Introduction to Neural Networks

Part One: Perceptron

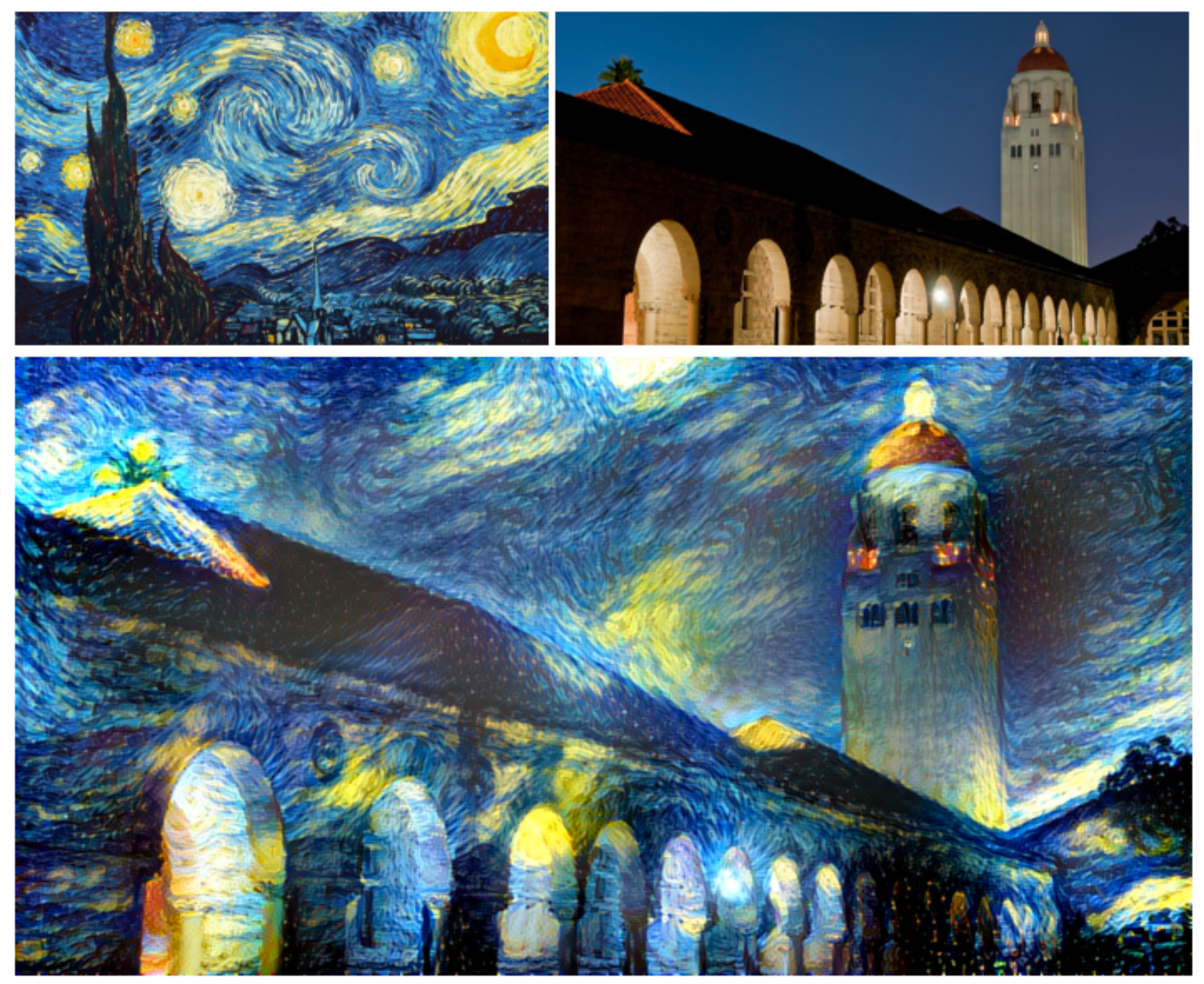
Portland Data Science Group
Hannes Hapke
@hanneshapke

1/26/2016

Why Machine Learning?
Why Neural Networks?

Algorithms that can **learn** from and **make predictions** on data.

Source: https://en.wikipedia.org/wiki/Machine_learning



Source: https://github.com/jcjohnson/neural-style
A Neural Algorithm of Artistic Style by Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

Instead of writing very specific code …

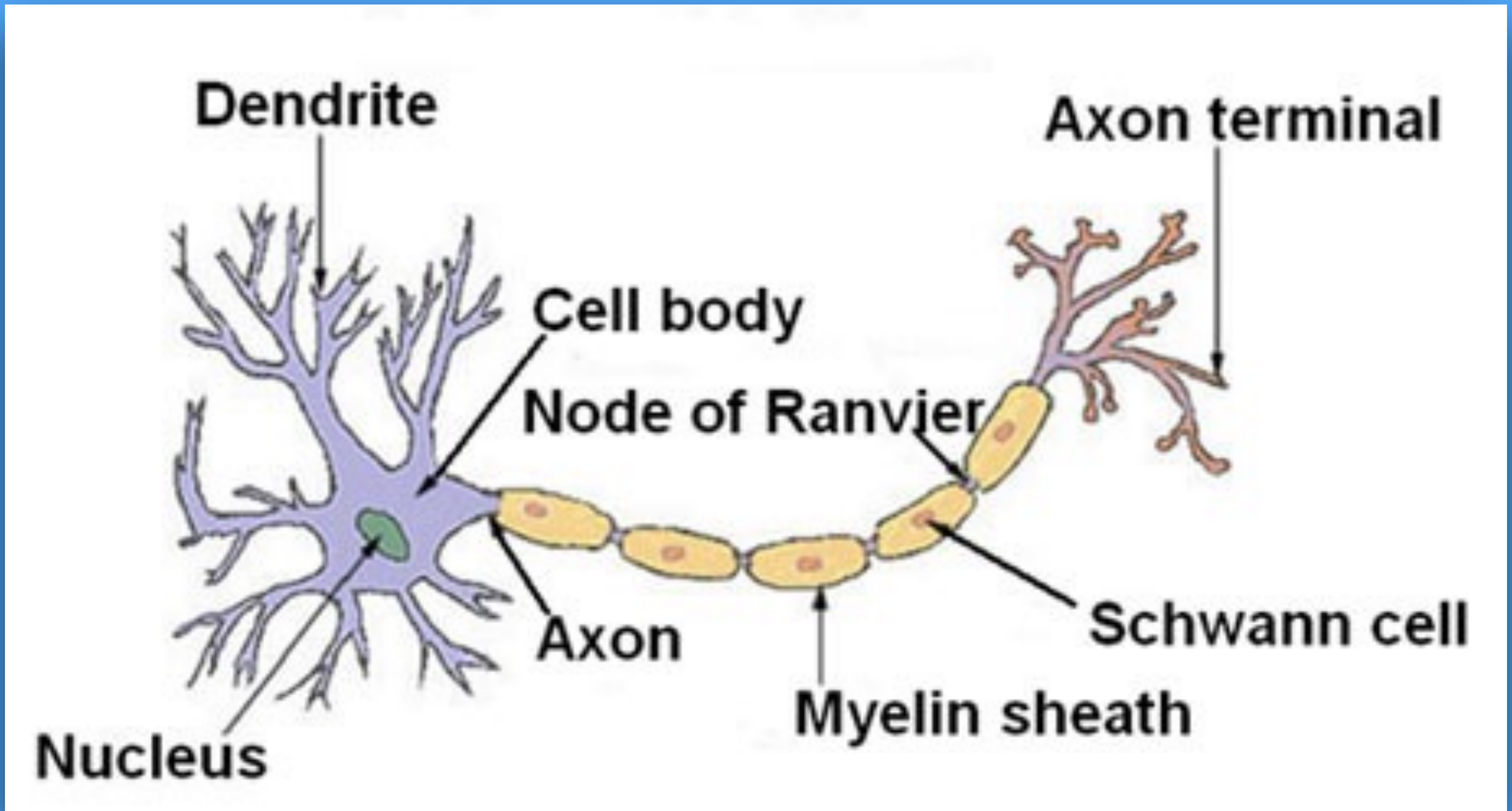
```
def _and(a, b):
    if a:
        if a:
            return True
    return False
```

You can apply a handful of machine learning algorithms to …

Recommend books, predict order volumes, predict flight delays, control your thermostat, drive a car, grade papers, detect fraud, identify animals, classify your emails, …

How does a single neuron work?

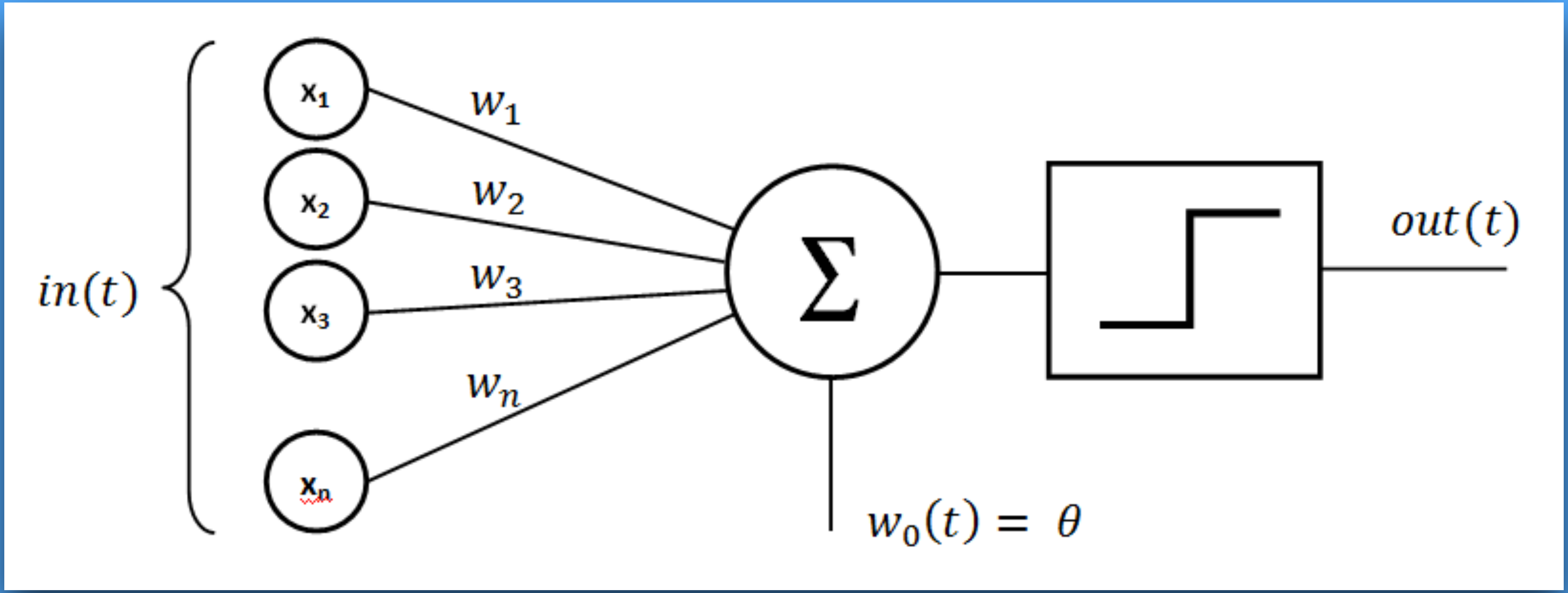
Source: https://en.wikipedia.org/wiki/Machine_learning



Source: https://upload.wikimedia.org/wikipedia/commons/b/bd/Neuron.jpg

How can a single neuron be represented mathematically?

Source: https://en.wikipedia.org/wiki/Machine_learning



Source: https://upload.wikimedia.org/wikipedia/commons/8/8c/Perceptron_moj.png

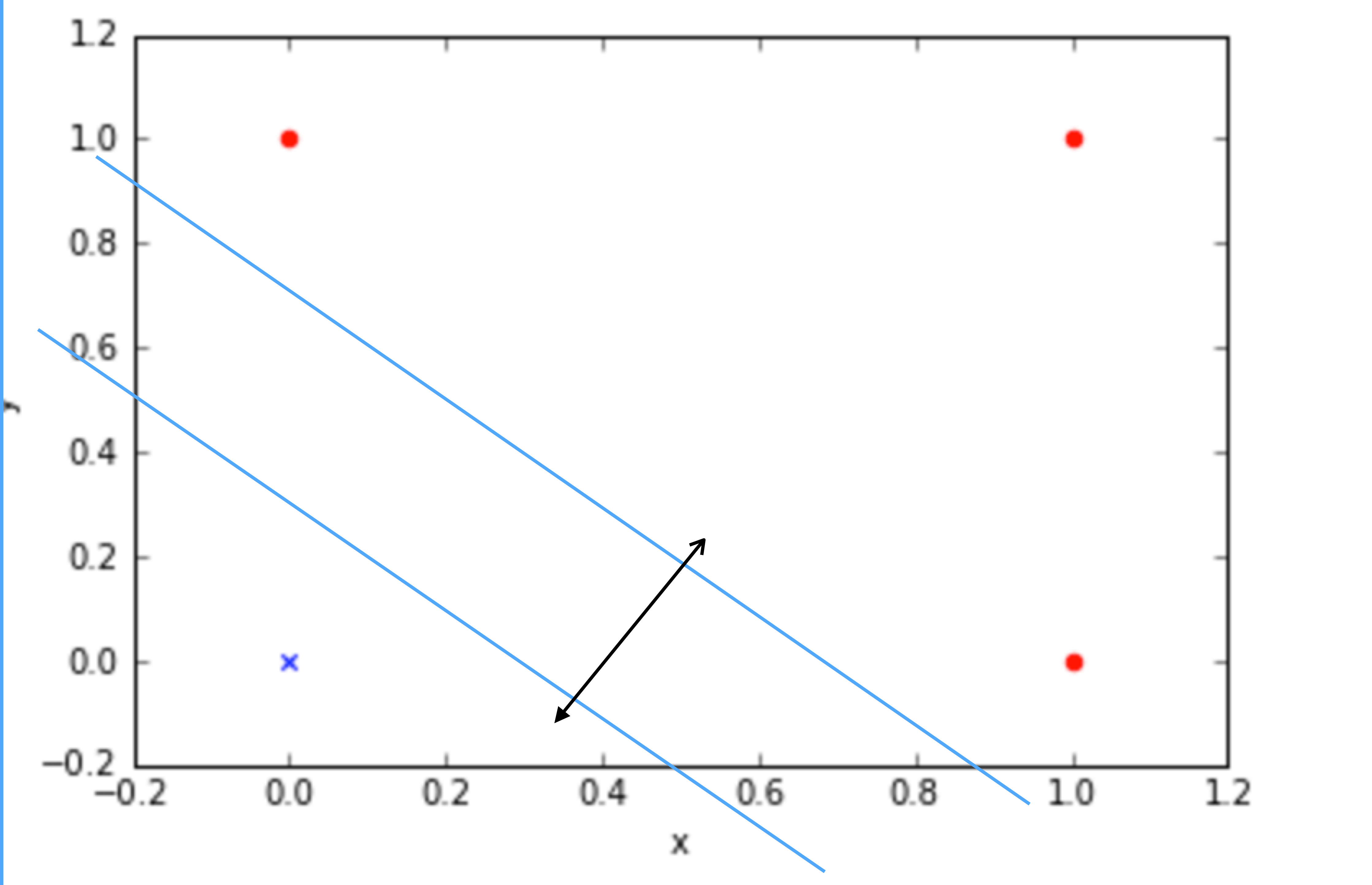
How to activate the neuron?

$$w = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$z = x^T * w + w_0$$

$$\phi(z) = \begin{cases} 1 \; if \; z > 0 \\ 0 \; otherwise \end{cases}$$

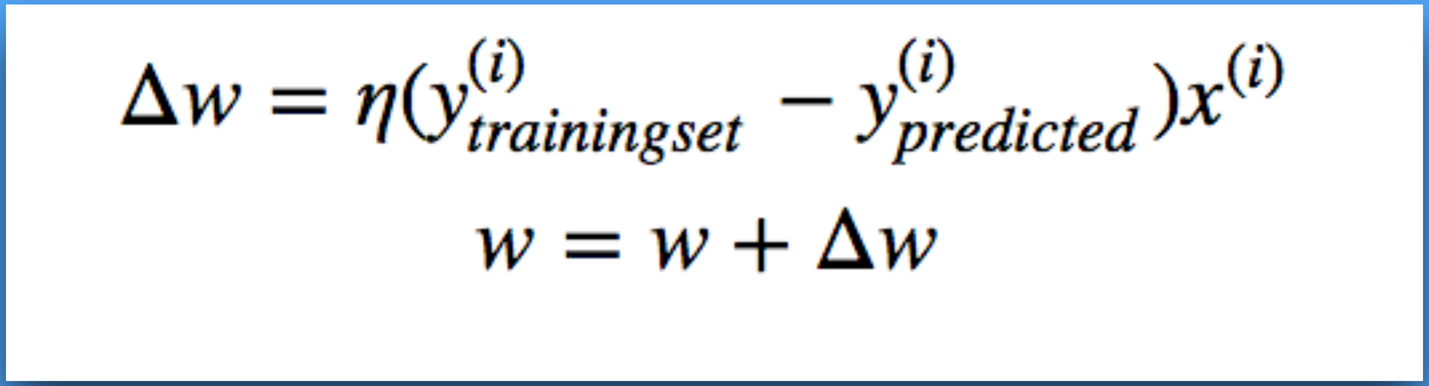
Why is a bias w_0 required?


1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
y
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
x

How does the Perceptron learn?

1. Initialize the weights with random values

2. For each training sample, compute the predicted y and update the weights accordingly



$$\Delta w = \eta(y^{(i)}_{trainingset} - y^{(i)}_{predicted})x^{(i)}$$

$$w = w + \Delta w$$

Implement your own Perceptron

How to get started?

0. Create a virtual environment with

```
mkvirtualenv perceptron
```

1. Fork the iPython notebook

   https://github.com/hanneshapke/PDX-data-perceptron.git

2. Install the requirements

```
pip install -r requirements.txt
```

3. Start the notebook with

```
ipython notebook
```

What to do?

1. Complete the net_input method

2. Complete the fit method
    2.1 Initialization of the Perceptron weights
    2.2 Calculate the difference of w
    2.3 Update the weights

3. Complete the predict method

What can you now predict?

- AND operation
- Identify Irises (setosa vs. versicolor)
- Credit card fraud/approval
- XOR operation

Questions along the way …

What works well?

What doesn’t work well?

Can you predict a reliable credit approval?

What’s the deal with the XOR operation?

Thanks to you

Thanks also to
@hobsonlane
@GrimmScientist
@uglyboxer
for their help to prepare this presentation

@hanneshapke

https://github.com/hanneshapke/PDX-data-perceptron.git

Please submit pull requests to share your solution.