Class Summary

Class 1
Topic: ML Overview and CNNs
Lab: Classifying digits with Keras

Class 2
Topic: Deep learning for image processing/Data augmentation
Labs: Build and improve SOTA classifier for color images (cifar)

Class 3
Topic: Transfer learning and fine-tunding
Lab: Custom Classification on small dataset

Class 4
Topic: Autoencoders/Compression
Lab: Exploring autoencoders

Class 5
Topic: Introduction to text processing/NLP and scimitar
Lab: Twitter text classifier with Sci-kit learn

Class 6
Topic: Recurrent Neural Networks and Time Series
Lab: Predict flu outbreaks over time

Class 7
Topic natural language processing
Lab: sentiment analysis on amazon reviews using LSTMs and GRUs

Class 8
Topic: Translation and seq2seq models
Lab: translation

Class 9
Topic Audio Processing
Lab: Build a simple speech recognizer

Class 10
Topic: Deploying deep learning models
Lab: Build and deploy emotion classifier

Setup Instructions http://bit.ly/hub-setup

Today’s Agenda

4:00-4:15	Introductions and Setup
4:15-4:45	ML Overview
4:45-5:45	Build CNN Classifier
5:45-6:00	Start Fashion Classifier

Who am I?

@l2k
lukas@wandb.com

Who Are You?

Why did you choose to sign up for this class?

19 responses

Do more hands-on machine learning projects
I'm interested in ML and its application and would like to do more of it in my day to day work
Stay up-to-date on ML concepts
interested in ML
Learn basics of ML
I took a course on Coursera a while ago. I want to brush up on the ML skills
Learn ML
A combination of reasons, but the highlights are 1) a general interest in expanding my skillset 2) an ill-defined desire to "predict failures" on our platform 3) learning more about what kind of ML work happens at Bloomberg
Learn neural network fundamentals
I want to learn more and refresh my knowledge about ML.
To understand ML and how it relates to real-life problems/projects
Learn Machine Learning concepts.

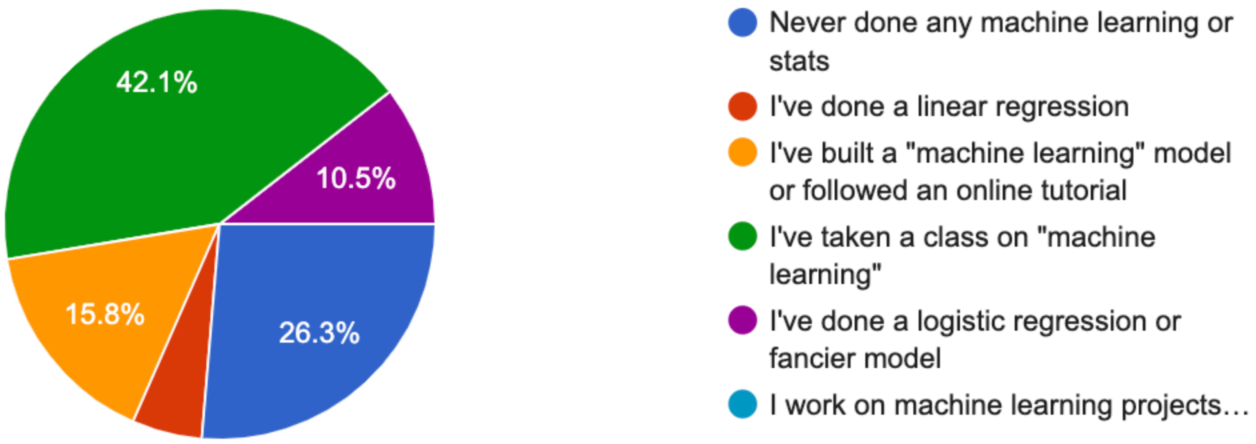
What are you hoping to get out of the class (esp. is there a topic you're hoping I will cover or emphasize?)

19 responses

In particular, any techniques that could be used to parse logs and either: 1) given a large training set of historical logs and outcomes, predict which logfiles in progress are likely to fail; 2) cluster log files into categories of error.
I can visualize what is happening inside a training model
Get a good overview of different methods out there and understand what they are best for or when to use it.
Be able to identify situations/cases where ML will solve the problem and tools/skills to do it.
Neural Networks, how to optimize NN, how to create datasets.
I would like to go breadth first and then dive deeper into some concepts with few problems to solve.
Nothing specific. See previous question
To see how much I forgot from school already.
A better understanding of the various types of models and when to use them
An overall practical understanding of ML and all the different tools to build and train and deploy models. I'm not so much interested in the mathematical details behind everything.
Why does machine learning work, to my knowledge it is never proved mathematically.

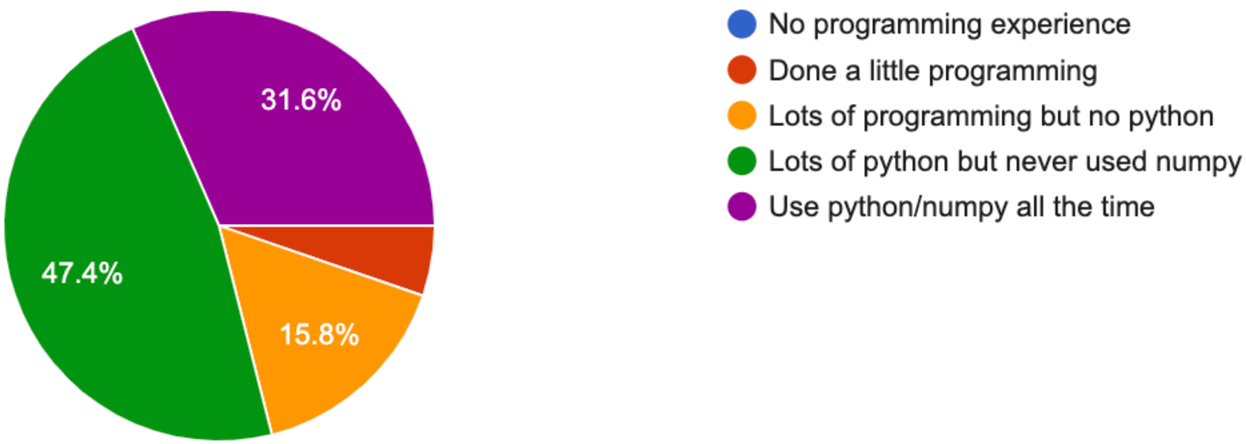
Machine Learning Experience

19 responses



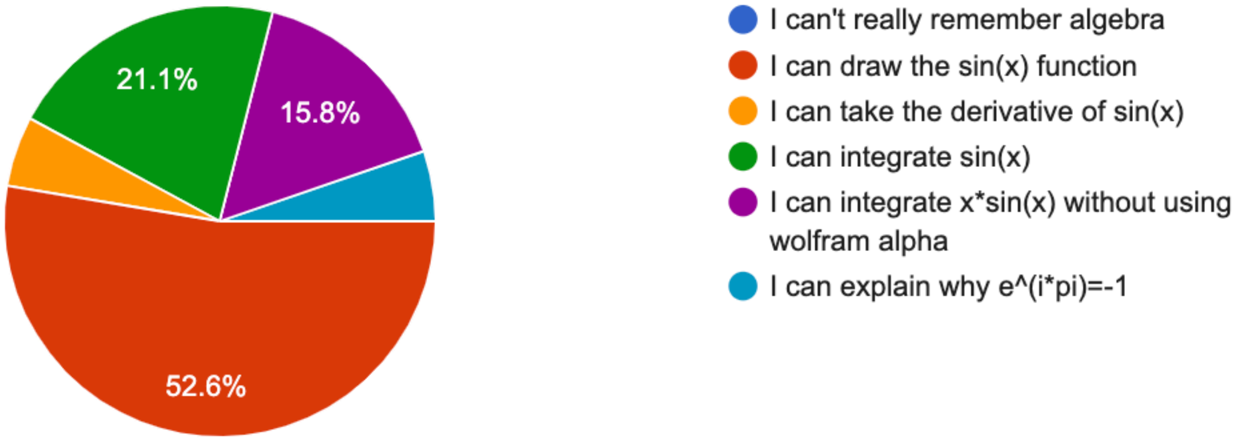
Familiarity with programming tools in class

19 responses



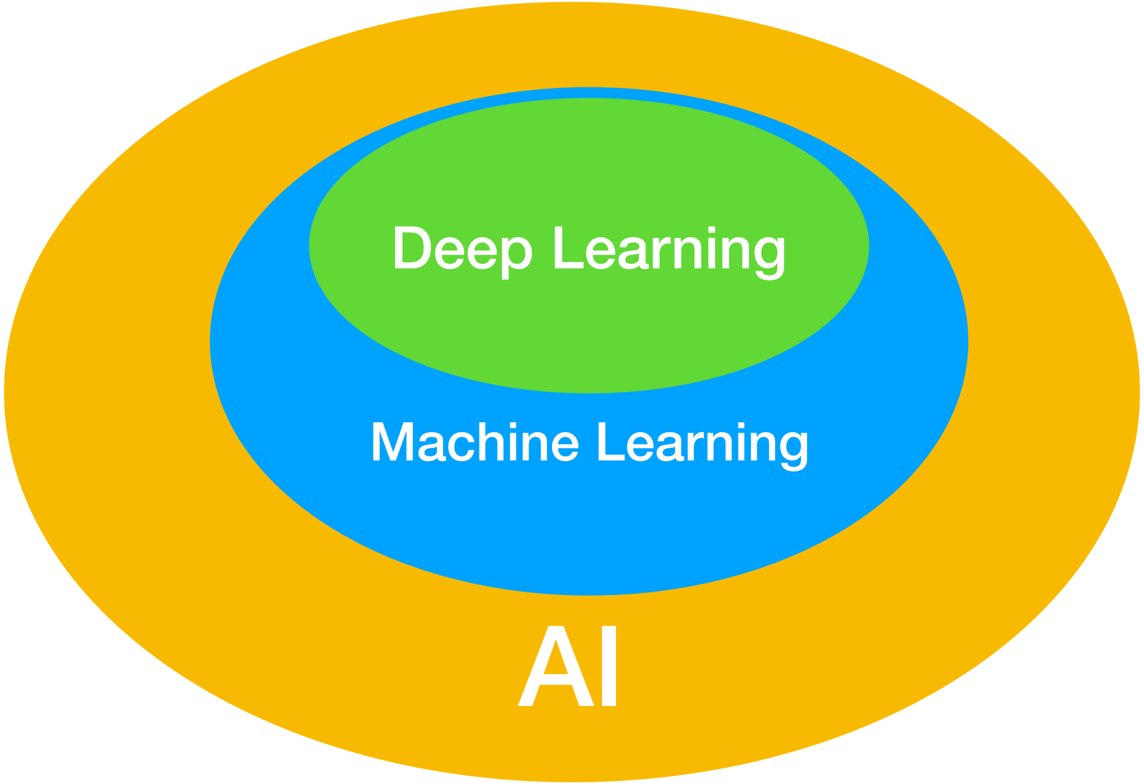
Briefly describe your level of math skills (none is ok!)

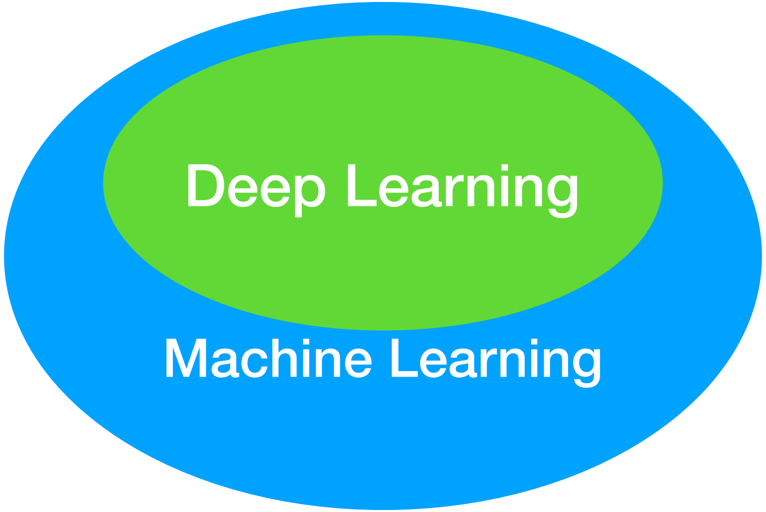
19 responses



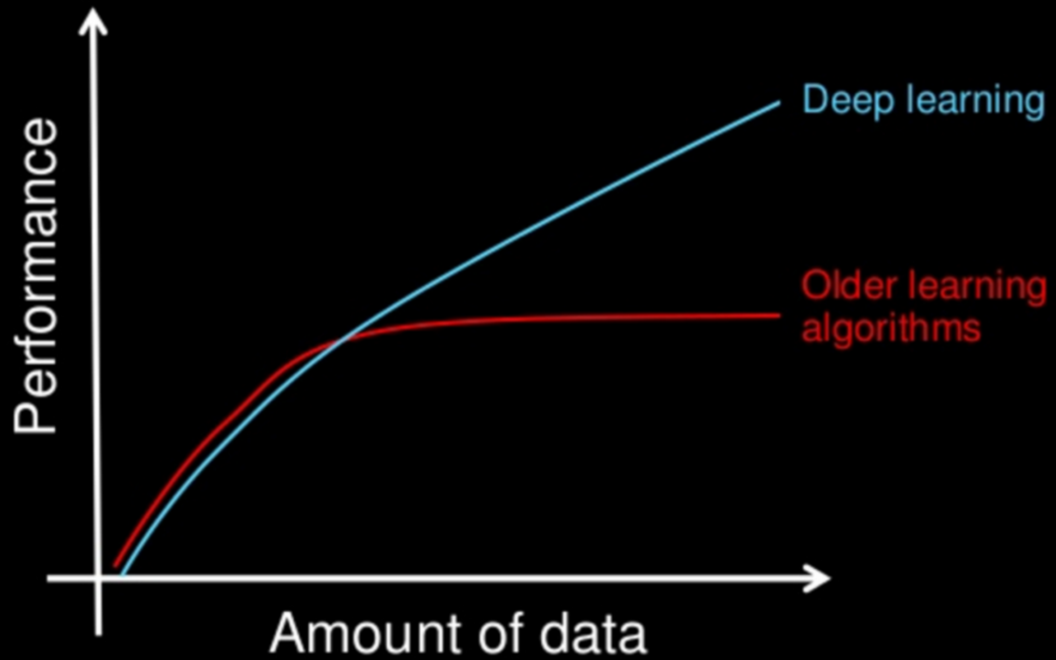
ML Overview

AI vs Machine Learning vs Deep Learning



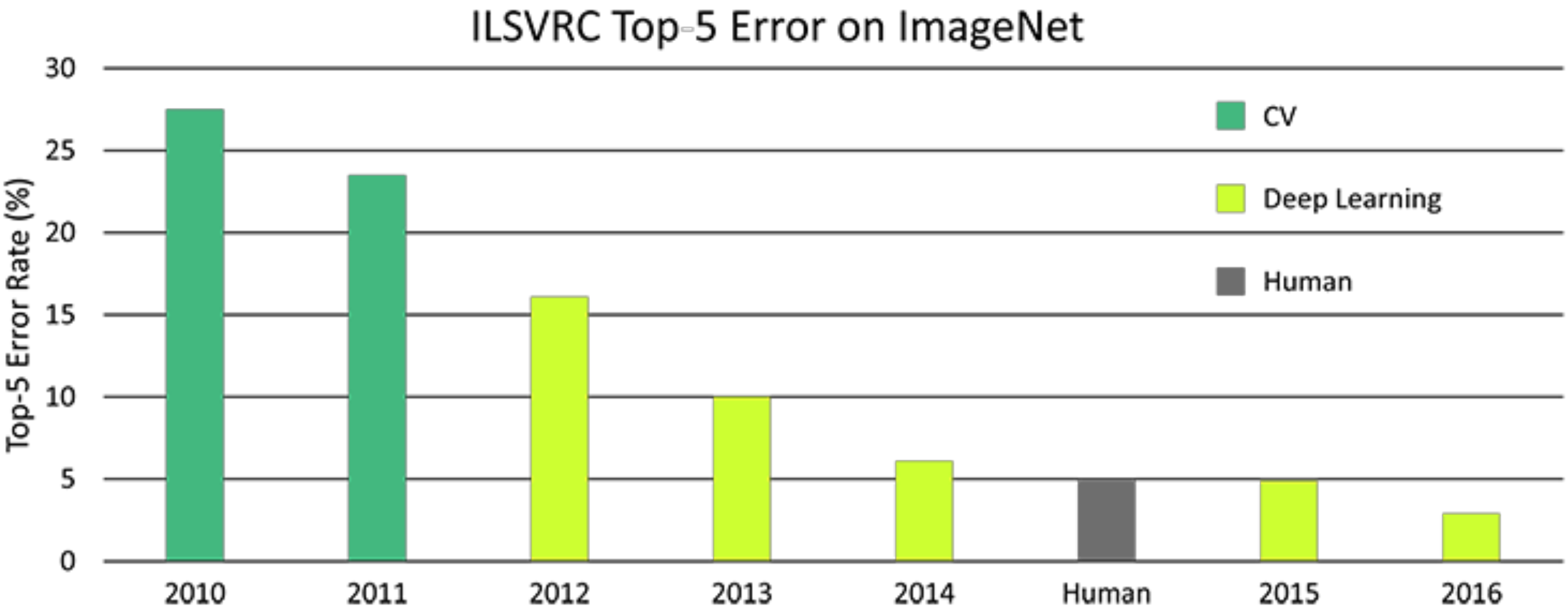

Deep Learning
Machine Learning

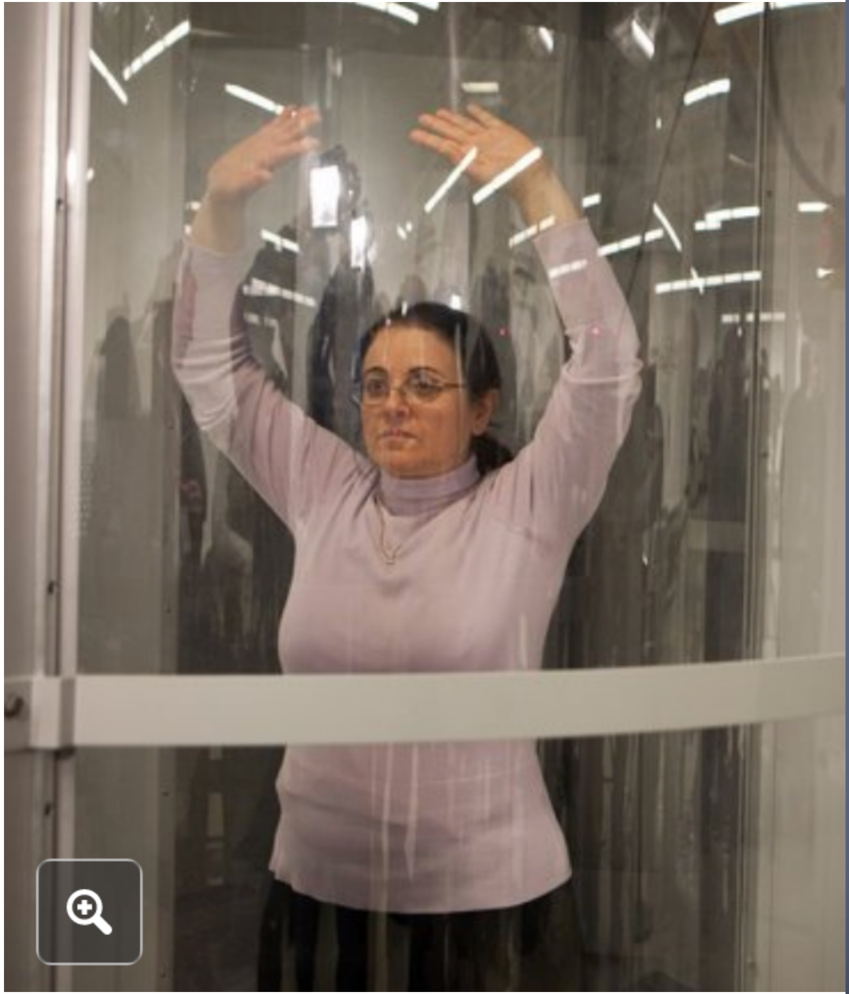
Why deep learning

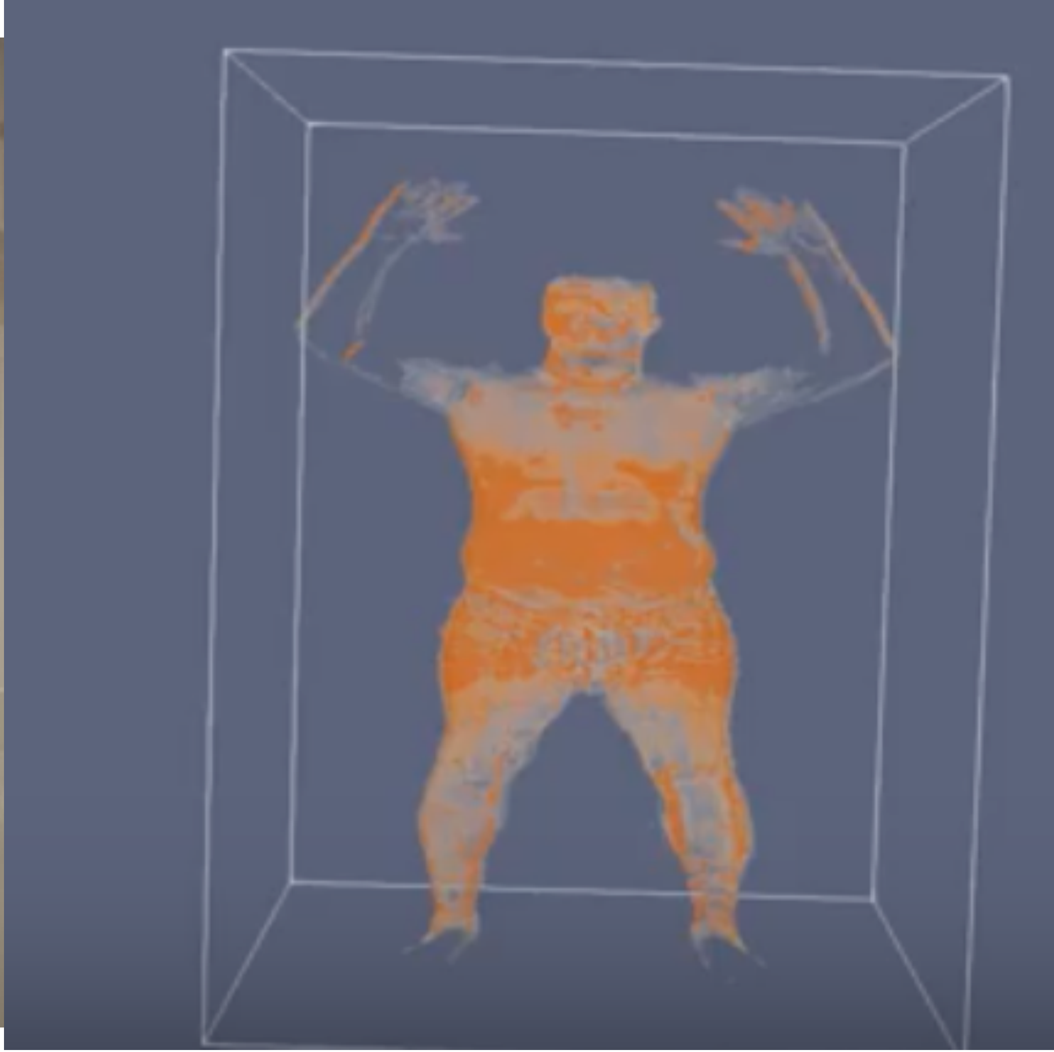


How do data science techniques scale with amount of data?

Image Recognition








entrupy
TAKE A PHOTO OF BAG
Image Captured


@teenybiscuit

Machine Learning 101
Problem

Input

??

Deep Learning

Label
Cat

Output

Statistics 101 Problem

Training Data

Age (Days)	Weight (Grams)
2	49
12	122
8	74
21	205
4	80

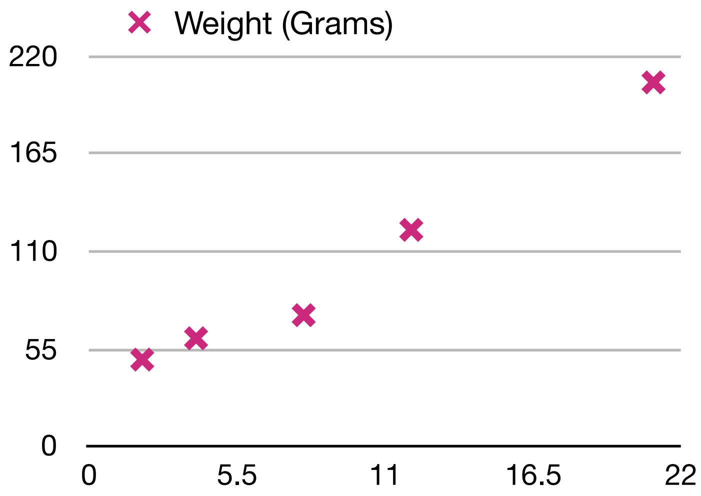
Test Data

Age (Days)	Weight (Grams)
18	??

Linear Regression

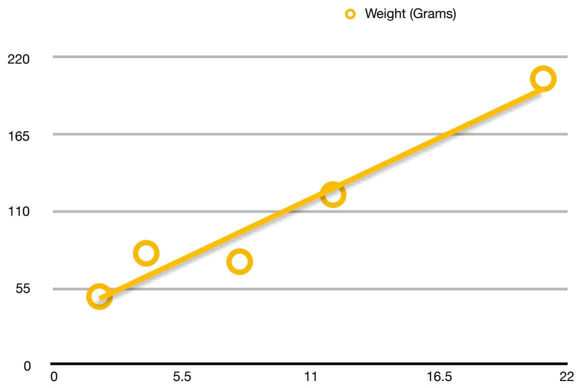
Training Data

Age (Days)	Weight (Grams)
2	49
12	122
8	74
21	205
4	61



Linear Regression

Linear Model

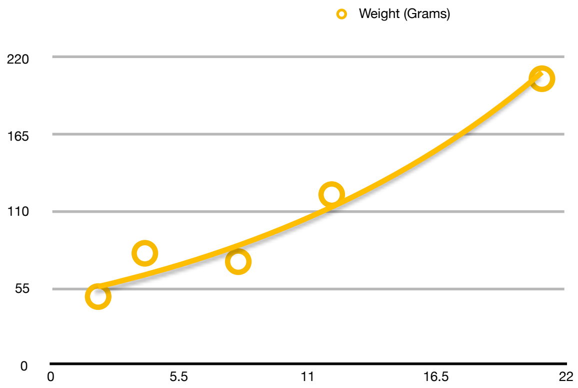


Test Data

Age (Days)	Weight (Grams)
18	**170**

Non-Linear Regression

Non-Linear Model

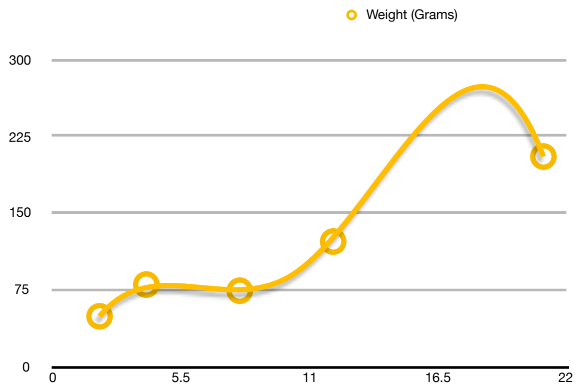


Test Data

Age (Days)	Weight (Grams)
18	**166**

Non Linear Regression

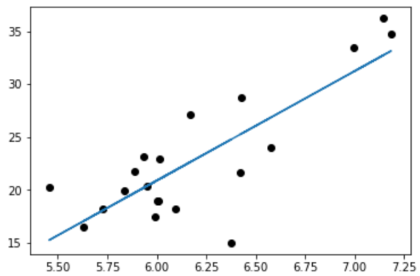
Non Linear Model

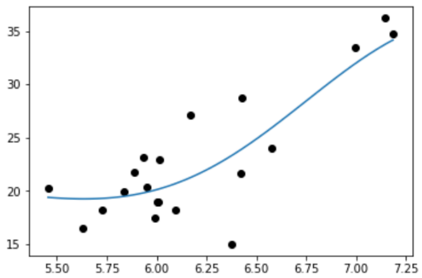


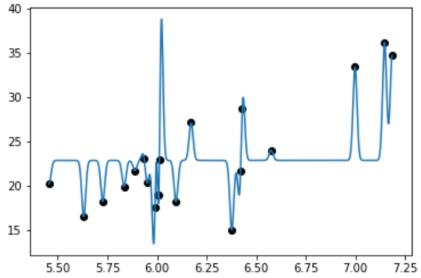
Test Data

Age (Days)	Weight (Grams)
18	**270**

Overfitting

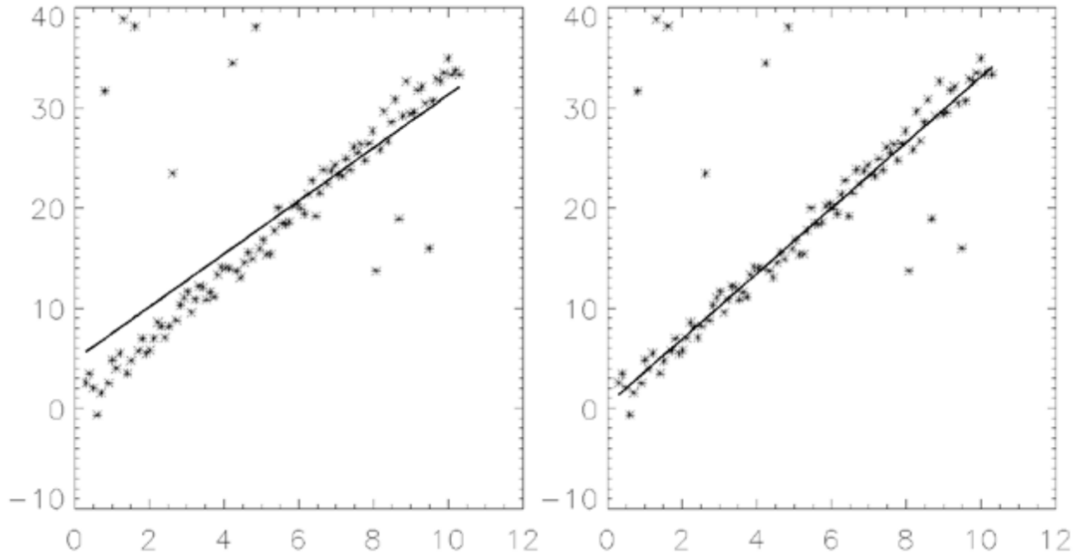






Non Linear Regression

What Even Makes a Model Good?



Multivariate Linear Regression

Training Data

Age (Days)	Diet	Weight (Grams)
2	0	49
12	0	122
8	0	74
21	0	205
4	1	80

Test Data

Age (Days)	Diet	Weight (Grams)
18	0	**??**

Deep Learning 101 Problem

Input

??

Deep Learning

Label
Cat

Output

Statistics API

Input: Fixed Length List of Numbers

Output: Fixed Length List of Numbers

Model generated from list of inputs (training data)

Machine Learning/Deep Learning API

Input: Fixed Length List of Numbers

Output: Fixed Length List of Numbers

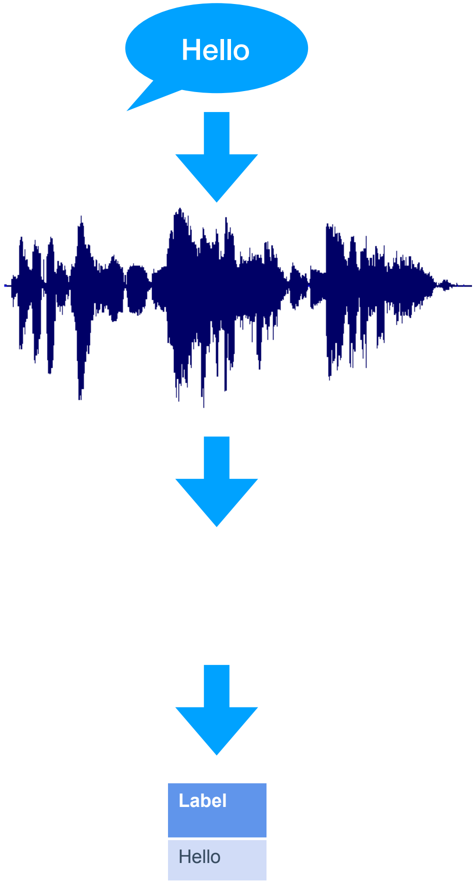
Model generated from list of inputs (training data)

Machine Learning on Speech

Input

Output

Machine Learning on Speech



Input

Feature Extraction

Machine Learning

Output

Machine Learning on Text

I love my iPhone! **Input**

??

Label
Positive Emotion

Output

Machine Learning on Text

I love my iPhone! **Input**

Feature Extraction

A	aardvark	…	hate	I	iphone	love	my	…	Zyzzyva
0	0	…	0	1	1	1	1	…	0

Machine Learning

Positive Emotion	Negative Emotion
1.0	0.0

Output

Deep Learning 101 Problem

Input

??

Deep Learning

Label
Cat

Output

Input

Feature Extraction

(0,0)	(0,1)	(0,2)	....	(1,0)	(1,1)	(1,2)	....	(2,0)	(2,1)	(2,2)
23	15	3		56	23	12		56	23	12

Machine Learning

Cat Score	Dog Score	Fish Score	Other Score
1.0	0.0	0.0	0.0

Output

Our Tech Stack

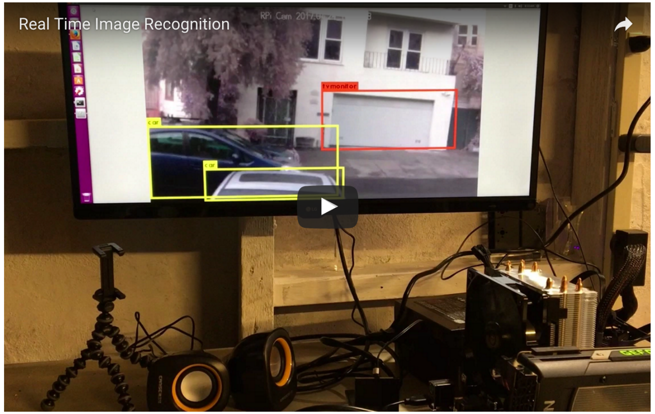

Keras
TensorFlow
CudNN
CUDA
GPU

Build your own box

Build a super fast deep learning machine for under $1,000

The adventures in deep learning and cheap hardware continue!

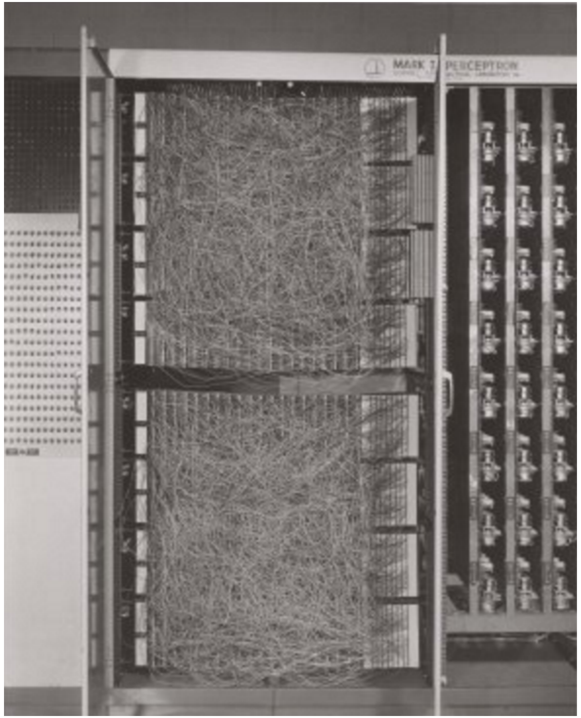
By Lukas Biewald. February 1, 2017



https://www.oreilly.com/learning/build-a-super-fast-deep-learning-machine-for-under-1000

Perceptrons

The First Perceptron



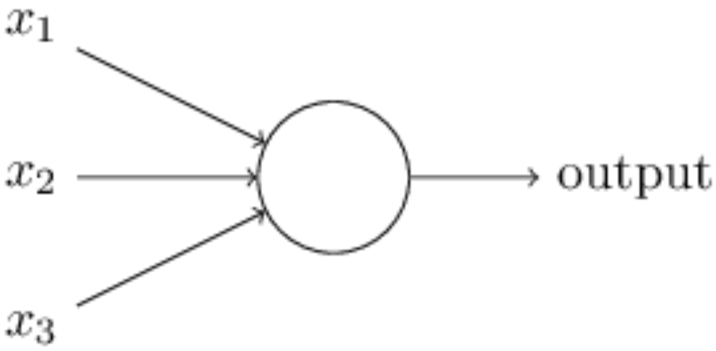
tion it has perceived itself. Ordinary computers remember only what is fed into them on punch cards or magnetic tape.

Later Perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech or writing in another language, it was predicted.

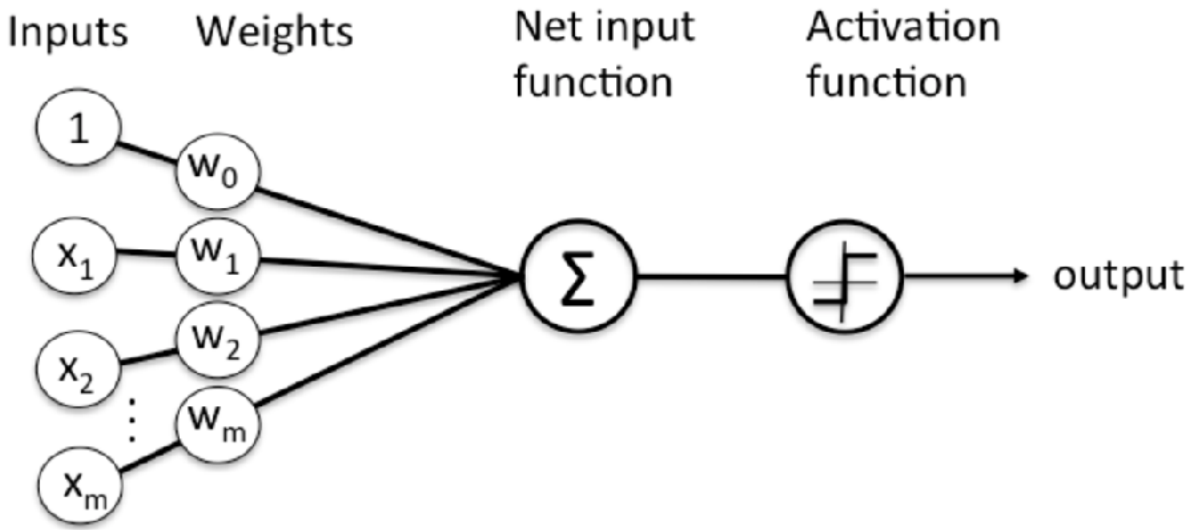
Mr. Rosenblatt said in principle it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

In today's demonstration, the "704" was fed two cards, one with squares marked on the left side and the other with squares on the right side.

Perceptron



Perceptron



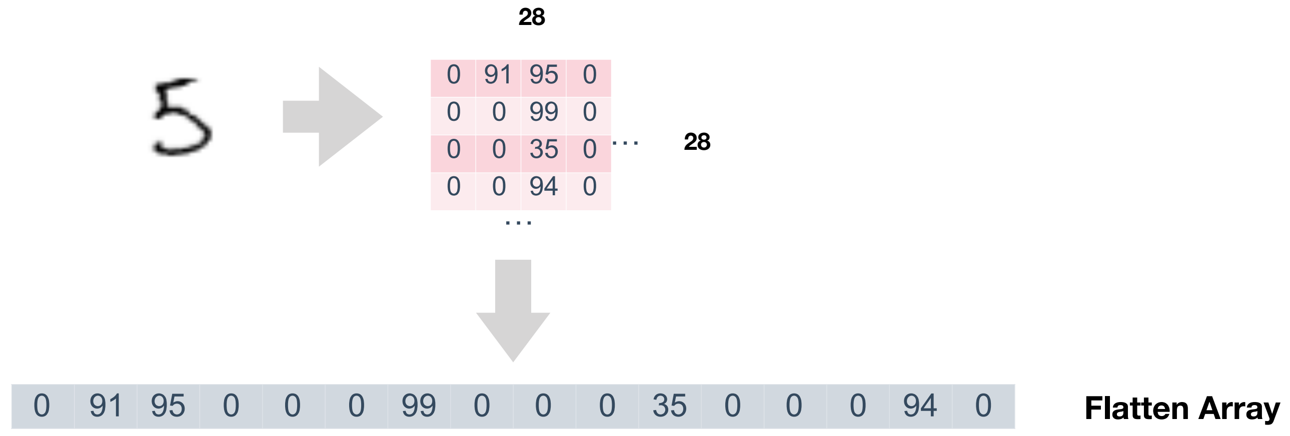
Schematic of Rosenblatt's perceptron.

Perceptron Algorithm

28

0	91	95	0	
0	0	99	0	
0	0	35	0	… 28
0	0	94	0	
	…			

Perceptron Algorithm



Perceptron Algorithm

28

0	91	95	0
0	0	99	0
0	0	35	0
0	0	94	0

... 28

...

0	91	95	0	0	0	99	0	0	0	35	0	0	0	94	0	Flatten Array
0.12	0.45	2.34	-0.34	1.21	-1.24	0.12	-0.45	12.4	0.45	-0.34	2.34	1.21	0.12	4.35	-2.34	* **Weights**

Perceptron Algorithm

28

0	91	95	0	...
0	0	99	0	
0	0	35	0	28
0	0	94	0	
...				

0	91	95	0	0	0	99	0	0	0	35	0	0	0	94	0	Flatten Array
0.12	0.45	2.34	-0.34	1.21	-1.24	0.12	-0.45	12.4	0.45	-0.34	2.34	1.21	0.12	4.35	-2.34	* Weights

Σ 0.3 **Output**

Perceptron Algorithm

28

0	91	95	0	
0	0	99	0	
0	0	35	0	... 28
0	0	94	0	
...				

0	91	95	0	0	0	99	0	0	0	35	0	0	0	94	0	Flatten Array
0.12	0.45	2.34	-0.34	1.21	-1.24	0.12	-0.45	12.4	0.45	-0.34	2.34	1.21	0.12	4.35	-2.34	* Weights

Σ	0.3	Output
	1.0	Activation
	= Is Five	**Interpret**

Multiclass Perceptron Algorithm

28

0	91	95	0
0	0	99	0
0	0	35	0
0	0	94	0

… 28

…

0	91	95	0	0	0	99	0	0	0	35	0	0	0	94	0	Flatten Array
0.12	0.45	2.34	-0.34	1.21	-1.24	0.12	-0.45	12.4	0.45	-0.34	2.34	1.21	0.12	4.35	-2.34	
0.45	0.12	-0.45	-0.34	-0.34	-0.34	12.4	-0.34	-1.24	-2.34	-0.34	-0.34	12.4	-0.2	4.35	1.3	* Weights
2.34	-0.34	-0.45	0.45	12.4	0.12	-0.45	-2.34	0.45	-1.24	-0.34	12.4	-0.34	4.35	-2.34	-0.6	

Σ	0.3	0.6	0.7	Output
	0.3	0.6	0.7	Activation Function

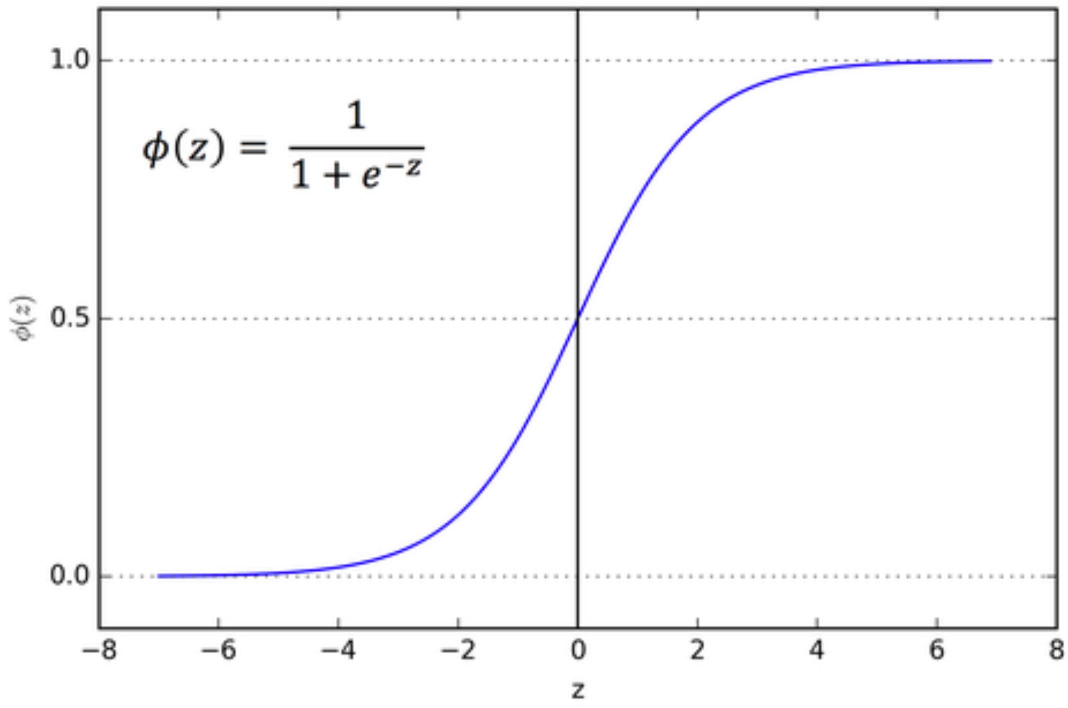
One hot encoding

Label
0
4
4
3
0
9

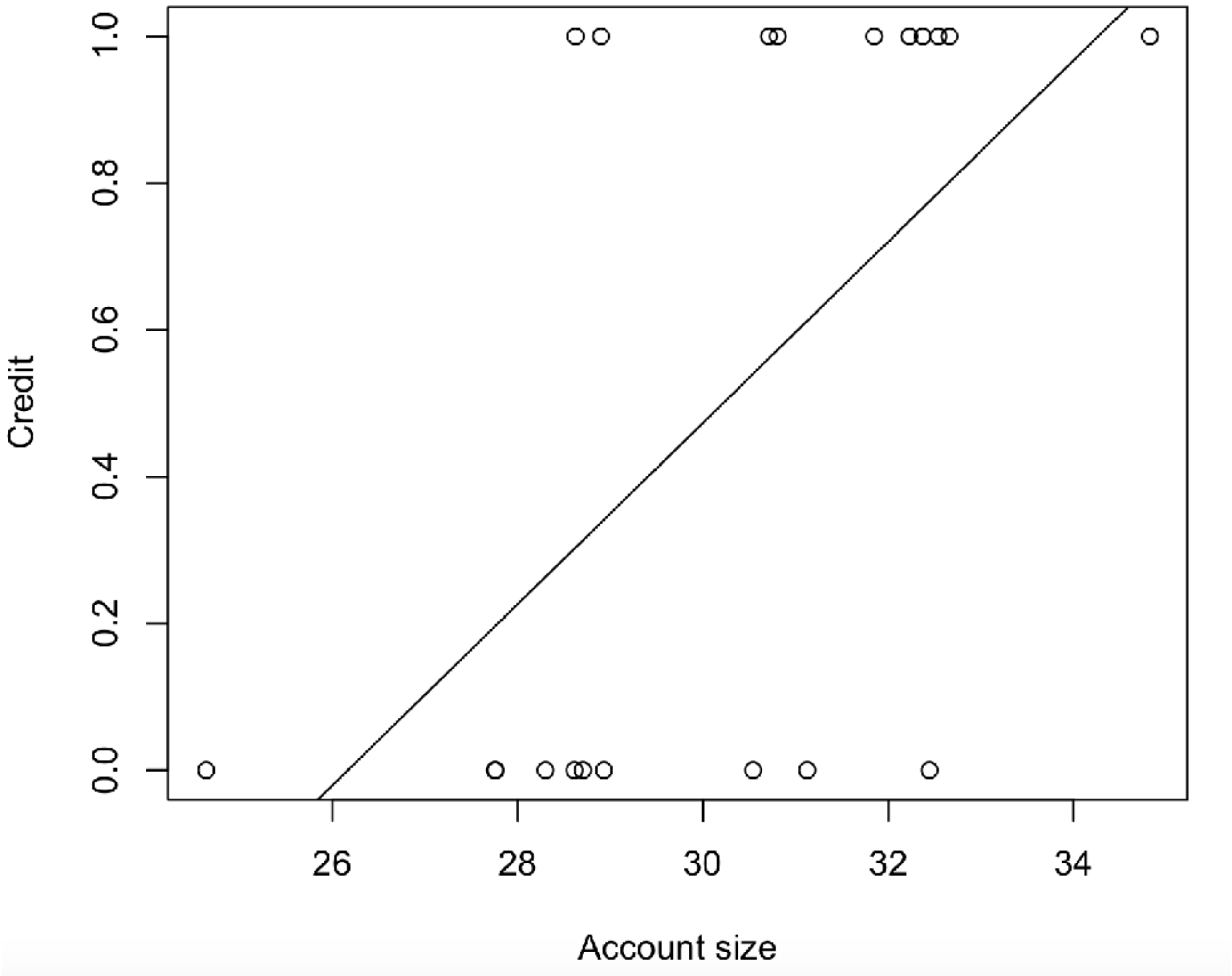
0	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1

Activation Functions

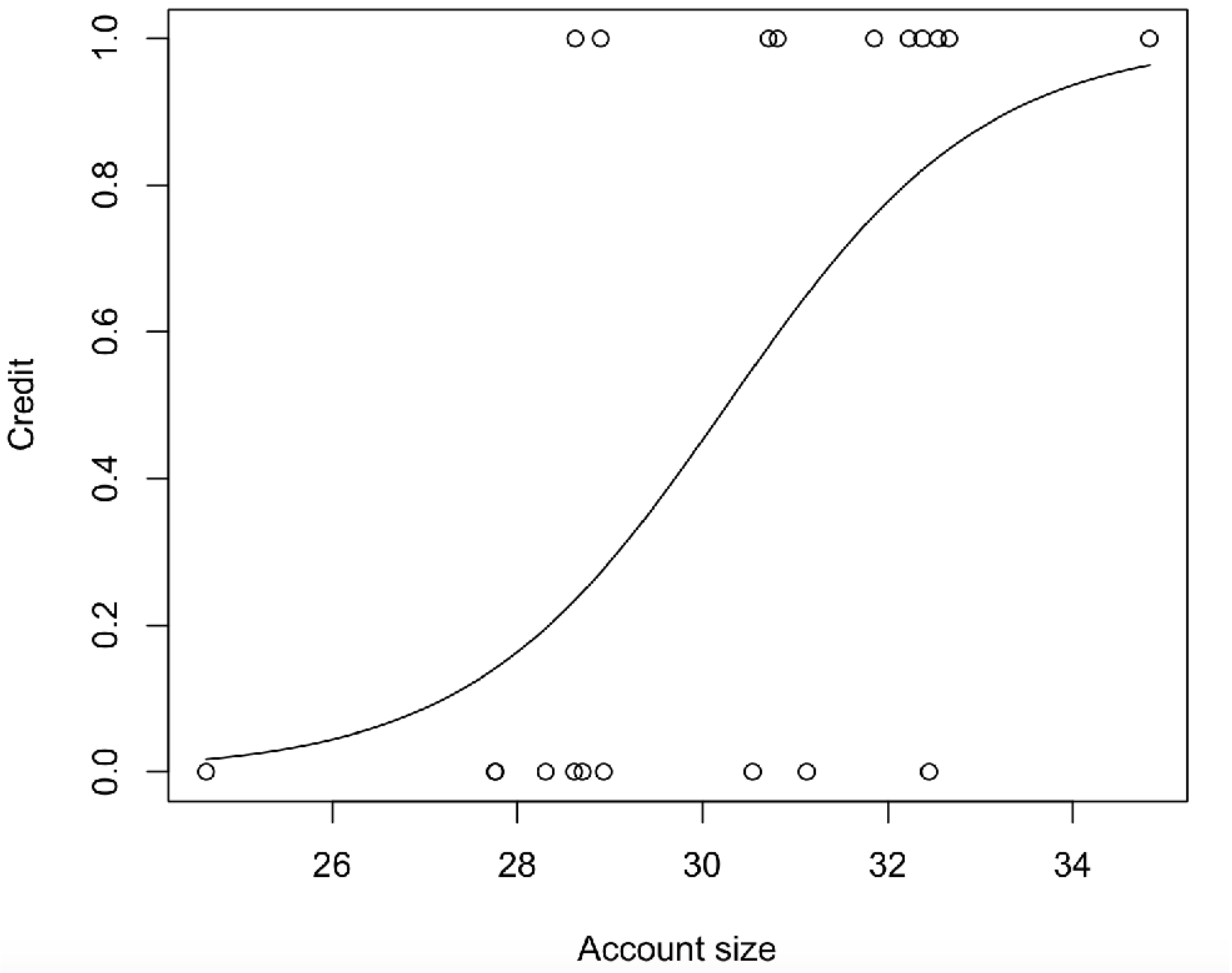
Sigmoid Activation Function



special case of softmax and logistic



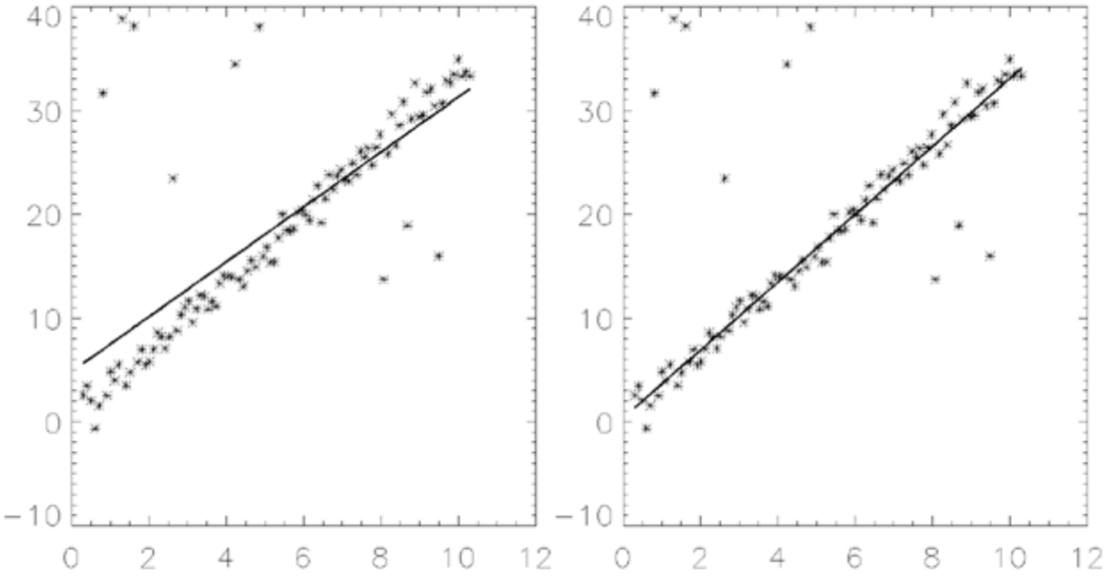
1.0
0.8
0.6
0.4
0.2
0.0
Credit
26
28
30
32
34
Account size



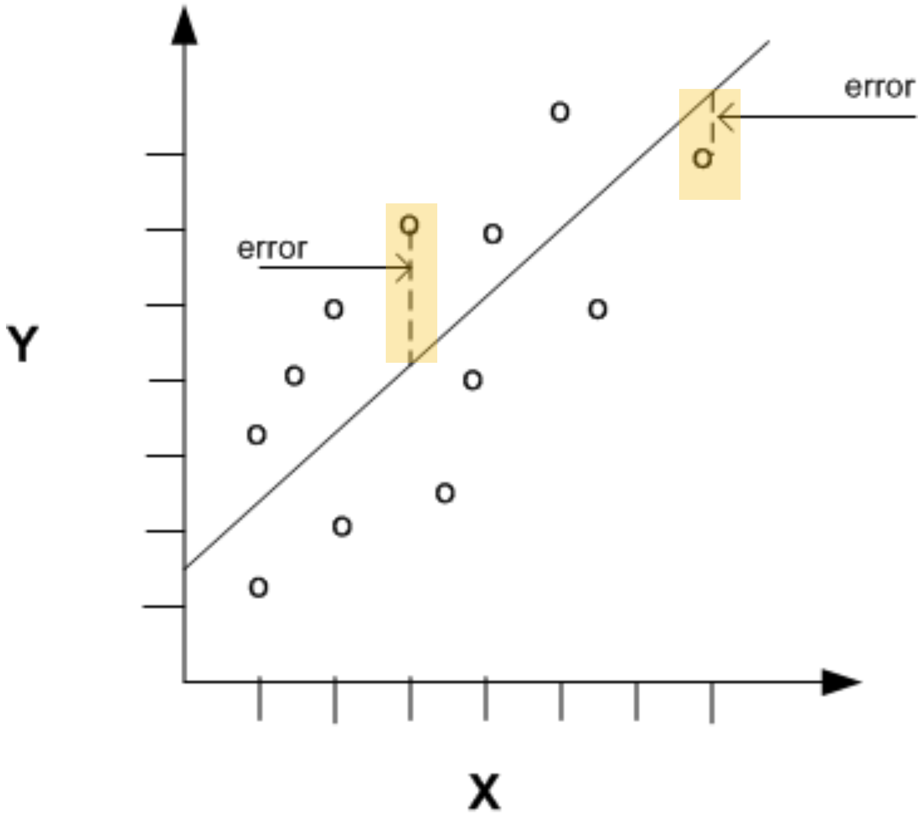
Credit
1.0
0.8
0.6
0.4
0.2
0.0
26
28
30
32
34
Account size

Loss Functions

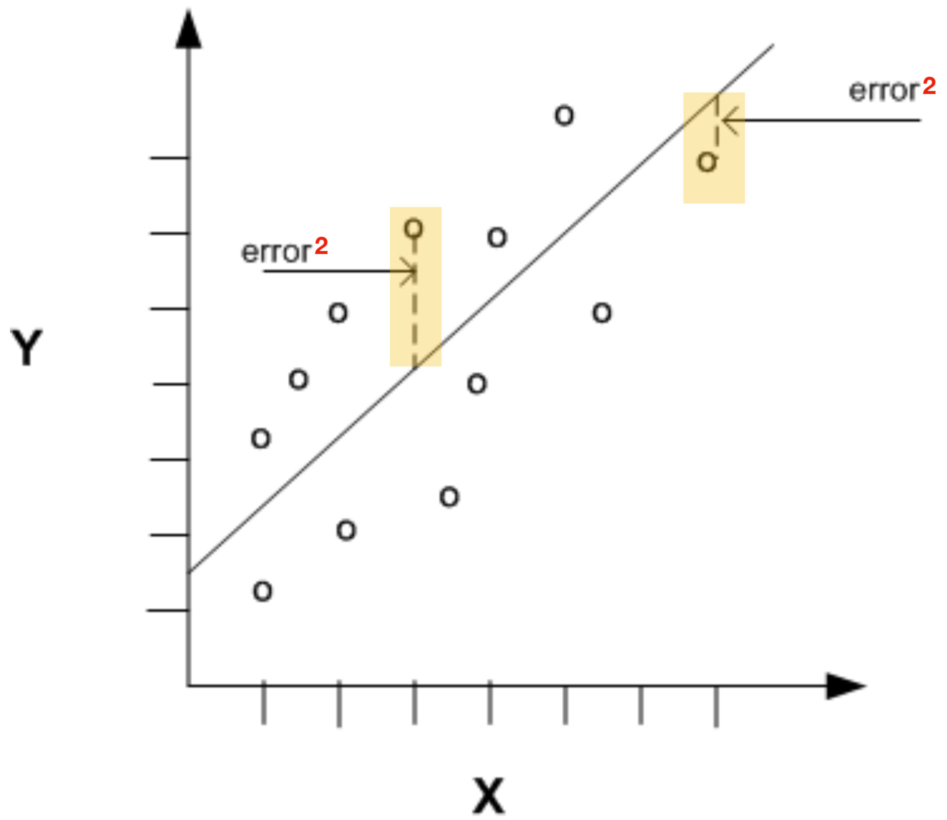
Poll: Which side is mean squared error and which was mean absolute error?



Mean Absolute Error

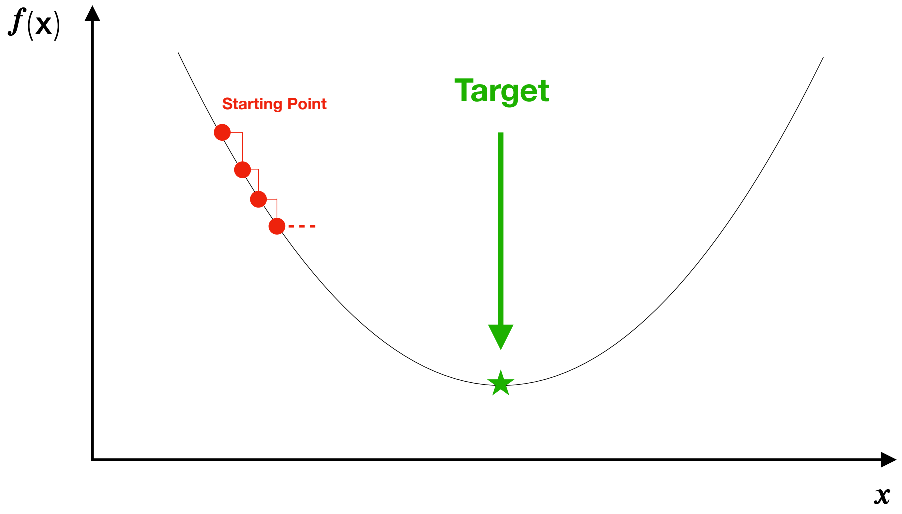


Mean Squared Error (MSE)

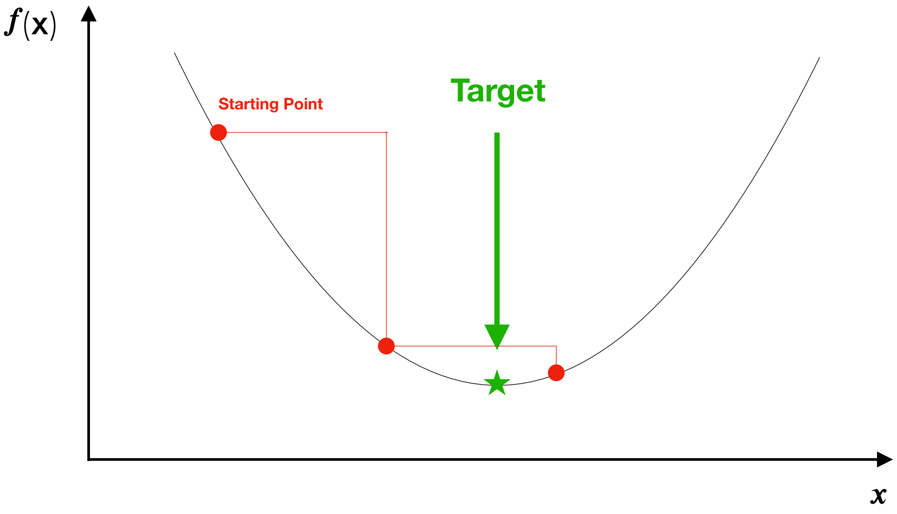


Learning Rate

Learning Rate too slow



Learning Rate too fast



examples/keras-perceptron/nn.py

A Recipe for Training Neural Networks

Apr 25, 2019

Some few weeks ago I posted a tweet on “the most common neural net mistakes”, listing a few common gotchas related to training neural nets. The tweet got quite a bit more engagement than I anticipated (including a webinar :)). Clearly, a lot of people have personally encountered the large gap between “here is how a convolutional layer works” and “our convnet achieves state of the art results”.

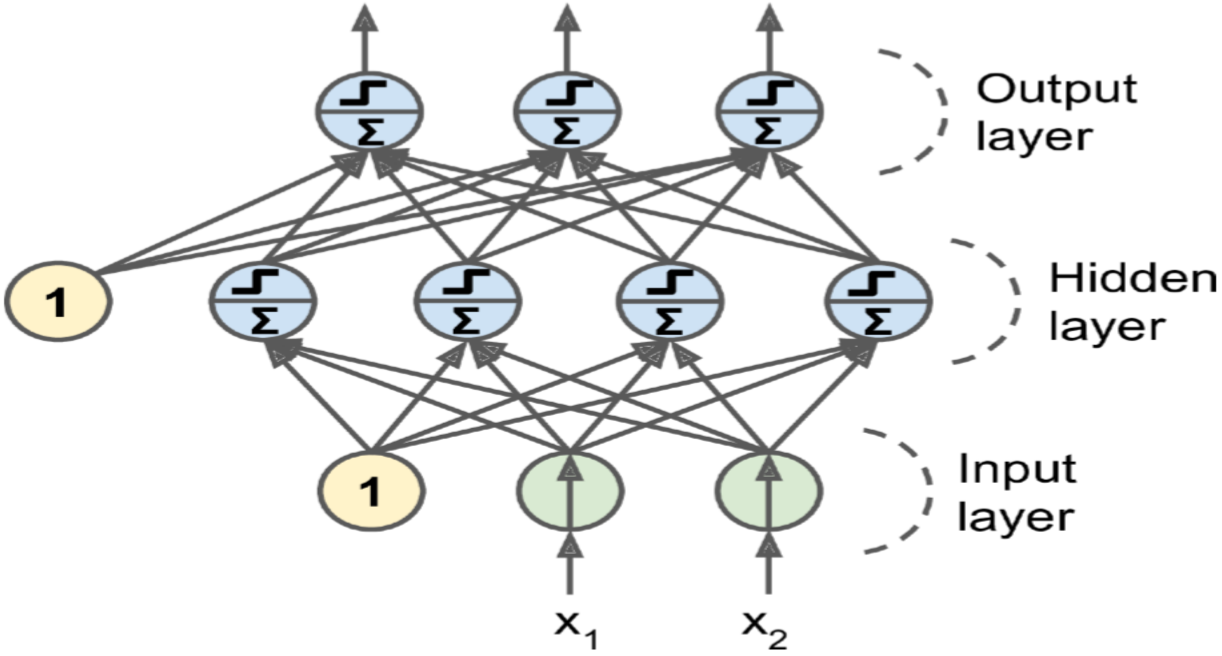
So I thought it could be fun to brush off my dusty blog to expand my tweet to the long form that this topic deserves. However, instead of going into an enumeration of more common errors or fleshing them out, I wanted to dig a bit deeper and talk about how one can avoid making these errors altogether (or fix them very fast). The trick to doing so is to follow a certain process, which as far as I can tell is not very often documented. Let’s start with two important observations that motivate it.

Simple steps for debugging any neural network

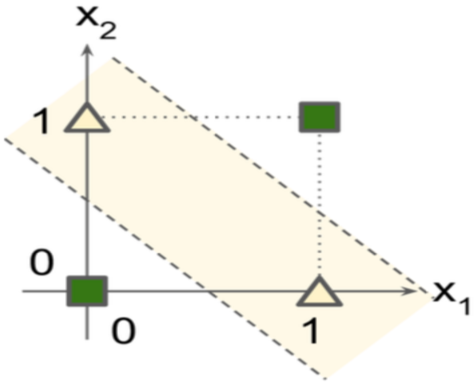
- Make your network as simple as possible
- Check if you can fit a small sample of your training data
- Look at the actual values the network is outputting
- Add complexity one step at a time

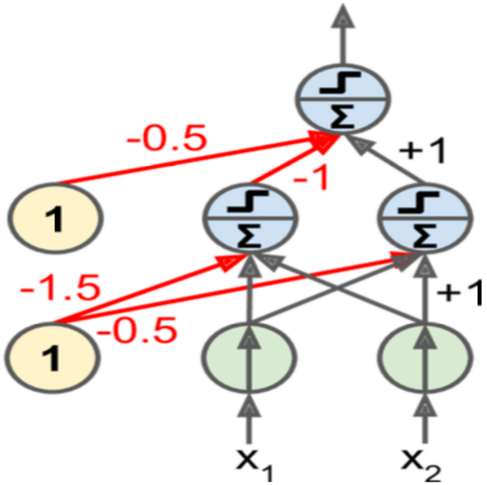
Multi-Layer Perceptrons and CNNs

Two Layers of Perceptrons

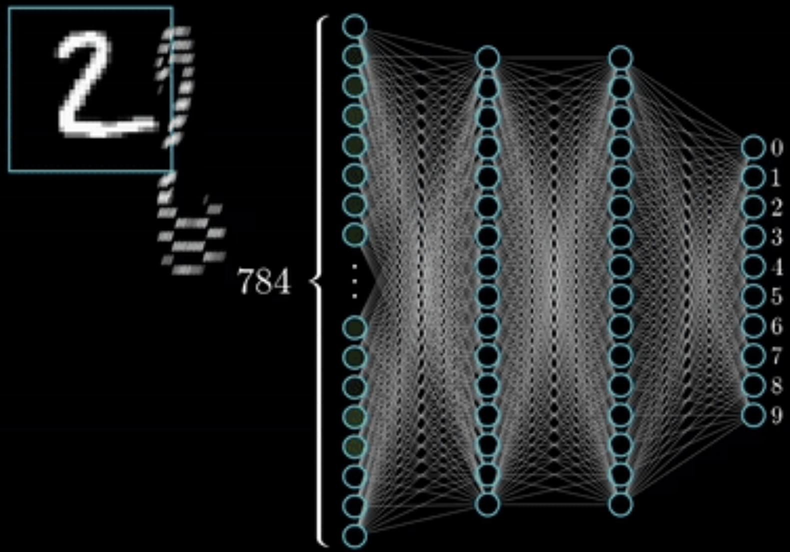


Multi-Payer Perceptrons Solving the “XOR” Problem



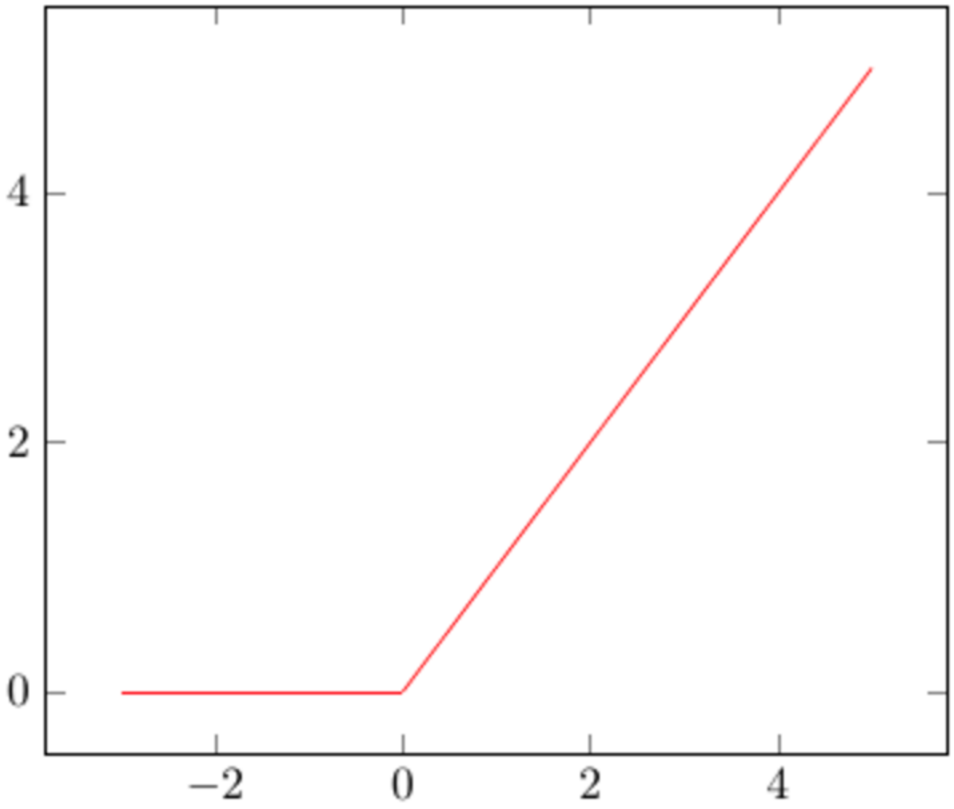


3Blue1Brown: video link

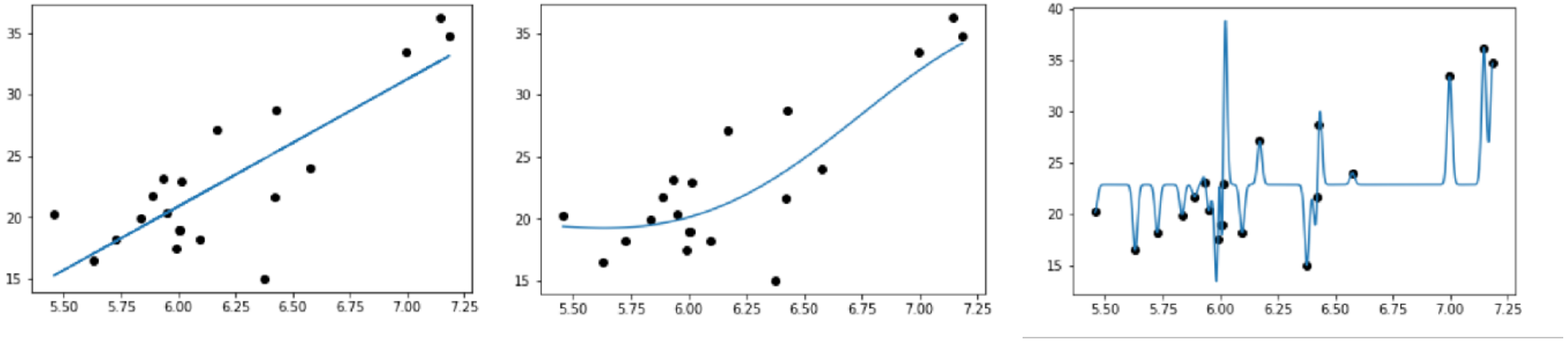


https://www.youtube.com/watch?v=Ilg3gGewQ5U&vl=en

MLP Activation Function: ReLU



Overfitting



Simple steps for improving any neural network

Ask “Am I overfitting”?

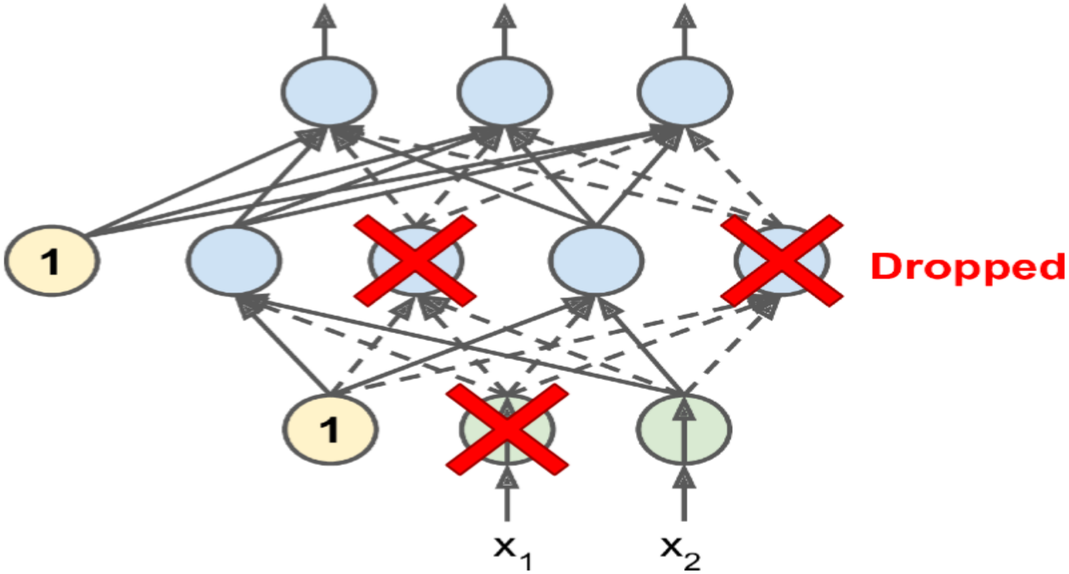
If overfitting, fix the overfitting

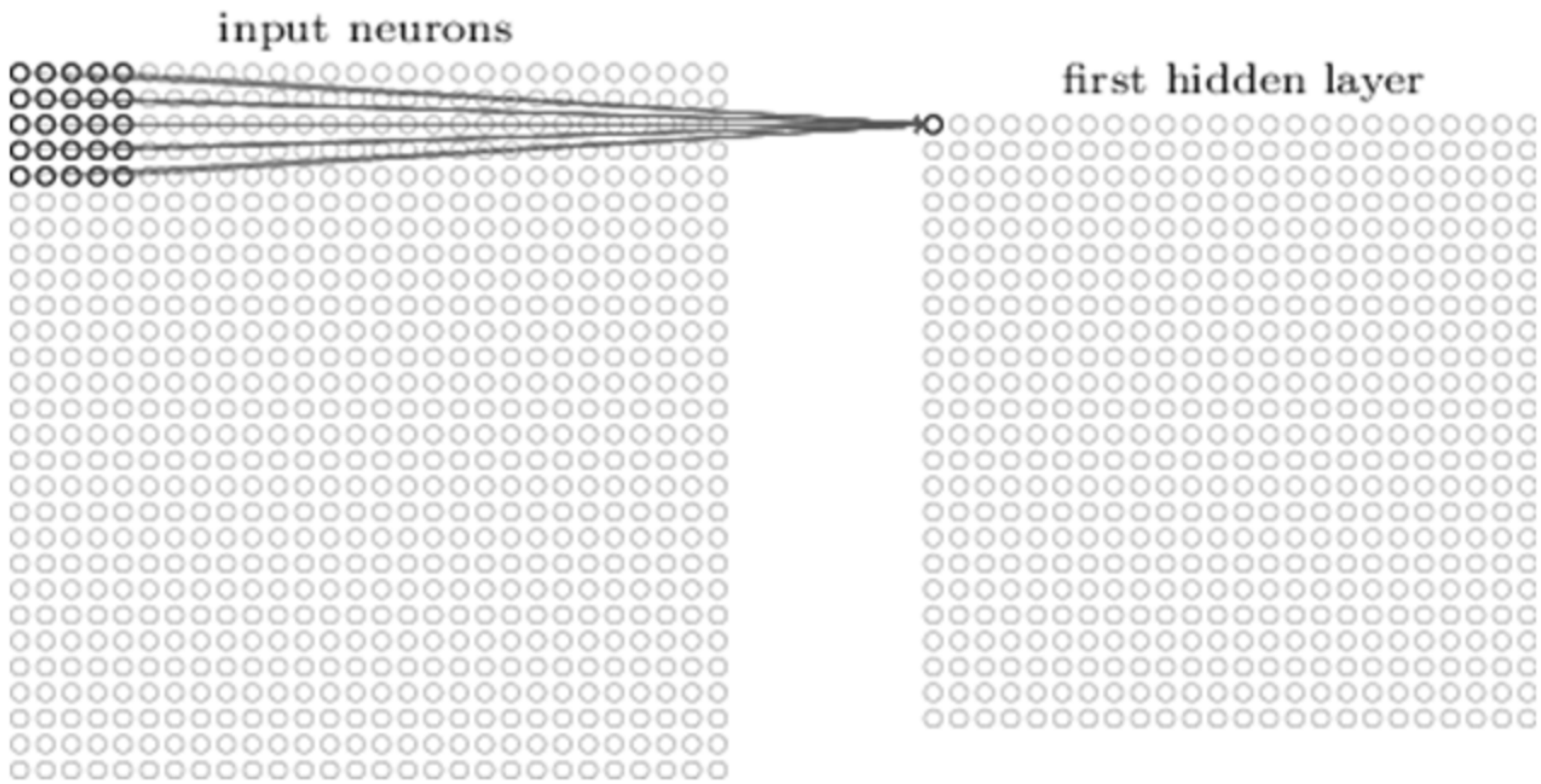
- more training data, dropout, regularization, data augmentation…

If not overfitting, try making the network more powerful

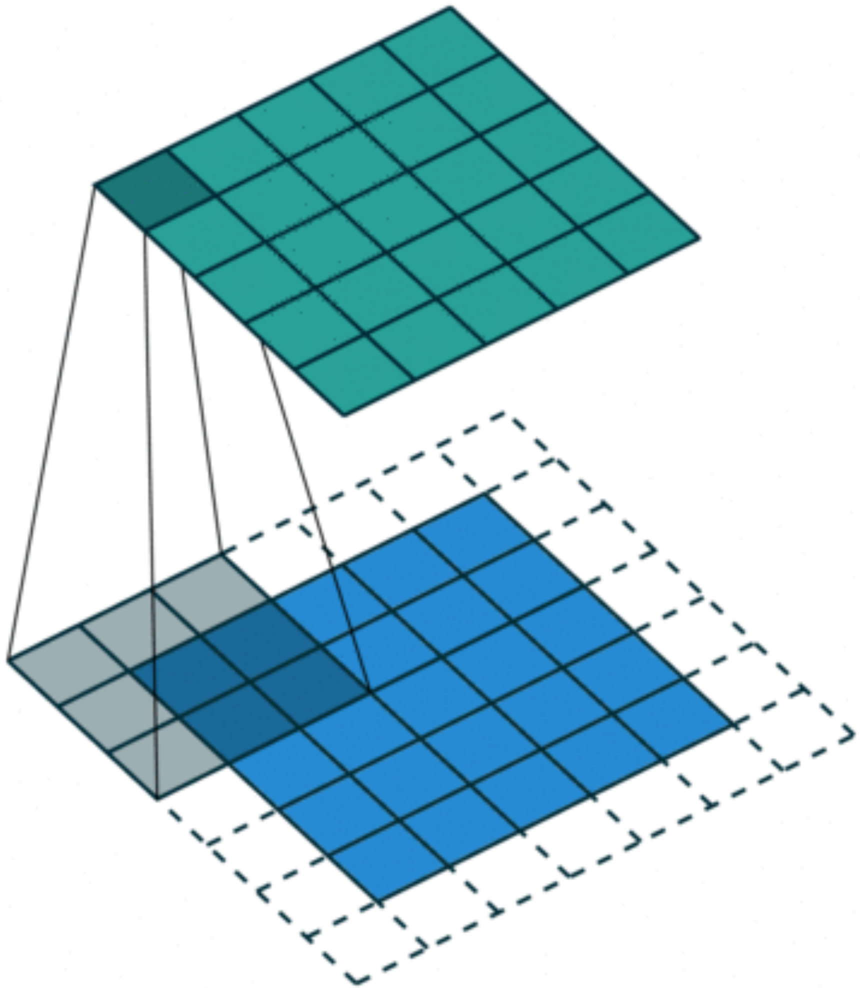
- more layers, larger layers
- one thing at a time!

Dropout

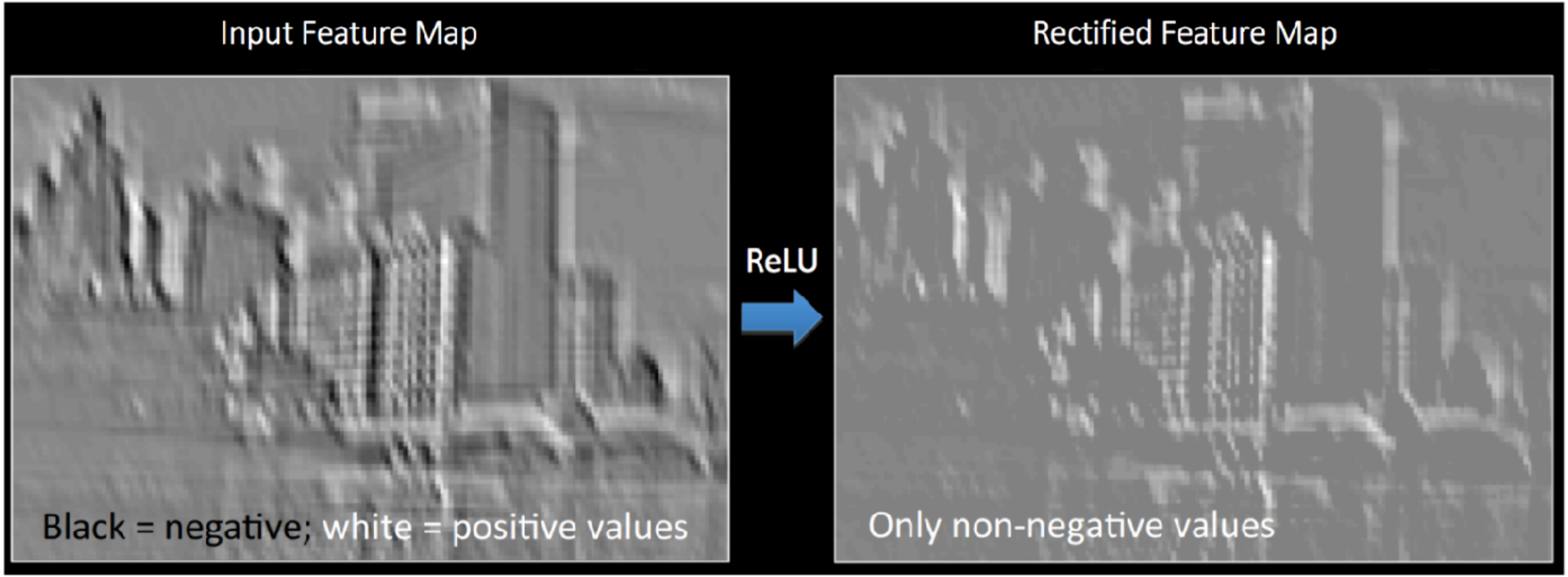




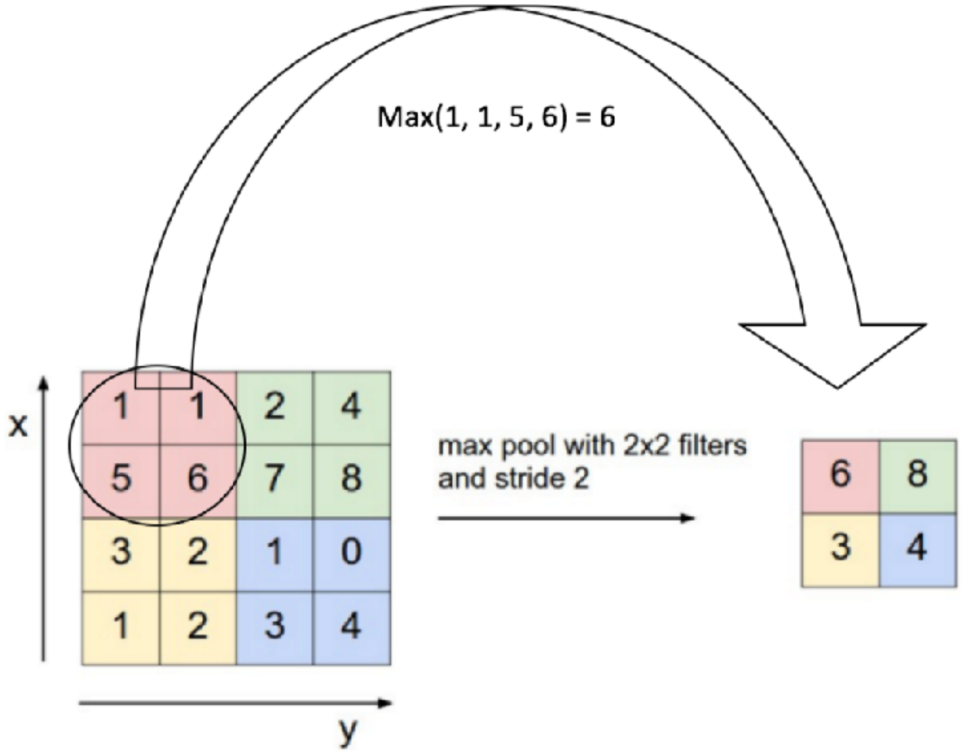
Visualization of 5 x 5 filter convolving around an input volume and producing an activation map



Activation Function (ReLU)

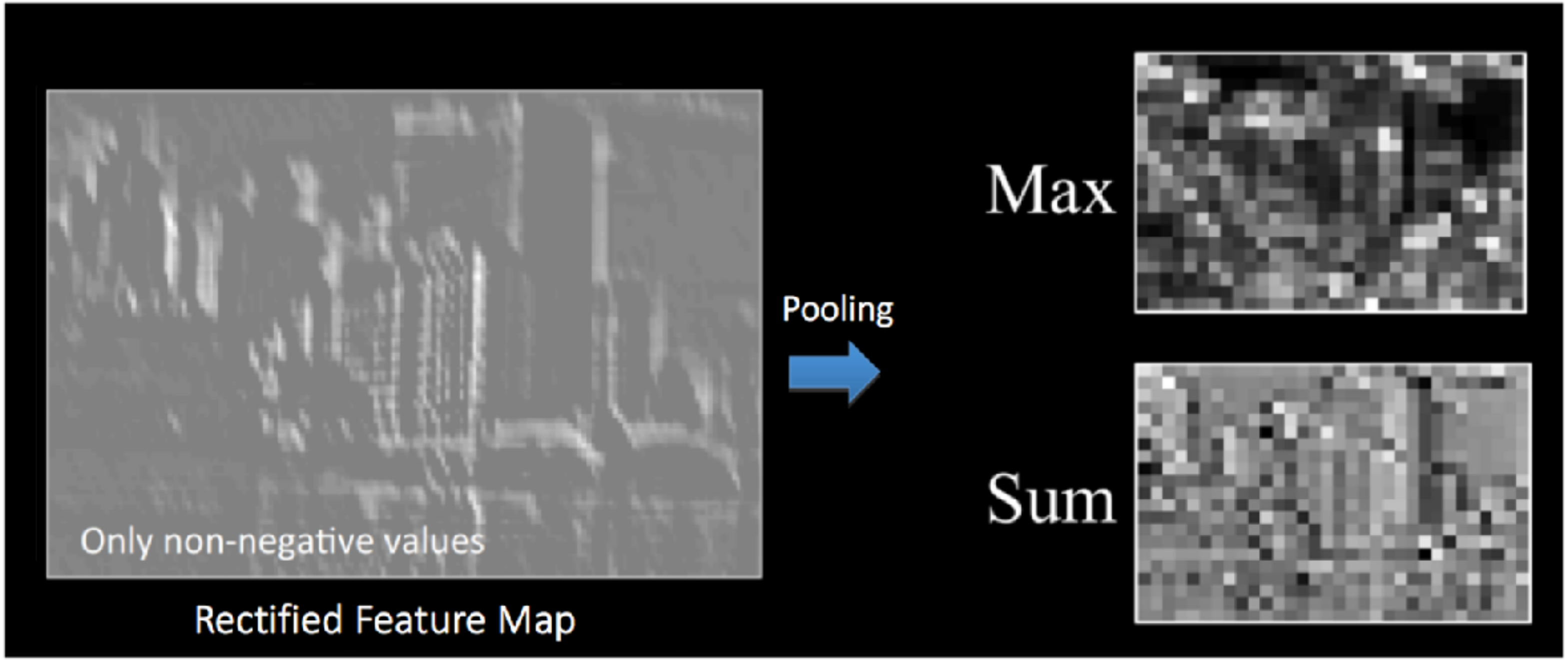


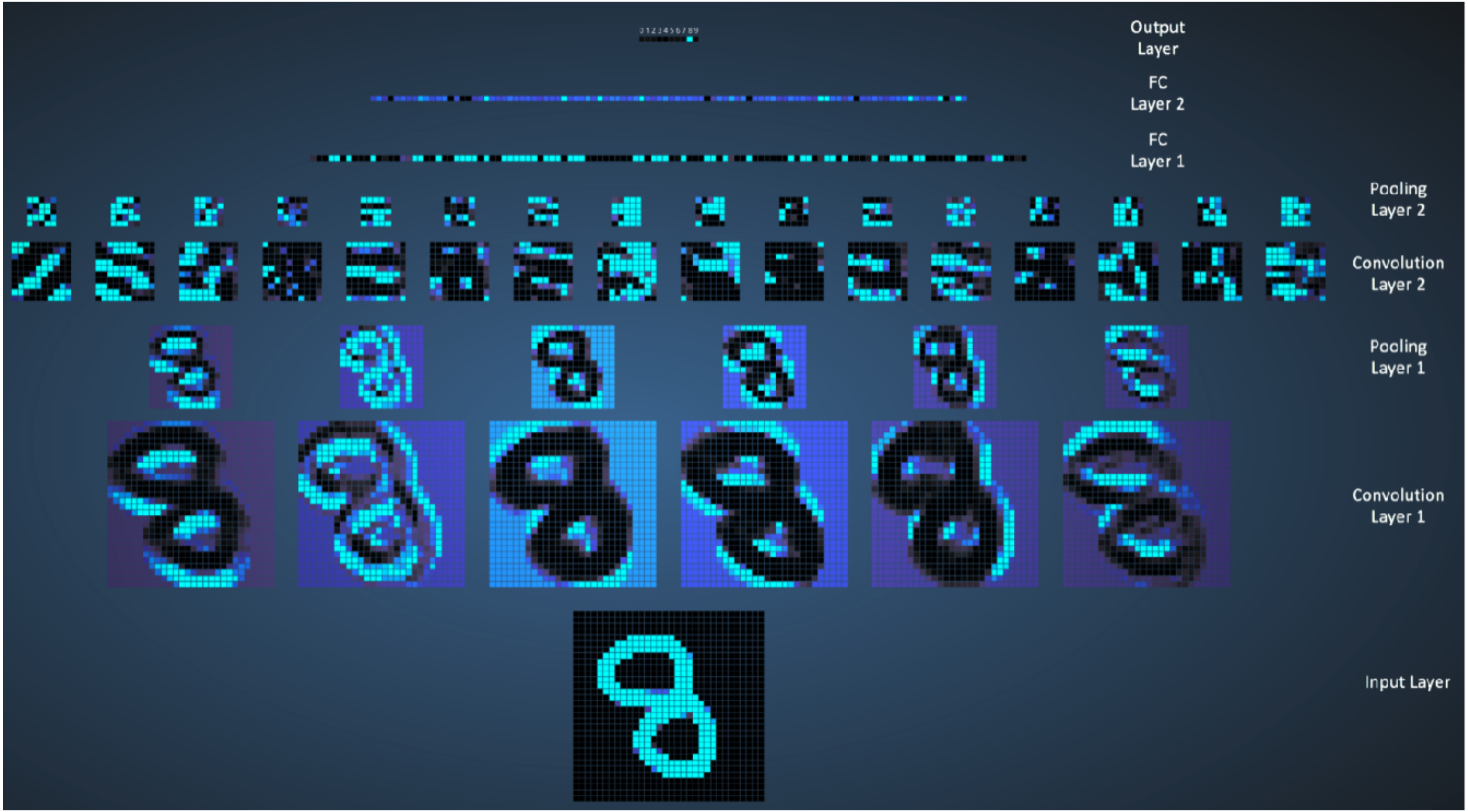
Pooling



Max(1, 1, 5, 6) = 6
x
1
1
2
4
5
6
7
8
3
2
1
0
1
2
3
4
y
max pool with 2x2 filters
and stride 2
6
8
3
4

Pooling


Only non-negative values
Rectified Feature Map
Pooling
Max
Sum


0123456789
Output Layer
FC Layer 2
FC Layer 1
Pooling Layer 2
Convolution Layer 2
Pooling Layer 1
Convolution Layer 1
Input Layer

ml-class/examples/keras-cnn/cnn.py

ml-class/examples/keras-fashion/nn.py