project2\_nn\_classification

EMVP

2023-05-01

#Q1  
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)  
  
#load data  
data <- read.csv('/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/project2/Titanic.csv')  
#exclude unnecessary 4 columns and empty values in age column   
data <- subset(data, select = -c(PassengerId,Name,Ticket,Cabin))  
data <- subset(data, is.na(Age) == FALSE)  
  
data$Embarked[data$Embarked == ""] = NA  
data <- subset(data, is.na(Embarked) == FALSE)  
data <- model.matrix(~., data = data)[,-1]  
data <- data.frame(data)  
dim(data)[1]

## [1] 712

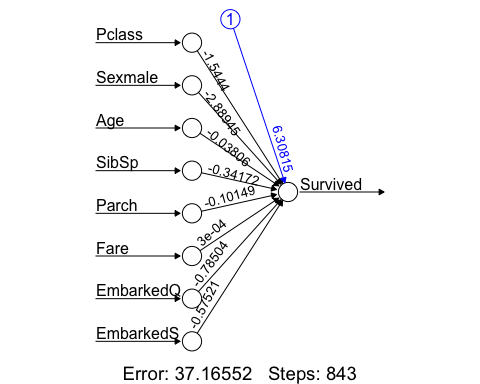
# 712 passengers are left after this step  
  
set.seed(123)  
training.samples <- data$Survived %>% createDataPartition(p = 0.75, list = FALSE)  
train.data <- data[training.samples, ]  
test.data <- data[-training.samples, ]  
str(train.data) # 534 obs

## 'data.frame': 534 obs. of 9 variables:  
## $ Survived : num 1 0 0 0 1 1 1 0 0 0 ...  
## $ Pclass : num 1 3 1 3 3 3 1 3 3 3 ...  
## $ Sexmale : num 0 1 1 1 0 0 0 1 1 0 ...  
## $ Age : num 38 35 54 2 27 4 58 20 39 14 ...  
## $ SibSp : num 1 0 0 3 0 1 0 0 1 0 ...  
## $ Parch : num 0 0 0 1 2 1 0 0 5 0 ...  
## $ Fare : num 71.28 8.05 51.86 21.07 11.13 ...  
## $ EmbarkedQ: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ EmbarkedS: num 0 1 1 1 1 1 1 1 1 1 ...

str(test.data) # 178 obs

## 'data.frame': 178 obs. of 9 variables:  
## $ Survived : num 0 1 1 1 1 0 0 0 0 1 ...  
## $ Pclass : num 3 3 1 2 2 2 3 1 3 3 ...  
## $ Sexmale : num 1 0 0 0 0 1 0 1 0 0 ...  
## $ Age : num 22 26 35 14 55 35 8 28 40 19 ...  
## $ SibSp : num 1 0 1 1 0 0 3 1 1 0 ...  
## $ Parch : num 0 0 0 0 0 0 1 0 0 0 ...  
## $ Fare : num 7.25 7.92 53.1 30.07 16 ...  
## $ EmbarkedQ: num 0 0 0 0 0 0 0 0 0 1 ...  
## $ EmbarkedS: num 1 1 1 0 1 1 1 0 1 0 ...

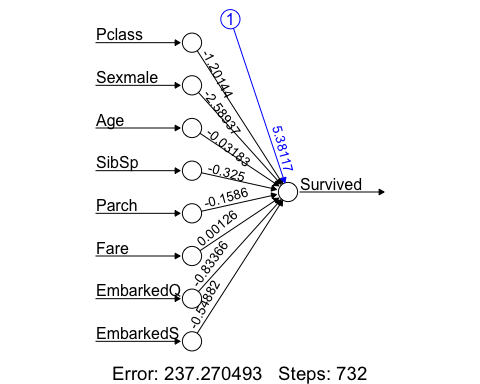
#Q2  
set.seed(123)  
model = neuralnet(Survived~., data = train.data, hidden = 0, err.fct = "sse", linear.output = F)  
plot(model, rep = "best")



probabilities = predict(model, test.data)  
predicted.classes = ifelse(probabilities > 0.5, 1, 0)  
  
confusionMatrix(factor(predicted.classes), factor(test.data$Survived), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 88 22  
## 1 12 56  
##   
## Accuracy : 0.809   
## 95% CI : (0.7434, 0.8639)  
## No Information Rate : 0.5618   
## P-Value [Acc > NIR] : 3.086e-12   
##   
## Kappa : 0.6065   
##   
## Mcnemar's Test P-Value : 0.1227   
##   
## Sensitivity : 0.7179   
## Specificity : 0.8800   
## Pos Pred Value : 0.8235   
## Neg Pred Value : 0.8000   
## Prevalence : 0.4382   
## Detection Rate : 0.3146   
## Detection Prevalence : 0.3820   
## Balanced Accuracy : 0.7990   
##   
## 'Positive' Class : 1   
##

#Q3  
set.seed(123)  
model = neuralnet(Survived~., data = train.data, hidden = 0, err.fct = "ce", linear.output = F)  
plot(model, rep = "best")



probabilities = predict(model, test.data)  
predicted.classes = ifelse(probabilities > 0.5, 1, 0)  
  
confusionMatrix(factor(predicted.classes), factor(test.data$Survived), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 83 22  
## 1 17 56  
##   
## Accuracy : 0.7809   
## 95% CI : (0.7129, 0.8393)  
## No Information Rate : 0.5618   
## P-Value [Acc > NIR] : 8.072e-10   
##   
## Kappa : 0.5518   
##   
## Mcnemar's Test P-Value : 0.5218   
##   
## Sensitivity : 0.7179   
## Specificity : 0.8300   
## Pos Pred Value : 0.7671   
## Neg Pred Value : 0.7905   
## Prevalence : 0.4382   
## Detection Rate : 0.3146   
## Detection Prevalence : 0.4101   
## Balanced Accuracy : 0.7740   
##   
## 'Positive' Class : 1   
##

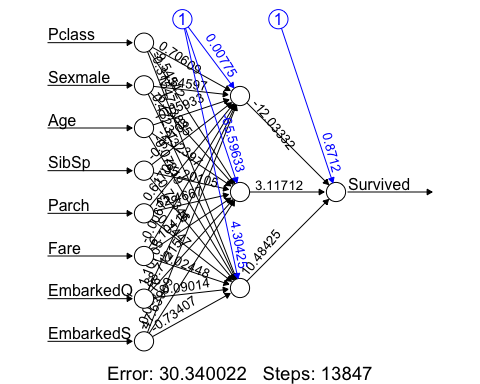
#Q4  
set.seed(123)  
model = glm(Survived~., family = binomial, data = train.data)  
  
model

##   
## Call: glm(formula = Survived ~ ., family = binomial, data = train.data)  
##   
## Coefficients:  
## (Intercept) Pclass Sexmale Age SibSp Parch   
## 5.385949 -1.202383 -2.589957 -0.031866 -0.325204 -0.158540   
## Fare EmbarkedQ EmbarkedS   
## 0.001253 -0.834327 -0.549692   
##   
## Degrees of Freedom: 533 Total (i.e. Null); 525 Residual  
## Null Deviance: 715.8   
## Residual Deviance: 474.5 AIC: 492.5

probabilities = model %>% predict(test.data, type = "response")  
predicted.classes = ifelse(probabilities > 0.5, 1, 0)  
  
confusionMatrix(factor(predicted.classes), factor(test.data$Survived), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 83 22  
## 1 17 56  
##   
## Accuracy : 0.7809   
## 95% CI : (0.7129, 0.8393)  
## No Information Rate : 0.5618   
## P-Value [Acc > NIR] : 8.072e-10   
##   
## Kappa : 0.5518   
##   
## Mcnemar's Test P-Value : 0.5218   
##   
## Sensitivity : 0.7179   
## Specificity : 0.8300   
## Pos Pred Value : 0.7671   
## Neg Pred Value : 0.7905   
## Prevalence : 0.4382   
## Detection Rate : 0.3146   
## Detection Prevalence : 0.4101   
## Balanced Accuracy : 0.7740   
##   
## 'Positive' Class : 1   
##

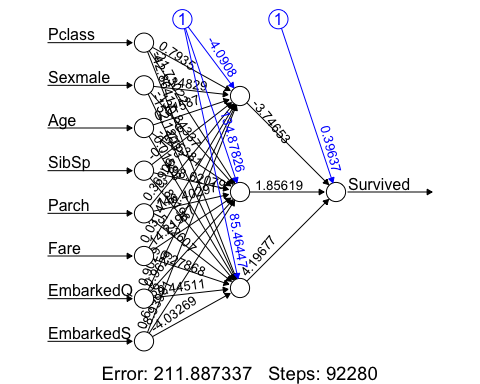
#Q5  
set.seed(123)  
model = neuralnet(Survived~., data = train.data, hidden = 3, err.fct = "sse", linear.output = F)  
plot(model, rep = "best")



probabilities = predict(model, test.data)  
predicted.classes = ifelse(probabilities > 0.5, 1, 0)  
  
confusionMatrix(factor(predicted.classes), factor(test.data$Survived), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 91 23  
## 1 9 55  
##   
## Accuracy : 0.8202   
## 95% CI : (0.7558, 0.8737)  
## No Information Rate : 0.5618   
## P-Value [Acc > NIR] : 2.61e-13   
##   
## Kappa : 0.6275   
##   
## Mcnemar's Test P-Value : 0.02156   
##   
## Sensitivity : 0.7051   
## Specificity : 0.9100   
## Pos Pred Value : 0.8594   
## Neg Pred Value : 0.7982   
## Prevalence : 0.4382   
## Detection Rate : 0.3090   
## Detection Prevalence : 0.3596   
## Balanced Accuracy : 0.8076   
##   
## 'Positive' Class : 1   
##

#Q6  
set.seed(123)  
model = neuralnet(Survived~., data = train.data, hidden = 3, err.fct = "ce", linear.output = F)  
plot(model, rep = "best")



probabilities = predict(model, test.data)  
predicted.classes = ifelse(probabilities > 0.5, 1, 0)  
  
confusionMatrix(factor(predicted.classes), factor(test.data$Survived), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 90 25  
## 1 10 53  
##   
## Accuracy : 0.8034   
## 95% CI : (0.7373, 0.8591)  
## No Information Rate : 0.5618   
## P-Value [Acc > NIR] : 1.006e-11   
##   
## Kappa : 0.592   
##   
## Mcnemar's Test P-Value : 0.01796   
##   
## Sensitivity : 0.6795   
## Specificity : 0.9000   
## Pos Pred Value : 0.8413   
## Neg Pred Value : 0.7826   
## Prevalence : 0.4382   
## Detection Rate : 0.2978   
## Detection Prevalence : 0.3539   
## Balanced Accuracy : 0.7897   
##   
## 'Positive' Class : 1   
##