Neural Networks: Representation

Termanteus

10 January 2020

Contents

I. Non-linear Classification

- For these types of problems, using linear/logistics regression still might be working if you keep adding more polynomial elements to the equations, but that would be computationally expensive ($O(\frac{n^2}{2})$ for squared term and so on,...), or causing overfiting so probably only fit for 2 features problems.
- Leads to **Neural Network**: which is more suitable for these scenarios.

II. Neural Networks

1. Model representation

- In this model, our x_0 input node is called "*bias unit*" (always equal to 1). We use the same *logistic function* as in classification (sometimes called sigmoid activation function); and theta is "weights":

$$\sigma = \frac{1}{1 + e^{-theta^T x}}$$

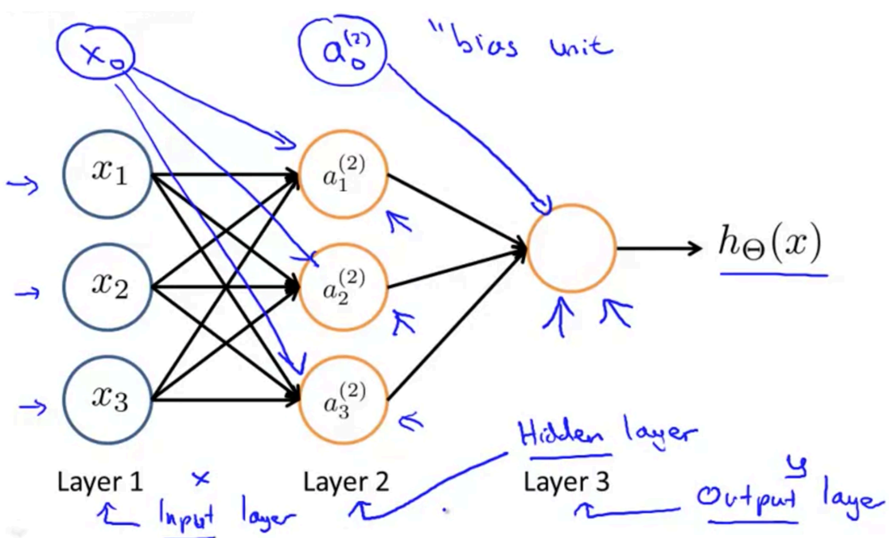

Figure 1: Simple Neural Network

- Mathematical Representation:

If we had one hidden layer, it would look like:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix} \rightarrow h_\theta(x)$$

The values for each of the "activation" nodes is obtained as follows:

$$\begin{aligned} a_1^{(2)} &= g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3) \\ a_2^{(2)} &= g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3) \\ a_3^{(2)} &= g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3) \\ h_\Theta(x) = a_1^{(3)} &= g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)}) \end{aligned}$$

- Each layer get its own matrix of weights, θ^j, determined as follow: "If network has s_j unit layer in j and s_{j+1} in layer j+1, then θ^j will be of dimension $s_{j+1}x(s_j + 1)$." (+1 because of the addition of the bias node: x_0).
- Vectorized implementation:
  - Define: $z_k^{(i)}$ such that:

$$\begin{aligned} a_1^{(2)} &= g(z_1^{(2)}) \\ a_2^{(2)} &= g(z_2^{(2)}) \\ a_3^{(2)} &= g(z_3^{(2)}) \end{aligned}$$

  - We can deduce to the vectorized equation:

$$\begin{aligned} z^{(j)} &= \Theta^{j-1}a^{j-1} \\ a^{(j)} &= g(z^{(j)}) \end{aligned}$$

- In the last layer, theta will have **one row** and a will be **one column**, leads to the result is a single number, the final result:

$$h_\Theta(x) = a^{j+1} = g(z^{j+1})$$

- Notice in the **last step** (layer j and j+1), we are do the **exact same thing** as we did in **logistic regression**.

2. Multiclass Classification

We can define our set of resulting classes as y:

$$y^{(i)} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\0\\1\end{bmatrix},$$

Each $y^{(i)}$ represents a different image corresponding to either a car, pedestrian, truck, or motorcycle. The inner layers, each provide us with some new information which leads to our final hypothesis function. The setup looks like:

$$\begin{bmatrix}x_0\\x_1\\x_2\\\cdots\\x_n\end{bmatrix} \rightarrow \begin{bmatrix}a_0^{(2)}\\a_1^{(2)}\\a_2^{(2)}\\\cdots\end{bmatrix} \rightarrow \begin{bmatrix}a_0^{(3)}\\a_1^{(3)}\\a_2^{(3)}\\\cdots\end{bmatrix} \rightarrow \cdots \rightarrow \begin{bmatrix}h_\Theta(x)_1\\h_\Theta(x)_2\\h_\Theta(x)_3\\h_\Theta(x)_4\end{bmatrix}$$

Our resulting hypothesis for one set of inputs may look like:

$$h_\Theta(x) = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$$

In which case our resulting class is the third one down, or $h_\Theta(x)_3$, which represents the motorcycle.

Figure 2: Multiclass Representation - Andrew NG