Report 1 (Week 1)

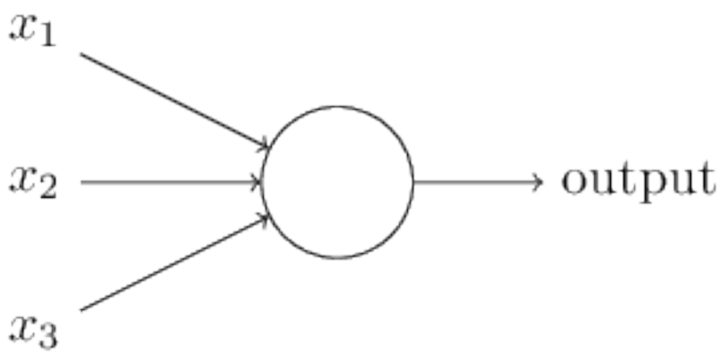
1. Introduction

A brief introduction about neural networks. What does it do, what place it has in machine learning studies etc.

.
.
.

1.1 Perceptrons

Basic things about perceptrons. This is an actual introduction to neural networks. It has a input and output scheme like this.

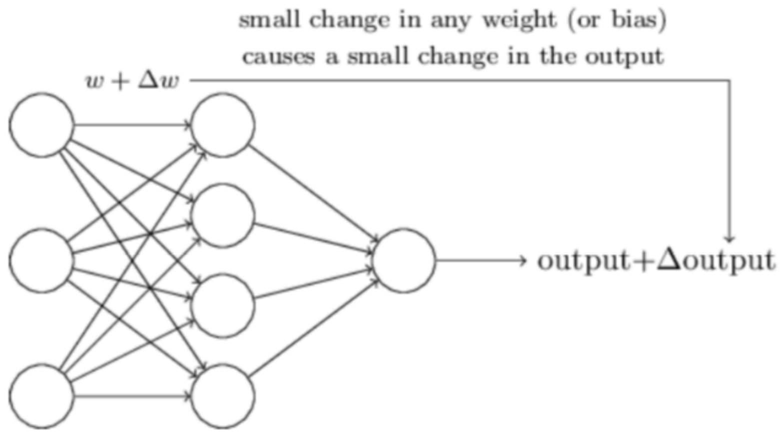


It has weights for the importance of the X_j. For example, if X_2 has more importance than X_3 or X_1, it has bigger weight and it directly affects the output.

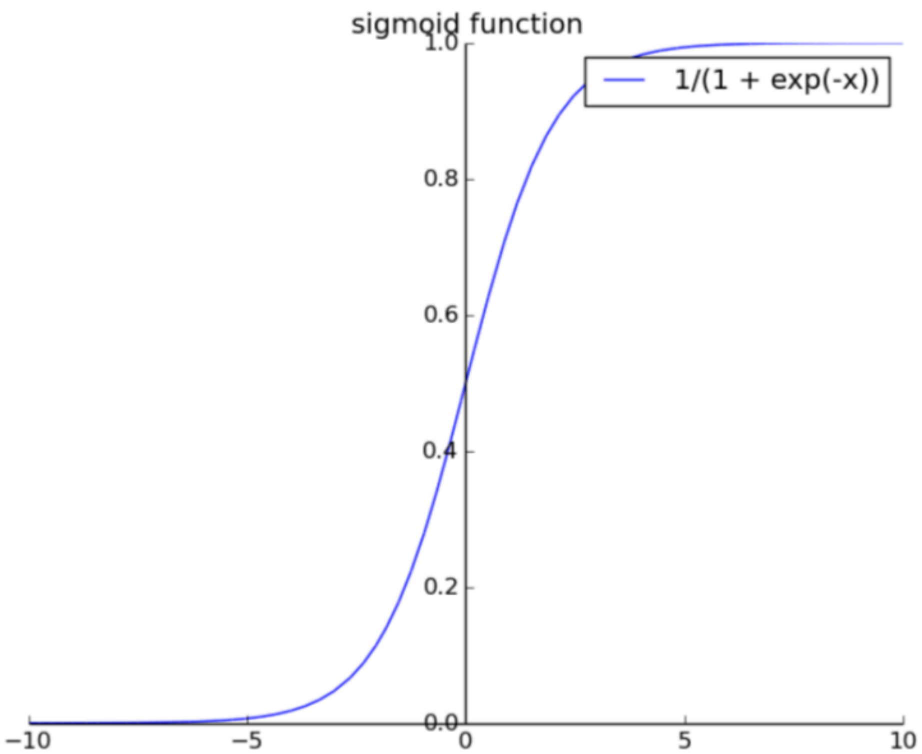
$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

1.2 Sigmoid neurons

This scheme explains the effects of changing weights to the output.

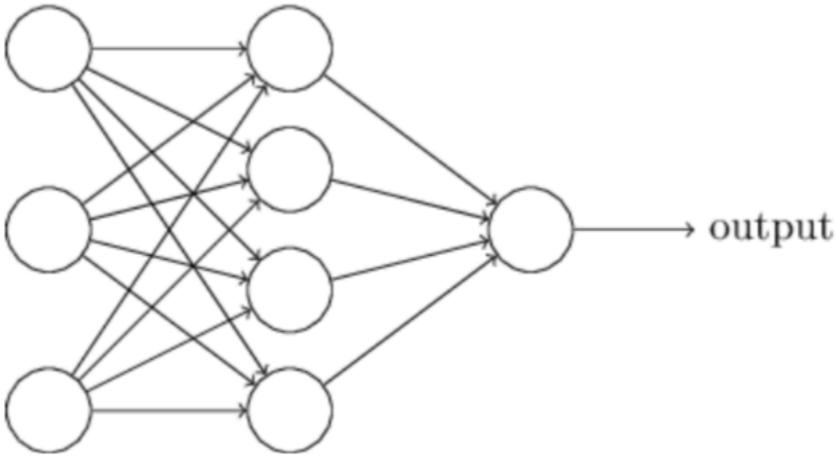


But this is not a healthy example with perceptrons. Little weight change can cause the output from 0 to 1 or from 1 to 0. And this is not a good thing to happen us. Therefore we are introduced with a new neuron that is called sigmoid neuron. It can have a value of any number in the range of from 0 to 1. Not just 0 or 1.

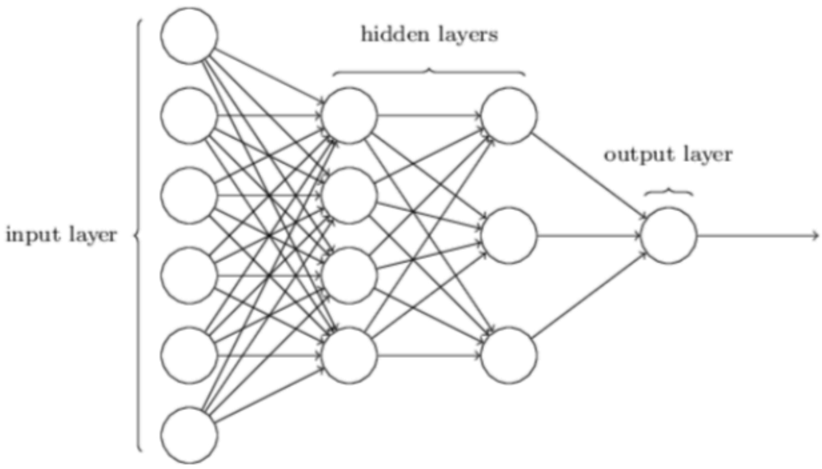


$$\frac{1}{1+exp(-\sum_j w_j x_j - b)}$$

1.3 The architecture of neural networks



This is a simple architecture of neural networks.



More complicated one with hidden layers etc.

1.4 Gradient Descent Algorithm

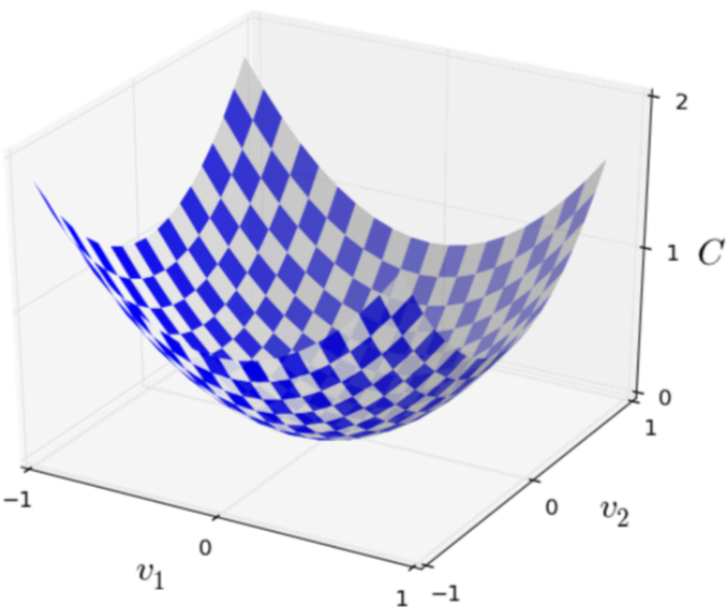
Gradient descent is a first-order optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point.

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$



Sample cost function with two variables.

1.5 Implementing our network to classify digits

Get the codes from https://github.com/mnielsen/neural-networks-and-deep-learning

It has an output like this:

```
Epoch 0: 9129 / 10000
Epoch 1: 9295 / 10000
Epoch 2: 9348 / 10000
...
Epoch 27: 9528 / 10000
Epoch 28: 9542 / 10000
Epoch 29: 9534 / 10000
```

Codes will be added.