Introduction to Neural Networks

DEEP LEARNING FUNDAMENTALS PART 1

databricks

Logistics

- We can’t hear you…
- Recording will be available…
- Slides will be available…
- Code samples and notebooks will be available…
- Queue up Questions…

VISION

Accelerate innovation by unifying data science, engineering and business

---

PRODUCT

Unified Analytics Platform powered by Apache Spark™

WHO WE ARE

- Founded by the original creators of Apache Spark
- Contributes **75%** of the open source code, **10x** more than any other company
- Trained **100k+** Spark users on the Databricks platform

About our speaker

Denny Lee

Technical Product Marketing Manager

Former:

- Senior Director of Data Sciences Engineering at SAP Concur
- Principal Program Manager at Microsoft
  - Azure Cosmos DB Engineering Spark and Graph Initiatives
  - Isotope Incubation Team (currently known as HDInsight)
  - Bing's Audience Insights Team
  - Yahoo!'s 24TB Analysis Services cube

Deep Learning Fundamentals Series

This is a three-part series:

- Introduction to Neural Networks
- Training Neural Networks
- Applying your Neural Networks

This series will be make use of Keras (TensorFlow backend) but as it is a fundamentals series, we are focusing primarily on the concepts.

Introduction to Neural Networks

- What is Deep Learning?
- What can Deep Learning do for you?
- What are artificial neural networks?
- Let’s start with a perceptron…
- Understanding the effect of activation functions

Upcoming Sessions

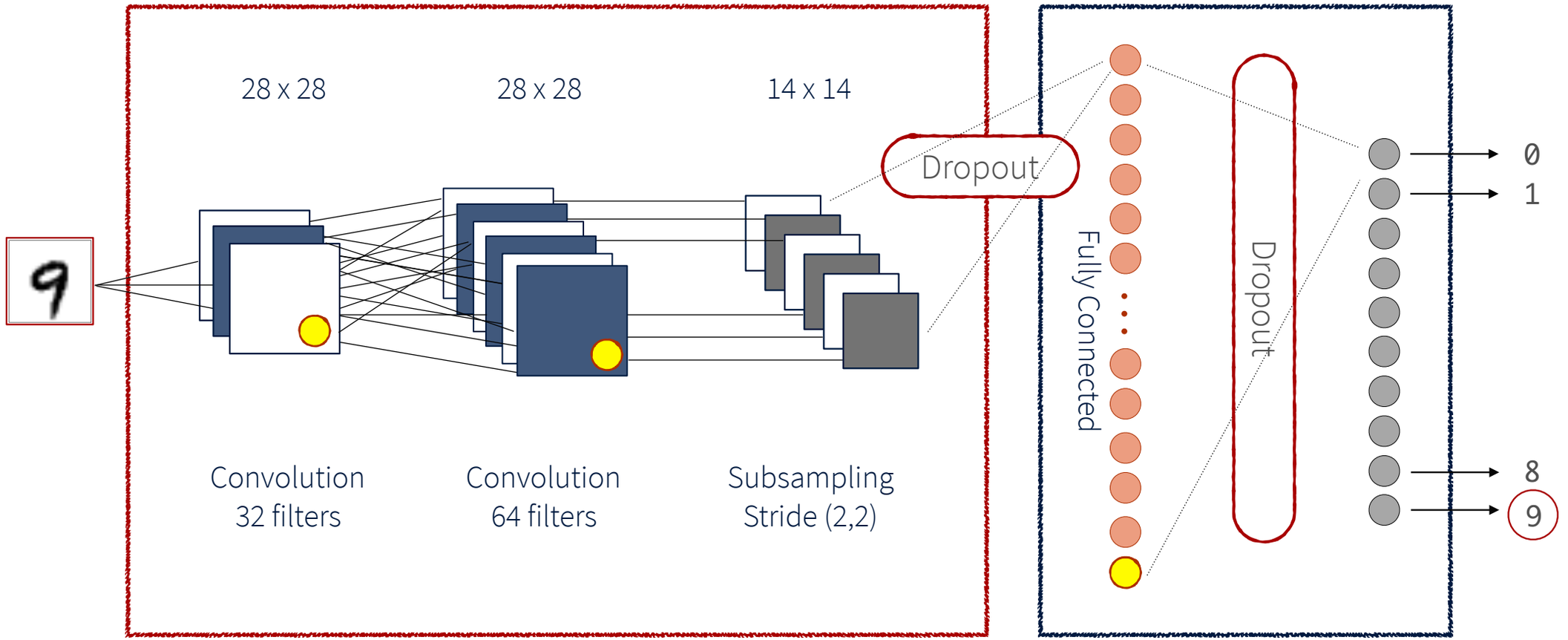
Training your Neural Network

- Tuning training
- Training Algorithms
- Optimization (including Adam)
- Convolutional Neural Networks

Applying your Neural Networks

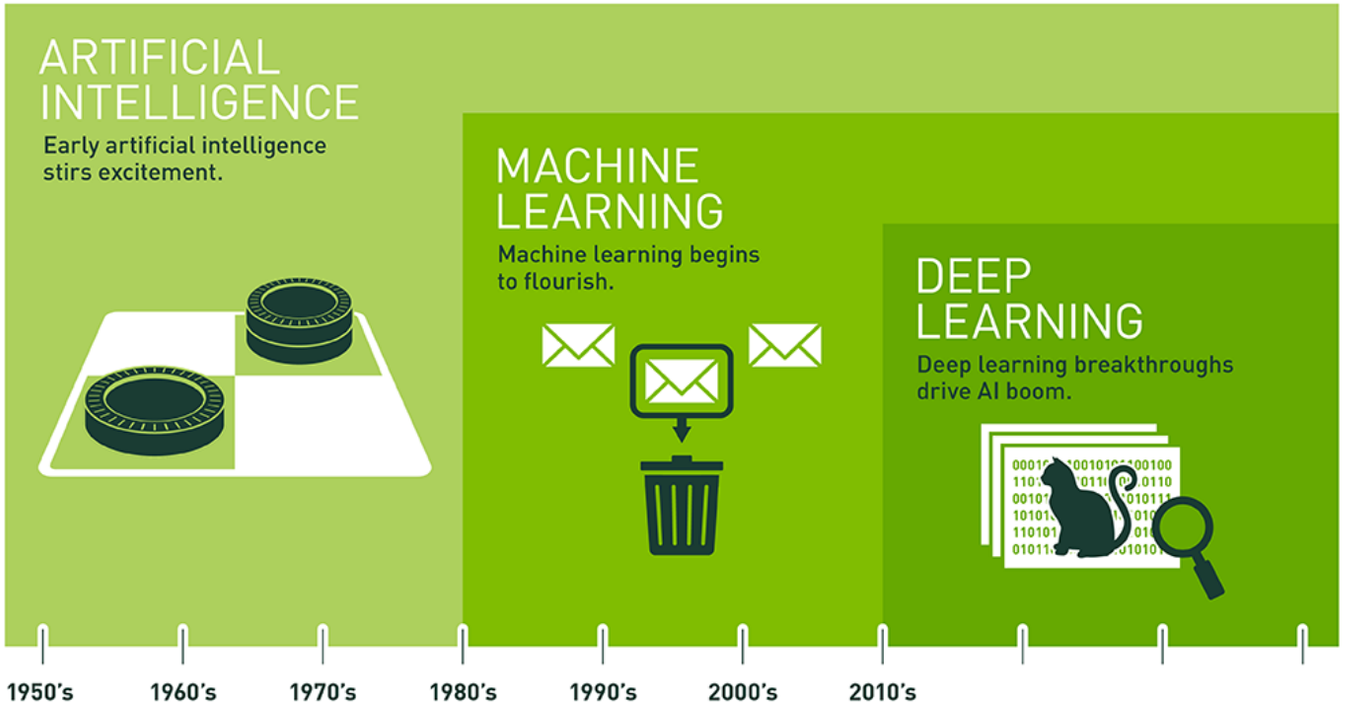
- Diving further into Convolutional Neural Networks (CNNs)
- CNN Architectures
- Convolutions at Work!

Convolutional Neural Networks



Deep Learning, ML, AI, . . . oh my!

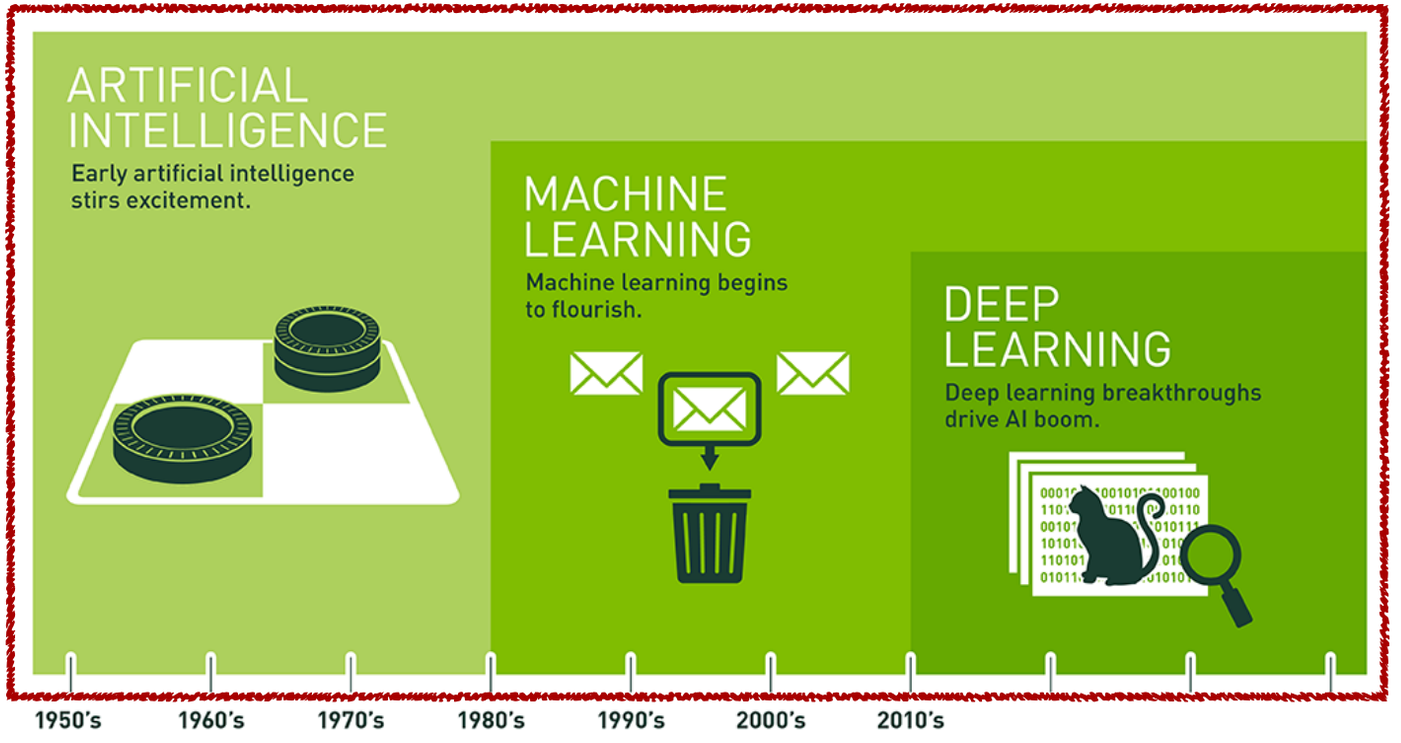
Deep Learning, ML, AI, . . . oh my!



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Source: http://bit.ly/2suufGJ

Deep Learning, ML, AI, . . . oh my!



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

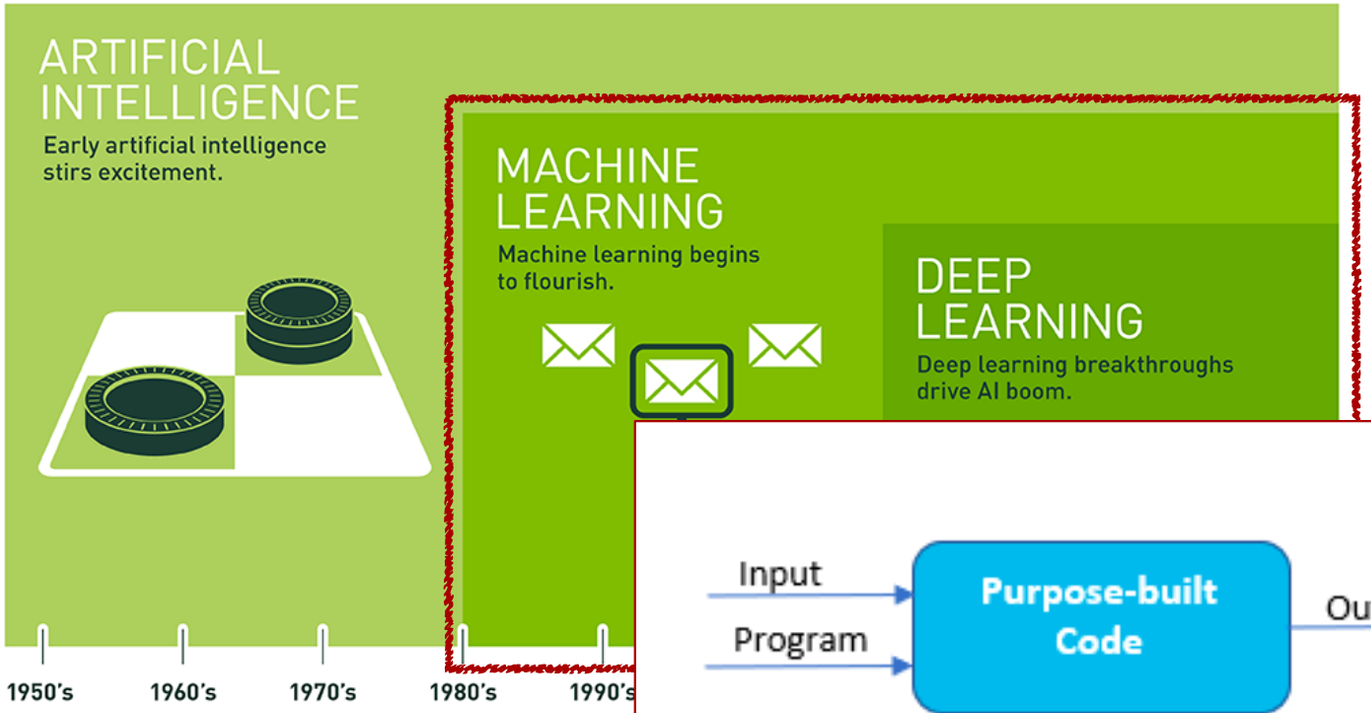
Artificial Intelligence

Artificial Intelligence is human intelligence exhibited by machines

[The world's best Dota 2 players just got destroyed by a killer AI from Elon Musk's startup]

[Elon Musk-funded Dota 2 bots spank top-tier humans, and they know how to trash talk]

Deep Learning, ML, AI, . . . oh my!



Since an early flush of optimism in the 1950s, smaller subse
deep learning, a subset of machine learning – have created

Machine Learning

Field of study that gives computers the ability to learn without being explicitly programmed.

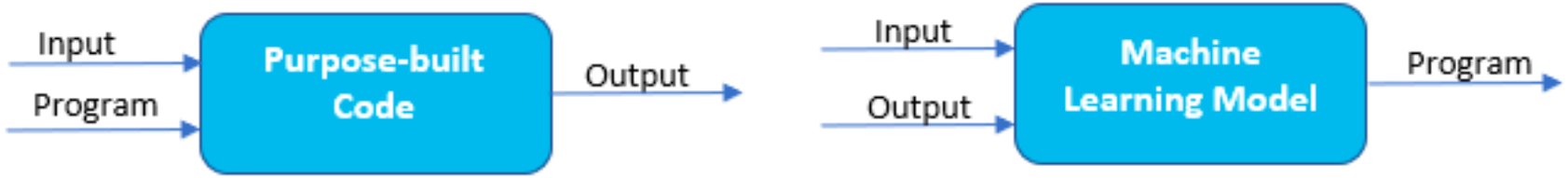
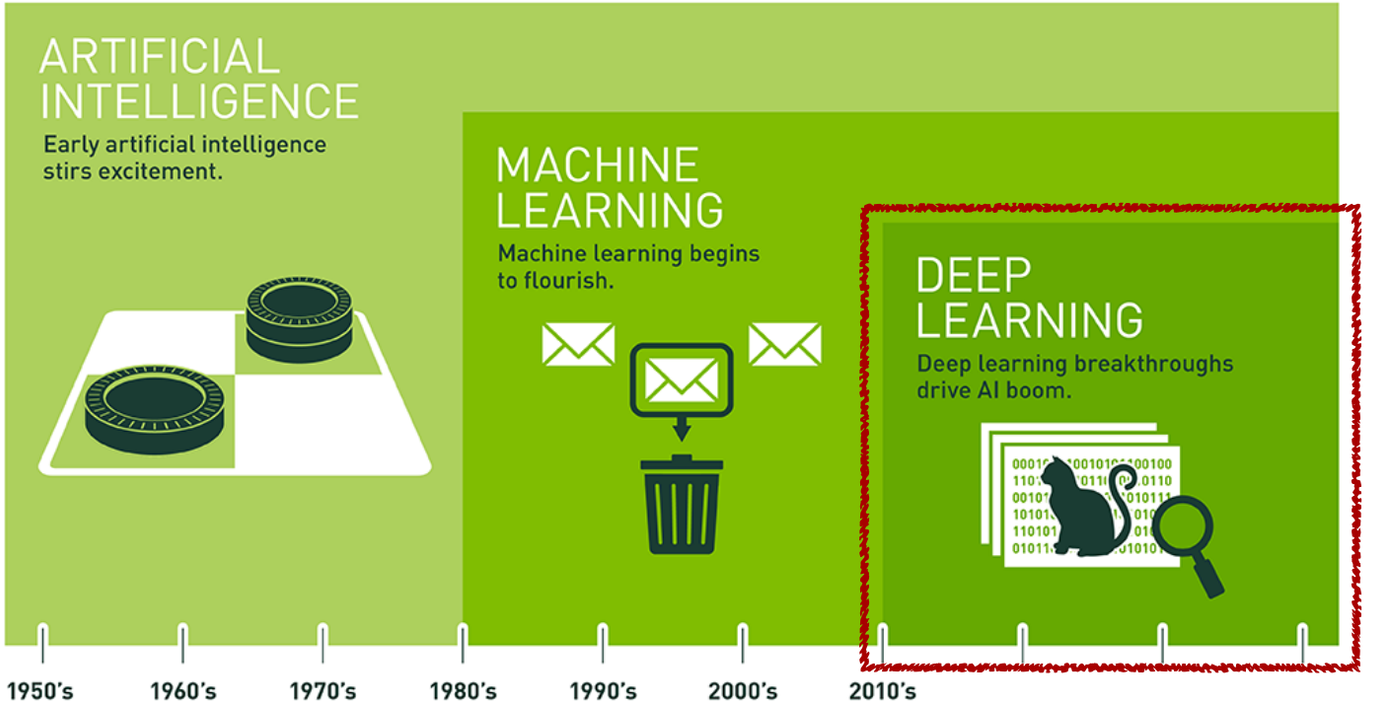


Figure 1:Purpose-built Code Vs Machine Learning Model

Source: https://bit.ly/2NnyrOz

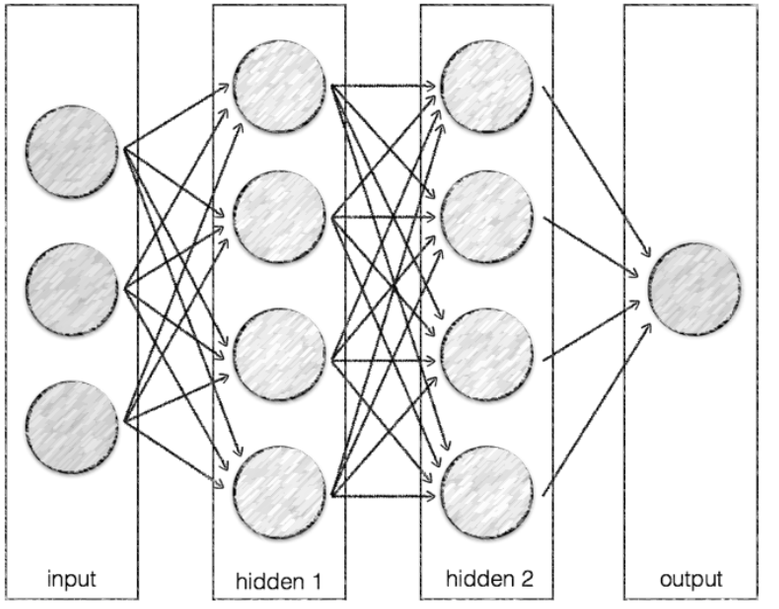
Deep Learning, ML, AI, . . . oh my!



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Deep Learning

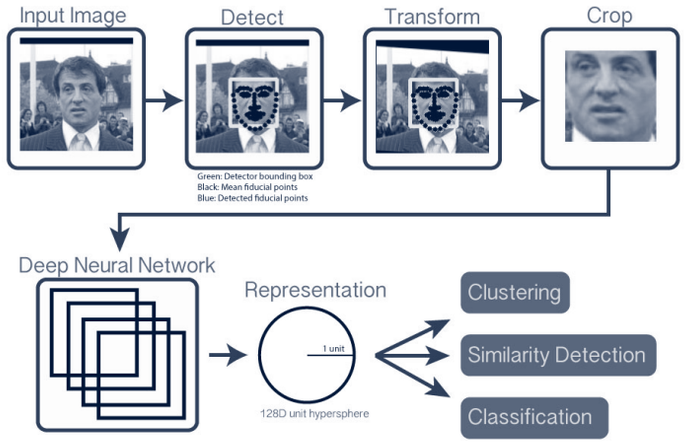
Using deep neural networks to implement machine learning



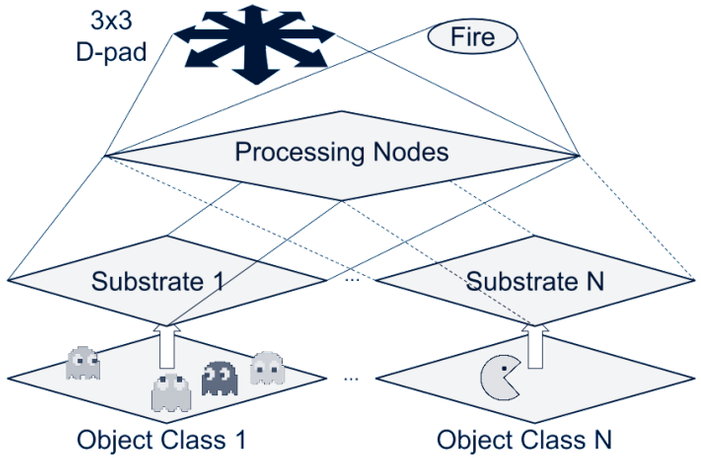
Ask not what AI can do for you….

Applications of neural networks

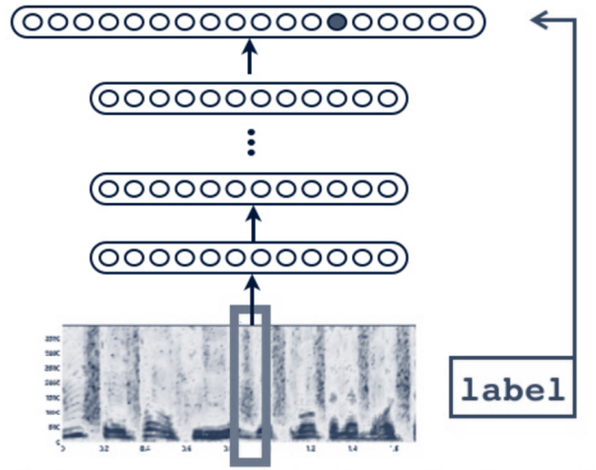
Object Recognition: Facial



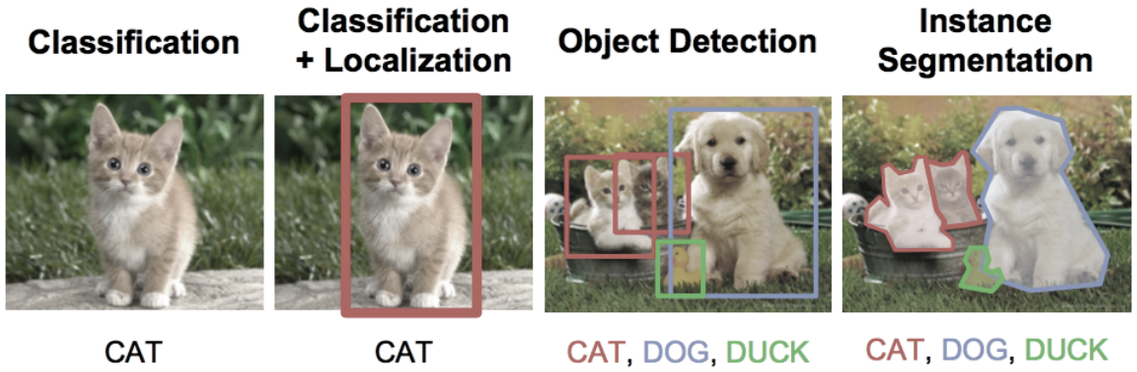
Game Playing



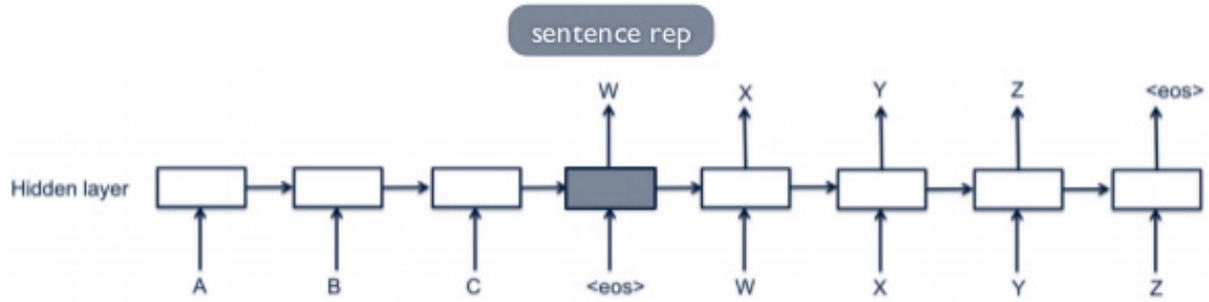
Object Recognition: Speech



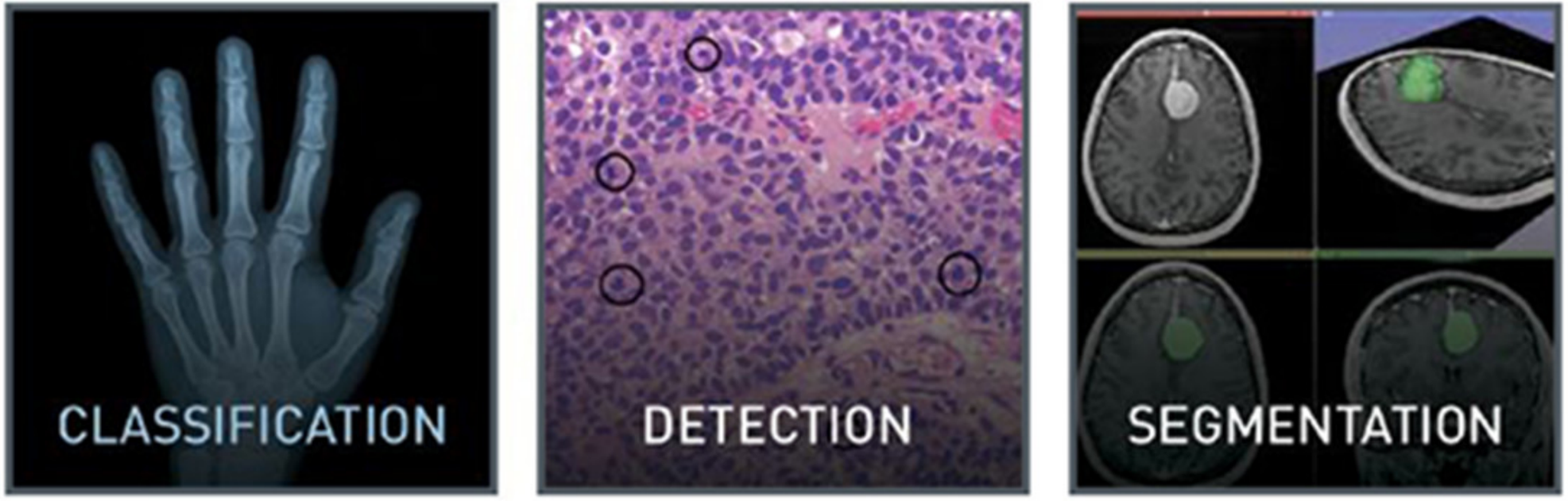
Object Classification



Language Translation



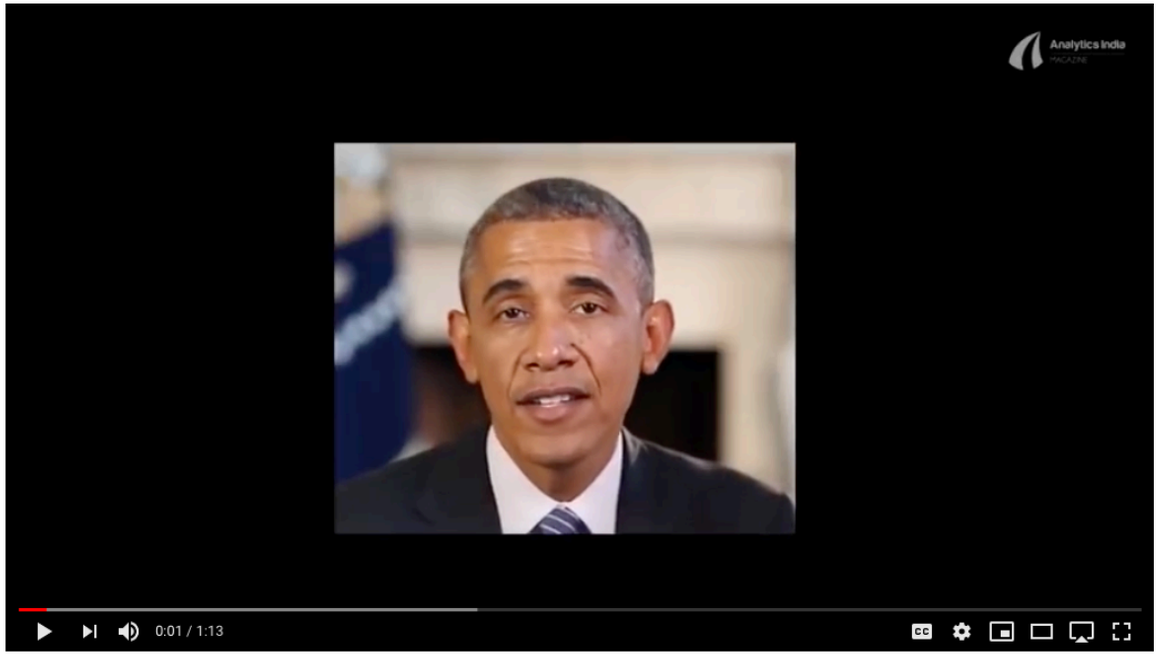
Healthcare



Source: http://bit.ly/2BXF5sR

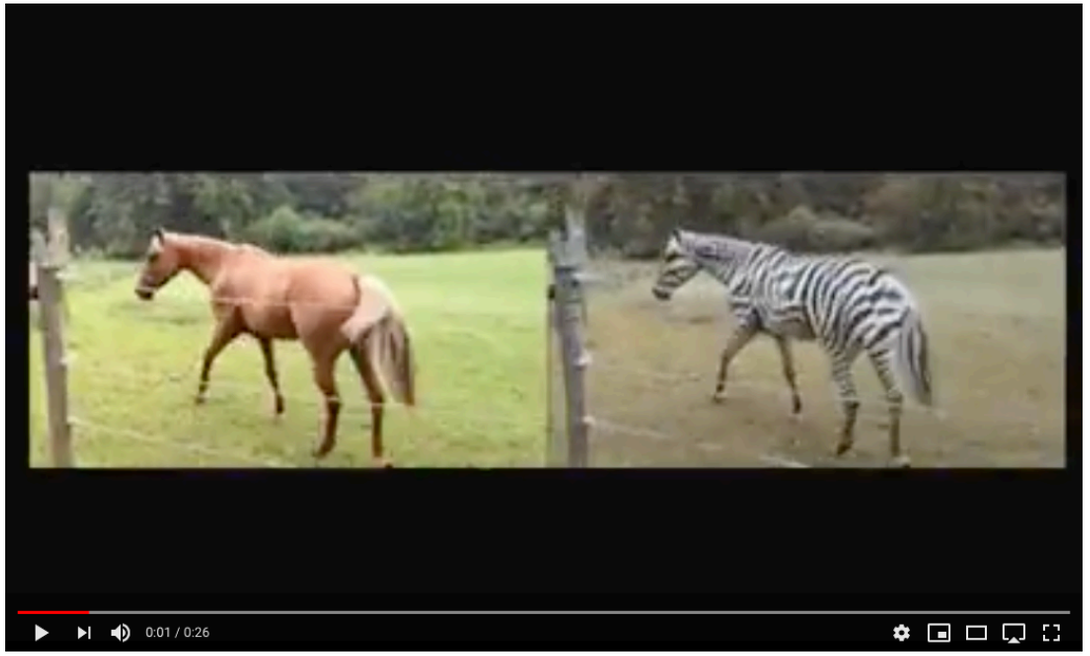
Faking videos

AI-generated "real fake" video of Barack Obama



Source: https://youtu.be/dkoi7sZvWiU

Turning a horse video into a zebra video in real time using GANs

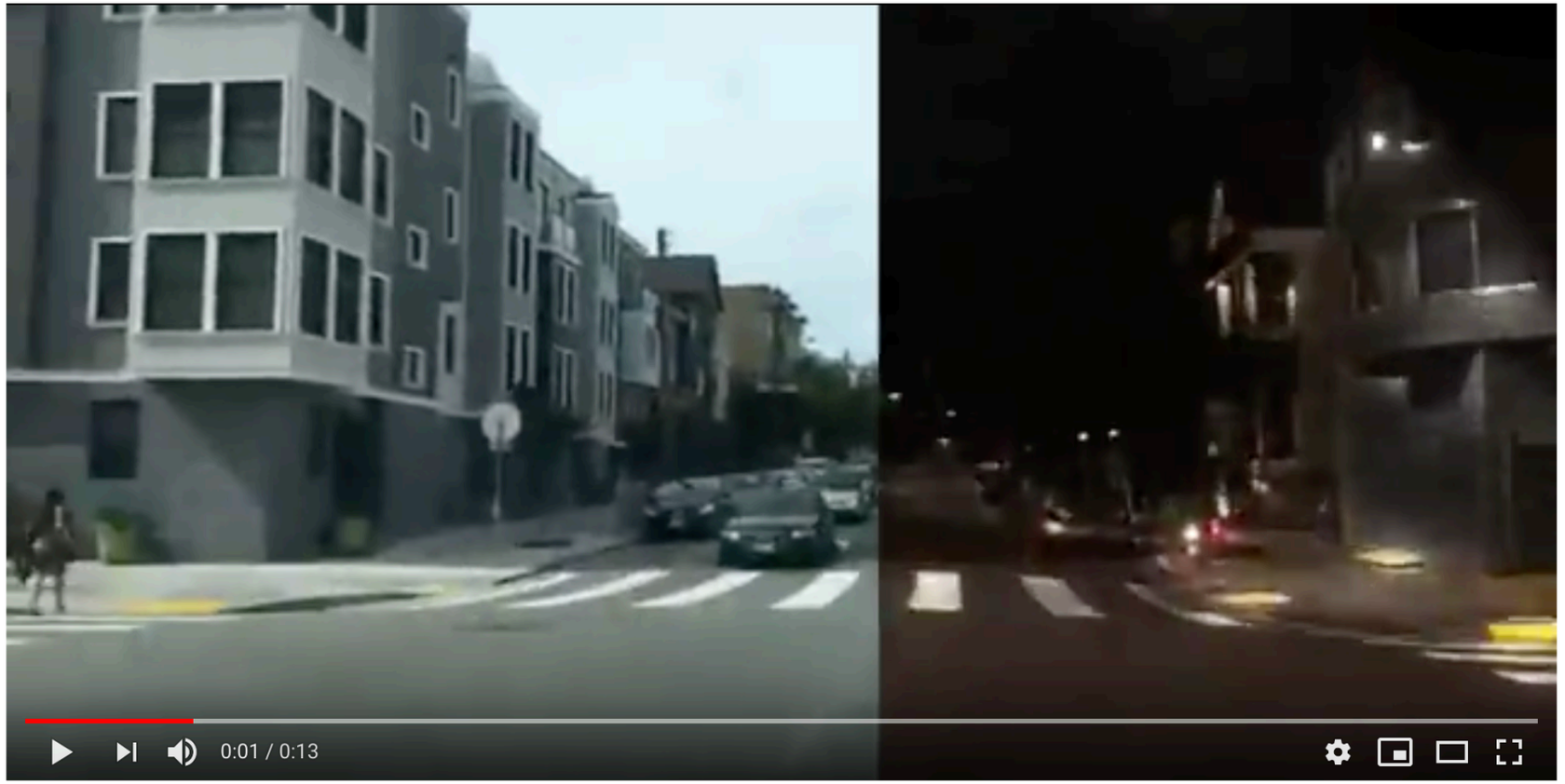


Source: https://youtu.be/JzgOfISLNjk

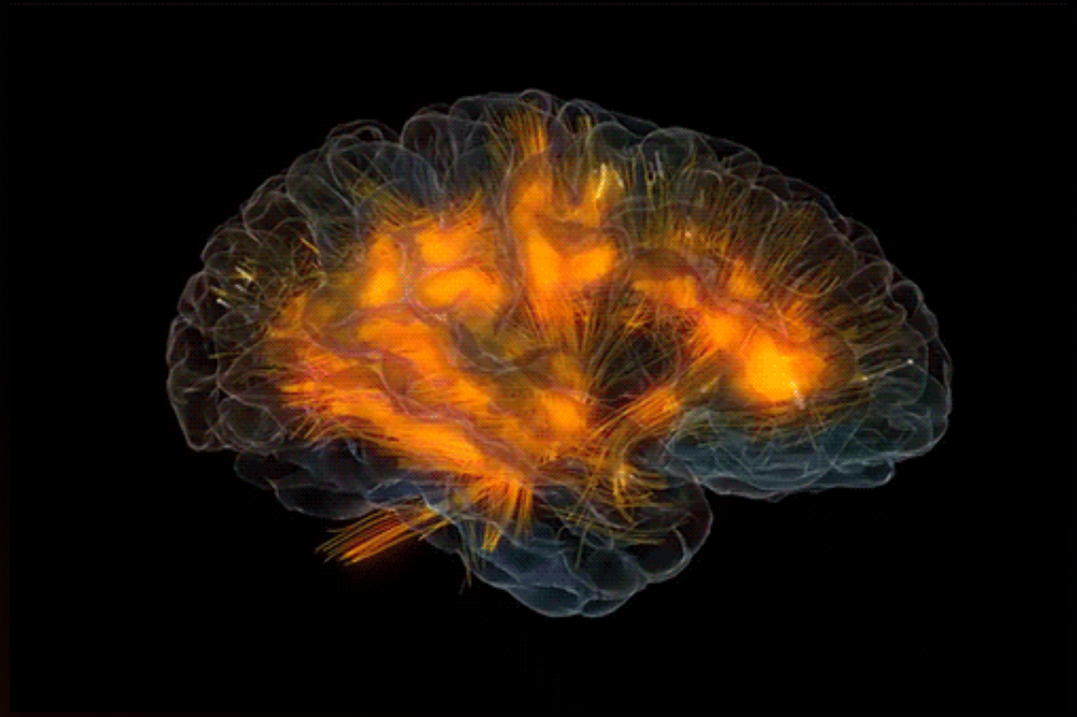
Self-driving cars

Source: https://youtu.be/URmxzxYlmtg?t=6m30s

Turning the day into night



Source: https://youtu.be/N7KbfWodXJE

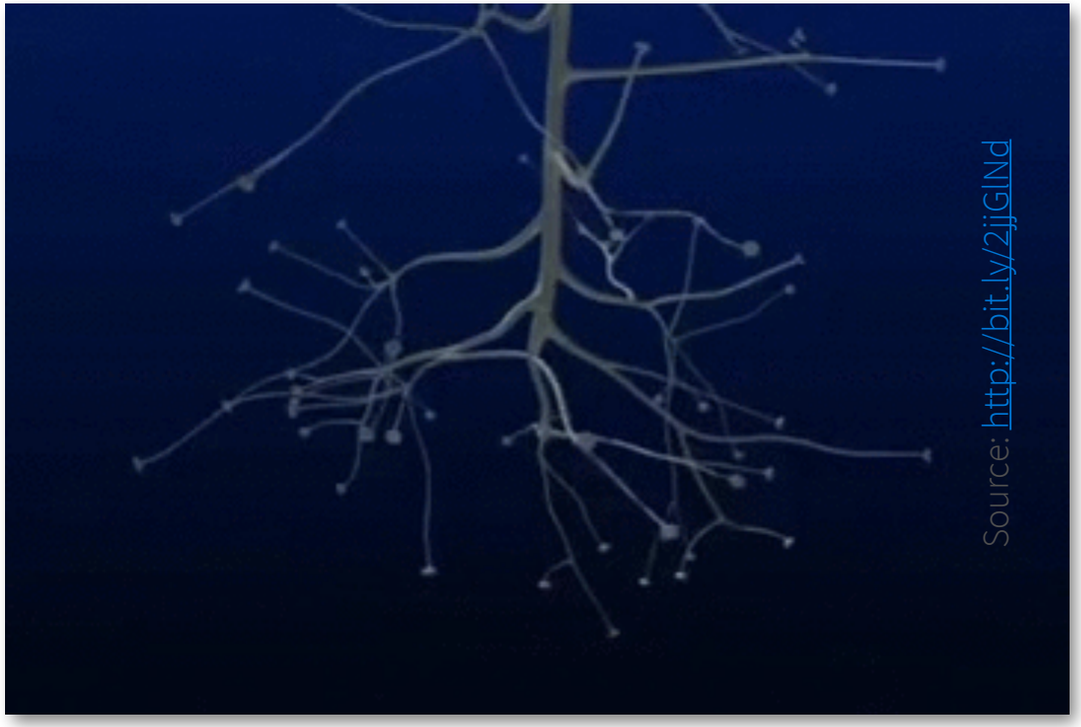


What are artificial neural networks?

Source: http://bit.ly/2jc9mYw

Conceptually, what are artificial neural networks?

- A *neuron* receives a signal, processes it, and propagates the signal (or not)
- The brain is comprised of around 100 billion neurons, each connected to ~10k other neurons: 10^{15} synaptic connections
- *ANNs* are a *simplistic* imitation of a brain comprised of dense net of simple structures

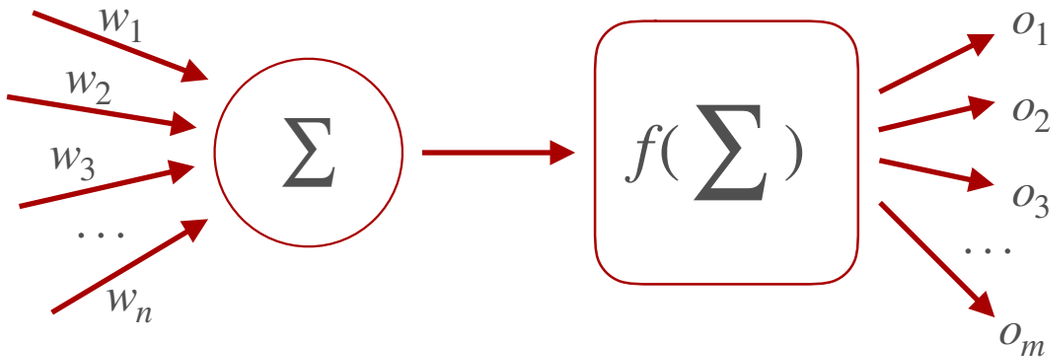


Conceptual mathematical model

- Receives input from n sources
- Computes weighted sum

$$h_1 = x_1 w_1 + x_2 w_2 + \ldots x_n w_n$$

- Passes through an activation function
- Sends the signal to m succeeding neurons



“Parallels”

A single neuron in the brain is an incredibly complex machine that even today we don’t understand. A single “neuron” in a neural network is an incredibly simple mathematical function that captures a minuscule fraction of the complexity of a biological neuron. So to say neural networks mimic the brain, that is true at the level of loose inspiration, but really artificial neural networks are nothing like what the biological brain does.

Andrew Ng
Medium Article: Google Brain’s Co-inventor Tells Why He’s Building Chinese Neural Networks

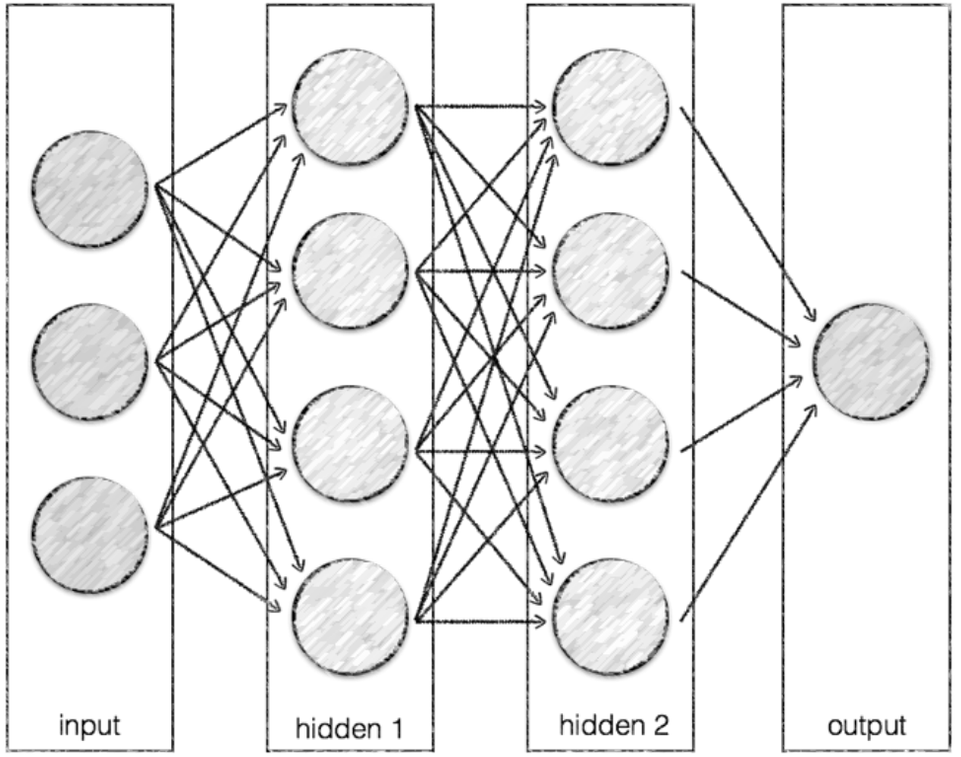
“Parallels”

A single neuron in the brain is an incredibly complex machine that even today we don’t understand. A single “neuron” in a neural network is an incredibly simple mathematical function that captures a minuscule fraction of the complexity of a biological neuron. So to say neural networks mimic the brain, that is true at the level of loose inspiration, but really artificial neural networks are nothing like what the biological brain does.

Andrew Ng
Medium Article: Google Brain’s Co-inventor Tells Why He’s Building Chinese Neural Networks

Artificial Neural Network

- Organized into layers of neurons as a black-box model
- Typically 3 or more: input, hidden and output
- For image detectors,
  - the first layers are organized similar as the visual cortex,
  - but activation functions are simpler for ANNs vs. biological neural networks



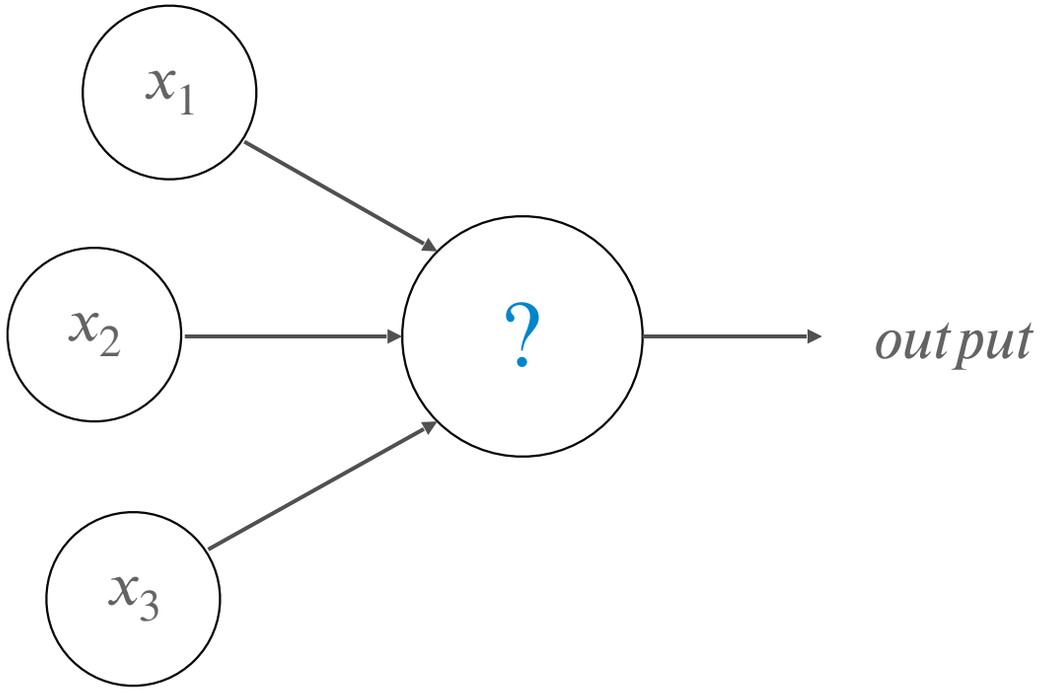
How perceptive of you…

or

The Immense Power of Simple Structures

Perceptron

Simplified (binary) artificial neuron



Do I snowboard this weekend?

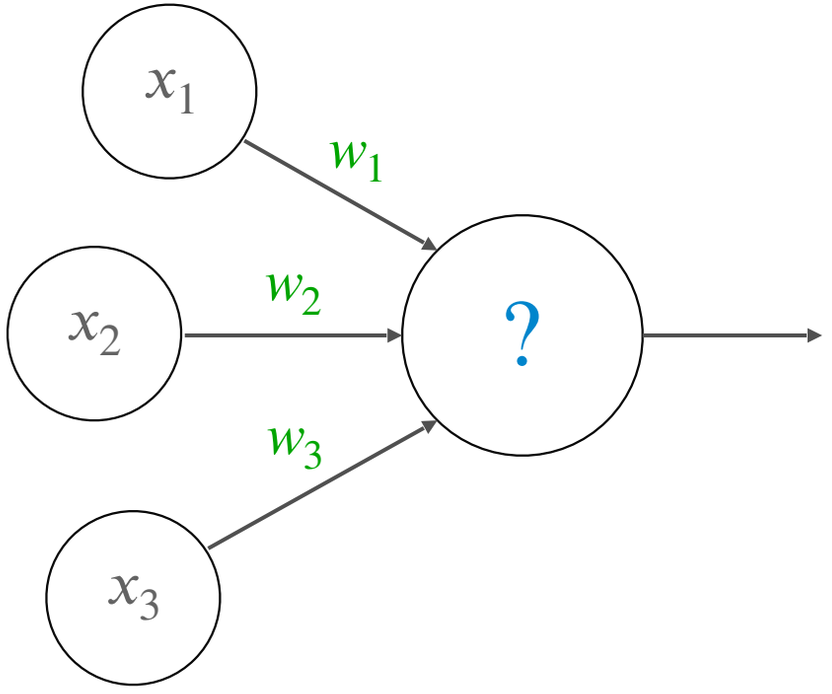
$x_1 \rightarrow$ *Is the weather good?*

$x_2 \rightarrow$ *Is the powder good?*

$x_3 \rightarrow$ *Am I in the mood to drive?*

Perceptron

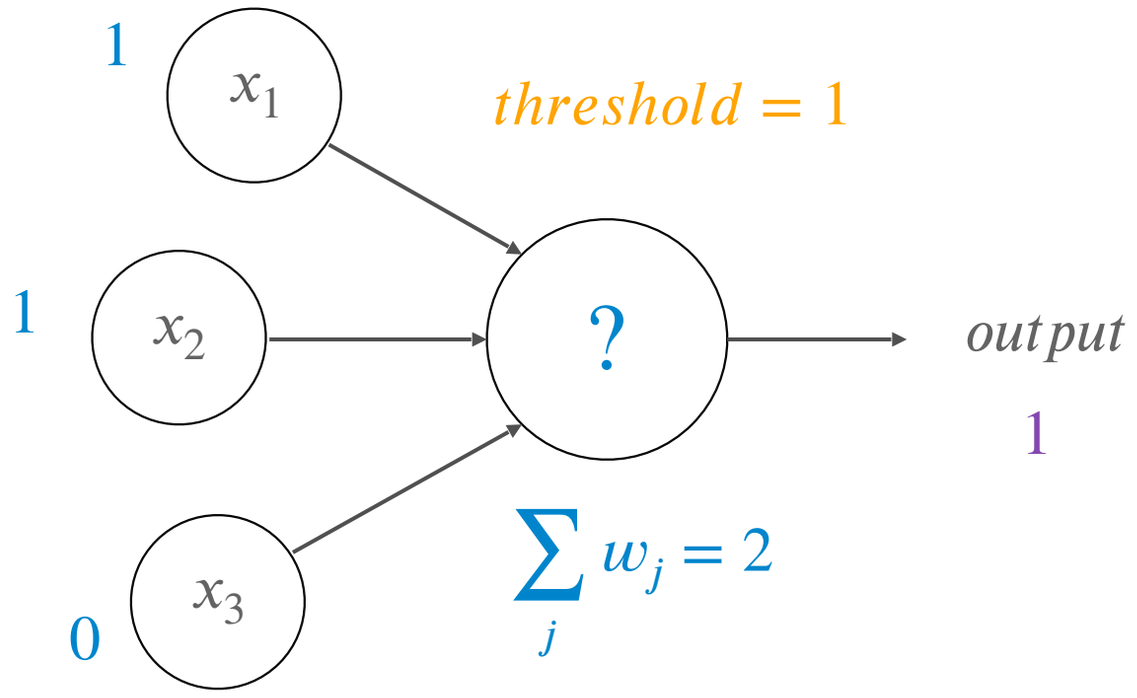
Simplified (binary) artificial neuron with weights



$$output = \begin{cases} 0, & \sum_{j=0}^{n} w_j x_j \leq threshold \\ 1, & \sum_{j=0}^{n} w_j x_j > threshold \end{cases}$$

Perceptron

Simplified (binary) artificial neuron; *no weights*



Do I snowboard this weekend?

$x_1 = 1$ (*good weather*)

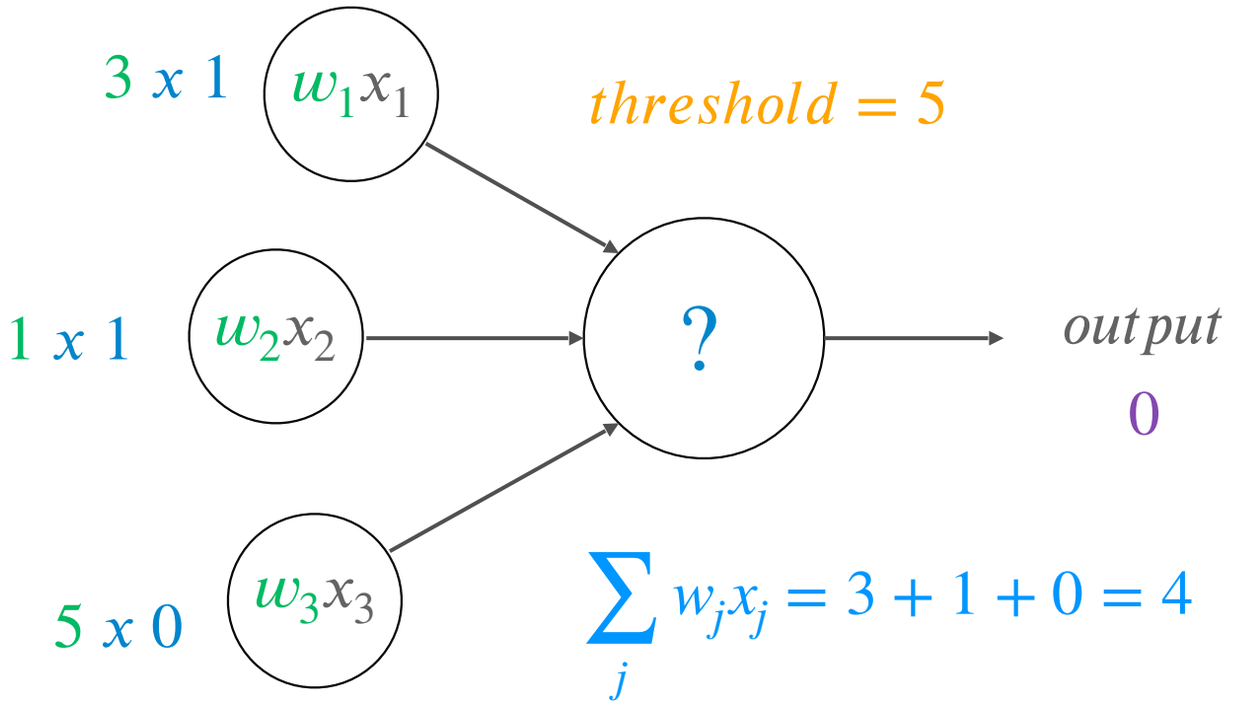
$x_2 = 1$ (*a lot of powder*)

$x_3 = 0$ (*driving sucks*)

Perceptron

Simplified (binary) artificial neuron; *add weights*

Persona: Après-ski'er



Do I snowboard this weekend?

$x_1 = 1$ (*good weather*)

$w_1 = 3$

$x_2 = 1$ (*a lot of powder*)

$w_2 = 1$

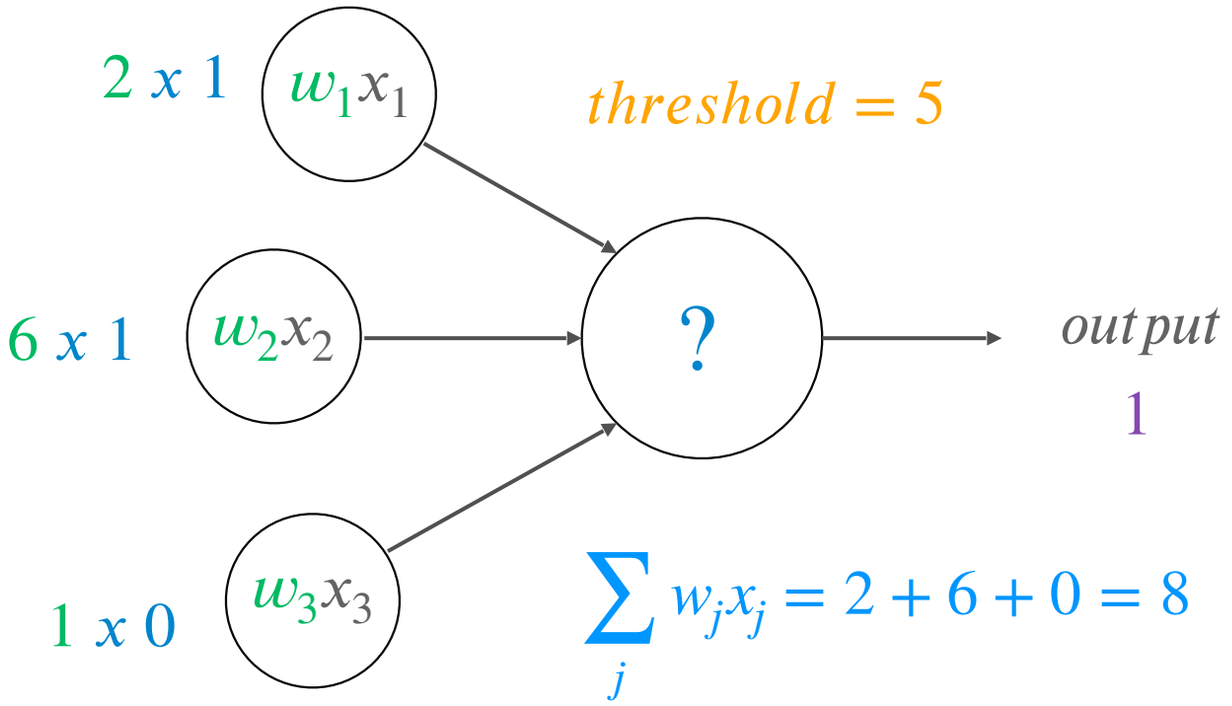
$x_3 = 0$ (*driving sucks*)

$w_3 = 5$

Perceptron

Simplified (binary) artificial neuron; *add weights*

Persona: Shredder



Do I snowboard this weekend?

$x_1 = 1$ (*good weather*)

$w_1 = 2$

$x_2 = 1$ (*a lot of powder*)

$w_2 = 6$

$x_3 = 0$ (*driving sucks*)

$w_3 = 1$

Introducing Bias

Perceptron needs to take into account the bias

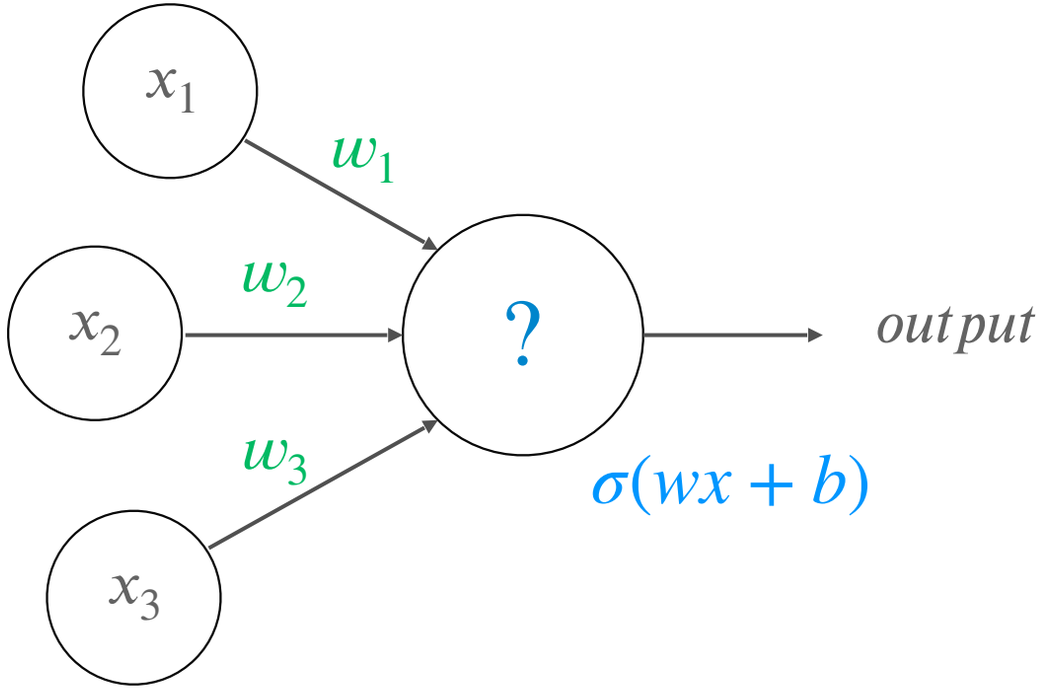
$$output = \begin{cases} 0, & wx + b \leq 0 \\ 1, & wx + b > 0 \end{cases}$$

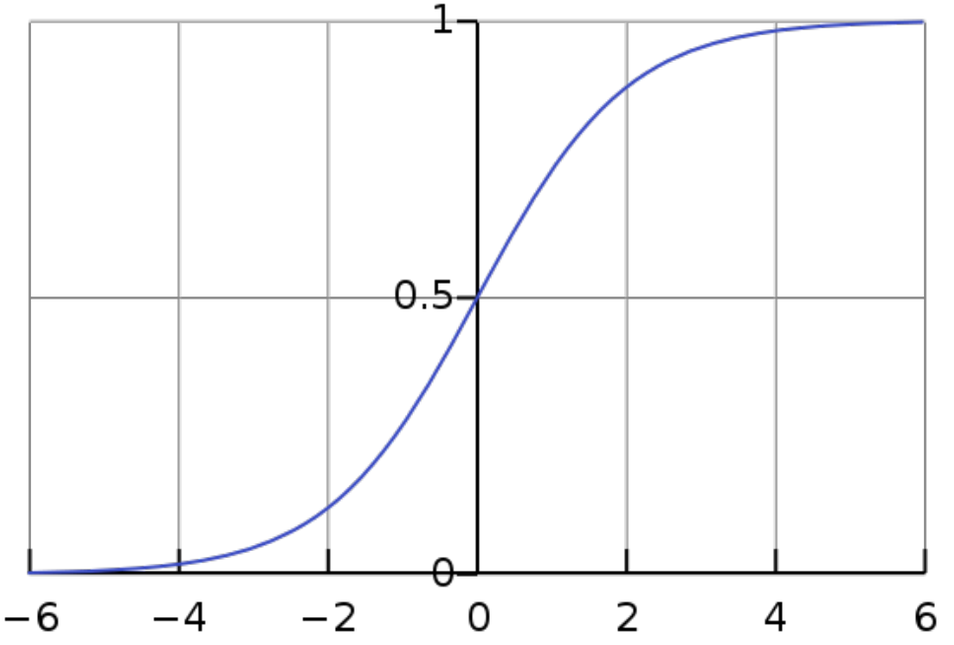
where b is how easy it is to get the perceptron to fire

e.g. Shredder has a strong positive bias to go to Whistler while Après-Ski'er bias is not as strong

Sigmoid Neuron

The more common artificial neuron



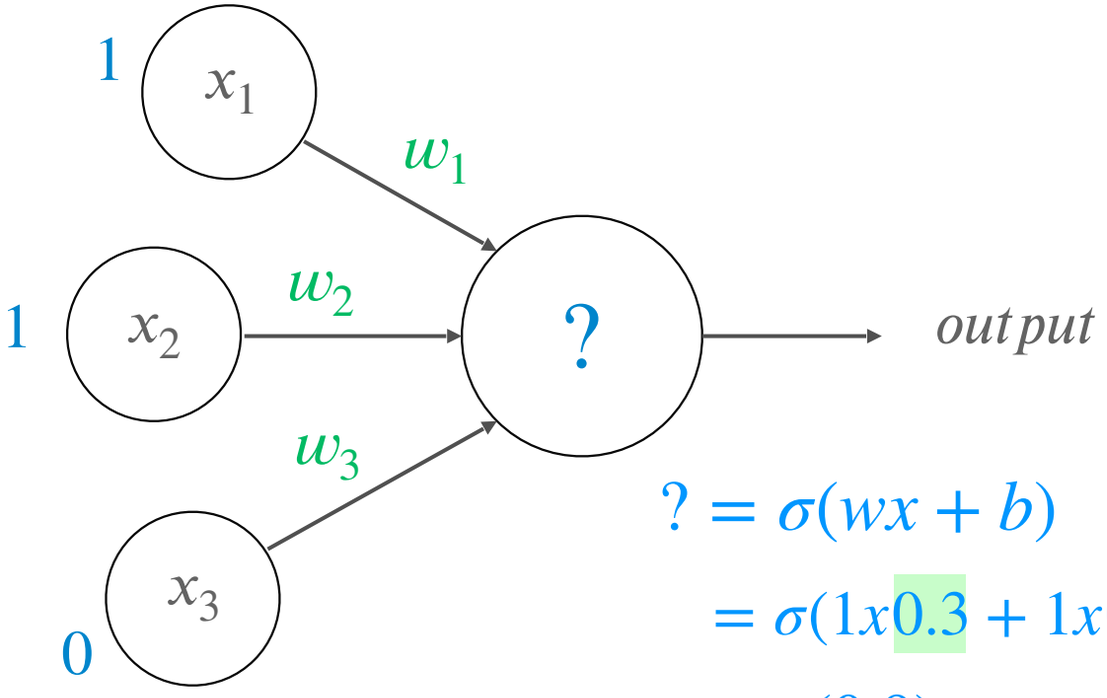


Instead of [0, 1], now (0…1)

Where output is defined by $\sigma(wx+b)$

Sigmoid Neuron

Persona: Shredder



$$? = \sigma(wx + b)$$
$$= \sigma(1x0.3 + 1x0.6 + 0x0.1)$$
$$= \sigma(0.9)$$
$$= 0.7109$$

Do I snowboard this weekend?

$x_1 = 1$ (*good weather*)

$w_1 = 0.3$

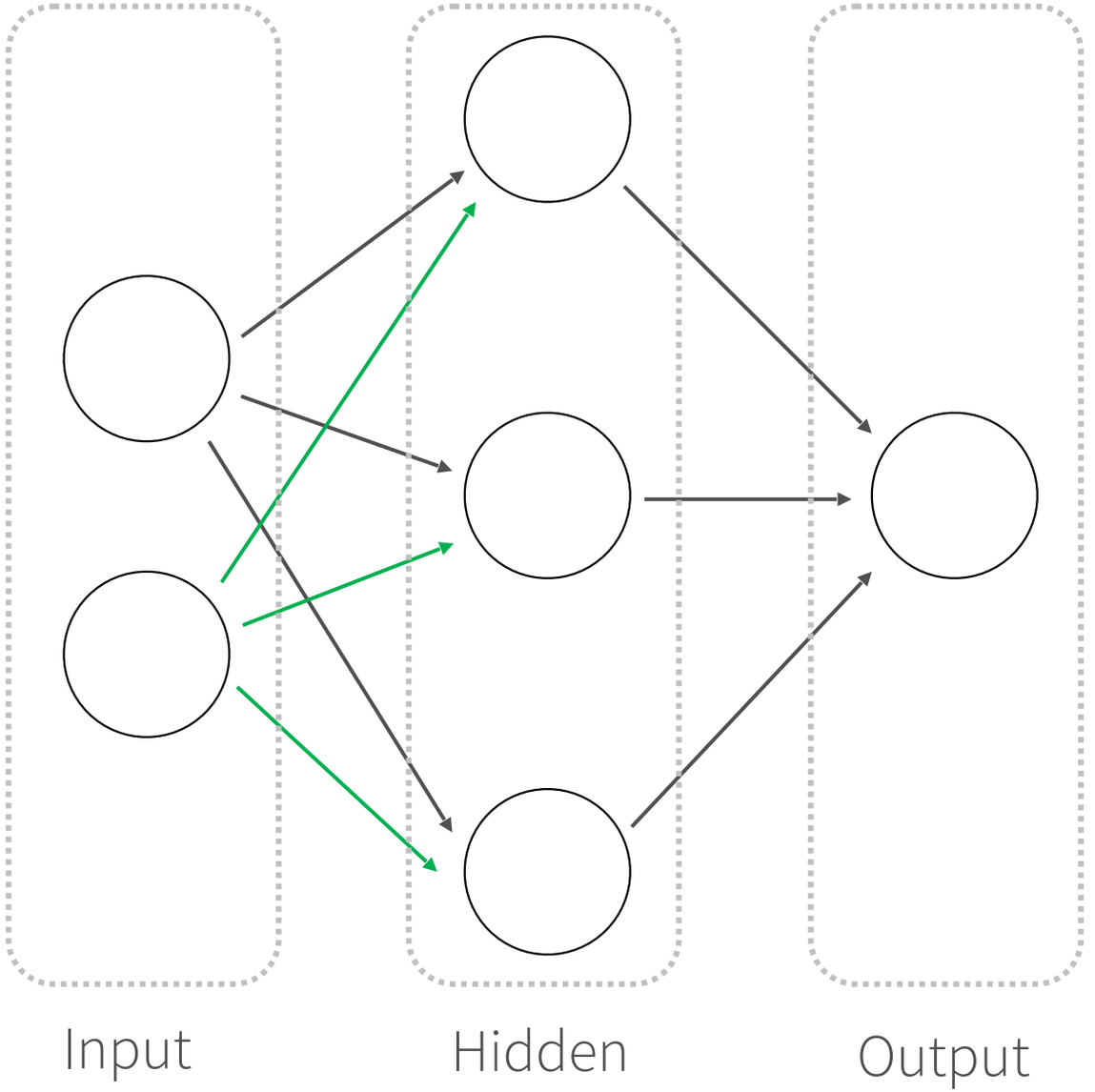
$x_2 = 1$ (*a lot of powder*)

$w_2 = 0.6$

$x_3 = 0$ (*driving sucks*)

$w_3 = 0.1$

Simplified Two-Layer ANN



Do I snowboard this weekend?

$x_1 \rightarrow$ *Apres Ski'er*

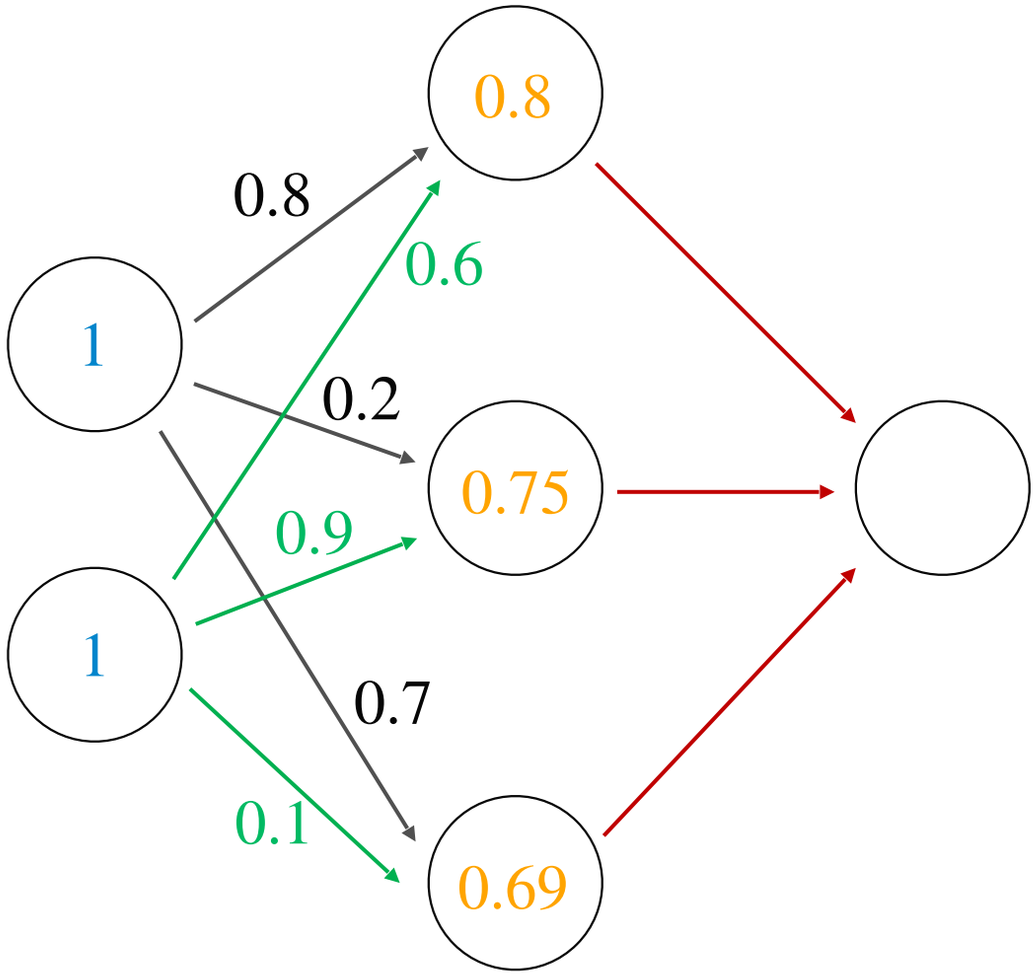
$x_2 \rightarrow$ *Shredder*

$h_1 \rightarrow$ *weather*

$h_2 \rightarrow$ *powder*

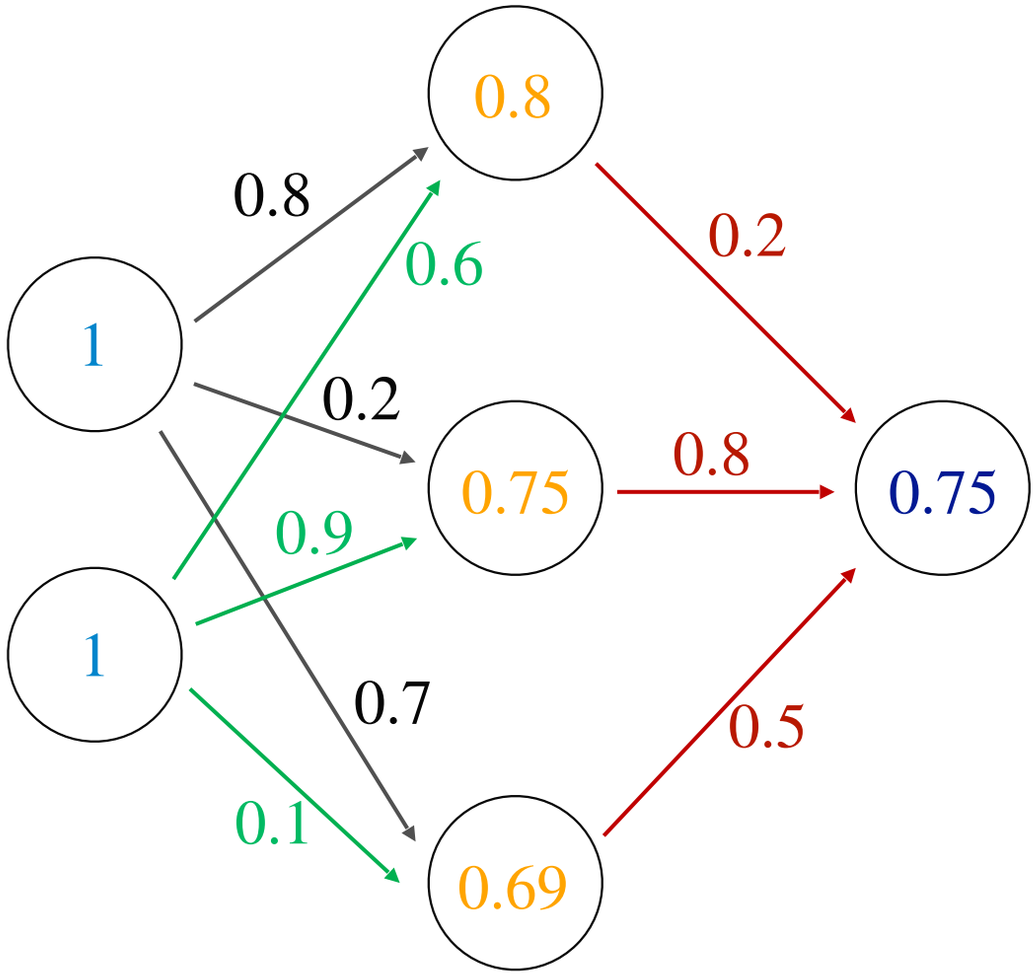
$h_3 \rightarrow$ *driving*

Simplified Two-Layer ANN



$$h_1 = \sigma(1x0.8 + 1x0.6) = 0.80$$
$$h_2 = \sigma(1x0.2 + 1x0.9) = 0.75$$
$$h_3 = \sigma(1x0.7 + 1x0.1) = 0.69$$

Simplified Two-Layer ANN



$$out = \sigma(0.2x0.8 + 0.8x0.75 + 0.5x0.69)$$

$$= \sigma(1.105)$$

$$= 0.75$$

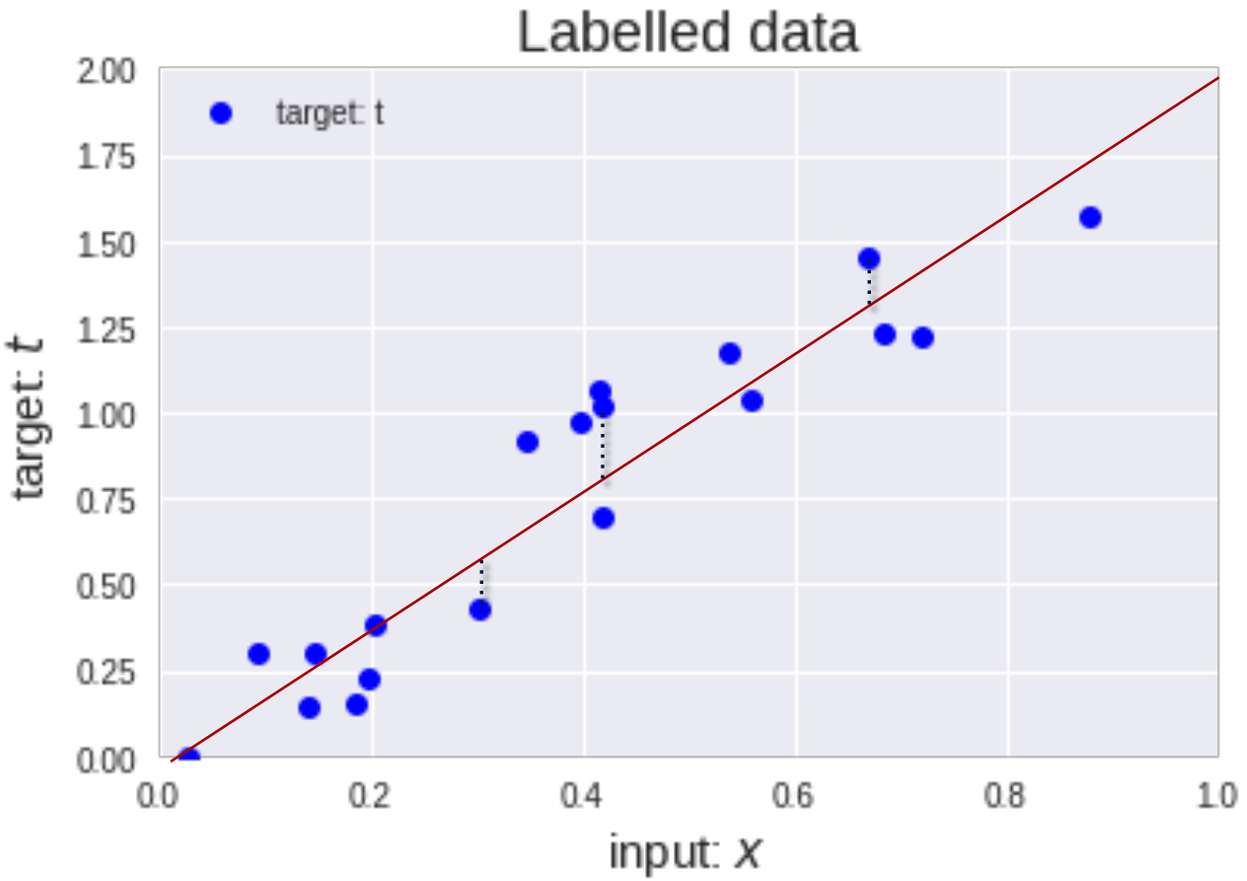
The Immense Power of Simple Structures

Great Resources

- Andrej Karpathy's CS231N Neural Networks
- Steve Miller's How to build a neural network
- Michael Nielsen's Neural Networks and Deep Learning, Chapter 1

Optimization Primer

Cost function



Source: https://bit.ly/2IoAGzL

For this linear regression example, to determine the best p (slope of the line) for

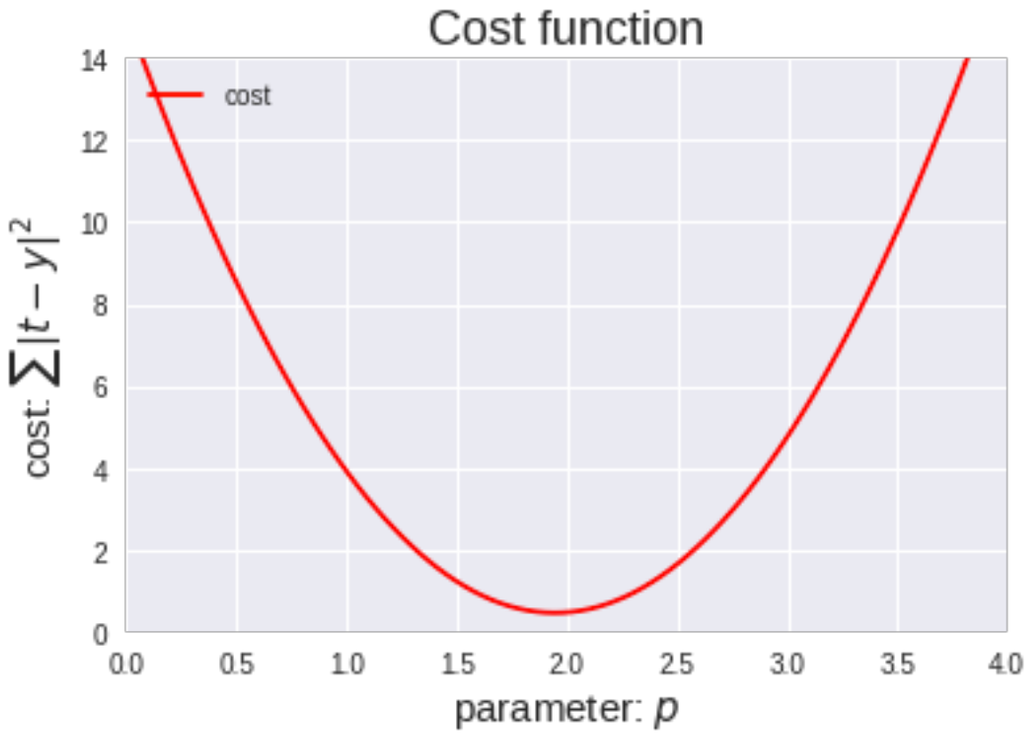
$$y = x \cdot p$$

we can calculate the cost function, such as Mean Square Error, Mean absolute error, Mean bias error, SVM Loss, etc.

For this example, we'll use sum of squared absolute differences

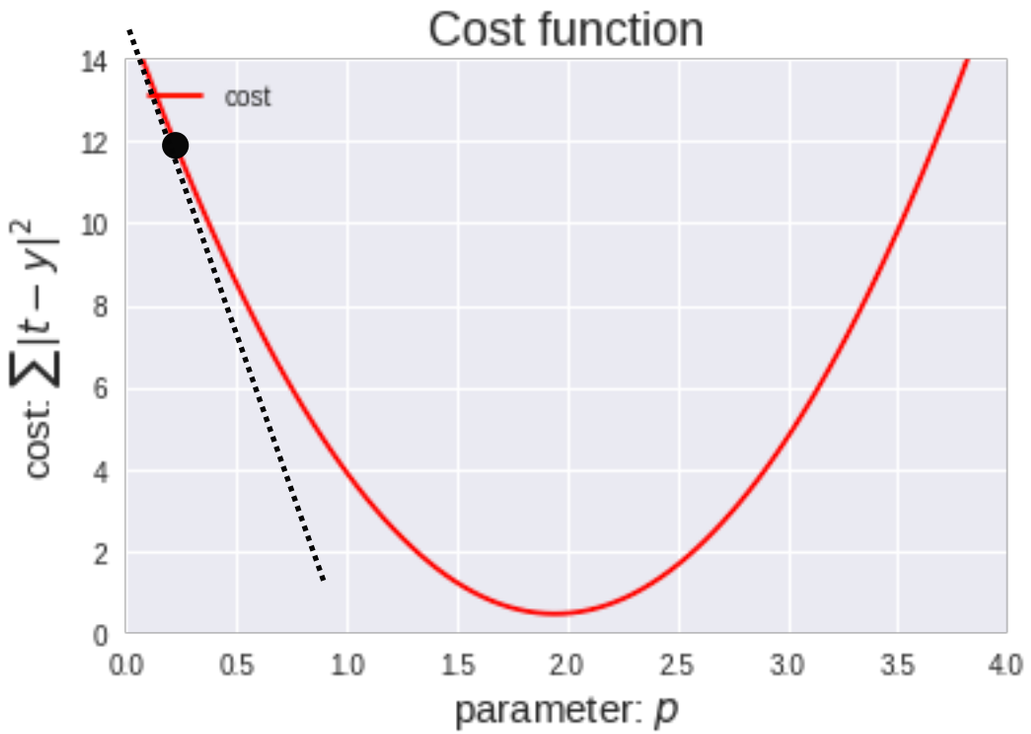
$$cost = \sum |t - y|^2$$

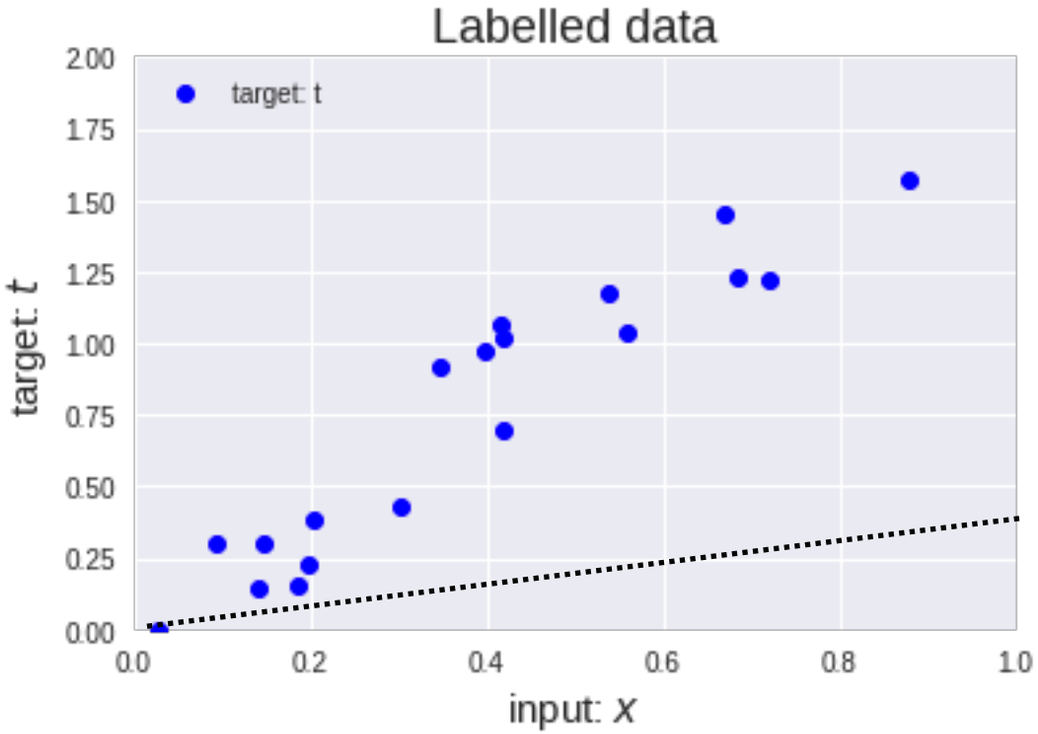
Visualize this cost function



Source: https://bit.ly/2IoAGzL

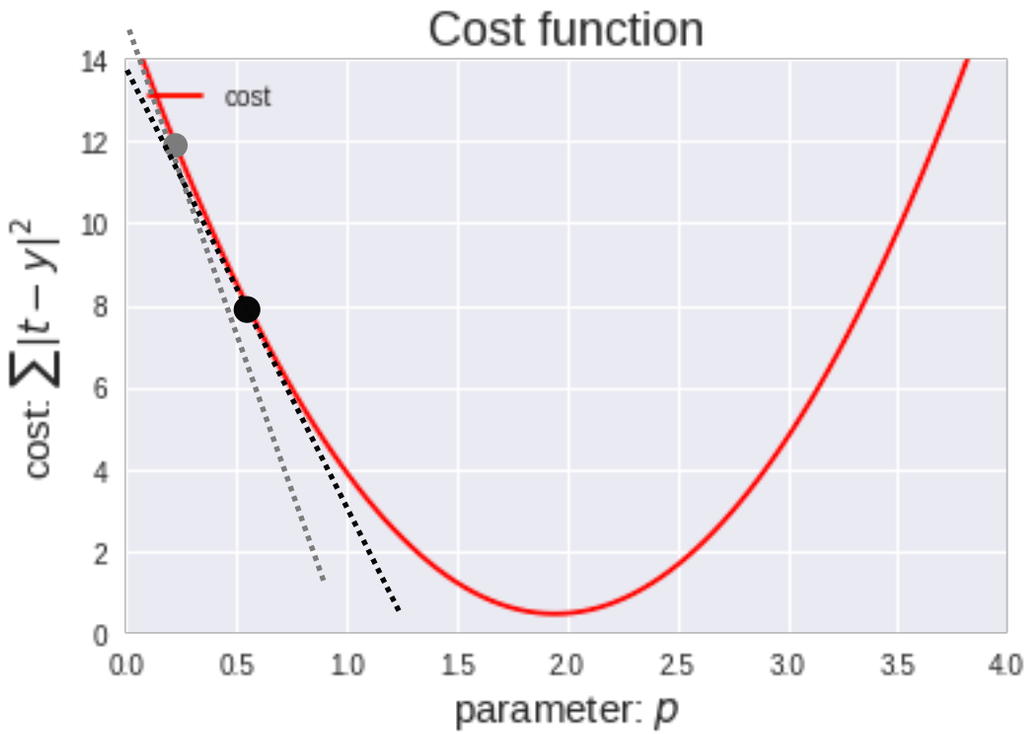
Calculate its derivative

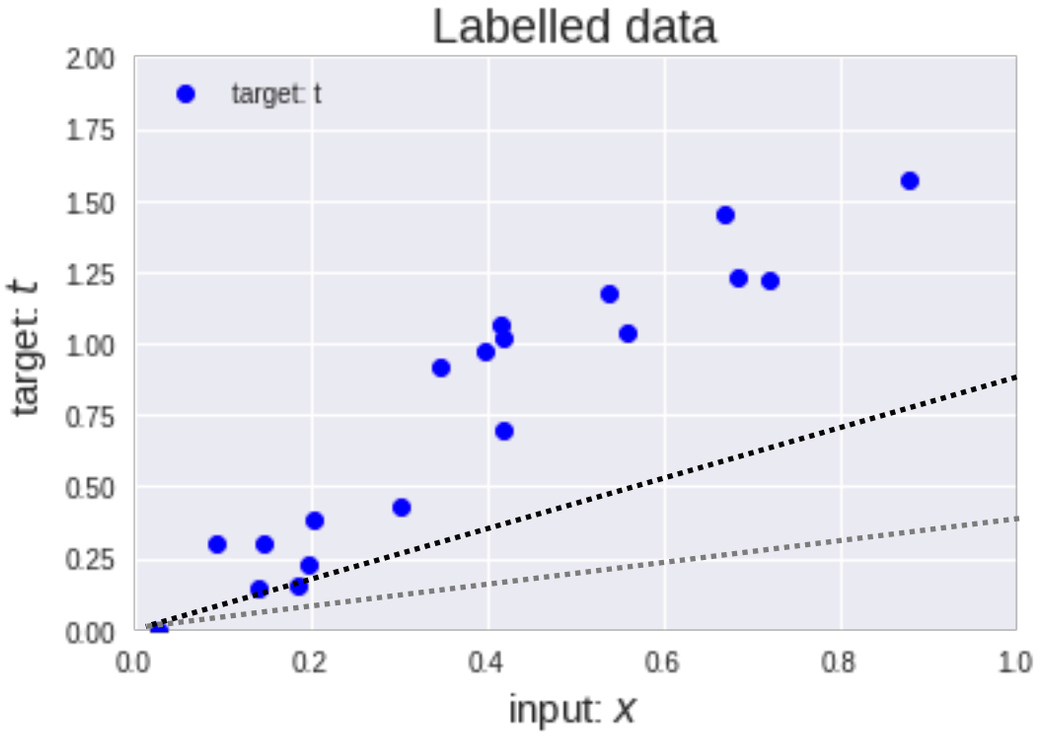




Source: https://bit.ly/2IoAGzL

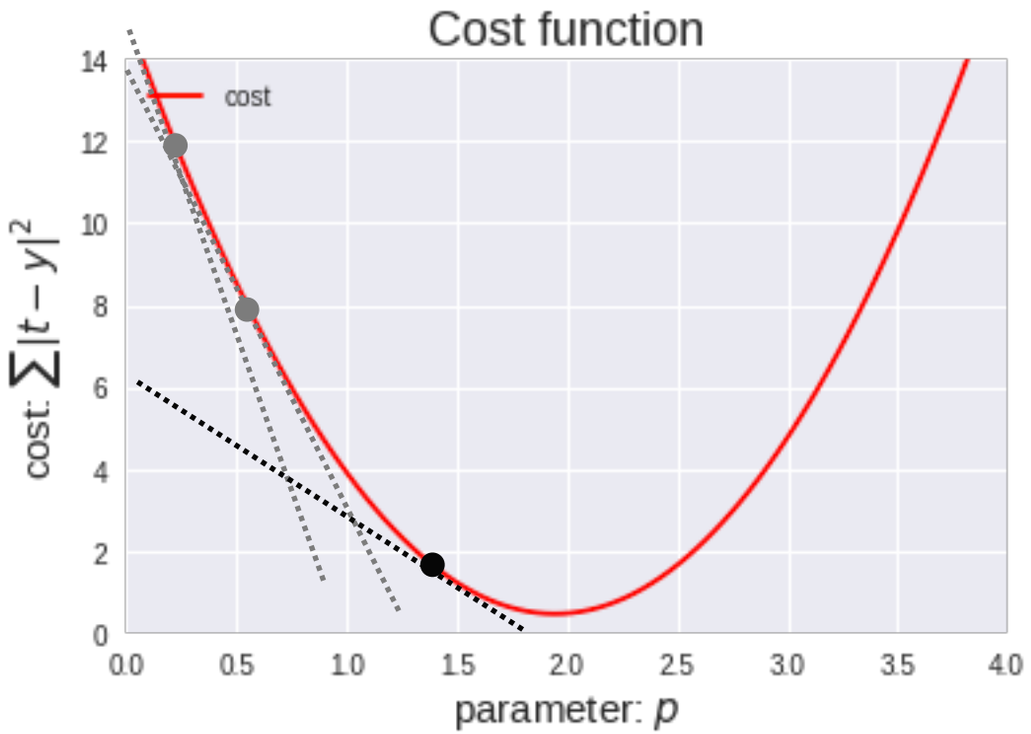
Gradient Descent

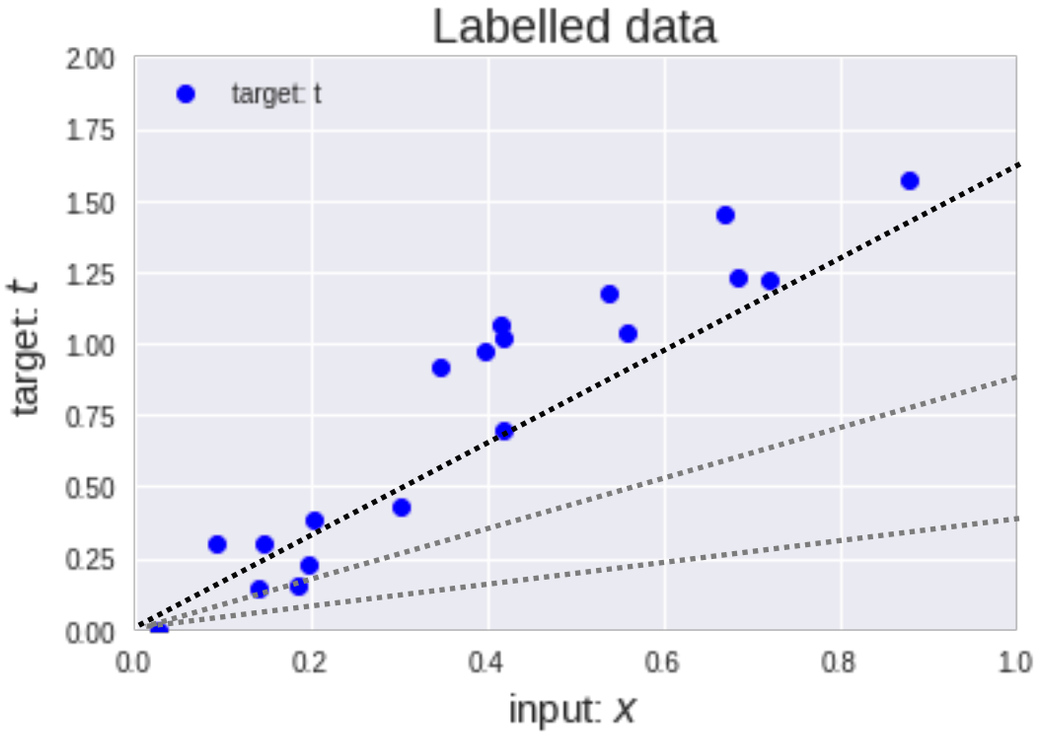




Source: https://bit.ly/2IoAGzL

Gradient Descent

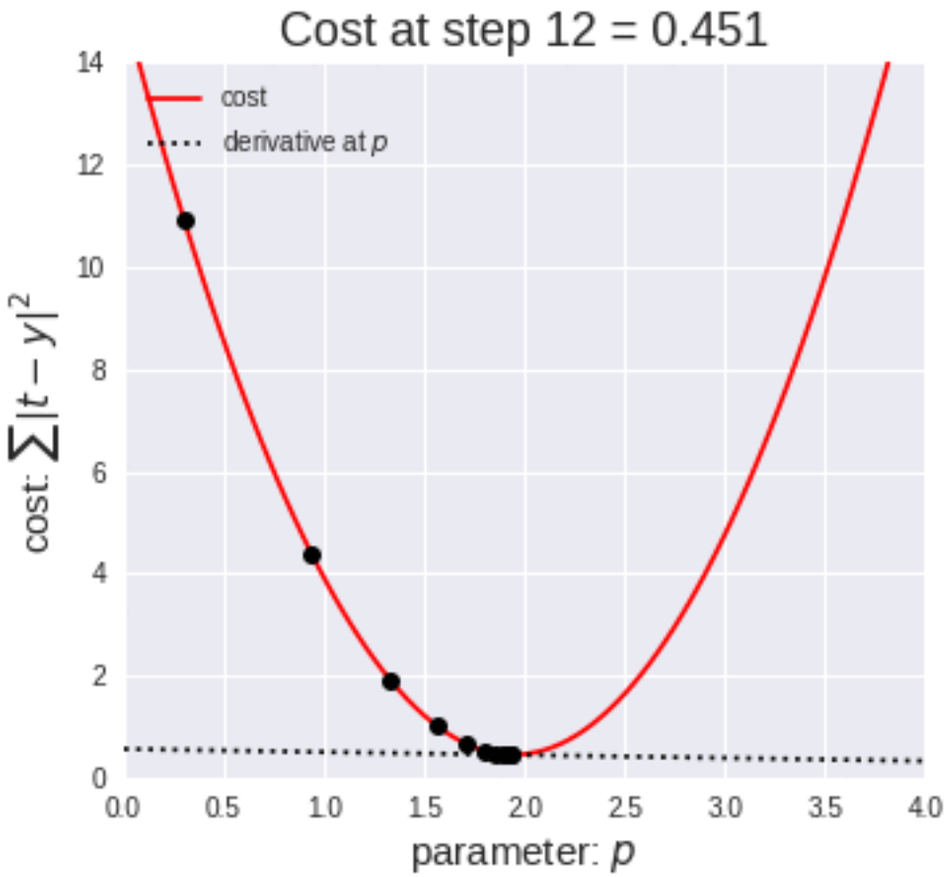


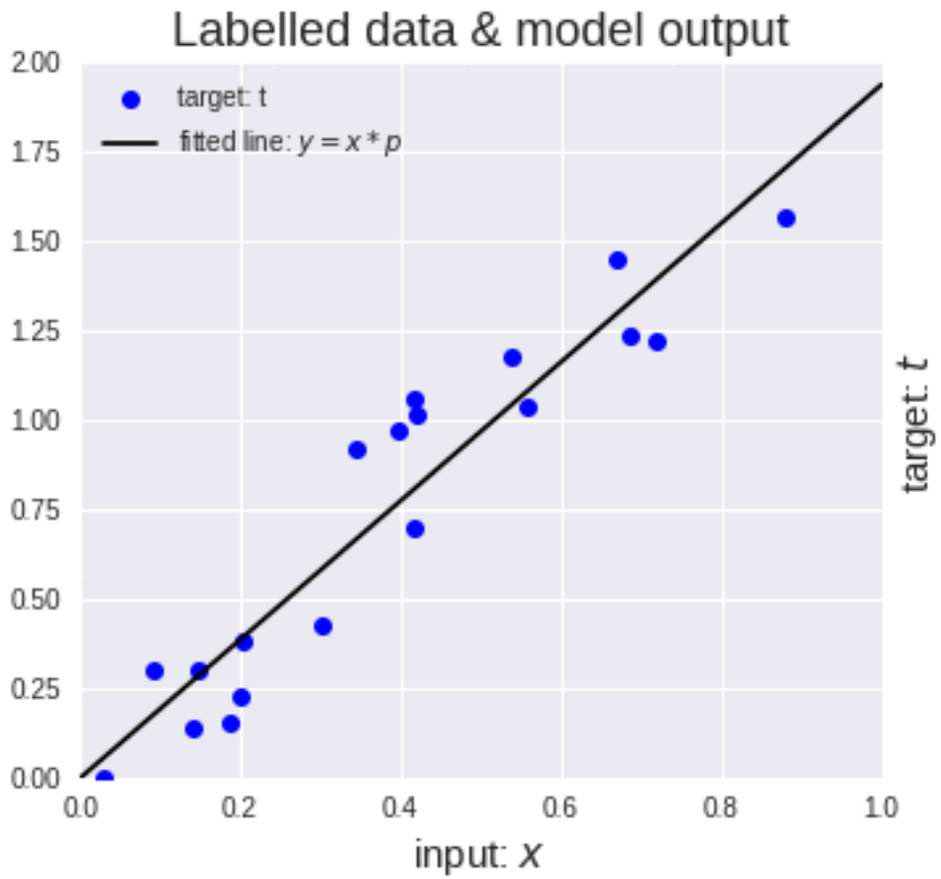


The goal is to find the lowest point of the cost function (i.e. least amount of cost or the minimum or *minima*). Gradient descent iteratively (i.e. step by step) descends the cost function curve to find the minima.

Source: https://bit.ly/2IoAGzL

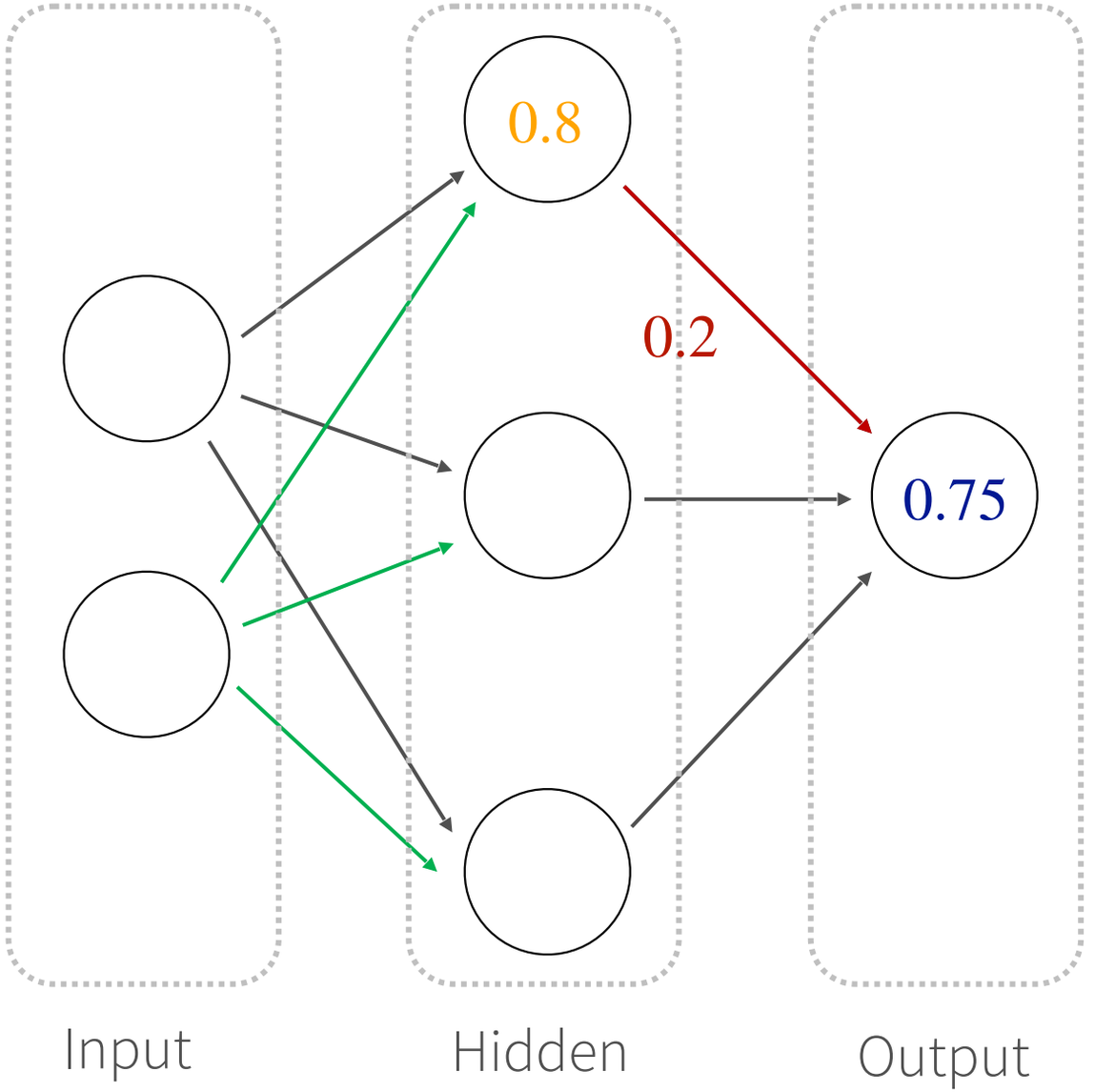
Gradient Descent Optimization



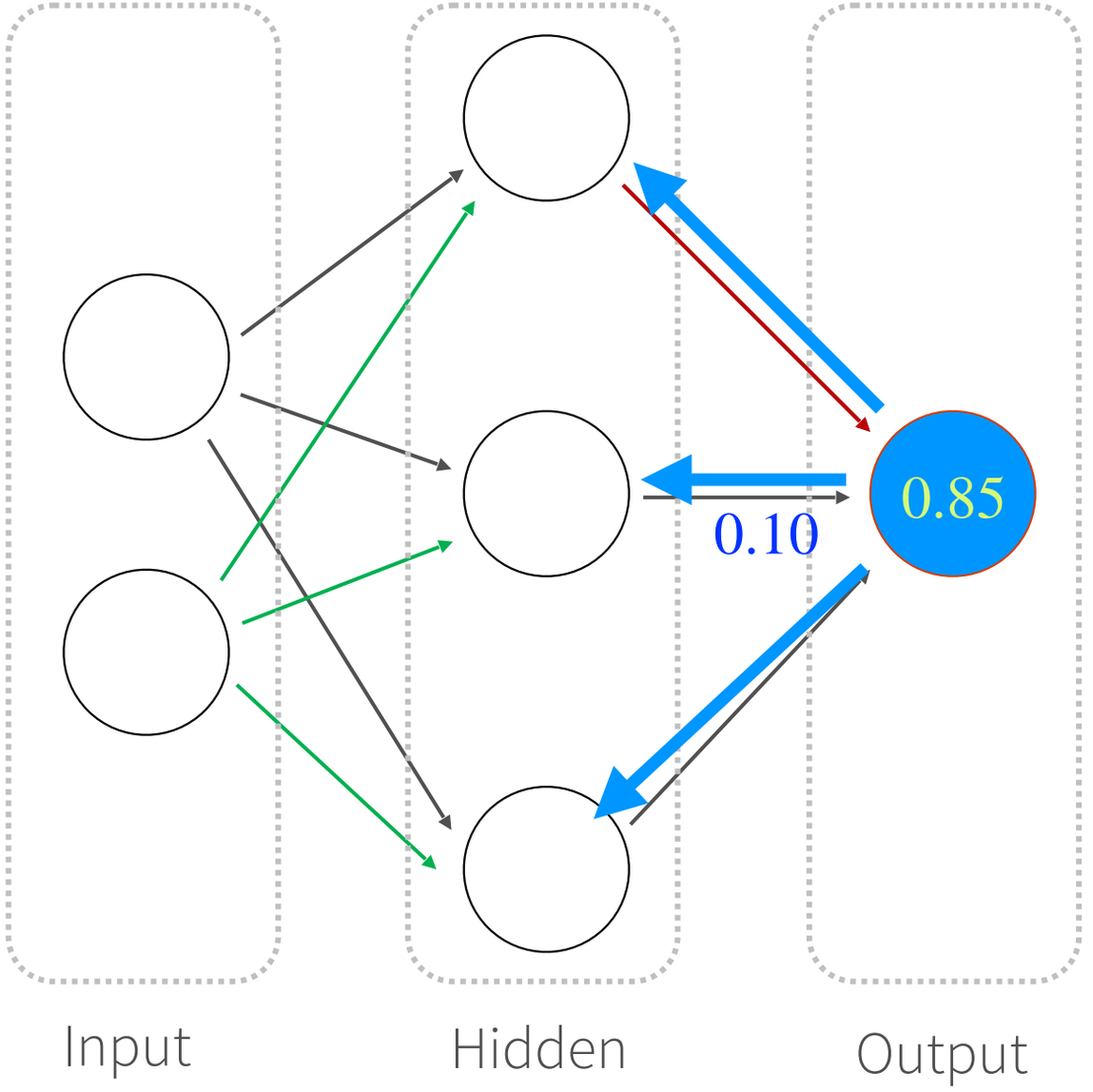


Source: https://bit.ly/2IoAGzL

Backpropagation



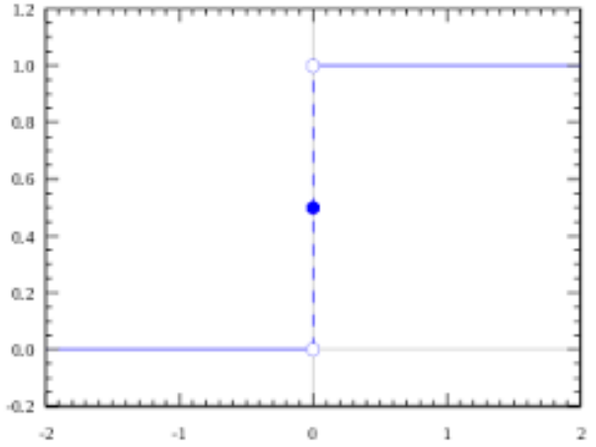
Backpropagation



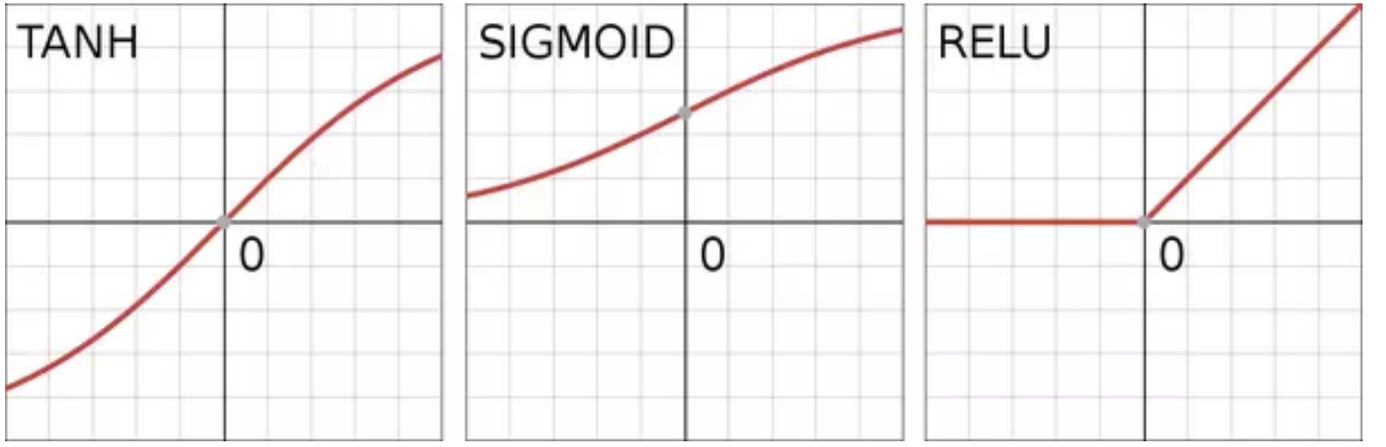
- Backpropagation: calculate the gradient of the cost function in a neural network
- Used by gradient descent optimization algorithm to adjust weight of neurons
- Also known as backward propagation of errors *as the error is calculated and distributed back through the network of layers*

Neurons . . . Activate!

Activation functions



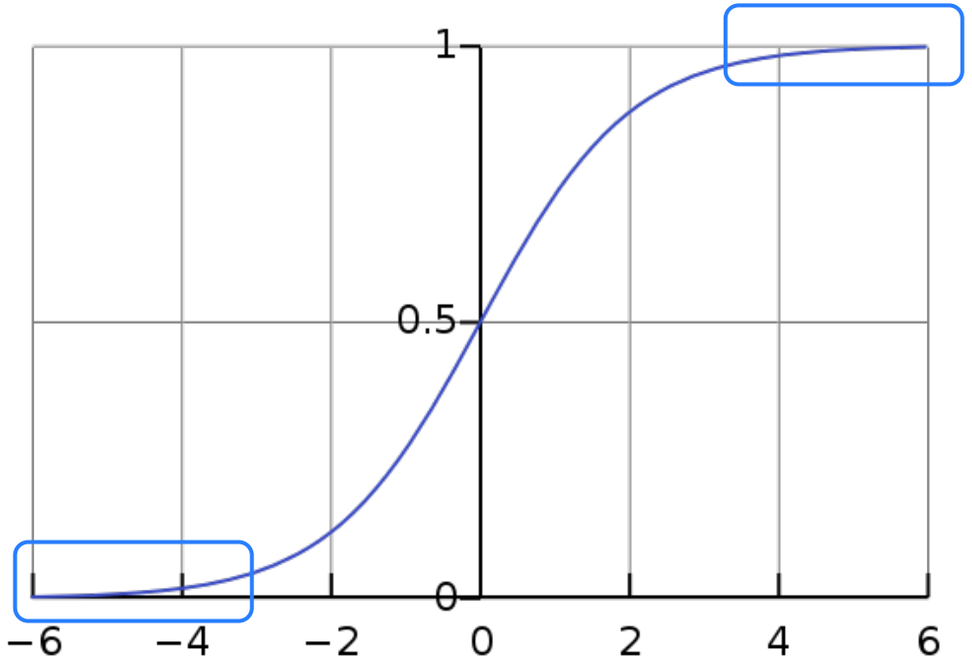
Source: https://bit.ly/2wv3kcd



Source: http://bit.ly/2tKhbMS

- Bias (threshold) activation function was proposed first
- Sigmoid and tanh introduce non-linearity with different codomains
- ReLU is one of the more popular ones because its simple to compute and very robust to noisy inputs

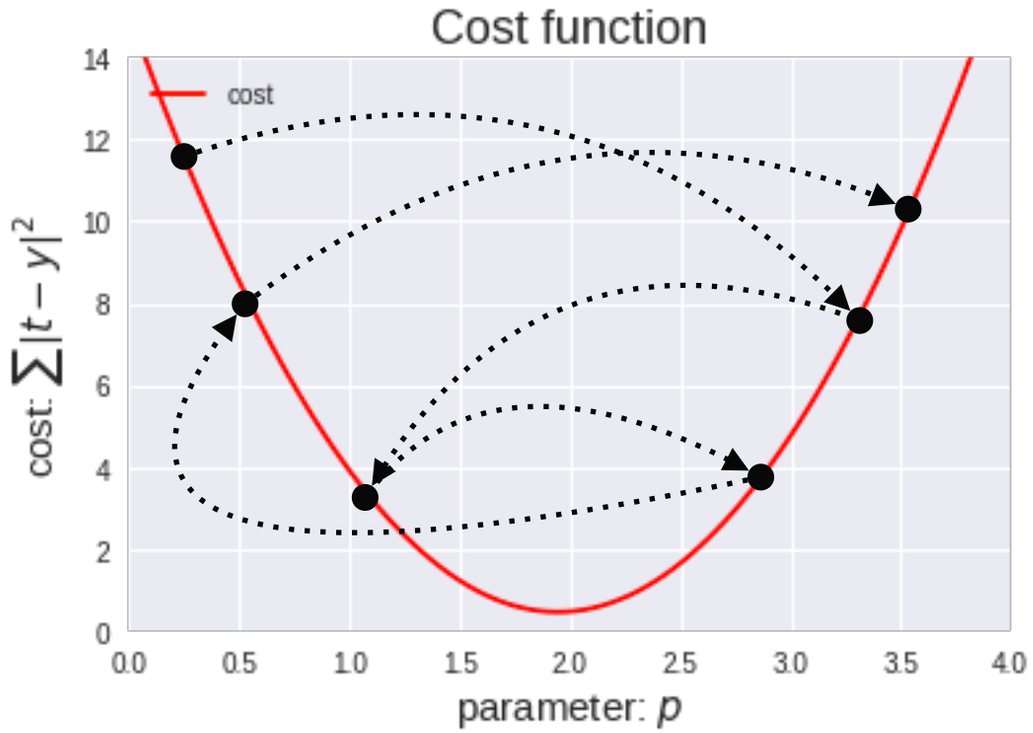
Sigmoid function



Source: http://bit.ly/2GgMbGW

- Sigmoid non-linearity squashes real numbers between [0, 1]
- Historically a nice interpretation of neuron firing rate (i.e. not firing at all to fully-saturated firing).
- Currently, not used as much because really large values too close to 0 or 1 result in gradients too close to 0 stopping backpropagation.

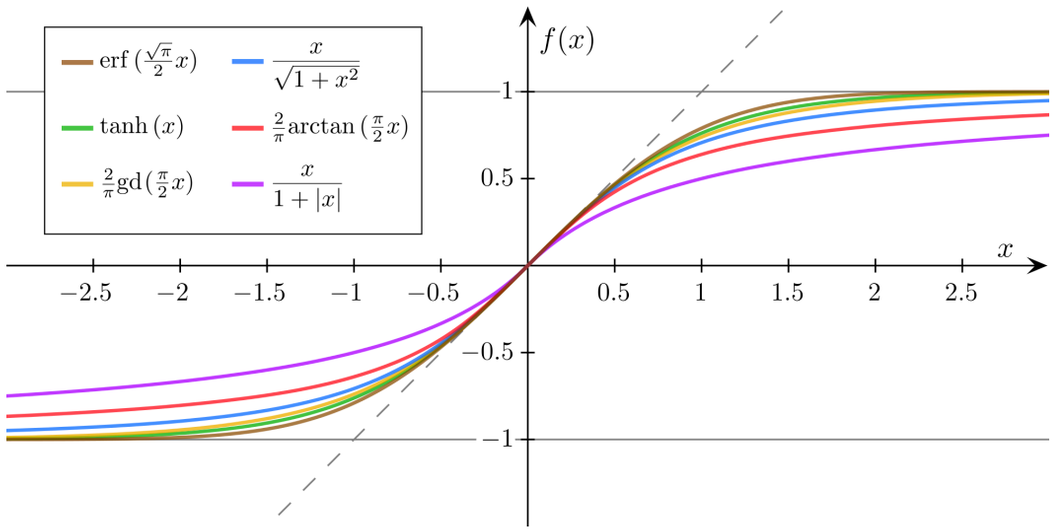
Sigmoid function (continued)



Source: https://bit.ly/2IoAGzL

Output is not zero-centered: During gradient descent, if all values are positive then during backpropagation the weights will become all positive or all negative creating zig zagging dynamics.

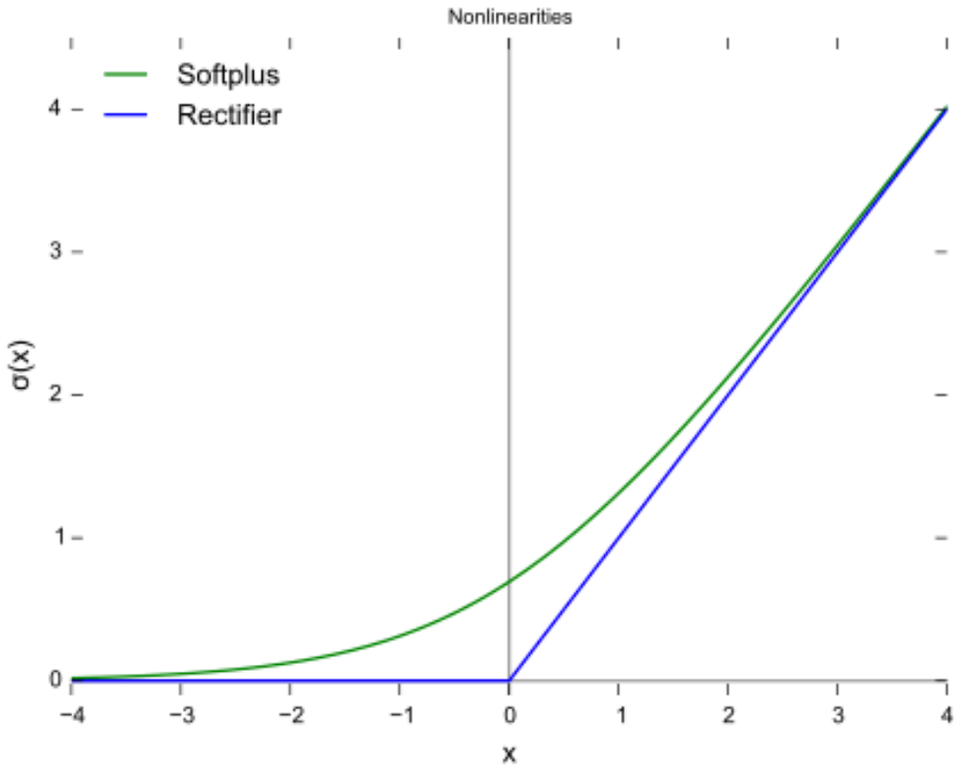
Tanh function



Source: http://bit.ly/2C4y89z

- Tanh function squashes real numbers [-1, 1]
- Same problem as sigmoid that its activations saturate thus killing gradients.
- But it is zero-centered minimizing the zig zagging dynamics during gradient descent.
- Currently preferred sigmoid nonlinearity

ReLU: Rectifier Linear Unit



Source: http://bit.ly/2EwRndsCC0

ReLU's activation is at threshold of zero

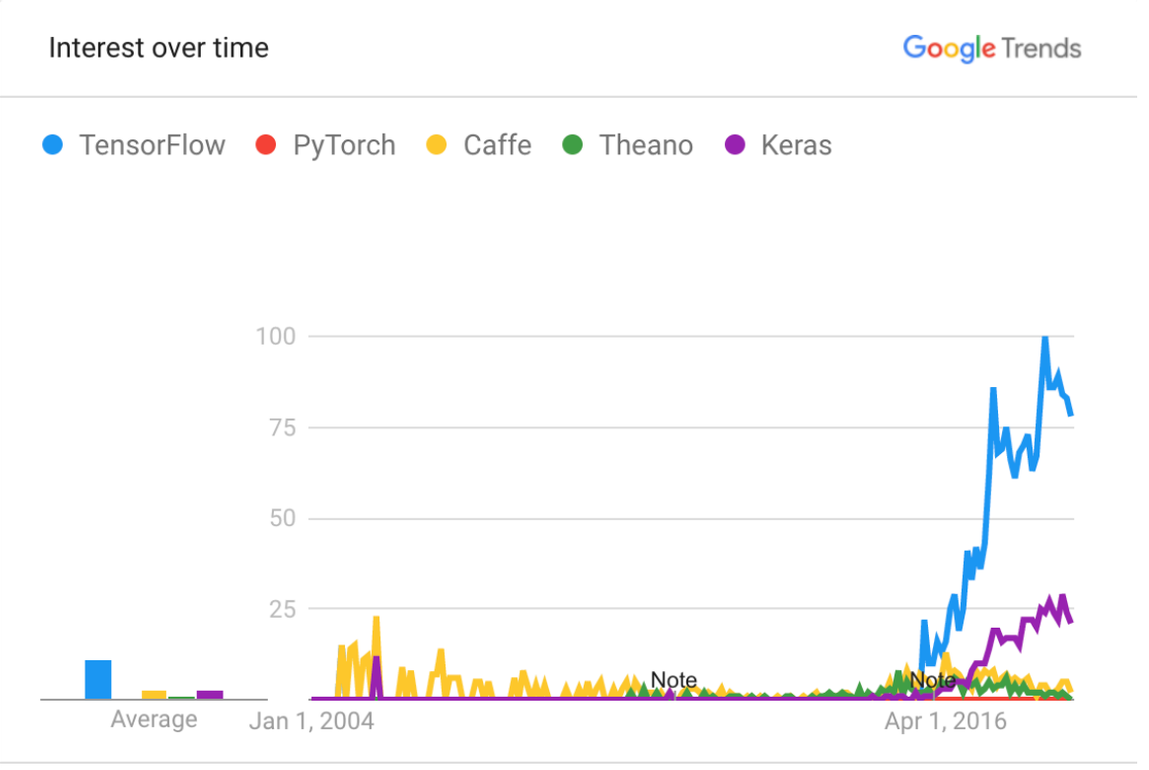
- Quite popular over the last few years
- Speeds up Stochastic Gradient Descent (SGD) convergence
- It is easier to implement due to simpler mathematical functions
- Sensitive to high learning rate during training resulting in "dead" neurons (i.e. neurons that will not activate across the entire dataset).

DEMO

Neurons . . . Activate!

What is TensorFlow

- Open source software library for numerical computation using data flow graphs
- Based loosely on neural networks
- Built in C++ with a Python interface
- Quickly gained high popularity
- Supports GPU computations



What is Keras

- Python Deep Learning Library
- High level neural networks API
- Runs on top of TensorFlow, CNTK, or Theano
- Simpler and easier to use making it easier for rapid prototyping

I’d like to thank…

Great References

- Machine Learning 101
- Andrej Karparthy's ConvNetJS MNIST Demo
- What is back propagation in neural networks?
- CS231n: Convolutional Neural Networks for Visual Recognition
  - Syllabus and Slides | Course Notes | YouTube
  - With particular focus on CS231n: Lecture 7: Convolution Neural Networks
- Neural Networks and Deep Learning
- TensorFlow

Great References

- Deep Visualization Toolbox
- What's the difference between gradient descent and stochastic gradient descent?
- Back Propagation with TensorFlow
- TensorFrames: Google TensorFlow with Apache Spark
- Integrating deep learning libraries with Apache Spark
- Build, Scale, and Deploy Deep Learning Pipelines with Ease

Attribution

Tomek Drabas

Brooke Wenig

Timothee Hunter

Cyrielle Simeone

Q&A

What's next?

Training your Neural Networks

October 9, 2018 | 09:00 PDT

https://dbricks.co/2pugWSC