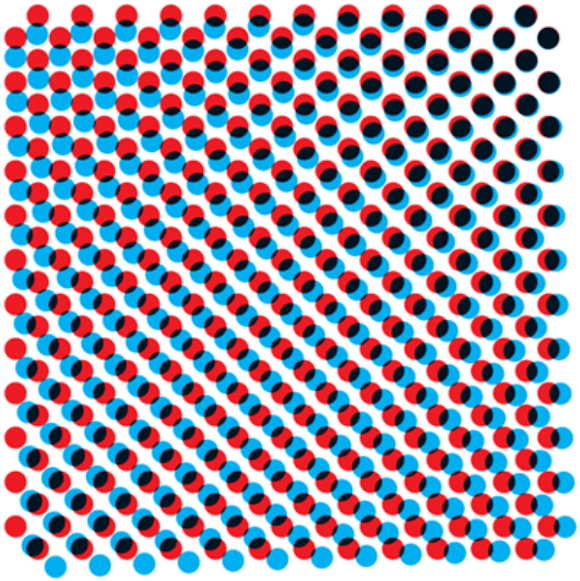


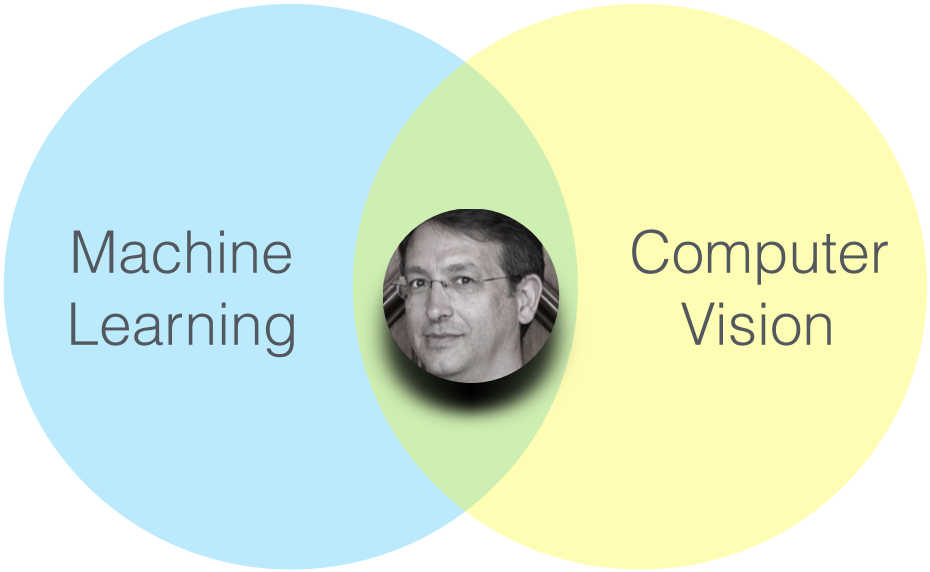
Deep Learning

or how to train large and highly complex models with deeply cascaded nonlinearities by using automatic differentiation and several tricks.

UNIVERSITAT DE BARCELONA

Girona, April 2017

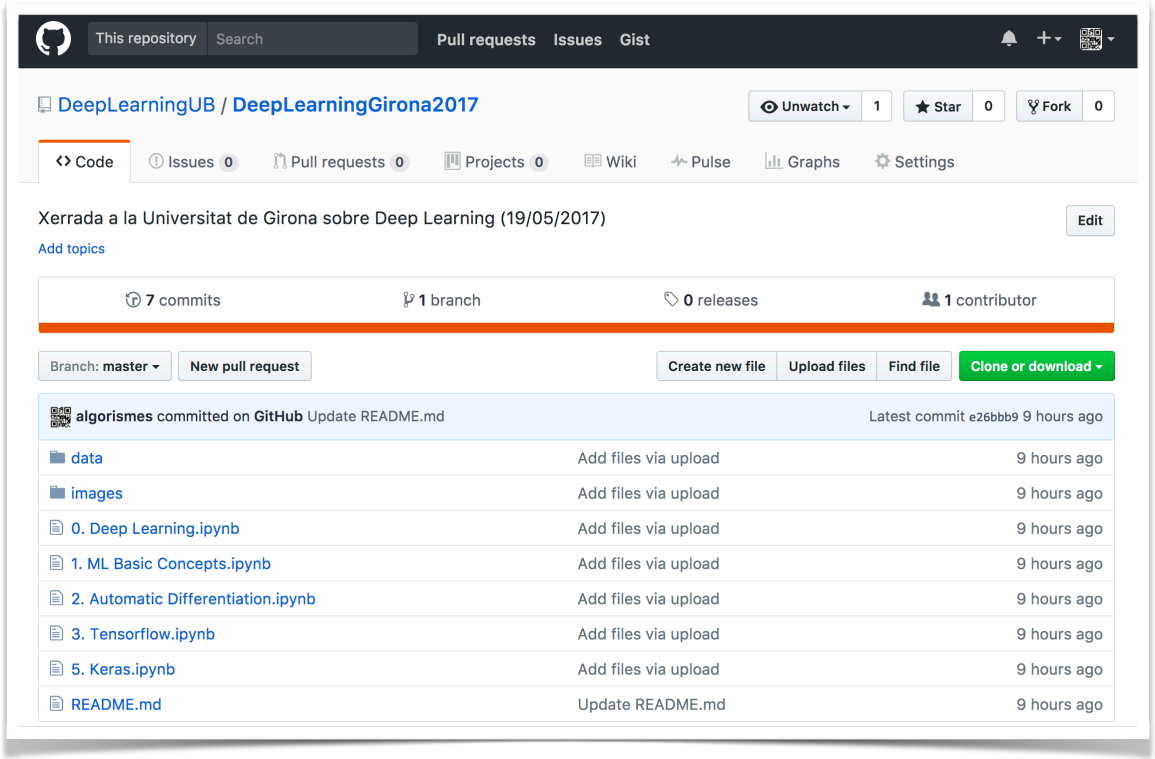
DAS



Jordi Vitrià
jordi.vitria@ub.edu

Since 2007, I am a Full Professor at Mathematics and Compute Science Department, Universitat de Barcelona. Before that I spent 20 years on the faculty of the Computer Science Department at the Universitat Autonoma de de Barcelona working on Computer Vision and Machine Learning. I am the Director of the DataScience@UB group at UB.

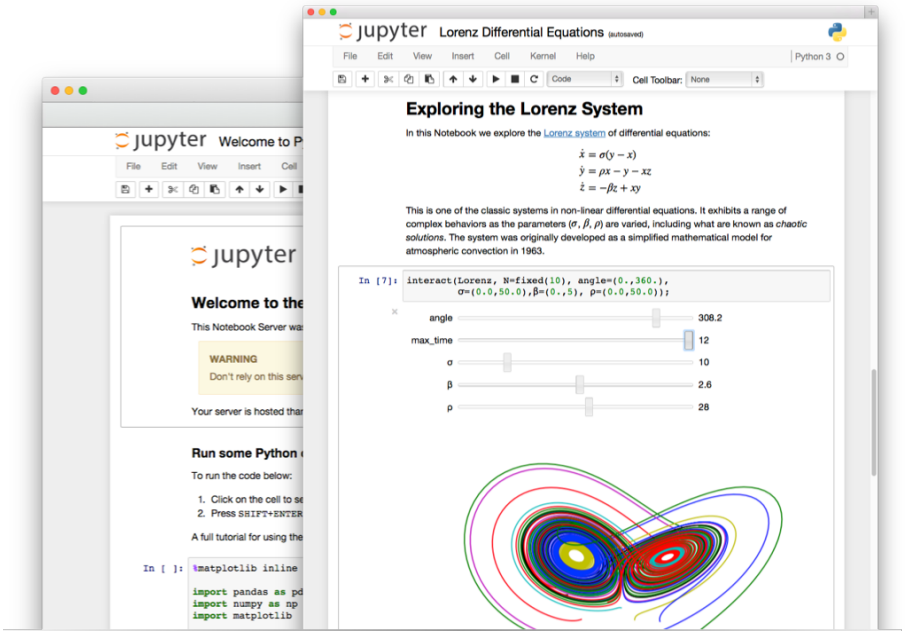
Repository



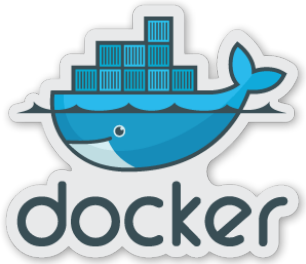
https://github.com/DeepLearningUB/DeepLearningGirona2017

Approach

We will illustrate all contents with Jupyter notebooks, a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text.



Approach



We will use a **Docker Container**.

Docker provides the ability to build a runtime environment that not only remains isolated from other running containers, but also can be deployed to multiple locations in a repeatable way. Docker also uses a text document – a Dockerfile – that contains all the commands to assemble an image, which will meet our need to document the build environment. Finally, Docker's runtime options enable us to attach GPU devices when deploying on remote servers.

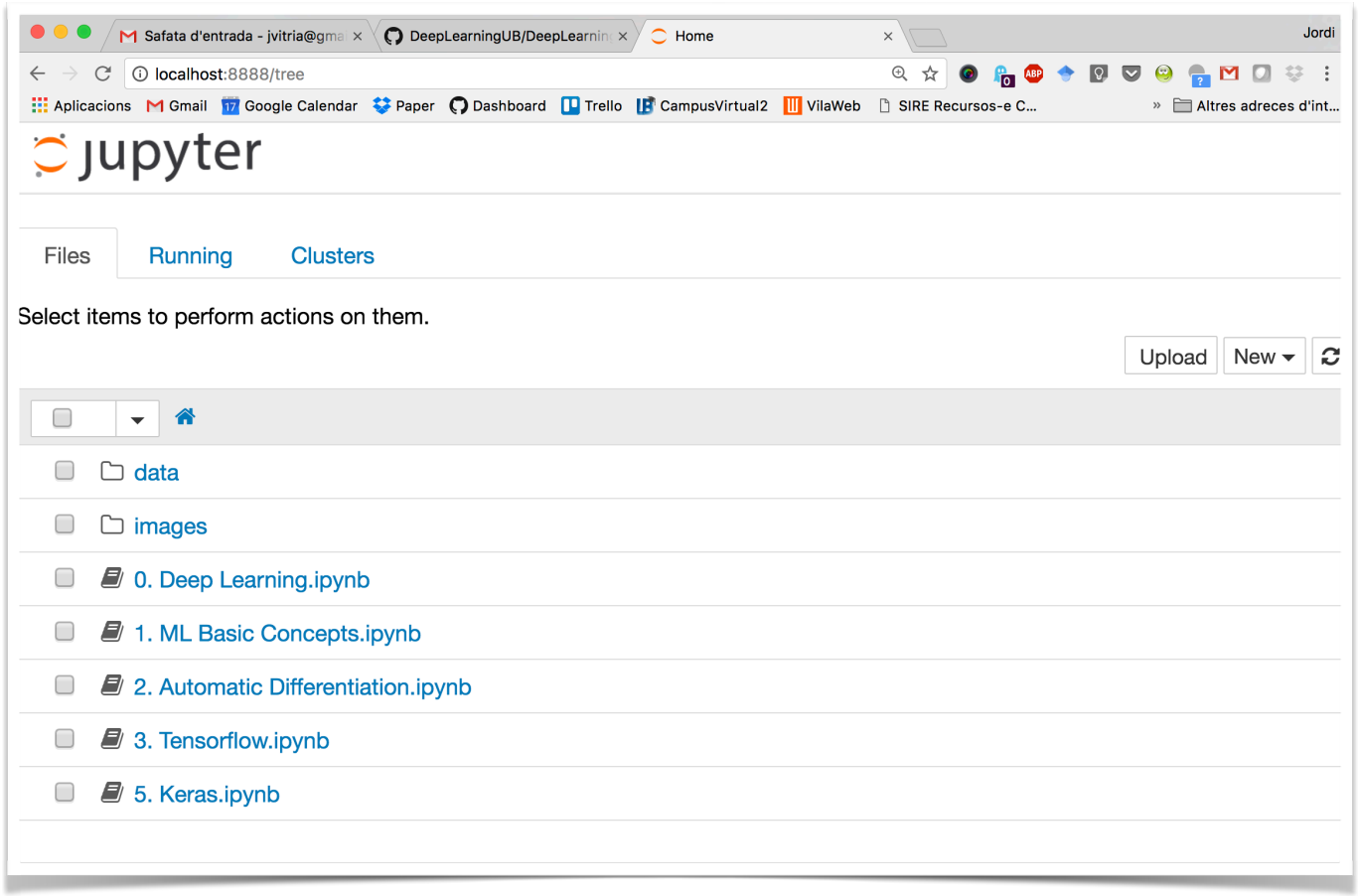
Software Installation

The best way to run the course software is to use a **Docker container**. There's full documentation on installing Docker at `docker.com`, but in a few words, the steps are:

- Go to `docs.docker.com` in your browser.
- Step one of the instructions sends you to download Docker.
- Run that downloaded file to install Docker.
- At the end of the install process a whale in the top status bar indicates that Docker is running, and accessible from a terminal.
- Click the whale to get `Preferences`, and other options.
- Open a command-line terminal, and run some Docker commands to verify that Docker is working as expected. Some good commands to try are `docker version` to check that you have the latest release installed, and `docker ps` and `docker run hello-world` to verify that Docker is running.
- By default, Docker is set to use 2 processors. You can increase processing power for the app by setting this to a higher number in `Preferences`, or lower it to have Docker for Mac use fewer computing resources.
- Memory - By default, Docker is set to use 2 GB runtime memory, allocated from the total available memory on your computer. You can increase the RAM on the app to get faster performance by setting this number higher (for example to 3) or lower (to 1) if you want Docker to use less memory.

Once Docker is installed, you can dowload the image of this course and download the git repository:

- In a terminal, go to your course folder and run (This operation requires a good internet connection; it will take some minutes): `docker pull datascienceub/deepub`
- MacOS & Linux: Run the `deepub` image on your system: `docker run -it -p 8888:8888 -p 6006:6006 -v /$(pwd):/notebooks datascienceub/deepub`
- Windows: Run the `deepub` image on your system: `docker run -it -p 8888:8888 -p 6006:6006 -v C:/your/course/folder:/notebooks datascienceub/deepub`
- Once these steps have been done, you can check the installation by starting your web browser and introducing this URL: `http://localhost:8888`.
- Open a new Jupyter notebook and execute this instruction in a code cell: `!git clone https://github.com/DeepLearningUB/DeepLearningGirona2017`.


Safata d'entrada - jvitria@gma
DeepLearningUB/DeepLearnin
Home
Jordi
localhost:8888/tree
Aplicacions
Gmail
Google Calendar
Paper
Dashboard
Trello
CampusVirtual2
VilaWeb
SIRE Recursos-e C...
Altres adreces d'int...
jupyter
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
data
images
0. Deep Learning.ipynb
1. ML Basic Concepts.ipynb
2. Automatic Differentiation.ipynb
3. Tensorflow.ipynb
5. Keras.ipynb

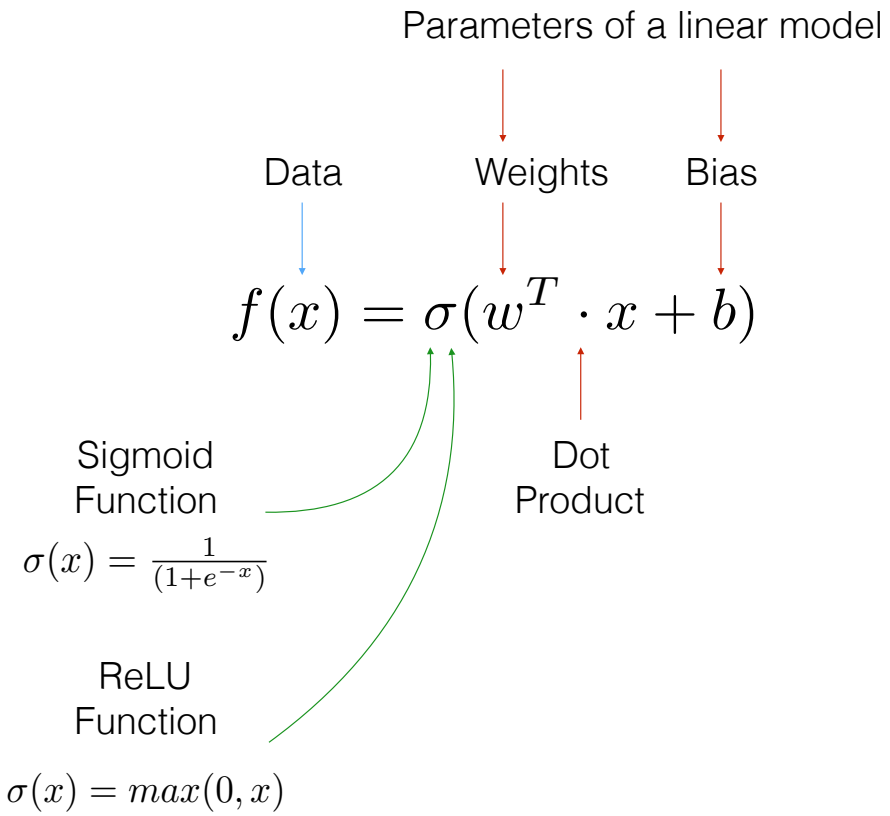
What is Deep Learning?

Learning from Data

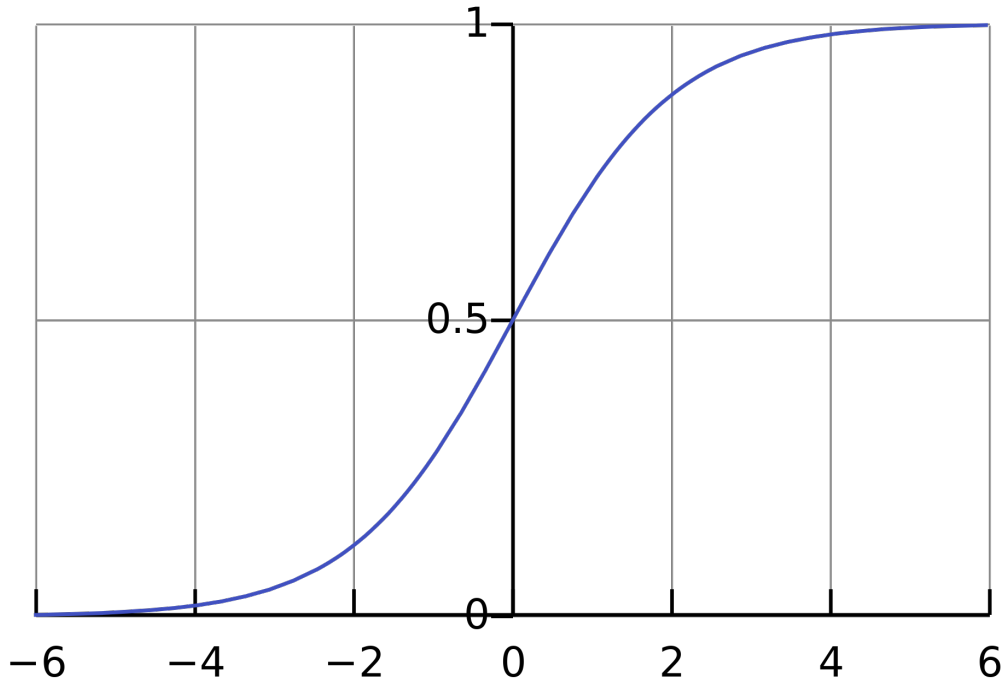
Training data: a set of $(x^{(m)}, y^{(m)})$ pairs.
Learn a function $f_w : x \to y$ to predict on new inputs x.

1. Choose a model function family f_w.
2. Optimize parameters w.

1-layer neural net model



Non-linearity



$$\sigma(x) = \frac{1}{(1+e^{-x})}$$

Non-linearity

$$\sigma(x) = max(0, x)$$

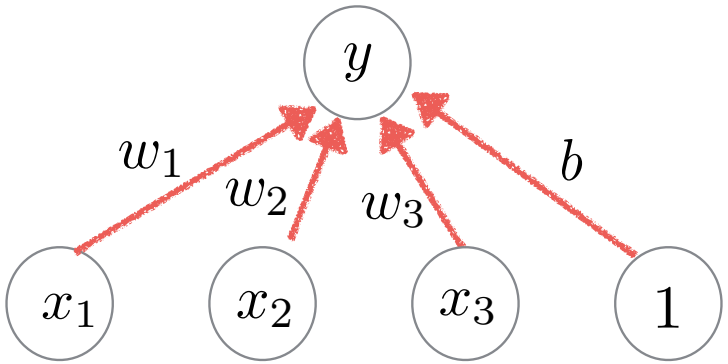
Non-linearity

Table 3: Non-linearities tested.

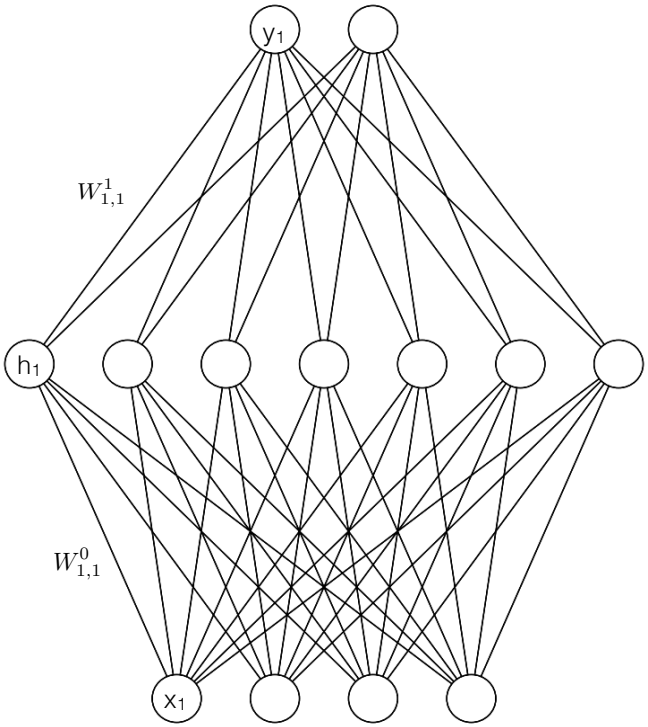
Name	**Formula**	**Year**
none	$y = x$	-
sigmoid	$y = \frac{1}{1+e^{-x}}$	1986
tanh	$y = \frac{e^{2x}-1}{e^{2x}+1}$	1986
ReLU	$y = \max(x, 0)$	2010
(centered) SoftPlus	$y = \ln(e^x + 1) - \ln 2$	2011
LReLU	$y = \max(x, \alpha x)$, $\alpha \approx 0.01$	2011
maxout	$y = \max(W_1 x + b_1, W_2 x + b_2)$	2013
APL	$y = \max(x,0) + \sum_{s=1}^{S} a_i^s \max(0, -x + b_i^s)$	2014
VLReLU	$y = \max(x, \alpha x)$, $\alpha \in 0.1, 0.5$	2014
RReLU	$y = \max(x, \alpha x)$, $\alpha = \text{random}(0.1, 0.5)$	2015
PReLU	$y = \max(x, \alpha x)$, α is learnable	2015
ELU	$y = x$, if $x \geq 0$, else $\alpha(e^x - 1)$	2015

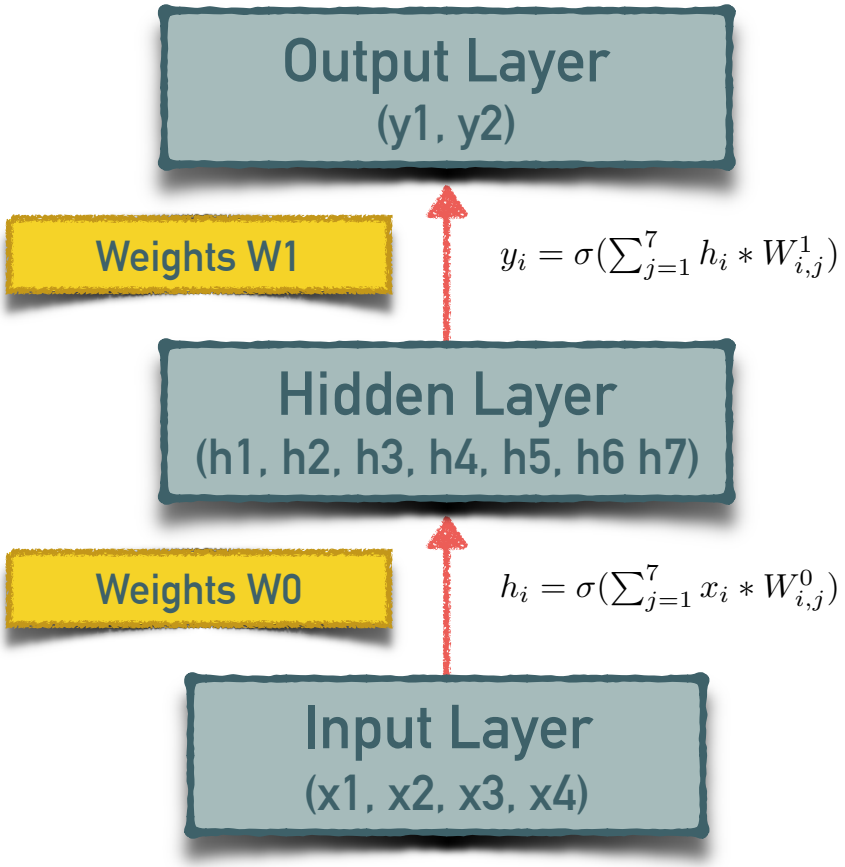
Graphical representation of 1-layer neural net model

$$f(x) = \sigma(w^T \cdot x + b)$$



2-layer neural net model





Multilayer Perceptrons

Convolutional Neural Networks

Recurrent Neural Networks (LSTM, GRU)

Non Supervised NN (Autoencoders, GAN, etc.)

....

Today, we can train very complex NN models because we have built in software a complex abstraction process that relies on several interesting (new and old) results:

Abstraction layers include:

- **Optimization & SGD.**
- **Automatic Differentiation.**
- **Computational Graphs and Data Flow Processing.**
- **Deep Learning Tricks.**