CME 216, ME 343 - Spring 2020

Eric Darve, ICME

We start our discussion of deep neural networks with the basic unit at the core of the model:

The Perceptron

The perceptron was invented in 1957 by Frank Rosenblatt.

We will see that it is quite similar to the basic formula used for SVM.

A perceptron is a function that takes as input a vector x and outputs a real number.

Its formula is given by

$$h_{w,b}(x) = \phi(w^T x + b)$$

x are the input variables.

This is the data used to make our prediction.

w and b are parameters to optimize.

- w is called the weight vector / matrix.
- b is called the bias. It shifts $w^T x$ by a constant.

There are many different choices for ϕ but most choices are associated with the idea of neuron activation and threshold activation.

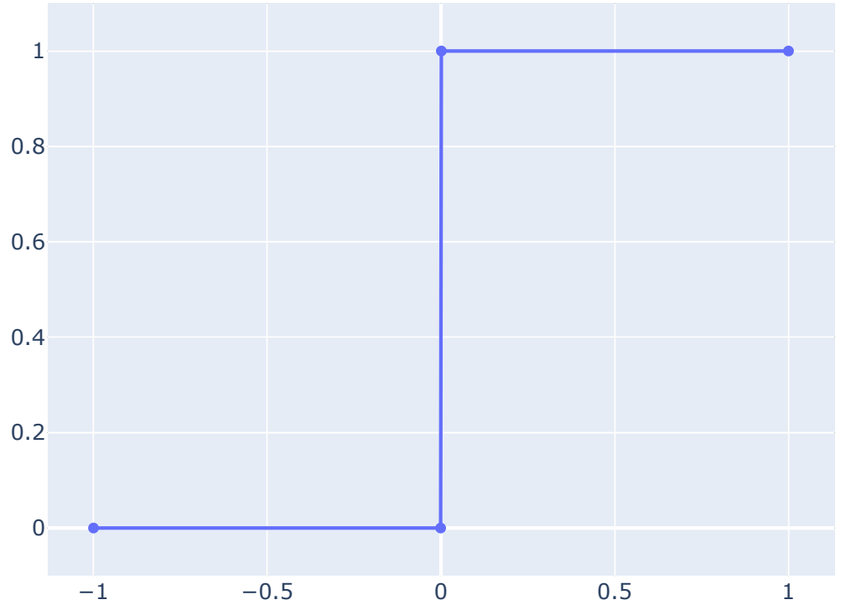
- Activation: the neuron is either on or off.
- Threshold: the neuron is off below a certain value and on above.

Mathematically, $\phi(z)$ is

- close to 0 when $z < 0$, and
- increases rapidly to 1 when $z > 0$.

The simplest example is the heaviside function.

heaviside function

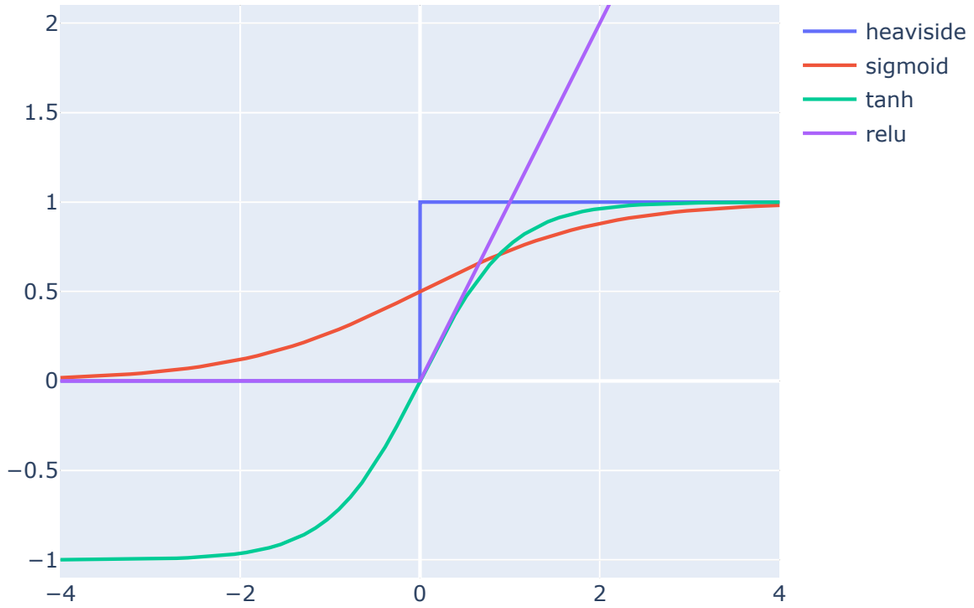


This function has some limitations that make it unsuitable for our optimization algorithms.

In particular, the slope is always 0.

Researchers have tried a variety of activation functions ϕ with better properties.

Some common choices are.


2
1.5
1
0.5
0
−0.5
−1
−4
−2
0
2
4
heaviside
sigmoid
tanh
relu

- sigmoid and tanh are very similar up to scaling and shifting
- relu has zero derivative when $x < 0$ which can cause some numerical problems for certain optimization algorithms.

Mathematical definitions

- heaviside(x): 0 if $x < 0$; 1 otherwise.
- sigmoid$(x) = \sigma(x) = 1/(1 + \mathbf{exp}(-x))$
- tanh$(x) = 2\sigma(2x) - 1$
- ReLU$(x) = \mathbf{max}(0, x)$

Optimization

How can we optimize w and b given some data?

We first need to define a loss function

$$L(w, b) = \sum_{i=1}^{m} (y_i - \phi(w^T x_i + b))^2$$

We can minimize the error by using a gradient descent method:

$$\Delta w = -\alpha \frac{\partial L}{\partial w}, \qquad \Delta b = -\alpha \frac{\partial L}{\partial b}$$

α = learning rate; Δw and Δb are increments in w and b.

Notation:

- y_i: training data
- $\hat{y}_i = \phi(w^T x_i + b)$: prediction using perceptron

$$L(w,b) = \sum_{i=1}^{m} (y_i - \phi(w^T x_i + b))^2 = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

Derivative with respect to w:

$$\frac{\partial L}{\partial w} = 2 \sum_{i=1}^{m} (\hat{y}_i - y_i)\ \phi'(w^T x_i + b)\ x_i$$

$$L(w, b) = \sum_{i=1}^{m} (y_i - \phi(w^T x_i + b))^2$$

Derivative with respect to b:

$$\frac{\partial L}{\partial b} = 2 \sum_{i=1}^{m} (\hat{y}_i - y_i)\ \phi'(w^T x_i + b)$$