Artificial Intelligence: *Modeling Human Intelligence with Networks*

Jeová Farias Sales Rocha Neto
jeova_farias@brown.edu

The Perceptron Algorithm!!

The perceptron

Algorithms

- What is an algrithm?

The perceptron

Algorithms

- What is an algrithm?
- Set of instructions, typically to solve a class of problems or perform a computation.

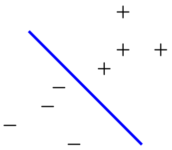
The perceptron

Algorithms

- What is an algrithm?
- Set of instructions, typically to solve a class of problems or perform a computation.

From last class

- We had a rule: separate the data using a line,



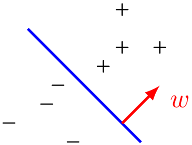
The perceptron

Algorithms

- Set of instructions, typically to solve a class of problems or perform a computation.

From last class

- We had a rule: separate the data using a line, defined by w.



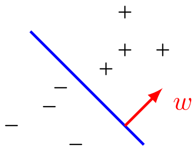
The perceptron

Algorithms

- Set of instructions, typically to solve a class of problems or perform a computation.

From last class

- We had a rule: separate the data using a line, defined by w.



- How do we find w automatically?

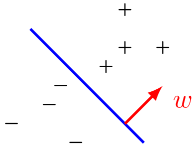
The perceptron

Algorithms

- Set of instructions, typically to solve a class of problems or perform a computation.

From last class

- We had a rule: separate the data using a line, defined by w.



- How do we find w automatically?
- The Perceptron Algorithm!

The perceptron

Set up

- In our exercises, we begin with a matrix *data* that contains the data: the points and the classes (negatives or positives).
- The matrix *data* will look like this:

$$data = \begin{bmatrix} 2.1 & 5.2 & 1 \\ 1.1 & -2.7 & -1 \\ 1.4 & 2.2 & -1 \\ \vdots & \vdots & \vdots \\ 3.5 & 1.7 & 1 \end{bmatrix}$$

The perceptron

Set up

- In our exercises, we begin with a matrix *data* that contains the data: the points and the classes (negatives or positives).
- The matrix *data* will look like this:

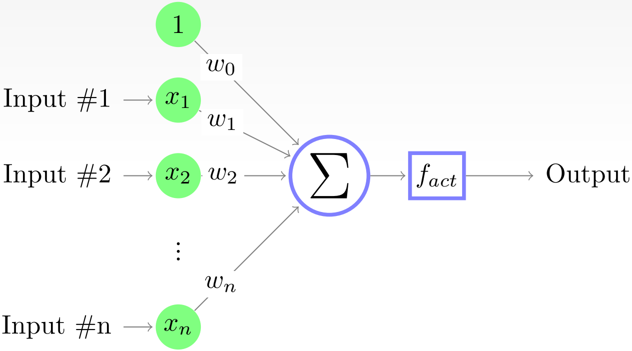
$$data = \begin{bmatrix} 2.1 & 5.2 & 1 \\ 1.1 & -2.7 & -1 \\ 1.4 & 2.2 & -1 \\ \vdots & \vdots & \vdots \\ 3.5 & 1.7 & 1 \end{bmatrix}$$

- We then break it up in two variables: a matrix X of points and a vector *classes* of classes:

$$X = \begin{bmatrix} 2.1 & 5.2 \\ 1.1 & -2.7 \\ \vdots & \vdots \\ 3.5 & 1.7 \end{bmatrix}, \quad classes = [1, -1, \ldots, 1]$$

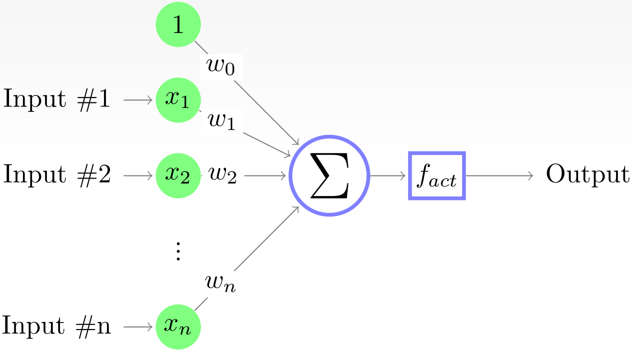
The perceptron

Remember the neuron model?



The perceptron

Remember the neuron model?



- We need to add a bias...

The perceptron

Adding the bias

- Each row of X is an input point $x = [x_1, x_2]$ and there are m data points.

The perceptron

Adding the bias

- Each row of X is an input point $x = [x_1, x_2]$ and there are m data points. We need to add a 1 before them:

$$X = \begin{bmatrix} 2.1 & 5.2 \\ 1.1 & -2.7 \\ \vdots & \vdots \\ 3.5 & 1.7 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2.1 & 5.2 \\ 1 & 1.1 & -2.7 \\ \vdots & \vdots & \vdots \\ 1 & 3.5 & 1.7 \end{bmatrix}$$

- This can be done in python by a function `np.concatenate()`.

The perceptron

The Algorithm

Algorithm 1 Perceptron - v2

Input: $X \in \mathbb{R}^{m\times 3}$ and $classes \in \mathbb{R}^m$
Output $w \in \mathbb{R}^3$

Initialize w
for $i = 1$ to m **do**
    $x = X[i,:]$
    $true\_class = classes[i]$,
    $predicted\_class = f_{act}(x \cdot w)$,
    if $predicted\_class == 1$ **and** $true\_class == -1$ **then**
        $w = w - x$
    end if
    if $predicted\_class == -1$ **and** $true\_class == 1$ **then**
        $w = w + x$
    end if
end for

The perceptron

Algorithm 2 Perceptron - v2

Input: $X \in \mathbb{R}^{m\times 3}$, $classes \in \mathbb{R}^m$, $num\_epochs \in \mathbb{Z}$
Output $w \in \mathbb{R}^3$

Initialize w
for $epoch = 1$ to num_epochs **do**
    for $i = 1$ to m **do**
        $x = X[i,:]$
        $true\_class = classes[i]$,
        $predicted\_class = f_{act}(x \cdot w)$,
        if $predicted\_class == 1$ **and** $true\_class == -1$ **then**
            $w = w - x$
        end if
        if $predicted\_class == -1$ **and** $true\_class == 1$ **then**
            $w = w + x$
        end if
    end for
end for

The perceptron

Algorithm 3 Perceptron - Final

Input: $X \in \mathbb{R}^{m \times 3}$, $classes \in \mathbb{R}^m$, $num\_epochs \in \mathbb{Z}$
Output $w \in \mathbb{R}^3$

Initialize w
for $epoch = 1$ to num_epochs **do**
    for $i = 1$ to m **do**
        $x = X[i,:]$,
        $true\_class = classes[i]$,
        $predicted\_class = f_{act}(x \cdot w)$,
        if $predicted\_class \times true\_class == -1$ **then**
            $w = w + true\_class \times x$
        end if
    end for
end for