HW5

Jingshi

2/24/2017

Question 2

a)

```
library(nnet)
set.seed(1234)
Advertising <- read.csv("Advertising.csv")

lm_fit <-lm(Sales ~ TV + Radio + Newspaper, data=Advertising)
set.seed(1234)
linear_fit <- nnet(Sales ~ TV + Radio + Newspaper,
                   size = 0,  data=Advertising, skip = T,
                   trace = F,maxit = 1000, linout = T)

summary(linear_fit)
```

```
## a 3-0-1 network with 4 weights
## options were - skip-layer connections  linear output units
##  b->o i1->o i2->o i3->o
##  2.94  0.05  0.19  0.00
```

```
summary(lm_fit)
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = Advertising)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.938889   0.311908   9.422   <2e-16 ***
## TV           0.045765   0.001395  32.809   <2e-16 ***
## Radio        0.188530   0.008611  21.893   <2e-16 ***
## Newspaper   -0.001037   0.005871  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

According to the output of summaries, two models have the same coefficients:

$$\beta_0 = 2.94,\ \beta_1 = 0.05,\ \beta_2 = 0.19,\ \beta_3 = 0.00$$

So they are essentially the same models. Theoratically, they should be the same because a neural network without hidden layer and with linear output is a linear regression.

b)

```
#anova(lm_fit)
# Calculate the SSE
sum(lm_fit$residuals^2)
```

```
## [1] 556.8253
```

```
sum(linear_fit$residuals^2)
```

```
## [1] 556.8253
```

According to the output, the SSE is 556.8253.

c)

```
# Make a neural network that fits Sales to the predictors with a 3 unit hidden layer.
set.seed(1234)
nnet_fit <- nnet(Sales ~ TV + Radio + Newspaper,
                   size = 3,  data=Advertising,
                   trace = F,maxit = 1000, linout = T)

#summary(nnet_fit)
# Calculate the SSE
sum(nnet_fit$residuals^2)
```

```
## [1] 5417.149
```

According to the output, the SSE is 5417.149.

d)

```
Advert2 <- as.data.frame(scale(Advertising))
Advert2$Sales <- Advertising$Sales
head(Advert2)
```

```
##           X          TV      Radio Newspaper Sales
## 1 -1.719098  0.96742460  0.9790656 1.7744925  22.1
## 2 -1.701821 -1.19437904  1.0800974 0.6679027  10.4
## 3 -1.684543 -1.51235985  1.5246374 1.7790842   9.3
## 4 -1.667266  0.05191939  1.2148065 1.2831850  18.5
## 5 -1.649989  0.39319551 -0.8395070 1.2785934  12.9
## 6 -1.632711 -1.61136487  1.7267010 2.0408088   7.2
```

```
head(Advertising)
```

```
##   X    TV Radio Newspaper Sales
## 1 1 230.1  37.8      69.2  22.1
## 2 2  44.5  39.3      45.1  10.4
## 3 3  17.2  45.9      69.3   9.3
```

```
## 4 4 151.5  41.3      58.5  18.5
## 5 5 180.8  10.8      58.4  12.9
## 6 6   8.7  48.9      75.0   7.2
```

According to the output, values of all variables except Sales are changed. The changed values are reduced into a much smaller scale (most are between -2 to 2).

e)

```
# Make a linear model that fits Sales to the scaled predictors in Advert2, using nnet() as in (a).
set.seed(1234)
linear_fit2 <- nnet(Sales ~ TV + Radio + Newspaper,
                    size = 0,  data=Advert2, skip = T,
                    trace = F,maxit = 1000, linout = T)
summary(linear_fit2)
```

```
## a 3-0-1 network with 4 weights
## options were - skip-layer connections  linear output units
##  b->o i1->o i2->o i3->o
## 14.02  3.93  2.80 -0.02
```

This model has exactly the same responses (Sales) as the linear model in (a) because the Sales in Advert2 and Advertising are the same.

```
# Make a neural network that fits Sales to the scaled predictors in Advert2 with a 3 unit hidden layer.
set.seed(1234)
nnet_fit2 <- nnet(Sales ~ TV + Radio + Newspaper,
                    size = 3,  data=Advert2,
                    trace = F,maxit = 1000, linout = T)
```

f)

```
# Compute the SSE for the neural network with 3 units that was made in (e)
# Calculate the SSE
sum(nnet_fit2$residuals^2)
```

```
## [1] 20.85593
```

According to the output, the SSE (20.85593) of nnet_fit2 decreased, compared with the SSE (556.8253) of the linear model that was made in (a) and to the SSE (5417.149) of the neural network mdel that was made in (c), because the predictors are reduced into a much smaller scale in the model.

Question 3

```
load("mnist_data.rdata")
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 3.3.2
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

a)

```
# A function to check whether a pixel has a zero variability
isZero<-function(j){
  res = paste("True, pixel",j,"has a zero variability.")
  for (i in 1 : 2115){
    if (images_df[i,j]!=0){
      res = paste("False, the variability of pixel",j,"is not zero.")
      break
    }
  }
  print(res)
}
# Test to see if pixel 100 has a zero variability.
isZero(100)
```

```
## [1] "False, the variability of pixel 100 is not zero."
```

```
# Test to see if pixel 101 has a zero variability.
isZero(101)
```

```
## [1] "False, the variability of pixel 101 is not zero."
```

The two variables I choose are feature 100 and 101. They have been verified to have non-zero variability.

```
# Fit a logistic regression model with these two features.
model.a<-glm(labels~X100+X101, data = images_df, family = binomial)
summary(model.a)
```

```
##
## Call:
## glm(formula = labels ~ X100 + X101, family = binomial, data = images_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.257  -1.257   1.100   1.100   2.847
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.184251   0.044137   4.175 2.99e-05 ***
## X100        -0.004841   0.003784  -1.279  0.20076
## X101        -0.013184   0.004752  -2.774  0.00553 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 2114  degrees of freedom
## Residual deviance: 2879.8  on 2112  degrees of freedom
## AIC: 2885.8
##
```

```
## Number of Fisher Scoring iterations: 5
```

```
# get predictions
a.predict<-predict(model.a, type = "response")


# compute roc
roc<-roc(images_df$labels, a.predict)

# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5192
```

The AUC is 0.5192 which is greater than 0.5. So the model is ok.

b)

```
# Create a neural net with one unit in the hidden layer.
set.seed(1234)
model.b <- nnet(labels~X100+X101,
                  size = 1,  data=images_df,
                  trace = F,maxit = 1000, linout = T)
summary(model.b)
```

```
## a 2-1-1 network with 5 weights
## options were - linear output units
##  b->h1 i1->h1 i2->h1
##  -0.99   2.41   0.03
##  b->o h1->o
##  0.70 -0.57
```

```
# get predictions
b.predict<-predict(model.b, type = "raw")


# compute roc
roc<-roc(images_df$labels, b.predict)
```

```
## Warning in roc.default(images_df$labels, b.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5191
```

The AUC is 0.5191 which is greater than 0.5 and slightly lower than the previous model. The model is ok.

c)

```
#Create a neural net with three unit in the hidden layer.
set.seed(1234)
model.c <- nnet(labels~X100+X101,
```

```
                  size = 3,  data=images_df,
                  trace = F,maxit = 1000, linout = T)
#summary(model.c)

# get predictions
c.predict<-predict(model.c, type = "raw")


# compute roc
roc<-roc(images_df$labels, c.predict)
```

```
## Warning in roc.default(images_df$labels, c.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5212
```

The AUC is 0.5212 which is greater than 0.5 and higher than the previous two models. The model is better.

d)

```
# Fistly, I would like to find out more features with non-zero variability
# besides 100 and 101.
# Test to see if pixel 102 has a zero variability.
isZero(102)
```

```
## [1] "False, the variability of pixel 102 is not zero."
```

```
# Test to see if pixel 103 has a zero variability.
isZero(103)
```

```
## [1] "False, the variability of pixel 103 is not zero."
```

```
# Test to see if pixel 104 has a zero variability.
isZero(104)
```

```
## [1] "False, the variability of pixel 104 is not zero."
```

```
# Test to see if pixel 105 has a zero variability.
isZero(105)
```

```
## [1] "False, the variability of pixel 105 is not zero."
```

```
# Test to see if pixel 106 has a zero variability.
isZero(106)
```

```
## [1] "False, the variability of pixel 106 is not zero."
```

```
# Test to see if pixel 107 has a zero variability.
isZero(107)
```

```
## [1] "False, the variability of pixel 107 is not zero."
```

```
# Test to see if pixel 108 has a zero variability.
isZero(108)
```

```
## [1] "False, the variability of pixel 108 is not zero."
```

```
# Test to see if pixel 150 has a zero variability.
isZero(150)
```

```
## [1] "False, the variability of pixel 150 is not zero."
```

```
# Test to see if pixel 151 has a zero variability.
isZero(151)
```

```
## [1] "False, the variability of pixel 151 is not zero."
```

```
# Test to see if pixel 152 has a zero variability.
isZero(152)
```

```
## [1] "False, the variability of pixel 152 is not zero."
```

It turns out that all the above features have non-zero variability.

```
# Now, create a neural net with four features (X100, X101, X102, X103)
# and 166 units in the hidden layer.
set.seed(1234)
model.d1 <- nnet(labels~X100+X101+X102+X103,
                 size = 166,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d1)

# get predictions
d.predict<-predict(model.d1, type = "raw")


# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5323
```

```
# Create a neural net with five features
# (X100, X101, X102, X103, X104)
# and 142 units in the hidden layer.
set.seed(1234)
model.d2 <- nnet(labels~X100+X101+X102+X103+X104,
                 size = 142,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d2)

# get predictions
d.predict<-predict(model.d2, type = "raw")

# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5338
```

As the number of units increase from 9 to 142, the AUC (0.5338) increases. Thus, this model is even better than the previous models.

```
# Create a neural net with six features
# (X100, X101, X102, X103, X104, X105)
# and 124 units in the hidden layer.
set.seed(1234)
model.d3 <- nnet(labels~X100+X101+X102+X103+X104+X105,
                 size = 124,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d3)

# get predictions
d.predict<-predict(model.d3, type = "raw")

# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5343
```

```
# Create a neural net with seven features
# (X100, X101, X102, X103, X104, X105, X106)
# and 111 units in the hidden layer.
set.seed(1234)
model.d4 <- nnet(labels~X100+X101+X102+X103+X104+X105+X106,
                 size = 111,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d4)

# get predictions
d.predict<-predict(model.d4, type = "raw")

# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5353
```

```
# Create a neural net with eight features
# (X100, X101, X102, X103, X104, X105, X106, X107)
```

```
# and 99 units in the hidden layer.
set.seed(1234)
model.d5 <- nnet(labels~X100+X101+X102+X103+X104+X105+X106+X107,
                 size = 99,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d5)

# get predictions
d.predict<-predict(model.d5, type = "raw")

# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5358
```

```
# Create a neural net with 9 features
# (X100, X101, X102, X103, X104, X105, X106, X107, X108)
# and 90 units in the hidden layer.
set.seed(1234)
model.d6 <- nnet(labels~X100+X101+X102+X103+X104+X105+X106+X107+X108,
                 size = 90,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d6)

# get predictions
d.predict<-predict(model.d6, type = "raw")

# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.5358
```

```
# Create a neural net with 12 features
# (X100, X101, X102, X103, X104, X105, X106, X107, X108, X150, X151, X152)
# and 71 units in the hidden layer.
set.seed(1234)
model.d7 <- nnet(labels~X100+X101+X102+X103+X104+X105+X106+X107+X108+X150+X151+X152,
                 size = 71,  data=images_df,
                 trace = F,maxit = 1000, linout = T)
#summary(model.d7)

# get predictions
d.predict<-predict(model.d7, type = "raw")
```

```
# compute roc
roc<-roc(images_df$labels, d.predict)
```

```
## Warning in roc.default(images_df$labels, d.predict): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a
## numeric vector.
```

```
# comput AUC
auc(roc)
```

```
## Area under the curve: 0.6795
```

Features with non-zero variability	Hidden Units	ROC
X100, X101, X102, X103	166	0.5323
X100, X101, X102, X103, X104	142	0.5338
X100, X101, X102, X103, X104, X105	124	0.5343
X100, X101, X102, X103, X104, X105, X106	111	0.5353
X100, X101, X102, X103, X104, X105, X106, X107	99	0.5358
X100, X101, X102, X103, X104, X105, X106, X107, X108	90	0.5358
X100, X101, X102, X103, X104, X105, X106, X107, X108, X150, X151, X152	71	0.6795

According to the outputs of ROC scores and the above table, as the number of features with non-zero variability increases, the highest number of hidden units that I can use for neural network model decreases and AUC increases. This implies that using more features with non-zero variability can improve the model by increasing its AUC.

Question 4

```
set.seed(20305)
mydf <- data.frame(matrix(rnorm(1100), ncol = 11, nrow = 100))
```

a)

```
# Fit z to the other 10 columns using multiple regression.
modela<-lm(X11~., data = mydf)
#summary(model)
sum(modela$residuals^2)
```

```
## [1] 98.80704
```

The sum of squares of the residuals is 98.80704.

b)

```
# Fit z to the other 10 columns, using a neural network with two
# hidden units and setting maxit = 2000 and decay = .01.
set.seed(20305)
modelb <- nnet(X11~.,size = 2,  data=mydf,
                   trace = F,maxit = 2000,decay = .01, linout = T)
sum(modelb$residuals^2)
```

```
## [1] 66.9531
```

This model fits better because it has a lower sum of residuals (66.9531) value than the previous model's (98.80704).

c)

```
# Fit z to the other 10 columns, using a neural network with 5
# hidden units and setting maxit = 2000 and decay = .01.
set.seed(20305)
modelc1 <- nnet(X11~.,size = 5,  data=mydf,
                  trace = F,maxit = 2000,decay = .01, linout = T)
sum(modelc1$residuals^2)
```

```
## [1] 12.8106
```

The sum of squares of the residuals is 12.8106.

```
# Fit z to the other 10 columns, using a neural network with 10
# hidden units and setting maxit = 2000 and decay = .01.
set.seed(20305)
modelc2 <- nnet(X11~.,size = 10,  data=mydf,
                  trace = F,maxit = 2000,decay = .01, linout = T)
sum(modelc2$residuals^2)
```

```
## [1] 0.07445059
```

The sum of squares of the residuals is 0.07445059.

Hidden Units	Sum of squares of the residuals
2	66.9531
5	12.8106
10	0.07445059

According to the output of sum of squared of the residuals and the above summarized table, as the number of hidden units increases, the sum of squares of the residuals decrease. This implies that the model fits better when the number of hidden units increase.