FINAL TUTORIAL

In lieu of a final exam, you'll create a mini-tutorial that:

- Identifies a raw data source
- Processes and stores that data
- Performs exploratory data analysis & visualization
- Derives insight(s) using statistics and ML
- Communicates those insights as actionable text

Individual or group project – 25% of final grade!

Will be hosted publicly online (GitHub Pages) and will strengthen your portfolio.

FINAL TUTORIAL

Deliverable: URL of your own GitHub Pages site hosting an .ipynb/.html export of your final tutorial

- https://pages.github.com/ – make a GitHub account, too!
- https://github.com/blog/1995-github-jupyter-notebooks-3

The project itself:

- ~1500+ words of Markdown prose
- ~150+ lines of Python
- Should be viewable as a **static webpage** – that is, if I (or anyone else) opens the link up, everything should render and I shouldn't have to run any cells to generate output

FINAL TUTORIAL RUBRIC

It is graded on a scale of 1-10:

Motivation: Does the tutorial make the reader believe the topic is important (a) in general and (b) with respect to data science?

Understanding: After reading the tutorial, does the reader understand the topic?

Further resources: Does the tutorial "call out" to other resources that would help the reader understand basic concepts, deep dive, related work, etc?

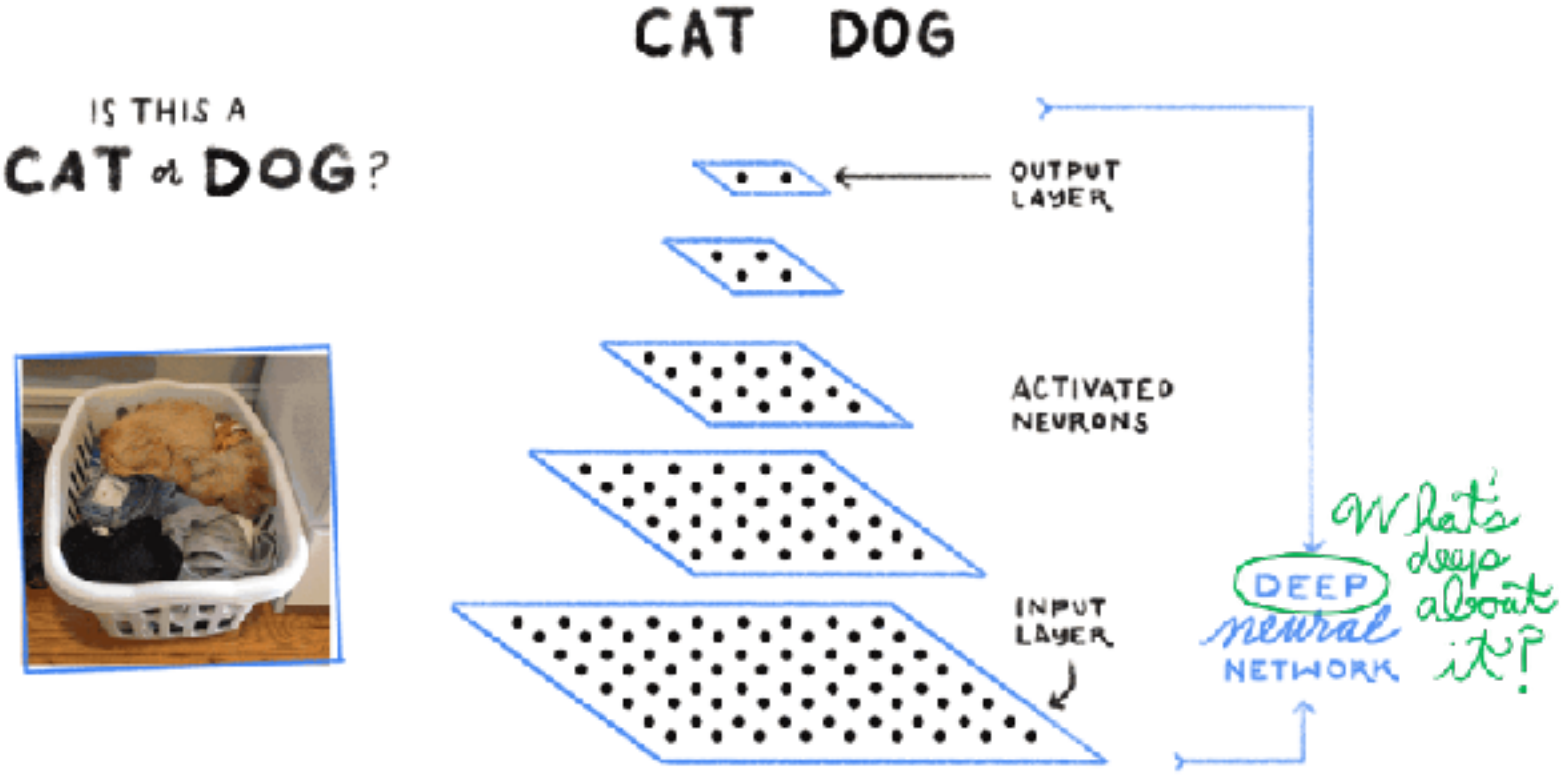
Prose: Does the prose in the Markdown portion of the .ipynb add to the reader's understanding of the tutorial?

Code: Does the code help solidify understanding, is it well documented, and does it include helpful examples?

Subjective Evaluation: If somebody linked to this tutorial from Hacker News, would people actually read the whole thing?

Thanks to: Zico Kolter

Artificial Neural Networks



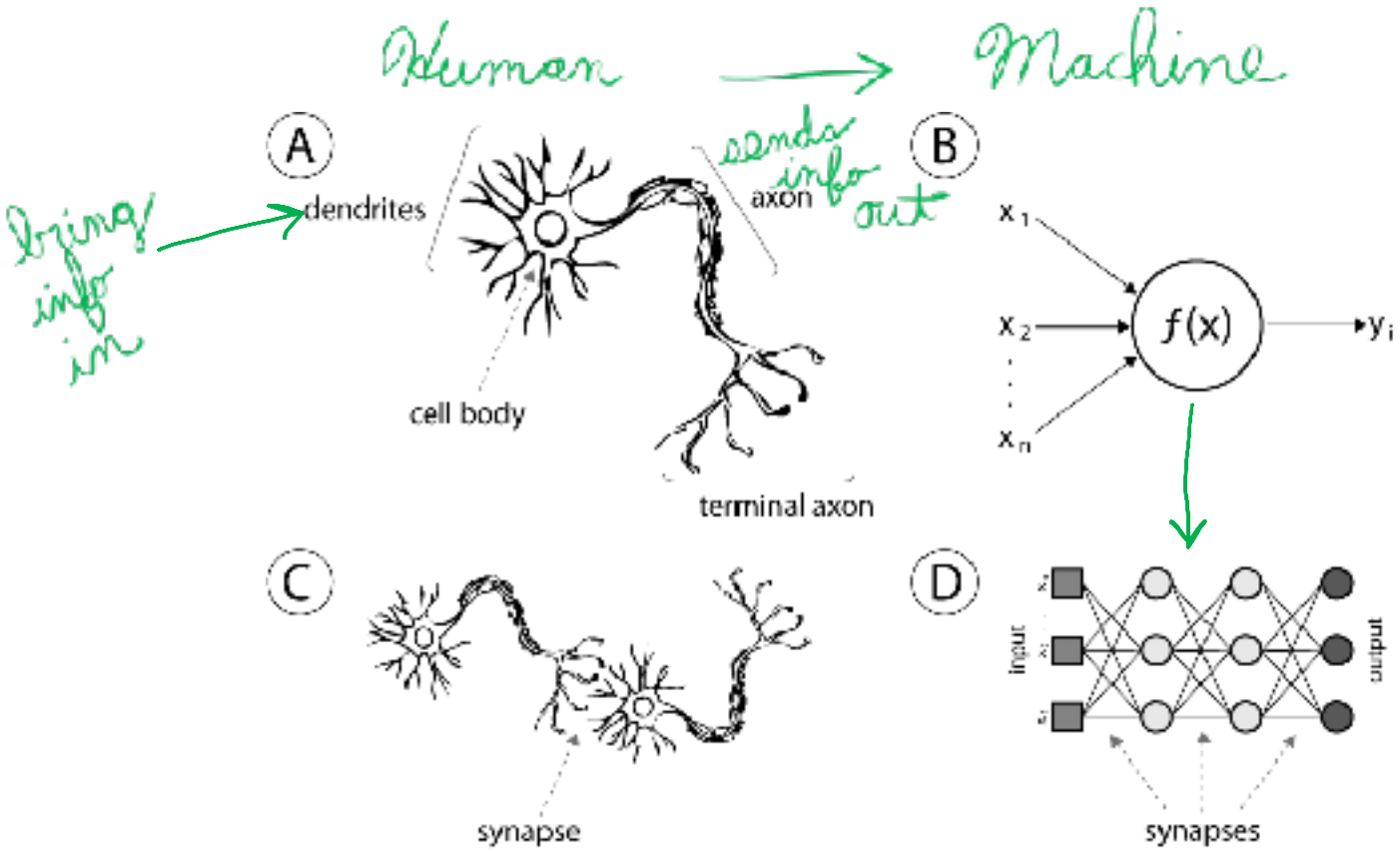
What are they?

Inspired by the Human Brain.

The human brain has about 86 Billion neurons and requires 20% of your body's energy to function.

These neurons are connected to between 100 Trillion to 1 Quadrillion synapses!

What are they?



What are they?

1. Originally developed by Warren McCulloch and Walter Pitts[3] (1943)
2. Started off as an unsupervised learning tool.
    1. Had problems with computing time and could not compute XOR
    2. Was abandoned in favor of other algorithms
3. Werbos's (1975) backpropagation algorithm
    1. Incorporated supervision and solved XOR
    2. But were still too slow vs. other algorithms e.g., Support Vector Machines
4. Backpropagation was accelerated by GPUs in 2010 and shown to be more efficient and cost effective

GPUS

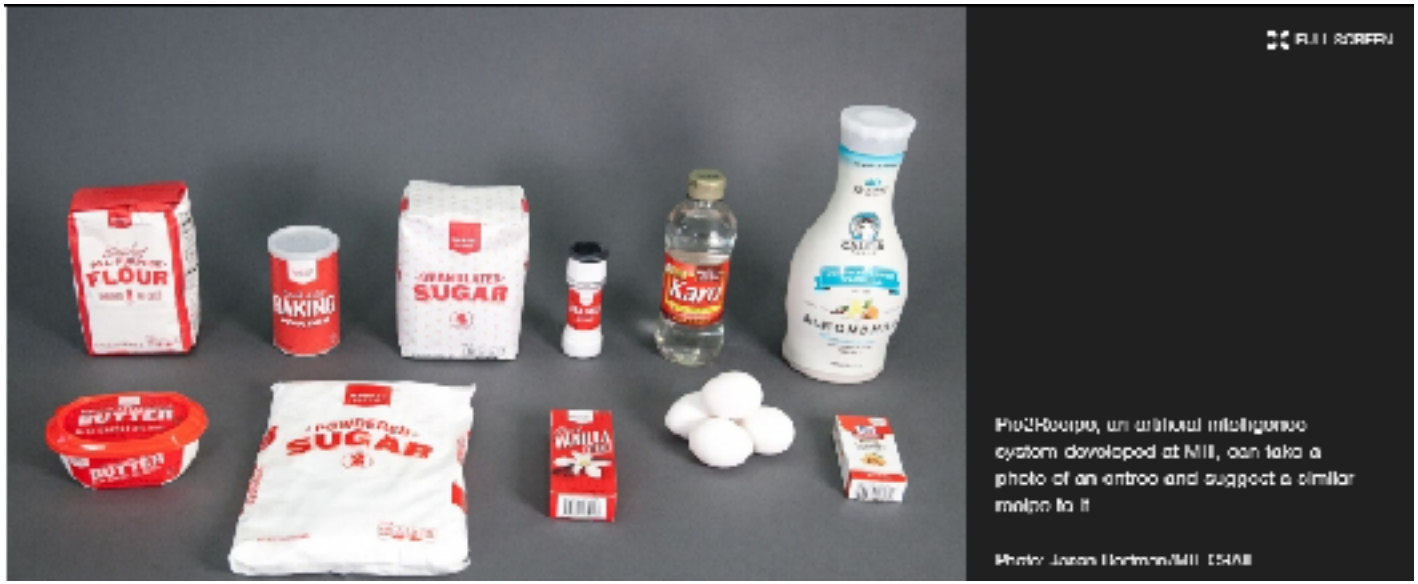
GPUS handle parallel operations much better (thousands of threads per core) but are not as quick as CPUs. However, the matrix multiplication steps in ANNs can be run in parallel resulting in considerable time + cost savings. The best CPUs handle about 50GB/s while the best GPUs handle 750GB/s memory bandwidth.

	CPU i9 Xseries	GeForce GTX 1080
Cores	18 (36 threads)	2560
Clock Speed (GHz)	4.4	1.6G
Memory	Shared	8GB
Price ($)	1799	549

Applications

http://news.mit.edu/2017/artificial-intelligence-suggests-recipes-based-on-food-photos-0720

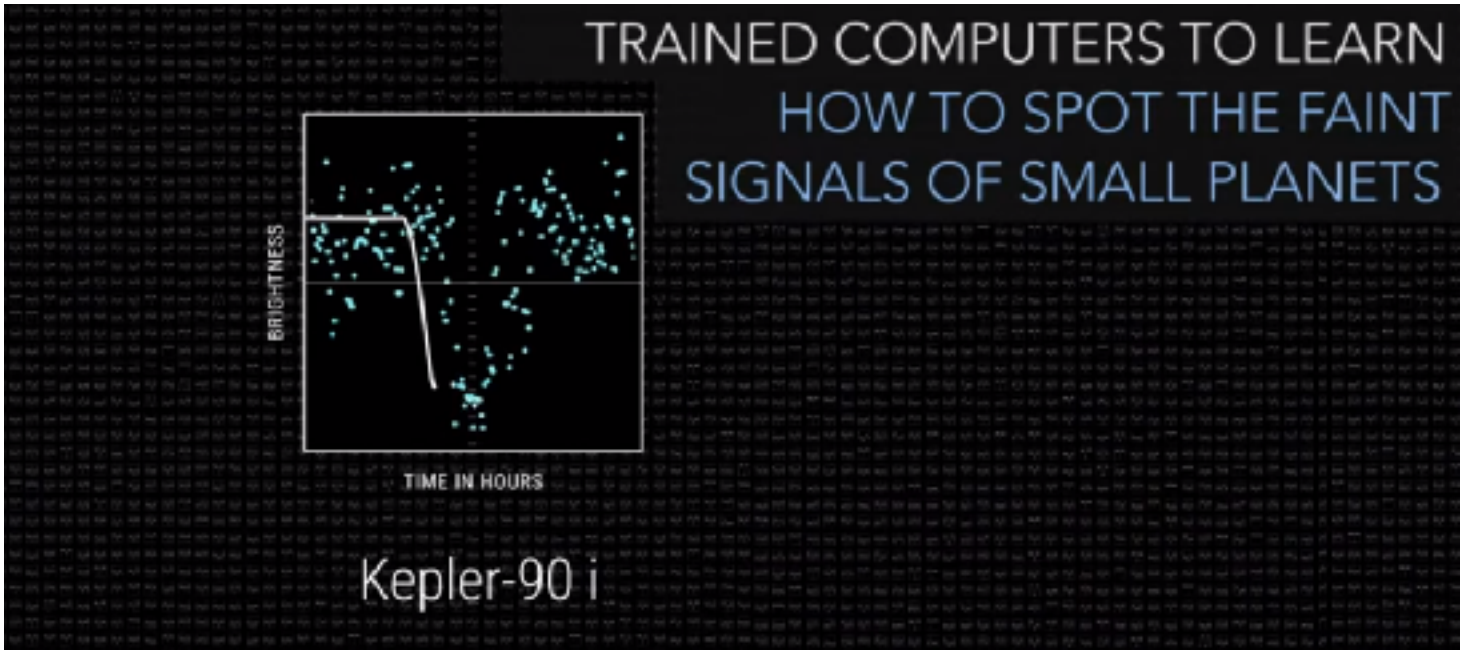
Image to food to ingredients to recipes.



Applications

https://www.nasa.gov/press-release/artificial-intelligence-nasa-data-used-to-discover-eighth-planet-circling-distant-star

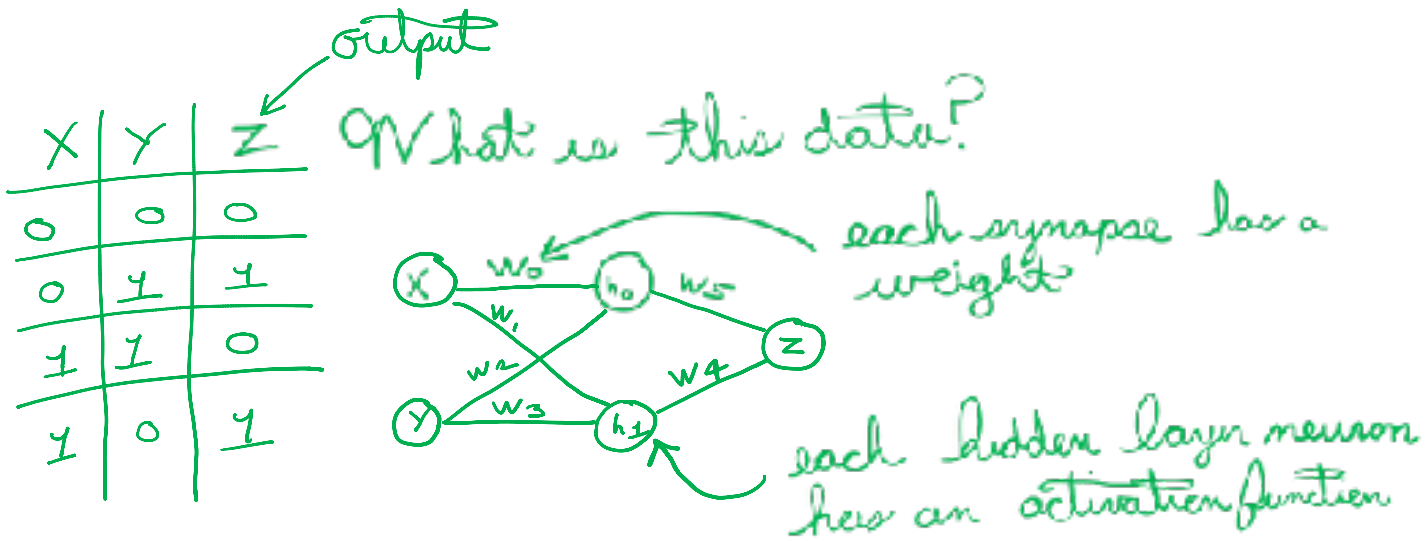
Images of light drop compared and new ones found.



Idea behind them

1. Obtain some structured data (always a good idea ☺️).
2. Use some subset of that data as training
3. Feed each training example through the network
    1. Calculate the error for each training example
    2. Update the weights for each neuron to minimize the error using Gradient Descent (Back Propagation)
    3. Feed in the data again until you reach the desired % error or trials run out
    4. If you reached % error or trials stop and go to the next training input
        1. Else (Back Propagation)

An example


output
What is this data?
X
Y
Z
0 0 0
0 1 1
1 1 0
1 0 1
each synapse has a weight
each hidden layer neuron has an activation function
w_0
w_1
w_2
w_3
w_4
w_5
h_0
h_1