Corrosion Science 193 (2021) 109904

Contents lists available at ScienceDirect

Corrosion Science

journal homepage: www.elsevier.com/locate/corsci

Machine learning modeling of time-dependent corrosion rates of carbon steel in presence of corrosion inhibitors

Mohammadreza Aghaaminiha [a], Ramin Mehrani [b], Martin Colahan [a], Bruce Brown [a], Marc Singer [a], Srdjan Nesic [a], Silvia M. Vargas [c], Sumit Sharma [a,*]

[a] Department of Chemical and Biomolecular Engineering, Ohio University, USA
[b] Department of Mechanical Engineering, Ohio University, USA
[c] BP America Production Company, USA

ARTICLE INFO

Keywords:
Corrosion inhibitors
Corrosion rates
Machine learning
Random forest

ABSTRACT

We have employed supervised machine learning methods to model measurements of corrosion rates of carbon steel as a function of time when corrosion inhibitors are added in different dosage and dose-schedules. The experiments show that the time-profile of corrosion rates depend on the dose schedule, while the final rates depend mainly on the environment severity. We find that Random Forest was the best algorithm that predicted the entire time-profile of corrosion rates with the mean squared error ranging from 0.005 to 0.093. Sensitivity of corrosion rates to changes in the environmental variables are well-predicted by the trained Random Forest model.

1. Introduction

Corrosion inhibitors are commonly used to mitigate internal corrosion of oil and gas pipelines [1–3]. These molecules are injected in parts-per-million (ppm) concentrations in a continuous or semi-continuous manner. It is understood that corrosion inhibitor molecules adsorb at the metal-water interfaces, with their adsorption chiefly governed by the strong affinity of the polar group for the metal surface as well as hydrophobic interactions between their tails [1–6]. It is important to ascertain the optimal dosage as well as the frequency of the doses of corrosion inhibitors needed to minimize corrosion of the pipelines [7]. For this purpose, extensive experimentation is carried out wherein the corrosion rates of steel specimen are measured in different environmental and operational conditions and by using different dosages and dose-schedules of inhibitors. These experiments are time-consuming and costly but are nevertheless necessary, in the absence of reliable theoretical models.

While mechanistic models of corrosion prediction in the absence of inhibitors have been developed and implemented for commercial purposes [8–12], there has been little success in incorporating the effect of corrosion inhibitors in these models. One difficulty is that the exact mode of action of corrosion inhibitors remains unclear [2]. Secondly, the efficiency of the inhibitors depends on a large number of environmental and operational factors, such as temperature, water chemistry, flow rate, etc. [13]. Thirdly, different inhibitor molecules are observed to perform optimally under different conditions for reasons unknown [14–17]. Lastly, corrosion inhibitor formulations used in the field are mixtures of molecules that supposedly work in synergy. Often, the composition of corrosion inhibitor formulations is not disclosed to the operator/researcher as it is proprietary knowledge belonging to the manufacturers, making it all difficult to model their behavior.

Along with experimental methods, molecular modeling has been employed for elucidating the molecular-level adsorption behavior of corrosion inhibitors [6,18–23]. However, molecular modeling can mostly help with understanding of the adsorption behavior of inhibitor molecules and not how they retard corrosion. Therefore, developing theoretical models to explain the performance of corrosion inhibitors remains a challenging task.

In the last decade, machine learning (ML) has been employed in a number of corrosion-related problems, such as for modeling CO_2 corrosion [24], detecting corrosion from automated image analysis [25], modeling corrosion defect growth in pipelines [26], material inspection [27], modeling corrosion rate in marine environment [28], finding corrosion initiation time of embedded steel in reinforced concrete [29],

* Correspondence to: Chemical and Biomolecular Engineering, Russ College of Engineering and Technology, Ohio University, 181 Stocker Center, Athens, OH 45701, USA.
E-mail address: sharmas@ohio.edu (S. Sharma).

https://doi.org/10.1016/j.corsci.2021.109904
Received 11 July 2021; Received in revised form 26 August 2021; Accepted 18 October 2021
Available online 22 October 2021

Table 1
Chemical composition of mild steel coupons used in experiments.

Composition	Elements									
	Cr	Mo	S	V	Si	C	Ni	Mn	P	Fe
Weight %	0.14	0.16	0.009	0.047	0.26	0.13	0.36	1.16	0.009	Balance

Table 2
Environmental and operational input variables (features) considered to model corrosion rate as a function of time. The last four features were added to achieve accurate modeling of the experiments.

Description	Range	Unit	Type
Corrosion inhibitor concentration	[0 – 500]	ppm	Numerical
Time	[0 – 160]	hour	Numerical
CO_2 partial pressure	[0.51 – 12]	bar	Numerical
Temperature	[80 – 130]	°C	Numerical
Corrosion inhibitor type	{*CI-1*, *CI-2*}	–	Categorical
Wall shear stress	(20, 277)	Pa	Numerical
Brine ionic strength	(0.615, 2.31, 0.51)	mol/L	Numerical
Brine type	{*A*, *B*}	–	Categorical
pH	{controlled 6, uncontrolled}	–	Categorical
Prior corrosion inhibitor concentration	[0 – 200]	ppm	Numerical
Initial corrosion rate	[0.05, 34.10]	mm/y	Numerical
Type of test	{sequential dosing, single dose}	–	Categorical

predicting electrochemical impedance spectra [30], modeling pipeline aging [31], and characterizing the spatial distribution of pitting corrosion [32].

In this work, we have performed regression using different ML algorithms (Artificial Neural Network, Random Forest, Support Vector Machines, and *K* Nearest Neighbors) to model experimental data of time-varying corrosion rates of mild steel specimens, when corrosion inhibitors were added to the system in different concentrations and dose-schedules. We demonstrate that the trained ML models are quite accurate in predicting the time-dependent and steady-state corrosion rates of laboratory experiments. Furthermore, a few experiments reported significant scatter in the data even in well-controlled experimental conditions. In these cases, ML models are found useful in predicting which experimental results may be less trustworthy.

2. Methodology

2.1. Description of the experimental data

The experimental data used was obtained from a series of experiments on corrosion inhibition of mild steel in CO_2 aqueous solutions. In the experiments, the inhibited corrosion rate, which changes over time, was measured by using Linear Polarization Resistance (LPR). The experiments were conducted independently at four different laboratories, using mild steel coupons with identical chemical composition, shown in Table 1, and two different organic corrosion inhibitors (*CI-1* and *CI-2*). The experimental matrix was designed to cover a large parameter space as shown in Table 2. Individual experiments lasted between one and seven days and were conducted at CO_2 partial pressures between 0.5 and 12 bar at temperatures between 80 °C and 130 °C. In some cases, the pH of the solution was controlled at pH 6 while in others that was not the case. Overall, there were 25 different experimental conditions, with many experiments replicated multiple times (26,855 corrosion rate data points in total). The corrosion inhibitors were added to the solution at different times, in different dosages, and by using various dosing schedules. Some of the experiments were conducted with "pre-corrosion", meaning that the steel specimens were allowed to corrode for some time, before the first dose of the inhibitor was added. In shorter duration experiments, the addition of the inhibitor was done as a single dose, while sequentially increasing dosing was performed in longer duration experiments. Fig. 1 shows plots of typical corrosion rate data taken from two experiments: A) with sequential dosing of the inhibitor, starting with 1 ppm and incrementally increasing up to 10 ppm, and B) when a single dose of inhibitor was added at 10 ppm. In both cases, inhibitor addition was done after a period of pre-corrosion of about 4.5 h. The corrosion rate started out high, at about 7.2 mm/y in both cases, and remained constant during the pre-corrosion period. From Fig. 1A, we can observe that already after the addition of 1 ppm of the inhibitor, the corrosion rate was reduced approximately by 10-fold, and that every next addition reduced the corrosion rate further. At the end of the experiment, the corrosion rate was about 0.04 mm/y obtained with 10 ppm inhibitor, which amounts to an inhibition efficiency of 99.4%. Comparing the sequential dosing with the single dose inhibitor injection, we can observe that a very similar final corrosion rate was achieved

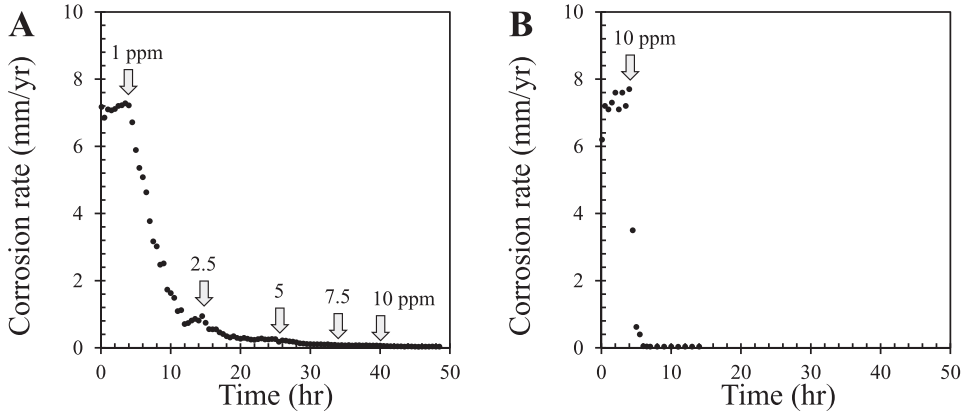


Fig. 1. Two typical types of corrosion experiments. Inhibitor added in A) sequential dosing, B) single dose. Arrows indicate the concentration in ppm and the time at which the corrosion inhibitor was injected into the system. In both experiments: inhibitor type = *CI-1*, p_{CO2} = 0.51 bar, T = 80 °C, pH = controlled 6, wall shear stress = 20 Pa, brine ionic strength = 2.31 M, and brine type = *A*.

(approximately 0.04 mm/y), suggesting that the dosing schedule was not an important variable if one is interested in the final corrosion rate. However, the kinetics profile in the two cases is significantly different. All other experiments in the dataset were of a similar nature, even if the details varied.

2.2. A brief overview of machine learning methods

Machine learning (ML) methods are suitable for developing predictive models in the cases where a large dataset is available, the outcome to be predicted depends on several variables, and when a mechanistic model of the relationship between the input variables and the outcome is not well established. Before we embark on describing our work in detail, it is useful to provide a brief introduction to ML methods used here for the readers who are experts in the corrosion field but not necessarily in ML. A reader well-versed in ML methods may skip this section. ML refers to a class of algorithms that get to learn how to perform a task, such as predicting the outcome of an experiment, when they are trained on the data obtained from some previously performed experiments [33,34]. The ML algorithms that learn from data wherein the outcome of the experiment is specified are called supervised learning [34]. Some popular supervised ML algorithms are Random Forest (RF), Artificial Neural Network (ANN), Supported Vector machines Regression (SVR), and K Nearest Neighbors (KNN).

To understand RF algorithm, one needs to first understand what is meant by a decision tree. In a decision tree, a dataset is split around input variables into smaller subsets. The split is performed so that the subsets that are formed have smaller variances in the outcome values. Each split can be thought of as a branch of the tree and each data subset as a leaf. The data are progressively split until some terminal condition is met. The terminal condition can be either that the maximum number of splits has been performed or the standard deviation of a subset has fallen below a cutoff value. The average value of the outcome in the terminal leaf, that is the leaf, which is not split any further, is the predicted value of the outcome for those set of input variables. As a simple example, imagine a decision tree which is used to predict the corrosion rate as a function of two key variables, pH and temperature, is created with a tree split with branches pH < 7 and pH ≥ 7. These branches are each further split for temperature < 350 K and temperature ≥ 350 K. Then, to predict the corrosion rate for a condition, say with pH $= 4.5$ and temperature $= 300$ K, the average value of the corrosion rate for the leaf: pH $< 7 \rightarrow$ temperature < 350 K is the predicted value of this decision tree. A RF is comprised of a multitude of decision trees. The decision trees are formed based on random subsets of the training data with replacement and using random subsets of features. RF algorithm reports the weighted mean value of the predictions from all decision trees. In general, using the outcome of many ML models to make the final prediction is called ensemble learning, and has been shown to significantly improve the prediction performance [35]. Therefore, RF is an ensemble learning method based on numerous decision trees. So, in the example discussed above, the corrosion rate is the output variable (label) and pH and temperature are the input variables (features) [36,37].

ANNs are networks of interconnected nodes that act as universal approximators, that is, they can approximate any continuous function to an arbitrary level of accuracy with a finite number of nodes [38]. The architecture of ANN is as follows: In the first layer (named input layer), each input variable, x_i is fed to a node. Each node in the input layer is connected to the nodes in the next layer (called first hidden layer). The connections between the nodes are assigned some weights, $w_{ij.}$ At each node in the first hidden layer, a weighted sum of the inputs from the nodes of the input layer is calculated, $F_j = \sum_i w_{ij} x_i$. The F_j is transformed via an activation function, such as a sigmoidal function, which is the output from each node, $x_j = \frac{1}{1+e^{-F_j}}$, and which becomes the input from node j in the first hidden layer to the nodes in the second hidden layer. This process is repeated for all the layers until the output layer is reached, which in regression problems is a single node that predicts the outcome. Training an ANN architecture entails adjusting the weights connecting the nodes so as to minimize the mean squared error between the predicted and the actual outcome values/classes [39,40].

SVR is another powerful supervised learning algorithm. In general, the relationship between the input variables and the label is non-linear. In SVR, the input variables are transformed to higher dimensions where the relationship may be better linearly separable. In a higher dimension, a linear regression line is fitted to the data. For example, a polynomial relationship, such as $y = a_0 + a_{11}x_1 + a_{12}x_2 + .. + a_{n1}x_1^n + a_{n2}x_2^n$ can be represented as a $2n$ dimensional hyperplane. There are special functions called kernels that help in determining the hyperplane in the higher dimension without increasing the computational cost [41].

KNN is a simple algorithm in which, for each data point, K nearest-neighbors are identified in the training set (where K is an integer), and the average value of their labels is reported as the outcome of that data point [42]. The K nearest-neighbors are identified by defining a *distance*. For numerical input variables, Euclidean distance is commonly used, whereas, for categorical variables, Hamming distance is used [43].

As discussed above, every ML algorithm has a specific architecture. For instance, an ANN is characterized by the number of hidden layers and the number of nodes per hidden layer; a RF is characterized by the number of trees and the maximum number of features that can be split. These parameters that define the architecture of a ML algorithm are called hyperparameters [44]. The performance of a ML algorithm on a dataset varies as one changes the hyperparameters. Therefore, values of the hyperparameters need to be tuned/adjusted to optimize performance. In order to tune the hyperparameters, first, the data should be split into a training set and a testing set. The training set is often taken as 70–80% of the entire dataset. The optimum values of the hyperparameters are found by evaluating the performance of all ML models on the training set. Once the hyperparameters are fixed, the best ML algorithm is trained on the training set, and then its performance is evaluated on the testing set.

In this work, we have developed a number of different predictive models of corrosion rates of carbon steel as a function of time, when corrosion inhibitors are added to the system in different dosages and dosing schedules, using the different ML algorithms (ANN, RF, SVR, KNN). The ML-based models are useful in predicting corrosion rates at different environmental and operational conditions and alleviate the need for further experimentation. Our analysis shows that RF is the best ML algorithm for this type of modeling for a given data. The details of the implementation are discussed next.

2.3. ML implementation methodology

To perform prediction using ML algorithms for any data, the following steps need to be taken: (1) data preprocessing, (2) tuning of hyperparameters for several ML models, (3) finding the optimum ML algorithm, and (4) training the optimum ML algorithm and testing its performance. In addition, trained ML algorithms are useful for performing sensitivity analysis to study the response in the outcome as a function of changes in input variables. Generally, K-fold cross-validation is performed on the training set to determine the best set of hyperparameters. Once the best set of hyperparameters is found, then the performance of the models is tested on the testing set. In this work, we have used a different approach because we are interested in generating complete time-profiles of 4 randomly selected experiments, while using the remaining 21 experiments as the training set. A more stringent test of ML algorithms as well a more practically useful approach is that the hyperparameters of the models are fine-tuned only once and not for every new set of 21 experiments. Therefore, our methodology involves two steps: first, selection of the best ML model, and second, training and testing of the best ML model. For the first step, we use all data-points for tuning the hyperparameters of each ML algorithm (ANN, RF, SVR, KNN). Once the hyperparameters are selected, then we randomly select 75% of

Table 3
Ranges of values for the hyperparameters of the studied algorithms. (*nHLs*: number of hidden layers, *nNodes*: number of nodes per hidden layer; *γ-gamma*: kernel coefficient, *cost*: regularization parameter; *nTR*: number of trees, *mFF*: maximum fraction of features to be split for a decision tree; and *K*: number of neighbors).

ANN		SVR		RF		KNN	
nHLs	*nNodes*	*γ*	*cost*	*nTR*	*mFF*	*K*	*weights*
1	2	1.0	1	10	0.6	3	uniform
2	4	0.1	5	50	0.7	4	distance
3	6	0.01	10	100	0.8	5	
4	8	0.001	100	200	0.9	6	
5	10	0.0001	1000	500	1.0	7	

the data-points for the training set and 25% for the testing set. The ANN, SVR, RF and KNN algorithms (with optimized hyperparameters) are then trained on the training set and their performance is compared against the testing set. Once the best ML algorithm is selected with the optimized set of hyperparameters, then the ML algorithm and the hyperparameters are never changed. In the second step, 4 experiments are randomly selected as testing set and 21 remaining experiments are selected as the training set. It should be noted that in this procedure, the testing set remains completely hidden during the training of the ML algorithm. Each of the steps needed for the ML implementation will be discussed in detail below.

Step 1. Data preprocessing. The first step is to ensure that the dataset does not have any missing or incorrect values [45,46]. Data points with missing values are either removed or the missing values are replaced with the mean or mode or a best guess value of that variable. If a variable has many missing values, then the variable may be removed from the dataset. Different variables have a different range of values and so it is important to rescale the variables. One way is to rescale the variables so that all the values are between 0 and 1. Another way to rescale the variables is to assume that the values are normally distributed and so one can convert them into standard normal variables by subtracting out the mean and dividing by the standard deviation. Table 2 lists the variables and their ranges in the experimental data of inhibited corrosion rate measurements that we have modeled in this work. The dataset includes eight numerical and five categorical features. Categorical features are converted to dummy variables. For example, if the categorical feature "Brine type" has two possible values, *A* and *B,* then a dummy variable for *A* can be created that takes two possible values {0, 1} with 1 representing brine type *A* and 0 representing brine type *B*. In our work, all numerical variables except *Time* were rescaled to have a standard normal distribution [46]. In the implementation of our ML strategy, we found that for accurate modeling of the experiments, some additional data preprocessing steps were needed. First, the *Time* was set to zero at each injection time of the corrosion inhibitor. Furthermore, the following new input features were added: (a) prior corrosion inhibitor concentration: this feature takes the value of the inhibitor concentration present in the system prior to a new dose; (b) initial corrosion rate: this feature specifies the corrosion rate at the beginning of the experiment; and (c) type of test: this categorical feature is set to 1 if the experiment involved more than one dose of corrosion inhibitor, that is for a sequential dosing experiment, and is set to 0 if there is only a single dose of corrosion inhibitor in the experiment.

Step 2. Tuning hyperparameters. One needs to determine the best hyperparameters for each ML algorithm (ANN, RF, SVR, KNN), that is the hyperparameters that will result in the smallest mean squared error (MSE) [44]. The best hyperparameters were found by performing a grid search over a range of hyperparameter values, same approach as our previous research work [47]. For each set of hyperparameter values, a 5-fold cross-validation method was used on the data. In 5-fold cross-validation, the data is divided into 5 subsets. The ML algorithm is then trained using the data in four subsets, and the performance of the trained algorithm is tested on the 5th subset. This procedure is repeated 5 times, each time choosing a different subset for testing the performance [33,34]. Then, the average performance of the ML algorithm (in our case, MSE score) is determined. Using this methodology, the best hyperparameter values of the different ML models (ANN, SVM, RF, and KNN) were determined. For ANN and RF models, 20 iterations of 5-fold cross-validation were performed.

Step 3. Finding the optimum ML model. After tuning the hyperparameters, one needs to select the best ML model out of ANN, RF, SVR and KNN models. For this purpose, the entire dataset was divided into a training set, comprising of 75% of the data, and the remaining as the testing set. The ML models were trained on the training set, and then the model with the best performance on the testing set, that is the one with the lowest MSE, was selected as the optimum ML model.

Steps 4. Training the optimum ML model and testing its performance. One useful and practical application of our ML modeling is to be able to predict results of complete experiments of corrosion rates as a function of time for given environmental and operational conditions and dosage schedule of corrosion inhibitors. To check the performance of our optimum ML model, a different strategy was adopted compared to the one used in **Step 3**, where 75% of data points used for training were selected from all the experiments, meaning that none of the individual experiments were completely hidden from the model. To make it more challenging for the model, the process of training was repeated but this time by using data from randomly selected 21 out of 25 experiments, which formed the training set; the remaining 4 experiments were completely hidden from the model and were later used as the testing set. In cases where an experiment was repeated multiple times and was randomly selected for the testing set, all its replicas were hidden from the model during training.

Sensitivity analysis**.** A trained ML model can also be used to study the behavior of the system as a function of changes in the values of the input features. In our study, the time-varying corrosion rate was predicted as a function of inhibitor type, inhibitor concentration, CO_2 partial pressure, temperature, wall shear stress, and brine type using the trained ML model.

3. Results and discussion

3.1. Identifying the best ML model

The ranges of the hyperparameters that were tested for each of the ML algorithms in this study are listed in Table 3. For ANN, the number of hidden layers (*nHLs*) and the number of nodes per hidden layer (*nNodes*) were the two hyperparameters that were varied. In SVR, two hyperparameters were varied: the kernel coefficient, *γ*, and the regularization parameter, *cost*. As discussed above, in SVR, the input variables are transformed to higher dimensions using kernel functions. We have used Gaussian functions as the kernels. Here, *γ* refers to the inverse of the standard deviation or the spread of the Gaussian functions used. The *cost* parameter controls how many kernel functions are employed in the SVR. For RF, we choose the number of decision trees, *nTR*, and the maximum fraction of input variables that can be used to create a decision tree, *mFF*, as the two hyperparameters. For KNN, the value of *K*, that is, the number of nearest neighbors that one should consider is taken as the hyperparameter. Alongside, we studied the performance of KNN models when equal weight is given to every neighbor (*weight* set to "uniform") and when nearer neighbors are given more weight than the farther ones (*weight* set to "distance"). It should be noted that for each ML algorithm,

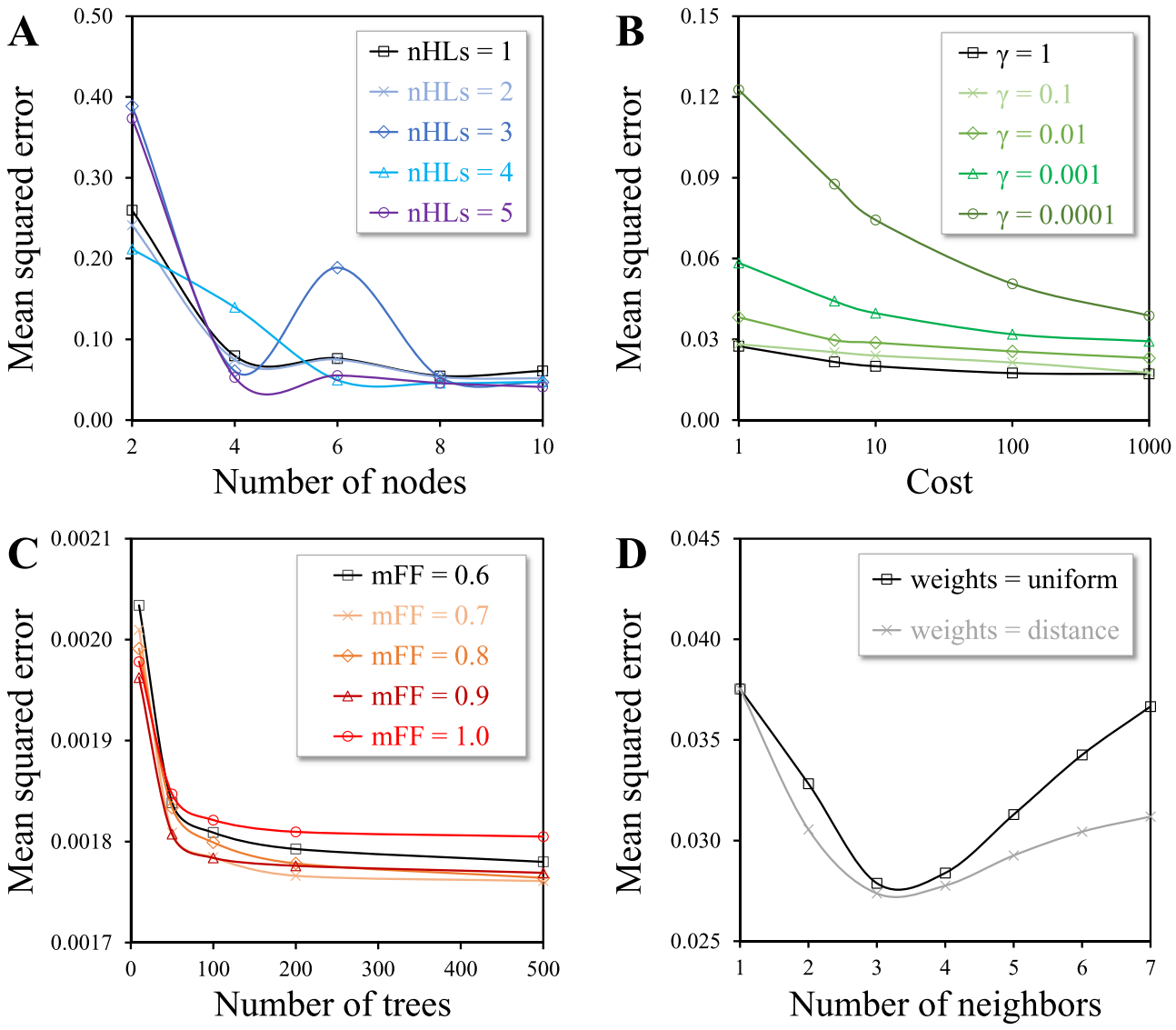


Fig. 2. Performance of **A)** Artificial Neural Network (ANN), **B)** Support Vector machine Regression (SVR), **C)** Random Forest (RF), and **D)** K Nearest Neighbors (KNN) on experimental data of corrosion rate for different values of the hyperparameters. Each point is the average of 20 independent iterations of training followed by testing on the testing set. Lines are guides for readability.

Table 4
Optimum values of the hyperparameters along with their MSE are reported. Note that the reported MSEs are the average of 20 independent iterations per testing action. (*nHLs*: number of hidden layers, *nNodes*: number of nodes per hidden layer; *γ-gamma*: kernel coefficient, *cost*: regularization parameter; *nTR*: number of trees, *mFF*: maximum fraction of features to be split for a decision tree; and *K*: number of neighbors).

	ANN		SVR		RF		KNN	
	nHLs	*nNodes*	*γ*	*cost*	*nTR*	*mFF*	*K*	*weights*
Hyperparameters	4	8	1	1000	500	0.7	3	distance
MSE	0.048 ± 0.010		0.017 ± 0.001		0.002 ± 0.001		0.027 ± 0.001	

only the hyperparameters that are understood to have the largest effect on the performance were studied, while choosing the default suggested values of the other hyperparameters. We have implemented the ML algorithms using the Scikit Learn package in Python, and the default hyperparameters for each ML algorithm can be found in the Scikit Learn documentation online [48].

The results of the 5-fold cross-validation of the ML algorithms for different values of the hyperparameters are shown in Fig. 2. For ANN (Fig. 2**A**), it was observed that the mean squared error (MSE) decreased with the increase in the number of nodes per hidden layer. The decrease in the MSE became gradual beyond 6 nodes. Increasing the number of nodes beyond 8 resulted in a slight increase in the MSE. Therefore, 8 nodes per hidden layer with 4 hidden layers was the optimum architecture. In the case of SVR (Fig. 2**B**), it was observed that the performance was sub-optimal for small values of γ (< 0.001) and was strongly dependent on the *cost*. For $\gamma > 0.01$, the performance showed significant improvement and was not overly sensitive to the *cost*. The MSE was observed to decrease with the *cost*, and so the SVM model with *cost* and γ set to 1000 and 1, respectively, was the optimum SVR model. In the case of RF model (Fig. 2**C**), the best performance (lowest MSE) was observed

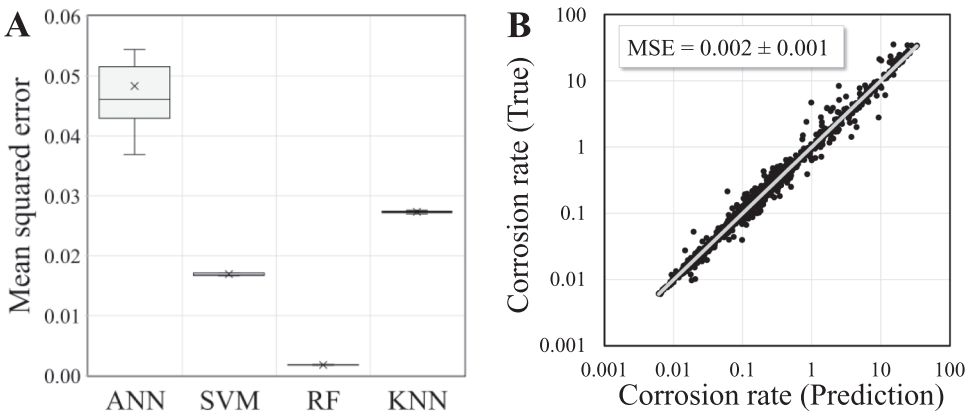


Fig. 3. A) Comparison of performance of the best model for each ML model. Error bars are obtained by performing 20 independent iterations of each model. **B)** Parity plot between the predicted corrosion rates (mm/y) by the RF model and the experimental values; The reported MSE value is the average of 20 independent iterations of training followed by testing on the testing set of the RF model. In both figures, predictions are made on the testing set which is randomly selected as 25% of the available 26,855 data points.

when the *mFF* was set to 0.7. Beyond 200 decision trees, some improvement in the performance was observed and so the RF model with 500 trees was the optimum RF model. In the case of KNN (Fig. 2**D**), a V-shaped curve of the MSE was obtained, implying the $K = 3$ as the optimum choice. Also, having the *weight* based on the distance rather than having a uniform weight for all the neighbors yielded lower MSE. The optimum set of hyperparameters for each ML model are reported in Table 4.

It was found that RF is the best algorithm among those studied with the MSE of 0.002 ± 0.001 (Fig. 3**A**). Therefore, RF was selected as the ML model for our investigations of corrosion rate as a function of time. The parity plot of the predictions of the RF model of the corrosion rate on the testing set, which is randomly selected as 25% of the whole 26,855 data points, is shown in Fig. 3**B**. The parity plot shows that the RF model is quite accurate in predicting the experimental results of corrosion rate measurements.

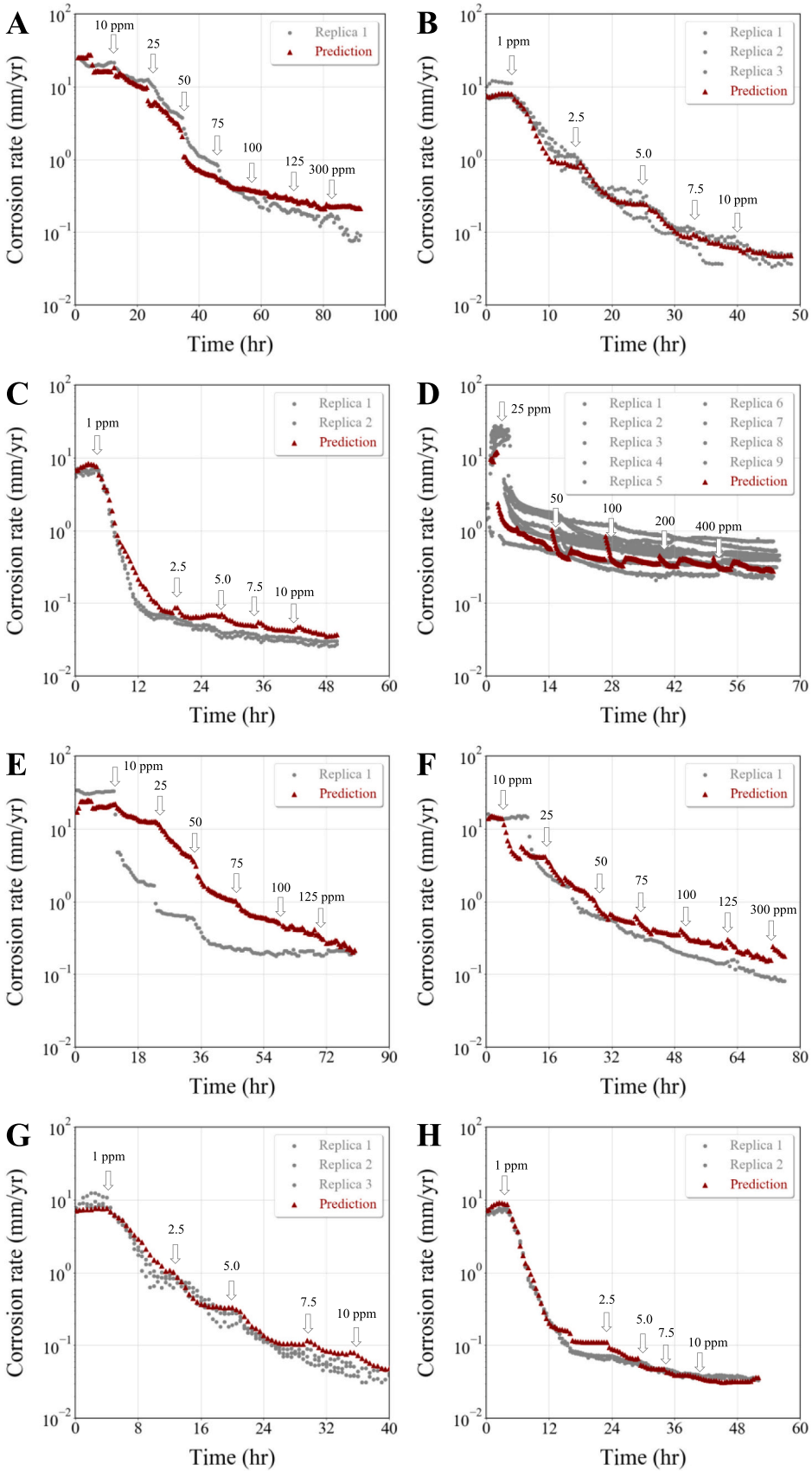
3.2. Predictions of time-dependent corrosion rates using RF

After selecting RF for our analysis, we need to evaluate its performance in predicting the entire kinetics of an experiment. Recall that now the RF model was trained using data from randomly selected 21 out of 25 experiments, while keeping the remaining 4 experiments completely hidden from the model as the testing set. The first set of results presented here are from experiments with sequential dosing of the inhibitor. Fig. 4 shows eight experiments with pre-corrosion and their prediction by the RF model. Corrosion rates are plotted on a logarithmic scale to be able to show on the same graph, both the high corrosion rates at the beginning of the experiment before inhibitor injection and the low corrosion rates after inhibition. It can be observed that the initial corrosion rates reflect the severity of the aqueous environment that also has an impact on the time-dependent corrosion rates. For example, the conditions of experiments shown in Fig. 4 **(A**, **D**, and **E)** were more aggressive, having the initial corrosion rates in the range 20 – 30 mm/y, which can be attributed primarily to the higher partial pressures of CO_2 in these experiments (p_{CO2} = 2.5, 1.57, and 2.5 bar respectively), This is clear when comparing to the conditions presented in Fig. 4 **(B**, **C**, **F**, **G**, and **H)** where the p_{CO2} = 0.51 bar and the initial corrosion rates were about 10 mm/y. It is observed that the time-dependent corrosion rates depend on the dose schedule of the inhibitors. One would imagine that the final corrosion rates (after inhibition) are also related to the inhibitor type and the inhibitor dosing, but this actually is not the case.[*] For example, for both *CI-1*, shown in Fig. 4**A** and Fig. 4**B** and *CI-2*, shown in Fig. 4**C** and Fig. 4**D** the final corrosion rate seems to be more related to the severity of the environment, rather than to the inhibitor type. While the experiments with severely corrosive conditions were given a higher concentration of initial dose of the corrosion inhibitor, the severity of the environment still had the major impact on the final corrosion rate observed. For example: in the experiments shown in Fig. 4 **(A**, **D**, and **E)** the final corrosion rate stayed relatively high, and in the experiments shown in the Fig. 4 **(D** and **E)** cases – it never decreased below 0.1 mm/y. This is mostly related to the high p_{CO2} in those experiments and a higher temperature (106 °C vs. 80 °C); it is well known that organic corrosion inhibitor performance deteriorates at a higher temperature. In the same experimental condition, shown in Fig. 4**D**, we see a substantial scatter of the inhibited corrosion rate results (with a factor of 5 in the 9 "replicated" experiments), which is primarily related to the uncontrolled pH in those experiments and compounded by the severe conditions for inhibition (high p_{CO2} and high temperature). Finally, there seems to be no influence of brine type or ionic strength on the corrosion rate under any condition used in this study.

When it comes to predictions, we can observe that the RF model was quite accurate in reproducing the experimentally observed values in all cases. This holds true even for the experiments shown in Fig. 4**D**, where a large scatter in the experimental results is seen, yet the RF model falls within the scatter band. In that condition, the prediction from the RF model shows some "jumps" in the corrosion rate at times when the corrosion inhibitors were injected into the system, which is clearly an unrealistic behavior of the model, but, nevertheless, the overall prediction from the model remains reasonable. An outlier to some extent is the case presented in Fig. 4**E** where the conditions were rather extreme (p_{CO2} = 2.5 bar, wall shear stress of 277 Pa), hence the RF model had more trouble in predicting the inhibited corrosion rate evolution, even if it got the final corrosion rate right. Except for the Fig. 4**E**, for which the mean squared error (MSE) is 0.362, for the other experiments, the overall MSE ranges from 0.008 to 0.057, which is a pretty good accuracy of the RF model.

Next, we look at single-dose experiments with pre-corrosion shown in Fig. 5. In all cases, we have similar initial corrosion rates, which are quite high: 5 – 10 mm/y, indicating a severe corrosion environment. Consequently, a high concentration of corrosion inhibitors was injected in these experiments. After inhibition, similar conclusions can be drawn as we had for the experiments with sequential dosing of the inhibitor, discussed above. One can clearly see that neither does the inhibitor type nor its dose directly correlates to the final corrosion rate. [†] The same is true for brine's ionic strength. It is understood that beyond the surface saturation concentration of corrosion inhibitors, little or no improvement in corrosion inhibition is reported [49]. The final corrosion rates were low (below 0.2 mm/y) in all cases, except in the experiment shown in Fig. 5**C**, which can be explained by the high temperature (132 °C), which made it hard for the organic inhibitor to perform well, even if the final dose was extremely high: 100 ppm. Just like in the previous example where sequential dosing was used, the large scatter in the inhibited corrosion rate was obtained in experiments where the pH was

[1] This conclusion is limited to the two inhibitors considered here and is not a general rule.



(caption on next page)

Fig. 4. Comparison between the predicted (red) and the experimentally reported (grey) corrosion rates as a function of time for sequential dosing experiments with pre-corrosion. Below are the details of the experiments.

	A	B	C	D	E	F	G	H
CI type	*CI-2*	*CI-2*	*CI-1*	*CI-1*	*CI-1*	*CI-2*	*CI-1*	*CI-2*
Exposure duration	~ 90 hrs.	~ 50 hrs.	~ 50 hrs.	~ 70 hrs.	~ 80 hrs.	~ 80 hrs.	~ 40 hrs.	~ 50 hrs.
p_{CO2}	2.5 bar	0.51 bar	0.51 bar	1.57 bar	2.5 bar	0.51 bar	0.51 bar	0.51 bar
Temperature	80 °C	80 °C	80 °C	106 °C	80 °C	80 °C	80 °C	80 °C
pH	controlled	controlled	controlled	uncontrolled	controlled	controlled	controlled	controlled
Wall shear stress	277 Pa	20 Pa	20 Pa	20 Pa	277 Pa	20 Pa	20 Pa	20 Pa
Brine ion. strength	0.51 M	0.615 M	2.31 M	2.31 M	0.51 M	0.51 M	0.615 M	2.31 M
Brine type	*A*	*A*	*B*	*B*	*A*	*A*	*A*	*B*
MSE	*0.034*	*0.011*	*0.024*	*0.057*	*0.362*	*0.053*	*0.009*	*0.008*

not controlled, shown in Fig. 5**A**. Despite all this complexity, and scatter, we can argue that the RF model performed quite well, in all cases staying within the error margins of the experimental results. The MSEs of the predictions range from 0.025 to 0.093 except for the Fig. 5**C** experiment, for which the MSE is 0.430. The same conclusion can be reached by inspecting the additional results shown in Fig. 6 obtained without pre-corrosion, with the MSEs of the predictions ranging from 0.006 to 0.037.

3.3. Sensitivity to input variables

Trained ML models can predict outcomes for conditions that have not been studied experimentally and thus can be employed to perform parametric studies to investigate how systematically changing one input affects the results. In Fig. 7, such a parametric study has been performed to determine the effect of changing the corrosion inhibitor type (A) and concentration (B), CO_2 partial pressure (C), temperature (D), wall shear stress (E), and brine type (F), on the corrosion rates. The baseline corrosion scenario was quite severe: high temperature (130 °C) and high $p_{CO2} = 12$ bar. In the simulation, a single dose of inhibitor was added after 4 h of pre-corrosion and the whole simulation lasted 22 h.

Clearly, the type of inhibitor injected, does not seem to matter, as shown in Fig. 7**A**, which is consistent with the experimental results shown in Fig. 4, Fig. 5, and Fig. 6. The effect of inhibitor concentration is shown in Fig. 7**B** and is logical – an increase in inhibitor concentration between 100 and 300 ppm leads to somewhat lower inhibited corrosion rates, whereas increasing the dose to 1000 ppm offers little or no additional benefits, according to the RF model. This seems to agree with our understanding that at some high inhibitor bulk concentration, a surface saturation concentration is reached [50]. Overall, it shows that the interpolation capability of the RF model (between 100 and 300 ppm) is excellent, and that this carries over to extrapolation when we look at concentrations up to 1000 ppm, which were not used in the experiments. On the other hand, the extrapolation of results towards lower concentrations (using 10 ppm which was never tested in single-dose experiments with pre-corrosion) gave erroneous results – a better inhibitor performance. This points to the danger of extrapolation using ML models such as RF, when the outcome is unpredictable. One possible remedy for this problem is to use active learning, which is a semi-supervised learning algorithm that continuously updates the training and testing sets and increases the extrapolation capability of ML models.

Similar results are seen when the p_{CO2} was varied, as shown in Fig. 7**C**, where the interpolation in the range of $p_{CO2} = 0.5$–12 bar follows the logical trends as seen in the experimental results – higher p_{CO2} leading to higher corrosion rates. However, when the p_{CO2} is increased further (leading to extrapolation) the sensitivity of the corrosion rate to p_{CO2} vanished, what is not an expected result. When it comes to temperature effects, shown in Fig. 7**D,** the interpolation capability of the RF model was fine, and the extrapolation to lower temperatures (e.g., 30 °C) worked as well, leading to even lower corrosion rates and better inhibition, as expected. Finally, as shown in Fig. 7**E** and Fig. 7**F**, there was no effect of the wall shear stress and brine type, as was deduced from the experimental results.

Finally, it is important to mention that throughout the course of this study, the ML modeling effort was continuously discussed with the experts in corrosion science and those with field experience, to ensure that (1) the models were meeting the desired level of accuracy in the predictions and (2) the modeling approach is logical and meaningful when it comes to practical applications. Their input into building, training, evaluating, and interpreting results from ML modeling cannot be overemphasized. Therefore, a hypothetical scenario where the role of corrosion experts can be readily replaced by various ML algorithms seems to be both unrealistic and unreliable. On the other hand, ML algorithm are another excellent addition to the toolbox of corrosion experts that can make them more effective, complement other methods they use to reach conclusions and make predictions, and can improve the corrosion experts' overall productivity and quality of performance. Trained ML models reduce engineering time and testing costs, and these models can continue to learn as new experimental data is obtained, thereby improving in their prediction capabilities.

4. Conclusions

Given that there are many environmental and operational variables that affect the corrosion rate of carbon steel in the presence of corrosion inhibitors, an exhaustive study of all possible combinations of the variables is prohibitive. Therefore, a well-trained ML model can be employed to alleviate the need for such experiments.

- Experiments, conducted by four independent laboratories, measuring the effect of dosage and dose schedule of corrosion inhibitors on the corrosion rates of carbon steel as a function of time are modeled using various supervised machine learning models [Artificial neural networks (ANN), random forest (RF), support vector machine (SVM), and *K* nearest neighbors (KNN)]. In these experiments, the severity of corrosion environment was varied by changing environmental conditions, including temperature, partial pressure of CO_2, brine concentration, ionic strength, wall shear-stress etc.
- The experiments show that the time-profile of corrosion rates depend on the dose schedule of the inhibitors, while the final rates depend mainly on the environment severity.
- Results show that the RF model outperformed the other models for this dataset. The entire time trend of the corrosion rate of mild steel is quite well predicted by the trained RF model with the mean squared error of the prediction in the range from 0.005 to 0.093.
- Sensitivity of corrosion rates to changes in the environmental variables are well-predicted by the trained RF model, which eliminates

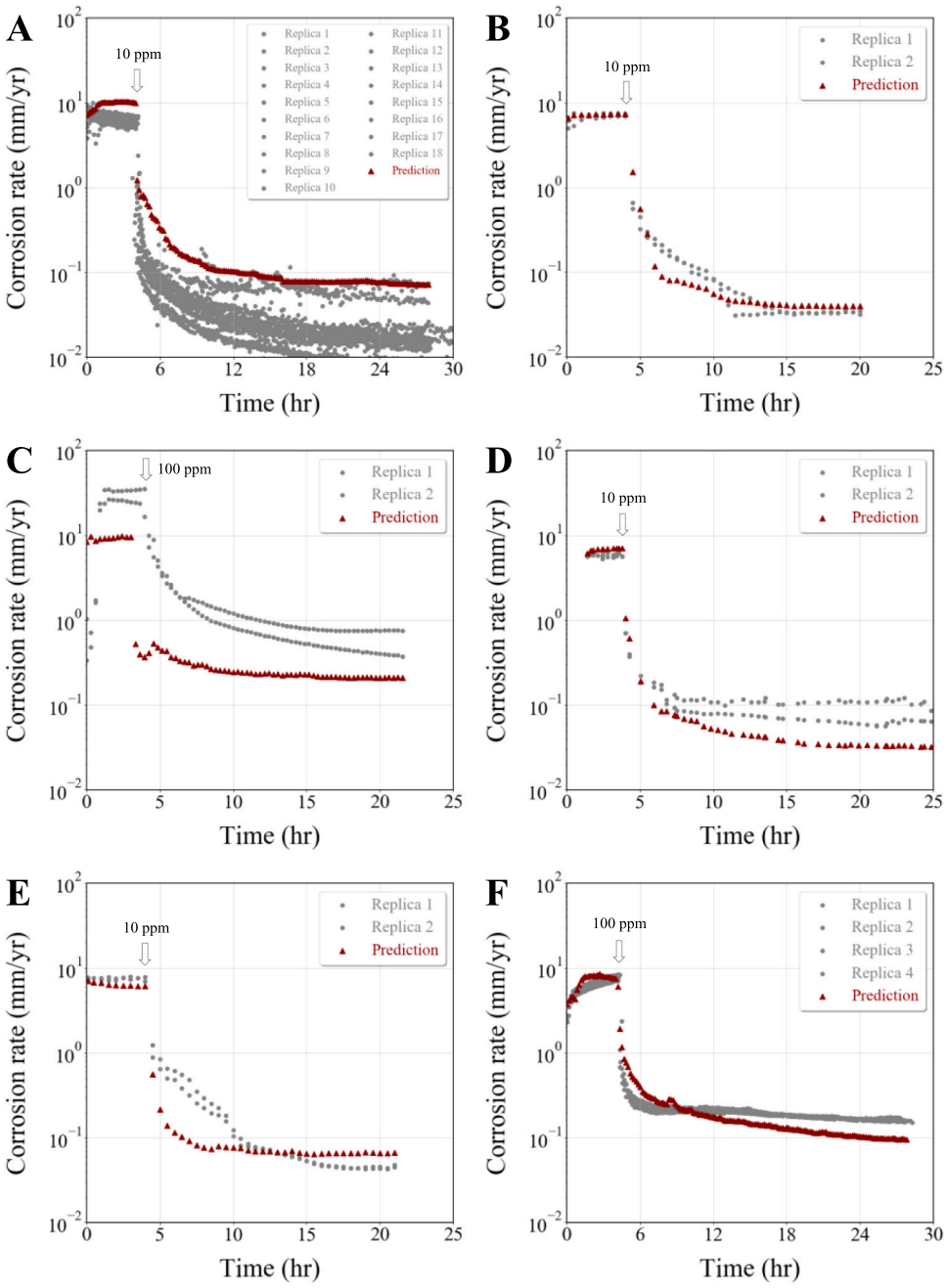


Fig. 5. Comparison between the predicted (red) and the experimentally reported (grey) corrosion rates as a function of time for single dose experiments with pre-corrosion. Below are the details of the experiments.

Conditions / Figure	A	B	C	D	E	F
CI type	*CI-2*	*CI-1*	*CI-1*	*CI-2*	*CI-2*	*CI-2*
Exposure duration	~ 30 hrs.	~ 20 hrs.	~ 22 hrs.	~ 25 hrs.	~ 21 hrs.	~ 28 hrs.
p_{CO2}	0.51 bar	0.51 bar	12 bar	0.51 bar	0.51 bar	0.51 bar
Temperature	80°C	80°C	132°C	80°C	80°C	120°C
pH	uncontrolled	controlled	uncontrolled	controlled	controlled	uncontrolled
Wall shear stress	20 Pa	20 Pa	20 Pa	20 Pa	20 Pa	20 Pa
Brine ionic strength	0.615 M	2.31 M	0.615 M	0.615 M	2.31 M	0.615 M
Brine type	*A*	*A*	*A*	*A*	*A*	*A*
MSE	*0.093*	*0.025*	*0.430*	*0.041*	*0.081*	*0.034*

.

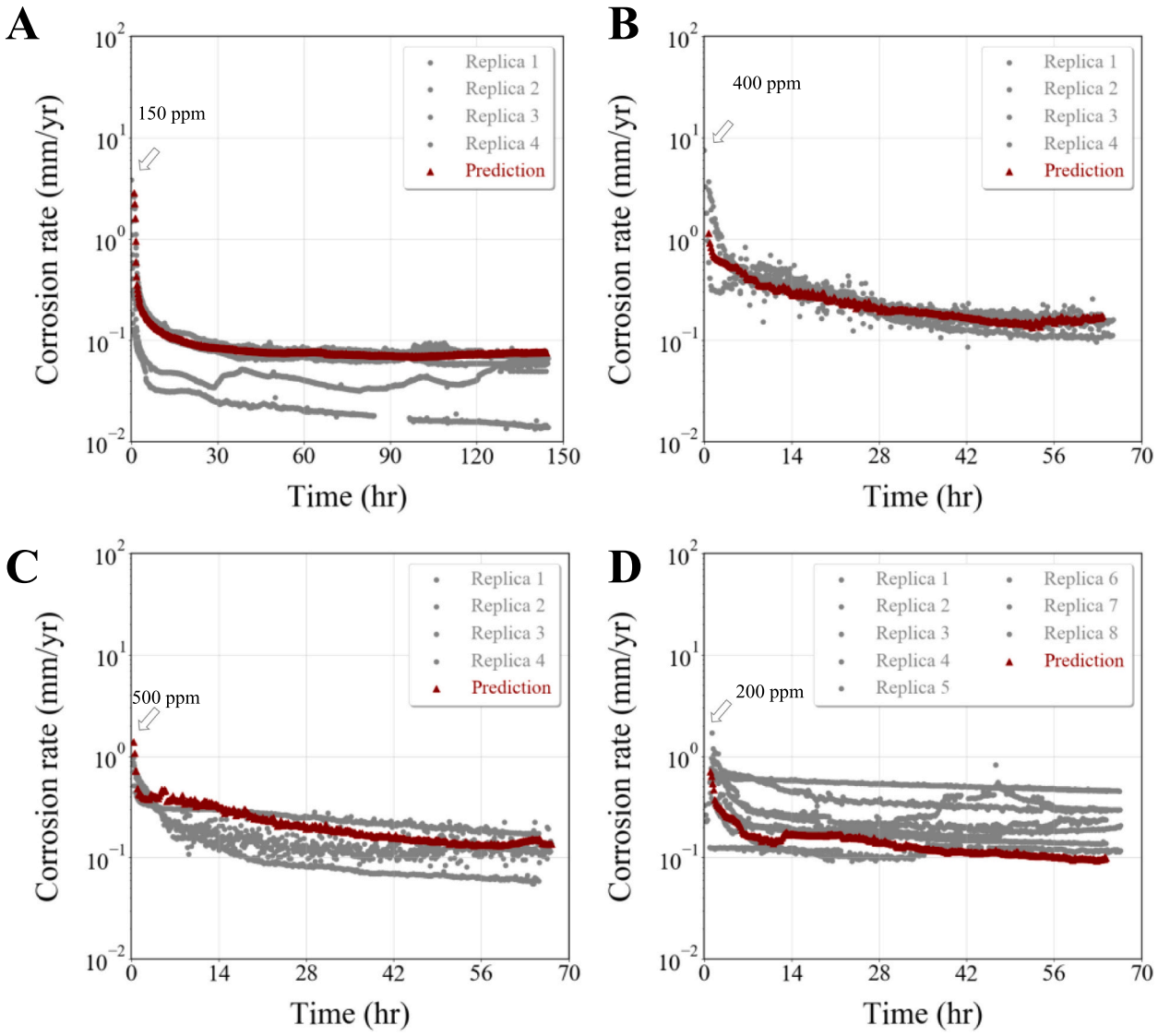


Fig. 6. Comparison between the predicted (red) and the experimentally reported (grey) corrosion rates as a function of time for single dose experiments without pre-corrosion. Below are the details of the experiments.

Conditions / Figure	A	B	C	D
CI type	*CI-1*	*CI-1*	*CI-1*	*CI-1*
Exposure duration	~ 145 hrs.	~ 65 hrs.	~ 68 hrs.	~ 68 hrs.
p_{CO2}	5.0 bar	6.9 bar	6.9 bar	1.57 bar
Temperature	94°C	130°C	130°C	106°C
pH	uncontrolled	uncontrolled	uncontrolled	uncontrolled
Wall shear stress	20 Pa	20 Pa	20 Pa	20 Pa
Brine ionic strength	0.615 M	0.615 M	0.615 M	2.31 M
Brine type	*A*	*A*	*A*	*B*
MSE	*0.007*	*0.009*	*0.037*	*0.006*

.

the need to perform extensive experiments for different solution conditions.

CRediT authorship contribution statement

Mohammadreza Aghaaminiha and Ramin Mehrani: Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft. **Martin Colahan:** Data curation, Methodology, Writing – review & editing. **Bruce Brown:** Conceptualization, Data curation, Methodology, Project administration, Supervision, Writing – review & editing. **Marc Singer:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – review & editing. **Srdjan Nesic:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – review & editing. **Silvia M. Vargas:** Conceptualization, Funding acquisition, Methodology, Supervision, Writing – review & editing. **Sumit Sharma:** Conceptualization, Funding acquisition, Methodology, Project administration, Resources, Supervision, Writing – review & editing.

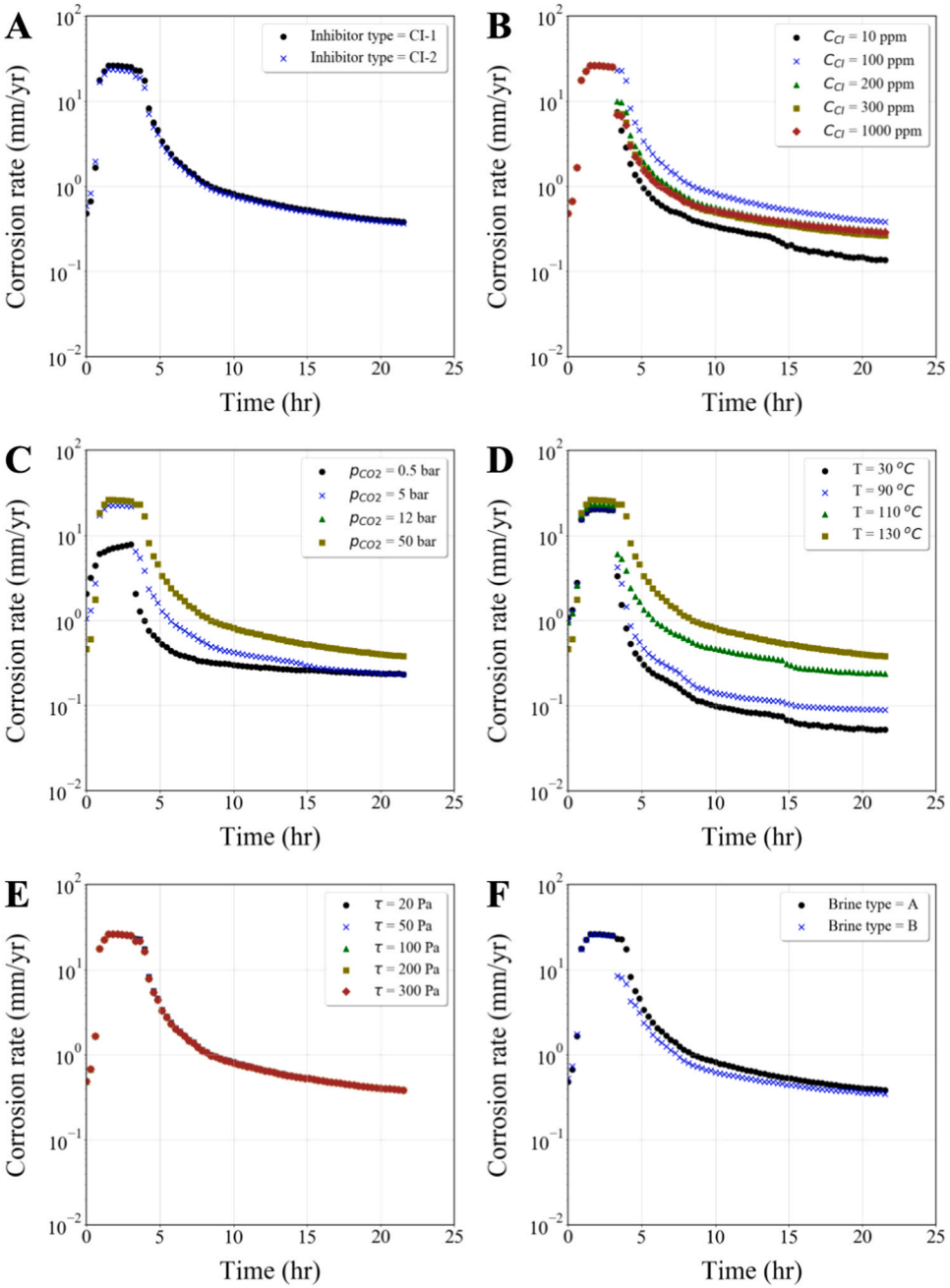


Fig. 7. A parametric study, using a trained RF model, to determine how the time-dependent corrosion rates are expected to vary with the A) inhibitor type, B) inhibitor concentration, C) CO2 partial pressure, D) temperature, E) wall shear stress, and F) brine type. Common condition in all figures, unless otherwise specified is as follow: inhibitor type = CI-1, inhibitor concentration = 100 ppm, exposure duration ~ 22 hrs., pCO2 = 12 bar, T = 130 oC, pH = uncontrolled, wall shear stress = 20 Pa, brine ionic strength = 0.615 M, brine type = A.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability statement

The raw/processed data required to reproduce these findings cannot be shared due to legal reasons. The Python script used for the analysis can be downloaded from the following link: https://github.com/ma947115/corrosion-inhibitor.

Acknowledgment

We would like to acknowledge British Petroleum America Production Company for funding and providing experimental data for this project. BP America Production Company reviewed the manuscript before it was submitted for publication. Computations were performed on Ohio Supercomputer Center (OSC) and National Science Foundation XSEDE supercomputers. The NSF XSEDE grant DMR 190005 is acknowledged for providing access to XSEDE supercomputers. SS thanks Ohio University Supercomputer Support Fund (OUSCSF) and for providing 1:1 matching funds for paying the OSC computational charges.

References

[1] J.A. Martin, F.W. Valone, The existence of imidazoline corrosion inhibitors, Corrosion 41 (1985) 281–287, https://doi.org/10.5006/1.3582003.

[2] A.J. McMahon, The mechanism of action of an oleic imidazoline based corrosion inhibitor for oilfield use, Colloids Surf. 59 (1991) 187–208, https://doi.org/10.1016/0166-6622(91)80247-L.

[3] A. Edwards, C. Osborne, S. Webster, D. Klenerman, M. Joseph, P. Ostovar, M. Doyle, Mechanistic studies of the corrosion inhibitor oleic imidazoline, Corros. Sci. 36 (1994) 315–325, https://doi.org/10.1016/0010-938X(94)90160-0.

[4] M. Jaschke, H.-J. Butt, H.E. Gaub, S. Manne, Surfactant aggregates at a metal surface, Langmuir 13 (1997) 1381–1384, https://doi.org/10.1021/la9607767.

[5] Y. Xiong, B. Brown, B. Kinsella, S. Nešić, A. Pailleret, Atomic force microscopy study of the adsorption of surfactant corrosion inhibitor films, Corrosion 70 (2013) 247–260, https://doi.org/10.5006/0915.

[6] X. Ko, S. Sharma, Adsorption and self-assembly of surfactants on metal–water interfaces, J. Phys. Chem. B 121 (2017) 10364–10370, https://doi.org/10.1021/acs.jpcb.7b09297.

[7] M. Finšgar, J. Jackson, Application of corrosion inhibitors for steels in acidic media for the oil and gas industry: a review, Corros. Sci. 86 (2014) 17–41, https://doi.org/10.1016/j.corsci.2014.04.044.

[8] B.F.M. Pots, Mechanistic models for the prediction of CO2 corrosion rates under multi-phase flow conditions, 1995. ⟨https://www.osti.gov/biblio/106132⟩ (accessed 11 January 2021).

[9] R. Zhang, M. Gopal, W.P. Jepson, Development of a mechanistic model for predicting corrosion rate in multiphase oil/water/gas flows, 1997, 30.

[10] A. Anderko, R.D. Young, M. Plains, Simulation of CO_2/H2S corrosion using thermodynamic and electrochemical models, NACE Corros. Conf. (1999) 19.

[11] S. Nesic, J. Cai, S. Wang, Y. Xiao, D. Liu, Ohio University multiphase flow and corrosion prediction software package MULTICORP V4.0, Ohio Univ. (2004).

[12] P.S. Abdar, M.B. Hariri, A. Kahyarian, S. Nesic, A revision of mechanistic modeling of mild steel corrosion in H2S environments, Electrochim. Acta 382 (2021), 138231, https://doi.org/10.1016/j.electacta.2021.138231.

[13] S. Nesic, Key issues related to modelling of internal corrosion of oil and gas pipelines – a review, Corros. Sci. 49 (2007) 4308–4338, https://doi.org/10.1016/j.corsci.2007.06.006.

[14] E. Gulbrandsen, S. Nesic, A. Stangeland, T. Burchardtt, Effect of precorrosion on the performance of inhibitors for CO_2 corrosion of carbon steel, Corrosion 98 (1998).

[15] Yu.P. Khodyrev, E.S. Batyeva, E.K. Badeeva, E.V. Platova, L. Tiwari, O. G. Sinyashin, The inhibition action of ammonium salts of O,O′-dialkyldithiophosphoric acid on carbon dioxide corrosion of mild steel, Corros. Sci. 53 (2011) 976–983, https://doi.org/10.1016/j.corsci.2010.11.030.

[16] R. Rihan, R. Shawabkeh, N. Al-Bakr, The effect of two amine-based corrosion inhibitors in improving the corrosion resistance of carbon steel in sea water, J. Mater. Eng. Perform. 23 (2014) 693–699, https://doi.org/10.1007/s11665-013-0790-x.

[17] M. Javidi, R. Chamanfar, S. Bekhrad, Investigation on the efficiency of corrosion inhibitor in CO_2 corrosion of carbon steel in the presence of iron carbonate scale, J. Nat. Gas. Sci. Eng. 61 (2019) 197–205, https://doi.org/10.1016/j.jngse.2018.11.017.

[18] S. Xia, M. Qiu, L. Yu, F. Liu, H. Zhao, Molecular dynamics and density functional theory study on relationship between structure of imidazoline derivatives and inhibition performance, Corros. Sci. 50 (2008) 2021–2029, https://doi.org/10.1016/j.corsci.2008.04.021.

[19] A. Kokalj, Formation and structure of inhibitive molecular film of imidazole on iron surface, Corros. Sci. 68 (2013) 195–203, https://doi.org/10.1016/j.corsci.2012.11.015.

[20] Y. Kurapati, S. Sharma, Adsorption free energies of imidazolinium-type surfactants in infinite dilution and in micellar state on gold surface, J. Phys. Chem. B 122 (2018) 5933–5939, https://doi.org/10.1021/acs.jpcb.8b02358.

[21] S. Sharma, X. Ko, Y. Kurapati, H. Singh, S. Nesic, Adsorption behavior of organic corrosion inhibitors on metal surfaces—some new insights from molecular simulations, Corrosion 75 (2018) 90–105, https://doi.org/10.5006/2976.

[22] H. Singh, S. Sharma, Disintegration of surfactant micelles at metal–water interfaces promotes their strong adsorption, J. Phys. Chem. B 124 (2020) 2262–2267, https://doi.org/10.1021/acs.jpcb.9b10780.

[23] H. Singh, S. Sharma, Free energy profiles of adsorption of surfactant micelles at metal-water interfaces, Mol. Simul. 0 (2020) 1–8, https://doi.org/10.1080/08927022.2020.1780231.

[24] S. Nesic, M. Nordsveen, N. Maxwell, M. Vrhovac, Probabilistic modelling of CO_2 corrosion laboratory data using neural networks, Corros. Sci. 43 (2001) 1373–1392, https://doi.org/10.1016/S0010-938X(00)00157-8.

[25] W.T. Nash, C.J. Powell, T. Drummond, N. Birbilis, Automated corrosion detection using crowdsourced training for deep learning, Corrosion 76 (2019) 135–141, https://doi.org/10.5006/3397.

[26] C.I. Ossai, A data-driven machine learning approach for corrosion risk assessment—a comparative study, Big Data Cogn. Comput. 3 (2019) 28, https://doi.org/10.3390/bdcc3020028.

[27] G. Sanchez, W. Aperador, A. Cerón, Corrosion grade classification: a machine learning approach, Indian Chem. Eng. 62 (2020) 277–286, https://doi.org/10.1080/00194506.2019.1675539.

[28] L. Yan, Y. Diao, Z. Lang, K. Gao, Corrosion rate prediction and influencing factors evaluation of low-alloy steels in marine atmosphere using machine learning approach, Sci. Technol. Adv. Mater. 21 (2020) 359–370, https://doi.org/10.1080/14686996.2020.1746196.

[29] B.A. Salami, S.M. Rahman, T.A. Oyehan, M. Maslehuddin, S.U.Al Dulaijan, Ensemble machine learning model for corrosion initiation time estimation of embedded steel reinforced self-compacting concrete, Measurement 165 (2020), 108141, https://doi.org/10.1016/j.measurement.2020.108141.

[30] X. Gong, C. Dong, J. Xu, L. Wang, X. Li, Machine learning assistance for electrochemical curve simulation of corrosion and its application, Mater. Corros. 71 (2020) 474–484, https://doi.org/10.1002/maco.201911224.

[31] R. Aulia, H. Tan, S. Sriramula, Prediction of corroded pipeline performance based on dynamic reliability models, Procedia CIRP 80 (2019) 518–523, https://doi.org/10.1016/j.procir.2019.01.093.

[32] A. Abass, K. Wada, H. Matsunaga, H. Remes, T. Vuorio, Quantitative characterization of the spatial distribution of corrosion pits based on nearest neighbor analysis, Corrosion 76 (2020) 861–870, https://doi.org/10.5006/3551.

[33] T.M. Mitchell, Does machine learning really work?, 11–11, AI Mag. 18 (1997), https://doi.org/10.1609/aimag.v18i3.1303.

[34] A. Géron, Hands-on machine learning with scikit-learn, keras, and tensorflow: concepts. Tools, and Techniques to Build Intelligent Systems, O'Reilly Media, Inc, 2019.

[35] L. Breiman, Random forests, Mach. Learn 45 (2001) 5–32, https://doi.org/10.1023/A:1010933404324.

[36] L. Breiman, Arcing classifier (with discussion and a rejoinder by the author), Ann. Stat. 26 (1998) 801–849, https://doi.org/10.1214/aos/1024691079.

[37] M. Aghaaminiha, S.A. Ghanadian, E. Ahmadi, A.M. Farnoud, A machine learning approach to estimation of phase diagrams for three-component lipid mixtures, Biochim. Biophys. Acta BBA - Biomembr. 1862 (2020), 183350, https://doi.org/10.1016/j.bbamem.2020.183350.

[38] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw. 4 (1991) 251–257, https://doi.org/10.1016/0893-6080(91)90009-T.

[39] J.M. Benitez, J.L. Castro, I. Requena, Are artificial neural networks black boxes? IEEE Trans. Neural Netw. 8 (1997) 1156–1164, https://doi.org/10.1109/72.623216.

[40] R. Bala, D.D. Kumar, Classification using ANN: a review, Int. J. Comput. Intell. Res. 13 (2017) 1811–1820.

[41] S.R. Gunn, Support vector machines for classification and regression, ISIS Tech. Rep. 14 (1998) 5–16.

[42] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1967) 21–27, https://doi.org/10.1109/TIT.1967.1053964.

[43] S. Xia, Z. Xiong, Y. Luo, L. Dong, G. Zhang, Location difference of multiple distances-based k-nearest neighbors algorithm, Knowl. -Based Syst. 90 (2015) 99–110, https://doi.org/10.1016/j.knosys.2015.09.028.

[44] M. Claesen, B. De Moor, Hyperparameter Search in Machine Learning, ArXiv150202127 Cs Stat., 2015.

[45] A. Famili, W.-M. Shen, R. Weber, E. Simoudis, Data preprocessing and intelligent data analysis, Intell. Data Anal. 1 (1997) 3–23, https://doi.org/10.3233/IDA-1997-1102.

[46] S.B. Kotsiantis, D. Kanellopoulos, P.E. Pintelas, Data preprocessing for supervised leaning, Int. J. Comput. Sci. 1 (2006) 111–117.

[47] M. Aghaaminiha, R. Mehrani, T. Reza, S. Sharma, Comparison of machine learning methodologies for predicting kinetics of hydrothermal carbonization of selective biomass, Biomass. Convers. Biorefin. (2021), https://doi.org/10.1007/s13399-021-01858-3.

[48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, É. Duchesnay, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[49] N. Moradighadi, S. Lewis, J.D. Olivo, D. Young, B. Brown, S. Nešić, Determining critical micelle concentration of organic corrosion inhibitors and its effectiveness in corrosion mitigation, Corrosion 77 (2020) 266–275, https://doi.org/10.5006/3679.

[50] T. Murakawa, S. Nagaura, N. Hackerman, Coverage of iron surface by organic compounds and anions in acid solutions, Corros. Sci. 7 (1967) 79–89, https://doi.org/10.1016/S0010-938X(67)80105-7.