Introduction to Machine Learning

Chap I: Basic concepts

Julien Donini
LPC/Université Clermont Auvergne

Foreword

Since about **10 years** a **new era** started for **Machine Learning**

- **ML librairies** became much accessible
- **Fast** execution of code (**G**raphics **P**rocessing **U**nits)
- High performance **computing**: data centers, clusters
- New ideas and **algorithms**: VAE (2013), GAN (2104), ADAM (2014)…
- Complexity is not an issue: **Deep Learning, ...**
- Extremely large area of **applications**: industry, science, ...

Objective of this lecture

Demistification of “Artificial Intelligence”

Basic **understanding** = stats and maths

Knowledge of some common algorithms

Motivate you to go further and **practice** ML

Outline of the ML course

Lectures

Chap I: Basic concepts on ML

Chap II: Regression

Chap III: Classification

Practice sessions

Introduction to Machine Learning

Code on Git:

https://github.com/judonini/MLcourses

Sections marked with (*) are more advanced → non examinable for DU students

This week

Practice sessions

S50

Lundi 12/12/2022	Mardi 13/12/2022	Mercredi 14/12/2022	Jeudi 15/12/2022	Vendredi 16/12/2022
Machine Learning 1/6 09h30 - 11h00 DONINI JULIEN ST PPHY 9211 M2 UP DU Data Scientist	**Machine Learning** 3/6 09h30 - 11h00 DONINI JULIEN ST PPHY 9213 M2 UP DU Data Scientist	**Machine Learning.** 2/4 09h30 - 12h30 DONINI JULIEN SCI 007 M2 UP DU Data Scientist	**Machine Learning** 6/6 09h30 - 12h00 DONINI JULIEN ST PPHY 9213 M2 UP DU Data Scientist	**Machine Learning.** 4/4 09h30 - 12h30 DONINI JULIEN SCI 004 M2 UP DU Data Scientist
Machine Learning 2/6 11h15 - 12h45 DONINI JULIEN ST PPHY 9211 M2 UP DU Data Scientist	**Machine Learning** 4/6 11h15 - 12h45 DONINI JULIEN ST PPHY 9213 M2 UP DU Data Scientist	**Machine Learning** 5/6 14h45 - 16h15 DONINI JULIEN ST PPHY 9211 M2 UP DU Data Scientist	**Machine Learning.** 3/4 13h00 - 15h00 DONINI JULIEN SCI 004 M2 UP DU Data Scientist	
Machine Learning. 1/4 14h30 - 17h30 DONINI JULIEN ST PHY2 214 M2 UP DU Data Scientist				

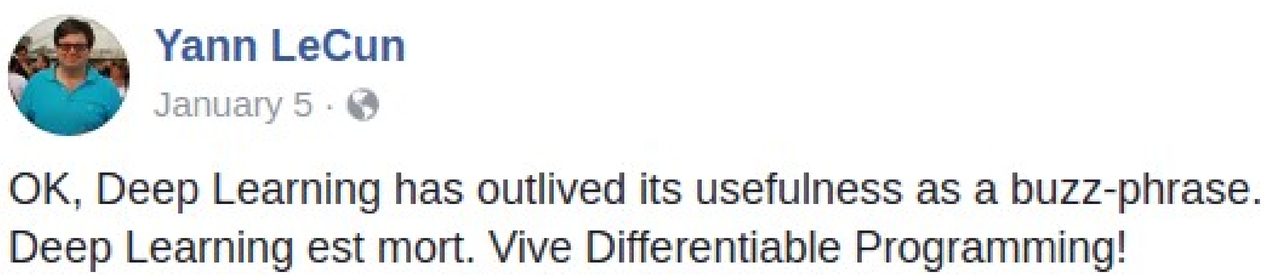
Machine Learning: statistics + computing + “learn” parameters from data

Big Data: same as above + techniques to handle lots of data

Artificial intelligence: same as above but sounds smarter

Data Science: same as above but sounds more scientific

Differentiable programming:



Applications

- **Speech** and **handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- …

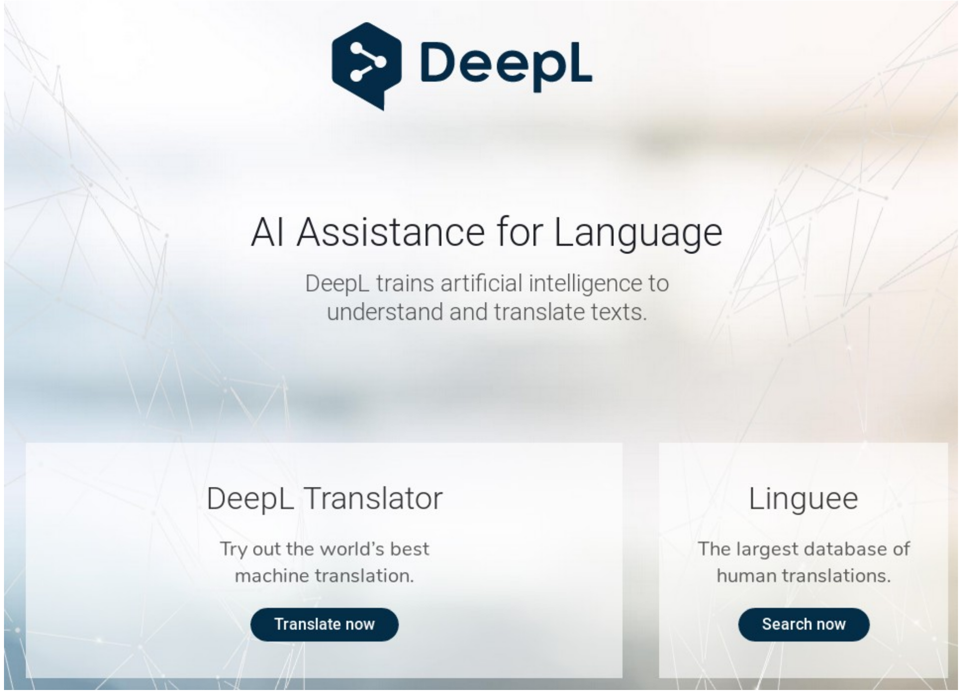
MNIST database
60,000 training images
10,000 testing images.

Human error rate ~ 2%

Best ML error rate: 0.2%

Applications

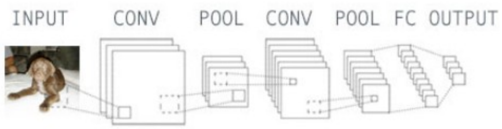
- **Speech** and **handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- …

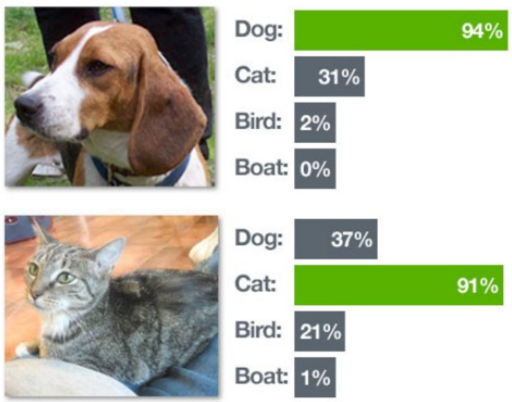


Deep learning translator
https://www.deepl.com

Applications

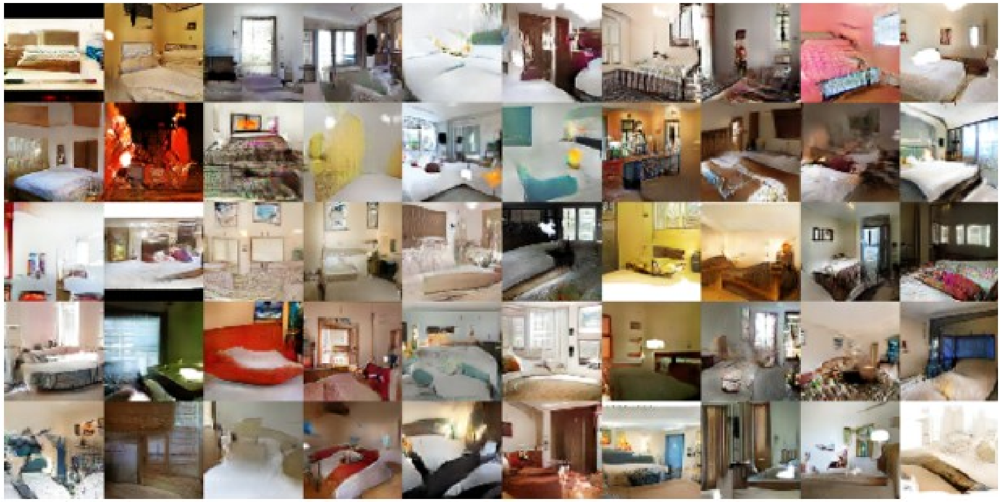
- **Speech** and **handwriting** recognition
- **Language** processing
- **Image** recognition
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- …





Applications

- **Speech** and **handwriting** recognition
- **Language** processing
- **Image** ~~recognition~~ **generation !**
- **Fraud** detection
- **Financial** market analysis
- **Search engines**
- **Spam** and **virus** detection
- **Medical** diagnosis
- **Robotic** control
- **Automation** (self-driving cars)
- **Advertising**
- **Physical** science
- …



(Arjovsky et al, 2017)

Warm up

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

What is Machine Learning

Data

weights

Output

$$y(x,{\color{blue}w}) = {\color{blue}t}$$

Target

What is Machine Learning

Data

weights

Output

$$y(x,w) = t$$

Target

Examples

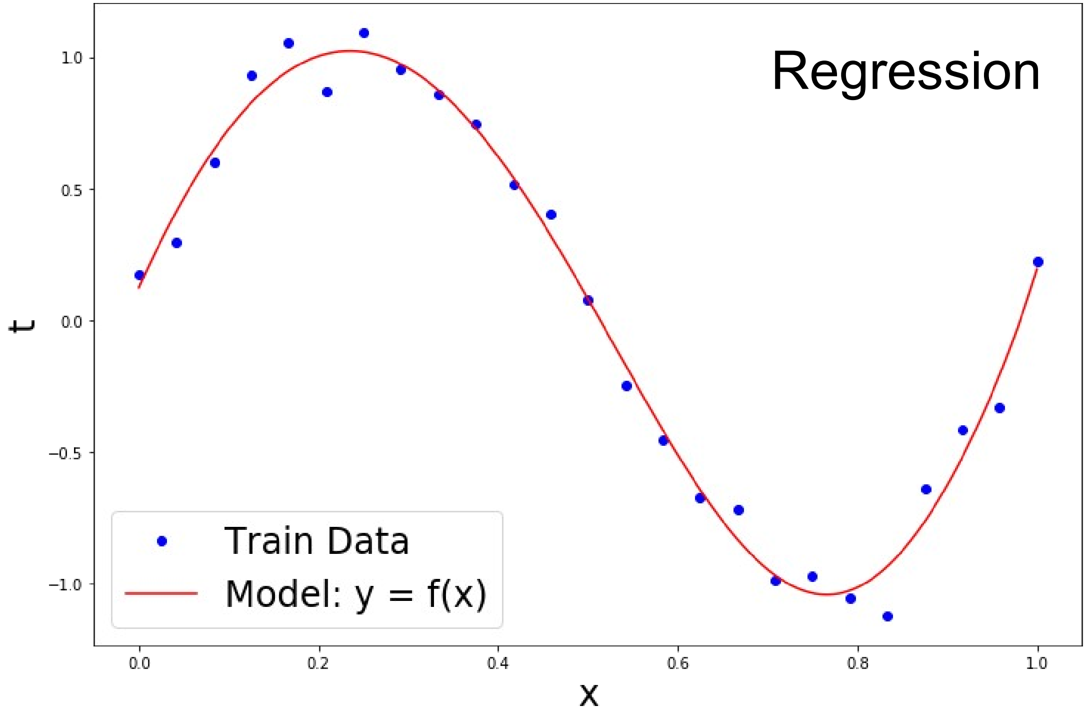
- **X** = {age, year, education, …} → **t**: income
- **X** = {image pixel values} → **t**: face recognition
- **X** = {list of words} → **t**: spam detection
- **X** = {E, p, …} → **t**: particle detection

  ...

Data
(features **x**)

Algorithm
y = f(x)

Objective
(target **t**)



Machine Learning Basics

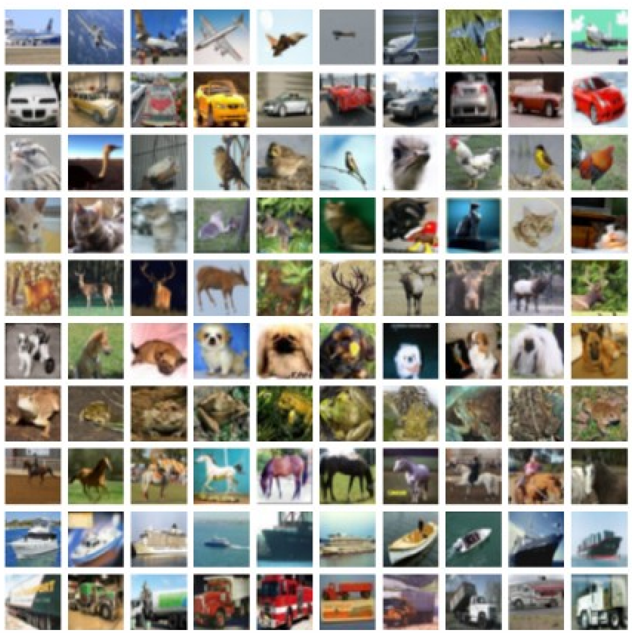
Data
(features **x**) → **Algorithm** **y = f(x)** → **Objective** (target **t**)

Classification → **Dog**

```
[[[ 7.4280e-02,  1.4022e-01, -2.2258e-02,  ..., -2.0172e-01,
    1.6240e-01,  5.5748e-02],
  [-1.1771e-02, -1.1327e-01,  3.0360e-01,  ...,  4.6299e-01,
    3.4765e-02,  2.2633e-02],
  [ 2.2252e-02,  2.1568e-01, -3.5726e-01,  ..., -7.4589e-02,
    7.0776e-02,  1.3573e-01],
  ...,
  [ 1.1035e-01, -2.4609e-01,  1.9962e-01,  ...,  2.4133e-01,
   -2.1069e-01,  1.9942e-01],
  [ 2.9337e-02,  2.4997e-01,  1.0341e-02,  ..., -3.1368e-01,
   -1.6878e-01, -1.4741e-02],
  [ 4.4006e-02,  5.1292e-02,  5.0462e-02,  ..., -8.1194e-02,
    1.6043e-01, -5.7106e-03]]],
```

Machine Learning Basics

Data
(features **x**) → **Algorithm** **y = f(x)** → **Objective** (target **t**)

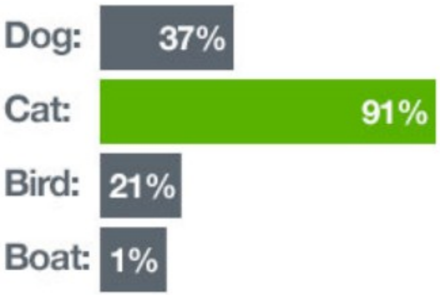


Training →

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

Data
(features **x**) → **Algorithm** **y = f(x)** → **Objective** (target **t**)

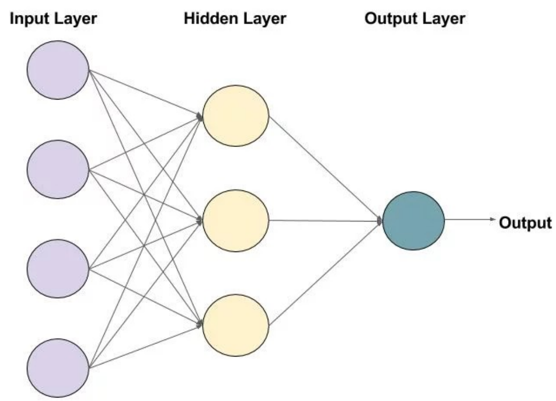
Test →



Example: Neural Networks

Data

(features **x**)



Objective

(target **t**)

y = f(**x**, **W**)

f : non-linear functions

W : parameters (weights)

Cost function

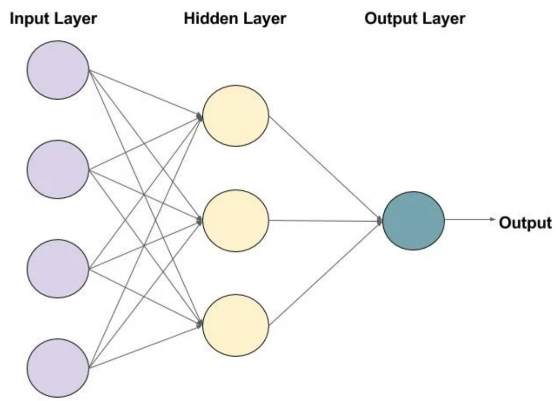
"how close is the network output to the objective ?"

$$\ell(\mathbf{y}, \mathbf{t})$$

Example: Neural Networks

Data

(features **x**)



Objective

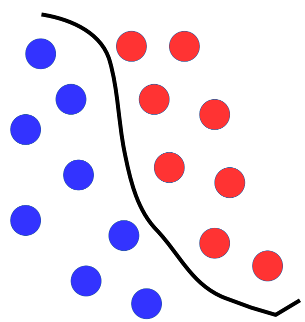
(target **t**)

Update of weights **W**

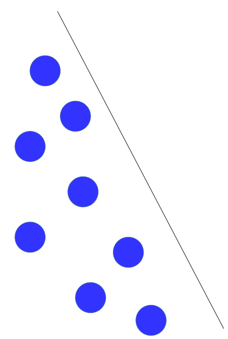
$$\mathbf{W} \to \mathbf{W} - \eta \sum_{N} \frac{\partial \ell(\mathbf{y}, \mathbf{t})}{\partial \mathbf{W}}$$

Common type of learning

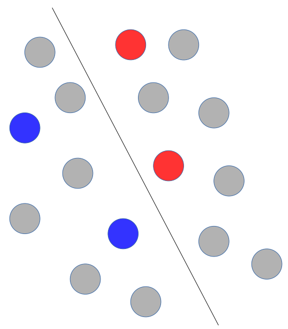
Supervised (labels are known)



Unsupervised (no labels)



Semi-supervised (few labels)



labels of class 1

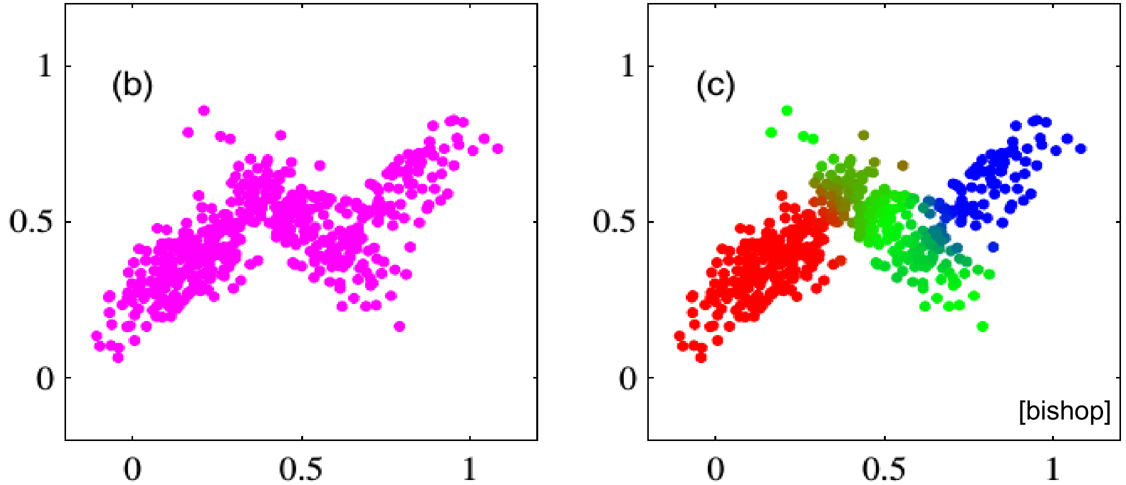
labels of class 2

unknown class

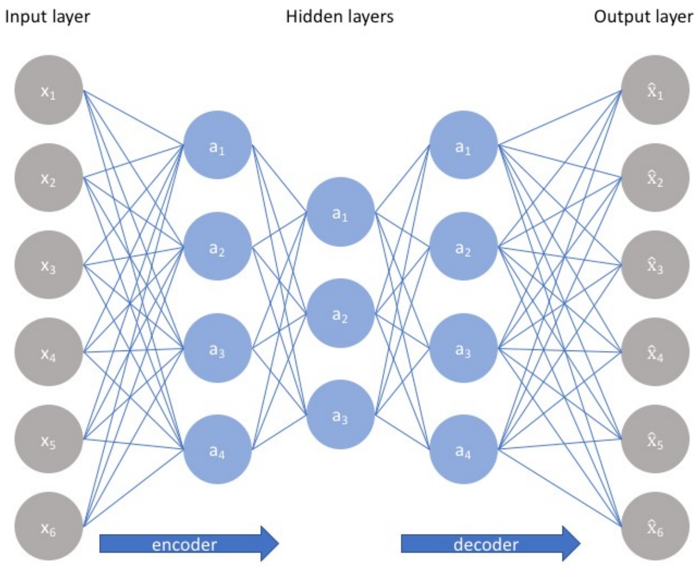
decision boundary

Unsupervised learning = no labels

Clustering

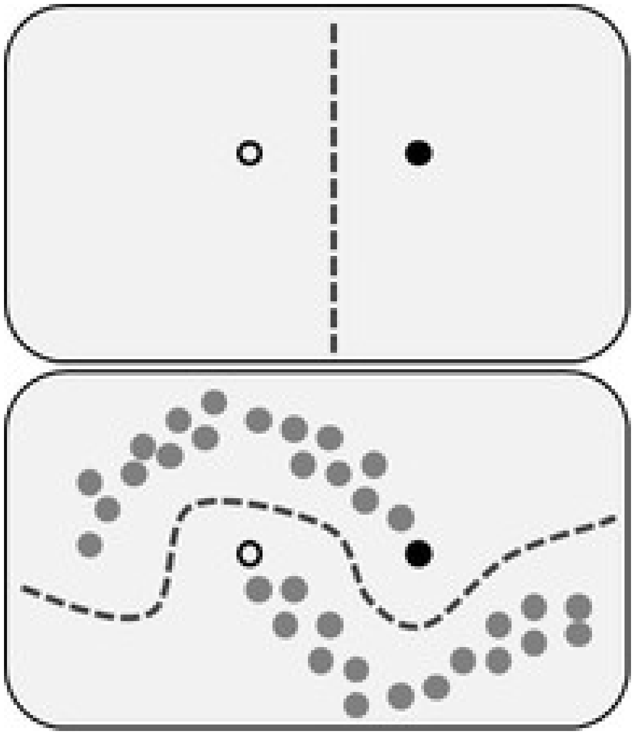


Dimensionality reduction



Semi-supervised learning

Semi-supervised learning = unlabelled data + few labels



[wikipedia]

Example of the influence of unlabelled data in semi-supervised learning.

The unlabelled data (grey dots) influence the separation of the two classes (decision surface)

Representation learning

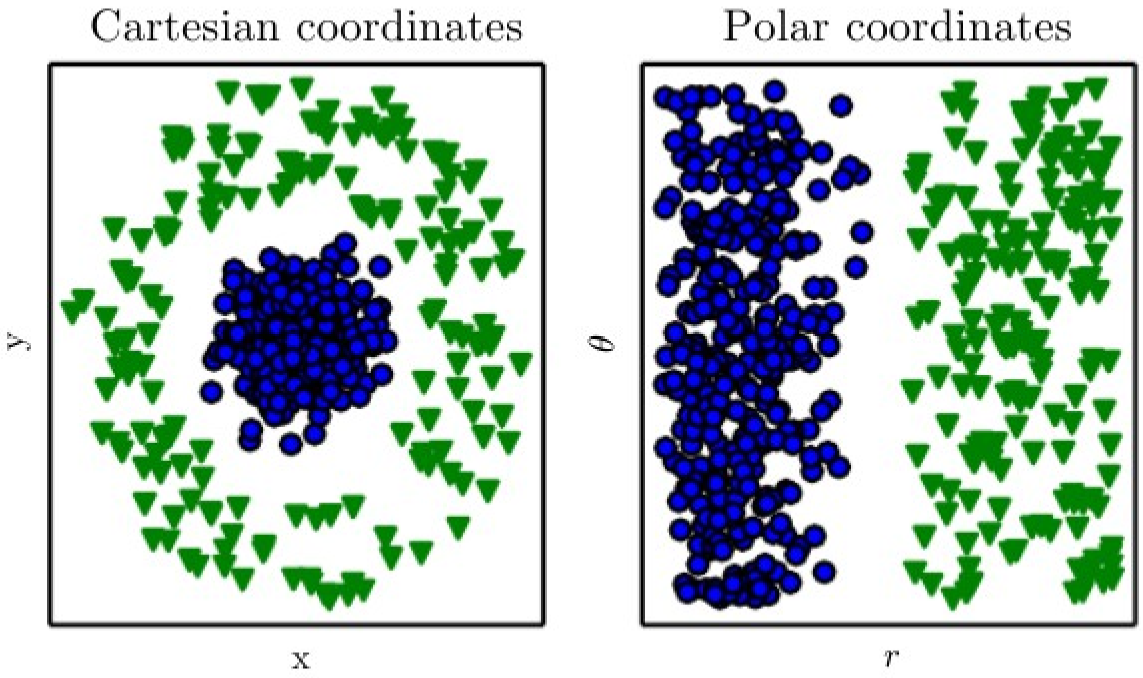
Representation is how we present the information (data) to solve a problem

Example: what is the best number representation to perform this division ?

Decimal	121 : 11
Roman	CXXI / XI
Binary	1111001 : 1011
Hex	79 : B
ASCII	y : VT

Representation is how we present the information (data) to solve a problem

Example: what are the best coordinates to separate this data ?



[deeplearningbook]

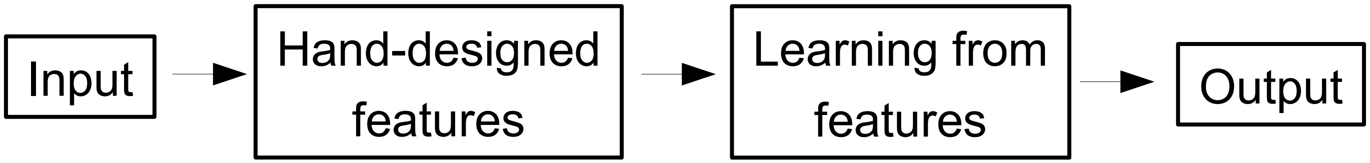
Representation learning

Representation is how we present the information (data) to solve a problem

For ML a good representation is one that makes the **learning task easier**

ML algorithms can also learn best representation: **representation learning**
→ Central to **deep learning** : learn complex representation from simpler ones

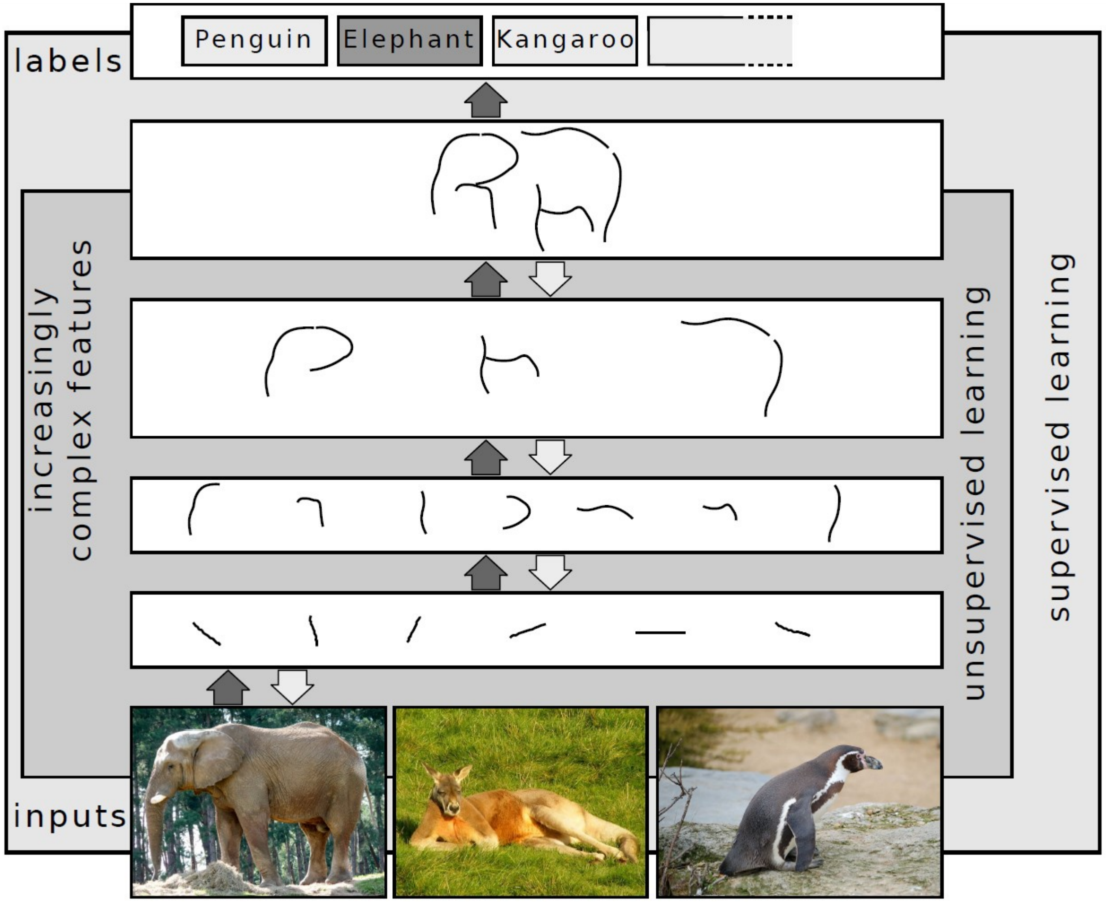
Classic ML



Deep learning

Deep learning

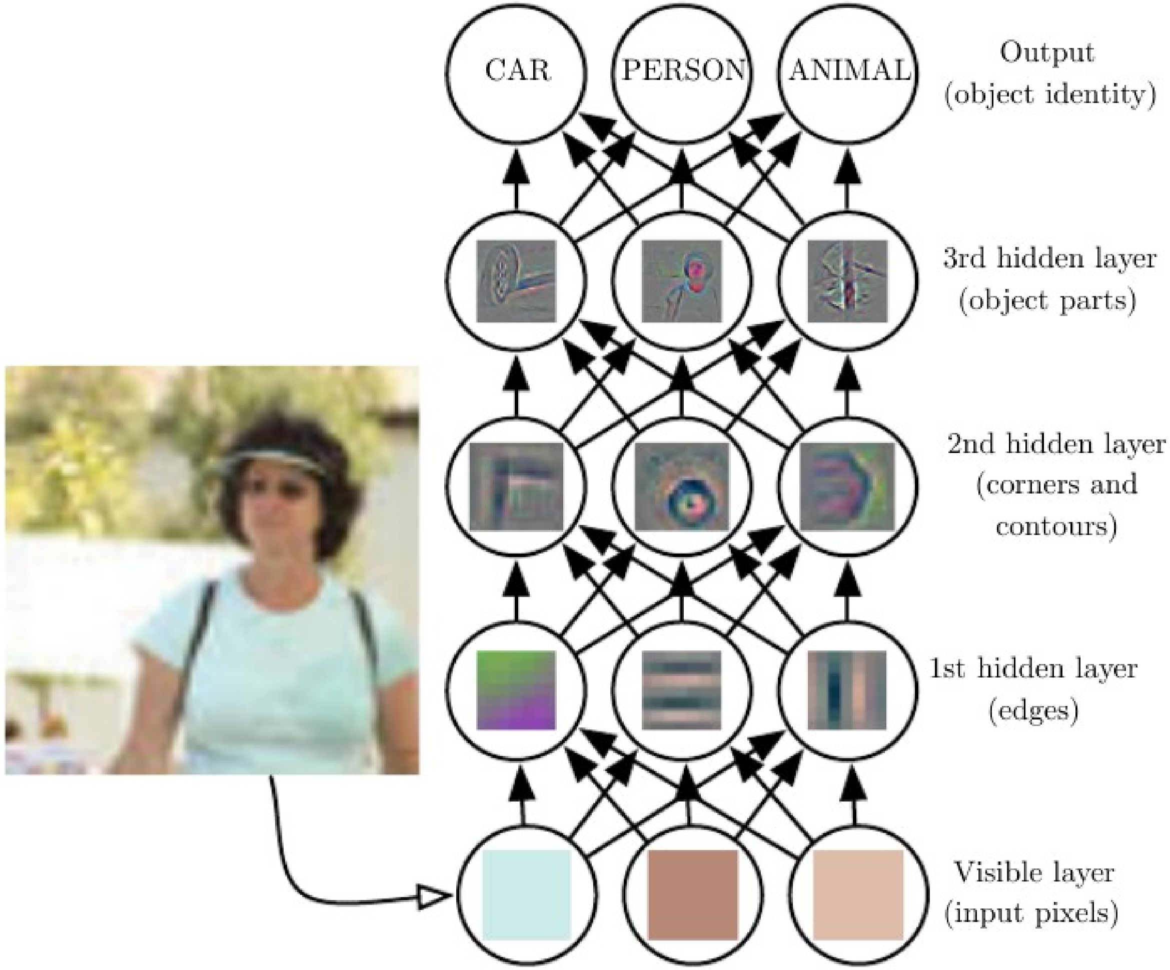
Class of **ML algorithms** (in general artificial neural networks) that use **multiple layer** to **extract higher level features** from raw data



[wikipedia]

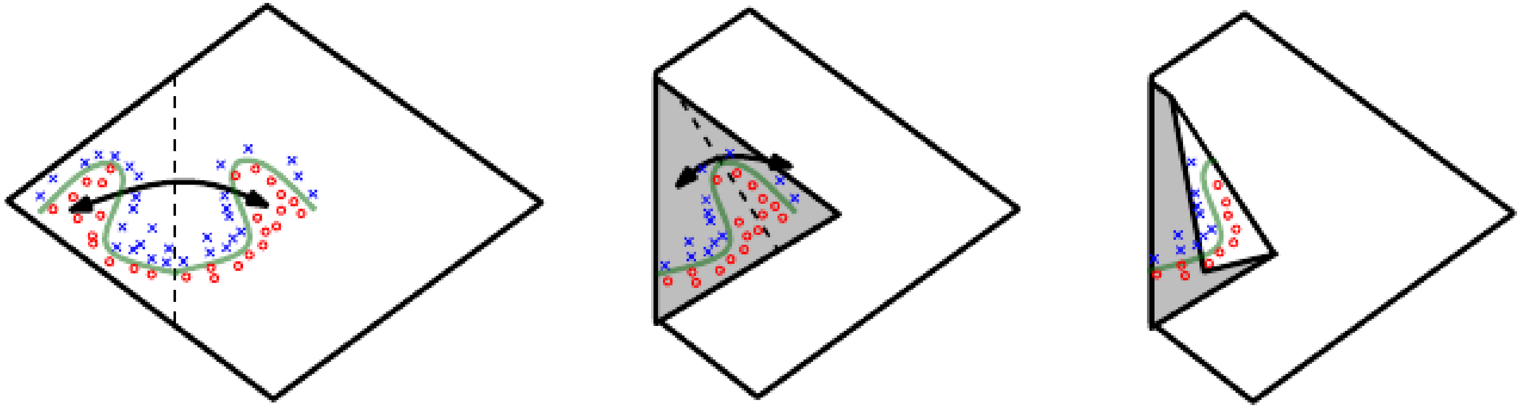
Deep learning

Class of **ML algorithms** (in general artificial neural networks) that use **multiple layer** to **extract higher level features** from raw data



[deeplearningbook]

Adding layers can help uncovering specific data patterns [Montufar, 1402.1869]:



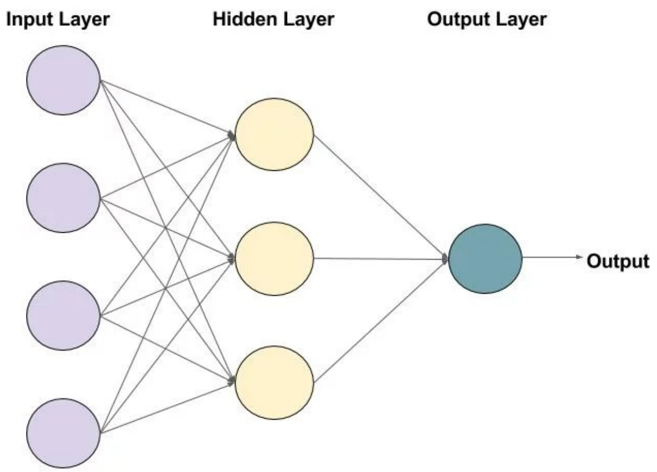
The absolute value activation function $g(x_1,x_2) \rightarrow |x_1|,|x_2|$ folds a 2D space twice.

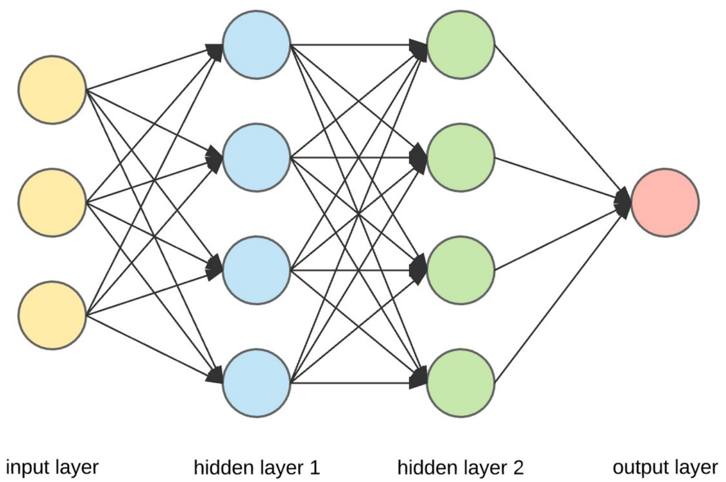
Each hidden layer of a deep neural network can be associated to a folding operator.

The folding can **identify symmetries** in the boundaries that the NN can represent.

Shallow Neural Networks

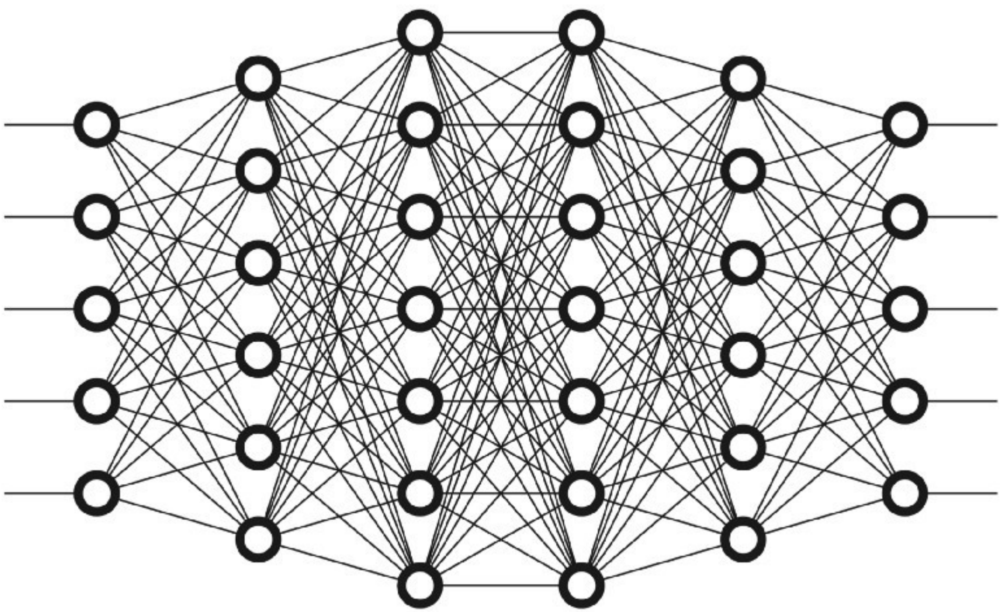
(1 or 2 hidden layers)

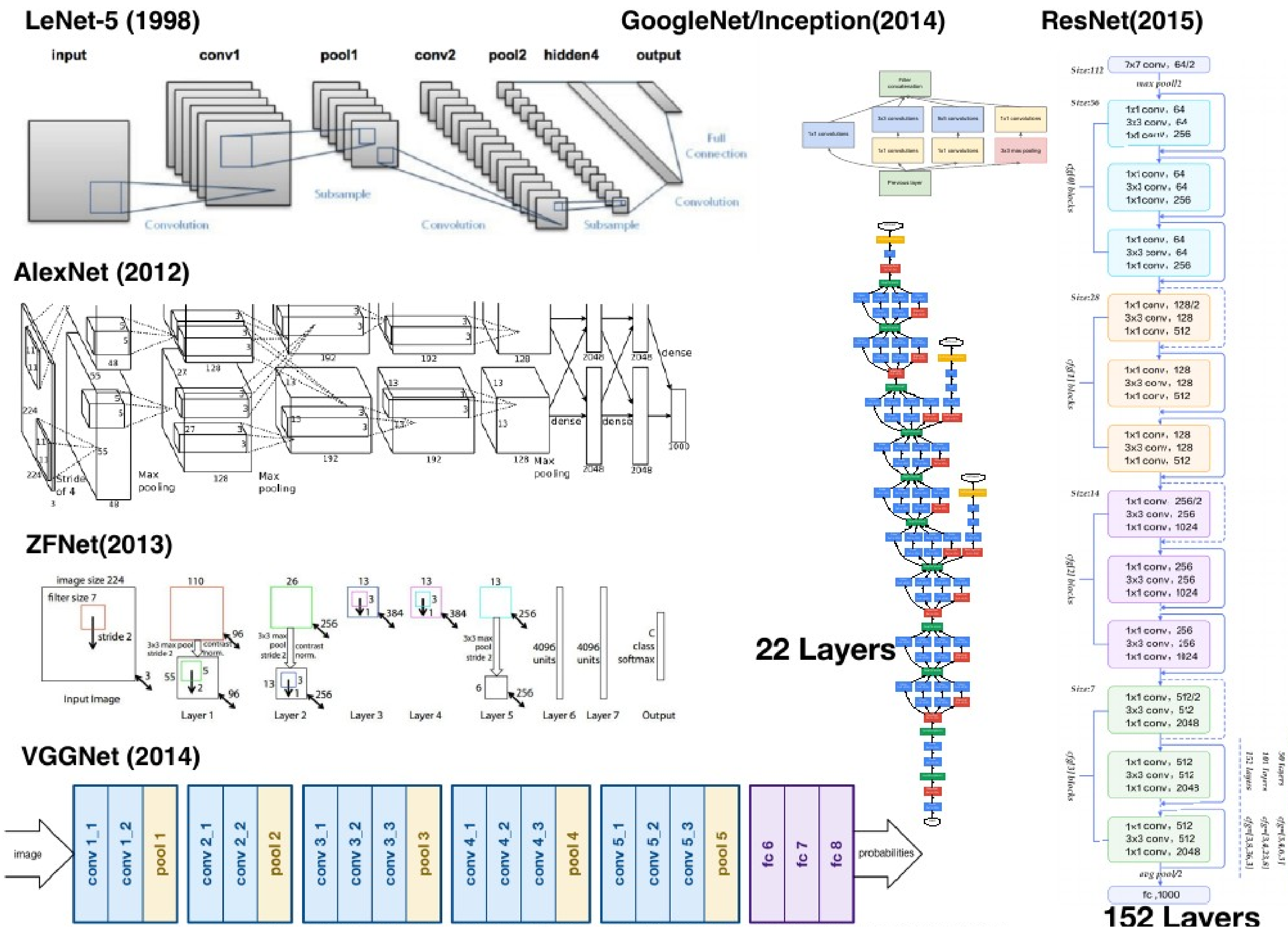




Deep Neural Network

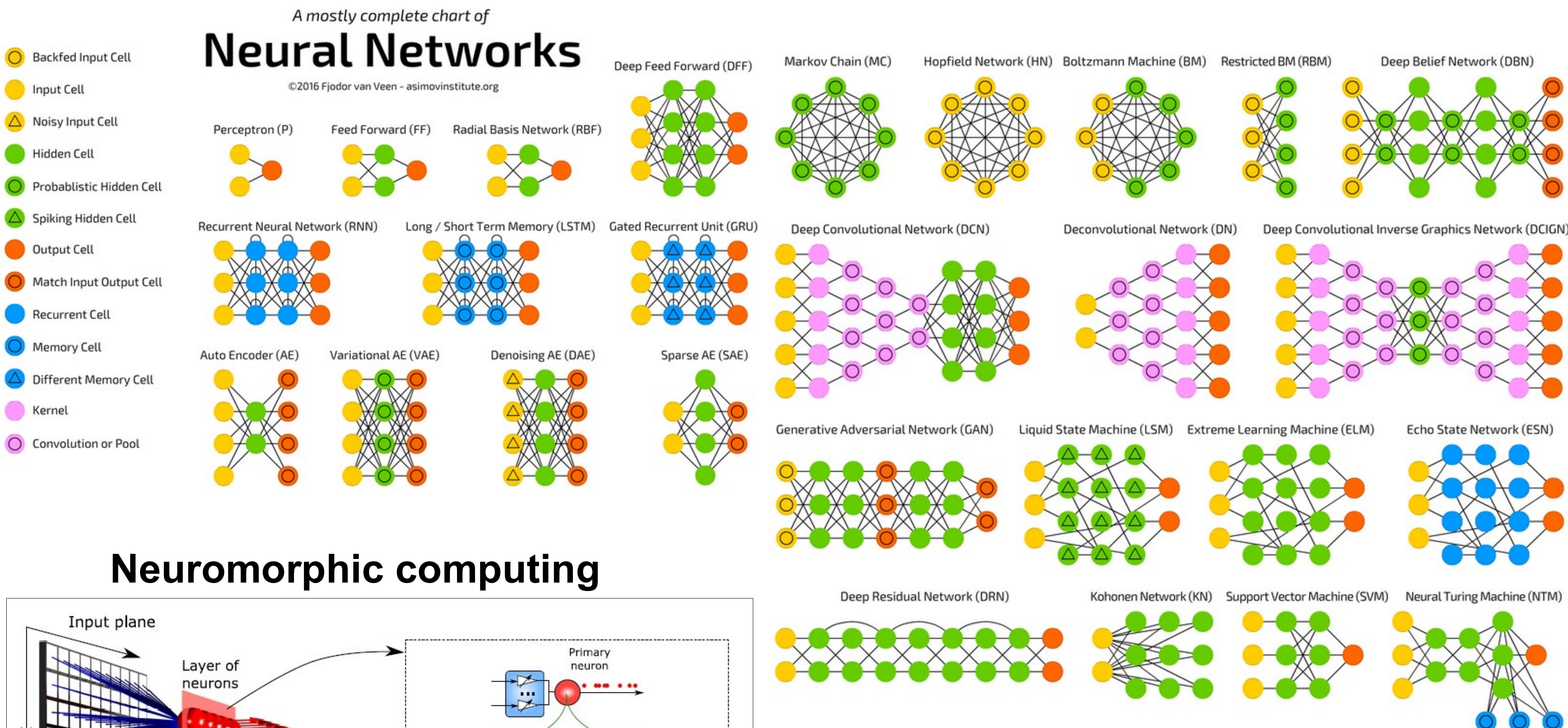
(more layers)





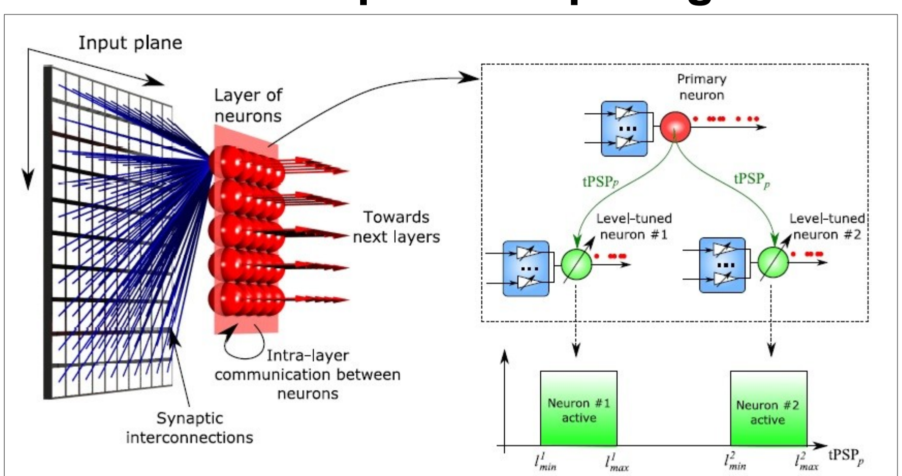
https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

Neural Networks today



http://www.asimovinstitute.org/neural-network-zoo/

Neuromorphic computing

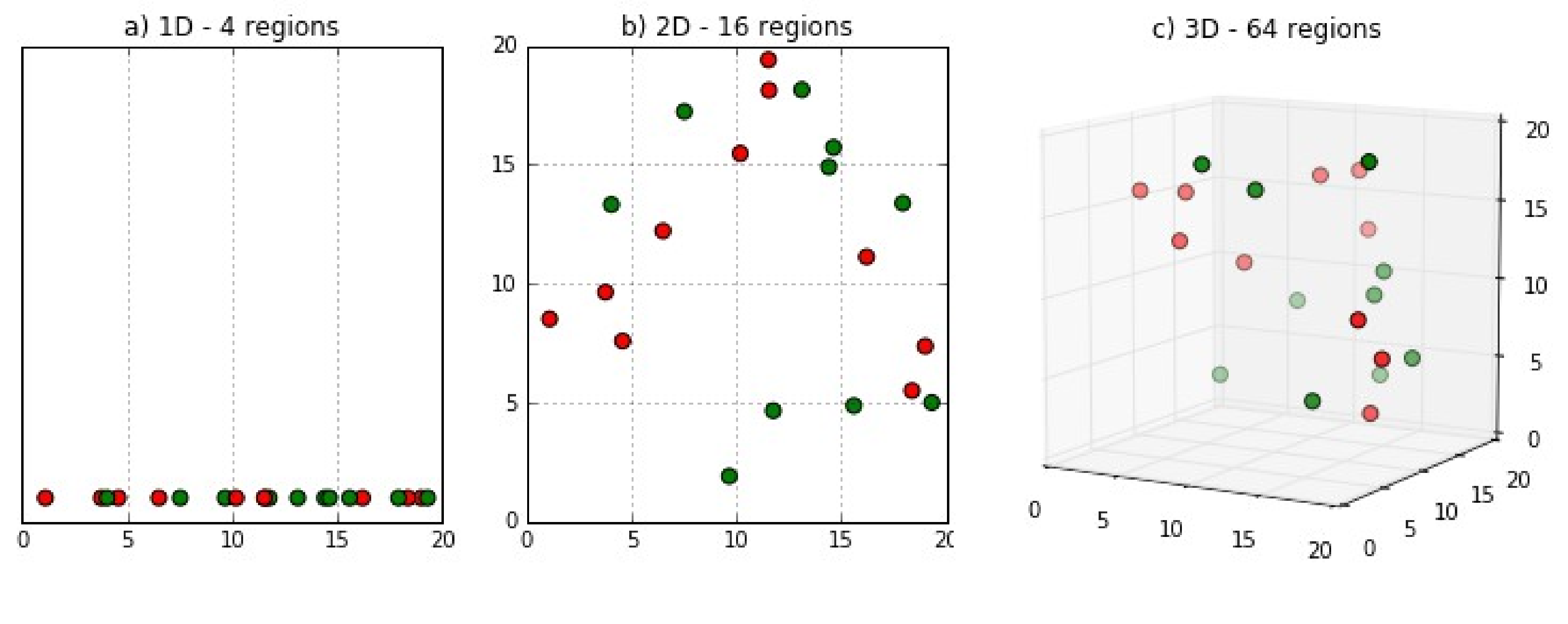


Limitations and difficulties

Deep learning algorithms are very efficient in many domains (in particular image recognition), but they require **a lot of data to train** because of large number of **dimensions** and high number of **hyperparameters.**

Illustration: the "curse of dimensionality

Simple example: classify each region depending on majority of labels



As dimensions grows, dimensions space increases exponentially.
Classification ok for ~2 variables, but fails at higher dimensions !

Limitations and difficulties

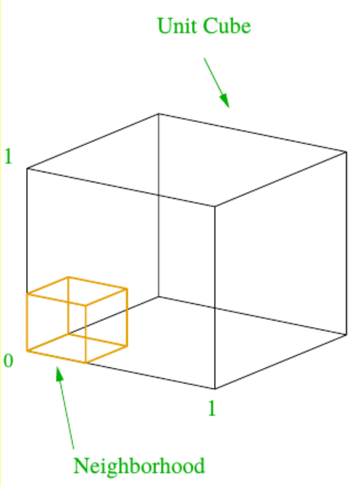
Deep learning algorithms are very efficient in many domains (in particular image recognition), but they require **a lot of data to train** because of large number of **dimensions** and high number of **hyperparameters.**

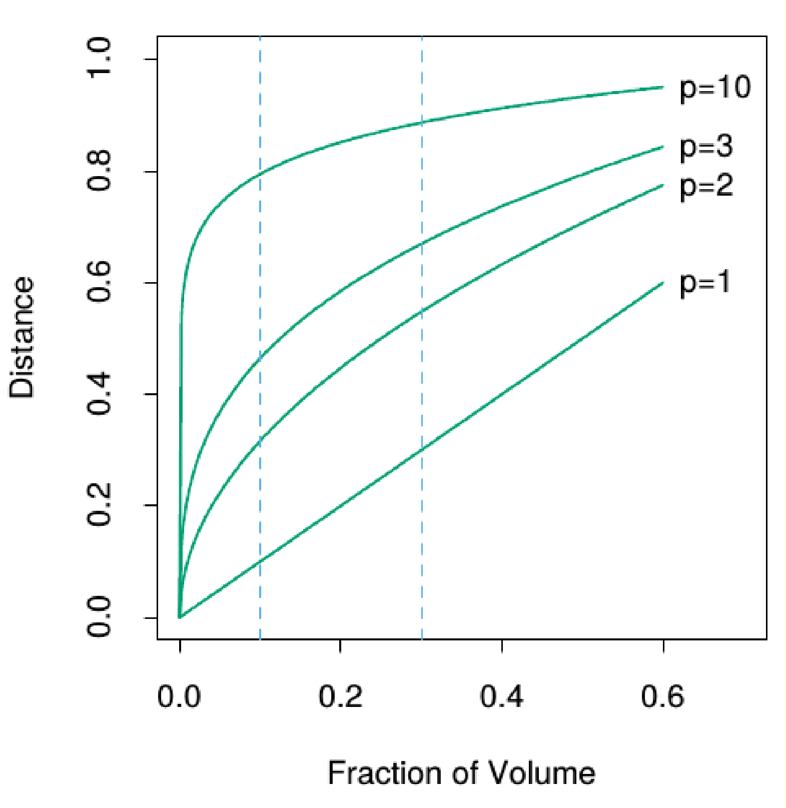
Illustration: the “curse of dimensionality

In a hypercube of length 1 and dimension **p** a fraction **r** of the volume corresponds to an edge length $\mathbf{e_p(r) = r^{1/p}}$

- p=10 and r=1%: $e_{10}(0.01)=0.63$
- p=10 and r=10%: $e_{10}(0.1)=0.80$

To contain 1% or 10% of the volume we must cover 63% or 80%, respectively, of the range of each input variable !





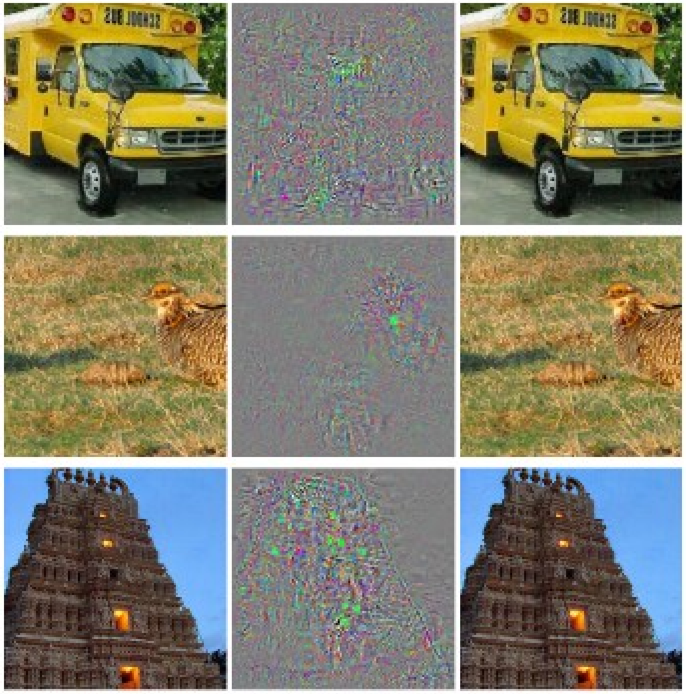
Algorithms get so **complex** that it is difficult to **interpret** what they really do !

Also their complexity can become a **weakness/threat**:

School bus

Bird (partridge)

Temple



[arxiv:1312.6199]

Limitations and difficulties

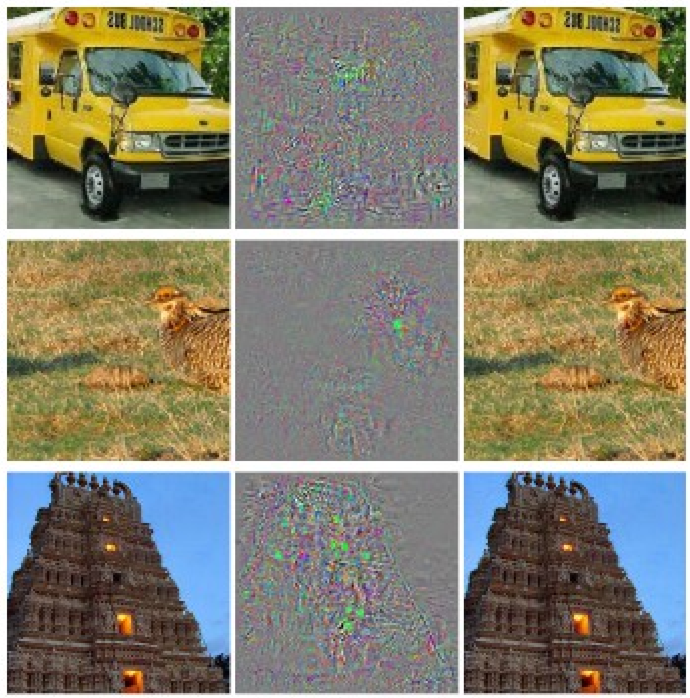
Algorithms get so **complex** that it is difficult to **interpret** what they really do !

Also their complexity can become a **weakness/threat**:

School bus

Bird (partridge)

Temple



Ostrich !

Ostrich !

Ostrich !

?!

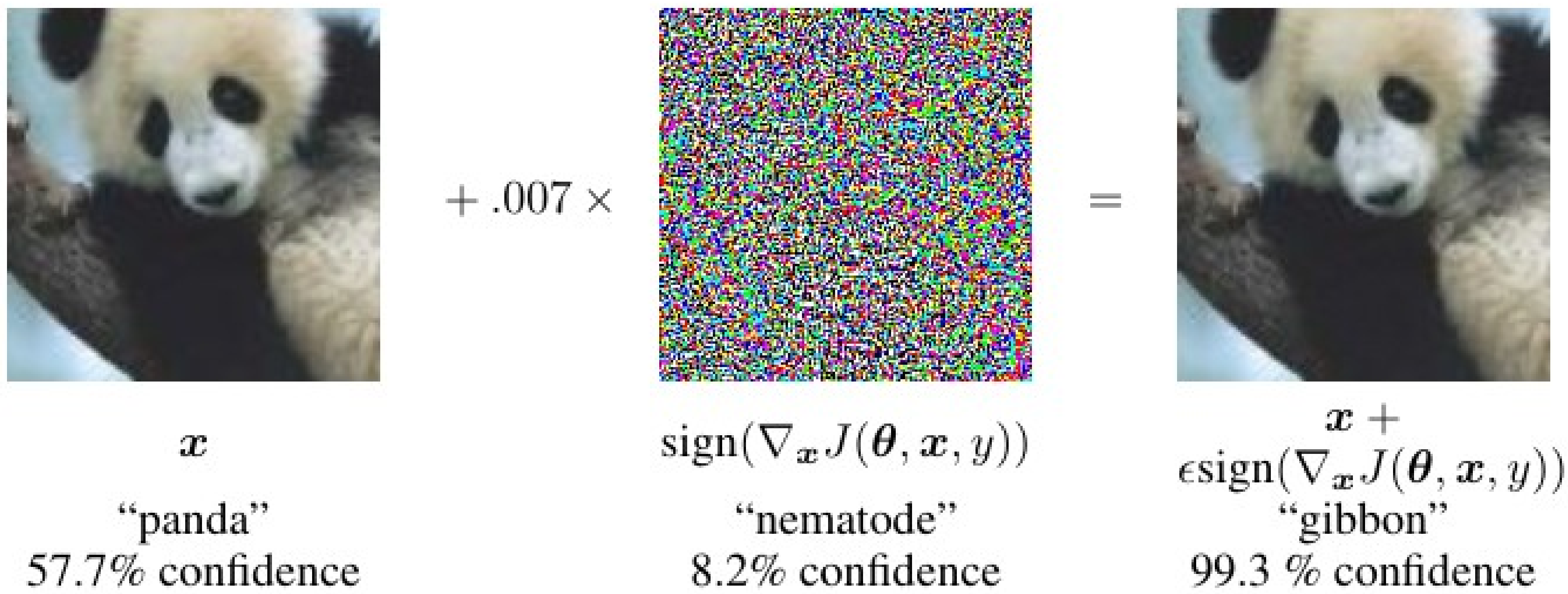
Perturbation

[arxiv:1312.6199]

Limitations and difficulties

Algorithms get so **complex** that it is difficult to **interpret** what they really do !

Also their complexity can become a **weakness/threat**:



[arxiv:1412.6572]

References

AAAH! JE SUIS VOTRE PRISONNIÈRE, MONSIEUR GASTON...
...MONSIEUR GASTON? OÙ ÊTES-VOUS?
ICI, 'MOISELLE JEANNE! PRÈS DU COURRIER DE 1969...
IL Y EN A UN QUI TRICHE: IL DÉPLACE DES LIVRES...
ROGNN-TUDJUU!
PAR ICI
818
SCEN: DOGLU
FRANQUIN

References (non exhaustive !) & credits

Classical Machine Learning textbooks

- **Elements of statistical learning** (ESL), Hastie et al., Springer
- **An Introduction to Statistical Learning** (ISLR), Hastie et al. Springer
  - Both books available online: http://web.stanford.edu/~hastie/pub.htm
- **Pattern Recognition and Machine Learning**, Bishop, Springer
- **Deep learning book,** I. Goodfellow et al, http://www.deeplearningbook.org/

A *lot* of courses, lectures and tutorial on the web

- **Online courses**: DataCamp, Coursera, Andrew Ng (http://cs229.stanford.edu/)
- **CERN** lectures (ex: Kagan https://indico.cern.ch/event/619370)
- **2 recommended lectures:**
  - **François Fleuret** (EPFL) **https://fleuret.org/ee559/**
  - **Gilles Louppe** (University Liège)https://github.com/glouppe/info8010-deep-learning
- ML **cheatsheet**: https://ml-cheatsheet.readthedocs.io/en/latest/index.html

Python resources

- A **Crash Course** in Python for Scientists : http://nbviewer.jupyter.org/gist/rpmuller/5920182
- Introduction to **scientific computing** with Python: http://github.com/jrjohansson/scientific-python-lectures
- Python **Tutorial**: https://www.codecademy.com/tracks/python

Notebooks basics

- **Installation** (recommended): https://www.anaconda.com/download
- **Jupyter** Notebook documentation: https://jupyter-notebook.readthedocs.io/en/stable/
- **Interactive** notebooks: https://mybinder.org/
- Introduction with **video** tutorial: https://www.youtube.com/watch?v=Duicsycntdo

Git

- **Git** documentation: https://book.git-scm.com/
- **Github**: https://github.com/
- **GitLab** (CERN) basics: https://gitlab.cern.ch/help/gitlab-basics/start-using-git.md
- **Tutorial** (in FR): https://github.com/clr-info/tuto-git

  https://openclassrooms.com/en/courses/1233741-gerez-vos-codes-source-avec-git

ML software and interfaces

Popular tools

- Data format: text, csv, images, HDF5, ...
- ML libraries: Keras+TensorFlow, Pytorch, scikit-learn (no DL), ...
- All kinds of popular algorithms: CNN, GAN, RNN, LSTM, AE, VAE ...

Tools specific for physicist (HEP)

- ROOT framework for data storage and processing
- Multivariate Analysis: TMVA for mostly BDT and (deep) NN
- PyMVA: Interface TMVA and Keras
- Several middleware file format conversion solutions:

arxiv:1807.02876

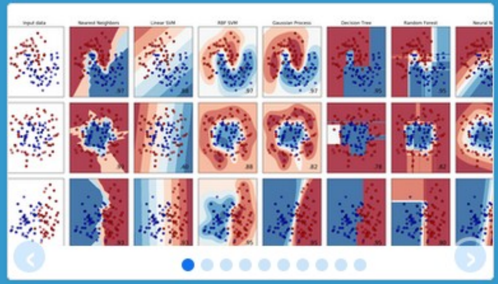
PyROOT	Python extension module that allows the user to interact with ROOT data/classes. [69]
root_numpy	The interface between ROOT and NumPy supported by the Scikit-HEP community. [65]
root_pandas	The interface between ROOT and Pandas dataframes supported by the DIANA/HEP project. [70]
uproot	A high throughput I/O interface between ROOT and NumPy. [71]
c2numpy	Pure C-based code to convert ROOT data into Numpy arrays which can be used in C/C++ frameworks. [72]
root4j	The hep.io.root package contains a simple Java interface for reading ROOT files. This tool has been developed based on freehep-rootio. [73]
root2npy	The `go-hep` package contains a reading ROOT files. This tool has been developed based on freehep-rootio. [73]
root2hdf5	Converts ROOT files containing TTrees into HDF5 files containing HDF5 tables. [74]

Scikit-learn (scikit-learn.org)

Home Installation Documentation Examples

Google Custom Search Search ×

Fork me on GitHub



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, … — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, … — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, … — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

News

On-going development: What's new (Changelog)

Community

About us See authors and contributing
More Machine Learning Find related projects

Who uses scikit-learn?

www.tensorflow.org

TensorFlow

Keras.io

Keras: The Python Deep Learning library

K Keras

You have just found Keras.

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result*

extensibility).

binations of the two.

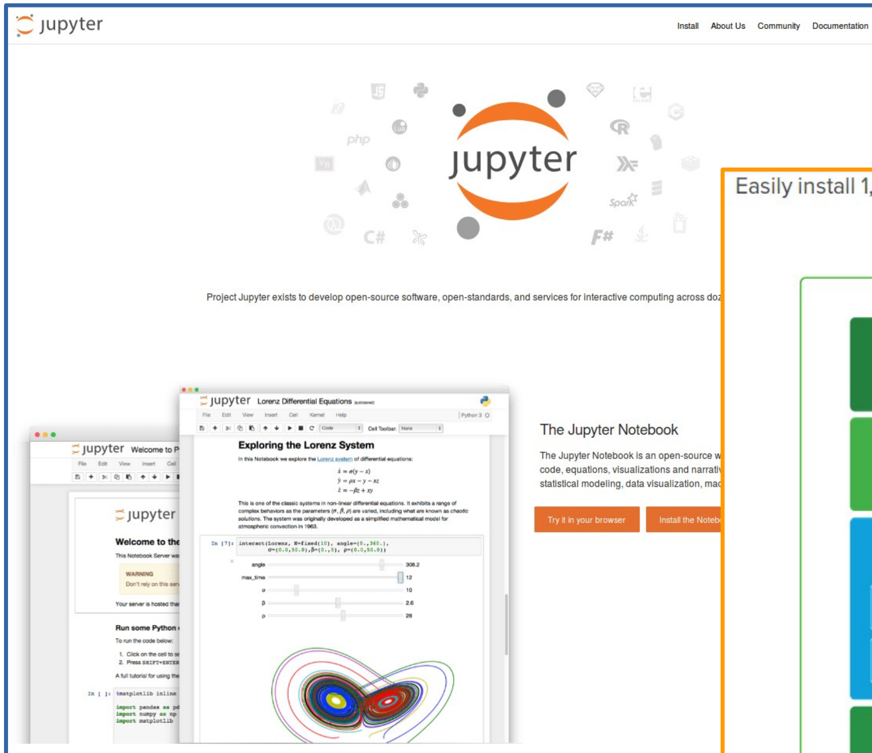
Pytorch.org

PyTorch   Get Started   Features

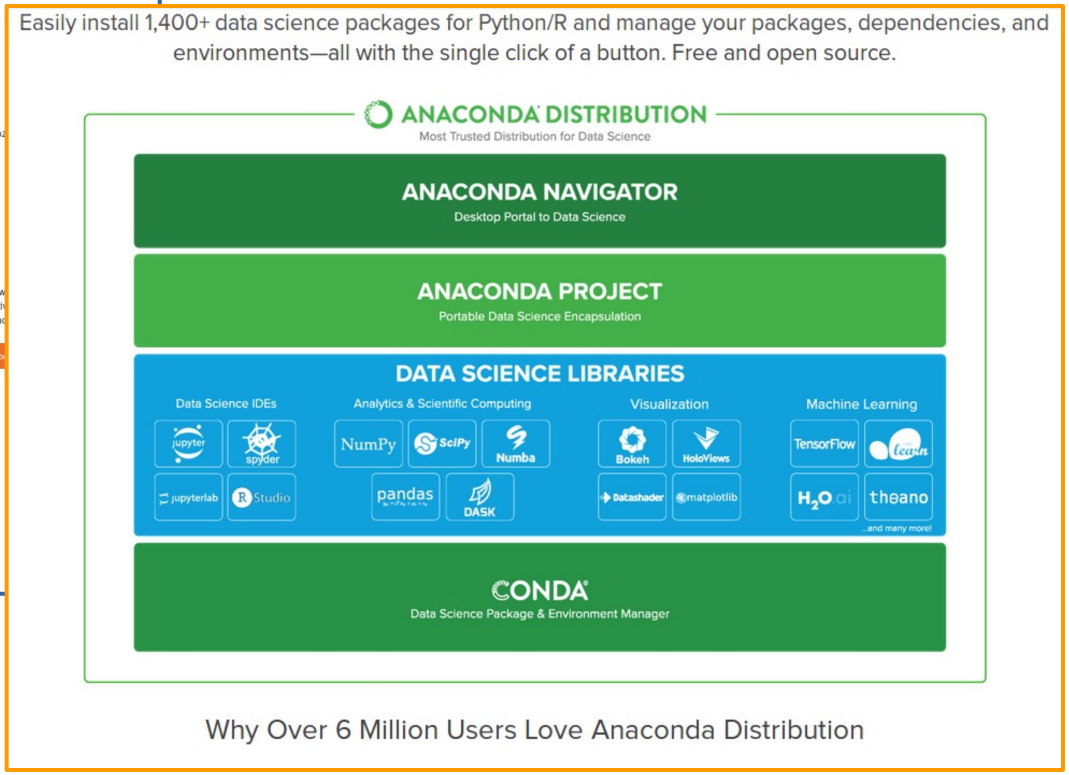
FROM RESEARCH TO PRODUCTION

An open source deep learning platform that provides a seamless path from research prototyping to production deployment.

Jupyter notebooks: jupyter.org



Install using Anaconda: www.anaconda.com



Google colab

https://colab.research.google.com/notebooks/intro.ipynb#recent=true

Resources

Lectures and exercices on ENT

https://ent.uca.fr/moodle/course/view.php?id=22704

Git hub

https://github.com/judonini/MLcourses

Practice sessions on local Linux computer cluster

Setup ML environment: conda activate mlearning

Summary

Machine Learning is based on statistics and computing (and lots of data !)

Not new (‘40s) but field in **exponential evolution** since ~10 years

Today: huge amount of **resources and tools** on the web

However real **understanding** of ML requires some efforts

Worth it because ML is evermore present in our work and life