Individual Assignmenet 9

Zixiao Wu

2022-11-17

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Exercises 8.4

```
library(neuralnet)
```

```
## Warning: 程辑包'neuralnet'是用R版本4.2.2 来建造的
```

```
library(nnet)
library(ggplot2)
```

```
## Warning: 程辑包'ggplot2'是用R版本4.2.2 来建造的
```

```
library(caret)
```

```
## Warning: 程辑包'caret'是用R版本4.2.2 来建造的
```

```
## 载入需要的程辑包：lattice
```

```
library(gains)
```

```
df=read.csv("C:/Users/wuzix/Desktop/EastWestAirlinesNN.csv")
attach(df)
df = na.omit(df)
dim(df)
```

```
## [1] 4985   16
```

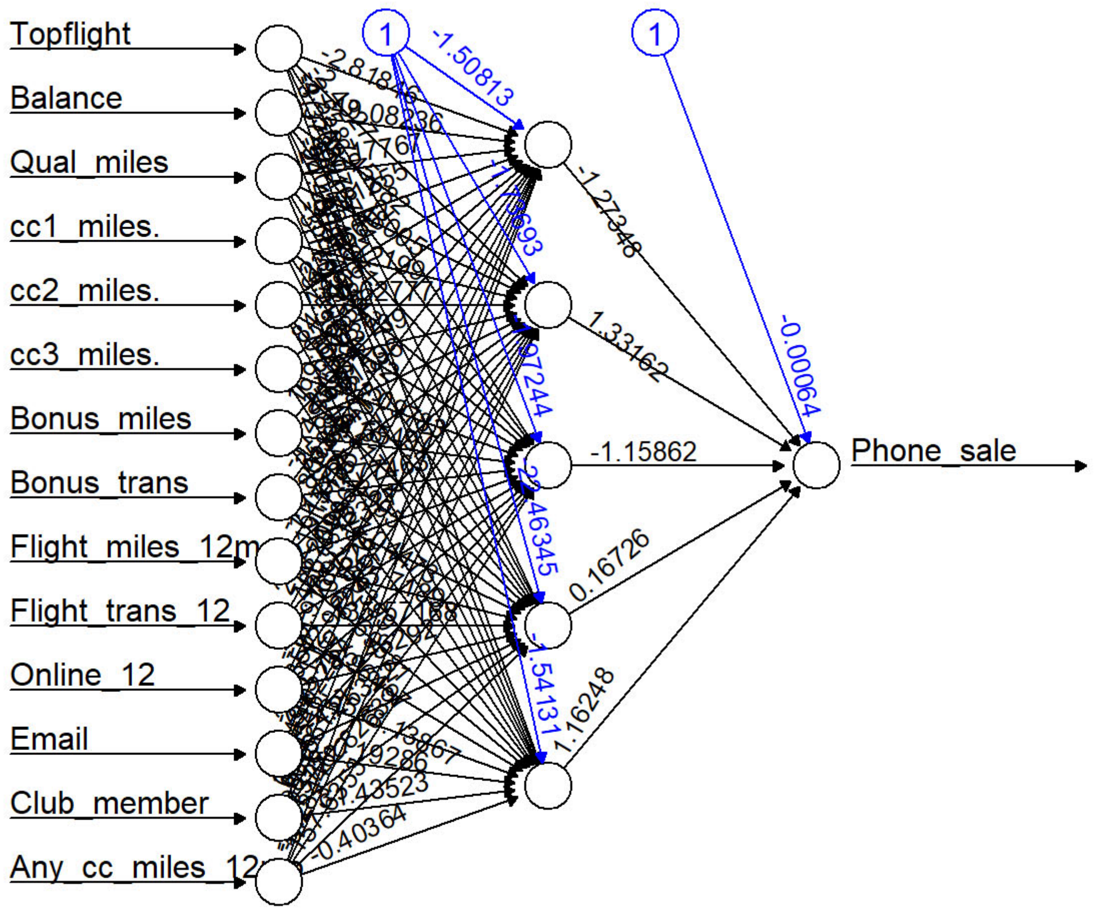
```
var = c("Topflight", "Balance", "Qual_miles", "cc1_miles.", "cc2_miles.", "cc3_miles.", "Bonus_
miles", "Bonus_trans", "Flight_miles_12mo", "Flight_trans_12", "Online_12", "Email", "Club_memb
er", "Any_cc_miles_12mo")
df = df[,c("Phone_sale",var)]
```

```
phone = df[,"Phone_sale"]
max_price=range(df['Phone_sale'])[2]
min_price=range(df['Phone_sale'])[1]

#Scale the numerical predictor and outcome variables to a 0-1 scale
numerical = c("Phone_sale", "Balance", "Qual_miles", "Bonus_miles", "Bonus_trans", "Flight_mile
s_12mo", "Flight_trans_12", "Online_12")
normvalues = preProcess(df[,numerical], method = 'range')
df[,numerical] = predict(normvalues, df[,numerical])
```

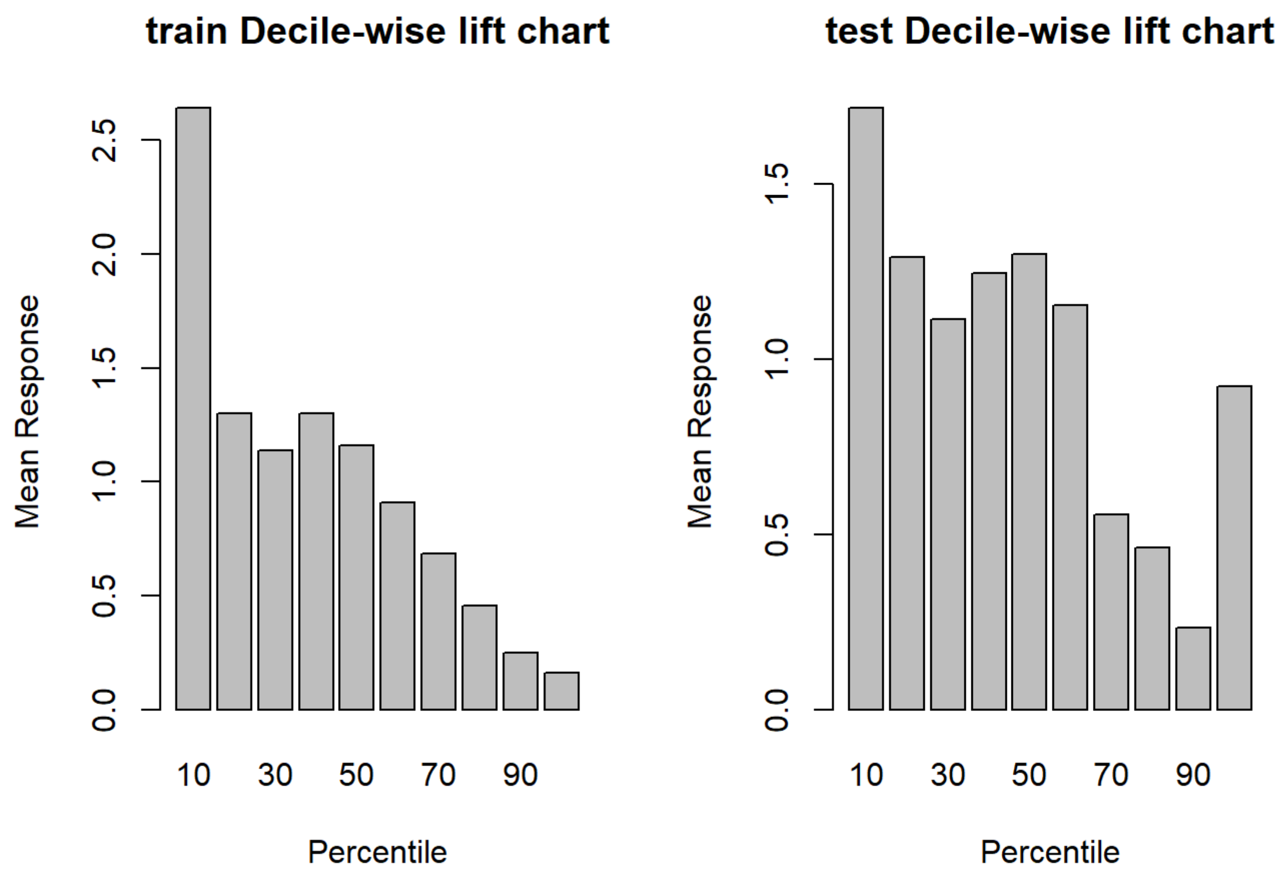
```
#bulid model
set.seed(114514)
train=sample(nrow(df),nrow(df)*0.65)
#neural network using 1 hidden layer 5 nodes
f=as.formula(paste('Phone_sale~',paste(names(df)[!names(df) %in% c('Phone_sale')],collapse='+'
)))
nn=neuralnet(f,data=df[train,],hidden=5)
```

```
plot(nn,rep = 'best')
```



```
#decile lift chart
pred.train = predict(nn,subset(df[train,],select = -c(Phone_sale)))
gain = gains(phone[train],pred.train)
pred.test = predict(nn,subset(df[-train,],select = -c(Phone_sale)))
gain1 = gains(phone[-train],pred.test)

par(mfcol=c(1,2))
barplot(gain$mean.resp / mean(phone[train]),names.arg = gain$depth, xlab = "Percentile",ylab =
"Mean Response", main = "train Decile-wise lift chart")
barplot(gain1$mean.resp / mean(phone[-train]),names.arg = gain1$depth, xlab = "Percentile",ylab
= "Mean Response", main = "test Decile-wise lift chart")
```



1,2. Reading the bar on the left of the decile-wise lift chart, we see that taking 10% of the data from training set that are ranked by the model as “the most probable 1’s” (having the highest propensities) yields, it has more than 2.5 times as many 1’s as would a random selection of 10% of the data. Also, the rate is nearly 1.5 in test set after modeling. So, we can see that the prediction capacity of the model is lower in the validation set than in the training set.

```
#build a rmse function
rmsef = function(nn,df,train,phone){
  pred.train = predict(nn,subset(df[train,],select = -c(Phone_sale)))
  pred.train.orig = pred.train*(max_price-min_price)+min_price
  train.rmse = sqrt(mean((phone[train]-pred.train.orig)^2))
  pred.test = predict(nn,subset(df[-train,],select = -c(Phone_sale)))
  pred.test.orig = pred.test*(max_price-min_price)+min_price
  test.rmse = sqrt(mean((phone[-train]-pred.test.orig)^2))
  #return rmse
  rmse = as.data.frame(rbind(train.rmse,test.rmse))
  return(rmse)
}
```

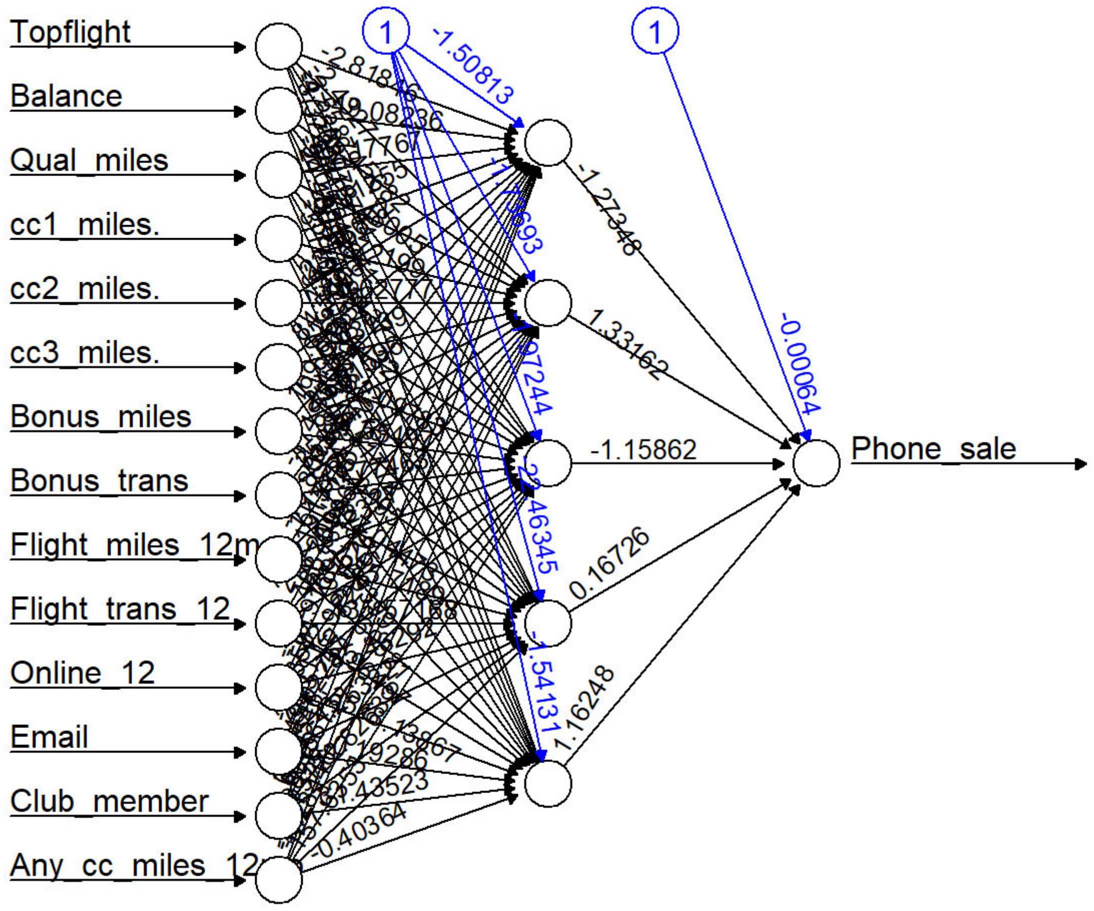
```
#compare two model
#another neuralnet model
nn1 =neuralnet(f,data=df[train,],hidden=1)
rmse = rmsef(nn,df,train,phone)
rmse1 = rmsef(nn1,df,train,phone)

rmse_df = cbind(rmse,rmse1)
names(rmse_df) = c('1 layer 5 nodes','1 layer 1 nodes')
rmse_df
```

```
##            1 layer 5 nodes 1 layer 1 nodes
## train.rmse       0.3232981       0.3361126
## test.rmse        0.3363286       0.3257248
```

3. From the rmsee of two models, we can find that though the model with 5 nodes has lower train rmse, its test rmse is higher than the model with 1 nodes. So we can assume that former model may have overfitting problem.

```
plot(nn,rep = 'best')
```



```
#check the Generalize Weight
head(nn$generalized.weights[[1]])
```

```
##               [,1]          [,2]          [,3]          [,4]          [,5]
## 4626  1.331304e+00  1.228876e+01  4.259879e+01  -16.396131088 -2.497283e+01
## 5026  4.205513e+02  8.619920e+03  3.895819e+03  714.621998082 -1.914858e+03
## 3906 -2.474942e-04 -4.853742e-03 -2.357612e-03   -0.000456202  1.099375e-03
## 1222 -2.175926e-03 -4.466302e-02 -2.024084e-02   -0.003632868  9.948901e-03
## 7778 -4.960892e+00 -1.658553e+02  4.453541e+00   43.311689110  1.162514e+02
## 3106 -2.479868e-03 -4.942908e-02 -2.342850e-02   -0.004420066  1.112887e-02
##               [,6]          [,7]          [,8]          [,9]          [,10]
## 4626  1.680483e+03  8.886481e+01 -1.462342e+02 -3.296283e+02    93.74129656
## 5026 -2.090728e+04 -4.490422e+03 -6.510498e+03  1.825697e+02  2953.01797883
## 3906  4.902526e-03  2.562636e-03  3.743161e-03  6.764973e-05    -0.00169301
## 1222  1.069187e-01  2.280511e-02  3.435300e-02 -1.782117e-04    -0.01556983
## 7778  4.834476e+02  3.364334e+02 -3.223850e+02  5.702766e+02   -99.63980949
## 3106  7.462248e-02  2.578719e-02  3.808301e-02  3.689799e-04    -0.01723758
##              [,11]         [,12]         [,13]         [,14]
## 4626 -3.793226e+03 -9.562201e-01  2.253175e+01  4.817957e+00
## 5026  4.042447e+03 -4.592227e+02 -1.169489e+03  1.541925e+03
## 3906 -1.563938e-03  2.610832e-04  6.484730e-04 -8.204713e-04
## 1222 -1.129541e-02  2.383064e-03  5.985056e-03 -7.993885e-03
## 7778  6.558055e+03 -7.487124e-01  1.765734e+01 -1.100681e+02
## 3106 -1.468976e-02  2.651057e-03  6.611034e-03 -8.532890e-03
```

4. From the Generalize Weight result, we can see that the 10th variable, Flight_trans_12, has a weight significantly greater than the other variables, shows that it may quite important for model prediction.