Neural nets unit:

- General introduction (last week Wednesday)
- Trying out neural nets (last week Friday, in lab)
- How to train your perceptron (**Today**)
- The feed-forward and back-propogation algorithms (Wednesday)
- Deep learning: CNNs (Friday and next week Monday)
- Deep learning in practice (next week Wednesday, in lab)

Today and next time:

- Implementing a perceptron
- Training a perceptron
- Implementing an MLP (the feed-forward algorithm)
- Training an MLP (the back-propagation algorithm)

A **perceptron** is a function $\mathbb{R}^D \to \mathbb{R}$ defined as

$$p(\boldsymbol{x}) = h(\boldsymbol{\theta} \cdot \boldsymbol{x} + b) = h\left(b + \sum_{i=0}^{D-1} \theta_i x_i\right)$$

where

- $\boldsymbol{\theta}$ is the vector of weights
- b is the bias term
- h is the activation function

N is the size of the data. n ranges over data points, as in $\mathbf{x}_n$. D is the dimensionality of the data points. i and k range over data point components, as in x_i.

Let $\boldsymbol{t}$ be the target values, that is t_n is the target value for data point $\boldsymbol{x}_n$. We're using t instead of y so that y can be used for the output unit of an MLP. Let η be the learning rate.

For a single perceptron with weights $\boldsymbol{\theta}$, the weights are updated based on data point $\boldsymbol{x_n}$ by the perceptron rule:

$$\boldsymbol{\theta}^{\text{new}} = \boldsymbol{\theta}^{\text{old}} + \eta(t_n - p(\boldsymbol{x_n}))\boldsymbol{x_n}$$

M is the number of hidden units. j and ℓ range over hidden units, as in z_j. We are assuming a single output unit y.

The weights in the output unit are θ_{yj}, that is, the jth weight (corresponding to the hidden unit z_j) in output unit y.

The weights of the hidden units are θ_{ji}, that is, the $i$the weight (corresponding to the input component x_i) in hidden unit z_j.

The sum of squares error, as a function of the parameters (weights) $\boldsymbol{\theta}$ is

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2} \sum_{n=0}^{N-1} (y(\boldsymbol{x}_n) - t_n)^2$$

Coming up:

Due Mon, Apr. 7:
Read excerpt from Geron introducing neural nets
(See Canvas)

Due Wed, Apr 9:
Read and respond to two articles about bias in algorithms
(See Canvas)

Due Wed, Apr 16:
Implement perceptron training, feed-forward, and back-propagation
(You know enough to do the first part)

Sometime between Mar 31 and Apr 17:
Make an office-hours appointment for project check-in
(Originally the deadline was Apr 11)