Individual_assignment10

Yuhan_Xu_474154

2019/11/15

Prefix

You are asked to develop a model to classify East-West customers as to whether they purchase a wireless phone service contract (outcome variable Phone_Sale). This model will be used to classify additional customers.

```
df = read.csv("./EastWestAirlinesNN.csv")
str(df)
```

```
## 'data.frame':    9974 obs. of  16 variables:
##  $ 锘緽D.           : int  NA 1 NA 2 NA 3 NA 4 NA 5 ...
##  $ Topflight        : int  NA 0 NA 0 NA 0 NA 0 NA 1 ...
##  $ Balance          : int  NA 28143 NA 19244 NA 41354 NA 14776 NA
97752 ...
##  $ Qual_miles       : int  NA 0 NA 0 NA 0 NA 0 NA 0 ...
##  $ cc1_miles.       : int  NA 0 NA 0 NA 1 NA 0 NA 1 ...
##  $ cc2_miles.       : int  NA 1 NA 0 NA 0 NA 0 NA 0 ...
##  $ cc3_miles.       : int  NA 0 NA 0 NA 0 NA 0 NA 0 ...
##  $ Bonus_miles      : int  NA 174 NA 215 NA 4123 NA 500 NA 43300 ...
##  $ Bonus_trans      : int  NA 1 NA 2 NA 4 NA 1 NA 26 ...
##  $ Flight_miles_12mo: int  NA 0 NA 0 NA 0 NA 0 NA 2077 ...
##  $ Flight_trans_12  : int  NA 0 NA 0 NA 0 NA 0 NA 4 ...
##  $ Online_12        : int  NA 0 NA 0 NA 0 NA 0 NA 0 ...
##  $ Email            : int  NA 1 NA 0 NA 1 NA 1 NA 1 ...
##  $ Club_member      : int  NA 0 NA 0 NA 0 NA 0 NA 0 ...
##  $ Any_cc_miles_12mo: int  NA 1 NA 0 NA 1 NA 0 NA 1 ...
##  $ Phone_sale       : int  NA 0 NA 0 NA 0 NA 0 NA 0 ...
```

Obviously, the first column “ID#” should not be used in model building.

```
df = df[,-1]
```

a

Q: Run a neural net model on these data, using a single hidden layer with 5 nodes. Remember to first convert categorical variables into dummies and scale numerical predictor variables to a 0-1 (use function preprocess() with method=“range” - see Chapter 7).

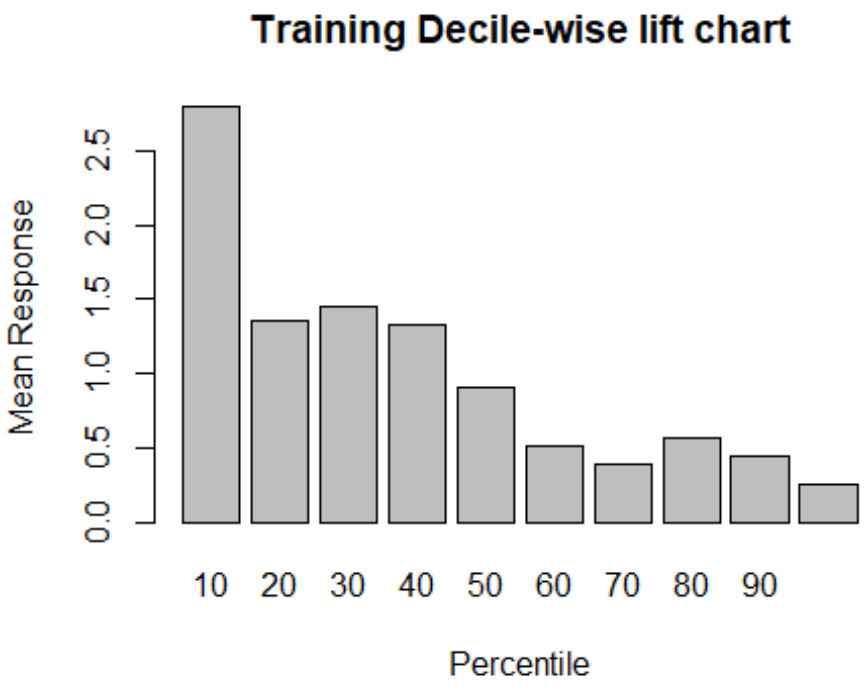
```
library(caret)
```

```
## Loading required package: lattice
```

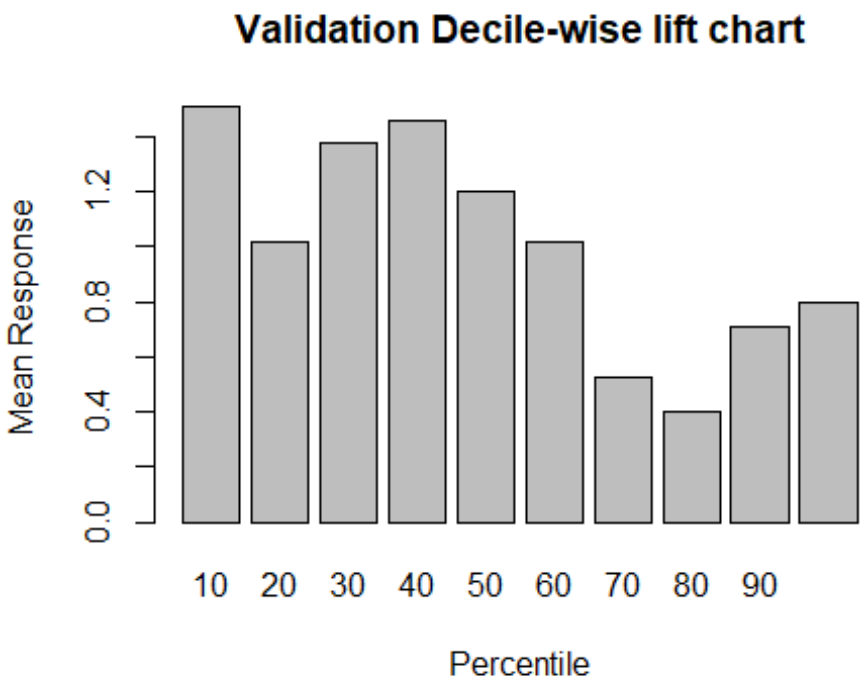
```
## Loading required package: ggplot2

standard = preProcess(df, method = "range")
df = predict(standard, df)
df = df[!is.na(df$Phone_sale),]
attach(df)

set.seed(1)
train = sample(nrow(df), nrow(df)*0.65)
df.train = df[train,]
df.test = df[-train,]
Phone_sale.train = df.train$Phone_sale
Phone_sale.test = df.test$Phone_sale

set.seed(1)
library(neuralnet)
nn1 = neuralnet(Phone_sale~., data = df.train, hidden = 5,
                linear.output = FALSE)
```

Q: Generate a deciles-wise lift chart for the training and validation sets.

```
library(gains)

prob.train1 = compute(nn1, subset(df.train, select =
                                    -c(Phone_sale)))$net.result
gain1 = gains(Phone_sale.train, prob.train1,)
barplot(gain1$mean.resp/mean(Phone_sale.train), names.arg =
gain1$depth,
        xlab = "Percentile", ylab = "Mean Response",
        main = "Training Decile-wise lift chart")
```




```
prob.test1 = compute(nn1, subset(df.test, select =
                                    -c(Phone_sale)))$net.result
gain1 = gains(Phone_sale.test, prob.test1,)
barplot(gain1$mean.resp/ mean(Phone_sale.test), names.arg =
gain1$depth, xlab = "Percentile", ylab = "Mean Response", main =
"Validation Decile-wise lift chart")
```



Q: Interpret the meaning (in business terms) of the leftmost bar of the validation decile- wise lift chart.
A: The leftmost bar means that when this model is used to predict test data, taking the 10% of the records that are ranked by the model as "the most probable 1's", that is having the highest probabilities, yields 1.5 times as many 1's as would a random selection fo 10% of the records.

b

Q: Comment on the difference between the training and validation lift charts.
A: The leftmost bar of training lift chart has a larger mean response than that of validation lift chart. When applying to training data set, the model gives a higher lift comparing to the results of test data set. Therefore, this model performs better on training data set than test data set.

c

Q: Run a second neural net model on the data, this time setting the number of hidden nodes to 1. Comment now on the difference between this model and the model you ran earlier, and how overfitting might have affected results.

```
set.seed(1)
nn2 = neuralnet(Phone_sale~., data = df.train, hidden = 1,
                linear.output = FALSE)
```

```
pred.train1 = ifelse(prob.train1 >= 0.5, 1, 0)
mean(pred.train1 != Phone_sale.train)

## [1] 0.1240741

prob.train2 = compute(nn2, subset(df.train, select =
                                    -c(Phone_sale)))$net.result
pred.train2 = ifelse(prob.train2 >= 0.5, 1, 0)
mean(pred.train2 != Phone_sale.train)

## [1] 0.1324074

pred.test1 = ifelse(prob.test1 >= 0.5, 1, 0)
mean(pred.test1 != Phone_sale.test)

## [1] 0.1386819

prob.test2 = compute(nn2, subset(df.test, select =
                                    -c(Phone_sale)))$net.result
pred.test2 = ifelse(prob.test2 >= 0.5, 1, 0)
mean(pred.test2 != Phone_sale.test)

## [1] 0.1295129
```

A: The training error rate of the model with 5 hidden nodes is 0.124. The training error rate of the model with 1 node is 0.132.

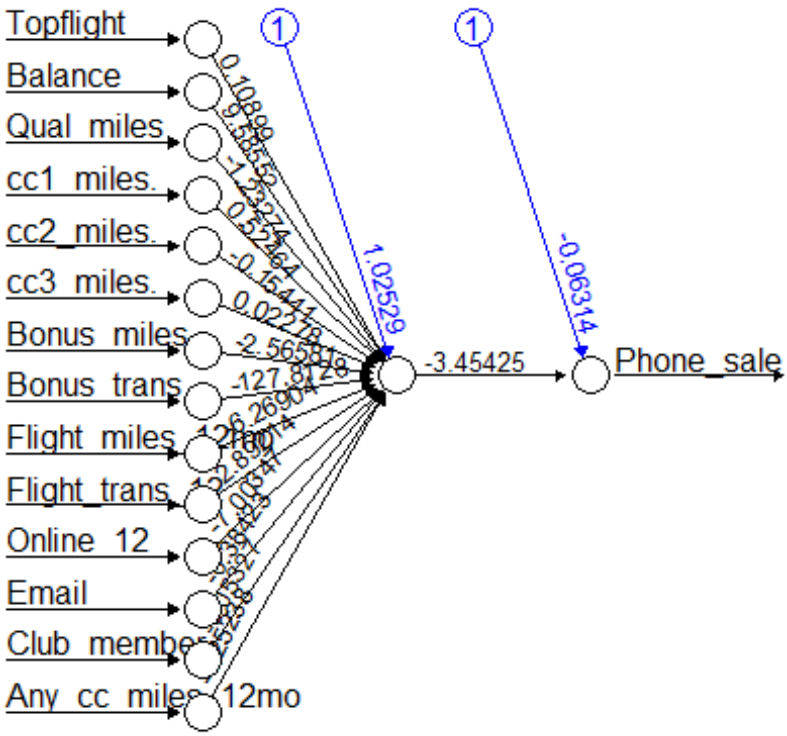
The test error rate of the model with 5 hidden nodes is 0.139. The test error rate of the model with 1 node is 0.130.

From the results, we know that a more complex model, that is the model with more hidden nodes, has a lower training error rate and a higher test error rate. Even though this model fits the training data well, when applying to the test data, it performs badly.
In a conclusion, a too complex model may overfit the training data set and have a bad performance when it is used to predict the results of test data.

d

Q: What sort of information, if any, is provided about the effects of the various variables?

```
plot(nn2, rep = "best")
```



A: The weights of each variables can indicate the effects of them on the results of neural network model.