

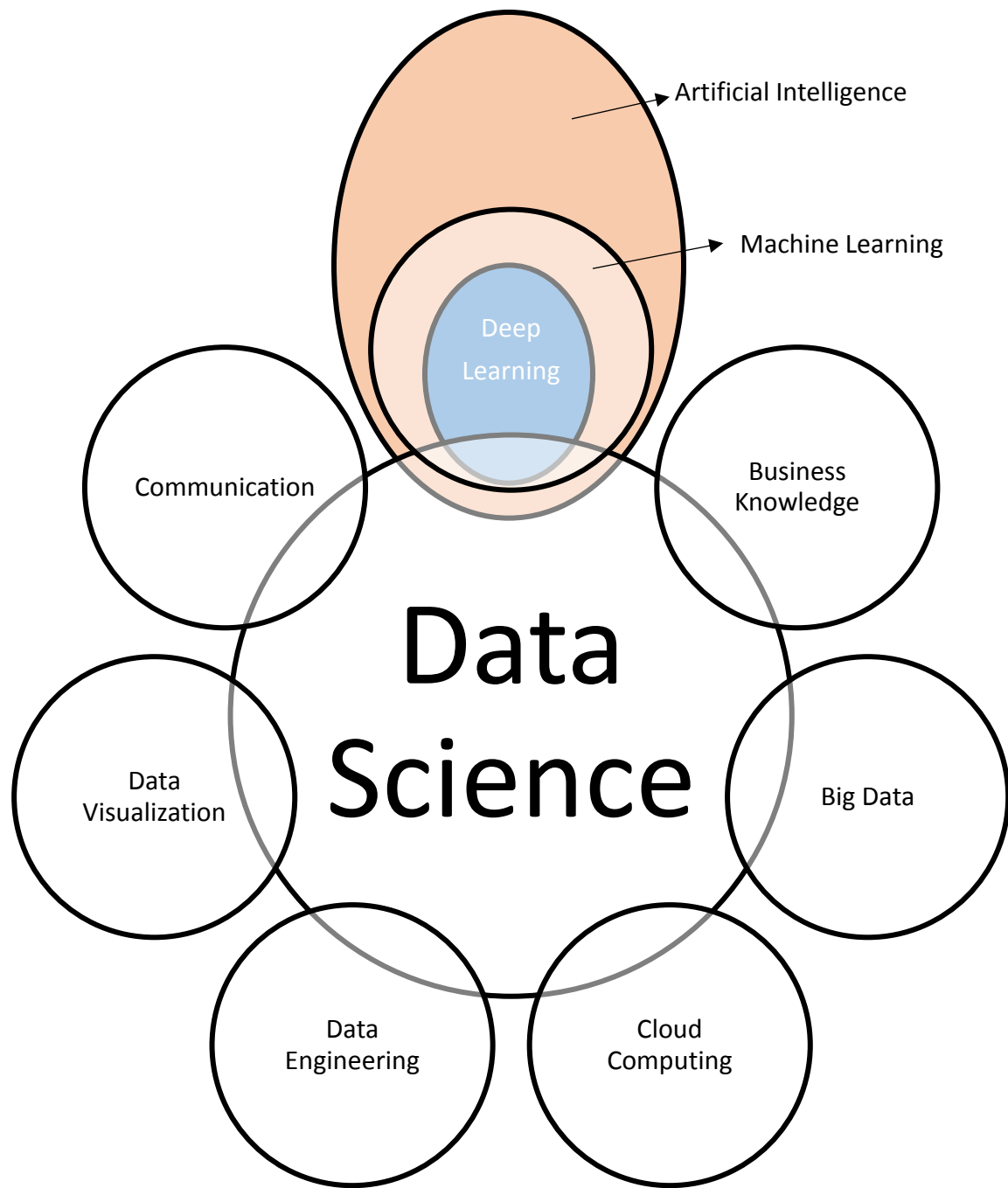
# Dipping your toes into Deep Learning!

Devashish Khatwani

October 2017

Data Science, Machine Learning, Deep Learning,  
Artificial Intelligence??

I AM CONFUSED!



1

Deep Learning is a subfield of Machine Learning

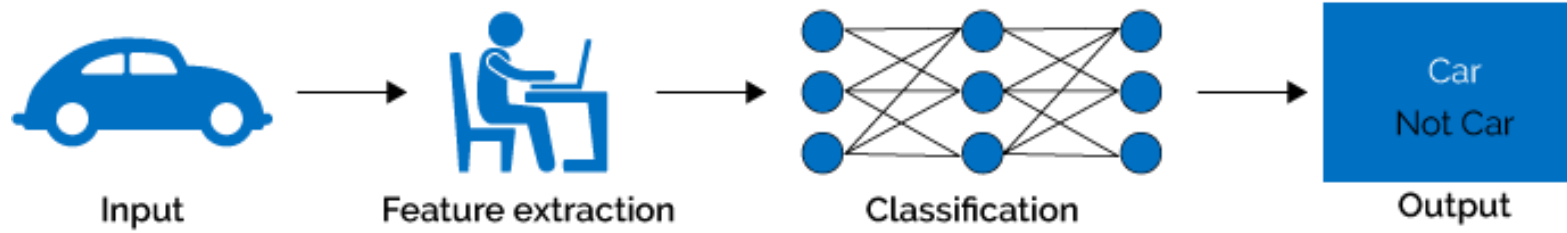
2

Machine Learning itself is a subfield of Artificial Intelligence

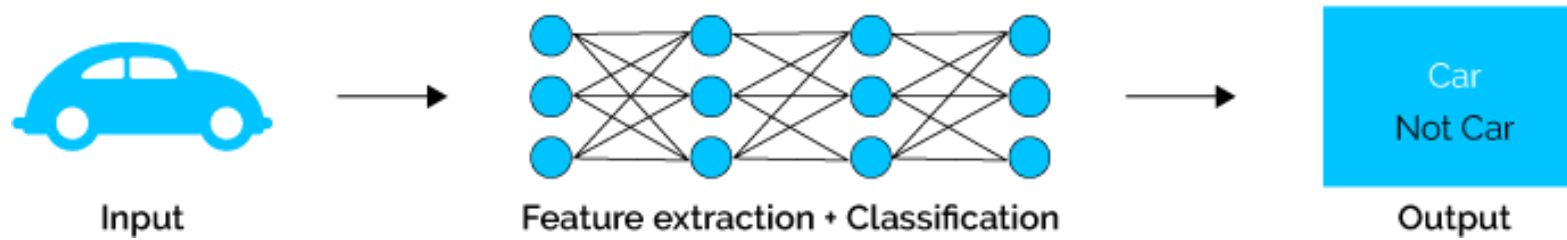
3

Data Science is a field which touches many subfields and is a more application oriented domain

# Machine Learning



# Deep Learning



1

No need of feature engineering

2

Bias Variance Tradeoff no longer applicable

3

Can handle massive amounts of data

4

Change in Train, Validation, Test dataset split mindset



Hardly a days goes by  
without a media mention



More Money flowing into AI based  
startups than ever before

TE Got a tip? [Let us know.](#) Follow Us [Facebook](#) [Twitter](#) [LinkedIn](#) [Google+](#) [YouTube](#) [Instagram](#) [Pinterest](#) [Snapchat](#) [Vimeo](#) [SoundCloud](#) [RSS](#)

News · Video · Events · Crunchbase [Message Us](#)  [Search](#)

[AdChoices](#)

**DISRUPT BERLIN** Time is running out on Early Bird savings for Disrupt Berlin. [Save 30% off tickets today](#)

cybernetics  
machine learning  
technology  
film  
Artificial Intelligence

**Kristen Stewart co-authored a paper on style transfer and the AI community lost its mind**  
Posted Jan 19, 2017 by [John Mannes \(@JohnMannes\)](#)

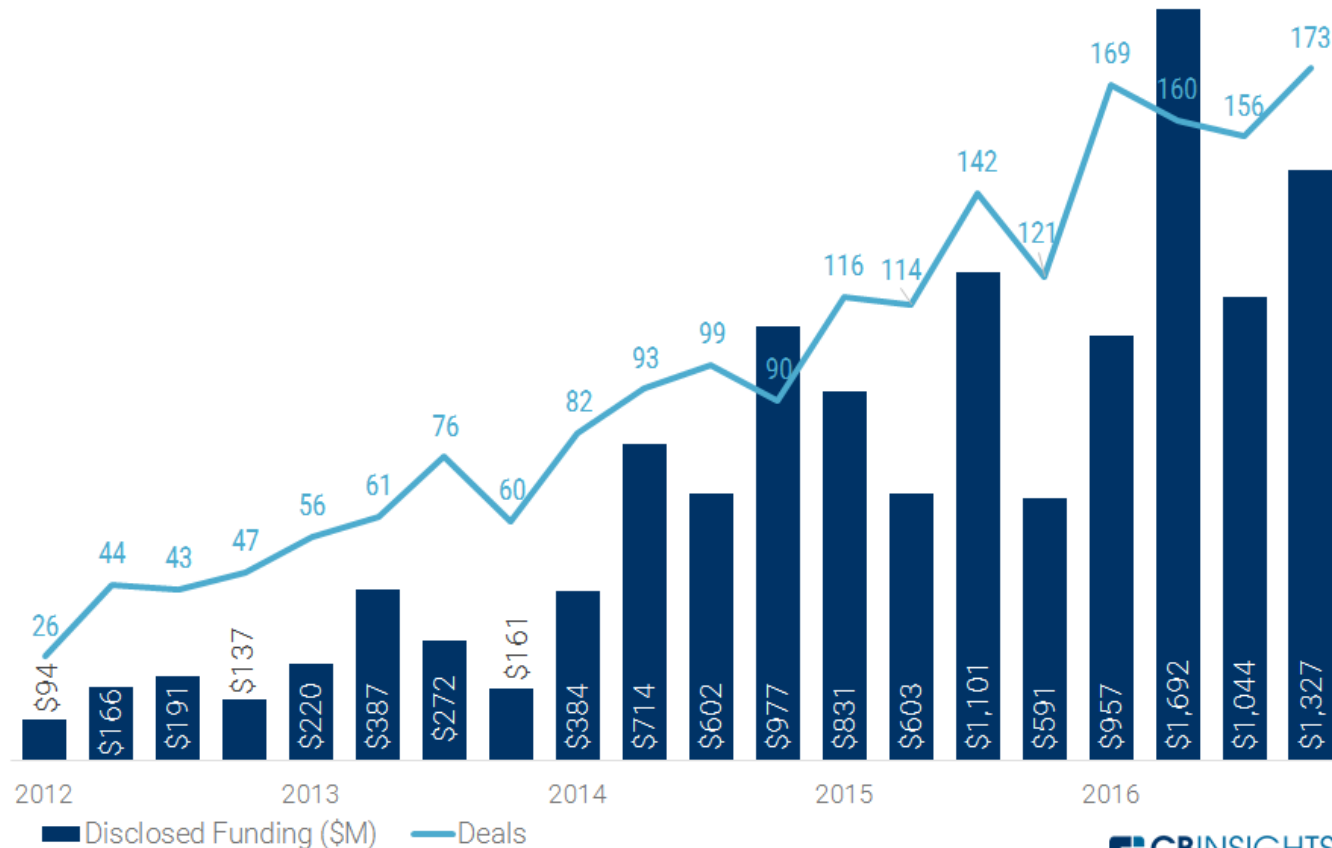
Popular Posts

[Download the new, completely redesigned](#)

[u=3](#) [u=4](#)



## AI QUARTERLY GLOBAL FINANCING HISTORY 2012 - 2016



Tech / #NewTech

SEP 18, 2017 @ 02:37 PM 2,381

## Deep Learning Could Finally Make Robots Useful

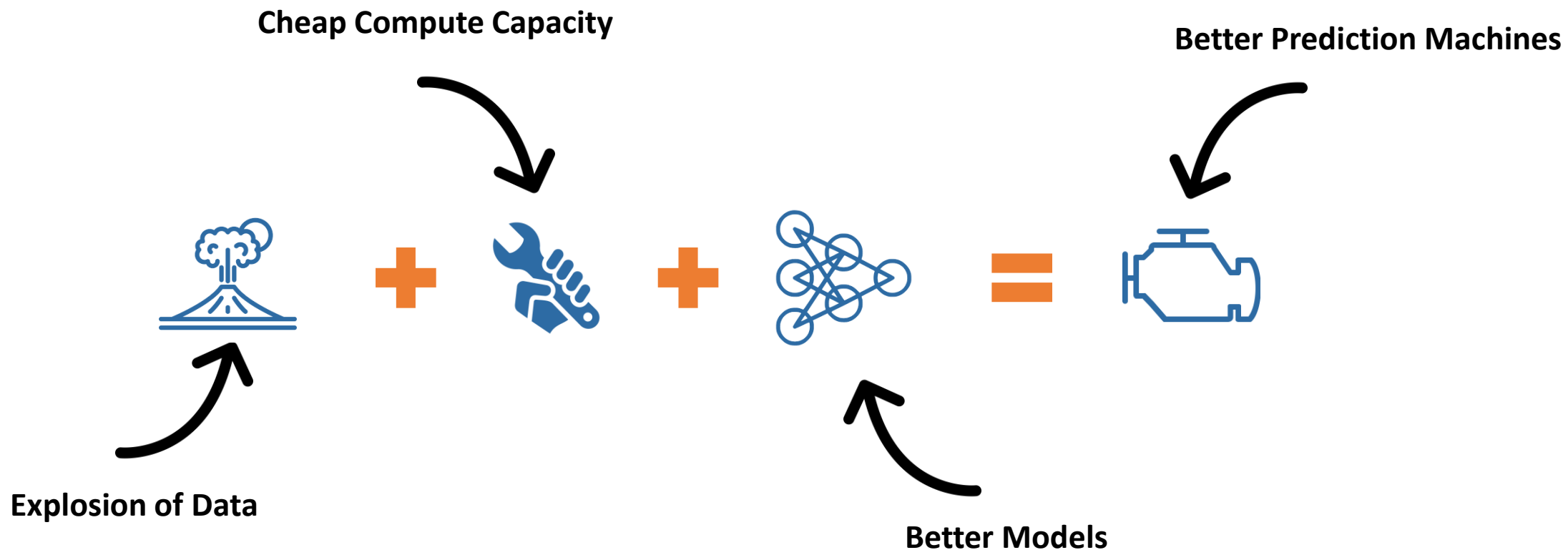


Aaron Tilley, FORBES STAFF  
[FULL BIO](#)

Why the hype, Bro?!

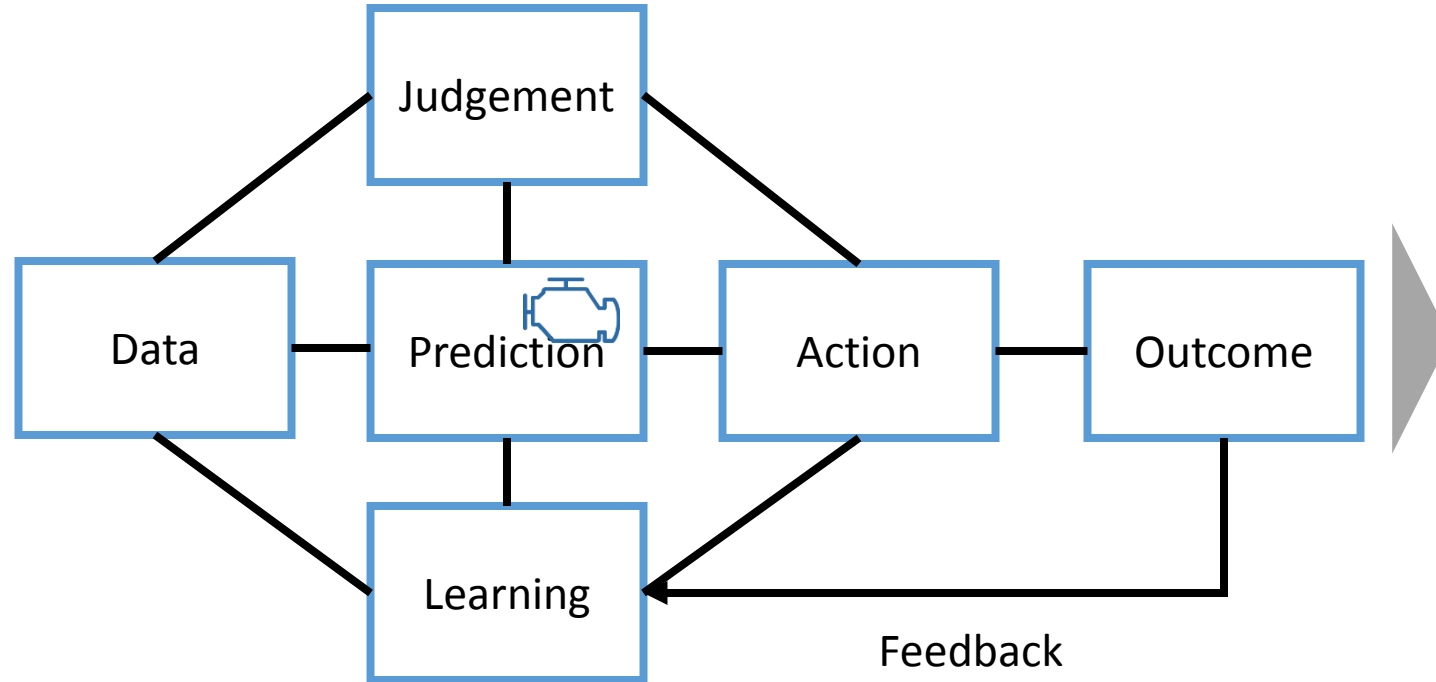


## Why Now?





## We have a better prediction machine, so what?



1

As the cost of a resource decreases we use more of it

2

The value of compliments goes up

3

The value of substitutes goes down

4

We start using the resource in places where we never used it before

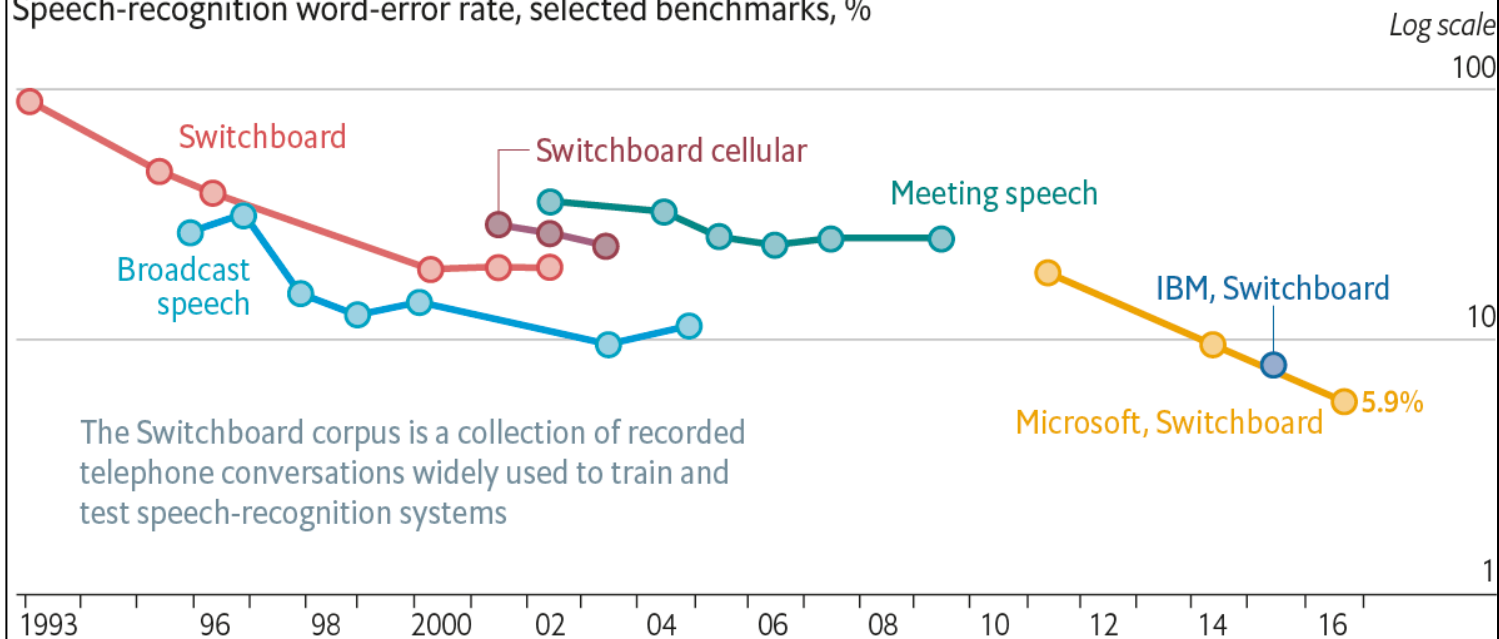




# Deep Learning Model have beaten Humans at Audio and Image recognition tasks in last 3 years

## Loud and clear

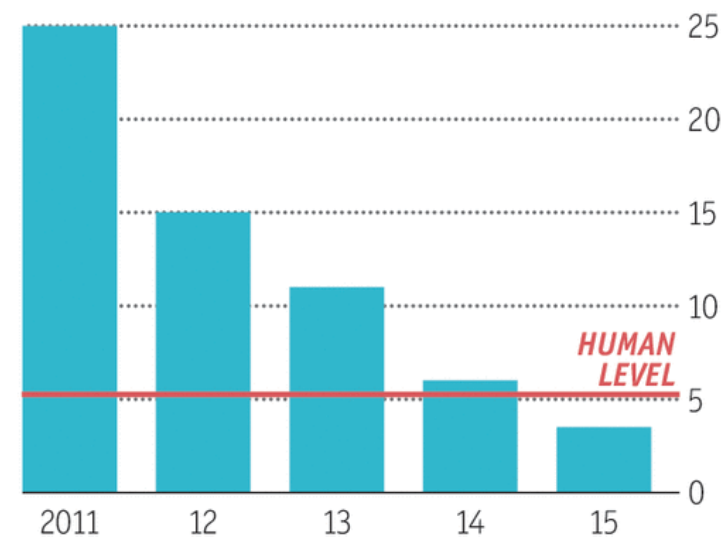
Speech-recognition word-error rate, selected benchmarks, %



Sources: Microsoft; research papers

## Ever cleverer

Error rates on ImageNet Visual Recognition Challenge, %

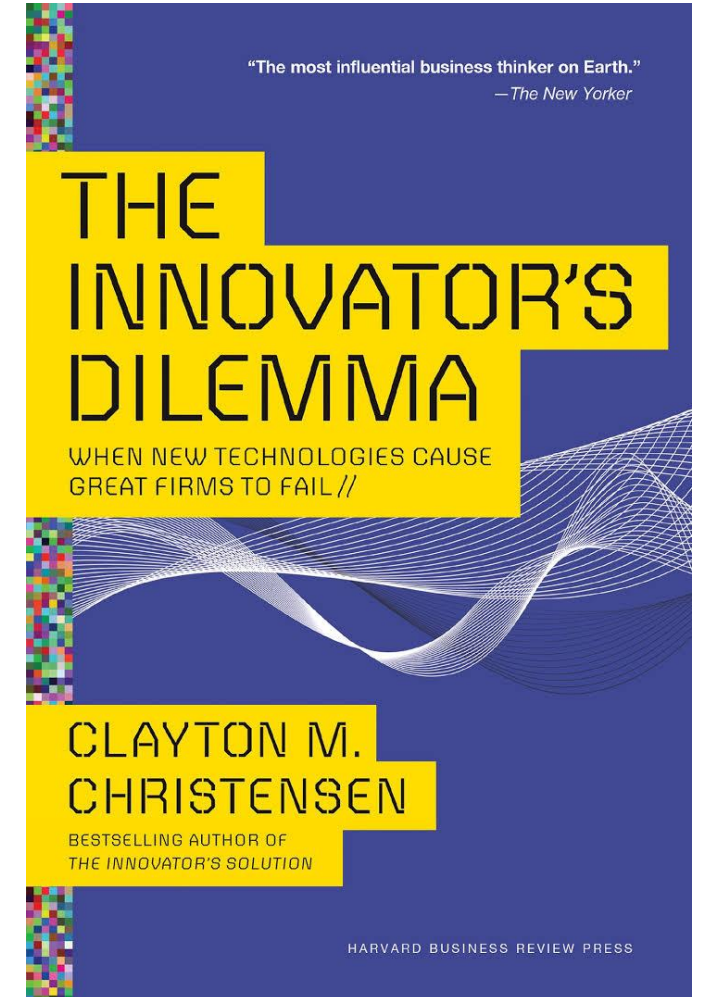
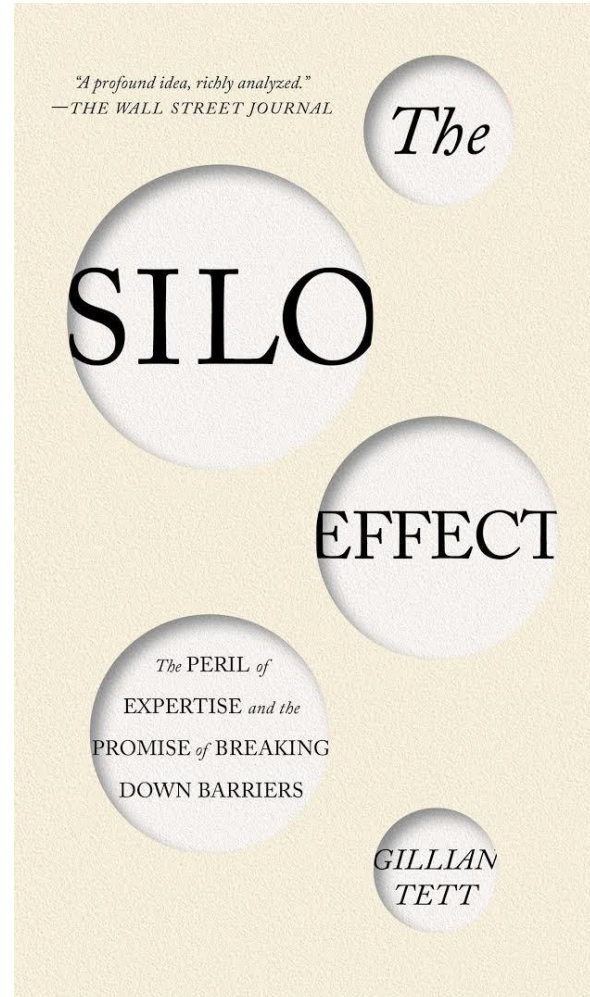
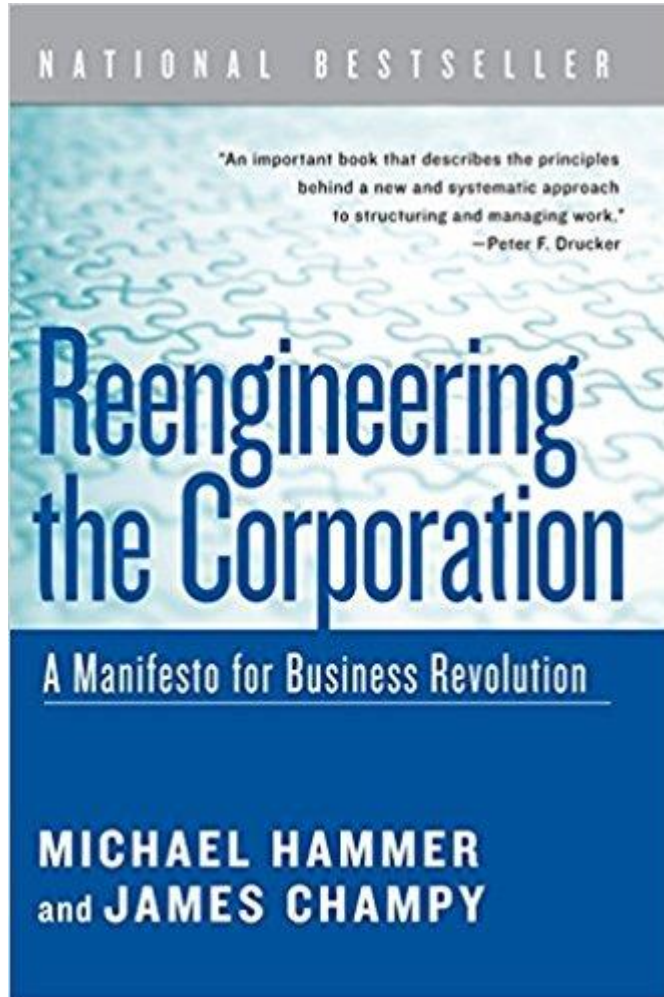


Sources: ImageNet; Stanford Vision Lab

Economist.com



## Tom's Christmas Reading List

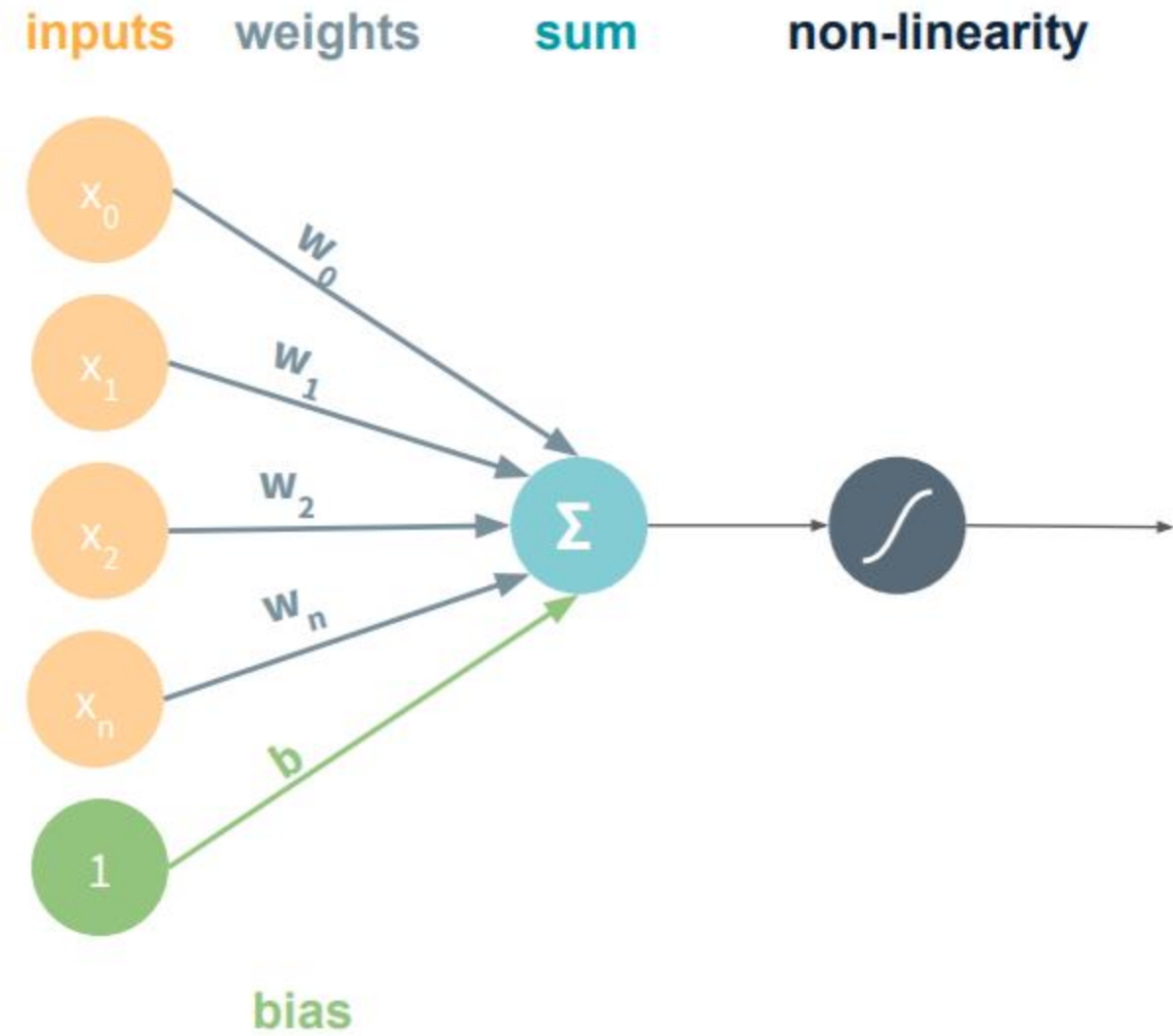


Lets get to the Meat and Potatoes now!

First the Potatoes(Pretty Pictures).



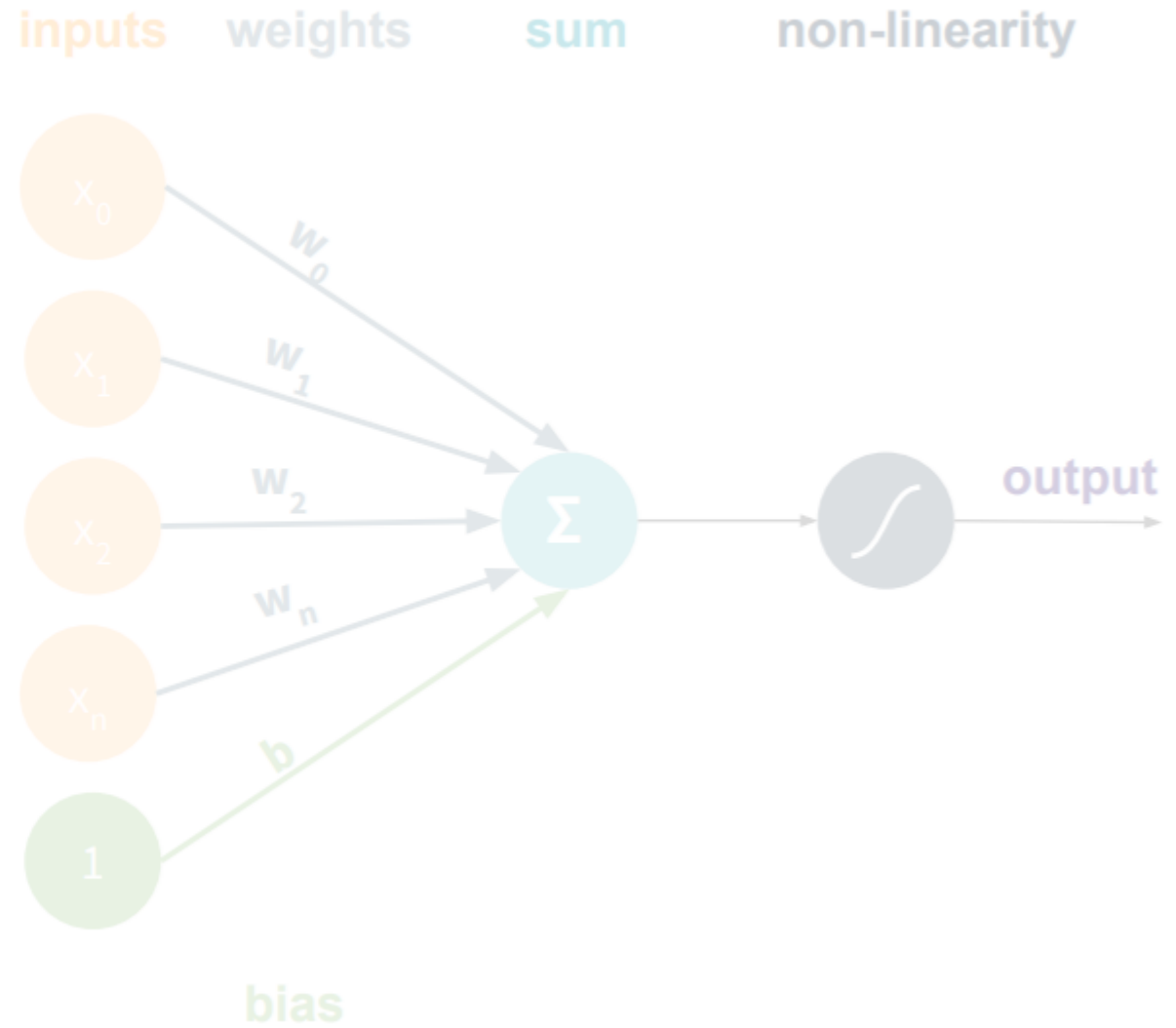
# The Perceptron





## The Perceptron: The Forward Pass

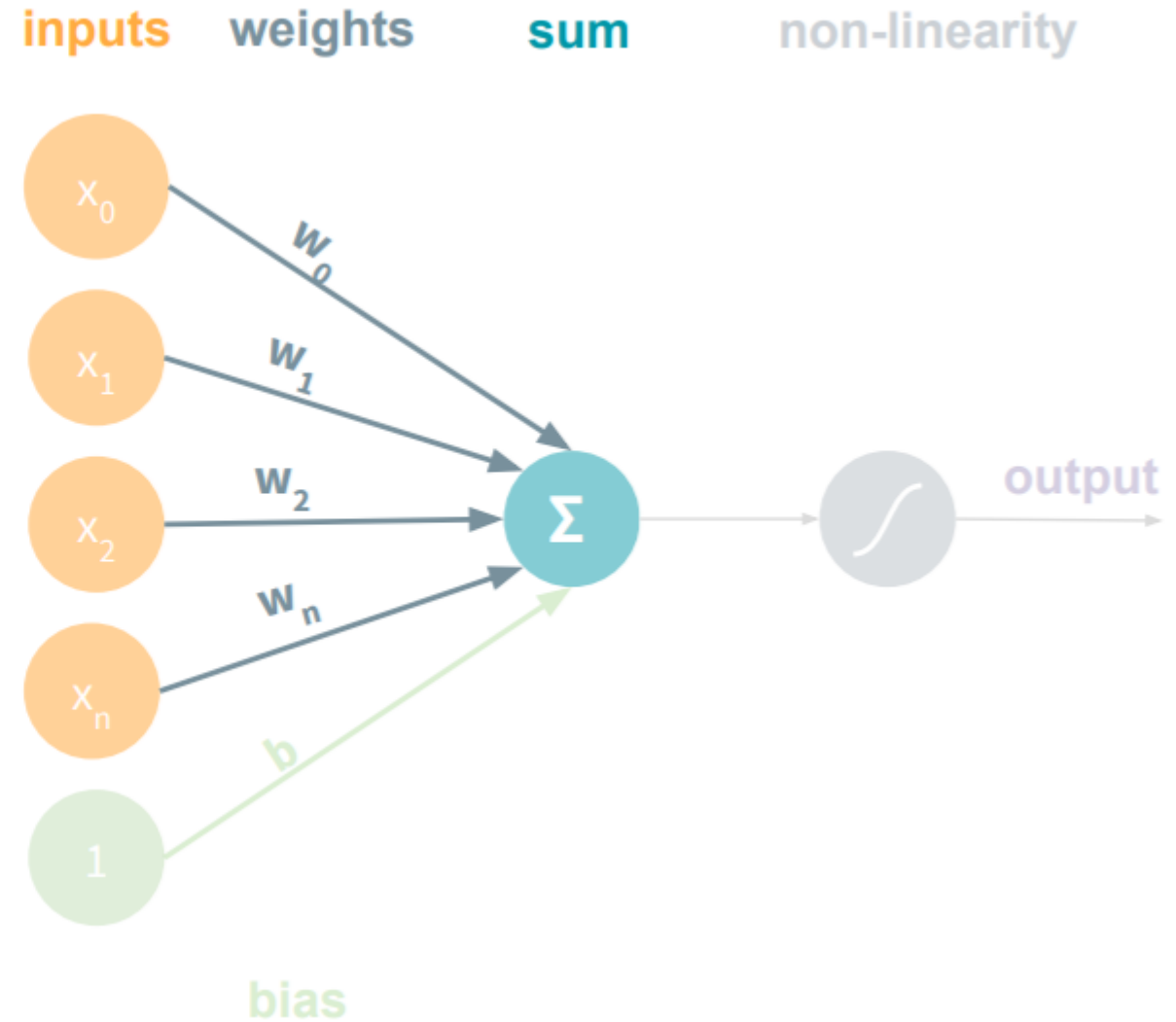
$output =$





## The Perceptron: The Forward Pass

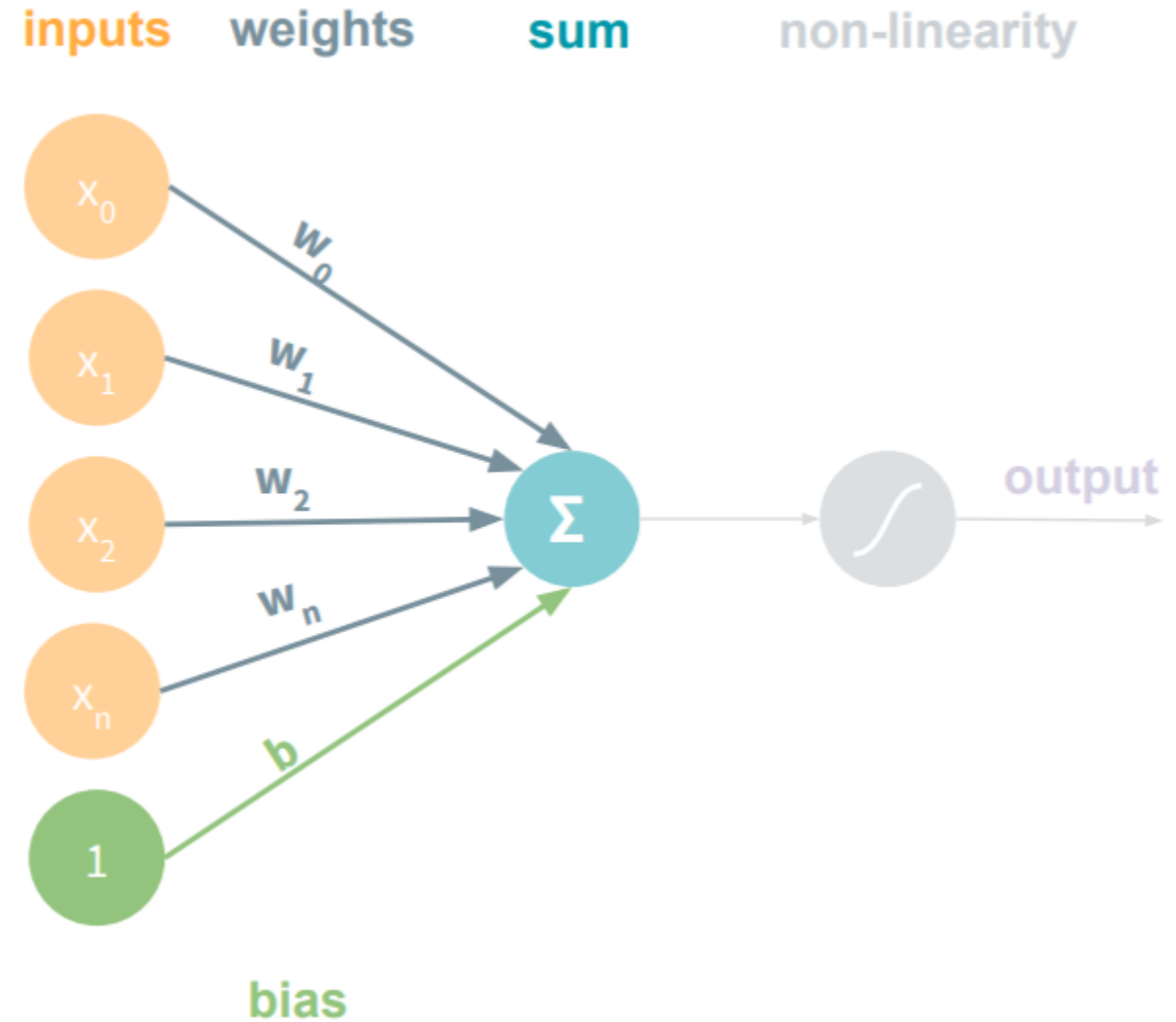
$$\text{output} = \sum_{i=0}^N x_i * w_i$$





## The Perceptron: The Forward Pass

$$\text{output} = \left( \sum_{i=0}^N x_i * w_i \right) + b$$

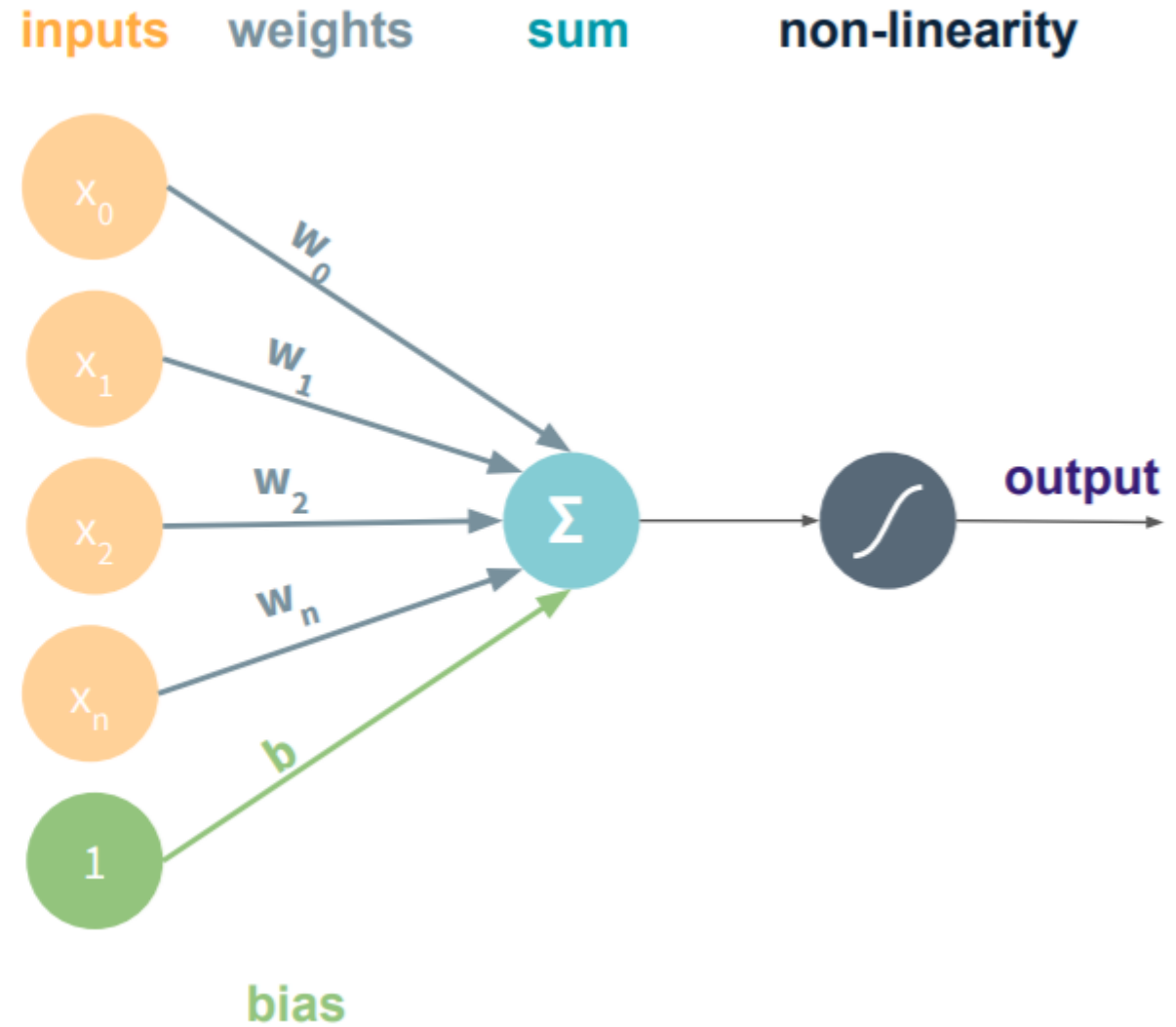






## The Perceptron: The Forward Pass

$$output = g\left(\sum_{i=0}^N x_i * w_i + b\right)$$



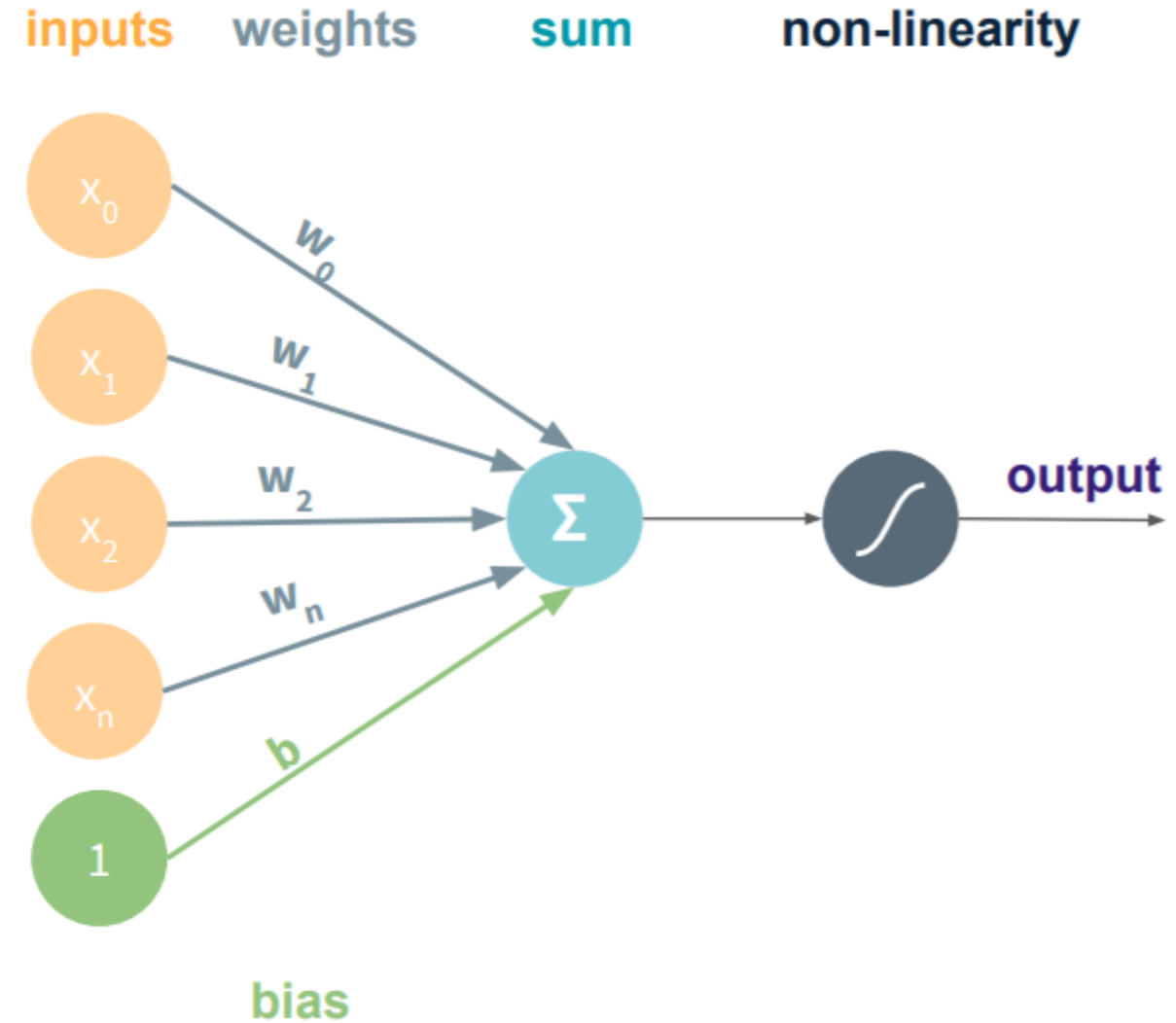


## The Perceptron: The Forward Pass

$$\text{output} = g(XW + b)$$

$$X = x_0, x_1, \dots, x_n$$

$$W = w_0, w_1, \dots, w_n$$





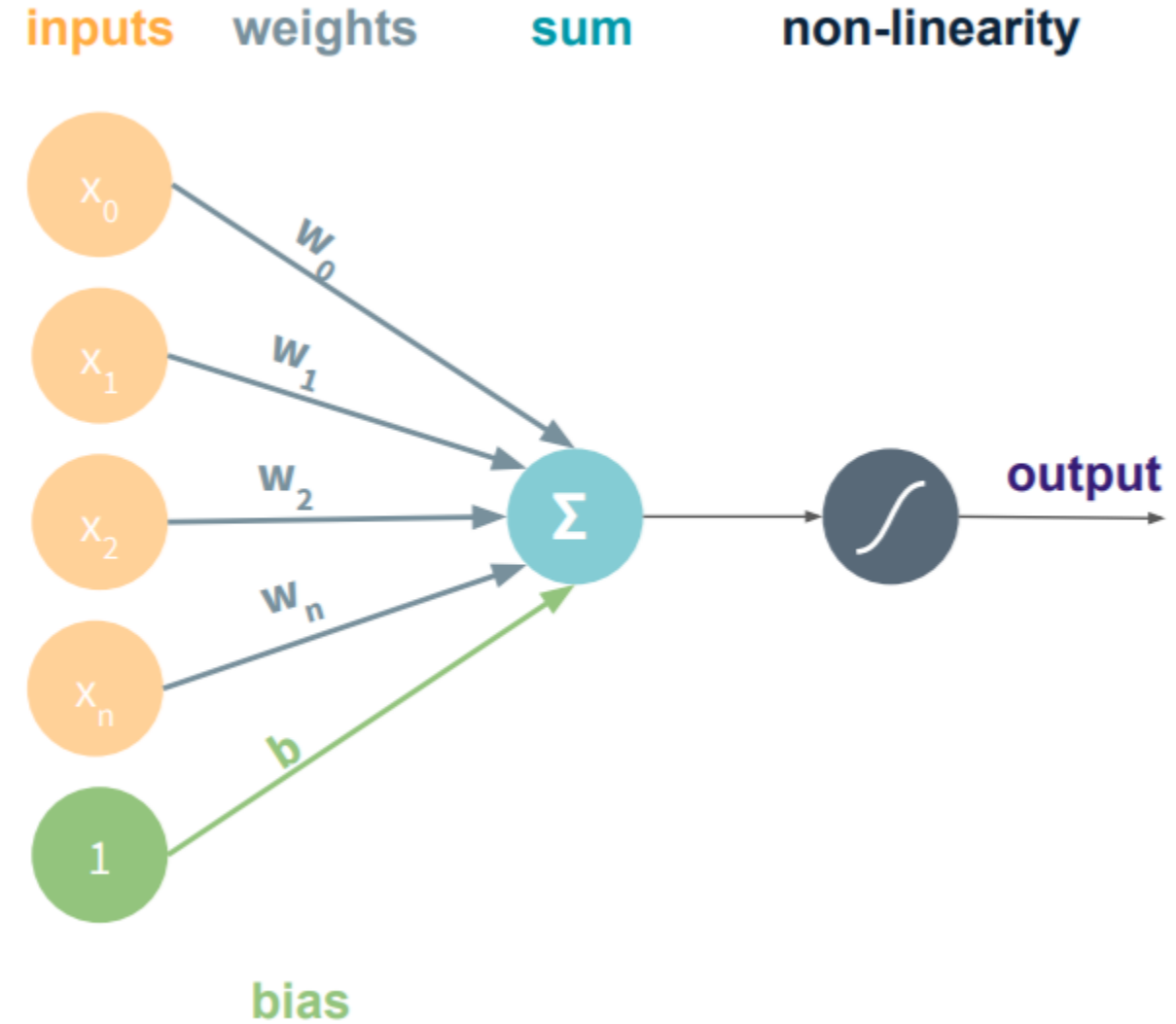
## The Perceptron: The Forward Pass

### Activation Function

$$\text{output} = g(XW + b)$$

$$X = x_0, x_1, \dots, x_n$$

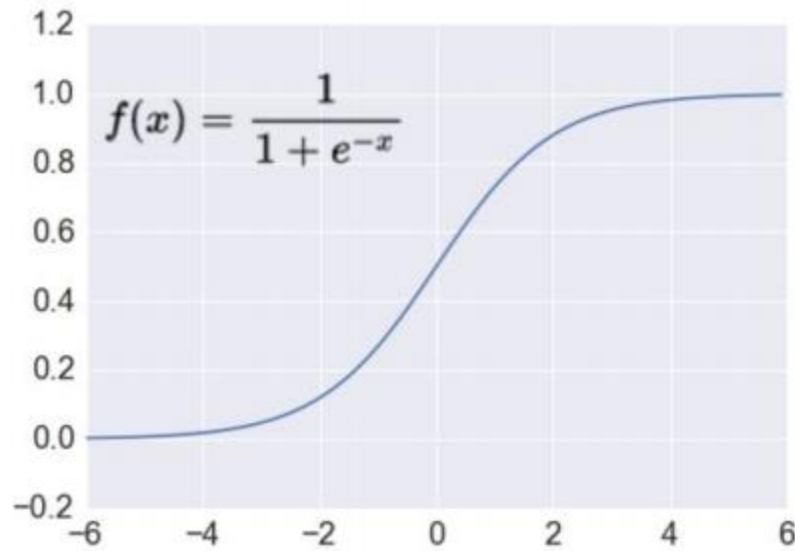
$$W = w_0, w_1, \dots, w_n$$



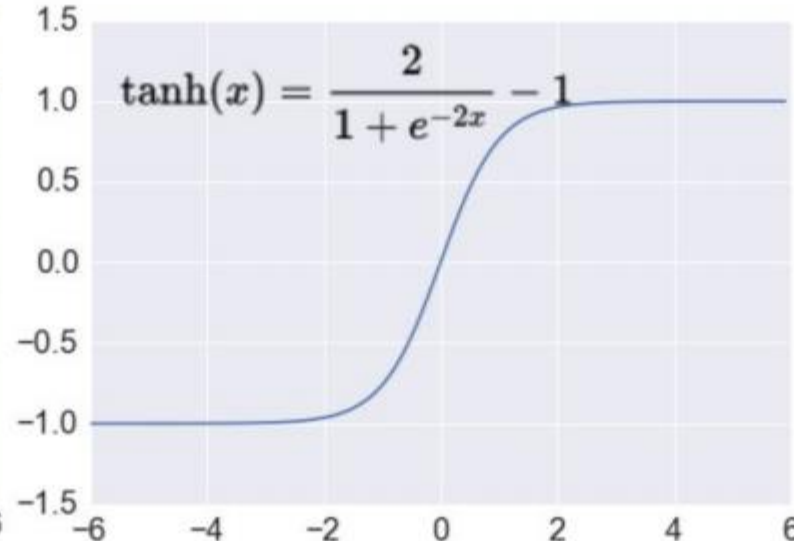


## Common Activation Functions

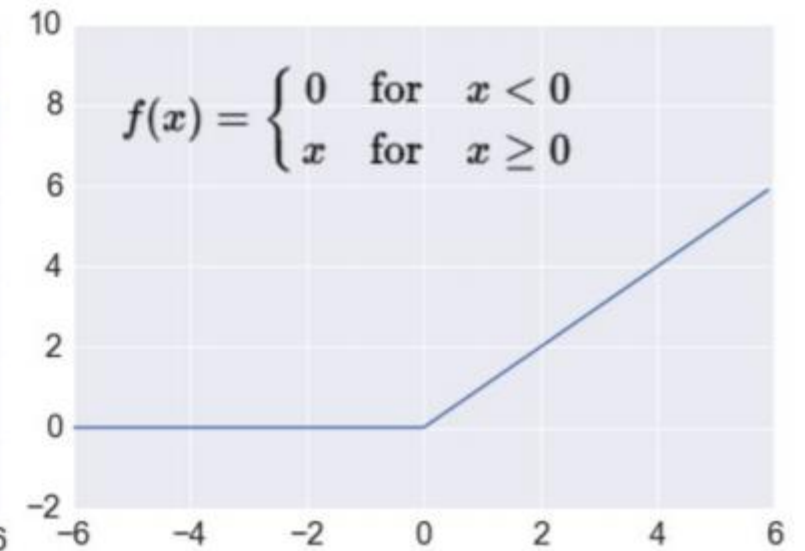
Sigmoid



TanH



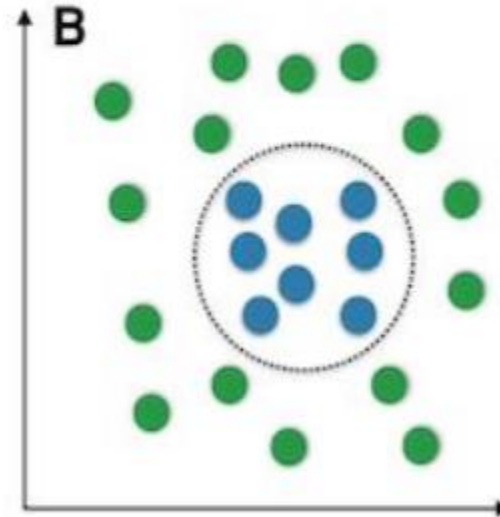
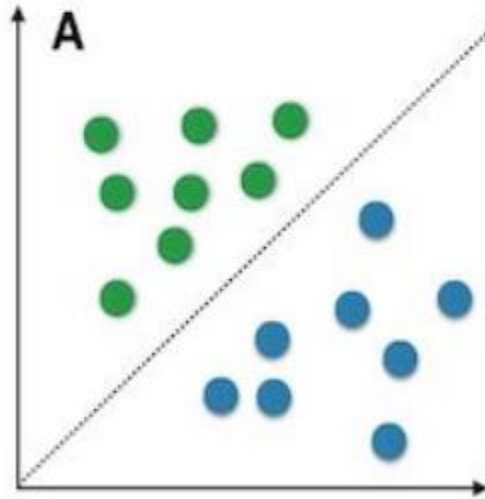
ReLU





## Importance of Activation Functions

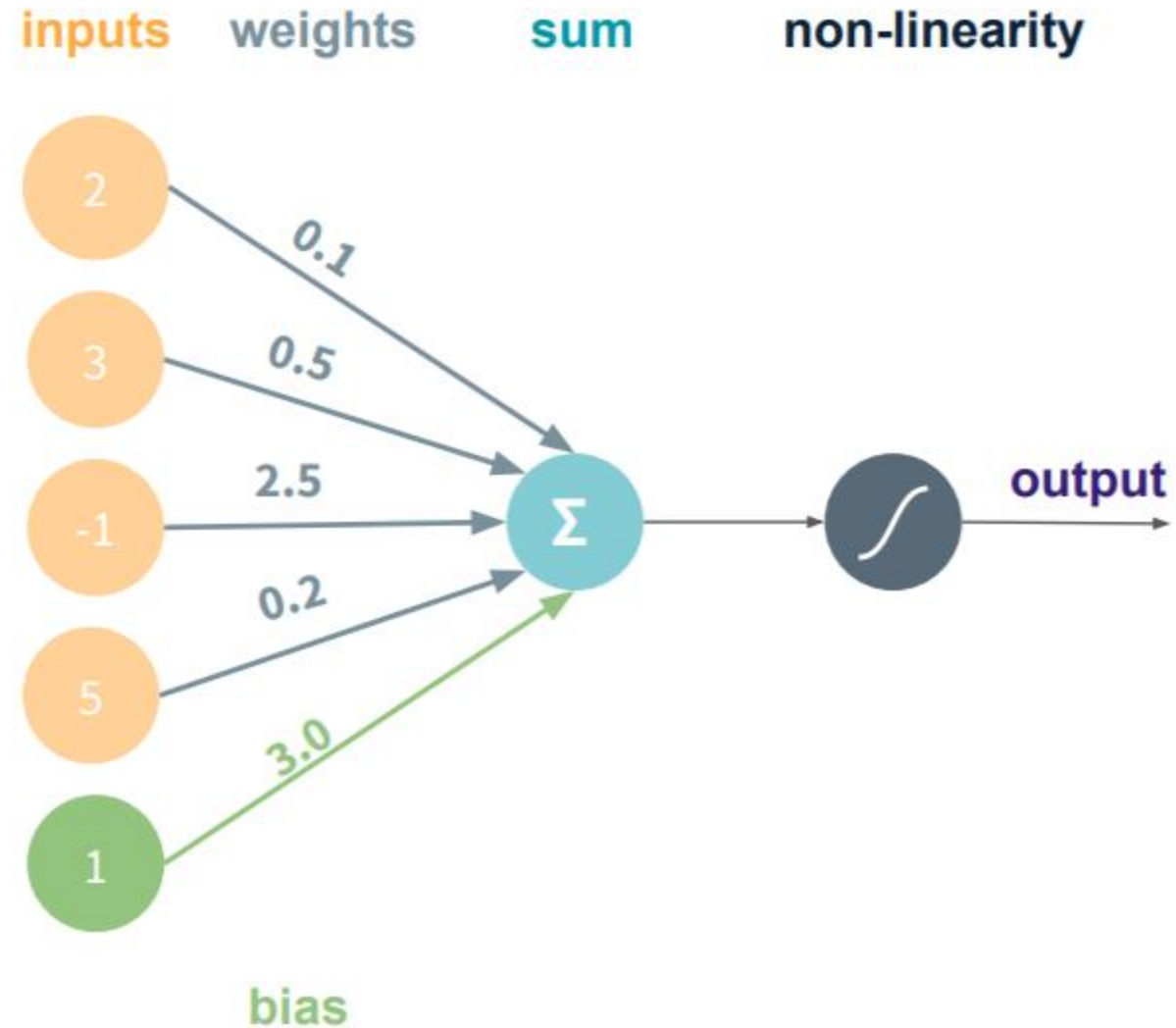
- Activation functions add non-linearity to our network's function
- Most real-world problems + data are **non-linear**





## The Perceptron: The Forward Pass

$$\text{output} = g(XW + b)$$





## The Perceptron: The Forward Pass

$$\text{output} = g(\text{$$

$$(2 \cdot 0.1) +$$

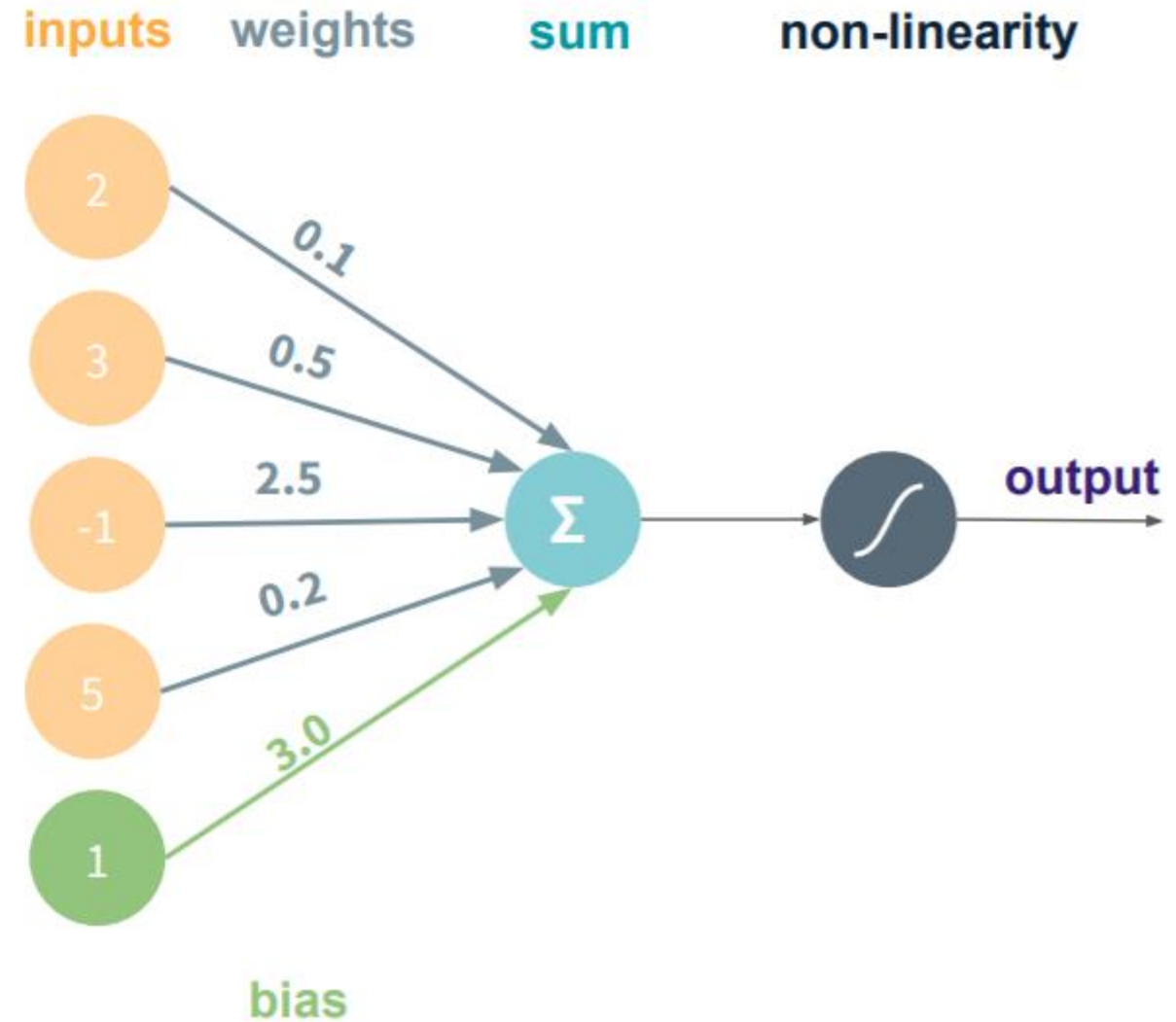
$$(3 \cdot 0.5) +$$

$$(-1 \cdot 2.5) +$$

$$(5 \cdot 0.2) +$$

$$(1 \cdot 3.0)$$

)

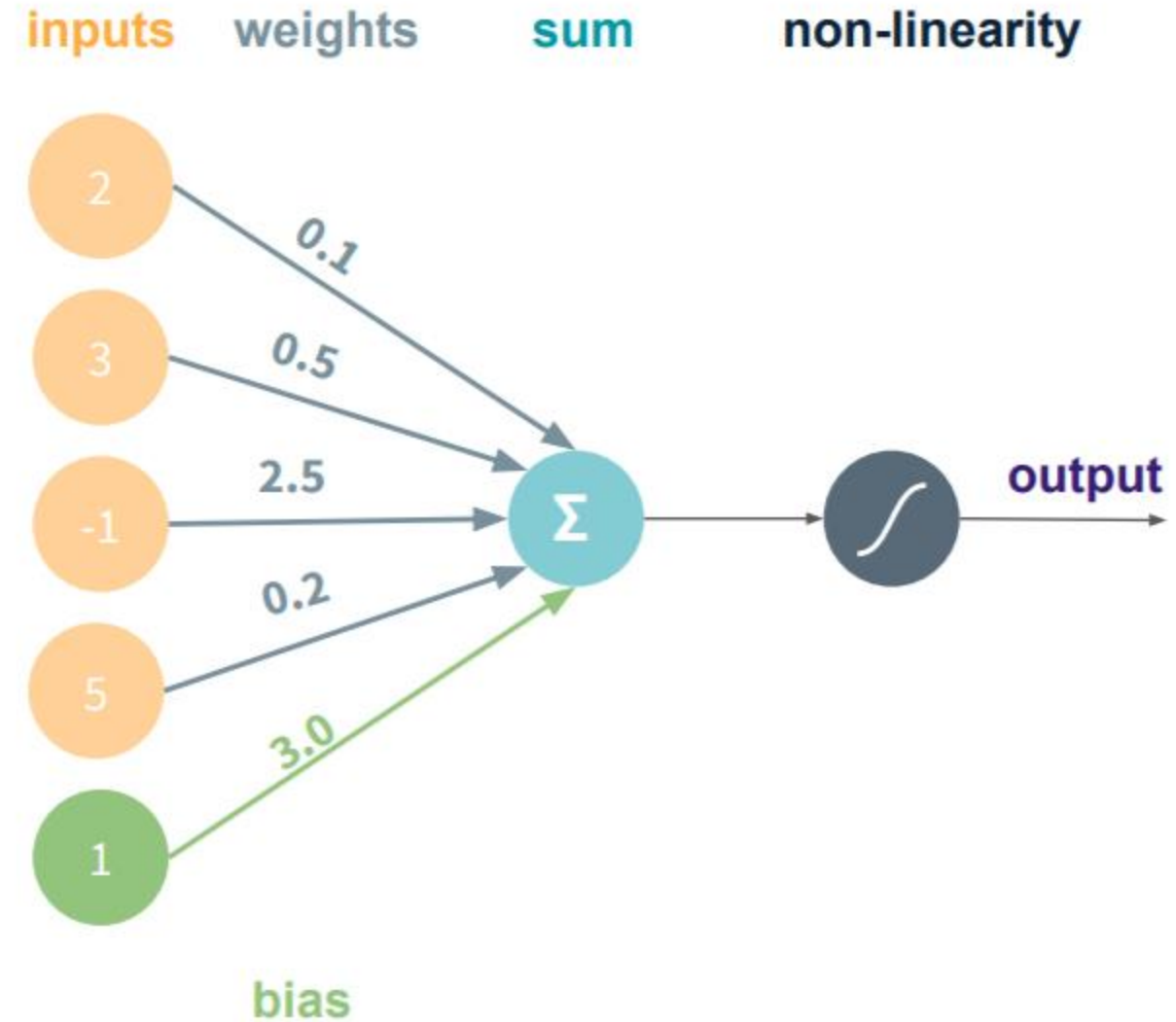




## The Perceptron: The Forward Pass

$$\text{output} = g(3.2) = \sigma(3.2)$$

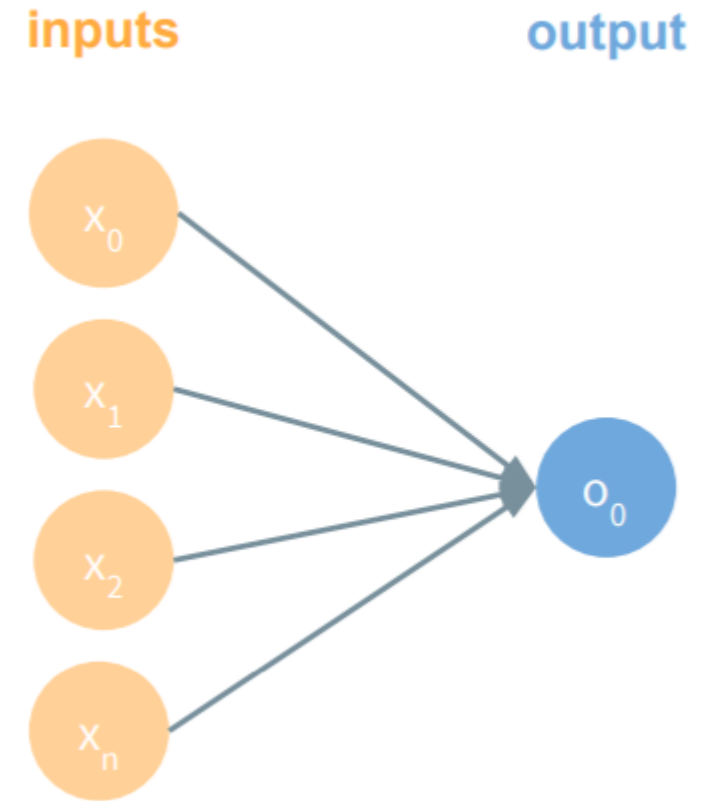
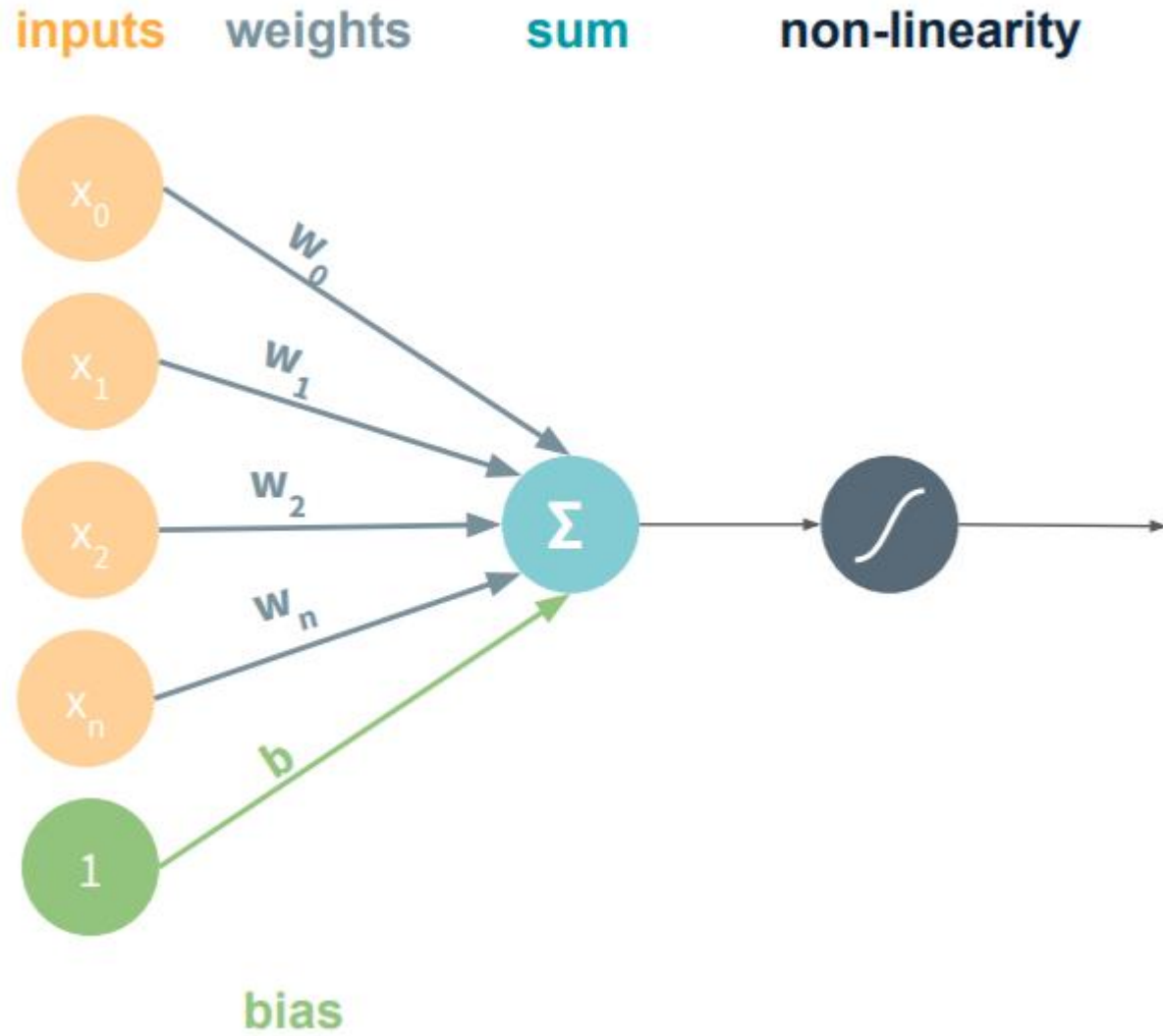
$$= \frac{1}{(1 + e^{-3.2})} = 0.96$$







# The Perceptron: Simplified

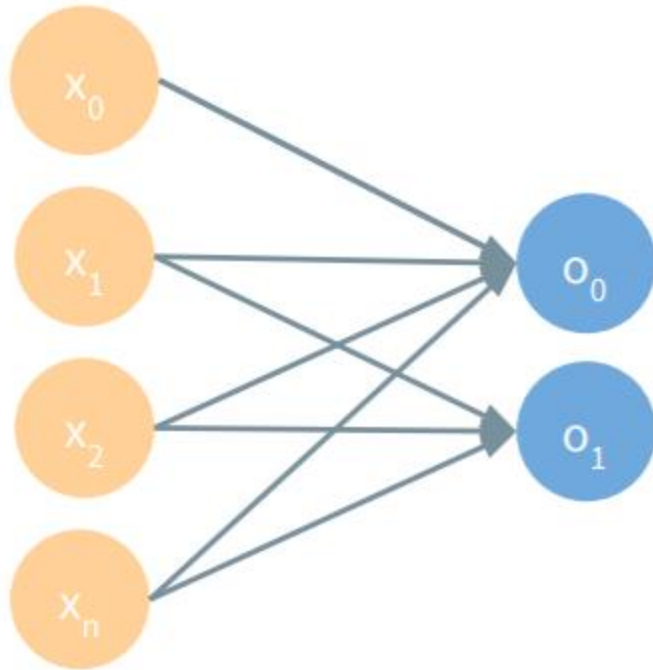




# The Perceptron: Multiple Output and Multi Layered

Input layer

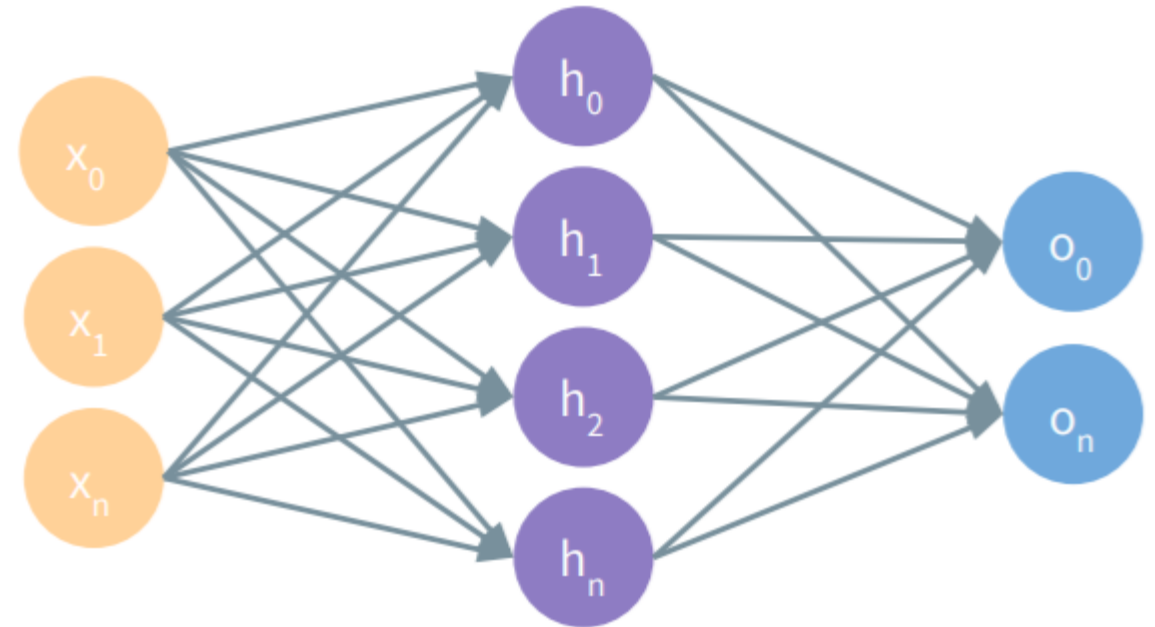
output layer



input  
layer

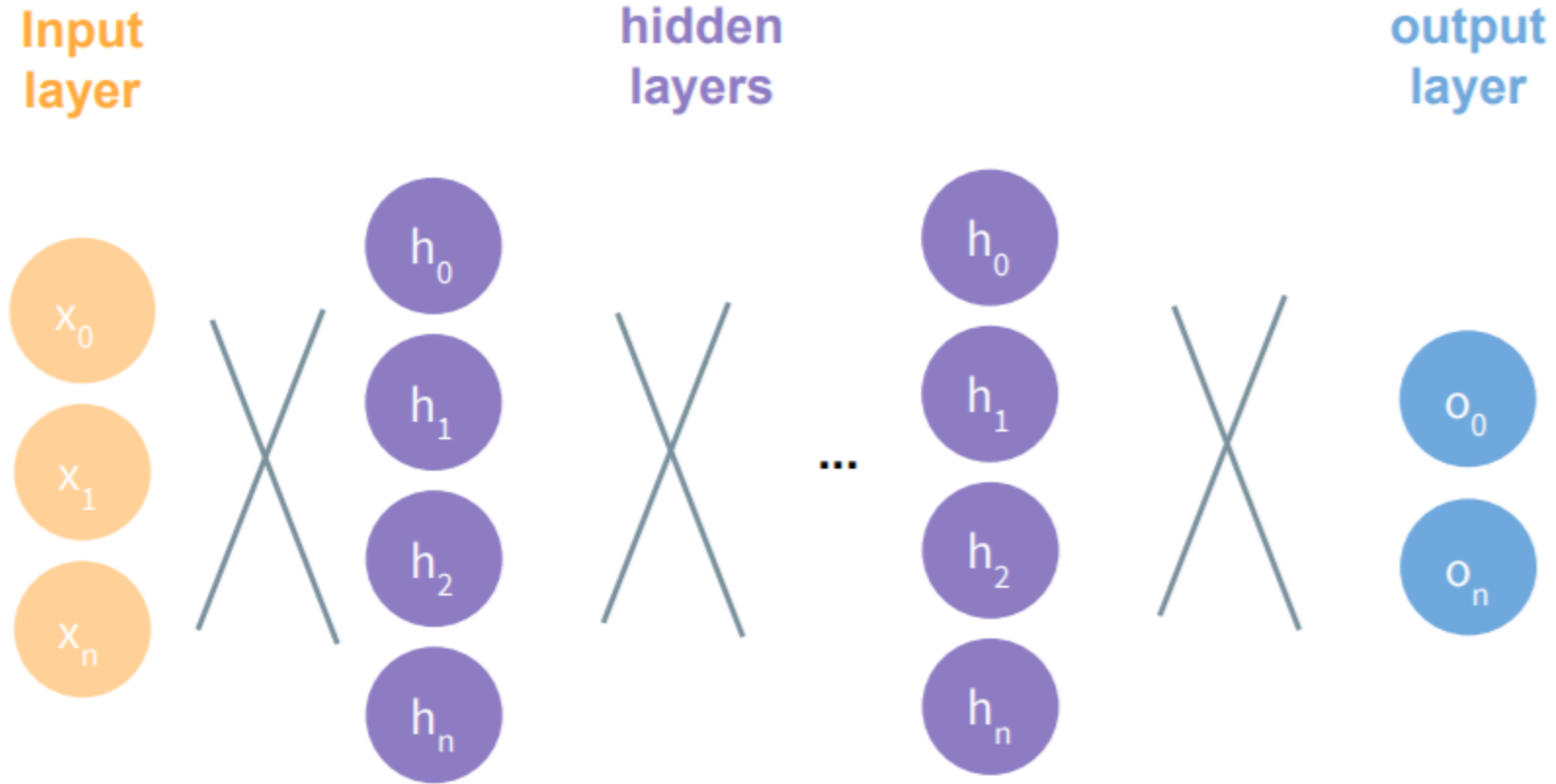
hidden  
layer

output  
layer





# Deep Neural Nets



Now comes the Meat (Math)!

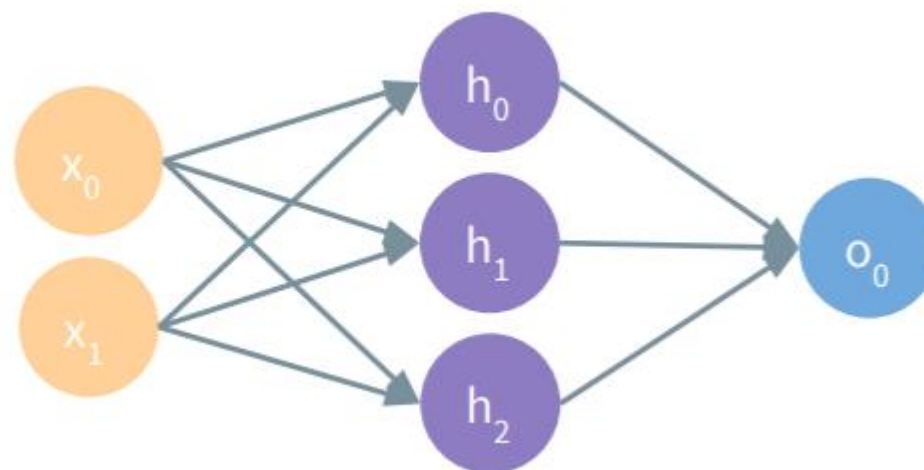


Will my flight be delayed?

Temperature  $-20^{\circ}\text{C}$

Wind speed 45 KM/hour

$[-20, 45]$



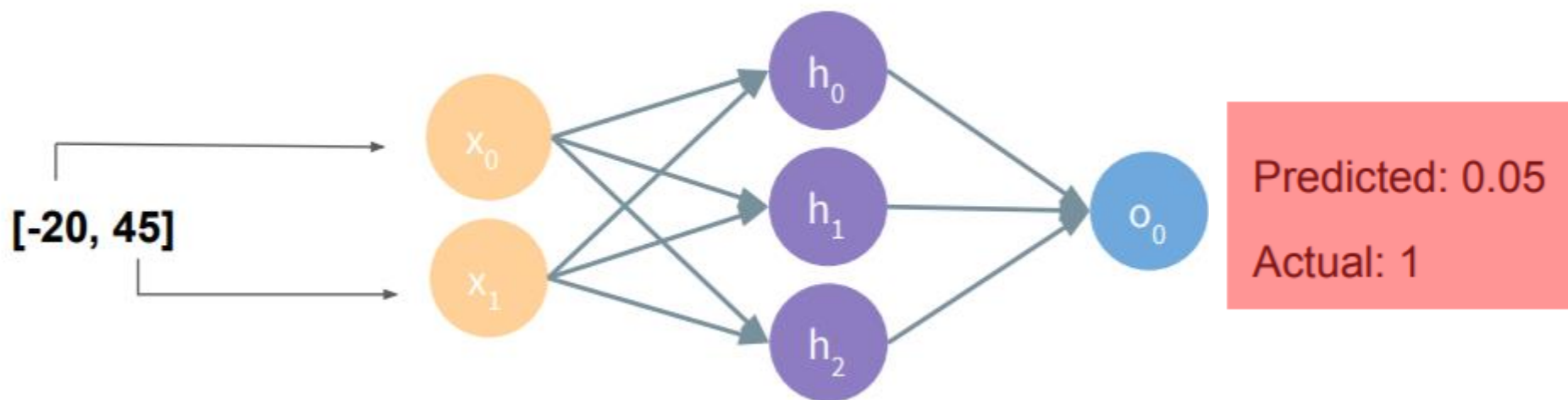
 DEPARTURES					
TIME	DESTINATION	FLIGHT	GATE	REMARKS	
12:39	LONDON	BA 903	31	CANCELLED	
12:57	SYDNEY	QF5723	27	CANCELLED	
13:08	TORONTO	AC5984	22	CANCELLED	
13:21	TOKYO	JL 608	41	DELAYED	
13:37	HONG KONG	CX5471	29	CANCELLED	
13:48	MADRID	IB3941	30	DELAYED	
14:19	BERLIN	LH5021	28	CANCELLED	
14:35	NEW YORK	AA 997	11	CANCELLED	
14:54	PARIS	AF5870	23	DELAYED	
15:10	ROME	AZ5324	43	CANCELLED	



## Quantifying Loss

Temperature -20° C

Wind speed 45 KM/hour



$$loss(\underbrace{f(x^{(i)}; \theta)}_{\text{Predicted}}, \underbrace{y^{(i)}}_{\text{Actual}})$$

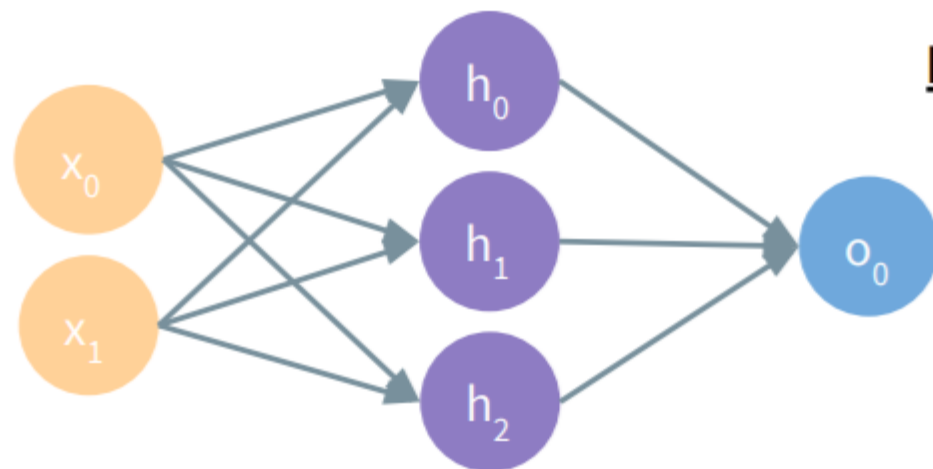
Predicted    Actual



## Quantifying Loss

### Input

[  
[-20, 45],  
[80, 0],  
[4, 15],  
[45, 60],  
]



### Predicted

[  
0.05  
0.02  
0.96  
0.35  
]

### Actual

[  
1  
0  
1  
1  
]

$$\text{total loss} := J(\theta) = \frac{1}{N} \sum_i \text{loss}(\underbrace{f(x^{(i)}; \theta)}_{\text{Predicted}}, \underbrace{y^{(i)}}_{\text{Actual}})$$



## Training the neural network – Minimize Loss Function

Minimize the number of incorrect predictions



Minimize the Loss function



Mathematically

$$\arg \min_{\theta} \frac{1}{N} \sum_i \text{loss}(f(x^{(i)}; \theta), y^{(i)})$$

$$\theta = W_1, W_2 \dots W_n$$

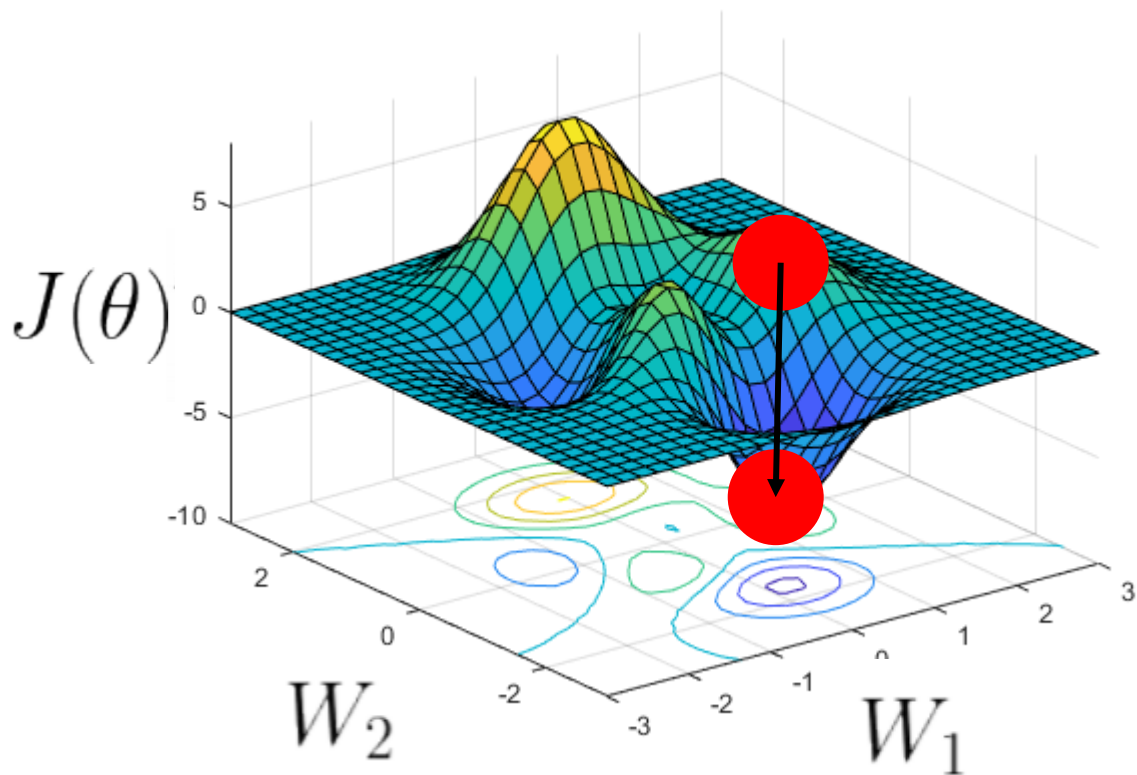






## Training the neural network – Gradient Descent

### Gradient Descent Algorithm



- Initialize  $\theta$  randomly
- For N Epochs
  - For each training example  $(x, y)$ :

- Compute Loss Gradient:  $\frac{\partial J(\theta)}{\partial \theta}$

- Update  $\theta$  with update rule:

$$\theta := \theta - \eta \frac{\partial J(\theta)}{\partial \theta}$$



## Training the neural network - Backpropagation



$$\frac{\partial J(\theta)}{\partial W_2} =$$



## Training the neural network - Backpropagation

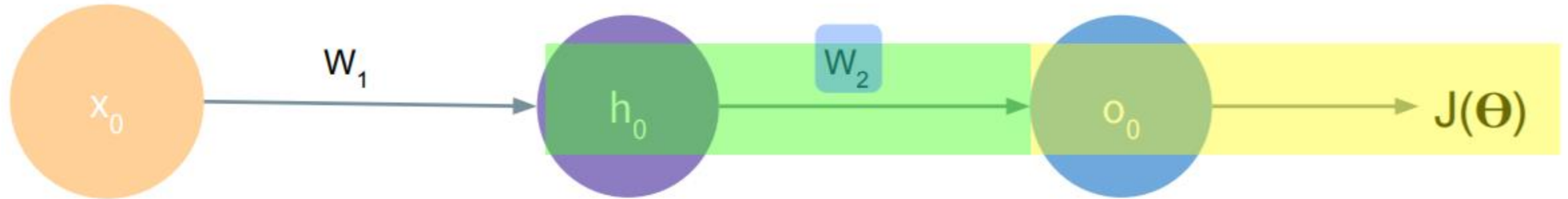


Apply the chain rule

$$\frac{\partial J(\theta)}{\partial W_2} = \frac{\partial J(\theta)}{\partial o_0}$$



## Training the neural network - Backpropagation



Apply the chain rule

$$\frac{\partial J(\theta)}{\partial W_2} = \frac{\partial J(\theta)}{\partial o_0} * \frac{\partial o_0}{\partial W_2}$$





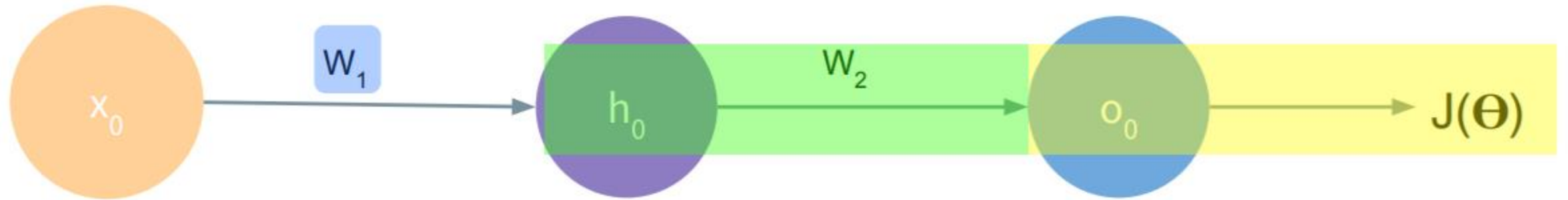
## Training the neural network - Backpropagation



$$\frac{\partial J(\theta)}{\partial W_1} =$$



## Training the neural network - Backpropagation



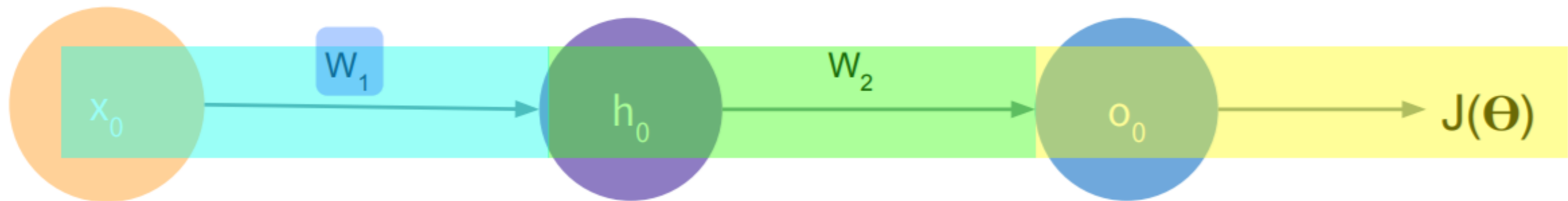
Apply the chain rule

$$\frac{\partial J(\theta)}{\partial W_1} = \frac{\partial J(\theta)}{\partial o_0} * \frac{\partial o_0}{\partial h_0}$$





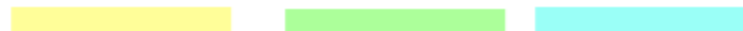
# Training the neural network - Backpropagation



Apply the chain rule

Apply the chain rule

$$\frac{\partial J(\theta)}{\partial W_1} = \frac{\partial J(\theta)}{\partial o_0} * \frac{\partial o_0}{\partial h_0} * \frac{\partial h_0}{\partial W_1}$$



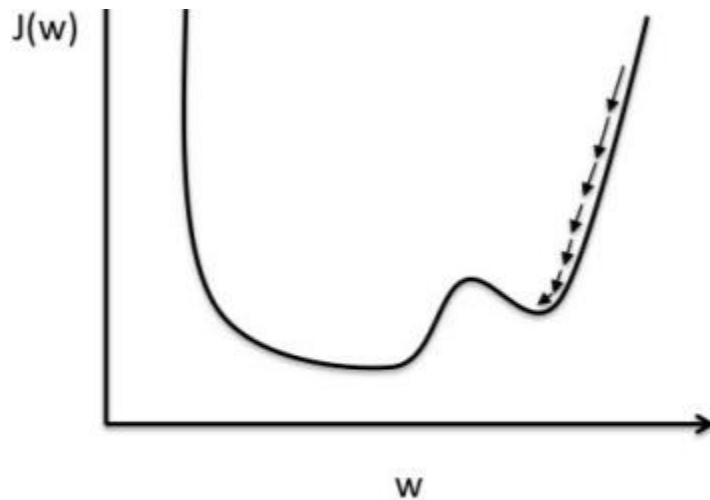


## Training the neural network – Learning Rate

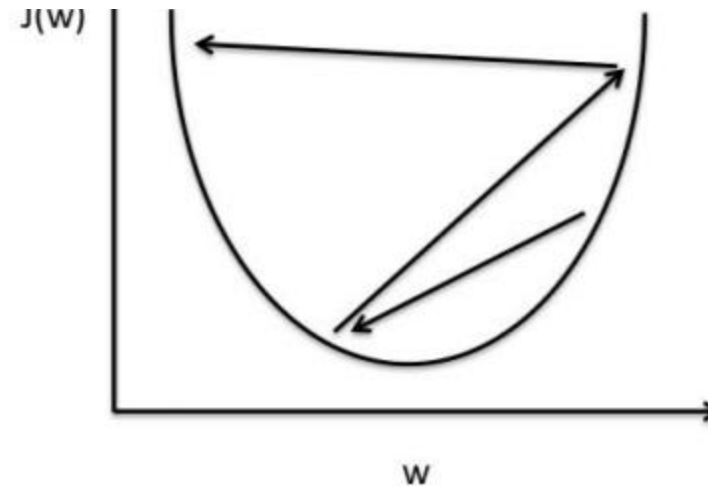
How to Choose Learning Rate?

**Update Rule:**

$$\theta := \theta - \eta \frac{\partial J(\theta)}{\partial \theta}$$



Small learning rate: Many iterations until convergence and trapping in local minima.



Large learning rate: Overshooting.





## Training the neural network – Learning Rate

Try different Learning Rates and observe if solution converges

OR

Use an adaptive learning rate which is larger initially and decreases as we get close to the minima point

Now comes the Wine (Programming)!