GENERAL ASSEMBLY

DATA SCIENCE

NEURAL NETWORK AND SVM

I. ARTIFICIAL NEURAL NETWORKS

II. SUPPORT VECTOR MACHINES

I. ARTIFICIAL NEURAL NETWORKS

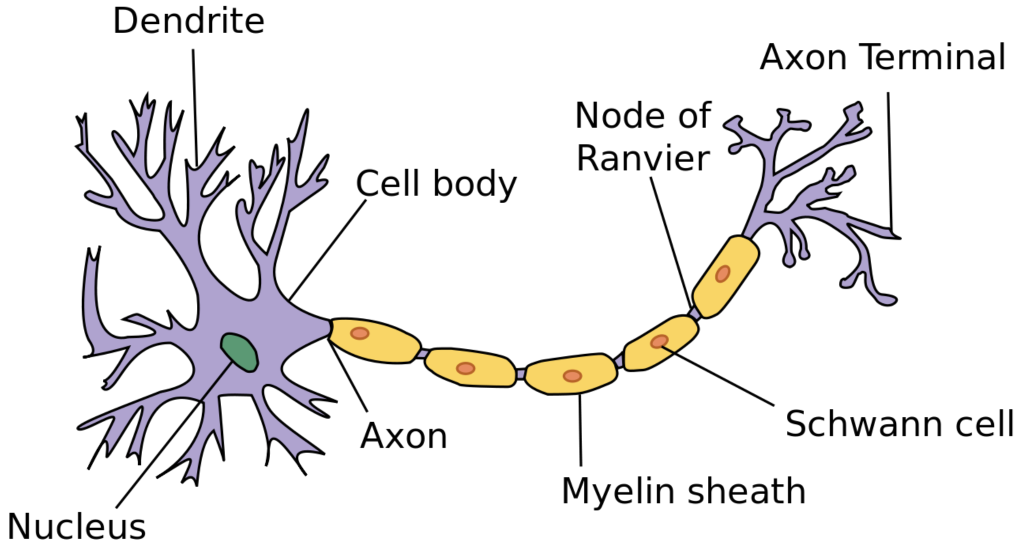
Artificial Neural Networks

A computational system
comprised of layers and each layer
is built of interconnected perceptrons

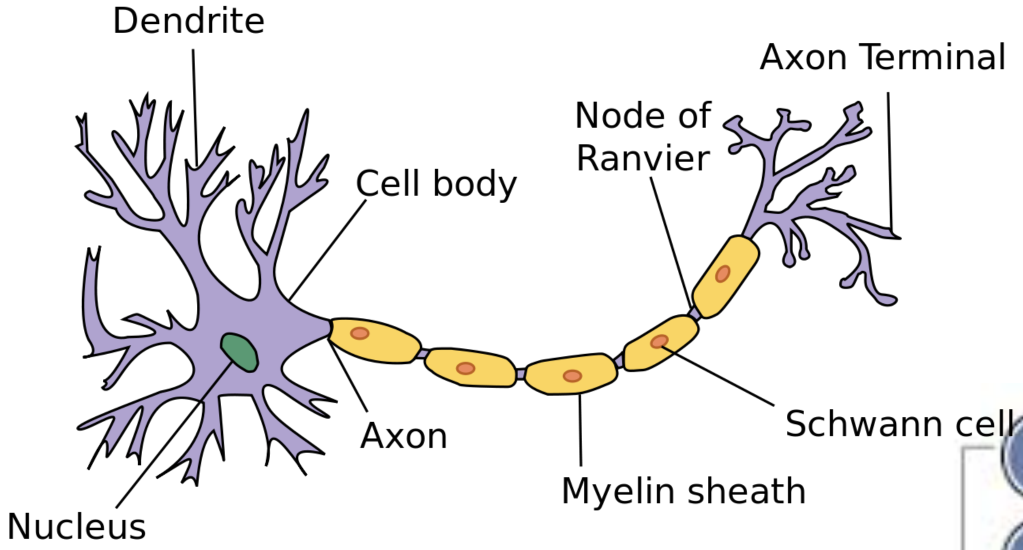
Artificial Neural Networks

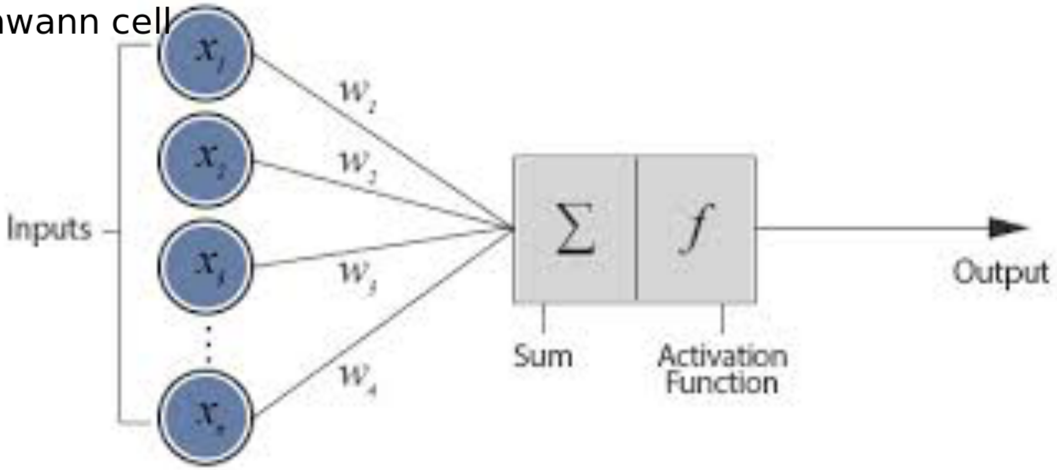
Built to model the animal nervous system

ARTIFICIAL NEURAL NETWORK



ARTIFICIAL NEURAL NETWORK





Artificial Neural Networks

A computational system
comprised of layers and each layer
is built of interconnected perceptrons

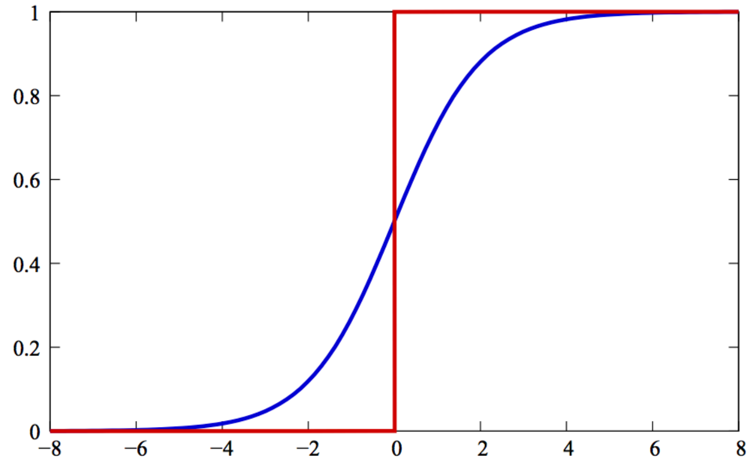
Single Perceptron

Takes in input and uses an activation function in order to output

Single Perceptron

$$f_{log}(z) = \frac{1}{1 + e^{-z}}$$

f_{log} is called logistic function



NOTE:
A single perception can be like a logistic regression in and of itself!

Takes in input and uses an activation function in order to output

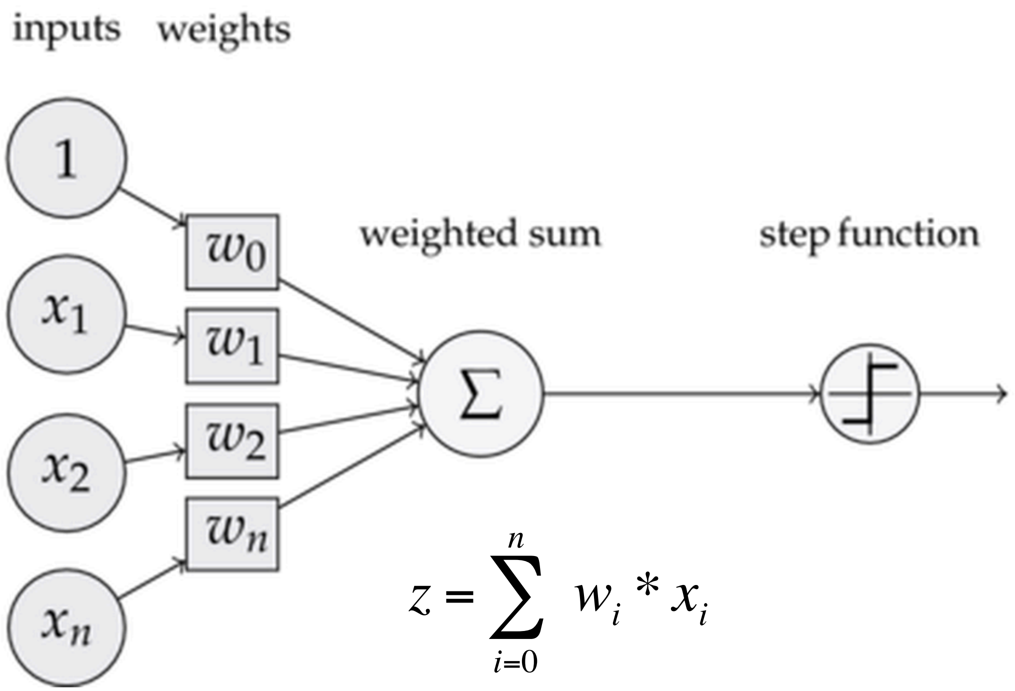
Single Perceptron

But what is z? A weighted sum on the inputs!

$$z = \sum_{i=0}^{n} w_i * x_i$$

Where w is the weight on input x

Single Perceptron



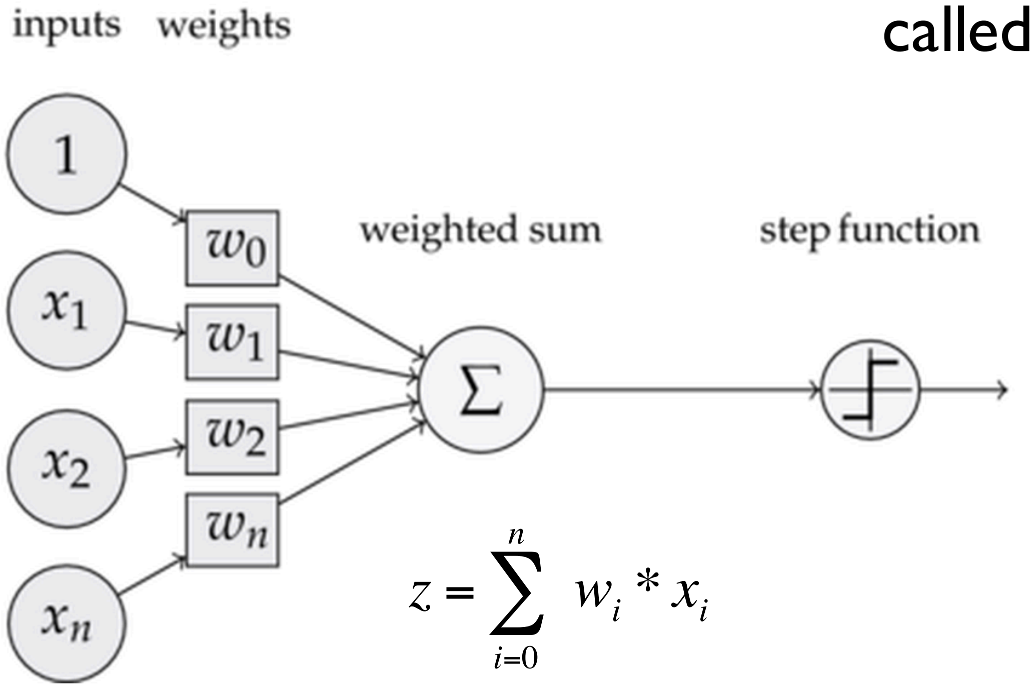
$$z = \sum_{i=0}^{n} w_i * x_i$$

$$f_{log}(z) = \frac{1}{1 + e^{-z}}$$

f_{log} is called logistic function

Single Perceptron

If f(z) if above a threshold, generally called theta, then the neuron "fires"



$$z = \sum_{i=0}^{n} w_i * x_i$$

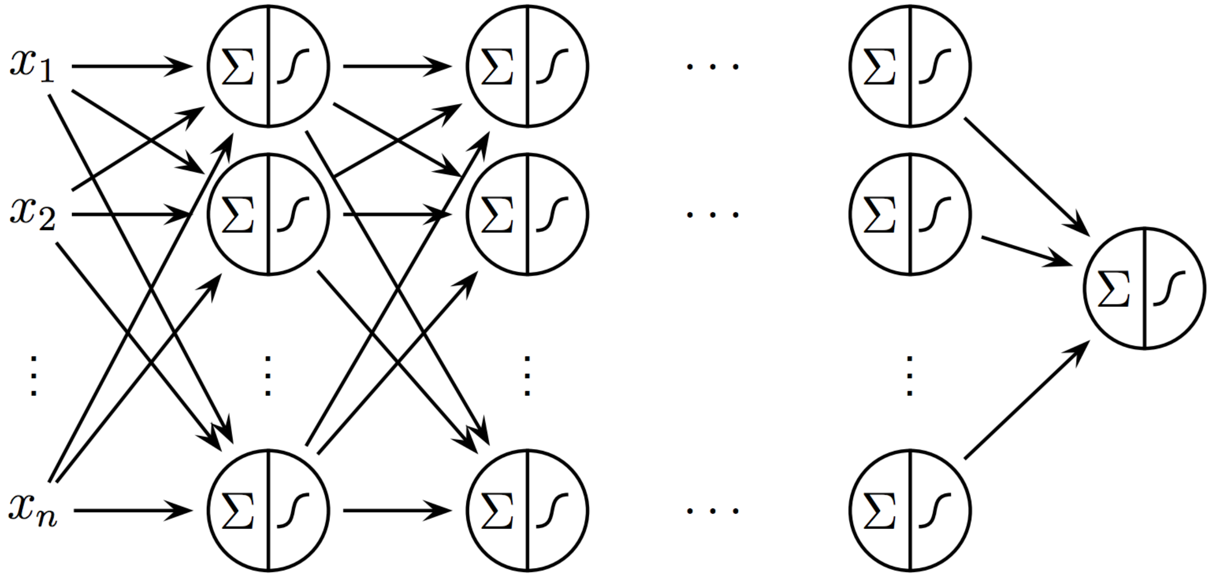
$$f_{log}(z) = \frac{1}{1+e^{-z}}$$

f_{log} is called logistic function

Artificial Neural Networks are also known as multi layer perceptrons

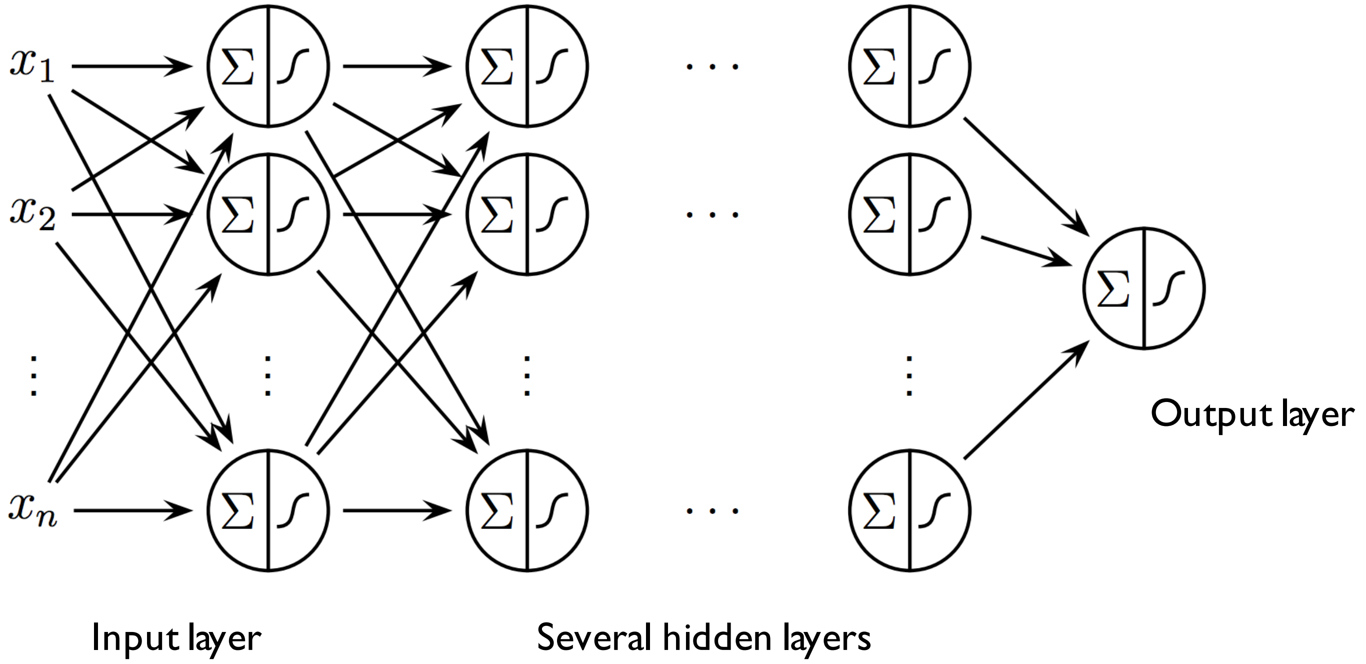
ARTIFICIAL NEURAL NETWORK

A multi layer perceptrons (MLP) is a finite acyclic graph. The nodes are neurons with logistic activation.



ARTIFICIAL NEURAL NETWORK

A multi layer perceptrons (MLP) is a finite acyclic graph. The nodes are neurons with logistic activation.



But how does it learn?!

Back-Propagation

As we train the model we update the sigmoid function weights in order to get the best predictions possible

If an observation goes through the model and is outputted as False when it should have been True, The logistic functions in the single perceptrons are changed slightly

Pros

- Online model (updates as you go)
  - Doesn’t need to be fit all of the time
- Very fast predictions
- Can approximate almost any type of function
- Can be used in a supervised and unsupervised manner
- Super cool

Cons

- Requires many training samples to be considered good
- Hard to describe what is happening
- Requires a lot of hardware / computation power
- Slow to train
- Sklearn only has unsupervised version
- Other versions are difficult to use

The most advanced ANN’s use thousand’s of neurons which is a lot right?

The most advanced ANN's use thousand's of neurons which is a lot right?

Sure but my dog has billions…….

http://deepdreamgenerator.com/

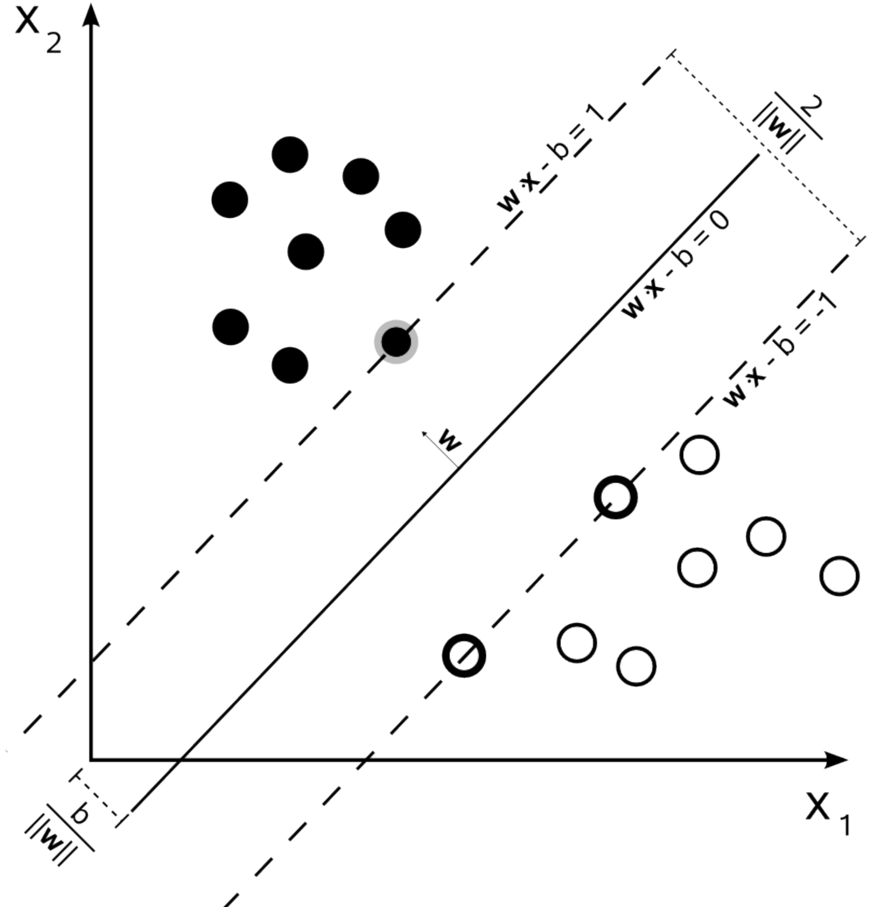
Google uses a supervised neural network to recognize content in photos.

http://deepdreamgenerator.com/

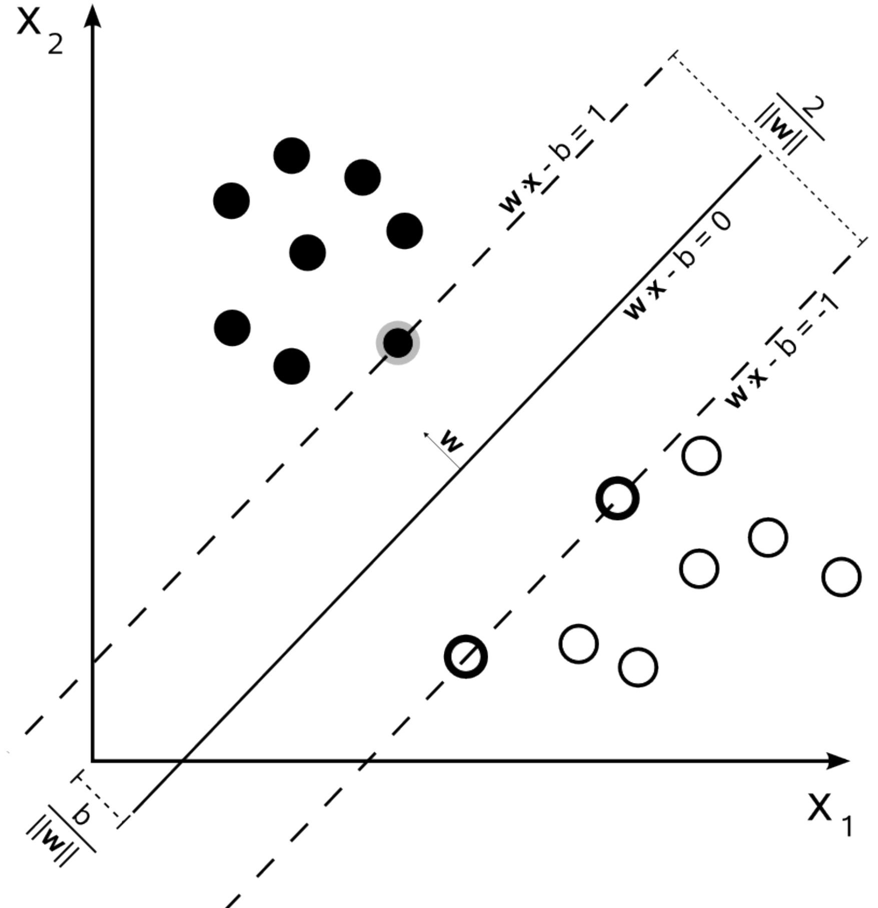
Turns out if you input an image you can ask the neural network to try and "re-create" the image as well

II. SUPPORT VECTOR MACHINES

Constructs a hyperplane to separate classes in space


x_2
x_1
$\mathbf{w}\cdot\mathbf{x} - b = 1$
$\mathbf{w}\cdot\mathbf{x} - b = 0$
$\mathbf{w}\cdot\mathbf{x} - b = -1$
$\frac{2}{\|\mathbf{w}\|}$
$\frac{b}{\|\mathbf{w}\|}$
$\mathbf{w}$

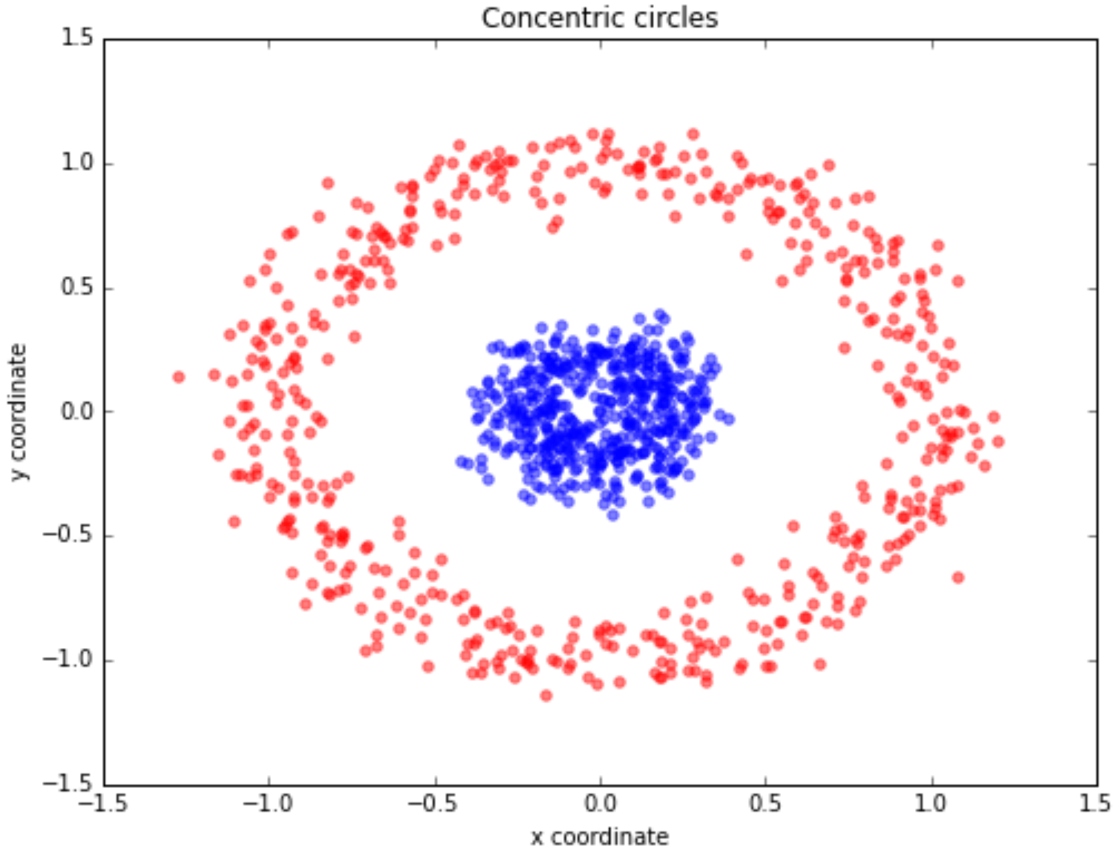
We want to maximize the width of the margin

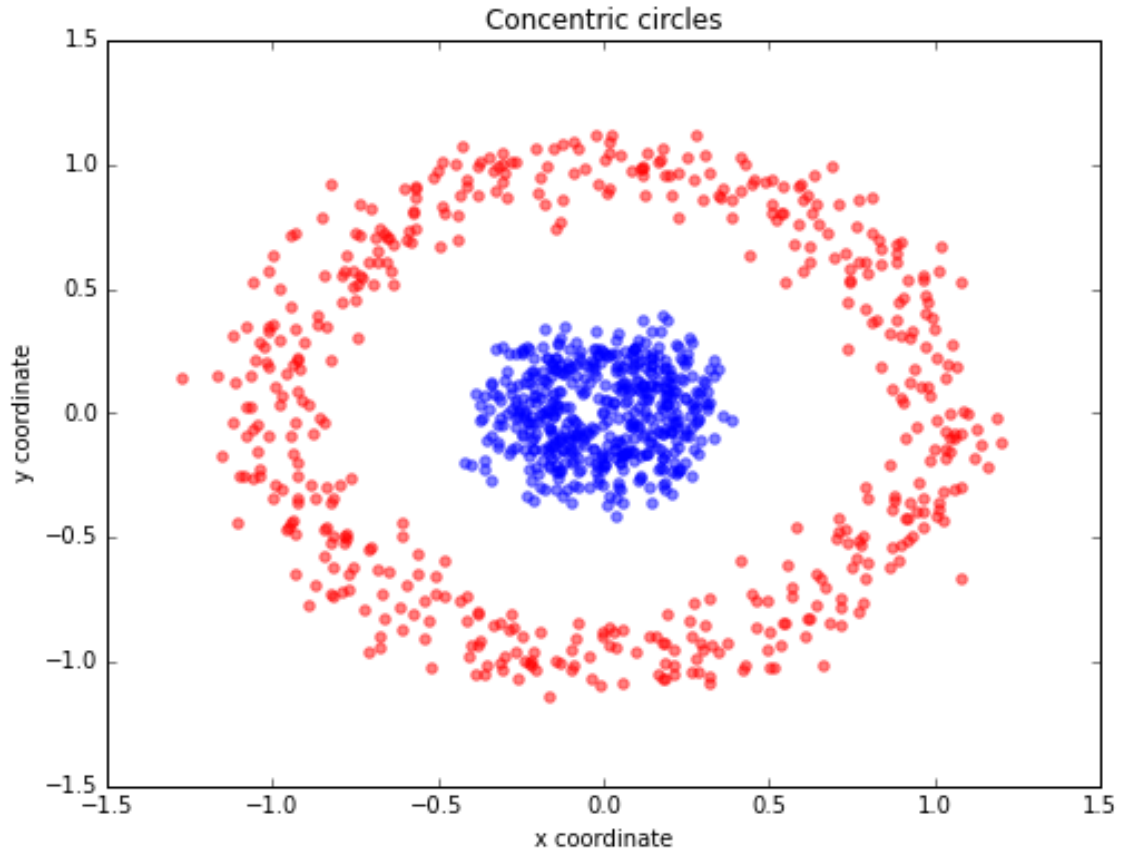

x_2
x_1
$\mathbf{w}\cdot\mathbf{x} - b = 1$
$\mathbf{w}\cdot\mathbf{x} - b = 0$
$\mathbf{w}\cdot\mathbf{x} - b = -1$
$\frac{2}{\|\mathbf{w}\|}$
$\frac{b}{\|\mathbf{w}\|}$
$\mathbf{w}$

What if there is no easy hyperplane?

What if there is no easy hyperplane?

Walk with me on this,
a mathematical journey


Concentric circles
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
y coordinate
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
x coordinate



Pretty much no hyperplane will separate this out, but what if we could add a third dimension?

Q. OK fine, but what if I have 100 predictors? How many dimensions should I project into?

A. An arbitrary amount, possible infinite..

Q. OK fine, but what if I have 100 predictors? How many dimensions should I project into?

A. An arbitrary amount, possible infinite..

OK but this can take time..

Kernel Trick

We assume a certain shape of the data and the kernel trick saves us MASSIVE computation time

Kernel Trick

Example:

Linear	(assumes a linear boundary)
Poly	(assumes a curved boundary)
Gaussian	(assumes a spherical boundary)

Pros

- Very fast training and predicting with kernel trick
- Built on solid mathematical foundation (unlike ANN)
- Very common and in sklearn

Cons

- A lot of “guess work” with kernels
- Hard to grasp math behind it (ok if you accept the black box)