**Non-Linear Classification.** What if we have 100 features? Order of $\frac{n^2}{2}$. $\binom{n}{2}$

**Neural Networks.** Algorithms that try to mimic the brain. Was very widely used in 80s and early 90s, popularity diminished in late 90s. Recent resurgence: state-of-the art technique for many applications.

**One Learning Algorithm hypothesis.** Rewire, part of the brain learns to do other things.

**Neuron model: Logistic unit.** Inputs: $x_1, x_2, x_3$ Output: $h_\Theta(x)$, where $x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$ (features), $\Theta = \begin{bmatrix} \Theta_0 \\ \Theta_1 \\ \Theta_2 \\ \Theta_3 \end{bmatrix}$ (weight, parameters). $x_0 = 1$, the bias unit or bias neuron.

Input layer -hidden layer - output layer. $a_i^{(j)}$ ="activation" of unit $i$ in layer $j$. $\Theta^{(j)}$ = matrix of weights controlling function mapping from layer $j$ to layer $j+1$.

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

If network has $s_j$ units in layer $j$, $s_{j+1}$ units in layer $j+1$, then $\Theta^{(j)}$ has dimension $s_{j+1} \times (s_j + 1)$. For example, 3 nodes in input layer and 3 nodes in hidden layer. $\Theta^{(1)}$ is $3 \times 4$.

**Forward propagation: Vectorized implementation.** Rewriting the equations,

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

Define $a^{(1)} = x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$, $z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}$. Then we can write,

$$
z^{(2)} = \Theta^{(1)} a^{(1)} \qquad a^{(2)} = g(z^{(2)})(\text{element-wise sigmoid})
$$

Add $a_0^{(2)} = 1$, $a^{(2)} \in \mathbb{R}^4$. Then

$$
z^{(3)} = \Theta^{(2)} a^{(2)} \qquad h_\Theta(x) = a^{(3)} = g(z^{(3)})
$$

**Neural Network learning its own features.** Every $\Theta$ learns features.

**Non-linear Classification example: XOR/XNOR.** $x_1, x_2$ are binary (0 or 1).

$$
y = x_1 XOR x_2 \qquad x_1 XNOR x_2
$$

XOR true when $i \neq j$, XNOR true when $i = j$. where XNOR = NOT $(x_1 XOR x_2)$

**Simple example: AND.** $x_1, x_2 \in \{0, 1\}, y = x_1 AND x_2$. Assign weights to be $x_1^{(1)}0 = -30, x_1^{(1)}1 = 20, x_1^{(1)}2 = 20$. In other words, $h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$.

Look at,

| $x_1$ | $x_2$ | $h_\Theta(x)$ |
|---|---|---|
| 0 | 0 | $g(-30) \approx 0$ |
| 0 | 1 | $g(-10) \approx 0$ |
| 1 | 0 | $g(-10) \approx 0$ |
| 1 | 1 | $g(10) \approx 1$ |

Note that the rightmost column is the truth table.

**Example: OR Function.** $g(-10 + 20x_1 + 20x_1)$

**Example: Negation.** NOT $x_1$ $h_\Theta(x) = g(10 - 20x_1)$

| $x_1$ | $h_\Theta(x)$ |
|---|---|
| 0 | $g(10) \approx 1$ |
| 1 | $g(-10) \approx 0$ |

Put large negative weight in front of the variable you want to negate. $(NOTx_1)AND(NOTx_2) = 1$ if and only if $x_1 = x_2 = 0$. $h_\Theta = g(10 - 20x_1 - 20x_2)$.

**Example:** $x_1 XNOR x_2$**.** Let $a_1^{(2)}$ be AND function. Let $a_2^{(2)}$ be $(NOTx_1)AND(NOTx_2)$. Adding a bias unit, $a_0^{(2)} = 1$. $a_1^{(3)}$ is the OR function.The truth table is,

| $x_1$ | $x_2$ | $a_1^{(2)}$ | $a_2^{(2)}$ | $h_\Theta(x)$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 |

**Multiple output units: one-vs-all.** Want to predict pedestrian, car, motorcycle, and truck. Want $h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ when pedestrian, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ when car, etc.

Training set: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)}$