Machine learning notes: week 4

Lars Yencken

Notes

Non-linear hypotheses

- Suppose you need a decision boundary that's non-linear
  - Then with logistic regression, you'd need polynomial features
  - As you get more features, even quadratic features scale very poorly
- When is the number of features *n* large?
  - Images have huge numbers of features, one per input pixel
  - 50×50 pixels is already 2500 features in grayscale
  - Quadratic features would add $\approx \frac{(50 \times 50)^2}{2} = 3.1$ million features

Enter neural networks

- Designed to mimic the brain
  - Is there one core learning algorithm used throughout the brain?
  - Neuroplasticity: you can learn to see with electrodes placed on your tongue, or by echolocation, amongst other examples of the brain's capacity to rewire
- Was widely used, then unpopular, now recently having a resurgence
  - Used effectively for deep learning by big companies like Google, for example automatically recognising object categories from images, like bicyles

Our hypothesis

- We call ours an Artificial Neural Network (ANN)
- An ANN consists of nodes, which are simplified models of neurons
  - Each has some inputs $x_1, \ldots, x_n$, and output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$
  - Look familiar? This is an instance of logistic regression!

- Multiple neurons can share the same input, and their outputs can be inputs to a new layer of nodes
  - Nodes connected directly to the input are in the *input layer*
  - Nodes giving the final answer are in the *output layer*
  - Everything in between is called the *hidden layer*
- Call the "activation" of unit i in layer j: $a_i^{(j)}$
- Call the matrix of weights mapping layer j to layer $j+1$: $\Theta^{(j)}$
  - Say layer j has s_j units, layer $j+1$ itself s_{j+1} units
  - Then $\Theta^{(j)}$ is of dimension $s_{j+1} \times (s_j + 1)$
  - The $+1$ is because of the bias weight for each node, much like the θ_0 we used in earlier regression
- Our hypothesis $h_\Theta(x)$ is built from the output layer, and depends recursively on the activation of earlier layers
- Starting with the input layer, we use *forward propagation* to calculate $h_\Theta(x)$
- Consider the hidden layer as being specially learned features, which are then what's actually used for logistic regression in the output layer

Examples

- Logistic regression can learn the linear function $y = x_1$ AND x_2
- But $y = x_1$ XOR x_2 is non-linear, and can't be learned this way
- We can prove that with one hidden unit, an ANN can learn

Multiclass with ANNs

- Create a network with output units, one per class
- This is similar to doing one-vs-all with logistic regression

Exercises

1. Modelling binary functions:
   (a) Create the neural net for x_1 OR x_2
   (b) Create the neural net for x_1 NOT x_2
   (c) Create the neural net for x_1 XNOR x_2