

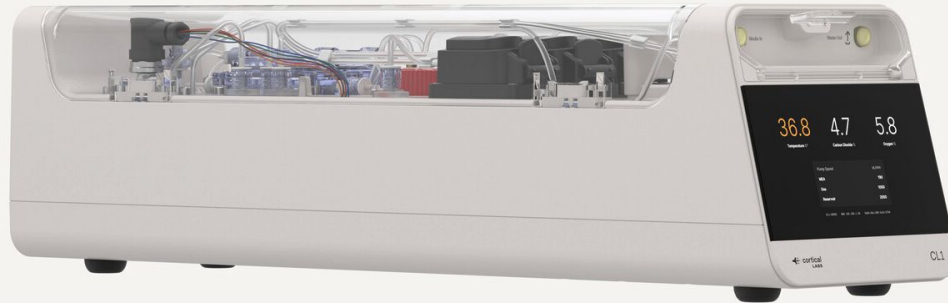
The Perceptron Learning Rule (a.k.a. The Perceptron Trick)

LIN 313 Language and Computers
UT Austin Fall 2025
Instructor: Gabriella Chronis

If ever the twain shall meet...

Introducing the world's first biological neural network: <https://corticalabs.com/cl1>

Silicon meets neuron



Real neurons are cultivated inside a nutrient rich solution, supplying them with everything they need to be healthy. They grow across a silicon chip, which sends and receives electrical impulses into the neural structure.

Admin

- HW 2 grades posted
- schedule re-arranged
 - more time on neural nets before midterm
 - word vectors after midterm
- study guide soon
- collab crib sheet

Overview 10/10

- The Perceptron
 - Applying the Perceptron algorithm (aka 'the forward pass')
 - Updating the Perceptron algorithm with the Perceptron trick (aka 'the backward pass')
- (or monday) Voting on classifiers

Pilot Annotation Results

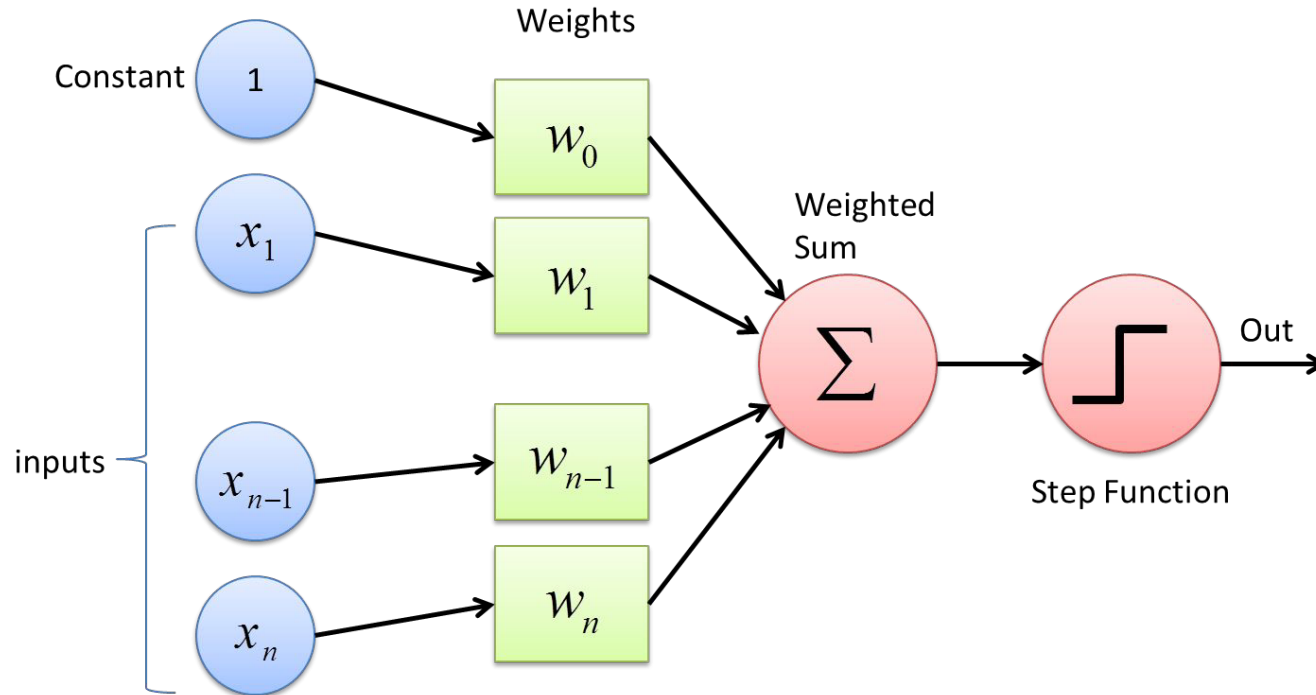
results:

https://docs.google.com/spreadsheets/d/1c3AnPdg9rI0Q5ZDEIjUy6t0FxV4g1_L8mUPW-irkZ20/

Perceptron Intuition

Should you buy a ticket to the next UT Game?

The Perceptron Algorithm



The Perceptron Algorithm (again)

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

\mathbf{w} is the weight vector ($w_1 w_2 \dots w_i$)

\mathbf{x} is the input vector ($x_1 x_2 \dots x_i$)

b is the bias, still a regular number

$\text{sign}()$ is the activation function

$\mathbf{w} \cdot \mathbf{x}$ is the **dot product** of \mathbf{w} and \mathbf{x}

*the **dot product** is just shorthand for the weighted sum!*

The Dot Product Definition

$$\mathbf{a} = \langle a_1, a_2, a_3 \rangle \quad \mathbf{b} = \langle b_1, b_2, b_3 \rangle$$

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

Vectors

The equation gets simpler if we think of the input as vectors.

A vector is an ordered list of numbers.

An ordered pair is a vector.

An ordered pair has a **geometric interpretation**.

A vector is an ordered pair with more numbers

A vector also has a geometric interpretation

Vector Dot Products

The Dot Product

Definition

$$\mathbf{a} = \langle a_1, a_2, a_3 \rangle \quad \mathbf{b} = \langle b_1, b_2, b_3 \rangle$$

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

To calculate the dot product, multiply each cell in vector **a** with the corresponding cell in vector **b**, then add them together.

The dot product of two **vectors** is always a **scalar**.
A scalar is just a plain old number.

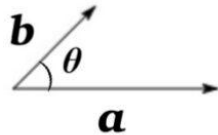
Meaning of the Dot Product

The dot product of two vectors tells you about what direction they are pointing.

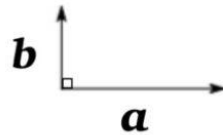
If the dot product is positive, the vectors are pointing in the same direction.

If the dot product is negative, the vectors are pointing in the opposite direction.

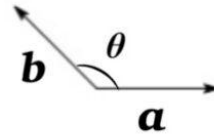
If the dot product is 0, the vectors are at right angles to each other.



$$\mathbf{a} \cdot \mathbf{b} > 0$$



$$\mathbf{a} \cdot \mathbf{b} = 0$$



$$\mathbf{a} \cdot \mathbf{b} < 0$$

Putting it all together

What does a **forward pass** through the Perceptron look like for our UT example?

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

It's an away game. two of my friends are actually going, but I'm broke and I have to write an essay and finish a lab write up this weekend.

$\mathbf{w} =$

$\mathbf{x} =$

$b =$

Putting it all together

What does a **forward pass** through the Perceptron look like for our UT example?

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

It's a home game, the weather is balmy. I have a ton of homework but all of my friends are going.

$\mathbf{w} =$

$\mathbf{x} =$

$b =$

Observation

There is a relationship between the *signs* of the weights and inputs and the *sign* of the dot product.

w_i	x_i	$x_i w_i$ is _____. Dot product is pushed towards _____
+	+	+
-	-	+
-	+	-
+	-	-

When the signs at $x_i w_i$ match, that connection pushes the output towards YES

Perceptron Learning

Perceptron learning rule
If $\hat{y} = y$, do nothing. Otherwise, for each w_i in \mathbf{w} : $w'_i = w_i + \alpha * y * x_i$

- \mathbf{w}'_i : the new weight at index i
(pronounced 'w eye prime')
 \mathbf{w}_i : the current weight at index i
 α : the learning rate (alpha) usually small
 y : the correct class label
 \mathbf{x}_i : the input at index i

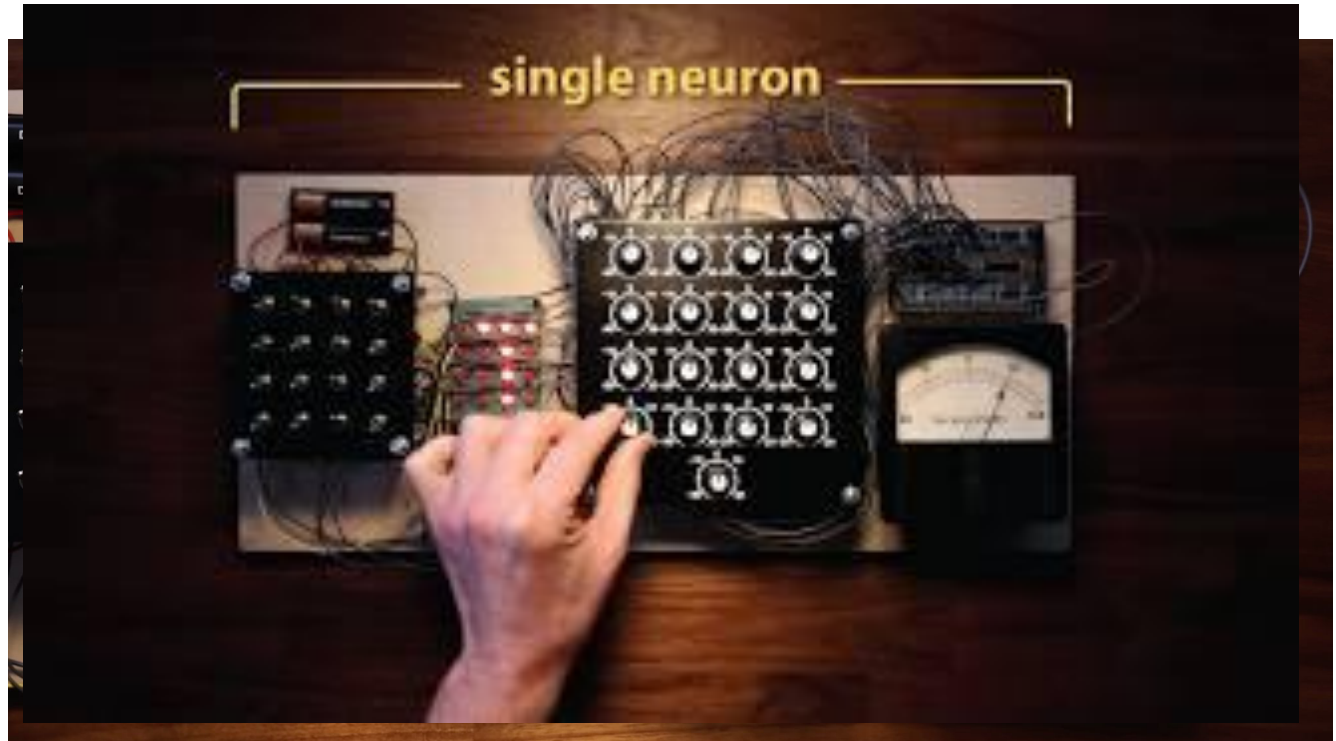
In English:

If we guessed the right answer, leave everything as is.

If we guessed the wrong answer, then move the weights a little bit towards a correct prediction for that input. Do this by updating all of the weights to look a little bit more like the input at that weight.

Visualizing the Perceptron rule with dials as weights

When the perceptron incorrectly outputs POS for a J shape, turn all of the dials for ON inputs a little bit towards NEG, and turn all of the dials for OFF inputs towards positive



Learning AND with the Perceptron Trick

We have a Perceptron with two weights. We want it to learn the logical AND function. (That is, we want to output 1 if and ONLY if $w_1 = 1$ AND $w_2 = 1$.)

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

Perceptron learning rule
If $\hat{y} = y$, do nothing. Otherwise, for each w_i in \mathbf{w} : $w'_i = w_i + \alpha * y * x_i$

x_1	x_2	\hat{y}

initial settings:

- learning rate $\alpha = 1$
- $\mathbf{w} = [0, 0]$
- $b = 0$

Training on real data

One run through training data through the update rule is called an "epoch."

We typically train neural nets with multiple epochs, i.e. the algorithm sees each training pair several times.