ams580\_quiz6

Mustafa Yigit Isik

2023-03-22

#QUESTION 1  
  
#Load packages  
if (!requireNamespace("tidyverse")) install.packages('tidyverse')

## Loading required namespace: tidyverse

if (!requireNamespace("caret")) install.packages('caret')

## Loading required namespace: caret

if (!requireNamespace("neuralnet")) install.packages('neuralnet')

## Loading required namespace: neuralnet

if (!requireNamespace("keras")) install.packages('keras')

## Loading required namespace: keras

if (!requireNamespace("e1071")) install.packages('e1071')

## Loading required namespace: e1071

library(tidyverse)

## ── Attaching packages  
## ───────────────────────────────────────  
## tidyverse 1.3.2 ──

## ✔ ggplot2 3.4.0 ✔ purrr 1.0.1   
## ✔ tibble 3.1.8 ✔ dplyr 1.0.10  
## ✔ tidyr 1.3.0 ✔ stringr 1.5.0   
## ✔ readr 2.1.4 ✔ forcats 1.0.0   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice  
##   
## Attaching package: 'caret'  
##   
## The following object is masked from 'package:purrr':  
##   
## lift

library(neuralnet)

##   
## Attaching package: 'neuralnet'  
##   
## The following object is masked from 'package:dplyr':  
##   
## compute

library(keras)  
library(e1071)  
  
#Import data  
data <- read.csv('/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/quizzes/quiz6/GreatUnknown.csv')  
data <- na.omit(data)  
cat("There are ", nrow(data), " observations left after this step" )

## There are 4601 observations left after this step

str(data)

## 'data.frame': 4601 obs. of 13 variables:  
## $ w1 : num 0.32 0.14 1.23 0.63 0.63 1.85 1.92 1.88 0.61 0.19 ...  
## $ w2 : num 0 0.21 0.19 0.31 0.31 0 0 0 0.3 0.38 ...  
## $ w3 : num 0 0.94 0.25 0.63 0.63 0 0.64 0 0.76 0 ...  
## $ w4 : num 0.32 0.14 0.06 0.31 0.31 0 0.96 0 0 0 ...  
## $ w5 : num 0 0 0.32 0 0 0 0 0 3.53 0.06 ...  
## $ w6 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w7 : num 0 0 0.06 0 0 0 0 0 0 0 ...  
## $ w8 : num 0 0 0.06 0 0 0 0 0 0 0 ...  
## $ w9 : num 0 0 0.01 0 0 0 0 0 0 0.04 ...  
## $ w10: num 0 0.132 0.143 0.137 0.135 0.223 0.054 0.206 0.271 0.03 ...  
## $ w11: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w12: num 0 0.18 0.184 0 0 0 0.054 0 0.203 0.081 ...  
## $ y : int 1 1 1 1 1 1 1 1 1 1 ...

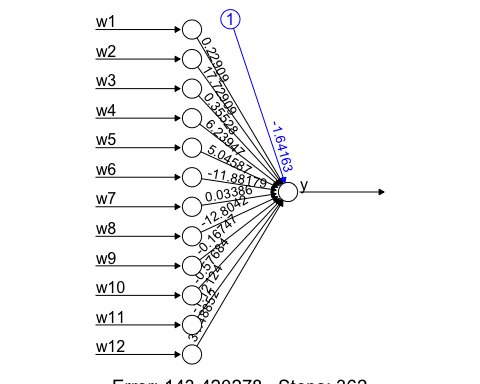
set.seed(123)  
split <- data$y %>% createDataPartition(p = 0.75, list = FALSE)  
train.data <- data[split, ]  
test.data <- data[-split, ]  
str(train.data) # 3451 obs

## 'data.frame': 3451 obs. of 13 variables:  
## $ w1 : num 0.32 0.14 1.23 0.63 0.63 1.92 0.61 0.19 0 0.38 ...  
## $ w2 : num 0 0.21 0.19 0.31 0.31 0 0.3 0.38 0.96 0.25 ...  
## $ w3 : num 0 0.94 0.25 0.63 0.63 0.64 0.76 0 1.92 0 ...  
## $ w4 : num 0.32 0.14 0.06 0.31 0.31 0.96 0 0 0 0 ...  
## $ w5 : num 0 0 0.32 0 0 0 3.53 0.06 0 0 ...  
## $ w6 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w7 : num 0 0 0.06 0 0 0 0 0 0.96 0 ...  
## $ w8 : num 0 0 0.06 0 0 0 0 0 0 0 ...  
## $ w9 : num 0 0 0.01 0 0 0 0 0.04 0 0.022 ...  
## $ w10: num 0 0.132 0.143 0.137 0.135 0.054 0.271 0.03 0 0.044 ...  
## $ w11: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w12: num 0 0.18 0.184 0 0 0.054 0.203 0.081 0 0 ...  
## $ y : int 1 1 1 1 1 1 1 1 1 1 ...

str(test.data) # 1150 obs

## 'data.frame': 1150 obs. of 13 variables:  
## $ w1 : num 1.85 1.88 0.9 1.27 0.76 2.94 1.16 0 0 1.46 ...  
## $ w2 : num 0 0 0.9 0.42 0.15 0 0 1.66 0 0.48 ...  
## $ w3 : num 0 0 0.9 1.27 0 0 0 0 0 0 ...  
## $ w4 : num 0 0 0 1.27 0 2.94 1.16 0 3.33 0.48 ...  
## $ w5 : num 0 0 0 0.42 0.1 0 0 0 0 0 ...  
## $ w6 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w7 : num 0 0 0 0.42 0 0 0 0 0 0 ...  
## $ w8 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ w9 : num 0 0 0 0 0.042 0.404 0 0 0 0 ...  
## $ w10: num 0.223 0.206 0 0.063 0.101 0.404 0.133 0 0.352 0.081 ...  
## $ w11: num 0 0 0 0 0.016 0 0 0 0 0 ...  
## $ w12: num 0 0 0 0.063 0.046 0 0 0 0 0.244 ...  
## $ y : int 1 1 1 1 1 1 1 1 1 1 ...

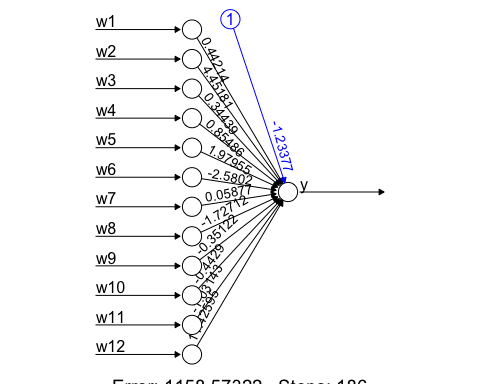
#QUESTION 2 - Part 1  
set.seed(123)  
model <- neuralnet(y~., data=train.data, hidden=0, err.fct="sse", linear.output=F)  
plot(model, rep = "best")



#QUESTION 2 - Part 2  
probs = predict(model, test.data)  
predicted.classes = ifelse(probs > .5, 1, 0)  
confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 649 75  
## 1 45 381  
##   
## Accuracy : 0.8957   
## 95% CI : (0.8765, 0.9127)  
## No Information Rate : 0.6035   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7795   
##   
## Mcnemar's Test P-Value : 0.008113   
##   
## Sensitivity : 0.8355   
## Specificity : 0.9352   
## Pos Pred Value : 0.8944   
## Neg Pred Value : 0.8964   
## Prevalence : 0.3965   
## Detection Rate : 0.3313   
## Detection Prevalence : 0.3704   
## Balanced Accuracy : 0.8853   
##   
## 'Positive' Class : 1   
##

#QUESTION 3 - Part 1  
set.seed(123)  
model <- neuralnet(y~., data=train.data, hidden=0, err.fct="ce", linear.output=F)  
plot(model, rep = "best")



#QUESTION 3 - Part 2  
probs = predict(model, test.data)  
predicted.classes = ifelse(probs > .5, 1, 0)  
confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 658 109  
## 1 36 347  
##   
## Accuracy : 0.8739   
## 95% CI : (0.8533, 0.8926)  
## No Information Rate : 0.6035   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7291   
##   
## Mcnemar's Test P-Value : 2.241e-09   
##   
## Sensitivity : 0.7610   
## Specificity : 0.9481   
## Pos Pred Value : 0.9060   
## Neg Pred Value : 0.8579   
## Prevalence : 0.3965   
## Detection Rate : 0.3017   
## Detection Prevalence : 0.3330   
## Balanced Accuracy : 0.8545   
##   
## 'Positive' Class : 1   
##

#QUESTION 4 - Part 1  
set.seed(123)  
model = glm(y~., family = binomial, data = train.data)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

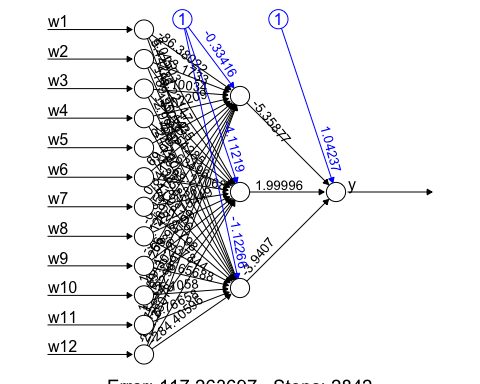
model

##   
## Call: glm(formula = y ~ ., family = binomial, data = train.data)  
##   
## Coefficients:  
## (Intercept) w1 w2 w3 w4 w5   
## -1.23379 0.44216 4.45185 0.34440 0.85478 1.97956   
## w6 w7 w8 w9 w10 w11   
## -2.58020 0.05878 -1.72696 -0.35121 -0.44284 -1.03140   
## w12   
## 11.42602   
##   
## Degrees of Freedom: 3450 Total (i.e. Null); 3438 Residual  
## Null Deviance: 4625   
## Residual Deviance: 2317 AIC: 2343

#QUESTION 4 - Part 2  
probabilities = model %>% predict(test.data, type = "response")  
predicted.classes = ifelse(probs > .5, 1, 0)  
confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 658 109  
## 1 36 347  
##   
## Accuracy : 0.8739   
## 95% CI : (0.8533, 0.8926)  
## No Information Rate : 0.6035   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7291   
##   
## Mcnemar's Test P-Value : 2.241e-09   
##   
## Sensitivity : 0.7610   
## Specificity : 0.9481   
## Pos Pred Value : 0.9060   
## Neg Pred Value : 0.8579   
## Prevalence : 0.3965   
## Detection Rate : 0.3017   
## Detection Prevalence : 0.3330   
## Balanced Accuracy : 0.8545   
##   
## 'Positive' Class : 1   
##

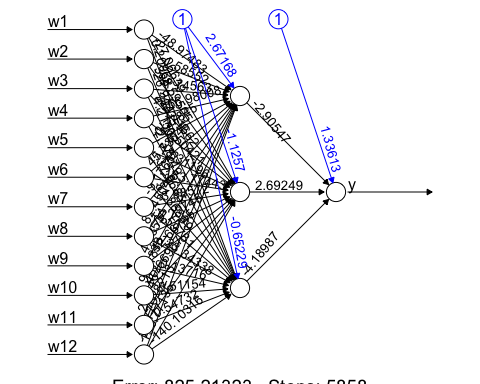
#QUESTION 5 - Part 1  
set.seed(123)  
model <- neuralnet(y~., data=train.data, hidden=3, err.fct="sse", linear.output=F)  
plot(model, rep = "best")



#QUESTION 5 - Part 2  
probs = predict(model, test.data)  
predicted.classes = ifelse(probs > .5, 1, 0)  
confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 652 65  
## 1 42 391  
##   
## Accuracy : 0.907   
## 95% CI : (0.8887, 0.9231)  
## No Information Rate : 0.6035   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.8039   
##   
## Mcnemar's Test P-Value : 0.03344   
##   
## Sensitivity : 0.8575   
## Specificity : 0.9395   
## Pos Pred Value : 0.9030   
## Neg Pred Value : 0.9093   
## Prevalence : 0.3965   
## Detection Rate : 0.3400   
## Detection Prevalence : 0.3765   
## Balanced Accuracy : 0.8985   
##   
## 'Positive' Class : 1   
##

#QUESTION 6 - Part 1  
set.seed(123)  
model <- neuralnet(y~., data=train.data, hidden=3, err.fct="ce", linear.output=F)  
plot(model, rep = "best")



#QUESTION 6 - Part 2  
probs = predict(model, test.data)  
predicted.classes = ifelse(probs > .5, 1, 0)  
confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 655 71  
## 1 39 385  
##   
## Accuracy : 0.9043   
## 95% CI : (0.8859, 0.9207)  
## No Information Rate : 0.6035   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7977   
##   
## Mcnemar's Test P-Value : 0.003119   
##   
## Sensitivity : 0.8443   
## Specificity : 0.9438   
## Pos Pred Value : 0.9080   
## Neg Pred Value : 0.9022   
## Prevalence : 0.3965   
## Detection Rate : 0.3348   
## Detection Prevalence : 0.3687   
## Balanced Accuracy : 0.8941   
##   
## 'Positive' Class : 1   
##

#QUESTION 7  
#It seems that the accuracy is quite close in part 5 and 6, (0.907 and 0.9043), respectively.  
#However, since the choice has to be made, due to slight advantage, has to be part 5,  
#which is sse with 3 hidden layers  
  
set.seed(123)  
model <- neuralnet(y~., data=train.data, hidden=3, err.fct="sse", linear.output=F)  
probs = predict(model, test.data)  
  
#I couldn't figure out column concat while in the quiz so I did write 2 csv files and then  
#got them together in excel, probably have a more elegant way to do but this also looks   
#foul-proof for quick quiz solution  
  
write.csv(probs, "/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/quizzes/quiz6/sse\_test\_output.csv", row.names=FALSE)  
write.csv(test.data, "/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/quizzes/quiz6/regular\_testdata.csv", row.names=FALSE)  
  
#I output the concat results under "test.data\_concatwith\_sse.pred.csv" and attached   
#with rmd, docx files

```