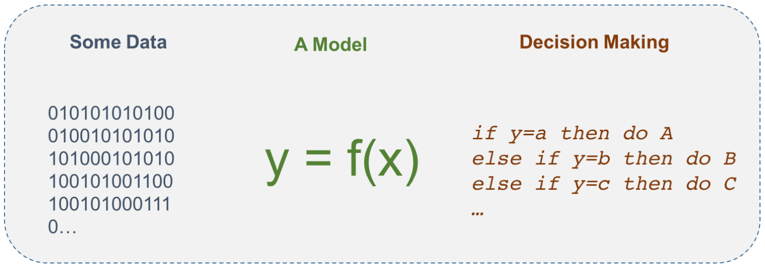
1 What is machine learning?

Machine learning refers to algorithms used to extract patterns from data and learn a mathematical model that could be used by a computer program to make intelligent decisions.



Machine learning is a field of computer science devoted to the design of algorithms that take as input some data set, extract a pattern from it which is typically expressed as a mathematical function, and finally it uses the function as part of a program that automatize a decision making problem. By data we mean digital information such as images, video, audio, text, or any other information that can be stored and accessed by a computer. By mathematical function we consider simple linear functions such as polynomials, non-linear functions such as deep neural networks, probability distribution functions, or even data structures such as trees and graphs. It is common to any of these models that we always need to estimate a set of parameters. By decision making process we are talking about any kind of computer program that is useful for solving, usually complicated tasks such as detecting parasites in blood sample images, classifying phrases written different language, or controlling a drone to accomplish a specific goal.

People involved in ML research and development comes from fields such as: computer science, artificial intelligence, statistics, neuroscience, psychology, robotics, bioinformatics, etc.

Based on the type of data available and the decisions needed, we can talk about three general kinds of machine learning problems:

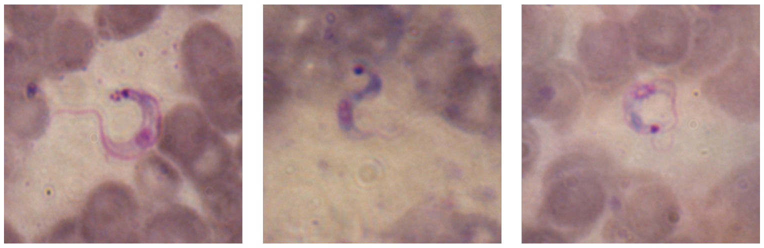
- Supervised Learning (SL)
- Unsupervised Learning (UL)
- Reinforcement Learning (RL)

One easy way to understand the differences between these types of machine learning problems is through the analysis of examples, and this is exactly what we will do in the next sections.

1.1 Supervised learning

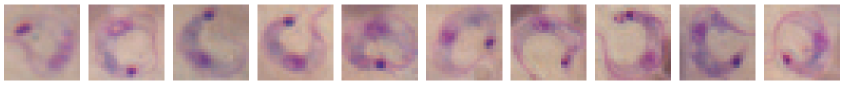
One of the most common machine learning problems is known as supervised learning. In supervised learning we are given a set of examples of how an object looks like and we need to estimate the parameters of a mathematical model capable to capture the patterns of the objects occurring in those examples. By pattern we mean the set of characteristics or regularities that make the object in question to be precisely that object and not other.

For example, consider the problem of detecting Chagas parasites in blood sample images. In this problem we are given a set of sample mages containing Chagas parasites. These samples are our examples of the objects that we need to recognize. In other word our mathematical model should be able to extract the patterns that make a Chagas parasite looks like a Chagas parasite and not other thing, for instance, a white blood cell.

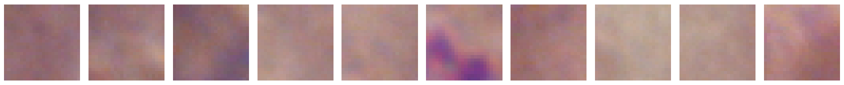


In this kind of machine learning problem in which we need to classify an object, which in this case it happens to be an image of a parasite, in whether the object is the one we are looking for or not, we count with labeled examples. These pairs of object and label are the reason we call this kind of learning a supervised learning problem. In this particular example we will have available a set of images of parasites (positive exanples) and a set of images of non-parasites (negative examples). When the number of possible outputs are only two, like in this case (parasite/non-parasite), we are talking about a binary classification problem.

Positive examples:



Negative examples:



In order to solve the Chagas detection problem we need to develop a mathematical model capable to take as input an image, and output one of the possible classes, also called categories. If we manage to generate such model, then we can program an algorithm to scan the whole image and evaluate each subimage with our model, always asking "is this a Chagas parasite?", and hopefully our model will answer with high precision with a one, whenever we are evaluating a subimage containing a parasite, and it will answer with a zero if the subimage does not contain a parasite at all.

In the classification problem we have function C mapping every input object in the set X to an output class in the set Y. Mathematically speaking we consider that every $\mathbf{x} \in X$ is a vector of real numbers in dimension d and the n outputs $y_i \in Y$ are our labels. When we have only $n = 2$ possible outputs, we are solving a binary classification problem.

$$\begin{aligned} &C : X \to Y, \\ &X \subseteq \mathbb{R}^d, \\ &Y = \{y_1, y_2, \ldots, y_n\} \end{aligned}$$

In a binary classification example one simple option of mathematical model could be a linear combination of features x_1 and x_2, and its posterior evaluation in a sigmoid function $C(z) = 1/(1 + e^{-z})$, which will give us as output a value always between 0 and 1. With the sigmoid function we consider any output value equal or larger than 0.5 as a one label, and any number less than 0.5 as a zero label.

$$C(\mathbf{x}) = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

Another kind of supervised learning problem highly related to the classification one is the regression problem. In regression problems we also count with set of labeled examples, however an important difference is that we do no talk anymore about labels, but simply as a numeric output, which usually is a real number. In this kind of supervised problem we one to predict the value that an object takes, for instance, the price of a house. Therefore, the outputs $y \in Y$ are considered real numbers.

$$\begin{aligned} &R : X \to Y, \\ &X \subseteq \mathbb{R}^d, \\ &Y \subseteq \mathbb{R}.(Continuous) \end{aligned}$$

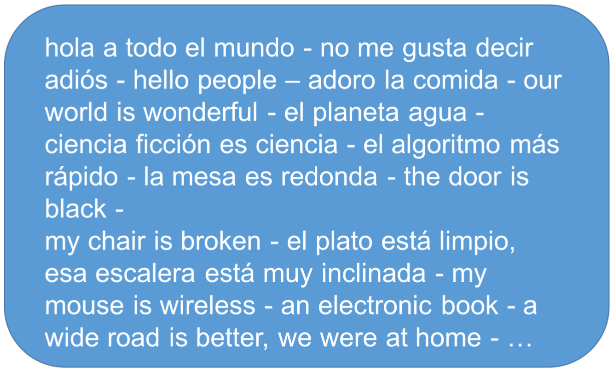
An example of linear model for regression could be the following.

$$R(\mathbf{x}) = w_o + w_1 x + w_2 x^2 + w_3 x^3$$

It is important to point out that in both of the model examples here defined, the one for binary classification and the one for regression, we consider every w_i to be a parameter of the corresponding model.

1.2 Unsupervised learning

In the unsupervised learning problem, we are given examples without their labels. The goal in this kind of problem is to discover patterns that will allow us to group our objects in sets that contain similar objects. This is called clustering. Take for example the problem of clustering text phrases written in two different languages. This is, given a set of text phrases written in two different languages, decide if a new text phrase belongs to one of the languages existing in your set of phrases.



hola a todo el mundo - no me gusta decir adiós - hello people – adoro la comida - our world is wonderful - el planeta agua - ciencia ficción es ciencia - el algoritmo más rápido - la mesa es redonda - the door is black -
my chair is broken - el plato está limpio, esa escalera está muy inclinada - my mouse is wireless - an electronic book - a wide road is better, we were at home - …

The unsupervised learning problem is similar to the classification one, but the data is not labeled this means that we do not know the category of each example. In the clustering problem we have to assign each of our examples to one of possible n existing clusters. The problem is that usually we do not even know the value of n. Therefore, it is common in this kind of machine learning problem to run our algorithms several times to discover which n seems to be the most pausible. Mathematically speaking we define the clustering problems as the following mapping.

$$\begin{aligned}
&C : X \to T,\\
&X \subseteq \mathbb{R}^d,\\
&T = \{t_1, t_2, \ldots, t_n\},
\end{aligned}$$

with unknown n. An example of clustering model could be one defined as the shortest distant to one of n estimated reference values t_i, as follows.

$$t = \arg\min_{t_i} \text{distance}(t_i, x)$$

One important algorithm of this kind is k-means, in which those reference values are known as centroids.

1.3 Reinforcement learning

Given a history of the commands used to control a drone, decide which is the best command to perform in order to avoid a collision with the ground. This is the kind of machine learning problems we face in reinforcement learning. We need to estimate a control function known as policy function. In this family of learning problems we always consider an agent that acts in an environment.

In reinforcement learning the agent must learn to perform a task, which means that it must go from one start state s_{start} to one final state s_{final} of its environment. The execution of the task can be seen as a sequence of actions $a_1, a_2, \ldots, a_n$. In order to learn the right sequence of actions the agent must interact with its environment, selecting at each moment one action a_{t+1} as a function of the current environment state s_t. This is $a_{t+1} = F(s_t)$.

Using Markov decision processes algorithms we can estimate how good a combination of state and action is, and store such values in a lookup table that we call Q-function:

S	A	$Q(s,a)$
1	1	-20
1	2	-23
1	3	-5
1	4	-22
2	1	-20
2	2	-5
2	3	-22
...	...	...
64	4	-24

One way to solve a reinforcement learning problem will involve estimating the Q-function. Some times the number of states and actions make it. impossible to store the Q values in a table and we will need to use a function approximator instead. Once we have estimated this function we can use it to control the system, which in our example is a drone. All we need to do is to perceive the current state of the drone and check in our Q-function which is the action with maximum value for that state. That action is expected to be the optimal for that current state and therefore will be the one which should be executed.

When working with reinforcement learning problems we need to apply the theory of Markov decision processes. A Markov decision process MDP $= (S, A, T, R)$ consists of:

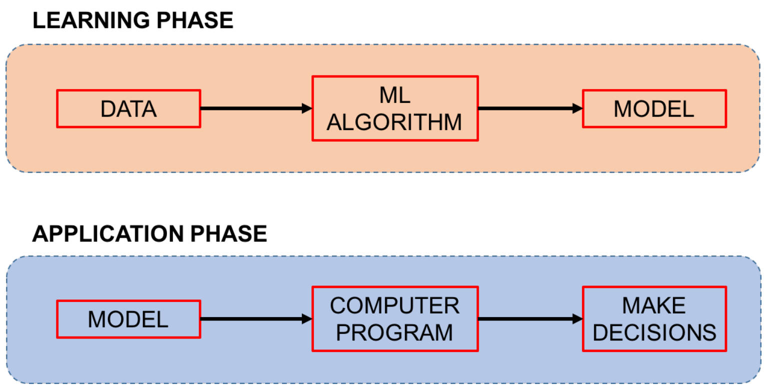
- A set of states $S = \{s_1, s_2, \ldots\}$
- A set of actions $A(s) = \{a_1, a_2, \ldots\}$
- A transition function (dynamics of the environment) $T : S \times A \rightarrow S$
- A reward function $R : S \rightarrow \mathbb{R}$

Once we have identified or defined these element of our MDP, we can apply different algorithms to estimate our Q-function. There are many ways to solve reinforcement learning problems, but most of them if not all, need to estimate the parameters of a policy function, which is the one that will help us control the system. In this example we are considering a policy function generated from the Q-function.

1.4 Practical aspects of machine learning

Developing and deploying machine learning algorithms usually involves at least two phases, namely, the learning phase and the application phase.

In the learning phase we need to estimate the parameters of our mathematical model, as we minimize some error measure, usually defined as a loss function. During the process of estimating the parameters, a process commonly known as training of the model, we must take especial care in minimizing the prediction error of our model with a set of examples never used during the training step, this set is called the test set. Since the test error is important to define the generalization error of our model or expected performance, there are different methods in which we can train and test our models. We will have time later to study those methods. The important point now is to understand that training and testing are the core tasks in the learning phase.



Once we have finished with the learning phase, we get to the application phase. In the application phase, the challenges are usually of an engineering nature. This phase is also called deployment of the machine learning system. In this phase our mathematical model is programmed and connected as a component of a larger software system. Leaving aside the difficulties of engineering the optimal way to deploy the machine learning model, the most important matter that we should consider in this phase is that very often, the model does not reach the same level of performance it reached during training and testing. This can be the consequence of multiple circumstances ignored during the learning phase. Therefore, at this point, we will need to discover the reason of the suboptimal performance of our model, go back to the learning phase, and make some modifications in the data set, the mathematical model, or both, with the goal of generating a better machine learning model. The learning and application phases are usually repeated until the machine learning system performs as needed.

Designing, training, testing, fine-tuning and deploying machine learning systems require technical knowledge of mathematics, data structures, algorithms complexity, and at least one programming language. If we were to devise methodology for this task, it could be summarize in the following five steps for developing machine learning systems:

1. Understanding the kind of machine learning problem (SL, UL, RL)
2. Understanding the mathematics (basic statistics, probability, calculus, linear algebra)
3. Understanding the algorithms (data structures, complexity)
4. Coding (Matlab/Octave/Python)
5. Experimenting (running programs and plotting graphs)