Athens Journal of Social Sciences- Volume 10, Issue 2, April 2023 – Pages 79-100

From Big Data to Machine Learning: An Empirical Application for Social Sciences

By Giovanni Di Franco[*] *& Michele Santurro*[±]

Machine learning (ML), and particularly algorithms based on artificial neural networks (ANNs), constitute a field of research lying at the intersection of different disciplines such as mathematics, statistics, computer science and neuroscience. This approach is characterized by the use of algorithms to extract knowledge from large and heterogeneous data sets. In this paper we will focus our attention on its possible applications in the social sciences and, in particular, on its potential in the data analysis procedures. In this regard, we will provide an example of application on sociological data to assess the impact of ML in the study of relationships between variables. Finally, we will compare the potential of ML with traditional data analysis models.

Keywords: *machine learning, artificial neural networks, supervised learning, linear models, nonlinear models*

Introduction

ML is an automatic learning process that takes place through the processing of usually very large data sets. The procedures of the past, defined with the “symbolic artificial intelligence” label, operated on algorithms constituted by a logical set of instructions by which a given output (usually called target) was encoded for all possible inputs. Contrarily, the new ML systems “learn” directly from data and estimate mathematical functions that discover representations of some input, or learn to link one or more inputs to one or more outputs in order to make predictions on new data (Jordan and Mitchell 2015).

In recent years in various human sciences: economics (Varian 2014, Blumenstock et al. 2015, Athey and Imbens 2017, Mullainathan and Spiess 2017), political science (Baldassarri and Goldberg 2014, Bonikowski and DiMaggio 2016), sociology (Barocas and Selbst 2016, Evans and Aceves 2016, Baldassarri and Abascal 2017), communication science (Hopkins and King 2010, Grimmer and Stewart 2013, Bail 2014), etc., ML has started to be applied both in academic research and in areas related to the management of services provided by the public administration (Athey 2017, Berk et al. 2021) or by private companies.

Overall, many different approaches and tools are included under the ML label (Kleinberg et al. 2015). There is no consensus about how much depth a model requires to qualify as deep. Discussions with deep learning (DL) experts have not yet yielded a conclusive response to this question. However, DL can be safely understood as the set of models that involve a greater amount of composition of

[*] Professor, Department of Social and Economic Sciences, Sapienza University of Rome, Italy.
[±] PhD Student, Department of Social and Economic Sciences, Sapienza University of Rome, Italy.

https://doi.org/10.30958/ajss.10-2-1 doi=10.30958/ajss.10-2-1

either learned functions or learned concepts than traditional ML does (Schmidhuber 2015, Goodfellow et al. 2016).

DL is not a breakthrough in the scientific sense, rather it is a relevant breakthrough in efficient coding that makes a difference in several contexts. In practical applications, DL is able to achieve higher accuracy on more complex tasks as compared with traditional ANNs, although it requires more computational resources. Furthermore, DL needs less manual interference to craft the right features or the suitable transformations of data. It performs exceptionally precise operations on data that come from different modalities, such as images, texts and videos (Schmidhuber 2015, Alpaydin 2016, Goodfellow et al. 2016).

So, the choice between ML or DL algorithms depends on the problem to be analysed. If the problem is relatively simple, it is preferable to use ML based on ANNs with few layers of hidden units; if the problem is complex or requires the achievement of very specific and rigorous objectives, it is considered more useful to resort to DL.

Here we will only consider ANNs that use supervised ML algorithms. In the supervised ML the algorithm observes an output for each input. This output gives the algorithm a target to predict and acts as a "teacher". On the contrary, unsupervised ML algorithms only observe the input and their task is to autonomously compute a function without a predetermined target (Hastie et al. 2009, Molina and Garip 2019). This work has two aims: a) to present ANN algorithms-based ML in a simple and intuitive manner; b) to apply it to sociological data by comparing the results obtained with the results of traditional statistical techniques, to evaluate its strengths and weaknesses.

When the question of progress in sociology is raised, it is in fact on the extent of theoretical progress that debate centres. In this regard, we would then suggest, empirical research will benefit in so far as its practitioners show a readiness to engage with methodological issues. Through such an engagement, we will be called upon to confront problems that arise by the development of the phenomena studied rather than by the development of science itself and will in this way be subject to two constraints or disciplines that we would view as salutary. The first requirement will be to move, as it were, from *explananda* to *explanantia*, from effects to causes – which, following Popper, we would take to be the way of science – rather than to go in the reverse direction. Second, theoretical models will more often come to be deployed in "data-rich" rather than "data-poor" contexts. This means that they will face relatively stringent empirical tests of their validity. In this way, serious explanatory efforts may be more readily distinguished from what Goldthorpe (2004) called sociological dandyism: a preoccupation with models, whether statistical or theoretical, on account more of their intrinsic elegance, refinement and subtlety than of what can be shown to follow from their sociological use that is of major substantive relevance, whether from the standpoint of pure or applied interests.

Literature Review

Historically, ANNs have been proposed to emulate some functions of the human brain and nervous system, within an approach called connectionism (Di Franco 1998, van der Maas et al. 2021).

Connectionism attempts to simulate biological intelligence on a computer by taking the brain as a physical organ as a model. This is a first and fundamental distinction between connectionism on the one hand and cognitivism based on symbolic artificial intelligence on the other (Grum 2022).

In connectionism the brain is the metaphor by which the mind is studied. For years there has been a bitter dispute between the opposing advocates of the two different approaches, for which readers are referred to Parisi (1989), Cammarata (1990) and Buscema (1994).

In the opinion of the psychologist Parisi (1989) this set of ideas, theories and computational techniques, starting from the second half of the 1980s, represents a scientific revolution in the study of the mind and brain.

The first attempts to set up intelligent systems on computers that emulated brain activity had been elaborated in neurocybernetics in the 1930s. In the 1940s McCulloch and Pitts built the first intelligent systems based on the simulation of brain activity. In the 1970s, research on ANNs experienced a period of stagnation as the systems that were created showed low efficiency. Since the second half of the 1980s, thanks to the availability of parallel computing systems and new learning algorithms, interest in ANNs has grown rapidly (Lu 2019, Toosi et al. 2021). At the same time, the limited progress of symbolic artificial intelligence in the construction of general intelligent systems through the symbolic manipulation techniques typical of this approach (expert systems, logical languages, semantic networks, etc.) and the technological interest in computer architectures closer to what appears to be the way the nervous system works (parallel rather than sequential), have fed the dispute between the advocates of cognitivism (or symbolic approach) and connectionism (Dietrich et al. 2021). According to Cammarata (1990) the symbolic approach is more suitable for simulating conscious intelligence processes, such as the human expert reasoning or theorem proving; but it does not seem to reflect the nature of many unconscious intelligence processes such as those related to image or sound recognition. It is however acknowledged that the connectionist paradigm, and in particular ANNs, not only can tackle new classes of problems, but also confer advantages in terms of simplicity and efficiency in solving problems already tackled. At present, the applications of ANNs range from sophisticated military technologies for guiding missiles to airport security systems, to encryption mechanisms, to shape and image recognition, to quality control in industrial automation processes, to adaptive noise cancellation in telecommunications, and many more (Balas et al. 2020, Echeberria 2022, Elliott 2022).

We can say that connectionism is an approach that opposes cognitivism trying to overcome the classic Cartesian distinction between mind and brain. As a result of this distinction, disciplines that dealt with mental capacities and intelligence, such as psychology, had separated from the neurosciences that studied the brain and central nervous system as organs of the human body (Boden 2006, Pecere 2020).

Connectionism is linked to a set of tools such as ANNs, nonlinear dynamic systems, complex dynamic systems, distributed parallel computing systems, associative memories, etc., which make use of computer simulation. The advantages offered by this approach are also very interesting for the social sciences. First of all, to simulate a phenomenon on a computer it is necessary to make explicit and formalize all the knowledge that is available. Furthermore, once one is able to simulate a certain phenomenon, it becomes possible to manipulate it in ways that would not be allowed with other research techniques, for ethical reasons and for other constraints dictated by the limited resources available in any scientific research (Fetzer 2004, Keuschnigg et al. 2018).

The use of the computer has made it possible to successfully study phenomena characterized by high dynamism, high parallelism and strong complexity, governed by rules of change that can be described by nonlinear equations, practically solvable only by using a computer (Strohmaier 2021).

Shortly, connectionism, by building artificial neural systems based exclusively on mathematical rules, attempts to build intelligent systems. The term ANN stands for a set of computational rules that simulate a behaviour typical of the brain structure of human beings. This is the fundamental difference of connectionism compared to the symbolic approach, whose intelligence model is based on the symbol manipulation through the use of rules that constitute the program to be executed. Unlike the computational models used in expert systems, in ANNs there is no program that specifies the operations to be executed, but the computational procedure is defined through the characteristics of the units and their connections (Davenport 2013).

A network learns and generalizes through the experience it acquires rather than through a program that determines its behaviour. The alternative that connectionism offers consists in the construction of artificial neural systems capable of learning, and subsequently generalizing, based on the experience that is administered to them. There are many types of ANNs, distinguished by architecture, learning rules, signal transfer functions, etc. There is no space here to present them all. We will focus in particular on those networks, called supervised, that have a goal to achieve in the training step, as opposed to those called unsupervised (also called autopoietic) which in the training step do not have a predetermined goal to achieve.

Methodology

ANNs consist of many computing units (called nodes or artificial neurons), usually organized into layers, with very simple operation and interconnected to each other. Through such networks, a signal (in the form of examples, called patterns) is passed, exciting or inhibiting the units. They, with appropriate mathematical rules, transfer the signal to other units until producing a quantitative output. In other words, each unit receives excitation (or inhibition) from the units from which connections arrive and, in turn, transmits excitation (or inhibition) to the units towards which it has connections.

Here we will mainly deal with feedforward networks, which have units arranged on at least three layers and unidirectional connections between each unit of one layer and all the other units of the next layer.

The excitation or inhibition that reaches a certain unit through the other units to which it is connected depends on the weights that characterize the links. If a connection weight is high, this causes a lot of activation; a low weight causes little activation. A positive weight transmits excitation; a negative weight inhibition.

What characterizes ANNs is the parallel processing: each node of the network constitutes an autonomous computing unit that carries out computations in parallel with all the others. In serial systems, on the other hand, operations are carried out in sequence, one after the other.

An ANN is able to learn a task, solve a problem, when the parallel propagation of the network activation reaches an equilibrium (when the function reaches its minimum value), namely, when the activation arrives at the output units of the network.

As said, the fundamental aspect of ANNs is their ability to learn, but it is important to be clear on this point. In fact, what ANNs learn and what allows them to perform tasks or solve problems are the weights that are assigned to the links and that regulate how much excitation or inhibition is propagated in the network and how this propagation takes place.

In other words, in ANNs, learning, that is, the acquisition of knowledge and ability, consists in a process of connection weight adjustment. ANNs are therefore intrinsically quantitative, they learn numerical weights, transform them mathematically and provide a quantitative result.

In the training step of a network, the initial state (i.e., the initial connection weights) is randomly defined, usually in a very small range (e.g., between -0.1 and +0.1). Some patterns are presented to the network, each associated, in supervised networks, with a target. The network must produce, for each pattern, an output as similar as possible to the target. The difference between the network output and its target is the error. Through a mechanism that is called error backpropagation (EBP), the network adjusts the connection weights until the distance between output and target is minimized.

The nodes emulate the brain’s nerve cells (neurons); the links between the nodes emulate the synaptic connections that exist between the axon of a neuron and the dendrites of another neuron. Indeed, research conducted so far with ANNs has allowed to reproduce only some, though important, characteristics of the human brain, which however are not reproducible in any other way.

In their mathematical characteristics, ANNs are part of a larger class of models formulated for the study of complex systems, with nonlinear dynamics, of the chaotic type, etc. These models have been introduced in the most innovative research sectors of different disciplines. Their general and abstract character makes them applicable to very different phenomena, and therefore of very broad potential interest, even for social phenomena.

The point is whether ANNs can be usefully applied in social research, besides as a complex of nonlinear data processing algorithms, also as a tool to simulate social phenomena (Capecchi 1996).

It is difficult to assimilate social phenomena to neurophysiological ones; for this reason, the analogies of the nodes of an ANN with neurons, of its connections with synapses, etc., that are possible in the study of the brain, are not possible in these other cases. However, it is a question of assessing whether the abstractness of the structures and processes postulated in ANNs, understood as models of complex nonlinear dynamic systems, does allow their application also to the study of social phenomena. In this case it is necessary to determine the interpretation to be given to concepts such as node, connection, excitation/inhibition, connection weight, learning rule, equilibrium and so on.

On the other hand, the use of ANNs allows the possibility of partially overcoming some limitations of the analyses conducted with traditional statistical techniques. For example, the use of ANNs does not require any hypothesis on the distributions of the system variables and their reciprocal associations. For this reason, it is possible to treat cardinal, ordinal and/or categorical variables (Di Franco 2017). By such approach the actual analysis of the system is left to the network, which alone creates its own criteria to reproduce its behaviour and consequently enables itself to formulate predictions on the system itself. In Fabbri and Orsini's (1993) judgement, this is both a strength and a weakness of ANNs: it is a strength because in this way the researcher is not conditioned by a priori hypotheses in the choice of the network units; the weakness consists in the fact that the network is not able to do other than reproduce the behaviour of the analysed system in a phenomenological manner, without contributing to the knowledge of the internal relationships between the single parts of the system. This problem, however, can be partially overcome as some devices, that allow us to interrogate the network about what it was able to reproduce, have been fine-tuned (Di Franco 1998).

If the simulation approach of ANNs to social phenomena proved to be possible and useful (Capecchi et al. 2010), this would allow significant progress in the social disciplines because it would also contribute to the foundation of a consistent basis of simulation concepts, models and techniques. If social phenomena can be thought of as complex dynamic systems[1] then it is necessary to accept the possibility of simulating them on a computer with more meaningful results than those obtainable with traditional data analysis tools.

We illustrate some key concepts of ANNs, and in particular of the feedforward networks with at least one hidden layer characterized by a learning technique called EBP. This type of network was proposed by a group of researchers at the University of California in San Diego (Rumelhart et al. 1986, 1987). Instead of resorting to mathematical formalizations we will use graphs.

[1] Where by complex dynamic system we mean a system made up of a large number of elements that interact on the basis of purely quantitative and non-symbolic rules, and that change over time, giving rise to complex collective dynamics.

As said, a characterizing aspect of ANNs is their ability to learn; learning consists in the search for the set of connection weights appropriate for each specific task. The network starts from a state where weights are randomly assigned; therefore, the resulting output is, at time t-zero, equally random. Through its training, a progressive adjustment of the connection weights of the network takes place until obtaining the set of weights that produces the desired output in the best possible way. But, even after long training, ANNs do not usually produce very accurate results. This feature, which could be a limitation especially for tasks where high accuracy is required, becomes interesting in classification and recognition tasks. In fact, in a classification task, similar objects can be placed in the same class; consequently, even patterns affected by noise, biases or missing data can be classified. This shows that the networks have a high noise tolerance; this feature is important considering that in data analysis one often comes across low quality data.

The most interesting peculiarity of neural models, however, is their ability to generalize: if a pattern different from the ones used for learning is presented as input, the network is able, within certain limits, to classify it in the "correct" way (provided that a class for that pattern exists).

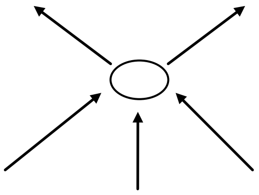
In social research, by exploiting the generalization ability of ANNs, cases with missing data could be processed without being forced to exclude them or replace them with the mean, the mode or the median of the relative distribution.

Schematically an ANN consists of:

- a large number of simple units (artificial neurons);
- a large number of links between the units (artificial synapses);
- a parallel and distributed control scheme;
- a learning algorithm.

A feedforward ANN is made up of a number of units connected by links which are, in this type of network, unidirectional. Excitation or inhibition is transmitted through the links from one unit to another. Each unit has a number of incoming links with other units and some outgoing links towards other units (see Figure 1).

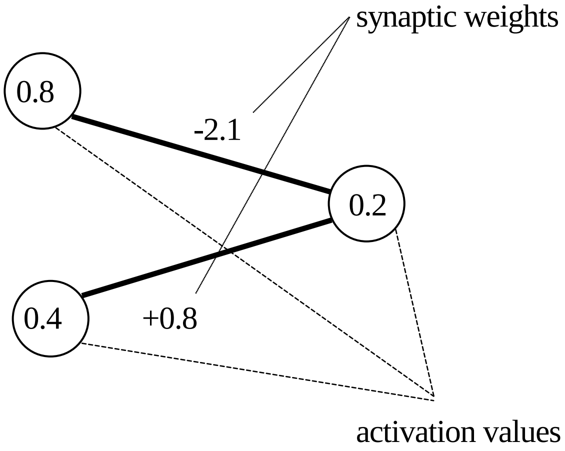
Figure 1. *Unit with Three Incoming and Two Outgoing Links*



So, there is a layer of input units that have no incoming links but only outgoing links. The activation state of these input units is determined from outside the network. And there is a second layer of output units that have only incoming links and no outgoing links. In practice, the activation state of the output units is read from the outside and tells us how the network reacted to the input from the outside.

The activation state of a unit is equal to a combination of all the excitations and inhibitions that reach that unit through its incoming links. Each amount of excitation or inhibition is weighted by a value, called connection weight, which characterizes each link. The weight can have a positive or negative sign, and this determines whether excitation (positive sign) or inhibition (negative sign) is transmitted (Figure 2).

Figure 2. *Connection Weights*



The amount of excitation or inhibition that comes through a certain link is actually determined by two factors. The first one is the connection weight (a link with weight +0.8 passes more activation than a link with weight +0.2). The second one is the activation state of the unit from which the link starts, which can be more or less high. The two factors are multiplied among themselves, and the result is the amount of excitation or inhibition that arrives at a certain unit through a certain link. In feedforward ANNs the activation state of a certain unit varies from a minimum (0) to a maximum (1). At a given moment, a number of excitations and inhibitions arrive at a certain unit.

The first thing the unit has to do is to compute all these excitations and inhibitions in a single value, which is called the net input for that unit. The net input is normally the algebraic sum of all the excitations and all the inhibitions that arrive at each node of the network.

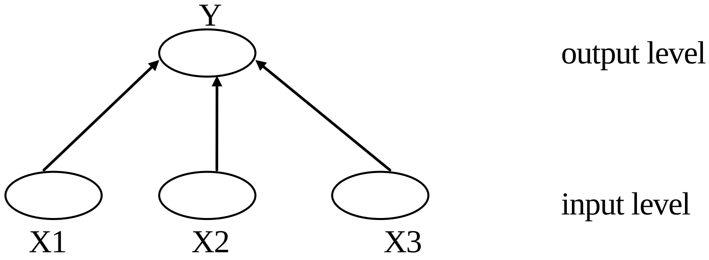
The net input is then transformed through a mathematical function in the activation state of the unit. The algorithm that transforms the net input into the activation state can be the logistic function, or sigmoid, with continuous values and saturation. Other algorithms (called transfer functions) can also be used, such as the identity or linear function without saturation, the linear function with saturation, the step function with binary or bipolar values, and others.

In the sigmoid the activation state varies between a minimum and a maximum (0 and 1). When the net input is 0, the activation state is 0.5. The algorithm is sensitive to small variations in the net input, which produce strong upward or downward deviations of the activation state in the central part of the range of variation.

Once the activation state has been computed through the algorithm of the sigmoid function, it is determined how each unit influences the other units with which it is linked. In feedforward networks the units are grouped in layers and the units of the same layer are not linked together; they can only be linked with units in other layers.

When the network has only one input and one output layer, it is called Perceptron. Each input unit is linked with each of the output units through a link whose intensity is measured by the weight; there are no horizontal links between output units; the propagation of signals is unidirectional from the input to the output (Figure 3).

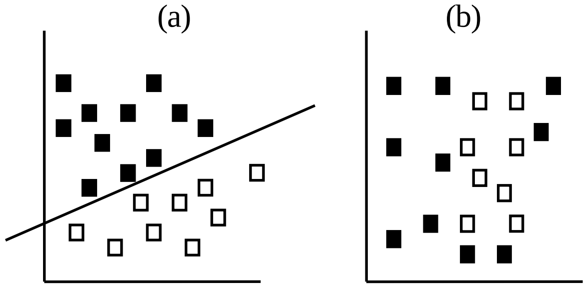
Figure 3. *The Perceptron (Two-Layer Feedforward Network Input -> Output)*



Each unit of a Perceptron has a set of inputs, each having a weight that represents the strength of the neuron's synaptic link.

Figure 4 shows the case of a Perceptron with two input and one output units. The two input units encode the coordinates of patterns with respect to the two-dimensional plane; the output unit encodes the type of pattern in two classes, A and B. The task of the Perceptron is to classify the input patterns into two distinct classes.

Figure 4. *A Perceptron with Two Input and One Output Units is Able to Identify the Two Classes in Situation (a) = Linearly Separable Patterns; but it Fails in Situation (b) = Nonlinearly Separable Patterns*



Two sets of patterns are presented. In the first set, the patterns of the two classes, distributed in a plane, can be separated by a line (the Perceptron is able to discriminate the two classes). In the second set, conversely, the patterns of the two classes cannot be separated by a line (consequently the Perceptron is not able to discriminate them correctly).

The global input of the neuron is an n-dimensional vector with associated weights. To obtain the output of the Perceptron, each element of the input vector is multiplied by its weight, and all the values thus obtained are added together. The unit gives 1 as output if the sum is greater than a certain threshold value, otherwise it gives 0. The major limitation of the Perceptron is its inability to perform classification tasks for nonlinearly separable problems (Figure 4).

In summary, a Perceptron does nothing but learn a series of direct associations between pairs of activation patterns. The network associates the output pattern with the input pattern by progressively adjusting the weights of the direct links between the input and the output units so as to store not a single association between an input and an output pattern, but as many associations as there are patterns to learn. In so doing, the Perceptron does not construct any internal representation of the different patterns it has learned, and therefore cannot highlight the similarities and differences between them. Precisely because the Perceptron cannot construct an autonomous internal representation, it is unable to make inferences on new characteristics of patterns, viz. on characteristics that it has not directly experienced.

When linear separation is impossible, the Perceptron is unable to solve even seemingly simple problems. This limitation can be overcome by adding to the network an additional layer of units placed between the input and output ones. This intermediate layer is called hidden precisely because it is inside the network and has no links with outside the system, unlike the input layer which receives information from the outside and the output layer which transmits information to the outside (Figure 5).

Figure 5. *Feedforward Network with One Input, One Hidden and One Output Layer*

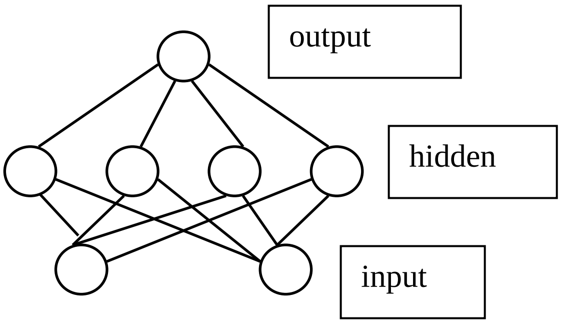
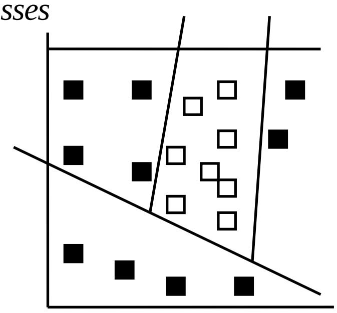


Figure 6 presents a case in which the input patterns of the two classes, distributed in a plane, cannot be linearly separated. However, they can be separated by drawing a number of lines (in this example, four). A possible solution to the problem consists in defining a multilayer neural architecture, which contains a sufficient number of hidden neurons to solve the task.

According to Kolmogorov's theorem (Cammarata 1990) a multilayer network provided with a sufficient number of hidden units is able to learn any function.

Figure 6. *Example of Input Patterns Divided into Two Nonlinearly Separable Classes*



A multilayer ANN is therefore able to recognize whether or not a given pattern belongs to class A by separating it from class B and to recognize the classes starting from a structure of links progressively adjustable in the training phase. Here the error is minimized by adjusting the connection weights by various criteria that guarantee, even in the case of linear nonseparability, the convergence of the iterative process towards the optimal solution. The way in which a network responds to an external activation pattern depends entirely on the connection weights between the units. What we want from a network is that it produces a certain activation pattern on the output units. But the weight of a specific link enters in determining the activation value of a given output unit by combining with the weights of the other links and with the activation state of the other hidden units. The activation states of the hidden units, in turn, depend on a large number of links proceeding from behind, and so on. So, it is almost impossible to determine the weight of each link, all the more if the fact that an output unit must be activated or not on a particular occasion depends on the overall activation pattern that is expected from all the output units on that occasion.

The common feature of many ANNs is that they initially have randomly chosen connection weights; there are criteria by which networks automatically adjust these weights until assigning them those values that allow to respond to a certain external stimulation as desired. At the beginning the network will give random responses to external stimuli. However, exposed to repeated experiences, the network progressively adjusts its weights so that they will produce the desired performance. Different learning techniques have been defined for ANNs. What we will consider is a supervised learning from the outside, that is, a learning in which there is a specific external target associated with each input pattern that each time imposes on the network the desired performance. The network consequently adjusts the connection weights until, after a number of times, it is able to reproduce approximately the desired output for each input.

In this paper we present the supervised learning criterion called EBP. EBP allows the network to compare, for each output unit, the obtained value with the desired value and to use the difference to adjust the connection weights in the right direction, so that after a number of learning cycles the connection weights determine the desired activation values on the output units. The EBP algorithm requires a multilayer network architecture: one input layer, one output layer and one or more hidden layers. Each unit of a layer is linked to the units of the preceding layers; there are no horizontal links between units of the same layer and the signal is propagated unidirectionally from the input to the output through the hierarchy of the hidden layers (feedforward networks). The procedure envisages two steps: in the first one, the input patterns are transmitted to the output, the activation value of the outputs is evaluated and compared with the target values; in the second one, the computed error is backpropagated from the output to the hidden layers and from these to the input layer.

The errors of the output units allow to estimate the errors of the hidden units and update the weights of their links. Thus, the errors of the input units are estimated and the weights of their links are updated. The algorithm is run for all the values of the training set until obtaining the correct values for the weights of all the links. After a first presentation of the patterns, for which the weights are updated, it is possible to proceed to other cycles of presentation until the mean squared error over the entire training set does not fall below a set threshold, or when new iterations do not correspond to a decrease in the error and therefore the network has reached a stable state, i.e. a minimum of the error function. For a number of times (which can vary from a few hundred to several thousand: each cycle carried out over the entire series is called epoch) it elaborates this series of training patterns.

A mathematical description of the EBP algorithm is in Cammarata (1990). The EBP algorithm can be applied to a network with any number of layers; the number of units can vary from layer to layer. The number of input units and output units is determined by the problem to be solved, whereas there are no criteria to determine the optimal number of hidden layers and that of the units they contain. Generally no more than one or two hidden layers are introduced, and the number of their units is usually limited. The advantage of the EBP algorithm compared to the previous weight adjustment techniques is its ability to compute an error not only for the output unit layer – for which it is very easy since they receive the target from the outside – but also for the hidden unit layer. In this way, through the learning, not only the connection weights between hidden units and output units, but also those between input units and hidden units are adjusted. In an ANN of the kind described above, the association between input and output is mediated by the hidden units and the links that connect, each with its weight, the input units with the hidden ones and these with the output units. We can say that an ANN develops an internal representation of the input, and its response to the input depends on that. This aspect of the networks recalls the concept of latent dimension typical of many traditional statistical techniques such as factor analysis (Di Franco and Marradi 2013). The internal representation of an input is obviously not symbolic; it is nothing more than the set of activation values that result on hidden units when the network receives an input. It is these activation values that determine the activation values of the output units, based on the connection weights between hidden and output units. And it is on the basis of this internal representation produced during the training phase that the network recognizes similarities and differences and is able to infer and generalize.

The EBP algorithm can be speeded up by adding a term to the weight adjustment formula that takes into account the update in the previous epoch: this additional term is controlled by a parameter ranging from zero to one called momentum. One drawback is the possibility that the algorithm does not reach the absolute minimum of the error function. In fact, the higher the value of the learning rate, the faster the network will learn. However, this entails the possibility of oscillations of the error function around a minimum value. On the other hand, a too small value of the learning rate can lead to too long training times, so the value of the parameter is often determined by trial and error. EBP guarantees that the convergence towards the global minimum occurs for a wide variety of tasks, in particular by avoiding the network falling into a local minimum, i.e., in a weight setting from which it cannot move but which does not correspond to the global minimum error that is trying to reach.

Learning ends when the value of the global error is low enough, and in any event it shows no signs of any further reduction by increasing the number of epochs. At this point the network has learned, namely, it is able to provide the approximately correct output for each input. Most importantly, the network demonstrates that it has an ability to go beyond what it has been explicitly taught. This ability manifests itself in various ways. If the network has learned to give a certain response to a pattern, it will give this response to a damaged, partial, obscured by noise version of this pattern too. If a pattern has been classified as belonging to a certain class, similar patterns never seen before will also be classified as belonging to that class. If the response that has been learned from the network for a certain pattern contains some unspecified parts, the network will be able to correctly infer the missing parts.

There are other factors, besides EBP, that can come into play in learning. For example, the learning rate, viz. the size of connection weight adjustment given a certain error, can be varied. As a rule, it is preferred to make small adjustments to have a gradual and smooth learning. Another factor that can be varied is the momentum, that is, if and how much the adjustment that I introduce now must be influenced by the adjustments introduced on the same weight previously. Then there is the bias, namely, an activation value that each unit tends to take regardless of the excitations and inhibitions that come from the other units. The bias is different from unit to unit and consists of excitation or inhibition that reaches the unit, through a link with a learnable weight, from a hypothetical special unit that always has activation equal to one. These additional mechanisms are indicative of the flexibility of multilayer networks and EBP learning.

All this variability of factors, which first of all derives from the fact that at the outset each network receives its own specific random assignment of weights, entails that the whole course of learning and its final result will vary from network to network. Thus, if the same experiment is repeated on different networks, that is, having an initial assignment of different weights and/or different values of the parameters described above, it will not be possible to have identical but only similar results. Other differences may arise from the training time and from the way in which the network examines the different patterns. Even by varying the training time and the order of presentation of the patterns, different results are obtained.

The training time poses a further problem: if a network undergoes long learning there is the risk of overtraining thus compromising its ability to generalize. In fact, if a network learns the patterns used during the training too well, it will be less able to classify new patterns, different from those used in the training set. To this goal, the testing set is used. In this step the network has already learned the weights used in the training set and now responds to new patterns that are submitted to it without each being associated with a target. It is therefore more important that the network is able to learn well the prototypes underlying the patterns, rather than being able to respond correctly to each input in the training set. The conclusion to be drawn is that the concept of prototype is central to ANNs as a basis for classification. This shifts the emphasis from the classes defined in terms of characteristics (as is normally done) to the classes defined in terms of prototype. The ability to extrapolate, to respond sensibly to the new things, is one of the most important features of ANNs, and one of their main advantages compared to traditional analysis systems. Each network responds sensibly to patterns that are new compared to those with which it was trained. However, the response is generally less good than that given for the training patterns: the network is more uncertain; if it has to classify the new pattern in class A, it gives an activation value of 0.8 or 0.7, instead of 0.9. Conversely, what happens with the prototypes is that, if the prototype pattern is presented to the network, and the latter has never seen it before, its response can be even better than that given to the patterns it has trained many times.

In short, the learning algorithm can be interpreted as the descent down any function from its generic point, whose coordinates are the initial randomly assigned weights and the initial error at its minimum point. The learning rate can be interpreted as the step of such descent.

Of course, local minima are possible. What the network looks for is a minimum value of the global error, i.e., the connection weight setting that gives the minimum error for all input patterns. Instead, a local minimum is a weight setting that keeps the error still quite high without the network being able to escape from this setting, as that would lead to an initial increase in the error and then a descent towards a lower error. In nonlinear functions there is not a single point of absolute minimum, but it is possible to find several local minima that would represent suboptimal solutions for the network.

Results and Discussion

We present an example of application[2] that consists of a comparison between a multiple linear regression model and an ANN Multilayer Perceptron.

[2] The data used in the example are taken from a matrix containing some information on the electoral polls published by the mass media in Italy from 1 January 2017 to 29 February 2020. The information relating to these polls was taken from the website www.sondaggipoliticoelettorali.it of the Presidency

We first present the results of multiple linear regression. The dependent variable is the percentage of voters who declared their intention to abstain or who declared their indecision regarding the election choice (label 'no-vot'). The independent variables are the following four: the number of days for carrying out the poll (label 'days'); the sample size (label 'n-sample'); the completeness index of the poll information (label 'ind-1'); the ratio between the interview attempts and the interviews carried out ('ind-2').

Table 1. *Multiple Regression Model Summary*

R	R Square	Adjusted R Square	Std. Error of the Estimate
0.563	0.317	0.311	8.4224

Predictors: (Constant), days, n-sample, ind-1, ind-2. Dependent Variable: no-vot.

Table 1 presents the fitting results of the model. Considering the adjusted R square, we find that the four independent variables reproduce just under a third (31.1%) of the variance of the dependent variable. Table 2 shows the regression coefficients and Table 3 the residual statistics.

Table 2. *Multiple Regression Coefficients*

Model	Unstandardized Coefficients		Standardized Coefficients		
	B	Std. Error	Beta	t	Sig.
(Constant)	21.168	2.609		8.114	0.000
days	-0.663	0.297	-0.105	-2.229	0.026
n-sample	0.007	0.001	0.284	4.986	0.000
ind-1	25.129	2.872	0.348	8.749	0.000
ind-2	-0.705	0.156	-0.226	-4.521	0.000

Dependent Variable: no-vot.

The analysis of the beta weights confirms that the contribution of the four independent variables is significant in explaining the variance of the dependent one.

The analysis of the residual statistics also shows a good fit of the model to the data (Table 3).

Table 3. *Multiple Regression Residual Statistics*

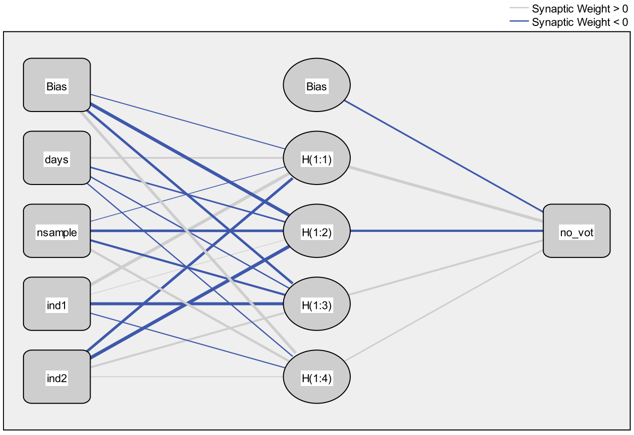
Model	Min.	Max.	Mean	Std. Deviation	N
Predicted Value	30.684	56.227	41.261	5.711	506
Residual	-26.179	35.316	0.000	8.389	506
Std. Predicted Value	-1.852	2.621	0.000	1.000	506
Std. Residual	-3.108	4.193	0.000	0.996	506

Dependent Variable: no-vot.

of the Council of Ministers, Department for Information and Publishing. For further information on the data matrix, see Di Franco (2018).

Let's now evaluate the results obtained with the ANN[3] comparing them with those obtained with the multiple linear regression (Figure 7).

Figure 7. *ANN Architecture*



Hidden layer activation function: Hyperbolic tangent

Output layer activation function: Identity

The cases submitted to the network are obviously the same 506 used in the regression. In this case, however, 70% of cases (359) were used in the training set and the remaining 30% (147) in the testing set. Table 4 presents the model summary. In the training step the relative error was equal to 0.225. In the testing step it grows slightly reaching the value of 0.327. Recall that in the testing step the network predicts the value of the dependent variable using the weights that it computed on the cases observed during the training. So basically, we assess the ability of the network to generalize what it has learned in the training.

Table 4. *ANN Model Summary*

Training	Sum of Squares Error	40.218
	Relative Error	0.225
	Stopping Rule Used	1 consecutive step(s) with no decrease in error
	Training Time	0:00:00.194
Testing	Sum of Squares Error	19.528
	Relative Error	0.327

We do not report the parameter estimates (i.e., the weights calculated for each node of the network) as their examination does not clarify the impact of each independent variable in the estimate of the dependent one.

The comparison between the results of the multiple regression and the ANN leaves no doubt about the better predictive performance of the network (Table 5). The correlation between the values predicted by the multiple regression and the

[3] For the ANN applications we used the Multilayer Perceptron procedure available in the SPSS program for Windows.

actual values of the dependent variable is equal to 0.563; the correlation between the values predicted by the ANN and the actual values of the dependent variable is thirty points higher, rising to 0.866.

Table 5. *Correlations between Predicted Values of Regression and ANN and Values of Dependent Variable*

No-vot	1		
Unstandardized Predicted Value: regression	0.563**	1	
Predicted Value for no_vot: ann	0.866**	0.391**	1

**. Correlation is significant at the 0.01 level (2-tailed).

Evidently in the relationship between the independent variables and the dependent one, the network managed to capture nonlinear trends which allow for a better estimate of the values.

Conclusions

At the end of this *excursus* on feedforward ANNs we can summarize the most important aspects by highlighting their strengths and weaknesses.

Do ANNs replace the other techniques of traditional sociology? No, they should be used essentially when we do not know how to solve a problem in another way or when we know how to solve it otherwise, but with less convenience or poorer results (Evans and Foster 2019). The logic with which these networks learn is the connectionist one and there are four great differences from the models of relations among variables and classification typical of quantitative methods: a) ANNs can define strategies of connection and predictions with variables that are not limited to the four levels indicated by Torgerson (1958) (nominal, ordinal, interval scales or relations) but also extend to fuzzy variables or variables represented by a pixel in an image (therefore different types of variables can be considered in the same case-by-variable matrix); b) ANNs use data to estimate the performance of alternative models (functions, regularization parameters) to choose the best one with respect to all of the variables (this process requires solving an optimization problem, discovering either linear or nonlinear associations among the variables themselves); c) such networks are bottom-up systems producing a data model as an end point of an iterative and a feedback loop process (this means that the weight connecting unit i to unit j has not the same value as the weight connecting unit j to unit i, as happens for instance in a correlation matrix); d) the connections between variables, being direct, have a clear conceptual meaning, indicating a relationship of faded excitement, inhibition, or indifference between every pair of variables or records (this situation is quite different from the clear separation of traditional quantitative methods between models of classification and models of relations between variables) (Kent 2009).

As the phenomenon of generalization demonstrates, ANNs are capable of learning, namely, they allow solving problems by associating the sought solution with data. Indeed, network learning techniques are applications of known statistical methods (stochastic approximation) to a new class of nonlinear regression models. In this sense the determination of the network weights can be interpreted as a nonlinear regression applied to an ANN function. The advantage is to have an extremely flexible function, avoiding the subjective components of the specification error, as the parameters implicitly determine which is the latent function that a network approximates (White 1989).

If the analytical form of the function underlying the problem under study is known, or can be assimilated to a known form, the problem of parameter estimation refers to the case of nonlinear least squares and the use of ANNs is not justified; it becomes so when one is not able to formulate reliable conjectures on such form. In this case, the use of networks is easier and more productive than other complex procedures with restrictive assumptions. The use of ANNs is therefore effective as a criterion for identifying hidden nonlinear relationships (Buskirk and Kirchner 2021).

The ability to learn is related to that to forecast. ANNs offer good performances both in univariate forecasting, that is, when one wants to predict the behaviour of a variable of a system that evolves over time on the basis of its past trend, and in multivariate analysis, when trying to predict the trend of a variable observing the past behaviour of several variables of the evolving system. Many studies have highlighted how ANNs allow good approximations and extrapolations to be made. Since a forecast problem can be referred to an approximation and extrapolation problem, it is possible to use networks to approximate the regularities present in the variations over time of the variable to be predicted. ANNs flexibly adapt to complex situations that change over time. They are also suitable for processing data that are incomplete or affected by noise or biases. By virtue of this ability to adapt to data, ANNs are very robust, viz. they have a high resistance to failures and malfunctions. Another important feature is the computational speed that derives from their parallelism and the very rapid input-output association, since the computations to be performed are weighted sums and threshold selections; therefore, they constitute a valid alternative to traditional techniques for performing complex computations (Aggarwal 2018).

The critical points of ANNs are, first of all, the long and scarcely incremental learning; in addition to requiring a large number of epochs before significantly reducing the error, learning must be repeated when the situation represented by the patterns undergoes substantial changes, unless such learning is continuous or unsupervised (Bartlett et al. 2021).

Obviously also for ANNs, as in any other case, it is necessary to have a data set that is rich and representative (of the problem under study) so that the training set and the testing set are effectively controllable (Molnar et al. 2020).

Other problems may arise from the low accuracy and the uncertain reliability of the results provided by ANNs: the past performances of a network do not guarantee those in the future. There is a risk that the generalization is not complete and that therefore most of the inputs do not recall correct outputs. Furthermore, there are no strict criteria to design the most suitable network for a given problem, but it is necessary to proceed by trial and error with, as mentioned, numerous degrees of freedom in the choice of each parameter. Moreover, each network has its own specificity. If the same experiment is repeated on another network, there will not be the same results, although in most cases they tend to converge. This is another interesting feature of ANNs; they are able to provide similar results in terms of performance with a variety of weight settings. Clearly what is important is not the value of a certain weight, but the overall set of all connection weights (Landi et al. 2010).

Finally, the criticism most frequently raised against the usefulness of ANNs is that, even when they succeed in the assigned task, they do not allow to explain their operation on a cognitive level (in the case of the sociological research we could say on the level of the analysis of relationships between variables). We expect from a model not only that it will be able to predict or reproduce its referent, but also that it will be transparent, that is, it will make us understand how it works, what mechanisms, processes and principles are behind it. ANNs, according to this criticism, risk obtaining the first goal, but not the second one. A network that was able to learn a certain task and is also capable of extending its performance to new situations, showing in this way that it has incorporated the mechanisms and principles underlying that task, may nevertheless be not very transparent as to these mechanisms and principles, not making them emerge clearly and thus not allowing their full explanation regarding the phenomenon in question. Their strictly quantitative nature, the interweaving of the links, the connection weights, the effects of a local phenomenon of activation on the rest of the network, are all factors that make the behaviour of networks dark as tools for explaining the relationships between variables (Guidotti et al. 2019, Longo et al. 2020).

How the role of mathematical thinking in the social sciences will evolve is difficult to predict, because neither mathematics nor social sciences are unchangeably fixed. Nonetheless, we recognize three main possibilities of application of ANNs. The first possibility consists in using ANNs together with statistical models to understand how different methods can contribute to the explanation of data concerning a single research. The second possibility consists in widening applications of ANNs to areas that are significant from a sociological point of view and that have not been analysed with ANNs. A third possibility consists in getting a better understanding on the way in which ANNs can contribute to the theory/explanation of sociological research. This part is very interesting because ANNs illustrate the concepts of explanation, prediction, etc. from a different perspective. As shown in the literature (Plebe and Grasso 2019), important considerations concerning applications of ANNs or their structure still remain to be explored. Answering these questions can help us push theory or generate new hypotheses. The results from ML provide not an end goal, but the starting point for further analysis and conceptualization. As such, ML tools complement, not replace, existing methods in sociology.

References

Aggarwal CC (2018) *Neural networks and deep learning. A textbook*. Cham, CH: Springer.

Alpaydin E (2016) *Machine learning. The new AI*. Cambridge, MA: The MIT Press.

Athey S (2017) Beyond prediction: using big data for policy problems. *Science* 355(6324): 483-485.

Athey S, Imbens GW (2017) The state of applied econometrics: causality and policy evaluation. *Journal of Economic Perspectives* 31(2): 3–32.

Bail CA (2014) The cultural environment: measuring culture with big data. *Theory and Society* 43(3–4): 465–482.

Balas VE, Kumar R, Srivastava R (eds.) (2020) *Recent trends and advances in artificial intelligence and internet of things*. Cham, CH: Springer.

Baldassarri D, Abascal M (2017) Field experiments across the social sciences. *Annual Review of Sociology* 43(1): 41–73.

Baldassarri D, Goldberg A (2014) Neither ideologues nor agnostics: alternative voters' belief system in an age of partisan politics. *American Journal of Sociology* 120(1): 45–95.

Barocas S, Selbst A (2016) Big data's disparate impact. *California Law Review* 104(3): 671–732.

Bartlett PL, Montanari A, Rakhlin A (2021) Deep learning: a statistical viewpoint. *Acta Numerica* 30(May): 87–201.

Berk R, Heidari H, Jabbari S, Kearns M, Roth A (2021) Fairness in criminal justice risk assessments: the state of the art. *Sociological Methods & Research* 50(1): 3–44.

Blumenstock J, Cadamuro G, On R (2015) Predicting poverty and wealth from mobile phone metadata. *Science* 350(6264): 1073–1076.

Boden MA (2006) *Mind as machine. A history of cognitive science*. Oxford, UK: Clarendon Press.

Bonikowski B, DiMaggio P (2016) Varieties of American popular nationalism. *American Sociological Review* 81(5): 949–980.

Buscema M (1994) *Squashing theory. Modello a reti neurali per la previsione dei sistemi complessi*. (Squashing theory. A neural network model for prediction of complex systems). Rome: Armando.

Buskirk TD, Kirchner A (2021) Why machines matter for survey and social science researchers: exploring applications of machine learning methods for design, data collection, and analysis. In CA Hill, PP Biemer, TD Buskirk, L Japec, A Kirchner, S Kolenikov, et al. (eds.), *Big Data Meets Survey Science. A Collection of Innovative Methods*, 11–62. Hoboken, NJ: Wiley.

Cammarata S (1990) *Reti neuronali. Una introduzione all'altra intelligenza artificiale* (Neural networks. An introduction to the other artificial intelligence). Milan: Etas.

Capecchi V (1996) Tre Castelli, una Casa e la Città inquieta. (Three castles, a house and the restless city). In C Cipolla, A De Lillo (eds.), *Il Sociologo e le Sirene. La Sfida dei Metodi Qualitativi*, 37–99. Milan: FrancoAngeli.

Capecchi V, Buscema M, Contucci P, D'Amore B (eds.) (2010) *Applications of Mathematics in Models, Artificial Neural Networks and Arts*. Dordrecht: Springer.

Davenport D (2013) The two (computational) faces of AI. In VC Müller (ed.), *Philosophy and Theory of Artificial Intelligence*, 43–58. Berlin: Springer.

Di Franco G (1998) Reti neurali artificiali e analisi dei dati per la ricerca sociale: un nuovo paradigma? (Artificial neural networks and data analysis for social research: a new paradigm?) *Sociologia e Ricerca Sociale* 19(56): 35–75.

Di Franco G (2017) *Tecniche e modelli di analisi multivariata.* (Techniques and models of multivariate analysis). Milan: FrancoAngeli.

Di Franco G (2018) *Usi e abusi dei sondaggi politico-elettorali in Italia. Una guida per giornalisti, politici e ricercatori*. (Uses and abuses of electoral polls in Italy. A guide for journalists, politicians and researchers). Milan: FrancoAngeli.

Di Franco G, Marradi A (2013) *Factor analysis and principal component analysis*. Milan: FrancoAngeli.

Dietrich E, Fields C, Sullins JP, Van Heuveln B, Zebrowski R (2021) *Great philosophical objections to artificial intelligence. The history and legacy of the AI wars*. London: Bloomsbury Academic.

Echeberria AL (ed.) (2022) *Artificial intelligence for business. Innovation, tools and practices*. Cham, CH: Palgrave Macmillan.

Elliott A (ed.) (2022) *The Routledge social science handbook of AI*. Abingdon, UK: Routledge.

Evans JA, Aceves P (2016) Machine translation: mining text for social theory. *Annual Review of Sociology* 42(1): 21–50.

Evans J, Foster JG (2019) Computation and the sociological imagination. *Contexts* 18(4): 10–15.

Fabbri G, Orsini R (1993) *Reti neurali per le scienze economiche. I modelli del connessionismo per l'analisi statistica e la simulazione dei comportamenti economici*. (Neural networks for economics. Connectionist models for statistical analysis and simulation of economic behaviour). Milan: Franco Muzzio Editore.

Fetzer JH (2004) The philosophy of AI and its critique. In L Floridi (ed.), *The Blackwell Guide to the Philosophy of Computing and Information*, 119–134. Malden, MA: Blackwell.

Goldthorpe JH (2004) Sociology as social science and cameral sociology: some further thoughts. *European Sociological Review* 20(2): 97–105.

Goodfellow I, Bengio Y, Courville A (2016) *Deep learning*. Cambridge, MA: The MIT Press.

Grimmer J, Stewart BM (2013) Text as data: the promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis* 21(3): 267–297.

Grum M (2022) *Construction of a concept of neuronal modeling*. Wiesbaden: Springer.

Guidotti R, Monreale A, Ruggieri S, Turini F, Giannotti F, Pedreschi D (2019) A survey of methods for explaining black box models. *ACM Computing Surveys* 51(5): 1–42.

Hastie T, Tibshirani R, Friedman J (2009) *The elements of statistical learning. Data mining, inference, and prediction.* New York, NY: Springer.

Hopkins DJ, King G (2010) A method of automated nonparametric content analysis for social science. *American Journal of Political Science* 54(1): 229–247.

Jordan MI, Mitchell TM (2015) Machine learning: trends, perspectives, and prospects. *Science* 349(6245): 255–260.

Kent R (2009) Case-centred methods and quantitative analysis. In D Byrne, CC Ragin (eds.), *The SAGE Handbook of Case-Based Methods*, 184–207. London: SAGE Publications.

Keuschnigg M, Lovsjö N, Hedström P (2018) Analytical sociology and computational social science. *Journal of Computational Social Science* 1(1): 3–14.

Kleinberg J, Ludwig J, Mullainathan S, Obermeyer Z (2015) Prediction policy problems. *American Economic Review* 105(5): 491–495.

Landi A, Piaggi P, Laurino M, Menicucci D (2010) Artificial neural networks for nonlinear regression and classification. In AE Hassanien, A Abraham, F Marcelloni, H Hagras, M Antonelli, T-P Hong (eds.), *Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications*, 115-120. Cairo, Egypt, November 29 - December 1, 2010. IEEE.

Longo L, Goebel R, Lecue F, Kieseberg P, Holzinger A (2020) Explainable artificial intelligence: concepts, applications, research challenges and visions. In A Holzinger, P Kieseberg, AM Tjoa, E Weippl (eds.), *Machine Learning and Knowledge Extraction. 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25–28, 2020, Proceedings, 1–16*. Cham, CH: Springer.

Lu Y (2019) Artificial intelligence: a survey on evolution, models, applications and future trends. *Journal of Management Analytics* 6(1): 1–29.

Molina M, Garip F (2019) Machine learning for sociology. *Annual Review of Sociology* 45(1): 1–25.

Molnar C, Casalicchio G, Bischl B (2020) Interpretable machine learning – A brief history, state-of-the-art and challenges. In I Koprinska, M Kamp, A Appice, C Loglisci, L Antonie, A Zimmermann, et al. (eds.), *ECML PKDD 2020 Workshops. Workshops of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, September 14–18, 2020, Proceedings*, 417–431. Cham, CH: Springer.

Mullainathan S, Spiess J (2017) Machine learning: an applied econometric approach. *Journal of Economic Perspectives* 31(2): 87–106.

Parisi D (1989) *Intervista sulle reti neurali. Cervello e macchine intelligenti*. (Interview on neural networks. Brain and intelligent systems). Bologna: Il Mulino.

Pecere P (2020) *Soul, mind and brain from Descartes to cognitive science. A critical history*. Cham, CH: Springer.

Plebe A, Grasso G (2019) The unbearable shallow understanding of deep learning. *Minds and Machines* 29(4): 515–553.

Rumelhart DE, McClelland JL, the PDP Research Group (1986) *Parallel distributed processing. Explorations in the microstructure of cognition. Volume 1: foundations*. Cambridge, MA: The MIT Press.

Rumelhart DE, McClelland JL, the PDP Research Group (1987) *Parallel distributed processing. Explorations in the microstructure of cognition. Volume 2: psychological and biological models*. Cambridge, MA: The MIT Press.

Schmidhuber J (2015) Deep learning in neural networks: an overview. *Neural Networks* 61(Jan): 85–117.

Strohmaier D (2021) Ontology, neural networks, and the social sciences. *Synthese* 199(1–2): 4775–4794.

Toosi A, Bottino AG, Saboury B, Siegel E, Rahmim A (2021) A brief history of AI: how to prevent another winter (a critical review). *PET Clinics* 21(4): 449–469.

Torgerson WS (1958) *Theory and methods of Scaling*. New York, NY: Wiley.

van der Maas HLJ, Snoek L, Stevenson CE (2021) How much intelligence is there in artificial intelligence? A 2020 update. *Intelligence* 87(Jul–Aug): 1–9.

Varian HR (2014) Big data: new tricks for econometrics. *Journal of Economic Perspectives* 28(2): 3–28.

White H (1989) Learning in artificial neural networks: a statistical perspective. *Neural Computation* 1(4): 425–464.