Slide 1

The objective of this practicum was to find a data source and create a Python or R script that analyzes the data in the data source. Although this is still the objective, I’ve gone a bit further, actually much further and developed an Artificial Neural Network commonly referred to as ANN.

The fascinating world of ANN is remarkable and amazing how it tries to mimic our brains in some ways. As we progress thru this presentation, I will show how the ANN compares to the human brain and discuss how technology mimics the neurons that our brains use every second of our lives.

This practicum was inspired by Udemy’s course called, “Deep Learning A-Z™: Hands-On Artificial Neural Networks.

My data set is from Kaggle and is a collection of 303 patients with and without Heart Disease. In this I intend to show you how to predict someone with Heart Disease as well as what the data tells us about heart disease.

Slide 2

##

What is an Artificial Neural Network?

First, we need to understand a little about our brains and the neural networks our brains use for thought, memory, emotion, and core activities such as heart beats, breathing etc.

##

Lets start with our brain

##

“The brain is one of the largest and most complex organs in the human body. It is made up of more than 100 billion nerves that communicate in trillions of connectiions called synapses…” from WebMD

##

Each of the 100 billion nerves are called Neurons, the core of what makes our brains work. And each of these neurons are connected via dendrites and axons.

Slide 3

It is important to understand the components of a neural network and how they relate to an artificial neural network. Let’s start with the Neuron.

##

A Neuron is?

**Neurons** (also known as neurones, nerve cells and nerve fibers) are electrically excitable cells in the nervous system that **function** to process and transmit information. In vertebrate animals, **neurons** are the **core** components of the brain, spinal cord and peripheral nerves. [*Neuron - ScienceDaily  
https://www.sciencedaily.com/terms/neuron.htm*](https://www.sciencedaily.com/terms/neuron.htm)

##

Slide 4

Neuron composition

##

This is a simplified image of what a Neuron is composed of. We are only concerned with the four main elements of a Neuron. They are the Soma (Cell Body), Axon (Transmitter), Dendrite (Receiver) and Synaptic Terminal (Synapse)

##

1. Cell Body or **Soma** is the nucleus of a [neuron](http://www.newworldencyclopedia.org/entry/Neuron) (nerve cell) from which one or more processes are received from or transmitted to ([dendrites](http://www.newworldencyclopedia.org/entry/Dendrite) and/or [axons](http://www.newworldencyclopedia.org/entry/Axon))

##

1. Axon is the Transmitter to other Neurons. It is much longer than Dendrites, by up to a 1000x longer is some cases.

##

1. Dendrites are the Receivers and mostly connect to Axons.

##

1. Synaptic terminals are where the Axon and Dendrite connect. More commonly known as Synapse

So, what does all the mean and what does it have to do with Artificial Neural Networks?

Slide 5

##

“Each individual neuron can form thousands of links with other neurons in this way, giving a typical brain well over 100 trillion synapses (up to 1,000 trillion, by some estimates).”

##

“Functionally related neurons connect to each other to form **neural networks** (also known as **neural nets** or **assemblies**). The connections between neurons are not static, though, they change over time. The more signals sent between two neurons, the stronger the connection grows…”

##

As we learn via inputs, the connections grow stronger and we retain more.

##

Since connections are not static and can change we now know the brain can be rewired due to brain injury or brain diseases.

Slide 6

##

Applying what we’ve learned

The discussions about Brain Neurons and how the brain learns so far is at the 40000 foot level if not greater. But the principles between Biology and an Artificial Neural Network are very similar. The following slides will help correlate how the Artificial Neural Network is similar to Neurons functioning in the Brain.

##

Input – This is simply some input where thru eyes, ears, smell, taste etc. This is sent to the Neuron for processing via a synapse connected to a Dendrite.

Neuron – Processes the Input and sends an Output via a Axon

Axon – Connects to other Neurons via the synapse becoming an Input to another Neuron.

The more often the input is processes by the Neuron the greater the Synapse connection and the greater learning and/or retention.

##

Slide 7

Lets look at a simplified ANN or Perceptron.

##

Here is the input layer

##

The hidden layer or neuron which sums up each row and applies a weight. Within this layer there are four types of Optimizers but we will only use the Rectifier

##

Finally the Output, it also uses an optimizer but since we are working within probabilities we will want to use a sigmoid optimizer

##

No we can see the inputs going into the Neuron with associated weights. When the ANN first starts it will automatically add weights to each input closest to zero.

##

This is called forward propagation

##

Once the dataset has gone thru and produced an output the ANN will update the Weights and go thru the dataset once again. The objective is to get the cost function close to zero. Each iteration is called an Epoch.

Slide 8

Let’s go a little bigger and more complicated than just one perceptron

##

Here we have our input layer once again.

##

Each input is a single row in the data set

Xx equal the columns in the row and represent the Independent Variable

Each Xx needs to be standardized for summation. This means that we need the values in each column to be within the same spectrum. For example Age is up to 100 but Net worth could be millions. We need this two numbers to be more standardized. Python has a special function called StandardScaler which does this for us.

##

Next we have our hidden layer (Neuron) where the values and weights are multiplied then summed.

##

The weights are initialized for the first synapse.

##

The ANN is smart enough to determine what inputs should be matched and fed to the Neuron. This maximizes the system so that it is not running calculations not required.

##

##

Nx is the weighted sum of the values in the row with the corresponding weight.

##

The neuron sends the results to the output where it is compared to the Actual and a cost function is generated. If the output is close to the actual then the learning is complete else the weights are updated thru back propagation. Each iteration of the data is an Epoch

Slide 9

##

Back propagation

I know I’ve already discussed back propagation in the previous slide however it is the crux of the effectiveness of an ANN

##

It is the most important step

##

It updates the weights based on the cost function

##

The cost function is ½ times the difference between the output and the actual squared.

##

The ANN back propogates until the Cost is mjnimized based on the number of Epoch you’ve told the system to run

##

Its ok to get close to 0 on the cost function but it should never be 100%

##

Slide 10

When creating our ANN the programming should follow certain steps.

##

Set the Weights to near zero

##

Load the data set and prepare it by ensuring it has been scaled, there are no nulls and all categories have been converted to numeric for summing

##

Forward propagate thru output

##

Run the cost function for accuracy

##

Back propagate by updating the Weights

##

Forward propagate

##

Continue based on number of Epoch chosen

##

Slide 11

##

I’m using Python spyder as my editor

##

Due to the mathematics involved a system with a high end GPU is preferable but you can use just you system CPU

##

The following are libraries specific to Machine Learning and are required to run ANN

Theano

##

Tensorflow – This is the heavy math lifter and manages all the computations

##

Keras library is the conduit that allows you to build the ANN

##

Lets get started building our ANN. Most of the material I’m presenting is based on the learning videos from Udemy – Deep Learning A-Z: Hands-On Artificial Neuron Network.

##

First we load the required libraries

##

We import the data and assign the X and y values. The X is the independent variables and y is the actual output or dependent variable

##

This may seem backwards but we need to establish the Train and test subsets of our data.

##

Now we can apply the scaler function that scales the data in both the Train and Test sets.

Slide 13

Here is the script for loading the data

##

Slide 14

##

There are 165 cases of Heart Disease and 138 with no Heart Disease

##

Selecting Head shows the first part of our data

##

Slide 15

##

The Train and Test datasets should only contain 12 columns out of the 13. The 13 is the actual column.

##

There should be 227 rows in Train. This is based on the total number of rows in the dataset (303) with a 75% going to Train

##

There should be 76 rows in Test. This is based on the total number of rows in the dataset (303) with a 25% going to Test

##

Slide 16

This defines what the columns represent and what the ANN is using to predict whether someone will have Heart Disease

##

Slide 17

Now, lets go build our ANN

##

First we need to load Keras to begin building the ANN

##

Next we need to call the ANN something so I came up with an awesome name…we’ll call it ANN.

##

Remember, an ANN consists of an Input layer, Hidden layer and output layer. We can create the Input and Hidden layer with one line of code. The units = 6 is simply taking the number of independent variables and dividing by 2 (this is binary). The Kernal\_initializer is set to Uniform and the calculation inside of the Neuron is to be set to Rectifier which uses the relu name and lastly set the input\_dim to the number of columns being processed. We can go into great discussions surrounding relu but according to the experts it is the best,

##

Slide 18

Continuing to build our ANN we need to add the Output Layer

##

You’ll notice this is very similar to the Input/Hidden layer from the previous slide. The only change is the units = 1 and the activation = Sigmoid. Again, we want the output to be probabilistic so we use sigmoid.

##

The ANN needs to be compiled in order to run. This is the most critical code of the ANN. It is where we define the type of algorithm to use, in this case we are using adam, the size of our batches and the number of epochs.

Optimizer: Adam stands for **Adaptive Moment Estimation.** Adaptive Moment Estimation (Adam) is another method that computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients like **AdaDelta** ,**Adam** also keeps an exponentially decaying average of past gradients M**(t)**, similar to momentum:

From towardsdatascience.com, “types of Optimization Algorithms used in Neural Networks and Ways to Optimize Gradien Descent”, anish singh walia June 10, 2017

https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f

##

Now the fun part, lets start training

Batches: If you have a large data set it is advisable to break it into manageable batches. In our case the data is small so I chose 1

NB\_Epochs: This is really dependent on you. If you are not getting the accuracy you expect then you can increase the number of times the data set is iterated thru. In our case I chose 25 because the data is small.

##

Slide 19

##

We now have a fully function ANN that has an accuracy of approximately 86%. The ANN ran 227 rows 25 times increase the accuracy as it went.

##

Slide 20

##

Lets see what we have created. Using a confusion\_matrix we can see from the test data the accuracy of the ANN

##

Here are the results: 24 No Heart Disease with 9 being wrong and 38 with Heart Disease and 5 being classified wrong. Adding them all up totals to the 76 test rows.

##

Slide 21

We can now add two new patients to see what the predicted results will be.

##

Here is the code to our new prediction based on our ANN

##

Here are the results

##

Slide 22

Now that we can predict if someone has the potential for heart disease we can now use graphs to plot what contributes to heart disease.

##

In this graph we see that heart disease is prevalent in the 55 to 60 age group

##

But in this graph we see that Heart Disease actually starts in the 40 to 50 range but the majority are in 50 to 60.

##

Slide 23

##

Cholesterol – Interesting slide, people with cholesterol levels 200 to 250 are more likely to have heart disease than those with higher cholesterol.

Men vs Women – here we see that men our more likely to have heart disease between 45 and 63 where women are more likely between 45 and 55.

##

Slide 24

##

Chest Pain when at a 2 is more likely to have Heart Disease than someone with 3. Its just enough to be uncomfortable. Where as 3 is likely a heart attack.

##

Thalassemia (thal-uh-SEE-me-uh) is an inherited blood disorder characterized by less hemoglobin and fewer red blood cells in your body than normal. Hemoglobin is the substance in your red blood cells that allows them to carry oxygen. The low hemoglobin and fewer red blood cells of thalassemia may cause anemia, leaving you fatigued.

Definition from the mayoclinic website

https://www.mayoclinic.org/diseases-conditions/thalassemia/symptoms-causes/syc-20354995

People with a 2 have the highest rate of heart disease.

##

Slide 25

Conclusion

I hope you have enjoyed not just learning about Artificial Neural Networks but Heart Disease as well. I have thoroughly enjoyed learning about both topics and know this will be of benefit to all.

Thank you for watching this slide presentation.