Data mining 5th Report  
(Student Number: 1651046, Name: Daiki Kubo)

1. Train a neural network that calculate square root of the numbers from 100 to 200. Please follow the following website: http://gekkoquant.com/2012/05/26/neural-networks-with-r-simple-example/

Try the neural network using some test data and make a short report (include the figure) on what you did (within 1 or 2 page).  
  
**Ans.**--------------------------------------------------------------------------------------- **Source Code:**library("neuralnet")  
  
traininginput <-  [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)(runif(50, [min](http://astrostatistics.psu.edu/su07/R/html/graphics/html/min.html)=100, [max](http://astrostatistics.psu.edu/su07/R/html/graphics/html/max.html)=200))  
trainingoutput <- [sqrt](http://astrostatistics.psu.edu/su07/R/html/graphics/html/sqrt.html)(traininginput)  
   
trainingdata <- [cbind](http://astrostatistics.psu.edu/su07/R/html/graphics/html/cbind.html)(traininginput,trainingoutput)  
[colnames](http://astrostatistics.psu.edu/su07/R/html/graphics/html/colnames.html)(trainingdata) <- [c](http://astrostatistics.psu.edu/su07/R/html/graphics/html/c.html)("Input","Output")  
   
# Train the neural network  
net.sqrt <- neuralnet(Output~Input,trainingdata, hidden=10, threshold=0.01)  
print(net.sqrt)  
   
# Test the neural network on some training data  
testdata <- as.data.frame((100:200)^2) # 100:200  
# testdata <- [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)((500:600)^2) # 500:600  
net.results <- compute(net.sqrt, testdata) #Run them through the neural network  
  
# Lets display a better version of the results  
cleanoutput <- [cbind](http://astrostatistics.psu.edu/su07/R/html/graphics/html/cbind.html)(testdata,[sqrt](http://astrostatistics.psu.edu/su07/R/html/graphics/html/sqrt.html)(testdata),  
  [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)(net.results$net.result))  
[colnames](http://astrostatistics.psu.edu/su07/R/html/graphics/html/colnames.html)(cleanoutput) <- [c](http://astrostatistics.psu.edu/su07/R/html/graphics/html/c.html)("Input","Expected Output","Neural Net Output")  
[print](http://astrostatistics.psu.edu/su07/R/html/graphics/html/print.html)(cleanoutput)

-----------------------------------------------------------------------------------------------------  
**Short Reports:**・ ソースコードの変更箇所は以下である。  
変更前: traininginput <-  [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)(runif(50, min=0, [max](http://astrostatistics.psu.edu/su07/R/html/graphics/html/max.html)=100))  
変更後: traininginput <-  [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)(runif(50, [min](http://astrostatistics.psu.edu/su07/R/html/graphics/html/min.html)=100, [max](http://astrostatistics.psu.edu/su07/R/html/graphics/html/max.html)=200))  
・ 使用したテストデータは、以下の２つである。  
1: testdata <- as.data.frame((100:200)^2) # 100:200  
2: testdata <- [as.data.frame](http://astrostatistics.psu.edu/su07/R/html/graphics/html/as.data.frame.html)((500:600)^2) # 500:600  
-----------------------------------------------------------------------------------------------------  
2. Apply SVM, NN and SOM to the dataset "wine {HDclassif}" in light of the tutorial.

Loading wine dataset > library(HDclassif)

> data(wine)

**Ans.**   
  
library("kernlab")

library("som")

library("nnet")

library("HDclassif")

data("wine")

data <- wine

train <- data[seq.int(1,178, by=2),]

test <- data[setdiff(1:178, train),]

# SVM

train.svm <- ksvm(class ~., data=train, type="C-bsvc", kernel　="rbfdot")

test.predict <- predict(train.svm,test)

table(test.predict,test$class)

# NN

train.nnet <- nnet(class ~., size=13, data=train)

test.predict <- predict(train.nnet, test, type="class")

table(test$class, test.predict)

# SOM

model <- som(data[,-1] , xdim=10, ydim=10)

label <- as.numeric(data[,1])

plot(model$visual$x, model$visual$y, pch = label,

col = label, cex = 2)

**Results:**

test.predict 1 2 3

1 59 0 0

2 0 70 0

3 0 1 48

test.predict 1

1 59

2 71

3 48

