

Article

Mathematical Approaches for the Characterization and Analysis of Molecular Markers in the Study of the Progression and Severity of Amyotrophic Lateral Sclerosis

Luisa Carracciuolo ^{1,*} , Ugo D'Amora ¹ , Raffaele Dubbioso ²  and Ines Fasolino ¹ 

¹ Institute of Polymers, Composites and Biomaterials (IPCB) of the National Research Council (CNR), 80078 Pozzuoli, Italy; ugo.damora@cnr.it (U.D.); ines.fasolino@cnr.it (I.F.)

² Department of Neurosciences, Reproductive Sciences and Odontostomatology, University of Naples Federico II, 80131 Naples, Italy; raffaele.dubbioso@unina.it

* Correspondence: luisa.carracciuolo@cnr.it

Abstract

Amyotrophic Lateral Sclerosis (ALS) is a progressive neurodegenerative disorder for which despite its severity, no validated biomarker currently exists to support early diagnosis, limiting therapeutic effectiveness and patient survival. In this context, mathematical modeling therefore becomes essential: it allows us to maximize the information obtainable from a limited number of samples, identify patterns that may not be directly observable, and estimate the relative contribution of different molecular markers to ALS progression. In this work, we propose methods for qualitatively and quantitatively evaluating the relevance of selected biomarkers in ALS classification and disease-state identification and laying the foundations for the definition of a protocol useful for constructing “digital twins” of the entire process of study, diagnosis, and treatment of the disease from the perspective of innovative precision medicine.

Keywords: Amyotrophic Lateral Sclerosis; molecular markers; Principal Components Analysis; Multiple Imputation by Chained Equations; deep neural networks

1. Introduction

Amyotrophic Lateral Sclerosis (ALS) is a rapidly progressive adult-onset neurodegenerative disorder that primarily targets upper and lower motor neurons in the cerebral cortex and spinal cord [1]. Despite its severity, no validated biomarker currently exists to support early diagnosis, limiting therapeutic effectiveness and patient survival. The predominantly sporadic nature of ALS, with only 5–10% of cases being hereditary, further complicates early detection. On average, life expectancy is approximately five years from symptom onset [2]. Although traditionally considered a “pure” motor disease, ALS also involves sensory and autonomic dysfunction, as demonstrated in punch skin biopsy studies [3,4]. Reductions in intraepidermal nerve fiber density have been reported in roughly 80% of patients, highlighting peripheral nervous system involvement [5,6]. Emerging evidence also implicates neuroinflammation, characterized by activated microglia, astrogliosis, and immune infiltration, in the spinal cord and peripheral nerves of both ALS patients and transgenic mouse models [7].

In this context, identifying early, reliable biomarkers is crucial to reducing diagnostic delays. One promising candidate pathway is the NOD-, LRR- and pyrin domain-containing protein 3 (NLRP-3) inflammasome. Although NLRP-3 activation has been extensively



Received: 9 December 2025

Revised: 28 January 2026

Accepted: 29 January 2026

Published: 5 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\) license](https://creativecommons.org/licenses/by/4.0/).

described in other neurodegenerative diseases, its role in ALS has only recently gained attention [8]. Pathological studies show *NLRP-3* activation in the brain and spinal cord of SOD1G93A mice and in human ALS tissue [9]. In SOD1 models, increased *NLRP-3* correlates with active caspase-1, elevated interleukin-18 (*IL-18*) and *IL-1 β* [10]. Upregulation of microglial *NLRP-3* has also been observed in TDP-43Q331K mice, and in vitro studies demonstrate that TDP-43 can trigger microglial NLRP3-mediated inflammation toxic to motor neurons [11]. Building on this rationale, the present work proposes the application of mathematical tools capable of extracting relevant insights from molecular data. A comprehensive biological characterization of ALS would ideally require the collection of a large number of skin biopsies and blood samples across multiple disease stages. However, such extensive sampling is often impractical due to the invasive nature of biopsies, patient burden, and clinical resource constraints. Mathematical modeling therefore becomes essential: it allows us to maximize the information obtainable from a limited number of samples, identify patterns that may not be directly observable, and estimate the relative contribution of different molecular markers to ALS progression.

Within this framework, we developed methods for qualitatively and quantitatively evaluating the relevance of selected biomarkers in ALS classification and disease-state identification. To generate the molecular dataset required for model training and validation, we used human skin biopsy and blood serum as minimally invasive tools to investigate the pathophysiological mechanisms underlying peripheral nerve degeneration in ALS. Specifically, we quantified inflammatory responses, such as *NLRP-3* inflammasome activation and cytokine production, alongside oxidative stress indicators (Superoxide Dismutase, SOD expression) and pathological phospho-TDP-43 accumulation. Skin and blood samples from ALS patients and healthy controls were processed using complementary analytical approaches, including Western blotting, ELISA, and fluorescence imaging.

By integrating these molecular measurements with mathematical analysis, our approach enables a more robust interpretation of limited clinical data and supports the identification of biomarker patterns that could be used in future diagnostic applications.

In particular, the novelty of our work, to the best of our knowledge regarding the current state-of-the-art, lies in an interdisciplinary approach that combines biochemical data from ALS patients with advanced computational techniques—such as Multiple Imputation Methods (MIMs), Principal Component Analysis (PCA), and neural networks (NNs)—to

- Integrate MIM, PCA, and NN within the ALS biomedical context;
- Use PCA and BiPlots, and NN for biomarker relevance interpretation by also using some simple techniques to open NN “black-box”;
- Combine a mathematical–biological workflow as a foundation for future “digital twin” construction.

To the best of our knowledge, this study is the first to formally combine mathematically interpretable multivariate modeling with biochemically validated correlations among oxidative, neuroinflammatory, and neurodegenerative biomarkers of ALS across both serum and peripheral tissues. Previous studies have shown that peripheral cutaneous nerve degeneration can mirror neurodegenerative mechanisms occurring in the central nervous system [5]. However, unlike earlier investigations, we specifically quantified protein biomarkers within peripherally innervated tissue and directly compared them with circulating serum markers. This integrative approach lays the groundwork for the future development of minimally invasive strategies (such as skin biosensors) aimed at detecting ALS progression and disease severity through cutaneous, peripherally innervated tissue.

The paper is structured as follows: Section 2 describes the biochemical analysis and the mathematical framework used for data analysis; Section 3 presents case-study results illustrating the application of these methods; and Section 4 discusses the implications

of these findings, summarizing the study's contributions and outlining future research directions in Section 5.

2. Materials and Methods

2.1. Collection of Skin Biopsies and Serum

Sixty patients diagnosed with ALS within eighteen months from symptom onset and classified as “definite”, “probable” or “probable laboratory-supported” according to the revised El Escorial criteria were enrolled at the ALS Center of Federico II University, along with thirty-two healthy controls selected from the patients' caregivers. Individuals under the age of eighteen and those with coexisting conditions that could affect the peripheral nervous system (e.g., glucose intolerance, diabetes, thyroid disorders, vitamin deficiencies) were excluded.

This study was designed as an exploratory analysis to investigate biochemical alterations in skin biopsies and serum, which required highly specific inclusion criteria and standardized sampling procedures, as reported in similar studies [12], providing sufficient statistical power for the primary outcomes.

2.2. Biochemical Assays

Skin biopsies and blood samples were collected from the sixty ALS patients to assess the expression levels of inflammasome pathway components in comparison to healthy controls. Specifically, two 3 mm punch biopsies were taken at baseline (T0) from the thigh, leg, and fingertip on the side most affected by the disease. Samples were fixed in cold Zamboni's solution (Sigma Aldrich, Milan, Italy) and cryoprotected in 20% sucrose phosphate-buffered saline (PBS, Sigma Aldrich, Milan, Italy). Then, skin biopsies were homogenized for Western blot, ELISA, and SOD assays. Blood was drawn into tubes containing anticoagulant and centrifuged ($1000\text{--}2000 \times g$ for 10 min in a refrigerated centrifuge) to obtain serum samples.

2.3. Mathematical Tools

Since the 1960s, figures such as John Wilder Tukey (the inventor, along with J. W. Cooley, of the FFT algorithm, one of the most elegant, beautiful, and powerful in the history of mathematics) argued that data analysis could be considered a discipline that could also be independent of mathematics. David Donoho, who was Tukey's student and is the author of some works related to the present and future of Data Science [13,14], in a position lecture presented at AMS Math Challenges 2020 [15], summarized Tukey's position as follows:

Data are here now; they will be coming on more and more in the future. We must analyze them, often using very humble means, and insistence on mathematics—for example, on deep mathematical analysis and proof—will likely distract us from recognizing these fundamental points.

But, in the same lecture, Donoho states

... while over the last forty years, data analysis has developed by separating itself from its mathematical parent, there is now need for a reconnection, which might yield benefits both for mathematics and for data analysis.

and

We are now in a setting where many very important data analysis problems are high-dimensional. Many of these high-dimensional data analysis problems require new or different mathematics. A central issue is the “curse of dimensionality”, which has ubiquitous effects through out the sciences.

Donoho well expressed the idea that, given the complexity of data, effective and efficient data analysis cannot (no longer) ignore a robust knowledge of mathematical approaches (both old and new).

Following this section, mathematical tools or approaches will be shortly introduced as a strategy to solve problems very relevant in the context of clinical research related to imputation of missing data, data dimensionality reduction, and building predictive models from data.

Clinical research frequently has missing data. When the values of the variables of interest are not measured or documented, this is known as missing data. A patient's unwillingness to answer a particular question, a patient's loss to follow-up, an investigator or mechanical error, or a doctor's failure to order certain investigations for a patient are some of the reasons why data may be missing [16–19]. Replacing the missing values of variables with reasonable values is one way to get around the limits of a full case analysis. Because one is imputing a value of the variable for those individuals with missing data on that variable, this method is known as “imputation” [20].

Large-scale patient data, genomic research, and medical imaging produce enormous volumes of high-dimensional data in healthcare research that need to be processed effectively. These datasets become computationally costly and difficult to analyze in the absence of appropriate dimensionality reduction. The term “dimensionality reduction” describes methods for generating a set of principal variables by reducing the high dimensional space of the original data into a low dimensional space. Healthcare researchers can improve disease prediction models and facilitate more precise clinical decision-making by extracting important features from complicated datasets through dimensionality reduction. Dimensionality reduction, for instance, aids in precision medicine by identifying important genetic markers linked to diseases in genomic investigations. Principal Component Analysis (PCA) is the best statistical method to reduce dimensionality and remove redundancy if every feature is equally relevant to the investigation [21]. Building a predictive model from data [22,23] has once again become a necessary approach, given the large amount of data available from which to extract knowledge. Techniques for creating these models, particularly those based on machine learning techniques, are quickly becoming crucial for the interpretation of intricate clinical and other data as well as for clinical decision support. The link between input and output signals of a complex system can be approximated using neural networks (NNs), which are highly parameterized, non-linear models with collections of processing units known as neurons. NNs are sometimes referred to as say black boxes, even though they can be more effective predictors than more traditional models (like linear regression). NN models are much harder to interpret than linear techniques, and it can be difficult to determine which descriptors (predictors) are most significant and how they relate to the property being modeled [24].

In the following part of this section, some methods will be proposed and briefly described. They could be used to solve the aforementioned problems afflicting clinical research based on data.

2.3.1. Handling Incomplete Data

As mentioned above, if the problem of missing data is very widespread in the field of medical research it is also true that it is a problem that affects many other contexts: see, for example, the long list available in [25] in which, in addition to the long list of problems related to medicine and health, applications related to management sciences, politics, psychology and sociology can also be highlighted.

For all these listed applications, a multiple imputation approach for a missing data problem is proposed since it has a proven effectiveness in multiple contests [17]. In particular, some form of the so called “chained equation approach” is proposed.

Here, we describe, and use, a version of that approach which is known as “Multivariate Imputation by Chained Equations (MICE)” [20,25,26]. Given the variables used in the imputation process, MICE operates on the assumption that missing data are “Missing At Random (MAR)”, meaning that the probability that a value is missing depends only on seen values and not on unobserved values [26]. Although using MICE when data are not MAR may lead to biased results, some research indicates that, even in these circumstances, MICE produces fewer biased estimates than naive approaches to managing the same censored values [17].

Definition 1 (The Multivariate Imputation by Chained Equations (MICE) method). *Let Y be a multivariate variable based on p univariate variables among which k are incomplete. Such variable Y is represented by n observations.*

Then,

$$Y = (Y_1, \dots, Y_k, \bar{Y}_1, \dots, \bar{Y}_{p-k})$$

where $\{Y_j, j = 1, \dots, k\}$ denotes the set of the incomplete variables and $\{\bar{Y}_j, j = 1, \dots, p - k\}$ denotes the set of $p - k$ complete variable. Let the observed and missing parts of Y_j be denoted by Y_j^{obs} and Y_j^{mis} , respectively, so $Y^{obs} = (Y_1^{obs}, \dots, Y_k^{obs}, \bar{Y}_1, \dots, \bar{Y}_{p-k})$ and $Y^{mis} = (Y_1^{mis}, \dots, Y_k^{mis})$ stand for the observed and missing data in Y . Let

$$Y_{-j} = (Y_1, \dots, Y_{j-1}, Y_{j+1}, \dots, Y_k, \bar{Y}_1, \dots, \bar{Y}_{p-k})$$

denotes the multivariate variable depending from all the variables in Y except Y_j .

Suppose that Y is partially observed random sample from the p -variate multivariate distribution $P(Y|\theta)$. We assume that the multivariate distribution of Y is completely specified by θ , a vector of unknown parameters.

The MICE method has the aim to obtain the multivariate distribution of θ , either explicitly or implicitly. Given the parameters $\theta_j, j = 1, \dots, k$, which are specific to the respective conditional variable Y_j , the MICE algorithm (see Algorithm A1 in Appendix A.1) implementing such method obtains the posterior distribution of θ by sampling iteratively from conditional distributions of the form

$$P(Y_1|Y_{-1}, \theta_1)$$

$$\vdots$$

$$P(Y_k|Y_{-k}, \theta_k).$$

Given a model $\mathcal{M}_j(Y_j^{obs}, Y_{-j}; \theta_j), j = 1, \dots, k$, defined by parameter θ_j , to impute from observed (and already imputed) marginal distributions, the iterative procedure can be considered a Gibbs sampler that is used m times to calculate a number m of independent samples $^{(i)}Y, i = 1, \dots, m$ of the complete variable where each of the $^{(i)}Y$ inherits the observed part Y^{obs} of Y filling the missing part Y^{mis} of Y by “imputed” observations. That is, if $Y_j^{(t)} = (Y_j^{obs}, Y_j^{(t)})$ is the j -th imputed variable at iteration t ,*

$$^{(i)}Y = (Y_1^{(maxit)}, \dots, Y_k^{(maxit)}, \{\bar{Y}_{j'}\}_{j'=1, \dots, p-k}).$$

All the m independent samples $^{(i)}Y, i = 1, \dots, m$ can be collected in the set X for further analysis.

Regarding the selection of the number of imputations m , the desired “relative efficiency” $E_{MI}(m, \lambda)$ of MI estimates [27] should be considered. $E_{MI}(m, \lambda)$ can be evaluated by using the following formula [17]

$$E_{MI}(m, \lambda) = \left(1 + \frac{\lambda}{m}\right)^{-1}, \quad (1)$$

where λ is the rate of missing data. See Table A1 in Appendix A.1 for the listing for some values of $E_{MI}(m, \lambda)$.

To overcome problems related with skewed data or data not described by linear models, the *Predictive Mean Matching* (PMM) method [28] is proposed as a flexible model $\mathcal{M}_j(\cdot, \cdot; \theta_j)$ for imputation. As described in [29], PMM is based on the following steps:

Model Building: A predictive model $\hat{y} = f(X)$ is built using complete cases to estimate the relationship between the target variable $\hat{y}$ (to be imputed) and a set of predictor variables represented by X .

Prediction Generation: This model is employed to predict values for both complete and incomplete cases. Let $\hat{y}_i$ be the predicted value for the i -th observation of $\hat{y}$.

Matching Process: For each incomplete case, PMM identifies one or several complete cases whose predicted values are closest in distance to the incomplete case is predicted value. For each missing instance i , calculate:

$$j_{dmin} = \arg \min_{j \in \mathcal{C}} \text{dist}(\hat{y}_i, \hat{y}_j)$$

where $\mathcal{C}$ is the indexes list for all completed observations.

Imputation by Matching: Replace the missing value $\hat{y}_i$ with the observed value from the best-matching candidate

$$\hat{y}_i = \hat{y}_{j_{dmin}}.$$

Optionally, multiple matches can be pooled to inject further randomness and variability into the imputation process.

2.3.2. Interpreting Multidimensional Data by Principal Components Analysis

In several fields, large datasets are becoming more and more common. Such datasets must be considerably reduced in dimensionality in an interpretable manner while maintaining the majority of the data’s content in order to be interpreted. Although many methods have been developed for this purpose, one of the oldest and most used is “*Principal Component Analysis (PCA)*”. Its concept is straightforward: lower a dataset’s dimensionality while maintaining as much “variability” (i.e., statistical information) as feasible [30].

PCA’s goal is to extract the important information from a set of observed data, described by several inter-correlated variables, to represent it as a set of new orthogonal variables called “principal components”, and to display the pattern of similarity of the observations and of the variables as points in appropriate maps called BiPlot [31]. The main uses of PCA are descriptive, rather than inferential [30]. “*PCA allows us to simultaneously describe the association between variables, as well as the resemblance among individuals. PCA can also be regarded to as a dimension reduction technique of quantitative variables, often employed as an intermediate step towards a subsequent model building phase [32]*”. Mathematically, PCA depends upon the solution to an eigenproblem or, alternatively, upon the singular value decomposition (SVD) of the (centered) data matrix. A complete description of the PCA approach can be found in [30,33]. For an informal but more descriptive approach in introducing PCA, we also suggest reading a paper by Aluja et al. [32]. Here, we propose some definitions and descriptions that are useful for our discussion of the proposed results.

Definition 2 (Principal Components Analysis (PCA) of a dataset Correlation Matrix). Let $\mathcal{D}$ a dataset with observations on p numerical variables, for each of n entities or individuals. These data values define an n times p data matrix $\mathbf{X}$, whose j -th column is the vector $\mathbf{x}_j$ of observations on the j -th variable. Let $\mathbf{Z}$ be the normalized data matrix of $\mathbf{X}$; then, for its generic elements, z_{ij} is valid as follows:

$$z_{ij} = \frac{y_{ij}}{s_j},$$

where $\bar{x}_j$ is the mean value of the n observations of variable j , where $y_{ij} = x_{ij} - \bar{x}_j$ and where s_j is the standard deviation of the elements $\{y_{ij}\}_{1,\dots,n}$, i.e., $s_j = \sqrt{\frac{\sum_i y_{ij}^2}{n-1}}$.

Let $\mathbf{R}$ be the sample covariance matrix related to $\mathbf{Z}$; $\mathbf{R}$ is a positive semi-definite matrix. Then, it has an eigen-decomposition such as the following one:

$$\mathbf{R} = \mathbf{A}\mathbf{\Lambda}\mathbf{A}^T, \quad (2)$$

where the p column vectors $\mathbf{a}_j \in \mathbb{R}^p$ of matrix $\mathbf{A}$ are the p linearly independent eigenvectors of $\mathbf{Z}$ defining an orthonormal set of vectors, i.e.,

$$\mathbf{a}_j \mathbf{a}_{j'}^T = \begin{cases} 1, & \text{if } j = j', \\ 0, & \text{otherwise.} \end{cases}$$

Let us define the set of vectors $\mathbf{b}_j, j = 1, \dots, p$ as

$$\mathbf{b}_j = \mathbf{Z}\mathbf{a}_j \quad (3)$$

Since the matrix $\mathbf{R}$ coincides with the correlation matrix of $\mathbf{X}$, the vectors defined in (3) are called Principal Components (PCs) of the Correlation Matrix of the $\mathcal{D}$ dataset, and this is called a “Normalized PCA”, which is the PC approach based on those vectors.

It can be shown [30,31] that $\mathbf{A}$ is a solution of the following optimization problem:

$$\mathbf{A} = \underset{\substack{\mathbf{Q}: \mathbf{Q}^T \mathbf{Q} = \mathbf{I} \\ \mathbf{Z}\mathbf{Q} \text{ orthogonal}}}{\arg \max} \quad \text{var}(\mathbf{Z}\mathbf{Q}). \quad (4)$$

Equation (4) expresses the essence of Principal Component Analysis which is related to identify a “projection matrix $\mathbf{Q}$ with orthonormal columns”, which transform the “cloud of points”, representing the observations $\mathbf{X}$ in a p dimensional space, in such a way that the new configuration, represented in a r -dimensional space $r \leq p$, is as close as possible to the original configuration (i.e., the distances among all different points are preserved as much as possible; see Figure 1). Let us call the new space a “Factorial Space”, or a “Factorial Plane” if $r = 2$.

When variables used in a dataset have different units of measurement, it is common practice to begin by standardizing the variables as in Definition 2. Correlation matrix PCs are therefore the best option for datasets where various scale variations are possible for each variable, since they are invariant to linear changes in units of measurement. For all these reasons, some statistical software assumes by default that a PCA means a normalized PCA [30].

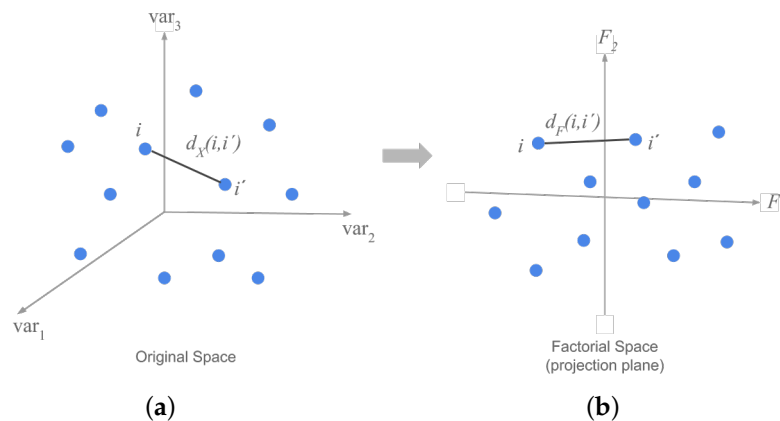


Figure 1. Cloud of points in the original p -dimensional space (a) and the cloud of transformed points in the new Factorial Space (b). $d_X(i, i')$ and $d_F(i, i')$ denote, respectively, the distance, between two different points i , and i' , in the original space and in the Factorial Space. Credits: Aluja et al. [32].

In standard PCA terminology, the elements of the eigenvectors a_j are commonly called “PC loadings”, while the elements of vectors b_j are called “PC scores”, as they are the values that each individual would score on a given PC [30].

It is noteworthy that PCA is related to a Singular Value Decomposition (SVD) of the matrix Z (see Proposition A1 in Appendix A.2).

Diagonal elements of matrix Λ can be used to evaluate the variance of the projection $projection(z_j, a_j)$ of column vectors z_j of Z (i.e., the variables) on the computed components $a_k, k = 1, \dots, p$. In particular,

$$var(projection(z_j, a_j)) = \Lambda_{jj}. \quad (5)$$

where a normalized PCA is considered as follows:

$$Var_{Tot} \sum_j^p var(projection(z_j, a_j)) = \sum_j^p \Lambda_{jj} = p.$$

It is also possible to define the *Relevance*(j) of the components $j, j = 1, \dots, p$ in representing the total variance Var_{Tot} , as well as the cumulative relevance *CumulativeRelevance*(j) of the first j components, as

$$Relevance(j) = \frac{\Lambda_{jj}}{p} \times 100 \quad (6)$$

$$CumulativeRelevance(j) = \sum_{j=1}^j Relevance(j) \quad (7)$$

Thanks to the reformulation (A2) of PCA, one could assert that PCA “... is at heart a dimensionality-reduction method, whereby a set of p original variables can be replaced by an optimal set of q derived variables, the PCs.” [30], where $q < p$. In fact, consider the “Reduced-Rank” Zq approximation of matrix Zr (where $q < r$) (see Proposition A2 in Appendix A.2).

The BiPlot is a helpful tool for data analysis that makes it possible to visually evaluate the structure of data matrices. It is particularly useful in Principal Component Analysis, where the BiPlot may exhibit variances and correlations of the variables, as well as inter-unit distances and unit clustering [34]. Also, thanks to PCA, this graphing method can exploit the opportunity offered by PCA to approximate the data matrix by a matrix product of dimension 2 [32]. In a BiPlot, the individuals z_i and the variables z_j , of a normalized data matrix $Z \in \mathbb{R}^{n \times p}$ defined as in Definition 2, are graphically represented, respectively,

as points and as vectors (i.e., arrows) in a bidimensional Cartesian system (see Proposition A3 in Appendix A.2 for its description). The BiPlot is based on an approximation $\hat{\mathbf{Z}}$ of $\mathbf{Z}$ defined by the product

$$\hat{\mathbf{Z}} = \hat{\mathbf{Z}}_1 \hat{\mathbf{Z}}_2^T,$$

where $\hat{\mathbf{Z}}_1 \in \mathbb{R}^{n \times 2}$ and $\hat{\mathbf{Z}}_2 \in \mathbb{R}^{p \times 2}$, and where the rows of $\hat{\mathbf{Z}}_1$ and of $\hat{\mathbf{Z}}_2$ represent, respectively, the individuals and the variables.

As suggested in [30], and thanks to the definition of the bidimensional Cartesian system $\mathcal{C}^{(j_1, j_2)}$ on which the BiPlot is based (see Proposition A3 in Appendix A.2), the BiPlot has the following properties:

- The cosine of the angle between any two vectors representing variables is the coefficient of correlation between those variables.
- Similarly, the cosine of the angle between any vector representing a variable and the axis representing a given PC is the coefficient of correlation between those two variables.
- The inner product between the markers for individual i and variable j gives the value of individual i on variable j . The practical implication of this result is that orthogonally projecting the point representing individual i onto the vector representing variable j recovers the value z_{ij} .
- The Euclidean distance between the markers for individuals i and i' is proportional to the “Mahalanobis distance” [35] between them (see [33] for more details).

Roughly speaking,

Interpreting Points: The relative location of the points can be interpreted. Points that are close together correspond to observations that have similar scores on the components displayed in the plot. To the extent that these components fit the data well, the points also correspond to observations that have similar values on the variables.

Interpreting Vectors: Both the direction and length of the vectors can be interpreted. Vectors point away from the origin in some direction. With the principal components under consideration, a vector direction is associated with the highest correlation. The squared multiple correlation between the projected variable and the components under consideration determines the vector’s length. As a result, variables with comparable response profiles and meanings within the context of the data are represented by vectors pointing in the same direction. The observations with the greatest amount of variable measures are those whose points project the farthest in the direction of the vector points. The points with the least amount are those that project at the opposite end. The amount of those projecting in the middle is average.

2.3.3. Building Predictive Model from Data by Neural Networks

Neural networks, a cornerstone of artificial intelligence (AI) and machine learning, are computational models built from data inspired by the structure and function of the human brain [36]. The concept of neural networks dates back to the mid-20th century, but it was not until the advent of powerful computers and the availability of large datasets in the 21st century that neural networks truly flourished.

A neural network consists of layers of neurons (nodes), where each neuron is a function that takes inputs, processes them using weights, biases, and an activation function, and produces an output. A deep neural network (DNN) can be considered the result of putting more than one level of neurons one after another. Figure 2 shows an example of a DNN composed of 3 layers. The numbers N_1 , N_2 and N_3 of “neurons” at each of the three layers are, respectively, 3, 4 and 2. If the number L of the DNN is equal to three, the DNN is called a “shallow neural network”.

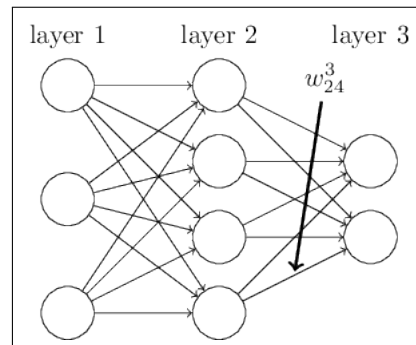


Figure 2. An example of a DNN which is composed of $L = 3$ layers. The numbers N_1 , N_2 and N_3 of “neurons” at each of the three layers are 3, 4 and 2, respectively. Credits: [37].

We recall that the term “Data fitting” denotes the process of constructing a mathematical function $g : \mathbb{R}^n \rightarrow \mathbb{R}^k$ (the model) that has “the best fit” to a series of data points $\{(x_i, y_i)\}_{i=1, \dots, m} : x_i \in \mathbb{R}^n, y_i \in \mathbb{R}^k$. Curve fitting can involve either interpolation where an exact fit to the data is required (i.e., $g(x_i) = y_i, \forall i = 1, \dots, m$) or smoothing in which a “smooth” g function is constructed that approximately fits the data (i.e., $\| [g(x_i)]_{i=1, \dots, m} - [y_i]_{i=1, \dots, m} \| < \epsilon$) for some small value for ϵ and some norm $\|\cdot\|$ defined on $\mathbb{R}^m$. The most widely used approach, especially in the field of machine learning based on neural networks, to build and use a data-driven model is related to a “Data fitting” smoothing process, where a function $f_\alpha : \mathbb{R}^n \rightarrow \mathbb{R}$, defined through a set of k parameters $\alpha = \{\alpha_j\}_{j=1, \dots, k}$, should be determined by a “learning” process on known information to be subsequently used to “predict/describe” new ones. In detail, this looks as follows:

Learning phase Given a set of m data points $\{(x_i, y_i)\}_{i=1, \dots, m}$, let $f_\alpha : \mathbb{R}^n \rightarrow \mathbb{R}$ be a function defined by a set of k parameters $\alpha = \{\alpha_j\}_{j=1, \dots, k} : \alpha_j \in \mathbb{R}$, organized in the vector $\alpha \in \mathbb{R}^k$. The aim of the learning process, given a loss (or “cost function”) C , is to compute the following minimum:

$$\alpha_{Best} = \operatorname{argmin}_{\alpha \in \mathbb{R}^k} C(\{(f_\alpha(x_i), y_i)\}_{i=1, \dots, m}) \quad (8)$$

Predict phase After learning from known information, the best fit function $f_{\alpha_{Best}}$, where α_{Best} is the solution of problem (8), $f_{\alpha_{Best}}$ can be used to “predict/describe” unknown information y_{unkown} about new data x_{new}

$$y_{unkown} = f_{\alpha_{Best}}(x_{new}) \quad (9)$$

Definition 3 defines the form of the fitting function $f^\mathcal{N}$ related to a DNN composed of L layers.

Definition 3 (Fitting function $f^\mathcal{N}$ of a DNN). Let $\mathcal{N}$ be a DNN composed of L layers and let N_l be the number of “neurons” in the l -th layer of $\mathcal{N}$. The fitting function $f^\mathcal{N}$ of $\mathcal{N}$ has the form of the following function compositions:

Let $f : A \rightarrow B$ and $g : B \rightarrow C$ be two functions. Then, the composition of f and g , denoted by $g \circ f$, is defined as the function $g \circ f : A \rightarrow C$ given by $g \circ f(x) = g(f(x)), \forall x \in A$.

Given a set of M functions $\{f_i | i = 1, \dots, M\}$ such that $f_i : A_i \rightarrow A_{i+1}$, with the symbol $\bigcirc_{i=1}^M f_i$, we have

$$\bigcirc_{i=1}^M f_i = f_M \circ f_{M-1} \circ \dots \circ f_2 \circ f_1.$$

$$f^{\mathcal{N}} \left(\mathbf{x}, \left\{ w_{i^l j^l}^{(l)} \right\}_{\substack{l=2, \dots, L \\ i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}} \right) = \bigcirc_{l=2}^L f_l^{\mathcal{N}} \left(\mathbf{x}^{l-1}, \left\{ w_{i^l j^l}^{(l)} \right\}_{\substack{i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}} \right), \quad (10)$$

where $w_{i^l j^l}^{(l)}$ is the so called “weight” from the j^l -th “neuron” in the $(l-1)$ -th layer to the i^l -th “neuron” in the l -th layer [37]. Each function $f_l^{\mathcal{N}}$ is defined as the composed function

$$f_l^{\mathcal{N}} \left(\mathbf{x}^{l-1}, \left\{ w_{i^l j^l}^{(l)} \right\}_{\substack{i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}} \right) = \sigma(\mathbf{y}^l) \quad (11)$$

where $\sigma(y)$ is the so called “activation function” and where

$$\mathbf{y}^l = \mathbf{W}^{(l)} \begin{bmatrix} \mathbf{x}^{l-1} \\ 1 \end{bmatrix} \quad (12)$$

$$\sigma(\mathbf{y}^l) = \left(\sigma(y_{i^l}^l) \right)_{i^l=1, \dots, N_l} \quad (13)$$

$$\mathbf{W}^{(l)} = \left(w_{i^l j^l}^{(l)} \right)_{\substack{i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}} \quad (14)$$

$$\mathbf{x}^1 = \mathbf{x} \quad (15)$$

The value

$$N = \sum_{l=2}^{L-1} N_l$$

of the total number of nodes in the hidden layers ($1 < l < N$) is called “Complexity of DNN $\mathcal{N}$ ”.

The success of such models is also due to their approximation capabilities for which neural networks (NNs) are known as “Universal Approximators” [38–42] in the sense that they can approximate arbitrarily well any continuous function of n variables on a compact domain [42]. Theorem A1 (see Appendix A.3) expresses a universal approximation concept in a more formal way [42,43].

If just the “shallow neural networks” $\mathcal{N}_{\theta}(n, k), \theta \in \Theta : L = 3$ are considered, some results about the computational complexity of computing a $O(\epsilon)$ approximation of a function g by a fitting function defined on $\mathcal{N}_{\theta}(n, k)$ is available from [42] and is stated by Theorem A2 in Appendix A.3.

Theorem A2 (see Appendix A.3) states that “shallow neural networks” are able to give any desirable approximation ϵ to any function $g \in W_m^n$ at a computational cost that grows exponentially with n unless the smoothness of the approximant is increased.

In [42], Poggio et al. answer to some questions about “Which classes of functions can it approximate and learn well?”, giving the message that

.. deep networks have the theoretical guarantee, which shallow networks do not have, that they can avoid the “curse of dimensionality” for an important class of problems, corresponding to ... hierarchically local compositional functions where all the constituent functions are local in the sense of bounded small dimensionality.

Since an optimization problem as in (8) should be solved in the context of machine learning based on DNNs to find a good approximation $f \in \mathcal{F}(n, k)$ of a function g that “models data”, particular attention should be paid to the conditions that guarantees the

existence of optimization problem solution and the effectiveness of the algorithms used to compute such solution numerically [44,45].

In the context of machine learning [46,47], the most commonly used algorithm to compute the solution of problem (8) is the *gradient descent algorithm* [48] based on the *Steepest Descent method* [49]. Gradient descent is an algorithm (see Algorithm A2 in Appendix A.3) for unconstrained mathematical optimization. It is an iterative algorithm used for finding a local minimum of a differentiable multivariate function. It is based on the idea to take repeated steps in the opposite direction of the gradient $\nabla_{\alpha} C(\alpha)$ (or approximate gradient) of the function $C(\alpha)$ at the current point, as this is the direction of steepest descent.

The gradient $\nabla_{\alpha} C(\alpha)$ is a vector in $\mathbb{R}^k$ defined as

$$\nabla_{\alpha} C(\alpha) = \begin{bmatrix} \frac{\partial C(\alpha)}{\partial \alpha_1} \\ \vdots \\ \frac{\partial C(\alpha)}{\partial \alpha_k} \end{bmatrix}$$

It is possible to guarantee the convergence to a local minimum under certain assumptions on the function C (for example, C is a convex function and ∇C is a Lipschitz functions) and particular choices of γ_n .

The “*Learning phase*” in the context of a DNN is then related with the aim to compute “the best values” for the “weights” $w = \left(w_{i^l, j^l}^{(l)} \right)_{\substack{l=2, \dots, L \\ i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}}$, given a “cost function” $C(w)$, by

using Algorithm A2, taking into consideration the form of the “fitting function” f defined in Definition 3. Due to the nature of the cost function C and fitting function f , the gradient estimation needed at line 6 of Algorithm A2 can be computed by the “*backpropagation method*” [50].

Some studies such as “*illuminating the NN black box*” contribute to the literature on this matter [51,52]. Among them, we mention those that use the set of weights w to interpret predictor variable contributions in neural networks, such as the Garson and Olden methods, where the second is an evolution of the first one [52,53]. An alternate and more adaptable way for assessing variable relevance is the Olden method. Relevance is determined by multiplying the raw input-hidden and hidden-output connection weights for each input and output neuron and then summing the product across all hidden neurons. Unlike Garson’s technique, which only takes into account the absolute magnitude, this method preserves the relative contributions of every connection’s weight in terms of both magnitude and sign. For instance, Garson’s approach may produce misleading outcomes based on the absolute magnitude, while the connection weights that change sign (i.e., from positive to negative) between the input-hidden to hidden-output layers could have a canceling effect. The ability of Olden’s algorithm to evaluate neural networks with multiple hidden layers and response variables is an additional advantage. In the case of of a shallow neural network, the Olden method calculates the relevance coefficient r_i^q of the i -th input factor to q -th output using the product of the connection weights among the input layer neurons, hidden layer neurons and output layer neurons of the neural network by via following formula:

$$r_i^q = \frac{\alpha_i^q}{\sum_{i'=1}^n \alpha_{i'}^q}, \quad (16)$$

where

$$\alpha_i^q = \sum_{j=1}^N \frac{w_{ij}^{(2)} w_{jq}^{(3)}}{\beta_j},$$

$$\beta_j = \sum_{i'=1}^n w_{i'j}^{(2)}.$$

Agreeing with Brownlee [54] who stated

... The objective of a neural network is to have a final model that performs well both on the data that we used to train it (e.g., the training dataset) and the new data on which the model will be used to make predictions ...

The main aim, in building such models, is to solve the challenging problem of defining models then can “generalize well” to new data [54]. Two cases indicate that the model fails in “generalization”: “Overfit” and “Underfit” models.

Underfit Model A model that performs poorly on a training dataset and does not perform well in predicting future observations reliably.

Overfit Model A model that corresponds too closely or exactly to a particular set of data and may therefore fail to fit to additional data or predict future observations reliably [55].

Underfitting by addressed by increasing the “capacity of the model”. Increasing such capacity is easily achieved by changing the structure of the model, such as adding more layers and/or more nodes to layers [54]. If an underfitting problem can be easily solved, this is not true for an overfitting case. Nonetheless, two ways can be considered to approach an overfit model based on an NN: training the network on more examples and changing the capacity of the network. As summarized by Brownlee [54], when the number of available data for training is limited, only the option of changing the network’s capacity can be taken advantage of by using one of the following two ways:

- By changing the network structure (number of weights). This is called “structural stabilization” [54].
- By changing the network parameters (values of weights) through the use of “regularization techniques” which, involving the addition of a penalty term to the “Cost function”, has the aim to constrain the values of the weights [54].

The term “regularization” is borrowed from the context of numerical analysis where “regularization approaches” are used to transform an ill-posed problem into a more “stable” one. A problem is said to be ill-posed if small changes in the given information cause large changes in the solution. This instability with respect to the data makes solutions unreliable because small measurement errors for uncertainties in parameters may be greatly magnified and lead to wildly different responses. For an introductory description of the main regularization techniques in the context of NN-based models, we suggest reading Brownlee [54].

In summary, despite all the limitations related to the stability, convergence and interpretability of the algorithms on which NNs are based, they can be considered very powerful tools to model (also well approximating) any general phenomena thanks to their ability to express non-linear models and their nature of “Universal Approximators”. In this study, we will, then, evaluate the applicability of a shallow neural network of complexity N to build models for “supervised classification” (see Section 3.5) because of the following reasons:

1. From Theorem A2, they can be considered able to give any desirable approximation of order ϵ to any function.

2. No information exists about observed data that gives the chance to reformulate the model by a hierarchically local compositional functions that can then justify a deep structure of an NN.

The choice for the values of complexity N is oriented by a compromise (see Equation (A13)) between the computational cost that grows exponentially with the dimension n of observations' space and the smoothness of the approximant that conditions model generalization.

3. Results

This section will describe and comment on the tests performed to analyze which of the markers described in Section 3.1, and represented by “mathematical” variables introduced in Section 3.2, can be considered most useful and significant in identifying the disease state.

Given the high level of missing data, these were first be subjected to an imputation process (see Section 3.3) in order to be then (1) analyzed using PC-based approaches (see Section 3.4) and (2) used to build NN-type predictive models for supervised classification (see Section 3.5). In Figure 3, the flow-chart of of the different phases, involving mathematical tools introduced in Section 2, of qualitative and quantitative analyses of data is shown.

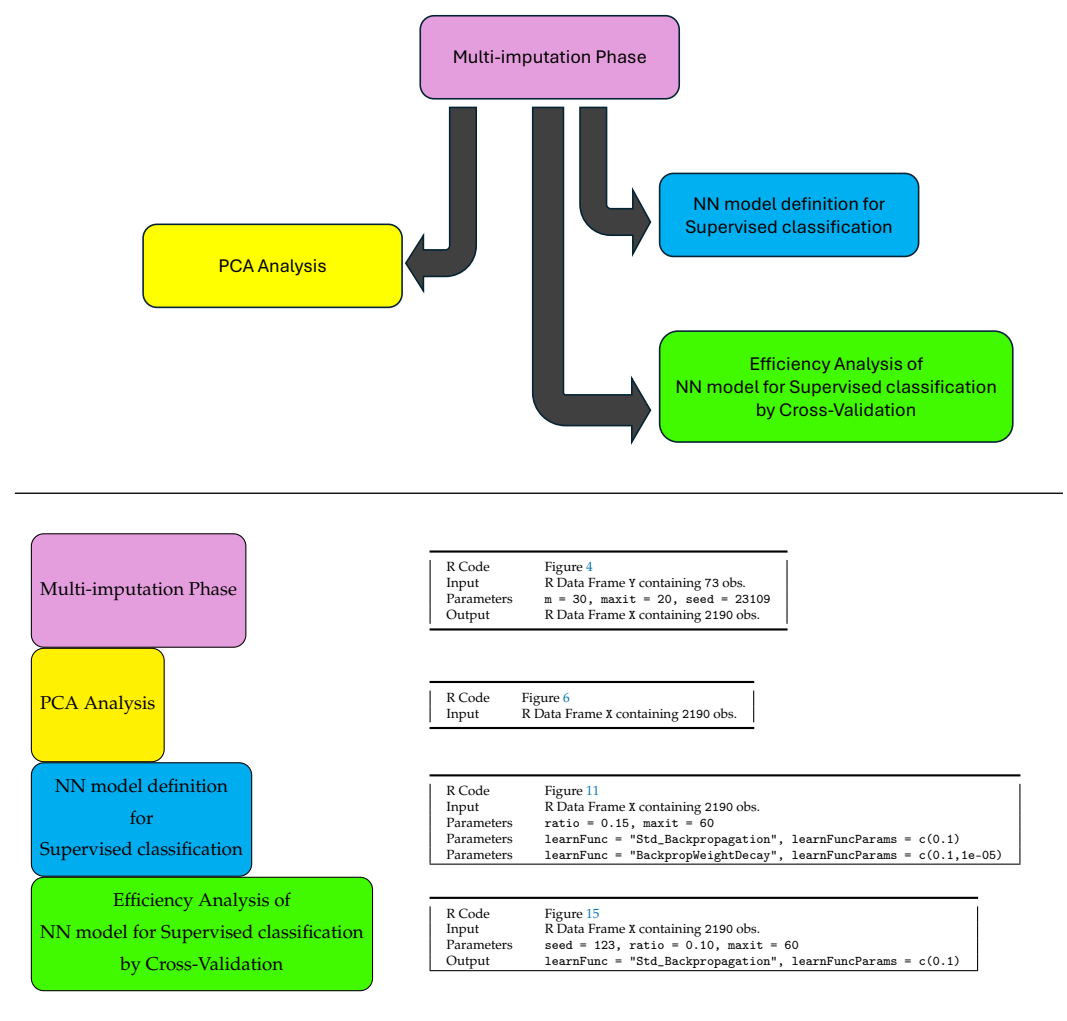


Figure 3. Flow-chart of the different phases, involving mathematical tools introduced in Section 2, of qualitative and quantitative analyses of data.

```

1  #
2  # Structure of censored data
3  #
4  str(Y)
5  tibble [73 × 9] (S3: tbl_df/tbl/data.frame)
6    $ IL18Skin      : num [1:73] NA NA NA 23.2 NA ...
7    $ IL18Serum     : num [1:73] 1432 NA NA NA 1297 ...
8    $ NLRP3bActinSkin: num [1:73] 7.5 NA NA 18.7 NA NA 13.7 NA NA 4.9 ...
9    $ TGFBSSkin     : num [1:73] 3035 1863 2221 NA 6890 ...
10   $ TGFBSSerum    : num [1:73] 78.9 NA NA NA 309.2 ...
11   $ IL10Skin      : num [1:73] NA 4.03 NA 4.35 NA ...
12   $ IL6Serum      : num [1:73] 4.7 NA NA NA 5.01 ...
13   $ SodSerum      : num [1:73] NA NA NA NA 75.6 ...
14   $ Status        : num [1:73] 1 1 1 1 1 1 1 1 1 1 ...
15  #
16  # Impute censored data Y by the mice procedure
17  #
18  m <- 30
19  imp <- mice(Y, m=m, maxit=20, seed = 23109, printFlag=FALSE)
20  print(imp)
21  Class: mids
22  Number of multiple imputations: 30
23  Imputation methods:
24      IL18Skin      IL18Serum NLRP3bActinSkin      TGFBSSkin      TGFBSSerum
25      "pmm"         "pmm"         "pmm"         "pmm"         "pmm"
26      IL10Skin      IL6Serum      SodSerum      Status
27      "pmm"         "pmm"         "pmm"         ""
28  PredictorMatrix:
29      IL18Skin IL18Serum NLRP3bActinSkin TGFBSSkin TGFBSSerum IL10Skin
30  IL18Skin      0      1      1      1      1      1
31  IL18Serum      1      0      1      1      1      1
32  NLRP3bActinSkin 1      1      0      1      1      1
33  TGFBSSkin      1      1      1      0      1      1
34  TGFBSSerum      1      1      1      1      0      1
35  IL10Skin      1      1      1      1      1      0
36      IL6Serum SodSerum Status
37  IL18Skin      1      1      1
38  IL18Serum      1      1      1
39  NLRP3bActinSkin 1      1      1
40  TGFBSSkin      1      1      1
41  TGFBSSerum      1      1      1
42  IL10Skin      1      1      1
43  #
44  # Plot density
45  #
46  densityplot(imp, ~IL18Skin + IL18Serum + NLRP3bActinSkin +
47    TGFBSSkin + TGFBSSerum + IL10Skin + IL6Serum + SodSerum,
48    layout = c(2, 4), allow.multiple=TRUE)
49  #
50  # Collect imputed data into the X data frame
51  #
52  X <- complete(imp,action="long")
53  str(X)
54  'data.frame':      2190 obs. of  9 variables:
55    $ IL18Skin      : num  96.2 85.5 222.5 23.2 23.3 ...
56    $ IL18Serum     : num  1432 404 191 1198 1297 ...
57    $ NLRP3bActinSkin: num   7.5 7.2 1.4 18.7 15.2 ...
58    $ TGFBSSkin     : num  3035 1863 2221 1621 6890 ...
59    $ TGFBSSerum    : num   78.9 64.5 64.5 78.9 309.2 ...
60    $ IL10Skin      : num   4.65 4.03 3.62 4.35 4.35 ...
61    $ IL6Serum      : num   4.7 4.7 9 4.7 5.01 ...
62    $ SodSerum      : num  76.7 87 64.6 87 75.6 ...
63    $ Status        : num   1 1 1 1 1 1 1 1 1 ...

```

Figure 4. R code of the imputation phase.

3.1. Biochemical Data Description

Results from biochemical assays are summarized in Table 1. Table 1 shows a higher level of IL-18 (ELISA assay) in the skin and serum of ALS patients compared to healthy controls. By contrast, higher levels of IL-1 β (ELISA assay) were observed only in the serum

of ALS patients compared to controls. A more evident increase in NLRP-3 expression (Western blot analysis) in the skin biopsies of ALS patients compared to controls was detected, thus suggesting inflammasome cascade activation. Results from the ELISA kit assay demonstrated an increase in TGF- β levels in the serum of patients affected by ALS. In the skin biopsies, these levels were even higher than those detected in the serum, thus highlighting the importance of the skin as a strategic tool to predict ALS progression. At the same time, no particular differences were observed in IL-10 (an anti-inflammatory cytokine) levels (ELISA assay) in the serum and skin of ALS patients. We observed significantly increased levels of IL-6 (ELISA assay) only in the serum of patients affected by ALS. Furthermore, no significant differences were observed in NEK-7 expression (Western Blot analysis) in the skin biopsies of ALS patients compared to controls. By contrast, a two-fold increase in p-TDP-43 expression (Western blot analysis) in the ALS skin biopsies compared to controls was observed, thus confirming the hypothesis of a correlation between NLRP-3 activation and nerve degeneration (p-TDP-43). Finally, we found that SOD activity was highly inhibited in the serum of ALS patients compared to healthy controls.

Table 1. Results from biochemical assays performed on skin biopsies and serum from healthy and ALS patients. Results are expressed as mean $\pm$ standard deviation.

Patients	IL-18 Skin (pg/mL)	IL-18 Serum (pg/mL)	IL-1 β Skin (pg/mL)	IL-1 β Serum (pg/mL)	NLRP-3/ β -Actin Ratio (Skin)	TGF- β Skin (pg/mL)	TGF- β Serum (pg/mL)
Healthy	12.2 $\pm$ 5.2	340.7 $\pm$ 130.9	24.9 $\pm$ 2.1	25.6 $\pm$ 1.3	1.4 $\pm$ 0.5	1224.8 $\pm$ 490.3	114.9 $\pm$ 62.5
ALS	131.8 $\pm$ 62.3	701.7 $\pm$ 380.2	27.5 $\pm$ 9.3	31.9 $\pm$ 8.5	15.9 $\pm$ 10.1	2469.4 $\pm$ 1432.8	138.2 $\pm$ 100.6
Patients	IL-10 Skin (pg/mL)	IL-10 Serum (pg/mL)	IL-6 Skin (pg/mL)	IL-6 Serum (pg/mL)	NEK7/ β -Actin Ratio	p-TDP43/GAPDH Ratio	Sod Serum (Inhibition Percentage)
Healthy	6.4 $\pm$ 2.0	23.8 $\pm$ 2.1	22.0 $\pm$ 1.3	5.1 $\pm$ 1.4	0.9 $\pm$ 0.1	1.2 $\pm$ 0.3	49.2 $\pm$ 28.7
ALS	7.2 $\pm$ 2.5	24.2 $\pm$ 5.5	27.5 $\pm$ 7.5	42.2 $\pm$ 33.1	1.0 $\pm$ 0.1	3.0 $\pm$ 0.8	77.5 $\pm$ 14.2

3.2. Mathematical Data and Computational Environment Description

A subset of the collected data are considered, consisting of data from fifty-six sick patients and seventeen healthy ones for a total of seventy-three observations. For each of the patient observations, the set of 14 biochemical markers described in Section 3.1 is represented as mathematical variables to which the Status variable is added to indicate the sick (Status = 1) or healthy (Status = 0) condition. In Table 2, such variables are listed together with

1. The name (column 1) and related marker description (column 5);
2. The numbers of available data for each marker (column 2) and its percentage on the total of observation (column 3);
3. The percentage of missing data for each marker (column 4).

All the tests were performed by using packages (whose full list is shown in Table 3) available for the R: a well-known language and environment for statistical computing. As suggested by its authors, R is an integrated suite of software facilities for data manipulation, calculation, and graphical display, where the term “environment” is intended to characterize it as a fully planned and coherent system designed around a true computer language, and allowing users to add additional functionality by defining new procedures [56]. For

computationally intensive tasks, C, C++, and Fortran code can be linked and called at run time. Furthermore, R packages exist that can implement strategies for introducing “parallelism” in R algorithms [56].

Table 2. List of variables associated with biochemical markers.

Variable Name	Number Available Data	Percentage Available Data	Percentage Missing Data	Markers Name
IL18Skin	28	38	62	IL-18 Skin (pg/mL)
IL18Serum	52	71	29	IL-18 Serum (pg/mL)
IL1BSkin	8	10	90	IL-1 β Skin (pg/mL)
IL1BSerum	8	10	90	IL-1 β Serum (pg/mL)
NLRP3bActinSkin	26	35	65	NLRP-3/ β -actin ratio (Skin)
TGFBSkin	22	30	70	TGF- β Skin (pg/mL)
TGFBSerum	23	31	69	TGF- β Serum (pg/mL)
IL10Skin	11	15	85	IL-10 Skin (pg/mL)
IL10Serum	21	28	72	IL-10 Serum (pg/mL)
IL6Skin	26	35	65	IL-6 Skin (pg/mL)
IL6Serum	31	42	58	IL-6 Serum (pg/mL)
NEK7bActin	11	15	85	NEK7/ β -actin ratio
pTDP43GAPDH	13	17	83	p-TDP43/GAPDH ratio
SodSerum	26	35	65	Sod serum (% of inhibition)
Tot	306	29	71	

Table 3. Details of used software. All URLs were accessed on 26 January 2026.

Name	Version	Reference Manual
R Environment	4.1.3	https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf
MICE Package	3.18.0	https://cran.r-project.org/web/packages/mice/mice.pdf
stats Package	4.1.3	https://cran.r-project.org/doc/manuals/r-release/fullrefman.pdf
RSNNS Package	0.4-17	https://cran.r-project.org/web/packages/RSNNS/RSNNS.pdf
NeuralNetTools Package	1.5.3	https://cran.r-project.org/web/packages/NeuralNetTools/NeuralNetTools.pdf
ggplot2 Package	3.5.2	https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf
DescTools Package	0.99.50	https://cran.r-project.org/web/packages/DescTools/DescTools.pdf

All the presented results are obtained by using computing resources whose details are listed in Table 4.

Table 4. Hardware and software specs of computing resources used for tests.

Processor type	Intel Xeon Gold 6240R CPU@2.40 GHz
Number of cores	48
OS	Linux CentOS 7

3.3. Imputation Phase

In this subsection, some details are given about the imputation phase of the data described in Section 3.2. The imputation process was performed by procedures from the mice R package.

Missingness in the dataset may follow a Missing At Random (MAR) mechanism, meaning that the probability that a value is missing is related to observed patient characteristics, rather than to the unobserved molecular measurements themselves. For example, patients with more advanced ALS or with greater functional impairment might be less able, or less willing, to undergo invasive procedures such as skin biopsies. In such cases, the missing molecular data are systematically associated with disease severity or functional status, which are variables available in the dataset, consistent with an MAR pattern.

The following 6 markers have been eliminated due to collinearity with the remaining ones, such as suggested by the procedure mice from MICE R package [25]: IL6Skin, IL10Serum, pTDP43GAPDH, NEK7bActin, IL1BSkin, IL1BSerum.

We observe that the mice R procedure, by default, checks for collinearity when fitting the imputation model. If a variable is almost “collinear” with another one (the correlation level between variables is above a certain threshold), it is removed and not imputed [57]. The `find.collinear` internal MICE procedure is used to establish collinearity between variables and is based on correlation matrix computation [58].

The remaining 8 markers, together with the Status variable, are organized in an R data frame Y whose structure is described on lines 5–14 in Figure 4.

The censored data Y is then imputed by the mice procedure (see line 19 in Figure 4) whose output (the `imp` object) contains, among the others, information about the following:

- The method used for the imputation of each univariate variable (see lines 23–27), which is the *Predictive Mean Matching (PMM)* (`pmm`) for all variables that must be imputed and is the empty method “” for the Status variable that does not require imputation since it does not contain censored data. Details about implementation of PMM in mice package can be found in [57].
- The *Predictor Matrix* (see line 28), which is a square matrix of size `ncol(Y)` containing 0/1 data. Each row in *Predictor Matrix* identifies which predictors are to be used for the variable in the row name. In the case of our approach, the Status variable is used as a predictor for all the censored variables, although it should not be imputed because such a variable can be considered as a relevant variable. We note that general advice from MICE experts is to include as many relevant variables including their interactions [25].

The number of imputations is chosen to be $m = 30$ according to maximum efficiency $E_{MI}(m, \lambda)$ obtainable when the percentage λ of censored data are greater than 70%. (see Table A1 in Appendix A.1). Instead, the value of the maximum number of iterations of the iterative process of the MICE algorithm (see line 6 of Algorithm A1 in Appendix A.1) is chosen to be the default value of mice procedure (that is `maxit = 20`) since such value is often considered sufficient [25]. The random seed used in imputation step by mice procedure is `seed = 23109` (see line 19 in Figure 4).

Once the MICE algorithm terminates, “it is important to inspect the imputations. In general, a good imputed value is a value that could have been observed had it not been missing [25]”. So, to check whether the imputations created by the MICE algorithm are plausible, density plots of both the observed and imputed values of all variables can be drawn. In Figure 5, the density trends of imputed (red lines) and observed data (blue line) for each of the 8 considered markers are plotted. Such plots, obtained by the R procedure `densityplot` (see lines 46–48 in Figure 4) using the density procedure from the R package `stats`, are based on the kernel density estimate (KDE) for the probability density of univariate variable [59]. Plots show that there are no discernible, meaningful disparities between the observed data and the imputed values.

At the end of the data imputation process, a data frame X comprising $73 \times 30 = 2190$ samples is obtained, whose structure is described on lines 54–63 in Figure 4.

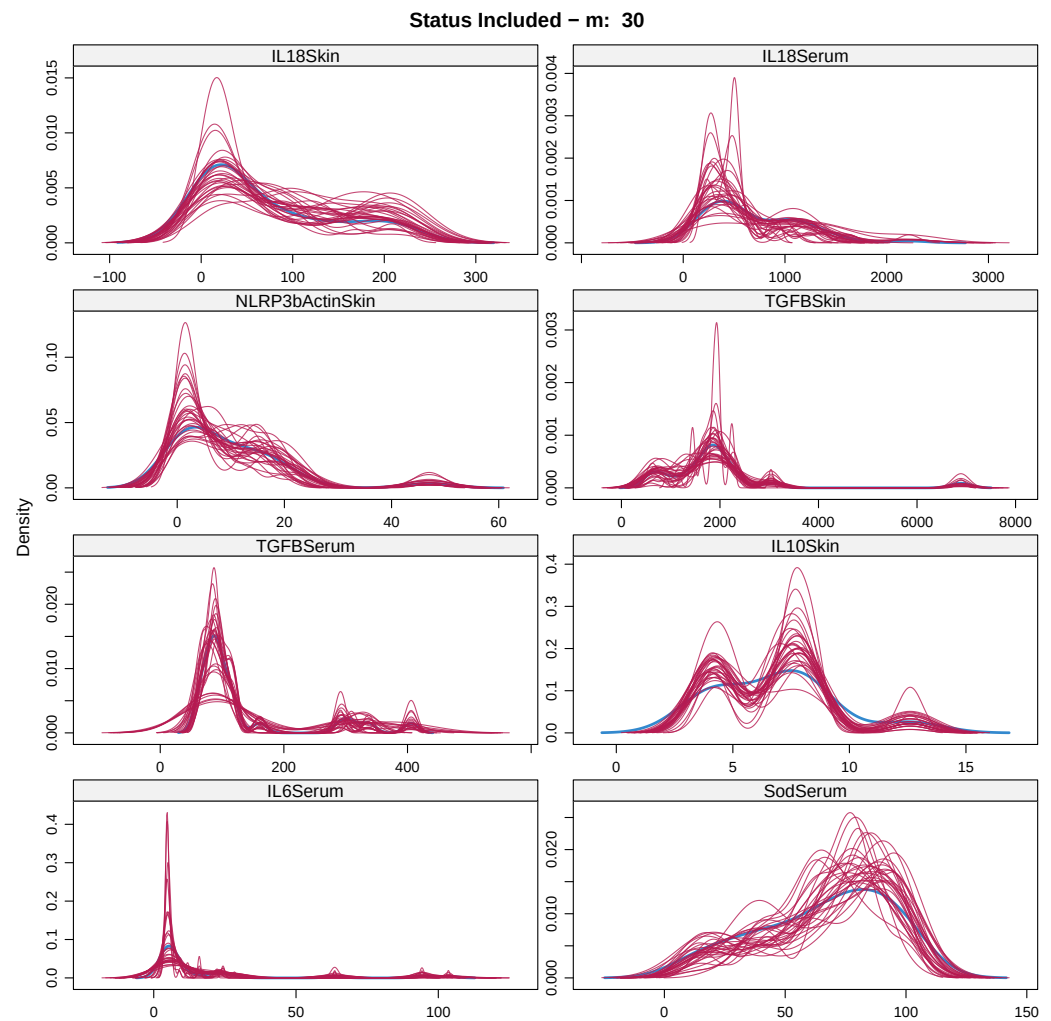


Figure 5. Kernel density estimates for the marginal distribution of the $m = 30$ imputed data (red lines) and available data (blue line) for each of the 8 considered markers.

3.4. Interpretation Phase

To qualitatively study the relevance of the molecular markers introduced in Section 3.2 in the identification of ALS disease state, we can study the “relations” between the variables associated with such markers and the variable Status representing the disease state by using the dataset of complete observations obtained by the MI described in Section 3.3 on which a normalized PCA is performed.

PCs are computed by using the procedure princomp from the R stats package. Such procedure is based on a computation of eigenvalues of the covariance matrix of an input data frame passed to the princomp procedure.

Two types of tests were performed on such data:

Test set n. 1 Normalized PCA of the full data frame X obtained at the end of the MI phase described in Section 3.3 to identify the correlations between the marker variables and the disease state one. See lines 2–20 of the code listed in Figure 6. This analysis aims to identify which marker variables are more correlated with the disease state.

Test set n. 2 Normalized PCA of the data frame XMarkers obtained from X preserving just the variables associated with the markers. See lines 25–38 of the code listed in Figure 6. This analysis aims to identify which groups of marker variables are more effective in distinguishing the disease state.

```

1  #
2  # X is the results of MI phase (see code in Figure 4)
3  #
4  #
5  #
6  # Test set n. 1
7  #
8  Z <- scale(X)           # Normalize data in X data ferme
9  data.pca <- princomp(Z) # Compute PC of Z data
10 summary(dataTotali.pca) # Show summary from PC of Z
11
12 Importance of components:
13      Comp.1      Comp.2      Comp.3      Comp.4      Comp.5
14 Standard deviation  1.617482  1.1451416  1.0575599  0.9996773  0.9511375
15 Proportion of Variance 0.290827  0.1457720  0.1243271  0.1110901  0.1005640
16 Cumulative Proportion 0.290827  0.4365991  0.5609262  0.6720163  0.7725803
17      Comp.6      Comp.7      Comp.8      Comp.9
18 Standard deviation  0.90934002  0.76880222  0.70103060  0.36938132
19 Proportion of Variance 0.09191967  0.06570298  0.05462982  0.01516721
20 Cumulative Proportion 0.86449998  0.93020297  0.98483279  1.00000000
21
22 #
23 # Test set n. 2
24 #
25 XMarkers <- X[,1:8]      # Preserve just variables associated with markers
26 ZMarkers <- scale(XMarkers) # Normalize data in XMarkers data ferme
27 Markers_data.pca <- princomp(ZMarkers) # Compute PC of ZMarkers data
28 summary(Markers_data.pca) # Show summary from PC of ZMarkers
29
30 Importance of components:
31      Comp.1      Comp.2      Comp.3      Comp.4      Comp.5
32 Standard deviation  1.3984442  1.1449447  1.0521130  0.9950700  0.9161805
33 Proportion of Variance 0.2445674  0.1639371  0.1384309  0.1238271  0.1049713
34 Cumulative Proportion 0.2445674  0.4085046  0.5469355  0.6707626  0.7757339
35      Comp.6      Comp.7      Comp.8
36 Standard deviation  0.8964635  0.71370163  0.69303184
37 Proportion of Variance 0.1005017  0.06370034  0.06006407
38 Cumulative Proportion 0.8762356  0.93993593  1.00000000

```

Figure 6. R code of the PCA phase.

Below is a series of descriptions of tests results:

- The “Proportion of Variance” values listed in Figure 6 for each component $j = 1, \dots, p$ (where $p = 9$ for Test set n. 1 and $p = 8$ for Test set n. 2) represent the *Relevance*(j) defined by Equation (6). Figure 7 shows the trend of *Relevance*(j) as a function of j for Test set n. 1 (a) and Test set n. 2 (b). Furthermore, the “Cumulative Proportion” values represent the *CumulativeRelevance*(j) defined by Equation (7).
- Regarding Test set n. 1, in Figure 8, PC-based BiPlots of the vectors $(z_j)^{\angle_{j_1, j_2}}$, representing the variable z_j in an underlying Cartesian system $\mathcal{C}_{TestSet1}^{(j_1, j_2)}$ (see Section 2.3.2), are shown. The following list of (j_1, j_2) for PC-based BiPlots is considered: $j_1 = 1$, $j_2 = 2, \dots, 9$. In each BiPlot, the color of the vectors $(z_j)^{\angle_{j_1, j_2}}$ is tied to their “representation accuracy” of the variable z_j in a related Cartesian system $\mathcal{C}_{TestSet1}^{(j_1, j_2)}$. In Figure 9, PC-based BiPlots of the vectors $(z_j)^{\angle_{j_1, j_2}}$ and the individuals z_i such as projected on the Cartesian system $\mathcal{C}_{TestSet1}^{(j_1, j_2)}$ (see Section 2.3.2) are shown. The following list of (j_1, j_2) for PC-based BiPlots is considered: $j_1 = 1$, $j_2 = 2, \dots, 9$. In each BiPlot, the gray color intensity of the vectors $(z_j)^{\angle_{j_1, j_2}}$ is tied to their “representation accuracy”.
- Regarding Test set n. 2, in Figure 10, PC-based BiPlots of the vectors $(z_j)^{\angle_{j_1, j_2}}$ and the individuals z_i such as projected on the Cartesian system $\mathcal{C}_{TestSet2}^{(j_1, j_2)}$ (see Section 2.3.2) are shown. The following list of (j_1, j_2) for PC-based BiPlots is considered: $j_1 = 1$, $j_2 = 2, \dots, 8$. In each BiPlot, the gray color intensity of the vectors $(z_j)^{\angle_{j_1, j_2}}$ is tied to their “representation accuracy”.
- In Figures 9 and 10, ellipses are plotted “around” individuals that represent a *concentration ellipse* [35] in normal probability where the percentage of the included individuals is 95%. Such plots are based on the `stat_ellipse` procedure from the R `ggplot2` package [60].

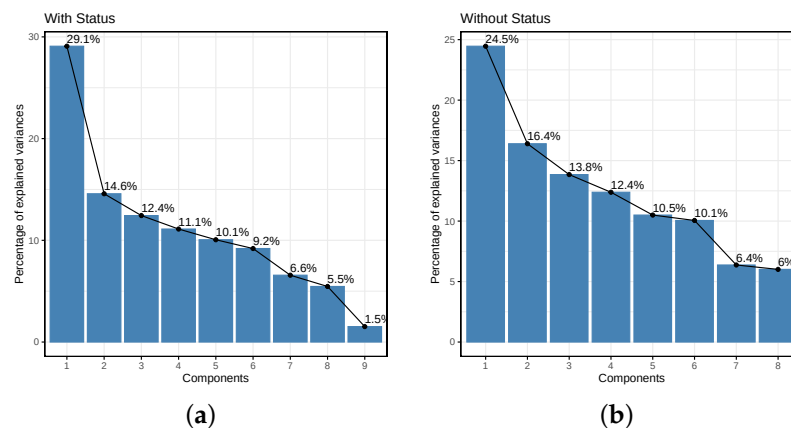


Figure 7. Proportion of variance explained by each principal component with the status parameter included (a) and excluded (b).

Below is a series of considerations based on the tests carried out:

- From the lines 16 and 34, we observe what the first five components of the PCs from both the test sets express about the 77% of the “information” contained in each dataset correlation matrix. Each of the components from the second one to the 5-th express 10%.
- The choice related to the couple (j_1, j_2) for a PC-based BiPlots is based on a combination of j_1, j_2 that expresses the same amount of “information” when $j_2 = 2, 3, 4, 5$: about 35–40%.
- From Figure 8, related to Test set n. 1, the following can be observed:
 1. High values for the “representation accuracy” $\cos^2$ of the variables Status and SodSerum in all the considered BiPlot spaces, confirming that any of the considered spaces can be potentially useful to get information about the relations among such variables.
 2. That in almost all the considered BiPlot spaces (i.e., excluding the “less significant” one), the variables Status and SodSerum are highly correlated.
 3. High values of “representation accuracy” are also present for both the variables IL18Skin and IL18Serum which express a good level of correlation with the Status variable (for example, see Figure 8a,b,d–f).
 4. Just some of the considered BiPlot spaces (see Figure 8b,c,g) show a good correlation level for the variable NLRP3bActinSkin (which has a medium level “representation accuracy”) with the Status one. Meanwhile, some other BiPlot spaces (see Figure 8c,d,f) show a good correlation level for the variable IL6Serum with Status, although the “representation accuracy” of IL6Serum is not so high.
 5. That the variables TGFBSerum and IL10Skin appear almost uncorrelated with Status in almost all the considered BiPlot spaces where their “representation accuracy” can be considered appreciable (see Figure 8a–c). The variable TGFBSkin shows a behavior, with respect to the Status variable, that makes it difficult to interpret: high correlation with not so high “representation accuracy” (i.e., see Figure 8a) and low correlation where its “representation accuracy” has a bigger value (i.e., see Figure 8b).
- From Figure 9, related to Test set n. 1, as counter-evidence of the effectiveness of representing individuals in a BiPlot space, it can be observed that the presence of the Statut variable separates sick individuals from healthy ones well.
- The same considerations on the “individual separation” are not possible when the Status variable is not used (Test set n. 2). Indeed, from Figure 10, we observe that

“concentration ellipses” for sick and healthy individuals can have a big overlap. We can also observe that the orientation and center of “concentration ellipses” for sick individuals seems to be, in the majority of BiPlot spaces which also coincide with the spaces that “carry-out the greatest amount of information”, particularly conditioned by the variables SodSerum, IL18Serum, IL18Skin, NLRP3bActinSkin and TGFBSkin (see Figure 10a–d).

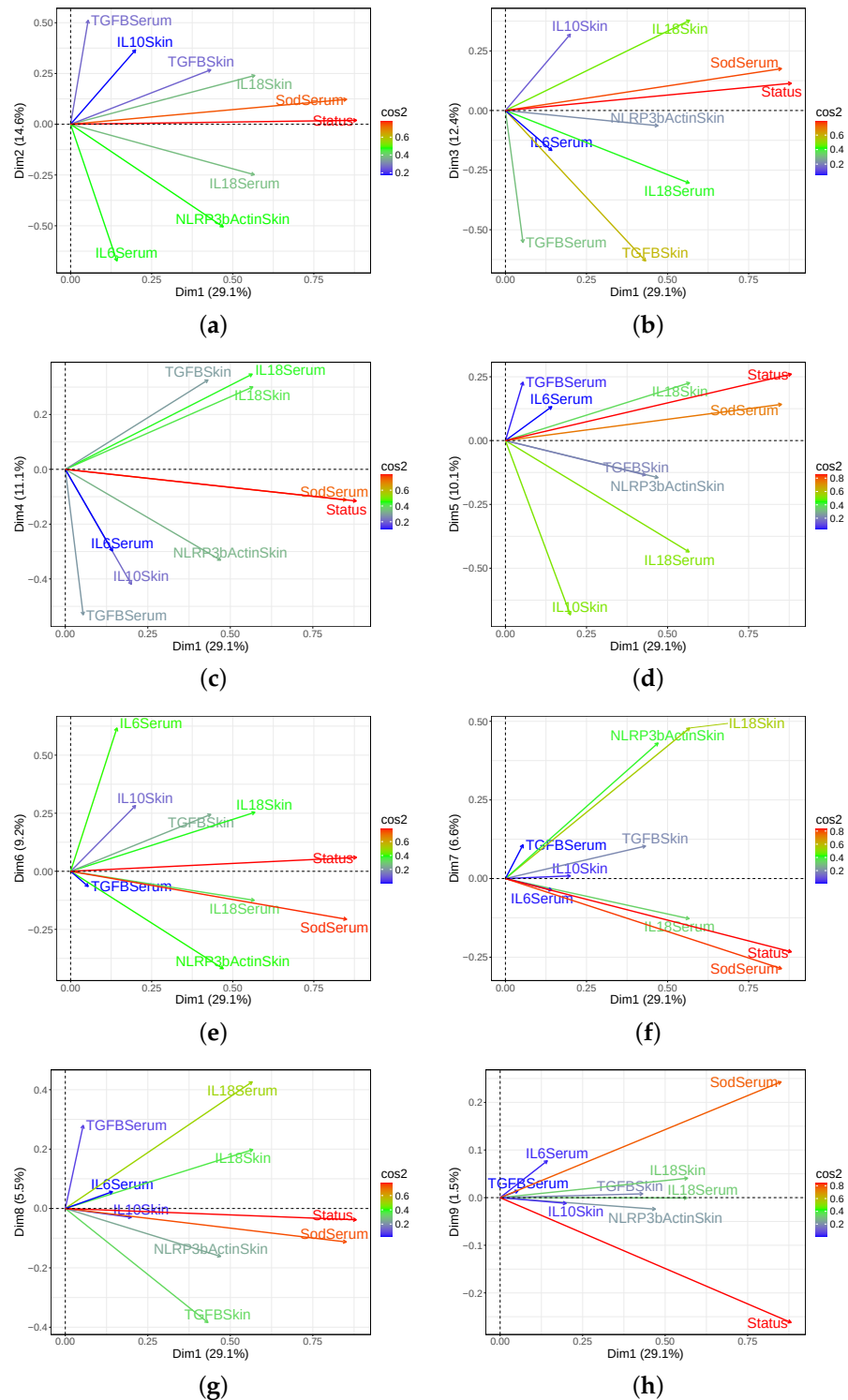
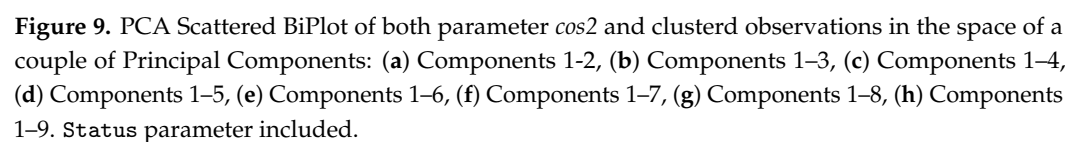


Figure 8. PCA $\cos^2$ in the space of a couple of Principal Components: (a) Components 1–2, (b) Components 1–3, (c) Components 1–4, (d) Components 1–5, (e) Components 1–6, (f) Components 1–7, (g) Components 1–8, (h) Components 1–9. Status parameter included.



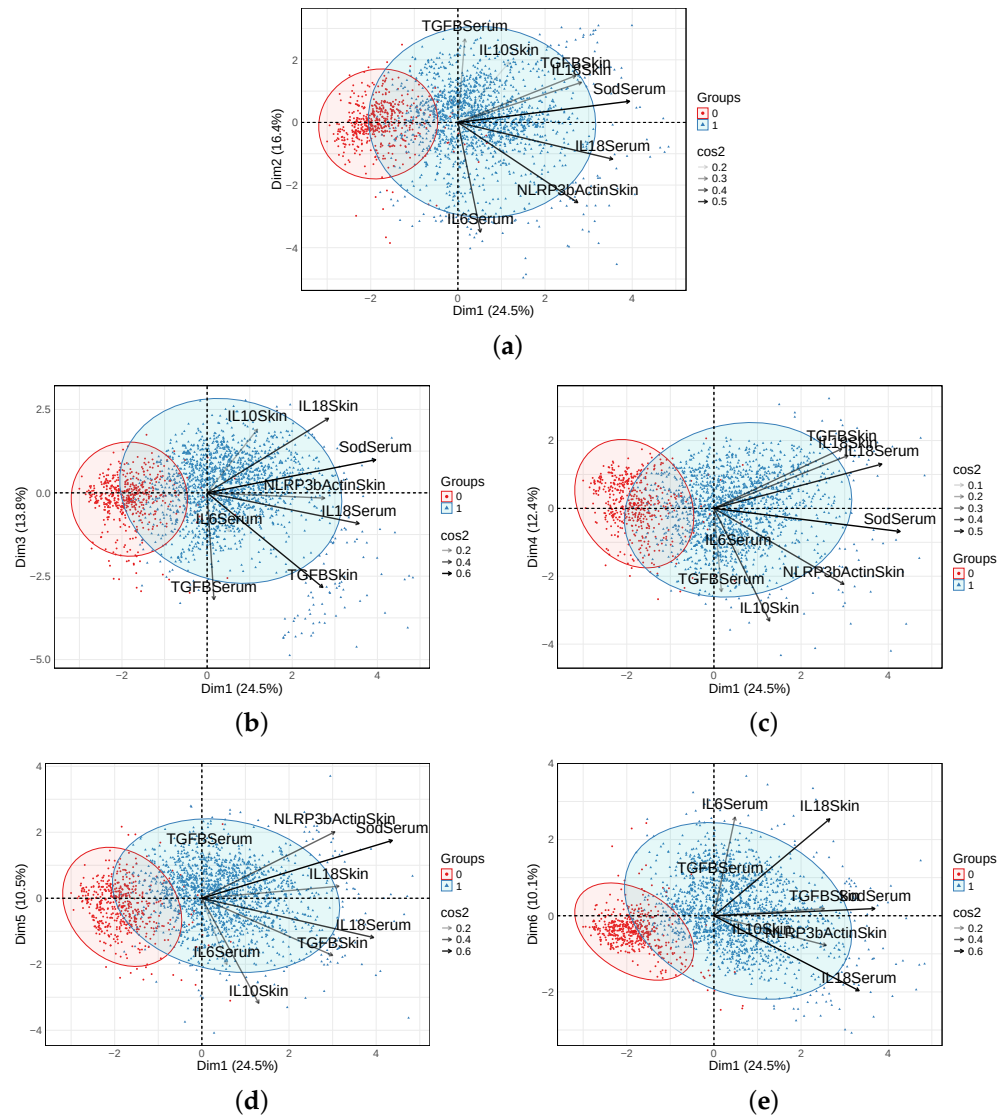


Figure 10. PCA scattered BiPlot of both parameter $\cos 2$ and clustered observations in the space of a couple of principal components: (a) components 1–2, (b) components 1–3, (c) components 1–4, (d) components 1–5, and (e) components 1–6. Status parameter excluded. Just the first six and more significant PCs are considered.

3.5. Provisional Method for Supervised Classification Based on Neural Network

To quantitatively study the relevance of the molecular markers introduced in Section 3.2 in the identification of the ALS disease state, we can analyze the results of NN-type predictive models for supervised classification. See Figure 11 for the R code used to perform the analysis.

Such approach is based on the following steps:

Define and “Learn” an NN-type model to classify sick and healthy individuals on the basis of markers from some known individual states.

The first action is to define the model $g : \mathcal{X} \rightarrow \mathcal{Y}$, clarifying what is the domain $\mathcal{X}$ and co-domain $\mathcal{Y}$ of g . In particular, we call $\mathcal{X}$ the space of parameters and define it as

$$\mathcal{X} = \{(x_1, \dots, x_{N_{Par}}), x_i \in \mathbb{R}\},$$

where $N_{par} = 8$ and where each x_i represents a marker according to the association defined in Table 5. The space

$$\mathcal{Y} = \{(y_1, y_2), y_i \in \{0, 1\}\}$$

is based on the couples

$$(y_1, y_2) = \begin{cases} (0, 1) & \text{if individual is classified as sick,} \\ (1, 0) & \text{if individual is classified as healthy.} \end{cases} \quad (17)$$

Table 5. Molecular markers' list and parameter associations.

Marker Name	Parameter ID
IL18Skin	1
IL18Serum	2
NLRP3bActinSkin	3
TGFBSkin	4
TGFBSerum	5
IL10Skin	6
IL6Serum	7
SodSerum	8

```

1  #
2  # X is the results of MI phase (see code in Figure 4)
3  #
4  #
5  #
6  # Considered value for N: 100 300 500 700 900 1100
7  #
8
9  x <- X[, 1:8]
10 y <- X[, 9:9]
11
12 yDec <- decodeClassLabels(y)
13 data <- splitForTrainingAndTest(x, yDec, ratio = 0.15)
14 data <- normTrainingAndTestSet(data)
15
16 mod <- mlp(data$inputsTrain, data$targetsTrain, size = c(N),
17             learnFunc = "Std_Backpropagation", learnFuncParams = c(0.1), maxit = 60,
18             inputsTest = data$inputsTest, targetsTest = data$targetsTest)
19
20 plotIterativeError(mod)
21
22 modWD <- mlp(data$inputsTrain, data$targetsTrain, size = c(N),
23              learnFunc = "BackpropWeightDecay", learnFuncParams = c(0.1, 1e-05), maxit = 60,
24              inputsTest = data$inputsTest, targetsTest = data$targetsTest)
25 plotIterativeError(modWD)
26
27
28 predictionsmod <- predict(mod, data$inputsTest)
29 plotRegressionError(predictionsmod[, 2], data$targetsTest[, 2])
30
31
32 olden(mod, out_var = 'Output_1')
33

```

Figure 11. R code of the NN-based analysis.

The second action is to improve the definition of g as an NN-based model; thus, we assume the following form for g :

$$g = h\left(f_{sigmoid}(x; w, N)\right),$$

where $f_{sigmoid}(x; w, N) \in \mathcal{F}^{L=3, \sigma=\sigma_{sigmoid}}(N_{par}, 2)$ and where $\mathcal{F}^{L=3, \sigma=\sigma_{sigmoid}}(N_{par}, 2)$ is the subset of $\mathcal{F}(N_{par}, 2)$ defined as

$$\mathcal{F}^{L=3, \sigma=\sigma_{sigmoid}}(N_{par}, 2) = \left\{ f_{\mathcal{N}_{\theta}(N_{par}, 2)}, \theta \in \Theta : L = 3, \sigma = \sigma_{sigmoid} \right\}.$$

We consider that $\sigma_{sigmoid}$ is a function defined as

$$\sigma_{sigmoid}(x) = \frac{1}{1 + e^{-x}}.$$

The sigmoid function is a nonlinear activation function widely used in neural networks and other computational models. Its output range makes it particularly suitable for modeling probabilities, as it can map any real value to the $[0, 1]$ interval, a property leveraged in binary classification problems [61].

Then, $f_{sigmoid}(\mathbf{x}; \mathbf{w}, N)$ is a function “defined” by a shallow network whose “activation function” is the so-called “Sigmoid function”, whose “complexity” N is equal to the number of neurons of the single “hidden layer”, and whose set of weights is

$$\mathbf{w} = \left\{ w_{i,j}^{(2)} \right\}_{i=1,\dots,N_{Par},j=1,\dots,N} \cup \left\{ w_{i,j}^{(3)} \right\}_{i=1,\dots,N,j=1,\dots,2}.$$

A shallow network is preferable since, at the status, no information about pure locally interaction among neurons is available (see considerations about the “Curse of Dimensionality” in Section 2.3.3 and Theorem A2 in Appendix A.3). Moreover, since the limited value of $n = N_{Par} = 8$, the complexity N remains limited to an order of $O(10^3)$ (see Figure 12) also because the considered use case does not require a very small value for ϵ : $\epsilon = O(10^{-1})$ could be considered a good value for ϵ since we have to just discriminate between the values of function $f_{sigmoid}(\mathbf{x}; \mathbf{w}, N)$ that are “near” zero or one. Figure 12 shows the trend of shallow neural network complexity N providing the accuracy of at least ϵ when $n = N_{Par} = 8$ [42].

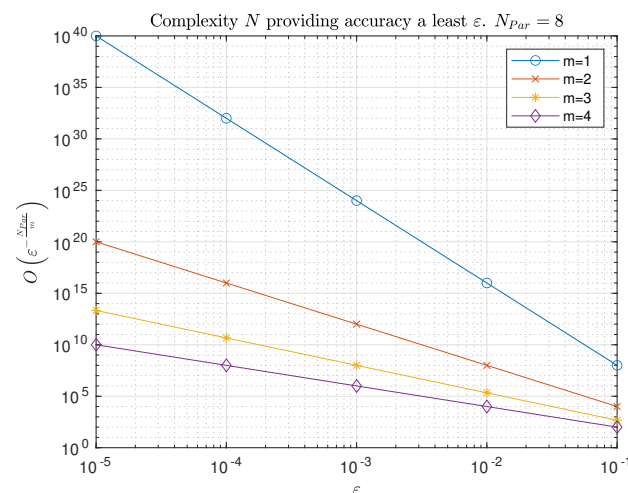


Figure 12. Trend of shallow neural network complexity N providing accuracy of at least ϵ . $N_{Par} = 8$.

Moreover, we consider that h is a function defined as a (Winner-Takes-All [WTA] strategy):

$$h : \mathbf{z} \in \mathbb{R}^2 \mapsto h(\mathbf{z}) \in \mathcal{Y} \quad \text{where} \quad h(\mathbf{z}) = \begin{cases} (0, 1) & \text{if } z_2 = \max(z_1, z_2), \\ (1, 0) & \text{if } z_1 = \max(z_1, z_2). \end{cases}$$

Verify NN-type model: To assess how the model will describe individuals (the “test individuals” $\{\mathbf{x}_i^{Test}, \mathbf{y}_i^{Test}\}_{i=1,\dots,m_{Test}}$) not used to “Learn” the model (that we identify by $\{\mathbf{x}_i^{Learn}, \mathbf{y}_i^{Learn}\}_{i=1,\dots,m_{Learn}}$). To “Learn” the parameters (the weights $\mathbf{w}$) defining the function of the form $f_{sigmoid}(\mathbf{x}; \mathbf{w}, N)$, we use the procedure `mlp` from the R `RSNNS` package. Such procedure, if data related to “test individuals” are passed to procedure (see line 18 in Figure 11), has the advantage that the “Cost function” in both the learning and test sets are

evaluated after each iteration (also called “epoch”) of the iterative process used to compute the minimum of such “Cost Function” (see Algorithm A2). Values of “Cost function” can later be used to analyze the training process, by the `plotIterativeError` procedure from the RSNNS package (see line 20 in Figure 11), for example. Training and test data can be both obtained from the input data X by the procedures `splitForTrainingAndTest` and `normTrainingAndTestSet` from the RSNNS package (see lines 13 and 14 in Figure 11). Data frame X is the output of the imputation process (see code in Figure 4) and collects data from $73 \times 30 = 2190$ individuals.

The considered “Cost Function” is

$$C\left(\left\{f_{\text{sigmoid}}(\mathbf{x}_i; \mathbf{w}, N), \mathbf{y}_i\right\}_{i=1, \dots, m}\right) = \sum_{i=1}^m \left\|f_{\text{sigmoid}}(\mathbf{x}_i; \mathbf{w}, N) - \mathbf{y}_i\right\|^2.$$

Such “Cost function” is known as the “Summed Squared Error (SSE)”, i.e., the sum of the squared errors of all patterns (the set of all the observations $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1, \dots, m}$) for every epoch.

See Figure 13 for results from the `plotIterativeError` procedure related to the neural network learning phase [53,62] for different values of neural network complexity N . This generates a plot (black line), normalized by dividing the SSE through the test set ratio (which is the amount of patterns in the test set divided by the amount of patterns in the training set) [63]. If a test set is provided, its SSE is also shown in the plot (red line).

See Figure 14 for the regression error plot obtained by the `plotRegressionError` procedure from the RSNNS package on test data (see lines 28 and 29 in Figure 11). As a classification is performed, ideally only the points (0,0) and (1,1) would be populated. The procedure `plotRegressionError` can be used to generate a regression plot which illustrates the quality of the regression. It has target values on the x -axis and fitted/predicted values on the y -axis. The optimal fit would yield a line through zero with gradient one. This optimal line is shown, as well as a linear fit to the actual data [63]. Regression error plot are showed for different values of complexity N . The `predict` procedure from the R RSNNS package can be used to obtain results on test data (see line 28 in Figure 11).

To analyze efficiency, the “confusion matrices” are used. Such matrices are obtained by the `confusionMatrix` procedure from the RSNNS package (see lines 44–47 in Figure 15). A confusion matrix, $C = (c_{i,j})_{i,j=1, \dots, n_{\text{classes}}}$, related to a classification problem of m individuals on n_{classes} , represents the amount of times the network erroneously classified an individual of the i -th class to be a member of the j -th one. See Figure 16 for “Mean values of Confusion Matrix Efficiency” $E^{\text{Mean}}(C)$ obtained as a mean value of the 10 values of Confusion Matrix Efficiency $E^k(C)$, $k = 1, \dots, 10$ computed by a 10-fold cross-validation process [54] implemented by the code listed in Figure 15, for both the training and test sets of individuals, as a function of the complexity N of the NN where $E(C)$ is defined as

$$E(C) = \frac{\sum_{i=1}^{n_{\text{classes}}} c_{i,i}}{m}.$$

$E(C)$ represents the percentage of classification successes compared to the total number of individuals.

In k -fold cross-validation, the original sample is randomly partitioned into k equal-sized subsamples, often referred to as “folds”. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k - 1$ subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data. The k results can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation,

and each observation is used for validation exactly once. Moreover, 10-fold cross-validation is commonly used [64] but, in general, k remains an unfixed parameter.

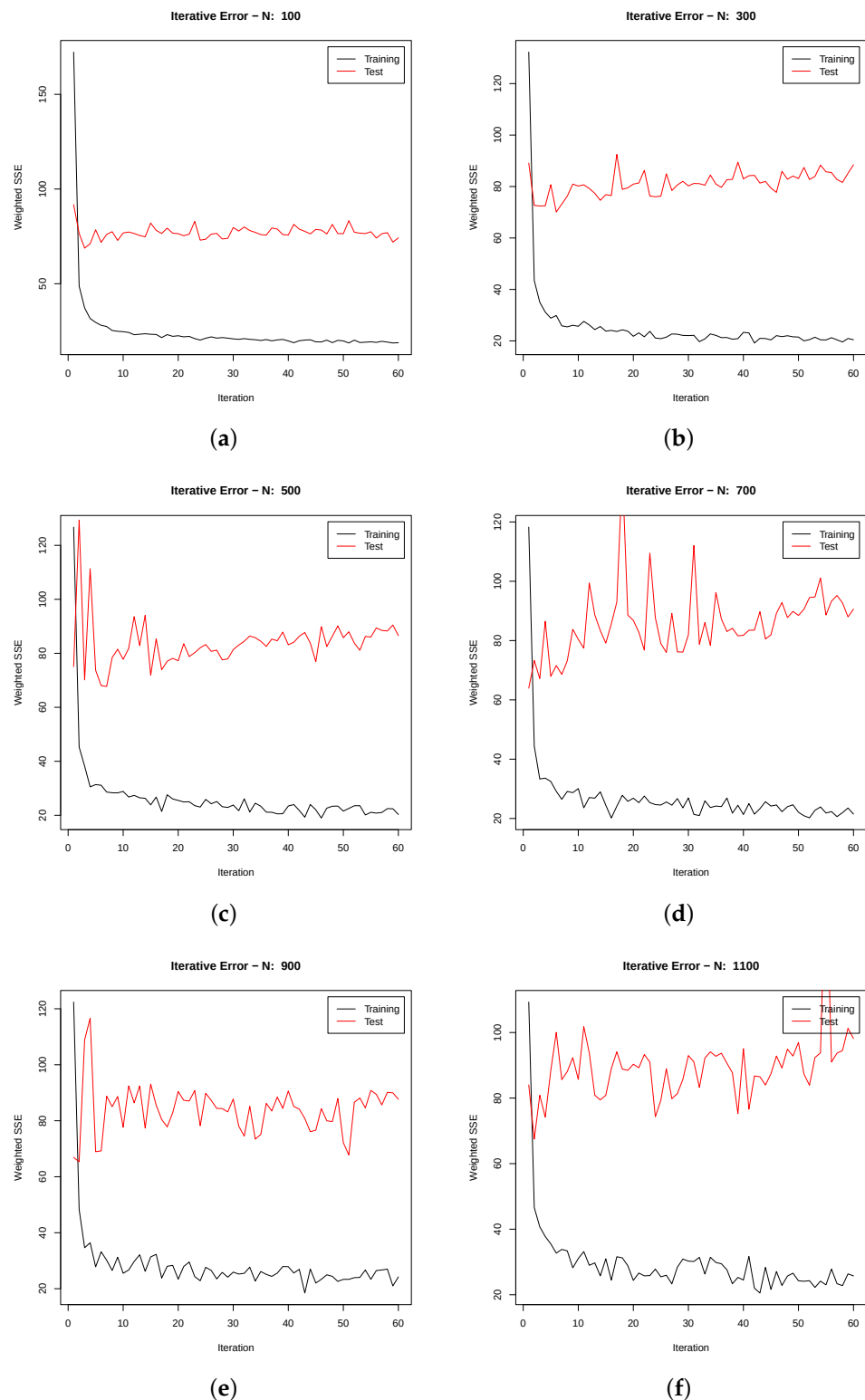


Figure 13. Convergence error plot for the test data. (a) $N = 100$, (b) $N = 300$, (c) $N = 500$, (d) $N = 700$, (e) $N = 900$, (f) $N = 1100$.

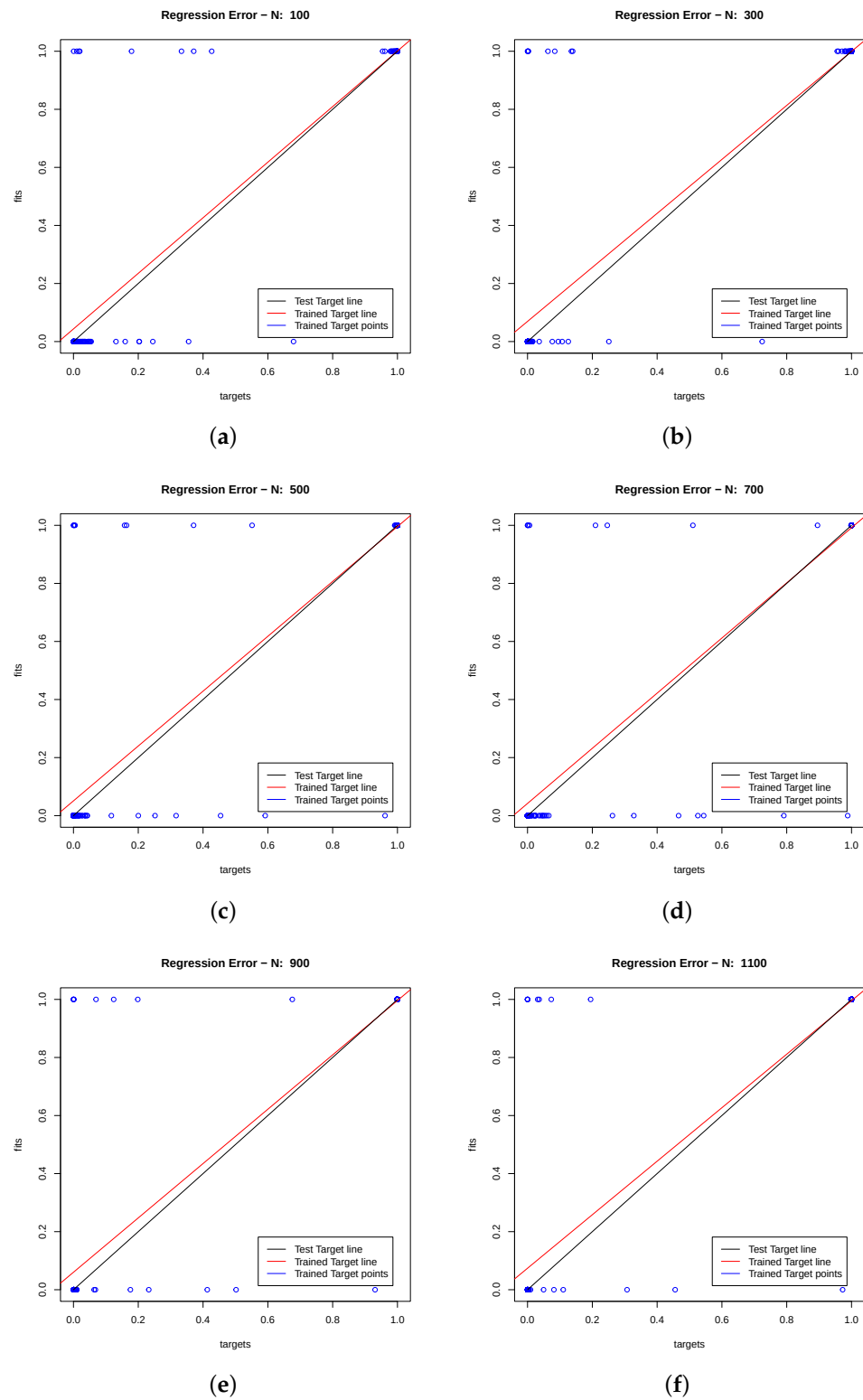


Figure 14. Regression error plot for the test data. (a) $N = 100$, (b) $N = 300$, (c) $N = 500$, (d) $N = 700$, (e) $N = 900$, (f) $N = 1100$.

```

1  #
2  # X is the results of MI phase (see code in Figure 4)
3  #
4
5  #
6  # Considered value for N: 100 300 500 700 900 1100
7  #
8
9  # Shuffle data
10 set.seed(123)
11 shuffled_data <- X[sample(1:nrow(datifinaliTotali)), ]
12
13 x <- shuffled_data[, 1:8]
14 y <- shuffled_data[, 9:9]
15
16 yDec <- decodeClassLabels(y)
17
18 ndatitotali <- nrow(X)
19 nfold <- 10
20 ndatixfold <- ndatitotali/nfold
21 nfoldmone <- nfold-1
22
23 TotAccuracyTrain <- 0
24 TotAccuracyTest <- 0
25 TotBS <- 0
26
27 for(k in 0:nfoldmone) {
28     stestrow <- k*ndatixfold+1
29     etestrow <- (k+1)*ndatixfold
30     traindatax <- x[-c(stestrow:etestrow), ]
31     traindatay <- yDec[-c(stestrow:etestrow), ]
32     testdatax <- x[c(stestrow:etestrow), ]
33     testdatay <- yDec[c(stestrow:etestrow), ]
34     dataxcv <- rbind(traindatax,testdatax)
35     dataycv <- rbind(traindatay,testdatay)
36
37     #
38     data <- splitForTrainingAndTest(dataxcv, dataycv, ratio = 0.10)
39     data <- normTrainingAndTestSet(data)
40     mod <- mlp(data$inputsTrain, data$targetsTrain, size = c(N),
41               learnFuncParams = 0.1, maxit = 60,
42               inputsTest = data$inputsTest, targetsTest = data$targetsTest)
43     predictionsmod <- predict(mod, data$inputsTest)
44
45     #
46     CFTrain <- confusionMatrix(data$targetsTrain, fitted.values(mod))
47     CFTest <- confusionMatrix(data$targetsTest, predictionsmod)
48     accuracyTrain <- sum(diag(CFTrain)) / sum(CFTrain)
49     accuracyTest <- sum(diag(CFTest)) / sum(CFTest)
50
51     #
52     classvaluesPRED <- class_matrix_to_vector(predictionsmod)
53     classvaluesTEST <- class_matrix_to_vector(data$targetsTest)
54     classvaluesTEST <- (classvaluesTEST-1)
55     classvaluesPRED <- matrix(classvaluesPRED-1)
56     BS <- BrierScore(classvaluesPRED,classvaluesTEST)
57
58     #
59     TotAccuracyTrain <- TotAccuracyTrain + accuracyTrain
60     TotAccuracyTest <- TotAccuracyTest + accuracyTest
61     TotBS <- TotBS + BS
62 }
63 MeanAccuracyTrain <- TotAccuracyTrain/nfold
64 MeanAccuracyTest <- TotAccuracyTest/nfold

```

Figure 15. R code of the cross-validation process used to evaluate mean values for Confusion Matrix Efficiency.

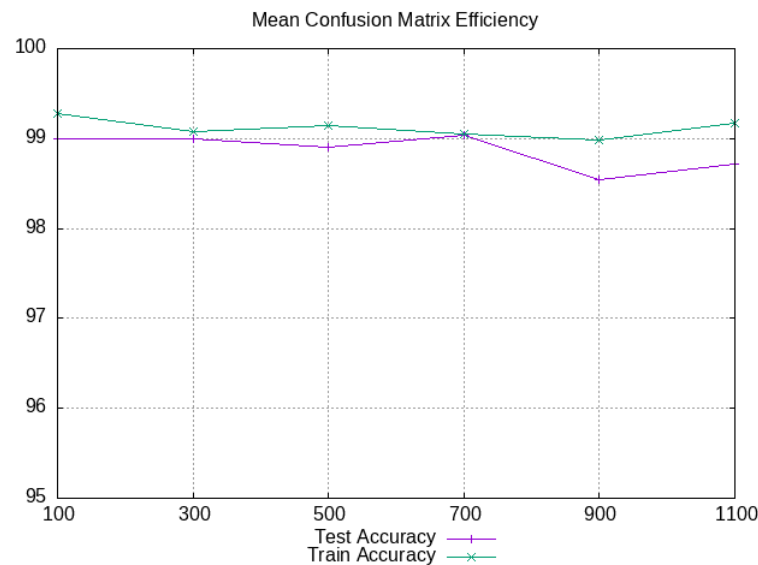


Figure 16. Mean confusion matrices' efficiency.

In Table 6, we show the Mean Brier Score [65] of the predicted values on test data. Such mean values are obtained by the BrierScore procedure from DescTools R package by the same cross-validation process described above (see lines 49–53 in Figure 15).

Results presented in Figure 16 indicate a very good level of efficiency $E(C)$ in both the training and test phases. Values listed in Table 6 also indicate a good level for Brier Scores for test data.

From all Figures 13 and 14, we observe that, for higher values of N , the neural network fitting function $f_{sigmoid}(x; w, N)$ suffers from an “overfitting” problem. High values of N seem to be related to fitting functions with a low level of “smoothness” [38]. All the results presented in those Figures suggest choosing $N \leq 700$ to prevent such a problem. Also, the use of “regularization techniques” aimed to limit the amplification of the weights values seems to be useful. Plots in Figure 17 show results from the plotIterativeError procedure related to the use of the “Weight Decay” in learning procedure [66] (see lines 24 and 25 in Figure 11). These plots show that divergence behavior throughout the learning phase on test data appears to be less prominent.

Table 6. Mean Brier Score trends as function of NN complexity N .

N	Mean Brier Score
100	0.0100457
300	0.0100457
500	0.0109589
700	0.0095890
900	0.0146119
1100	0.0127854

Analyze the relative importance of marker variables: to assess the relevance of markers in identifying sick status. See Figure 18 for results from an *Olden-type Analysis* of the neural network learning phase [53,62]. The plots are obtained for different values of N by the olden procedure from the NeuralNetTools package [67] (see line 32 in Figure 11). That procedure implements the Olden’s connection weights method described in Section 2.3.3. *Olden-type Analysis*, by computing the relevance coefficient $r_{i=1, \dots, N_{Par}}^2$ for all the N_{Par} parameters, finds that the parameters numbered 3, 1, 7, and 8, respectively, associated with the markers NLRP3bActinSkin, IL18Skin, IL6Serum, and SodSerum (see Table 5), are the

most relevant in identifying sick status (second output variable `out_var='Output_1'`, see line 32 in Figure 11).

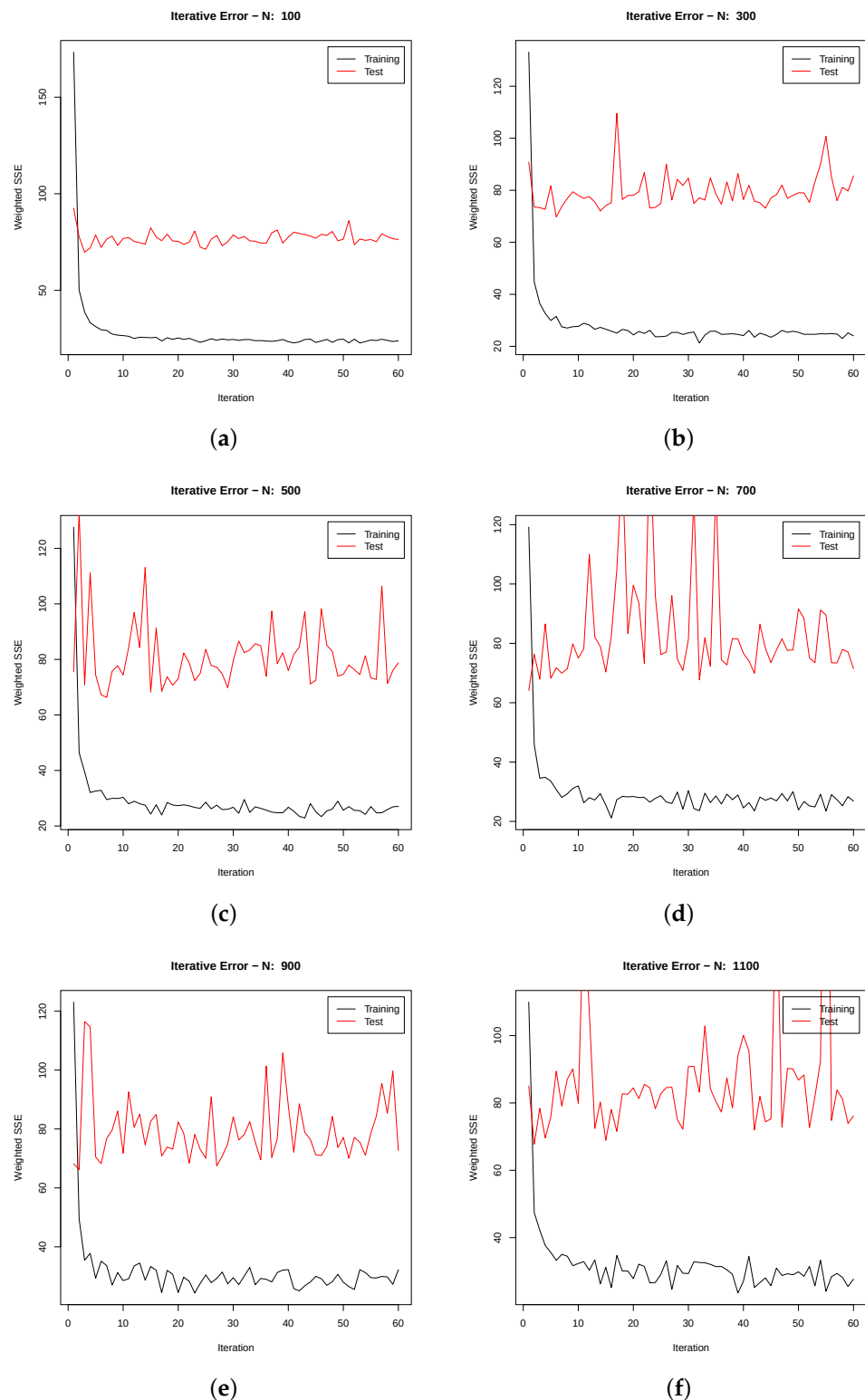


Figure 17. Convergence error plot for the test data where learning function is based on “weight decay”. (a) $N = 100$, (b) $N = 300$, (c) $N = 500$, (d) $N = 700$, (e) $N = 900$, (f) $N = 1100$.

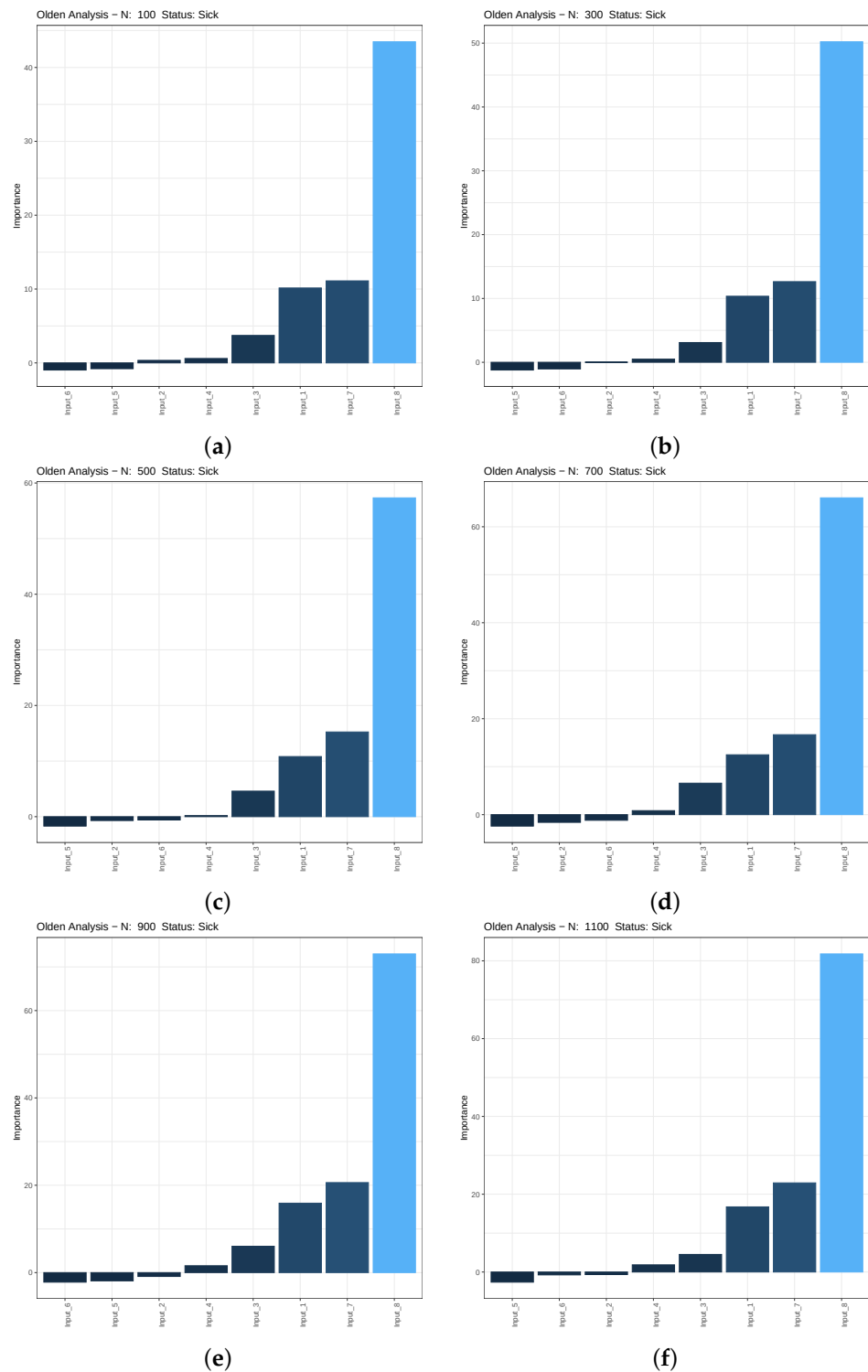


Figure 18. Parameters' relevance in identifying sick status evaluated by an Olden-type analysis. (a) $N = 100$, (b) $N = 300$, (c) $N = 500$, (d) $N = 700$, (e) $N = 900$, (f) $N = 1100$.

4. Discussion

Based on the results presented in this study, both the qualitative and quantitative analyses described in Sections 3.4 and 3.5 indicate that the most significant molecular markers for identifying and characterizing ALS disease are *NLRP-3* in skin, *IL-18* in skin, *IL-6* in serum, *SOD* in serum, *TGF- β* in skin, and *pTDP-43* in skin.

For example, the correlations highlighted from PCA can be interpreted as in the following. The activated *NLRP-3* (pyroptosis) in skin amplifies local inflammation by

the release of *IL-18* [68]. Thus, *IL-18* in the dermis is a direct marker of inflammasome activation. *IL-6* is also strictly connected to a pyroptosis cascade and it is correlated with an inflammatory response in the acute phase and associated with oxidative stress and *SOD* dysfunction. In this scenario, altered *SOD* triggers activation of *TGF- β* with an imbalance of neurogenesis and neurodegeneration, stimulates *NLRP-3*, releasing *IL-18* in tissues (including skin), thus contributing to disease progression [69]. Finally, *pTDP-43* deposits in skin activate inflammatory responses, including *NLRP-3*, *IL-6*, *IL-18* and *TGF- β* . This interpretation highlights how PCA dimensions capture biologically relevant pathways, reinforcing the clinical significance of our multivariate approach.

From a biochemical perspective, the findings suggest that *NLRP-3*, as a component of the inflammasome cascade, may play a role in ALS pathogenesis, and its gene expression levels could serve as a biomarker for predicting ALS onset and progression in human patients. Moreover, the detection of pathological *TDP-43* aggregates in skin biopsies from living ALS patients, compared to controls, points to a strong link between inflammasome activation and neurodegenerative processes.

Additionally, persistent alterations in the *TGF- β* system associated with *NLRP-3* activation have emerged as a potential therapeutic target for ALS. Our results also revealed elevated *IL-6* levels in the serum of ALS patients. To further investigate oxidative stress related to ALS disorders, we assessed total *SOD* antioxidant activity in ALS patient serum to determine whether *SOD* maintains its protective role. The results showed a marked inhibition of *SOD* activity in ALS patients, confirming that oxidative stress contributes significantly to ALS development and progression by suppressing antioxidant enzyme function. Finally, among all cytokines analyzed, *IL-18* production appears to be the most specific marker of inflammasome activation, closely linked to *TDP-43* aggregation triggered by *NLRP-3* activation.

In our study, the multivariate models based on the combination of mathematical and biochemical methods allowed identifying the most influential contributors to ALS disease classification and diagnosis (*NLRP-3* in skin, *IL-18* in skin, *IL-6* in serum, *TGF- β* in skin, *SOD* in serum, and *pTDP-43* in skin). These markers are known to be involved in biological processes, e.g., neuroinflammation, oxidative stress, which aligns with current understanding of ALS pathophysiology. Furthermore, the combination of neuroinflammation marker (*NLRP-3* and its related cytokines), oxidative stress marker (*SOD*) and neurodegeneration marker (*pTDP-43*) improved classification accuracy, suggesting potential synergistic roles in disease progression. In addition, this study was performed on tissues obtained from living ALS patients, complemented by matched serum measurements.

Previous investigations have predominantly relied on animal models (e.g., *SOD1G93A* mice/rats) [9,70], and post-mortem human tissues [71], which are invaluable for mechanistic insight but limited for real-time diagnosis and treatment decisions. This is the first study to combine mathematically interpretable multivariate modeling with biochemically validated correlation between oxidative, neuroinflammatory and neurodegenerative ALS biomarkers across serum and peripheral tissue, enabling biologically grounded diagnosis. Indeed, by measuring biochemically validated pathway nodes in viable patient tissue, our work establishes a new, minimally invasive diagnostic approach that can support severity ALS stratification and progression monitoring, thereby informing therapeutic management in the clinical setting.

5. Conclusions and Future Directions

To overcome some limitations of this study, which are principally due to (1) the high missingness, (2) the modest sample size, and (3) the generalizability of the findings on

more wide and complete cohorts, new studies already in progress will be focused on the following:

- Acquiring a more complete and wide datasets for which an effective assessment of missingness type for data (MNAR, MCAR, or MAR) [16] is available.
- Proposing and validating some evolution of the NN-based models, described in such study, that can solve the generalization problems related to small cohorts size very typical of clinical settings.

However, all the above considered, this study should be interpreted as a methodological and exploratory analysis rather than a definitive validation ready for clinical application. It provided us with a preliminary opportunity to design a “proof-of-concept”, which we envision as the starting point for future work.

About the identifying the most promising mathematical methodologies and techniques to use in ALS characterization, we will not ignore the analysis of “the state-of-the-art” as described in the literature [72,73]. Thanks to a deeper study of the “state-of-the-art”, our work will evolve with activities that allow the development of predictive models that can be used to improve the knowledge of the phenomena underlying the origin and evolution of the ALS disease. It could also lay the foundations for the definition of a protocol useful for constructing “digital twins” [74,75] of the entire process of study, diagnosis, and treatment of the disease from the perspective of innovative precision medicine. More in details, we intend to deepen all the aspects related with the “generalization” of the NN-based models in the sense introduced at the end of Section 2.3.3 with particular attention to “regularization techniques” and to the NN structure. Indeed, in addition to carrying out comparative tests with classic classification methods such as Random Forest ones [76] (see first very preliminary results proposed in Appendix B.1), we intends to investigate the use of graph neural networks (GNNs) that are considered very effective tools for learning on graph-structured data since have shown cutting-edge performance in a variety of applications, including biological network modeling and social network analysis [77]. Indeed, compared to analyzing data separately, the special capability of graphs to capture the structural relationships between data allows for the extraction of additional insights. The use of GNNs could also lay the basis for DNNs able overcome the “curse of dimensionality” problem since the fitting functions of DNN are based on hierarchically compositional functions which are local in the sense of bounded small dimensionality (see Section 2.3.3). Other key tools to build data-driven models potentially of interest are the recurrent neural network (RNNs) [78]: a class of neural networks designed for processing sequential data, such as text, speech, and time series [78].

Considering the aims of our work, related to ALS biomarkers identification and patient stratification, a multi-view *classification* [36], combined with a “Features Selection” operation [79], seems to be the most promising approach [80,81] since the multiomics nature of the problem. The combination of GNNs with RNNs will be investigated as a tool to predict ALS disease progression starting from time series collected data, the so called “longitudinal data” (see [82] for an example for such kind of application).

From the point of view of Computational Scientists, our work intend to be an useful case study to be used in the development of algorithms, such as the optimization algorithms that are the basis and of interest for many other application contexts, in the new High-Performance Computing (HPC) contexts made available in the nascent Exascale Computing Era [23,83–90].

Furthermore, the application of ML techniques in critical areas, such as healthcare, has made it necessary to understand their underlying mechanisms and their often opaque outputs. Explainable AI (XAI) [91] has emerged as a response to this demand, as it seeks to develop methods for explaining AI systems and their outputs “illuminating the NN black

box". Indeed, the increasing use of AI-based systems, especially in critical areas, has made XAI an area of study with significant practical and ethical value.

Author Contributions: Conceptualization, L.C. and I.F.; methodology, L.C.; software, L.C.; validation, L.C., U.D., R.D. and I.F.; data curation, U.D., R.D. and I.F.; writing—original draft preparation, L.C.; writing—review and editing, L.C., U.D., R.D. and I.F.; founding acquisition: R.D. and I.F. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge support from Progetto MUR PRIN2022 INFLAMM-ALS, Grant No. 2022BNZLMN.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and approved by the Comitato Etico Campania 3 of Ufficio Locale Comitato Etico-A.O.U. "Federico II" del (protocol number 100/17/ES01 (14 May 2017), protocol number 151/2023 (13 September 2023)).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to privacy reasons.

Acknowledgments: Luisa Carracciolo is a member of the "Gruppo Nazionale Calcolo Scientifico—Istituto Nazionale di Alta Matematica (GNCS-INdAM)". This work was carried out using the computational resources available at the scientific datacenter of the University of Naples Federico II.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ALS	Amyotrophic Lateral Sclerosis
PCA	Principal Component Analysis
MI	Multiple Imputation
NN	Neural Network

Appendix A. Mathematical Tools—Further Material

Appendix A.1. Handling Incomplete Data

Algorithm A1 The MICE algorithm.

```

1: procedure MICE( $Y, k, p, m, \{\mathcal{M}_j(Y_j^{obs}, Y_{-j}; \theta_j)\}_{j=1, \dots, k}, X$ )
2: Input:  $Y, k, p, m, \{\mathcal{M}_j(Y_j^{obs}, Y_{-j}; \theta_j)\}_{j=1, \dots, k}$ 
3: Output:  $X$ 
4:   parallel for  $i = 1, \dots, m$  do
5:     Initialize  $Y_{-1}^{(0)}$   $\triangleright$  Impute, by simple method, each uni-variate variable  $Y_j, j = 2, \dots, k$ 
6:     for  $t = 1, \dots, maxit$  do
7:       for  $j = 1, \dots, k$  do
8:          $\theta_j^{(t)} \leftarrow P(\theta_j | Y_j^{obs}, Y_{-j}^{(t)})$   $\triangleright$  Estimate parameter  $\theta_j$  defining model  $\mathcal{M}_j(\cdot, \cdot; \theta_j)$ 
9:          $Y_j^{*(t)} \leftarrow \mathcal{M}_j(Y_j^{obs}, Y_{-j}^{(t)}; \theta_j^{(t)})$   $\triangleright$  Impute missing values of  $Y_j$  by  $\mathcal{M}_j(\cdot, \cdot; \theta_j^{(t)})$ 
10:      end for
11:    end for
12:     $^{(i)}Y \leftarrow \left( \{Y_j^{(maxit)}\}_{j=1, \dots, k}, \{\tilde{Y}_{j'}\}_{j'=1, \dots, p-k} \right)$   $\triangleright$  Form the  $i$ -th complete data sample  $^{(i)}Y$ 
13:     $X \leftarrow X \cup ^{(i)}Y$   $\triangleright$  Add to the set  $X$  of completed data samples
14:  end parallel for
15: end procedure

```

Table A1. Relative Efficiency of Multiple Imputation with an Increasing Number of Imputations and Proportion of Censored Data (see table 1 in [17]).

No. of Imputations (m)	Proportion of Censored Data (λ)				
	10%	30%	50%	70%	90%
3	0.97	0.91	0.86	0.81	0.77
5	0.98	0.94	0.91	0.88	0.85
10	0.99	0.97	0.95	0.93	0.92
20	1.00	0.99	0.98	0.97	0.96
30	1.00	0.99	0.98	0.98	0.97

Appendix A.2. Interpretating Multidimensional Data by Principal Components Analysis

Proposition A1 (Relation between PCA and SVD of the matrix $\mathbf{Z}$). Any arbitrary matrix of dimension $n \times p$ and rank r (where necessarily, $r \leq \min(n, p)$), matrix $\mathbf{Z}$ can be written as

$$\mathbf{Z} = \mathbf{U}\mathbf{\Delta}\mathbf{V}^T, \quad (\text{A1})$$

where $\mathbf{U}, \mathbf{V}$ are $n \times r$ and $p \times r$ matrices with orthonormal columns ($\mathbf{U}^T\mathbf{U} = \mathbf{I}_r = \mathbf{V}^T\mathbf{V}$, with $\mathbf{I}_r$ the $r \times r$ identity matrix) and $\mathbf{\Delta}$ is an $r \times r$ diagonal matrix.

Since $\mathbf{R} = \mathbf{Z}^T\mathbf{Z}$, from Equation (A1) the following relations follows

$$\begin{aligned} \mathbf{R} &= \mathbf{Z}^T\mathbf{Z} \\ &= [\mathbf{U}\mathbf{\Delta}\mathbf{V}^T]^T [\mathbf{U}\mathbf{\Delta}\mathbf{V}^T] \\ &= [\mathbf{V}\mathbf{\Delta}\mathbf{U}^T\mathbf{U}\mathbf{\Delta}\mathbf{V}^T] \\ &= \mathbf{V}\mathbf{\Delta}^2\mathbf{V}^T \\ &= \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T, \end{aligned} \quad (\text{A2})$$

where, if $r = p$, $\mathbf{\Lambda} = \mathbf{\Delta}^2$ and $\mathbf{V} = \mathbf{A}$.

Proposition A2 (Reduced rank approximation $\mathbf{Z}_q$ of a matrix $\mathbf{Z}$ whose rank is equal to r (where $q < r$)). To build a “Reduced rank approximation $\mathbf{Z}_q$ ” of matrix $\mathbf{Z}$ whose rank is equal to r , the Truncated Singular Value Decomposition (TSVD) can be used. TSVD represents the best approximation of an arbitrary matrix in the spectral and Frobenius norms [83]. Then the matrix $\mathbf{Z}_q = \mathbf{U}_q\mathbf{\Delta}_q\mathbf{V}_q^T$, where $\mathbf{U}_q$ and $\mathbf{V}_q$ are matrices built on the first q rows of $\mathbf{U}$ and $\mathbf{V}$, and $\mathbf{\Delta}_q$ is the diagonal matrix built on the top-left $q \times q$ block of the matrix $\mathbf{\Delta}$, is the best “Reduced-Rank” $\mathbf{Z}_q$ approximation of matrix $\mathbf{Z}$. For matrix $\mathbf{Z}_q$ is valid

$$\|\mathbf{Z} - \mathbf{Z}_q\|_2 = \Delta_{r+1, r+1}, \quad (\text{A3})$$

$$\|\mathbf{Z} - \mathbf{Z}_q\|_F = \sqrt{\sum_{j=q+1}^r \Delta_{jj}^2}, \quad (\text{A4})$$

where $\|\cdot\|_2$ and $\|\cdot\|_F$ denote, respectively, the L2 and Frobenius norms.

Proposition A3 (Description of bidimensional Cartesian $\mathcal{C}^{(j_1, j_2)}$ system used by a BiPlot representation). Lets consider the reduced-rank approximation $\hat{\mathbf{Z}} = \mathbf{Z}_{(j_1, j_2)}$ that is built retaining only the j_1 -th and the j_2 -th singular values of $\mathbf{Z}$. Then, for matrices $\hat{\mathbf{Z}}_1$ and $\hat{\mathbf{Z}}_2$, the following expression can be considered

$$\hat{\mathbf{Z}}_1 = \mathbf{U}_{(j_1, j_2)}, \quad (\text{A5})$$

$$\hat{\mathbf{Z}}_2^T = \mathbf{\Delta}_{(j_1, j_2)} \mathbf{V}_{(j_1, j_2)}^T, \quad (\text{A6})$$

where $\mathbf{U}_{(j_1, j_2)}$ and $\mathbf{V}_{(j_1, j_2)}$ are matrices built on the j_1 -th and the j_1 -th rows of $\mathbf{U}$ and $\mathbf{V}$, and $\Delta_{(j_1, j_2)}$ is the diagonal matrix built on the j_1 -th and the j_1 -th diagonal elements of Δ . We recall that, if $j_1 = 1$ and $j_2 = 2$, we obtain the reduced-rank approximation $\hat{\mathbf{Z}} = \mathbf{Z}_{q=2}$ based on TSVD.

Consider the bidimensional Cartesian $\mathcal{C}^{(j_1, j_2)}$ system whose axes coincide with the two orthonormal vectors represented by the columns of the matrix $\mathbf{V}_{(j_1, j_2)}$. Each individuals $\mathbf{z}_i$ is represented as a point $P_i^{I(C1, C2)} = (IC1_i, IC2_i)$ whose coordinates are

$$IC1_i = \mathbf{b}_{ij_1} \quad (\text{A7})$$

$$IC2_i = \mathbf{b}_{ij_2}, \quad (\text{A8})$$

where the vectors $\mathbf{b}_j$ are defined as

$$\mathbf{b}_j = \mathbf{Z}(\mathbf{V}_{(j_1, j_2)})_j, j = 1, 2.$$

Indeed, each variable $\mathbf{z}_j$ is represented as a vector $(\mathbf{z}_j)^{\angle_{j_1, j_2}}$, starting at the origin of the Cartesian system, and ending into a point $P_j^{V(C1, C2)} = (VC1_j, VC2_j)$ whose coordinates are

$$VC1_j = \mathbf{c}_{1j}, \quad (\text{A9})$$

$$VC2_j = \mathbf{c}_{2j}, \quad (\text{A10})$$

where the vectors $\mathbf{c}_j$ are defined as

$$\mathbf{c}_j = \frac{1}{\sqrt{n}} (\Delta_{(j_1, j_2)} \mathbf{V}_{(j_1, j_2)}^T)_j, j = 1, 2$$

The squared lenght in L2 norm of the vector $(\mathbf{z}_j)^{\angle_{j_1, j_2}}$ is defined as $\cos 2(\mathbf{z}_j, \mathbf{V}_{(j_1, j_2)})$ of the projection of the variable $\mathbf{z}_j$ in the bidimensional Cartesian system whose axes coincide with the two orthonormal vectors $(\mathbf{V}_{(j_1, j_2)})_{j=1,2}$, i.e.,

$$\cos 2(\mathbf{z}_j, \mathbf{V}_{(j_1, j_2)}) = \left\| (\mathbf{z}_j)^{\angle_{j_1, j_2}} \right\|_2^2 = \mathbf{c}_{1j}^2 + \mathbf{c}_{2j}^2. \quad (\text{A11})$$

$\cos 2(\mathbf{z}_j, (\mathbf{V}_{(j_1, j_2)})_{j=1,2})$ shows the “representation accuracy” of variable $\mathbf{z}_j$ in the space spanned by the components $(\mathbf{V}_{(j_1, j_2)})_{j=1,2}$.

Appendix A.3. Building Predictive Model from Data by Neural Networks

Theorem A1 (DNN as Universal Approximator). Let the space Θ of the tuple θ of the parameters

$$\theta = \left(L, \{N_l\}_{l=2, \dots, L}, \left\{ w_{i^l j^l}^{(l)} \right\}_{\substack{l=2, \dots, L \\ i^l=1, \dots, N_l \\ j^l=1, \dots, N_{l-1}+1}}, \sigma \right)$$

defining the DNN $\mathcal{N}_\theta(n, k)$ acting on input $\mathbf{x} \in \mathbb{R}^n$ and output $\mathbf{y} \in \mathbb{R}^k$. Let $\mathcal{F}(n, k)$ be the functions set

$$\mathcal{F}(n, k) = \left\{ f^{\mathcal{N}_\theta(n, k)} : \theta \in \Theta \right\}$$

of all the functions $f^{\mathcal{N}_\theta(n, k)}$ whose form is defined as in Equation (10).

If σ is a measurable function, then $\mathcal{F}(n, k)$ is fundamental in the space $C(n, k)$ of all the continuous functions $g : \mathbb{R}^n \leftarrow \mathbb{R}^k$ if and only if σ is not a polynomial.

Theorem A2 (Curse of Dimensionality Theorem). Let $\mathcal{F}^{L=3}(n, k)$ be the subset of $\mathcal{F}(n, k)$ defined as

$$\mathcal{F}^{L=3}(n, k) = \{f^{\mathcal{N}_{\theta}(n, k)}, \theta \in \Theta : L = 3\}.$$

If σ is a measurable function and is not a polynomial, then $\mathcal{F}^{L=3}(n, k)$ is fundamental in the space $C(n, k)$, and for every $\epsilon > 0$ and every function $g \in C(n, k)$, there is a function $f^{\mathcal{N}_{\theta}(n, k)} \in \mathcal{F}^{L=3}(n, k)$ such that

$$\|g - f^{\mathcal{N}_{\theta}(n, k)}\| < \epsilon, \quad (\text{A12})$$

where the considered norm $\|\cdot\|$ is defined as

$$\|g(\mathbf{x})\| = \sup_{\mathbf{x} \in \mathbb{R}^n} |g(\mathbf{x})|.$$

Moreover, let W_m^n be the set of all the functions $g(\mathbf{x})$ in $C(n, k)$ with continuous partial derivatives of orders up to $m : 1 \leq m < \infty$, such that

$$\|g(\mathbf{x})\| + \sum_{1 \leq |\mathbf{k}|_1 \leq m} \|\mathcal{D}^{\mathbf{k}} g(\mathbf{x})\| \leq 1,$$

where $\mathcal{D}^{\mathbf{k}}$ denotes the partial derivative indicated by the multi-integer $\mathbf{k} \geq 1$, and $|\mathbf{k}|_1$ is the sum of the components of $\mathbf{k}$. If σ is infinitely differentiable, for every $g \in W_m^n$, the complexity N of network that provide accuracy at least ϵ to g is

$$N = O\left(\epsilon^{-\frac{n}{m}}\right), \quad (\text{A13})$$

and is the best possible.

Algorithm A2 The Gradient descent Algorithm.

```

1: procedure GRADIENTDESCENDENT( $C(\alpha)$ ,  $\alpha_0$ ,  $\alpha_{Best}$ ,  $maxit$ )
2: Input:  $C$ ,  $\alpha_0$ ,  $maxit$ 
3: Output:  $\alpha_{Best}$ 
4:    $n \leftarrow 1$ 
5:   repeat
6:      $\alpha_n \leftarrow \alpha_{n-1} - \gamma_n \nabla_{\alpha} C(\alpha_{n-1})$ 
7:      $n \leftarrow n + 1$ 
8:   until  $n > maxit$ 
9:    $\alpha_{Best} \leftarrow \alpha_n$ 
10: end procedure

```

Appendix B. Mathematical Tools—Further Results

Appendix B.1. Random Forest Parameter Relevance

This appendix describes some very preliminary results about the use of methods other than those based on NNs to assess parameters relevance in our case study.

These results were obtained using the procedures of the RandomForest package of R. In Figure A1 the R, code used is listed.

In Figure A2, instead, it is represented such relevance measured by the Mean Decrease Accuracy metrix. For details about such metrix and the RandomForest R package refer to Liaw et al. [92]. Presented results confirm those presented in the main part of this study.

```

1  #
2  # X is the results of MI phase (see code in Figure 4)
3  #
4
5  data <- X[,1:8]
6  data[,9] <- as.factor(X[,9])
7  colnames(data)[9] <- "Status"
8
9  data.RfImportance <- randomForest(Status ~ ., data = data,
10                                   importance = TRUE,
11                                   proximity = TRUE)
12  importance(data.RfImportance,type=1)
13
14  stitle <- paste('Random Forest Parameters Relevance')
15  varImpPlot(data.RfImportance, type =1, main=stitle)
16

```

Figure A1. R code of the Random Forest analysis.

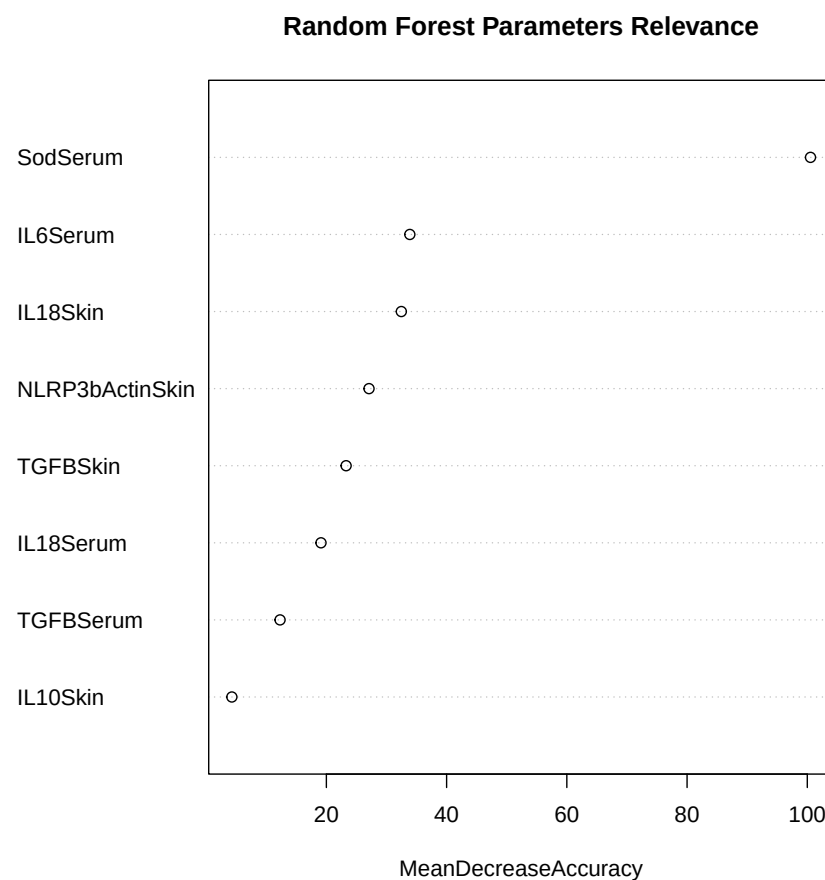


Figure A2. Parameters relevance in identifying sick status evaluated by Random Forest Training process.

References

1. Feldman, E.L.; Goutman, S.A.; Petri, S.; Mazzini, L.; Savelieff, M.G.; Shaw, P.J.; Sobue, G. Amyotrophic lateral sclerosis. *Lancet* **2022**, *400*, 1363–1380. [\[CrossRef\]](#) [\[PubMed\]](#)
2. Van Es, M.A.; Hardiman, O.; Chio, A.; Al-Chalabi, A.; Pasterkamp, R.J.; Veldink, J.H.; van den Berg, L.H. Amyotrophic lateral sclerosis. *Lancet* **2017**, *390*, 2084–2098. [\[CrossRef\]](#)
3. Piccione, E.A.; Sletten, D.M.; Staff, N.P.; Low, P.A. Autonomic system and amyotrophic lateral sclerosis. *Muscle Nerve* **2015**, *51*, 676–679. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Pugdahl, K.; Fuglsang-Frederiksen, A.; de Carvalho, M.; Johnsen, B.; Fawcett, P.R.W.; Labarre-Vila, A.; Liguori, R.; Nix, W.A.; Schofield, I.S. Generalised sensory system abnormalities in amyotrophic lateral sclerosis: A European multicentre study. *J. Neurol. Neurosurg. Psychiatry* **2007**, *78*, 746–749. [\[CrossRef\]](#) [\[PubMed\]](#)

5. Nolano, M.; Provitera, V.; Manganelli, F.; Iodice, R.; Stancanelli, A.; Caporaso, G.; Saltalamacchia, A.; Califano, F.; Lanzillo, B.; Picillo, M.; et al. Loss of cutaneous large and small fibers in naive and L-dopa-treated PD patients. *Neurology* **2017**, *89*, 776–784. [CrossRef]
6. Truini, A.; Vergari, M.; Biasiotta, A.; La Cesa, S.; Gabriele, M.; Di Stefano, G.; Cambieri, C.; Cruccu, G.; Inghilleri, M.; Priori, A. Transcutaneous spinal direct current stimulation inhibits nociceptive spinal pathway conduction and increases pain tolerance in humans. *Eur. J. Pain* **2011**, *15*, 1023–1027. [CrossRef]
7. Liu, Z.; Cheng, X.; Zhong, S.; Zhang, X.; Liu, C.; Liu, F.; Zhao, C. Peripheral and Central Nervous System Immune Response Crosstalk in Amyotrophic Lateral Sclerosis. *Front. Neurosci.* **2020**, *14*, 575. [CrossRef]
8. Voet, S.; Srinivasan, S.; Lamkanfi, M.; van Loo, G. Inflammasomes in neuroinflammatory and neurodegenerative diseases. *EMBO Mol. Med.* **2019**, *11*, EMM201810248. [CrossRef]
9. Johann, S.; Heitzer, M.; Kanagaratnam, M.; Goswami, A.; Rizo, T.; Weis, J.; Troost, D.; Beyer, C. NLRP3 inflammasome is expressed by astrocytes in the SOD1 mouse model of ALS and in human sporadic ALS patients. *Glia* **2015**, *63*, 2260–2273. [CrossRef]
10. Debye, B.; Schmülling, L.; Zhou, L.; Rune, G.; Beyer, C.; Johann, S. Neurodegeneration and NLRP3 inflammasome expression in the anterior thalamus of SOD1(G93A) ALS mice. *Brain Pathol.* **2018**, *28*, 14–27. [CrossRef]
11. Zhao, W.; Beers, D.R.; Bell, S.; Wang, J.; Wen, S.; Baloh, R.H.; Appel, S.H. TDP-43 activates microglia through NF- κ B and NLRP3 inflammasome. *Exp. Neurol.* **2015**, *273*, 24–35. [CrossRef] [PubMed]
12. Nowowiejska, J.; Baran, A.; Pryczynicz, A.; Hermanowicz, J.M.; Sieklucka, B.; Pawlak, D.; Flisiak, I. Gasdermin B (GSDMB) in psoriatic patients—A preliminary comprehensive study on human serum, urine and skin. *Front. Mol. Biosci.* **2024**, *11*, 1382069. [CrossRef] [PubMed]
13. Donoho, D. 50 Years of Data Science. *J. Comput. Graph. Stat.* **2017**, *26*, 745–766. [CrossRef]
14. Donoho, D. Data Science at the Singularity. *Harv. Data Sci. Rev.* **2024**, *6*, 1–51. Available online: <https://hdsr.mitpress.mit.edu/pub/g9mau4m0> (accessed on 26 January 2026). [CrossRef]
15. Donoho, D.L. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lect.* **2000**, *1*, 32. Available online: https://www.researchgate.net/publication/220049061_High-Dimensional_Data_Analysis_The_Curses_and_Blessings_of_Dimensionality (accessed on 26 January 2026).
16. Haukoos, J.S.; Newgard, C.D. Advanced Statistics: Missing Data in Clinical Research—Part 1: An Introduction and Conceptual Framework. *Acad. Emerg. Med.* **2007**, *14*, 662–668. [CrossRef]
17. Haukoos, J.S.; Newgard, C.D. Advanced Statistics: Missing Data in Clinical Research—Part 2: Multiple Imputation. *Acad. Emerg. Med.* **2007**, *14*, 669–678. [CrossRef]
18. Newgard, C.D.; Lewis, R.J. Missing Data: How to Best Account for What Is Not Known. *JAMA* **2015**, *314*, 940–941. [CrossRef]
19. Li, P.; Stuart, E.A.; Allison, D.B. Multiple Imputation: A Flexible Tool for Handling Missing Data. *JAMA* **2015**, *314*, 1966–1967. [CrossRef]
20. Austin, P.C.; White, I.R.; Lee, D.S.; van Buuren, S. Missing Data in Clinical Research: A Tutorial on Multiple Imputation. *Can. J. Cardiol.* **2021**, *37*, 1322–1331. [CrossRef]
21. Weeraratne, N.; Hunt, L.; Kurz, J. Optimizing PCA for Health and Care Research: A Reliable Approach to Component Selection. *arXiv* **2025**, arXiv:2503.24248. [CrossRef]
22. Bradley, W.; Kim, J.; Kilwein, Z.; Blakely, L.; Eydenberg, M.; Jalvin, J.; Laird, C.; Boukouvala, F. Perspectives on the integration between first-principles and data-driven modeling. *Comput. Chem. Eng.* **2022**, *166*, 107898. [CrossRef]
23. Carracciulo, L.; D’Amora, U. Mathematical Tools for Simulation of 3D Bioprinting Processes on High-Performance Computing Resources: The State of the Art. *Appl. Sci.* **2024**, *14*, 6110. [CrossRef]
24. Zhang, Z.; Beck, M.W.; Winkler, D.A.; Huang, B.; Sibanda, W.; Goyal, H.; on behalf of AME Big-Data Clinical Trial Collaborative Group. Opening the black box of neural networks: Methods for interpreting neural network models in clinical applications. *Ann. Transl. Med.* **2018**, *6*, 216. [CrossRef]
25. van Buuren, S.; Groothuis-Oudshoorn, K. MICE: Multivariate Imputation by Chained Equations in R. *J. Stat. Softw.* **2011**, *45*, 1–67. [CrossRef]
26. Azur, M.J.; Stuart, E.A.; Frangakis, C.; Leaf, P.J. Multiple imputation by chained equations: What is it and how does it work? *Int. J. Methods Psychiatr. Res.* **2011**, *20*, 40–49. [CrossRef]
27. Rubin, D.B. *Multiple Imputation for Nonresponse in Surveys*; Wiley Series in Probability and Statistics; Wiley: Hoboken, NJ, USA, 1987; p. 258. [CrossRef]
28. Little, R.J.A. Missing-Data Adjustments in Large Surveys. *J. Bus. Econ. Stat.* **1988**, *6*, 287–296. [CrossRef]
29. Lee, S. Mastering Predictive Mean Matching for Accurate Data Imputation. 2025. Available online: <https://www.numberanalytics.com/blog/predictive-mean-matching-data-imputation-guide> (accessed on 26 January 2026).
30. Jolliffe, I.T.; Cadima, J. Principal component analysis: A review and recent developments. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2016**, *374*, 20150202. [CrossRef]
31. Abdi, H.; Williams, L.J. Principal component analysis. *WIREs Comput. Stat.* **2010**, *2*, 433–459. [CrossRef]

32. Aluja, T.; Morineau, A.; Sanchez, G. PCA4DS: Principal Component Analysis for Data Science. 2018. Available online: <https://pca4ds.github.io> (accessed on 26 January 2026).
33. Jolliffe, I.T. *Principal Component Analysis*; Springer Series in Statistics; Springer: Berlin/Heidelberg, Germany, 2002.
34. Gabriel, K.R. The Biplot Graphic Display of Matrices with Application to Principal Component Analysis. *Biometrika* **1971**, *58*, 453–467. [\[CrossRef\]](#)
35. Friendly, M.; Monette, G.; Fox, J. Elliptical Insights: Understanding Statistical Methods through Elliptical Geometry. *Stat. Sci.* **2013**, *28*, 1–39. [\[CrossRef\]](#)
36. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [\[CrossRef\]](#) [\[PubMed\]](#)
37. Nielsen, M. *Neural Networks and Deep Learning*; Determination Press: San Francisco, CA, USA, 2016. Available online: <http://neuralnetworksanddeeplearning.com/index.html> (accessed on 26 January 2026).
38. Augustine, M.T. A Survey on Universal Approximation Theorems. *arXiv* **2024**, arXiv:2407.12895. [\[CrossRef\]](#)
39. Mhaskar, H.N. Approximation properties of a multilayered feedforward artificial neural network. *Adv. Comput. Math.* **1993**, *1*, 61–80. [\[CrossRef\]](#)
40. Chui, C.K.; Li, X.; Mhaskar, H.N. Limitations of the approximation capabilities of neural networks with one hidden layer. *Adv. Comput. Math.* **1996**, *5*, 233–243. [\[CrossRef\]](#)
41. Pinkus, A. Approximation theory of the MLP model in neural networks. *Acta Numer.* **1999**, *8*, 143–195. [\[CrossRef\]](#)
42. Poggio, T.; Mhaskar, H.; Rosasco, L.; Miranda, B.; Liao, Q. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *Int. J. Autom. Comput.* **2017**, *14*, 503–519. [\[CrossRef\]](#)
43. Leshno, M.; Lin, V.Y.; Pinkus, A.; Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Netw.* **1993**, *6*, 861–867. [\[CrossRef\]](#)
44. Bubeck, S. Convex Optimization: Algorithms and Complexity. *Found. Trends Mach. Learn.* **2015**, *8*, 231–357. [\[CrossRef\]](#)
45. Garrigos, G.; Gower, R.M. Handbook of Convergence Theorems for (Stochastic) Gradient Methods. *arXiv* **2024**, arXiv:2301.11235. [\[CrossRef\]](#)
46. Hochreiter, S.; Younger, A.S.; Conwell, P.R. Learning to Learn Using Gradient Descent. In *Artificial Neural Networks—ICANN 2001*; Dorffner, G., Bischof, H., Hornik, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2001; pp. 87–94.
47. Amari, S.I. Backpropagation and stochastic gradient descent method. *Neurocomputing* **1993**, *5*, 185–196. [\[CrossRef\]](#)
48. Poliak, B. *Introduction to Optimization*; Translations series in mathematics and engineering; Optimization Software, Publications Division: New York, NY, USA, 1987.
49. Curry, H.B. The method of steepest descent for non-linear minimization problems. *Quart. Appl. Math.* **1944**, *2*, 258–261. [\[CrossRef\]](#)
50. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [\[CrossRef\]](#)
51. Pentos, K. The methods of extracting the contribution of variables in artificial neural network models—Comparison of inherent instability. *Comput. Electron. Agric.* **2016**, *127*, 141–146. [\[CrossRef\]](#)
52. Gevrey, M.; Dimopoulos, I.; Lek, S. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecol. Model.* **2003**, *160*, 249–264. [\[CrossRef\]](#)
53. Olden, J.D.; Jackson, D.A. Illuminating the “black box”: A randomization approach for understanding variable contributions in artificial neural networks. *Ecol. Model.* **2002**, *154*, 135–150. [\[CrossRef\]](#)
54. Brownlee, J. *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions*; Machine Learning Mastery: San Juan, PR, USA, 2018.
55. Oxford English Dictionary. Overfitting Noun. 2024. Available online: https://www.oed.com/dictionary/overfitting_n?tl=true (accessed on 26 January 2026).
56. R Core Team. *R: A Language and Environment for Statistical Computing*; R Foundation for Statistical Computing: Vienna, Austria, 2013; ISBN 3-900051-07-0. Available online: <http://www.R-project.org/> (accessed on 26 January 2026).
57. MICE Package Reference Manual. *mice.impute.pmm* Imputation by Predictive Mean Matching. 2025. Available online: <https://cran.r-project.org/web/packages/mice/refman/mice.html#mice.impute.pmm> (accessed on 26 January 2026).
58. MICE Package Developer. *find.collinear* Internal MICE Procedure. 2024. Available online: <https://github.com/amices/mice/blob/7df3487a56cd49dfc9192a8ebe9697bfa7258ca/R/internal.R#L84-L91> (accessed on 26 January 2026).
59. Rosenblatt, M. Remarks on Some Nonparametric Estimates of a Density Function. *Ann. Math. Stat.* **1956**, *27*, 832–837. [\[CrossRef\]](#)
60. Wickham, H. *ggplot2: Elegant Graphics for Data Analysis*; Springer: New York, NY, USA, 2016. Available online: <https://ggplot2-book.org/> (accessed on 26 January 2026).
61. Choudhury, A.D.; Banerjee, R.; Kimbahune, S.; Pal, A. Chapter 3—Sensor signal analytics. In *New Frontiers of Cardiovascular Screening Using Unobtrusive Sensors, AI, and IoT*; Choudhury, A.D., Banerjee, R., Kimbahune, S., Pal, A., Eds.; Academic Press: Cambridge, MA, USA, 2022; pp. 61–89. [\[CrossRef\]](#)
62. Olden, J.D.; Joy, M.K.; Death, R.G. An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecol. Model.* **2004**, *178*, 389–397. [\[CrossRef\]](#)

63. Bergmeir, C.; Benítez, J.M. Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. *J. Stat. Softw.* **2012**, *46*, 1–26. [\[CrossRef\]](#)
64. McLachlan, G.; Do, K.; Ambroise, C. *Analyzing Microarray Gene Expression Data*; Wiley Series in Probability and Statistics; Wiley: Hoboken, NJ, USA, 2004. [\[CrossRef\]](#)
65. Brier, G.W. Verification of forecasts expressed in terms of probability. *Mon. Weather Rev.* **1950**, *78*, 1–3. [\[CrossRef\]](#)
66. Gupta, A.; Lam, M. The weight decay backpropagation for generalizations with missing values. *Ann. Oper. Res.* **1998**, *78*, 165–187. [\[CrossRef\]](#)
67. Beck, M.W. NeuralNetTools: Visualization and Analysis Tools for Neural Networks. *J. Stat. Softw.* **2018**, *85*, 1–20. [\[CrossRef\]](#) [\[PubMed\]](#)
68. Xu, W.; Huang, Y.; Zhou, R. NLRP3 inflammasome in neuroinflammation and central nervous system diseases. *Cell. Mol. Immunol.* **2025**, *22*, 341–355. [\[CrossRef\]](#) [\[PubMed\]](#)
69. Peters, S.; Zitzelsperger, E.; Kuespert, S.; Iberl, S.; Heydn, R.; Johannesen, S.; Petri, S.; Aigner, L.; Thal, D.R.; Hermann, A.; et al. The TGF- β System As a Potential Pathogenic Player in Disease Modulation of Amyotrophic Lateral Sclerosis. *Front. Neurol.* **2017**, *8*, 669. [\[CrossRef\]](#)
70. Gugliandolo, A.; Giacompo, S.; Bramanti, P.; Mazzon, E. NLRP3 Inflammasome Activation in a Transgenic Amyotrophic Lateral Sclerosis Model. *Inflammation* **2018**, *41*, 93–103. [\[CrossRef\]](#)
71. Arseni, D.; Hasegawa, M.; Murzin, A.G.; Kametani, F.; Arai, M.; Yoshida, M.; Ryskeldi-Falcon, B. Structure of pathological TDP-43 filaments from ALS with FTL. *Nature* **2022**, *601*, 139–143. [\[CrossRef\]](#)
72. Grollemund, V.; Pradat, P.-F.; Querin, G.; Delbot, F.; Le Chat, G.; Pradat-Peyre, J.-F.; Bede, P. Machine Learning in Amyotrophic Lateral Sclerosis: Achievements, Pitfalls, and Future Directions. *Front. Neurosci.* **2019**, *13*, 135. [\[CrossRef\]](#)
73. Chia, R.; Moaddel, R.; Kwan, J.Y.; Rasheed, M.; Ruffo, P.; Landeck, N.; Reho, P.; Vasta, R.; Calvo, A.; Moglia, C.; et al. A plasma proteomics-based candidate biomarker panel predictive of amyotrophic lateral sclerosis. *Nat. Med.* **2025**, *31*, 3440–3450. [\[CrossRef\]](#)
74. Singh, M.; Fuenmayor, E.; Hinchey, E.P.; Qiao, Y.; Murray, N.; Devine, D. Digital Twin: Origin to Future. *Appl. Syst. Innov.* **2021**, *4*, 36. [\[CrossRef\]](#)
75. Kamel Boulos, M.N.; Zhang, P. Digital Twins: From Personalised Medicine to Precision Public Health. *J. Pers. Med.* **2021**, *11*, 745. [\[CrossRef\]](#)
76. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [\[CrossRef\]](#)
77. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The Graph Neural Network Model. *IEEE Trans. Neural Networks* **2009**, *20*, 61–80. [\[CrossRef\]](#) [\[PubMed\]](#)
78. Tealab, A. Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Comput. Inform. J.* **2018**, *3*, 334–340. [\[CrossRef\]](#)
79. Theng, D.; Bhoyar, K.K. Feature selection techniques for machine learning: A survey of more than two decades of research. *Knowl. Inf. Syst.* **2023**, *66*, 1575–1637. [\[CrossRef\]](#)
80. Rad, H.; Su, Z.; Trinh, A.; Newton, M.A.; Shamsani, J.; Karim, A.; Sattar, A.; NYGC ALS Consortium. Amyotrophic lateral sclerosis diagnosis using machine learning and multi-omic data integration. *Heliyon* **2024**, *10*, e38583. [\[CrossRef\]](#)
81. Wang, T.; Shao, W.; Huang, Z.; Tang, H.; Zhang, J.; Ding, Z.; Huang, K. MOGONET integrates multi-omics data using graph convolutional networks allowing patient classification and biomarker identification. *Nat. Commun.* **2021**, *12*, 3445. [\[CrossRef\]](#)
82. Nguyen, M.; He, T.; An, L.; Alexander, D.C.; Feng, J.; Yeo, B.T.T. Predicting Alzheimer’s disease progression using deep recurrent neural networks. *NeuroImage* **2020**, *222*, 117203. [\[CrossRef\]](#)
83. Carracciolo, L.; Mele, V. New Strategies Based on Hierarchical Matrices for Matrix Polynomial Evaluation in Exascale Computing Era. *Mathematics* **2025**, *13*, 1378. [\[CrossRef\]](#)
84. Carracciolo, L.; Lapegna, M. Implementation of a non-linear solver on heterogeneous architectures. *Concurr. Comput. Pract. Exp.* **2018**, *30*, e4903. [\[CrossRef\]](#)
85. Mele, V.; Constantinescu, E.M.; Carracciolo, L.; D’Amore, L. A PETSc parallel-in-time solver based on MGRIT algorithm. *Concurr. Comput. Pract. Exp.* **2018**, *30*, e4928. [\[CrossRef\]](#)
86. Carracciolo, L.; Mele, V.; Szustak, L. About the granularity portability of block-based Krylov methods in heterogeneous computing environments. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6008. [\[CrossRef\]](#)
87. Carracciolo, L.; Casaburi, D.; D’Amore, L.; D’Avino, G.; Maffettone, P.; Murli, A. Computational simulations of 3D large-scale time-dependent viscoelastic flows in high performance computing environment. *J. Non-Newton. Fluid Mech.* **2011**, *166*, 1382–1395. [\[CrossRef\]](#)
88. Carracciolo, L.; D’Amore, L.; Murli, A. Towards a parallel component for imaging in PETSc programming environment: A case study in 3-D echocardiography. *Parallel Comput.* **2006**, *32*, 67–83. [\[CrossRef\]](#)
89. Murli, A.; D’Amore, L.; Carracciolo, L.; Ceccarelli, M.; Antonelli, L. High performance edge-preserving regularization in 3D SPECT imaging. *Parallel Comput.* **2008**, *34*, 115–132. [\[CrossRef\]](#)

90. D'Amore, L.; Constantinescu, E.; Carracciuolo, L. A Scalable Space-Time Domain Decomposition Approach for Solving Large Scale Nonlinear Regularized Inverse Ill Posed Problems in 4D Variational Data Assimilation. *J. Sci. Comput.* **2022**, *91*, 59. [[CrossRef](#)]
91. Longo, L.; Brcic, M.; Cabitza, F.; Choi, J.; Confalonieri, R.; Del Ser, J.; Guidotti, R.; Hayashi, Y.; Herrera, F.; Holzinger, A.; et al. Explainable Artificial Intelligence (XAI) 2.0: A manifesto of open challenges and interdisciplinary research directions. *Inf. Fusion* **2024**, *106*, 102301. [[CrossRef](#)]
92. Liaw, A.; Wiener, M. Classification and Regression by randomForest. *R News* **2002**, *2*, 18–22. Available online: <https://journal.r-project.org/articles/RN-2002-022/RN-2002-022.pdf> (accessed on 26 January 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.