Chapter 1

A Non-technical Introduction to Machine Learning

Olivier Colliot

Abstract

This chapter provides an introduction to machine learning for a non-technical readership. Machine learning is an approach to artificial intelligence. The chapter thus starts with a brief history of artificial intelligence in order to put machine learning into this broader scientific context. We then describe the main general concepts of machine learning. Readers with a background in computer science may skip this chapter.

Key words Machine learning, Artificial intelligence, Supervised learning, Unsupervised learning

1 Introduction

Machine learning (ML) is a scientific domain which aims at allowing computers to perform tasks without being explicitly programmed to do so [1]. To that purpose, the computer is trained using the examination of examples or experiences. It is part of a broader field of computer science called *artificial intelligence* (AI) which aims at creating computers with abilities that are characteristic of human or animal intelligence. This includes tasks such as perception (the ability to recognize images or sounds), reasoning, decision-making, or creativity. Emblematic tasks which are easy to perform for a human and are inherently difficult for a computer are, for instance, recognizing objects, faces, or animals in photographs or recognizing words in speech. On the other hand, there are also tasks which are inherently easy for a computer and difficult for a human, such as computing with large numbers or memorizing exactly huge amounts of text. Machine learning is the AI technique that has achieved the most impressive successes over the past years. However, it is not the only approach to AI, and conceptually different approaches also exist.

Machine learning also has close ties to other scientific fields. First, it has evident strong links to statistics. Indeed, most machine learning approaches exploit statistical properties of the data. Moreover, some classical approaches used in machine learning were actually invented in statistics (for instance, linear or logistic regression). Nowadays, there is a constant interplay between progress in statistics and machine learning. ML has also important ties to signal and image processing, ML techniques being efficient for many applications in these domains and signal/image processing concepts being often key to the design or understanding of ML techniques. There are also various links to different branches of mathematics, including optimization and differential geometry. Besides, some inspiration for the design of ML approaches comes from the observation of biological cognitive systems, hence the connections with cognitive science and neuroscience. Finally, the term *data science* has become commonplace to refer to the use of statistical and computational methods for extracting meaningful patterns from data. In practice, machine learning and data science share many concepts, techniques, and tools. Nevertheless, data science puts more emphasis on the discovery of knowledge from the data, while machine learning focuses on solving tasks.

This chapter starts by providing a few historical landmarks regarding artificial intelligence and machine learning (Subheading 2). It then proceeds with the main concepts of ML which are foundational to understand other chapters of this book.

2 A Bit of History

As a scientific endeavor, artificial intelligence is at least 80 years old. Here, we provide a very brief overview of this history. For more details, the reader may refer to [2]. A non-exhaustive timeline of AI is shown in Fig. 1.

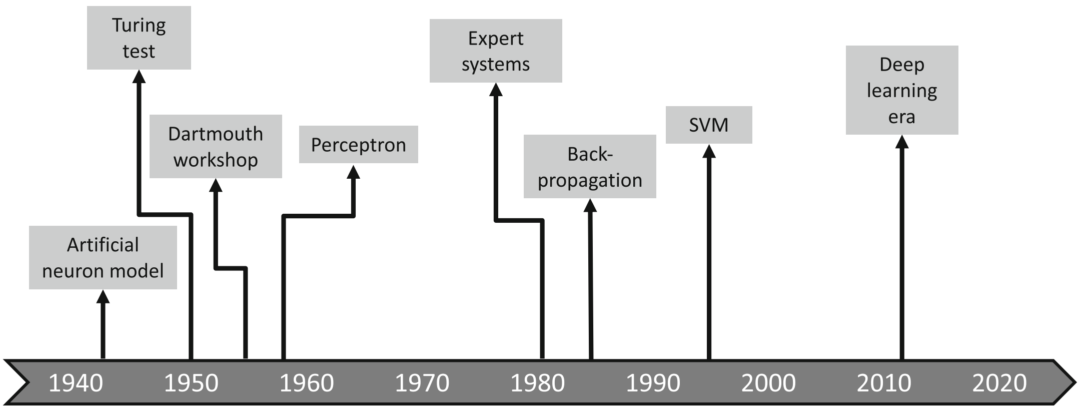


Fig. 1 A brief timeline of AI with some of the landmark advances

Even if this is debatable, one often considers AI to emerge in the 1940s–1950s with a series of important concepts and events. In 1943, the neurophysiologist Warren McCulloch and the logician Walter Pitts proposed an artificial neuron model, which is a mathematical abstraction of a biological neuron [3], and showed that sets of neurons can compute logical operations. In 1948, the mathematician and philosopher Norbert Wiener coined the term "cybernetics" [4] to designate the scientific study of control and communication in humans, animals, and machines. This idea that such processes can be studied within the same framework in both humans/animals and machines is a conceptual revolution. In 1949, the psychologist Donald Hebb [5] described a theory of learning for biological neurons which was later influential in the modification of the weights of artificial neurons.

In 1950, Alan Turing, one of the founders of computer science, introduced a test (the famous "Turing test") for deciding if a machine can think [6]. Actually, since the question *can a machine think?* is ill-posed and depends on the definition of thinking, Turing proposed to replace it with a practical test. The idea is that of a game in which an interrogator is given the task of determining which of two players A and B is a computer and which is a human (by using only responses to written questions). In 1956, the mathematician John McCarthy organized what remained as the famous Dartmouth workshop and which united ten prominent scientists for 2 months (among which were Marvin Minsky, Claude Shannon, Arthur Samuel, and others). This workshop is more important by its scientific program than by its outputs. Let us reproduce here the first sentences of the proposal written by McCarthy et al. [7] as we believe that they are particularly enlightening on the prospects of artificial intelligence:

> We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

There was no major advance made at the workshop, although a reasoning program, able to prove theorems, was presented by Allen Newell and Herbert Simon [8] at this occasion. This can be considered as the start of symbolic AI (we will come back later on the two main families of AI: symbolic and connexionist). Let us end the 1950s with the invention, in 1958, of the perceptron by Frank Rosenblatt [9], whose work was built upon the ideas of McCulloch, Pitts, and Hebb. The perceptron was the first actual artificial neuron. It was able to recognize images. This is an important landmark for several reasons. The perceptron, with some modifications, is still the building block of modern deep learning algorithms. To mimic an artificial neuron (Fig. 2), it is composed of a set of inputs (which correspond to the information entering the synapses) x_i, which are linearly combined and then go through a non-linear function g to produce an output y. This was an important advance at the time, but it had strong limitations, in particular its inability to discriminate patterns which are not linearly separable. More generally, in the field of AI as a whole, unreasonable promises had been made, and they were not delivered: newspapers were writing about upcoming machines that could talk, see, write, and think; the US government funded huge programs to design automatic translation programs, etc. This led to a dramatic drop in research funding and, more generally, in interest in AI. This is often referred to as the first AI winter (Fig. 3).

Even though research in AI continued, it was not before the early 1980s that real-world applications were once again considered possible. This wave was that of expert systems [10], which are a type of symbolic AI approach but with domain-specific knowledge. Expert systems led to commercial applications and to a real boom in the industry. A specific programming language, called LISP [11], became dominant for the implementation of expert systems. Companies started building LISP machines, which were dedicated computers with specific architecture tailored to execute LISP efficiently. One cannot help thinking of a parallel with current hardware dedicated to deep learning. However, once again, expectations were not met. Expert systems were very large and complex sets of rules. They were difficult to maintain and update. They also had poor performances in perception tasks such as image and speech recognition. Academic and industrial funding subsequently dropped. This was the second AI winter.

At this stage, it is probably useful to come back to the two main families of AI: symbolic and connexionist (Fig. 4). They had important links at the beginning (see, e.g., the work of McCulloch and Pitt aiming to perform logical operations using artificial neurons), but they subsequently developed separately. In short, these two families can be described as follows. The first operates on symbols through sets of logical rules. It has strong ties to the domain of predicate logic. Connexionism aims at training networks of artificial neurons. This is done through the examination of training examples. More generally, it is acceptable to put most machine learning methods within the connexionist family, even though they don't rely on artificial neuron models, because their underlying principle is also to exploit statistical similarities in the training data. For a more detailed perspective on the two families of AI, the reader can refer to the very interesting (and even entertaining!) paper of Cardon et al. [12].

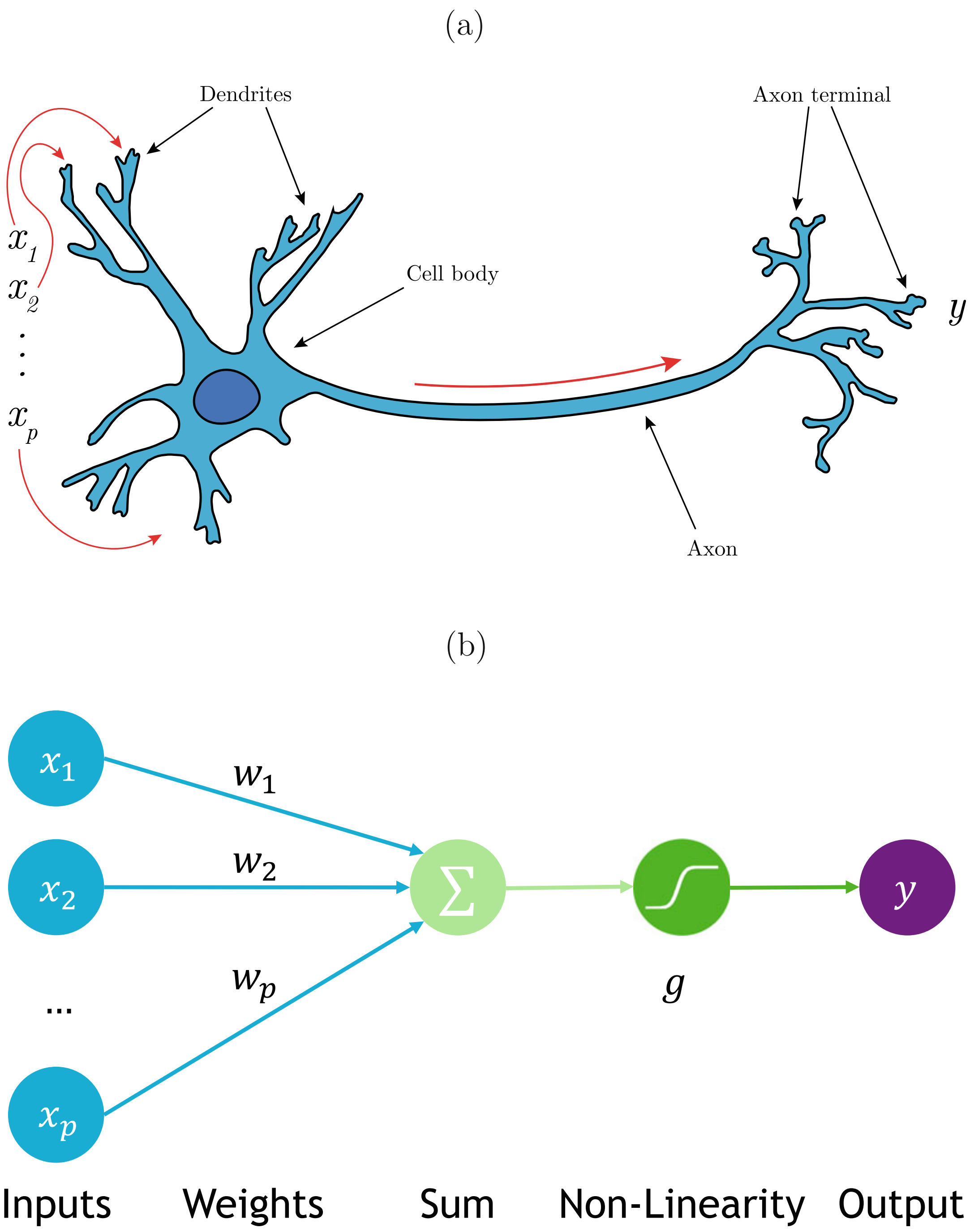


Fig. 2 (**a**) Biological neuron. The synapses form the input of the neuron. Their signals are combined, and if the result exceeds a given threshold, the neuron is activated and produces an output signal which is sent through the axon. (**b**) The perceptron: an artificial neuron which is inspired by biology. It is composed of the set of inputs (which correspond to the information entering the synapses) x_i, which are linearly combined with weights w_i and then go through a non-linear function g to produce an output y. Image in panel (**a**) is courtesy of Thibault Rolland

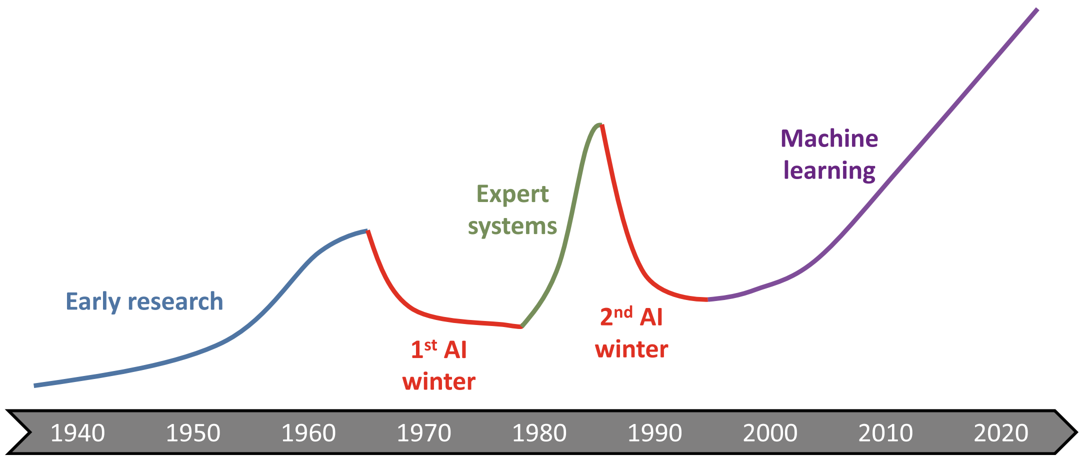


Fig. 3 Summers and winters of AI

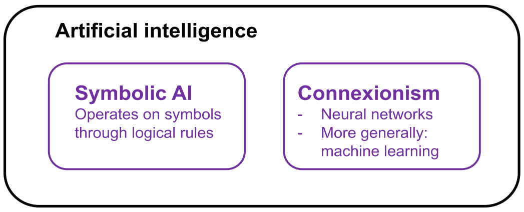

Artificial intelligence
Symbolic AI
Operates on symbols through logical rules
Connexionism
- Neural networks
- More generally: machine learning

Fig. 4 Two families of AI. The symbolic approach operates on symbols through logical rules. The connexionist family actually not only encompasses artificial neural networks but more generally machine learning approaches

Let us come back to our historical timeline. The 1980s saw a rebirth of connexionism and, more generally, the start of the rise of machine learning. Interestingly, it is at that time that two of the main conferences on machine learning started: the International Conference on Machine Learning (ICML) in 1980 and Neural Information Processing Systems (NeurIPS, formerly NIPS) in 1987. It had been known for a long time that neural networks with multiple layers (as opposed to the original perceptron with a single layer) (Fig. 5) could solve non-linearly separable problems, but their training remained difficult. The back-propagation algorithm for training multilayer neural networks was described by David Rumelhart, Geoffrey Hinton, and Ronald Williams [13] in 1986, as well as by Yann LeCun in 1985 [14], who also refined the procedure in his PhD thesis published in 1987. This idea had actually been explored since the 1960s, but it was only in the 1980s that it was efficiently used for training multilayer neural networks. Finally, in 1989, Yann LeCun proposed the convolutional neural network [15], an architecture inspired by the organization of the visual cortex, whose principle is still at the core of

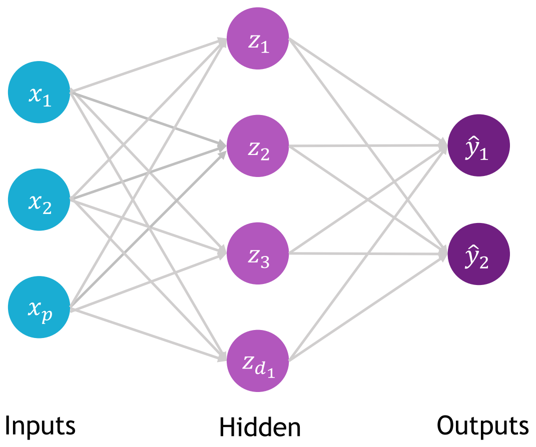


Fig. 5 A multilayer perceptron model (here with only one hidden layer, but there can be many more)

state-of-the-art algorithms for many image processing and recognition tasks. Multilayer neural networks demonstrated their utility in several real-world applications such as digit recognition on checks and ZIP codes [16]. Nevertheless, they would not become the dominant machine learning approach until the 2010s. Indeed, at the time, they required considerable computing power for training, and there was often not enough training data.

During the 1980s and 1990s, machine learning methods continued to develop. Interestingly, connections between machine learning and statistics increased. We are not going to provide an overview of the history of statistics, but one should note that many statistical methods such as linear regression [17], principal component analysis [18], discriminant analysis [19], or decision trees [20] can actually be used to solve machine learning tasks such as automatic categorization of objects or prediction. In the 1980s, decision trees witnessed important developments (see, e.g., the ID3 [21] and CART [21] algorithms). In the 1990s, there were important advances in the statistical theory of learning (in particular, the works of Vladimir Vapnik [22]). A landmark algorithm developed at that time was the support vector machine (SVM) [23] which worked well with small training datasets and could handle non-linearities through the use of kernels. The machine learning field continued to expand through the 2000s and 2010s, with new approaches but also more mature software packages such as scikit-learn [24]. More generally, it is actually important to have in mind that what is currently called AI owes more to statistics (and other mathematical fields such as optimization in particular) than to modeling of brain circuitry and that even approaches that take inspiration from neurobiology can actually be viewed as complex statistical machineries.

2012 saw the revival of neural networks and the beginning of the era of deep learning. It was undoubtedly propelled by the considerable improvement obtained on the ImageNet recognition challenge which contains 14 million natural images belonging to 20,000 categories. The solution, proposed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton [25], was a convolutional neural network with a large number of layers, hence the term deep learning. The building blocks of this solution were already present in the 1980s, but there was not enough computing power nor large training datasets for them to work properly. In the interval, things had changed. Computers had become exponentially more powerful, and, in particular, the use of graphical processing units (GPU) considerably sped up computations. The expansion of the Internet had provided massive amounts of data of various sorts such as texts and images. In the subsequent years, deep learning [26] approaches became increasingly sophisticated. In parallel, efficient and mature software packages including TensorFlow [27], PyTorch [28], or Keras [29], whose development is supported by major companies such as Google and Facebook, enable deep learning to be used more easily by scientists and engineers.

Artificial intelligence in medicine as a research field is about 50 years old. In 1975, an expert system, called MYCIN, was proposed to identify bacteria causing various infectious diseases [30]. More generally, there was a growing interest in expert systems for medical applications. Medical image processing also quickly became a growing field. The first conference on Information Processing in Medical Imaging (IPMI) was held in 1977 (it existed under a different name since 1969). The first SPIE Medical Image Processing conference took place in 1986, and the Medical Image Computing and Computer-Assisted Intervention (MICCAI) conference started in 1998. Image perception tasks, such as segmentation or classification, soon became among the key topics of this field, even though the methods came in majority from traditional image processing and not from machine learning. In the 2010s, machine learning approaches became dominant for medical image processing and more generally in artificial intelligence in medicine.

To conclude this part, it is important to be clear about the different terms, in particular those of artificial intelligence, machine learning, and deep learning (Fig. 6). Machine learning is *one* approach to artificial intelligence, and other radically different approaches exist. Deep learning is a specific type of machine learning approach. It has recently obtained impressive results on some types of data (in particular, images and text), but this does not mean that it is the universal solution to all problems. As we will see in this book, there are tasks for which other types of approaches perform best.

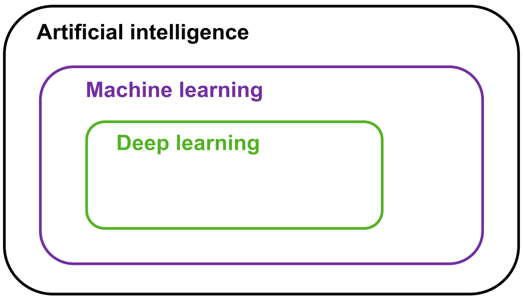


Fig. 6 Artificial intelligence, machine learning, and deep learning are not synonymous. Deep learning is a type of machine learning which involves neural networks with a large number of hidden layers. Machine learning is one approach to artificial intelligence, but other approaches exist

3 Main Machine Learning Concepts

As aforementioned, machine learning aims at making a computer capable of performing a task without explicitly being programmed for that task. More precisely, it means that one will not write a sequence of instructions that will directly perform the considered task. Instead, one will write a program that allows the computer to learn how to perform the task by examining examples or experiences. The output of this learning process is a computer program itself that performs the desired task, but this program was not explicitly written. Instead, it has been learned automatically by the computer.

In 1997, Tom Mitchell gave a more precise definition of a **well-posed machine learning problem** [31]:

> A computer program is said to learn from **experience E** with respect to some **task T** and some **performance measure P**, if its performance at task T, as measured by P, improves with experience E.

He then provides the example of a computer that learns to play checkers: task T is playing checkers, performance measure P is the proportion of games won, and the training experience E is playing checker games against itself. Very often, the experience E will not be an actual action but the observation of a set of examples, for instance, a set of images belonging to different categories, such as photographs of cats and dogs, or medical images containing tumors or without lesions. Please refer to Box 1 for a summary.

Box 1: Definition of machine learning
Machine learning definition [31]:

> a computer program is said to learn from **experience E** with respect to some **task T** and some **performance measure P**, if its performance at task T, as measured by P, improves with experience E.

(continued)

Box 1 (continued)

Example: learning to detect tumors from medical images

- **Task T**: detect tumors from medical image
- **Performance measure P**: proportion of tumors correctly identified
- **Experience E**: examining a dataset of medical images where the presence of tumors has been annotated

3.1 Types of Learning

One usually considers three main types of learning: supervised learning, unsupervised learning, and reinforcement learning (Box 2). In both supervised and unsupervised learning, the experience E is actually the inspection of a set of examples, which we will refer to as *training examples* or *training set.*

Box 2: Supervised, Unsupervised, and Reinforcement learning

- **Supervised learning.** Learns from labeled examples, i.e., examples for which the output that we are trying to learn is known
  - Example 1. The task is computer-aided diagnosis (a classification problem), and the label can be the diagnosis of each patient, as defined by an expert physician.
  - Example 2. The task is the prediction of the age of a person from a set of biological variables (e.g., a brain MRI). This is a regression problem. The label is the true age of a given person in the training set.
- **Unsupervised learning.** Learns from unlabeled examples
  - Example 1. Given a large set of newspaper articles, automatically cluster them into groups dealing with the same topic based only on the text of the article. The topics can, for example, be economics, politics, or international affairs. The topics are not known a priori.
  - Example 2. Given a set of patients with autism spectrum disorders, the aim is to discover a cluster of patients that share the same characteristics. The clusters are not known a priori. Examples 1 and 2 will be referred to as clustering tasks.
  - Example 3. Given a large set of medical characteristics (various biological measurements, clinical and cognitive tests, medical images), find a small set of variables that best explain the variability of the dataset. This is a dimensionality reduction problem.

(continued)

Box 2 (continued)

- **Reinforcement learning.** Learns by iteratively performing actions to maximize some reward
  - Classical approach used for learning to play games (chess, go, etc.) or in the domain of robotics
  - Currently few applications in the domain of brain diseases

3.1.1 Supervised Learning

In supervised learning, the machine learns to perform a task by examining a set of examples for which the output is known (i.e., the examples have been labeled). The two most common tasks in supervised learning are classification and regression (Fig. 7). Classification aims at assigning a category for each sample. The examples can, for instance, be different patients, and the categories are the different possible diagnoses. The outputs are thus discrete. Examples of common classification algorithms include logistic regression (in spite of its name, it is a classification method), linear discriminant analysis, support vector machines, random forest classifiers, and deep learning models for classification. In regression, the output is a continuous number. This can be, for example, the future clinical score of a patient that we are trying to predict. Examples of common regression methods include simple or multiple linear regression, penalized regression, and random forest regression. Finally, there are many other tasks that can be framed as a supervised learning problem, including, for example, data synthesis, image segmentation, and many others which will be described in other chapters of this book.

3.1.2 Unsupervised Learning

In unsupervised learning, the examples are not labeled. The two most common tasks in unsupervised learning are clustering and dimensionality reduction (Fig. 8). Clustering aims at discovering groups within the training set, but these groups are not known a priori. The objective is to find groups such that members of the same group are similar, while members of different groups are dissimilar. For example, one can aim to discover disease subtypes which are not known a priori. Some classical clustering methods are *k*-means or spectral clustering, for instance. Dimensionality reduction aims at finding a space of variables (of lower dimension than the input space) that best explain the variability of the training data, given a larger set of input variables. This produces a new set of variables that, in general, are not among the input variables but are combinations of them. Examples of such methods include principal component analysis, Laplacian eigenmaps, or variational autoencoders.

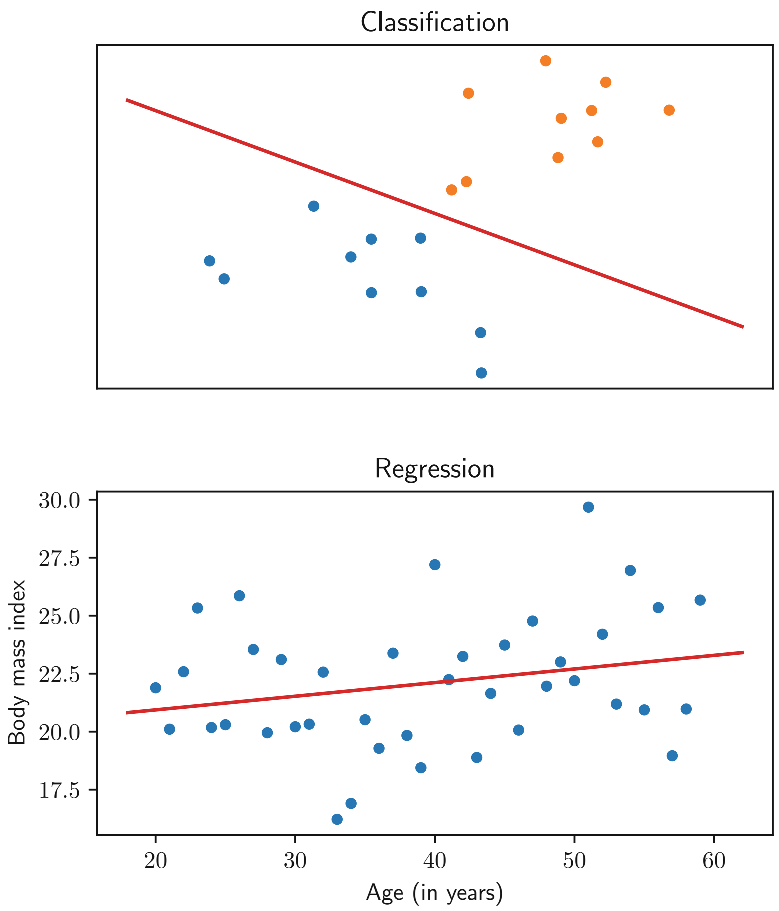


Fig. 7 Two of the main supervised learning tasks: classification and regression. The upper panel presents a classification task which aims at linearly separating the orange and the blue class. Each sample is described by two variables. The lower panel presents a linear regression task in which the aim is to predict the body mass index from the age of a person. *Figure courtesy of Johann Faouzi*

3.1.3 Reinforcement Learning

In reinforcement learning, the machine will take a series of actions in order to maximize a reward. This can, for example, be the case of a machine learning to play chess, which will play games against itself in order to maximize the number of victories. These methods are widely used for learning to play games or in the domain of robotics. So far, they have had few applications to brain diseases and will not be covered in the rest of this book.

3.1.4 Discussion

Unsupervised learning is obviously attractive because it does not require labels. Indeed, acquiring labels for a training set is usually time-consuming and expensive because the labels need to be assigned by a human. This is even more problematic in medicine because the labels must be provided by experts in the field. It is thus in principle attractive to adopt unsupervised strategies, even for

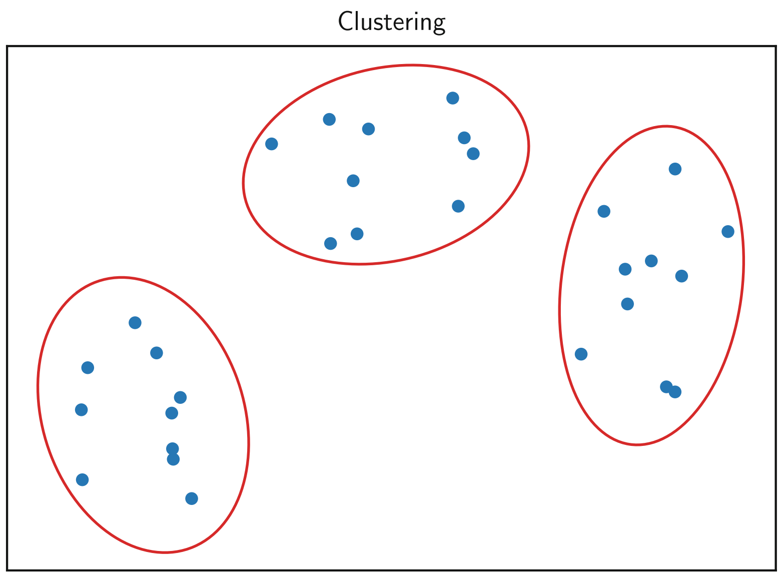


Fig. 8 Clustering task. The algorithm automatically identifies three groups (corresponding to the red circles) from unlabeled examples (the blue dots). The groups are not known a priori. *Figure courtesy of Johann Faouzi*

tasks which could be framed as supervised learning problems. Nevertheless, up to now, the performances of supervised approaches are often vastly superior in many applications. However, in the past years, an alternative strategy called self-supervised learning, where the machine itself provides its own supervision, has emerged. This is a promising approach which has already led to impressive results in different fields such as natural language processing in particular [32–34].

3.2 Overview of the Learning Process

In this section, we aim at formalizing the main concepts underlying most supervised learning methods. Some of these concepts, with modifications, also extend to unsupervised cases.

The task that we will consider will be to provide an output, denoted as y, from an input given to the computer, denoted as x. At this moment, the nature of x does not matter. It can, for example, be any possible photograph as in the example presented in Fig. 9. It could also be a single number, a series of numbers, a text, etc. For now, the nature of y can also be varied. Typically, in the case of regression, it can be a number. In the case of classification, it corresponds to a label (for instance, the label "cat" in our example). For now, you do not need to bother about how these data (images, labels, etc.) are represented in a computer. For those without a background in computer science, this will be briefly covered in Subheading 3.3.

Learning will aim at finding a function f that can transform x into y, that is, such that $y = f(x)$. For now, f can be of any type—

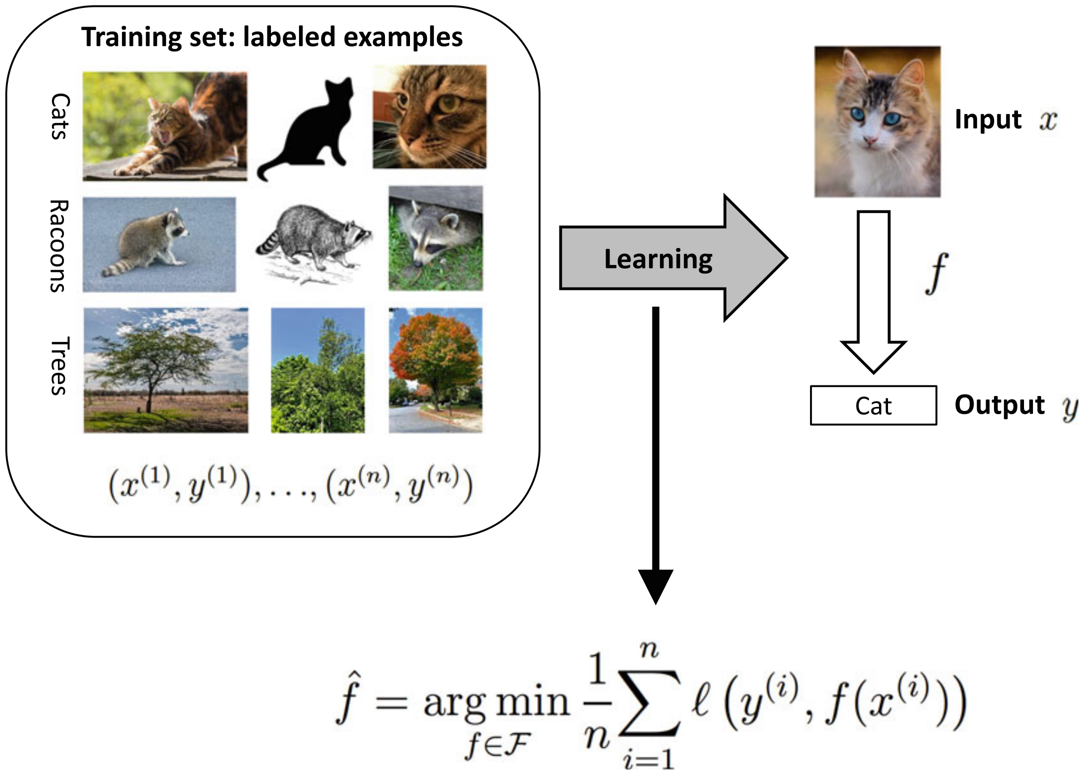

Training set: labeled examples
Cats
Racoons
Trees
$(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$
Learning
Input x
f
Cat
Output y
$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell\left(y^{(i)}, f(x^{(i)})\right)$

Fig. 9 Main concepts underlying supervised learning, here in the case of classification. The aim is to be able to recognize the content of a photograph (the input x) which amounts to assigning it a label (the output y). In other words, we would like to have a function f that transforms x into y. In order to find the function f, we will make use of a training set $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$ (which in our case is a set of photographs which have been labeled). *All images come from https:/commons.wikimedia.org/ and have no usage restriction*

just imagine it as an operation that can associate a given x with a given y. In Chap. 3, the functions f will be artificial neural networks. Learning aims at finding a function f which will provide the correct output for each given input. Let us call the loss function and denote ℓ a function that measures the error that is made by the function f. The loss function takes two arguments: the true output y and the predicted output $f(x)$. The lower the loss function value, the closer the predicted output is to the true output. An example of loss function is the classical least squares loss $\ell(y, f(x)) = (y - f(x))^2$, but many others exist. Ideally, the best function f would be the one that produces the minimal error for any possible input x and associated output y, not only those which we have at our disposal, but any other possible new data. Of course, we do not have any possible data at our disposal. Thus, we are going to use a set of data called the training set. In supervised learning, this set is labeled, i.e., for each example in this set, we know the value of both x and y. Let us denote as $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$ the n examples of the training set which are *n* pairs of inputs and outputs. We are now going to search for the function *f* that makes the minimum error over the *n* samples of the training set. In other words, we are looking for the function which minimizes the average error over the training set. Let us call this average error the cost function:

$$J(f) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(y^{(i)}, f(x^{(i)})\right)$$

Learning will then aim at finding the function $\hat{f}$ which minimizes the cost function:

$$\hat{f} = \underset{f \in \mathcal{F}}{\arg\min}\ \frac{1}{n}\sum_{i=1}^{n} \ell\left(y^{(i)}, f(x^{(i)})\right)$$

In the above equation, *argmin* indicates that we are interested in the function *f* that minimizes the cost *J*(*f*) and not in the value of the cost itself. $\mathcal{F}$ is the space that contains all admissible functions. $\mathcal{F}$ can, for instance, be the set of linear functions or the set of neural networks with a given architecture.

The procedure that will aim at finding *f* that minimizes the cost is called an *optimization procedure*. Sometimes, the minimum can be find analytically (i.e., by directly solving an equation for *f*), but this will rarely be the case. In other cases, one will resort to an iterative procedure (i.e., an algorithm): the function *f* is iteratively modified until we find the function which minimizes the cost. There are cases where we will have an algorithm that is guaranteed to find the global minimum and others where one will only find a local minimum.

Minimizing the errors on the training set does not guarantee that the trained computer will perform well on new examples which were not part of the training set. A first reason may be that the training set is too different from the general population (for instance, we have trained a model on a dataset of young males, and we would like to apply it to patients of any gender and age). Another reason is that, even if the training set characteristics follow those of the general population, the learned function *f* may be too specific to the training set. In other words, it has learned the training set "by heart" but has not discovered a more general rule that would work for other examples. This phenomenon is called *overfitting* and often arises when the dimensionality of the data is too high (there are many variables to represent an input), when the training set is too small, or when the function *f* is too flexible. A way to prevent overfitting will be to modify the cost function so that it not only represents the average error across training samples but also constrains the function *f* to have some specific properties.

Table 1
Example where the input is a series of number. Here each patient is characterized by several variables

	Age (years)	Height (cm)	Weight (kg)
Patient 1	52.5	172	52
Patient 2	75.1	182	78
Patient 3	32.7	161	47
Patient 4	45	190	92

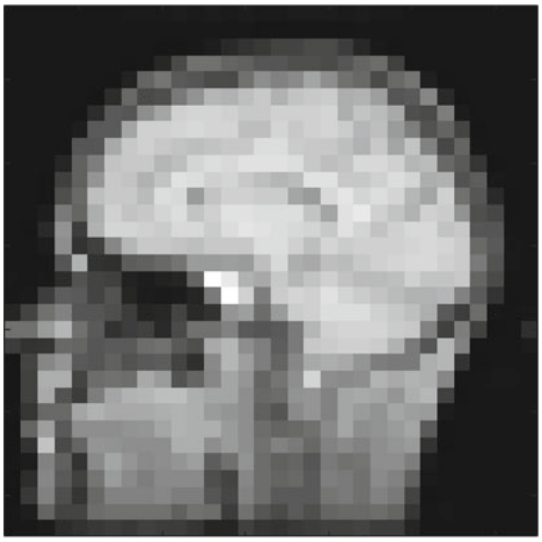
3.3 Inputs and Features

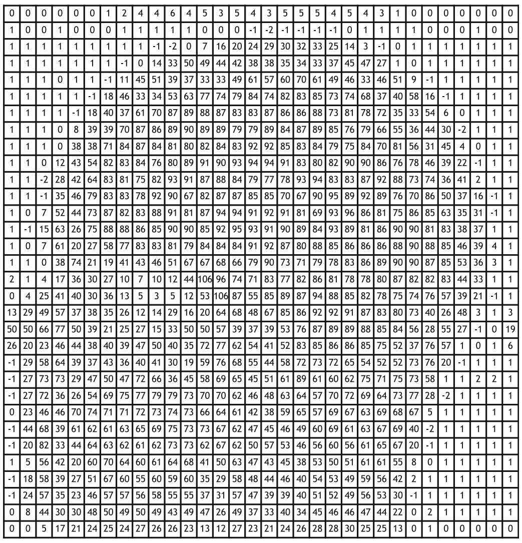
In the previous section, we made no assumption on the nature of the input x. It could be an image, a number, a text, etc.

The simplest form of input that one can consider is when x is a single number. Examples include age, clinical scores, etc. However, for most problems, characterization of a patient cannot be done with a single number but requires a large set of measurements (Table 1). In such a case, the input can be a series of numbers $x_1, \ldots, x_p$ which can be arranged into a vector:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_p \end{bmatrix}$$

However, there are cases where the input is not a vector of numbers. This is the case when the input is a medical image, a text, or a DNA sequence, for instance. Of course, in a computer, everything is stored as numbers. An image is an array of values representing the grayscale intensity of each pixel (Fig. 10). A text is a sequence of characters which are each coded as a number. However, unlike in the example presented in Table 1, these numbers are not meaningful by themselves. For this reason, a common approach is to extract features, which will be series of numbers that meaningfully represent the input. For example, if the input is a brain magnetic resonance image (MRI), relevant features could be the volumes of different anatomical regions of the brain (this specific process is done using a technique called image segmentation which is covered in another chapter). This would result in a series of numbers that would form an input vector. The development of efficient methods for extracting meaningful features from raw data is important in machine learning. Such an approach is often called *feature engineering*. Deep learning methods allow for avoiding extracting features by providing an end-to-end approach from the raw data to the output. In some areas, this has made feature engineering less important, but there are still applications where the so-called handcrafted features are competitive with deep learning methods.





0	0	0	0	0	0	1	2	4	4	6	4	5	3	5	4	3	5	5	5	4	5	4	3	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1	1	1	1	1	0	0	0	-1	-2	-1	-1	-1	-1	0	1	1	1	1	0	0	0	1	0	0
1	1	1	1	1	1	1	1	1	-1	-2	0	7	16	20	24	29	30	32	33	25	14	3	-1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	-1	0	14	33	50	49	44	42	38	38	35	34	33	37	45	47	27	1	0	1	1	1	1	1	1
1	1	1	0	1	1	-1	11	45	51	39	37	33	33	49	61	57	60	70	61	49	46	33	46	51	9	-1	1	1	1	1	1
1	1	1	1	1	-1	18	46	33	34	53	63	77	74	79	84	74	82	83	85	73	74	68	37	40	58	16	-1	1	1	1	1
1	1	1	1	-1	18	40	37	61	70	87	89	88	87	83	83	87	86	86	88	73	81	78	72	35	33	54	6	0	1	1	1
1	1	1	0	8	39	39	70	87	86	89	90	89	89	79	79	89	84	87	89	85	76	79	66	55	36	44	30	-2	1	1	1
1	1	1	0	38	38	71	84	87	84	81	80	82	84	83	92	92	85	83	84	79	75	84	70	81	56	31	45	4	0	1	1
1	1	0	12	43	54	82	83	84	76	80	89	91	90	93	94	94	91	83	80	82	90	90	86	76	78	46	39	22	-1	1	1
1	1	-2	28	42	64	83	81	75	82	93	91	87	88	84	79	77	78	93	94	83	83	87	92	88	73	74	36	41	2	1	1
1	1	-1	35	46	79	83	83	78	92	90	67	82	87	87	85	85	70	67	90	95	89	92	89	76	70	86	50	37	16	-1	1
1	0	7	52	44	73	87	82	83	88	91	81	87	94	94	91	92	91	81	69	93	96	86	81	75	86	85	63	35	31	-1	1
1	-1	15	63	26	75	88	88	86	85	90	90	85	92	95	93	91	90	89	84	93	89	81	86	90	90	81	83	38	37	1	1
1	0	7	61	20	27	58	77	83	83	81	79	84	84	84	91	92	87	80	88	85	86	86	86	88	90	88	85	46	39	4	1
1	1	0	38	74	21	19	41	43	46	51	67	67	68	66	79	90	73	71	79	78	83	86	89	90	90	87	85	53	36	3	1
2	1	4	17	36	30	27	10	7	10	12	44	106	96	74	71	83	77	82	86	81	78	78	80	87	82	82	83	44	33	1	1
0	4	25	41	40	30	36	13	5	3	5	12	53	106	87	55	85	89	87	94	88	85	82	78	75	74	76	57	39	21	-1	1
13	29	49	57	37	38	35	26	12	14	29	16	20	64	68	48	67	85	86	92	92	91	87	83	80	73	40	26	48	3	1	3
50	50	66	77	50	39	21	25	27	15	33	50	50	57	39	37	39	53	76	87	89	89	88	85	84	56	28	55	27	-1	0	19
26	20	23	46	44	38	40	39	47	50	40	35	72	77	62	54	41	52	83	85	86	86	85	75	52	37	76	57	1	0	1	6
-1	29	58	64	39	37	43	36	40	41	30	19	59	76	68	55	44	58	72	73	72	65	54	52	52	73	76	20	-1	1	1	1
-1	27	73	73	29	47	50	47	72	66	36	45	58	69	65	45	51	61	89	61	60	62	75	71	75	73	58	1	1	2	2	1
-1	27	72	36	26	54	69	75	77	79	79	73	70	70	62	46	48	63	64	57	70	72	69	64	73	77	28	-2	1	1	1	1
0	23	46	46	70	74	71	71	72	73	74	73	66	64	61	42	38	59	65	57	69	67	63	69	68	67	5	1	1	1	1	1
-1	44	68	39	61	62	61	63	65	69	75	73	73	67	62	47	45	46	49	60	69	61	63	67	69	40	-2	1	1	1	1	1
-1	20	82	33	44	64	63	62	61	62	73	73	62	67	62	50	57	53	46	56	60	56	61	65	67	20	-1	1	1	1	1	1
1	5	56	42	20	60	70	64	60	61	64	68	41	50	63	47	43	45	38	53	50	51	61	61	55	8	0	1	1	1	1	1
-1	18	58	39	27	51	67	60	55	60	59	60	35	29	58	48	44	46	40	54	53	49	59	56	42	2	1	1	1	1	1	1
-1	24	57	35	23	46	57	57	56	58	55	55	37	31	57	47	39	39	40	51	52	49	56	53	30	-1	1	1	1	1	1	1
0	8	44	30	30	48	50	49	50	49	43	49	47	26	49	37	33	40	34	45	46	46	47	44	22	0	2	1	1	1	1	1
0	0	5	17	21	24	25	24	27	26	26	23	13	12	27	23	21	24	26	28	28	30	25	25	13	0	1	0	0	0	0	0

Fig. 10 In a computer, an image is represented as an array of numbers. Each number corresponds to the gray level of a given pixel. Here the example is a slice of an anatomical MRI which has been severely undersampled so that the different pixels are clearly visible. Note that an anatomical MRI is actually a 3D image and would thus be represented by a 3D array rather than by a 2D array. *Image courtesy of Ninon Burgos*

3.4 Illustration in a Simple Case

We will now illustrate step by step the above concepts in a very simple case: univariate linear regression. Univariate means that the input is a single number as in the example shown in Fig. 7. Linear means that the model f will be a simple line. The input is a number x and the output is a number y. The loss will be the least squares loss: $\ell(y, f(x)) = (y - f(x))^2$. The model f will be a linear function of x that is $f(x) = w_1 x + w_0$ and corresponds to the equation of a line, w_1 being the slope of the line and w_0 the intercept. To further simplify things, we will consider the case where there is no intercept, i.e., the line passes through the origin. Different values of w_1 correspond to different lines (and thus to different functions f) and to different values of the cost function $J(f)$, which can be in our case rewritten as $J(w_1)$ since f only depends on the parameter w_1 (Fig. 11). The best model is the one for which $J(w_1)$ is minimal.

How can we find w_1 such that $J(w_1)$ is minimal? We are going to use the derivative of J: $\frac{dJ}{dw_1}$. A minimum of $J(w_1)$ is necessarily such that $\frac{dJ}{dw_1} = 0$ (in our specific case, the converse is also true). In our case, it is possible to directly solve $\frac{dJ}{dw_1} = 0$. This will nevertheless not be the case in general. Very often, it will not be possible to solve this analytically. We will thus resort to an iterative algorithm. One classical iterative method is *gradient descent*. In the general case, f depends not only one parameter w_1 but on a set of parameters $(w_1, \ldots, w_p)$ which can be assembled into a vector $\boldsymbol{w}$. Thus, instead of working with the derivative $\frac{dJ}{dw_1}$, we will work with the gradient $\nabla_{\boldsymbol{w}} J$. The gradient is a vector that indicates the direction that one should follow to climb along J. We will thus follow the opposite of the gradient, hence the name gradient descent. This process is illustrated in Fig. 12, together with the corresponding algorithm.

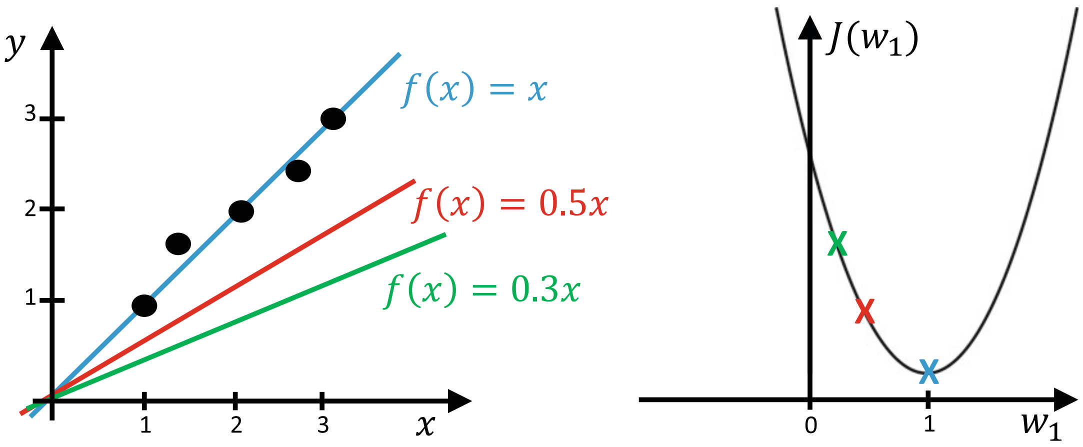
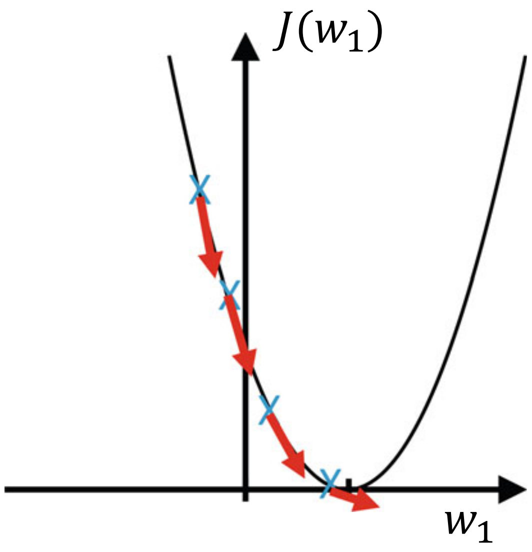


Fig. 11 We illustrate the concepts of supervised learning on a very simple case: univariate linear regression with no intercept. Training samples correspond to the black circles. The different models $f(x) = w_1 x$ correspond to the different lines. Each model (and thus each value of the parameter w_1) corresponds to a value of the cost $J(w_1)$. The best model (the blue line) is the one which minimizes $J(w_1)$; here it corresponds to the line with a slope $w_1 = 1$



```
**repeat**
    $w_1 \leftarrow w_1 - \eta \frac{dJ}{dw_1}$
**until** $convergence$;
```

Fig. 12 Upper panel: Illustration of the concept of gradient descent in a simple case where the model f is defined using only one parameter w_1. The value of w_1 is iteratively updated by following the opposite of the gradient. Lower panel: Gradient descent algorithm where η is the learning rate, i.e., the speed at which w_1 will be updated

4 Conclusion

This chapter provided an introduction to machine learning (ML) for a non-technical readership (e.g., physicians, neuroscientists, etc.). ML is an approach to artificial intelligence and thus needs to be put into this larger context. We introduced the main concepts underlying ML that will be further expanded in Chaps. 2–6. The reader can find a summary of these main concepts, as well as notations, in Box 3.

Box 3: Summary of main concepts

- The input x
- The output y
- The training samples $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$
- The model: transforms the input into the output

$$f \text{ such that } y = f(x)$$

- The set of possible models $\mathcal{F}$
- The loss: measures the error between the predicted and the true output, for a given sample

$$\ell(y, f(x))$$

- The cost function: measures the average error across the training samples

$$J(f) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(y^{(i)}, f\left(x^{(i)}\right)\right)$$

- Learning process: finding the model which minimizes the cost function

$$\hat{f} = \arg\min_{f \in \mathcal{F}} J(f)$$

Acknowledgements

The author would like to thank Johann Faouzi for his insightful comments. This work was supported by the French government under management of Agence Nationale de la Recherche as part of the “Investissements d’avenir” program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute) and reference ANR-10-IAIHU-06 (Institut Hospitalo-Universitaire ICM).

References

1. Samuel AL (1959) Some studies in machine learning using the game of checkers. IBM J Res Dev 3(3):210–229
2. Russell S, Norvig P (2002) Artificial intelligence: a modern approach. Pearson, London
3. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5(4):115–133
4. Wiener N (1948) Cybernetics or control and communication in the animal and the machine. MIT Press, Cambridge
5. Hebb DO (1949) The organization of behavior. Wiley, New York
6. Turing AM (1950) Computing machinery and intelligence. Mind 59(236):433–360
7. McCarthy J, Minsky ML, Rochester N, Shannon CE (1955) A proposal for the Dartmouth summer research project on artificial intelligence. Research Report. http://raysolomonoff.com/dartmouth/boxa/dart564props.pdf
8. Newell A, Simon H (1956) The logic theory machine–a complex information processing system. IRE Trans Inf Theory 2(3):61–79
9. Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev 65(6): 386
10. Buchanan BG, Shortliffe EH (1984) Rule-based expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project. Addison-Wesley, Boston
11. McCarthy J (1960) Recursive functions of symbolic expressions and their computation by machine, part I. Commun ACM 3(4): 184–195
12. Cardon D, Cointet JP, Mazières A, Libbrecht E (2018) Neurons spike back. Reseaux 5:173–220. https://neurovenge.antonomase.fr/RevengeNeurons_Reseaux.pdf
13. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088): 533–536
14. Le Cun Y (1985) Une procédure d'apprentissage pour réseau à seuil assymétrique. Cognitiva 85:599–604
15. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural Comput 1(4): 541–551
16. Matan O, Baird HS, Bromley J, Burges CJC, Denker JS, Jackel LD, Le Cun Y, Pednault EPD, Satterfield WD, Stenard CE et al (1992) Reading handwritten digits: a zip code recognition system. Computer 25(7):59–63
17. Legendre AM (1806) Nouvelles méthodes pour la détermination des orbites des comètes. Firmin Didot
18. Pearson K (1901) On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2(11):559–572
19. Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugenics 7(2):179–188
20. Loh WY (2014) Fifty years of classification and regression trees. Int Stat Rev 82(3):329–348
21. Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106
22. Vapnik V (1999) The nature of statistical learning theory. Springer, Berlin
23. Boser BE, Guyon IM, Vapnik VN (1992) A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on computational learning theory, pp 144–152
24. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B et al (2011) Scikit-learn: Machine learning in python. J Mach Learn Res 12:2825–2830
25. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems, vol 25, pp 1097–1105
26. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
27. Abadi M, Agarwal A et al (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/, software available from tensorflow.org
28. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L et al (2019) PyTorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems, vol 32, pp 8026–8037

29. Chollet F et al (2015) Keras. https://github.com/fchollet/keras
30. Shortliffe E (1976) Computer-based medical consultations: MYCIN. Elsevier, Amsterdam
31. Mitchell T (1997) Machine learning. McGraw Hill, New York
32. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781
33. Radford A, Narasimhan K, Salimans T, Sutskever I (2018) Improving language understanding by generative pre-training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf
34. Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805