

Article

Damping Ratio Prediction for Redundant Cartesian Impedance-Controlled Robots Using Machine Learning Techniques

José Patiño ¹, Ángel Encalada-Dávila ¹, José Sampietro ², Christian Tutivén ¹, Carlos Saldarriaga ^{1,*}
and Imin Kao ³

¹ Faculty of Mechanical Engineering and Production Sciences (FIMCP), ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, Campus Gustavo Galindo, Km. 30.5 Vía Perimetral, Guayaquil EC09015863, Ecuador

² Facultad de Ingenierías, Universidad Ecotec, Km. 13.5 Samborondón, Samborondón EC092302, Ecuador

³ Department of Mechanical Engineering, Stony Brook University, Stony Brook, NY 11794, USA

* Correspondence: cxsaldar@espol.edu.ec

Abstract: Implementing impedance control in Cartesian task space or directly at the joint level is a popular option for achieving desired compliance behavior for robotic manipulators performing tasks. The damping ratio is an important control criterion for modulating the dynamic response; however, tuning or selecting this parameter is not easy, and can be even more complicated in cases where the system cannot be directly solved at the joint space level. Our study proposes a novel methodology for calculating the local optimal damping ratio value and supports it with results obtained from five different scenarios. We carried out 162 different experiments and obtained the values of the inertia, stiffness, and damping matrices for each experiment. Then, data preprocessing was carried out to select the most significant variables using different criteria, reducing the seventeen initial variables to only three. Finally, the damping ratio values were calculated (predicted) using automatic regression tools. In particular, five-fold cross-validation was used to obtain a more generalized model and to assess the forecasting performance. The results show a promising methodology capable of calculating and predicting control parameters for robotic manipulation tasks.

Keywords: robotic manipulator; Cartesian impedance; MCK system; machine learning; XGBoost; random forest; support vector regressor; LightGBM; CatBoost

MSC: 68T40; 68T07



Citation: Patiño, J.; Encalada-Dávila, Á.; Sampietro, J.; Tutivén, C.; Saldarriaga, C.; Kao, I. Damping Ratio Prediction for Redundant Cartesian Impedance-Controlled Robots Using Machine Learning Techniques. *Mathematics* **2023**, *11*, 1021. <https://doi.org/10.3390/math11041021>

Academic Editors: Shuai Li, Dechao Chen, Vasilios N. Katsikis, Predrag S. Stanimirovic, Dunhui Xiao and Mohammed Aquil Mirza

Received: 27 December 2022

Revised: 28 January 2023

Accepted: 10 February 2023

Published: 17 February 2023



Copyright: © 2023 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/>).

1. Introduction

In order to achieve the desired compliant behavior for a robotic manipulator performing a task, a popular choice is implementing impedance control at either the task (Cartesian) space or directly at the joint level. Among these common tasks are wiping, deburring, polishing, and several others for human–robot interaction, in which safety becomes an important feature that must be handled in a careful manner [1].

Usually, in the one-DoF (Degree of Freedom) mass-damper-spring case or in discrete systems that can be decoupled, a set of control criteria such as damping ratios ζ or natural frequencies ω_n can be imposed according to particular (scalar) parameters. After the system is no longer decoupled, the tuning or selection of the parameters is not straightforward, and it becomes much more complicated in cases of unconstrained discrete systems or redundant serial kinematic chains, which are not directly solvable at the joint space level.

In this work, a new methodology is applied to perform a joint space analysis based on the theory of mechanical vibrations of discrete unconstrained systems. As per the expansion theorem, the solution of unconstrained systems such as the one shown in Figure 1 can be

obtained by separately solving for the rigid body and non-rigid body modes of motion; then, the linear combination of all these obtained modes form the complete response of the system. Analogous to the case of discrete systems, a closed-form solution to redundant robotic manipulators performing impedance control can be found that allows for modulating the dynamic response and selecting an appropriate set of stiffness and damping parameters according to the control criteria.

In general, in robotics there is a rule of thumb that tasks which are complex for any human being are likely straightforward for any robot, while tasks that are easy for humans end up being highly complex for any robot. Most recent advances in machine learning (ML) can successfully teach planar and general robots to perform complex tasks such as locomotion [2,3], and manipulation [4–6]. Furthermore, there are ML frameworks that provide baseline algorithms [7] which can be employed to solve general tasks. Other approaches focus these algorithms on performing industrial tasks using physical robots [8,9]. Meanwhile, other strategies have overcome the data hurdle by integrating ML with physical simulations [10,11] and employing sim-to-real techniques to transfer learned tasks from virtual to physical robots [12].

Regarding the novel points of this work, we propose ML techniques to predict or compute the damping ratio values that correspond to the dominant oscillatory mode of motion of the multi-dimensional impedance control system of a redundant robotic manipulator. Recently, ML or deep learning techniques have been widely applied in the robotic control field, for example, in optimal impedance control for redundant manipulators (i.e., [13]). However, in this case, the application of ML techniques for predicting the damping ratio is particularly novel. The proposed methodology differs from theory-based damping ratio calculation in that it does not require the application of mathematical equations or known physical principles in order to make estimates; instead, ML techniques use algorithms to estimate the damping ratio from historical data. A number of advantages of using ML techniques are listed below.

- Improved accuracy; machine learning can analyze large amounts of historical data and learn complex patterns involving the damping ratio, thereby improving prediction accuracy.
- Handling of redundant variables; machine learning is capable of handling redundant variables that can occur in a system controlled by redundant Cartesian impedance, making predictions more robust.
- Ability to adapt to different conditions; machine learning algorithms can be adjusted to adapt to different operating conditions, which improves the generalization of the model.
- Modal analysis data; data are obtained from modal analysis on the robot system instead of from arbitrary trial and error practices.
- Continuous improvement; machine learning allows systems to improve over time, meaning that they can be updated with new data and their performance can be improved.
- Use of different techniques; the use of different machine learning techniques such as XGBoost, support vector regression, random forest, LightGBM, and CatBoost allows different models to be compared and the best one applied to a given robot control system.

The rest of this article is structured as follows. In Section 2, the current limitations on the computation of damping ratio values are described and the need for a new methodology is shown. Section 3 states the most important basis for understanding the concept of Cartesian impedance control. In Section 4, the dynamic response modulation of a general system with damping is described mathematically. The 3R planar robot experimental setup and data acquisition process are introduced in Section 5. A full explanation of data preprocessing is detailed in Section 6, along with the proposed prediction methodologies. The obtained results are presented in Section 7. Lastly, the main conclusions are provided in Section 8, along with possible future research directions.

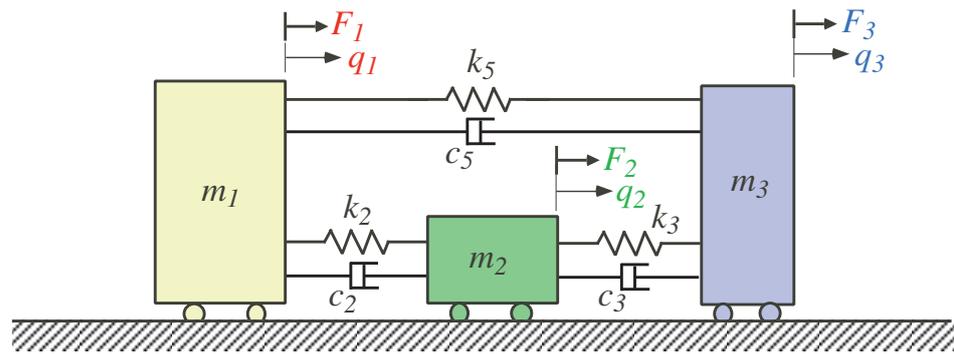


Figure 1. Non-conservative and unconstrained three-DoF discrete mechanical system.

2. Related Works

For general mechanical systems, the damping ratio and natural frequency are a common and appropriate set of control criteria for the dynamic response of the system. There are different methods for obtaining or computing these control criteria, among which are modelling, system identification, and automatic learning through previous data, among others that combine these principles [14,15]. In other applications, such as structural engineering, machine learning tools have been used to obtain equivalent damping ratios of reinforced concrete walls in seismic events [16]. However, for redundant robotic systems performing impedance control with multidimensional mass, damping, and stiffness parameters, the computation and optimization of these control criteria, that is, the damping ratio, is not straightforward and cannot be directly treated as a decoupled system of equations, at least not from a sound fundamental point of view.

In most cases, the translational and rotational Cartesian impedance control parameters, particularly damping, are chosen in one of the following ways: First, through trial and error on the robot setup by experimentally testing for stability regions [17], computationally obtaining values based on previous suitable data [18], imposing linear combinations involving the Cartesian stiffness, or arbitrarily assuming a set of m or n (depending on whether the equations are specified at the Cartesian or joint level, respectively) decoupled **M-C-K** (mass-damping-stiffness) equations (which in general is not the case) and choosing a Cartesian space damping matrix as $\mathbf{C}_C = 2\sqrt{\mathbf{K}_C}$ without considering either the effects of the off-diagonal elements of the inertia-mass matrix or performing a joint space analysis of the response using the configuration-dependant Jacobian matrix of the robot.

In order to choose or compute an appropriate set of impedance parameters, a joint space-based analysis for stiffness [19–21] and damping makes sense, as what is physically needed to be applied is a set of joint torques at the motors of the robot within a certain desired or limited range.

For redundant robotic manipulators, where there are more joints than task coordinates ($n > m$), after the Cartesian impedance system equations are mapped into the joint space those extra DoFs make the system impossible to solve analytically except by handling and solving for the rigid-body and non-rigid-body or zero-potential-energy and non-zero-potential-energy motions [22] separately. When the system is solvable, the effect of each element of the stiffness and damping matrices can be observed and used to generate parameter studies that allow an appropriate or optimal set of values to be chosen that modulate the dynamic response according to the desired control criteria in the modal space.

3. Cartesian Impedance Control

Robotic impedance control grows out of the one-DoF second-order mechanical system with a mass, damper, spring, and external force $m\ddot{x} + c\dot{x} + kx = f$. When this type of system is applied to a robotic manipulator, this concept is extended to multiple DoF matrix equations with coupled non-diagonal mass, damping, and stiffness matrices with certain dynamic behavior governed by the aforementioned matrices [23,24]. At the Cartesian or task space level [25], the impedance control of a robot becomes

$$\mathbf{M}_C \ddot{\mathbf{x}}(t) + \mathbf{C}_C \dot{\mathbf{x}}(t) + \mathbf{K}_C \mathbf{x}(t) = \mathbf{f} \tag{1}$$

with \mathbf{x} being the vector of translational and rotational displacements, \mathbf{f} the vector of external forces and moments, and the mass, damping, and stiffness matrices with subscript C expressed in the Cartesian space. From the dynamic equation of motion of a robot provided by

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}(t) + \mathbf{G}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}}(t) + \mathbf{v}(\mathbf{q}) = \boldsymbol{\tau}_m + \boldsymbol{\tau}_{ext} \tag{2}$$

where \mathbf{q} is the vector of the n joint angles, \mathbf{M} is the manipulator’s mass matrix, \mathbf{G} is the matrix containing the centrifugal and Coriolis terms, and \mathbf{v} is the gravity vector term, if we choose the motor torques $\boldsymbol{\tau}_m$ as $[-\mathbf{K}(\mathbf{q})\mathbf{q}(t) - \mathbf{C}(\mathbf{q})\dot{\mathbf{q}}(t) + \mathbf{v}(\mathbf{q}) + \mathbf{G}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}(t)]$, the system in Equation (2) becomes

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}(t) + \mathbf{C}\dot{\mathbf{q}}(t) + \mathbf{K}\mathbf{q}(t) = \boldsymbol{\tau}_{ext} \tag{3}$$

where the external torques $\boldsymbol{\tau}_{ext}$ and the mass, damping, and stiffness matrices establish the joint space impedance control system of equations. The respective Cartesian stiffness and damping matrices \mathbf{K}_C and \mathbf{C}_C can be mapped to the joint space using the manipulator Jacobian matrix $\mathbf{J}(\mathbf{q})$ [26,27]

$$\mathbf{K} = \mathbf{J}^T \mathbf{K}_C \mathbf{J} + \left[\left(\frac{\partial \mathbf{J}^T}{\partial q_1} \mathbf{f} \right) \left(\frac{\partial \mathbf{J}^T}{\partial q_2} \mathbf{f} \right) \cdots \left(\frac{\partial \mathbf{J}^T}{\partial q_n} \mathbf{f} \right) \right] + \mathbf{J}^T \mathbf{C}_C \mathbf{J} \tag{4}$$

$$\mathbf{C} = \mathbf{J}^T \mathbf{C}_C \mathbf{J} \tag{5}$$

making Equations (1) and (3) equivalent to each other.

4. Dynamic Response Modulation of a General System with Damping

The dynamic response of a multi-dimensional non-conservative mechanical system can be obtained in the modal space [14], in which the multiple DoFs in Equation (3) are decoupled such that each DoF can be independently analyzed to obtain and tune the damping ratios and natural frequencies of the response accordingly. In the case of a redundant robot performing impedance control-related tasks defined in the Cartesian space, the system in the joint space becomes positive semidefinite due to the additional ($r = n - m$) DoFs of the robot. This makes both the joint stiffness and damping matrices singular when they are mapped from the Cartesian space using Equations (4) and (5). The methodology presented here reduces the system to a positive definite system that can be solved analytically. If the robot is non-redundant, the system is positive definite and transformation to a reduced space is not necessary.

First, the stiffness and damping matrices \mathbf{K}_C and \mathbf{C}_C are mapped from the Cartesian to the joint space using Equations (4) and (5). Assuming that the external forces and changes in Jacobian are negligible, only the first term in Equation (4) is used to map the stiffness matrix. After that, the redundancy or positive semi-definite property can be taken care of by removing the zero-potential-energy (ZP) mode of motion \mathbf{u}_0 using a constraint matrix \mathbf{S} and obtaining a system in a reduced space defined as \mathbf{q}'

$$E_p = \frac{1}{2} \mathbf{u}_0^T \mathbf{K} \mathbf{u}_0 = 0 \tag{6}$$

$$\mathbf{u}_0 \in N(\mathbf{K}) \Rightarrow \mathbf{K} \mathbf{u}_0 = 0 \tag{7}$$

$$\mathbf{q} = \mathbf{S} \mathbf{q}' \tag{8}$$

When we remove the r ZP mode(s) that belongs to the null space of the stiffness matrix (The ZP mode is analogous to the rigid body mode in theory of vibrations with no changes in the potential energy E_p of the system), Equation (8) can be substituted into Equation (3):

$$\mathbf{S}^T \mathbf{M}(\mathbf{q}) \mathbf{S} \ddot{\mathbf{q}}' + \mathbf{S}^T \mathbf{C} \mathbf{S} \dot{\mathbf{q}}' + \mathbf{S}^T \mathbf{K} \mathbf{S} \mathbf{q}' = \mathbf{S}^T \boldsymbol{\tau}_{ext} \tag{9}$$

$$\implies \mathbf{M}'(\mathbf{q}) \ddot{\mathbf{q}}' + \mathbf{C}' \dot{\mathbf{q}}' + \mathbf{K}' \mathbf{q}' = \mathbf{S}^T \boldsymbol{\tau}_{ext} \tag{10}$$

This reduced system in the \mathbf{q}' space is now positive definite and solvable for the $(n - r)$ non-zero potential energy (NZP) modes of motion of the system by establishing the state vector $\mathbf{z}(t) = [\mathbf{q}'(t)^T \dot{\mathbf{q}}'(t)^T]^T$ in the state space representation:

$$\dot{\mathbf{z}}(t) = \mathbf{A} \mathbf{z}(t) + \mathbf{B} \mathbf{S}^T \boldsymbol{\tau}_{ext}(t) \tag{11}$$

where the \mathbf{A} and \mathbf{B} matrices have dimensions of $2(n - r) \times 2(n - r)$ and $2(n - r) \times (n - r)$, respectively, and are given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -(\mathbf{M}')^{-1} \mathbf{K}' & -(\mathbf{M}')^{-1} \mathbf{C}' \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{0} \\ (\mathbf{M}')^{-1} \end{bmatrix} \tag{12}$$

This linear system has a solution that is mainly governed by the $(n - r)$ pairs of eigenvalues λ of the \mathbf{A} matrix, which are contained in the $\boldsymbol{\Lambda}$ matrix, which has a solution

$$\mathbf{z}(t) = \begin{bmatrix} \mathbf{q}'(t) \\ \dot{\mathbf{q}}'(t) \end{bmatrix} = \mathbf{X} e^{\boldsymbol{\Lambda} t} \mathbf{Y}^T \mathbf{z}(0) + \int_0^t \mathbf{X} e^{\boldsymbol{\Lambda} \tau} \mathbf{Y}^T \mathbf{B} \boldsymbol{\tau}_{ext}(t - \tau) d\tau \tag{13}$$

where \mathbf{X} and \mathbf{Y} are the right and left eigenvectors of the system. The corresponding damping ratios ζ in the modal space of the \mathbf{A} matrix can be computed and systematically improved:

$$\lambda_i = a \pm ib; \quad \omega_{di} = b; \quad \zeta_i = \cos(\gamma); \quad \gamma = \tan^{-1}(b/a) \tag{14}$$

After the solution is obtained in the modal space, this NZP solution can be mapped back into the physical space using the constraint matrix in Equation (8). Because the dynamic equation of impedance control of the robot can be solved analytically, the trend or behavior of the system can be obtained, as the elements of the damping matrix are iteratively changed or updated. A parameter study can be generated for all diagonal and off-diagonal elements of \mathbf{C}_C , allowing the dynamic response of the system to be modulated through damping in a consistent and coherent manner.

As can be seen, this analysis in the joint space of the robot must be performed in order to modulate the dynamic response. This becomes an iterative process of determining the elements of the damping matrix that affect the most each mode of motion (vibration) as the robot moves and interacts with the environment. It is worth pointing out that, the methodology explained here can be applied in higher dimensions as well as in case of more redundant DoFs for robotic tasks; this becomes a matter of handling the redundancies by successively imposing constraint matrices (\mathbf{S}) and reducing the system until it becomes positive definite.

Using machine learning tools and data collected applying the proposed methodology with modal analysis, a function that computes the optimum damping ratio can be obtained.

5. Experimental Setup and Data Collection

For a redundant robotic manipulator performing Cartesian impedance control, the analytical tool previously described is used to find an optimal damping matrix that generates the best or highest possible damping ratio of the dominant NZP (oscillatory) mode of motion having the least number of damping parameters tuned.

This work focuses on the case of a 3R planar robot, for which the set of optimal damping elements is obtained considering different paths and Cartesian stiffness matrices. Because there is no inertia reshaping, the configuration-dependent mass matrix is obtained directly at the joint space, and because for an intended task there is a Cartesian stiffness matrix associated with it, a damping matrix needs to be obtained. The were data collected for each point of a given path and then fed into the following algorithm:L

$$\mathbf{M}(\mathbf{q}) = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

$$\mathbf{K}_C = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$$

$$\mathbf{C}_C = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

It is worth pointing out that the damping matrix was chosen after performing the analysis described in the previous Section iteratively in order to establish a behavior and find the most suitable damping values. An experimental test consisted of finding the damping matrix elements for all the points or robot configurations in a given path. The duration of each experimental test for an intended path was 1993.74 s, with a sampling frequency of about 0.053 Hz. As a consequence, 105 samples (optimal damping values in a path) were obtained for each experiment. In other words, it takes 18.988 s to generate one sample. Table 1 shows a description of the trials carried out for each path type in the workspace of the robot. For the circular paths, the radii values were between 0.2 and 1.15 m, for the square paths the sides lengths were between 0.3 and 1.626 m, and for the straight line paths the lengths were between 0.8 and 2 m.

Table 1. Experiments performed for each type of path: circle, square, and straight line.

Path	Total Experiments	Total Samples
Circle	54	6195
Square	54	6232
Straight Line	54	5448
Total	162	17,875

To better understand the mechanics of a 3R planar robot, Figure 2 illustrates. The motion of this robot is determined by its three revolute joints; their angular positions are described by q_1 , q_2 , and q_3 . For this theoretical robot, the range of angular movement is considered to be infinite in all its joints, meaning that for any $i \in (1, 2, 3)$, $-\infty < q_i < \infty$.

In this way, the revolute joints do not impose any restriction on the robot's workspace. On the other hand, the link lengths restrict the robot's reach. This study considers $L_1 = 0.5$ m, $L_2 = 0.35$ m, and $L_3 = 0.3$ m, with corresponding link masses of $m_1 = 4$ kg, $m_2 = 3$ kg, and $m_3 = 2.5$ kg.

Because a redundant 3R planar robot is being considered, the number of joint space variables should be greater than the number of controlled task space variables. In Figure 2, q_i are the three joint space variables and x_e , y_e , and ϕ are the task space variables. No access to control ϕ is assumed in order to impose redundancy on the model. This means that the size of the manipulator Jacobian is 2×3 . Consequently, the mapping from the Cartesian to the joint space provides a positive semi-definite system with singular \mathbf{K} and \mathbf{C} matrices, which is not solvable unless the system is reduced to a positive definite one using the methodology previously described in this paper.

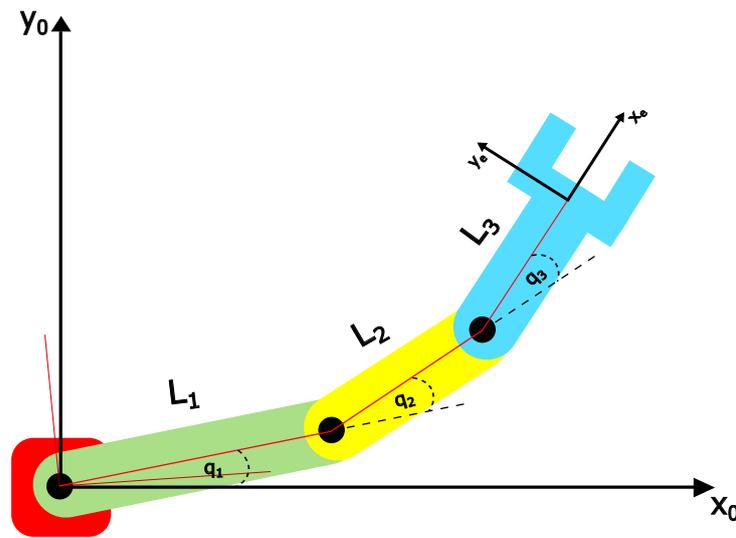


Figure 2. 3R planar robot on which this study is based. In our research, $L_1 = 0.5$ m, $L_2 = 0.35$ m, and $L_3 = 0.3$ m, while the joint space variables q_1 , q_2 , and q_3 have an infinite range of angular movement.

Local Optimum Damping Based on Modal Analysis

Before extending to a path, it is necessary to explain how to obtain the optimal damping values for a given point or joint configuration, after which generalizing it for paths is a matter of applying the algorithm to all the points that are part of the pre-established path.

Figure 3 describes the process of obtaining an optimum damping value for a configuration of a redundant 3R planar robot. For greater comprehension, we explain this by an example in which the end effector is located at $\mathbf{x} = [x, y]^T = [0.5 \text{ m}, 0.6 \text{ m}]^T$.

The input data are provided by the joint configuration vector \mathbf{q} , which corresponds to the Cartesian \mathbf{x} , the inertia mass matrix $\mathbf{M}(\mathbf{q})$, a desired stiffness matrix in the Cartesian space \mathbf{K}_C , and the configuration-dependent Jacobian matrix of the robot $\mathbf{J}(\mathbf{q})$. All of these data except for the Cartesian stiffness matrix depend on the current robot configuration. The joint configuration vector is obtained by applying inverse kinematics to \mathbf{x} , $\mathbf{q} = [q_1, q_2, q_3]^T = [1.659, -1.979, 1.105]^T$ rad. Considering this configuration,

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} 2.013 & 0.4634 & 0.2542 \\ 0.4634 & 0.6215 & 0.1339 \\ 0.2542 & 0.1339 & 0.0750 \end{bmatrix}$$

$$\mathbf{J}(\mathbf{q}) = \begin{bmatrix} -0.6000 & -0.1020 & -0.2121 \\ 0.5000 & 0.5443 & 0.2121 \end{bmatrix}$$

$$\mathbf{K}_C = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$$

Then, mapping the Cartesian stiffness matrix to the joint space using Equation (4) and assuming negligible changes in Jacobian, the joint stiffness matrix \mathbf{K} is obtained:

$$\mathbf{K} = \begin{bmatrix} 61.00 & 33.33 & 23.33 \\ 33.33 & 30.67 & 13.71 \\ 23.33 & 13.71 & 9.000 \end{bmatrix}$$

As expected from theory, due to the redundancy this joint stiffness matrix is positive semi-definite. In order to remove the ZP mode, a state vector in the null space of \mathbf{K} must be

found and normalized with respect to $\mathbf{M}(\mathbf{q})$ such that $\mathbf{u}_0^T \mathbf{M} \mathbf{u}_0 = 1$. For this illustrative example, the normalized vector is

$$\mathbf{u}_0 = \begin{bmatrix} -0.9013 \\ -0.2037 \\ 2.647 \end{bmatrix}$$

When the constraint matrix \mathbf{S} is imposed to the system such that $\mathbf{q} = \mathbf{S} \mathbf{q}'$, the redundancies are removed. This means that \mathbf{K} and \mathbf{M} are reduced to \mathbf{K}' and \mathbf{M}' through Equations (8)–(10):

$$\mathbf{S} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -21.38 & -3.284 \end{bmatrix}$$

$$\mathbf{K}' = \begin{bmatrix} 3178 & 295.5 \\ 295.5 & 37.7 \end{bmatrix}$$

$$\mathbf{M}' = \begin{bmatrix} 25.43 & 2.032 \\ 2.032 & 0.5510 \end{bmatrix}$$

From this point on, a parametric study is carried out with the purpose of analyzing the influence of the damping matrix on the damping ratios of the system. The damping ratios affect the dynamic response of the robot under impedance control, and are very important. To accomplish this parameter study, one element from the main diagonal of the damping matrix is iteratively increased while the other one remains with a low value ($0.1 \frac{Ns}{m}$) in order to obtain the effect of the element in study over the damping ratios. Please note that it is important to maintain the positive definite property of the damping matrix. A step of 0.001 was used for the iterations, beginning with a $\mathbf{C}_c = \text{diag}(0.1, 0.1) \frac{Ns}{m}$ for the first iteration ($j = 1$) in both parametric studies. Note that for the first iterations, the results of both parameterizations are very alike, as the values are very low.

For both the c_{11} and c_{22} parametric studies, the iteration $j = 40,000$ is shown as an example in order to clarify the procedure.

For the first parametric study (c_{11}), we have

$$\mathbf{C}_c = \begin{bmatrix} 40.09 & 0 \\ 0 & 0.1 \end{bmatrix}$$

Mapping the Cartesian damping matrix onto the joint space with the help of Equation (5), we have

$$\mathbf{C} = \begin{bmatrix} 14.46 & 2.481 & 5.114 \\ 2.481 & 0.4466 & 0.8789 \\ 5.114 & 0.8789 & 1.809 \end{bmatrix}$$

Reducing \mathbf{C} , we have

$$\mathbf{C}' = \mathbf{S}^T \mathbf{C}_c \mathbf{S}$$

$$\mathbf{C}' = \begin{bmatrix} 622.7 & 93.91 \\ 93.91 & 14.18 \end{bmatrix}$$

The next step is to obtain the system matrix (\mathbf{A}) from the space state representation, as described in Equation (12):

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -116.4 & -8.726 & -15.41 & -2.319 \\ -106.9 & -36.19 & -113.6 & -17.18 \end{bmatrix}$$

To obtain the damping ratios, the eigenvalues of the **A** matrix are needed, and can be obtained using the equations in (14):

$$\zeta_{1,2} = 0.0782; \zeta_{3,4} = 1$$

For the second parametric study (element c_{22}), the same procedure is followed:

$$C_c = \begin{bmatrix} 0.1 & 0 \\ 0 & 40.09 \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -116.4 & -8.726 & -31.38 & -1.188 \\ -106.9 & -36.19 & 70.59 & 2.632 \end{bmatrix}$$

$$\zeta_{1,2} = 0.0975; \zeta_{3,4} = 1$$

For the complete parametric study at this robot configuration, 49,901 iterations have been considered; the whole analysis results are illustrated in Figure 4.

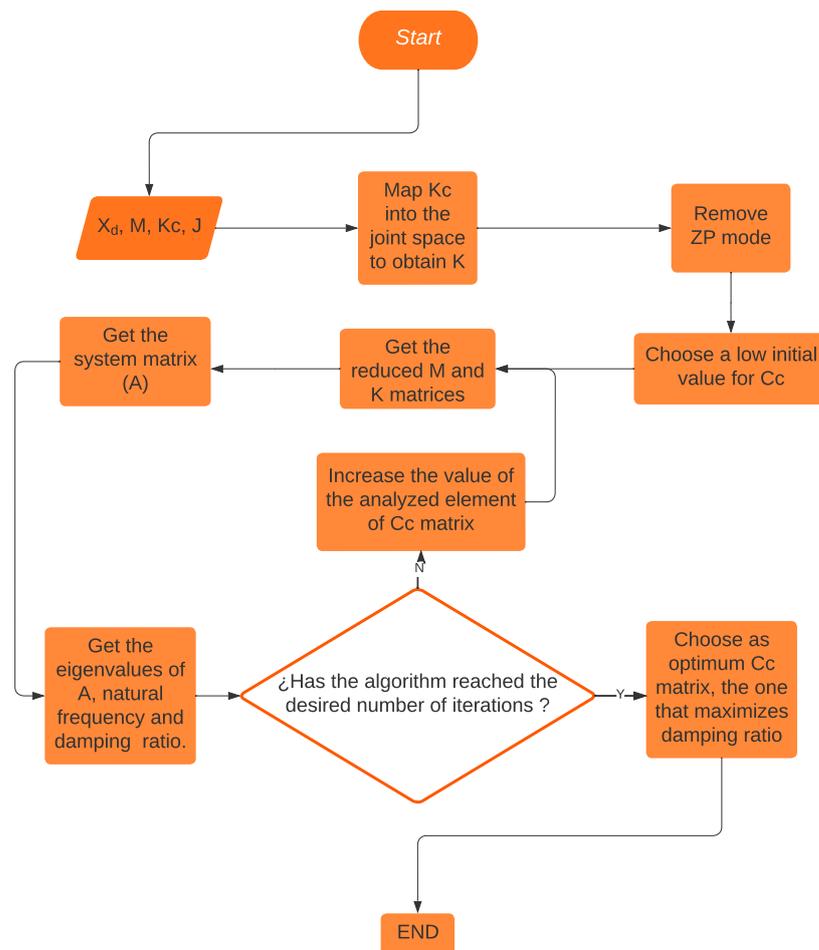
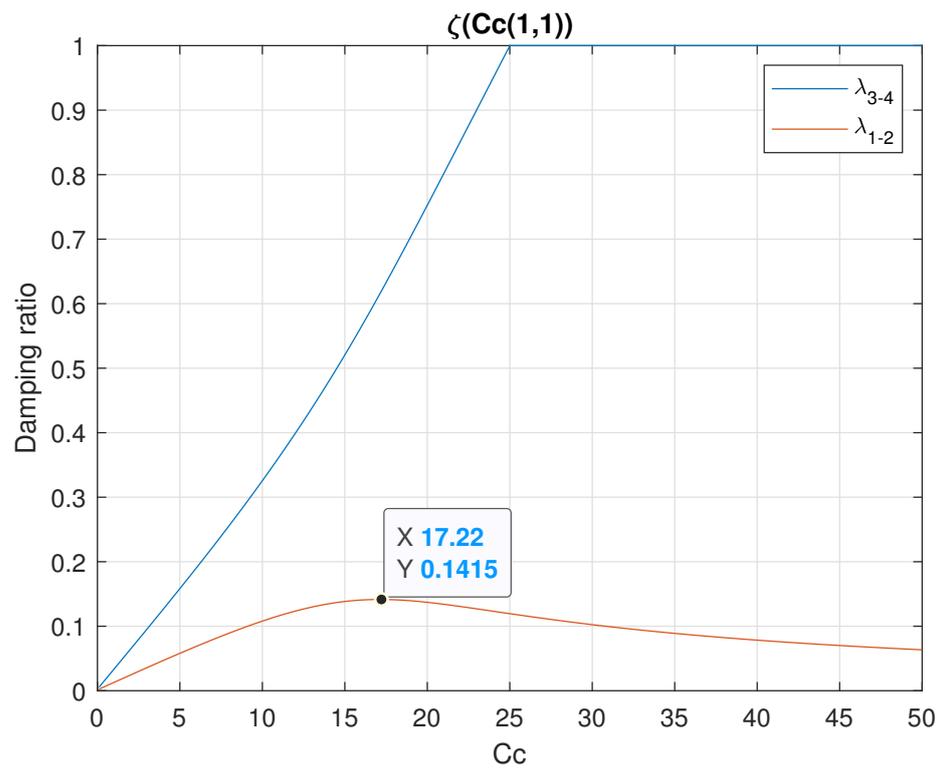
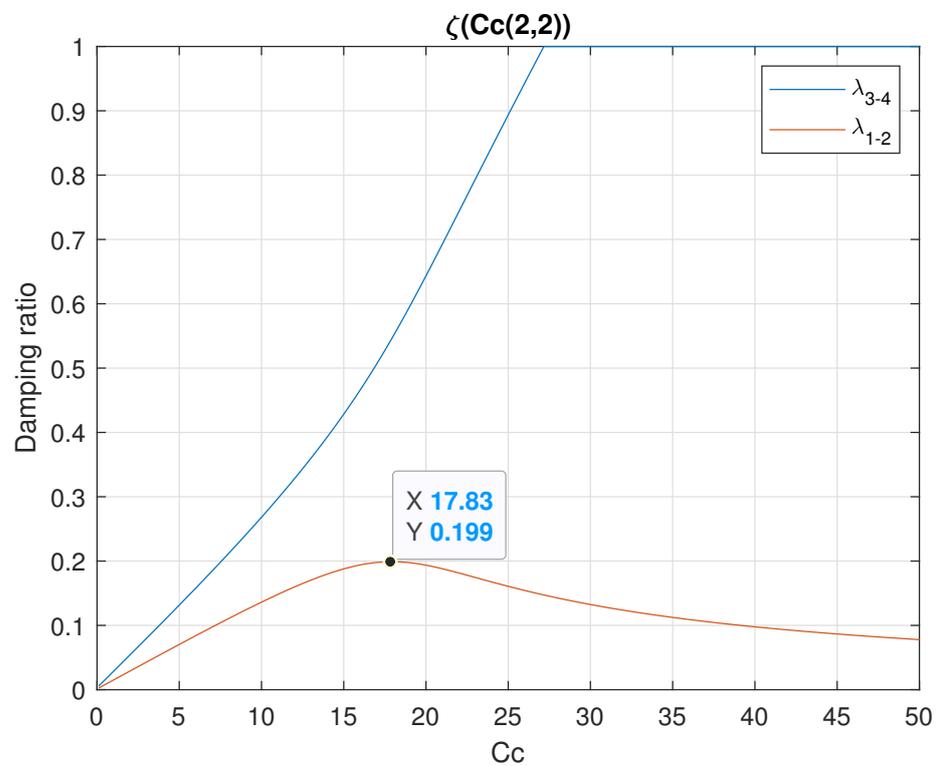


Figure 3. The algorithm used to select Cartesian damping matrices with an associated damping ratio. The algorithm increases a certain element from the diagonal of C_c and finds the damping ratios related to the eigenvalues of the system matrix, which is the matrix from the state space representation that includes the input variables mapped onto the joint space after reducing them using the removed ZP mode(s). This is performed iteratively until an optimum value is identified.



(a) Parametric study of element c_{11} .



(b) Parametric study of element c_{22} .

Figure 4. Plots showing the results of the parametric study of damping based on variation of the main diagonal elements of C_c (1,1) and (2,2). Because the parametric study of element c_{22} has more influence on λ_{1-2} than element c_{11} , the optimum value is $c_{22} = 17.83$, which generates a set of damping ratios $\zeta_{1,2} = 0.2$ and $\zeta_{3,4} = 0.55$.

From observation and experimentation, it became understood that the most critical mode is λ_{1-2} , as it is not significantly influenced by any of the elements of the damping matrix. The element with the greatest influence on mode 1-2, comparing Figure 4a,b, is c_{22} , and the higher damping ratio values for this mode were obtained with the variation of this element. Because the damping ratios for mode 1-2 have a unique maximum, the optimum values for the elements of the damping matrix can be chosen. Thus, for this configuration, the optimum damping matrix is

$$\mathbf{C}_c = \begin{bmatrix} 0.1000 & 0 \\ 0 & 17.83 \end{bmatrix}$$

Leading to the following damping ratios:

$$\zeta_{1,2} = 0.2; \zeta_{3,4} = 0.55$$

This analysis for one robot configuration corresponding to a point in the intended path is then extended for each configuration along the path.

6. Methodology

In the creation of data-driven machine learning models, the adage “garbage in, garbage out” is applicable [28]. In actuality, real-world data are frequently erroneous and deficient in specific behaviors or patterns, and are frequently inconsistent and incomplete. Data preprocessing is therefore essential for overcoming the aforementioned problems and for preparing the data to create data models [29]. Data cleansing, normalization, feature discovery (i.e., feature extraction, feature selection, and feature learning), and unbalanced data management are often included in data preprocessing activities.

In this section, all the steps used to preprocess the data are described, including splitting of the data into training, validation, and testing sets as well as several preliminary considerations regarding the data, feature selection and redundant variable removal, and data normalization using scaling strategies.

Then, the basis of the five proposed ML algorithms are addressed. Likewise, the details of the ML algorithm-based architectures are described. Finally, the technique employed for hyperparameter tuning (grid search plus CV) is outlined.

6.1. Data Split

Typically, the full dataset (see Figure 5) can be divided into two or more subsets, i.e., for training, validation, and testing. Training and testing datasets are the most common in ML models; however, a validation dataset is frequently required as well. Training sets are usually employed to estimate the parameters of the model and fit the data. After that, a test set is used to evaluate the model and measure its performance. When a third set is used (validation), it is commonly intended to change or calibrate the hyperparameters of the model. If only the training and testing sets are used, another strategy, such as cross-validation (CV), can be applied to calibrate the model, and a third dataset is not required. In this work, only two datasets, training and testing, are used. Moreover, during testing a CV strategy is applied to calibrate the model.

There is no fixed guideline for splitting data. There are several strategies to perform the task, such as random sampling, stratified random sampling, or non-random sampling. Considering that our experiments are classified into three path types (square, straight line, and circle) it is important to ensure that the data are balanced (correctly distributed) with respect to those paths in the training and testing sets; thus, stratified random sampling is suitable.

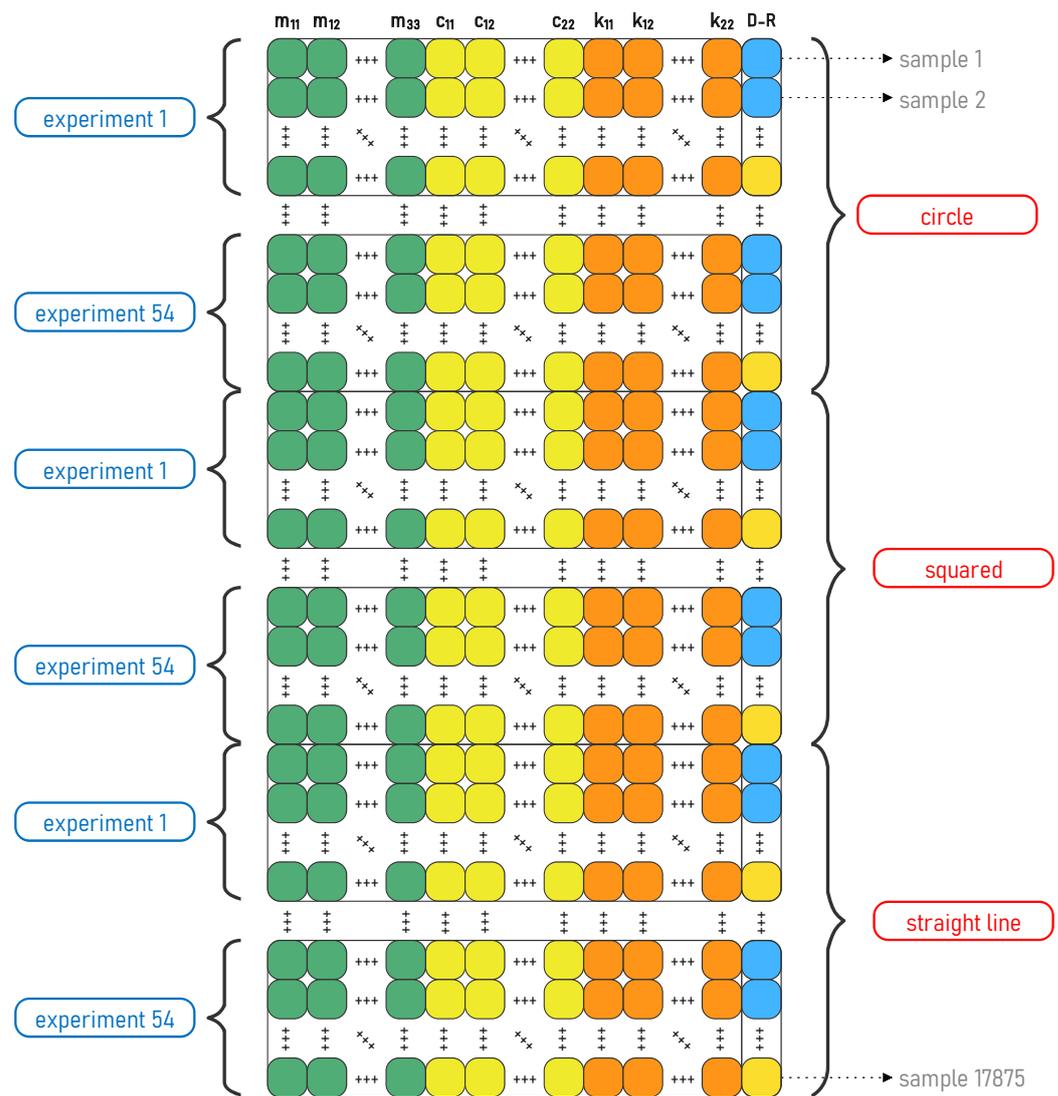


Figure 5. Structure of the full dataset.

From Table 1, the number of total samples for each path can be observed. Each path in the dataset is split into 80% for training and 20% for testing. Next, Table 2 shows the total samples corresponding to each dataset (training and testing). Taking the total samples for training (14,299) and testing (3576) and dividing by the total samples (17875), it can be confirmed that the total distribution is 80% and 20%, respectively, as expected.

Table 2. Total samples intended for training and testing.

Path	Training Samples	Testing Samples	Total Samples
Circle	4956	1239	6195
Square	4985	1247	6232
Straight Line	4358	1090	5448
Total Samples	14,299	3576	17,875

6.2. Preliminary Considerations

Based on Section 5, the input data are composed of the predictor variables (elements of matrices M, K, and C) and the target (damping-ratio). Regarding the predictors, as they are elements of matrices there are several considerations to be taken into account. For instance, the K matrix is joint-form, which implies that it is a diagonal matrix. Thus, all the entries

outside the main diagonal (drawn from left to right) are zero, i.e., the elements k_{12} and k_{21} . A similar case is the C matrix; as it is a diagonal matrix, its elements c_{12} and c_{21} are zero. In consequence, as those four elements are always zero they do not provide useful information to the input data, and must be removed.

To ensure that the rest of features (elements of matrices) can provide useful information to the input data, the standard deviation was computed for each feature in order to measure how many features vary (a measure of variance) throughout the dataset. If a feature does not vary, it is unnecessary and should be removed. Below, Table 3 shows the standard deviation computed on the predictors.

Table 3. Standard deviation (STD) of predictor variables into the full dataset.

M	m_{11}	m_{12}	m_{13}	m_{21}	m_{22}	m_{23}	m_{31}	m_{32}	m_{33}
STD	0.991	0.548	0.133	0.548	0.178	0.089	0.133	0.089	1.994×10^{-14}
C	c_{11}	c_{22}							
STD	23.973	23.684							
K	k_{11}								
STD	272.354								

From Table 3 it can be seen that the elements from matrices **K** and **C** have quite a significant deviation, unlike matrix **M**, in which there are elements with low but still significant deviation (i.e., with at least two significant decimal values). However, the element m_{33} has a standard deviation equal to 1.994×10^{-14} , which is completely negligible. This element must be removed, as it does not vary at all throughout the dataset.

In summary, with these preliminary considerations several elements of the matrices have been removed. The remaining eleven parameters include eight for **M**, two for **C**, and one for **K**.

6.3. Feature Selection

The process of choosing a subset of pertinent features to be used in model creation is known as feature selection, or sometimes as variable selection or attribute selection. Feature selection is usually carried out for several purposes. First, it can enhance a machine-learning algorithm’s performance [28]. For instance, certain features could be noise, or may not be important for a specific problem. These features contribute to overfitting, which can lead to biased or undesirable variation in the problem result. Second, feature selection is carried out to increase the model’s interpretability [28]. The model is made simpler when certain features are eliminated. In order to better understand which features contribute the most to the model, feature selection additionally rates the relevance of the features. Third, feature selection lowers the amount of resources needed for computation and data collection [28]. For instance, if sensor data are utilized as features, feature selection aids in reducing the number of sensors, lowering the cost of the sensor system, as well as the costs associated with data gathering, storage, and processing. Finally, feature selection works to lessen the possibility of the “curse of dimensionality,” similar to dimensionality reduction.

In this work, considering that the full dataset contains several experiments and each is performed under different execution conditions, there is not any guarantee that the predictor variables have a linear relationship, either between themselves or with the predicted or target variables. Under this scenario, a linear measure of correlation such as the Pearson correlation is not appropriate; instead, other techniques for measuring correlation that are able to capture nonlinear relationships should be employed, such as Spearman’s rank correlation, distance correlation (DC), mutual information (MI), ANOVA f-test, Chi-square test, and variance checking, among others. In this work, three different methods, Spearman, MI, and DC, are used to measure the feature–target (feature selection)

and feature–feature (redundant variables removal) correlation. Each method and its details are explained below.

6.3.1. Spearman’s Rank Correlation

Known as Spearman’s rho, this is a non-parametric measure of the association between two variables. In the context of this study, it can be used to measure the correlation between different features of the robot, such as its control parameters and damping ratio.

As is well known, Spearman’s correlation varies in a range from -1 to $+1$, where -1 is associated with the maximum negative relationship between two variables and $+1$ represents the maximum positive one [30]. Based on [31], an absolute correlation coefficients between 0.00 and 0.10 represents a negligible correlation, i.e., those predictor variables should be removed. Figure 6a shows the Spearman correlation matrix among the predictor variables and with respect to the variable to be predicted.

The last row (from top to bottom) contains the correlation coefficients between the predictor variables and the target (feature–target). It can be seen that these coefficients vary in a range of -0.42 to $+0.04$ (moderate, weak, and negligible correlations). In this sense, as there are no higher coefficients and not very many predictor variables, it is not suitable to remove them. Therefore, in Section 6.4 all of the features are evaluated in order to eliminate those that are redundant.

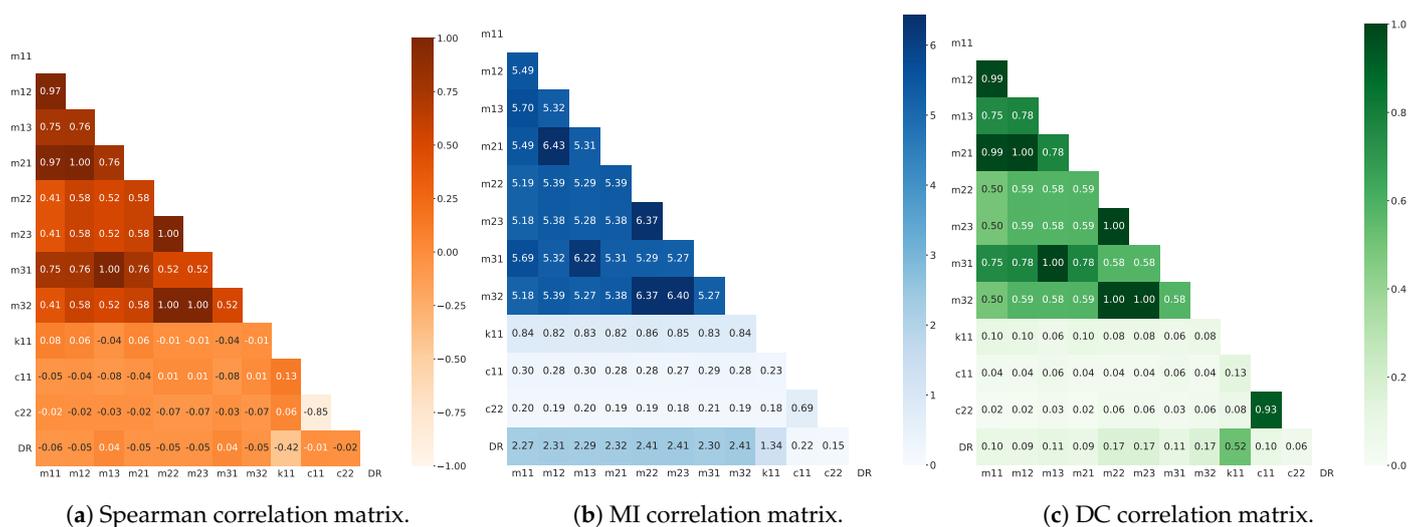


Figure 6. Feature–target and feature–feature correlation matrices computed with different techniques: (a) Spearman, (b) MI, and (c) DC.

6.3.2. MI Correlation

MI is a concept used in feature selection that measures the dependence between two variables [32,33]. In this work, MI can be used to measure the dependence between the input features of the robot, such as its control parameters, and the output feature, such as the damping ratio. The MI between two variables is a non-negative value, with larger values indicating stronger dependence between the variables.

Figure 6b shows the feature–feature and feature–target MI correlation matrix. As mentioned above, the last row from top to bottom contains the correlation coefficients between the features and the target. After observing these values, the variables with coefficients closer to 0 are eliminated, considering a cut-off value of 1.5; Considering this, the remained features are m_{11} , m_{12} , m_{13} , m_{21} , m_{22} , m_{23} , m_{31} , and m_{32} . This means that, in Section 6.4, those features are evaluated to determine which are redundant and should be removed.

6.3.3. DC

DC is a statistical measure that quantifies the dependence between two random variables. In feature selection, it can be used to measure the dependence between the input

features and output feature of a model [34]. DC has the advantage of being able to detect nonlinear relationships between variables, making it a useful tool for feature selection in situations where the relationships between features and the output are nonlinear [35].

The DC coefficient is a value between 0 and 1, where 0 indicates no dependence between the variables and 1 indicates perfect dependence. The coefficient is calculated as the square root of the ratio of the distance covariance and the product of the distance variances of the two variables.

Figure 6c shows the feature–feature and feature–target DC matrix. As in the previous cases, the values of the last row from top to bottom correspond to the feature–target correlation coefficients. Values between 0.00 and 0.10 represent a negligible correlation, i.e., those features should be removed. Considering this, the remaining features are m_{11} , m_{13} , m_{22} , m_{23} , m_{31} , m_{32} , k_{11} , and c_{11} . In Section 6.4, these features are evaluated to determine which are redundant and should be removed.

6.4. Redundant Variables Removal

Accuracy and elapsed time are two crucial factors to be considered when ML models are built, as it is desirable for the model to have excellent performance, i.e., the best accuracy and the least elapsed time for the training and testing stages. Feature quality is a key point to achieve this, as redundant variables (two or more predictors that explain the same information) result in poor model performance [36].

Regarding the Spearman and DC correlation, when the absolute correlation coefficient fluctuates from 0.90 to 1.00 it can be considered as a very strong correlation [31]; thus, if two predictor variables have a very strong correlation one of those must be removed to avoid redundant features. On the other hand, concerning MI correlation, if the correlation coefficients are less than 1.50 (cut-off value) it means that those variables are sufficiently independent of each other.

Table 4 summarizes the feature selection and redundant variables removal process carried out to select the predictor variables used in the next stage of the methodology. The “Feature Selection” column summarizes the discussion about the correlation of all the features against the target in order to determine which should be kept or removed. Thus, in this part, only the features that contain a (✓) are assessed in the next column, “Redundancy Analysis”, to determine whether they are redundant with another feature, while those that contain a (✗) symbol are excluded from the next analysis, as they are not strongly correlated with the target.

The column “Redundancy Analysis” recaps what was discussed in the previous paragraph about two features being considered redundant or not. Based on that, the (✓) symbol determines the non-redundant features that are kept, while the (✗) symbol points out the features that are redundant and should be removed; (***) indicates features that were not evaluated because they were excluded in the previous feature selection phase.

Finally, as three methods are selected to perform redundant analysis, a feature is kept only if all three, or at least two, provide a (✓) symbol to the feature; otherwise, it is not considered for integrating the predictor set and receives a (✗) symbol.

Therefore, after this thorough feature selection process, the remaining features from the initial dataset are

- M matrix: m_{11} , m_{13} , and m_{22}
- K matrix: k_{11}
- C matrix: c_{11}

Table 4. Feature selection and redundancy analysis for all the features from the full dataset, applying three different correlation measurement techniques; (✓) and (✗) determine whether or not the feature is taken into account after each analysis, respectively, while (***) marks the feature as excluded in that specific analysis.

Features	Feature Selection (Feature-Target)			Redundancy Analysis (Feature-Feature)			Final Votation
	Spearman	MI	DC	Spearman	MI	DC	
m_{11}	✓	✓	✓	✓	✓	✓	✓
m_{12}	✓	✓	✗	✗	✗	***	✗
m_{13}	✓	✓	✓	✓	✗	✓	✓
m_{21}	✓	✓	✗	✗	✗	***	✗
m_{22}	✓	✓	✓	✓	✗	✓	✓
m_{23}	✓	✓	✓	✗	✗	✗	✗
m_{31}	✓	✓	✓	✗	✗	✗	✗
m_{32}	✓	✓	✓	✗	✗	✗	✗
k_{11}	✓	✗	✓	✓	***	✓	✓
c_{11}	✓	✗	✓	✓	***	✓	✓
c_{22}	✓	✗	✗	✓	***	***	✗

6.5. Data Normalization

Predictors can come from different data sources, which can have varying order of magnitude. To accelerate training or improve a model’s generalization capability, a data scaling strategy can be applied [37]. There are several normalization or standardization techniques, including Min-Max, Standard, Robust, and L1, among others [29]. Based on our experiments’ nature (simulation), there are no outliers; thus, the Min-Max technique is applied, considering that this strategy is usually weak to the presence of outliers. Min-Max scales data in the [0, 1] range using a linear transformation based on the original data range. Thus, through this process it is ensured that data are scaled to the same proportions and in the same range for all features.

Next, as indicated at the beginning of this section, it is essential to recall the foundations of the proposed ML techniques as well as the details of their algorithm-based architectures.

6.6. XGBoost: eXtreme Gradient Boosting

This algorithm, which is applicable to both regression and classification, was first developed by Chen et al. [38]. It is an ensemble in which boosting properties are leveraged by aggregating new models to adjust errors in the performance of the existing models. This task is recursive until no noticeable improvements are detected, then the task stops. When boosting is combined with a gradient basis, the new models that are added can help to predict the residuals or errors of prior models. Finally, the models aggregated to make a final prediction. During this step, a gradient descent algorithm is employed to minimize the loss when new models are added. Notably, XGBoost won on 17 of the 29 machine learning tasks launched by Kaggle by 2015.

As with many other machine learning algorithms, XGBoost requires a careful tuning process to determine the optimal performance. Moreover, this algorithm has several hyperparameters that need to be tuned; for that reason, the tuning process needs to be performed appropriately. Strategies such as Grid Search can be used to search for optimum hyperparameter combinations.

6.7. RF: Random Forest

An RF is an ensemble of decision trees used to forecast the output from a set of predictors [39]. On a training dataset, decision trees are built to obtain the prediction results in either mode (in case of classification issues) or the mean prediction (in case of regression issues) of the individual trees [40]. During the training stage, a bootstrap sample function is employed to randomly split the dataset into homogeneous subsets. While a random subset

is used to train each decision tree, the rest of the subsets are taken to validate the tree and compute the model accuracy [40,41]. Moreover, similar to XGBoost, this algorithm requires tuning of the hyperparameters to achieve its best performance.

6.8. LightGBM

This is an open-source gradient-boosting framework that uses tree-based learning algorithms. It is designed to handle large-scale data and has been widely used in various applications, such as recommendation systems, click-through rate prediction, and machine learning competitions. LightGBM is particularly efficient at handling large datasets and high-dimensional data, as it uses a histogram-based algorithm to split the tree leaves, which reduces the complexity of the model and speeds up the training process [42].

Compared to XGBoost, while both are powerful gradient-boosting libraries that have been used in many machine learning tasks with good performance, LightGBM is generally faster, more memory efficient, and is better able to handle categorical features and missing values, making it more suitable for large-scale data.

6.9. CatBoost

This algorithm can be used for both classification and regression tasks. For regression problems, it uses a different loss function, such as mean squared error or mean absolute error, to optimize the model. It permits using other loss functions as well [43]. CatBoost is particularly suitable for datasets with a large number of categorical features and missing values, as it has built-in handling for both.

Compared to RF, CatBoost is generally more powerful and accurate for regression; however, it can be more complex and require more computational resources. Compared to XGBoost and LightGBM, CatBoost has better ability to handle categorical features and missing values, which can be beneficial when working with datasets with a high number of categorical features.

6.10. SVR: Support Vector Regressor

Support vector machine is a well-known ML algorithm to deal with data classification challenges where high-dimensional data and small training datasets are not an issue. However, computational resource demands can reach high levels if caution is not taken [44]. This algorithm is extremely well equipped to deal with regression issues.

6.11. Hyperparameter Tuning Process for Models

The hyperparameter tuning process can be performed on a grid or by random search. Random searching (non-discrete or parametric values) tends to consume a higher level of computational resources than grid search; thus, in this study we have employed grid search. In grid search, reasonable ranges (parametric values) must be initially proposed as the hyperparameter combinations are set and tested on this basis.

On the other hand, K-fold cross-validation is used to assess model performance during the tuning process, with cross-validation adopted as a strategy to avoid overfitting. For this work, $K = 5$ is taken as a popular fold-division number, i.e., the data are divided in five folds. Four-fold validation is typically intended for training, with the remaining data used for validation. After that, the data used for validation pass through to become part of the training folds, while one of the four folds initially used for training passes through to become the new validation fold. This iterative task executes until all folds have been used for validation. When this process is complete, the data are reshuffled and cross-validation is repeated; for our case, three repetitions were set.

The five proposed models each have several distinct hyperparameters that need to be tuned. Table 5 presents the most popular hyperparameters for each model and defines the parametric values used for cross-validation.

Table 5. Parametric values for hyperparameters of different ML algorithms. *** Not Applicable.

Hyper-Parameter	Algorithm				
	XGBoost	SVR	RF	LightGBM	CatBoost
colsample_bytree	[1, 0.75, 0.5, 0.25, 0.1]	***	***	***	***
learning_rate	[1, 0.75, 0.5, 0.3, 0.1]	***	***	[1, 0.75, 0.5, 0.3, 0.1]	[0.01, 0.05, 0.1]
max_depth	[5, 6, 10, 50, 100]	***	[5, 6, 10, 50, 100]	***	[4, 6, 8]
alpha	[0, 0.5, 1]	***	***	***	***
n_estimators	[10, 50, 100, 500, 1000]	***	[10, 50, 100, 500, 1000]	***	***
C	***	[1, 10, 50, 100, 500, 1000]	***	***	***
ccp_alpha	***	***	[0, 0.5, 1]	***	***
num_leaves	***	***	***	[20, 25, 30, 35, 40, 45]	***
min_data_in_leaf	***	***	***	[10, 15, 20, 25, 30, 35, 40, 45]	***
feature_fraction	***	***	***	[0.5, 0.6, 0.7, 0.8, 0.9, 1]	***
bagging_fraction	***	***	***	[0.5, 0.6, 0.7, 0.8, 0.9, 1]	***
iterations	***	***	***	***	[30, 50, 100]
l2_leaf_reg	***	***	***	***	[1, 3, 5, 7, 9]

6.12. Feature Importance

This process refers to the relative importance of different features in a dataset for a given model. It is a measure of the impact that each feature has on the model’s predictions. There are a variety of feature importance measures, such as the permutation-based importance (PbI), mean decrease impurity (MDI), and mean decrease accuracy (MDA), that can be used to determine the importance of each feature. Both this method and feature selection are important steps in the process of building machine learning models, and can help to both improve the performance of the model and to understand the relationships between the features and the output.

In this study, the MDI (or Gini importance) [45,46] and PbI [47] are used to perform this task. On the one hand, the Gini importance looks at how a random forest is constructed and measures how each feature decreases the impurity of the split; the feature with the highest decrease is selected for the internal node. For each feature, it can describe the average decrease in the impurity. The average over all the trees in the forest is the measure of the feature importance. On the other hand, PbI, apart from analyzing feature importance, is employed to overcome the drawbacks of default feature importance computed with MDI. PbI randomly shuffles each feature and computes the change in the model’s performance. The features which impact the performance the most are the most essential ones.

In this work, RF is the algorithm employed to determine feature importance, following training and tuning using a five-variable predictor set. Note that after choosing the features with the highest impact for the RF model, it should be re-trained and re-tuned for predicting damping ratios. The other models are trained only once, and perform the prediction using the most important features.

Figure 7 shows the feature importance analysis using the two above-mentioned methods, MDI and PbI. Gini importance analysis (see Figure 7a) performed over the variables, as mentioned in Section 6.4, clearly shows that m_{11} and m_{13} are the features with the least impact on the model, and should be removed. This idea is supported by PbI analysis (see Figure 7b), where the same two variables have the least impact on the model. In conclusion, the predictor set is finally reduced to three variables: k_{11} , c_{11} , and m_{22} .

As a final point, Figure 8 shows the flow of the entire applied methodology and summarizes all the steps described in Section 6.

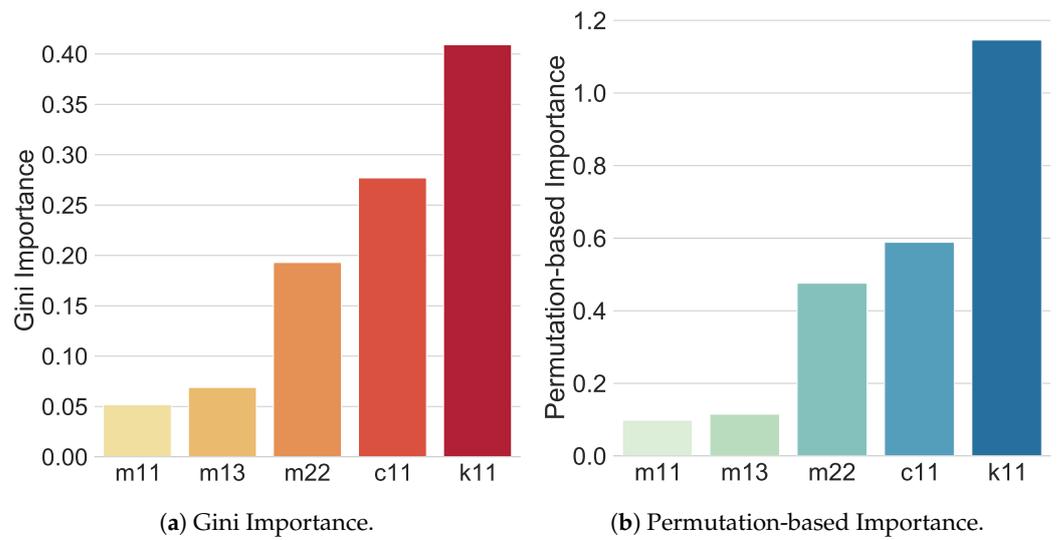


Figure 7. Feature importance analysis based on MDI and PbI methods; the results of both methods agree that the most important features for an RF model are k_{11} , c_{11} , and m_{22} .

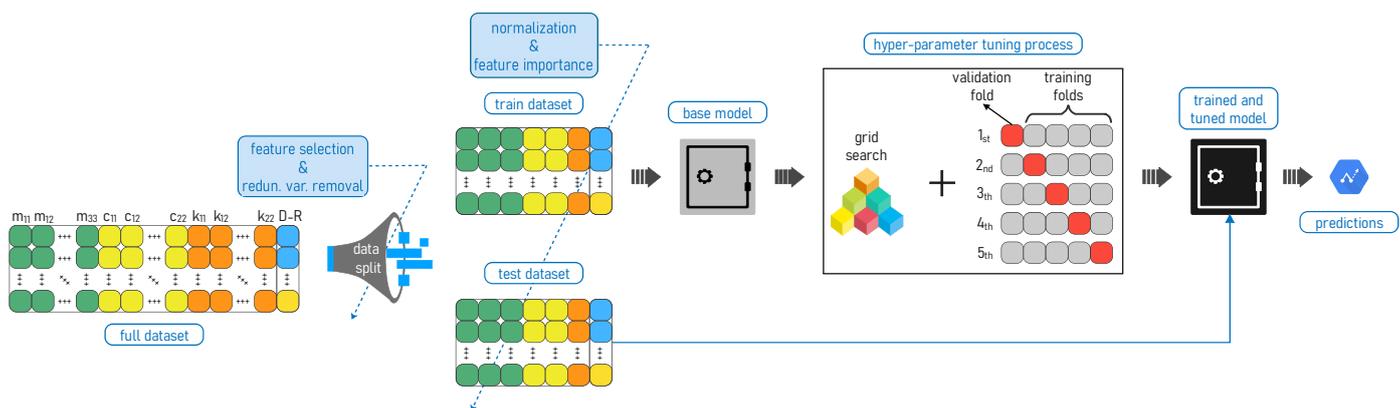


Figure 8. Flowchart of the proposed data processing methodology.

7. Results and Discussion

Through K-fold cross-validation, each one of the five stated algorithms was trained and validated to obtain the best possible combination of hyperparameters allowing for the highest accuracy (R^2 score) in predictions for optimum damping ratios. Thus, Table 6 shows the most suitable hyperparameters that could be found for the LightGBM, CatBoost, XGBoost, SVR, and RF models.

With the chosen hyperparameters, each model was retrained and retested. At this point, it should be noted that one-way testing was usually performed, with no splitting or cross-validation. When evaluating model performance on the test dataset, the model may suffer from overfitting to the test data. Using cross-validation, the model’s performance is estimated on different subsets of the data, leading to a more unbiased estimate of the model’s performance. Furthermore, cross-validation provides a more reliable estimate of the model’s performance by averaging the performance over multiple subsets of the data. This is especially useful when the sample size is small or the data are unbalanced, in which case a single training–testing split may not be representative of the model’s performance. In this work, testing is carried out using cross-validation.

Table 6. Chosen values for hyperparameters of the different ML models through the K-fold cross-validation method. *** Not Applicable.

Hyper-Parameter	Algorithm				
	XGBoost	SVR	RF	LightGBM	CatBoost
colsample_bytree	1.00	***	***	***	***
learning_rate	0.10	***	***	0.51	0.01
max_depth	6	***	100	***	4
alpha	0	***	***	***	***
n_estimators	1000	***	1000	***	***
C	***	1000	***	***	***
ccp_alpha	***	***	0	***	***
num_leaves	***	***	***	45	***
min_data_in_leaf	***	***	***	45	***
feature_fraction	***	***	***	0.83	***
bagging_fraction	***	***	***	0.50	***
iterations	***	***	***	***	30
l2_leaf_reg	***	***	***	***	9

Classic model assessment characteristics to measure performance are the training and testing time, an error metric such as the root mean square error or RMSE, and the training and testing accuracy (R^2 score). Table 7 shows these characteristics in order to assess and discuss the models' performance.

Table 7. Evaluation of the models based on cross-validation, accuracy, RMSE, and elapsed time.

Algorithm	K-Fold	Accuracy (%)		RMSE		Elapsed Time (s)	
		Train	Test	Train	Test	Train	Test
XGBoost	1	82.32	78.94	0.1187	0.1287	18.25	0.05
	2	86.14	85.32	0.0978	0.1033		
	3	96.99	90.90	0.0349	0.0632		
	4	92.38	87.06	0.0679	0.0879		
	5	92.97	89.11	0.0670	0.0849		
	Average	90.17	86.27	0.0773	0.0936		
RF	1	81.89	75.83	0.1202	0.1379	57.00	1.11
	2	85.92	84.67	0.0986	0.1056		
	3	97.64	84.71	0.0309	0.0819		
	4	92.74	84.20	0.0663	0.0971		
	5	93.50	86.12	0.0644	0.0958		
	Average	90.34	83.11	0.0761	0.1036		
LightGBM	1	76.83	75.00	0.1359	0.1402	0.55	0.01
	2	80.50	80.16	0.1161	0.1201		
	3	87.22	84.61	0.0719	0.0821		
	4	77.98	76.60	0.1155	0.1182		
	5	83.11	82.40	0.1038	0.1079		
	Average	81.13	79.75	0.1087	0.1137		
SVR	1	38.13	38.52	0.2221	0.2199	96.46	1.39
	2	47.98	52.05	0.1896	0.1867		
	3	73.60	70.40	0.1034	0.1139		
	4	67.74	66.73	0.1398	0.1409		
	5	66.79	71.25	0.1456	0.1379		
	Average	58.85	59.79	0.1601	0.1599		
CatBoost	1	20.67	20.64	0.2601	0.2498	1.65	0.01
	2	22.40	23.67	0.2515	0.2356		
	3	18.05	19.75	0.2316	0.1876		
	4	19.03	19.21	0.1822	0.2196		
	5	18.62	19.25	0.2215	0.2312		
	Average	19.75	20.50	0.2229	0.2248		

Comparing the five models, it is noticeable that both the XGBoost and RF models have the highest average accuracy in the tests (86.27% and 83.11%, respectively), and outperform the LightGBM, SVR, and CatBoost models. During training, the average accuracy of XGBoost is quite similar to that of RF. In terms of RMSE, XGBoost has the lowest value in testing. Thus far, considering these characteristics, XGBoost has the best performance. However, in terms of the elapsed time needed to train or test a model, LightGBM is much faster than XGBoost. This is a great advantage if a model is intended to be deployed into production, as in the ML lifecycle the control time is quite critical.

As can be seen, CatBoost performs poorly, reaching an average accuracy of 20.50% in testing and 19.75% in training. The reasons for this could be due to a number of aspects. On the one hand, this model is probably too complex for the given dataset, memorizing the training data instead of generalizing to new data. Another possibility is that the model is too simple for the given dataset, and is not able to capture the underlying patterns in the data.

It is always important to visualize the results in graphics; as such, Figure 9 shows the scatter plots for the different tested models, with real datapoints plotted against the predicted values. Following this idea, it is expected that better prediction will result in datapoints being plotted closer to the straight curve, with a slope equal to 1. The XGBoost model meets almost all the conditions to be the model with the highest performance, followed by the RF model, then the LightGBM model, and finally the SVR model. In terms of accuracy, the XGBoost and RF models have almost the same percentage; Figure 9a,c shows that most of datapoints are nearer to the straight curve. However, Figure 9d shows that many datapoints are far from the straight curve, indicating the low accuracy and performance of the model. It should be noted that the scatter plot figure for the CatBoost model is not shown due to its very poor performance.

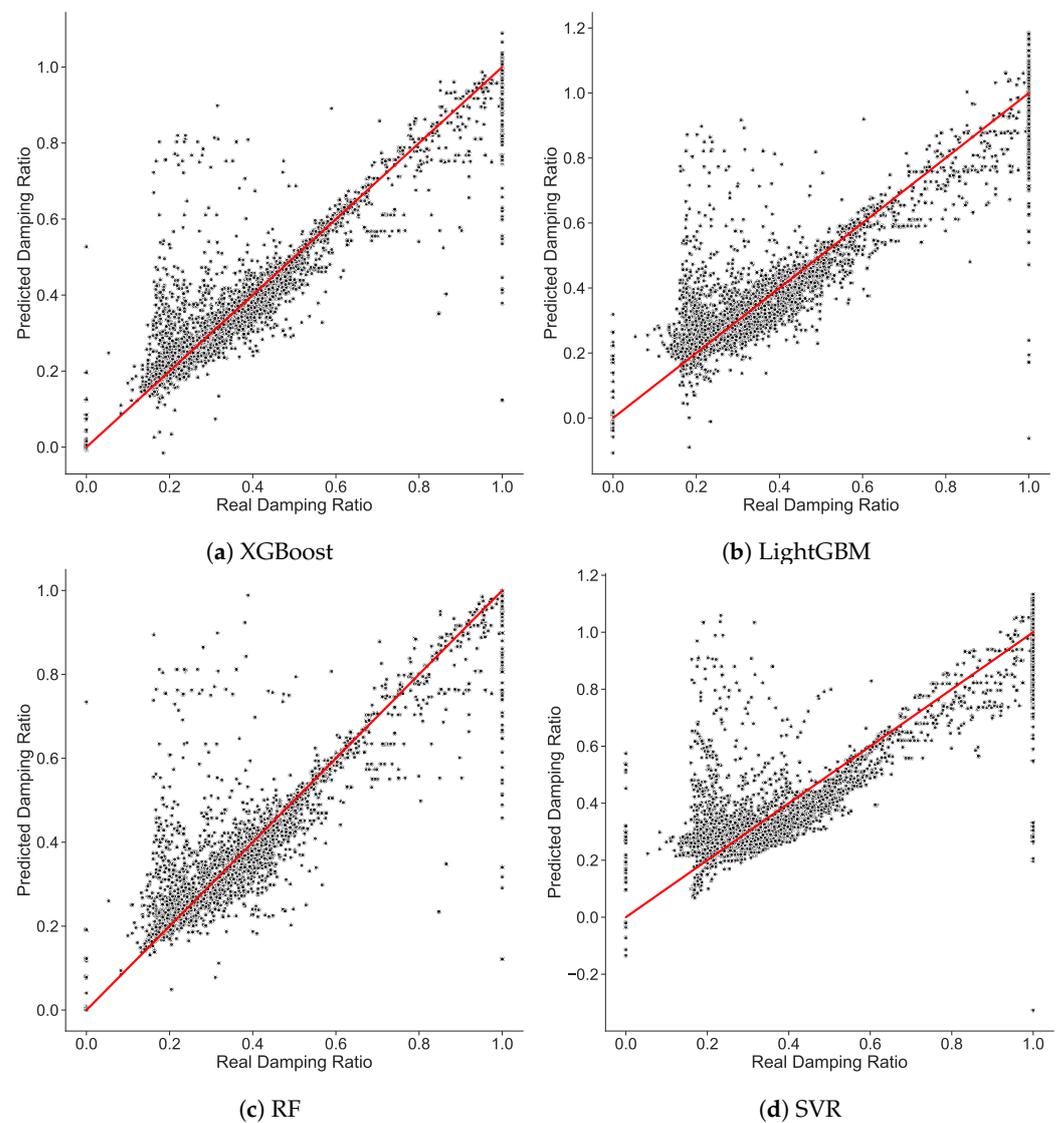


Figure 9. Plot of actual test data values of the damping ratio against their predicted values for (a) XGBoost, (b) LightGBM, (c) RF, and (d) SVR models.

On the other hand, recalling that the dataset was built with simulations of three different path types, it is important to analyze how the prediction of the data points associated with each path perform. Based on this, the XGBoost model has the best performance, Figure 10 highlights the location distribution of the datapoints associated with each different path type.

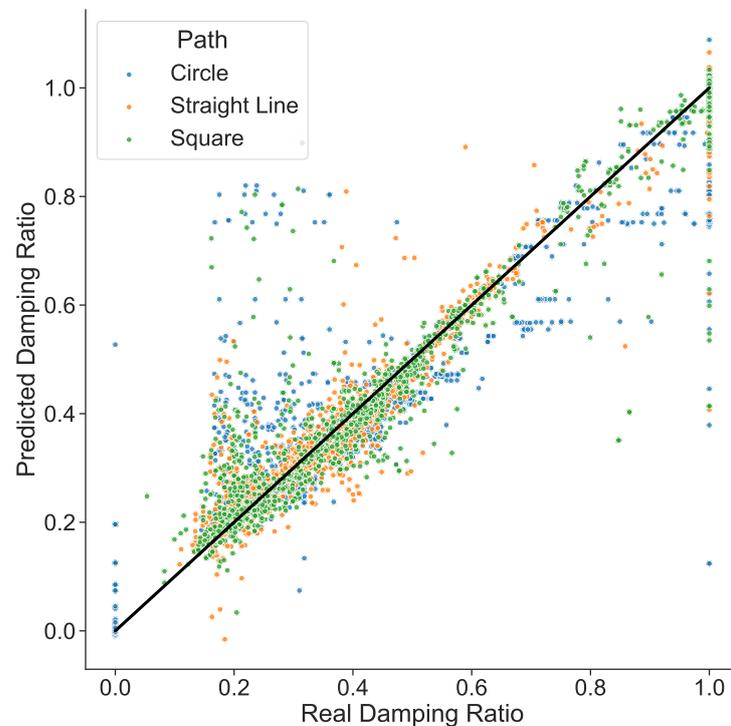


Figure 10. Plot of actual damping ratio test data values against their predicted values for each different path type when using the extreme gradient boosting model.

Considering each path, it is possible to observe the trend of the prediction results. Essentially, the straight line and square paths have the best predictions performance, while the circle has a larger error. This behavior is corroborated by Figure 10, where the furthest data points of the trend line are the blue ones that belong to the circle path, while the nearest ones belong to the other available paths.

In addition, as can be seen in Figure 10, there are outliers at the extreme real damping ratio equal to one. This shows that the model is better at predicting certain damping ratio values than others. For this reason, we decided to carry out an analysis of sample frequency distribution for each value of the damping ratio in the training dataset. Despite the fact that, as shown in Table 2, the dataset is relatively balanced in terms of the number of samples for each type of trajectory, it can be seen in Figure 11 that this is not the case for the number of samples for each value of the damping ratio.

It should be remembered that data preprocessing (feature selection and redundant variables removal) was carried out considering this imbalance of samples in terms of damping ratio values. This could have led to the elimination of variables that would have helped the model to predict whether they correspond to one damping ratio or another. For this reason, as mentioned in the next section, in future work it is important to ensure a balanced dataset in terms of the number of samples per value of the damping factor.

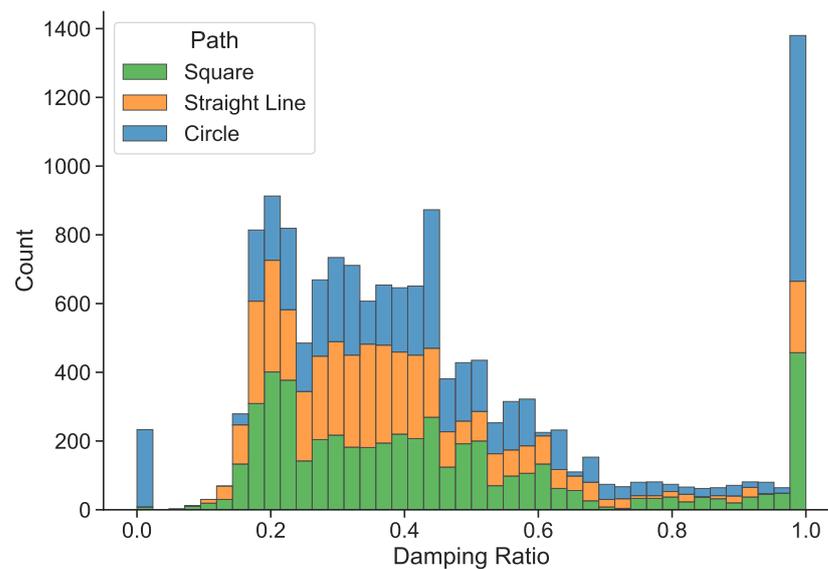


Figure 11. Damping ratio frequency distribution considering the path (robot trajectory). The distribution is similar to a normal distribution, with particular peaks at 0 and 1 values.

8. Conclusions and Future Work

This study has shown, as expected, that different configurations of a robotic manipulator result in different dynamic responses as the robot moves along a given path. To maintain a dynamic response as close as possible to the desired level, joint space analysis with corresponding damping value updates must be performed and implemented. A parameter study based on the modal analysis and optimization for damping ratio values is computationally expensive, especially when the robot includes more DoFs or task coordinates that need to be analyzed.

This study proposes and validates an innovative and advanced damping factor prediction method. Specifically, instead of using the seventeen variables that normally make up the matrices, the five ML models tested here only need one value from the \mathbf{M} matrix (m_{22}), one value from the \mathbf{K}_C matrix (k_{11}), and one value from the \mathbf{C}_C matrix (c_{11}), providing a total of only three variables. Furthermore, the stated strategy works under the different path types (circle, square, and straight line) to which the 3R planar robot is subject.

The three main contributions of this work are: (i) preprocessing (data cleaning, normalization, feature importance, and redundant variables removal) of the variables from the matrices derived from various time and experiments; (ii) selection of the best model hyperparameters based on the grid search technique; and (iii) the training and testing of five different ML models using K-fold cross-validation. Furthermore, the K-fold cross-validation strategy utilized in the proposed method eliminates the need for a validation dataset. The developed damping ratio prediction approaches with the best hyperparameters work very well, with accuracy (R^2 score) results higher than 86% and RMSE outcomes less than 0.094. With the XGBoost and RF models in particular, a significant overall accuracy greater than 83.1% is attained. The XGBoost model achieves an inference time on the test data of 0.05 s, making it extremely adaptable to real-world scenarios. Comparing the test time with the time required to generate a sample (18.99 s), sample generation is 380 times slower than the ML model. These findings demonstrate the potential of ML models for the creation of robot damping prediction systems.

It is important to clarify that there are several limitations and considerations that can affect the performance of machine learning algorithms used for damping ratio prediction, among which are the following:

- **Data quality:** machine learning algorithms require a large dataset to train and validate the model, and their performance can suffer if the data are incomplete, noisy, or irrelevant.

- Model choice: The performance of machine learning algorithms can depend on model choice, and it may be necessary to try several different models in order to find the best fit for a particular problem.
- Overfitting: machine learning algorithms can overfit the model to the training data, which can reduce the model's accuracy on new data.
- Generalization: machine learning algorithms may not generalize well to new data if not enough training data have been used or if all relevant variables have not been considered.

As future work, we intend to extend the present analysis for prediction of optimal damping ratios in real robot applications in addition to simulated robots. An important point is to generate a balanced dataset in terms of the number of examples per type of trajectory, as well as the number of samples for each value of the damping ratio. Furthermore, it is desirable to expand the methodology to higher-dimensional and highly redundant systems, as this can be an excellent tool for system identification and modulation of dynamic response.

Author Contributions: Conceptualization, C.S. and C.T.; methodology, J.P. and Á.E.-D.; software, J.P. and Á.E.-D.; validation, Á.E.-D. and C.T.; formal analysis, C.S. and I.K.; investigation, J.P. and Á.E.-D.; resources, C.S.; data curation, Á.E.-D.; writing—original draft preparation, C.S., J.P., C.T., Á.E.-D., I.K. and J.S.; writing—review and editing, C.S., C.T., I.K. and J.S.; visualization, J.P. and Á.E.-D.; supervision, C.S., C.T. and I.K.; project administration, C.S. and C.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this work are available on request from the corresponding author.

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

Sample Availability: Not applicable.

Abbreviations

The following abbreviations are used in this manuscript:

DoF	Degrees of Freedom
MCK	Mass–Damping–Stiffness
ML	Machine Learning
XGBoost	eXtreme Gradient Boosting
RF	Random Forest
SVR	Support Vector Regressor
RMSE	Root Mean Square Error

References

1. Ajoudani, A.; Zanchettin, A.M.; Ivaldi, S.; Albu-Schaeffer, A.; Kosuge, K.; Khatib, O. Progress and prospects of the human–robot collaboration. *Auton. Robot.* **2018**, *42*, 957–975. [[CrossRef](#)]
2. Phaniteja, S.; Dewangan, P.; Guhan, P.; Sarkar, A.; Krishna, K.M. A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017; IEEE: Macau, China, 2017; pp. 1818–1823. [[CrossRef](#)]
3. Hwangbo, J.; Lee, J.; Dosovitskiy, A.; Bellicoso, D.; Tsounis, V.; Koltun, V.; Hutter, M. Learning agile and dynamic motor skills for legged robots. *Sci. Robot.* **2019**, *4*, eaau5872. [[CrossRef](#)] [[PubMed](#)]
4. Andrychowicz, O.M.; Baker, B.; Chociej, M.; Józefowicz, R.; McGrew, B.; Pachocki, J.; Petron, A.; Plappert, M.; Powell, G.; Ray, A.; et al. Learning dexterous in-hand manipulation. *Int. J. Robot. Res.* **2020**, *39*, 3–20. [[CrossRef](#)]
5. Fazeli, N.; Oller, M.; Wu, J.; Wu, Z.; Tenenbaum, J.B.; Rodriguez, A. See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion. *Sci. Robot.* **2019**, *4*, eaav3123. [[CrossRef](#)] [[PubMed](#)]
6. Baressi Šegota, S.; Anđelić, N.; Šercer, M.; Meštrić, H. Dynamics Modeling of Industrial Robotic Manipulators: A Machine Learning Approach Based on Synthetic Data. *Mathematics* **2022**, *10*, 1174. [[CrossRef](#)]

7. Guo, H.; Zhuang, X.; Chen, P.; Alajlan, N.; Rabczuk, T. Stochastic deep collocation method based on neural architecture search and transfer learning for heterogeneous porous media. *Eng. Comput.* **2022**, *38*, 5173–5198. [[CrossRef](#)]
8. Johannink, T.; Bahl, S.; Nair, A.; Luo, J.; Kumar, A.; Loskyll, M.; Ojea, J.A.; Solowjow, E.; Levine, S. Residual Reinforcement Learning for Robot Control. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 6023–6029. [[CrossRef](#)]
9. Johannsmeier, L.; Gerchow, M.; Haddadin, S. A Framework for Robot Manipulation: Skill Formalism, Meta Learning and Adaptive Control. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 5844–5850. [[CrossRef](#)]
10. Aichele, F.; Schenke, B.; Eckstein, B.; Groz, A. A Framework for Robot Control Software Development and Debugging Using a Real-Time Capable Physics Simulation. In Proceedings of the ISR 2016: 47st International Symposium on Robotics, Munich, Germany, 21–22 June 2016; pp. 1–8.
11. Andrychowicz, M.; Wolski, F.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Pieter Abbeel, O.; Zaremba, W. Hindsight Experience Replay. In *Proceedings of the Advances in Neural Information Processing Systems*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
12. Peng, X.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 1–8. [[CrossRef](#)]
13. Xu, Z.; Li, X.; Li, S.; Wu, H.; Zhou, X. Dynamic neural networks based adaptive optimal impedance control for redundant manipulators under physical constraints. *Neurocomputing* **2022**, *471*, 149–160. [[CrossRef](#)]
14. Meirovitch, L. *Fundamentals of Vibrations*; McGraw-Hill: New York, NY, USA, 2001.
15. Ljung, L. Perspectives on system identification. *Annu. Rev. Control* **2010**, *34*, 1–12. [[CrossRef](#)]
16. Yaghoubi, S.T.; Deger, Z.T.; Taskin, G.; Sutcu, F. Machine learning-based predictive models for equivalent damping ratio of RC shear walls. *Bull. Earthq. Eng.* **2023**, *21*, 293–318. [[CrossRef](#)]
17. Ficuciello, F.; Romano, A.; Villani, L.; Siciliano, B. Cartesian impedance control of redundant manipulators for human-robot co-manipulation. In Proceedings of the 2014 IEEE/RSJ IROS, Chicago, IL, USA, 14–18 September 2014; pp. 2120–2125. [[CrossRef](#)]
18. Penco, L.; Hoffman, E.M.; Modugno, V.; Gomes, W.; Mouret, J.B.; Ivaldi, S. Learning Robust Task Priorities and Gains for Control of Redundant Robots. *IEEE Robot. Autom. Lett.* **2020**, *5*, 2626–2633. [[CrossRef](#)]
19. Ajoudani, A.; Tsagarakis, N.G.; Bicchi, A. On the role of robot configuration in Cartesian stiffness control. In Proceedings of the 2015 IEEE ICRA, Seattle, WA, USA, 26–30 May 2015; pp. 1010–1016. [[CrossRef](#)]
20. Yamamoto, K. Resolved Multiple Viscoelasticity Control for a Humanoid. *IEEE Robot. Autom. Lett.* **2018**, *3*, 44–51. [[CrossRef](#)]
21. Zhou, S.; Gao, H.; Xu, C.; Jia, Z.; Lin, J.; Han, Q.; Luo, Z. Kinematic Modeling and Stiffness Analysis of a 3-DOF 3SPS + 3PRS Parallel Manipulator. *Mathematics* **2022**, *10*, 4465. [[CrossRef](#)]
22. Saldarriaga, C.; Chakraborty, N.; Kao, I. Damping Selection for Cartesian Impedance Control With Dynamic Response Modulation. *IEEE Trans. Robot.* **2022**, *38*, 1915–1924. [[CrossRef](#)]
23. Hogan, N. Impedance Control: An Approach to Manipulation: Part I, part II, part III. *J. Dyn. Syst. Meas. Control* **1985**, *107*, 1–24. [[CrossRef](#)]
24. Villani, L.; De Schutter, J. Force Control. In *Springer Handbook of Robotics*; Siciliano, B., Khatib, O., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 161–185. [[CrossRef](#)]
25. Khatib, O. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE J. Robot. Autom.* **1987**, *3*, 43–53. [[CrossRef](#)]
26. Chen, S.F.; Kao, I. Conservative Congruence Transformation for Joint and Cartesian Stiffness Matrices of Robotic Hands and Fingers. *Int. J. Robot. Res.* **2000**, *19*, 835–847. [[CrossRef](#)]
27. Saldarriaga, C.; Chakraborty, N.; Kao, I. Joint Space Stiffness and Damping for Cartesian and Null Space Impedance Control of Redundant Robotic Manipulators. In Proceedings of the 2019 International Symposium on Robotics Research, Hanoi, Vietnam, 6–10 October 2019; Springer: Cham, Switzerland, 2022; pp. 410–426.
28. Kang, M.; Tian, J. Machine Learning: Data Pre-processing. In *Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things*; John Wiley and Sons Ltd.: Hoboken, NJ, USA, 2018; pp. 111–130.
29. Hossen, M.S. Data preprocess. In *Machine Learning and Big Data: Concepts, Algorithms, Tools and Applications*; Wiley: Hoboken, NJ, USA, 2020; pp. 71–103.
30. Akoglu, H. User’s guide to correlation coefficients. *Turk. J. Emerg. Med.* **2018**, *18*, 91–93. [[CrossRef](#)] [[PubMed](#)]
31. Schober, P.; Boer, C.; Schwarte, L. Correlation Coefficients: Appropriate Use and Interpretation. *Anesth. Analg.* **2018**, *126*, 1763–1768. [[CrossRef](#)] [[PubMed](#)]
32. Sharmin, S.; Shoyaib, M.; Ali, A.A.; Khan, M.A.H.; Chae, O. Simultaneous feature selection and discretization based on mutual information. *Pattern Recognit.* **2019**, *91*, 162–174. [[CrossRef](#)]
33. Zhou, H.; Wang, X.; Zhu, R. Feature selection based on mutual information with correlation coefficient. *Appl. Intell.* **2022**, *52*, 5457–5474. [[CrossRef](#)]
34. Brankovic, A.; Hosseini, M.; Piroddi, L. A distributed feature selection algorithm based on distance correlation with an application to microarrays. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2019**, *16*, 1802–1815. [[CrossRef](#)] [[PubMed](#)]

35. Tan, H.; Wang, G.; Wang, W.; Zhang, Z. Feature selection based on distance correlation: A filter algorithm. *J. Appl. Stat.* **2022**, *49*, 411–426. [[CrossRef](#)] [[PubMed](#)]
36. Radha, R.; Muralidhara, S. Removal of redundant and irrelevant data from training datasets using speedy feature selection method. *Int. J. Comput. Sci. Mob. Comput.* **2016**, *5*, 359–364.
37. Huang, L.; Qin, J.; Zhou, Y.; Zhu, F.; Liu, L.; Shao, L. Normalization Techniques in Training DNNs: Methodology, Analysis and Application. *arXiv* **2020**, arXiv:2009.12836.
38. Chen, T.; Guestrin, C. XGBoost. In Proceedings of the the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016 ; ACM: New York, NY, USA, 2016. [[CrossRef](#)]
39. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
40. Rasaei, Z.; Bogaert, P. Spatial filtering and Bayesian data fusion for mapping soil properties: A case study combining legacy and remotely sensed data in Iran. *Geoderma* **2019**, *344*, 50–62. [[CrossRef](#)]
41. Were, K.; Bui, D.T.; Dick, Ø.B.; Singh, B.R. A comparative assessment of support vector regression, artificial neural networks, and random forests for predicting and mapping soil organic carbon stocks across an Afrotropical landscape. *Ecol. Indic.* **2015**, *52*, 394–403. [[CrossRef](#)]
42. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the Advances in Neural Information Processing Systems*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
43. Dorogush, A.V.; Ershov, V.; Gulin, A. CatBoost: Gradient boosting with categorical features support. *arXiv* **2018**. [[CrossRef](#)]
44. Wu, X.; Kumar, V.; Ross Quinlan, J.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G.J.; Ng, A.; Liu, B.; Yu, P.S.; et al. Top 10 algorithms in data mining. *Knowl. Inf. Syst.* **2008**, *14*, 1–37. [[CrossRef](#)]
45. Algehyne, E.A.; Jibril, M.L.; Algehainy, N.A.; Alamri, O.A.; Alzahrani, A.K. Fuzzy Neural Network Expert System with an Improved Gini Index Random Forest-Based Feature Importance Measure Algorithm for Early Diagnosis of Breast Cancer in Saudi Arabia. *Big Data Cogn. Comput.* **2022**, *6*, 13. [[CrossRef](#)]
46. Nembrini, S.; Koenig, I.R.; Wright, M.N. The revival of the Gini importance? *Bioinformatics* **2018**, *6*, 3711–3718. Available online: <https://academic.oup.com/bioinformatics/article/34/21/3711/4994791> (accessed on 9 February 2023). [[CrossRef](#)] [[PubMed](#)]
47. Kaneko, H. Cross-validated permutation feature importance considering correlation between features. *Anal. Sci. Adv.* **2022**, *3*, 278–287. [[CrossRef](#)]

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.