MATH 4322 Final Project Group 9

Neural Network Model

```r
# Imported libraries.
library(readr)
library(NeuralNetTools)
```

```
Warning: package 'NeuralNetTools' was built under R version 4.3.3
```

```r
library(nnet)

set.seed(35)

# Import the data from the csv in the correct format.
setwd("~/RStudio/Spring 2024/MATH 4322/Project/MATH4322")
cardio.data <- read_delim("cardio_train.csv", delim = ";",
                          escape_double = FALSE, trim_ws = TRUE)
```

```
Rows: 70000 Columns: 13
-- Column specification --------------------------------------------------------
Delimiter: ";"
dbl (13): id, age, gender, height, weight, ap_hi, ap_lo, cholesterol, gluc, ...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

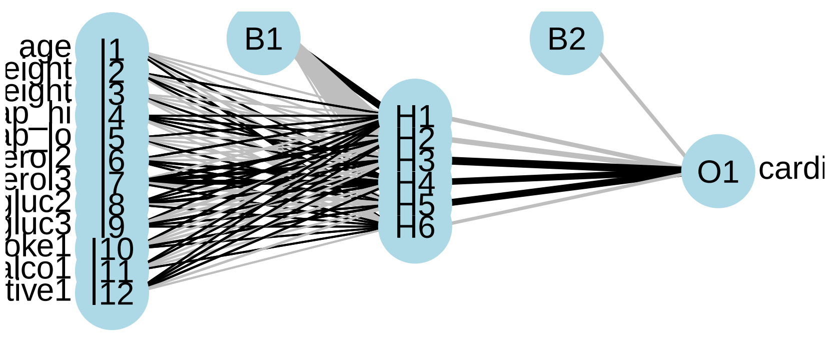
```r
# Take all of the classification variables as their factors.
cardio.data$gender = as.factor(cardio.data$gender)
cardio.data$cholesterol = as.factor(cardio.data$cholesterol)
cardio.data$gluc = as.factor(cardio.data$gluc)
cardio.data$smoke = as.factor(cardio.data$smoke)
cardio.data$alco = as.factor(cardio.data$alco)
cardio.data$active = as.factor(cardio.data$active)
cardio.data$cardio = as.factor(cardio.data$cardio)

# Separate the data into 80% training and 20% testing.
sample = sample(1 : nrow(cardio.data), floor(nrow(cardio.data) * 0.8))
cardio.train = cardio.data[sample, ]
cardio.test = cardio.data[-sample, ]

# Build the neural network.
cardio.model = nnet(cardio ~ . - id - gender, data = cardio.train,
             size = 6, rang = 0.5, decay = 5e-2, maxit = 5000)
```

```
# weights:  85
initial  value 39022.927542
iter  10 value 38814.263045
iter  20 value 38643.556537
iter  30 value 38619.193201
iter  40 value 38599.346136
iter  50 value 36518.868686
iter  60 value 33905.075190
iter  70 value 33593.251770
iter  80 value 33294.114889
iter  90 value 32249.152366
iter 100 value 31962.110688
iter 110 value 31514.126189
iter 120 value 31401.592821
iter 130 value 30969.960281
iter 140 value 30942.104981
iter 150 value 30906.273917
iter 160 value 30885.923042
iter 170 value 30711.736212
iter 180 value 30656.327509
iter 190 value 30620.581492
iter 200 value 30577.575309
iter 210 value 30552.096076
iter 220 value 30538.342087
iter 230 value 30529.178590
iter 240 value 30526.349646
iter 250 value 30518.701286
iter 260 value 30510.719464
iter 270 value 30472.858266
iter 280 value 30454.156349
iter 290 value 30441.490266
iter 300 value 30427.124792
iter 310 value 30409.836919
iter 320 value 30377.488109
iter 330 value 30362.476826
iter 340 value 30343.864594
iter 350 value 30334.529335
iter 360 value 30329.206365
iter 370 value 30325.401306
iter 380 value 30315.604903
iter 390 value 30298.275813
iter 400 value 30279.330848
iter 410 value 30274.533158
iter 420 value 30273.247609
iter 430 value 30272.060312
iter 440 value 30271.366855
iter 450 value 30270.638513
iter 460 value 30270.316483
iter 470 value 30270.096038
iter 480 value 30269.520857
iter 490 value 30267.257482
iter 500 value 30266.472018
iter 510 value 30266.124146
iter 520 value 30265.729148
iter 530 value 30265.277089
iter 540 value 30265.184041
iter 540 value 30265.183814
iter 540 value 30265.183803
final  value 30265.183803
converged
```

```
# Print the model's information, plot, summary, and garson plot.
print(cardio.model)
```

```
a 12-6-1 network with 85 weights
inputs: age height weight ap_hi ap_lo cholesterol2 cholesterol3 gluc2 gluc3 smoke1 alco1 act
output(s): cardio
options were - entropy fitting  decay=0.05
```

```r
plotnet(cardio.model)
```

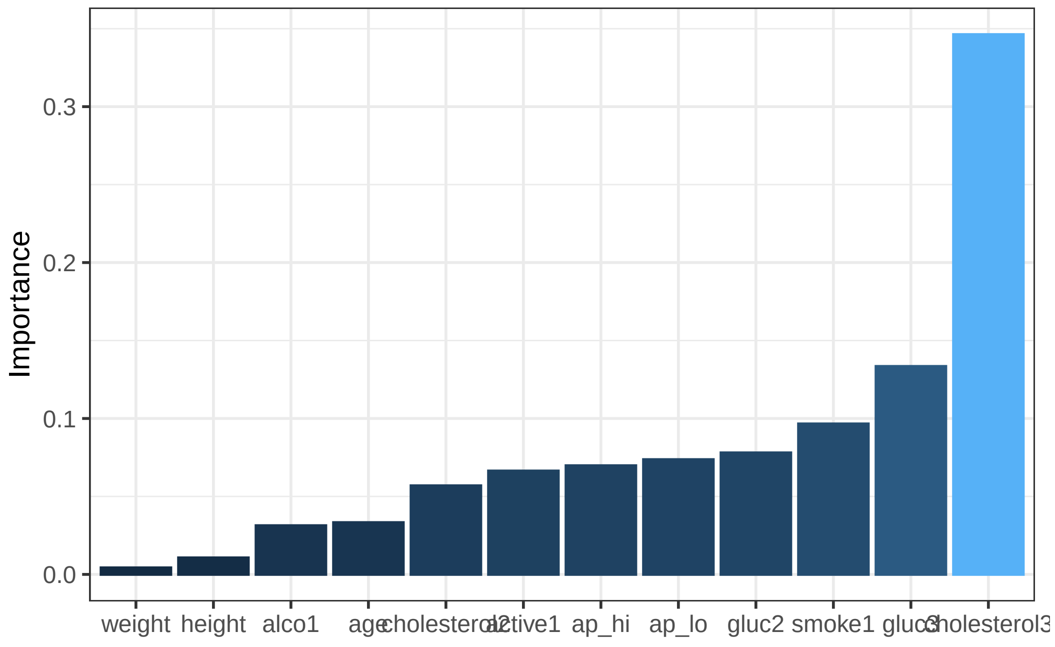


```r
summary(cardio.model)
```

```
a 12-6-1 network with 85 weights
options were - entropy fitting  decay=0.05
  b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1  i9->h1
   6.54    0.00    0.01   -0.01    0.05    0.00    0.41   -0.35    0.31    0.26
i10->h1 i11->h1 i12->h1
   0.44    0.27    0.19
  b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2  i9->h2
 -14.86    0.00   -0.01   -0.01    0.18    0.00   -0.05    1.78    0.02   -1.21
i10->h2 i11->h2 i12->h2
  -0.33    0.07    0.47
  b->h3  i1->h3  i2->h3  i3->h3  i4->h3  i5->h3  i6->h3  i7->h3  i8->h3  i9->h3
 -13.31    0.00   -0.01    0.00    0.14    0.00   -0.02    1.93    0.02   -1.05
i10->h3 i11->h3 i12->h3
  -0.36   -0.04    0.20
  b->h4  i1->h4  i2->h4  i3->h4  i4->h4  i5->h4  i6->h4  i7->h4  i8->h4  i9->h4
  -8.07    0.00    0.00    0.02    0.04   -0.01    0.55    3.44    1.27    0.48
i10->h4 i11->h4 i12->h4
```

```
  -0.61   -0.24    0.27
  b->h5  i1->h5  i2->h5  i3->h5  i4->h5  i5->h5  i6->h5  i7->h5  i8->h5  i9->h5
  -0.41    0.00    0.01    0.00   -0.02    0.03    0.25   -1.93   -0.54    0.19
i10->h5 i11->h5 i12->h5
   0.50   -0.03   -0.43
  b->h6  i1->h6  i2->h6  i3->h6  i4->h6  i5->h6  i6->h6  i7->h6  i8->h6  i9->h6
   0.00    0.03   -0.01    0.00   -0.05   -0.07    0.00    0.00    0.00    0.00
i10->h6 i11->h6 i12->h6
   0.00    0.00    0.00
  b->o  h1->o  h2->o  h3->o  h4->o  h5->o  h6->o
 -2.20  -3.08  -4.15   7.27   5.53   6.03  -1.75
```

```
garson(cardio.model)
```



```
# Training prediction values.
cardio.train.predict = predict(cardio.model, cardio.train, type = "class")
cardio.train.values = cardio.train$cardio

# Testing prediction values.
cardio.test.predict = predict(cardio.model, cardio.test, type = "class")
cardio.test.values = cardio.test$cardio
```

```r
# Training confusion matrix.
cardio.train.table = table(cardio.train.values, cardio.train.predict)
cardio.train.table
```

```
                   cardio.train.predict
cardio.train.values     0     1
                  0 21537  6477
                  1  8356 19630
```

```r
# Testing confusion matrix.
cardio.test.table = table(cardio.test.values, cardio.test.predict)
cardio.test.table
```

```
                  cardio.test.predict
cardio.test.values    0    1
                 0 5386 1621
                 1 2073 4920
```

Questions:

Include the inputs and the outputs of the data.

- Input
  - age
    - Shown in days.
  - height
    - Shown in cm.
  - weight
    - Shown in float kg.
  - gender
    - Shown as a categorical code.
  - ap_hi
    - Systolic blood pressure.
  - ap_lo

        - Diastolic blood pressure.
    - cholesterol
        - Shown in three different ways: 1: normal cholesterol, 2: above normal cholesterol, and 3: well above cholesterol.
    - gluc
        - Shown in three different ways: gluc1: normal glucose, gluc2: above normal glucose, and gluc3: well above normal glucose.
    - smoke
        - Binary classification variable: smoke0: non-smoker, smoke1: smoker.
    - alco
        - Binary classification variable: alco0: low alcohol intake, alco1: high alcohol intake.
    - active
        - Binary classification variable: active0: low physical activity, active1: regular physical activity.

Include the question you are wanting to answer and the response variable.

- Output (response variable)
    - cardio
        - Indicates the presence or absence of cardiovascular disease.

Question we are trying to answer:

- How do lifestyle factors correlate with the risk of developing cardiovascular disease?

Why are you using a neural network as one of your models?

- We choose a Neural network model as we wanted to see if the dataset contained more complex data that possibly not linear, as well as investigating all possible interactions between the input variables and having an effect on the response variable.

What are the advantages and disadvantages of using a neural network model?

- Advantages of using a neural network model is that it can handle complex data that is not linear and adapt to changing input, also neural networks can detect all possible interactions between predictor variables, and it can also be developed using multiple different training algorithms.

- While some of the disadvantages can be the “black-box” nature and have limited ability to identify possible casual relationships, and also limited to the computational power that we have access to, as having additional data or adding more complexity could affect the time to compute a model.

Draw a sketch of the neural network, labeling each node.

Why did you choose to use only 1 hidden layer?

- We decided to use a single hidden layer as our data did not deviate that much from a linear relationship and did not have as much noise thus the data not being as complex, we decided a single hidden layer would be optimal.

Why does the hidden layer have only 6 nodes?

- We decided that using 6 nodes in the hidden layer would be a good starting point for our model as it was half of our initial input nodes which was 12. However, we found out it’s essential to validate this choice through testing and adjust based on the specific needs and outcomes of your model’s performance on validation datasets and could be implemented in further iterations of the model.

Why was entropy used as the activation function?

- Regarding the nnet() function in R, “entropy” refers to the cross-entropy loss function, not an activation function. Cross-entropy is used to optimize classification accuracy by penalizing the difference between predicted probabilities and actual class labels. It complements the logistic and softmax activation functions used in nnet() for binary and multi-class classification, respectively. Thus, the term “entropy” describes the loss function used for training the model efficiently.

Why are the initial weights 0.5?

- The initial weights in a neural network play a crucial part in the learning process, as it affect the speed of convergence in a neural network. The reason why weights can't generally be set to 0 is because the neurons in the network would receive the same signal and, therefore, will create inefficiency in training. This is why we decided to use 0.5 to make sure that the neurons are small enough to ensure that there’s no numerical instability—allowing the learning to be normal, rather than too slow or diverging.

What can you inference from the garson plot of the neural network?

- The Garson plot is used to interpret the relative importance of input features in a neural network. It decomposes the neural network weights into contributions by each input variable toward the output. In the Garson plot that we produced, it can be inference that the variables cholesterol, gluc, and smoke were the variables that had the most significant factors in the cardio dataset, as these factors had the strongest correlation to increased risk of heart disease.

How does the neural network help inference the factors that contribute to heart disease?

- By analyzing the weights and influence of different inputs in the network, you can identify which factors are most predictive of cardiovascular diseases. This insight can help in understanding risk factors better and potentially guiding preventive measures as people who have significant predictors as lifestyle factors can be aware of the at-risk factors.

Summarize your findings.

- The neural network model, trained on various patient health metrics, can predict the presence of cardiovascular disease with a reasonable level of accuracy. The model identifies key predictors and their influence, offering insights into the underlying patterns and risk factors associated with cardiovascular conditions.

How does the neural network compare to the logistic model?

- The neural network model had a slightly lower test error rate than the logistic model with a 26.38% testing error rate compared to a 27.74% testing error rate of the logistic model. Thus displaying that the neural network model performed better on the testing data set than the logistic model. The neural network model also better indicated which predictors are more statistically significant in causing heart disease.

Introductory paragraph?

In this report, we employ a neural network to investigate the correlation between lifestyle factors—such as diet, exercise, and smoking—and the risk of cardiovascular disease. Using inputs ranging from physiological data to behavioral patterns, we aim to predict the occurrence of cardiovascular conditions. This analysis is designed to highlight key lifestyle influences on heart health and inform preventative strategies.