HW4

Sheng Zhang

April 23, 2017

```
## Q1

# Read in data
advertising <- read.csv("./Advertising.csv", header = TRUE)
# advertising <- read.csv("./Spring 2017/Machine Learning/Rmd files/HW4/Advertising.csv", header = TRUE
advertising <- scale(advertising[,-1]) # standardize all inputs to have zero mean and unit variance


# Sample training and test sets
set.seed(1)
advertising.selection.id <- sample(1:nrow(advertising), 200)
advertising.train.id <- sample(advertising.selection.id, 150)
advertising.test.id <- advertising.selection.id[-advertising.train.id]
advertising.train <- advertising[advertising.train.id, ]
advertising.test <- advertising[advertising.test.id, ]
x.train <- advertising.train[,-4]
y.train <- advertising.train[,4]
x.test <- advertising.test[,-4]
y.test <- advertising.test[,4]

# a)
# Build a one-hidden-layer neural network
library(RSNNS)
```

```
## Warning: package 'RSNNS' was built under R version 3.3.3
```

```
## Loading required package: Rcpp
```

```
advertising.nn1 <- RSNNS::mlp(x=x.train,y=y.train,size = c(2), maxit = 10000, learnFuncParams = 0.01, li
summary(advertising.nn1)
```

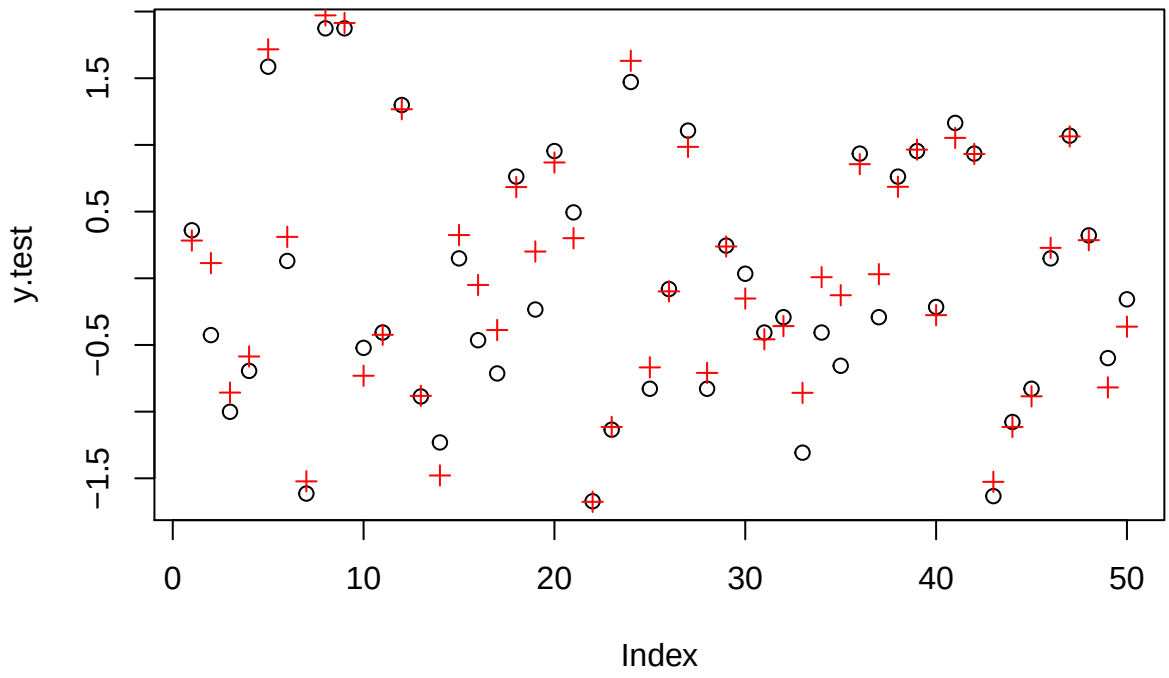
```
## SNNS network definition file V1.4-3D
## generated at Sat Apr 29 17:12:30 2017
##
## network name : RSNNS_untitled
## source files :
## no. of units : 6
## no. of connections : 8
## no. of unit types : 0
## no. of site types : 0
##
##
## learning function : Std_Backpropagation
## update function   : Topological_Order
##
##
```

```
## unit default section :
##
## act      | bias     | st | subnet | layer | act func     | out func
## ---------|----------|----|--------|-------|--------------|-------------
##  0.00000 |  0.00000 | i  |      0 |     1 | Act_Logistic | Out_Identity
## ---------|----------|----|--------|-------|--------------|-------------
##
##
## unit definition section :
##
## no. | typeName | unitName        | act      | bias     | st | position | act func     | out func | si
## ----|----------|-----------------|----------|----------|----|----------|--------------|----------|---
##   1 |          | Input_TV        | -0.82282 | -0.00831 | i  | 1,0,0    | Act_Identity |          |
##   2 |          | Input_Radio     |  0.23143 | -0.26172 | i  | 2,0,0    | Act_Identity |          |
##   3 |          | Input_Newspaper | -0.37900 |  0.17073 | i  | 3,0,0    | Act_Identity |          |
##   4 |          | Hidden_2_1      |  0.11825 | -2.89872 | h  | 1,2,0    |||
##   5 |          | Hidden_2_2      |  0.06712 | -2.11667 | h  | 2,2,0    |||
##   6 |          | Output_1        | -0.63844 | -238.62894 | o  | 1,4,0    | Act_Identity |          |
## ----|----------|-----------------|----------|----------|----|----------|--------------|----------|---
##
##
## connection definition section :
##
## target | site | source:weight
## -------|------|-----------------------------------------------------------------------------------
##      4 |      | 3:-0.04908, 2:-0.04654, 1:-1.07166
##      5 |      | 3:-0.01779, 2: 1.44009, 1: 1.03937
##      6 |      | 5: 2.76215, 4:-6.96684
## -------|------|-----------------------------------------------------------------------------------
```

```
nn1.pred <- predict(advertising.nn1,x.test)
mse <- sum((nn1.pred-y.test)^2)/length(y.test)
mse
```

```
## [1] 0.04150914
```

```
# Plot the results
plot(y.test)
points(nn1.pred, col = "red", pch = 3)
```



1)

a) The results are shown from the R output above.

b) I chose 2 as the number of hidden units because there are 3 input units and 1 output unit, so 2, which is between 3 and 1, is an appropriate choice for the number of hidden units. Alternatively, I could choose 3 as the number of hidden units and look at the model results with regularization. For the learning parameter, I selected 0.01 after comparing model performance with difference learning rates, but could use cross validation to select the best learning parameter as well.

```
## Q2

# Specify 9 kinds of hidden unit configurations
size_list <- list(c(2),c(3),c(3,3),c(3,2),c(2,2),c(3,3,3),c(3,3,2),c(3,2,2),c(2,2,2))
nn.pred <- nn1.pred

for (i in 1:9)
{
  for (j in 1:4) # Use 4 different starting values for each configuration
  {
    advertising.nn_temp <- RSNNS::mlp(x=x.train,y=y.train,size = size_list[[i]], maxit = 10000, learnFu
    nn.pred <- cbind(nn.pred, predict(advertising.nn_temp,x.test))
  }
}
```

```
# Select 30 models from 36 models estimated
nn.pred <- nn.pred[,-1]
selection.id <- sample(1:36,30)
nn.pred <- nn.pred[,selection.id]

# Calculate the ensemble predictions and calculate MSE
nn_ensemble.pred <- rowMeans(nn.pred)
mse_ensemble <- sum((nn_ensemble.pred-y.test)^2)/length(y.test)
mse_ensemble
```

```
## [1] 0.007502991
```

2)

I selected 30 neural networks by varying both hidden layer configuration (number of hidden of layers and number of hidden units in each layer) and starting values. The MSE of the ensemble method turns out to be about 0.0075, which is much smaller than the MSE obtained from Q1, suggesting the ensemble method improves the predictve accuracy by a lot.

The reason for the improvement is perhaps because calibrated weights will depend on the chosen random starting points and averaging predictions with different starting points reduce the variance by a lot.

```
## Q3

library(rugarch)
```

```
## Warning: package 'rugarch' was built under R version 3.3.3
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
data("dji30ret")
summary(dji30ret)
```

```
##        AA                  AXP                  BA
##  Min.   :-0.2745595   Min.   :-0.3034304   Min.   :-0.1938568
##  1st Qu.:-0.0114593   1st Qu.:-0.0109291   1st Qu.:-0.0098007
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0001608   Mean   : 0.0001687   Mean   : 0.0003058
##  3rd Qu.: 0.0116377   3rd Qu.: 0.0114812   3rd Qu.: 0.0105709
##  Max.   : 0.2087337   Max.   : 0.1712035   Max.   : 0.1439727
##       BAC                   C                   CAT
##  Min.   :-0.3420588   Min.   :-0.3056056   Min.   :-0.244156
##  1st Qu.:-0.0093365   1st Qu.:-0.0111602   1st Qu.:-0.010575
##  Median : 0.0000000   Median : 0.0000000   Median : 0.000000
```

```
##  Mean   : 0.0001149   Mean   : 0.0000796   Mean   : 0.000378
##  3rd Qu.: 0.0100293   3rd Qu.: 0.0116803   3rd Qu.: 0.011141
##  Max.   : 0.2698774   Max.   : 0.4572902   Max.   : 0.137371
##       CVX                  DD                   DIS
##  Min.   :-0.1812526   Min.   :-0.2018984   Min.   :-0.3426451
##  1st Qu.:-0.0082829   1st Qu.:-0.0093255   1st Qu.:-0.0102932
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004538   Mean   : 0.0001774   Mean   : 0.0002886
##  3rd Qu.: 0.0095239   3rd Qu.: 0.0095836   3rd Qu.: 0.0105821
##  Max.   : 0.1894765   Max.   : 0.1086964   Max.   : 0.1756133
##       GE                   GM                   HD
##  Min.   :-0.1947441   Min.   :-0.3727220   Min.   :-0.3386365
##  1st Qu.:-0.0083683   1st Qu.:-0.0119502   1st Qu.:-0.0115889
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0002751   Mean   :-0.0002715   Mean   : 0.0006922
##  3rd Qu.: 0.0093459   3rd Qu.: 0.0115692   3rd Qu.: 0.0127656
##  Max.   : 0.1275967   Max.   : 0.3009365   Max.   : 0.1315251
##       HPQ                  IBM                  INTC
##  Min.   :-0.2263815   Min.   :-0.268161   Min.   :-0.2488610
##  1st Qu.:-0.0125577   1st Qu.:-0.009243   1st Qu.:-0.0139915
##  Median : 0.0000000   Median : 0.000000   Median : 0.0000000
##  Mean   : 0.0003792   Mean   : 0.000249   Mean   : 0.0005553
##  3rd Qu.: 0.0135366   3rd Qu.: 0.009469   3rd Qu.: 0.0158734
##  Max.   : 0.1591410   Max.   : 0.123635   Max.   : 0.2265276
##       JNJ                  JPM                  AIG
##  Min.   :-0.2043813   Min.   :-0.3234769   Min.   :-0.9362581
##  1st Qu.:-0.0078036   1st Qu.:-0.0111299   1st Qu.:-0.0089217
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004993   Mean   : 0.0002586   Mean   :-0.0002978
##  3rd Qu.: 0.0084695   3rd Qu.: 0.0111094   3rd Qu.: 0.0094814
##  Max.   : 0.1153126   Max.   : 0.2239172   Max.   : 0.3585320
##       KO                   MCD                  MMM
##  Min.   :-0.2828628   Min.   :-0.1827990   Min.   :-0.2257926
##  1st Qu.:-0.0080020   1st Qu.:-0.0093024   1st Qu.:-0.0074074
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004333   Mean   : 0.0004514   Mean   : 0.0003322
##  3rd Qu.: 0.0089027   3rd Qu.: 0.0099834   3rd Qu.: 0.0082499
##  Max.   : 0.1791113   Max.   : 0.1030806   Max.   : 0.1049975
##       MRK                  MSFT                 PFE
##  Min.   :-0.3119154   Min.   :-0.379490   Min.   :-0.1892420
##  1st Qu.:-0.0090772   1st Qu.:-0.010643   1st Qu.:-0.0096386
##  Median : 0.0000000   Median : 0.000000   Median : 0.0000000
##  Mean   : 0.0003447   Mean   : 0.000787   Mean   : 0.0003885
##  3rd Qu.: 0.0100307   3rd Qu.: 0.012848   3rd Qu.: 0.0107528
##  Max.   : 0.1224923   Max.   : 0.178465   Max.   : 0.0989399
##       PG                   T                    UTX
##  Min.   :-0.3598917   Min.   :-0.1352280   Min.   :-0.3029024
##  1st Qu.:-0.0074074   1st Qu.:-0.0087377   1st Qu.:-0.0083770
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004917   Mean   : 0.0003364   Mean   : 0.0004517
##  3rd Qu.: 0.0085107   3rd Qu.: 0.0096619   3rd Qu.: 0.0098064
##  Max.   : 0.1980376   Max.   : 0.1505242   Max.   : 0.1278841
##       VZ                   WMT                  XOM
##  Min.   :-0.1931448   Min.   :-0.1249721   Min.   :-0.2676996
```

```
##  1st Qu.:-0.0087802   1st Qu.:-0.0100137   1st Qu.:-0.0078818
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0002848   Mean   : 0.0004985   Mean   : 0.0004964
##  3rd Qu.: 0.0089366   3rd Qu.: 0.0104713   3rd Qu.: 0.0090419
##  Max.   : 0.1365130   Max.   : 0.1146918   Max.   : 0.1653925
```

```
# Construct the data matrix with Ys and one-lag Xs
dji30ret$AVG <- rowMeans(dji30ret)
summary(dji30ret)
```

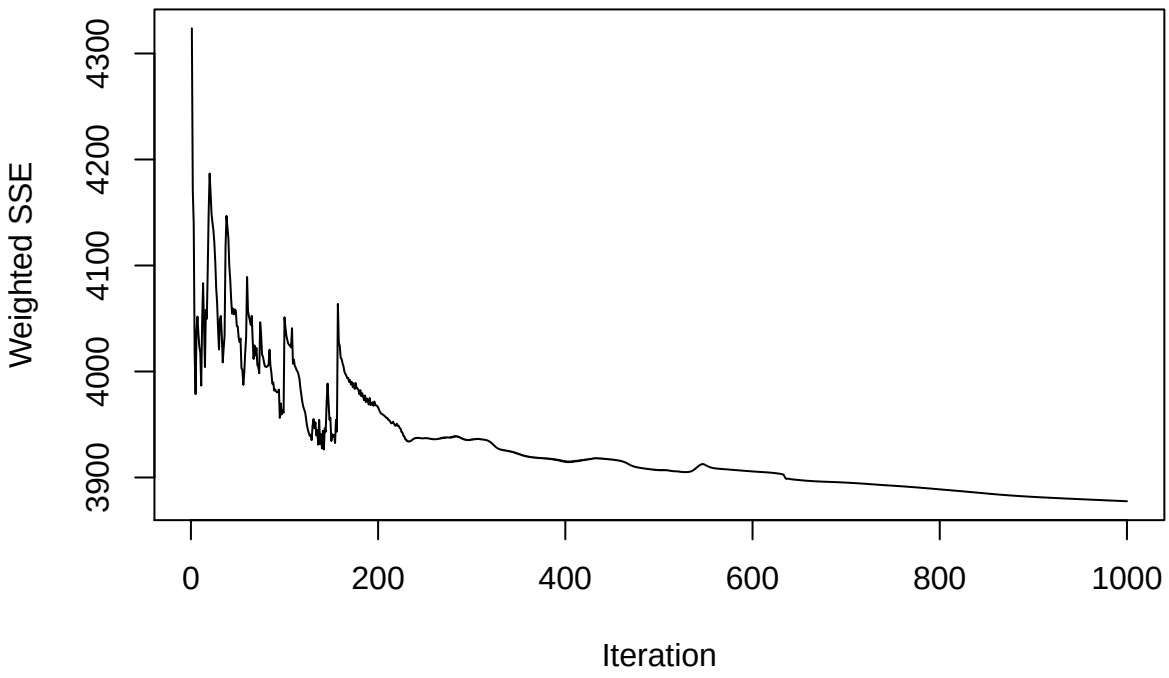
```
##        AA                  AXP                  BA
##  Min.   :-0.2745595   Min.   :-0.3034304   Min.   :-0.1938568
##  1st Qu.:-0.0114593   1st Qu.:-0.0109291   1st Qu.:-0.0098007
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0001608   Mean   : 0.0001687   Mean   : 0.0003058
##  3rd Qu.: 0.0116377   3rd Qu.: 0.0114812   3rd Qu.: 0.0105709
##  Max.   : 0.2087337   Max.   : 0.1712035   Max.   : 0.1439727
##       BAC                   C                   CAT
##  Min.   :-0.3420588   Min.   :-0.3056056   Min.   :-0.244156
##  1st Qu.:-0.0093365   1st Qu.:-0.0111602   1st Qu.:-0.010575
##  Median : 0.0000000   Median : 0.0000000   Median : 0.000000
##  Mean   : 0.0001149   Mean   : 0.0000796   Mean   : 0.000378
##  3rd Qu.: 0.0100293   3rd Qu.: 0.0116803   3rd Qu.: 0.011141
##  Max.   : 0.2698774   Max.   : 0.4572902   Max.   : 0.137371
##       CVX                   DD                  DIS
##  Min.   :-0.1812526   Min.   :-0.2018984   Min.   :-0.3426451
##  1st Qu.:-0.0082829   1st Qu.:-0.0093255   1st Qu.:-0.0102932
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004538   Mean   : 0.0001774   Mean   : 0.0002886
##  3rd Qu.: 0.0095239   3rd Qu.: 0.0095836   3rd Qu.: 0.0105821
##  Max.   : 0.1894765   Max.   : 0.1086964   Max.   : 0.1756133
##        GE                   GM                   HD
##  Min.   :-0.1947441   Min.   :-0.3727220   Min.   :-0.3386365
##  1st Qu.:-0.0083683   1st Qu.:-0.0119502   1st Qu.:-0.0115889
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0002751   Mean   :-0.0002715   Mean   : 0.0006922
##  3rd Qu.: 0.0093459   3rd Qu.: 0.0115692   3rd Qu.: 0.0127656
##  Max.   : 0.1275967   Max.   : 0.3009365   Max.   : 0.1315251
##       HPQ                  IBM                 INTC
##  Min.   :-0.2263815   Min.   :-0.268161   Min.   :-0.2488610
##  1st Qu.:-0.0125577   1st Qu.:-0.009243   1st Qu.:-0.0139915
##  Median : 0.0000000   Median : 0.000000   Median : 0.0000000
##  Mean   : 0.0003792   Mean   : 0.000249   Mean   : 0.0005553
##  3rd Qu.: 0.0135366   3rd Qu.: 0.009469   3rd Qu.: 0.0158734
##  Max.   : 0.1591410   Max.   : 0.123635   Max.   : 0.2265276
##       JNJ                  JPM                  AIG
##  Min.   :-0.2043813   Min.   :-0.3234769   Min.   :-0.9362581
##  1st Qu.:-0.0078036   1st Qu.:-0.0111299   1st Qu.:-0.0089217
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004993   Mean   : 0.0002586   Mean   :-0.0002978
##  3rd Qu.: 0.0084695   3rd Qu.: 0.0111094   3rd Qu.: 0.0094814
##  Max.   : 0.1153126   Max.   : 0.2239172   Max.   : 0.3585320
##        KO                   MCD                  MMM
##  Min.   :-0.2828628   Min.   :-0.1827990   Min.   :-0.2257926
```

```
##  1st Qu.:-0.0080020   1st Qu.:-0.0093024   1st Qu.:-0.0074074
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004333   Mean   : 0.0004514   Mean   : 0.0003322
##  3rd Qu.: 0.0089027   3rd Qu.: 0.0099834   3rd Qu.: 0.0082499
##  Max.   : 0.1791113   Max.   : 0.1030806   Max.   : 0.1049975
##       MRK                  MSFT                PFE
##  Min.   :-0.3119154   Min.   :-0.379490   Min.   :-0.1892420
##  1st Qu.:-0.0090772   1st Qu.:-0.010643   1st Qu.:-0.0096386
##  Median : 0.0000000   Median : 0.000000   Median : 0.0000000
##  Mean   : 0.0003447   Mean   : 0.000787   Mean   : 0.0003885
##  3rd Qu.: 0.0100307   3rd Qu.: 0.012848   3rd Qu.: 0.0107528
##  Max.   : 0.1224923   Max.   : 0.178465   Max.   : 0.0989399
##        PG                    T                   UTX
##  Min.   :-0.3598917   Min.   :-0.1352280   Min.   :-0.3029024
##  1st Qu.:-0.0074074   1st Qu.:-0.0087377   1st Qu.:-0.0083770
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0004917   Mean   : 0.0003364   Mean   : 0.0004517
##  3rd Qu.: 0.0085107   3rd Qu.: 0.0096619   3rd Qu.: 0.0098064
##  Max.   : 0.1980376   Max.   : 0.1505242   Max.   : 0.1278841
##        VZ                   WMT                 XOM
##  Min.   :-0.1931448   Min.   :-0.1249721   Min.   :-0.2676996
##  1st Qu.:-0.0087802   1st Qu.:-0.0100137   1st Qu.:-0.0078818
##  Median : 0.0000000   Median : 0.0000000   Median : 0.0000000
##  Mean   : 0.0002848   Mean   : 0.0004985   Mean   : 0.0004964
##  3rd Qu.: 0.0089366   3rd Qu.: 0.0104713   3rd Qu.: 0.0090419
##  Max.   : 0.1365130   Max.   : 0.1146918   Max.   : 0.1653925
##       AVG
##  Min.   :-0.2259355
##  1st Qu.:-0.0053170
##  Median : 0.0005850
##  Mean   : 0.0003255
##  3rd Qu.: 0.0061890
##  Max.   : 0.1151826
```

```r
xfactors <- dji30ret[-nrow(dji30ret),-ncol(dji30ret)]
yfactors <- dji30ret[-1,ncol(dji30ret)]
dji_data <- as.matrix(cbind(xfactors,yfactors))
dji_data <- scale(dji_data)

# a)
# Split into training and test sets
split_date.id <- which(dji_data[,1]==dji_data["2005-12-30",1]) + 1
dji_train <- dji_data[1:split_date.id,]
dji_test <- dji_data[split_date.id:nrow(dji_data),]

# Fit a Elman neural network
library(RSNNS)
x_train <- dji_train[,-ncol(dji_train)]
y_train <- dji_train[,ncol(dji_train)]
dji.elman <- elman(x_train, y_train, size = c(5), learnFuncParams = c(0.1), maxit = 1000)
plotIterativeError(dji.elman)
```




```
summary(dji.elman)
```

```
## SNNS network definition file V1.4-3D
## generated at Sat Apr 29 17:14:51 2017
##
## network name : RSNNS_untitled
## source files :
## no. of units : 41
## no. of connections : 190
## no. of unit types : 0
## no. of site types : 0
##
##
## learning function : JE_BP
## update function   : JE_Order
##
##
## unit default section :
##
## act      | bias     | st | subnet | layer | act func     | out func
## ---------|----------|----|--------|-------|--------------|-------------
##  1.00000 |  0.00000 | i  |      0 |     1 | Act_Logistic | Out_Identity
## ---------|----------|----|--------|-------|--------------|-------------
##
##
```

```
## unit definition section :
##
## no. | typeName | unitName | act      | bias     | st | position | act func     | out func | sites
## ----|----------|----------|----------|----------|----|----------|--------------|----------|-------
##   1 |          | inp1     |  0.47778 | -0.91907 | i  |  1, 1, 0 | Act_Identity |          |
##   2 |          | inp2     |  0.92574 | -0.12462 | i  |  1, 2, 0 | Act_Identity |          |
##   3 |          | inp3     |  0.13042 | -0.48560 | i  |  1, 3, 0 | Act_Identity |          |
##   4 |          | inp4     |  0.81756 |  0.06269 | i  |  1, 4, 0 | Act_Identity |          |
##   5 |          | inp5     |  0.56515 | -0.87587 | i  |  1, 5, 0 | Act_Identity |          |
##   6 |          | inp6     |  0.00867 |  0.36288 | i  |  1, 6, 0 | Act_Identity |          |
##   7 |          | inp7     |  2.39554 |  0.89557 | i  |  1, 7, 0 | Act_Identity |          |
##   8 |          | inp8     |  0.71180 |  0.24236 | i  |  1, 8, 0 | Act_Identity |          |
##   9 |          | inp9     |  0.85728 | -0.75426 | i  |  1, 9, 0 | Act_Identity |          |
##  10 |          | inp10    |  0.49591 |  0.53197 | i  |  1,10, 0 | Act_Identity |          |
##  11 |          | inp11    | -1.04595 | -0.73817 | i  |  1,11, 0 | Act_Identity |          |
##  12 |          | inp12    |  0.79020 | -0.01690 | i  |  1,12, 0 | Act_Identity |          |
##  13 |          | inp13    |  0.18284 | -0.88500 | i  |  1,13, 0 | Act_Identity |          |
##  14 |          | inp14    | -0.09957 |  0.06298 | i  |  1,14, 0 | Act_Identity |          |
##  15 |          | inp15    |  0.84702 |  0.48217 | i  |  1,15, 0 | Act_Identity |          |
##  16 |          | inp16    |  1.58807 | -0.88437 | i  |  1,16, 0 | Act_Identity |          |
##  17 |          | inp17    |  0.48330 |  0.24819 | i  |  1,17, 0 | Act_Identity |          |
##  18 |          | inp18    |  0.70600 |  0.76945 | i  |  1,18, 0 | Act_Identity |          |
##  19 |          | inp19    |  0.84114 | -0.84202 | i  |  1,19, 0 | Act_Identity |          |
##  20 |          | inp20    | -0.36057 |  0.84574 | i  |  1,20, 0 | Act_Identity |          |
##  21 |          | inp21    |  1.31152 |  0.08409 | i  |  1,21, 0 | Act_Identity |          |
##  22 |          | inp22    |  1.55195 |  0.97354 | i  |  1,22, 0 | Act_Identity |          |
##  23 |          | inp23    |  1.04920 |  0.17025 | i  |  1,23, 0 | Act_Identity |          |
##  24 |          | inp24    |  1.02713 | -0.06105 | i  |  1,24, 0 | Act_Identity |          |
##  25 |          | inp25    |  0.89237 |  0.96711 | i  |  1,25, 0 | Act_Identity |          |
##  26 |          | inp26    |  0.48688 | -0.98152 | i  |  1,26, 0 | Act_Identity |          |
##  27 |          | inp27    |  0.58224 | -0.57759 | i  |  1,27, 0 | Act_Identity |          |
##  28 |          | inp28    |  0.49715 | -0.77045 | i  |  1,28, 0 | Act_Identity |          |
##  29 |          | inp29    | -0.67272 |  0.74644 | i  |  1,29, 0 | Act_Identity |          |
##  30 |          | inp30    |  2.44071 |  0.74061 | i  |  1,30, 0 | Act_Identity |          |
##  31 |          | hid1     |  0.00000 | -29.75155 | h  |  7, 1, 0 |||
##  32 |          | hid2     |  0.00000 | -36.35191 | h  |  7, 2, 0 |||
##  33 |          | hid3     |  0.00000 | -40.83426 | h  |  7, 3, 0 |||
##  34 |          | hid4     |  0.00000 | -30.64052 | h  |  7, 4, 0 |||
##  35 |          | hid5     |  0.00000 | -34.01869 | h  |  7, 5, 0 |||
##  36 |          | out1     | -0.00000 | 6470.86182 | o  | 13, 1, 0 | Act_Identity |          |
##  37 |          | con1     |  0.00780 |  0.50000 | sh  |  4,32, 0 | Act_Identity |          |
##  38 |          | con2     |  0.00000 |  0.50000 | sh  |  4,33, 0 | Act_Identity |          |
##  39 |          | con3     |  0.00000 |  0.50000 | sh  |  4,34, 0 | Act_Identity |          |
##  40 |          | con4     |  0.00042 |  0.50000 | sh  |  4,35, 0 | Act_Identity |          |
##  41 |          | con5     |  0.00000 |  0.50000 | sh  |  4,36, 0 | Act_Identity |          |
## ----|----------|----------|----------|----------|----|----------|--------------|----------|-------
##
##
## connection definition section :
##
## target | site | source:weight
## -------|------|-----------------------------------------------------------------------------------
##     31 |      | 41:-2.17612, 40:-4.18329, 39: 4.19223, 38: 6.08403, 37: 0.07002, 30:-5.80402, 29:-3.
##                 26:-4.21796, 25: 0.82071, 24:-1.59751, 23:-1.13426, 22:-10.43551, 21:-6.95692, 20:-2
```

```
##                 17:-6.51633, 16: 5.45805, 15: 1.60647, 14:-2.99970, 13:-1.76194, 12:-1.55023, 11:-0.8
##                  8: 0.98661,  7: 3.63648,  6: 0.10857,  5:-3.29318,  4: 3.19364,  3:-9.52617,  2:-6.4
##     32 |      | 41:-3.29806, 40:-9.51252, 39:-2.22531, 38: 7.07721, 37: 4.13384, 30: 8.99229, 29: 4.9
##                 26:-9.90123, 25:-2.40001, 24:-8.01598, 23:-3.72813, 22:-2.20097, 21:-3.96336, 20: 1.0
##                 17:-4.47132, 16: 0.21522, 15:-7.95894, 14:-6.05867, 13: 0.47383, 12:-3.47754, 11:-2.1
##                  8:-1.59087,  7: 0.21432,  6:-3.64856,  5: 7.08441,  4: 3.94321,  3:-2.52364,  2:-0.0
##     33 |      | 41: 6.21000, 40:-0.27678, 39:-1.78365, 38: 6.19060, 37: 2.29166, 30:-0.10326, 29: 1.0
##                 26:-4.37074, 25:-3.78208, 24:-6.97006, 23:-5.20621, 22:-3.23019, 21:-3.53245, 20: 5.3
##                 17:-9.63702, 16:-0.08717, 15:-7.80373, 14:-0.47953, 13: 2.31084, 12:-7.66372, 11:-5.7
##                  8: 1.80991,  7: 6.45970,  6: 4.16970,  5: 1.45316,  4:-3.56986,  3: 1.69088,  2: 5.5
##     34 |      | 41:-3.10743, 40: 1.11616, 39:-7.36880, 38:-4.51558, 37:-5.85712, 30:-7.11287, 29:-1.7
##                 26:-6.51610, 25: 0.24658, 24:-5.01959, 23:-5.46858, 22:-3.33413, 21: 0.88455, 20: 0.0
##                 17:-3.74158, 16:-2.89301, 15:-2.27218, 14:-0.65667, 13:-1.36925, 12:-7.86950, 11:-6.0
##                  8:-3.91973,  7: 0.57402,  6:-0.96416,  5: 3.77501,  4: 0.75505,  3:-0.10111,  2:-6.0
##     35 |      | 41: 7.26769, 40:-6.66623, 39:-6.61548, 38:10.34146, 37:-7.16306, 30:-4.06458, 29:-7.1
##                 26:-5.05459, 25:-1.70842, 24: 0.14433, 23:-7.31349, 22:-3.02293, 21:-0.95516, 20: 1.0
##                 17:-3.43398, 16:-6.01401, 15:-4.77596, 14: 0.15178, 13:-2.53260, 12:-3.01854, 11:-2.9
##                  8: 3.45483,  7: 7.31434,  6: 5.43265,  5: 1.12381,  4:-5.09163,  3: 1.55712,  2:-0.0
##     36 |      | 35:-0.66226, 34:-0.07983, 33: 1.65654, 32:-1.49724, 31: 0.42017
##     37 |      | 37: 0.30000, 31: 1.00000
##     38 |      | 38: 0.30000, 32: 1.00000
##     39 |      | 39: 0.30000, 33: 1.00000
##     40 |      | 40: 0.30000, 34: 1.00000
##     41 |      | 41: 0.30000, 35: 1.00000
## -------|------|-------------------------------------------------------------------------------------
```

```
# b)
# Make predictions and calculate MSE
x_test <- dji_test[,-ncol(dji_test)]
y_test <- dji_test[,ncol(dji_test)]
elman.pred <- predict(dji.elman, x_test)
mse_elman <- sum((elman.pred-y_test)^2)/length(y_test)
mse_elman
```

```
## [1] 2.19211
```

3)

a) The results are shown from the R output above.

b) The results are shown from the R output above.

c) Elman neural network is an appropriate model for time-series prediction, as the context layer within the Elman neural network "remember" the previous internal state of the network by storing hidden layer neuron values. The number of hidden units chosen for the model is 5, much fewer than the number of input units, which is 30, suggesting that overfitting may not occur. We could also use regularization to minimize the risk of overfitting.

```
## Q4
# Build transformed dataset with 5 lagged returns
library(ISLR)
data("Weekly")
summary(Weekly)
```

```
##       Year           Lag1                Lag2                Lag3          
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4                Lag5              Volume       
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202  
##  Median :  0.2380   Median :  0.2340   Median :1.00268  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821  
##      Today          Direction 
##  Min.   :-18.1950   Down:484  
##  1st Qu.: -1.1540   Up  :605  
##  Median :  0.2410             
##  Mean   :  0.1499             
##  3rd Qu.:  1.4050             
##  Max.   : 12.0260
```

```
# Split into training and test sets
sp_500.x_train <- head(scale(as.matrix(Weekly[,2:6])),889)
sp_500.y_train <- head(scale(as.matrix(Weekly[,8])),889)

# Build the Deep Belief Net (DBN)
library(deepnet)
sp_500.dbn <- dbn.dnn.train(sp_500.x_train, sp_500.y_train, hidden = c(4,3), output = "linear", learnin
```

```
## begin to train dbn ......

## training layer 1 rbm ...

## training layer 2 rbm ...

## dbn has been trained.

## begin to train deep nn ......

## ####loss on step 10000 is : 0.326920

## ####loss on step 20000 is : 0.838359

## ####loss on step 30000 is : 0.258900

## ####loss on step 40000 is : 0.733237

## ####loss on step 50000 is : 0.559317

## ####loss on step 60000 is : 0.395439
```

```
## ####loss on step 70000 is : 0.722998

## ####loss on step 80000 is : 0.297470

## deep nn has been trained.
```

```r
# Make predictions and calculate MSE
sp_500.x_test <- tail(scale(as.matrix(Weekly[,2:6])),200)
sp_500.y_test <- tail(scale(as.matrix(Weekly[,8])),200)
dbn.pred <- nn.predict(sp_500.dbn, sp_500.x_test)
mse_dbn <- sum((dbn.pred-sp_500.y_test)^2)/length(sp_500.y_test)
mse_dbn
```

```
## [1] 2.04773
```

4)

a) The results are shown from the R output above.

b) The results are shown from the R output above.

c) I used c(4,3) as the number of hidden units because the number of inputs is 5, so I want to make the number of hidden units smaller or equal to 5 to extract meaningful features from the input. I also tried many other hidden layer configurations that satisfy this rule. Overall, My finding is that after trying 100+ set of parameters for the Deep Belief Network, the DBN still produces predictions that do not make much sense. Specifically, the distribution of the predictions do not align very closely with the distribution of the y_test variable. This might be because the package has some problem handling this dataset.