Data 630 – Fall 2018

Assignment 4 – Backpropagation Neural Network Analysis of Heart Disease

Kenneth Lulie

Dr. Gates

University of Maryland University College

November 11, 2018

**Neural Network Analysis of Heart Disease Survival using the Hosmer Dataset**

**Introduction**

The term heart disease covers many different conditions that negatively affect your heart. Diseases under this label include blood vessel diseases such as coronary artery disease, arrhythmias (heart rhythm problems), as well as heart defects present at birth which are called congenital heart defects (Mayoclinic, 2018). Initial symptoms may present as chest related pain or discomfort (angina), shortness of breath, pain or weakness in your legs or arms, or pain in the neck, jaw, throat, upper abdomen or back. Many people are not diagnosed with cardiovascular disease until their first heart attack occurs.

The impact of heart disease is hard to overstate, as one in every four deaths in the United States is from Heart Disease (CDC). Heart disease is the leading cause of death in the U.S. of both men and women with over 630,000 people dying of heart disease every year. Heart disease has only the human impact on the individual men and women it affects and their families, but also negatively impacts the economy as a whole. The CDC estimates that heart disease costs the United States 200 billion dollars every year when factoring in expenses related to health care services, medications, and lost productivity. The risk factors of heart disease are well studied. The major risk factors are high blood pressure, high LDL cholesterol, and smoking with about one in every two Americans having at least one of these three risk factors. Other medical conditions as well as lifestyle choices can also increase risk such as diet, diabetes, physical inactivity, obesity, and excessive alcohol use.

While heart disease is a serious condition, when the condition is properly diagnosed and treated it may be survivable. In this analysis, the impact of different factors such as Age, Sex, Peak Cardiac Enzyme, Cardiogenic Shock Complications, Left Heart Failure Complications, MI Order, and MI Type will be analyzed to see if they allow predictions of if a patient will survive their initial hospital admission for heart disease. The specific aim of this analysis is to use a backpropagation Neural Network algorithm to build a model based on these variables which will predict the probability of a patient surviving their initial admission so informed medical decisions regarding treatment may be taken. The model will also be able to produce a confidence rating between 0 and 1 showing the probability the model assigns to survival. The model will be tuned to different parameters to attempt to improve overall accuracy and recall.

**Analysis and Model Demonstration**

**Subsection: Data Information, Cleaning, and Preprocessing**

The data used in this analysis comes from the UMUC provided list of approved datasets, originally provided by D.W. Hosmer and S. Lemeshow in 1998. The dataset is that of patient information for 481 patients admitted for heart disease, presumably after suffering a heart attack, who, if they survived, were readmitted after a second heart attack. The patients have been made anonymous with a numeric ID. The data includes a classification showing if the patient survived the initial admission as well as if they survived the follow up admission.

The analysis of this study was implemented using Rstudio version 3.5.1. There were 481 patients in the dataset provided. There were no missing values in the observations provided. The data as provided contained 14 variables, specifically ID, AGE, SEX, CPK (Peak Cardiac Enzyme), SHO (Cardiogenic Shock Complications), CHF (Left Heart Failure Complications), MIORD (MI Order), MITYPE, Year (Cohort Year), YRGRP (Grouped Cohort Year), LENSTAY (Length of Hospital Stay in Days), DSTAT (Discharge status from Hospital), LENFOL (Total length of follow-up from hospital admission), FSTAT (Status as of Last-Follow up).

The data was reviewed for missing values and unique identifiers. No missing values were found but as ID was a unique reference variable, it was removed. The data was also reviewed outliers or other possible errors in the data. No outliers or errors in the data were found. The data was also reviewed for data types to ensure that the data was of the correct type. The continuous variables of Age, and CPK were scaled to prepare the data for the backpropagation Neural Network algorithm. Additionally, as the MITYPE was a factorial variable with 3 values, it was transformed into two separate Boolean variables, MIQ and MINQ, and then the MITYPE variable was removed. MIQ represents a value of 1 in MITYPE which indicates the presence of Q-waves, MINQ represent a value of 2 which indicates the absence of Q waves and both at 0 represent a value of 3 in MITYPE which indicates the results were Indeterminate. As the purpose of the analysis is to create a model to predict the patients discharge status after the first admission, variables were removed that would not be known at the time of the first admission, specifically the length of the follow up time, the length of the initial stay, and the follow up discharge status. The variables identifying the year the patient was admitted was also removed as the historical differences between the years is beyond the scope of this analysis. The remaining variables were all Boolean and required no further data preprocessing.

A descriptive analysis including mean, median, 1st and 3rd quartiles, min and max of the Age and CPK variables are provided in **Table 1**. A boxplot distribution of Age shows us that as expected, while the age range covers almost all adult ages from 24 to 98, the patients are generally over 60 the typical age of when heart disease problems start **(Figure 1)**. A summary of the frequency of the categorical variables are presented in **Table 2**. Of the 481 patients, 308 were discharged from their initial admission alive, and 173 were discharged dead.

Table 1 Distribution of Numeric Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Age | | CPK | |
| Min | 24 | Min | 10 |
| 1st Qu. | 59 | 1st Qu. | 270 |
| Median | 68 | Median | 587 |
| Mean | 67.48 | Mean | 941.52 |
| 3rd Qu. | 77 | 3rd Qu. | 1146 |
| Max. | 98 | Max. | 9000 |

The distribution appears to be skewed towards males, with 287 patients as male and 194 patients as female. This is concurrent with previous expectations as heart disease affects men at a higher rate. Only 6 patients had MITYPE of ‘Indeterminate’. The distribution also shows that Cardiogenic Shock Complications and Left Heart Failure Complications are fairly common with 40.9% and 40.7% patients presenting them respectively. The review of these categorical variables shows that each of them has one value more common than the other but generally have enough observations in each category to retain for further analysis.

**Subsection: Analysis and Model Methods**

The preprocessed data was used in this analysis with the backpropagation Neural Net algorithm for classification to predict if the patient would be discharged dead or alive. This Neural Net algorithm also predicts the confidence from 0 to 1 of its prediction for each observation which allows for further tuning of threshold cutoffs, which could also be used for categorizing patients into different risk groups. Neural Net algorithms are well known

Table 2. Distribution of Categorical Variables

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sex | | Cardiogenic Shock Complications | | Left Heart Failure Complications | |
| Male | 287 | No | 287 | No | 285 |
| Female | 194 | Yes | 194 | Yes | 196 |
| MI Order | | MI Type | | Discharge Status from Hospital | |
| First | 308 | Q-Wave | 280 | Alive | 308 |
| Recurrent | 173 | Not Q-Wave | 195 | Dead | 173 |
| Indeterminate | 6 |

for being a “black box” as the models it generates are infamously difficult to understand for humans. However, it is possible to decipher the model with different methods such as changing variable values or removing variables but such an analysis is out of scope for this project.

The backpropagation Neural Net algorithm is a neural network learning algorithm (Han, 2011). A neural network is a model of connected inputs and outputs where each connection has a certain weight associated to it. As the neural network learns, the weights are adjusted to be able to correctly predict the class label of the input tuples. Neural Networks have long training times and are best used in applications where this is acceptable. One major criticism of Neural Networks is that they have poor human interpretability. It is very difficult for humans to look at a model of a Neural Network and visually understand the meaning behind the connections in the model. Some of the advantages of Neural Networks are that they are very tolerant of ‘noisy’ data as well as their ability to classify patterns on which they were not trained. Additionally, they are capable of handling both continuous inputs and outputs unlike typical decision tree algorithms. Additionally, as Neural Networks are essentially parallel, it is possible to use parallelization techniques to reduce the time necessary to run or train models.

Backpropogation Neural Networks is a multiplayer feed-forward neural network (Han, 2011). It consists of one input player, one or more hidden layers, and one final output layer. Each layer is made up of units. The input neurons take their values from the input tuples. These values are then sent to the hidden layer and weighted based on each connection. The hidden layer then transmits to the next hidden layer again changing the values based on the weights until it passes to the final output layer which will give the prediction for that specific tuple. The units for the input layer are called input units, the hidden and output layer are called neurodes, based on their resemblance to Neurons in the human brain from which Neural Networks were inspired. In addition to the weights changing the values as they go through each connection, each Neurode has an additional property called a Bias which are constant weights additionally changing the values as they flow through each specific Neurode. Backpropagation Neural Networks are feed-forward because none of the weights cycles back to the neurodes of a previous layer, and they are fully-connected in that each unit provides inputs to each unit in the next layer. Each output unit takes as its input the weighted sum of the outputs from units in the previous layer (Han, 2011). From a statistical point of view, the model performs nonlinear regression and with enough hidden layers and training sample a multi-layer feed-forward Neural Network can closely approximate any function which is what makes such a powerful tool.

To measure the effectiveness of the backpropagation Neural Network models produced in this analysis, confusion matrixes will be used to review the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). From these numbers Error Rate (FP + FN)/Total Number of Predictions), Accuracy (TP + TN/Total Number of Predictions), Sensitivity (TP/ Total Actual Positives), and Specificity (TN/Total Actual Negatives), Precision (TP/TP + FP), Recall (TP/ TP + FN) can be calculated and used to measure the model. However, as the purpose of the model is assisting in calculating the probability a patient will be discharged dead so proper medical interventions may be taken, the two main criteria will be overall accuracy and recall. Accuracy is important as an overall tool to judge how well the algorithm is working and as the purpose is to identify the riskiest patients Recall is important as false positives are more acceptable than false negatives. The reasoning behind why false positives are more acceptable than false negatives is that additional care on a low risk patient is better than not enough care for high risk patient.

This analysis reviewed the results of three backpropagation Neural Networks model at three different parameters using the nn Neural Network algorithm in R from the library “Neuralnet”. The data was randomly sampled with 70% going to a training dataset, and the remaining 30% going to the testing dataset. Each model was trained on the same training set, and then used to predict discharge classifications for the same test dataset. The models were evaluated on their scores of Accuracy and Recall for their classifications of the Test data set. Each model was measured using a Confusion Matrix, with the Error Rate, Accuracy, Sensitivity, Specificity, Precision, and Recall calculated. Each Model also had the accuracy scored on the training data set compared against to the accuracy scored on the test data set to measure over-fitting. Additionally, the cross entropy error and total number of steps to build the model were also measured.

The first model was trained using all remaining variables in the dataset, Age, Sex, CPK, SHO, CHF, MIORD, MIQ and MINQ to classify discharge status. The model was predicted using the NN method from the NeuralNet R Library using default parameters with one hidden layer with two neurodes, the error calculated using cross-entropy, and linear output set to false. The second model was trained with the same parameters, with the exception of using two hidden layers with two neurodes in each layer and with the maximum number of steps permitted to train the network increased to 1,000,000 to permit the added complexity. The third model was trained with the same parameters as the second, with the exceptions that each of the two hidden layers had five neurodes each.

|  |  |  |  |
| --- | --- | --- | --- |
| Model 1 - Train Data - Confusion Matrix | | | |
|  | Predicted False | | Predicted Positive |
| Actual False | 268 | | 31 |
| Actual Positive | 0 | | 16 |
| Error Rate | Accuracy | | Sensitivity |
| 9.84% | 90.16% | | 100.00% |
| Specificity | Precision | | Recall |
| 94.37% | 34.04% | | 100.00% |
|  |  | |  |
| Model 1 - Test Data - Confusion Matrix | | | |
|  | | Predicted False | Predicted Positive |
| Actual False | | 131 | 30 |
| Actual Positive | | 0 | 5 |
| Error Rate | | Accuracy | Sensitivity |
| 18.07% | | 81.93% | 100.00% |
| Specificity | | Precision | Recall |
| 96.32% | | 14.29% | 100.00% |

**Results**

The first Neural Network model produced a back-propagation Neural Network with 8 input units, 1 hidden layer with 2 neurodes, and one output unit. A visualization of the Neural Network was also created (**Figure 2)**. The written weights and biases of the Neural Network are shown in **Appendix 1**. The Neural Network took 2507 steps to train, and had a cross-entropy error of 63.89 which shows that the Network was not able to find a perfect model for the training set and it trained fairly quickly, both positive signs of the model not extremely over-fitting. The Neural Network model had an accuracy of 90.6% and a recall of 100% when classifying the training dataset, and an accuracy of 81.93% and a recall of 100% when classifying the test data set **(Table 3)**. The accuracy decreased 8.67% from the training data to the test data indicating a modest amount of over-fitting, although recall stayed at 100% for each data set.

The second Neural Network model produced a backpropagation Neural Network with 8 input units, 2 hidden layer with 2 neurodes each, and one output unit. A visualization of the Neural Network was created (**Figure 2**) The written weights and biases of the Neural Network are shown in **Appendix 2**. The Neural Network took 44,649 steps to train, and had a cross-entropy error of 59.73 which shows that the model has some improvement compared to the first model in error, but took about 18 times as long to train as the first model. The Neural Network model had an accuracy of 92.38% and a recall of 81.08% when classifying the training dataset, and an accuracy of 71.85% and a recall of 82.47% when classifying the test data set **(Table 4)**. The accuracy decreased 20.53% from the training data to the test data indicating strong over-fitting, while recall was lower than the last model.

The third Neural Network model produced a backpropagation Neural Network with 8 input units, 2 hidden layer with 5 neurodes each, and one output unit. A visualization of the Neural Network was created (**Figure 4)**. The written weights and biases of the Neural Network are shown in **Appendix 3**. The Neural Network took 231,116 steps to train, and had a cross-entropy error of 0.004 which shows that the model has high potential for over-fitting with such a low error given the relatively small dataset used. The Neural Network model had an accuracy of 100% and a recall of 100% when classifying the training dataset, and an accuracy of 75.9% and a recall of 44.68% when classifying the test data set **(Table 5)**. The steps taken show that the model took 92 times as long to train as the initial model produced.

Of the three models, the first and simplest model had the greatest scores of accuracy and recall on the test data, while the other two models showed signs of overfitting. This appears to be due to the extra amount of hidden layer neurodes in the second and third models which allowed the model more opportunities to over-fit. The extra complexity of the third model allowed it to score a perfect 100% accuracy of the train data, but the performance of only 75.9% on the train data showed this was only due to over-fitting. We can see from this that the complexity of the neural network needs to be commensurate with the amount of data available to train it, with too little data and too much complexity leading to over-fitting and that for this dataset the simplest model was also the superior model.

**Conclusion**

This analysis showed that the backpropagation Neural Network algorithim combined with the variables provided could be used to build a model to classify patients into higher risk of death at discharge, and lower risk of death at discharge with high accuracy and high recall. This shows its potential suitability for the purpose of assisting in health care outcome predictions. Additionally, it shows the importance of using separate data to train backpropagation Neural Networks than the data used to evaluate its performance. The third model which had a 24.1% drop in accuracy from train data to test data shows the potential dangers of allowing a backpropagation model to over-fit without testing. This information could be useful in a strategy to increase the complexity of the neural network in-line with the amount of data provided which is applicable to any general application of the algorithm.

Future studies could improve on the limitations of this analysis. The dataset used was relatively small at only 481 observations. A higher number could yield much higher accuracy and allow for greater complexity in backpropagation neural network models. Second, the dataset is aged with the oldest data from 1975 and the most recent data from 1988. As health care would be expected to improve from this time, the data could be out of date and using more recent data to train could increase the model if it was used on patients today. Third, more variables related to the patients’ health and treatment could help train the network to be more accurate.

**References**

Han, Kamber, and Pei (2011). Data Mining: Concepts and Techniques, Third Edition Retrieved September 14, 2018 from http://hanj.cs.illinois.edu/cs412/bk3/01.pdf

Heart Disease Facts & Statistics. (n.d.). Retrieved from https://www.cdc.gov/heartdisease/facts.htm

Heart disease. (2018, March 22). Retrieved from https://www.mayoclinic.org/diseases-conditions/heart-disease/symptoms-causes/syc-20353118

Figure 1. Boxplot Distribution of Age of Patients

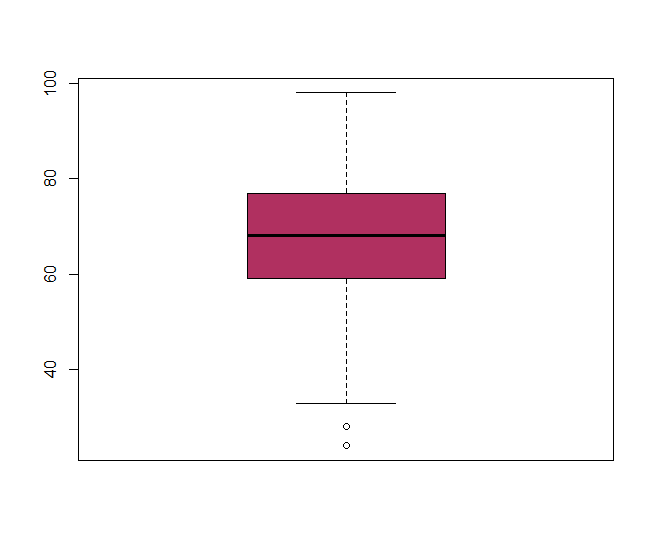


Figure 2. 1st Neural Network Model

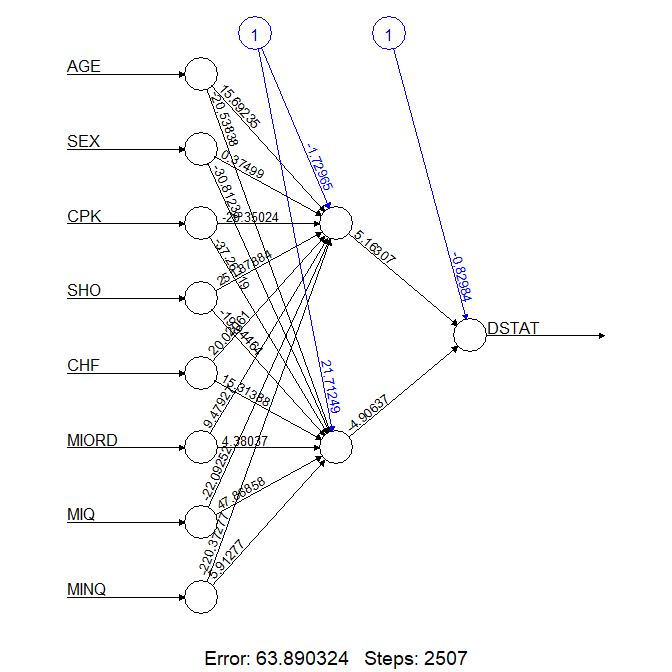


Figure 3. Second Neural Network Model

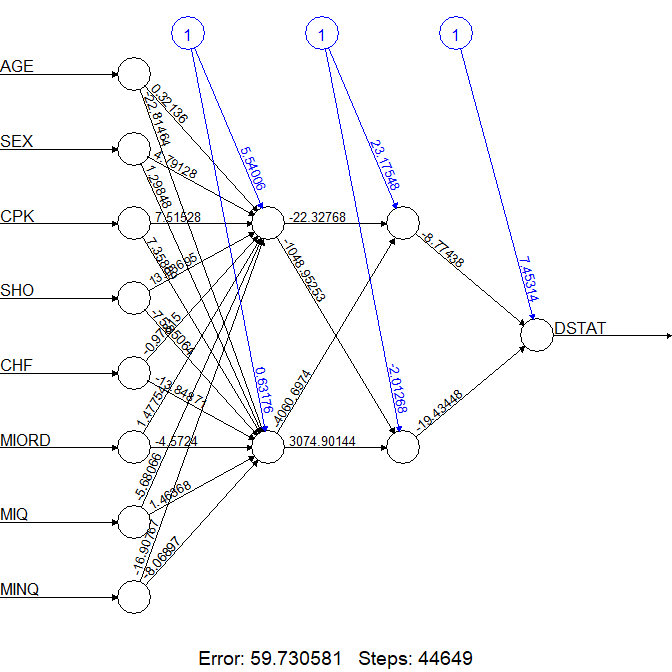


Table 4. Confusion Matrix of Test and Train Data for Model 2

|  |  |  |
| --- | --- | --- |
| Model 2 - Train Data - Confusion Matrix | | |
|  | Predicted False | Predicted Positive |
| Actual False | 261 | 17 |
| Actual Positive | 7 | 30 |
| Error Rate | Accuracy | Sensitivity |
| 7.62% | 92.38% | 81.08% |
| Specificity | Precision | Recall |
| 89.69% | 63.83% | 81.08% |
|  |  |  |
| Model 2 - Test Data - Confusion Matrix | | |
|  | Predicted False | Predicted Positive |
| Actual False | 57 | 51 |
| Actual Positive | 34 | 160 |
| Error Rate | Accuracy | Sensitivity |
| 28.15% | 71.85% | 82.47% |
| Specificity | Precision | Recall |
| 26.27% | 75.83% | 82.47% |

Figure 4. Third Neural Network Model

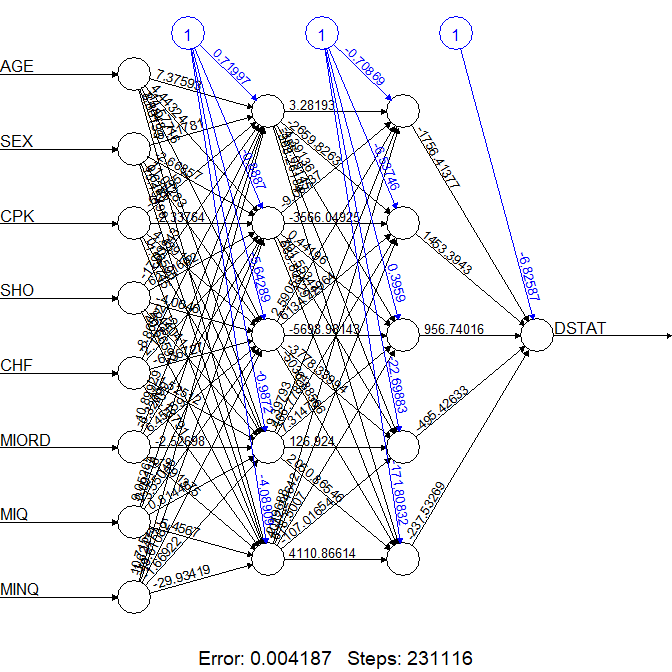


Table 5. Confusion Matrix of Test and Train Data for Model 3

|  |  |  |
| --- | --- | --- |
| Model 3 - Train Data - Confusion Matrix | | |
|  | Predicted False | Predicted Positive |
| Actual False | 268 | 0 |
| Actual Positive | 0 | 47 |
| Error Rate | Accuracy | Sensitivity |
| 0.00% | 100.00% | 100.00% |
| Specificity | Precision | Recall |
| 85.08% | 100.00% | 100.00% |
|  |  |  |
| Model 1 - Train Data - Confusion Matrix | | |
|  | Predicted False | Predicted Positive |
| Actual False | 105 | 14 |
| Actual Positive | 26 | 21 |
| Error Rate | Accuracy | Sensitivity |
| 24.10% | 75.90% | 44.68% |
| Specificity | Precision | Recall |
| 83.33% | 60.00% | 44.68% |

**Appendix 1**

**Neural Network 1 Weights and Biases**

1

error 63.890324370300

reached.threshold 0.009457098404

steps 2507.000000000000

Intercept.to.1layhid1 -1.729646418721

AGE.to.1layhid1 15.692349521669

SEX.to.1layhid1 0.374992647440

CPK.to.1layhid1 -29.350244611753

SHO.to.1layhid1 251.878840571050

CHF.to.1layhid1 20.020609965843

MIORD.to.1layhid1 9.479267702936

MIQ.to.1layhid1 -22.092524116347

MINQ.to.1layhid1 -220.372773073506

Intercept.to.1layhid2 21.712486722468

AGE.to.1layhid2 -20.538376295079

SEX.to.1layhid2 -30.812320724225

CPK.to.1layhid2 -37.261191375469

SHO.to.1layhid2 -19.944635323416

CHF.to.1layhid2 15.313881208702

MIORD.to.1layhid2 4.380374665105

MIQ.to.1layhid2 47.868582226283

MINQ.to.1layhid2 5.912774749330

Intercept.to.DSTAT -0.829841209808

1layhid.1.to.DSTAT 5.163065556501

1layhid.2.to.DSTAT -4.906374453087

**Appendix 2**

**Neural Network 2 Weights and Biases**

1

error 59.730581138728

reached.threshold 0.009734731335

steps 44649.000000000000

Intercept.to.1layhid1 5.540061626426

AGE.to.1layhid1 0.321359369014

SEX.to.1layhid1 4.791280340347

CPK.to.1layhid1 7.515278233035

SHO.to.1layhid1 13.036948434859

CHF.to.1layhid1 -0.978153576252

MIORD.to.1layhid1 1.477544270743

MIQ.to.1layhid1 -5.680662201539

MINQ.to.1layhid1 -16.907670997391

Intercept.to.1layhid2 0.631763863636

AGE.to.1layhid2 -22.814637386563

SEX.to.1layhid2 1.298484676030

CPK.to.1layhid2 7.358888301988

SHO.to.1layhid2 -75.850641623663

CHF.to.1layhid2 -13.848710782470

MIORD.to.1layhid2 -4.572396622598

MIQ.to.1layhid2 1.463684450821

MINQ.to.1layhid2 -8.068969538503

Intercept.to.2layhid1 23.175477199397

1layhid.1.to.2layhid1 -22.327679610032

1layhid.2.to.2layhid1 -4060.697402876528

Intercept.to.2layhid2 -2.012678355960

1layhid.1.to.2layhid2 -1048.952525379266

1layhid.2.to.2layhid2 3074.901435984182

Intercept.to.DSTAT 7.453139465585

2layhid.1.to.DSTAT -8.774379179346

2layhid.2.to.DSTAT -19.434480339319

**Appendix 3**

**Neural Network 3 Weights and Biases**

1

error 0.004186726013

reached.threshold 0.009323318676

steps 231116.000000000000

Intercept.to.1layhid1 0.719970863571

AGE.to.1layhid1 7.375932107434

SEX.to.1layhid1 -5.717813450573

CPK.to.1layhid1 -6.879511326331

SHO.to.1layhid1 -17.033430943243

CHF.to.1layhid1 -8.865622958065

MIORD.to.1layhid1 -10.899289034278

MIQ.to.1layhid1 9.052650404915

MINQ.to.1layhid1 10.715202341840

Intercept.to.1layhid2 -0.888696978482

AGE.to.1layhid2 4.443241807691

SEX.to.1layhid2 -2.668566924030

CPK.to.1layhid2 -2.337638179937

SHO.to.1layhid2 62.616019608882

CHF.to.1layhid2 2.178149607856

MIORD.to.1layhid2 -6.320992742299

MIQ.to.1layhid2 0.091175329033

MINQ.to.1layhid2 -1.624739358723

Intercept.to.1layhid3 -5.642890665092

AGE.to.1layhid3 -14.577159901685

SEX.to.1layhid3 11.012827317404

CPK.to.1layhid3 4.439187627951

SHO.to.1layhid3 -4.064602223481

CHF.to.1layhid3 -6.367267767492

MIORD.to.1layhid3 6.451586111733

MIQ.to.1layhid3 -3.550479003649

MINQ.to.1layhid3 -49.610816667963

Intercept.to.1layhid4 -0.987200274212

AGE.to.1layhid4 1.484795223054

SEX.to.1layhid4 0.628795149307

CPK.to.1layhid4 0.165948469860

SHO.to.1layhid4 2.440414184365

CHF.to.1layhid4 -1.525117482966

MIORD.to.1layhid4 -2.526977023641

MIQ.to.1layhid4 0.814476825521

MINQ.to.1layhid4 -1.669216375464

Intercept.to.1layhid5 -4.089093144227

AGE.to.1layhid5 8.801330911900

SEX.to.1layhid5 15.154987484908

CPK.to.1layhid5 -1.057450256126

SHO.to.1layhid5 20.665915955662

CHF.to.1layhid5 -22.187912763195

MIORD.to.1layhid5 -11.913746069955

MIQ.to.1layhid5 -5.456697419614

MINQ.to.1layhid5 -29.934194025578

Intercept.to.2layhid1 -0.708690180646

1layhid.1.to.2layhid1 3.281929965784

1layhid.2.to.2layhid1 -9.067369943894

1layhid.3.to.2layhid1 2.590532329436

1layhid.4.to.2layhid1 9.397934150993

1layhid.5.to.2layhid1 -2.796877248421

Intercept.to.2layhid2 -6.537464363540

1layhid.1.to.2layhid2 -2659.826299796217

1layhid.2.to.2layhid2 -3566.049245042169

1layhid.3.to.2layhid2 6134.229637354191

1layhid.4.to.2layhid2 466.778035060385

1layhid.5.to.2layhid2 4032.246422932941

Intercept.to.2layhid3 0.395901206973

1layhid.1.to.2layhid3 -4.391360248840

1layhid.2.to.2layhid3 0.441964310314

1layhid.3.to.2layhid3 -5698.981433450537

1layhid.4.to.2layhid3 7.314733156839

1layhid.5.to.2layhid3 678.500697744069

Intercept.to.2layhid4 -22.698832346620

1layhid.1.to.2layhid4 -355.091449443545

1layhid.2.to.2layhid4 201.553435200460

1layhid.3.to.2layhid4 -3778.339942028691

1layhid.4.to.2layhid4 126.923997887510

1layhid.5.to.2layhid4 -107.016538381068

Intercept.to.2layhid5 -171.808318296223

1layhid.1.to.2layhid5 -368.261844827189

1layhid.2.to.2layhid5 583.805289000348

1layhid.3.to.2layhid5 -5036.885663730584

1layhid.4.to.2layhid5 2050.865457662521

1layhid.5.to.2layhid5 4110.866141953700

Intercept.to.DSTAT -6.825872902152

2layhid.1.to.DSTAT -1756.413774541400

2layhid.2.to.DSTAT 1453.394298242335

2layhid.3.to.DSTAT 956.740155446259

2layhid.4.to.DSTAT -495.426325902766

2layhid.5.to.DSTAT -237.532693861361

**Appendix 4**

> # Assignment 4

> # by Kenneth Lulie, Data 630 - Ami Gates

> # Created 11/4

> #Work on 11/5 through 11/11

>

>

> #Standard introductory screenshots

> Sys.time()

[1] "2018-11-11 09:54:42 EST"

> Sys.info()

sysname release version nodename machine login user effective\_user

"Windows" ">= 8 x64" "build 9200" "LAPTOP-QHQ0RPOH" "x86-64" "Kenneth" "Kenneth" "Kenneth"

> R.version

\_

platform x86\_64-w64-mingw32

arch x86\_64

os mingw32

system x86\_64, mingw32

status

major 3

minor 5.1

year 2018

month 07

day 02

svn rev 74947

language R

version.string R version 3.5.1 (2018-07-02)

nickname Feather Spray

>

> #load the library to use later.

> library("neuralnet")

>

> #Set Working DIrecty

> setwd("D:/UMUC/630/Week 9/Assignment 4")

>

> #import data

> heart<-read.csv(file="whas1.csv", head=TRUE, sep=",")

>

>

>

>

>

>

>

> #Run the summary command

> summary(heart)

ID AGE SEX CPK SHO CHF MIORD MITYPE YEAR

Min. : 1 Min. :24.00 Min. :0.0000 Min. : 10.0 Min. :0.000 Min. :0.0000 Min. :0.0000 Min. :1.00 Min. :1.000

1st Qu.:121 1st Qu.:59.00 1st Qu.:0.0000 1st Qu.: 270.0 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:1.00 1st Qu.:2.000

Median :241 Median :68.00 Median :0.0000 Median : 587.0 Median :0.000 Median :0.0000 Median :0.0000 Median :1.00 Median :3.000

Mean :241 Mean :67.48 Mean :0.4033 Mean : 941.5 Mean :0.079 Mean :0.4075 Mean :0.3597 Mean :1.43 Mean :3.422

3rd Qu.:361 3rd Qu.:77.00 3rd Qu.:1.0000 3rd Qu.:1146.0 3rd Qu.:0.000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:2.00 3rd Qu.:5.000

Max. :481 Max. :98.00 Max. :1.0000 Max. :9000.0 Max. :1.000 Max. :1.0000 Max. :1.0000 Max. :3.00 Max. :6.000

YRGRP LENSTAY DSTAT LENFOL FSTAT

Min. :1.000 Min. : 1.00 Min. :0.0000 Min. : 1 Min. :0.0000

1st Qu.:1.000 1st Qu.: 8.00 1st Qu.:0.0000 1st Qu.: 150 1st Qu.:0.0000

Median :2.000 Median :12.00 Median :0.0000 Median :1420 Median :1.0000

Mean :1.971 Mean :13.86 Mean :0.1705 Mean :1735 Mean :0.5177

3rd Qu.:3.000 3rd Qu.:17.00 3rd Qu.:0.0000 3rd Qu.:2551 3rd Qu.:1.0000

Max. :3.000 Max. :71.00 Max. :1.0000 Max. :5843 Max. :1.0000

> #check also with str

> str(heart)

'data.frame': 481 obs. of 14 variables:

$ ID : int 1 2 3 4 5 6 7 8 9 10 ...

$ AGE : int 62 78 81 79 60 72 60 83 78 72 ...

$ SEX : int 1 1 1 1 1 0 1 1 0 1 ...

$ CPK : int 485 910 320 3290 2500 99 1200 160 66 99 ...

$ SHO : int 1 0 1 1 1 0 0 0 0 0 ...

$ CHF : int 1 1 1 1 1 0 0 0 1 0 ...

$ MIORD : int 0 1 0 1 1 0 0 0 1 0 ...

$ MITYPE : int 1 1 1 1 1 1 1 1 1 1 ...

$ YEAR : int 1 1 1 1 1 1 1 1 1 1 ...

$ YRGRP : int 1 1 1 1 1 1 1 1 1 1 ...

$ LENSTAY: int 1 1 1 1 2 2 2 3 3 3 ...

$ DSTAT : int 1 1 1 1 1 1 0 1 1 0 ...

$ LENFOL : int 1 1 1 1 2 2 2 3 3 5586 ...

$ FSTAT : int 1 1 1 1 1 1 1 1 1 0 ...

>

>

>

> #arrays start at 1 in python

> #Remove ID, unique identifier

> heart<-heart[2:14]

>

> ##Dstat is the initial discharge, LENFOL is the amount of time that passed from INITIAL admission to follow up admission.

> #Therefore, its a more rewarding question to ask if they will survive the initial admission

> #The purpose will therefore be to make a model that can help doctors on initial admission identify the

> #likely hood of death from initial discharge.

>

> #For this purpose, we will remove all data that doctors would not know after running tests on initial admissions.

> #Or would not be relevant.

> #Therefore we will remove, the length of the stay, the Length of follow up time, the follow up discharge (FSTAT)

> #and Year and YRGRP as we want this to be more useful on a present day basis.

> #This leaves AGE, SEX, CPK, SHO, CHF, MIORD,MITYPE to predict off of.

>

> #remove remaining variables

> heart <- subset(heart, select = -c(YEAR, YRGRP, LENSTAY, LENFOL, FSTAT))

>

>

> #Make boxplot

> boxplot(heart$AGE, col="maroon")

>

> #Scale down all numeric before running neural network

> #do not need to scale categorical

> heart$AGE<-scale(heart$AGE)

> heart$CPK<-scale(heart$CPK)

>

> #Scaling looks appropriate.

> summary(heart)

AGE.V1 SEX CPK.V1 SHO CHF MIORD MITYPE DSTAT

Min. :-3.429223 Min. :0.0000 Min. :-0.822857 Min. :0.000 Min. :0.0000 Min. :0.0000 Min. :1.00 Min. :0.0000

1st Qu.:-0.669089 1st Qu.:0.0000 1st Qu.:-0.593192 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:1.00 1st Qu.:0.0000

Median : 0.040660 Median :0.0000 Median :-0.313177 Median :0.000 Median :0.0000 Median :0.0000 Median :1.00 Median :0.0000

Mean : 0.000000 Mean :0.4033 Mean : 0.000000 Mean :0.079 Mean :0.4075 Mean :0.3597 Mean :1.43 Mean :0.1705

3rd Qu.: 0.750409 3rd Qu.:1.0000 3rd Qu.: 0.180603 3rd Qu.:0.000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:2.00 3rd Qu.:0.0000

Max. : 2.406490 Max. :1.0000 Max. : 7.118252 Max. :1.000 Max. :1.0000 Max. :1.0000 Max. :3.00 Max. :1.0000

>

> #However, MITYPE has 3 levels and is not ordinal.

> #1 is Q-Wave, 2 is not Q-wave, and 3 is indeterminate.

> #We will therefore make 2 new variables, QMITYPE and QMITYPEN. If both are 0, it indicates the third level.

> #We will then remove MITYPE

>

>

> #Create new variables, default 0

> heart$MIQ <- 0

> heart$MINQ <- 0

>

> #assign values to these new variables based on MITYPE

> heart$MIQ[heart$MITYPE == 1] <- 1

> heart$MINQ[heart$MITYPE == 2] <- 1

>

> #Remove MITYPE

> heart <- subset(heart, select = -c(MITYPE))

>

>

>

>

> #add DSTAT2 to end, remove DSTAT, change DSTAT2 to DSTAT

> #to make the other formulas that rely on it being the end work.

> heart$DSTAT2 <-heart$DSTAT

> heart <- subset(heart, select = -c(DSTAT))

> heart$DSTAT <-heart$DSTAT2

> heart <- subset(heart, select = -c(DSTAT2))

>

>

>

> #make sure that the result is reproducible

> set.seed(12345)

> #split the data into a training and test set

> ind <- sample(2, nrow(heart), replace = TRUE, prob = c(0.7, 0.3))

> train.data <- heart[ind == 1, ]

> test.data <- heart[ind == 2, ]

>

>

> ##### Start of NN 1

>

>

>

> #Build the model.

> ##1 hidden layer, with 2 nodes.

> nn<-neuralnet(formula = DSTAT~AGE+SEX+CPK+SHO+CHF+MIORD+MIQ+MINQ,

+ data = train.data, hidden=2, err.fct="ce", linear.output = FALSE)

>

>

> #names command displays the available neural network properties

> names(nn)

[1] "call" "response" "covariate" "model.list" "err.fct" "act.fct"

[7] "linear.output" "data" "net.result" "weights" "startweights" "generalized.weights"

[13] "result.matrix"

>

> #a matrix containing the reached threshold, needed steps,

> #error, AIC and BIC (if computed) and weights for every repetition. Each heart represents one repetition.

> nn$result.matrix

1

error 63.890324370300

reached.threshold 0.009457098404

steps 2507.000000000000

Intercept.to.1layhid1 -1.729646418721

AGE.to.1layhid1 15.692349521669

SEX.to.1layhid1 0.374992647440

CPK.to.1layhid1 -29.350244611753

SHO.to.1layhid1 251.878840571050

CHF.to.1layhid1 20.020609965843

MIORD.to.1layhid1 9.479267702936

MIQ.to.1layhid1 -22.092524116347

MINQ.to.1layhid1 -220.372773073506

Intercept.to.1layhid2 21.712486722468

AGE.to.1layhid2 -20.538376295079

SEX.to.1layhid2 -30.812320724225

CPK.to.1layhid2 -37.261191375469

SHO.to.1layhid2 -19.944635323416

CHF.to.1layhid2 15.313881208702

MIORD.to.1layhid2 4.380374665105

MIQ.to.1layhid2 47.868582226283

MINQ.to.1layhid2 5.912774749330

Intercept.to.DSTAT -0.829841209808

1layhid.1.to.DSTAT 5.163065556501

1layhid.2.to.DSTAT -4.906374453087

>

>

> #Plot showing the NN

> plot(nn)

>

> #Compute values against train data and round

> mypredict1 <- compute(nn, train.data[, 1:8])$net.result

> mypredict1 <-apply(mypredict1, c(1), round)

>

> #Build the confusion matrix

> table(mypredict1, train.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 268 31

1 0 16

> #Accuracy

> mean(mypredict1==train.data$DSTAT)

[1] 0.9015873016

> #90.158

> # Actual

> #Predicted 0 1

> # 0 268 31

> # 1 0 16

>

>

>

> ##Run on the test data, use all variables except the dependent.

> testPred1 <- compute(nn, test.data[, 1:8])$net.result

> testPred1<-apply(testPred1, c(1), round)

> #make confusion matrix

> table(testPred1, test.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 131 30

1 0 5

> #check accuracy

> mean(testPred1==test.data$DSTAT)

[1] 0.8192771084

> #test accuracy, 84.337

> # Actual

> #Predicted 0 1

> # 0 129 24

> # 1 2 11

>

>

>

>

>

>

>

>

> ##### Start of NN 2

>

>

>

> #Build the model.

> #using 2 hidden layers, with 3 nodes each.

> nn2<-neuralnet(formula = DSTAT~AGE+SEX+CPK+SHO+CHF+MIORD+MIQ+MINQ,

+ data = train.data, hidden=c(2,2), err.fct="ce", linear.output = FALSE, stepmax = 1000000)

> ## will take a while

>

>

> #make plot of second model.

> plot(nn2)

>

>

>

> #Run on train data

> mypredict2<-compute(nn2, nn$covariate)$net.result

> mypredict2<-apply(mypredict2, c(1), round)

>

> #Build the confusion matrix

> table(mypredict2, train.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 261 17

1 7 30

> mean(mypredict2==train.data$DSTAT)

[1] 0.9238095238

> #92.3

> # Actual

> #Predicted 0 1

> # 0 261 17

> # 1 7 30

>

>

>

> #Run again on test data.

> testPred2 <- compute(nn2, test.data[, 1:8])$net.result

> testPred2<-apply(testPred2, c(1), round)

> #Make table.

> table(testPred2, test.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 121 16

1 10 19

> #Check accuracy

> mean(testPred2==test.data$DSTAT)

[1] 0.843373494

> #.84337

>

> # Actual

> #Predicted 0 1

> # 0 121 16

> # 1 10 19

>

>

>

>

>

> ####### Model 3

>

>

> #Build the model.

> #using 2 hidden layers, with 5 nodes each.

> ## We will remove

> nn3<-neuralnet(formula = DSTAT~AGE+SEX+CPK+SHO+CHF+MIORD+MIQ+MINQ,

+ data = train.data, hidden=c(5 , 5), err.fct="ce", linear.output = FALSE, stepmax = 1000000)

> ### will take a long to run, aprox 5 minutes on my machine

>

>

> #make plot of second model.

> plot(nn3)

>

>

> #Run on train data

> mypredict3<-compute(nn3, nn$covariate)$net.result

> mypredict3<-apply(mypredict3, c(1), round)

>

> #Build the confusion matrix

> table(mypredict3, train.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 268 0

1 0 47

> mean(mypredict3==train.data$DSTAT)

[1] 1

>

> #1

> #Actual

> #Predicted 0 1

> # 0 268 0

> # 1 0 47

>

>

>

> #Run again on test data.

> testPred3 <- compute(nn3, test.data[, 1:8])$net.result

> testPred3<-apply(testPred3, c(1), round)

> #Make table.

> table(testPred3, test.data$DSTAT, dnn =c("Predicted", "Actual"))

Actual

Predicted 0 1

0 105 14

1 26 21

> #Check accuracy

> mean(testPred3==test.data$DSTAT)

[1] 0.7590361446

> #.819

>

>

> # Actual

> #Predicted 0 1

> # 0 113 12

> # 1 18 23