Article

Using Machine Learning to Predict the Performance of a Cross-Flow Ultrafiltration Membrane in Xylose Reductase Separation

Reza Salehi [1], Santhana Krishnan [1], Mohd Nasrullah [2] and Sumate Chaiprapat [1,3,*]

[1] Department of Civil and Environmental Engineering, Faculty of Engineering, Prince of Songkla University, Hat Yai Campus, Songkhla 90110, Thailand
[2] Faculty of Civil Engineering Technology, University of Malaysia Pahang, Lebuhraya Tun Razak, Gambang 26300, Malaysia
[3] PSU Energy Systems Research Institute, Research and Development Office, Prince of Songkla University, Songkhla 90110, Thailand
* Correspondence: sumate.ch@psu.ac.th

Abstract: This study provides a new perspective for xylose reductase enzyme separation from the reaction mixtures—obtained in the production of xylitol—by means of machine learning technique for large-scale production. Two types of machine learning models, including an adaptive neuro-fuzzy inference system based on grid partitioning of the input space and a boosted regression tree were developed, validated, and tested. The models' inputs were cross-flow velocity, transmembrane pressure, and filtration time, whereas the membrane permeability (called membrane flux) and xylitol concentration were considered as the outputs. According to the results, the boosted regression tree model demonstrated the highest predictive performance in forecasting the membrane flux and the amount of xylitol produced with a coefficient of determination of 0.994 and 0.967, respectively, against 0.985 and 0.946 for the grid partitioning-based adaptive neuro-fuzzy inference system, 0.865 and 0.820 for the best nonlinear regression picked from among 143 different equations, and 0.815 and 0.752 for the linear regression. The boosted regression tree modeling approach demonstrated a superior capability of predictive accuracy of the critical separation performances in the enzymatic-based cross-flow ultrafiltration membrane for xylitol synthesis.

Keywords: adaptive neuro-fuzzy inference system; boosted regression trees; cross-flow ultrafiltration; grid partitioning; (non)linear regression; xylitol; xylose reductase

check for updates

Citation: Salehi, R.; Krishnan, S.; Nasrullah, M.; Chaiprapat, S. Using Machine Learning to Predict the Performance of a Cross-Flow Ultrafiltration Membrane in Xylose Reductase Separation. *Sustainability* **2023**, *15*, 4245. https://doi.org/10.3390/su15054245

Academic Editors: Andreas Kanavos, Lidietta Giorno and Emmanouil Papaioannou

Received: 17 December 2022
Revised: 19 February 2023
Accepted: 22 February 2023
Published: 27 February 2023

1. Introduction

Xylose reductase (XR) is a member of the aldose reductase or aldehyde reductase (ALR) family (EC 1.1.1.21), which belongs to the aldo-keto reductase (AKR) superfamily of enzymes [1,2]. It catalyzes the reduction of xylose (found in hemicellulose hydrolysates from lignocellulosic biomass) to xylitol [3], which has enormous applications in the pharmaceutical, food, and beverage industries [4] as its global market size is expected to increase from USD 1 billion in 2022 [5] to USD 1.37 billion by 2025 [6].

XR has been reported to be found in the cytoplasm of a wide variety of microorganisms, including bacteria, molds, algae, and yeasts [5,7,8]. However, as has appeared in the literature, only yeast species have been extensively studied. Some examples of yeast XRs include *Candida shehatae* [9], *Candida tropicalis* [10–12], *Candida guilliermondii* [13,14], *Pichia fermentans* [6], *Chaetomium themophilum* [15], *Spathaspora arborariae/passalidarum* [16], *Barnettozyma populi/california/salicaria* and *Cyberlindnera mrakii* [7], *Neurospora crassa* [17], and *Kluyveromyces* sp. [18,19]. In general, XR prefers NADPH (reduced form of nicotinamide adenine dinucleotide phosphate as a coenzyme (donor of hydrogen atoms)) to convert xylose to xylitol. However, dual coenzyme specificity for NADPH and NADH has also been reported [8,15,16].

One of the major challenges associated with the enzymatic-based xylitol production process is downstream processing. It is referred to further process the reaction mixture for the recovery of the targeted component of interest; for example, the separation of the XR enzyme, as it is not available commercially [14], is highly desired. With respect to the fact that downstream processes are potential cost drivers [20], a proper design for the downstream process is required to offer good purification yield for the component of interest while being as simple as possible and cost-efficient to make their practical application feasible for large-scale industrial processes. In this regard, membrane filters have proven to be valuable tools because they are energy-efficient technologies, and their scale-up is relatively easy [21–23]. Membrane filters are categorized in accordance with the types of driving forces and the size of pores, among which the tangential-flow filtration (more often known as cross-flow (CF) filtration) with an ultrafiltration (UF) membrane has become the most attractive and promising technology [24] because it is economically viable and offers high productivity along with good purity level for the targeted product of interest. The performance of such a process is strongly influenced by several operating parameters, most notably CF velocity, transmembrane pressure (i.e., the difference between the applied and the osmotic pressure), and filtration time (referred to hereafter as CFV, TMP, and FT, respectively) [20]. Hence, the development of a model to accurately predict the performance of CF–UF membrane process is highly desired as it could help engineers and asset managers to better control the operating parameters.

Background

Numerous mathematical models describing the mechanism of solute(s) transport through the CF–UF membrane system are available in the literature. The first one—the gel layer model—was introduced by Michaels [25]; however, this model does not take into account the osmotic pressures of macromolecular solutes. Hence, such a model seems not to be appropriate for ultrafiltration of protein solutions because Vilker et al. [26] observed that the concentrated solutions of bovine serum albumin can exert osmotic pressure. Bellara and Cui [27] proposed a nonparameterized model to predict the permeate flux for protein (i.e., bovine serum albumin) solutions in CF–UF tubular membrane. The proposed model was a combination of a hydrodynamic model to support the growth of boundary layer, and Maxwell–Stefan equations [28] to describe electrostatic interactions generated through filtration of charged molecules. The model was numerically solved with (and without) taking into account viscose integrations within the concentration polarization layer, and tested against data (TMP = 40 kPa, pH of 5.4 and 7.4). When the model considered viscous integrations, an excellent correlation between the model predictions and measured data was observed. One concern associated with Bellara and Cui's model [27] is neglecting the electro-viscous effects in order to keep the model simple. Ahmad et al. [29] developed a model to predict the volume of permeate flux, gel layer resistance, and the rejection of each solute in a multiple-solute solution using a pilot-scale CF–UF membrane, where the feed was a complex biological solution (i.e., the pretreated palm oil mill effluent containing the solutes of ammonical nitrogen, carbohydrate constituents, and crude protein). The model was constructed based on osmotic pressure-/resistance-in-series-/gel-polarization-model taking into account mass balance analysis. The authors estimated the model parameters (e.g., membrane resistance, mass transfer and back transport coefficients of the solutes, and permeating coefficient) by means of Levenberg–Marquardt and Gauss–Newton algorithms. The model predictions were in a good agreement with the experimental results. In the study conducted by Karasu et al. [30], a model was developed for separation of protein from a synthetic whey concentrate suspension in a CF–UF system by modifying the compressive yield stress model for permeate flux through a fouling layer on a dead-end UF membrane that was introduced by [31–33]. In Karasu et al.'s model [30], it was assumed that (i) some of the solid particles on the fouling layer surface are continuously detached by shear stress created by the influent flow, and the rate of particle removal (volume per unit time and per unit membrane area) is constant; (ii) the effect of the concentration polarization layer on the shear stress on the top part of the fouling layer is negligible; and (iii) shear stress and cake compression are functions of CF velocity and TMP, respectively. The authors used the removal rate of the fouling layer as a fitting factor, which was estimated by trial-and-error method so that the model-predicted ratios of filtration time-to-volume of permeate per unit membrane area best fitted the corresponding measured values. The results indicated that permeate flux and filtration time predicted by the proposed model correlated well with the measured data in steady-state condition. However, the model was not able to adequately fit the experimental data in the beginning of the time course of permeate. Nguyen et al. [34] modified Karasu et al.'s model [30] considering that membrane fouling (using a pore blockage model) and the growth of the cake layer (using a compressive yield stress model) take place simultaneously. Their results demonstrated that such a combined model well described the entire time course of the permeate. Kirschner et al. [35] modified Hermia's models [36] (i.e., intermediate pore blocking and cake filtration models reported for filtration and fouling in dead-end membrane under constant pressure) to describe fouling during filtration using a polyether sulfone flat sheet UF membrane operated under constant CF filtration; intermediate pore blocking refers to a phenomenon where solid particles are allowed to either deposit onto an unobstructed membrane's surface area or on top of an early deposited solid particle, whereas, in cake formation, solid particles fully cover the membrane's surface area in some layers. The authors also used Field et al.'s model [37] to take into account the CF filtration term in their model. A latex bead suspension (200 ppm 0.22 μm) and a soybean oil emulsion (200 ppm) were separately utilized as foulant. According to the simulation results, below the threshold flux (i.e., the flux below which cake formation is insignificant and beyond which cake formation governs the fouling mechanism), the intermediate pore blocking model fitted the experimental values well. However, above the threshold flux, the combination of the intermediate pore blocking model and the cake filtration model produced the best fit.

Even though various mathematical models have been developed to predict the efficiency of ultrafiltration of a complex mixture, they might not be able to successfully model the permeation flux decline because of occurring a wide range of different and complicated phenomena during fouling of the UF membrane, which could be associated with (an unknown) interaction between feed compositions, the membrane itself (i.e., nature and surface properties), and operating/hydrodynamic conditions [38,39]. In addition, equations involved in the mathematical models may be complex and there may be a need for simplifications by incorporating assumptions; in such case, if the assumptions are far from reality, these could lead to an under/overestimation in the model-predicted values. Furthermore, the mathematical models describing behavior of UF processes often suffer from a number of limitations. For instance, they are not able to describe the flux-time behavior under unsteady state conditions. Moreover, a given mathematical model can be applied only for a particular feed under certain conditions [40]. For these reasons, alternative methods (much simpler and more flexible in comparison with mathematically established models, and with high prediction accuracy) for predicting UF process performance are nowadays in high demand. In this context, one approach that can be recruited is machine leaning (ML) modeling (known as an easy-to-use black-box technique). The ML models do not require detailed/theoretical knowledge of mechanisms involved in the process or the relationships between the parameters that govern it. They are constructed based on only a measured input–output dataset. Such modeling approach is a robust and powerful tool possessing high generalization ability [41].

Wei et al. [42] used a single hidden layer feedforward backpropagation neural network (FBPNN) to predict the dynamic permeate flux of cross-flow ultrafiltration of colloidal suspensions (i.e., spherical silica colloids with particle diameter of 65 nm) as a function of TMP, FT, ionic strength, and zeta potential (data taken from Bowen et al. [43]). The results showed that the FBPNN model satisfactory described the nonlinear behavior of permeate flux, offering a training and testing R^2 value of 0.997 and 0.914, respectively, against the linear multiple regression (MR) and nonlinear MR with the training and testing R^2 values of 0.752 and 0.800, and 0.809 and 0.848, respectively. In another study, Krippl et al. [39] developed a single hidden layer FBPNN model and successfully predicted the permeate flux of cross-flow ultrafiltration process as a function of CF (100–300 mL min^{-1}), TMP (1.3–2.5 bar), and the initial protein bulk concentration (1.9–23.2 g L^{-1}) (i.e., bovine serum albumin and lysozyme from chicken egg white), where two types of membranes were used (i.e., hydrophobic polyether sulfone and hydrophilic stabilized cellulose-based membrane). The authors further combined the FBPNN model with a mechanistic model such that the hybrid model showed a high predictive accuracy in predicting the duration of filtration in both batch and fed-batch continuous modes.

Yet, to the best of the authors' knowledge, the application of machine learning models in predicting the performance of CF–UF membrane process in XR enzyme purification has never been explored. This inspired the authors to conduct this study aimed to develop, validate, and test two different machine learning models, including an adaptive neuro-fuzzy inference system based on grid partitioning of the input space (ANFIS-GP) (an advanced ML model possessing the advantage of both numerical and linguistic knowledge, which is more transparent to the user/reader compared to other neural network models), and boosted regression trees (an easy-to-understand ML model even for people without an analytical background) to predict the performance of a CF–UF membrane process in the separation of XR as a function of the operating parameters CFV, TMP, and FT. The predictive performances of these two models were compared with each other, and with that of the best nonlinear MR picked from among 143 different regressions, and of a linear MR.

In this study, initially, the dataset is given followed by modeling approaches including ANFIS-GP, boosted regression trees, and (non)linear regression. Next, the modeling results are discussed and compared. Then, sensitivity analysis for the best predictive model is provided. Finally, this study ends with conclusions.

2. Methodology

A diagram illustrating the workflow of this study is depicted in Figure 1; refer to text for further details.

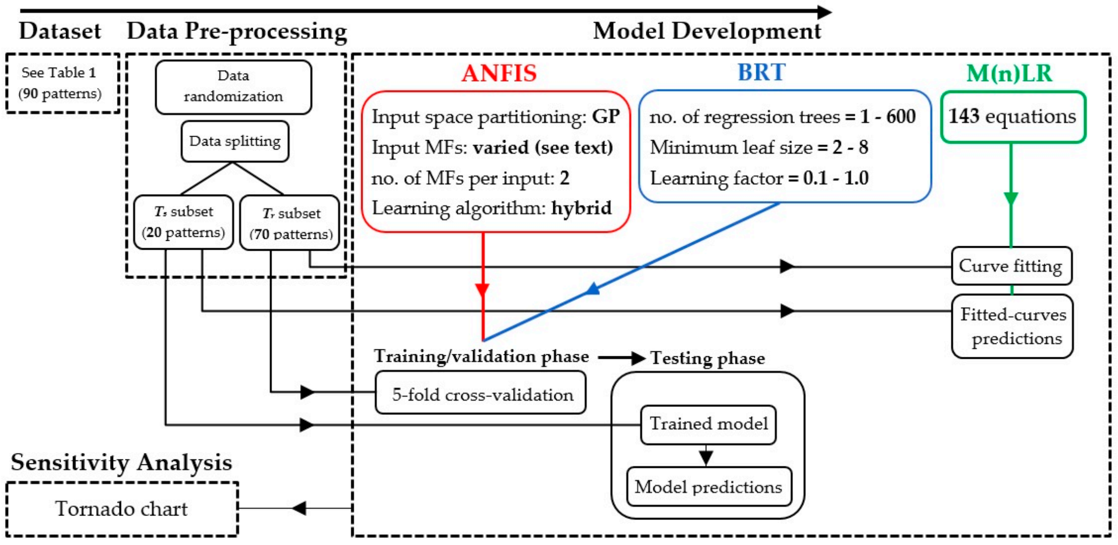

Figure 1. A schematic illustration of the workflow of this study (Tr: training subset; Ts: testing subset; ANFIS: adaptive neuro-fuzzy inference system; BRT: boosted regression tree; M(n)LR: multiple (non)linear regression; see text for further details).

2.1. Dataset

The data used in this study were obtained from Krishnan et al. [20], who operated a lab-scale CF–UF membrane for separation of XR enzyme from the reaction mixture (i.e., xylitol, xylose, glucose, arabinose, acetic acid, and XR) during xylitol synthesis. The reaction mixture was subjected to ultrafiltration, where a filtration process was performed at TMP and CFV values of 1.2 bar and 1.06 cm s^{-1}, respectively, to assess how the filtration time—ranging from 10 min to 100 min—influenced the process performance (i.e., the membrane permeability—called membrane flux—and the amount of xylitol produced). The authors also conducted two other series of experiments for the purpose of investigating the effects of TMP and CFV on the process performance: (i) TMP varied between 0.8 and 1.6 bar at a constant CFV value of 1.06 cm s^{-1}, and (ii) CFV varied between 0.58 and 1.2 cm s^{-1} while TMP was held constant at 1.2 bar. The experimental conditions in all the runs are presented in Table 1.

Table 1. Operating conditions used in the experimental UF membrane runs [20].

CFV = 0.58 cm s^{-1}					**CFV = 0.70 cm s^{-1}**					**CFV = 0.82 cm s^{-1}**				
Exp. code	**TMP (bar)**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**TMP (bar)**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**TMP (bar)**	**FT (min)**	y_1	y_2 (g L^{-1})
E1	1.2	10	0.5321	15.51	E2	1.2	10	0.5976	15.66	E3	1.2	10	0.6397	15.80
		20	0.4865	15.48			20	0.5439	15.61			20	0.5623	15.84
		30	0.4586	15.34			30	0.4859	15.52			30	0.5157	15.70
		40	0.4351	15.12			40	0.4553	15.36			40	0.4906	15.62
		50	0.4232	15.00			50	0.4478	15.20			50	0.4940	15.45
		60	0.4253	14.89			60	0.4497	15.07			60	0.4764	15.40
		70	0.4054	14.56			70	0.4467	14.76			70	0.4781	15.37
		80	0.3967	14.23			80	0.4397	14.50			80	0.4718	15.26
		90	0.3932	14.00			90	0.4387	14.33			90	0.4684	15.20
		100	0.3937	13.00			100	0.4365	14.30			100	0.4673	15.00
CFV = 1.20 cm s^{-1}					**CFV = 1.06 cm s^{-1}**					**TMP = 0.8 bar**				
Exp. code	**TMP (bar)**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**TMP (bar)**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**CFV (cm s^{-1})**	**FT (min)**	y_1	y_2 (g L^{-1})
E4	1.2	10	0.7643	16.27	E5	1.2	10	0.6639	15.97	E6	1.06	10	0.5296	15.40
		20	0.6753	16.10			20	0.5572	15.71			20	0.4820	15.23
		30	0.6381	16.00			30	0.5233	15.31			30	0.4570	15.00
		40	0.5923	15.92			40	0.5018	15.14			40	0.4183	14.65
		50	0.5529	15.85			50	0.4808	14.94			50	0.4176	14.60
		60	0.5343	15.63			60	0.4674	14.63			60	0.3932	14.58
		70	0.5319	15.55			70	0.4585	14.57			70	0.3723	14.56
		80	0.5307	15.54			80	0.4634	14.34			80	0.3685	14.50
		90	0.5318	15.40			90	0.4593	14.17			90	0.3652	14.20
		100	0.5287	15.42			100	0.4572	13.95			100	0.3622	13.24
TMP = 1.0 bar					**TMP = 1.4 bar**					**TMP = 1.6 cm s^{-1}**				
Exp. code	**CFV (cm s^{-1})**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**CFV (cm s^{-1})**	**FT (min)**	y_1	y_2 (g L^{-1})	**Exp. code**	**CFV (cm s^{-1})**	**FT (min)**	y_1	y_2 (g L^{-1})
E7	1.06	10	0.5941	15.66	E8	1.06	10	0.7188	16.10	E9	1.06	10	0.7832	16.25
		20	0.5412	15.61			20	0.6320	16.06			20	0.6732	16.00
		30	0.5165	15.52			30	0.6072	15.89			30	0.6532	15.90
		40	0.4953	15.36			40	0.5532	15.72			40	0.6238	15.76
		50	0.4618	15.20			50	0.5395	15.60			50	0.5844	15.65
		60	0.4482	15.07			60	0.5064	15.46			60	0.5583	15.63
		70	0.4371	14.76			70	0.4841	15.55			70	0.5367	15.55
		80	0.4382	14.50			80	0.4591	15.36			80	0.5347	15.54
		90	0.4359	14.33			90	0.4586	15.20			90	0.5328	15.40
		100	0.4382	14.30			100	0.4573	15.21			100	0.5320	15.42

TMP: transmembrane pressure, CFV: cross-flow velocity; FT: filtration time; y_1: normalized flux (i.e., ratio of the permeate flux to the pure water flux); y_2: xylitol concentration.

2.2. Modeling Approaches

To construct predictive models (described in Sections 2.2.1–2.2.3), FT (x_1), TMP (x_2), and CFV (x_3) were considered as the inputs, while the normalized flux (y_1), and xylitol concentration (y_2) were the outputs. As seen in Table 1, the raw dataset consists of a total number of 90 input–output data pairs (referred to hereafter as observations). Before using the raw dataset, it was randomized using Excel (version 2016, Microsoft Corp., Redmond, WA, USA) and subsequently split into two disjoint subsets: training subset and testing subset. Out of 90 observations, 70 observations (corresponding to about 78% of the dataset) were used as training subset to develop the models, while the remaining 22% of the dataset (i.e., 20 observations) was served as testing subset in order to evaluate the prediction power of the trained (developed) models; note that the training subset was further divided into five subsets in order to perform a cross-validation (CV) technique to avoid the models from overfitting the training data. The training and testing subsets were stored in the workspace of MATLAB® (trial version, R2020a) (MathWorks Inc., Natick, MA, USA) in the form of arrays, in which each row represented an observation.

The following subsections provide different modeling approaches used for predicting the performance of the UF membrane process in the separation of XR from the reaction mixture—obtained in xylitol synthesis—as a function of FT, TMP, and CFV.

2.2.1. ANFIS Model

ANFIS, a computational intelligence technique, integrates the learning ability of an artificial neural network (ANN) with the ability of an FIS in knowledge interpretation. The ANFIS modeling technique is used to model complex tasks with nonlinearity behavior, which is not easy to be modeled mathematically. Such models are built based on if-then rules and a collection of input–output data pairs [44,45].

Let us suppose that an FIS is composed of two inputs (x_i; i = 1 and 2)—each characterized by two fuzzy sets A_i and B_i that are specified with an appropriate membership function (MF)—and one output $f(x_1,x_2)$. Furthermore, let us suppose that the FIS is in the form of the first-order Takagi–Sugeno FIS [46]. The relationship between inputs and output can be described in accordance with the following rule sets, each consists of an "if part" (called premise or antecedent part) and a "then part" (called conclusion or consequent part):

Rule (1):

$$\text{If } (x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2), \text{ then } f_1(x_1, x_2) = a_1 x_1 + b_1 x_2 + c_1 \tag{1}$$

Rule (2):

$$\text{If } (x_1 \text{ is } B_1 \text{ and } x_2 \text{ is } B_2), \text{ then } f_2(x_1, x_2) = a_2 x_1 + b_2 x_2 + c_2 \tag{2}$$

where A_i and B_i denote the fuzzy sets pertaining to x_i (i = 1 and 2), and each is specified by an appropriate MF (note that parameters associated with Ai and Bi are referred to as antecedent parameters); $f(x_1,x_2)$ is the output function of the i-th rule (i = 1 and 2); ai, bi, and ci are referred to as antecedent parameters, corresponding to the i-th output function (i = 1 and 2).

A schematic representation of a two-input one-output first-order Takagi–Sugeno FIS with two fuzzy rules is displayed in Figure 2.

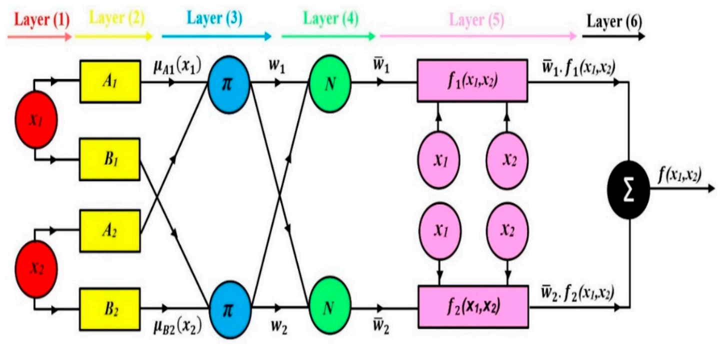

Figure 2. A schematic representation of a two-input one-output first-order Takagi–Sugeno FIS with two fuzzy rules. x_1 and x_2 are the inputs, A_i and B_i are the fuzzy sets pertaining to the input x_i (i = 1 and 2), $\mu_{Ai}(x_i)$ is the membership function of fuzzy set A associated with x_i (i = 1 and 2), $\mu_{Bi}(x_i)$ is the membership function of fuzzy set B associated with x_i (i = 1 and 2), the nodes marked as "π" apply a product operation, the nodes marked as "N" normalize the firing strength w_i (i = 1 and 2), the single node marked as $\sum$ performs a summation function, $f_i(x_1, x_2)$ represents the output function of the i-th rule (i = 1 and 2), and a rectangle represents an adaptive node while a circle represents a fixed node (an adaptive node has modifiable parameters while a fixed node does not have modifiable parameters).

It can be observed from Figure 2 that the ANFIS model is a six-layer feed-forward neural network wherein each layer, except layer 1, consists of processing units (more often called nodes) that perform particular functions on their incoming signals and generate outputs, which are considered as inputs for the next layer; layer 1, called input layer, has no processing units; in other words, the input layer nodes only transfer the input data to the next layer. The reader is referred to [47,48] for a detailed description of each layer.

- ANFIS-GP

One of the most popular techniques developed to identify the structure of the ANFIS model is based on grid partitioning (GP) of the input space. This technique—with respect to the number/type of MFs assigned to each dimension of a given input space—splits the input space into a number of subspaces (called fuzzy regions each characterized by a fuzzy if-then rule) using an axis-paralleled partition. The total number of fuzzy rules equals the number of all possible combinations of all the inputs [49]. Figure 3 illustrates an example of grid partitioning of a two-dimensional input space with three triangular MFs assigned to each dimension.

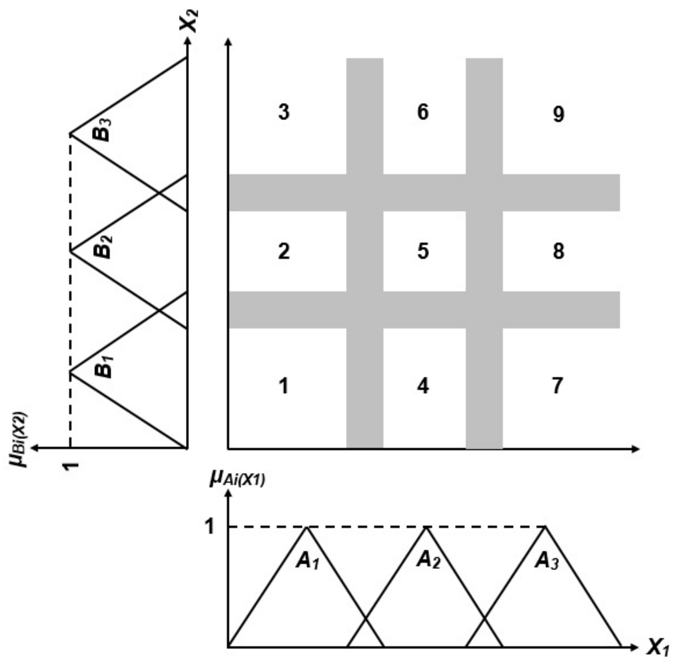

Figure 3. Grid partitioning of an input domain composed of two input variables (X_1 and X_2) each assigned three triangular membership functions. A_i and B_i are the fuzzy sets pertaining to the input X_i (i = 1 and 2), and $\mu_{ji}(X_i)$ is the membership function of the j-th fuzzy set (A or B) associated with X_i (i = 1 and 2).

Let us suppose that an input space contains a collection of n inputs $\{x_1, x_2, \ldots, x_n\}$. Applying GP technique on such an input space yields $m_1 \times m_2 \times \ldots \times m_n$ fuzzy rules where m_i stands for MFs associated with the i-th input x_i. It can be implied that the number of fuzzy rules rapidly rises as the number of MFs increases. Therefore, training the ANFIS model becomes computationally expensive because the number of fitting parameters increases. According to Jang [50], GP is an appropriate technique only for problems with a small number of input variables (e.g., smaller than six). In this study, the problem at hand deals with three inputs. Hence, applying the ANFIS-GP is suitable.

- ANFIS Parameters

ANFIS models contain two parameter sets: (1) antecedent parameter set and (2) consequent parameter set. Let us suppose an ANFIS is created based on the first-order Takagi–Sugeno FIS. The number of antecedent and consequent parameters can be computed according to Equations (3) and (4).

$$N_A = N \times P \times M \tag{3}$$

$$N_c = N_R = \times (N+1) \tag{4}$$

where N_A is the number of antecedent parameters; N is the number of inputs to the ANFIS model; P is the number of input MFs (for example, in the case of generalized bell-shaped MF; P equals (3)); M is the number of MFs assigned to each input; N_C is the number of consequent parameters; and N_R is the number of fuzzy rules.

A large body of literature has reported that a combination of the error backpropagation (BP) and the least squares estimation (LSE) methods, called a hybrid algorithm, is very efficient to optimize ANFIS parameters [50–53]. Each iteration of the hybrid algorithm includes two passes as follows:

(1) Forward pass: The input signals go forward through the network, layer by layer, until the defuzzification layer wherein the LSE method is applied to determine the consequent parameters while the antecedent parameters remain constant.
(2) Backward pass: The error signals are back propagated from the output layer to the input layer and gradient descent is used to adjust the antecedent parameters, while the consequent parameters remain constant.

The ANFIS output is calculated using the consequent parameters determined in the forward pass.

In this study, an ANFIS-GP model was implemented in MATLAB® (trial version, R2020a) (MathWorks Inc., Natick, MA, USA) using ANFIS Editor GUI (graphical user interface), which can be displayed by typing "anfisedit" in the MATLAB command window. Initially, the training and testing subsets were loaded from the MTALAB workspace, and then, an FIS in the form of the first-order Takagi–Sugeno FIS was created considering that a five-fold CV technique was applied to secure the model against overfitting the data.

For the problem at hand, various ANFIS models were developed differing in the types of input MFs to choose the best model as the one with minimum validation error. Six types of input MFs, including Gaussian MF (*gaussmf*), generalized bell-shaped MF (*gbellmf*), trapezoidal MF (*trapmf*), triangular MF (*trimf*), the product of two sigmoidal MFs (*psigmf*), and a combination of two Gaussian MFs (*gauss2mf*) were examined (refer to the study of Clemente [54] for the mathematical definition of these MFs). The number of MFs per input was set to two; note that an increase beyond two was found to be inappropriate due to the creation of an ANFIS model in which the total number of trainable parameters was greater than the number of observations in the training subset. Each ANFIS-GP model was trained using the hybrid algorithm, and the training error goal was set to zero. The algorithm stopped learning when the training error reached zero. Otherwise, the algorithm continued iterating up to a certain iteration beyond which the training error was not further decreased. In case none of these two stopping criteria were satisfied, iterating was progressed up the predetermined number of iterations (i.e., 100 iterations in this study). Once the model training process was complete, the model was tested on the testing subset (unseen during training process). The performance of the ANFIS-GP models was assessed using three statistical indices, namely coefficient of determination (R^2), root mean squared error (*RMSE*), and Nash–Sutcliffe efficiency coefficient (*NSE*) defined by Equations (5), (6), and (7), respectively [55–57].

$$R^2 = \left(\frac{\sum_{i=1}^{n} (y_i - \overline{y}) \left(y_i^p - \overline{y^p} \right)}{\sqrt{\sum_{i=1}^{n} (y_i - \overline{y})^2} \sqrt{\sum_{i=1}^{n} (y_i^p - \overline{y^p})^2}} \right)^2 \quad 0 \le R^2 \le 1 \tag{5}$$

$$RMSE = \left(\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^p\right)^2\right)^{0.5} \quad 0 \leq RMSE \leq \infty \tag{6}$$

$$NSE = 1 - \frac{\sum_{i=1}^{n}\left(y_i - y_i^p\right)^2}{\sum_{i=1}^{n}\left(y_i - \overline{y}\right)^2} \quad -\infty \leq NSE \leq 1 \tag{7}$$

where y_i and y_i^p prepresent the measured values of the output, and the model predicted values for the i-th observations, respectively; $\overline{y}$ and $(\overline{y^p})$ represent the average value of y_i and y_i^p, respectively (i = 1, 2, . . . , n); and n is the total number of observations (in the training or testing subset), on which R^2, *RMSE,* and *NSE* are calculated.

From Equations (5)–(7), a perfect model would expect to achieve R^2 and *NSE* values equal to unity, and an *RMSE* value of zero (i.e., y_i^p = y_i).

2.2.2. Boosted Regression Trees

Boosted regression trees is a modelling approach composed of two algorithms, including regression trees (RT) and boosting [58]. The following subsections describe a brief introduction to RT and boosting algorithms.

- RT Algorithm

RT, a decision support tool dealing with continuous (numeric) predictors [59,60], is among the most common machine learning algorithms, which has been widely used in various fields (e.g., engineering, science, finance, etc.) due to its simplicity and high interpretability [61–63]. The RT algorithm was first introduced by Breiman et al. [64], and it consists of a series of decision nodes on input variables called explanatory or predictor variables. A decision node is represented by a conditional statement, i.e., a binary question with the options of being "true" or "false". This results in the creation of two arcs (branches) from each node; note that if the conditional statement is true, the observations correspond to the given predictor variable fall to the right branch, otherwise, they fall to the left branch. Each branch may lead to a decision node, which is branched out again, or may end up in an unsplit node called terminal (output) node or leaf depending on the minimum number of data points that a node can take in order to be split. Since this approach forms a tree-shaped structure, it is known as RT. Most RTs are drawn upside down, which means that the parent node (called root node) is the topmost, and the leaves are at the bottom of the tree [65–67]. The algorithm examines binary splits for all predictors to find their corresponding cut-off values that offer the best split. The best predictor, which is assigned to the root node, is selected according to the sum of the squared residuals index after data splitting; this process is repeated for new nodes (i.e., child nodes generated from the root node, grandchild nodes, etc.) until a stopping condition is satisfied (for instance, minimum leaf size (*mls*) is used as a stopping parameter); the splitting of nodes is terminated if the number of data points per node becomes smaller than the *mls*. The mathematical background of the RT model is given in detail below [65,68].

Let us suppose that a given system is described with a training dataset consisting of n observations $\{x_i, y_i\}_1^n$ with $x \in R^m$ and $y \in R$, where x represents an m-element vector (m denotes the number of predictor variables) and y is the corresponding output variable. The objective is to use this dataset to find a function, $f(x)$, mapping x into y, which minimizes the expected value of a particular loss function $L(y, f(x))$ on the training dataset; an often used L function is the sum of squared residuals (*SSR*), where residuals are the difference between y and $f(x)$. This function, $f(x)$, can then be served to predict the output of query observations where only predictor variables are known.

First, all observations are sorted according to the values of each predictor [69]. Let us suppose that $\{x_1, x_2, \ldots, x_n\}$ are the sorted values of a given predictor x_j. Any value between x_i and x_{i+1} will split the set to the same two subsets, and thus, the possible number of splits that needs to be tested is n-1. In general, the midpoint of each interval $[x_i, x_{i+1}]$ is considered the splitting point. Then, the algorithm starts with a root node where, for a given predictor x_j, we choose the cut-off point (c) such that the binary splitting of the observations into the right-hand node with $x_j \geq c$ and the left-hand node with $x_j < c$ results in the greatest possible reduction in the accumulative *SSR* on both the right- and left-hand nodes (*SSR* index is used as a criterion to assess the quality of a given split). All predictors and all possible *c* values that correspond to each of the predictors are examined. A particular predictor with a *c* value that produces the overall minimal *SSR* is selected as the variable, which should reside on the root node. Mathematically, it can be expressed as follows:

$$SSR_j = \sum_{i \in LN} (y_i - y^*_{LN})^2 + \sum_{i \in RN} (y_i - y^*_{RN})^2 \tag{8}$$

where SSR_j is sum of squared residual correspond to the *j*-th predictor, and y^*_{LN} and y^*_{RN} represent the averaged outputs of the series of observations corresponding to the left-hand node and to the right-hand node, respectively.

This process is repeated to find the best predictor and *c* value to split the dataset further so that the *SSR* in each of the prior created nodes is minimized. The process continues until a stopping condition is satisfied. For example, it stops once the size of none of the leaves is greater than *mls*, which is set by the user. When the tree is formed, the output of a given observation, i.e., new (test) observation, is made by averaging the outputs of all the training observations in the leaf where the test observation belongs.

- Boosting

Single RT-based models—despite widespread use due to their simplicity and high interpretability, being able to deal with missing variables and to handle a mixture of continuous and categorical predictor variables, and insensitive to outliers [58,67] —suffer from low accuracy in term of prediction [60]. In addition, they are often unstable, which means that small changes in a training dataset may lead to forming a tree with different series of splits [70–73]. Hence, the boosting algorithm that was first proposed by Schapire [74]—later developed by Freund [75] and Freund and Schapire [76]—can be served as a means to enhance predictive performance of RT models [58,77]. Boosting of RT (referred to hereafter as BRT) employs an iterative algorithm to build a final model, sequentially adding many—typically hundreds or thousands—single RTs; a single RT is referred to as a base learner or weak learner due to having low prediction power [78]. The concept of the BRT model, graphically shown in Figure 4, is that each additional RT further reduces the prediction error [67].

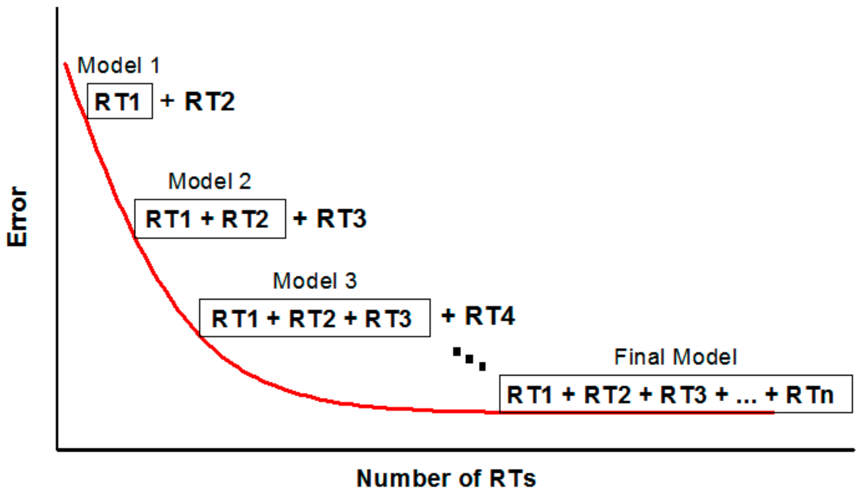

Figure 4. A graphical representation of BRT model concept.

Figure 5 displays a general schematic diagram depicting how BRT works; note that only two RTs are shown, and for simplicity, the RTs were supposed to be shallow with only one internal node. In other words, the left- or right-hand node (here the left-hand one) originating from the root node and the nodes from the single internal node are unsplit nodes (leaves). The output of a BRT model is calculated as the sum of the outputs of all trees multiplied by a learning rate.

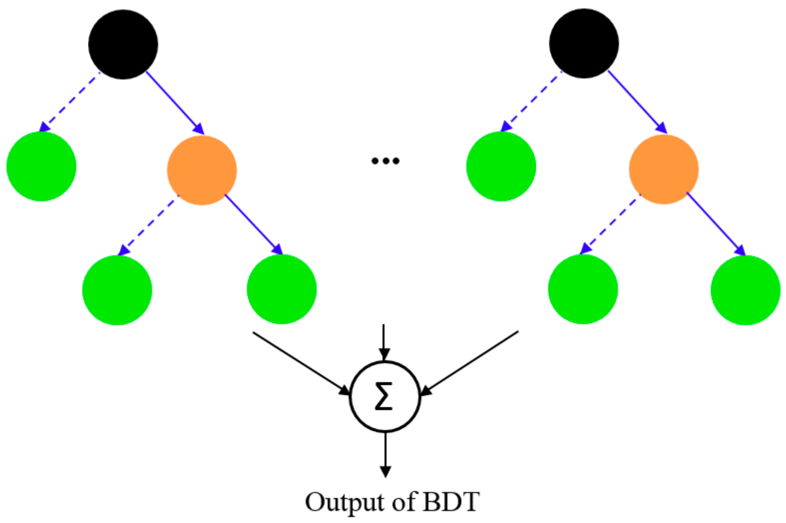

Figure 5. A general schematic diagram depicting the components of a BDT. A black circle corresponds to the root node of the tree, whereas a brown corresponds to the internal node. The green circles represent unsplit nodes (leaves). A solid blue arrow indicates that the conditional statement in the preceding node is satisfied, while a dashed blue arrow indicates that the conditional statement has failed. The circle labeled as Σ applies a summation function on its incoming signals, where a signal is defined as the weighted output of an RT (weighting coefficient known as learning rate (α) that take values smaller than 1.0).

Mathematically, the BRT's output (F) can be computed in accordance with Equation (8) [58].

$$F(x) = \sum_{k=1}^{N_t} \alpha f_k(x) \tag{9}$$

where $f_k(x)$ is the function of k-th RT built on the dataset $\{X, r\}$ where X is the predictor space and r is the vector of residuals produced from the previously grown RT. When k equals one, r equals y, where y is the vector of outputs in the training observations. In other words, the first RT is grown on the dataset $\{X, y\}$. It is worth mentioning that each tree in a BRT model has a quite different structure compared with the others as they grow on different datasets.

α is the learning rate.

N_t is the total number of RTs.

- BRT Tuning Parameters

The BRT model has several tuning parameters, which should be adjusted in order to optimize the model accuracy. The key parameters are as follows [58,67,79]:

Number of trees (N_{tr}): In case of setting N_{tr} to small values, the model may be subject to underfitting, which means that the model in unable to fit even the training dataset. In contrast, the more trees, the better the model learns. However, the training time increases, and the model would tend to overfitting. This means that the model learns noises (irregular patterns) rather than relevant patterns; in other words, the model offers good predictions on the training dataset; however, it produces a poor prediction performance on the test dataset. Hence, this parameter should be carefully controlled. One of the possible solutions to avoid under-/over-fitting is to use the CV technique, which allows selecting the optimum value of N_{tr} according to the performance of CV. For a detailed description of the CV technique, the reader is referred to the studies of Carslaw and Taylor [80], Fedotenkova [81], and Salehi and Lestari [82].

Depth of trees: The deeper the trees grow, the more the model is prone to overfitting.

Learning rate (α): It is often called step size or shrinkage factor, which takes a value in the range (0,1]; α is a measure of how fast or slow the model converges. Note that in case of setting α to a very small value (<<1), a very large value of N_{tr} is needed to attain high performance; however, this comes at a cost, which means that training the model might become computationally too expensive.

In this study, the BRT model was implemented in MATLAB using the Regression Learner App, a subdivision of the Machine Learning and Deep Learning group, which can be found by clicking on the *Apps* tab in the MATLAB toolbar. Initially, the training and testing subsets were loaded from the MATLAB workspace, and then a five-fold CV was selected as a preventative technique against overfitting.

There are some parameters, i.e., N_{tr}, *mls* and α, that are critical to the development of an accurate BRT model. For the problem under consideration here, N_{tr}, *mls*, and α varied in the following ranges: N_{tr} from 1 up to 600 (with a total number of 23 data points), *mls* from 2 to 8 with an increment step size of 1, and α from 0.1 to 1.0 with an increment step size of 0.1. It was assumed that only one of these three parameters varied at a time, and the rest of the two parameters were held constant. This assumption resulted in the creation of 1610 different BRT models for each output (y_1 and y_2), among which the best model was picked as the one that yielded the least validation error; the smaller the validation error, the better the model generalization power. The performance of the BRT model was assessed using the statistical indices (R^2, *NSE*, and *RMSE*) given by Equations (5)–(7).

2.2.3. Multiple Regression Analysis

Multiple regression is a statistical technique that is widely used in numerous problems to establish a relationship between a series of predictor variables and a dependent variable called output variable. Consider output variable y, which depends on "m" predictor variables ($x_1, x_2, \ldots, x_m$). For "n" observation, if the relationship between these variables is given by Equation (10), it is called a multiple linear regression (MLR).

$$y_i = a_0 + \sum_{j=1}^{m} a_j x_{ji} \qquad i = 1, 2, \ldots, n \tag{10}$$

where x_{ji} denotes the i-th observation of x_j; y_i represents i-th observation of the response; "a_0" is a constant (the model intercept); and "a_j's" are the linear regression coefficients; these coefficients can be estimated using the LSE method. The computational procedure to determine the regression coefficients has been well described in [83,84].

An extended form of MLR is the multiple nonlinear regression (MnLR) in which an output variable is described by a function that is a nonlinear combination of the predictors. Though the MLR is often used, there are so many problems in which the application of a MnLR is more suitable because the output behaves nonlinearly with changes in the predictors. A challenge that is often associated with the use of MnLR is what type of nonlinear function gives the best description of the relationship between predictors and output, because MnLR can take many different functional forms.

In this study, MLR analysis was first performed to appraise the performance of the conventional regression method for predicting the performance of the UF membrane process in the separation of XR. The training subset was fitted to an MLR using MATLAB to estimate the MLR coefficients. Then, the testing subset was used to assess the prediction accuracy of the fitted MLR in terms of R^2, *RMSE*, and *NSE* given by Equations (5) and (6). Second, MnLR analysis was performed; to achieve this, the curve fitting in the DataFit software (trial version 9.1.32, Copyright© 1995–2014, Oakdale Engineering, USA) was used. The training subset was fitted to various MnLR models (a total number of 143 different models) to estimate the MnLR models' coefficients. The fitted models were then fed with the testing subset to pick the best model as the one with the highest prediction accuracy (the higher R^2, NSE, and the smaller *RMSE*).

2.2.4. Sensitivity Analysis

A sensitivity analysis was performed to investigate how the changes in input variables influence the output of a given model. In this regard, a tornado chart (sometimes called butterfly chart) was constructed, which is based on one-at-a-time input variation [85–87]. The tornado chart demonstrates the sensitivity of output in terms of "swings" displayed as bars; a swing corresponding to the input variable X_i is defined as the absolute difference between the output values (Y_i) calculated at the values of $X_{i\ \max}$ and $X_{i\ \min}$, while all other input variables (X_j, j # i) are held at their median values. The created swings are sorted horizontally in descending order of their width so that the overall chart (i.e., the stacked horizontal bars) looks similar to the shape of a typical tornado. The wider the swing, the higher the model sensitivity to the input variable tested; in other words, the most important input variable that affects the model output is at the top of the chart, and the least important is at the bottom. Figure 6 illustrates the procedure of building a tornado chart for a given output variable (Y) as a function of input variables X_i (i =1 to n). The blue and red bars—extended from the vertical dash line (Y value when medians are used for all input variables)—represent the values of $Y_i|X_{i,\ min}$ and $Y_i|X_{i,max}$, respectively. The green bars represent swings, which are then categorized so that the widest bar is placed at the top of the chart, followed by the second-widest bar, and so on.

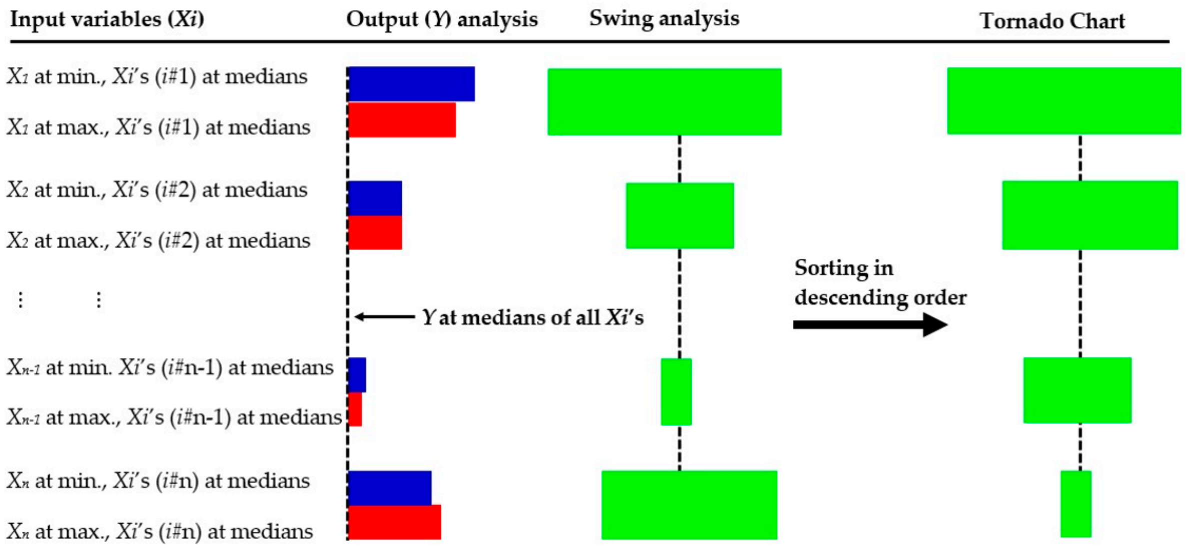
Input variables (Xi)
Output (Y) analysis
Swing analysis
Tornado Chart
X1 at min., Xi's (i#1) at medians
X1 at max., Xi's (i#1) at medians
X2 at min., Xi's (i#2) at medians
X2 at max., Xi's (i#2) at medians
Y at medians of all Xi's
Sorting in descending order
Xn-1 at min. Xi's (i#n-1) at medians
Xn-1 at max., Xi's (i#n-1) at medians
Xn at min., Xi's (i#n) at medians
Xn at max., Xi's (i#n) at medians

Figure 6. Graphical illustration of sensitivity analysis using tornado chart (note: the swing associated with the input variable X_i (i = 1 to n) is the absolute ($Y_i|X_{i,\ max} - Y_i|X_{i,min}$) while all other input variables X_j (j # i) are set at their median values).

3. Results and Discussions

3.1. Evaluation of ANFIS-GP Model

Figure 7 shows the validation performance, in terms of R^2 and $RMSE$, of the ANFIS-GP when the model output was the normalized flux. It is clear from Figure 7 that changes in the type of input MF influence the validation results. The best input MF, out of six MFs (i.e., *trimf*, *trapmf*, *gbellmf*, *gaussmf*, *gauss2mf*, and *psigmf*) examined to fit the training subset, was selected as the one yielded the highest R^2 and the least $RMSE$. As seen in Figure 7, the use of *trimf* offered the best results (R^2 = 0.9649, $RMSE$ = 0.0154). Hereafter, the ANFIS-GP model with normalized flux as the model output and *trimf* as the best input MF will be referred to as ANFIS-GP1.

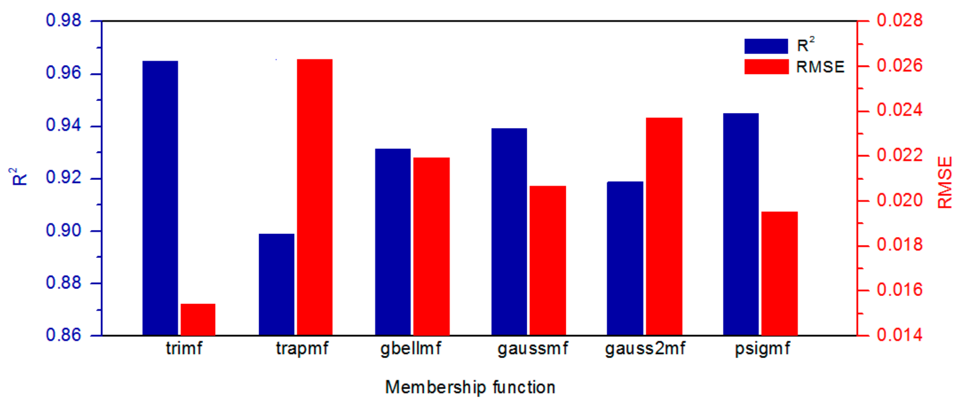

Figure 7. Validation results for the ANFIS-GP (with normalized flux as the model output) as a function of the types of membership functions.

The structure of the ANFIS-GP1 is graphically represented in Figure 8. There are three nodes in the input layer, corresponding to the three inputs. In the second layer, two nodes are connected to each input node (in total six nodes), which correspond to two MFs, in the form of *trimf*, for each input. The third layer contains eight (i.e., 2^3) nodes equivalent to eight if-then rules. The rules processing results are given by eight nodes in the next layer, and then a weighted average method is applied in order to obtain the output. A detailed description of the fuzzy if-then rules, and the optimal values of the input MFs' parameters (i.e., antecedent parameters) and output MFs' parameters (i.e., consequent parameters) of the ANFIS-GP1 are given in the Supplementary Material (Tables S1–S3).

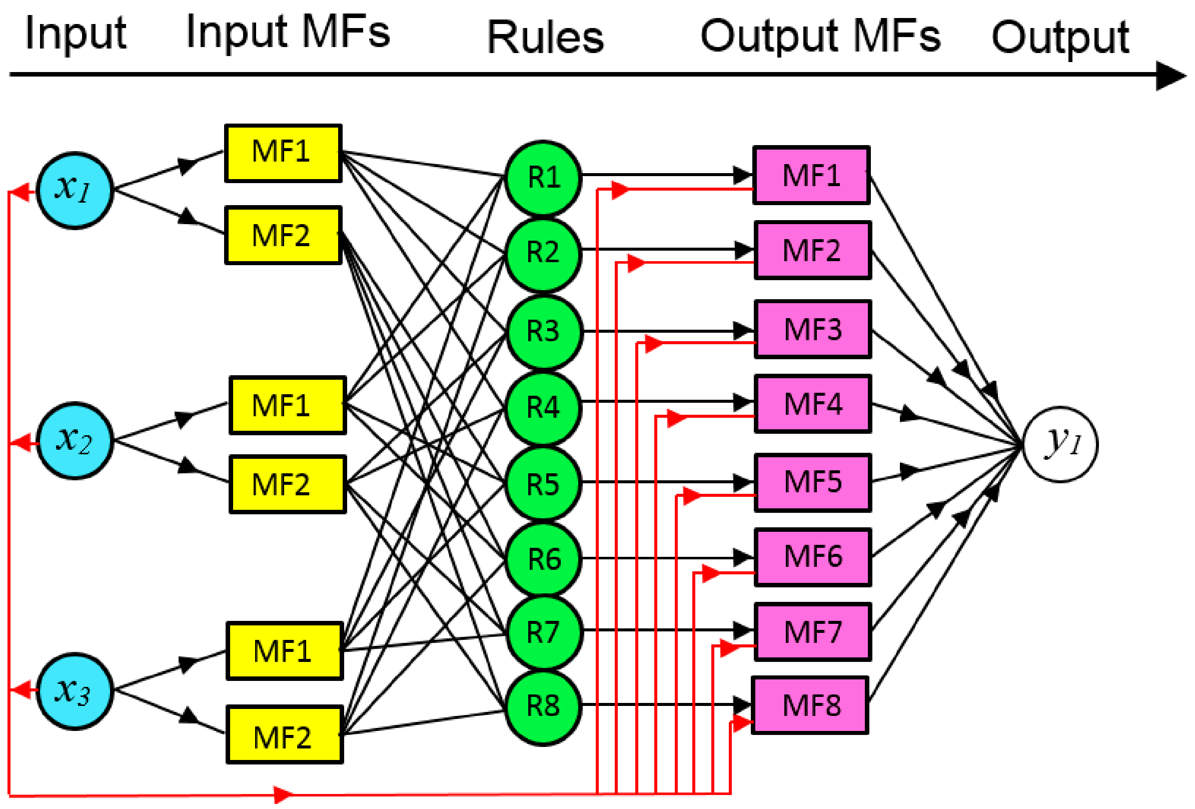

Figure 8. Structure of the ANFIS-GP1 (x_1: FT, x_2: TMP, x_3: CFV (refer to Table 1 for FT, TMP, and CFV values); y_1: predicted normalized flux; "MF" and "R" denote membership function and fuzzy if-then rule, respectively (refer to Tables S1–S3 in the Supplementary Materials for a detailed description of the fuzzy if-then rules and the optimal values of the inputs/output MFs' parameters).

Regarding the ANFIS-GP model with xylitol concentration as the model output, *gaussmf* was selected as the best input MF, which gave R^2 and *RMSE* values equal 0.8366 and 0.2394, respectively (see Figure 9). Hereafter, the ANFIS-GP model with xylitol concentration as the model output and *gaussmf* as the best input MF will be referred to as ANFIS-GP2. The structure of the ANFIS-GP2 is the same as the structure of the ANFIS-GP1 shown in Figure 8, except that the input MFs were in the form of *gaussmf* instead of *trimf*. A detailed description of the fuzzy if-then rules and the optimal values of the input/output MFs' parameters of the ANFIS-GP2 are given in the Supplementary Materials (Tables S4–S6).

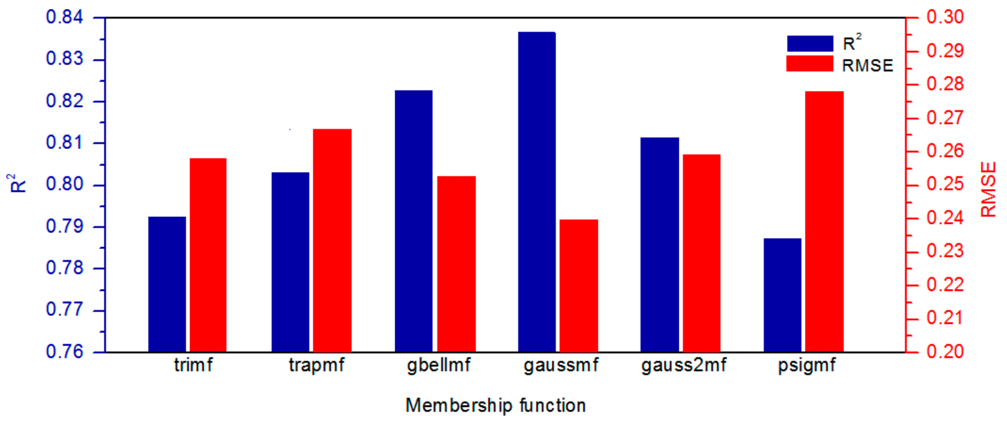

Figure 9. Validation results for the ANFIS-GP (with xylitol concentration as the model output) as a function of the types of membership functions.

To visualize the prediction accuracy of the ANFIS-GP1 and ANFIS-GP2 on the entire dataset (i.e., both the training and testing subsets), the scattered diagram—in which the measured data plotted against the models' predicted values—was constructed (Figure 10); a distinction first needs to be made between the 1:1 line and the line of best fit (regression line). The 1:1 line (more often called the 45° line or 100% correlation line) is often used as a reference when comparing two data sets in a two-dimensional scatter plot. If the corresponding data points from the two data sets are equal to each other, the corresponding scatters lie exactly on the 1:1 line. The line of best fit refers to a straight line through a scatter plot of data points that best represents the relationship between the data points. The equation for such a line is typically created using the least squares method.

It can be observed from Figure 10 that the data points on the plots have been dispersed well enough around the 1:1 line with R^2 and *NSE* values of 0.9845 and 0.9843 for ANFIS-GP1 and of 0.9459 and 0.9453 for the ANFIS-GP2, respectively. These results indicate the high prediction accuracy of the models as they can satisfactorily explain the variability in outputs; the ANFIS-GP1 and ANFIS-GP2 do not explain only about 1.6% and 5.5%, respectively, of the total variability in the outputs.

3.2. Evaluation of BRT Model

Figure 11 shows the validation error of the BRT model (with the normalized flux as the output) as a function of *mls* and N_{tr}, while α was set to its optimal value (α = 0.3) that was obtained via trial-and error-method. As seen in Figure 11, the smallest error (*RMSE* = 0.0184) was achieved when *mls* = 5 and N_{tr} = 200 (note that further increase in N_{tr}, beyond 200, showed a very little effect on the decrease in error); hereafter, the optimal BRT model with the normalized flux as the output will be referred to as BRT1. As an example, the structure of the tree no. 200 in the trained BRT1 is illustrated in Figure 12.

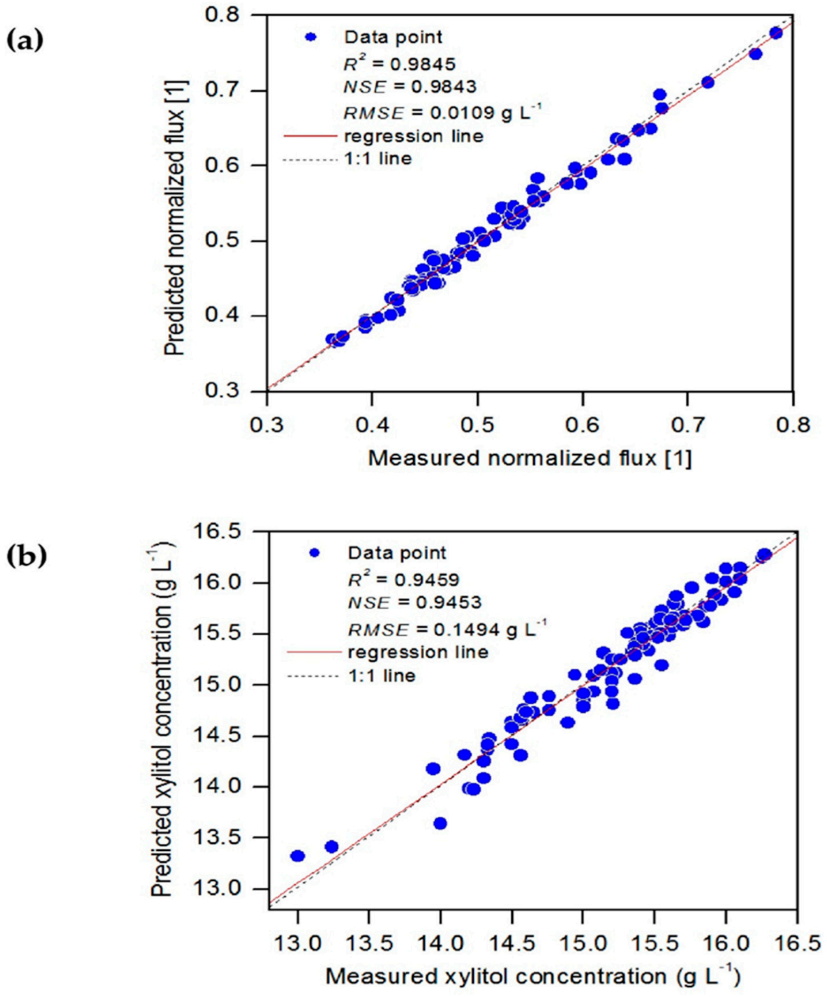

Figure 10. Scatter plot of the measured and predicted (**a**) normalized flux using ANFIS-GP 1 and (**b**) xylitol concentration using the ANFIS-GP2 on the entire dataset.

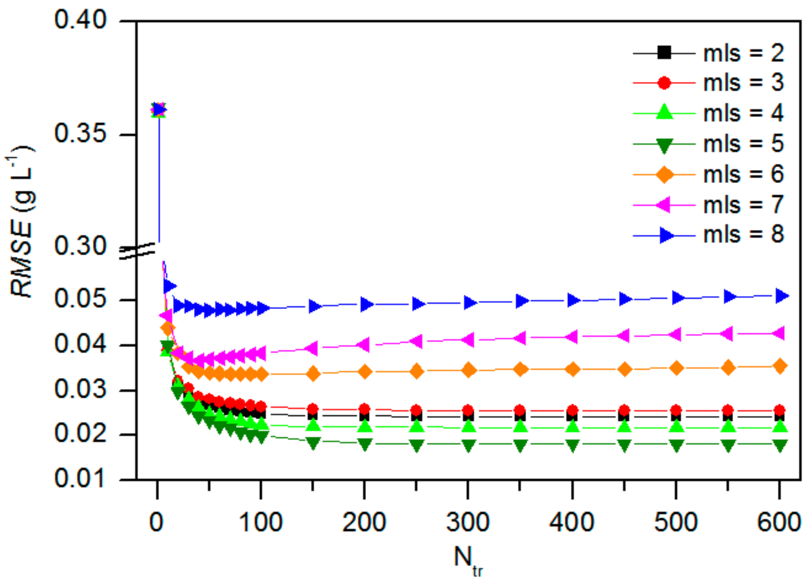

Figure 11. Validation *RMSE* curves for the BRT model (with normalized flux as the output) as a function of N_{tr} and *mls* (α = 0.3).

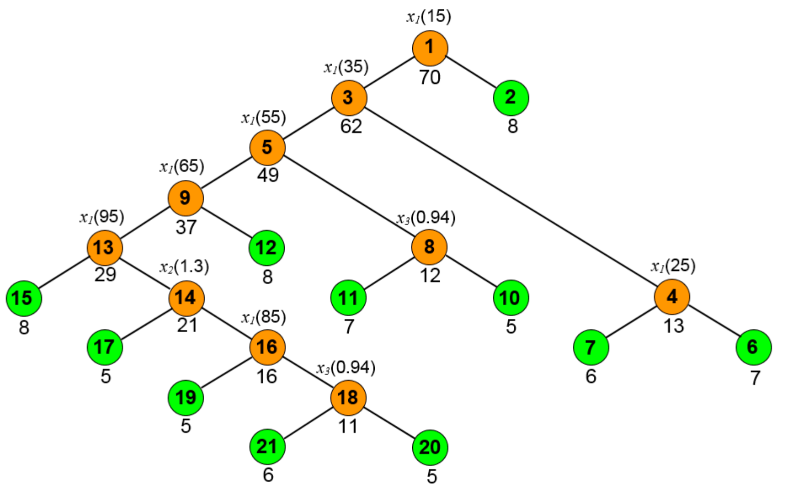

Figure 12. Graphical representation of the structure of weak learner (tree) no. 200 in the trained BRT1. (Notes: the total number of nodes in the tree equals 21; a brown circle indicates a node that can be branched out, whereas a green circle indicates a terminal node (called unsplit node or leaf node) that cannot be branched out; the number inside the nodes represent node number; the number under the each node represents the size of the node (the number of observations that satisfy the condition for the node); x_i on the top of the nodes denotes the cut predictor; and the numbers in parentheses represent the cut-off point (c) associated with x_i (for each node, the right-hand branch is chosen if $x_i \geq c$, and the left-hand branch is chosen if $x_i < c$).

Figure 13 shows the validation error of the BRT model (with xylitol concentration as the output) as a function of *mls* and N_{tr}, while α was set to its optimal value (α = 0.1). It can be observed in Figure 13 that the smallest error (*RMSE* = 0.2566) was obtained when *mls* = 2 and N_{tr} = 80; hereafter, the optimal BRT model with xylitol concentration as the output will be referred to as BRT2. As an example, the structure of tree no. 80 in the trained BRT2 is illustrated in Figure 14.

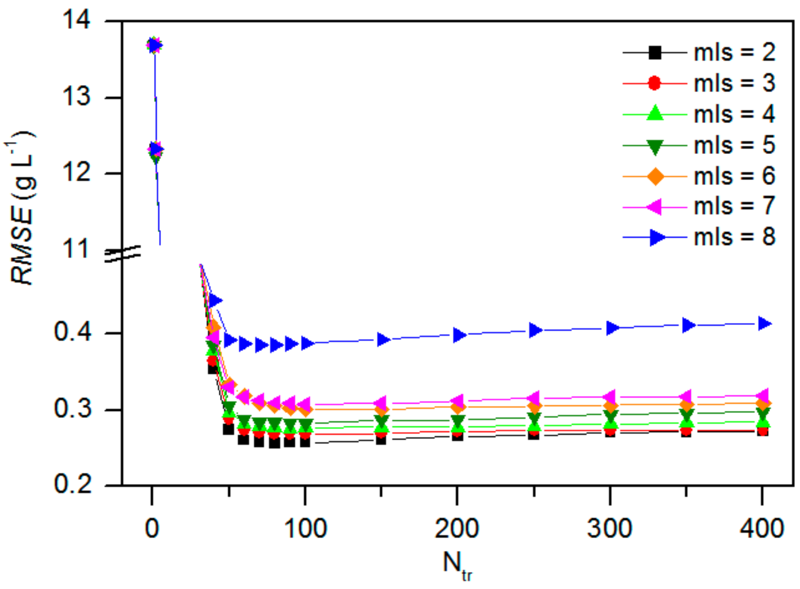

Figure 13. Validation RMSE curves for the BRT model (with xylitol concentration as the output) as a function of N_{tr} and *mls* (α = 0.1).

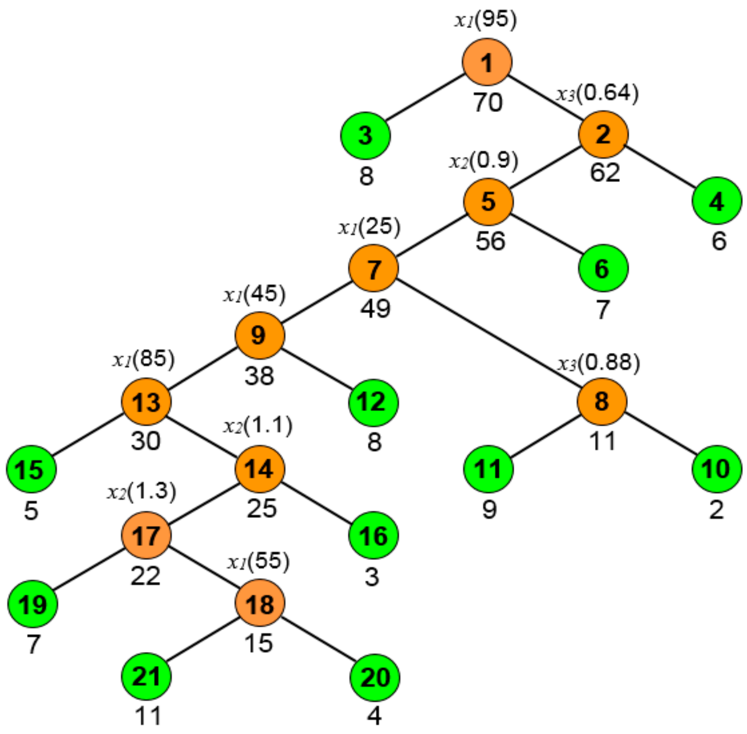

Figure 14. Graphical representation of the structure of weak learner (tree) no. 80 in the trained BRT2 (see the notes in the caption of Figure 12).

Figure 15 shows the prediction accuracy of models BRT1 and BRT2 on the entire dataset (i.e., both the training and testing subsets) as a scatter plot of the measured and the model-predicted values. From Figure 15, the BRT1 and BRT2 offered an R^2 value of 0.9944 and 0.9669, respectively, and an *NSE* value of 0.9937 and 0.9664, respectively, indicating that the established models could predict the output variables well; the BRT1 and BRT2 do not explain only about 0.6% and 3.3%, respectively, of the total variability in the outputs.

3.3. *Evaluation of Regression Models*

The MLR and MnLR equations obtained using the training subset have the following form:

$$\begin{gathered}\text{MLR1:}\\ y_1 = -0.0019x_1 + 0.2167x_2 + 0.1745x_3 + 0.1845\end{gathered} \tag{11}$$

$$\begin{gathered}\text{MLR2:}\\ y_2 = -0.0150x_1 + 1.5183x_2 + 0.9941x_3 + 13.2684\end{gathered} \tag{12}$$

$$\text{MnLR1}: y_1 = -0.1547 + \frac{2.3579}{\ln(A)} - \frac{16.0759}{A^2} A = 0.3905x_1 - 35.6471x_2 - 36.1295x_3 + 92.6519 \tag{13}$$

$$\text{MnLR2}: y_2 = 13.1374 + \frac{7.0660 \times A}{\ln(A)} + \frac{0.6351}{\sqrt{A}} A = -0.1041x_1 + 9.9640x_2 + 8.2077x_3 - 5.3888 \tag{14}$$

where y_1 and y_2 represent the outputs (the normalized flux and xylitol concentration, respectively); x_1: FT; x_2: TMP; and x_3: CFV (refer to Table 1 for FT, TMP, and CFV values).

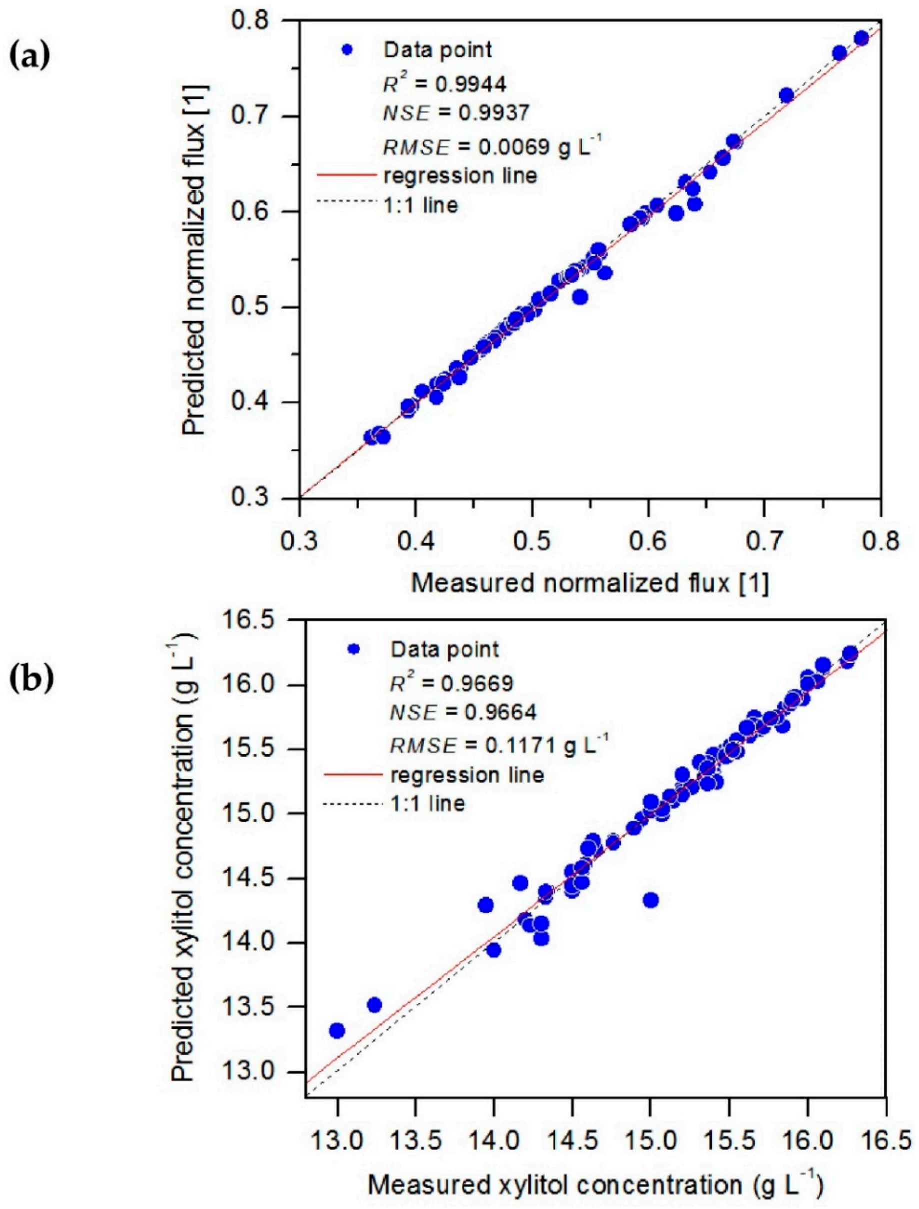

Figure 15. Scatter plot of the measured and predicted (**a**) normalized flux using BRT1 and (**b**) xylitol concentration using the BRT2 on the entire dataset.

Figure 16 shows a scatter plot of the measured and the predicted values using the MLR1, MLR2, MnLR1, and MnLR2 models on the entire dataset (i.e., both the training and testing subsets). From this figure, the MLR 1 and MLR2 produced a relatively poor linear correlation with R^2value of 0.8149 and 0.7519, and *NSE* values of 0.8145 and 0.7507, respectively. It can also be seen from Figure 16 that the MnLR1 and MnLR2 produced a linear correlation with R^2 value of 0.8653 and 0.8199, and *NSE* values of 0.8641 and 0.8190, respectively. These findings indicate that (non)linear regression could not be accurate enough to correctly predict the normalized flux and xylitol concentration for the system under consideration in this study.

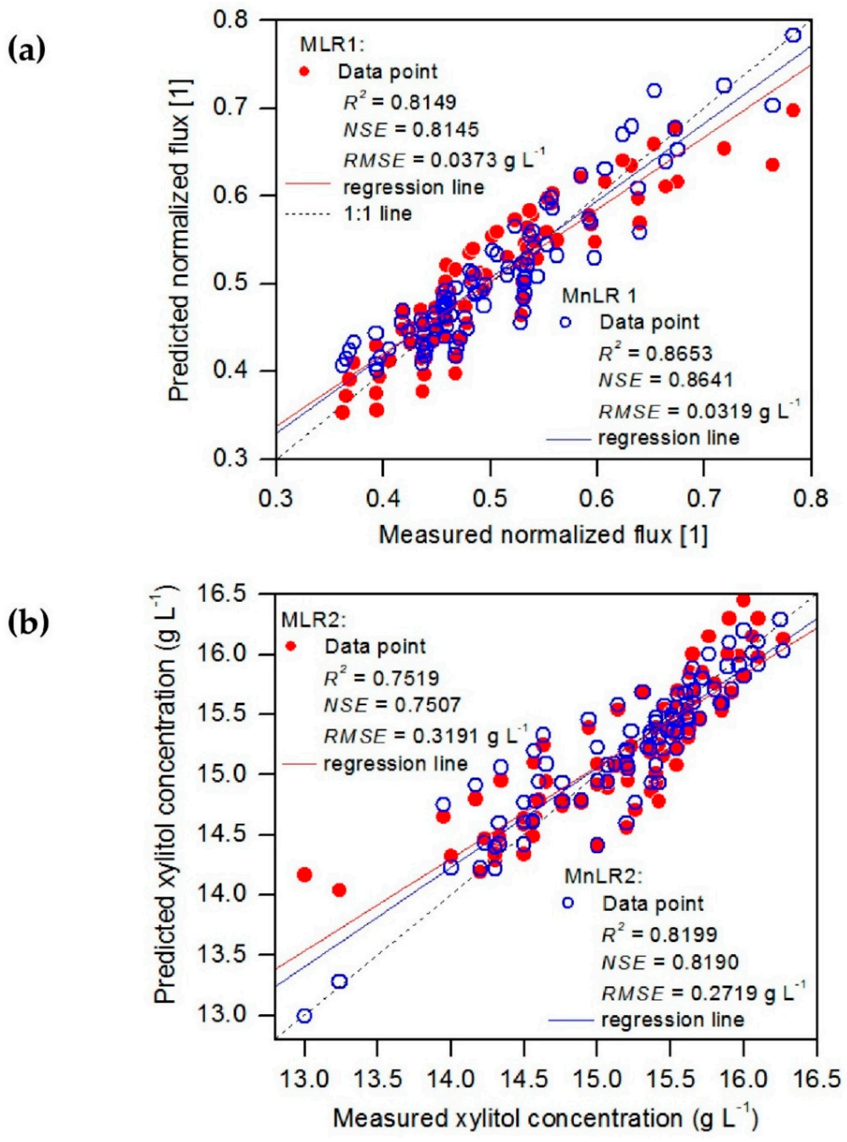

Figure 16. Scatter plot of the measured and predicted (**a**) normalized flux using M(n)LR1 and (**b**) xylitol concentration using M(n)LR2 on the entire dataset.

3.4. Comparison of the Models

A performance comparison of the models developed (i.e., ANFIS-GP, BRT, and M(n)LR) on the entire dataset is summarized in Table 2; three statistical indices (R^2, *NSE*, and *RMSE*) were used to make the comparison. It can be seen from Table 2, when the normalized flux was used as the model's output (called ANFIS-GP1 model), an R^2 value of 0.9845 and an *NSE* value of 0.9843—corresponding to an *RMSE* value of 0.0109—were obtained. In case of using xylitol concentration as the model's output (called ANFIS-GP2), R^2, *NSE*, and *RMSE* were found to be 0.9459, 0.9453, and 0.1494 g L^{-1}, respectively. These findings demonstrate that the ANFIS-GP model performs better than M(n)LR models, which yielded R^2, *NSE*, and *RMSE* values in the ranges of 0.7519–0.8653, 0.7507–0.8641, and 0.0319–0.3191, respectively. In addition to high prediction accuracy of the ANFIS-GP model, its development was straightforward because the number of input variables—for the problem presented in this study—was as small as three inputs. However, in case of a given problem involving many inputs, ANFIS-GP modeling could not be a feasible method because the number of fuzzy rules increases exponentially with an increase in the number of inputs, and consequently, this leads to an increase in the number of trainable parameters so that the model becomes computationally expensive. For instance, an ANFIS-GP model fed with n input variables would generate M^n fuzzy rules, where M denotes the number of MFs for each input.

Table 2. Performance comparison of the developed models in this study.

Model Output	Model	R^2	*NSE*	Unit	*RMSE*
Normalized flux	BRT1	0.9944	0.9937	1	0.0069
	ANFIS-GP1	0.9845	0.9843	1	0.0109
	MnLR1	0.8653	0.8641	1	0.0319
	MLR1	0.8149	0.8145	1	0.0373
Xylitol concentration	BRT2	0.9669	0.9664	1	0.1171
	ANFIS-GP2	0.9459	0.9453	1	0.1494
	MnLR2	0.8199	0.8190	1	0.2719
	MLR 2	0.7519	0.7507	1	0.3191

When comparing the performance of the developed ANFIS-GP and BRT models (see Table 2), it can be observed that the BRT produced an R^2 value of 0.9669–0.9944 and an *NSE* value of 0.9664–0.9937, which are greater than those obtained using ANFIS-GP model. In addition, the BRT model gave an *RMSE* value of 0.0069–0.1171, which is approximately 22–37% smaller than that produced by the ANFIS model.

Overall, it can be concluded that, among the models developed, the BRT model offered the highest R^2 and *NSE* values and produced the least error, and hence, it is the best choice for predicting the normalized flux and xylitol concentration for the system under consideration in this study. However, it should be noted that BRT models suffer from two major limitations: (i) they are greatly unstable; a small change in the training dataset can lead to form a completely different structure yielding different results, and (ii) such models are prone to overfit the training data and to be computationally expensive. In order to overcome this issue, caution must be taken to properly set the leaf node size (i.e., the number of data points assigned to each leaf node) for any given problem.

3.5. Sensitivity Analysis for Model BRT

The sensitivity analysis, in the form of tornado chart, for a model BRT that produced a smaller *RMSE* (higher R^2) in comparison with models ANFIS-GP, MnLR, and MLR (see Table 2) is shown in Figure 17. As seen in Figure 17a (tornado chart for model BRT1), the widest swing is associated with input variable X_1, which is 0.199. This implies that the highest impact on the model output comes from variable X_1, followed by input variables X_2 and X_3 showing a lower impact, with swing widths of 0.165 and 0.110, respectively. Figure 16b (tornado chart for model BRT2), similar to Figure 16a, demonstrates that the input variable X_1 has the highest impact on the model output with a swing width of 1.60 g L^{-1} compared to those of X_2 and X_3 with swing widths of 1.00 g L^{-1} and 0.77 g L^{-1}, respectively.

3.6. Model Calibration and Validation Need

Although the modelling approaches and the models developed have demonstrated such a meaningful predictive power, a need for model calibration and validation has to be addressed. The machine learning models presented in this study were learned from and used to make predictions on the data taken from the experimental runs conducted by our previous work [20]. As the size of the original (raw) dataset was considered small (only 90 observations, called patterns), the authors initially decided to randomly split the raw dataset into only two disjoint subsets (i.e., training (T_r): 70 patterns, and testing (T_s): 20 patterns). Then, the cross validation (a very common, useful, and proven technique in the field of data mining) was performed to avoid overfitting the training data and to tune the models' parameters. Then, the prediction accuracy of the developed (trained) model was assessed on the T_s subset.

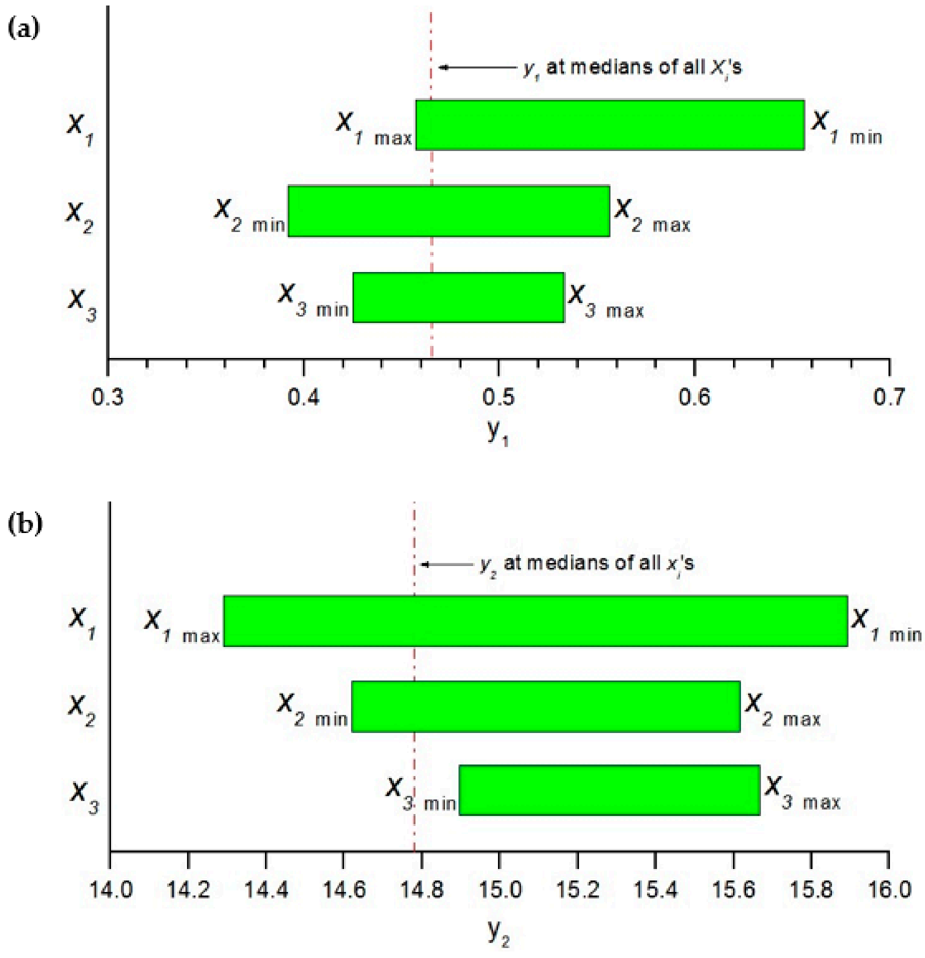

Figure 17. Sensitivity analysis for the assessment of input variable importance on the estimation of (**a**) y_1 using model BRT1 and (**b**) y_2 using model BRT2. (Notes: the minimum, median, and maximum values of the input variables X_1, X_2, and X_3 are (10, 55, and 100), (0.8, 1.2, and 1.6), and (0.58, 1.06, and 1.20), respectively; X_1: filtration time (min); X_2: transmembrane pressure (bar); X_3: cross-flow velocity (cm s^{-1}); y_1: normalized flux (i.e., ratio of the permeate flux to the pure water flux); y_2: xylitol concentration (g L^{-1}); vertical dash line: y_1 (and y_2) value at medians of all input variables.

Each pattern in the T_s subset was never seen during the training phase. In other words, at least one of the data points (input variables in the pattern) of the j-th pattern (j = 1—20) in the T_s subset had a different value compared to that of the i-th pattern (i =1–70) in the T_r subset. By this, the authors mean that the T_r and T_s are composed of independent patterns. In other words, a reasonable judgment on prediction capability of the trained model can be made by testing the model on the T_s subset.

If any data point of the j-th pattern in the T_s does not have the same value as found for that of i-th pattern in the T_r, the T_r and T_s subsets would be definitely independent, and a better judgment can be made on the prediction accuracy of the model. Recognizing the limited experimental runs in our work, we therefore recommend additional datasets of the truly independent experiments be performed when the mathematical/statistical modeling works are to be initiated by the existing data.

4. Conclusions

This study, for the first time, demonstrated the application of a machine learning (ML) modeling approach to predict the performance of a cross-flow ultrafiltration (CF-UF) membrane for the separation of xylose reductase (XR) from the reaction mixture obtained in the production of xylitol, as a function of cross-flow velocity, transmembrane pressure, and filtration time. Two types of ML-based models, including a grid partitioning-based adaptive neuro-fuzzy inference system (ANFIS-GP) and boosted regression trees (BRT) were developed, validated, and tested. The prediction accuracy of the BRT and ANFIS-GP models was compared as well as with that of the (non)linear regressions. The modeling results clearly indicated the BRT model yielded a superior predictive performance on membrane permeability with an excellent R^2 value of 0.994 and on the amount of xylitol produced with R^2 value of 0.966. This implies that the BRT could be considered as a valuable computational tool for predicting the performance of CF–UF membrane for enzymatic production of xylitol. Future research needs to be directed towards other types of commercial enzymes and on a scale-up study to determine the suitability of the separation process in industrial applications, particularly for long-term usage with respect to the production of high yield and productivity.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/su15054245/s1, Table S1: Fuzzy if-then rules for the ANFIS-GP1 used in this study; Table S2: Antecedent parameters of the ANFIS-GP1 used in this study; Table S3: Consequent parameters of the ANFIS-GP1 used in this study; Table S4: Fuzzy if-then rules for the ANFIS-GP2 used in this study; Table S5: Antecedent parameters of the ANFIS-GP xylitol model used in this study; Table S6: Consequent parameters of the ANFIS-GP xylitol model used in this study.

Author Contributions: Writing—original draft, validation, formal analysis, interpretation, writing—review and editing, R.S.; Writing—original draft, investigation, formal analysis, resources, writing—review and editing, S.K.; Writing—review and editing, M.N.; Supervision, project administration, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by a Postdoctoral Fellowship from Prince of Songkla University, Thailand. This study was also supported by research grant-(RDU2203102) from the University of Malaysia Pahang, Malaysia.

Data Availability Statement: All data supporting the conclusions of this study are included in the paper or in its Supplementary Materials.

Acknowledgments: The authors thank the support from Biogas and Biorefinery Laboratory at the Faculty of Engineering and PSU Energy Systems Research Institute, Prince of Songkla University, Thailand.

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

Nomenclature

AKR	aldo-keto reductase
ALR	aldehyde reductase
ANFIS-GP	adaptive neuro-fuzzy inference system based on grid partitioning on the input space
ANN	artificial neural network
A_i, B_i	fuzzy sets pertaining to input variable x_i
a_i, b_i, c_i	antecedent parameters corresponding to the i-th output function
BP	backpropagation
BRT	boosted regression tree
c	cut-off point
CF-UF	cross-flow ultrafiltration
CV	cross-validation
FBPNN	feedforward backpropagation neural network
FIS	fuzzy inference system
f_i	output function of the i-th fuzzy rule
GP	grid partitioning
GUI	graph user interface

gaussmf	Gaussian membership function
gauss2mf	combination of two Gaussian membership function
gbellmf	generalized bell-shaped membership function
LSE	least squared estimation
M	number of membership functions assigned to each input
MF	membership function
MLR	multiple linear regression
mls	minimum leaf size
MnLR	multiple nonlinear regression
N	number of inputs to the ANFIS model
N_A	number of antecedent parameters
N_C	number of consequent parameters
N_R	number of fuzzy rules
n	total number of observations
NADPH	reduced form of nicotinamide adenine dinucleotide phosphate (donor of hydrogen atoms)
NSE	Nash–Sutcliffe efficiency coefficient
P	number of input MFs
psigmf	product of two sigmoidal membership function
R^2	coefficient of determination
$RMSE$	root mean squared error
r	vector of residuals
RT	regression tree
SSR	sum of squared residuals
trapmf	trapezoidal membership function
trimf	triangular membership function
UF	ultrafiltration
w_i	synaptic weight assigned to the signal leaving the i-th neuron with the hid den layer
XR	xylose reductase
x_1	FT (filtration time; min)
x_2	TMP (transmembrane pressure; bar)
x_3	CFV (cross-flow velocity; cm s^{-1})
y_1	normalized flux (i.e., ratio of the permeate flux to the pure water flux)
y_2	xylitol concentration (g L^{-1})
$y_i{}^p$	model-predicted value for the i-th observation
$\overline{y}$	averaged value of y_i
$y\|_{x_i,\ min}$	output value computed at x_i set to its minimum value
$y\|_{x_i,\ max}$	output value computed at x_i set to its maximum value
$\mu_{ji}\ (x_i)$	membership function of j-th fuzzy set associated with input variable x_i
α	learning rate (step size or shrinkage factor)

References

1. Jez, J.M.; Penning, T.M. The aldo-keto reductase (AKR) superfamily: An update. *Chem. Biol. Interact.* **2001**, *130–132*, 499–525. [CrossRef]
2. Krishnan, S.; Nasrullah, M.; Kamyab, H.; Suzana, N.; Munaim, M.S.A.; Wahid, Z.A.; Ali, I.H.; Salehi, R.; Chaiprapat, S. Fouling characteristics and cleaning approach of ultrafiltration membrane during xylose reductase separation. *Bioprocess Biosyst. Eng.* **2022**, *45*, 1125–1136. [CrossRef]
3. Azizah, N. Biotransformation of xylitol production from xylose of lignocellulose biomass using xylose reductase enzyme: Review. *J. Food Life Sci.* **2019**, *3*, 103–112.
4. Umai, D.; Kayalvizhi, R.; Kumar, V.; Jacob, S. Xylitol: Bioproduction and applications-A review. *Front. Sustain.* **2022**, *3*, 826190. [CrossRef]
5. Lugani, Y.; Sooch, B.S. Fermentative production of xylitol from a newly isolated xylose reductase producing *Pseudomonas putida* BSX-46. *LWT-Food Sci. Technol.* **2020**, *134*, 109988. [CrossRef]

6. Narisetty, V.; Castro, E.; Durgapal, S.; Coulon, F.; Jacob, S.; Kumar, D.; Awasthi, M.K.; Pant, K.K.; Parameswaran, B.; Kumar, V. High level xylitol production by *Pichia fermentans* using non-detoxified xylose-rich sugarcane bagasse and olive pits hydrolysates. *Bioresour. Technol.* **2021**, *342*, 126005. [CrossRef]
7. Saha, B.C.; Kennedy, G.J. Production of xylitol from mixed sugars of xylose and arabinose without co-producing arabitol. *Biocatal. Agric. Biotechnol.* **2020**, *29*, 101786. [CrossRef]
8. Lugani, Y.; Puri, M.; Sooch, B.S. Recent insights, applications and prospects of xylose reductase: A futuristic enzyme for xylitol production. *Eur. Food Res. Technol.* **2021**, *247*, 921–946. [CrossRef]
9. Ho, N.W.Y.; Lin, F.P.; Huang, S.; Andrewst, P.C.; Tsao, G.T. Purification, characterization, and amino terminal sequence of xylose reductase from *Candida shehatae*. *Enzym. Microb. Technol.* **1990**, *12*, 33–39. [CrossRef]
10. Eryasar, K.; Karasu-Yalcin, S. Evaluation of some lignocellulosic byproducts of food industry for microbial xylitol production by *Candida tropicalis*. *3 Biotech* **2016**, *6*, 202. [CrossRef]
11. Ariyan, M.; Uthandi, S. Xylitol production by xylose reductase over producing recombinant *Escherichia coli* M15. *Madras Agric. J.* **2019**, *106*, 205–209. [CrossRef]
12. Kim, S.; Lee, J.; Sung, B.H. Isolation and characterization of the stress-tolerant *Candida tropicalis* YHJ1 and evaluation of its xylose reductase for xylitol production from acid pre-treatment wastewater. *Front. Bioeng. Biotechnol.* **2019**, *7*, 12. [CrossRef]
13. Walsh, M.K.; Khliaf, H.F.; Shakir, K.A. Production of xylitol from agricultural waste by enzymatic methods. *Am. J. Agric. Biol. Sci.* **2018**, *13*, 1–8. [CrossRef]
14. Kklaif, H.F.; Nasser, J.M.; Shakir, K.A. Production of xylose reductase and xylitol by *Candida guilliermondii* using wheat straw hydrolysates. *Iraqi J. Agric. Sci.* **2020**, *51*, 1653–1660.
15. Quehenberger, J.; Reichenbach, T.; Baumann, N.; Rettenbacher, L.; Divne, C.; Spadiut, O. Kinetics and predicted structure of a novel xylose reductase from *Chaetomium thermophilum*. *Int. J. Mol. Sci.* **2019**, *20*, 185. [CrossRef]
16. Mouro, A.; dos Santos, A.A.; Agnolo, D.D.; Gubert, G.F.; Bon, E.P.S.; Rosa, C.A.; Fonseca, C.; Stambuk, B.U. Combining xylose reductase from *Spathaspora arborariae* with xylitol dehydrogenase from *Spathaspora passalidarum* to promote xylose consumption and fermentation into xylitol by *Saccharomyces cerevisiae*. *Fermentation* **2020**, *6*, 72. [CrossRef]
17. Woodyer, R.; Simurdiak, M.; van der Donk, W.A.; Zhao, H. Heterologous expression, purification, and characterization of a highly active xylose reductase from *Neurospora crassa*. *Appl. Environ. Microbiol.* **2005**, *71*, 1642–1647. [CrossRef]
18. Mueller, M.; Wilkins, M.R.; Banat, I.M. Production of xylitol by the thermotolerant *Kluyveromyces marxianus* IMB strains. *Bioprocess. Biotech.* **2011**, *1*, 1000102e. [CrossRef]
19. Dasgupta, D.; Ghosh, D.; Bandhu, S.; Agrawal, D.; Suman, S.K.; Adhikari, D.K. Purification, characterization and molecular docking study of NADPH dependent xylose reductase from thermotolerant *Kluyveromyces* sp. IIPE453. *Process Biochem.* **2016**, *51*, 124–133. [CrossRef]
20. Krishnan, S.; Suzan, B.N.; Wahid, Z.A.; Nasrullah, M.; Munaim, M.S.A.; MD Din, M.F.B.; Taib, S.M.; Li, Y.Y. Optimization of operating parameters for xylose reductase separation through ultrafiltration membrane using response surface methodology. *Biotechnol. Rep.* **2020**, *27*, e00498. [CrossRef]
21. Desiriani, R.; Kresnowati, M.T.A.P.; Wenten, I.G. Membrane-based downstream processing of microbial xylitol production. *Int. J. Technol.* **2017**, *8*, 1393–1401. [CrossRef]
22. Conidi, C.; Drioli, E.; Cassano, A. Membrane-based agro-food production processes for polyphenol separation, purification and concentration. *Curr. Opin. Food Sci.* **2018**, *23*, 149–164. [CrossRef]
23. Cordova, A.; Astudillo, C.; Illanes, A. Membrane technology for the purification of enzymatically produced oligosaccharides. In *Separation of Functional Molecules in Food by Membrane Technology*; Elsevier: Amsterdam, The Netherlands, 2019; pp. 113–153.
24. Mukherjee, S. Isolation and purification of industrial enzymes: Advances in enzyme technology. *Adv. Enzym. Technol. (Biomass Biofuels Biochem.)* **2019**, 41–70. [CrossRef]
25. Michaels, A.S. New separation techniques for the CPI. *Chem. Eng. Process.* **1968**, *64*, 31–42.
26. Vilker, V.L.; Colton, C.K.; Smith, K.A.; Green, D.L. The osmotic pressure of concentrated protein and lipoprotein solutions and its significance to ultrafiltration. *J. Membr. Sci.* **1984**, *20*, 63–77. [CrossRef]
27. Bellara, S.R.; Cui, Z. A Maxwell-Stefan approach to modeling the cross flow ultrafiltration of protein solutions in tubular membranes. *Chem. Eng. Sci.* **1998**, *53*, 2153–2166. [CrossRef]
28. Wesselingh, J.A.; Vonk, P. Ultrafiltration of a large polyelectrolyte. *J. Membr. Sci.* **1995**, *99*, 21–27. [CrossRef]
29. Ahmad, A.L.; Chong, M.F.; Bhatia, S. Ultrafiltration modeling of multiple solutes system for continuous cross flow process. *Chem. Eng. Sci.* **2006**, *61*, 5057–5069. [CrossRef]
30. Karasu, K.; Yoshikawa, S.; Kentish, S.E.; Stevens, G.W. A model for cross flow filtration of dairy whey based on the rheology of the compressible cake. *J. Membr. Sci.* **2009**, *341*, 252–260. [CrossRef]
31. Landman, K.A.; Sirakoff, C.; White, L.R. Dewatering of flocculated suspensions by pressure filtration. *Phys. Fluids Fluid Dyn.* **1991**, *3*, 1495–1509. [CrossRef]
32. Landman, K.A.; White, L.R.; Eberl, M. Pressure filtration of flocculated suspensions. *AIChE J.* **1995**, *41*, 1687–1700. [CrossRef]
33. Lawrence, N.D. Characterisation and Performance of Ultrafiltration Membranes in the Dairy Industry. Ph.D. Thesis, University of Melbourne, Melbourne, Australia, 1998.
34. Nguyen, T.A.; Yoshikawa, S.; Karasu, K.; Ookawara, S. A simple combination model for filtrate flux in cross flow ultrafiltration of protein suspension. *J. Membr. Sci.* **2012**, *403–404*, 84–93. [CrossRef]

35. Kirschner, A.Y.; Cheng, Y.H.; Paul, D.R.; Field, R.W.; Freeman, B.D. Fouling mechanisms in constant flux cross flow ultrafiltration. *J. Membr. Sci.* **2019**, *574*, 65–75. [CrossRef]
36. Hermia, J. Constant pressure blocking filtration laws—Application of power-law non-Newtonian fluids. *Trans. Am. Inst. Chem. Eng.* **1982**, *60*, 183–187.
37. Field, R.W.; Wu, J.J. Modeling of permeability loss in membrane filtration: Re-examination of fundamental fouling equations and their link to critical flux. *Desalination* **2011**, *283*, 68–74. [CrossRef]
38. Cheryan, M. *Ultrafiltration and Microfiltration Handbook*; CRC Press: Roca Raton, FL, USA, 1998.
39. Krippl, M.; Durauer, A.; Duerkoa, M. Hybrid modeling of cross filtration: Predicting the flux evolution and duration of ultrafiltration processes. *Sep. Purif.* **2020**, *248*, 117064. [CrossRef]
40. Sargolzaei, J.; Saghatoleslami, N.; Mosavi, S.M.; Khoshnoodi, M. Comparative study of artificial neural network (ANN) and statistical methods for predicting the performance of ultrafiltration process in the milk industry. *Iran J. Chem. Chem. Eng.* **2006**, *25*, 67–76.
41. Teodosiu, C.; Pastravanu, O.; Macoveanu, M. Neural network models for ultrafiltration and backwashing. *Water Res.* **2000**, *34*, 4371–4380. [CrossRef]
42. Wei, A.L.; Zeng, G.M.; Huang, G.H.; Liang, J.; Li, X.D. Modeling of a permeate flux of cross-flow membrane filtration of colloidal suspensions: A wavelet network approach. *Int. J. Environ. Sci. Technol.* **2009**, *6*, 395–406. [CrossRef]
43. Bowen, W.R.; Jones, M.G.; Yousef, H.N.S. Prediction of the rate of cross flow membrane ultrafiltration of colloids: A neural network approach. *Chem. Eng. Sci.* **1998**, *53*, 3793–3802. [CrossRef]
44. Hussain, W.; Merigo, J.M.; Reza, M.R. Predictive intelligence using ANFIS-induced OWAWA for complex stock market prediction. *Int. J. Intell. Syst.* **2022**, *37*, 4586–4611. [CrossRef]
45. Al-qaness, M.A.A.; Ewees, A.A.; Fan, H.; Abualigah, L.; Abd Elaziz, M. Boosted ANFIS model using augmented marine predator algorithm with mutation operators for wind power forecasting. *Appl. Energy* **2022**, *314*, 118851. [CrossRef]
46. Takagi, T.; Sugeno, M. Fuzzy identification of systems and its application to modeling and control. *IEEE Trans. Syst. Man Cybern.* **1985**, *15*, 116–132. [CrossRef]
47. Zivkovic, M.; Bacanin, N.; Venkatachalam, K.; Nayyar, A.; Djordjevic, A.; Strumbeger, I.; Al-Turjman, F. COVID-19 cases prediction by using hybrid machine learning and beetle antennae search approach. *Sustain. Cities Soc.* **2021**, *66*, 102669. [CrossRef]
48. Salehi, R.; Chaiprapat, S. Predicting H_2S emission from gravity sewer using an adaptive neuro-fuzzy inference system. *Water Qual. Res. J.* **2022**, *57*, 20–39. [CrossRef]
49. Khaled, B.; Abdellah, A.; Noureddine, D.; Salim, H.; Sabeha, A. Modelling of biochemical oxygen demand from limited water quality variable by ANFIS using two partition methods. *Water Qual. Res. J.* **2018**, *53*, 24–40. [CrossRef]
50. Jang, J.S.R. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Trans. Syst. Man Cybern.* **1993**, *23*, 665–685. [CrossRef]
51. Jang, J.S.R.; Sun, C.T.; Mizutani, E. *Neuro-Fuzzy and Soft Computing (A Computational Approach to Learning and Machine Intelligence)*; Prentice Hall: Hoboken, NJ, USA, 1997.
52. Karim, R.; Dilwar, F.; Siddique, R.A. Predictive modeling of surface roughness in MQL assisted turning of SiC-Al alloy composites using artificial neural network and adaptive neuro fuzzy inference system. *J. Adv. Res. Manuf. Mater. Sci. Metall. Eng.* **2018**, *5*, 12–28.
53. Keneni, B.M. Evolving Rule Based Explainable Artificial Intelligence for Decision Support System of Unmanned Aerial Vehicles. Master's Thesis, University of Toledo, Electrical and Computer Science Engineering Department, Toledo, OH, USA, 2018.
54. Clemente, L. ANFIS Modelling of PIV Tests. Master's Thesis, Department of Chemical and Materials Engineering, Polytechnic of Turin, Turin, Italy, 2019.
55. Deo, R.C.; Downs, N.J.; Adamowski, J.F.; Parisi, A.V. Adaptive neuro-fuzzy inference system integrated with solar zenith angle for forecasting sub-tropical photo-synthetically active radiation. *Food Energy Secur.* **2019**, *8*, e00151. [CrossRef]
56. Dehghani, M.; Riahi-Madvar, H.; Hooshyaripor, F.; Mosavi, A.; Shamshirband, S.; Zavadskas, E.K.; Chau, K.W. Prediction of hydropower generation using grey wolf optimization adaptive neuro-fuzzy inference system. *Energies* **2019**, *12*, 289. [CrossRef]
57. Belvederesi, C.; Dominic, J.A.; Hassan, Q.K.; Gupta, A.; Achari, G. Predicting river flow using an AI-based sequential adaptive neuro-fuzzy inference system. *Water* **2020**, *12*, 1622. [CrossRef]
58. Elith, J.; Leathwick, J.R.; Hastie, T. A working guide to boosted regression trees. *J. Anim. Ecol.* **2008**, *77*, 802–813. [CrossRef]
59. Carty, D.M. An Analysis of Boosted Regression Trees to Predict the Strength Properties of Wood Composites. Master's Thesis, Graduate School at Tennessee Research Creative Exchange, University of Tennessee, Knoxville, TN, USA, 2011.
60. Friedman, J.H.; Meulman, J.J. Multiple additive regression trees with application in Epidemiology. *Stat. Med.* **2003**, *22*, 1365–1381.
61. Chang, Y. Economic forecasting by a piecewise regression tree method. *Int. J. Manag. Appl. Sci.* **2017**, *3*, 12–15.
62. Koc, Y.; Eyduran, E.; Akbulut, O. Application of regression tree method for different data from animal science. *Pak. J. Zool.* **2017**, *49*, 599–607. [CrossRef]
63. Naumets, S.; Lu, M. Investigation into explainable regression trees for construction engineering applications. *J. Constr. Eng. Manag.* **2021**, *147*, 04021084. [CrossRef]
64. Breiman, L.; Friedman, J.; Olshen, R.; Stone, C. *Classification and Regression Trees*; Chapman and Hall/CRC: London, UK, 1984.
65. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning with Application in R*; Springer Science + Business Media: New York, NY, USA, 2013.
66. Das, S. Reconstruction of Gamma-Ray Direction Using Boosted Decision Trees and the Disp Parameter. Master's Thesis, Department of Physics, McGill University, Montreal, QC, Canada, 2019.

67. Akalin, A. *Computational Genomics with R*; Chapman and Hall/CRC: New York, NY, USA, 2020. [CrossRef]
68. Ceballos-Santos, S.; González-Pardo, J.; Carslaw, D.C.; Santurtún, A.; Santibáñez, M.; Fernández-Olmo, I. Meteorological normalisation using boosted regression trees to estimate the impact of COVID-19 restrictions on air quality levels. *Int. J. Environ. Res. Public Health* **2021**, *18*, 13347. [CrossRef]
69. Mehta, M.; Agrawal, R.; Rissanen, J. SLIQ: A fast scalable classifier for data mining. In *International Conference on Extending Database Technology*; Springer: Berlin, Heidelberg, 1996. [CrossRef]
70. Sutton, C.D. Classification and regression trees, bagging, and boosting. *Handb. Stat.* **2005**, *24*, 303–329.
71. Bastos, J. *Credit Scoring with Boosted Decision Trees*; CEMAPRE, School of Economics and Management (ISEG), Technical University of Lisbon: Lisbon, Portugal, 2008.
72. Roe, B.P.; Yang, H.J.; Zhu, J.; Liu, Y.; Stancu, I.; McGregor, G. Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nucl. Instrum. Methods Phys. Res. A* **2005**, *543*, 577–584. [CrossRef]
73. Hastie, T.; Tibshirani, R.; Friedman, J.H. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: NewYork, NY, USA, 2001.
74. Schapire, R.E. The strength of weak learnability. *Mach. Learn.* **1990**, *5*, 197–227. [CrossRef]
75. Freund, Y. Boosting a week learning algorithm by majority. *Inf. Comput.* **1995**, *121*, 256–285. [CrossRef]
76. Freund, Y.; Schapire, R.E. A Short introduction to boosting. *J. Jpn. Soc. Artif. Intell.* **1999**, *14*, 771–780. (In Japanese)
77. Ridgeway, G. The state of boosting. *Comput. Sci. Stat.* **1999**, *31*, 172–181.
78. Latif, S.D. Developing a boosted decision tree regression prediction model as a sustainable tool for compressive strength of environmentally friendly concrete. *Environ. Sci. Pollut. Res.* **2021**, *28*, 65935–65944. [CrossRef] [PubMed]
79. Xie, Y.; Zhu, C.; Lu, Y.; Zhu, Z. Towards optimization of boosting models for formation lithology identification. *Math. Probl. Eng.* **2019**, *13*, 5309852. [CrossRef]
80. Carslaw, D.C.; Taylor, P.J. Analysis of air pollution data at a mixed source location using boosted regression trees. *Atmos. Environ.* **2009**, *43*, 3563–3570. [CrossRef]
81. Fedotenkova, M. Extraction of Multivariate Components in Brain Signals Obtained during General Anesthesia. Ph.D. Thesis, Faculty of science and technology, University of Lorraine, Nancy, France, 2016.
82. Salehi, R.; Lestari, R.A.S. Predicting the performance of a desulfurizing bio-filter using an artificial neural network (ANN) model. *Environ. Eng. Res.* **2021**, *26*, 200462. [CrossRef]
83. Draper, N.R.; Smith, H. *Applied Regression Analysis*; John Wiley & Sons: New York, NY, USA, 1998.
84. Montgomery, D.C. *Design and Analysis of Experiments*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
85. Eschenbach, T.G. Spiderplots versus tornado diagrams for sensitivity analysis. *Interfaces* **1992**, *22*, 40–46. [CrossRef]
86. Bassam, A.; Tzuc, M.; Soberanis, M.E.; Ricalde, L.J.; Cruz, B. Temperature estimation for photovoltaic array using an adaptive neuro fuzzy inference system. *Sustainability* **2017**, *9*, 1399. [CrossRef]
87. Tran, Q.H.; Huh, J.; Nguyen, V.B.; Kang, C.; Ahn, J.H.; Park, I.J. Sensitivity analysis for ship-to-shore container crane design. *Appl. Sci.* **2018**, *8*, 1667. [CrossRef]