Neural-Network-for-Classification.R

patriciamaya

2020-11-03

```
#Neural Network for Classification

#We'll use ISLR's built in College Data Set which has several features of a college and a
#categorical column indicating whether or not the School is Public or Private.

library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.0.2
```

```
data(College)
str(College) #all variables are numerical besides Private
```

```
## 'data.frame':    777 obs. of  18 variables:
##  $ Private    : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Apps       : num  1660 2186 1428 417 193 ...
##  $ Accept     : num  1232 1924 1097 349 146 ...
##  $ Enroll     : num  721 512 336 137 55 158 103 489 227 172 ...
##  $ Top10perc  : num  23 16 22 60 16 38 17 37 30 21 ...
##  $ Top25perc  : num  52 29 50 89 44 62 45 68 63 44 ...
##  $ F.Undergrad: num  2885 2683 1036 510 249 ...
##  $ P.Undergrad: num  537 1227 99 63 869 ...
##  $ Outstate   : num  7440 12280 11250 12960 7560 ...
##  $ Room.Board : num  3300 6450 3750 5450 4120 ...
##  $ Books      : num  450 750 400 450 800 500 500 450 300 660 ...
##  $ Personal   : num  2200 1500 1165 875 1500 ...
##  $ PhD        : num  70 29 53 92 76 67 90 89 79 40 ...
##  $ Terminal   : num  78 30 66 97 72 73 93 100 84 41 ...
##  $ S.F.Ratio  : num  18.1 12.2 12.9 7.7 11.9 9.4 11.5 13.7 11.3 11.5 ...
##  $ perc.alumni: num  12 16 30 37 2 11 26 37 23 15 ...
##  $ Expend     : num  7041 10527 8735 19016 10922 ...
##  $ Grad.Rate  : num  60 56 54 59 15 55 63 73 80 52 ...
```

```
#DATA PREPROCESSING
# Create vector of column Max and Min values
maxs = apply(College[ , 2:18], 2, max)
# apply(x, margin, function).
#If margin = 1, function is applied on the rows. If margin = 2, function is applied on the columns
mins = apply(College[ , 2:18], 2, min)
#normalizing data
scaled.data = as.data.frame(scale(College[ , 2:18], center = mins, scale = maxs - mins))

#SPLIT DATA
set.seed(1234)
ind = sample(2, nrow(College), replace = T, prob = c(0.7, 0.3))
```

```
TrainData = College[ind == 1, ]
TestData = College[ind == 2, ]

library(nnet)
nn = nnet(Private ~ ., data=TrainData, linout=F, size=10, decay=0.01, maxit=1000)
```

```
## # weights:  191
## initial  value 485.887287
## iter  10 value 247.762510
## iter  20 value 174.812331
## iter  30 value 119.555050
## iter  40 value 117.766013
## iter  50 value 109.355874
## iter  60 value 105.428478
## iter  70 value 104.587145
## iter  80 value 100.003301
## iter  90 value 99.254662
## iter 100 value 99.069631
## iter 110 value 97.800280
## iter 120 value 96.287717
## iter 130 value 95.060649
## iter 140 value 94.493642
## iter 150 value 92.765077
## iter 160 value 92.464401
## iter 170 value 91.896218
## iter 180 value 90.482196
## iter 190 value 90.260756
## iter 200 value 90.130014
## iter 210 value 89.849747
## iter 220 value 89.589006
## iter 230 value 89.284073
## iter 240 value 89.013106
## iter 250 value 88.720591
## iter 260 value 88.388993
## iter 270 value 88.370376
## iter 280 value 88.299777
## iter 290 value 88.273862
## iter 300 value 88.259755
## iter 310 value 88.237144
## iter 320 value 88.217238
## iter 330 value 88.078857
## iter 340 value 88.019195
## iter 350 value 87.837137
## iter 360 value 87.776408
## iter 370 value 87.690785
## iter 380 value 87.487479
## iter 390 value 87.392482
## iter 400 value 87.303044
## iter 410 value 87.218736
## iter 420 value 87.028616
## iter 430 value 86.845645
## iter 440 value 86.665065
## iter 450 value 86.619367
## iter 460 value 86.605685
## iter 470 value 86.550946
## iter 480 value 86.431257
## iter 490 value 86.377424
## iter 500 value 86.235417
## iter 510 value 85.775320
## iter 520 value 85.717657
## iter 530 value 85.602845
## iter 540 value 85.395966
## iter 550 value 85.308607
## iter 560 value 84.589309
## iter 570 value 83.462937
## iter 580 value 82.013836
## iter 590 value 81.392742
## iter 600 value 81.147700
## iter 610 value 79.974230
## iter 620 value 78.664159
## iter 630 value 78.466769
## iter 640 value 77.553235
## iter 650 value 77.468005
## iter 660 value 76.498758
## iter 670 value 72.680294
## iter 680 value 70.825021
## iter 690 value 70.274280
## iter 700 value 69.980880
## iter 710 value 69.105281
## iter 720 value 68.734577
## iter 730 value 68.528781
## iter 740 value 67.405952
## iter 750 value 67.194740
## iter 760 value 66.906339
## iter 770 value 66.703103
## iter 780 value 66.630486
## iter 790 value 66.426696
## iter 800 value 66.129554
## iter 810 value 65.887121
## iter 820 value 65.453291
## iter 830 value 65.208523
## iter 840 value 65.123731
## iter 850 value 64.821433
## iter 860 value 64.616198
## iter 870 value 64.520869
## iter 880 value 64.039870
## iter 890 value 63.733603
## iter 900 value 63.382449
## iter 910 value 63.199916
## iter 920 value 62.951946
## iter 930 value 62.858501
## iter 940 value 62.766967
## iter 950 value 61.931618
## iter 960 value 61.465205
## iter 970 value 61.223511
## iter 980 value 60.904719
## iter 990 value 60.478556
## iter1000 value 60.317879
## final  value 60.317879
## stopped after 1000 iterations
```

```
#The weights will be learned with a weight updating rate of 0.01 (the parameter decay).
#The parameter linout (linear out-put) indicates that the target variable is continuous or not.
#The maxit parameter sets the maximum number of iterations of the weight convergence algorithm.
library(devtools)
```

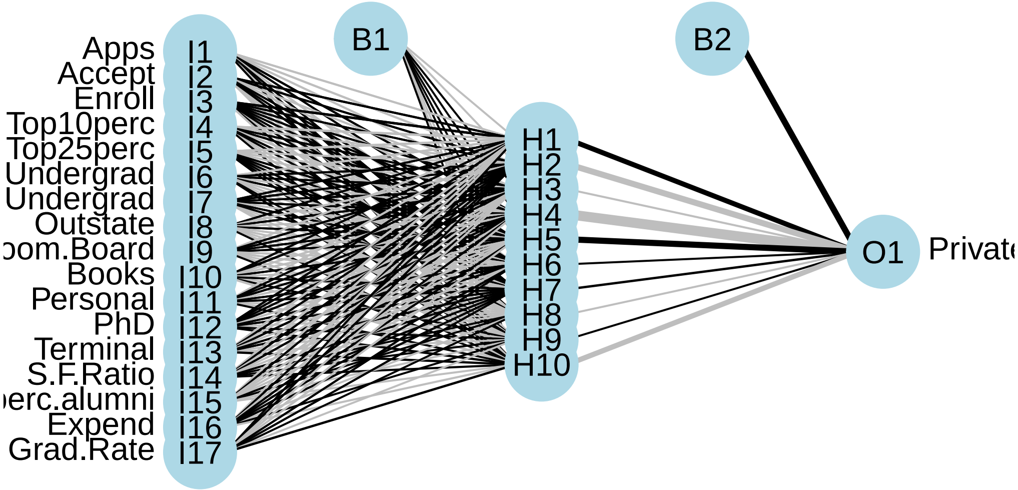
```
## Warning: package 'devtools' was built under R version 4.0.2

## Loading required package: usethis

## Warning: package 'usethis' was built under R version 4.0.2
```

```
source_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4
```

```
## SHA-1 hash of file is 74c80bd5ddbc17ab3ae5ece9c0ed9beb612e87ef
```

```
plot.nnet(nn)
```

```
## Loading required package: scales

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'
## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'
## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'

## Loading required package: reshape

## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'reshape'
```



```
summary(nn)
```

```
## a 17-10-1 network with 191 weights
## options were - entropy fitting  decay=0.01
##   b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1  i9->h1
##   -0.01   -0.03    0.14    0.41   -0.32   -0.31   -0.17    0.03    0.04   -0.02
## i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1
##    0.05   -0.04    0.15   -0.16   -0.34   -0.08    0.00    0.01
##   b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2  i9->h2
##    0.00   -0.04    0.09    0.00   -0.04   -0.05    0.03    0.08   -0.10    0.09
## i10->h2 i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2
##    0.23    0.01    0.03    0.01    0.01    0.01    0.00    0.01
##   b->h3  i1->h3  i2->h3  i3->h3  i4->h3  i5->h3  i6->h3  i7->h3  i8->h3  i9->h3
##    0.00   -0.01   -0.01    0.00    0.00    0.00    0.01    0.00    0.00    0.02
## i10->h3 i11->h3 i12->h3 i13->h3 i14->h3 i15->h3 i16->h3 i17->h3
##    0.00    0.00    0.00    0.00    0.00    0.00   -0.01    0.00
##   b->h4  i1->h4  i2->h4  i3->h4  i4->h4  i5->h4  i6->h4  i7->h4  i8->h4  i9->h4
##    0.00    0.10    0.00    0.03    0.02    0.00   -0.04    0.01   -0.03    0.00
## i10->h4 i11->h4 i12->h4 i13->h4 i14->h4 i15->h4 i16->h4 i17->h4 
##   -0.04    0.03    0.12    0.07    0.02   -0.01    0.03   -0.01 
##   b->h5  i1->h5  i2->h5  i3->h5  i4->h5  i5->h5  i6->h5  i7->h5  i8->h5  i9->h5 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.01    0.00 
## i10->h5 i11->h5 i12->h5 i13->h5 i14->h5 i15->h5 i16->h5 i17->h5 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.01    0.00 
##   b->h6  i1->h6  i2->h6  i3->h6  i4->h6  i5->h6  i6->h6  i7->h6  i8->h6  i9->h6 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
## i10->h6 i11->h6 i12->h6 i13->h6 i14->h6 i15->h6 i16->h6 i17->h6 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
##   b->h7  i1->h7  i2->h7  i3->h7  i4->h7  i5->h7  i6->h7  i7->h7  i8->h7  i9->h7 
##    0.00    0.04    0.02    0.02    0.00    0.00   -0.02    0.02   -0.03   -0.03 
## i10->h7 i11->h7 i12->h7 i13->h7 i14->h7 i15->h7 i16->h7 i17->h7 
##    0.02    0.01    0.00    0.00    0.00    0.00    0.01    0.01 
##   b->h8  i1->h8  i2->h8  i3->h8  i4->h8  i5->h8  i6->h8  i7->h8  i8->h8  i9->h8 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
## i10->h8 i11->h8 i12->h8 i13->h8 i14->h8 i15->h8 i16->h8 i17->h8 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
##   b->h9  i1->h9  i2->h9  i3->h9  i4->h9  i5->h9  i6->h9  i7->h9  i8->h9  i9->h9 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
## i10->h9 i11->h9 i12->h9 i13->h9 i14->h9 i15->h9 i16->h9 i17->h9 
##    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 
##   b->h10  i1->h10  i2->h10  i3->h10  i4->h10  i5->h10  i6->h10  i7->h10 
##     0.01     0.08    -0.22    -0.16     0.00     0.02     0.19    -0.10 
##  i8->h10  i9->h10 i10->h10 i11->h10 i12->h10 i13->h10 i14->h10 i15->h10 
##     0.04     0.02    -0.41     0.05     0.28     0.23     0.10    -0.03 
## i16->h10 i17->h10 
##    -0.09     0.36 
##   b->o  h1->o  h2->o  h3->o  h4->o  h5->o  h6->o  h7->o  h8->o  h9->o h10->o 
##   4.18   3.05  -4.09  -0.03  -8.11   4.18   0.00   0.35   0.00   0.00  -3.24
```

```
# You could also use wts to get the best weights found and fitted.values to get the fitted
#values on training data
nn$wts
```

```
##   [1] -8.346814e-03 -3.180840e-02  1.383018e-01  4.092997e-01 -3.184893e-01
##   [6] -3.070548e-01 -1.693836e-01  3.159437e-02  3.780833e-02 -1.987079e-02
##  [11]  4.510113e-02 -3.709286e-02  1.461482e-01 -1.564430e-01 -3.425698e-01
##  [16] -7.996408e-02 -1.764706e-03  5.723435e-03 -3.809041e-04 -4.158933e-02
##  [21]  8.783067e-02  8.942135e-04 -3.537822e-02 -4.959560e-02  3.248600e-02
##  [26]  7.563627e-02 -9.572936e-02  9.037993e-02  2.284504e-01  1.140116e-02
##  [31]  2.915567e-02  1.233496e-02  1.016949e-02  6.662271e-03  4.209715e-04
##  [36]  1.021499e-02  7.504003e-04 -9.645047e-03 -1.110022e-02  1.439678e-04
##  [41]  1.267905e-03 -5.782217e-06  8.102928e-03  4.318705e-03  1.361195e-03
##  [46]  1.744511e-02 -1.383612e-03  2.214859e-03  1.813942e-03 -1.276769e-03
##  [51] -2.855351e-04 -3.203392e-04 -9.915250e-03 -5.918951e-04  2.125271e-03
##  [56]  1.021530e-01 -4.062111e-04  3.147085e-02  2.264027e-02 -2.814349e-03
##  [61] -4.003394e-02  1.496051e-02 -3.245579e-02  4.834716e-03 -4.022348e-02
##  [66]  2.774757e-02  1.215735e-01  6.872600e-02  2.032940e-02 -1.382640e-02
##  [71]  2.704714e-02 -1.350968e-02  9.291463e-04  8.177963e-04  2.559609e-03
##  [76]  2.047389e-03 -1.374688e-03  2.312299e-04  1.307436e-03 -1.461727e-03
##  [81]  5.645802e-03  4.444709e-03  2.380481e-03 -1.189458e-03 -3.715773e-04
##  [86] -1.970768e-03 -3.566358e-04 -1.228527e-03  7.355922e-03 -1.113603e-03
##  [91]  2.064103e-03 -9.792936e-04  1.359719e-04  1.912850e-03  1.734161e-03
##  [96]  1.922780e-03 -8.638021e-04  1.132063e-03 -1.341269e-03 -1.731161e-03
## [101] -1.777638e-03  3.921842e-05  3.322880e-04  7.994634e-04 -1.064268e-03
## [106] -2.337382e-04  1.325078e-04  2.913920e-04  1.839477e-03  3.618418e-02
## [111]  1.868118e-02  2.398438e-02  3.675391e-04  4.058815e-03 -1.549374e-02
## [116]  2.376740e-02 -2.782790e-02 -2.597028e-02  1.534999e-02  6.567838e-03
## [121]  1.441848e-03  6.705053e-04  1.281724e-03  2.313750e-03  1.439441e-02
## [126]  5.681919e-03 -2.082583e-04  1.628096e-03 -2.149866e-03 -7.148944e-04
## [131] -1.048437e-03  1.176948e-03 -1.005163e-03 -1.055594e-03 -2.203453e-03
## [136] -2.592809e-03 -1.287332e-03 -1.827870e-03 -1.810755e-03 -2.666722e-04
## [141]  2.063867e-03  1.982976e-03 -4.849844e-04  1.700788e-04 -1.894135e-04
## [146] -1.238095e-03 -1.154004e-03  3.898819e-04 -1.035475e-03  1.102433e-03
## [151]  2.200860e-04 -1.723675e-03 -1.498730e-03 -1.518853e-03  2.100760e-03
## [156] -2.030408e-03  4.571084e-04 -1.704186e-03  4.035629e-05 -1.338792e-03
## [161]  2.769769e-05 -4.918236e-04  8.152693e-03  7.554874e-02 -2.219631e-01
## [166] -1.586807e-01  1.505371e-03  2.233091e-02  1.947682e-01 -1.010053e-01
## [171]  3.711506e-02  2.019794e-02 -4.143202e-01  4.890080e-02  2.843147e-01
## [176]  2.348469e-01  9.914329e-02 -3.337568e-02 -9.274277e-02  3.641261e-01
## [181]  4.180649e+00  3.051322e+00 -4.089120e+00 -2.562073e-02 -8.110837e+00
## [186]  4.181220e+00  1.249743e-03  3.483872e-01 -5.223906e-04  6.956624e-04
## [191] -3.235796e+00
```

```r
#nn$fitted.values

nn.preds = predict(nn, TestData, type = "class")
nn.preds
```

```
##   [1] "Yes" "Yes" "Yes" "No"  "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes"
##  [13] "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes"
##  [25] "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "No" 
##  [37] "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "No" 
##  [49] "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes"
##  [61] "Yes" "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "No"  "Yes" "Yes"
##  [73] "Yes" "No"  "Yes" "No"  "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes"
##  [85] "No"  "Yes" "Yes" "Yes" "No"  "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "No" 
##  [97] "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "No"  "No" 
## [109] "Yes" "Yes" "No"  "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "No" 
## [121] "No"  "No"  "Yes" "No"  "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "Yes"
## [133] "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes"
## [145] "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "Yes"
## [157] "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes"
## [169] "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "No"  "No"  "No"  "No"  "Yes" "Yes"
## [181] "Yes" "Yes" "Yes" "No"  "No"  "No"  "No"  "Yes" "No"  "No"  "Yes" "Yes"
## [193] "Yes" "No"  "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No" 
## [205] "Yes" "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"  "Yes" "No"  "No" 
## [217] "Yes" "No"  "No"  "Yes" "Yes" "Yes" "Yes" "No"  "Yes" "No"  "Yes" "No" 
## [229] "No"  "Yes" "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "No"
```

```r
#CONFUSION MATRIX
table(TestData$Private, nn.preds)
```

```
##      nn.preds
##        No Yes
##   No   55  10
##   Yes  16 157
```

```r
#Accuracy
(55+157)/238
```

```
## [1] 0.8907563
```