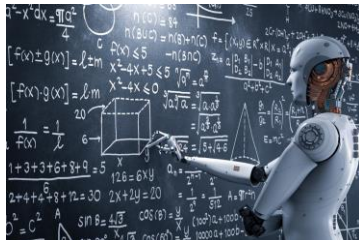


Introduction to ML

Olivier Caelen

What is artificial intelligence ?

Any technique that allows the computer to imitate human behavior



Credit: Civilization - FIRAXIS GAMES / 2K GAMES

AI for strategy-
games

• **Brut-force AI** (infeasible when the decision space is too high)

• **Rule-based AI**; e.g., expert system

e.g., Age of empires II AI Scripting:
<https://gist.github.com/Andygmb/1e3a6d9d444b2dfa8c40>

• **AI with learning**

This training focused on these techniques

What are AI, ML and DL ?

Artificial Intelligence

Any technique that allows the computer to imitate human behavior



Machine Learning

Ability to learn without being explicitly programmed



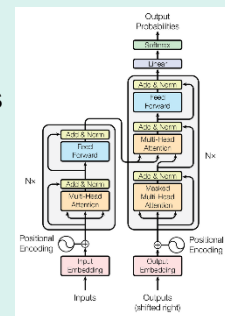
Deep learning

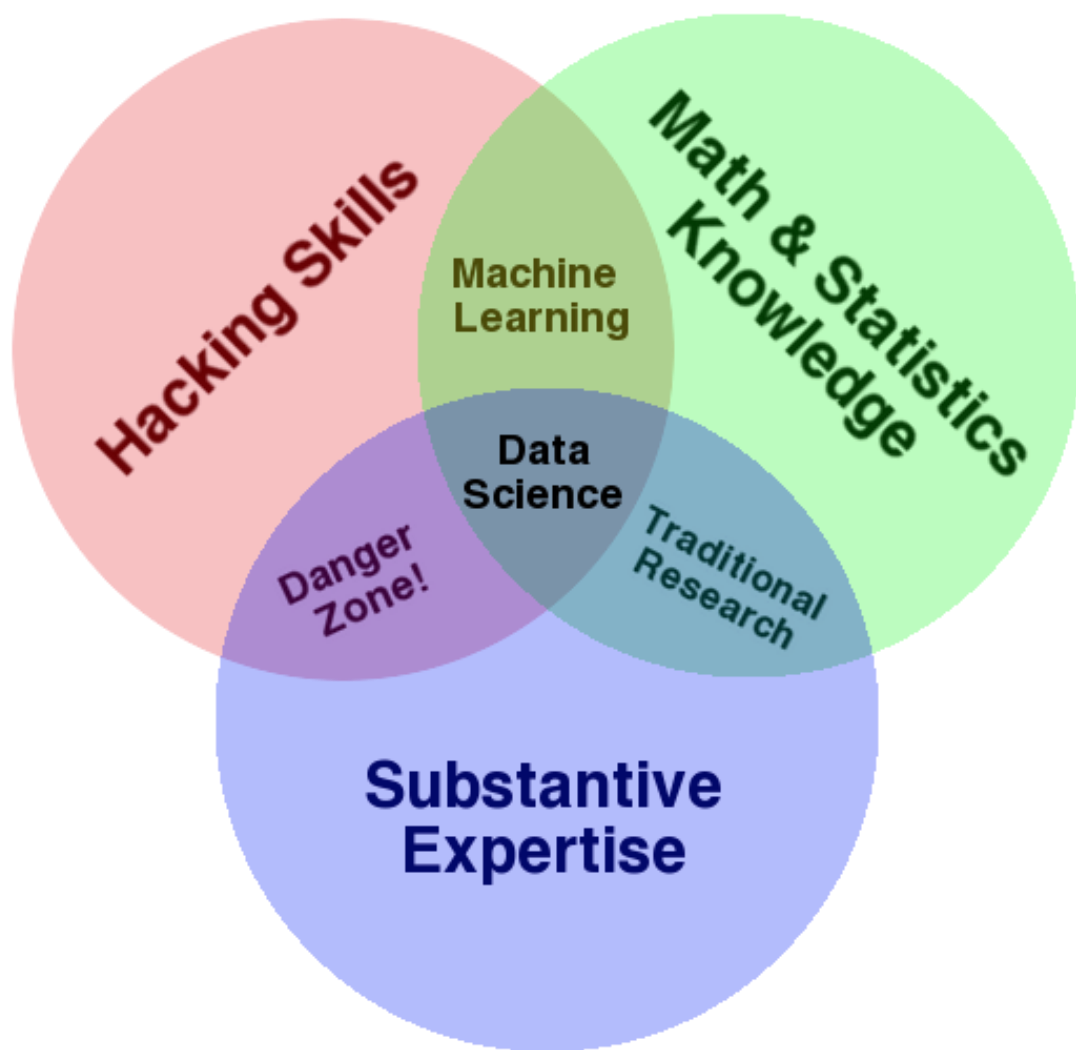
Extract patterns from data using artificial neural networks



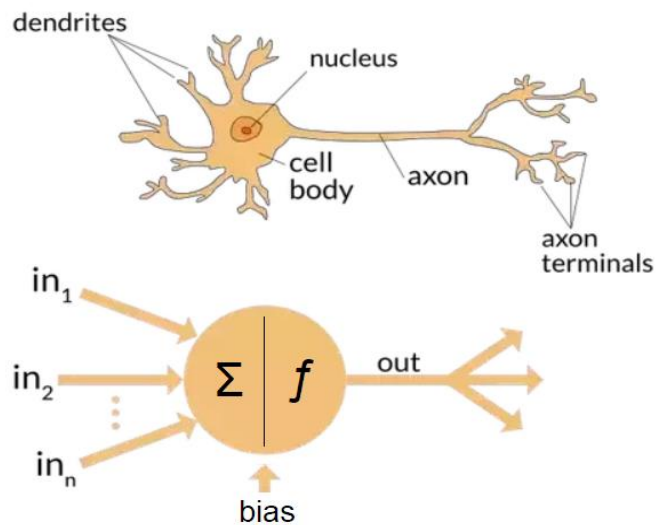
Transformers

Models for sequence-to-sequence tasks using attention mechanisms

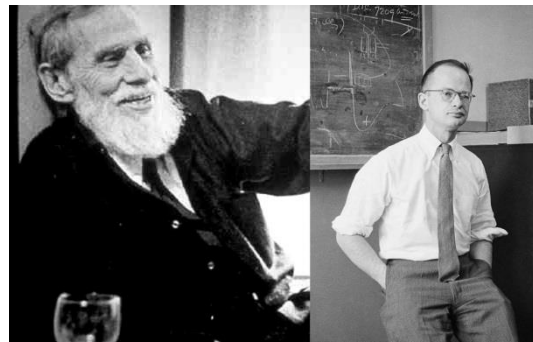




HISTORY: A first ARTIFICIAL NEURAL NETWORK



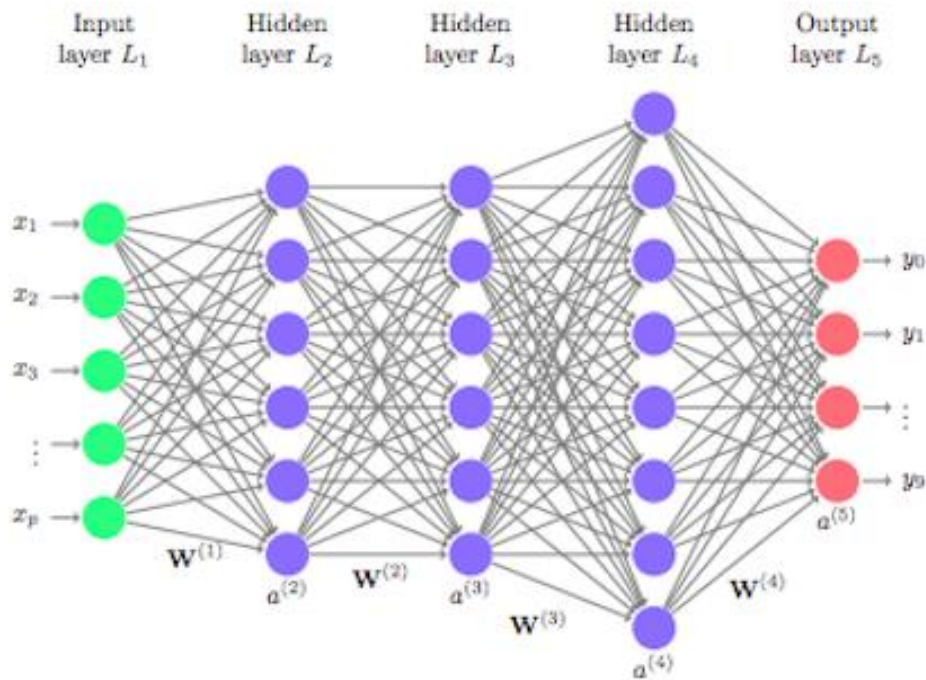
Frank
Rosenblatt
1958




McCulloch & Pitts
1943

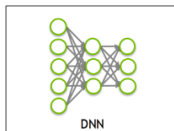
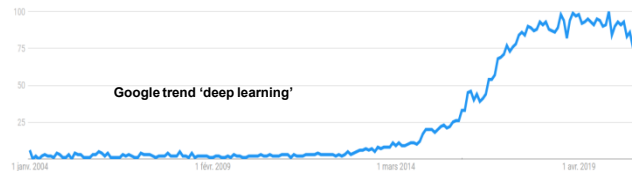
What is deep learning?

Roughly speaking, it's that :

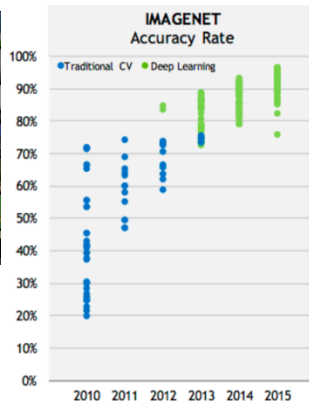
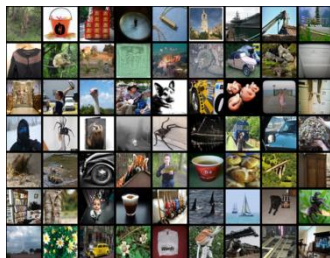


Why deep learning?

- Deep learning systems are neural networks
 - Exist since the 70s (and even before).
 - What has changed:
 - more data
 - new hardware
 - new algorithms
 - new software
- 
- A word cloud with a black border containing various terms related to big data. The most prominent words are 'BIG', 'DATA', and 'VOLUME' in large purple letters. Other visible words include 'PROCESSING', 'STORAGE', 'ANALYSIS', 'NETWORKS', 'SYSTEMS', 'CAPACITY', 'GROWTH', 'INFORMATION', 'TECHNOLOGY', 'COMPUTATION', 'PERFORMANCE', 'SCALABILITY', 'RESILIENCE', 'AGILITY', 'INTEGRITY', 'SECURITY', 'EFFICIENCY', 'EFFECTIVENESS', 'PRODUCTIVITY', 'QUALITY', 'RELIABILITY', 'AVAILABILITY', 'FLEXIBILITY', 'ADAPTABILITY', 'INNOVATION', 'CREATIVITY', 'COLLABORATION', 'COMMUNICATION', 'COOPERATION', 'COORDINATION', 'CONSISTENCY', 'CONTINUITY', 'COURTESY', 'COURTSHIP', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY', 'COURTESY'.



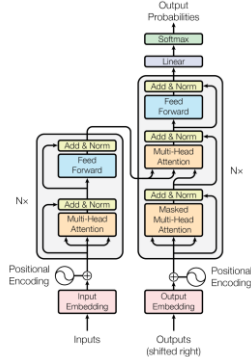
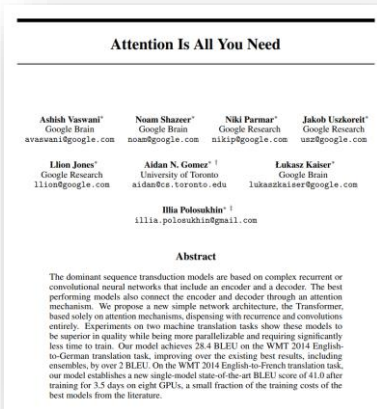
14,197,122 indexed images



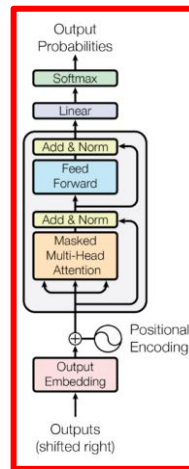
Some landmarks:

- **2012:** 'AlexNet' deep network. Provided 10% increase in accuracy on state of the art image classification (ImageNet)
- **2015:** Classification accuracy of deep learning for images reported to outperform humans
- **2016:** AlphaGo from Google DeepMind outperforms Go world champion
- **2017-2020:** Spectacular progresses in generative models, both in computer vision (GANs) and NLP (Transformers)
- **2018:** Turing award for deep learning

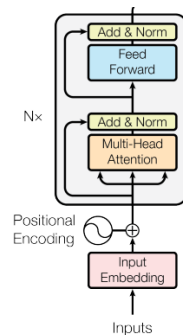
2017: Transformers



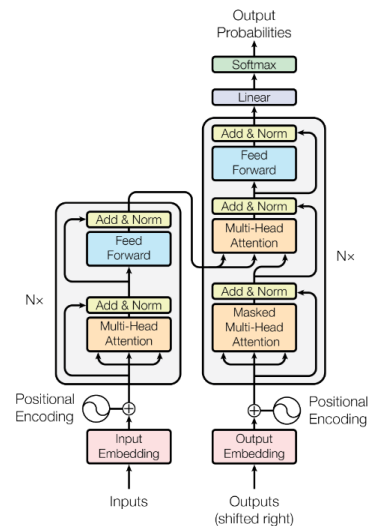
GPT



BERT



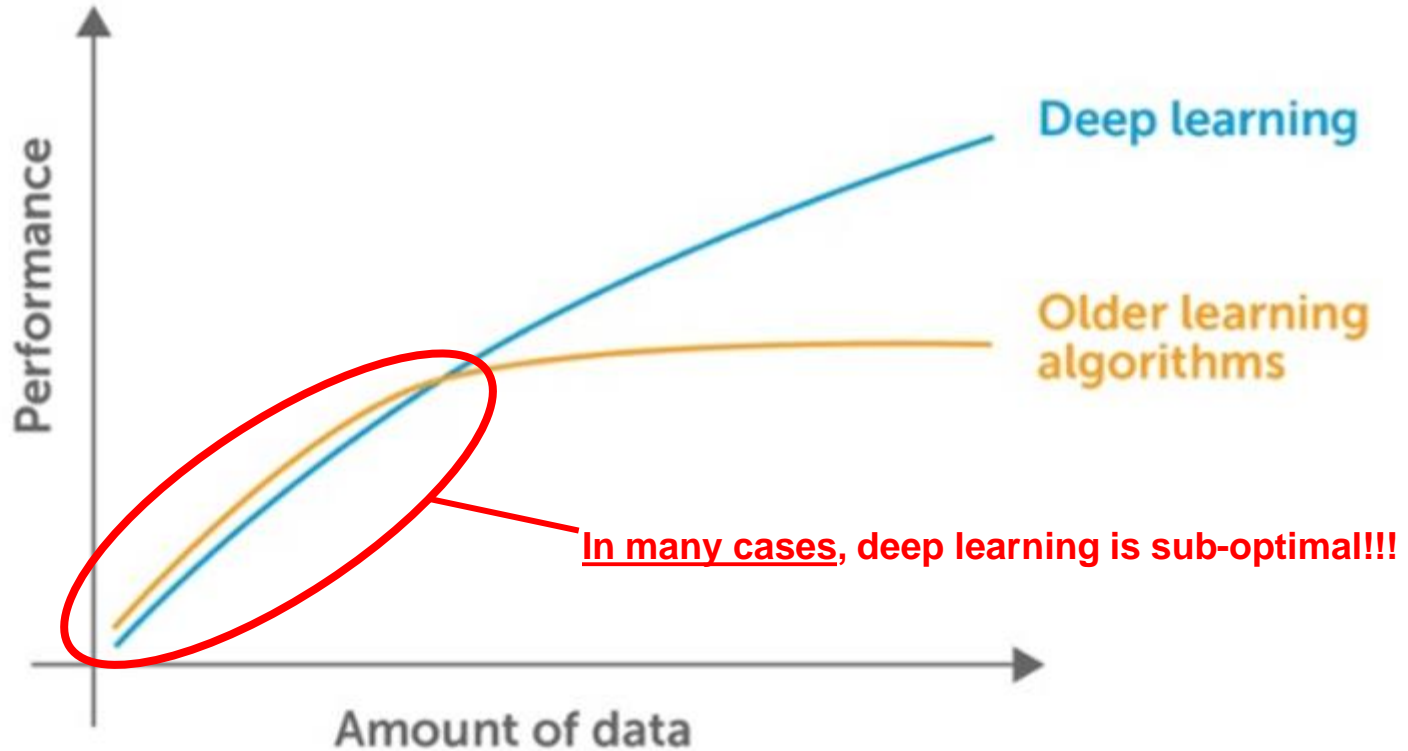
T5



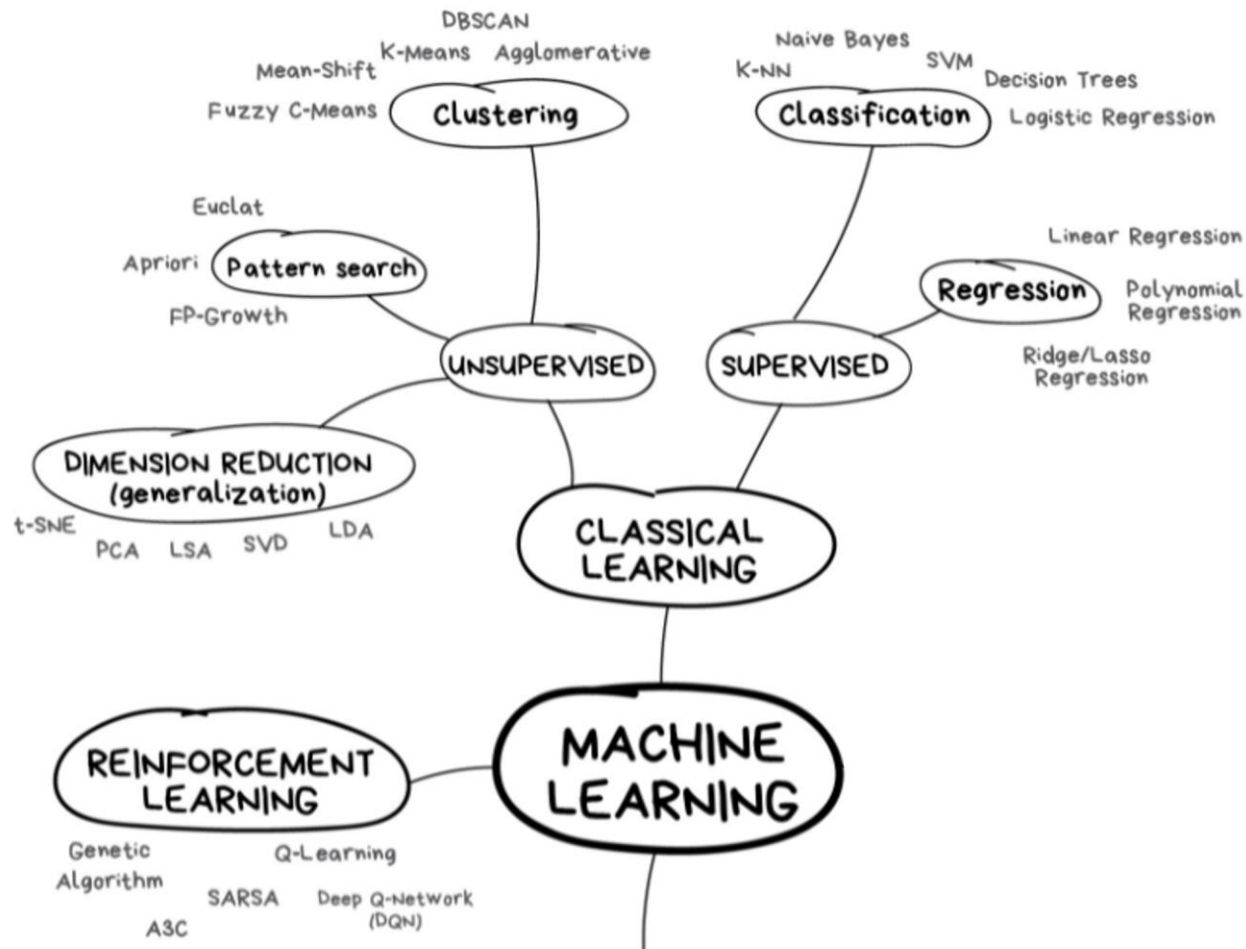
This paper has been a **groundbreaking** contribution to the field of AI in recent years.

2018	GPT-1	117 million parameters
2019	GPT-2	1.5 billion parameters
2020	GPT-3	175 billion parameters
2022	GPT-3.5 (ChatGPT)	175 billion parameters
2023	GPT-4	???

Why deep learning? ... and why not !



Credit: <https://medium.datadriveninvestor.com/when-not-to-use-neural-networks-89fb50622429>



Supervised vs unsupervised learning

Supervised

X_1	X_2	X_3	Y

Target

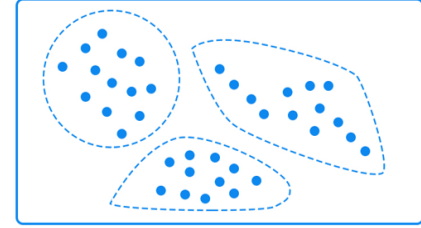


Unsupervised

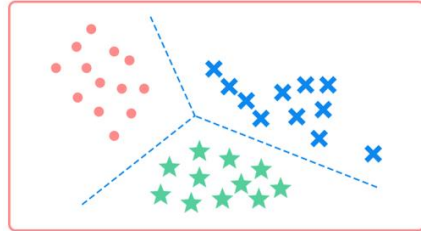
X_1	X_2	X_3

Target is a “*number*” (size, price, age, ...) → regression
Target is a “*category*” (Yes/No, a color, ...) → classification

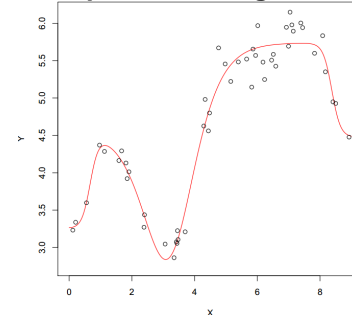
Unsupervised (clustering)



Supervised - classification



Supervised - Regression



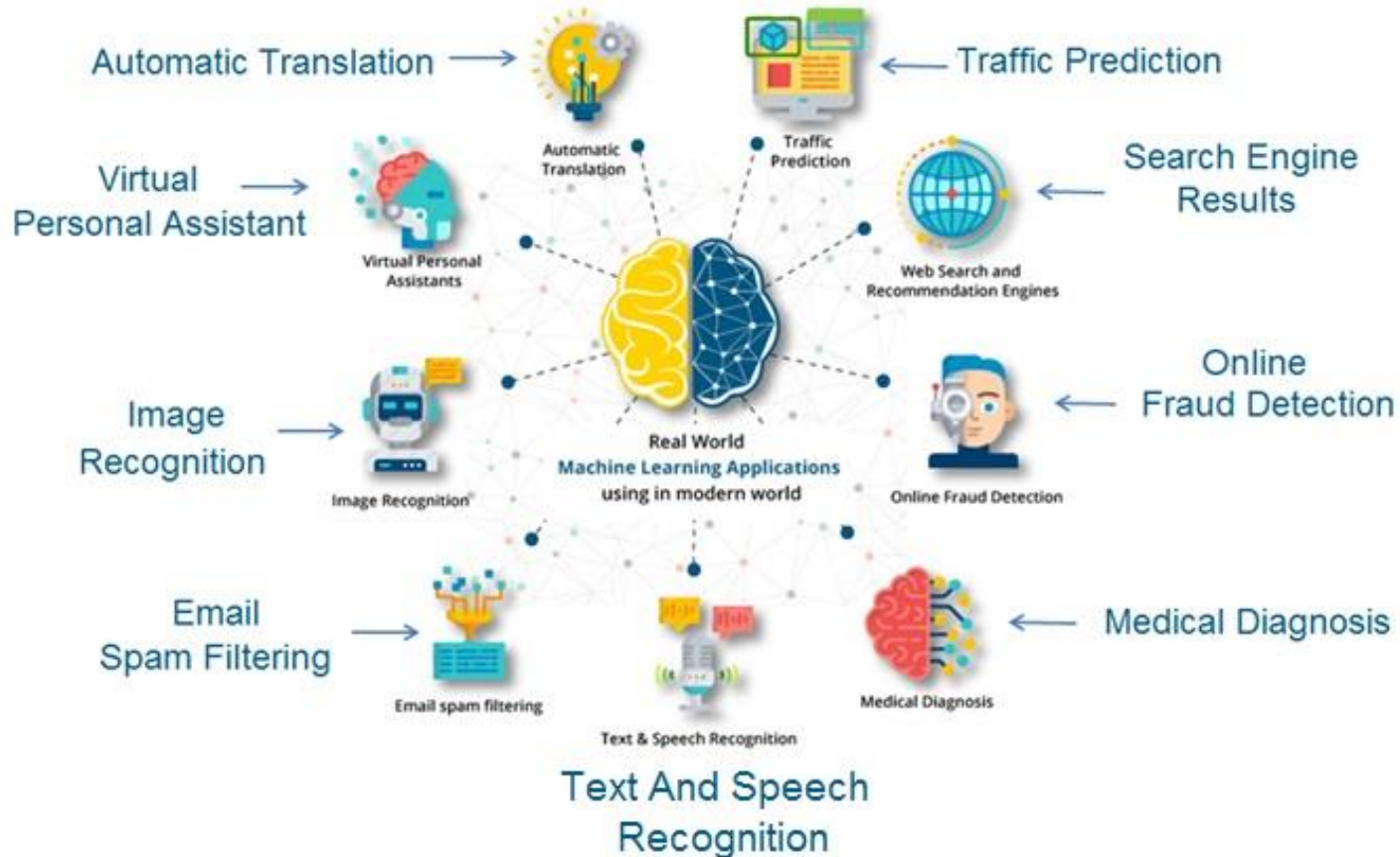
Question for you: Why does an engineer need machine learning?

Let's brainstorm!

Non-exhaustive list of other engineering application fields:

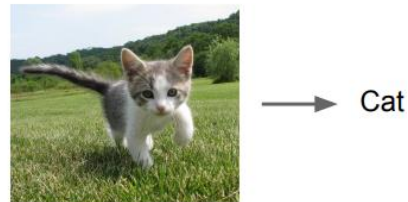
- Maintenance, default detection
- Logistics, scheduling, supply chains
- Cybersecurity, spamfilter, virus detection
- Prediction, e.g.: next word, density distribution, etc.
- Classification: signal types, automatic tagging, etc.
- Recommendation: social media, marketing, music/film, etc.
- Image correction
- Smart house
- ...

Real World Applications Of Machine Learning



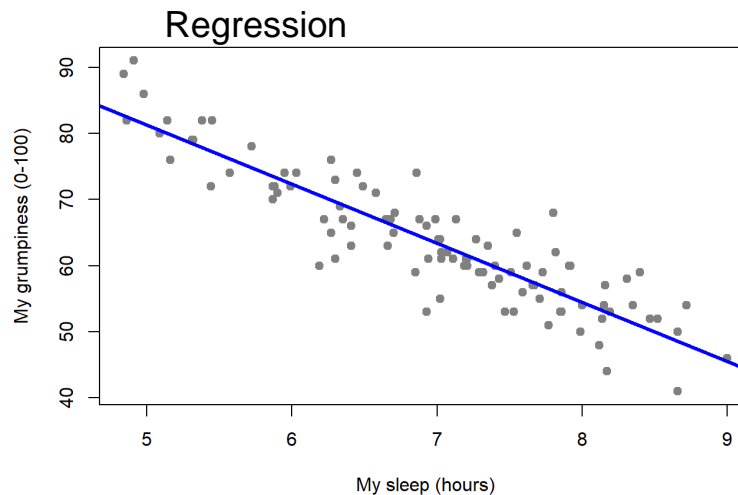
First 'Formalization'

Supervised learning



Classification

- **Learning to predict something**
- **Data:**
 - x = input variable, y = variable to predict
- **Goal:**
 - Learn the mapping : $x \rightarrow y$
- **Examples:**
 - Fraud detection
 - Image classification
 - Price regression



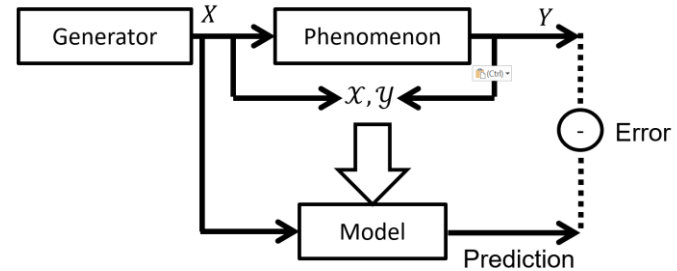
Supervised learning

- Assume that we have a dataset where the lines are observations and columns are variables.
- In a supervised learning problem, we have (at least) one variable which is the “target” (output variable).
 - The other variables are called the input variables
- Two types of target variables:
 - Numeric \rightarrow regression problem
 - Nominal \rightarrow classification problem
- In both cases, the dataset is used to produce a model

X1	X2	Target
1,57	0,3	A
12,3	0,98	A
32,7	-1,3	B
1,8	0,01	A
7,5	1,1	B
3,2	1,2	B
1,2	-0,5	A
...

X1	X2	Target
1,57	0,3	4.2
12,3	0,98	6.8
32,7	-1,3	-12.6
1,8	0,01	42.6
7,5	1,1	11
3,2	1,2	0
1,2	-0,5	19.5
...

Actors in supervised learning



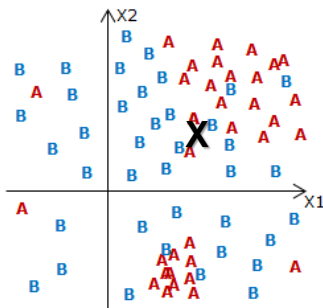
- A generator produces independent input values from an unknown distribution
- An unknown phenomenon assigns an output to each input
- Samples are collected in a dataset (X, Y)
- A learning algorithm is used on the samples (X, Y) to produce a model
- The model can be used to make predictions on **new inputs**.
- Goal: error on predictions must be minimized

Supervised learning (classification)

Given a set of data \mathcal{X}, \mathcal{Y}
with input/target variables:

X1	X2	Target
1,57	0,3	A
12,3	0,98	A
32,7	-1,3	B
1,8	0,01	A
7,5	1,1	B
3,2	1,2	B
1,2	-0,5	A
...

It's a supervised learning problem.



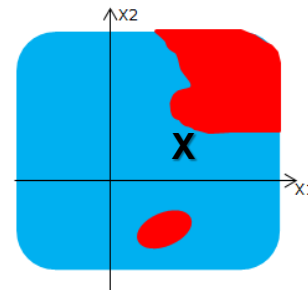
Practical examples:

- ▶ Bought a certain product?
- ▶ Repay their loan or not?
- ▶ Fraudulent transaction?

Let "X" be a new point, how can I know
the best class ("A" or "B") of this point?

➡ Via classification algorithms.

- Trees
- Logistic regression
- Neural Network
- ...

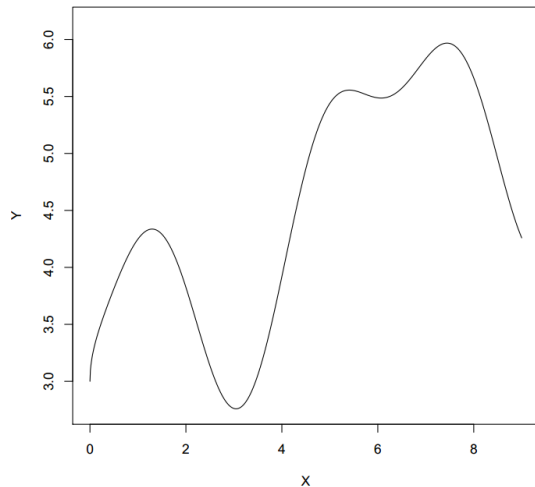


- The classification algorithm has split the input space in an optimal way.
- The new point can now be labeled.
- And this result can be used for all new points!

Think at this same problem in a higher dimension!

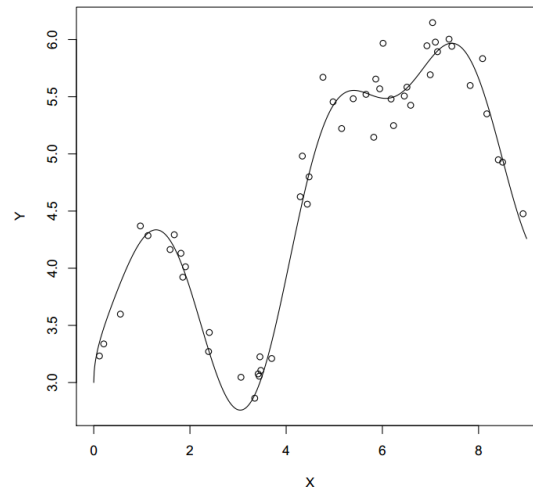
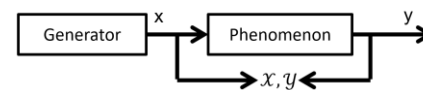
Supervised learning

Regression (I)



Assume that we have an unknown input/output phenomenon.

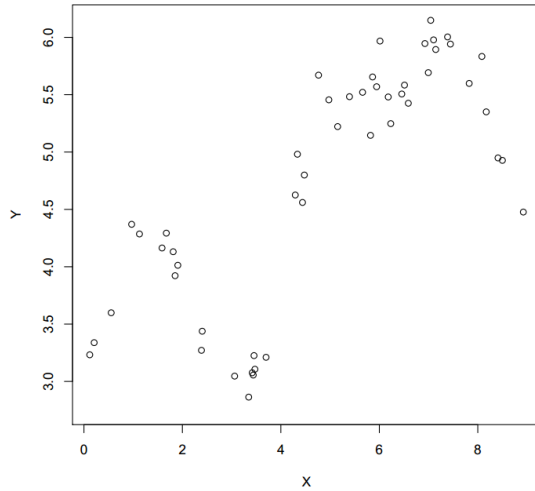
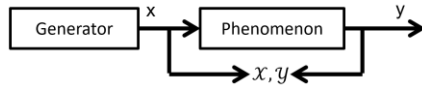
Regression (II)



- Some observations about the unknown phenomenon are available.
- There is noise in the data.

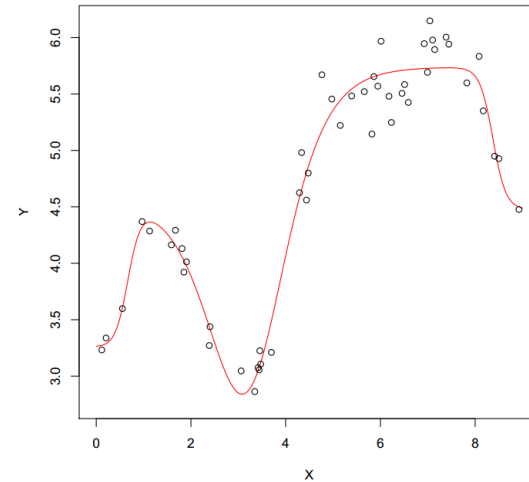
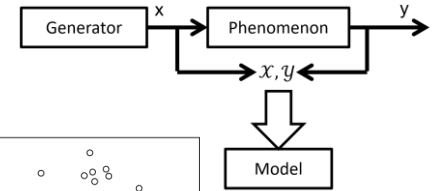
Supervised learning

Regression (III)



- Phenomenon is unknown!
- We only have the samples.

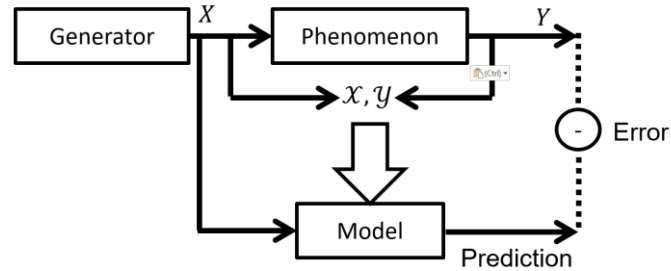
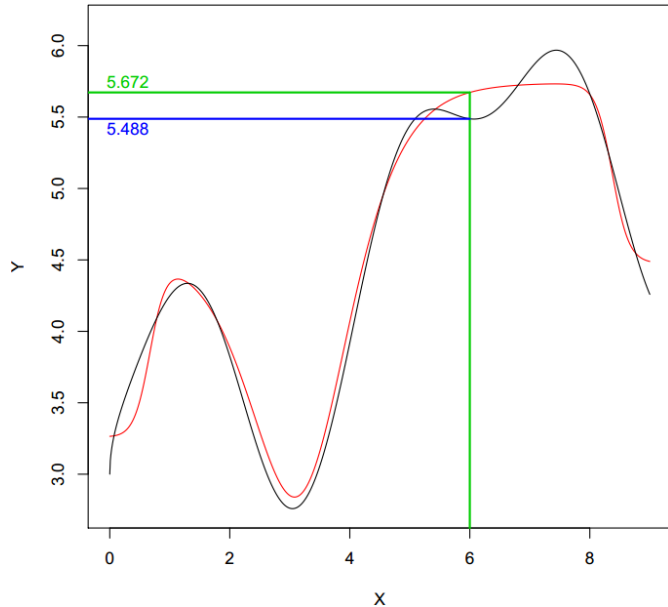
Regression (IV)



We use a learning algorithm to create a model $h(\cdot, \alpha)$ from the data.

Supervised learning

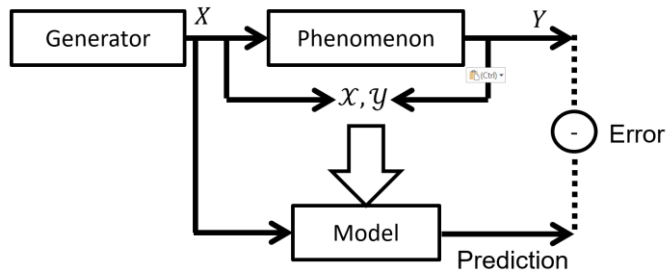
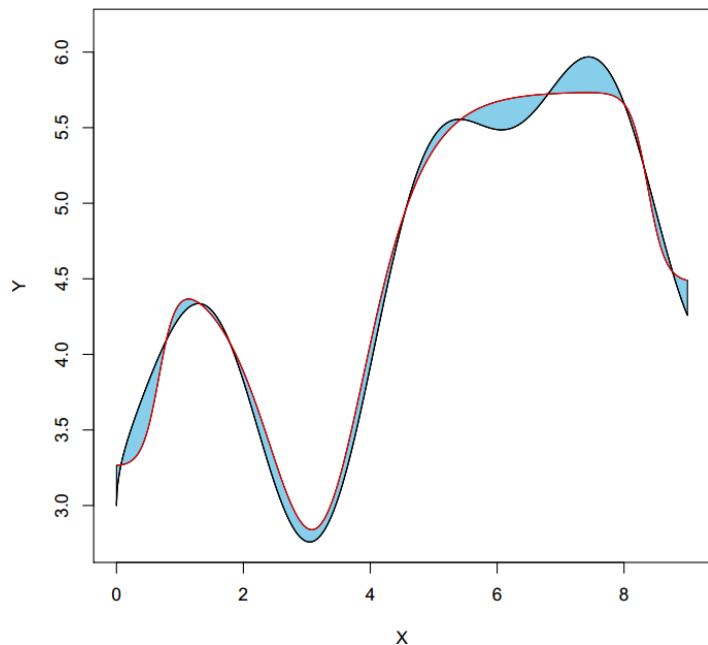
Regression (V)



- The model is an estimation of the true unknown input/output phenomenon.
- It makes errors.

Supervised learning

Regression (VI)



- The surface in blue measures the error of the model on the input domain.
- Without access to the real phenomenon, the goal is to find - based on the available data - the best model which minimizes the blue surface.

Think at this same problem in a higher dimension!

Unsupervised learning

- **No supervision, just analyze and discover**

- **Data:**

- x = descriptive variables

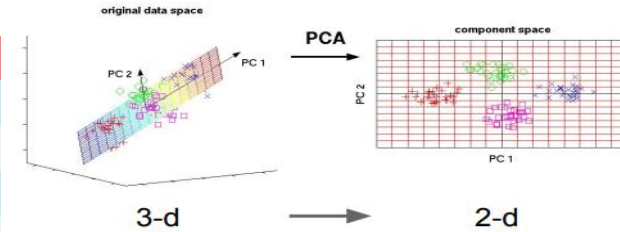
- **Goal:**

- Compress
 - regroup
 - learn a hidden structure
 - ...

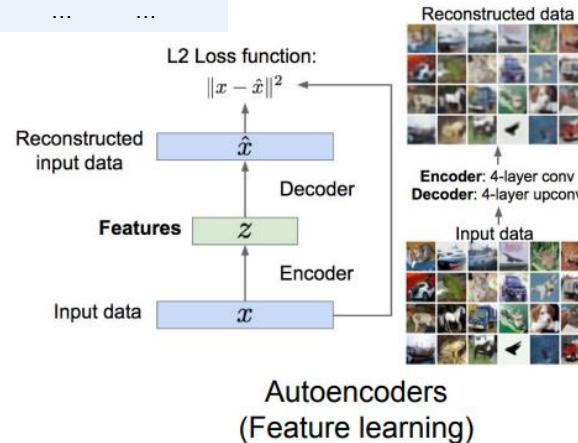
- **Examples:**

- Image compression/denoising
 - Client segmentation
 - Detect anomalies

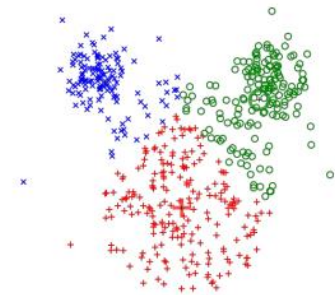
X1	X2	Target
1,57	0,3	A
12,3	0,98	A
32,7	-1,3	B
1,8	0,01	A
7,5	1,1	B
3,2	1,2	B
1,2	-0,5	A
...



Principal Component Analysis
(Dimensionality reduction)



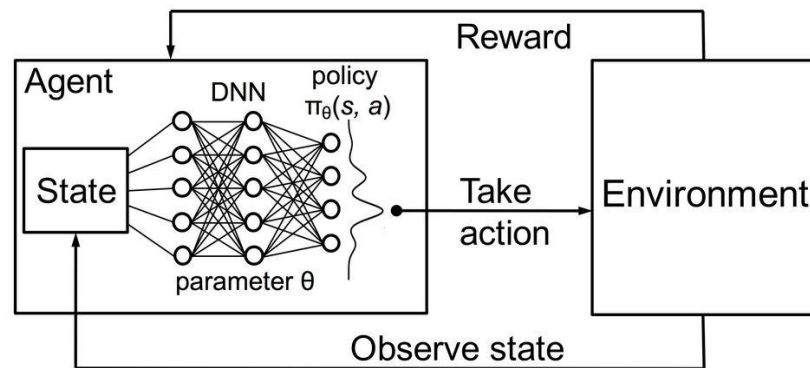
Autoencoders
(Feature learning)



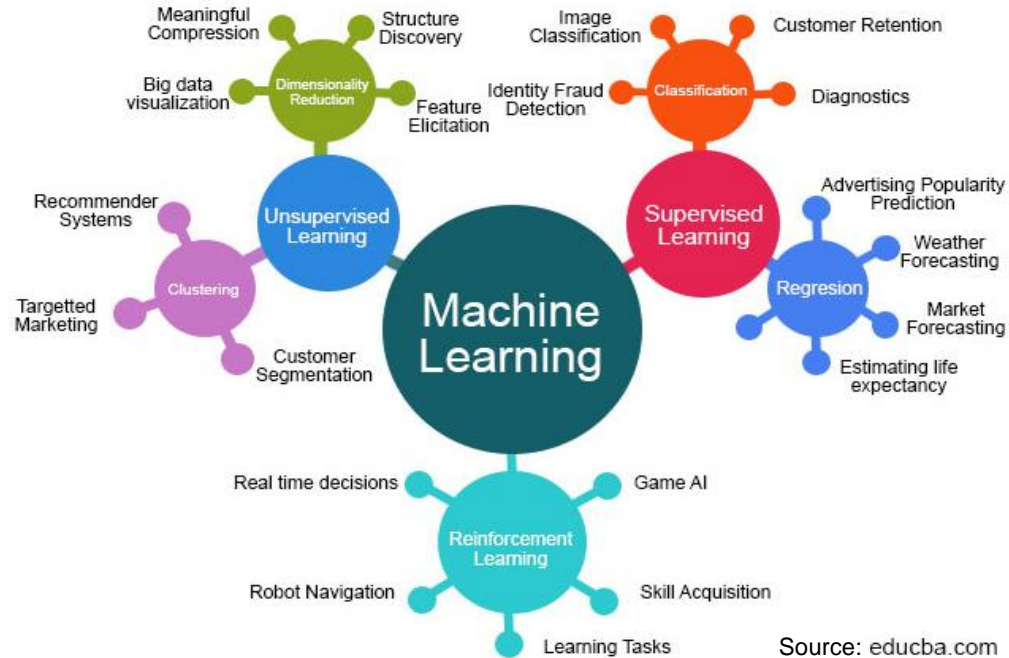
K-means clustering

Reinforcement learning

- Take actions, get a reward, improve your strategy
- **Data:**
 - x s = environment state
 - y a = action to take
 - a is not given for each s
 - Reward (negative or positive) sometimes given (after series of actions)
- **Goal:**
 - Learn the policy $\pi : s \rightarrow a$
- **Examples:**
 - Learn best strategies in games or other complex environment
 - Robotics
 - https://www.youtube.com/watch?v=VMp6pq6_QjI



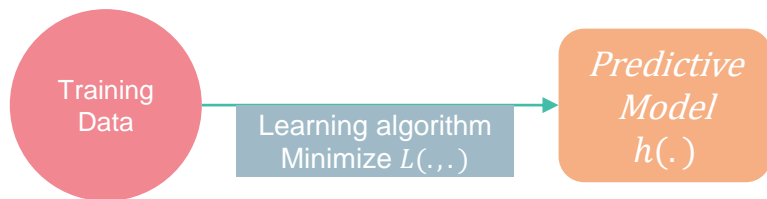
Reinforcement learning is not covered in this course



Source: educba.com

Supervised learning in practice

OK, BUT REALLY... HOW ?



Concretely:

data = $(x_i, y_i) \ i = 1, \dots, N$
→ Training data

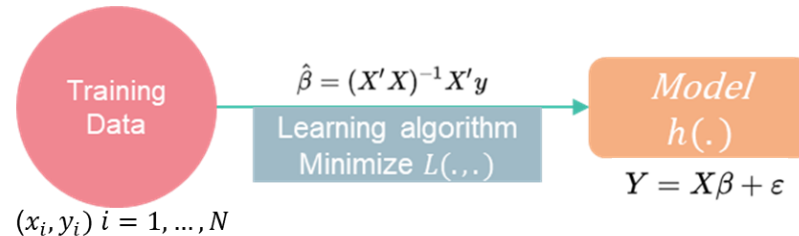
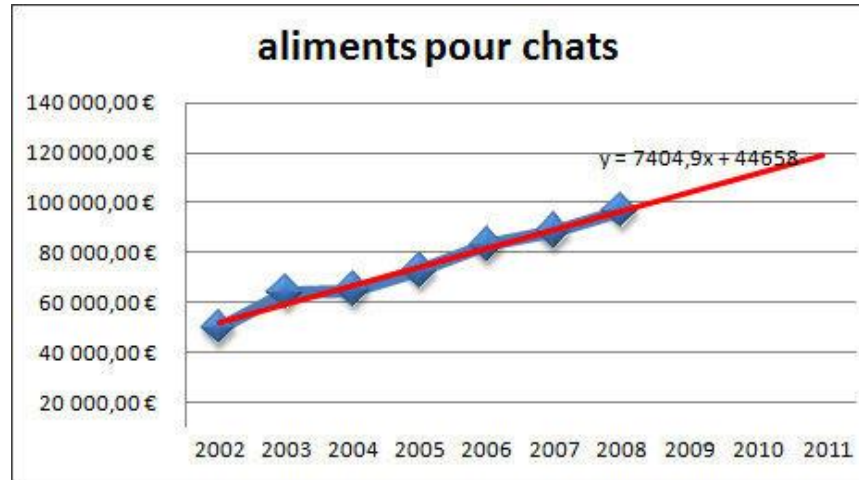
The goal is to find a function h , such that $h(x_i) = y_i$ for all i .
 $h(x_i)$ denoted as \hat{y}_i

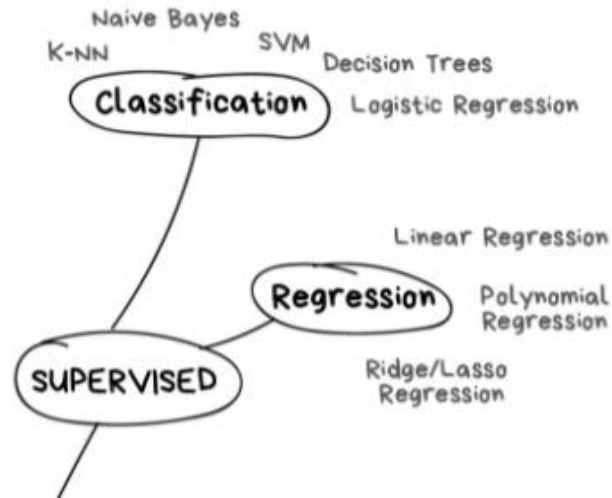
In practice, goal is to minimize $L(y_i, \hat{y}_i)$

The algorithm will take the training data set as input and will search a function $h()$ (a.k.a. predictive model) minimizing the loss function $L(.,.)$ computed on the training set.

This definition is not complete and will be updated in some slides!

LINEAR REGRESSION is Machine learning!



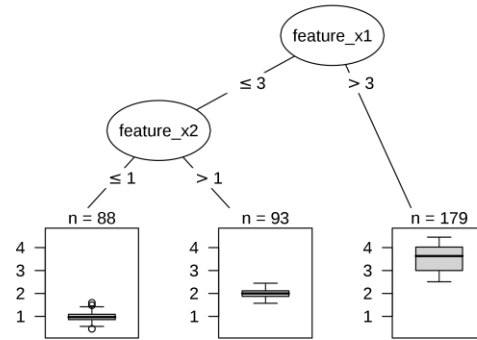
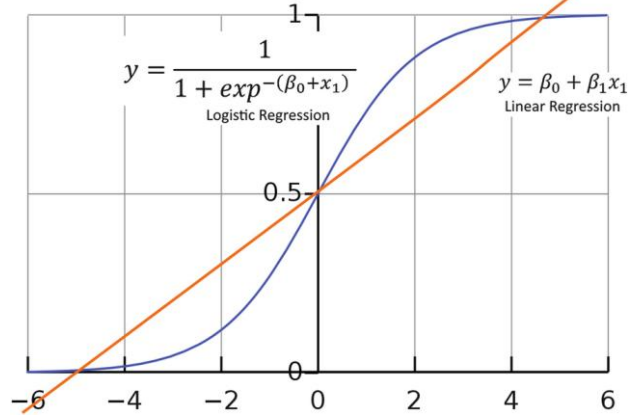


- Several **types of model** can be used for learning
 - Knn,
 - Naïve Bayes
 - SVM
 - ...
- They all have their advantages depending on the type of problem
- Each model type has also ***hyper parameters*** to control their “complexity”
- Roughly speaking:
 - applying an overly simple model to a complex problem
→ **underfitting**
 - applying an overly complex model to a simple problem
→ **overfitting**

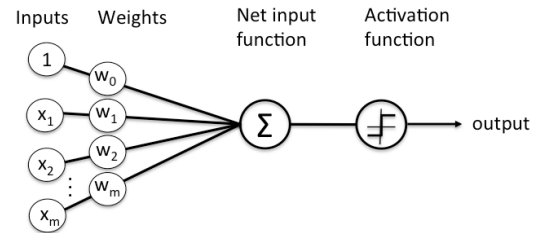
Warning: make the distinguish between a *type of model* (with its hyperparameters) and a *predictive model* (with its parameters).

Several types of models can be used

Logistic regression



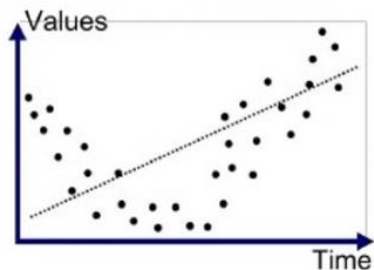
Decision Tree



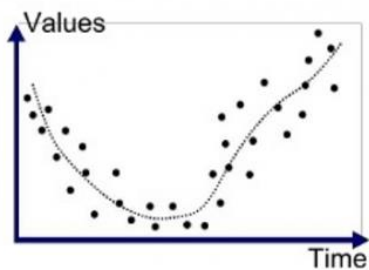
Perceptron

Underfitting & overfitting

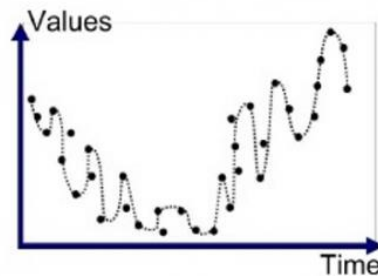
The dot points are the training data



Underfitted



Good Fit/Robust

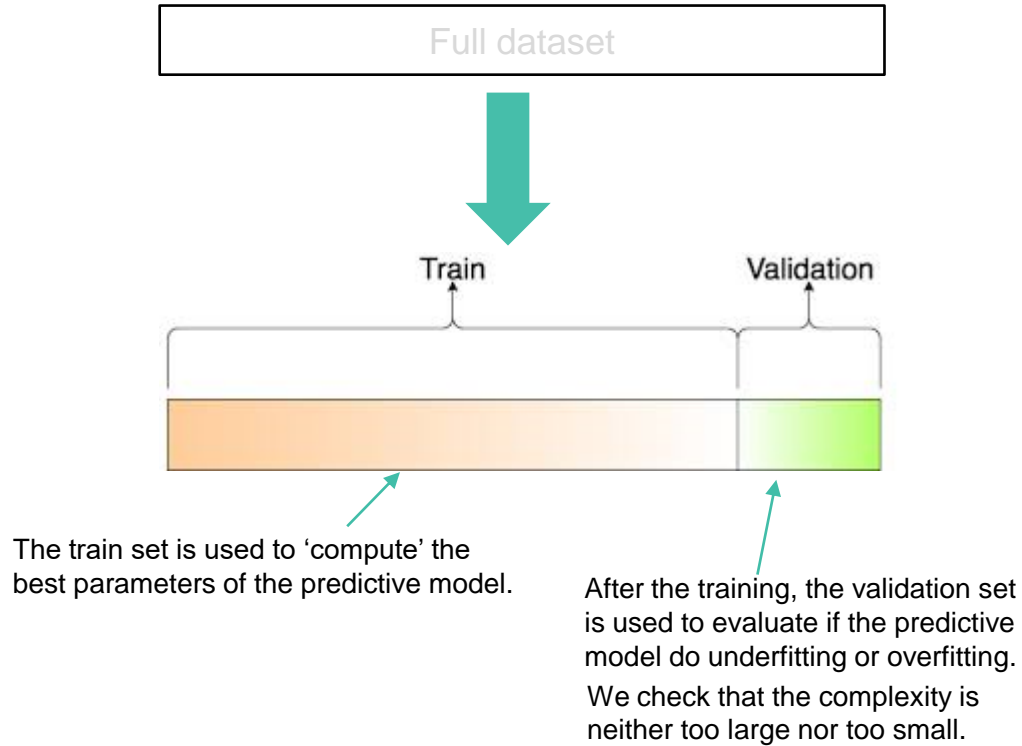


Overfitted

Question for you:
do you have any idea how to
avoid overfitting?

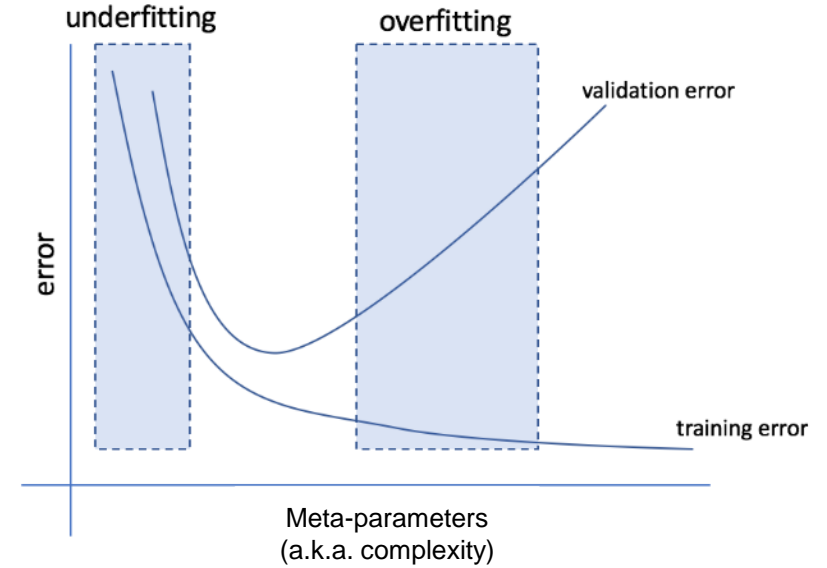
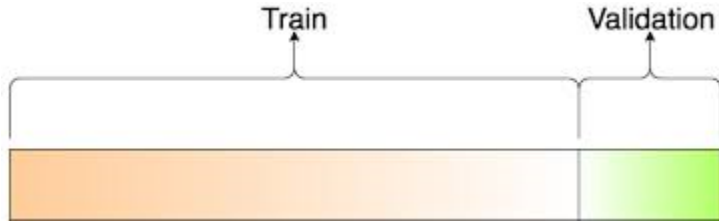
This function minimize $L(y_i, \hat{y}_i)$!

→ trying to minimize the error on the training set is therefore not a
“sufficient” criterion.



Learning is a *nested optimization* task.

Data set → training and validation sets

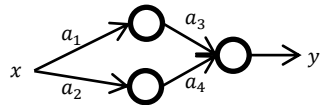


Supervised Learning (SL): optimization in Λ

Based on a training set \mathcal{d} , a supervised machine learning algorithm searches in a set of predictive models Λ an optimal model $h(x, \alpha)$ which should estimate at best the unknown function f .

α is a vector that contains the parameters, and we can write $\alpha \in \Lambda$ if $h(., \alpha)$ is a model in the type of models Λ .

type of models Λ	Model $h(., \alpha)$
Set of quadratic functions: $y = a_2x^2 + a_1x + a_0$	$\begin{cases} a_2 = 3 \\ a_1 = 5 \Rightarrow 3x^2 + 5x + 8 \\ a_0 = 8 \end{cases}$
Set of linear functions: $y = a_1x + a_0$	$\begin{cases} a_1 = 10 \\ a_0 = 2 \Rightarrow 10x + 2 \end{cases}$
Multilayer perceptron with 2 hidden nodes: $y = s(a_1x)a_3 + s(a_2x)a_4$	$\begin{cases} a_4 = 0,5 \\ a_3 = 2,2 \\ a_2 = 1,5 \\ a_1 = 8,5 \end{cases}$

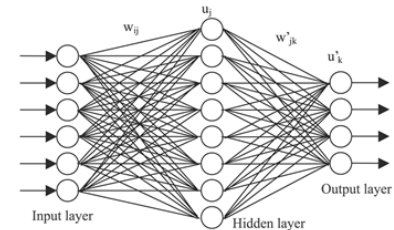


Let Λ^* be a set containing all the possible models and let us define the following sequence: $\Lambda_1 \subset \Lambda_2 \subset \Lambda_3 \subset \dots \subset \Lambda_*$

Example:

- $\Lambda_1 \equiv$ "Neural network with 1 hidden node"
- $\Lambda_2 \equiv$ "Neural network with 2 hidden nodes"
- $\Lambda_3 \equiv$ "Neural network with 3 hidden nodes"
- ...
- $\Lambda_* \equiv$ "Neural network with ∞ hidden nodes"

Question for you: can you give another example?



Question for you

What is the difference between a *type of model* and a *predictive model* ?

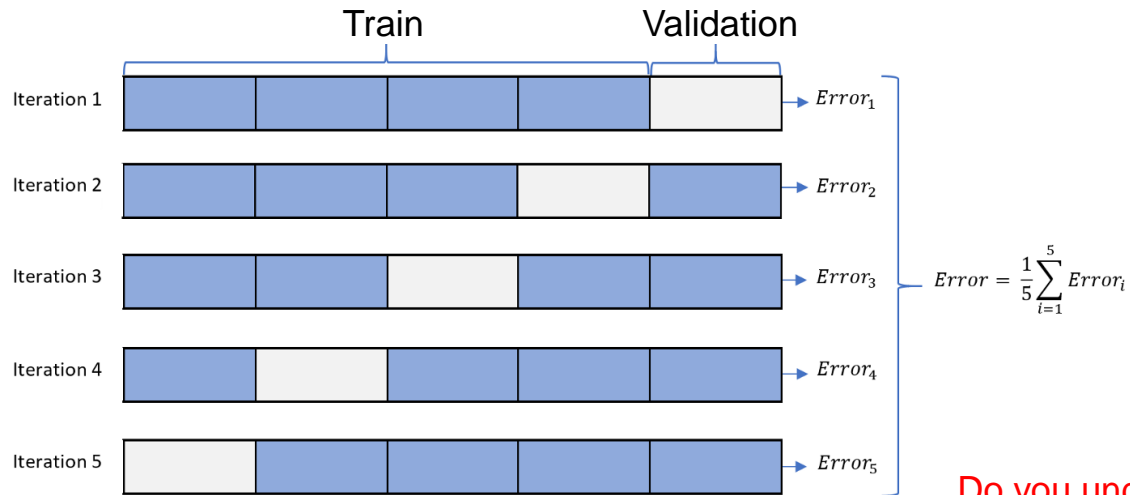
We evaluate a predictive model on data that it has never seen during training
→ We assess the “generalization” capacity of a prediction model.

Question for you: Do you understand why it is not possible to use the training set to assess the generalization capacity of a predictive model?

Answer: Because the parameters (e.g., the weights) of the predictive model have been optimized to minimize the error on this training set.

The error of the model on the training set is therefore optimistic with respect to the generalization error.

k fold cross validation



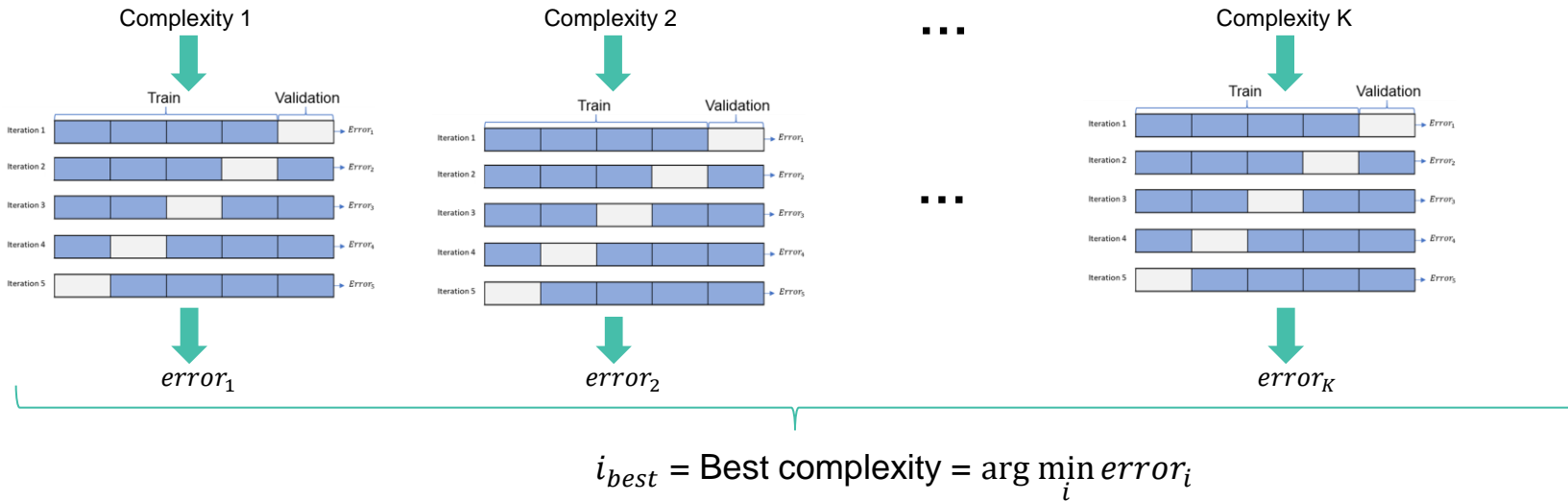
Question for you:

- What is the value of k in this example?
- What is the maximum value of k?

Do you understand that here we no longer evaluate the generalization error of a prediction model, but we evaluate the generalization error of something related to the “type of model with a given complexity”!!

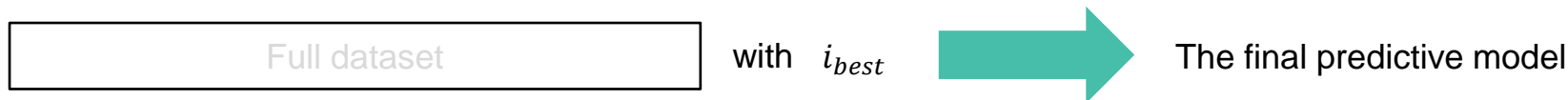
Optimal complexity selection (aka model selection)

- There are several ways to do model selection, but the easiest and most common way is to compare the generalization error of different complexities and select the one with the smallest error.
- This can be done by cross-validation or simple validation



Final predictive model

- i_{best} is the optimal complexity computed on the validation set.
- It is common to then use the full data set with i_{best} to build the final predictive model.

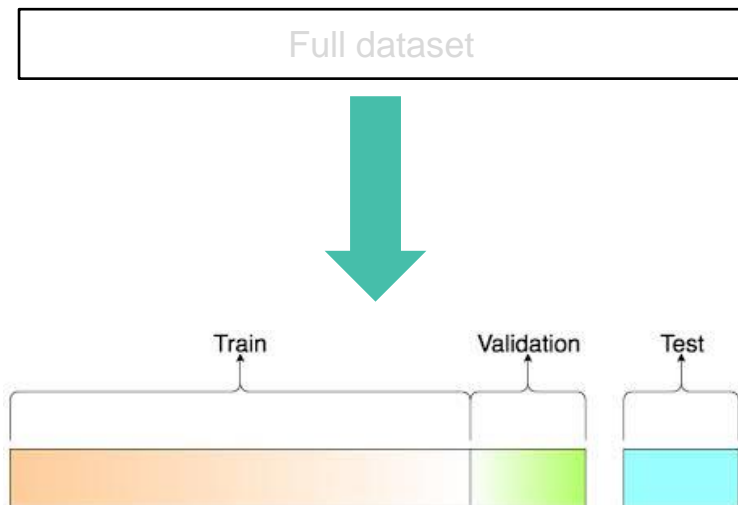


Question for you: Do you understand why it is not possible to use the validation set to assess the generalization capacity of this predictive model?

Answer: The hyper parameters have been optimized to minimize the error on the validation set.
We can no longer say that this validation error measures the error of the model on a dataset that it has never seen

Question for you: do you have any idea of how to measure the generalization error of this final predictive model?

Train, validation and test splits

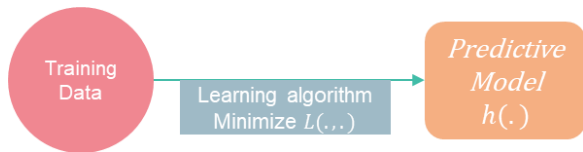


- You cannot touch this *test set* at any time during learning.
- To have a correct estimate of the generalization error, we test the predictive model on the test set only at the end, when the final predictive model is built.

Performance measures in classification & regression

Performance measures

- We have seen previously that during the learning process, the algorithm must minimize a loss function.



- An error function must also be computed on the validation set when searching for the best complexity.



- And of course, to evaluate the predictive capacity of the final predictive model on the test set, we also need an error function.

- Each type of problem (classification, regression ...) has its own set of error measures.
- There are many choices of error measures for the same type of problem.
- The choice of the error measure will have an important impact on the final prediction model.
- It is not mandatory to use the same error measure for each step: learning, validation and testing.
- Not all error measurements can be used at each step. This is often the case for the learning step.
- The error measure that can be used during learning step depends on the type of model
 - E.g., Neural networks require that the error measure can be derived
 - E.g., Tree based models has their specific error measure (i.e., Entropy/Gini)

Model evaluation: performance measures in classification - The confusion matrix

- The **confusion matrix** is typically used in machine learning to evaluate or to visualize the behavior of models in classification contexts.

		Prediction	
		1	0
Actual	1	True Positive (TP)	False Negative (FN)
	0	False Positive (FP)	True Negative (TN)

Example

		Actual	
		Pregnant	Not
Predicted	Pregnant	45 TP	55 FP
	Not	5 FN	395 TN

- It is a square matrix in which the rows represent the actual class of the instances and the columns their predicted class (we can sometimes also see the opposite).
- If we are handling a binary classification task, then the confusion matrix is a 2×2 matrix that reports the number of *true positives* (TP), *true negatives* (TN), *false positives* (FP), and *false negatives* (FN) as follows:
- This matrix contains all the raw information about the predictions done by a classification model on a given data set.
- To evaluate the generalization accuracy of a model, it is common to use a testing data set which was not used during the learning process of said model.

Model evaluation: performance measures in classification

Many synthetic one-dimensional performance indicators can be extracted from a 2×2 confusion matrix.

		Prediction	
		1	0
Actual	1	True Positive (TP)	False Negative (FN)
	0	False Positive (FP)	True Negative (TN)

$$\text{Accuracy: } \frac{TP+TN}{TP+FN+FP+TN}$$

$$\text{Precision: } \frac{TP}{TP+FP}$$

$$\text{Recall: } \frac{TP}{TP+FN}$$

$$\text{F1 score: } \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Question for you:

- These measures cannot be used during learning by all machine learning algorithms (like, neural net), why?
- Compute accuracy, precision, recall & F1 score

$$\text{Recall: } \frac{TP}{TP+FN}$$

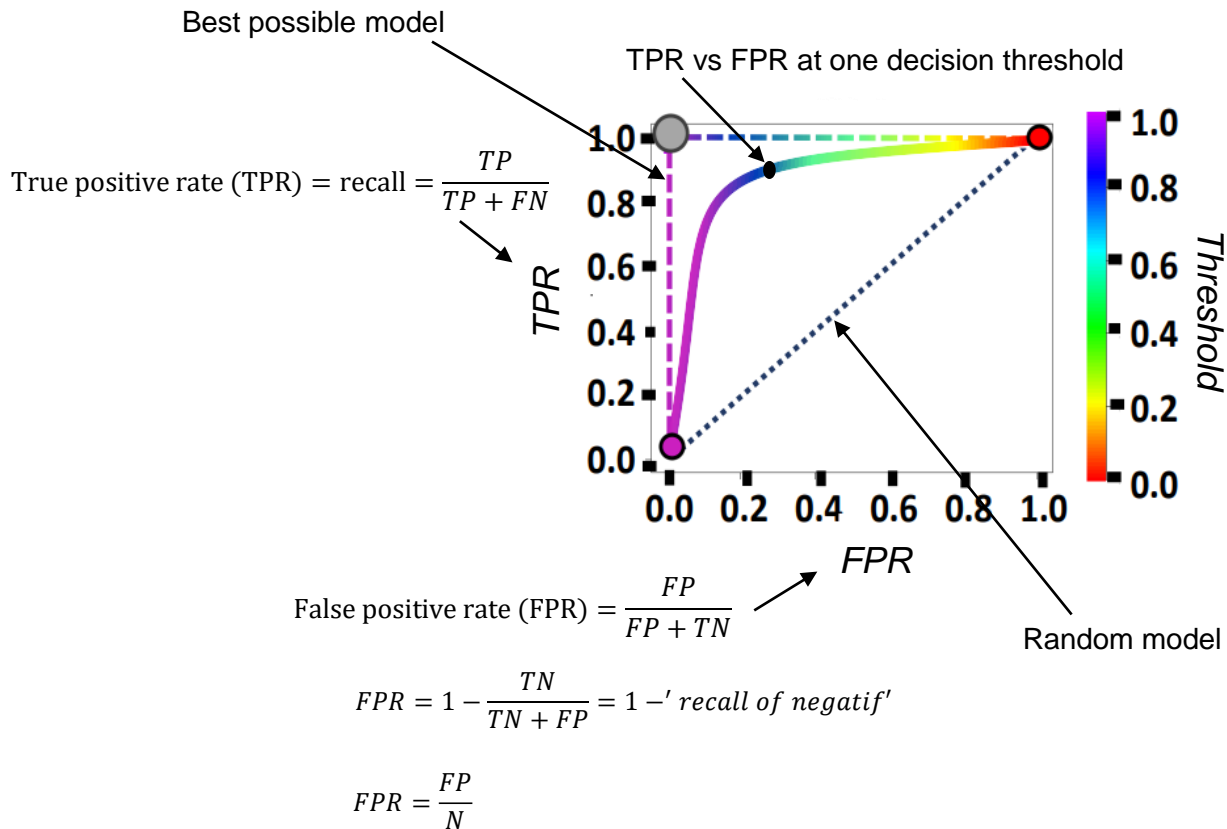
Probabilistic classifiers & threshold

- In binary classification context, rather than directly returning a class many models return a score in $[0,1]$.
- This score measures the likelihood that the observation x belongs to the positive class.
- A threshold γ can then be used to decide if the observation x belongs to the negative or the positive class.

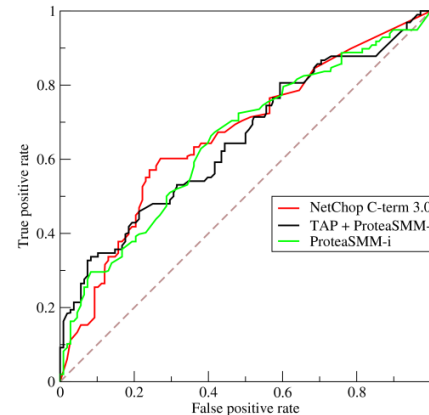
$$h(x) = \begin{cases} 1 & \text{if } g(x) > \gamma \\ 0 & \text{else} \end{cases} \quad \text{where} \quad g(x) \in [0,1]$$

- The error measures presented previously (recall, f-score ...) are computed directly from $h()$. As we will see next error measures can also be computed directly from $g()$.

Receiver operating characteristic (ROC) curve



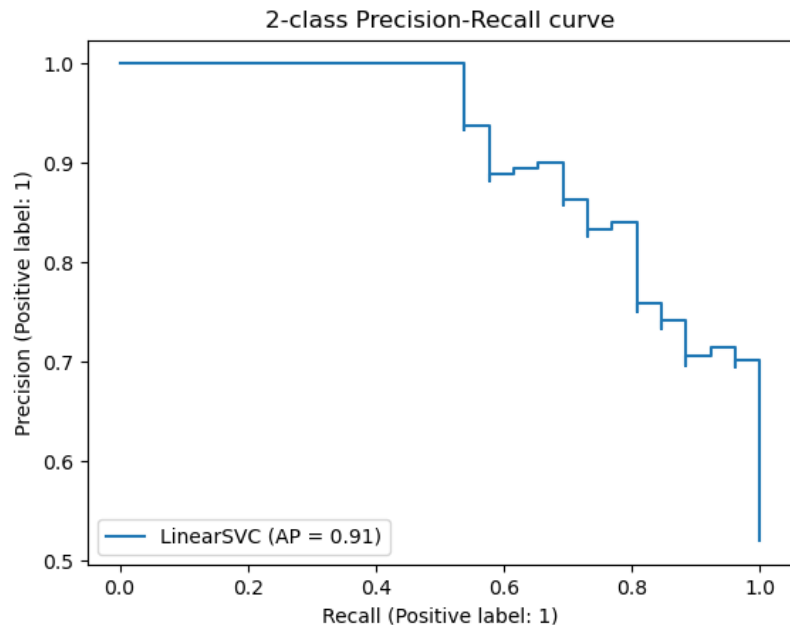
Roc curve can be used to compare different models.



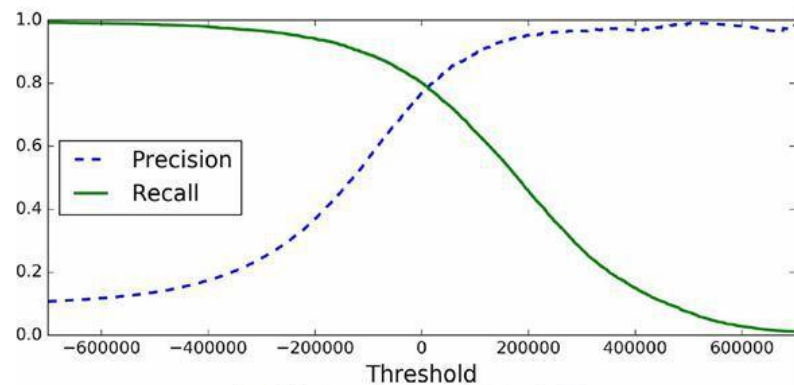
It is common to use the **Area under the ROC Curve (AUC)** to summarize the ROC curve in a *single number*.

Question for you: what is the AUC of the best and the random models?

Precision-Recall (PR) curves



Precision-Recall threshold curves



- Not all error measurements can be used at each step (learning, validation & test).
- This is often the case for the learning step.
- For instance, Neural networks require that the error measure can be derived

Binary cross-entropy - Commonly used by neural networks for binary-classification problems

Let

- $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ be a dataset with N observations
- $g(x_i) \in [0, 1]$ be the output of the model
- $y_i \in \{0, 1\}$ be the true output



Binary cross-entropy:

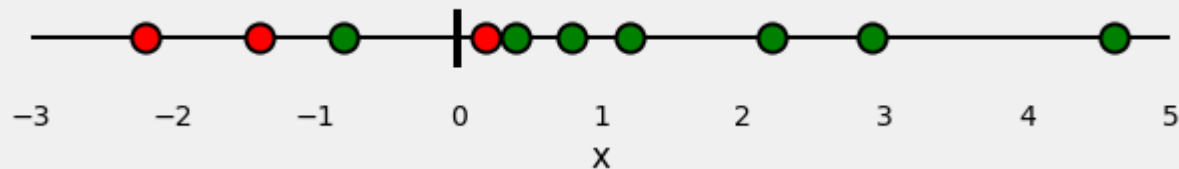
$$H(g) = -\frac{1}{N} \sum_{i=1}^N y_i \cdot \log(g(x_i)) + (1 - y_i) \cdot \log(1 - g(x_i))$$

Reading this formula:

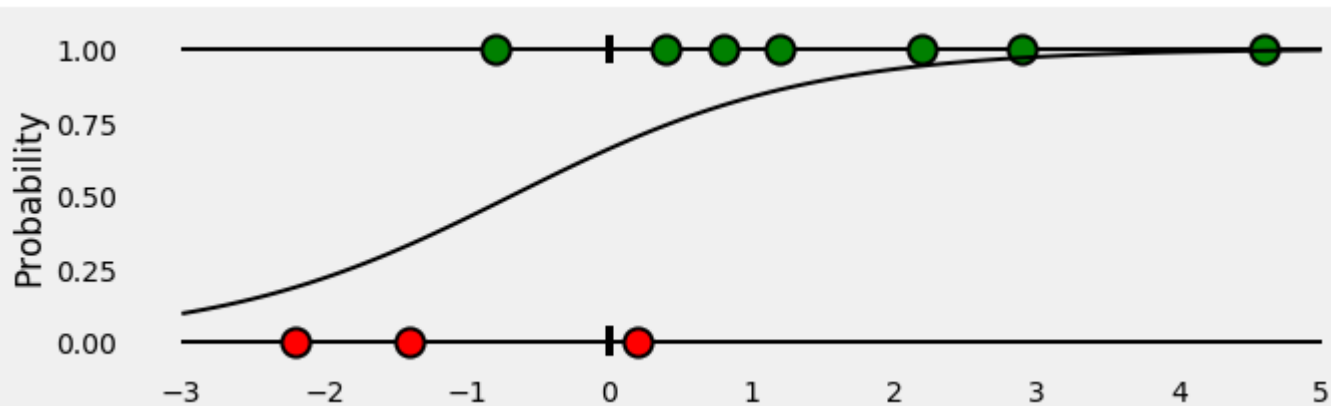
- For each "y=1", it adds $-1 \times \log(g(x_i))$ to the loss.
- For each "y=0", it adds $-1 \times \log(1 - g(x_i))$ to the loss

Binary cross-entropy — the visual way (1)

Credit : <https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a>



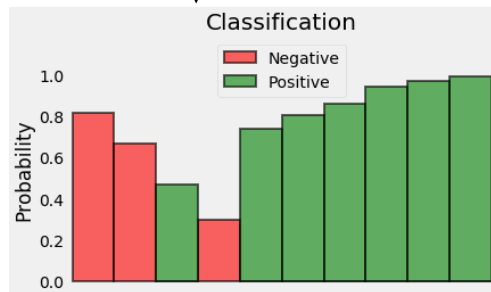
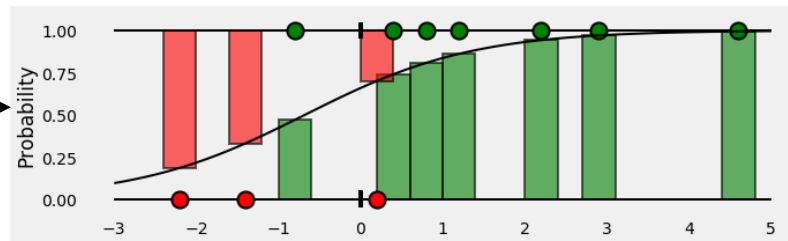
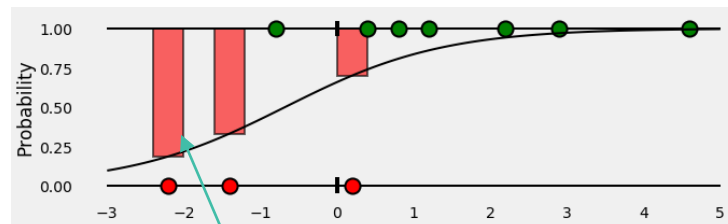
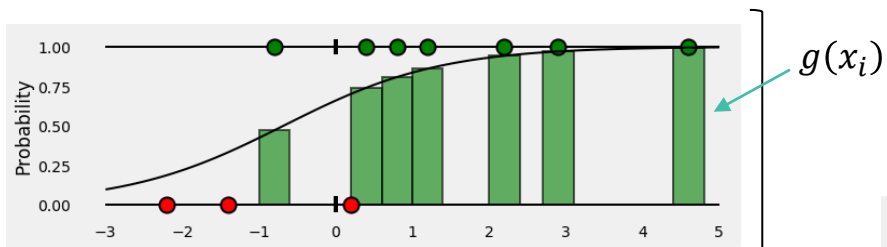
Training data



Logistic regression model

Binary cross-entropy — the visual way (2)

Credit : <https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a>



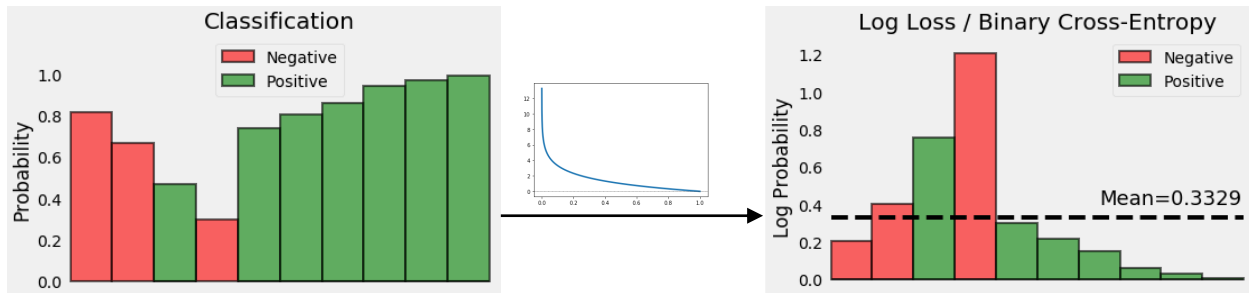
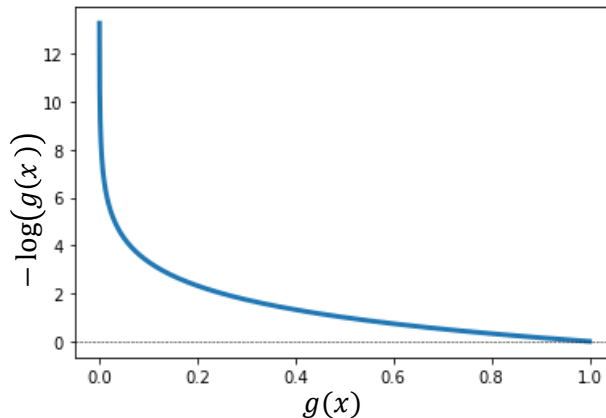
If the **probability** associated with the **true class** is **1.0**, we need its **loss** to be **zero**.
Conversely, if that **probability** is **low**, say **0.01**, we need its **loss** to be **HUGE**!

Binary cross-entropy — the visual way (3)

If the **probability** associated with the **true class** is **1.0**, we need its **loss** to be **zero**.

Conversely, if that **probability is low**, say **0.01**, we need its **loss** to be **HUGE**!

It turns out, taking the **negative log of the probability** suits us well enough for this purpose (*since the log of values between 0.0 and 1.0 is negative, we take the negative log to obtain a positive value for the loss*).



Binary cross-entropy:

$$H(g) = -\frac{1}{N} \sum_{i=1}^N y_i \cdot \log(g(x_i)) + (1 - y_i) \cdot \log(1 - g(x_i))$$

Finally, we compute the **mean of all these losses**.

The **binary cross-entropy** of this example is **0.3329**.

Multi-class cross-entropy loss - Commonly used by neural networks for multi-class problems

$$CE(\mathbf{g}_y(x_i), \mathbf{p}(y|x_i)) = - \sum_y \mathbf{p}(y|x_i) \log(\mathbf{g}_y(x_i))$$



	$\mathbf{g}_y(x_i)$	LABELS
P(Dog x_i)	0.80	1.00
P(Cat x_i)	0.03	0.00
P(Fox x_i)	0.03	0.00
P(Elephant x_i)	0.02	0.00
P(Lion x_i)	0.01	0.00
P(Mouse x_i)	0.00	0.00
P(Bird x_i)	0.03	0.00
P(Pokémon x_i)	0.04	0.00
P(Fish x_i)	0.01	0.00
P(None x_i)	0.03	0.00

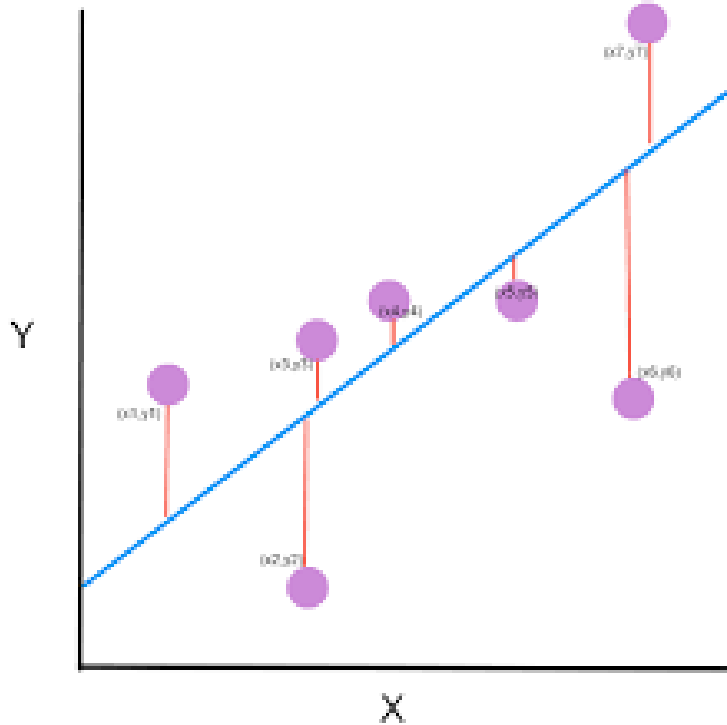
$\mathbf{g}_y(x_i)$
0.7
0.2
0.1
0.9

$\mathbf{p}(y x_i)$
1
0
0
0



$$CE(\mathbf{g}_y(x_i), \mathbf{p}(y|x_i)) = -\log(0.7) = 0.5145$$

Model evaluation: performance measures *in regression*



$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

Question for you:

- can you suggest other error measures in regression?