  1.225119e-87
##  [80,] 5.004293e-134 1.000000e+00  1.056413e-87
##  [81,] 3.950726e-134 1.000000e+00  1.165121e-87
##  [82,] 3.950728e-134 1.000000e+00  1.165110e-87
##  [83,] 3.950719e-134 1.000000e+00  1.165149e-87
##  [84,] 9.833574e-140 9.748248e-01  2.102510e-03
##  [85,] 3.931748e-134 1.000000e+00  1.252602e-87
##  [86,] 3.935455e-134 1.000000e+00  1.234979e-87
##  [87,] 3.939259e-134 1.000000e+00  1.217167e-87
##  [88,] 3.949773e-134 1.000000e+00  1.169352e-87
##  [89,] 3.950664e-134 1.000000e+00  1.165396e-87
##  [90,] 3.950631e-134 1.000000e+00  1.165540e-87
##  [91,] 3.950565e-134 1.000000e+00  1.165833e-87
##  [92,] 3.947948e-134 1.000000e+00  1.177509e-87
##  [93,] 3.950712e-134 1.000000e+00  1.165181e-87
##  [94,] 3.963047e-134 1.000000e+00  1.163605e-87
##  [95,] 3.950554e-134 1.000000e+00  1.165884e-87
##  [96,] 3.950700e-134 1.000000e+00  1.165236e-87
##  [97,] 3.950616e-134 1.000000e+00  1.165607e-87
##  [98,] 3.950599e-134 1.000000e+00  1.165683e-87
##  [99,]  8.442560e-36 1.000000e+00 2.140057e-128
## [100,] 3.950650e-134 1.000000e+00  1.165457e-87
## [101,] 5.824826e-148 1.323586e-74  1.000000e+00
## [102,] 5.890137e-148 1.466120e-74  1.000000e+00
## [103,] 5.824999e-148 1.323948e-74  1.000000e+00
## [104,] 5.926021e-148 1.550117e-74  1.000000e+00
## [105,] 5.824861e-148 1.323659e-74  1.000000e+00
## [106,] 5.824835e-148 1.323605e-74  1.000000e+00
## [107,] 1.554797e-141 1.169254e-15  1.000000e+00
## [108,] 5.826848e-148 1.327806e-74  1.000000e+00
## [109,] 5.827432e-148 1.329029e-74  1.000000e+00
## [110,] 5.824830e-148 1.323596e-74  1.000000e+00
## [111,] 6.967196e-148 6.841402e-74  1.000000e+00
## [112,] 5.859330e-148 1.397269e-74  1.000000e+00
## [113,] 5.826299e-148 1.326660e-74  1.000000e+00
## [114,] 5.828538e-148 1.331343e-74  1.000000e+00
## [115,] 5.824843e-148 1.323623e-74  1.000000e+00
## [116,] 5.825097e-148 1.324152e-74  1.000000e+00
## [117,] 6.701982e-148 4.792305e-74  1.000000e+00
## [118,] 5.824892e-148 1.323724e-74  1.000000e+00
## [119,] 5.824824e-148 1.323583e-74  1.000000e+00
## [120,] 4.004307e-140 1.010585e-02  9.993550e-01
## [121,] 5.824880e-148 1.323698e-74  1.000000e+00
## [122,] 5.873690e-148 1.428995e-74  1.000000e+00
## [123,] 5.824836e-148 1.323608e-74  1.000000e+00
## [124,] 7.163264e-147 1.312849e-64  1.000000e+00
## [125,] 5.826952e-148 1.328024e-74  1.000000e+00
## [126,] 6.004820e-148 1.749790e-74  1.000000e+00
## [127,] 7.000102e-144 3.499212e-37  1.000000e+00
## [128,] 1.093108e-142 3.103865e-26  1.000000e+00
## [129,] 5.824982e-148 1.323911e-74  1.000000e+00
## [130,] 1.002822e-145 4.275131e-54  1.000000e+00
## [131,] 5.825590e-148 1.325180e-74  1.000000e+00
## [132,] 5.854775e-148 1.387336e-74  1.000000e+00
## [133,] 5.824864e-148 1.323665e-74  1.000000e+00
## [134,] 2.538703e-134 1.000000e+00  8.997565e-85
## [135,] 4.162597e-140 1.435956e-02  9.988453e-01
## [136,] 5.824834e-148 1.323603e-74  1.000000e+00
## [137,] 5.824855e-148 1.323646e-74  1.000000e+00
## [138,] 7.540323e-148 1.412717e-73  1.000000e+00
## [139,] 4.258771e-140 1.764738e-02  9.983729e-01
## [140,] 5.831933e-148 1.338473e-74  1.000000e+00
## [141,] 5.824838e-148 1.323612e-74  1.000000e+00
## [142,] 5.825952e-148 1.325936e-74  1.000000e+00
## [143,] 5.890137e-148 1.466120e-74  1.000000e+00
## [144,] 5.824844e-148 1.323624e-74  1.000000e+00
## [145,] 5.824831e-148 1.323596e-74  1.000000e+00
## [146,] 5.825079e-148 1.324114e-74  1.000000e+00
## [147,] 5.919471e-148 1.534472e-74  1.000000e+00
## [148,] 5.910490e-148 1.513250e-74  1.000000e+00
## [149,] 5.825172e-148 1.324307e-74  1.000000e+00
## [150,] 3.458805e-147 1.651978e-67  1.000000e+00
```

출력 노드에서 나오는 확률값, 각 범주에 속하 확률을 제공

자녀 부모 정보에 대한 신경망 구축

```
data(infert)
net.infert = neuralnet(case~parity + induced + spontaneous,hidden = c(20,20),data = infert, linear.outpu
```

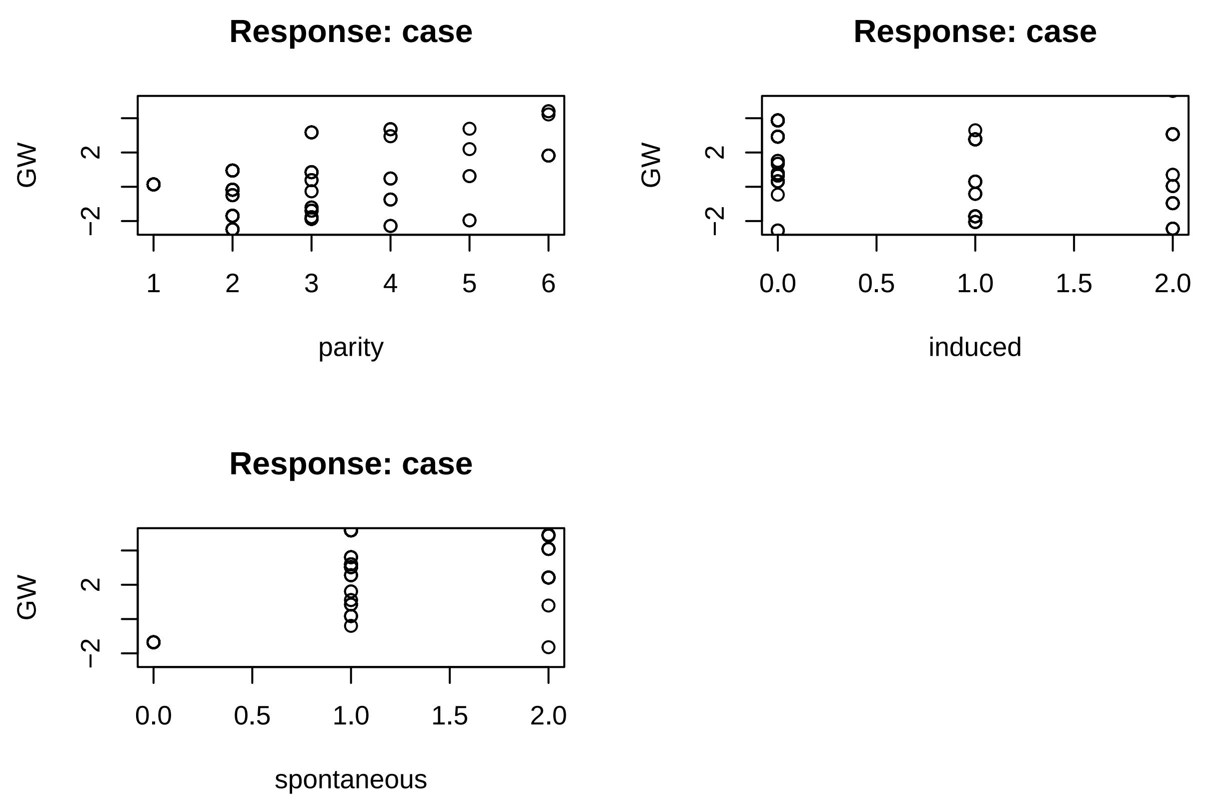
신경망 그래프

```
plot(net.infert)
```

generalized weights를 그래프로 표현

```
head(net.infert$generalized.weights[[1]])#    =
##              [,1]        [,2]        [,3]
## [1,]  4.2180984 -9.73598917   0.7884575
## [2,] -3.7250865  2.76519502 -24.8831866
## [3,]  1.8203716 -2.44269567  -4.1290248
## [4,]  3.3582870 -3.47801373  39.7624096
## [5,] -1.3947504  0.30026685   3.6106865
## [6,]  0.4812687  0.04284231   0.1735865
```

```
par(mfrow=c(2,2))
#gwplot(net.infert, selected.covariate='age', min=-2.5, max=5)
gwplot(net.infert, selected.covariate='parity', min=-2.5, max=5)
gwplot(net.infert, selected.covariate='induced', min=-2.5, max=5)
gwplot(net.infert, selected.covariate='spontaneous', min=-2.5, max=5)
```



보스턴 집 값 예측에 신경망 활용

회귀를 활용하여 예측

```
#
apply(Boston,2,function(x) sum(is.na(x)))
```

```
##     crim      zn   indus    chas     nox      rm     age     dis     rad     tax
##       0       0       0       0       0       0       0       0       0       0
## ptratio   black   lstat    medv
##       0       0       0       0
```

```
index <- sample(1:nrow(Boston),round(0.75*nrow(Boston)))
train <- Boston[index,]
test <- Boston[-index,]
lm.fit <- glm(medv~., data=train)
summary(lm.fit)
```

```
##
## Call:
## glm(formula = medv ~ ., data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -11.4148   -2.7243   -0.6633    1.7111   26.6663
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.112e+01  5.978e+00   5.205 3.24e-07 ***
## crim        -1.449e-01  3.545e-02  -4.086 5.41e-05 ***
## zn           4.182e-02  1.592e-02   2.627  0.00899 **
## indus        2.906e-02  7.004e-02   0.415  0.67849
## chas         2.372e+00  9.614e-01   2.467  0.01408 *
## nox         -1.828e+01  4.264e+00  -4.285 2.33e-05 ***
## rm           4.606e+00  4.909e-01   9.383  < 2e-16 ***
## age         -7.329e-05  1.494e-02  -0.005  0.99609
## dis         -1.468e+00  2.264e-01  -6.487 2.85e-10 ***
## rad          3.584e-01  7.336e-02   4.886 1.54e-06 ***
## tax         -1.324e-02  4.150e-03  -3.191  0.00154 **
## ptratio     -9.221e-01  1.487e-01  -6.200 1.52e-09 ***
## black        9.246e-03  3.233e-03   2.860  0.00448 **
## lstat       -4.981e-01  5.839e-02  -8.530 3.92e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 22.45949)
##
##     Null deviance: 33624.1  on 379  degrees of freedom
## Residual deviance:  8220.2  on 366  degrees of freedom
## AIC: 2276.6
##
## Number of Fisher Scoring iterations: 2
```

```
pr.lm <- predict(lm.fit,test)
MSE.lm <- sum((pr.lm - test$medv)^2)/nrow(test)

MSE.lm
```

```
## [1] 24.16145
```

MSE가 20정도로 나옴

```
maxs <- apply(Boston, 2, max)
mins <- apply(Boston, 2, min)
scaled <- as.data.frame(scale(Boston, center = mins, scale = maxs - mins))#
#center =      scale =
train_ <- scaled[index,]
test_ <- scaled[-index,]
```

신경망

```
n <- names(train_)
f <- as.formula(paste("medv ~", paste(n[!n %in% "medv"], collapse = " + ")))
net.Boston <- neuralnet(f,data=train_,hidden=c(5,3), linear.output=T)

plot(net.Boston)
net.Boston$result.matrix
```

```
##                                [,1]
## error                  4.752871e-01
## reached.threshold      9.525397e-03
## steps                  1.973000e+03
## Intercept.to.1layhid1  5.192067e-01
## crim.to.1layhid1      -1.891732e+01
## zn.to.1layhid1         6.893425e-01
## indus.to.1layhid1     -6.167413e-01
## chas.to.1layhid1       1.023418e+00
## nox.to.1layhid1       -3.012210e+00
## rm.to.1layhid1         1.768811e+00
## age.to.1layhid1       -3.609727e-01
## dis.to.1layhid1       -2.376003e+00
## rad.to.1layhid1        7.623086e+00
## tax.to.1layhid1       -3.130355e-02
## ptratio.to.1layhid1    5.311414e-01
## black.to.1layhid1     -5.466352e-01
## lstat.to.1layhid1     -2.291100e-01
## Intercept.to.1layhid2  9.584730e-01
## crim.to.1layhid2      -1.702517e+00
## zn.to.1layhid2         3.901101e-01
## indus.to.1layhid2     -2.259311e-01
## chas.to.1layhid2       4.556995e-01
## nox.to.1layhid2        5.071831e-02
## rm.to.1layhid2        -4.696682e+00
## age.to.1layhid2        8.539596e-01
## dis.to.1layhid2       -3.300933e-03
## rad.to.1layhid2        1.507388e+00
## tax.to.1layhid2        1.870233e+00
## ptratio.to.1layhid2    1.558057e+00
## black.to.1layhid2      3.963617e-01
## lstat.to.1layhid2      1.455291e-01
## Intercept.to.1layhid3 -2.082331e-01
## crim.to.1layhid3      -5.507540e-01
## zn.to.1layhid3        -1.244564e+02
## indus.to.1layhid3      2.948997e+00
## chas.to.1layhid3       1.617577e+00
## nox.to.1layhid3       -7.244699e+00
## rm.to.1layhid3        -3.578617e-01
## age.to.1layhid3        3.166062e-01
## dis.to.1layhid3       -2.132694e+01
## rad.to.1layhid3        2.067424e+00
## tax.to.1layhid3        1.657648e+00
## ptratio.to.1layhid3   -4.231633e-01
## black.to.1layhid3      2.266722e-01
## lstat.to.1layhid3     -2.162685e+00
## Intercept.to.1layhid4  2.968962e+00
## crim.to.1layhid4       3.286304e+01
## zn.to.1layhid4        -2.287018e+00
## indus.to.1layhid4      3.443417e+00
## chas.to.1layhid4      -1.984470e+02
## nox.to.1layhid4       -7.246500e+00
## rm.to.1layhid4        -1.587337e+00
## age.to.1layhid4       -3.836926e+00
## dis.to.1layhid4        5.431711e+00
## rad.to.1layhid4        2.510979e+00
## tax.to.1layhid4       -3.361985e+00
## ptratio.to.1layhid4   -6.401357e+00
## black.to.1layhid4     -2.067264e+00
## lstat.to.1layhid4      2.353154e+01
## Intercept.to.1layhid5  2.628674e+00
## crim.to.1layhid5      -1.381890e+01
## zn.to.1layhid5         1.378274e+02
## indus.to.1layhid5     -5.863969e+00
## chas.to.1layhid5       6.927080e+00
## nox.to.1layhid5        6.015227e-01
## rm.to.1layhid5        -5.704983e-01
## age.to.1layhid5       -2.752306e+00
## dis.to.1layhid5        2.359942e+01
## rad.to.1layhid5        1.781378e+00
## tax.to.1layhid5       -5.305759e+00
## ptratio.to.1layhid5   -7.688933e-01
## black.to.1layhid5      1.912419e+00
## lstat.to.1layhid5      1.829977e+00
## Intercept.to.2layhid1 -2.817884e-01
## 1layhid1.to.2layhid1  -6.008027e-01
## 1layhid2.to.2layhid1   1.132491e+00
## 1layhid3.to.2layhid1   3.840779e+00
## 1layhid4.to.2layhid1  -7.287820e-01
## 1layhid5.to.2layhid1  -7.139670e-01
## Intercept.to.2layhid2 -5.796448e-01
## 1layhid1.to.2layhid2  -3.237940e+00
## 1layhid2.to.2layhid2   1.978046e+00
## 1layhid3.to.2layhid2   4.214747e+00
## 1layhid4.to.2layhid2  -5.654571e-01
## 1layhid5.to.2layhid2   5.381272e+01
## Intercept.to.2layhid3  2.724728e-01
## 1layhid1.to.2layhid3   7.291899e-01
## 1layhid2.to.2layhid3  -2.155890e+00
## 1layhid3.to.2layhid3   2.422160e+00
## 1layhid4.to.2layhid3  -8.021115e-01
## 1layhid5.to.2layhid3  -9.124011e-01
## Intercept.to.medv      1.119094e-01
## 2layhid1.to.medv      -7.126562e-01
## 2layhid2.to.medv       2.585153e-01
## 2layhid3.to.medv       2.067023e+00
```

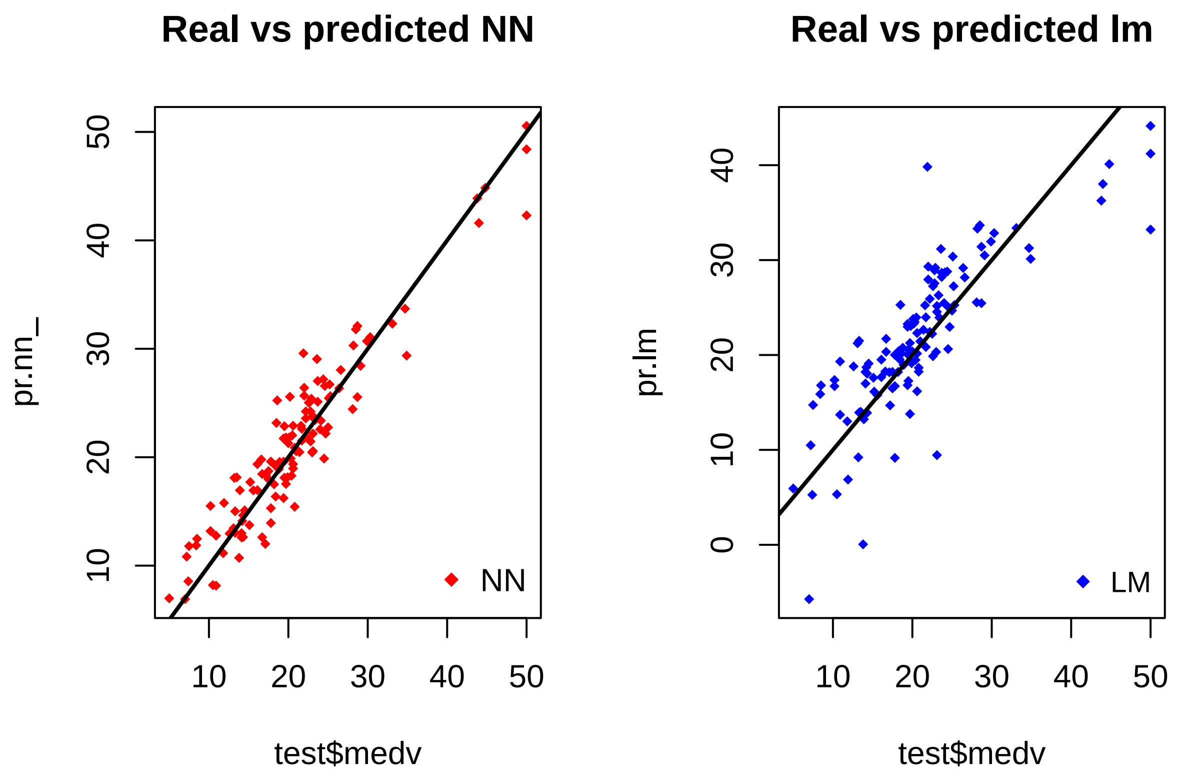
```
pr.nn <- compute(net.Boston,test_[,1:13])
pr.nn_ <- pr.nn$net.result*(max(Boston$medv)-min(Boston$medv))+min(Boston$medv)#
test.r <- (test_$medv)*(max(Boston$medv)-min(Boston$medv))+min(Boston$medv)
MSE.nn <- sum((test.r - pr.nn_)^2)/nrow(test_)
c(MSE.lm,MSE.nn)
```

```
## [1] 24.161452  7.081776
```

MSE를 비교해보면 신경망을 이용한 예측에서 훨씬 좋은 성과를 보여주고 있는 것을 확인 할 수 있다

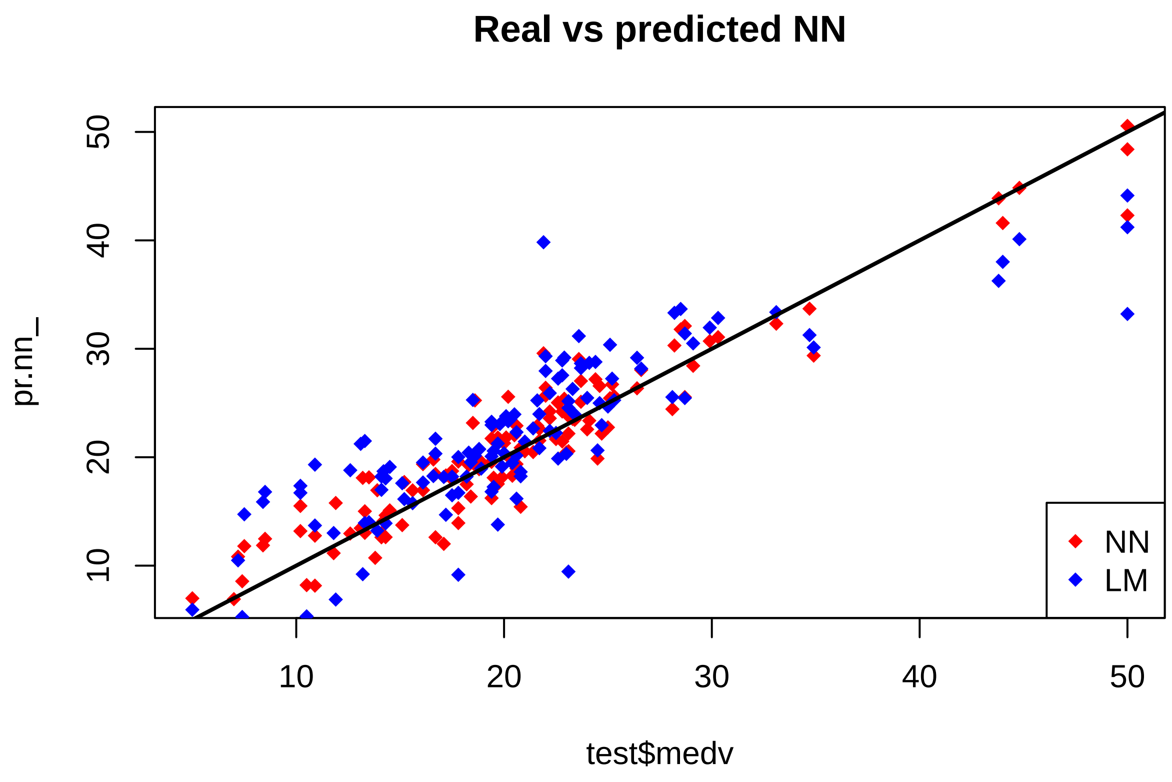
그래프를 이용하여 회귀와 NN의 예측값 비교

```
par(mfrow=c(1,2))
plot(test$medv,pr.nn_,col='red',main='Real vs predicted NN',pch=18,cex=0.7)
abline(0,1,lwd=2)
legend('bottomright',legend='NN',pch=18,col='red', bty='n')
plot(test$medv,pr.lm,col='blue',main='Real vs predicted lm',pch=18, cex=0.7)
abline(0,1,lwd=2)
legend('bottomright',legend='LM',pch=18,col='blue', bty='n', cex=.95)
```



그래프를 비교해보아도 NN이 훨씬 잘 예측하고 있는 것을 확인 할 수 있다 x축 = 실제 y축 = 예측 => 직선에 가까울수록 더 예측을 잘 한다고 할 수 있음

```r
plot(test$medv,pr.nn_,col='red',main='Real vs predicted NN',pch=18,cex=1)
points(test$medv,pr.lm,col='blue',pch=18,cex=1)
abline(0,1,lwd=2)
legend('bottomright',legend=c('NN','LM'),pch=18,col=c('red','blue'))
```


Real vs predicted NN
pr.nn_
test$medv
10
20
30
40
50
10
20
30
40
50
NN
LM