Target: Species (setosa, versicolor, and virginica)

Inputs: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width.

##01.환경설정 및 데이터 불러오기

library(nnet)

data(iris)

##02.훈련 및 검증 데이터 분리 (7:3)

set.seed(1111)

train.idx <- c(sample(1:50,35),sample(51:100,35),sample(101:150,35))

iris.train <- iris[train.idx , ]

iris.test <- iris[-train.idx, ]

#200번 반복하여 가장 좋은 모델을 뽑아라 (Neural Network modeling)

model.nnet<-nnet(Species~.,data=iris.train,size=2,decay=5e-04,maxit=200)

##은닉노드,출력노드 수가 나옴.

summary(model.nnet)

##nnet 함수의 결과에서 제공하는 항목 들을 살펴볼 수 있다.

names(model.nnet)

##가장 좋은 가중치의 집합(모수추정치)

model.nnet$wts

##훈련 데이터를 모델로 적합한 결과(output추정치)

head(model.nnet$fitted.values)

##훈련 데이터의 잔차

head(model.nnet$residuals)

##테스트 집합에 대한 예측 결과를 벡터의 형태로 제공,predict 함수를 이용하여 예측이 잘 되었는지 확인

predicted <- predict(model.nnet, iris.test, type = "class")

predicted

##오분류표를 만들기 위해서 table 함수를 이용하여 실제 결과와 예측 결과에 대한 교차표를 작성한다.  
actual <- iris.test$Species  
confusion.matrix <- table(actual, predicted)

confusion.matrix

##비율(%)로 나타낸 오분류표는 prop.table 함수를 이용하여 다음과 같이 구한다. round 함수를 이용하여 소수 첫째 자리에서 반올림하였다.  
round(prop.table(confusion.matrix) \* 100, digit = 1)

##hidden unit의 수에 따른 test error

test.err<-function(h.size)

{

model.nnet <- nnet(Species~.,data=iris.test,size= h.size,decay=5e-4,trace=F)

actual<-iris.test$Species

predicted <- predict(model.nnet, iris.test, type = "class")

err<-mean(actual!=predicted)

c(h.size,err)

}  
out<- t(sapply(1:10,FUN=test.err))

plot(out,type="b",xlab="The number of Hidden units",ylab="Test Error")

# 시각화 R 코드 함수 다운로드

library(devtools)

source\_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4684a5/nnet\_plot\_update.r')

# 신경망 모형 시각화

library(reshape2)

plot.nnet(model.nnet)

#Lek 프로파일 방법을 사용하여 lekprofile 함수로 민감도 분석도 가능하다.

library(NeuralNetTools)

lekprofile(model.nnet)

.##02.훈련 및 검증 데이터 분리 (7:3)

set.seed(1234)

train.idx <- c(sample(1:59,41),sample(60:118,41),sample(119:178,42))

wine.train <- wine[train.idx , ]

wine.test <- wine[-train.idx, ]

--------------------------------------------------------------------------------------------------------------

wine<-read.csv("C:/Users/Kang/Desktop/wine.csv")

##01.환경설정 및 데이터 불러오기

library(nnet)

##02.훈련 및 검증 데이터 분리 (7:3)

wine.scale <- cbind(wine[1], scale(wine[-1]))

size <- nrow(wine.scale)

set.seed(100)

index <- c(sample(1:size, size \* 0.7))

wine.train <- wine.scale[index, ]

wine.test <- wine.scale[-index, ]

#200번 반복하여 가장 좋은 모델을 뽑아라 (Neural Network modeling)

model.nnet<-nnet(Type~.,data=wine.train,size=5,decay=5e-04,maxit=200)

##은닉노드,출력노드 수가 나옴.

summary(model.nnet)

##nnet 함수의 결과에서 제공하는 항목 들을 살펴볼 수 있다.

names(model.nnet)

##가장 좋은 가중치의 집합(모수추정치)

model.nnet$wts

##훈련 데이터를 모델로 적합한 결과(output추정치)

head(model.nnet$fitted.values)

##훈련 데이터의 잔차

head(model.nnet$residuals)

##테스트 집합에 대한 예측 결과를 벡터의 형태로 제공,predict 함수를 이용하여 예측이 잘 되었는지 확인

predicted <- predict(model.nnet, wine.test, type = "class")

predicted

##오분류표를 만들기 위해서 table 함수를 이용하여 실제 결과와 예측 결과에 대한 교차표를 작성한다.

actual <- wine.test$Type

confusion.matrix <- table(actual, predicted)

confusion.matrix

##비율(%)로 나타낸 오분류표는 prop.table 함수를 이용하여 다음과 같이 구한다. round 함수를 이용하여 소수 첫째 자리에서 반올림하였다.

round(prop.table(confusion.matrix) \* 100, digit = 1)

##hidden unit의 수에 따른 test error

test.err<-function(h.size)

{

model.nnet <- nnet(Type~.,data=wine.test,size= h.size,decay=5e-4,trace=F)

actual<-wine.test$Type

predicted <- predict(model.nnet, wine.test, type = "class")

err<-mean(actual!=predicted)

c(h.size,err)

}

out<- t(sapply(1:5,FUN=test.err))

plot(out,type="b",xlab="The number of Hidden units",ylab="Test Error")

# 시각화 R 코드 함수 다운로드

library(devtools)

source\_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4684a5/nnet\_plot\_update.r')

# 신경망 모형 시각화

library(reshape2)

plot.nnet(model.nnet)

#Lek 프로파일 방법을 사용하여 lekprofile 함수로 민감도 분석도 가능하다.

library(NeuralNetTools)

lekprofile(model.nnet)

<http://www.datamarket.kr/xe/board_BoGi29/6641>  
<http://m.blog.daum.net/sys4ppl/3>  
<http://blog.naver.com/PostView.nhn?blogId=leedk1110&logNo=220804356934>

http://statkclee.github.io/deep-learning/r-nnet.html