ML2 – Perceptron

Introduction

One way to automate classification of objects using their features is to find a decision line in feature space which can separate the classes. In this activity we learn about a simple algorithm to find the decision line. It takes its inspiration from brain cells or neurons. A simplified schematic of a biological neuron is shown in Figure 1. Brain cells are known to receive electrical signals from other neurons through its dendrites and synapses and fire an electrical signal to other neurons through its axon. Brain cells are massively interconnected. There is estimated to be in the order of 10^{11} neurons in the human brain. Each neuron is estimated to be connected to 10^4 other neurons.

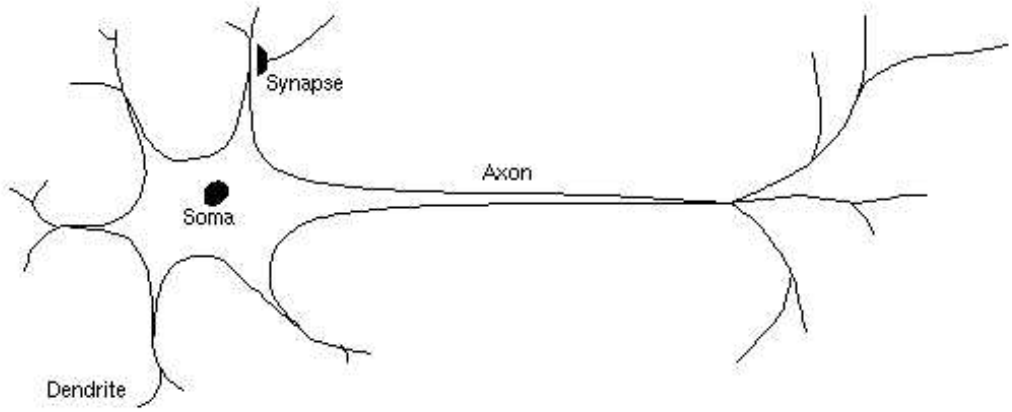

Figure 1: Simplified biological neuron illustration.(Image from https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html)

The earliest mathematical model of a neuron (published 1943 [1]) is the McCulloch-Pitts Neuron (MCP) and is shown in Figure 2. Each input x_i enters through a dendrite with synaptic strength or weight w_i,. The neuron sums these weighted inputs ($x_i w_i$) and lets the sum, a , act on an activation function, g. The result, z, is then fired through the output to other neurons.

In equations the sequence from input to output are as follows:

1. Multiply inputs x_i to corresponding synaptic weights w_i and then sum into a:

$$a = x_1 w_1 + x_2 w_2 + x_3 w_3 \ldots = \sum x_i w_i = \boldsymbol{x}^T \boldsymbol{w}$$

2. The output z is then equal to the output of an activation function acting on a,

$$z = g(a) \quad .$$

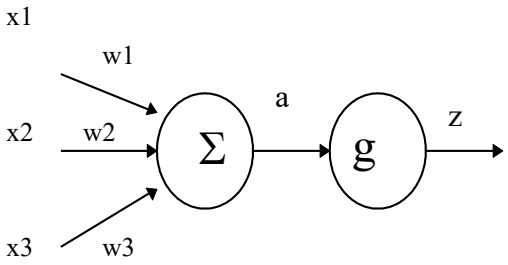

Figure 2.Artificial Neuron

The activation function can be any thresholding function such as a step function, a sigmoid function, or even a Gaussian function.

The Perceptron Algorithm

Just like a biological neuron, an artificial neuron "learns" by adjusting its weights based on a weight-change rule that minimizes a certain cost function. In 1958 Frank Rosenblatt proposed the first Perceptron learning algorithm [2]. A perceptron can perform classification through supervised learning if the classes are linearly separable in feature space. This means, if the feature space is two-dimensional, the decision boundary which separates the two classes is a line (Figure 3), if three-dimensional, the decision boundary is a plane, or if hyperdimensional, the decision surface is a hyperplane.

Suppose we have two classes of objects, ω_1 and ω_2, which are both described by features x_1 and x_2. In the context of Activity 1, ω_1 may be bananas, ω_2 may be oranges, x_1 is color and x_2 is aspect ratio.

We will follow some notation rules:

- Subscripts of x are the feature components. Corresponding weights w will be indicated by subscripts as well. Thus x_j and w_j are the jth feature and corresponding synaptic weight, respectively. The feature vector is denoted by $\boldsymbol{x}=[x_1 x_2]^T$.
- Superscripts in x indicate the input sample number. The corresponding perceptron output z is also indicated by a superscript. Thus x_j^i indicate the jth feature of the ith sample while z^i is the output of the perceptron due to input $\boldsymbol{x}^i$.

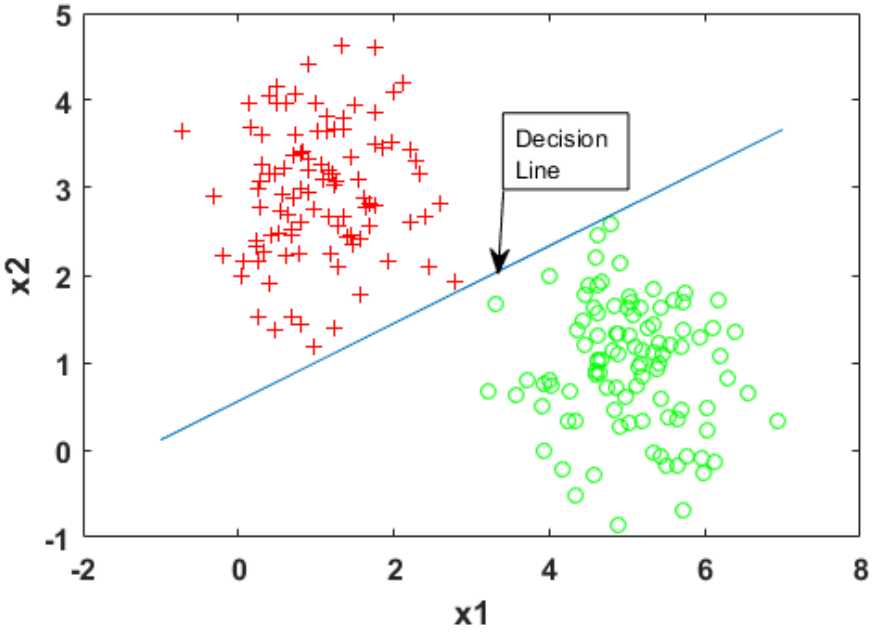

Figure 3: The two classes shown are linearly separable. A decision line can separate the two classes in feature space.

The perceptron weight change rule is as follows:

For each sample i

1. Include a constant bias input x_0 = 1 in all sample inputs. Thus, a two-feature vector will become 3 (that is $[1 \quad x_1^i \quad x_2^i]^T$). Let d^i be the class label of the feature: $d^i=1$ if the feature belongs to class 1, $d=-1$ if the feature belongs to class 2.
2. Initialize the weights to some random number. Note the x_o will have its corresponding weight w_o .

3. Calculate the perceptron output as in Steps 1 and 2 above. Here, we will use the step function as the thresholding function,

$$z=g(a)=\begin{Bmatrix} 1 & \text{if } a\geq 0 \\ -1 & \text{otherwise} \end{Bmatrix}$$

4. Calculate the weight change Δw_j as

   $\Delta w_j=\eta\left(d^i-z^i\right)x_j^i$.

   Here, η is the learning rate and is assigned a number between 0 and 1.

5. Update the weights as $w_j=w_j+\Delta w_j$

6. Iterate through samples. Stop when $\left(d^i-z^i\right)$ is small or a number of iterations is reached.

Procedure

1. Using your fruit feature data, take 2 classes at a time and compute the decision surface between the two classes using the perceptron algorithm.

2. Plot your data in feature space and overlay the decision line. To draw the decision line, note that an alternative form of the equation of the line is given by:

   $Ax+By=C$ or $-C+ax+By=0$.

   In terms of slope and y-intercept form, $y=\frac{C}{B}-\frac{A}{B}x$ where $m=\frac{-A}{B}$ and $b=\frac{C}{B}$.

   The product of inputs and weights make up an equation of the line:

   $w_o+w_1x_1+w_2x_2=0$ (note: $x_o=1$). With $C=-w_0$, $A=w_1$, and $B=w_2$ the decision line will have slope $m=-w_1/w_2$ and y-intercept $b=-w_0/w_2$.

3. If there are more than two classes, compute another decision line for the other pair of classes.

4. You can now test your perceptron if it can classify fruits. Using $w_o+w_1x_1+w_2x_2=0$ enter features of an unclassified fruit in x1 and x2. If the result of the equation is greater than 0 (positive) then the feature point is above the decision line so the fruit belongs to class 1, if negative

then the feature point is below the decision line thus the fruit belongs to class 2.

References

1. McCulloch, Warren S., and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity." The bulletin of mathematical biophysics 5, no. 4 (1943): 115-133
2. Rosenblatt, Frank. "The perceptron: A probabilistic model for information storage and organization in the brain." Psychological review 65, no. 6 (1958): 386

