CISC 271 Class 27

Classification – Single Artificial Neuron

Text Correspondence: Hastie *et al.*, 2009 [6], pp 130–132

Main Concepts:

- *Biological inspiration: cortical nerve cell*
- *Hyperplane with augmented vectors*
- *Neural networks use binary classes 0 and 1*

Sample Problem, Machine Inference: How can we model a neuron?

Historically, the idea of an *artificial neuron* dates to 1943, when McCulloch and Pitts [9] began to produce computational models of basic nerve cells. From early electrophysiology, McCulloch and Pitts observed that some neurons had a behavior that was roughly approximated as a linear sum of the inputs. If the j^{th} electrical input to a neuron was a value x_j, the input appeared to be weighted by a cell-specific value w_j. Taking n such inputs, the linear sum was

$$\sum_{j=1}^{n} x_j w_j = \vec{x} \cdot \vec{w} = \vec{x}^T \vec{w} = u \tag{27.1}$$

An illustration of a human neuron is shown in Figure 27.1.

If the weighted sum in Equation 27.1 was greater than a cell-specific threshold value β, the cell would "fire" and produce an output; otherwise, the cell would not have significant electrical activity. The actual electrophysiology is considerably more complicated, involving time-dependent sequences of electrical "spikes", but this simple mathematical model is still used in many current neural networks. A common terminology in machine learning is that the value β is referred to as the *bias* value. This condition for "firing" an artificial neuron can be written as

$$\sum_{j=1}^{n} x_j w_j \geq \beta \tag{27.2}$$

The way that an artificial neuron "fires" is also referred to as its *activation function*, written here as $\phi(\cdot)$. The activation function maps the linear sum u in Equation 27.1 to a real number, which we will call the *score*, as

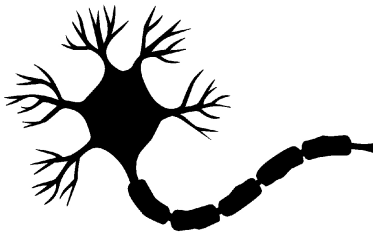

Figure 27.1: Illustration of a human neuron. The small projections from the cell body represent dendrites that provide electrical inputs to the neuron. The long projection represents the axon, which transmits the output to other cells. The axon is wrapped by electrically insulating cells that help to preserve the electrical signal from the neuron.

$$z = \phi(u) \quad \text{where} \quad u \stackrel{\text{def}}{=} \vec{x}^T \vec{w} + b \quad \text{and} \quad b = -\beta \tag{27.3}$$

Depending on how the artificial neuron is specified, the score z in Equation 27.3 might be any real number so $z \in \mathbb{R}$, or the score might be restricted to an interval such as $z \in [0\ 1]$. We need to read carefully when we are studying other source material on this topic.

A commonly used diagram illustrates the inputs to an artificial neuron as "arriving" from the left. The values x_i are weighted by the values w_i; the bias b is associated with a constant input value $x_{n+1} \stackrel{\text{def}}{=} +1$. The weighted sum of these are the scalar value u, which passes through the activation function $\phi(u)$ to produce an output z on the right of the diagram. Such an artificial neuron is often illustrated as in Figure 27.2.

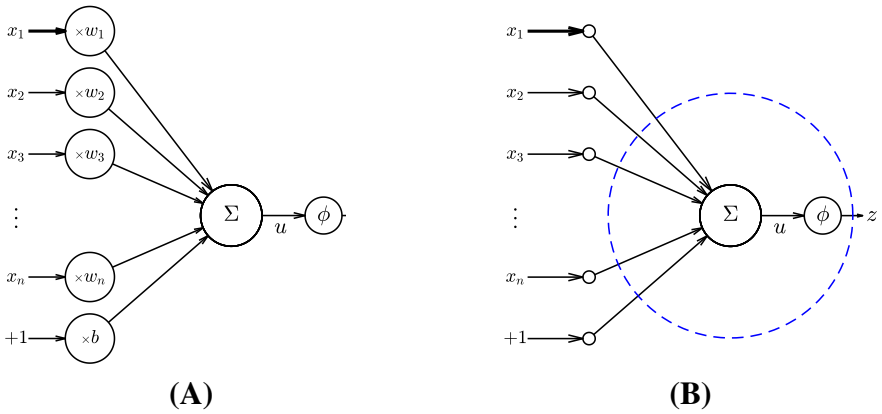

Figure 27.2: Illustrations of an artificial neuron. The input vector $\vec{x}$ is scaled by the weights $w_1, w_2, \ldots, w_n$ and biased by b; the result u is the input to the activation function ϕ that produces the neuron output z. (A) Data flow for an artificial neuron. (B) External variables are input $\vec{x}$ and output z.

The score z is a real number that might not be the "final" output of an artificial neuron. In some applications, the output of the artificial neurons must be *quantized* into two distinct classes. Although most of machine learning uses the classes $\{-1, +1\}$, in neural networks the biological inspiration is used so that the quantization is

$$q(z) \in \{0, 1\} \tag{27.4}$$

Consider the usual case in data analysis, in which there are m observations. In neural networks, the i^{th} observation is usually written as a "vector"[1] The linear sum for this "vector" is

$$u_i = \vec{x}_i^T \vec{w} + b \tag{27.5}$$

Combining Equation 27.5 and Equation 27.3, the artificial neuron will "fire" for this observation if and only if

$$u_i \geq 0 \tag{27.6}$$

We will explore artificial neural networks for binary classification problems. For such networks, it is important that the labels of the observations correspond to the quantization of Equation 27.4. Here, we require that each label y_i is

$$y_i \in \{0, 1\} \tag{27.7}$$

27.1 Hyperplane For An Artificial Neuron

Consider Equation 27.5; this computes the pseudo-distance of a vector $\vec{x}_i$ to a hyperplane $\mathbb{H}$ that is specified by a general normal vector $\vec{w}$ and a bias scalar b. We might find the notational overhead associated with the bias scalar b to be tedious.

Let us can consider an augmentation of the weight vector $\vec{w}$ and the observation "vector" $\vec{x}_i$. We will define the augmented objects as

$$\hat{w} \stackrel{\text{def}}{=} \begin{bmatrix} \vec{w} \\ b \end{bmatrix} \quad \text{and} \quad \hat{x}_i \stackrel{\text{def}}{=} \begin{bmatrix} \vec{x}_i \\ 1 \end{bmatrix} \tag{27.8}$$

We can use Equation 27.8 to accomplish the computation of Equation 27.5 succinctly, by writing

$$u_i = \hat{x}_i^T \hat{w} \tag{27.9}$$

[1] This is a vector in the weakest sense: as an ordered list of real numbers. An observation might not be a member of a vector space.

In a very simple model of an artificial neuron, we can omit the activation function $\phi(\cdot)$ of Equation 27.3; equivalently, we can use the identity function that maps the input to the output. This implies that we can take the pseudodistance u_i to the hyperplane $\mathbb{H}$, computed with Equation 27.9, and quantize u_i as 0 for a negative pesudo-distance and 1 as a non-negative pseudo-distance.

A common quantization is the *Heaviside* function, named after the mathematician and physicist Oliver Heaviside. This is also call the *step* function and it plays a prominent part in neural networks. The Heaviside function is defined as

$$H(u) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if} \quad u < 0 \\ 1 & \text{if} \quad u \geq 0 \end{cases} \tag{27.10}$$

Our simple model of an artificial neuron will be that it uses Equation 27.10 to classify an augmented observation "vector" $\hat{x}_i$ by using the augmented weight vector $\hat{w}$, with the computation

$$q(\hat{w}; \hat{x}_i) = H(\hat{x}_i^T \hat{w}) \tag{27.11}$$