Introduction

Introduction to Deep Learning

Learning outcomes

UNIVERSITÉ DU LUXEMBOURG

After this lesson you will be able to:

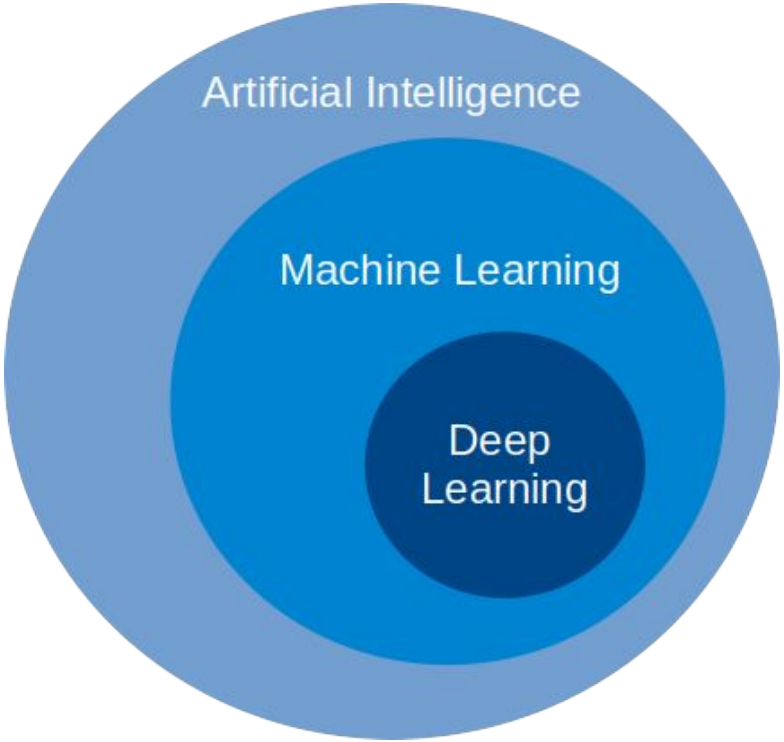
- Understand ML foundations
- Understand DL foundations
- Formulate problems from a ML/DL perspective

Introduction

“Machine Learning is a lot like teenage sex. Everybody talks about it. Only some really know how to do it. Everyone thinks everyone else is doing it. So, everyone claims they’re doing it.”

– https://medium.com/@peterx/dcdc08f17d17

Where we are



Artificial Intelligence

Try to create machines that can simulate human behavior

Machine Learning

Automatically learn from data without explicit programming

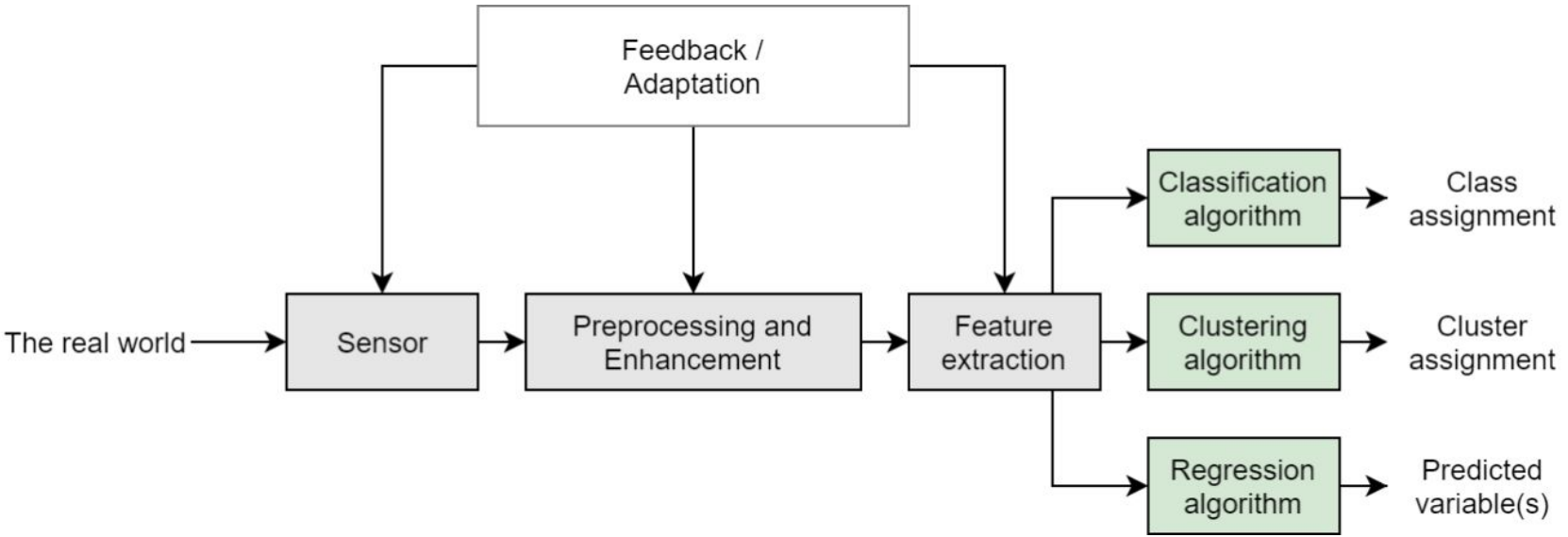
Deep Learning

Machine learning with neural networks

Machine Learning

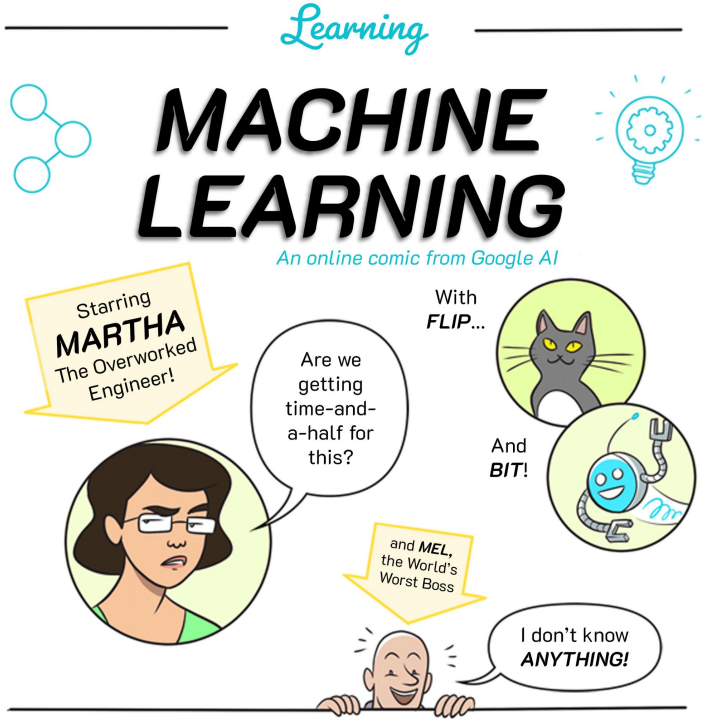
Pattern recognition and inference

Often guided by examples



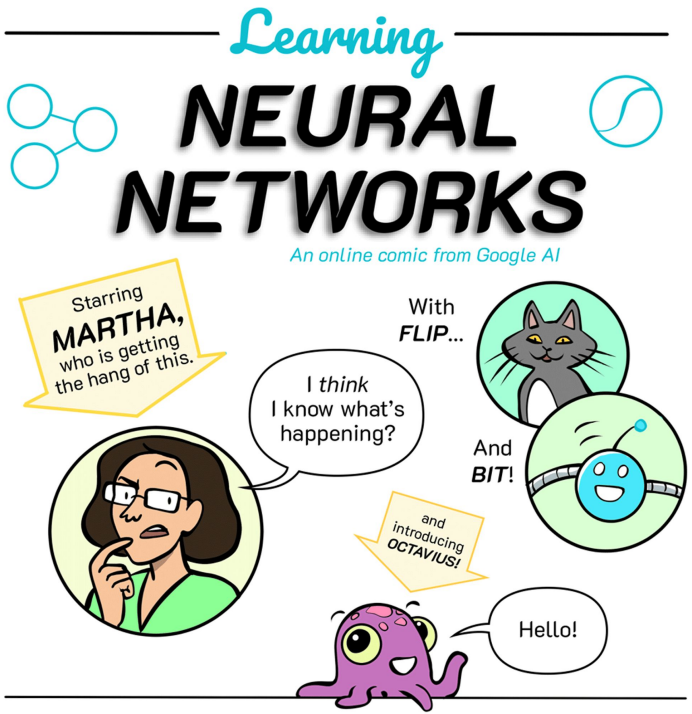
https://huspi.com/blog-open/pattern-recognition-in-machine-learning/

ML preliminaries



https://cloud.google.com/products/ai/ml-comic-1/

DL preliminaries



https://cloud.google.com/products/ai/ml-comic-2

Why is it called DL?

Input layer

One or more hidden layers

Output layer

https://xkcd.com/1838/

Why DL *now*?

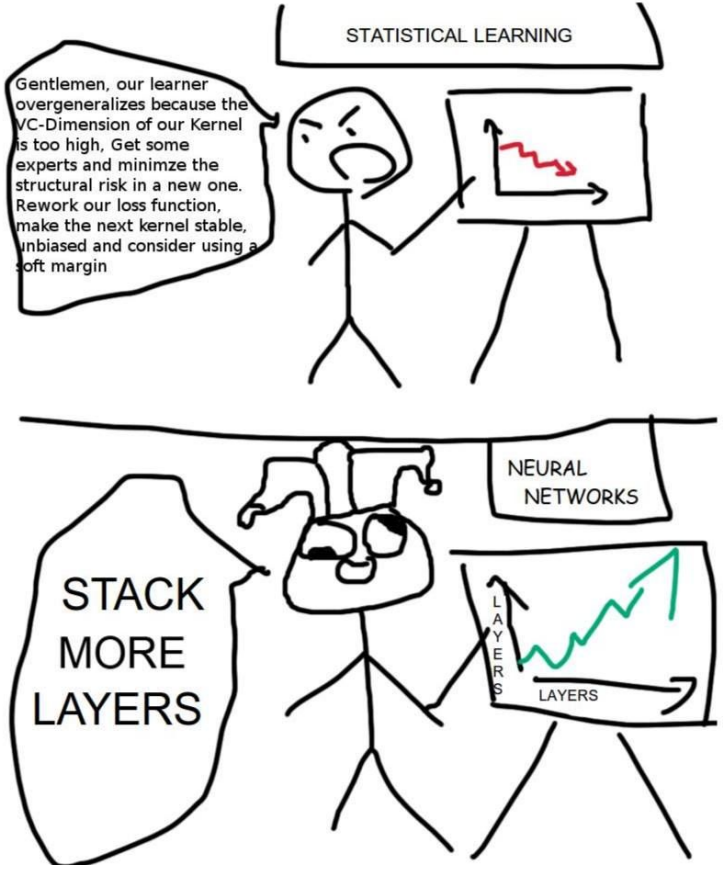
More hardware

TensorFlow
PYTORCH
theano
mxnet
Microsoft CNTK

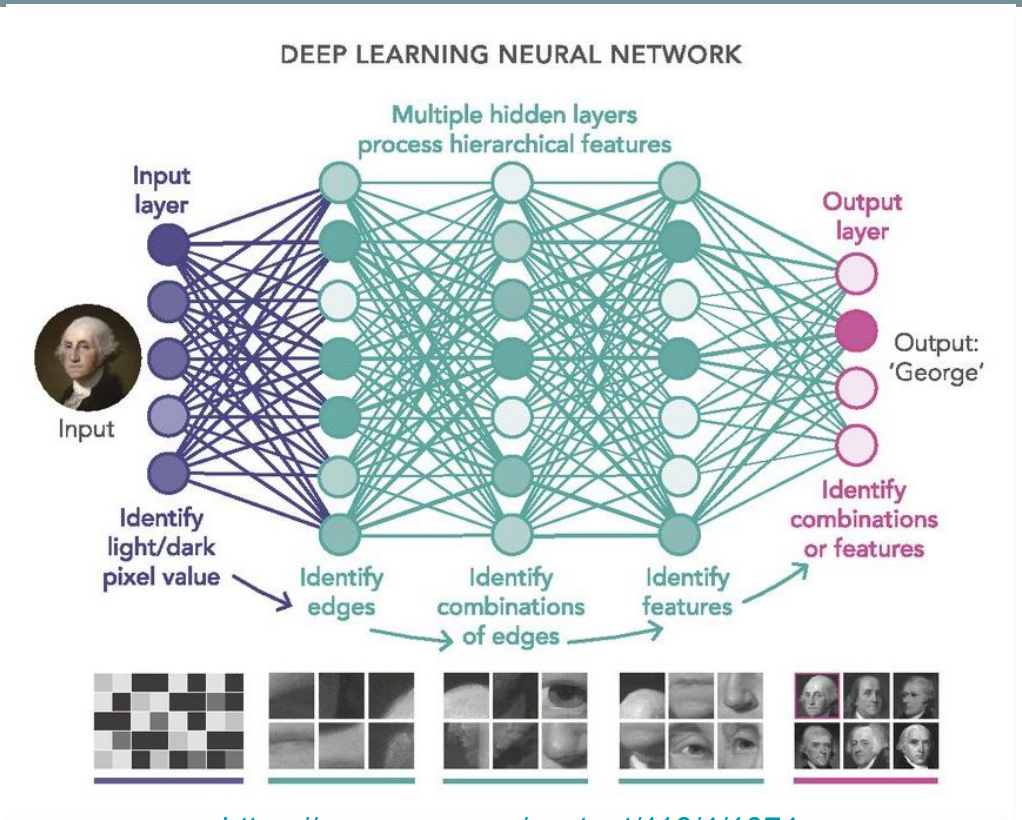
More software

More data

Why DL *now*?

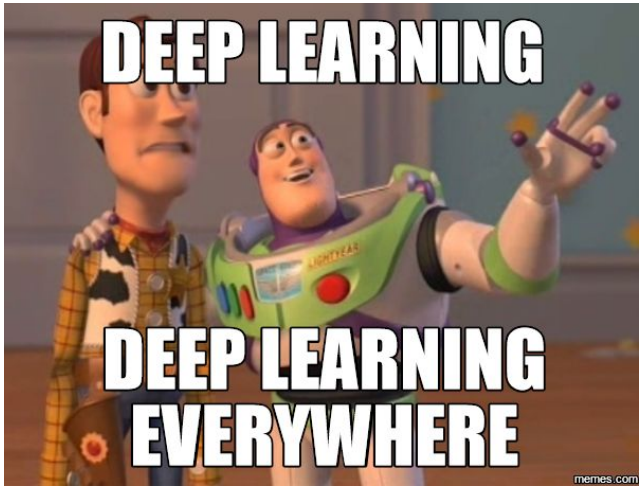

STATISTICAL LEARNING
Gentlemen, our learner overgeneralizes because the VC-Dimension of our Kernel is too high, Get some experts and minimze the structural risk in a new one. Rework our loss function, make the next kernel stable, unbiased and consider using a soft margin
NEURAL NETWORKS
STACK MORE LAYERS
LAYERS
LAYERS

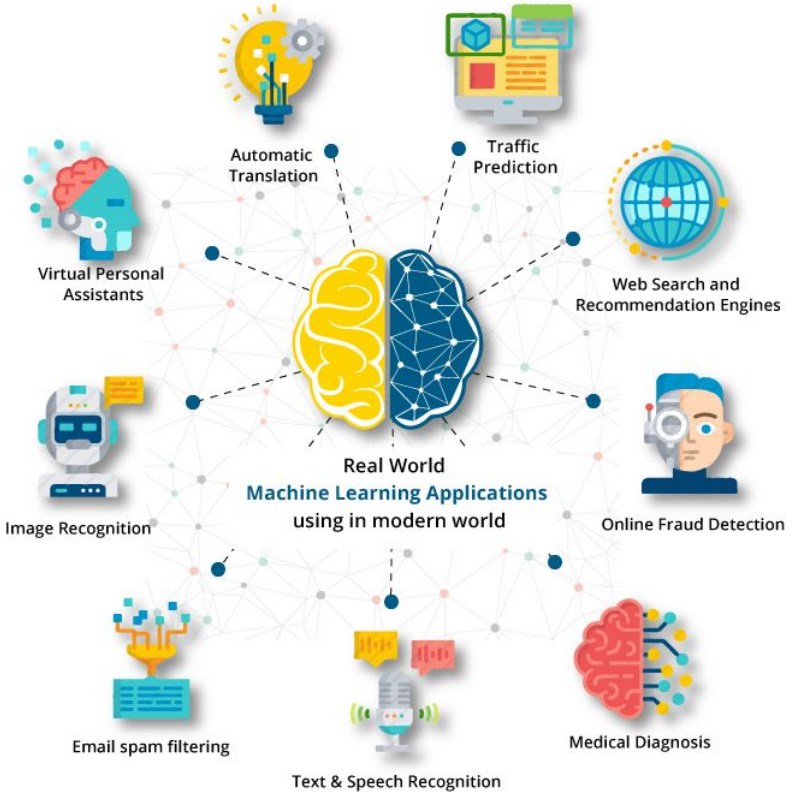
Example



https://www.pnas.org/content/116/4/1074

Applications





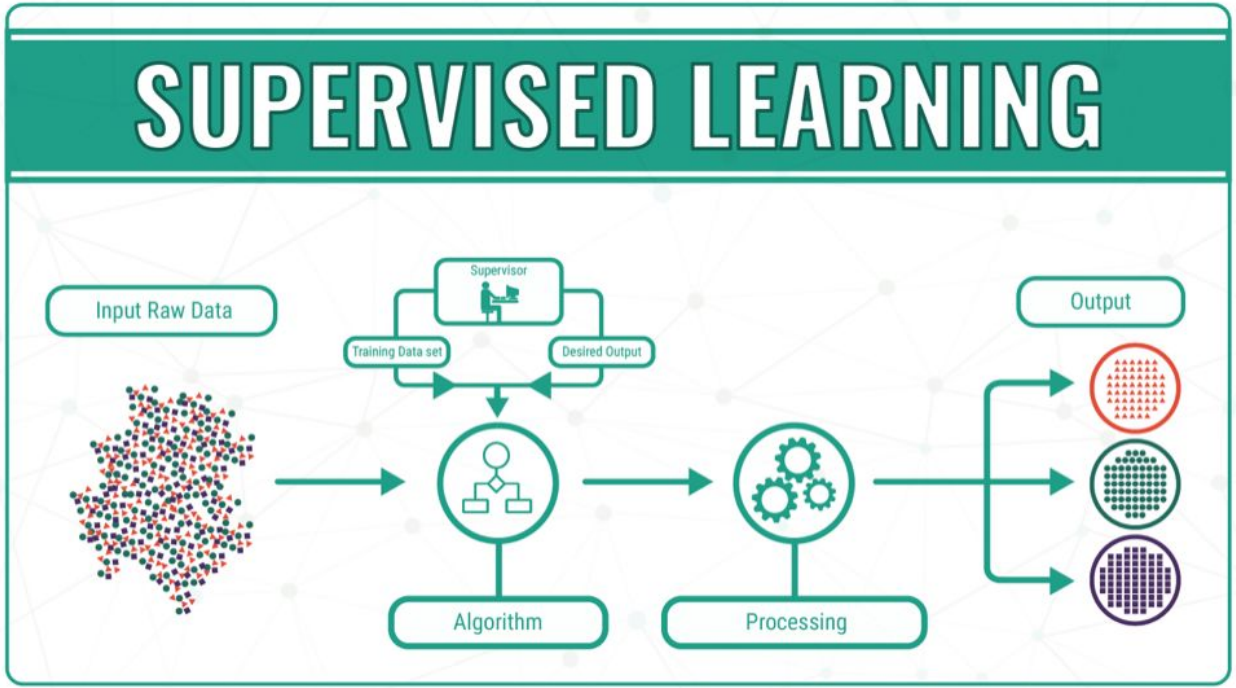
Learning paradigms

Examples:
Classification
Regression

Examples:
Autoencoders
Clustering

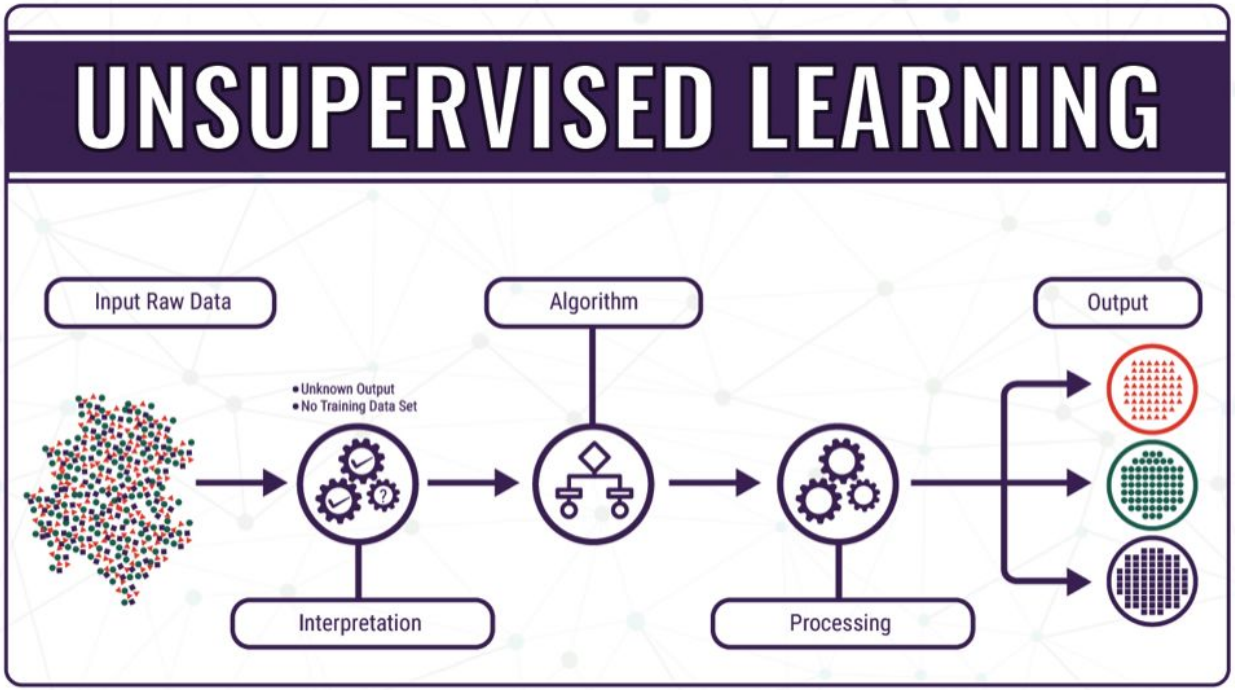
Examples:
Navigation
Planning

Learning paradigms



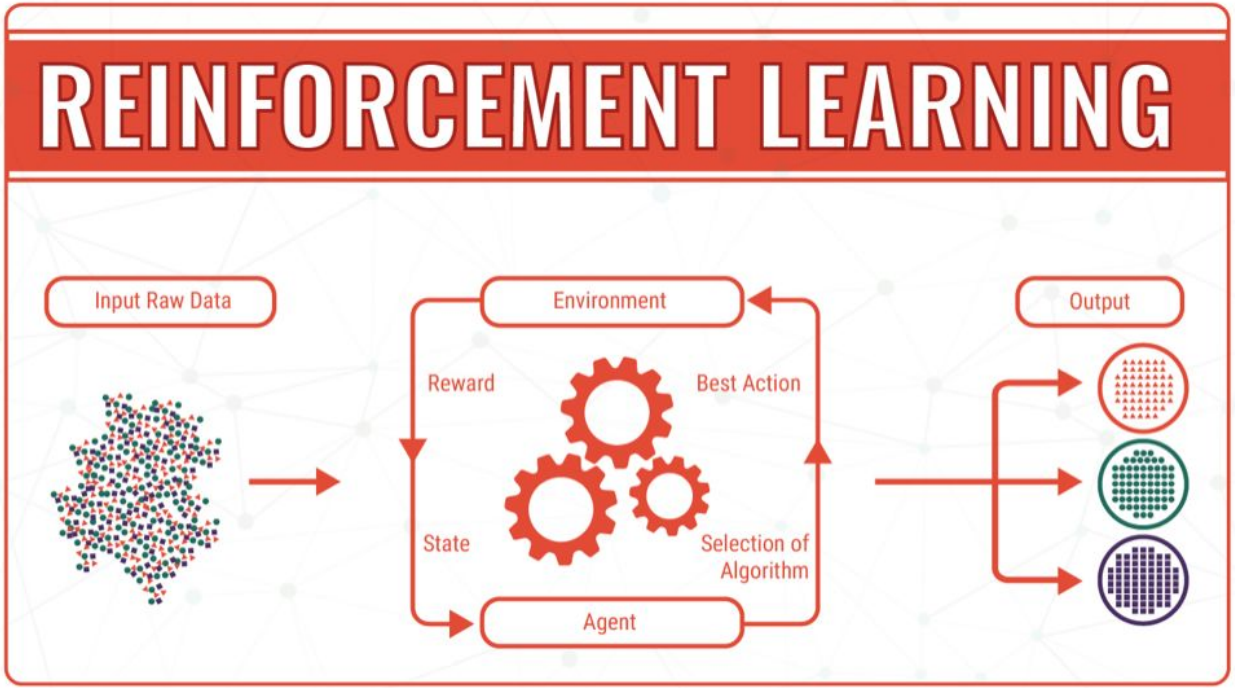
https://www.linkedin.com/pulse/machine-learning-explained-understanding-supervised-ronald-van-loon

Learning paradigms



https://www.linkedin.com/pulse/machine-learning-explained-understanding-supervised-ronald-van-loon

Learning paradigms



https://www.linkedin.com/pulse/machine-learning-explained-understanding-supervised-ronald-van-loon

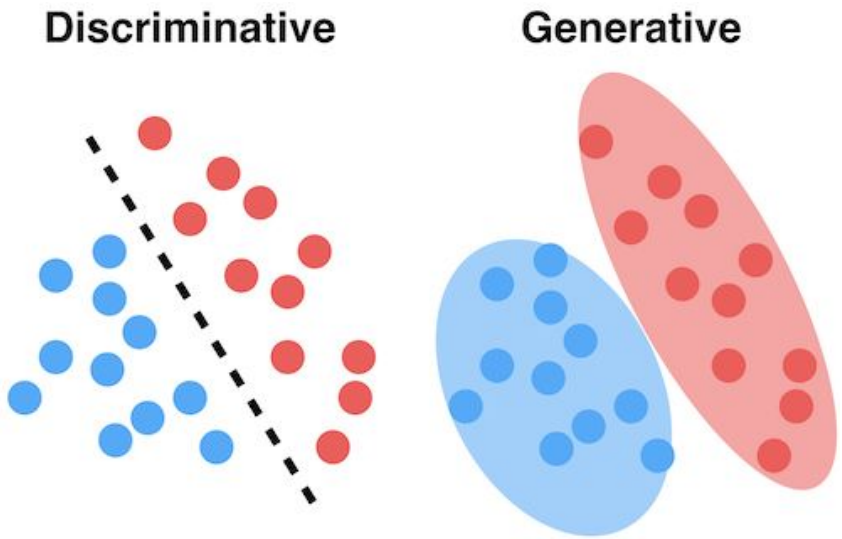
Modeling paradigms

Discriminative modeling

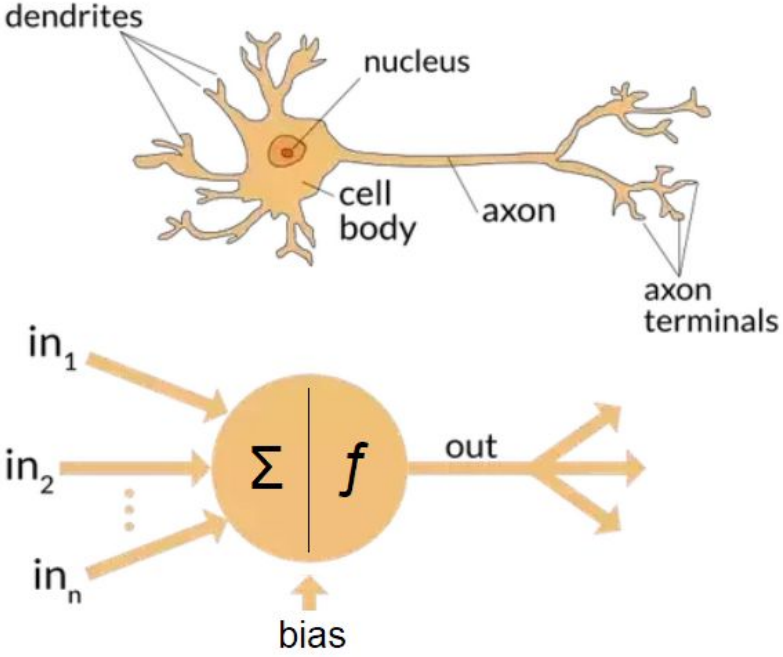
- Learn class boundaries
- Supervised learning

Generative modeling

- Learn class distributions
- Unsupervised learning

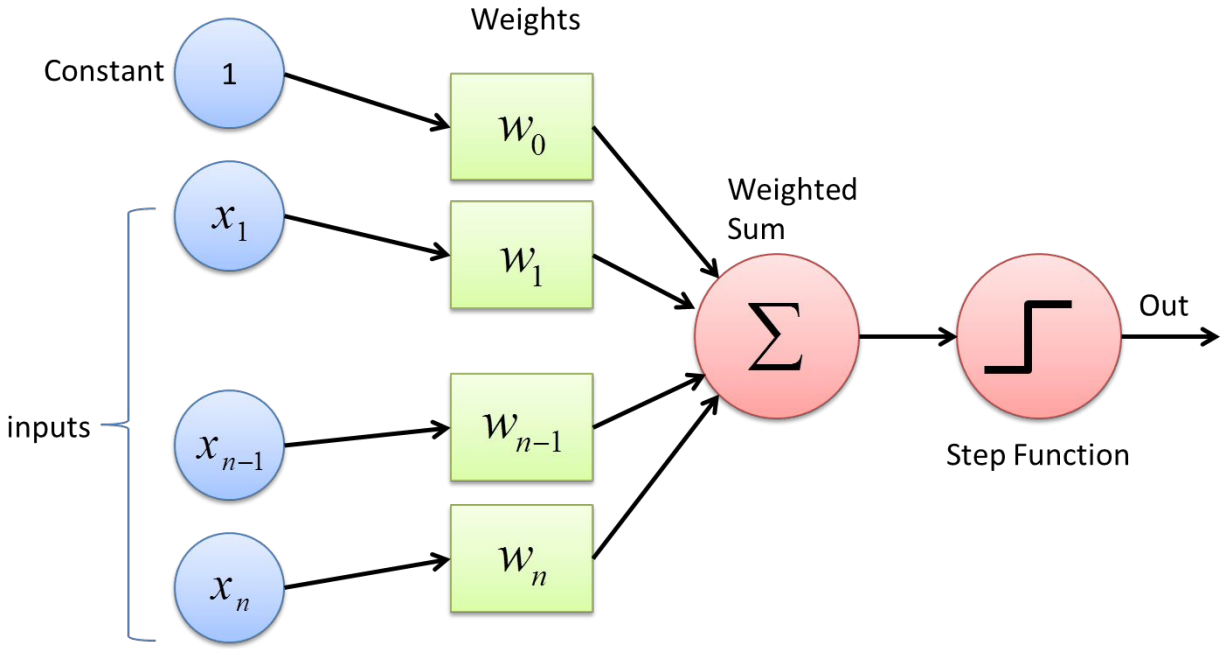


Perceptron algorithm



https://towardsdatascience.com/a8b46db828b7

Perceptron algorithm



https://towardsdatascience.com/626217814f53

Perceptron algorithm

UNIVERSITÉ DU LUXEMBOURG

$$\hat{y} = f(z) = \begin{cases} 1 & w_0 + \sum_{i=1}^{n} x_i w_i > \theta \\ 0 & \text{otherwise} \end{cases}$$

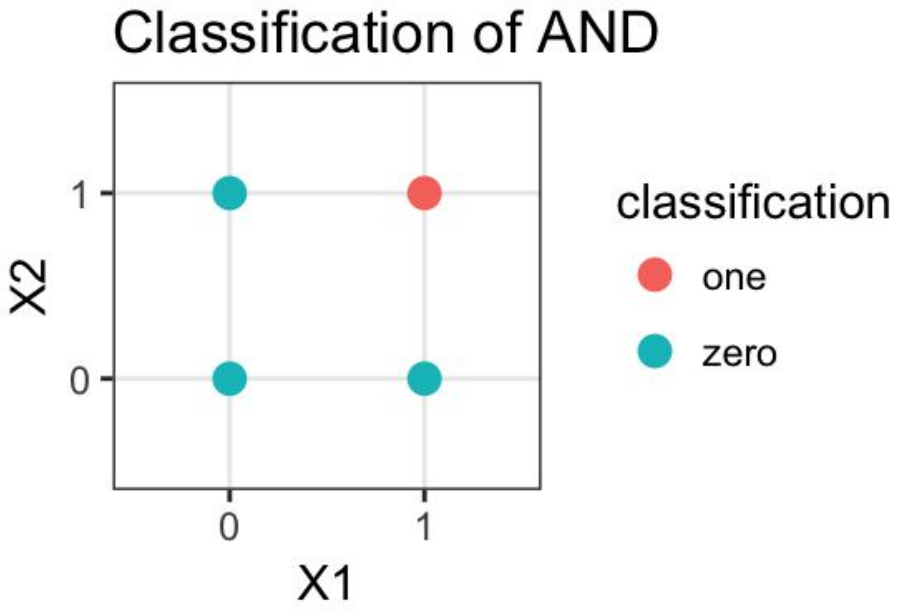
$$\hat{y} = f(z) = \begin{cases} 1 & \sum_{i=0}^{n} x_i w_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

Perceptron algorithm

UNIVERSITÉ DU LUXEMBOURG

1. Initialize weights $\mathbf{w}$
2. For N iterations (or until convergence) do:
   2.1. Process each instance $f(z) = f(\mathbf{w}^T \mathbf{x})$
   2.2. Update weights: w_i += [y - $f(z)$] x_i
3. Return weights

Perceptron algorithm



x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

Perceptron algorithm

Let $y = f(b, x_1, x_2)$ and $\mathbf{w} = (0,0,0)$.

Train $\mathbf{x} = (1,1,1)$; $y = 1$

$z = \mathbf{x} \cdot \mathbf{w}^T = 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 0 = 0 \rightarrow f(z) = 0$

$\mathbf{w} = (0,0,0) + (1 - 0) \cdot (1,1,1) = (1,1,1)$

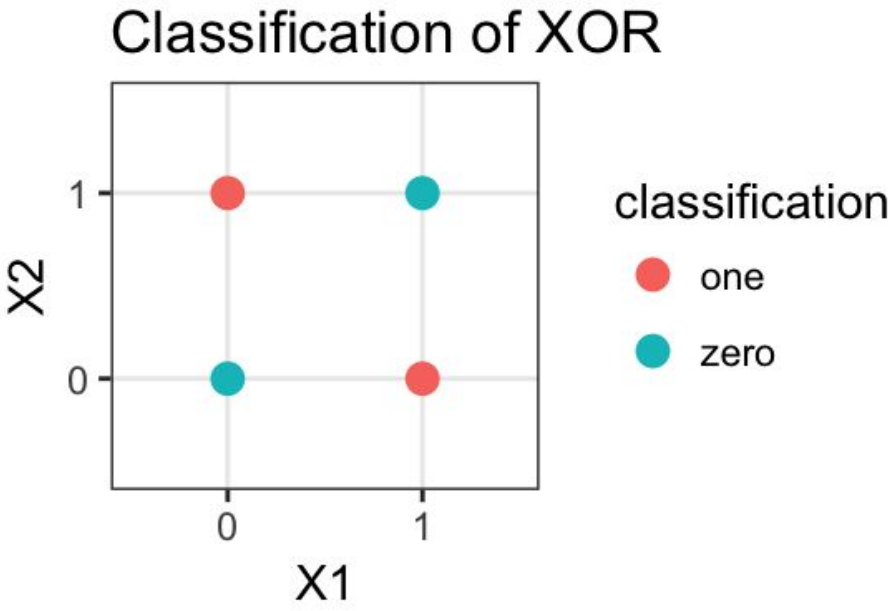
Train $\mathbf{x} = (1,1,0)$; $y = 0$

$z = \mathbf{x} \cdot \mathbf{w}^T = 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 1 > 0 \rightarrow f(z) = 1$

$\mathbf{w} = (1,1,1) + (0 - 1) \cdot (1,1,0) = (0,0,1)$

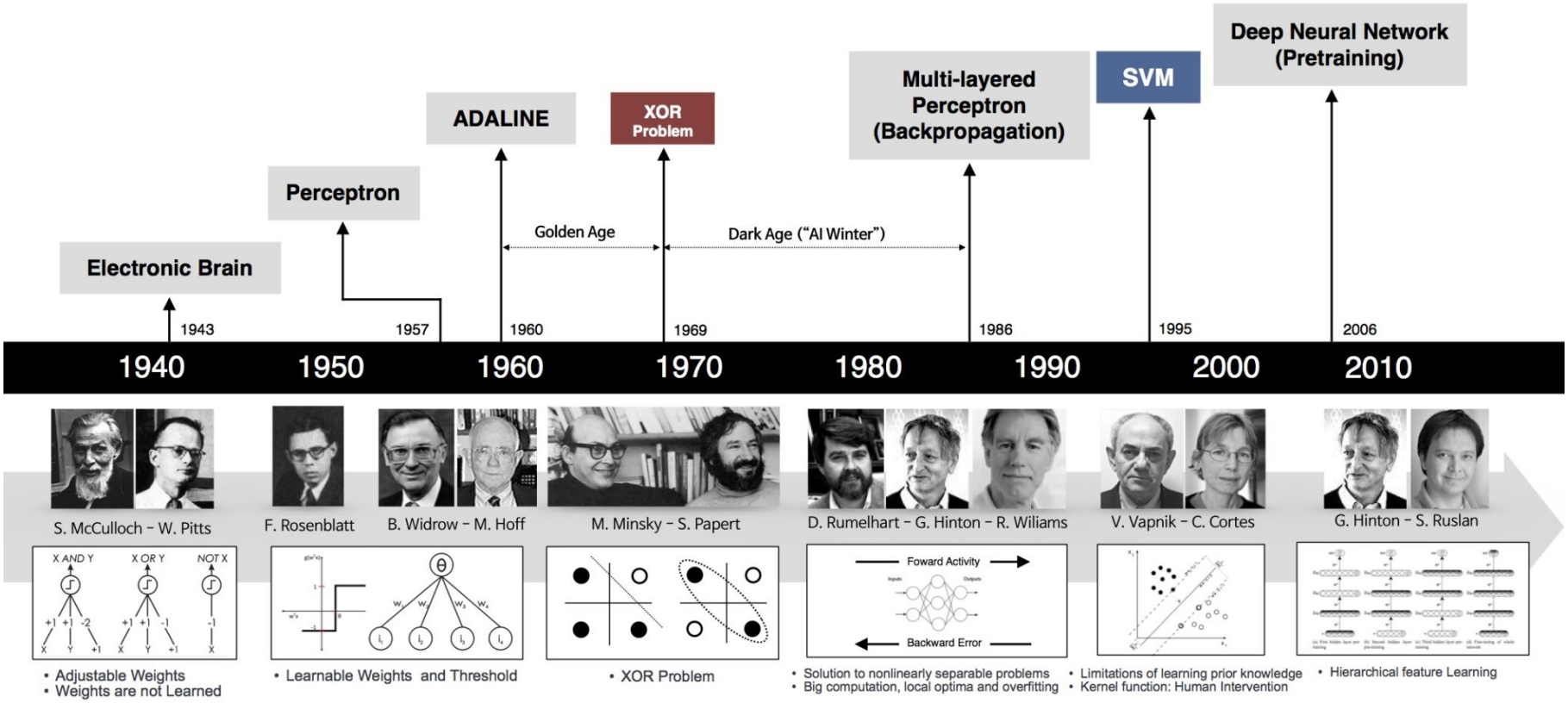
x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

Perceptron algorithm

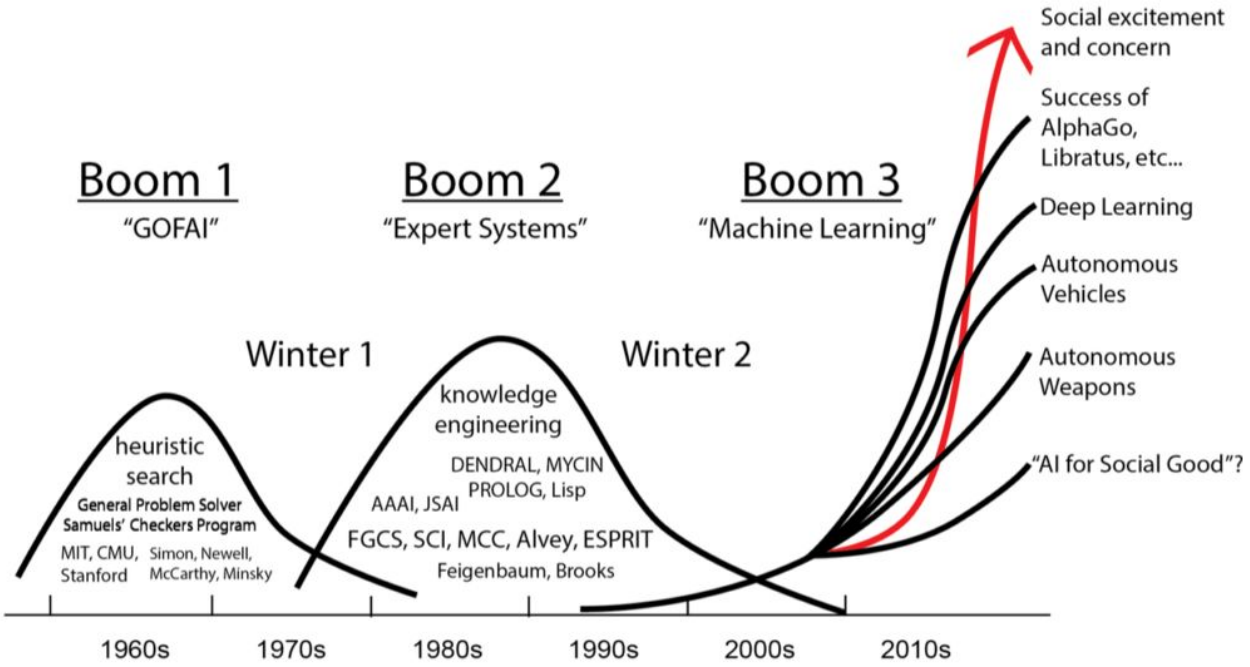


x_1	x_2	y
1	1	0
1	0	1
0	1	1
0	0	0

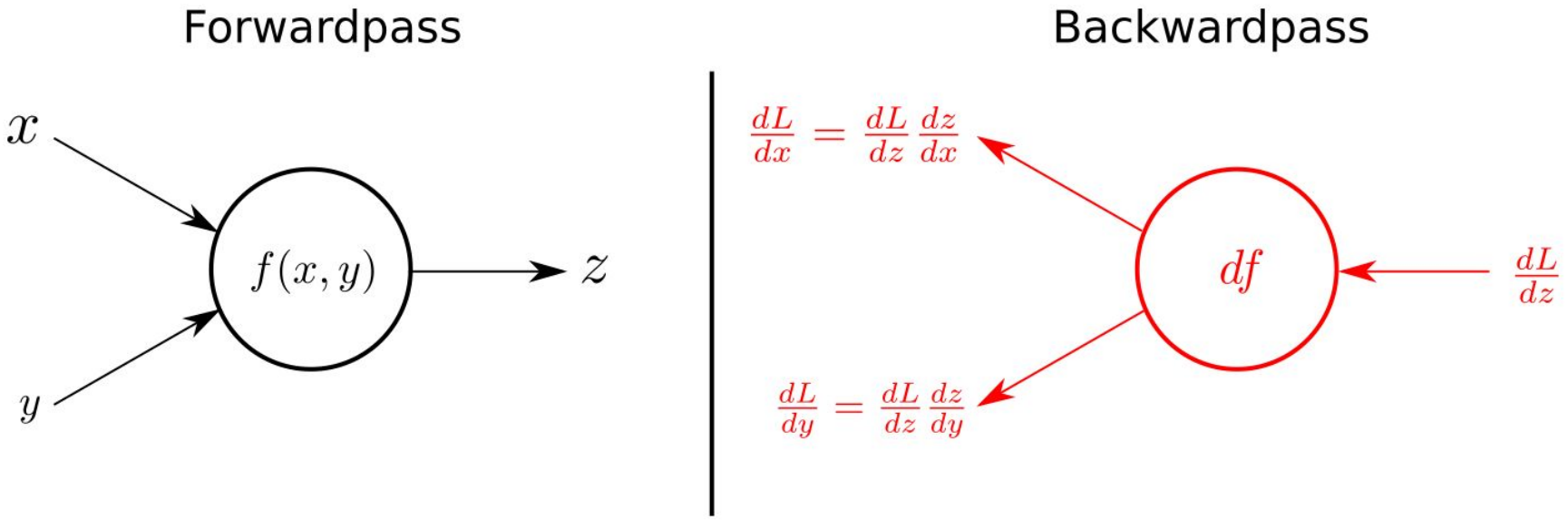
Timeline


Deep Neural Network (Pretraining)
SVM
Multi-layered Perceptron (Backpropagation)
XOR Problem
ADALINE
Perceptron
Electronic Brain
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
X AND Y
X OR Y
NOT X
Foward Activity
Backward Error
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

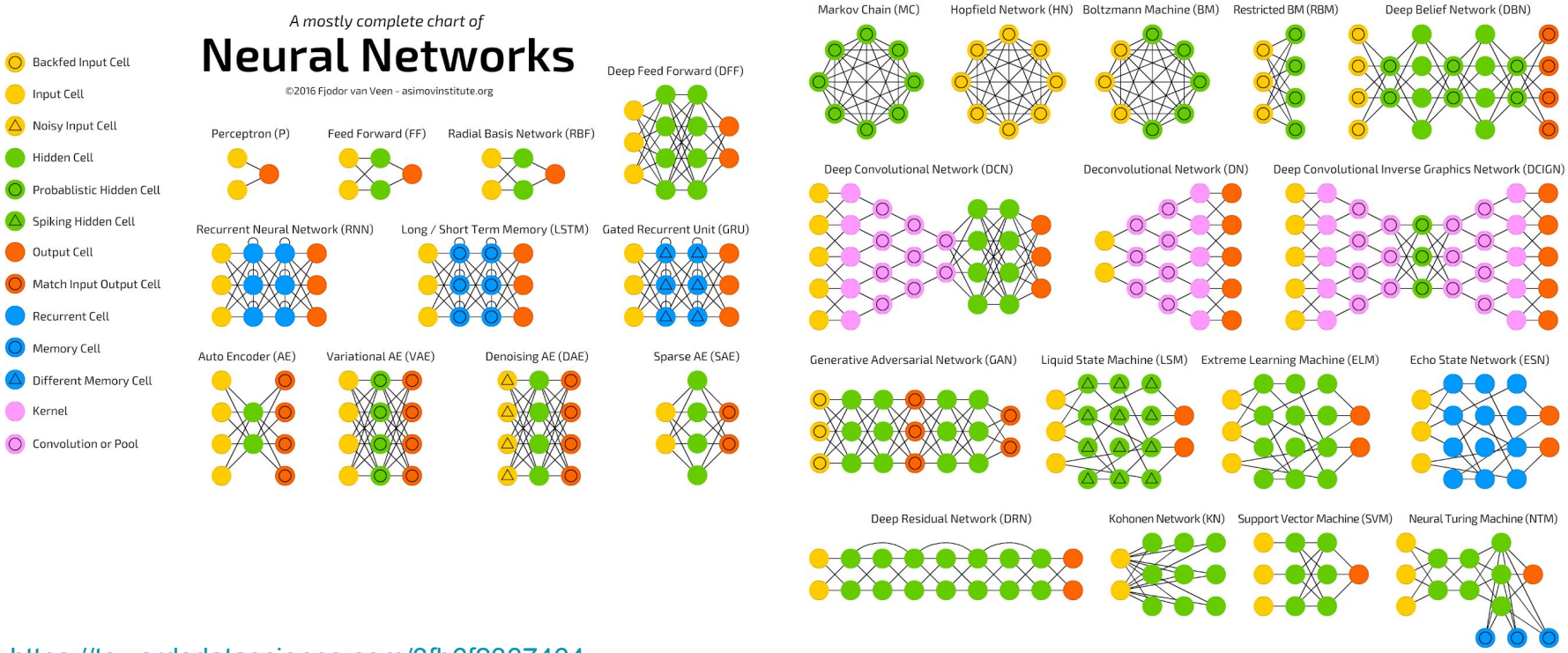
AI winters


Boom 1
"GOFAI"
Boom 2
"Expert Systems"
Boom 3
"Machine Learning"
Winter 1
Winter 2
heuristic search
General Problem Solver
Samuels' Checkers Program
MIT, CMU, Stanford
Simon, Newell, McCarthy, Minsky
knowledge engineering
DENDRAL, MYCIN
PROLOG, Lisp
AAAI, JSAI
FGCS, SCI, MCC, Alvey, ESPRIT
Feigenbaum, Brooks
Social excitement and concern
Success of AlphaGo, Libratus, etc...
Deep Learning
Autonomous Vehicles
Autonomous Weapons
"AI for Social Good"?
1960s
1970s
1980s
1990s
2000s
2010s

Backprop

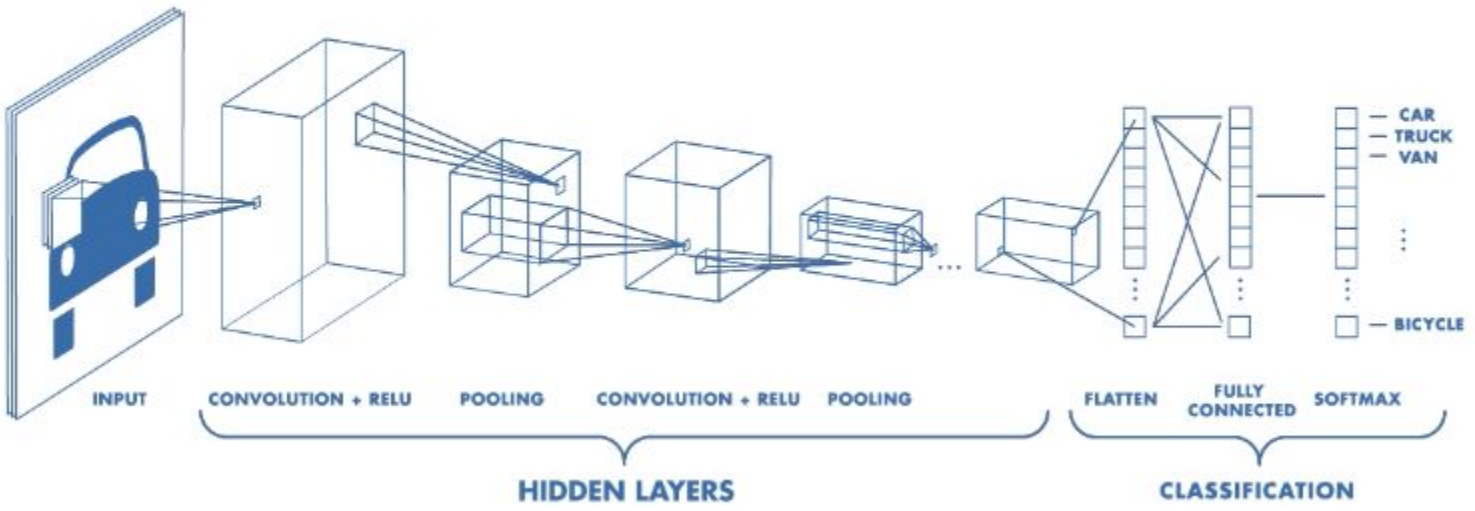

Forwardpass
Backwardpass
x
y
$f(x, y)$
z
$\frac{dL}{dx} = \frac{dL}{dz}\frac{dz}{dx}$
$\frac{dL}{dy} = \frac{dL}{dz}\frac{dz}{dy}$
df
$\frac{dL}{dz}$

Some architectures



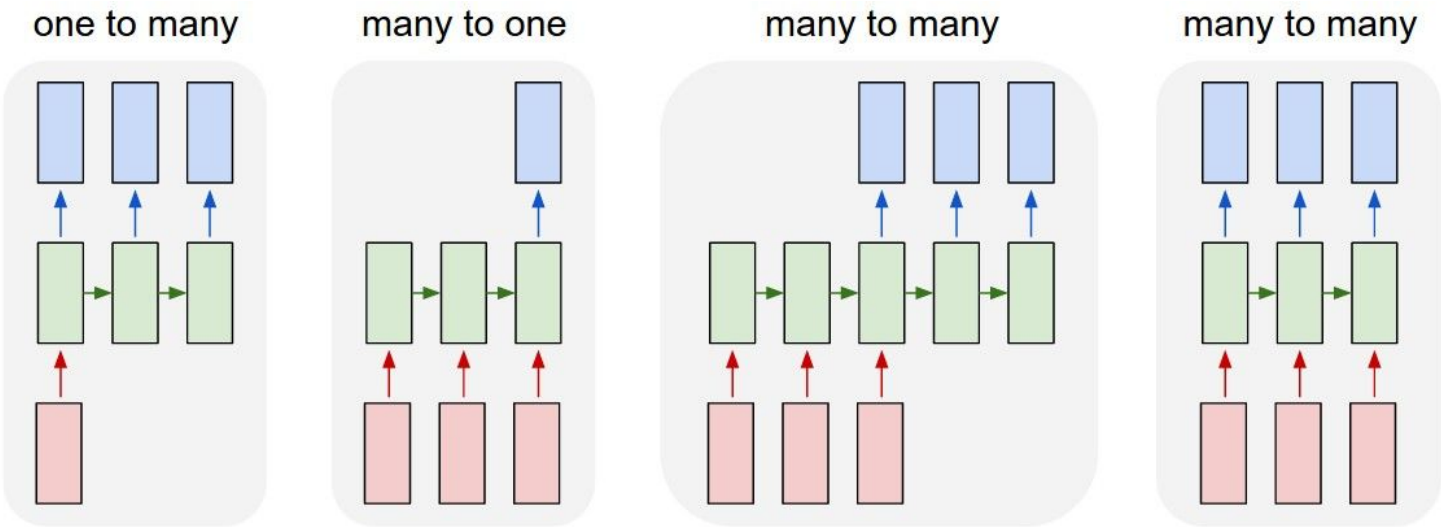
https://towardsdatascience.com/3fb6f2367464

Convolutional Neural Nets



https://towardsdatascience.com/3bd2b1164a53

Recurrent Neural Nets



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Building blocks

Layers

Activation functions

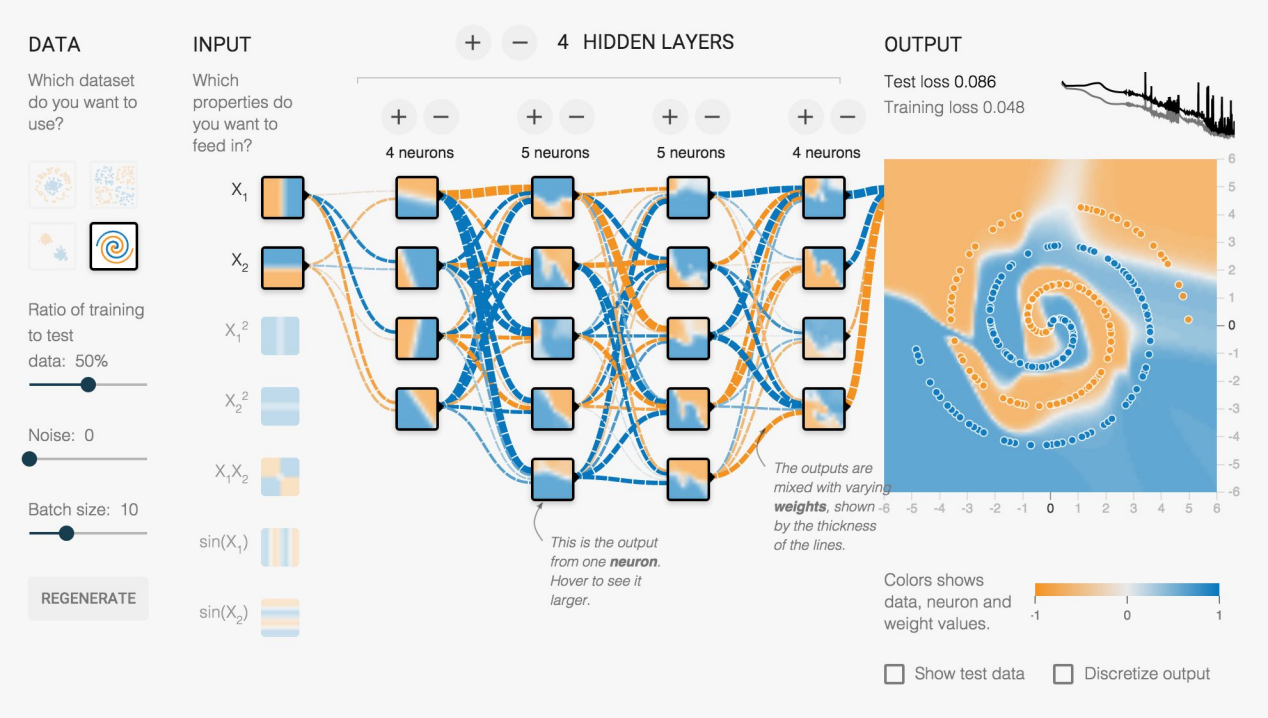
Loss functions

Optimizers

Regularizers

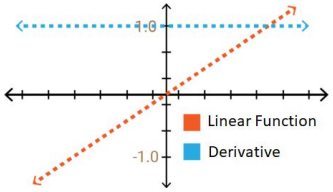
Evaluation metrics

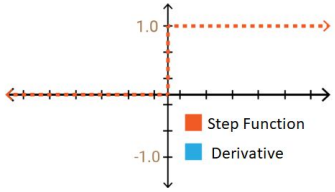
Layers

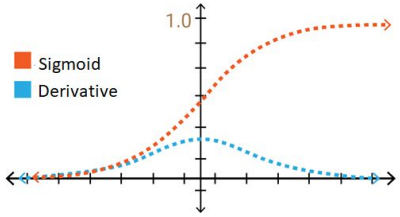


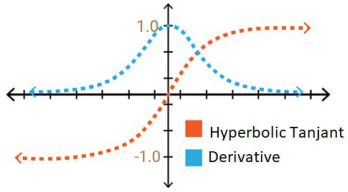
https://blogs.scientificamerican.com/sa-visual/unveiling-the-hidden-layers-of-deep-learning/

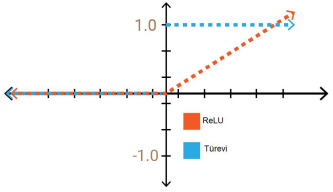
Activation functions

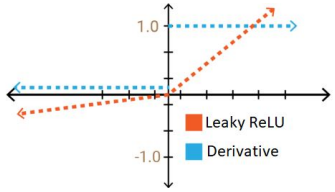


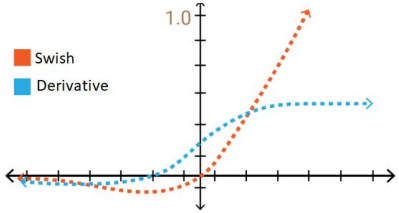










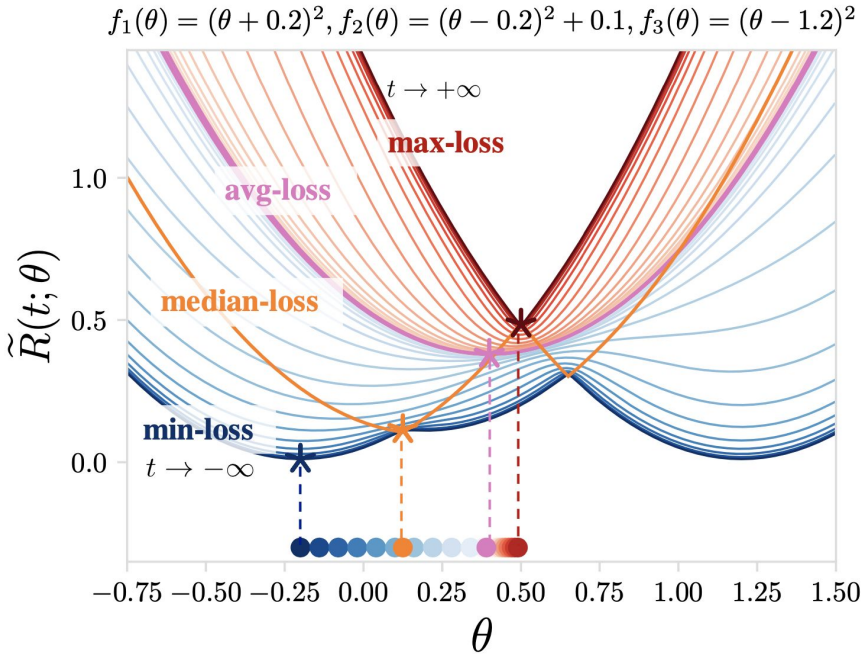


https://towardsdatascience.com/706ac4284c8a and https://en.wikipedia.org/wiki/Activation_function

Loss functions

Goal: Minimize empirical loss

$$J(\mathbf{W}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(\hat{y}_i, y_i)$$



https://pythonawesome.com/tilted-empirical-risk-minimization-in-python/

Loss functions for **classification** tasks

Binary Crossentropy loss for *C* = 2 classes:

$$\mathcal{L}(\widehat{y_i}, y_i) = -\frac{1}{N}\sum_i^N y_i \log \widehat{y_i} + (1 - y_i) \log (1 - \widehat{y_i})$$

Categorical Crossentropy loss for *C* > 2 classes:

$$\mathcal{L}(\widehat{y_i}, y_i) = -\frac{1}{N}\sum_i^N \sum_c^C \mathbb{1}_{y_i \in C_c} \log \widehat{y_i}$$

Note: $\widehat{y_i} = p(y_i)$

Loss functions for **regression** tasks

UNIVERSITÉ DU LUXEMBOURG

Mean Squared Error loss:

$$\mathcal{L}(\hat{y}_i, y_i) = (y_i - \hat{y}_i)^2$$

Mean Absolute Error loss:

$$\mathcal{L}(\hat{y}_i, y_i) = |y_i - \hat{y}_i|$$

Notes:

RMSE = √MSE

MAE ≤ RMSE

RMSE ≤ MAE · √n

Loss functions for **regression** tasks

UNIVERSITÉ DU LUXEMBOURG

CASE 1: Evenly distributed errors

ID	Error	|Error|	Error^2
1	2	2	4
2	2	2	4
3	2	2	4
4	2	2	4
5	2	2	4
6	2	2	4
7	2	2	4
8	2	2	4
9	2	2	4
10	2	2	4

MAE	RMSE
2.000	2.000

CASE 2: Small variance in errors

ID	Error	|Error|	Error^2
1	1	1	1
2	1	1	1
3	1	1	1
4	1	1	1
5	1	1	1
6	3	3	9
7	3	3	9
8	3	3	9
9	3	3	9
10	3	3	9

MAE	RMSE
2.000	2.236

CASE 3: Large error outlier

ID	Error	|Error|	Error^2
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	20	20	400

MAE	RMSE
2.000	6.325

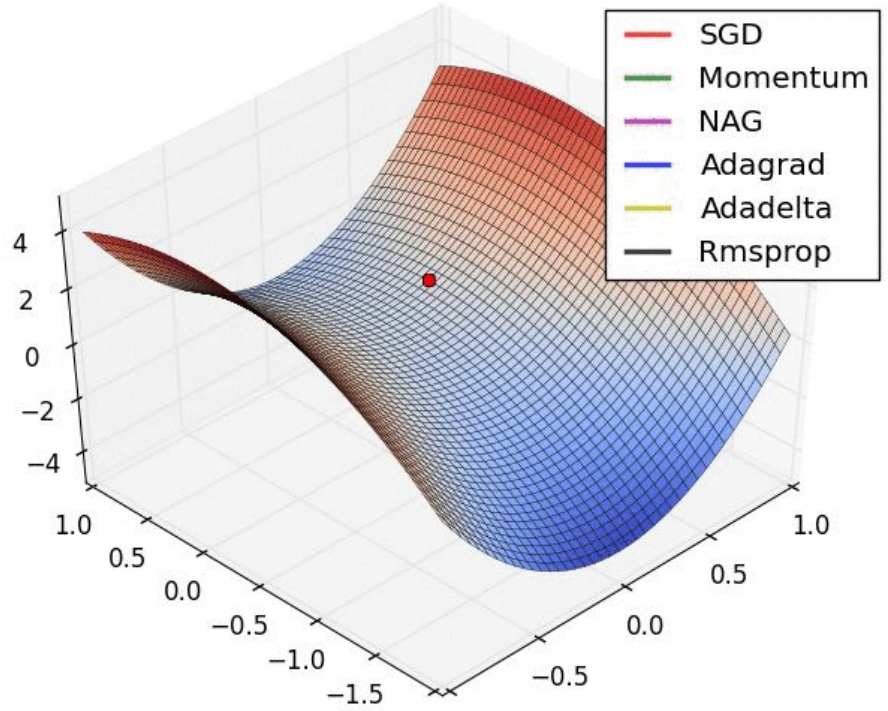
https://medium.com/human-in-a-machine-world/e60ac3bde13d

Optimizers

Mostly based on Gradient Descent:

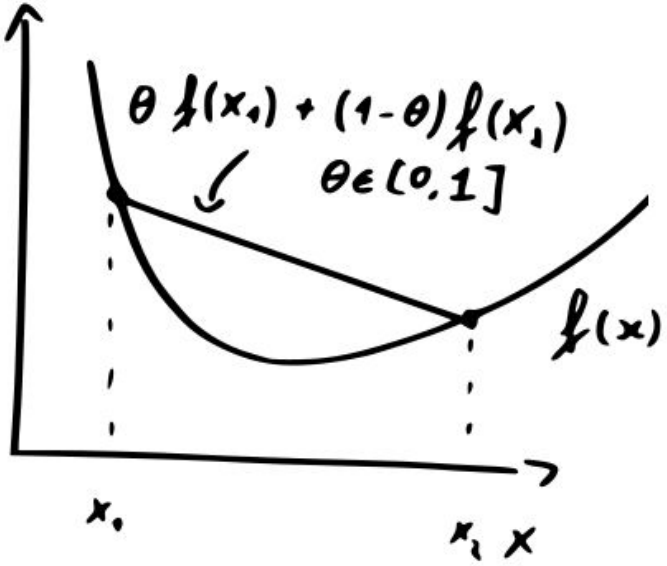
$\theta = \theta - \eta \cdot \nabla J(\mathbf{W}; \theta)$

- SGD
- RMSProp
- Adam
- AdaGrad
- AdaDelta

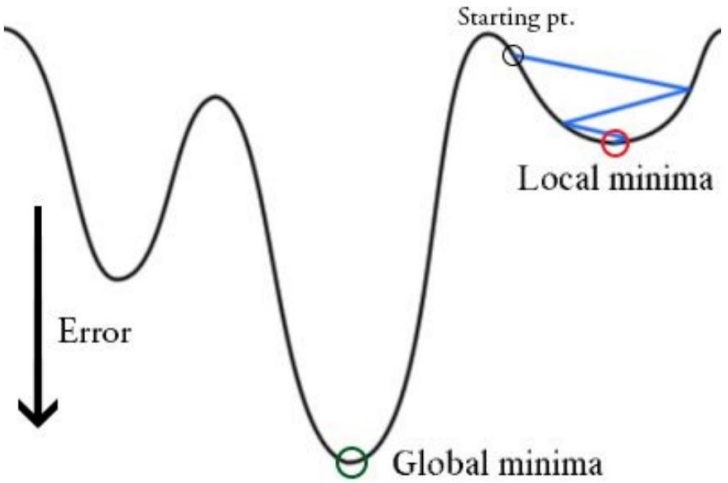


https://ruder.io/optimizing-gradient-descent/

Non-convex optimization



https://michielstock.github.io/posts/2018/2018-03-07-ConvexSummary/



https://www.cs.ubc.ca/labs/lci/mlrg/slides/non_convex_optimization.pdf

Regularizers

UNIVERSITÉ DU LUXEMBOURG

Goal: Improve model generalization

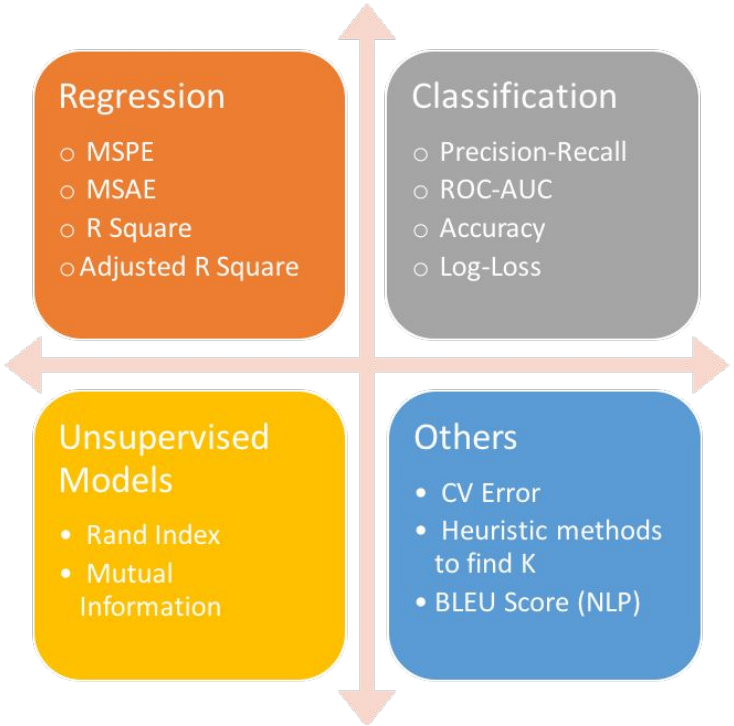
- Dropout*
- BatchNorm
- Early stopping*
- L1 & L2 penalty
- Adaptive learning rate*
- Data augmentation

* Very very useful, really

Note: Double Descent

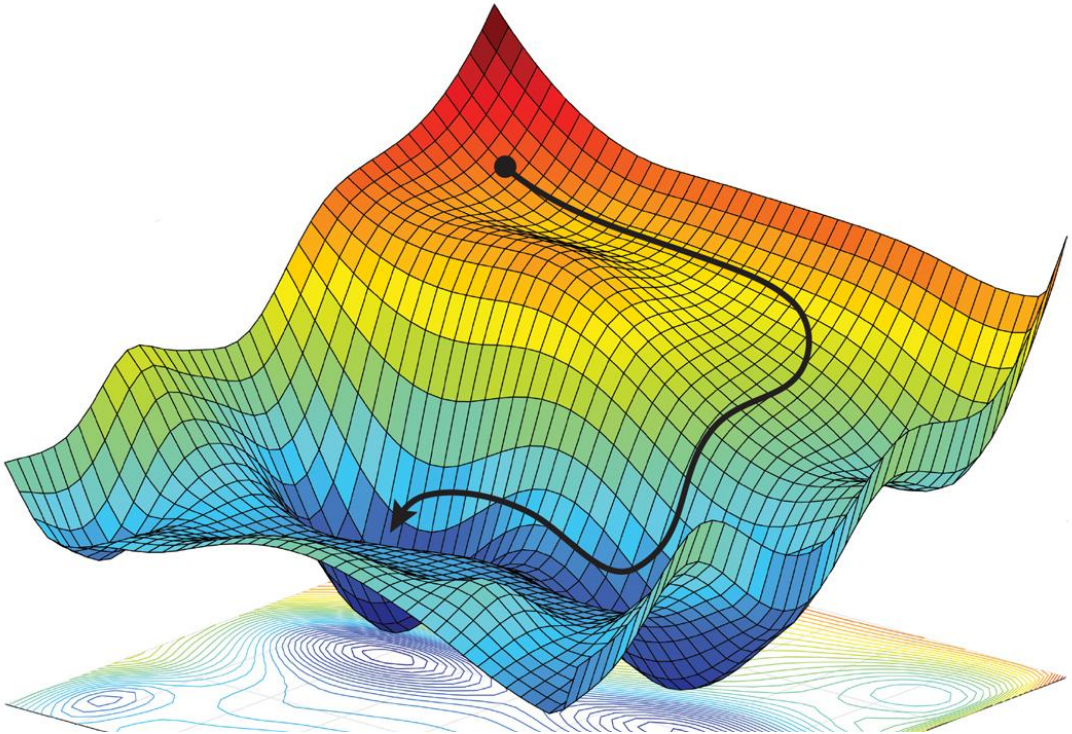
https://openai.com/blog/deep-double-descent/

Evaluation metrics



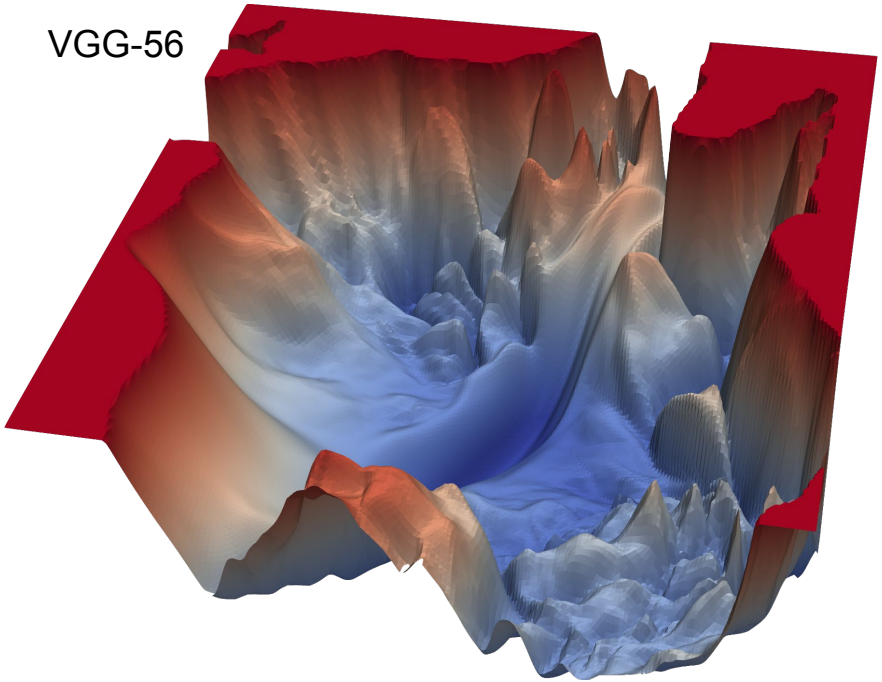
https://medium.com/usf-msds/a99d7d7414e4

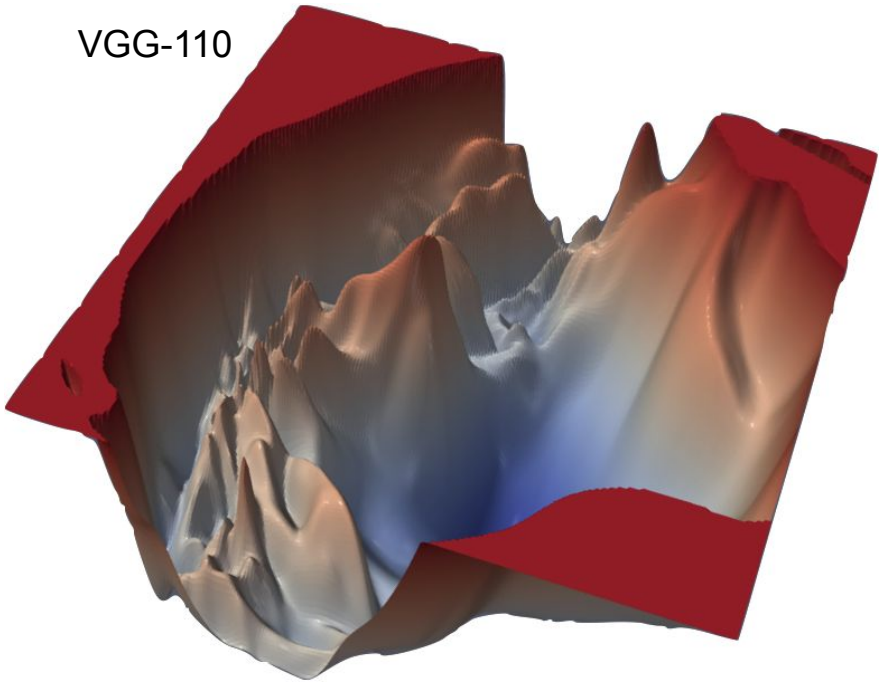
DL in practice



https://www.science.org/content/article/ai-researchers-allege-machine-learning-alchemy

Loss landscapes





https://www.cs.umd.edu/~tomg/projects/landscapes/

Generalization

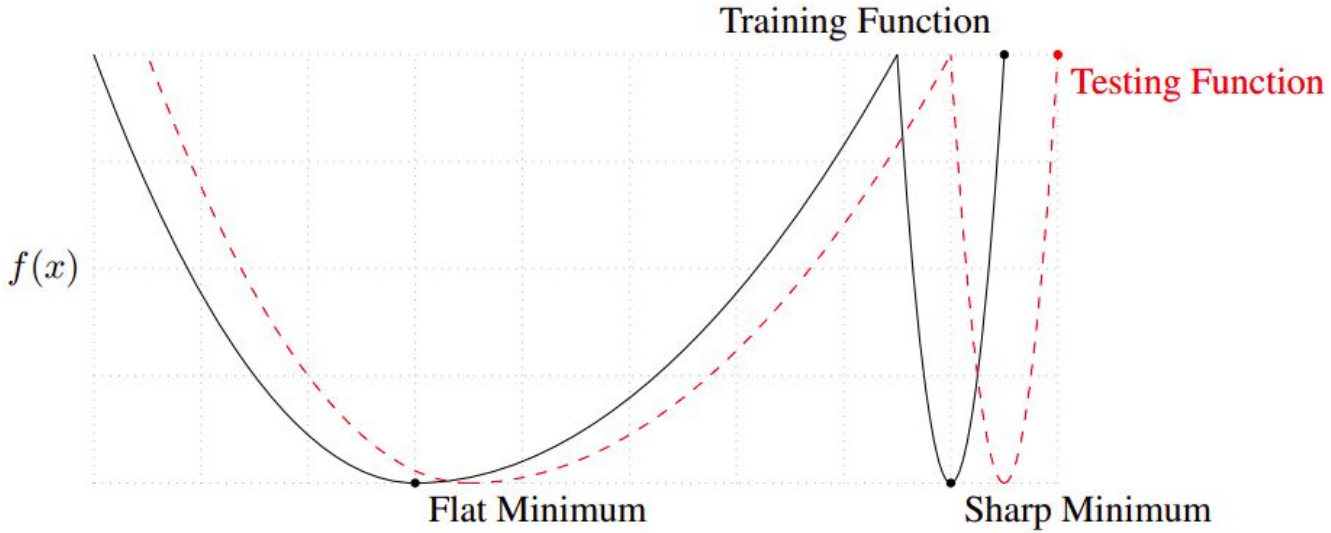
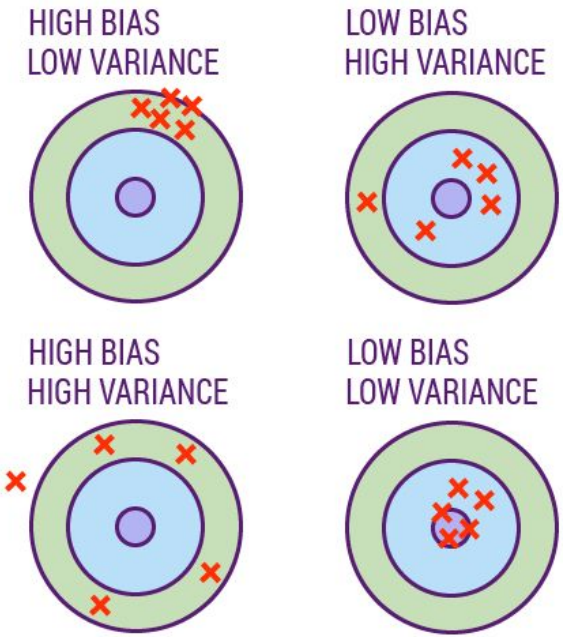
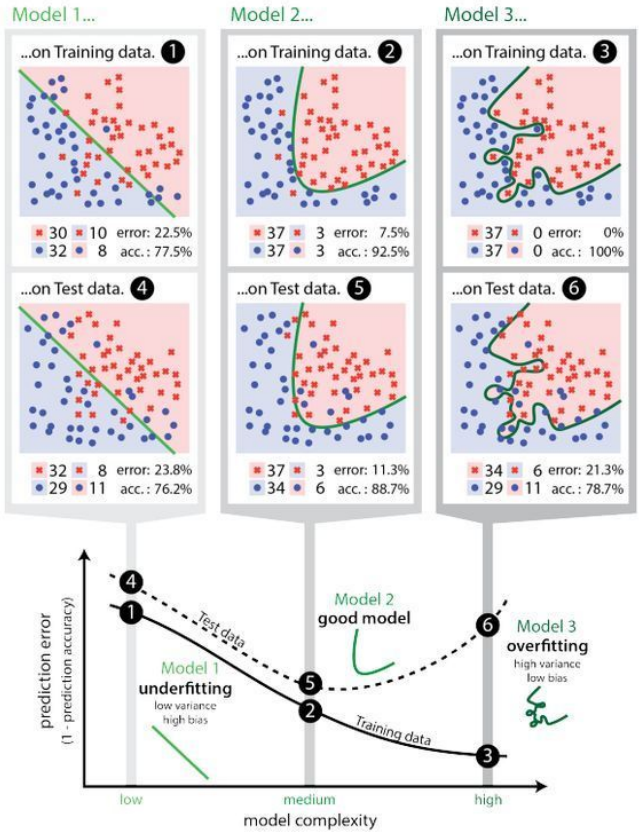


Figure 1: A Conceptual Sketch of Flat and Sharp Minima. The Y-axis indicates value of the loss function and the X-axis the variables (parameters)

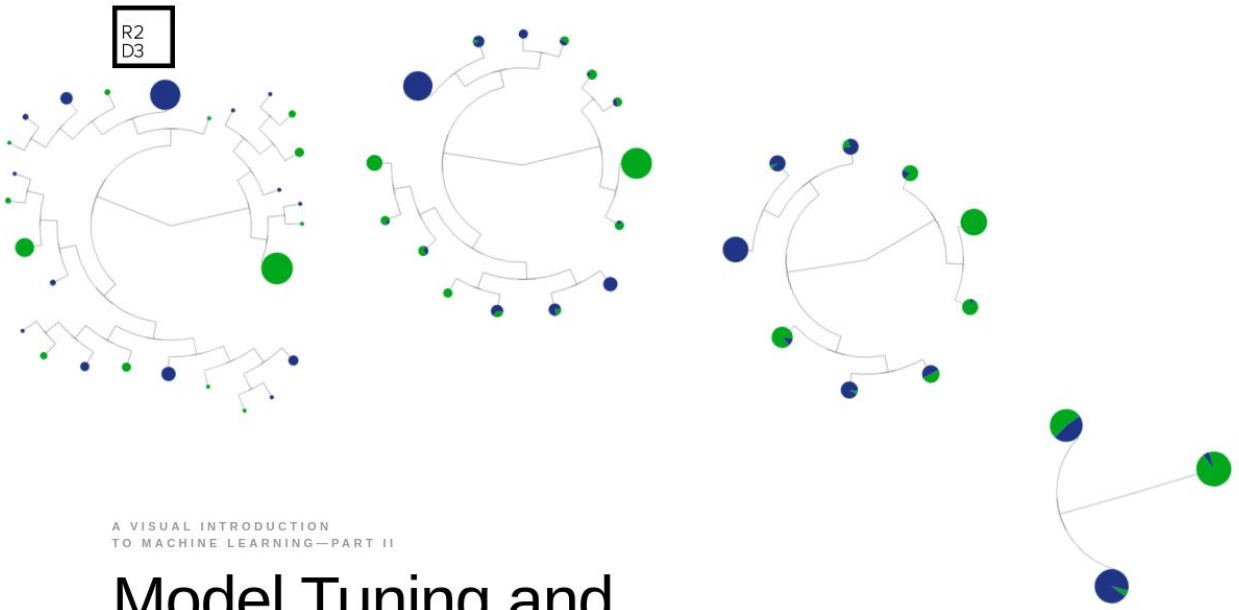
https://arxiv.org/pdf/1609.04836.pdf

Bias-Variance tradeoff



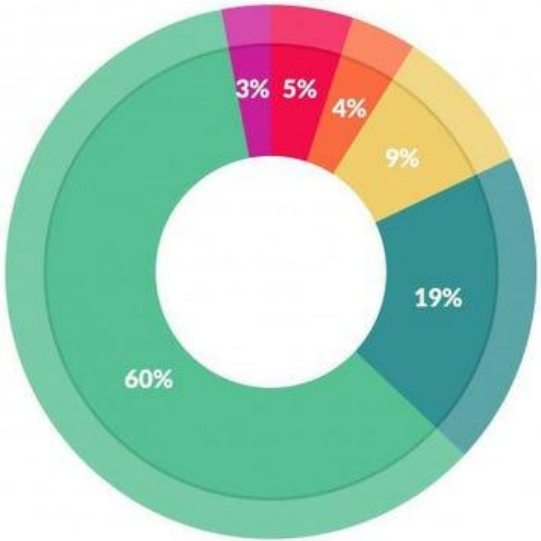


Bias-Variance tradeoff **interactive demo**



http://www.r2d3.us/visual-intro-to-machine-learning-part-2/

Data science

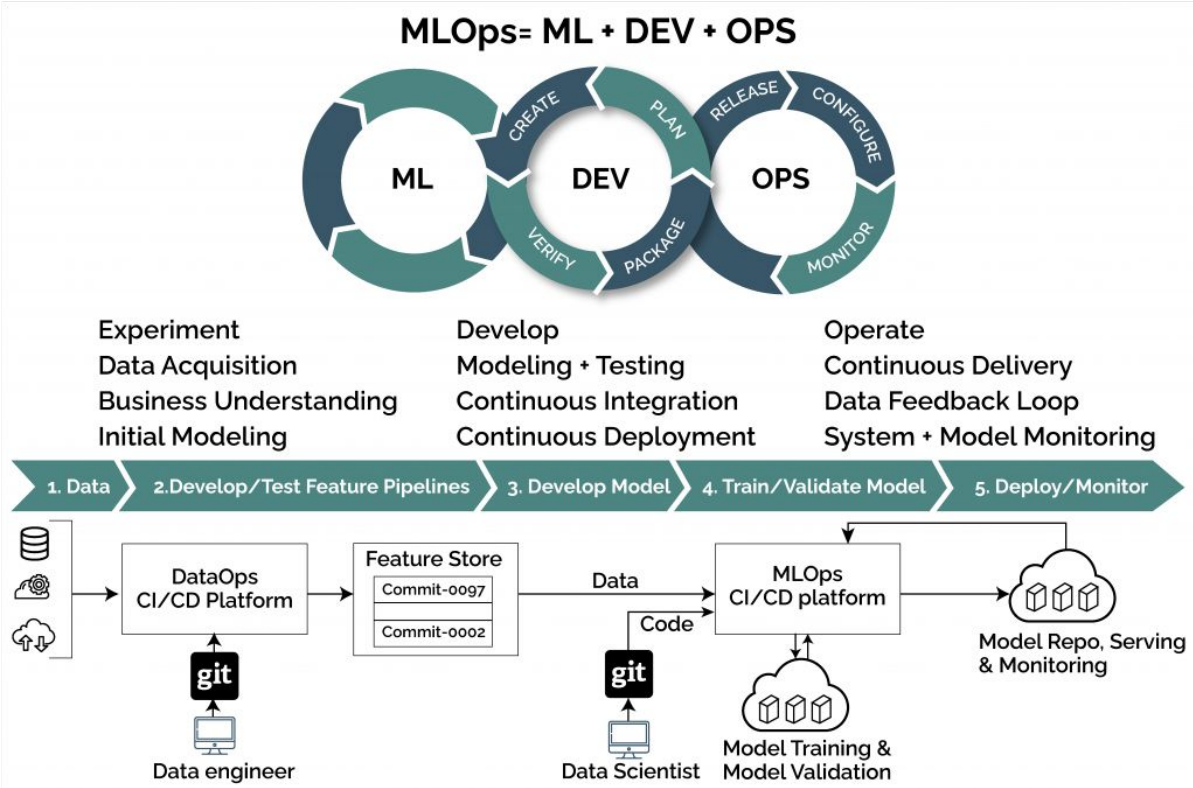


What data scientists spend the most time doing

- *Building training sets: 3%*
- *Cleaning and organizing data: 60%*
- *Collecting data sets; 19%*
- *Mining data for patterns: 9%*
- *Refining algorithms: 4%*
- *Other: 5%*

https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/

DL in practice



https://radiant.digital/the-fundamentals-of-mlops-the-enabler-of-quality-outcomes-in-production-environments/

Limitations of DL

Lack of reasoning

Lack of generalization →DL excels at *automating* tasks

Lack of prior knowledge → solvable with transfer learning

Lack of interpretability (black-box models) →XAI working on this

Lots of training data often needed →Recent research on *Efficient DL*

Models prone to adversarial noise →Hot topic!

DL is a “too-empirical” field

Recommended readings

1. Zachary C. Lipton, Jacob Steinhardt. Troubling Trends in Machine Learning Scholarship. arXiv:1807.03341, 2018.
2. D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, D. Dennison. Hidden Technical Debt in Machine Learning Systems. Proc. NeurIPS, 2015.

Some tricks and tips

UNIVERSITÉ DU LUXEMBOURG

http://cs231n.github.io/neural-networks-3/

http://karpathy.github.io/2019/04/25/recipe/

https://towardsdatascience.com/3ca24c31ddc8

https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks

Bibliography

1. F. Chollet. **Deep Learning with Python**, 2021
2. M. Nielsen. **Neural Networks and Deep Learning**, 2015
3. I. Goodfellow, Y. Bengio, A. Courville. **Deep Learning**, 2016
4. A. Gibson, J. Patterson. **Deep Learning: A Practitioners Approach**, 2017
5. R. O. Duda, P. E. Hart, D. G. Stork. **Pattern Classification**, 2000