

*Department of Electrical and Electronics Lebanese University (ULFG III)*

**Prepared by: Raneem Jaafar 5602**

**Hassan Jawad 5264**

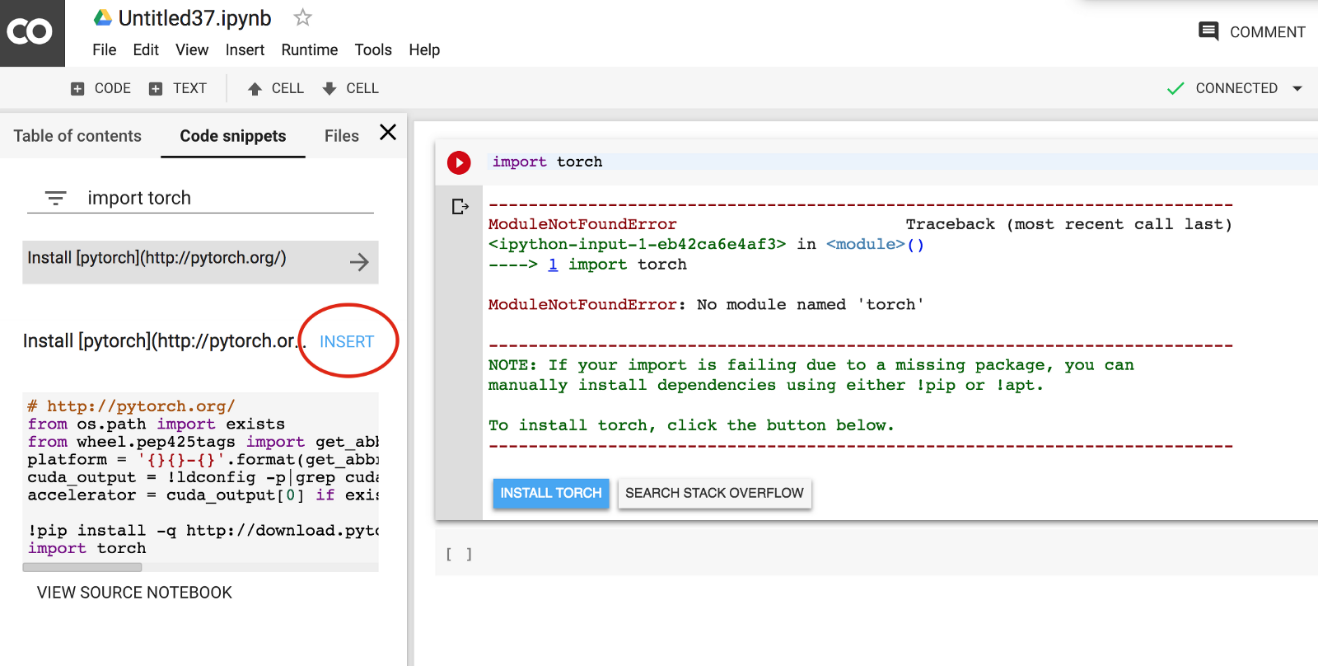
**Presented to: Dr. Mohammad Aoude**

*Submitted 22 July*

***Table of content***:

* Google colab------------------------------------------------------3
* Machine learning ------------------------------------------------4
* Neural network ---------------------------------------------------4
* Our project --------------------------------------------------------6
* Appendix -----------------------------------------------------------8

***What is Google colab?***

****Google Colab is a platform that allows you to run code directly on the cloud, this means that you can use very powerful hardware to run your code and the only requirement to do it is to have a Google account. In Google Colab you can only use **Python** as a programming language but this is fairly enough for the features it offers. Every line of code you write is automatically saved on your **Google Drive** storage and you can easily access your project notebook whenever you want, wherever you are.

***What is machine learning?***

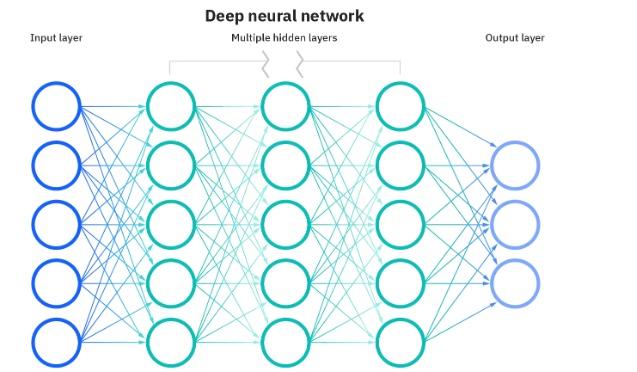
Machine Learning is a way of taking data and turning it into insights. We use computer power to analyze examples from the past to build a model that can predict the result for new examples.

We encounter machine learning models every day. For example, when Netflix recommends a show to you, they used a model based on what you and other users have watched to predict what you would like. When Amazon chooses a price for an item, they use a model based on how similar items have sold in the past. When your credit card company calls you because of suspicious activity, they use a model based on your past activity to recognize anomalous behavior.

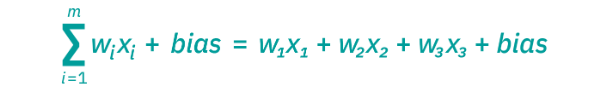
***Neural Network:***

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of [machine learning](https://www.ibm.com/cloud/learn/machine-learning) and are at the heart of [deep learning](https://www.ibm.com/cloud/learn/deep-learning) algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

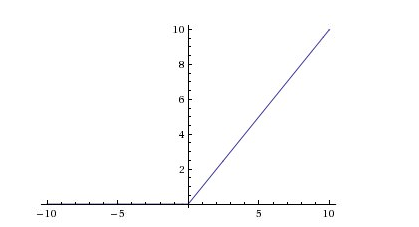
Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and [artificial intelligence](https://www.ibm.com/cloud/learn/what-is-artificial-intelligence), allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google’s search algorithm.



***How does neural network works?***

Think of each individual node as its own [linear regression](https://www.ibm.com/analytics/learn/linear-regression) model, composed of input data, weights, a bias (or threshold), and an output. The formula would look something like this:

Activision function = a = ∑wixi + bias = w1x1 + w2x2 + w3x3 + bias



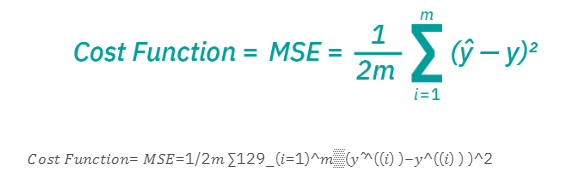
Output = RelU(a)= max(0,a)

output = f(x) = 1 if ∑w1x1 + b>= 0; a if ∑w1x1 + b < 0

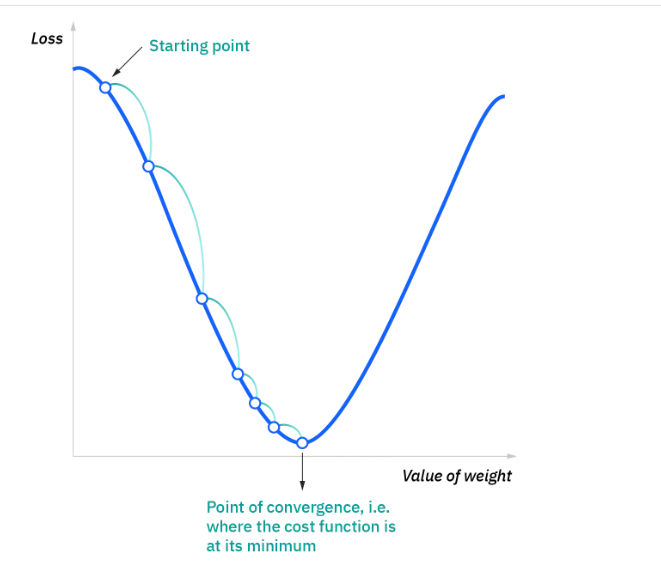
Once an input layer is determined, weights are assigned. These weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, it “fires” (or activates) the node, passing data to the next layer in the network. This results in the output of one node becoming in the input of the next node. This process of passing data from one layer to the next layer defines this neural network as a feed forward network.

As we start to think about more practical use cases for neural networks, like image recognition or classification, we’ll leverage supervised learning, or labeled datasets, to train the algorithm. As we train the model, we’ll want to evaluate its accuracy using a cost (or loss) function. This is also commonly referred to as the mean squared error (MSE). In the equation below,

* *i*represents the index of the sample,
* y-hat is the predicted outcome,
* y is the actual value, and
* *m* is the number of samples



𝐶𝑜𝑠𝑡 𝐹𝑢𝑛𝑐𝑡𝑖𝑜𝑛= 𝑀𝑆𝐸=1/2𝑚 ∑129\_(𝑖=1)^𝑚▒(𝑦 ̂^((𝑖) )−𝑦^((𝑖) ) )^2

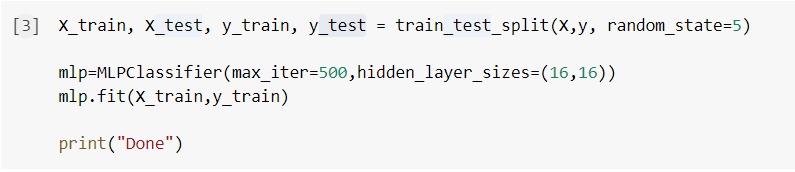
Ultimately, the goal is to minimize our cost function to ensure correctness of fit for any given observation. As the model adjusts its weights and bias, it uses the cost function and reinforcement learning to reach the point of convergence, or the local minimum. The process in which the algorithm adjusts its weights is through gradient descent, allowing the model to determine the direction to take to reduce errors (or minimize the cost function). With each training example, the parameters of the model adjust to gradually converge at the minimum.

Most deep neural networks are feed forward, meaning they flow in one direction only, from input to output. However, you can also train your model through back propagation; that is, move in the opposite direction from output to input. Back propagation allows us to calculate and attribute the error associated with each neuron, allowing us to adjust and fit the parameters of the model(s) appropriately.

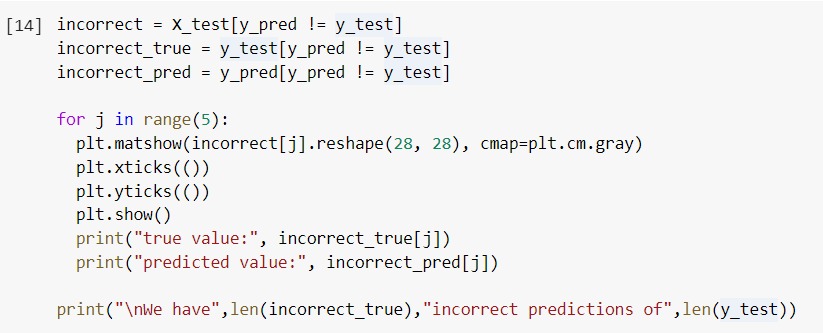
***Our project:***

Our project takes an image of a handwritten digit as an input and it detect which digit is it.

The data (70 000 data point) set is made of images, every picture is 28x28 pixel (784 pixel), and every pixel has a number **[from 0 to 255 where 0== black, 255== white].**

****The data set is divided into two categories:

Testing is 25% of the data set (17500 data point for testing)

****Training is 75% of the data set (52500 data point for training)

This model uses Neural Network for prediction.

Training set is used to train the model while testing set to see the performance of this model.

The accuracy function determines the percentage of correct prediction.

When we calculate the accuracy we see that its not 100% so there is some incorrect predictions but their percentage is low.

We plotted the pixels of some incorrect predictions that is the image and the true value with the predicted value that is incorrect.

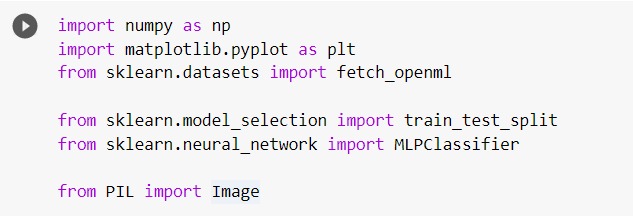
Furthermore, we added the ability for the user to give the model a picture as an input.

If the picture is not a grayscale, it will be converted to grayscale and if it is not 28x28 pixels it is also converted.

Once done, the image will be ready and fed to the model for detecting the value of the digit in it.

This model is trained on white digits written on black background, so we took a step forward and we added a piece of code to convert the white background written digits into white digits on a black background just like the images that this model is trained for.

***Appendix:***



* Matplotib : to plot the image
* Sklearn.datasets : took the dataset from it
* Numpy: provides a multidimensional array object