REPORT

수강과목	:	전산통계
담당교수	:	노윤환
학 과	:	통계학과
학 번	:	201611531
이 름	:	정호재
제출일자	:	2019.11.21

<과제2>

첨부된 두 csv 파일을 각각 신경망 모형을 이용하여 분석하고, 분석결과에 대해 기술하세요. 각 반응변수는 변수설명.hwp 파일에 빨간 색으로 표시되어 있습니다.

1. dividendinfo.csv 데이터는 은닉층 2개, 은닉노드는 각 2개, 1개로 설정하여 분석하세요. c(2,1)

(변수작성)

```
> library(nnet)
> library(devtools)
> source_url('https://gist.githubusercontent.com/Peque/41a9e20d6687f2f3108d/raw/85e14f3a292e126f1454864427e3a189c2fe33f3/nnet_plot_update.r')
> library(NeuralNetTools)
> library(clusterGeneration)
> library(scales)
> library(reshape)
> library(caret)
```

fcfps: Free cash flow per share (in $)
earnings_growth: Earnings growth in the past year (in %)
de: Debt to Equity ratio
mcap: Market Capitalization of the stock
current_ratio: Current Ratio (or Current Assets/Current Liabilities)
dividend: a value of 1 to a stock that pays a dividen
a value of 0 to a stock that does not pay a dividend

```
> divi<-read.csv("C:/Users/Administrator/Desktop/dividendinfo.csv",header=T, stringsAsFactors = F)
```

반응 변수 dividend을 제외한 나머지 변수들을 정규화한다.

```
> divi$dividend<-as.factor(divi$dividend)
> divi<-cbind(divi[1], scale(divi[-1]))
> str(divi)
'data.frame':    200 obs. of  6 variables:
 $ dividend       : Factor w/ 2 levels "0","1": 1 2 2 1 2 2 2 1 2 2 ...
 $ fcfps          : num  0.378 2.105 0.402 -1.435 0.527 ...
 $ earnings_growth: num  -2.0545 -0.7169 -0.6996 0.0918 -0.6097 ...
 $ de             : num  -0.982 -1.001 -1.861 -0.419 -0.294 ...
 $ mcap           : num  0.681 1.137 0.772 -0.16 1.426 ...
 $ current_ratio  : num  -0.806 -0.047 0.661 0.613 1.374 ...
```

```
> apply(divi,2,function(x) sum(is.na(x)))
        dividend             fcfps earnings_growth                de
               0                 0               0                 0
            mcap    current_ratio
               0                 0
```

데이터에 결측치는 없다.

데이터의 정확도를 높여주기 위하여 0.75%로 train, test set을 구분한다.

```
> set.seed(123456)
> index <- sample(1:nrow(divi),round(0.75*nrow(divi)))
> train <- divi[index,]
> test <- divi[-index,]
```

① 순전파

(풀이)

분리된 train set에 각각 은닉노드가 2개, 1개인 은닉층 2개로 설정하여 순전파를 적용한다.

```
> nn.divi<-nnet(dividend~fcfps+earnings_growth+de+mcap+current_ratio, data=train,
size=c(2,1), rang=.1, decay=5e-4, maxit=200)
# weights:  15
initial  value 104.536961
iter  10 value 9.802119
iter  20 value 4.201768
iter  30 value 3.905266
iter  40 value 3.900207
iter  50 value 3.777005
iter  60 value 2.635959
iter  70 value 1.017343
iter  80 value 0.951001
iter  90 value 0.897089
iter 100 value 0.891926
iter 110 value 0.887465
iter 120 value 0.886481
iter 130 value 0.886097
iter 140 value 0.886034
iter 150 value 0.886012
iter 160 value 0.886008
iter 170 value 0.886006
final  value 0.886005
converged
```

```
> summary(nn.divi)
a 5-2-1 network with 15 weights
options were - entropy fitting  decay=5e-04
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1 i5->h1
  1.37   4.82   3.49  -6.18  12.64  16.62
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2 i5->h2
 10.73   3.54  -5.42  -0.36  -1.29   1.34
  b->o  h1->o  h2->o
  5.48 21.37 -15.30
```

5-2-1구조의 신경망이고, 15개의 가중치를 갖는다.

test set과 train set을 적용한 신경망 모형의 반응변수를 분할표를 사용하여 비교한다.

반응변수는 범주형자료이므로 type='class'을 적용한다.

```
> confusion_matrix<-table(test$dividend, predict(nn.divi, newdata = test,
type='class'))
> confusion_matrix
    0  1
0  21  5
1   2 22
```

86%(43/50)의 결정력을 가진다.

② 역전파

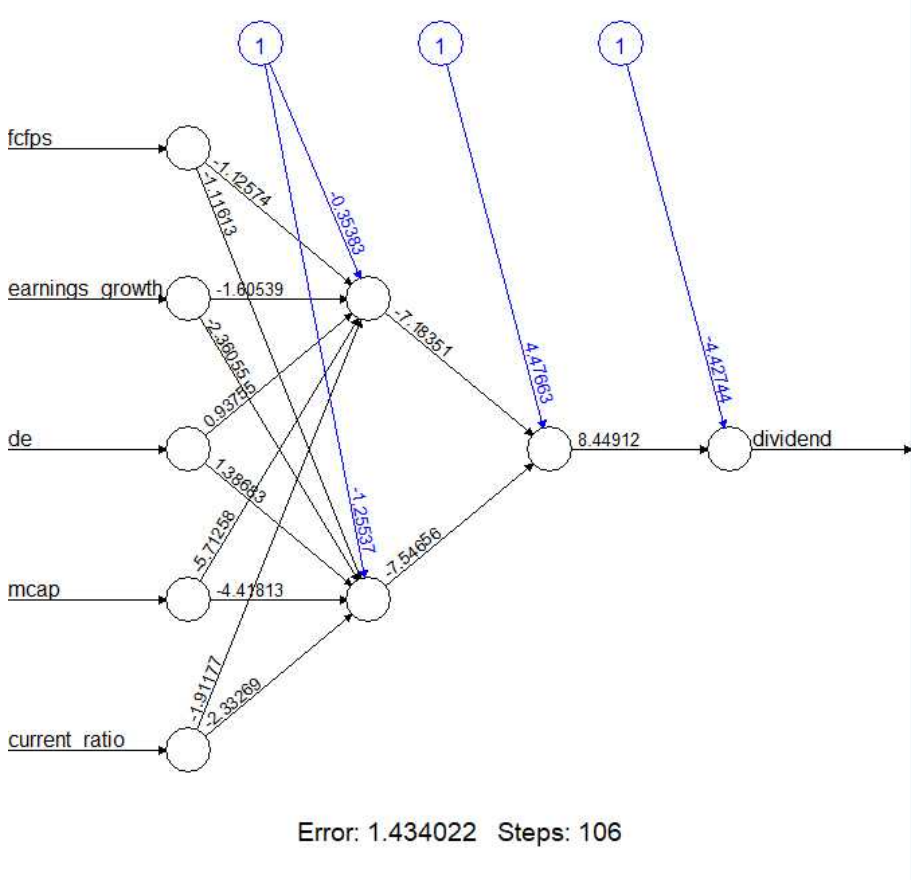
(풀이)

```
> divi<-read.csv("C:/Users/user/Desktop/dividendinfo.csv",header=T,
stringsAsFactors = F)
> divi<-cbind(divi[1], scale(divi[-1]))
> index <- sample(1:nrow(divi),round(0.75*nrow(divi)))
> train <- divi[index,]
> test <- divi[-index,]
```

neuralnet함수를 이용한다. 이 때 반응변수가 범주형 자료임으로 linear.output=F으로 두었다.

```
> library(neuralnet)
> net.divi <- neuralnet(dividend~fcfps+earnings_growth+de+mcap+current_ratio,
hidden=c(2,1), data=data.train, linear.output=F)
> net.dividend$result.matrix
                                      [,1]
error                          1.434021663
reached.threshold              0.007546174
steps                        106.000000000
Intercept.to.1layhid1         -0.353829321
fcfps.to.1layhid1             -1.125735950
earnings_growth.to.1layhid1   -1.605393568
de.to.1layhid1                 0.937550777
mcap.to.1layhid1              -5.712582214
current_ratio.to.1layhid1     -1.911773864
Intercept.to.1layhid2         -1.255368084
fcfps.to.1layhid2             -1.116132185
earnings_growth.to.1layhid2   -2.360554409
de.to.1layhid2                 1.386826477
mcap.to.1layhid2              -4.418133671
current_ratio.to.1layhid2     -2.332685396
Intercept.to.2layhid1          4.476630308
1layhid1.to.2layhid1          -7.183513498
1layhid2.to.2layhid1          -7.546558370
Intercept.to.dividend         -4.427439284
2layhid1.to.dividend           8.449117213
```

```
> plot(net.dividend)
```

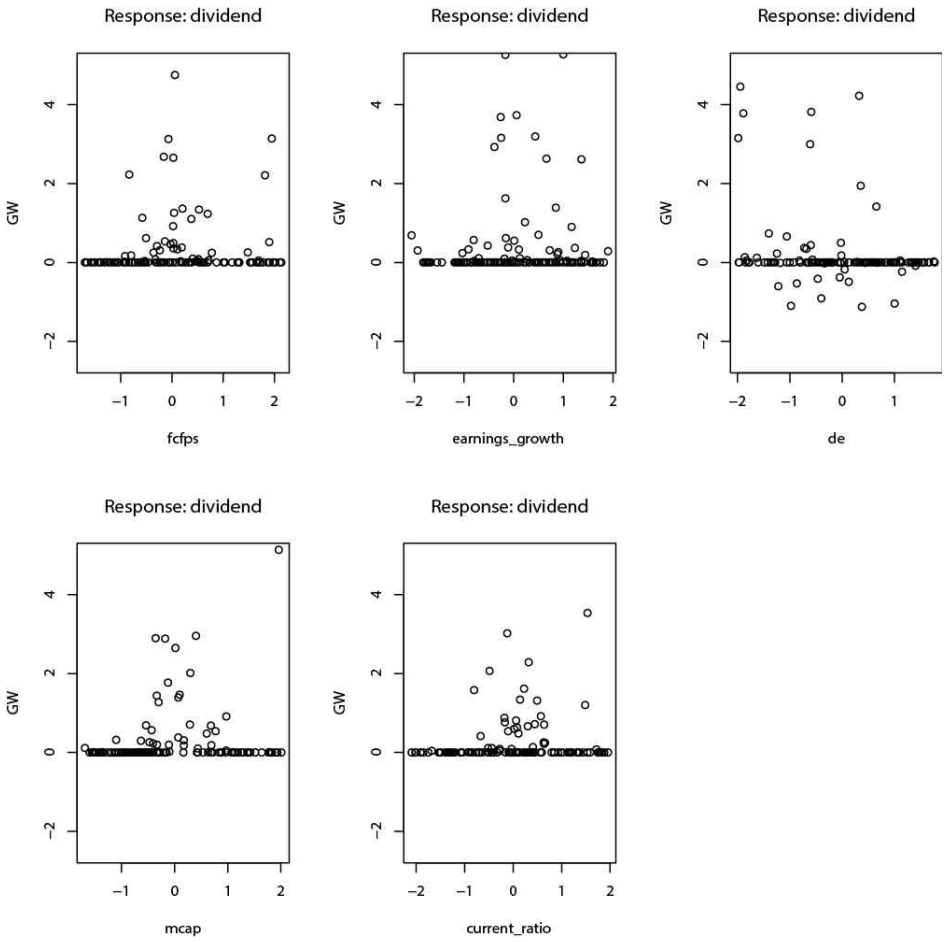


test set과 train set을 적용한 신경망 모형의 반응변수를 분할표를 사용하여 비교한다. 반응변수는 범주형자료이므로 type='class'을 적용한다. 0.5이상의 확률은 1으로 두고 나머지는 0으로 두어 분할표를 작성한다.

```
> net.results <- compute(net.dividend, test)
> results <- data.frame(actual = test$dividend, prediction = net.results$net.result)
> results$prediction <- ifelse(results$prediction>=0.5,1,0)
> pred<-table(results[,1],results[,2])
> pred
    0  1
0  25  1
1   1 23
```

96%(48/50)의 결정력을 가진다.
이는 앞의 순전파보다 더 높은 결정력을 갖는다.

```
> par(mfrow=c(2,3))
> gwplot(net.divi, selected.covariate='fcfps', min=-2.5, max=5)
> gwplot(net.divi, selected.covariate='earnings_growth', min=-2.5, max=5)
> gwplot(net.divi, selected.covariate='de', min=-2.5, max=5)
> gwplot(net.divi, selected.covariate='mcap', min=-2.5, max=5)
> gwplot(net.divi, selected.covariate='current_ratio', min=-2.5, max=5)
```



대부분 변수들의 가중치가 0에서 벗어나지 않으므로 모형에서 큰 영향을 주지 않는다. 따라서 순전파보다 역전파가 모형에 더 적합하다.

2. gasoline.csv 데이터는 은닉층 2개, 은닉노드는 각 2개, 1개로 설정하여 분석하세요. c(2,1)

(변수작성)
consumption: Spend (in $) on gasoline per year for a particular vehicle
capacity: Capacity of the vehicle's fuel tank (in litres)
gasoline: Average cost of gasoline per pump
hours: Hours driven per year by owner

```
> gas<-read.csv("C:/Users/Administrator/Desktop/gasoline.csv",header=T)
> apply(gas,2,function(x) sum(is.na(x)))
> str(gas)
'data.frame':    40 obs. of  4 variables:
 $ consumption: int  1043 1307 1328 1341 1401 1406 1432 1433 1446 1452 ...
 $ capacity   : int  58 60 61 56 56 59 60 59 62 58 ...
 $ gasoline   : num  3.95 3.96 3.82 3.74 3.94 ...
 $ hours      : int  12577 13452 14973 14250 13788 13466 14278 14547 12363
14320 ...
> apply(gas,2,function(x) sum(is.na(x)))
consumption    capacity    gasoline       hours
          0           0           0           0
```

데이터에 결측치는 없다.

```
> summary(gas$consumption)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1043    1457    1622    1595    1711    1980
```

neural network를 사용하기 전 데이터를 정규화 시킨다.
이를 위해 최대, 최솟값을 활용하여 [0,1] 간격으로 데이터를 scale한다.
그 후 데이터의 정확도를 높여주기 위하여 0.75%로 train, test set을 구분한다.

```
> maxs <- apply(gas, 2, max)
> set.seed(123456)
> maxs <- apply(gas, 2, max)
> mins <- apply(gas, 2, min)
> scaled <- as.data.frame(scale(gas, center = mins, scale = maxs - mins))
> index <- sample(1:nrow(gas),round(0.75*nrow(gas)))
> train <- scaled[index,]
> test <- scaled[-index,]
```

① 순전파

(풀이)

분리된 train set에 각각 은닉노드가 2개, 1개인 은닉층 2개로 설정하여 순전파를 적용한다.

```
> nn.gas<-nnet(consumption~capacity+gasoline+hours, data=train, size=c(2,1),rang=.1,
decay=5e-4, maxit=200)
# weights:  11
initial  value 1.489286
iter  10 value 0.508052
iter  20 value 0.364872
iter  30 value 0.321489
iter  40 value 0.286542
iter  50 value 0.279528
iter  60 value 0.275404
iter  70 value 0.273337
iter  80 value 0.271645
iter  90 value 0.269593
iter 100 value 0.268510
iter 110 value 0.267859
iter 120 value 0.267016
iter 130 value 0.264869
iter 140 value 0.263374
iter 150 value 0.262360
iter 160 value 0.261703
iter 170 value 0.261182
iter 180 value 0.261160
iter 190 value 0.261139
iter 200 value 0.261137
final  value 0.261137
stopped after 200 iterations

> summary(nn.gas)
a 3-2-1 network with 11 weights
options were - decay=5e-04
 b->h1 i1->h1 i2->h1 i3->h1
 -2.88   1.97   4.55   0.47
 b->h2 i1->h2 i2->h2 i3->h2
  1.27  -3.35  -4.38   2.78
 b->o h1->o h2->o
-3.16  3.91  4.01
```

3-2-1구조의 신경망이고, 11개의 가중치를 갖는다.

scale시킨 값을 원래의 데이터로 되돌린다.

```
> original<-(test$consumption)*(max(gas$consumption)-min(gas$consumption))
+min(gas$consumption)
> pred<-predict(nn.gas, newdata = test,type='raw')*(max(gas$consumption)
-min(gas$consumption))+min(gas$consumption)
> temp<-data.frame(original,prediction=pred)
> temp
   original prediction
4      1341   1474.838
5      1401   1387.870
8      1433   1586.537
9      1446   1443.393
11     1459   1493.623
13     1474   1556.841
17     1570   1593.171
20     1616   1671.508
24     1641   1723.902
37     1833   1872.929
```

test set과 train set을 적용한 신경망 모형을 MSE를 통해 비교한다.

```
> MSE <- sum((temp[,1] - temp[,2])^2) /  nrow(test)
> MSE
[1] 6181.174
```

순전파에서의 MSE는 6181.174이다.

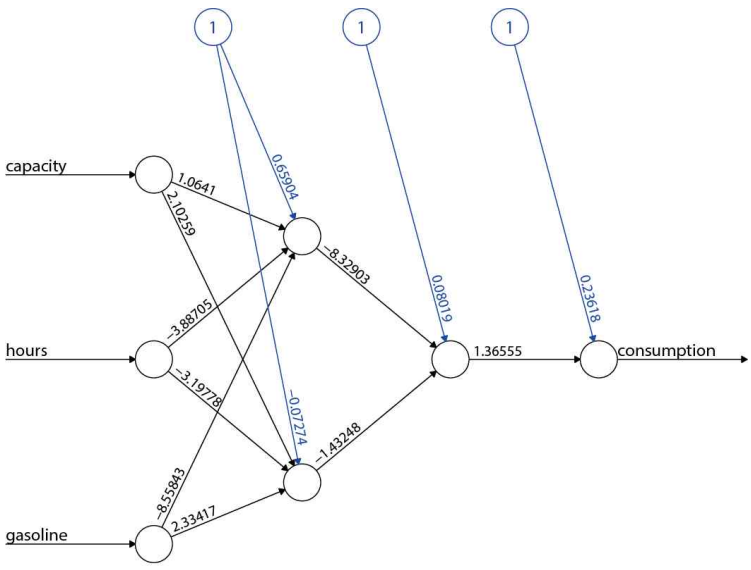
② 역전파

(풀이)

neuralnet함수를 이용한다.

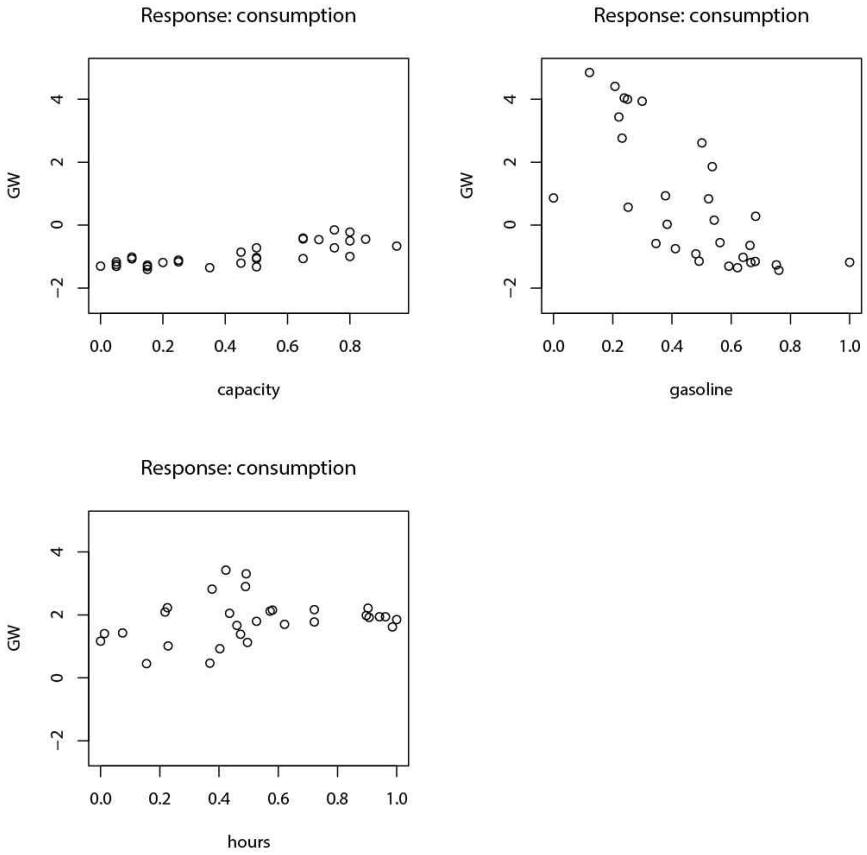
```
> net.gas <- neuralnet(consumption~capacity+hours+gasoline, data=train, hidden=c(2,1), linear.output=T)
> net.gas$result.matrix
                                    [,1]
error                        0.119952358
reached.threshold            0.008720488
steps                      136.000000000
Intercept.to.1layhid1        0.659040645
capacity.to.1layhid1         1.064101954
hours.to.1layhid1           -3.887053792
gasoline.to.1layhid1        -8.558426487
Intercept.to.1layhid2       -0.072738065
capacity.to.1layhid2         2.102593766
hours.to.1layhid2           -3.197784989
gasoline.to.1layhid2         2.334174052
Intercept.to.2layhid1        0.080191610
1layhid1.to.2layhid1        -8.329031716
1layhid2.to.2layhid1        -1.432476949
Intercept.to.consumption     0.236184639
2layhid1.to.consumption      1.365553436
```

```
> plot(net.gas)
```



Error: 0.119952 Steps: 136

```
> par(mfrow=c(2,2))
> gwplot(net.gas, selected.covariate='capacity', min=-2.5, max=5)
> gwplot(net.gas, selected.covariate='gasoline', min=-2.5, max=5)
> gwplot(net.gas, selected.covariate='hours', min=-2.5, max=5)
```



변수들은 0에 벗어나있으므로 모형에 영향을 준다.

```
> pre_nn<-compute(net.gas,test[,2:4])
> pre_nn_<-pre_nn$net.result*(max(gas$consumption)-min(gas$consumption))
+min(gas$consumption)
> test_r <- (test$consumption)*(max(gas$consumption)-min(gas$consumption))
+min(gas$consumption)
> MSE_nn <- sum((test_r-pre_nn_)^2)/nrow(test)
> temp2<-data.frame(original,prediction=pre_nn_)
> temp2
   original  prediction
4        1341    1405.205
5        1401    1425.718
8        1433    1646.780
9        1446    1289.056
11       1459    1593.116
```

```
13      1474    1516.169
17      1570    1499.234
20      1616    1727.277
24      1641    1808.870
37      1833    1627.508
> MSE_nn
[1] 15219.14
```

역전파에서의 MSE는 15219.14이므로 역전파보다 순전파의 모형이 더 적절하다.

```
> results <- data.frame(actual = test$consumption, prediction =
net.gas$net.result[[1]])
> results
       actual prediction
32 0.0000000  0.6069536
30 0.4300961  0.7862625
15 0.4364995  0.5049800
13 0.5378869  0.4752622
14 0.6115261  0.4760387
7  0.6254002  0.3038781
19 0.6638207  0.5769951
4  0.7267876  0.3752412
40 0.8356457  0.8751104
6  0.8399146  0.4903334
24 0.0000000  0.6903717
18 0.4300961  0.5043474
26 0.4364995  0.5766783
39 0.5378869  0.8818771
28 0.6115261  0.6461794
23 0.6254002  0.7204086
22 0.6638207  0.6821136
5  0.7267876  0.4259749
8  0.8356457  0.5325683
17 0.8399146  0.4748905
33 0.0000000  0.7626625
2  0.4300961  0.3716272
3  0.4364995  0.3720619
29 0.5378869  0.6750760
27 0.6115261  0.6411093
11 0.6254002  0.5142156
37 0.6638207  0.8878635
```

```
12 0.7267876   0.3540499
38 0.8356457   0.8943408
34 0.8399146   0.7800726

> predicted=results$prediction * abs(diff(range(gas$consumption))) +
min(gas$consumption)
> actual=results$actual * abs(diff(range(gas$consumption))) + min(gas$consumption)
> comparison=data.frame(predicted,actual)
> deviation=((actual-predicted)/actual)
> comparison=data.frame(predicted,actual,deviation)
> accuracy=1-abs(mean(deviation))
> accuracy
[1] 0.9589546
```

역전파 모형은 약 95.89546%의 정확도를 가진다.