Introduction to Neural Networks

Dr. Osita Onyejekwe:

Logistic Regression : Classification algorithm :

Quick Intro on deep learning : Subset of Machine learning
: Computer Vision
Uses : : Natural Language Processing
: Speech Recognition

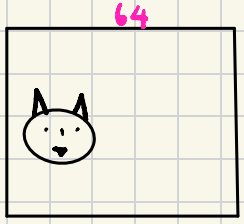
Deep learning is extremely computationally expensive.
- People have had to find techniques to parallelize the code.
- Data available has been growing exponentially since the internet bubble.
- This type of algorithm is able to learn when there is a plethora of data.
  - models are very flexible.
  - The more data available the better the algorithm can learn salient features within the data.
- Algorithms:

(I) Logistic Regression

Goal: Find Cats in images. (Binary classification)

If there is a cat in the image, we want to output a number close to 1
If there is no cat in the image, we want to output 0.

For now let us just assume that we are dealing with just one cat in the image and no more.



Note that in computer Science, images can be documented/represented as 3-D matrices.

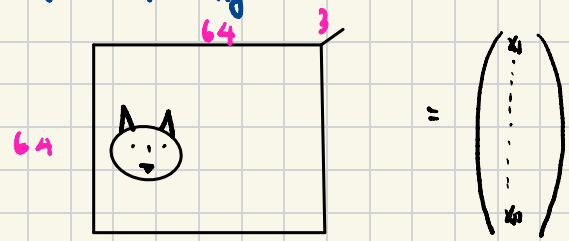
- Assuming the image on the left is a color image of size 64 x 64, we have 64 x 64 x 3 numbers to represent those pixels.

3 ← RGB Channel

- Every pixel in an image can be represented by 3 numbers.

- One representing the red filter, one representing the blue filter and another representing the green filter.

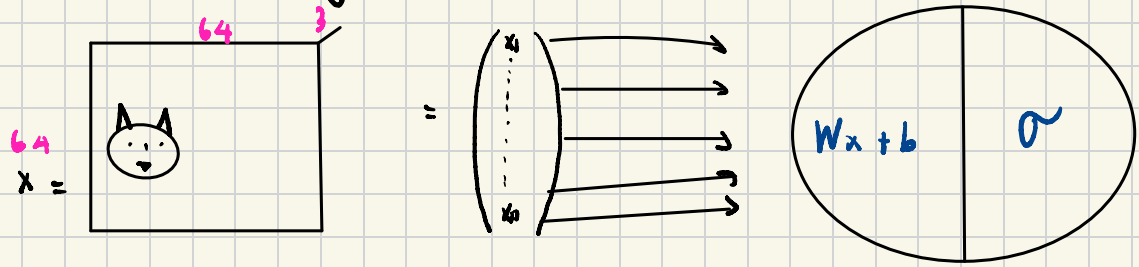
- We will now flatten this image into a vector before using logistic regression to see if there is a cat in the image.



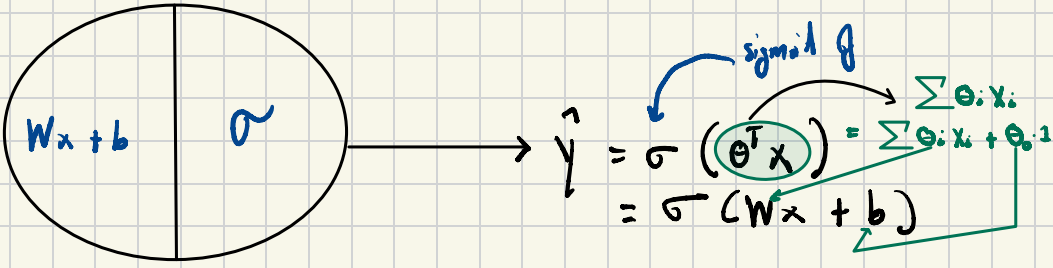
∴ taking all the numbers in this matrix and flatten it into a vector.

- We can now use our logistic regression because we have a vector input.

- We will now take the vectors and push them into an operation known as the logistic operation, which has one part which is $Wx + b$, where x is the image, and the second part is going to be the sigmoid.



- The sigmoid function is a function that takes a number between $-\infty$ and ∞, and maps it between 0 & 1. Very common for a classification problem.



Question: What is the shape of the matrix W?

Answer: $\hat{y} = \sigma(Wx + b)$

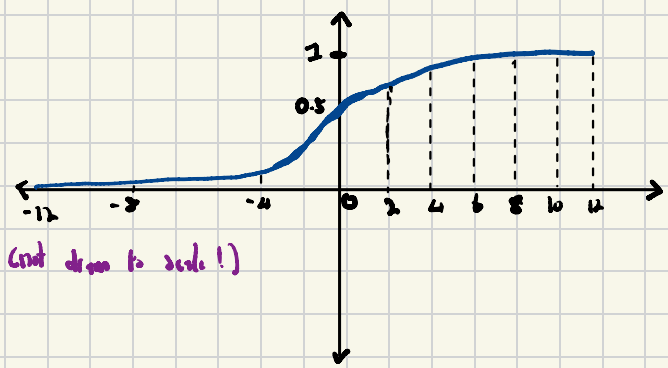
- $\hat{y}$: (1×1) or $(1,1)$
- W: $(1, 12288)$
- x: $(64 \times 64 \times 3, 1)$ $= 12,288$

∴ We have a row vector as our parameter.

Once we have the model we need to train it. The process of training is that

i) initialize parameters w, b (weights, bias) (find the right parameters)
ii) Then optimize them. Find the optimal w, b.
iii) After finding the optimal $\{w, b\}$, we them to predict.
$\Rightarrow \hat{y} = \sigma(wx + b)$ to predict.

Recall that this is a sigmoid function:



(Click down to scale!)

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$= \frac{e^{x}}{1 + e^{x}}$$

$$= 1 - \sigma(-x)$$

- Note that finding the optimal W means defining the loss function, which is the objective.
- In machine learning, it is often the case such that whereby you have a function that you know you want to find (actually effective) but you do not know the value of its parameters. In order to find them

you are going to use a proxy, that is going to be your loss function. If you manage to minimize the loss function you will find the right parameters.

∴ We define a loss function that is the logistic loss as

minimize

$$\mathcal{L} = -[\, y \log \hat{y} + (1-y) \log(1-\hat{y}) \,]$$

originating from Maximum Likelihood Estimation, starting from a probabilistic model.

∴ The idea is now how do we minimize this function, using a gradient descent algorithm. ⇒ iteratively compute the derivative of the loss function with respect to our parameters.

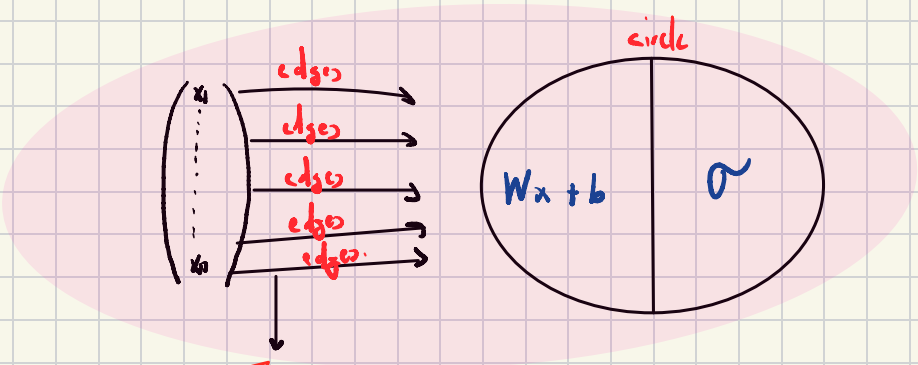
$$\begin{cases} w = w - \alpha \dfrac{\delta \mathcal{L}}{\delta w} \\ b = b - \alpha \dfrac{\delta \mathcal{L}}{\delta b} \end{cases}$$

- At every step we will update them to make this loss function go a little smaller.
- At every point you compute the derivative of the loss with respect to your parameters.
- Note that : $\sigma' = \sigma(1-\sigma)$

Q: How many parameters does this logistic Regression have?

A: 12,288 weights + 1 bias = 12,289 parameters.

Note that every circle has a bias & every edge has a weight.

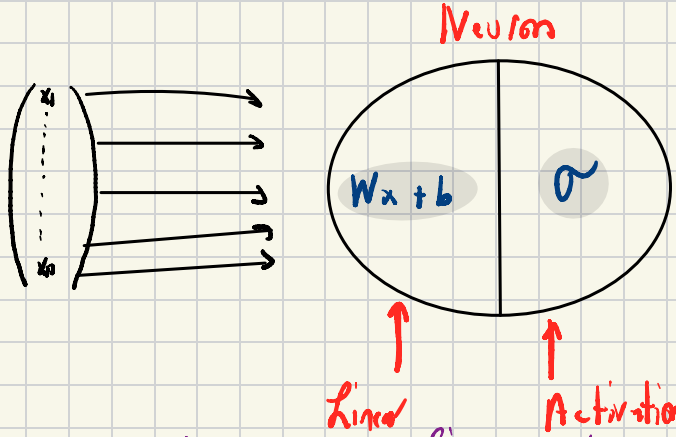


$\Rightarrow$ Every weight corresponds to an edge.

2 Important Equations to know:

① **Neuron = Linear + Activation**

(A neuron is an operation that has 2 parts, 1) Linear 2) Activation)



Take the output of the linear part & put it into the activation (Sigmoid) function in this case however it can be other functions).

② **Model = architecture + parameters.**

architecture: one neuron neural network

parameters: w, b

- Here we are trying to train a logistic Regression in order to be able to use it

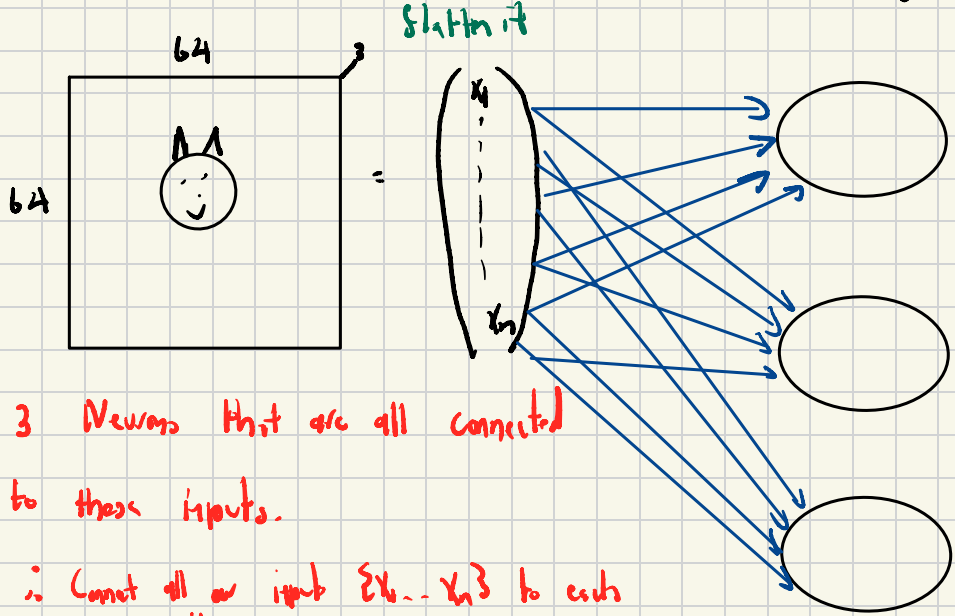
When people say "we have skipped a model" in industry, what they are saying is that we have found the right parameters with the right architecture

For this series, we will modify the architecture extensively (adding neurons all over) and the parameters will always be called w & b. They will become bigger and bigger because we have more data. We will need more parameters to understand what a cat is.

Goal 2: Find Cat/Lion/Iguana in image

3 - types of animals

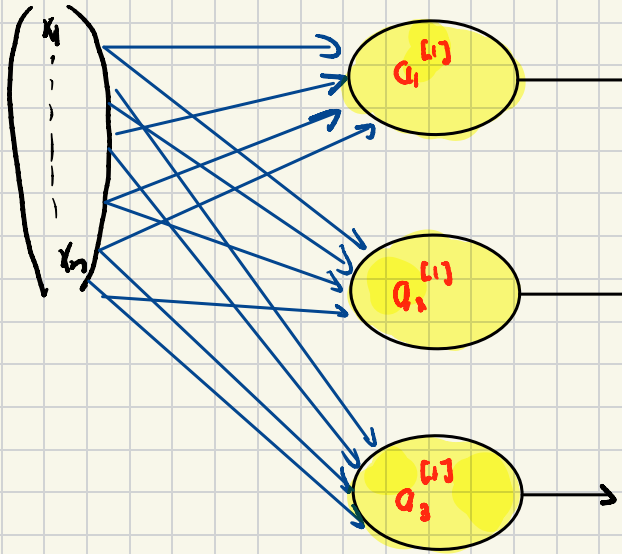
⇒ Put 2 more circles/neurons & do the same thing.



3 Neurons that are all connected to these inputs.

∴ Connect all our inputs $\{x_1 \ldots x_{3}\}$ to each one of these neurons.

We will use a specific set of Notation here:



- The square brackets represent what we will call the layer. If you look at this network, it looks like there is just one layer currently present. In this one layer, the neurons do not communicate with each other.
- The index that is the subscript to the "a" is the number identifying the neuron inside the layer.
- Then we have $\hat{y}$, and instead of being a single number as it was before, is know a vector of size 3.

Question: How many parameters does this network have?
Answer: 3 x what we had before because we added 2 more Neurons.

Note that we are dealing with Binary Classifications with images and Kebabs.

- A tensor for an image, which has a cat would probably be a vector with a one and two Zeros.
- One should represent the network as follows. With the following Kebab.

Presence ← 1 → p(cat)

Presence ← 0 → p(Kim)

Presence ← 0 → p(guma)

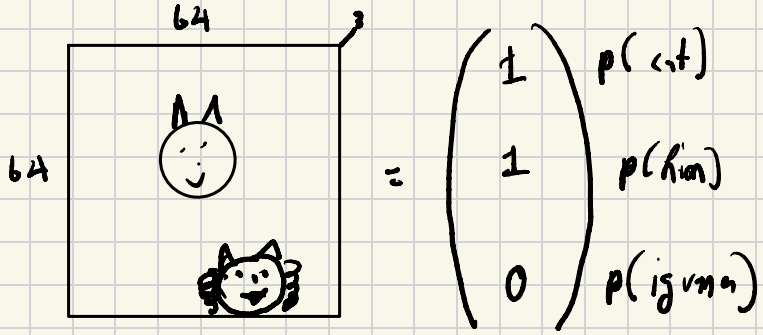
} Assume we use this schema to represent the data set.

∴ We train this network using the same technique here such that you initialize all of your weights & biases with a starting value, optimize the loss function by using gradient descent, and use $y = \_\_\_$ to predict.

- The neuron a_1 will be responsible for detecting cats because we decided to label our data set in such a manner. Thus, after training, the neuron a_1 is naturally going to detect cats.
- If we changed the labeling schema & said the second entry would correspond to the cat, then after training, you will detect that the neuron a_2 is responsible for detecting a Cat... etc
- The network will evolve depending on how you label your data set
- Do you think that this network could still be robust to different angles in the same picture?

- This cat now has a friend that is a lion.

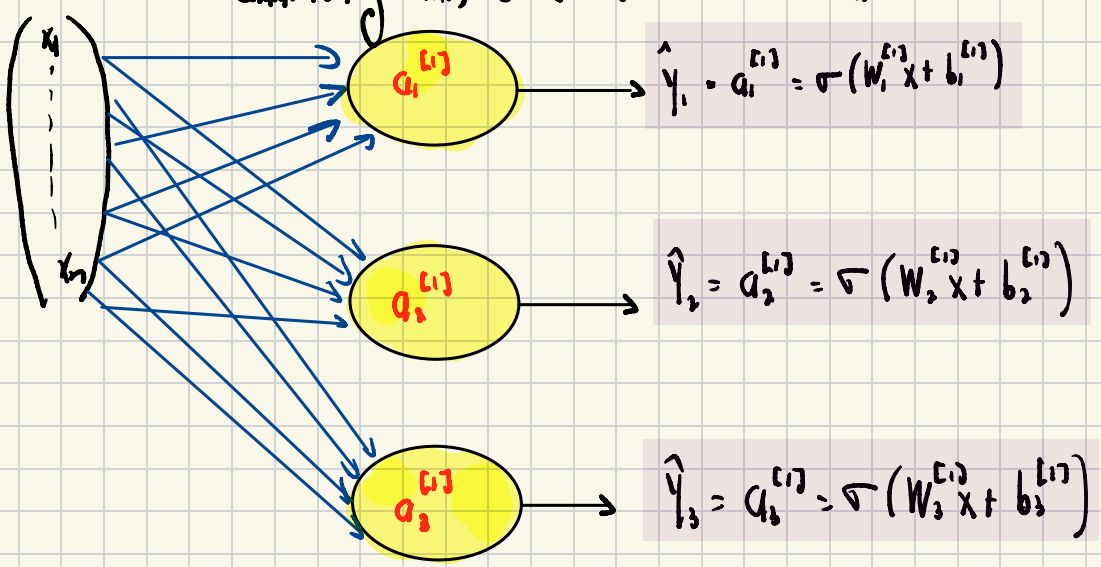
↓



- The network only sees (1, 1, 0) & an image. It does not see which "1" corresponds to the cat or the lion. With neural networks you do not need to tell them everything. If you have enough data they will figure them out.

- Even though you will have multiple of those animals either together or alone in a picture, the neuron will learn what it's looking for. and will understand that a specific "1" corresponds to a specific animal and yes, it is going to be robust. You just need alot of data of course to achieve this.

- This system is also going to robust because the 3 neurons are not communicating with each other. & thus we can train them



independently from one another. Further, the Sigmoid in $\hat{y}_1$ does not depend on the sigmoid in $\hat{y}_2$ & does not depend on the sigmoid in $\hat{y}_3$

⇒ That we can have $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ as an output. So You can think of it as 3 logistic Regressions. ⇒ We will not call this a Neural Network yet.

You can I guess also think of this as a 3-Neuron Network but NOT a Neural Network

Question: Is there a case such that we have a constraint that leads to only one possible outcome? Such that there is no Cat and Lion, it is either a cat or a lion.

Answer: Think about health care. There are many models that are made to determine whether or not someone has a skin disease based on cell microscopic images. So you want to classify a specific disease over a large number of diseases.

⇒ The model we have will still work but it will not be optimal because it's going to take longer to train.

i.e. one disease might be extremely rare and one of the neurons is never going to be trained

i.e. You are working in a Zoo & there is 1 iguana & thousands of lions and cats. The third neuron is never going to be trained

if you would want to start with another model whereby you put the constraint such that "There is only one disease that we want to predict."

Then let the model learn with all the Neurons Training Together by creating interactions between them.