Ch11RandallPlyler

Randall Plyler

2/26/2022

```
#### Table 11.2
accidents.df <- read.csv("C:/Users/randa/Dropbox/Masters/Winter/TBANLT 560 Data Mining/Files/AccidentsFu

#install.packages('neuralnet')
library(neuralnet)
df <- read.csv("C:/Users/randa/Dropbox/Masters/Winter/TBANLT 560 Data Mining/Files/DMBA-R-datasets/DMBA-

df$Like <- df$Acceptance=="like"
df$Dislike <- df$Acceptance=="dislike"

set.seed(1)
nn <- neuralnet(Like + Dislike ~ Salt + Fat, data = df, linear.output = F, hidden = 3)

# display weights
nn$weights
```

```
## [[1]]
## [[1]][[1]]
##           [,1]      [,2]      [,3]
## [1,] -1.061144  3.057022  3.337952
## [2,]  2.326024 -3.408663 -4.293214
## [3,]  4.106435 -6.525668 -5.929419
##
## [[1]][[2]]
##            [,1]      [,2]
## [1,] -0.3495333 -1.677856
## [2,]  5.8777146 -3.606625
## [3,] -5.3529201  5.620330
## [4,] -6.1115039  6.696287
```

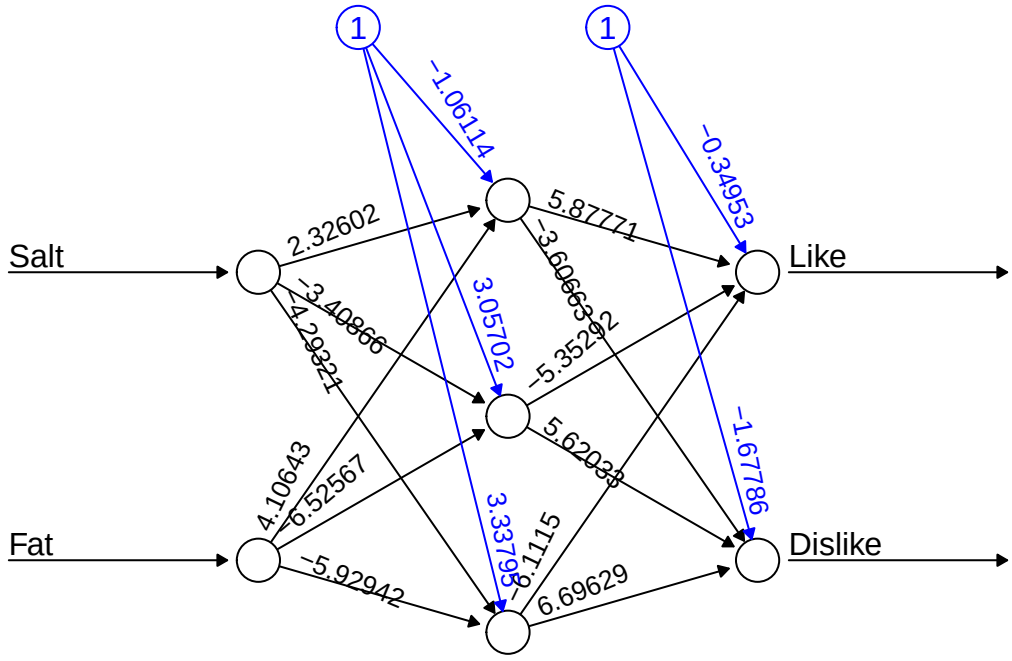
```
# display predictions
prediction(nn)
```

```
## Data Error:  0;

## $rep1
##   Salt Fat         Like    Dislike
## 1  0.1 0.1 0.0002415536 0.99965512
## 2  0.4 0.2 0.0344215787 0.96556788
## 3  0.5 0.2 0.1248666748 0.87816828
```

```
## 4  0.9 0.2 0.9349452648 0.07022732
## 5  0.8 0.3 0.9591361793 0.04505631
## 6  0.5 0.4 0.8841904620 0.12672438
##
## $data
##   Salt Fat Like Dislike
## 1  0.1 0.1    0       1
## 2  0.4 0.2    0       1
## 3  0.5 0.2    0       1
## 4  0.9 0.2    1       0
## 5  0.8 0.3    1       0
## 6  0.5 0.4    1       0
```

```
# plot network
plot(nn, rep="best")
```




```
#### Table 11.3

library(caret)
```

```
## Loading required package: ggplot2
```

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

```
predict <- compute(nn, data.frame(df$Salt, df$Fat))
predicted.class=apply(predict$net.result,1,which.max)-1
confusionMatrix(as.factor(ifelse(predicted.class=="1", "dislike", "like")), as.factor(df$Acceptance))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction dislike like
##    dislike       3    0
##    like          0    3
##
##                Accuracy : 1
##                  95% CI : (0.5407, 1)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 0.01563
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0
##             Specificity : 1.0
##          Pos Pred Value : 1.0
##          Neg Pred Value : 1.0
##              Prevalence : 0.5
##          Detection Rate : 0.5
##    Detection Prevalence : 0.5
##       Balanced Accuracy : 1.0
##
##        'Positive' Class : dislike
##
```

```
#### Table 11.6, 11.7
library(neuralnet)
library(nnet)
library(caret)
library(e1071)
```

```
accidents.df <- read.csv("accidentsnn.csv")

head(accidents.df)
```

```
##   ALCHL_I PROFIL_I_R SUR_COND VEH_INVL MAX_SEV_IR
## 1       2          0        1        1          0
## 2       2          1        1        1          2
## 3       1          0        1        1          0
## 4       2          0        2        2          1
## 5       2          1        1        2          1
## 6       2          0        1        1          0
```

```r
# selected variables
vars <- c("ALCHL_I", "PROFIL_I_R", "VEH_INVL")
# partition the data
set.seed(3)
training=sample(row.names(accidents.df), dim(accidents.df)[1]*0.6)
validation=setdiff(row.names(accidents.df), training)

# when y has multiple classes - need to dummify
trainData <- cbind(accidents.df[training,c(vars)],
                   class.ind(accidents.df[training,]$SUR_COND),
                   class.ind(accidents.df[training,]$MAX_SEV_IR))
str(trainData)
```

```
## 'data.frame':    599 obs. of  11 variables:
##  $ ALCHL_I   : int  2 1 2 2 1 2 2 2 1 2 ...
##  $ PROFIL_I_R: int  0 0 0 0 0 0 0 0 1 1 ...
##  $ VEH_INVL  : int  2 1 3 1 3 1 2 1 1 2 ...
##  $ 1         : num  0 1 0 1 1 1 0 1 1 1 ...
##  $ 2         : num  1 0 1 0 0 0 1 0 0 0 ...
##  $ 3         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ 4         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ 9         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ 0         : num  0 1 0 0 0 1 0 0 0 0 ...
##  $ 1         : num  1 0 1 0 0 0 1 0 0 1 ...
##  $ 2         : num  0 0 0 1 1 0 0 1 1 0 ...
```

```r
names(trainData) <- c(vars,
                   paste("SUR_COND_", c(1, 2, 3, 4, 9), sep=""), paste("MAX_SEV_IR_", c(0, 1, 2), sep="
str(trainData)
```

```
## 'data.frame':    599 obs. of  11 variables:
##  $ ALCHL_I     : int  2 1 2 2 1 2 2 2 1 2 ...
##  $ PROFIL_I_R  : int  0 0 0 0 0 0 0 0 1 1 ...
##  $ VEH_INVL    : int  2 1 3 1 3 1 2 1 1 2 ...
##  $ SUR_COND_1  : num  0 1 0 1 1 1 0 1 1 1 ...
##  $ SUR_COND_2  : num  1 0 1 0 0 0 1 0 0 0 ...
##  $ SUR_COND_3  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SUR_COND_4  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SUR_COND_9  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ MAX_SEV_IR_0: num  0 1 0 0 0 1 0 0 0 0 ...
##  $ MAX_SEV_IR_1: num  1 0 1 0 0 0 1 0 0 1 ...
##  $ MAX_SEV_IR_2: num  0 0 0 1 1 0 0 1 1 0 ...
```

```r
validData <- cbind(accidents.df[validation,c(vars)],
                   class.ind(accidents.df[validation,]$SUR_COND),
                   class.ind(accidents.df[validation,]$MAX_SEV_IR))
str(validData)
```

```
## 'data.frame':    400 obs. of  11 variables:
##  $ ALCHL_I   : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ PROFIL_I_R: int  1 0 1 1 0 0 0 0 0 0 ...
##  $ VEH_INVL  : int  1 1 1 2 2 1 1 1 2 1 ...
```

```
##  $ 1         : num  1 0 0 1 1 1 1 1 0 1 ...
##  $ 2         : num  0 1 0 0 0 0 0 0 1 0 ...
##  $ 3         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ 4         : num  0 0 1 0 0 0 0 0 0 0 ...
##  $ 9         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ 0         : num  0 0 0 0 1 1 1 1 0 1 ...
##  $ 1         : num  0 0 1 1 0 0 0 0 1 0 ...
##  $ 2         : num  1 1 0 0 0 0 0 0 0 0 ...
```

```
names(validData) <- c(vars,
                      paste("SUR_COND_", c(1, 2, 3, 4, 9), sep=""), paste("MAX_SEV_IR_", c(0, 1, 2), sep="
```

```
# run nn with 2 hidden nodes
# use hidden= with a vector of integers specifying number of hidden nodes in each layer
nn <- neuralnet(MAX_SEV_IR_0 + MAX_SEV_IR_1 + MAX_SEV_IR_2 ~
                  ALCHL_I + PROFIL_I_R + VEH_INVL + SUR_COND_1 + SUR_COND_2
                + SUR_COND_3 + SUR_COND_4, data = trainData, hidden = 2)

training.prediction <- compute(nn, trainData[,-c(8:11)])
training.class <- apply(training.prediction$net.result,1,which.max)-1
confusionMatrix(as.factor(training.class), as.factor(accidents.df[training,]$MAX_SEV_IR))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2
##          0 339   0  32
##          1   0 175  45
##          2   0   0   8
##
## Overall Statistics
##
##                Accuracy : 0.8715
##                  95% CI : (0.842, 0.8972)
##     No Information Rate : 0.5659
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7621
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity            1.0000   1.0000  0.09412
## Specificity            0.8769   0.8939  1.00000
## Pos Pred Value         0.9137   0.7955  1.00000
## Neg Pred Value         1.0000   1.0000  0.86971
## Prevalence             0.5659   0.2922  0.14190
## Detection Rate         0.5659   0.2922  0.01336
## Detection Prevalence   0.6194   0.3673  0.01336
## Balanced Accuracy      0.9385   0.9469  0.54706
```

```
validation.prediction <- compute(nn, validData[,-c(8:11)])
validation.class <-apply(validation.prediction$net.result,1,which.max)-1
confusionMatrix(as.factor(validation.class), as.factor(accidents.df[validation,]$MAX_SEV_IR))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2
##          0 210   0  22
##          1   0 124  37
##          2   2   0   5
##
## Overall Statistics
##
##                Accuracy : 0.8475
##                  95% CI : (0.8085, 0.8813)
##     No Information Rate : 0.53
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7301
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity            0.9906   1.0000  0.07812
## Specificity            0.8830   0.8659  0.99405
## Pos Pred Value         0.9052   0.7702  0.71429
## Neg Pred Value         0.9881   1.0000  0.84987
## Prevalence             0.5300   0.3100  0.16000
## Detection Rate         0.5250   0.3100  0.01250
## Detection Prevalence   0.5800   0.4025  0.01750
## Balanced Accuracy      0.9368   0.9330  0.53609
```