Applied Soft Computing Journal 96 (2020) 106535

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Fuzzy regression functions with a noise cluster and the impact of outliers on mainstream machine learning methods in the regression setting

Srinivas Chakravarty [a], Haydar Demirhan [a,*], Furkan Baser [b]

[a] School of Science, Discipline of Mathematical Sciences, RMIT University, Melbourne, VIC, Australia
[b] Department of Actuarial Sciences, Faculty of Applied Sciences, Ankara University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 7 February 2020
Received in revised form 1 June 2020
Accepted 5 July 2020
Available online 11 July 2020

Keywords:
Artificial neural network
Fuzzy regression function
Outlier
Robustness
Support vector machine

ABSTRACT

The presence of outliers in the dependent and/or independent features distorts predictions with machine learning techniques and may lead to erroneous conclusions. It is important to implement methods that are robust against the outliers to make reliable predictions and to know the accuracy of the existing methods when data is contaminated with outliers. The focus of this study is to propose a robust fuzzy regression functions (FRFN) approach against the outliers and evaluate the performance of the proposed and several mainstream machine learning approaches in the presence of outliers for the regression problem. The proposed FRFN approach is based on fuzzy k-means clustering with a noise cluster. We compare the accuracy of Artificial Neural Networks (ANN), Support Vector Machines (SVM) and the proposed FRFN approaches with different training algorithms/kernel functions via simulated and real benchmark datasets. In total, accuracies of 36 ANN, SVM, and FRNF implementations with training algorithms and kernel and loss functions have been evaluated and compared to each other with samples containing outliers via a Monte Carlo simulation setting. It is observed in both Monte Carlo simulations and applications with benchmark dataset that FRFN with ANN trained with Bayes regularization algorithm and FRFN with SVM with Gaussian kernel outperforms the classical implementations of ANN and SVMs under the existence of outliers. The proposed noise cluster implementation considerably increases the robustness of fuzzy regression functions against outliers.

1. Introduction

There is a large body of research that proposes the use of Machine Learning (ML) methods such as Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Fuzzy Regression Functions (FRFs), Adaptive Neuro-Fuzzy Inference System, etc. to explain or predict a dependent feature using a set of independent features in a regression setting. ML methods constitute a large bunch of highly accurate algorithms for the analysis of challenging datasets with great accuracy. The observations that are located significantly far from the center of a dataset are considered to be unusual observations and called outliers [1]. Outliers, if not adequately managed, can potentially distort the results of analyses by ML methods such as SVMs, ANNs, and FRFs [2–7]. In practice, ML algorithms are trained over the training split of a dataset and validated over the test split of the same dataset. When there are outlier observations in the dataset, they are randomly allocated to either training or test sets during the implementation. If most of the outliers randomly go into the training dataset, the model is trained under the impact of extreme information due to the outliers; hence, we do not get accurate validation results over the test set. If most of the outliers randomly get allocated to the test set, then we will get misleading test performance for models. The k-fold cross-validation procedure has a potential to mitigate the impact of outliers. Due to the averaging done in cross-validation, the impact of outliers is reduced to some degree, but they still have an impact on the results due to the random allocation of observations to the folds.

Although there is considerable literature around the use of ML methods for the detection of outliers in various contexts, the number of studies that are focused on the impact of outliers on ML methods for both regression and classification problems is quite limited. Research on improving the robustness of the SVM against outliers has recently started to appear in ML literature. Yang and Dong [2] proposed a new robust loss function and introduced robust SVM frameworks for both classification and regression. They compared the performance of the least squares support vector regression (LSSVR), weighted LSSVR, SVR with pinball loss function, and robust SVR which is found superior

* Corresponding author.
E-mail address: haydar.demirhan@rmit.edu.au (H. Demirhan).

https://doi.org/10.1016/j.asoc.2020.106535
1568-4946/

to the other SVRs. In their study, very high magnitude outliers, which are very far from the center of the dataset, are created synthetically by multiplying 20% of the benchmark datasets by 10. In the same context, Xu et al. [8] and Xing and Ji [9] worked on the rescaled hinge loss functions to make the SVM more robust against outliers for the classification problem. Zhu et al. [10] proposed another SVM approach based on pinball loss function and criticized the hinge loss SVM to be prone to the outliers for the classification problem. Recognizing that the performance of ML methods can be negatively impacted by outliers and insignificant features in the data, Li and Zhou [1] proposed a model averaging method accompanied by an outlier detection method to make ML more robust to such contaminants. Panagopoulos et al. [5] designed an SVM algorithm to be used under the existence of outliers in the regression problem. They conducted an experimental study with benchmark datasets to compare the accuracy of their approach to other SVM variations and standard regression methods and observed a considerable gain in accuracy with their approach. In predicting short-term traffic conditions accurately, Yan and Yu [11] proposed a model which deals effectively with the negative effects of outliers for SVM, thereby enhancing the robustness and the ability to generalize. A homotopy algorithm has been proposed by Suzumura et al. [12] in the SVM context to obtain a good local solution from a non-convex optimization problem and to also optimize the robustness–efficiency trade-off. The algorithm computes robust SVM solutions by gradually suppressing the influence of outliers as simulated annealing. Singla and Shukla [13] provides a comprehensive review of SVMs and their variants that employ robust-statistics against outliers until 2019.

For ANN, the literature around the outlier problem is more focused on the detection of outliers rather than their impact on the accuracy of the method. However, there are recent studies focusing on the use of robust statistics along with neural networks. Araújo Júnior et al. [14] compared the performance of ANN to linear and quantile regressions and observed a superior performance with ANN. They asserted that ANN is a very robust approach in the presence of outliers. Egrioglu et al. [4], Yolcu et al. [15], Aladag et al. [16] used robust statistics to improve the strength of ANN against the outliers for forecasting with time series data. Bas et al. [17] proposed a robust ANN algorithm against outliers based on the multiplicative neuron models that use Huber's loss function as fitness function for time series data. Lin et al. [18], Liao et al. [19], and Beliakov et al. [20] proposed neural networks based on least trimmed squares estimators to gain robustness against outliers in the regression setting. Khamis et al. [21] worked on the impact of the percentage and magnitude of outliers on an ANN with sigmoid activation function. Their numerical study is limited in the sense that it considers only one activation function. They observed that when the training data features contain outliers, the accuracy of the modeling decreases as the concentration and the magnitude of outliers increase. They concluded that variations in the percentage and magnitude of outliers in the test data may affect modeling accuracy at higher levels.

Although robust statistical methods have been hybridized with ANNs to strengthen the robustness of ANNs against outliers, kernel functions are more utilized in the fuzzy context. Fuzzy methods that employ least squares as their objective function are weak against the outliers [22]. Oi and Endo [22] replaced the least squares with the least absolute deviation to overcome this weakness. Akbari and Hesamian [23] used an extended version of the classical kernel fitting to estimate the fuzzy smooth function and proposed a kernel-based weighted criterion to reduce the sensitivity of the method to outliers. They evaluated the proposed approach for fuzzy regression models with fuzzy predictors and fuzzy responses over a small-scale simulated data and an example dataset and observed that the method is insensitive to the outliers. Neto and de Carvalho [24] proposed the use of exponential-type kernel functions to introduce a new robust regression approach for interval-valued observations as in the fuzzy regression. Their approach penalizes having outliers in the mid-points of the fuzzified observations and its performance is compared to other robust interval regression methods via an extensive Monte Carlo study. Li et al. [25] used the least absolute deviation trapezoidal fuzzy numbers to propose a robust fuzzy regression approach to the presence of outliers. They tested the robustness of their approach using some small-scale numerical examples. Kula and Apaydin [6,7] proposed using triangular fuzzy numbers along with a weight matrix defined over the membership function of the residuals, when independent variables are crisp and the dependent variable is fuzzy. In fuzzy cluster analysis, Davé [26] proposed assigning noisy observations into a noise cluster to mitigate their impact on the cluster centers. None of these studies compares the accuracy of fuzzy methods to their straightforward alternatives, SVMs and ANNs, in the presence of outliers.

A typical approach to handle outliers is to eliminate them based on outlier detection algorithms [27,28] or to replace them with other defined values such as mean [29] if the researcher does not rely on a method that is claimed to be robust against outliers. It is important to understand the source of outliers in the data and keep them in the analysis if they are not measurement errors and generated by the system under consideration. When the outliers are not dropped from the dataset, it is essential to understand their impact on the accuracy of the predictions. SVMs and ANNs are widely employed for the regression problem alternatively to each other and readily implemented by software packages. The FRF method is used along with SVM at the inference phase in the literature [30–33]. Therefore, it introduces fuzzification before the implementation of SVM at the inference phase. The implementation of FRF method requires traditional fuzzy k-means clustering proposed by Bezdek [34]. Although fuzzification would be useful to mitigate the impact of outliers, traditional fuzzy k-means clustering is still prone to the outliers in the dataset. To improve the clustering step, we propose to replace fuzzy k-means clustering with the noise cluster approach of Davé [26]. There is no knowledge of the impact of this fuzzification and the noise cluster approach on the robustness of the FRF method in the literature. To the best of our knowledge, no study in the literature focuses on the impact of outliers on FRF and compares accuracy of the SVM, ANN, and FRF methods under different scenarios of outlier contamination for regression problems. To address these gaps and demonstrate the accuracy of FRF with noise cluster (FRFN), we employ both SVM and ANN at the inference phase of FRFN to create FRFN.SVM and FRFN.ANN approaches and conduct numerical studies with the Monte Carlo (MC) method and real benchmark datasets. Our MC study includes scenarios composed of the combinations of different sample sizes, test and training contamination rates, and magnitudes of outliers. Linear, polynomial, Gaussian, and sigmoid kernel functions and pinball loss function are considered for SVM. Use of Huber loss function and different training algorithms is considered to assess the robustness of ANN against outliers in the literature. We consider 12 distinct algorithms to train ANNs in our study. Corresponding FRFN versions are included in the study for all considered SVM and ANN implementations. In total, we simultaneously compare and test the robustness of 36 different ML approaches composed of ANNs with 12 different training algorithms and one with the Huber loss function, SVMs with 4 different kernels and one with the pinball loss function, and 18 FRFN implementations against outliers. Huber and pinball loss functions are recent approaches

shown to be robust against outliers in ANN and SVM contexts in independent studies under different conditions. The results of our study shed a light on the robustness and accuracy of SVM and ANN under the existence of outlier observations. We demonstrate how we can improve the robustness of both SVM and ANN methods against outliers when they are implemented under a FRFN approach. The results reveal that the fuzzification step and fuzzy k-means clustering with a noise cluster under FRF framework are successful in the mitigation of the impact of contamination by outliers. We also conduct a numerical study on 15 datasets from UCI benchmark, R data repository, and Kaggle database [35–37] to compare the performances of the 36 ML approaches under 4 different outlier contamination rates to imitate zero, low, moderate, and high levels of contamination in practice in our simulations by injecting outliers into real data. The results of both MC simulation and numerical studies present performance comparisons among the training algorithms of neural networks, kernel functions of SVMs, and their FRFN versions when there is no outlier in the dataset as well as when outliers at different rates of contamination and magnitudes are introduced to the dataset.

The motivations of this study are to (i) propose a robust FRFN framework against contamination of data with outlier observations, (ii) discover the impact of different magnitudes of outlier contamination on the accuracy of mainstream and recently prosed ML methods as well as the proposed FRFN framework, and (iii) assess the accuracy of ML methods with different training algorithms, kernel and loss functions when there is no outlier contamination. Contributions of this study are (i) development of fuzzy regression functions approaches with a noise cluster with ANNs and SVMs to robustify these ML methods against outliers, (ii) an extensive comparison of the accuracies of ANNs and SVMs and the proposed FRFNs with different training algorithms, kernel and loss functions under the existence of outliers in the research data using Monte Carlo simulations and benchmark datasets, and (iii) accuracy comparisons of all the mentioned methods when the dataset is uncontaminated with outliers using Monte Carlo simulations and benchmark datasets. To achieve the contributions, we (i) utilize the noise cluster approach to handle the impact of outliers at the classification step of the fuzzy regression functions approach, (ii) employ ANNs within the FRFN framework for the first time, and (iii) implement ANNs and SVMs with many different training algorithms, kernel and loss functions under an extensive simulation space and 15 benchmark datasets.

In Section 2, the methodologies behind SVM and ANN methods are given and the implementation of FRFNs is explained. The impact of outliers on each method is discussed in terms of the theoretical characteristics of the methods. In Section 3, the MC simulation study and its results are presented. The simulation space of the numerical study is described and the impacts of outliers on each of the considered ML methods are presented. In Section 4, the datasets used for the numerical study are introduced and the performance results of the considered ML approaches with real datasets are presented. In Section 5, discussions and conclusions are presented and recommendations are made for the use of the considered ML methods under different conditions.

2. Methods

2.1. Artificial neural networks

ANNs are designed to learn in a way similar to how the human brain does [38]. They are composed of artificial neurons, edges, and a number of layers where artificial neurons are generally aggregated into [38,39]. An ANN is used for classification as well as regression tasks [40].

Let x_l and y_k, $l = 1, \ldots, N_i$; $k = 1, \ldots, N_0$ be input and output vectors, respectively. The input vector is transmitted through the hidden layer defined as in Eq. (1):

$$z_j = \sum_{l=1}^{N_i} \omega_{jl} x_l + \omega_{j0}, \tag{1}$$

where $j = 1, \ldots, N_k$, N_k is the number of hidden layers, ω_{jl} is the weight of input x_l, and ω_{j0} is the bias in layer j. The inputs pass through a layer of activation function, $f\left(z_j\right)$, that can be sigmoid or hyperbolic tangent function and accounts for the nonlinearity of the relationship between inputs and outputs. Then the output is created by $y_k = f\left(v_k\right)$, where $j = 1, \ldots, N_0$ and $\left(v_k\right)$ is another layer of filters [41].

The impact of outliers on ANN is directly related to the weights, biases, and the activation function used in the structure of a neural network. In this study, we use sigmoid activation function within multilayer perceptron structure. We consider Huber loss function with ANN, which is recently considered by Bas et al. [17] for contaminated time series data. We also consider 12 distinct training algorithms to estimate the weights. Among these algorithms, Gradient Descent Backpropagation (GD) [41] algorithm updates the weights and biases in the direction of the negative gradient. The learning speed of the GD method is directly proportional to the speed of learning through the learning rate parameter. A large learning rate defines bigger steps in the search and makes the algorithm unstable whereas a small learning rate lowers the speed of convergence. In the search directions where outlier observations are clustered, learning rate can artificially increase and cause the algorithm to be overtrained with outliers. Gradient Descent with Momentum (GDM) [41] is a variation of GD algorithm with the ability to skip local minimums on the error surface. This feature makes GDM algorithm more robust against low-valued outliers than GD algorithm. Variable Learning Rate Gradient Descent (GDX) [41] is another variation of GD algorithm. GDX uses the momentum feature to avoid local minima and employs a variable learning rate to have the flexibility to use a larger learning rate when it is far from the optimal solution and a smaller learning rate when it is close to the optimal. Then again, the outliers can confuse the algorithm in the selection of variable learning rate such that when it is close to the optimal, a cluster of outliers can misleadingly push the algorithm to use a larger learning rate. We implemented damped least-squares with Levenberg–Marquardt (LM) [42] algorithm which uses larger step sizes based on the gradient descendent rule at the beginning and smaller step sizes as it progresses. LM is expected to be robust against outliers due to the adaptation of the step sizes while searching for the optimum. The impact of outliers would decrease by decreasing step sizes near the optimum. Bayesian Regularization (BR) [43,44] algorithm uses Gaussian priors on weights along with maximum likelihood learning with the Gaussian noise to produce more robust weights against overfitting. BR algorithm is designed to mitigate overfitting problem occurs during the neural network training [44]. In this sense, we expect BR to be robust against the existence of outliers since it does not get overtrained by the outliers. Broyden–Fletcher–Goldfarb–Shanno Quasi-Newton (BFG) [45] algorithm is based on the second order partial derivatives of the error function with respect to the weights and biases. Therefore, BFG algorithm considers unit change in the direction of error function at the values of the weights and biases through the optimization iterations. The outliers impact the amount of unit change in the BFG algorithm through the second derivatives; hence, we expect the BFG algorithm to be affected by the outliers in the training set. One Step Secant (OSS) [46] algorithm uses the identity matrix instead of the matrix of second order partial derivatives (Hessian

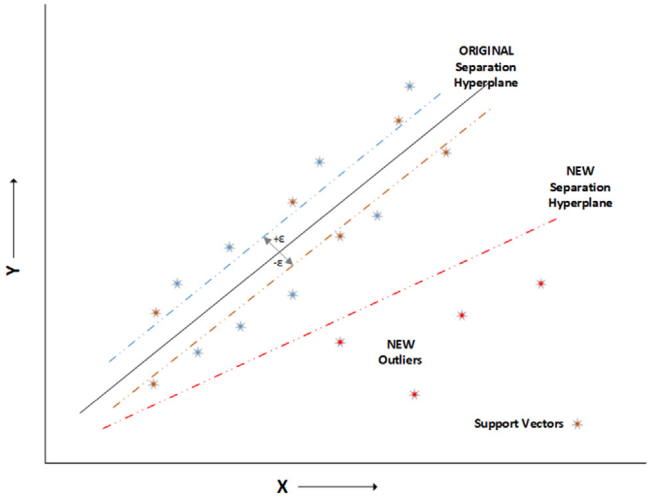

Fig. 1. General description of SVM in the presence of outliers.

matrix) of the previous iteration. The influence of outliers on OSS can be lower than BFG due to the use of identity matrix instead of a possibly contaminated Hessian matrix at every new iteration. Resilient Backpropagation (RB) [47] algorithm is based on the sign of the partial derivative for each weight independently. Since the sign of the gradient is related to the direction of search (not to the magnitude), this feature makes RB less prone to the outliers in the training split. Scaled Conjugate Gradient (SCG) [48] algorithm uses LM method to scale the step sizes in the search direction. Due to its utilization of LM algorithm, SCG can also be impacted by the outliers in the training dataset. Conjugate Gradient with Powell/Beale Restarts (CGB) [49] algorithm resets the search direction based on an orthogonality test between current and previous gradient. Outliers can be influential on the test to reset the search direction; hence, they can cause delusive direction changes during the training. Fletcher–Powell Conjugate Gradient (CGF) [50] employs the ratio of norm squared of current and previous gradients to set the new search direction during the search. Like CGB, outliers can cause false direction changes in the CGF algorithm. Polak–Ribiére Conjugate Gradient (CGP) [41] algorithm replaces norm squared of current gradient with the inner product of the previous change in the gradient and the current gradient. So, the magnitude of change is included in the selection of the new direction in the gradient search. CGP will be impacted by the outliers more than both CGB and CGF since it includes both the norm squared and the step size in the calculation of the new direction. We utilized MATLAB® 2019b Statistics and Machine Learning Toolbox for the implementation of all training algorithms with their default argument values and **ANN2** R package for the implementation of Huber loss function [51].

2.2. Support vector machines

SVMs are supervised learning models with linear algorithms which can analyze data for classification and regression tasks [38]. General description of SVM is given in Fig. 1 [52].

Fig. 1 shows a hyperplane labeled "ORIGINAL Separation Hyperplane", which depicts the data points with a margin of $\pm\varepsilon$ between the support vectors and the hyperplane. As soon as a new outlier point is introduced, the hyperplane moves to a new position (labeled "NEW Separation Hyperplane") due to the pull effect exerted by the outliers. We consider SVM quantile regression with the pinball loss function as a potential method against outliers [13]. The detrimental impacts of outliers, such as margin reduction and breaking of linear separability can be minimized to various degrees using different kernels. Mainly used kernels are Gaussian, linear, polynomial, and sigmoid kernels [53–55]. Since the linear kernel does not incorporate any functional form to mitigate the impact of outliers, it is the most susceptible kernel to outliers. A higher order with polynomial kernel can show a better performance on a test set but can lead to overfitting [55] as well. The Gaussian kernel is the most common one used in practice [55]. Since the Gaussian kernel contains a negative exponent with a squared term, causing outliers to have a reduced impact, we expect to get most robust results with this kernel among SVM implementations. The Sigmoid kernel assigns a higher degree of confidence to a test point which is farther than the hyperplane as compared to a point which is closer to the hyperplane [53]. Since the Sigmoid kernel is designed to treat the impact of large negative values and large positive values of the features as being similar, and depending on the steepness of its slope, the Sigmoid kernel would give robust results to outliers. In this study, we implemented these kernels using MATLAB® 2019b Statistics and Machine Learning Toolbox with their default settings. SVMs with Pinball loss function are implemented using **qrsvm** R package [56].

2.3. Fuzzy regression functions

As discussed by Baser and Demirhan [57], the method of FRFs results in an improved closeness of system outputs and model outputs compared to the results from traditional fuzzy rule-based mechanisms. Under FRF, the system structure is analyzed by using the fuzzy c-means (FCM) clustering algorithm. By using FRF method, additional membership values are obtained through FCM clustering when describing the relationship between the input features and the target output [30,32].

FRF systems involve training and inference processes. Training data consists of randomly selected observations from the complete data. Testing or validation data is used in the inference process to assess the trained model. Implementation of FRF method is outlined in Fig. 2. The crucial steps of FRF implementation are clustering I/O data (clustering step) and the estimation of FRF cluster parameters (estimation step). At the clustering step, any outliers in the training dataset will be influential since they stay away from the center of the dataset constituting a separate cluster. The previous implementations of FRF methods include FCM clustering at this step. The estimation step is where we apply any other ML method to estimate the parameters of FRFs for each cluster. Mainly, SVMs are employed at this step. The training stage consists of five steps described in Algorithm C.1 and the inference stage consists of four steps described in Algorithm C.2 of Appendix C given in the supplementary material.

The estimation of FRF cluster parameters determines the full FRF method. In this study, we consider SVM and ANN to create FRF–SVM and FRF–ANN approaches. FRF–SVM is implemented by Baser and Demirhan [57] and to the best of our knowledge there is no implementation of FRF–ANN in the literature. Although the impact of outliers on the accuracy of FRF is expected to be lower than both SVM and ANN implementations due to the fuzzification step, the FCM clustering is still prone to the impact of outliers. In this study, we improve on FRF approach to make it more robust against the outliers. Clustering methods, including the FCM, can severely be biased when there are even a few outliers in the dataset due to the noise created by the outliers [26,58]. Fig. 3 outlines the proposed FRFN approach. We use a fuzzy clustering method that is robust against outliers and introduce ANN into FRF implementation to robustify FRF against impact of outlier observations.

Davé and Sen [58] propose to create a noise cluster within the objective functional type clustering algorithms such as fuzzy

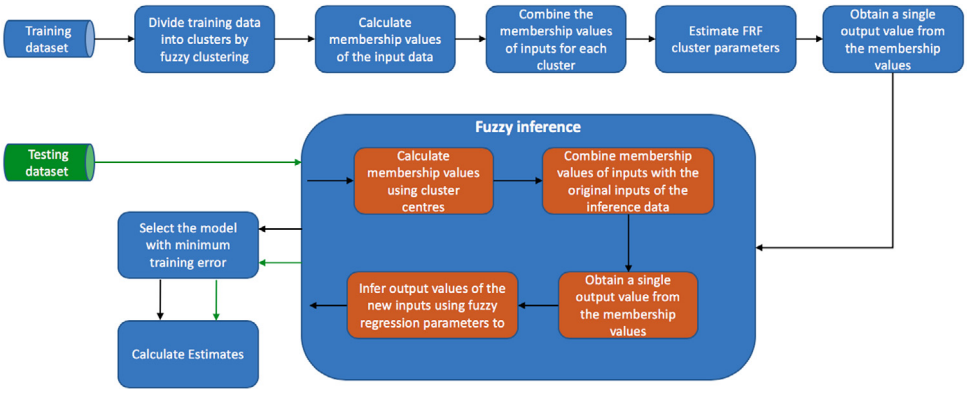

Fig. 2. Graphical description of FRF method.

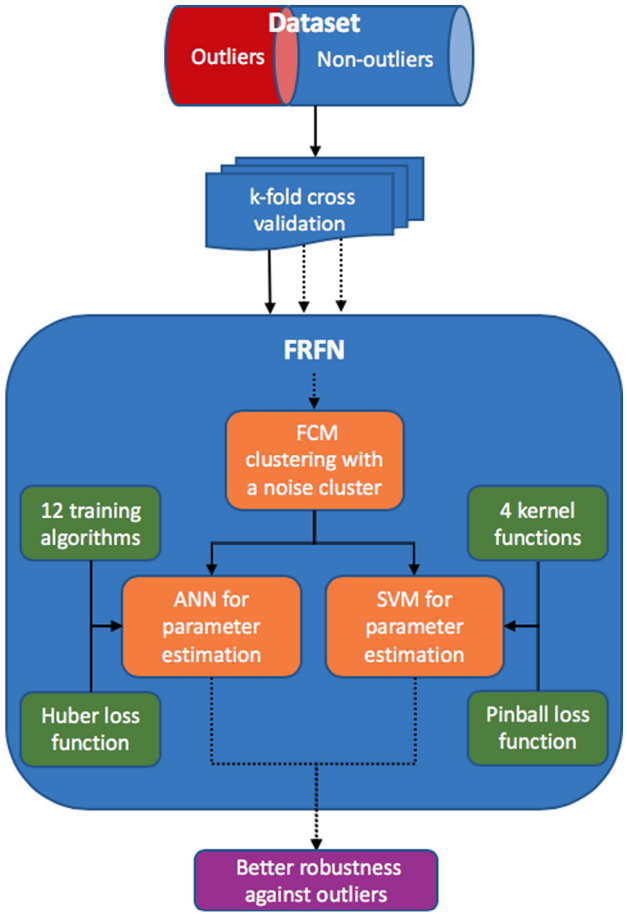

Fig. 3. Flowchart of the proposed fuzzy regression functions with noise cluster approach.

k -means (FKM). Then, the noisy outlier values are pushed into a *noise cluster* to avoid the bias they create. At the beginning of clustering, all points have equal *a priori* probability of belonging to the noise cluster and this probability increases for outliers and decreases for non-outlier observations as the algorithm progresses with updating cluster memberships and centers. We modify FRF approach by replacing fuzzy c-means clustering with fuzzy k-means clustering with the noise cluster. We abbreviate this approach as FRFN throughout the article. The FRFN approach is expected to be robust against the outliers due to both fuzzification and treating outliers in a separate noise cluster. Optimum membership values and the number of clusters are determined using cluster validity indexes such as Bezdek's partition coefficient, Xie-Beni index, partition entropy index, weighted Inter-intra index, fuzzy silhouette width [59]. FKM noise clustering algorithm determines both the membership values and the number of clusters through its iterations using cluster validity indexes. This is a desired feature of noise cluster approach since the optimization of multiple indexes is not fully objective and unfeasible for simulation studies where the identification of the number of clusters and the membership values needs to be carried out by the researcher for each simulated dataset.

Suppose $(\mathbf{x}_i, y_i)$ is an observation in the training sample, $\mathbf{v}_k(\mathbf{x}, y)$ shows the cluster centers and $\mu_{k,i}(\mathbf{x}, y) \in [0, 1]$ represents interactive (input–output) fuzzy membership values for $i = 1, 2, \ldots, n$, $k = 1, 2, \ldots, c$, n is the number of observations in the training dataset, and $n > c$. Then, the membership values are obtained as in Eq. (2) [30,32]:

$$\mu_{k,i}(\mathbf{x}, y) = \left(\sum_{l=1}^{n} \left(d_{k,i}(\mathbf{x}, y) / d_{l,i}(\mathbf{x}, y) \right)^{2/m-1} \right)^{-1}, \tag{2}$$

where $m > 1.1$ is the degree of fuzziness, which is set to 2 in this study, and

$$d_{k,i}(\mathbf{x}, y) = \left\| (\mathbf{x}_i, y_i) - \mathbf{v}_i(\mathbf{x}, y) \right\|. \tag{3}$$

The cluster centers are found by optimization of the function in Eq. (4) [34]:

$$f(\mu, \mathbf{v}) = \sum_{i=1}^{n} \sum_{k=1}^{c} \left[\mu_{k,i}(\mathbf{x}, y) \right]^m \left[d_{k,i}(\mathbf{x}, y) \right]^2. \tag{4}$$

When one of the clusters is designated as the noise cluster, Eqs. (2)–(4) apply for $k = 1, 2, \ldots, c-1$ and for $k = c$, we replace Eq. (3) with

$$d_{k,i}(\mathbf{x}, y) = \delta^2, \tag{5}$$

where δ is the distance to the noise cluster called *noise distance* and Eq. (4) is relabeled as $f_N(\mu, \mathbf{v})$ [26]. Cluster centers that make $f_N(\mu, \mathbf{v})$ globally minimum are found by

$$\mathbf{v}_k(\mathbf{x}, y) = \left(\sum_{i=1}^{n} \left[\mu_{k,i}(\mathbf{x}, y) \right]^m \mathbf{x}_i \right) \left(\sum_{i=1}^{n} \left[\mu_{k,i}(\mathbf{x}, y) \right]^m \right)^{-1}, \quad k = 1, 2, \ldots, c-1 \tag{6}$$

and the distance between observations and $\mathbf{v}_c(\mathbf{x}, y)$ is equal to δ. If δ is smaller than a suitable value, non-outlier observations are erroneously assigned into the noise cluster. However, with a δ greater than a suitable value, noise cluster misses the outliers; hence, they are assigned to regular clusters. For the specification of δ, Davé [26] provides an approach based on the Euclidean

Table 1
Simulation space of the MC simulation study.

Sample size	200, 1000
Contamination %	0, 5, 10, 15, 20
Contamination scenario	Balanced, Slightly unbalanced, Moderately unbalanced, Highly unbalanced
Outlier multiplier	1.5, 3, 6

distance between objects and prototypes from FKM algorithm. The function **FKM.noise()** from the R package **fclust** [60] is used to implement Davé's [26] noise cluster approach at the first step of Fig. 2.

Both FRFN.ANN and FRFN.SVM approaches are implemented by MATLAB scripts (prepared by the authors) that communicate with R to call **FKM.noise()** function at the step S1 of Algorithm C1. MATLAB® 2019b Statistics and Machine Learning Toolbox is employed to implement SVM and ANN estimation methods under the FRFN setting where we consider the same training and kernel algorithms for ANN and SVM as mentioned in Sections 2.1 and 2.2, respectively. The details of software implementation are given in Figure B1 of Appendix B.

3. MC simulation study

3.1. Simulation space and statistical comparison of results

To demonstrate the accuracy of the considered and proposed methods against outliers, we need to introduce outliers into the dataset with different characteristics. For each combination of these characteristics, we get a simulation scenario and all the scenarios constitute the simulation space. Since the distribution of outliers across the folds of k-fold cross validation is done randomly in practice and simulations need to be replicated for generalizability of the results, we follow a MC simulation approach in this study. Simulation space of our MC study is composed of sample size, contamination rate, contamination scenario, and the magnitude of outliers. The levels of these factors of the simulation space are given in Table 1. Sample size represents the number of records (training + testing) created for a simulation scenario to obtain reasonably sized samples after splitting. We consider a relatively small and a large sample size. The contamination rate ranges between 5% and 20% to represent low, moderate, and high degree of outlier contamination that can occur in practice. While 5% and 10% contamination represent low levels of outlier occurrence in the data, 15% and 20% contamination depict moderate to high contamination. We also consider the performance of the methods with uncontaminated datasets with 0% contamination rate. An outlier is typically an observation falling more than 1.5 times the interquartile range (*IQR*, the difference between 75th and 25th percentiles) above the third quartile or below the first quartile. Outlier multiplier is a multiplication factor used to push outliers beyond 1.5, 3, or 6 times *IQR* to create three different magnitudes such as slightly, moderately, and highly influential outliers.

The k-fold cross-validation is a frequently used technique in practice. Since the dataset is shuffled randomly and split into folds, the outliers in the observed dataset will be distributed across the folds in balanced and unbalanced patterns depending on the randomness. To represent this case in our numerical study, we consider the distribution of outliers across the 5-folds under different contamination scenarios. Contamination rates of the folds under each contamination scenario are presented in Table 2.

When the outliers are equally or slightly unequally distributed across the folds, at every iteration of cross-validation, both test and training sets will have outliers. For moderately unbalanced distribution of outliers across the folds, the possibility of having outliers in the training set and no or a small number of outliers in the test set will be high. Consequently, the model will be trained under the impact of outliers, but testing will be done free of outlier impact or vice versa. For the highly unbalanced distribution of the outliers, when the fold containing highest proportion of outliers is taken as the test set, it will challenge the testing performance of the models in an unfair way. However, when this fold is included in the training sample, the impact will be similar to a slightly unbalanced scenario. Thus, the impact of outliers is highest in moderately unbalanced distribution of outliers across the folds.

Table 2
Distribution of outliers across the folds in 5-fold cross-validation for each contamination rate.

Contamination rate	Fold	Balanced	Slightly unbalanced	Moderately unbalanced	Highly unbalanced
5%	1	1%	2%	2%	2%
	2	1%	0%	0%	3%
	3	1%	1%	0%	0%
	4	1%	1%	2%	0%
	5	1%	1%	1%	0%
10%	1	2%	3%	3%	2%
	2	2%	0%	3%	6%
	3	2%	3%	0%	1%
	4	2%	2%	2%	1%
	5	2%	2%	2%	0%
15%	1	3%	4%	3%	3%
	2	3%	2%	3%	9%
	3	3%	4%	2%	1%
	4	3%	2%	5%	1%
	5	3%	3%	2%	1%
20%	1	4%	5%	7%	6%
	2	4%	3%	1%	12%
	3	4%	4%	1%	1%
	4	4%	4%	7%	1%
	5	4%	4%	4%	0%

The simulation space is composed of 120 combinations (simulation scenarios) of the components. Each simulation scenario is considered under each of ANN, SVM, FRFN.ANN, and FRFN.SVM methods, related membership functions, and training algorithms and kernel and loss functions. We used the default values of kernel parameters in MATLAB such that the order of the polynomial kernel is taken as 3 and the scale of Gaussian kernel is taken as 1 for SVMs. Outliers can impact SVMs through the used kernels. Optimization algorithms used to optimize kernel parameters are also negatively impacted by the outliers. One of the aims of our study is to compare the robustness of different kernels against outliers. If we use optimization, we are not able to separate the impact of outliers on kernel function and optimization process. Therefore, we fix the kernel parameters at their default values in MATLAB to be able to assess the superiority of kernels over each other instead of doing optimization on them. In total, 36 different implementations of ANN, SVM, FRFN.ANN, and FRFN.SVM with 12 training algorithms, 4 kernel functions, and 2 loss functions are considered in the simulations. The overall number of simulation runs is 4995. The computer on which we ran the simulations has an Intel Core i5-8250 processor (2.5–3.1 GHz) and an 8 GB, 1600 MHz DDR3L memory. All the parameter values used for ANNs and SVMs are tabulated in Table D1–D4 of Appendix D.

To create the independent features for the regression problem, four independent features were generated for the simulations under different distributional assumptions to cater for a variety of scenarios in practice. Gamma ($\alpha = 1, \beta = 1$), Truncated-normal ($\mu = 25, \sigma = 15, a = -10, b = 40$), and Uniform ($a = 0, b = 300$) distributions were arbitrarily used to generate real-valued features X_1, X_2, and X_3, and Uniform ($a = 0, b = 100$) distribution was used to generate X_4 feature represented by percentages. A

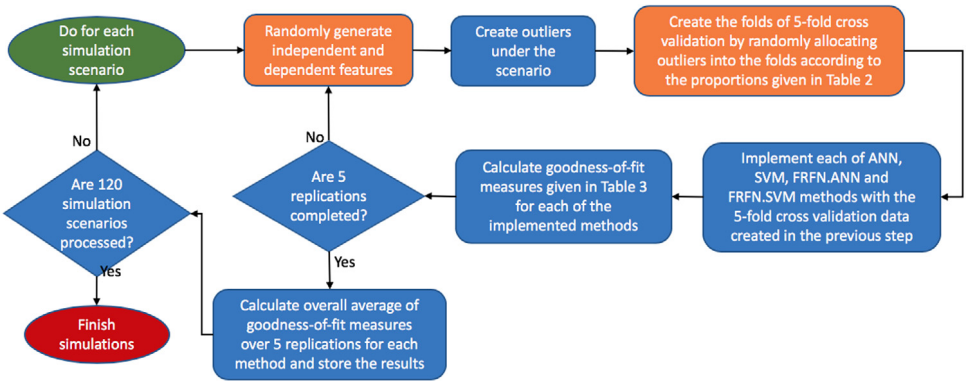

Fig. 4. Flow chart of simulation study.

Table 3
Goodness-of-fit measures and their formulations.

GOF measure	Formula	
Mean Absolute Error (MAE) [39]	$MAE = \frac{1}{n}\sum_{i=1}^{n}\lvert Y_i - \hat{Y}_i\rvert$	(7)
Percent Bias (PBIAS) [39]	$PBIAS = 100\frac{\sum_{i=1}^{n}(\hat{Y}_i - Y_i)}{\sum_{i=1}^{n} Y_i}$	(8)
Mean Squared Error (MSE) [39]	$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2$	(9)

$\overline{Y}$: The mean of observations. σ_Y: The standard deviation of observations. n: Sample size.

linear relationship between the dependent feature Y and the matrix of independent features $\boldsymbol{X} = (X_{ij})$ is set by $\boldsymbol{Y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$ model where $\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2, \beta_3, \beta_4]^T = [140, 12, 2, 0.25, 0.75]^T$ and $\varepsilon_i \sim N(0, 1), i = 1, \ldots, n; j = 0, \ldots, 4$. In this setting, the elements of $\boldsymbol{\beta}$ are arbitrarily chosen. Our main aim is to monitor the prediction accuracy of the methods via the k-fold cross validation. Therefore, random selection of model parameters $\boldsymbol{\beta}$ and arbitrary selection of the distributional parameters do not impose any limitations on the generalizability of our results.

We artificially replaced 0%, 5%, 10%, 15%, and 20% of the generated data values with outliers of different magnitudes defined by each simulation scenario. The predicted values of the dependent target feature are compared to the corresponding values from the initial simulated data to obtain the goodness-of-fit (GOF) measures. We use 3 GOF measures between predicted and simulated values over the test split. Let $\hat{\boldsymbol{Y}}$ be the vector of predicted values. Then, the formulations of GOF measures used in this study are given Eqs. (7) to (9) in Table 3.

The flow chart of MC simulation study that involves 5 replications of 120 scenarios is given by Fig. 4. Also, Figure B1 of Appendix B shows the details of the software implementation of simulations. In Fig. 4, the steps colored orange show those include random generations.

To apply a statistical test to figure out the significance of the difference between GOF measures attached to each method, Demšar [61] proposes to implement a well-known non-parametric omnibus test, namely the Friedman test. If a significant difference between the performances of the considered ML methods is observed by the Friedman test [62], we proceed to the post-hoc Nemenyi test to detect the individual differences between the methods. The null hypothesis of the Friedman test states that all the ML methods have the same performance in terms of the considered GOF measure. The test statistic for the Friedman test is given in Eq. (10)

$$\chi_F^2 = \frac{12N}{k(k+1)}\sum_{j=1}^{k}\left(R_j - \frac{(k+1)}{2}\right)^2, \tag{10}$$

where $R_j = \frac{1}{N}\sum_{i=1}^{N} r_j^i$, r_j^i is the rank of the jth method on the ith dataset, $i = 1, \ldots, N$, and $j = 1, \ldots, k$ [62]. χ_F^2 statistic is distributed as χ_{k-1}^2 under the null hypothesis for large N and k. Demšar [61] suggests $N > 10$ and $k > 5$. For smaller N and k, critical values are available by Sheskin [63]. Iman and Davenport [64] suggest use of the F-statistic in Eq. (11) to avoid the conservativeness of the χ_F^2 statistic.

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} \tag{11}$$

which is distributed as F-distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedoms. After rejecting the null hypothesis of the Friedman test, Nemenyi test [65] is applied using the critical difference

$$CD = q_\alpha\left(\frac{k(k+1)}{6N}\right)^{0.5}, \tag{12}$$

where q_α is tabulated by Demšar [61] [See Table 5(a)] for $\alpha = 0.05$ and 0.10. We conclude the significance of the difference between the GOF performance of two methods if the corresponding average ranks between the methods differ by at least the value of CD.

3.2. Results

We present the impact of outliers on the ANN, SVM, FRFN.ANN and FRFN.SVM methods in the following subsections. In all the plots displayed in this section, we averaged the values of GOF measures over the scenarios that are not shown in the plots. For example, if the test outlier contamination is shown in the plot, the plotted values of a GOF measure are averaged over the rest of scenarios and the values of Mean GOF measure are displayed for each method. We use log-scale to make the differences more visible in the plots where appropriate.

3.2.1. Impact of contamination rate

The impact of outliers on the considered methods over the test split in terms of logarithm of mean MSE is shown in Fig. 5.

When there is no outlier contamination in the dataset, FRFN.ANN and ANN trained with BR algorithm have the highest performance in terms of mean MSE. FRFN.ANN trained with BR is less sensitive to the high contamination rate, 20%, than its ANN counterpart. In addition to these methods, FRFN.SVM and SVM with the Gaussian kernel are consistently robust against increasing outlier contamination in terms of mean MSE. The proposed FRFN implementation of SVM with all the considered kernels are more robust against outliers than their pure SVM implementations.

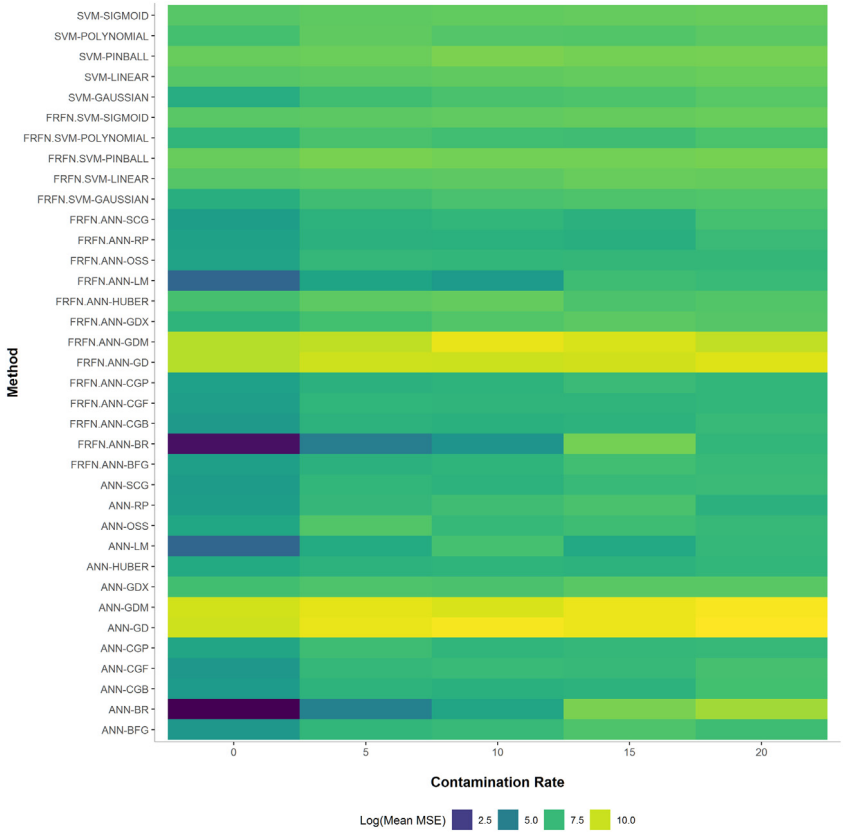

Fig. 5. Impact of outlier contamination in terms of logarithm of MSE in the breakdown of contamination rates.

The impact of outliers on the considered methods is plotted in terms of MAE and PBIAS in Fig. 6. The methods using LM and BR training algorithms perform best in terms of mean MAE with ANN and FRFN.ANN implementations when there is no outlier contamination. When we increase the rate of outlier contamination, ANN with Huber loss function and FRFN.SVM and SVM with the Gaussian kernel show a stable robustness against the increasing contamination rate of outliers. FRFN.ANN with BR or LM training has the highest mean MAE accuracy for all contamination rates. FRFN.ANN implementations are more robust against outliers than their regular ANN versions in terms of MAE. They are less sensitive to the increasing contamination rate among the considered methods. ANN trained with GD and GDM are the methods with the lowest MAE accuracy both in the presence and the absence of outliers.

In terms of mean PBIAS, we get different results than those obtained with mean MSE and mean MAE. The methods do not respond to the increasing contamination rate in a gradually increasing way in terms of PBIAS. But, in general, the higher the contamination rate the higher the mean PBIAS. When there is no contamination, ANN and FRFN.ANN with BR training has the lowest PBIAS. With low degree of contamination, FRFN.ANN-BR and FRFN.ANN-SCG perform best. For moderate contamination rates, ANN trained with BR has better performance. For high contamination rates, SVM.GAUSSIAN, its FRFN implementation, and FRFN.ANN-BR are less prone to the existence of outliers in terms of PBIAS accuracy. Almost all FRFN.ANNs have a better PBIAS performance than their ANN correspondents with 20% contamination rate.

All the considered GOF measures lead to the conclusion that ANNs trained with Gradient Descent or Gradient Descent with Momentum does not perform as good as the rest of the methods. The main reason of this is that they get overtrained due to the existence of outliers which follows our conceptual assessment in Section 2.1. The proposed FRFN implementation improves their performance by the noise cluster approach but they are still under the impact of outliers and far behind compared to the other methods. For SVM, linear and sigmoid kernels do not perform well in terms of all the GOF measures. The Gaussian and polynomial kernels perform well against outliers with FRFN.SVM and SVM. The functional structure of these kernels helps to mitigate the impact of outliers. For ANN and FRFN.ANN, BR training performs better than the other training algorithms. The robustness of BR algorithm against overfitting helps it to be robust against the outliers as well. While the performance of FRFN.ANN implementation is comparable to the pure ANN when there is no outlier, the proposed FRFN implementation becomes more robust against the outliers with the increasing contamination rates due to assigning outliers to the noise cluster.

3.2.2. Impact of contamination scenario

The random distribution of outliers across the folds of the k-fold cross-validation will most probably be unbalanced for different magnitudes in practice. Figs. 7 and 8 show mean MSE, mean MAE, and mean PBIAS performances of methods in the log scale under different cases of possible formations of the folds.

When the outliers are equally distributed to the folds, all the methods perform better than the other contamination scenarios in terms of the considered GOF measures. This shows the usefulness of the k-fold cross-validation in mitigating the impact of outliers. Considering the observations are randomly assigned to the folds in practice, having a fully balanced distribution of the outliers over the folds is unlikely. Thus, the performance of almost all methods are impacted by the unbalanced distributions or existence of outliers. Since we cannot control how unbalanced the folds will be in practice, we need to identify the methods that

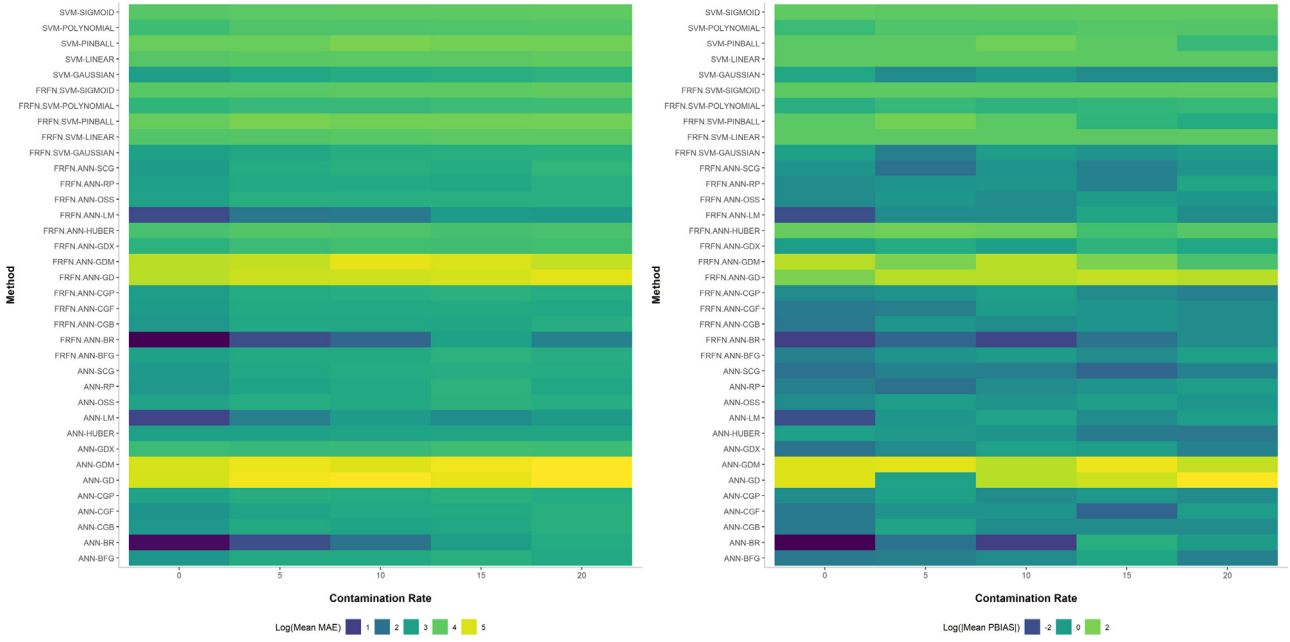

Fig. 6. Impact of outlier contamination in terms of MAE and PBIAS in the breakdown of contamination rates.

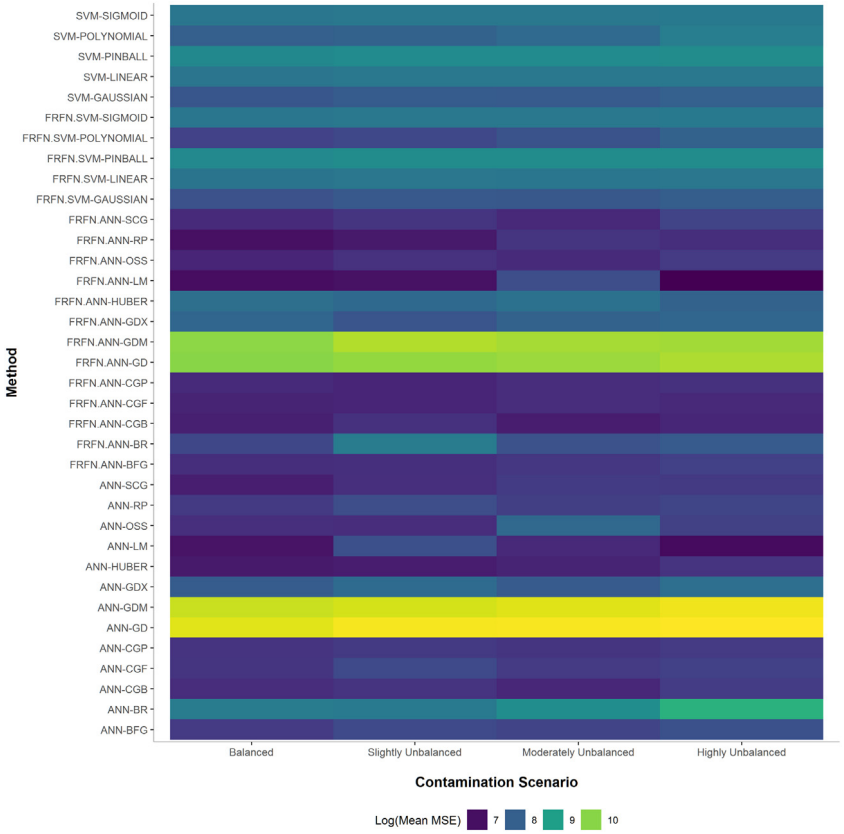

Fig. 7. Impact of outlier contamination in terms of logarithm of mean MSE in the breakdown of contamination scenarios.

perform well under the unbalanced distribution of outliers in the folds.

As expected, the impact of outliers is highest in moderately unbalanced distribution of outliers across the folds. The FRFN implementation improves the robustness of SVM with polynomial and Gaussian kernels and ANN trained with RP, OSS, LM, GDM, GD, CGP, CGF, and BR in terms of mean MSE in nearly all contamination scenarios. When the distribution of outliers across the folds is slightly unbalanced FRFN.ANNs trained with LM and RP are the most accurate methods. When it is moderately unbalanced, FRFN.ANN-CGB is the most accurate method. However, when it is highly unbalanced, the best performing method is FRFN.ANN trained with LM in terms of mean MSE. From Fig. 8, FRFN.ANN and ANN trained with BR and LM give the

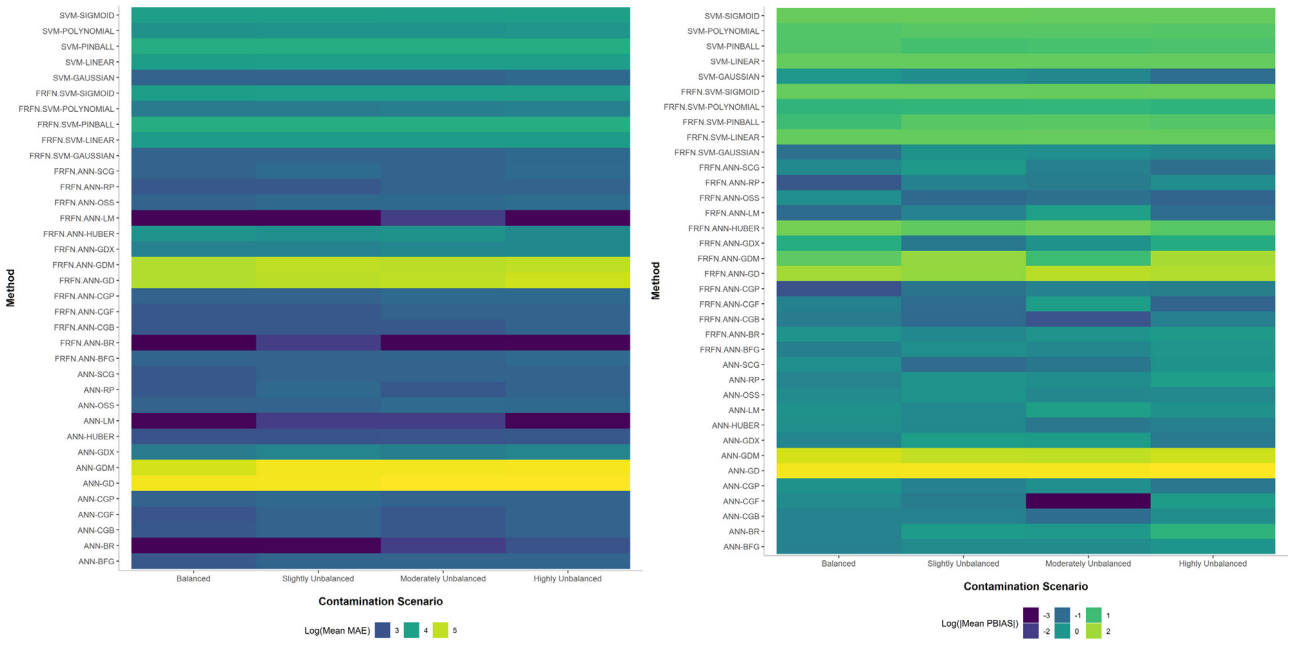

Fig. 8. Impact of outlier contamination in terms of MAE and PBIAS in the breakdown of contamination scenarios.

most promising performance in terms of mean MAE. FRFN.ANN trained with BR has the best mean MAE performance for balanced and moderately unbalanced contamination scenarios. For the rest of the scenarios, FRFN.ANN-LM has the best mean MAE performance. In terms of mean PBIAS, FRFN.ANN trained with CGB and RP have the best performance for balanced distribution of outliers. For the slightly unbalanced case, FRFN.ANN trained with OSS has the best PBIAS performance. FRFN.ANN trained with CGB is the most robust method when the distribution is moderately unbalanced. When it is highly unbalanced, FRFN.ANN trained with OSS and CGF give the most promising mean PBIAS results. Among the SVMs, SVM with Gaussian kernel perform best against the outliers among the contamination scenarios.

In the overall sense, the proposed FRFN implementation improves the robustness of both ANN and SVM methods against outliers. FRFN.ANN trained with LM and BR are the most robust methods against the outliers in terms of mean MSE and mean MAE among the considered methods. In terms of mean PBIAS, ANNs trained with conjugate gradient methods are found more robust than the other methods. Similar with the contamination rates, ANNs trained with gradient descent and its variations are not as robust as the other considered methods when the outliers are distributed across the folds randomly.

3.2.3. Impact of outlier magnitude and sample size

The impact of the magnitude of outliers and the sample size on the ML methods are displayed in Figs. 9 and 10 in terms of logarithms of mean MSE, mean MAE, and mean PBIAS.

Both mean MSE and MAE increase with increasing outlier magnitude. However, mean PBIAS is at its lowest level for moderate sized outliers for some of the methods. Since PBIAS shows the relative error against the original value of an observation, mean PBIAS does not reflect a monotone change along with the magnitude of the outliers. For low magnitude outliers, FRF.ANN-LM and ANN-LM give the best MAE results. In terms of MSE, FRF.ANN-BR and ANN-BR also perform well. When the outlier magnitude is increased to 3, FRF.ANN-BR is the only method that is robust to the increase in the magnitude of outliers in terms of both mean MSE and MAE. FRF.ANN-LM and FRF.ANN-BR have the best mean MSE and MAE performances for the large magnitude outliers. In terms of the mean relative magnitude of errors, mean PIBAS shows that FRF.ANN-BR and FRFN.SVM-GAUSSIAN are the best methods for moderate magnitude outliers. For low magnitude outliers, ANN-CGP gives the highest PBIAS accuracy. As for the large outliers, FRF.ANN-SCG gives the lowest mean relative error.

For the smaller sample size, ANN with Huber loss function and LM and BR training functions give the lowest MSE, FRF.ANN-BR and ANN.BR produce the lowest MAE, and ANN-BR has the lowest PBIAS. When the sample size is increased to 1000, FRFN implementations perform better in terms of all GOF measures. FRFN.ANN-LM is the best performing method with the sample size of 1000. Also, FRFN.ANN-BR gives promising results in terms of mean MAE, FRF.ANN-RP and FRF.ANN-CGF provide good results in terms of mean MSE for large samples.

3.2.4. Overall comparison of the methods

To compare the methods in an overall sense, we tabulate the method that gives the smallest value to each GOF measure for each simulation scenario that has a contamination rate greater than 0% in Tables A1–A4 of Appendix A. Then, we count the number of times each method beats others for all the considered simulation scenarios. The results are summarized in Table 4 for each GOF measure.

For all GOF measures, FRFN.ANN trained with BR has the highest number of wins, its pure ANN implementation has the second highest number of wins and LM training comes the third most accurate method. SVM and FRFN.SVM methods do not provide the smallest GOF values as frequently as ANN and FRFN.ANN implementations over the set of simulation scenarios in terms of both mean MAE and mean MSE. From Table A1, when the contamination rate is 5%, ANN-BR gives more accurate results than FRFN.ANN-BR for high magnitude outliers whereas FRFN.ANN-BR is the most accurate method for low to moderate magnitude outliers in terms of mean MAE. We have a similar pattern when we consider mean MSE. But FRFN.ANN-BR also gives the most accurate results in some cases with high magnitude outliers in addition to ANN-BR and ANN-LM. When the contamination rate increases to 10% (Table A2), FRFN.ANN-BR gives the best results in most of the considered simulation scenarios with high

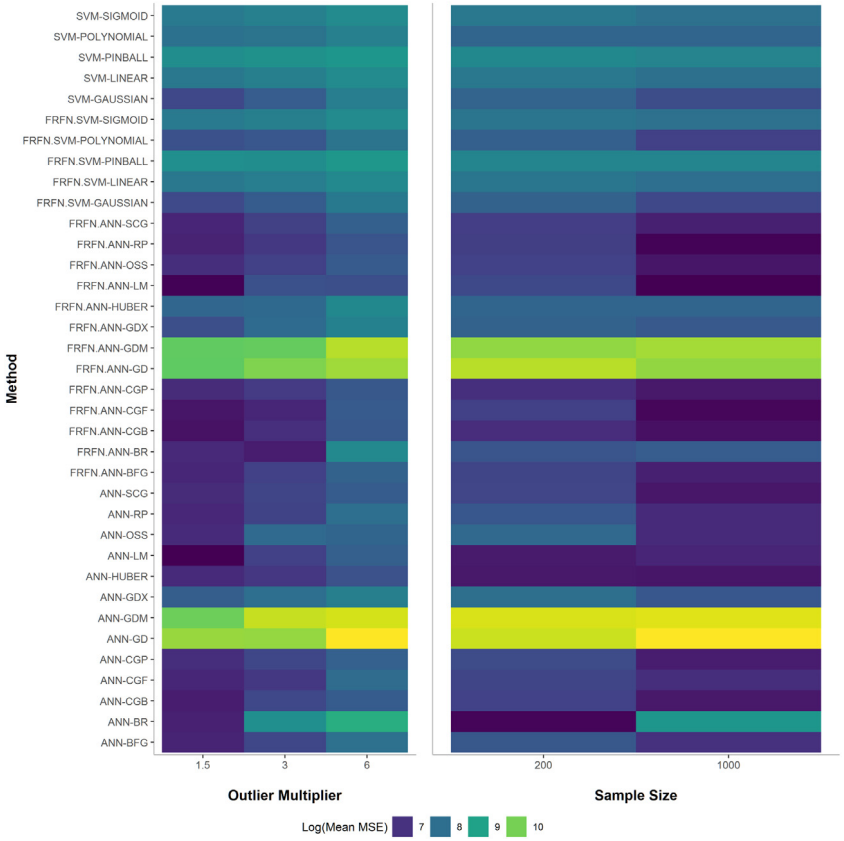

Fig. 9. Impact of outlier magnitude and sample in terms of logarithm of mean MSE.

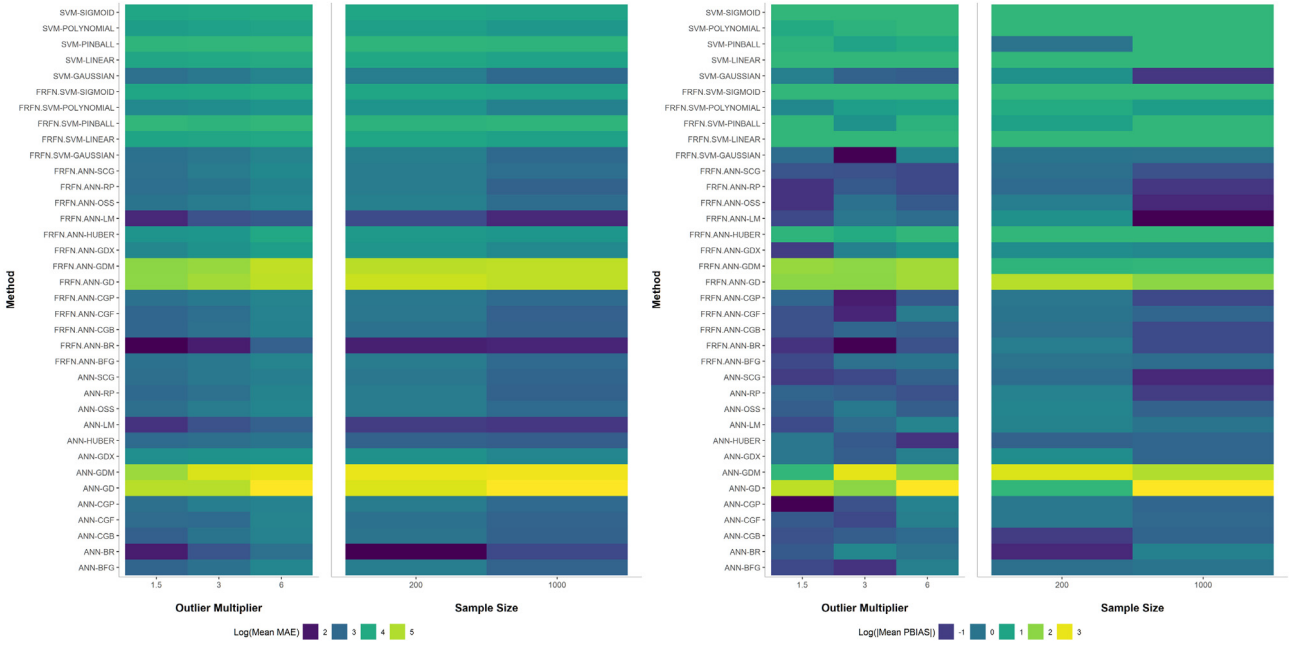

Fig. 10. Impact of outlier magnitude and sample size in terms of logarithm of mean MAE and PBIAS.

magnitude outliers in terms of both mean MAE and mean MSE. FRFN implementations with different training algorithms give the most accurate PBIAS results in 75% of the scenarios. When the contamination rate is 15% (Table A3), FRFN.ANN implementations provide the minimum mean MAE and MSE mostly for low magnitude outliers. When the magnitude of outliers increases, pure ANN implementations become more robust in a greater number of scenarios than those of FRFN.ANNs. In terms of mean PBIAS, FRFN.ANNs trained with BR, LM, CGB, CGP, and SCG and FRFN.SVM with Polynomial kernel have the best performance with low magnitude outliers. When the contamination rate is 20% (Table A4), this pattern moves towards the moderate magnitude outliers. For almost all the cases with moderate or high magnitude outliers, FRFN.ANN-BR gives the minimum MAE and ANN-BR

Table 4
The number of times each method achieved the best GOF measure over all simulation scenarios where the contamination rate is greater than 0%.

MAE	
Method	Wins
FRFN.ANN-BR	58
ANN-BR	29
ANN-LM	8
FRFN.ANN-LM	1
Total	96

MSE	
Method	Wins
FRFN.ANN-BR	45
ANN-BR	22
ANN-LM	14
FRFN.ANN-LM	8
ANN-CGB	2
ANN-CGP	2
FRFN.ANN-CGB	2
FRFN.ANN-RP	1
Total	96

PBIAS	
Method	Wins
FRFN.ANN-BR	36
ANN-BR	14
FRFN.ANN-LM	13
ANN-LM	4
ANN-CGF	3
FRFN.ANN-OSS	3
FRFN.ANN-CGB	3
FRFN.ANN-BFG	3
SVM-GAUSSIAN	2
ANN-OSS	2
ANN-GDX	2
ANN-CGB	2
FRFN.SVM.POLYNOMIAL	1
FRFN.ANN-SCG	1
ANN-SCG	1
FRFN.ANN-CGP	1
FRFN.SVM.GAUSSIAN	1
ANN-RP	1
ANN-CGP	1
ANN-BFG	1
FRFN.ANN-RP	1
Total	96

Table 5
Average ranks for the top four methods in the breakdown of GOF measures.

GOF measure	ANN-BR	ANN-LM	FRFN.ANN-BR	FRFN.ANN-LM
MAE	2.17	3.04	1.67	3.13
MSE	2.54	2.88	1.86	2.72
PBIAS	2.43	2.88	1.89	2.66

is the best performing method for low magnitude outliers. We observe that FRFN.ANN-BR gives the minimum MSE with a larger sample size and it mostly has the best MSE performance for the moderate or low magnitude outliers. For PBIAS, ANNs with Huber loss function gives the best results for slightly and moderately balanced cases with moderate to high magnitude outliers. FRFN.ANNs trained with BR, LM, OSS, RP, BFG, CGB algorithms give the best PBIAS performance across all outlier magnitudes and for all highly unbalanced cases.

Lastly, we implement the Friedman test in an omnibus way to statistically test the null hypothesis that there is no significant difference between the GOF performances of 36 methods. The average ranks of methods for each GOF measure are tabulated in Table A5 of Appendix A. The values of the Friedman test statistic calculated by using Eq. (11) are 577.04, 333.16, and 222.58 for MAE, MSE, and PBIAS, respectively. Comparing these values with the F-table value of 1.39 at 5% level of significance, we strongly conclude that there is a significant difference between MAE, MSE, and PBIAS performances of the considered methods. Table A5 shows that the top four methods have a considerably higher average rank performance than the other methods which is consistent with the inferences made from the MC simulation results. Therefore, we compare GOF performances of ANN-BR, ANN-LM, FRFN.ANN-BR, and FRFN.ANN-LM methods to focus on the difference between top performing methods. Average ranks of ANN-BR, ANN-LM, FRFN.ANN-BR, and FRFN.ANN-LM methods over 96 scenarios are given in Table 5, where we consider only four methods.

Based on the average ranks given in Table 5, the Friedman tests statistics with $k = 4$ and $N = 96$ for MAE, MSE, and PBIAS are 40.31, 12.81, and 11.71, respectively. Since the corresponding F-table value is 2.60 at 5% level of significance, we conclude the existence of a significant difference between the top four methods in terms of all the considered GOF measures. Then, to proceed with the post-hoc Nemenyi test, we read the q_α of Eq. (12) from Table 5(a) of Demšar [61] as 2.569 for $k = 4$ and $\alpha = 0.05$, and calculate $CD = 0.479$. From Table 5, the top two methods are ANN-BR and FRFN.ANN-BR for all GOF measures. The absolute differences between the average ranks of ANN-BR and FRFN.ANN-BR are 0.500, 0.677, and 0.547 for MAE, MSE, and PBIAS, respectively. Since all the absolute differences are greater than 0.479, we conclude that the overall MAE, MSE, and PBIAS performances of FRFN.ANN-BR are significantly greater than those of ANN-BR at 5% level of significance.

4. Numerical study

To compare the performance of the proposed FRFN.ANN and FRFN.SVM implementations to their pure ANN and SVM counterparts with real data, we refer to the benchmark database of UCI ML Repository [35], Rdatasets repository [36], and Kaggle data compiled by Scott [37]. We work on 15 datasets detailed in Table 6. Our benchmark datasets include small, moderate, and large samples in terms of the ratio of the number of independent variables to the sample size. All the dependent features in the datasets are measured in the interval scale.

Table 6
Details of benchmark datasets used in the numerical study.

Name	Number of independent features	Sample size	Attribute type	Web link
Airfoil self-noise	5	1503	Real	[a]/Airfoil+Self-Noise
Alcohol	2	45	Real	[b]/robustbase/alcohol.csv
Aricraft	4	23	Real	[b]/robustbase/aircraft.csv
Auto MPG	8	398	Cat., Real	[a]/auto+mpg
Bfox	2	30	Real	[b]/carData/Bfox.csv
Bigcity	1	49	Real	[b]/boot/bigcity.csv
Concrete slump test	9	103	Real	[a]/Concrete+Slump+Test
Energy efficiency	7	768	Int., Real	[a]/energy+efficiency
Fish	2	159	Real	[c]/aungpyaeap/fish-market
Gas sensor	4	5636	Real	[a]/Gas+sensor+array+temperature+modulation
Insurance	4	1338	Cat., Real	[c]/mirichoi0218/insurance
Motor	3	94	Int., Real	[b]/boot/motor.csv
Porsche	2	30	Real	[b]/Stat2Data/PorschePrice.csv
Toxicity	9	38	Real	[b]/robustbase/toxicity.csv
Wine Quality	11	4898	Real	[a]/Wine+Quality

[a] https://archive.ics.uci.edu/ml/datasets.
[b] https://raw.github.com/vincentarelbundock/Rdatasets/master/csv.
[c] https://www.kaggle.com.

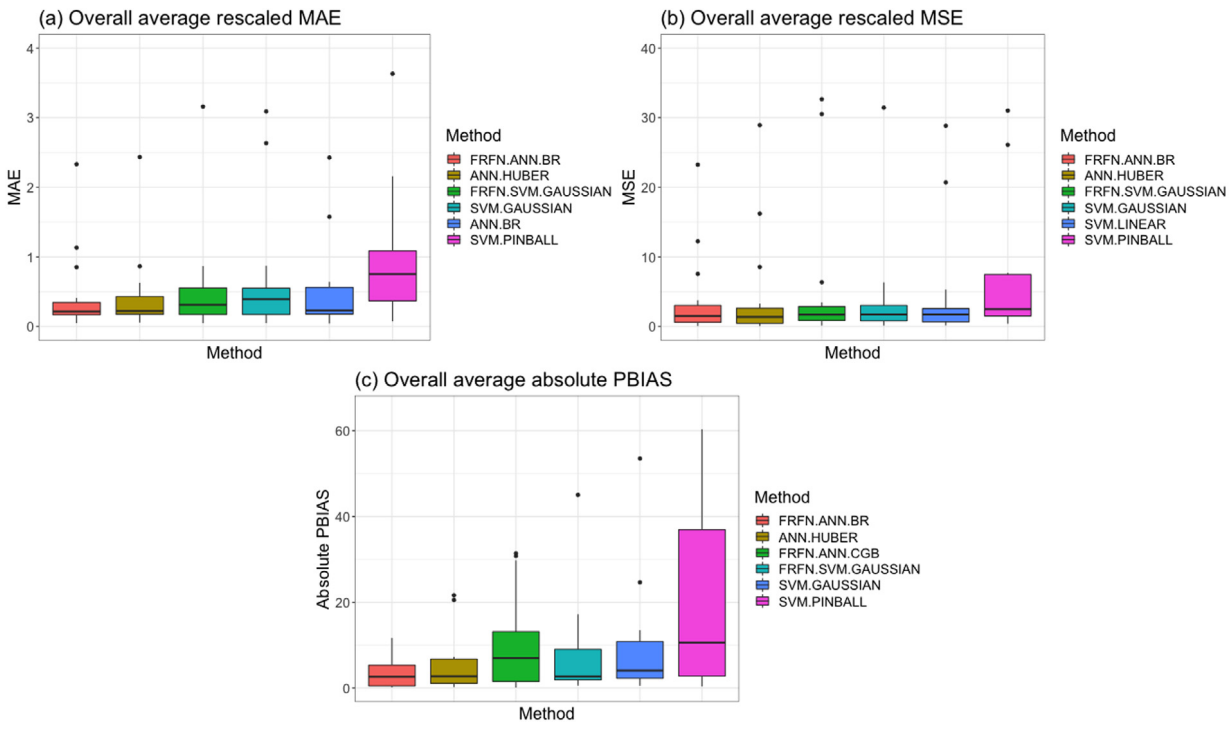

Fig. 11. Overall average of rescaled MAE, MSE and absolute PBIAS over 5%, 10%, 15%, and 20% contamination rates.

We randomly introduce outliers in the benchmark datasets to set the rate of outliers to 5%, 10%, 15%, or 20% of the observations according to $3 \cdot IQR$ criterion. Then, we implement 5-fold cross validation by randomly choosing the folds. We rescale MAE and MSE by dividing the formulas given in Table 3 by the average of dependent variable ($\overline{y}$) of each dataset due to the difference of scales among the datasets in the numerical study. The top five performing methods according to overall averages of rescaled MAE, rescaled MSE and absolute PBIAS and SVM.PNINBALL are presented in Fig. 11. Since FRFN.ANN-BR has the smallest average MAE and variation, it performs best followed by ANN.HUBER and the FRF implementation of SVM with Gaussian kernel. FRFN.ANN-BR is the second best in terms of overall average MSE and performs best in terms of overall average absolute PBIAS.

In terms of each GOF measure, overall average of ranks of methods over the considered contamination rates and the datasets are reported in Table 7. For MAE, FRFN.ANN-BR has the smallest overall average rank. In terms of MSE ranks, ANN.HUBER outperforms FRFN.SVM-GAUSSIAN and FRFN.ANN-BR. This is consistent with the observation of Ma et al. [66], who note that the MSE performs well under Gaussian noise but it is sensitive to large outliers in training the ANN. In terms of absolute PBIAS, FRFN.ANN-BR and ANN.HUBER have very close over average ranks.

We use the average ranks in Table 7 to implement Friedman test in the omnibus sense among the top five methods and the post-hoc Nemenyi test for the top pairs as described in Section 3.1. From Table 7, the Friedman tests statistics, χ_F^2, with $k = 5$ and $N = 60$ for rescaled MAE, rescaled MSE, and absolute PBIAS are 342 947, 332 592, and 328 384, respectively. Since these values are too large compared to the χ^2 table value of 9.48 at 4 degrees of freedom and 5% level of significance, we straightforwardly conclude that there is a significant difference between the MAE, MSE, and PBIAS performance of the top five methods. For $k = 5$ and $\alpha = 0.05$, we read $q_\alpha = 2.728$ from Table 5(a) of Demšar [61]; hence, the critical difference for the Nemenyi test is obtained as $CD = 0.787$. For MAE, since the difference between the average ranks of FRFN.ANN-BR and ANN.HUBER is 1.34 and it is greater than 0.676, we conclude that the MAE performance of FRFN.ANN-BR is significantly better than that of ANN.HUBER. However, the MSE performance of ANN.HUBER is significantly better than that of FRFN.SVM-GAUSSIAN ($2.30 > 0.787$). For PBIAS performance, there is no significant difference between FRFN.ANN-BR and ANN.HUBER ($0.13 < 0.787$) and this pair is significantly better than FRFN.ANN-CGB ($2.73 > 0.787$).

In the overall sense, we find the performance of the proposed FRFN implementations of ANN and SVM methods better than their straightforward versions for the real datasets under the presence of outlier contamination. Consistent with the simulation study, FRFN.ANN-BR and FRFN.SVM-GAUSSIAN methods give more accurate results than the other considered implementations of ANN and SVM with real datasets. Although ANN.HUBER gives better MSE ranking across the benchmark datasets, its MSE values are more variable than those of FRFN.ANN-BR (Fig. 11-b)

For uncontaminated version of the real datasets, the overall averages of rescaled MAE, rescaled MSE, and absolute PBIAS over the benchmark datasets are given for top 10 performing methods in Table 8.

Consistent with the simulation results, we find that FRFN.ANN-BR and ANN.HUBER outperform other methods in terms of all rescaled MAE, rescaled MSE and absolute PBIAS. FRFN.SVM with Gaussian kernel is a promising method for the real datasets in terms of rescaled MSE. CGB training gives the third overall rescaled MAE, rescaled MSE, and absolute PBIAS performance with ANN. FRFN.ANNs with different versions of training algorithms based on partial derivatives and gradients such as CGB, CGF, and SCG are accurate methods in terms of overall average PBIAS.

To assesses the execution times of the methods, we displayed the box plots of elapsed time for each method over 15 benchmarking datasets in Fig. 12. The computer processed the methods with benchmarking datasets is the same as the one used for the simulations. The details of the computer are given in Section 3.1.

Table 7
Overall average of ranks and their standard deviations (St.Dev.) over all outlier contamination rates for top five performing methods according to rescaled MAE, rescaled MSE, and absolute PBIAS. Rank difference shows the difference between two succeeding average ranks for testing.

	Method	Average rank	St.Dev.	Rank difference
Rescaled MAE	FRFN.ANN-BR	5.33	4.62	–
	ANN-HUBER	6.67	6.93	1.34
	FRFN.SVM-GAUSSIAN	7.47	4.60	0.80
	SVM-GAUSSIAN	7.67	7.28	0.20
	ANN-BR	9.40	10.63	1.73
Rescaled MSE	ANN-HUBER	7.53	7.33	–
	FRFN.SVM-GAUSSIAN	9.83	6.11	2.30
	SVM-GAUSSIAN	10.33	7.74	0.50
	SVM-LINEAR	10.36	9.27	0.03
	FRFN.ANN-BR	10.40	10.04	0.03
Absolute PBIAS	ANN-HUBER	8.93	7.00	–
	FRFN.ANN-BR	9.07	8.31	0.13
	FRFN.ANN-CGB	11.80	8.59	2.73
	FRFN.SVM-GAUSSIAN	1193	8.97	0.13
	SVM-GAUSSIAN	12.33	8.99	0.4

Table 8
Top ten methods according to overall averages of rescaled MAE, rescaled MSE, and absolute PBIAS over the benchmark datasets when is no outliers in the dataset.

Method	Rescaled MAE	Method	Rescaled MSE	Method	Absolute PBIAS
FRFN.ANN-BR	0.213	FRFN.ANN-BR	0.095	FRFN.ANN-BR	1.752
ANN-HUBER	0.228	ANN-HUBER	0.102	ANN-HUBER	1.808
ANN-CGB	0.286	ANN-CGB	0.138	ANN-CGB	2.819
SVM-LINEAR	0.305	FRFN.SVM-GAUSSIAN	0.146	ANN-CGF	2.953
ANN-RP	0.305	SVM-LINEAR	0.147	FRFN.ANN-OSS	3.048
ANN-LM	0.313	ANN-RP	0.154	ANN-RP	3.251
ANN-CGF	0.313	FRFN.ANN-OSS	0.159	FRFN.ANN-CGB	4.703
FRFN.SVM-GAUSSIAN	0.336	FRFN.SVM-LINEAR	0.186	ANN-BFG	4.740
ANN-SCG	0.338	FRFN.ANN-CGP	0.190	FRFN.ANN-CGF	5.006
FRFN.ANN-OSS	0.338	ANN-CGF	0.191	FRFN.ANN-CGP	5.475

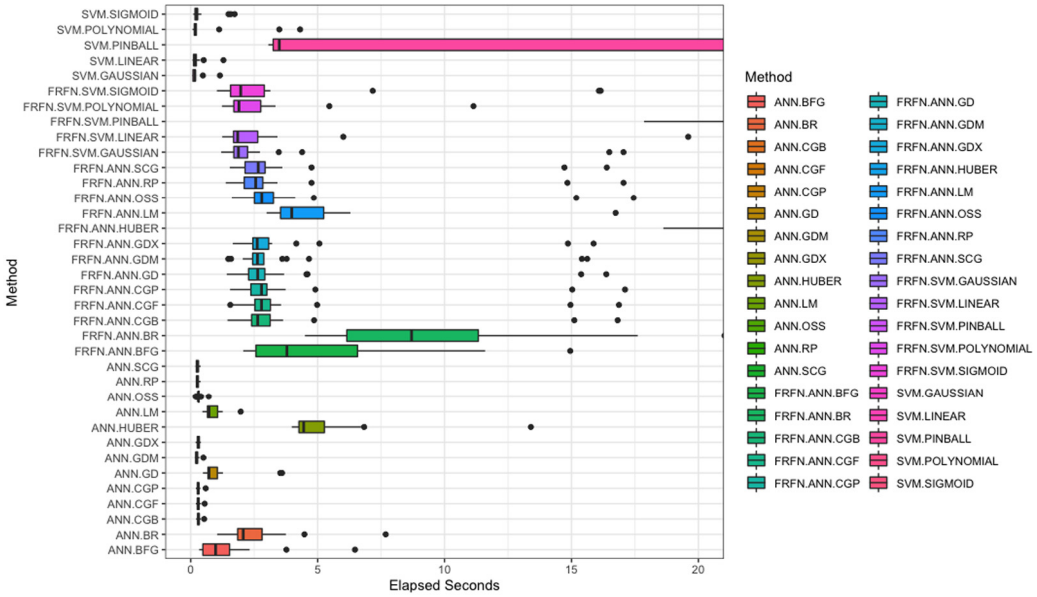

Fig. 12. Box plots of processing times of the methods for the benchmarking datasets.

Due to the additional layer of complexity in FRFN implementation, pure SVMs and ANNs require less computational resources to process. However, most of the FRFN implementations are completed under 5 s, which is worthwhile compared to the accuracy gain. Since SVM.Pinball and ANN.Huber implementations need R and MATLAB to communicate with each other, they require considerably longer time to finish than other ANN implementations.

5. Conclusions and discussion

Almost all the mainstream machine learning (ML) methods are prone to the impact of outliers in the dataset. An algorithm trained under the existence of outliers tends to give inaccurate predictions due to the noise created by the outliers. On the other hand, if the algorithm is tested with a contaminated test set while it is not trained under the existence of outliers, it can

erroneously be classified as unsuitable for predictions. The k-fold cross-validation has the potential to mitigate the impact of outliers through randomly shuffling the dataset. However, due to the randomness, possibility of distributing outliers evenly across the folds is low. Thus, the problem of training a model with a contaminated training set and testing it with an uncontaminated or slightly contaminated test set (or vice versa) remains. It is crucial to know the reaction and the degree of robustness of ML methods against the existence of outliers. In this sense, although the ML methods are used to investigate existence of outliers in the literature, their performance has not been thoroughly tested under the presence of outliers in the dataset.

In this article, we investigate robustness of mainstream ML methods, namely support vector machines (SVM) and artificial neural networks (ANN) for the regression problem. We modify the classical fuzzy regression functions method by using fuzzy k-means clustering with a noise cluster (FRFN) at its clustering step. We conduct an extensive Monte Carlo study and a numerical study with real benchmarking datasets to monitor the robustness of the SVMs, ANNs, and our proposed FRFs against various scenarios of sample size, rate and magnitude of outliers, and their distribution across the folds in cross-validation. With the Monte Carlo study, in addition to the performance of the methods with contaminated data, we provide a performance comparison of SVM, ANN and modified FRF methods with uncontaminated data in our study. We consider 12 training algorithms for ANN and 4 kernels for SVM and implement FRFN.SVM and FRFN.ANN methods under all the considered kernels and training algorithms. In total, we compare the performance of 36 methods over a simulation space of 120 scenarios by applying 5-fold cross-validation for each method in the Monte Carlo study and we consider 15 real datasets with differing characteristics in the numerical study.

The main findings of our study are summarized as follows:

- When there is no outlier contamination in the dataset, ANN and FRFN.ANN trained with Bayesian Regularization (BR) gives the most accurate results. FRFN implementations further improves the performance pure ANN implementation. FRFN.ANN trained with the Levenberg–Marquardt (LM) algorithm also provides promising results. Gaussian kernel is the best performing kernel for the SVM and FRFN-SVM. Although the accuracy of ANN trained with gradient decent algorithms is improved by FRFN implementation, their performance is the worst among the considered methods. In terms of percentage bias, ANNs trained with algorithms based on gradient and partial derivative have superior performance.
- Even when the contamination rate is low, such as 5%, the prediction accuracy of all methods is negatively impacted. The accuracy of the methods decreases along with increasing contamination rate. FRFN.ANN trained with BR performs best when the magnitude of outliers is low to moderate. This implies that the proposed FRF method is very sensitive against rare and low magnitude outliers. This is a desired feature for an ML method in the sense that FRFN.ANN-BR can safely be used even when the outliers in a dataset are not detected accurately. When the magnitude of rare outliers is high, in which case outliers can easily be detected, ANN trained with BR gives the best results. It is recommended to use ANN-BR when outliers are evident, and they contaminate the sample at a low rate. However, if there are outliers of a small magnitude creating a low rate of contamination in the sample, use of FRFN.ANN-BR is recommended.
- For a moderate contamination rate, such as 10%, FRFN.ANN-BR gives best results in most of the scenarios including all the considered outlier magnitudes. We suggest the use of FRFN.ANN-BR for this case.
- When the contamination rate is moderate to high, such as 15% and 20%, FRFN.ANN trained with BR algorithm performs best in most of the scenarios. In some scenarios, ANN implementations also provide us with promising accuracy based on the distribution of outliers across the folds in 5-fold cross-validation. However, in practice, it is not possible to know the allocation of outliers across the folds during cross-validation. Since it gives the best performance for most of the considered scenarios, use of FRFN.ANN-BR is suggested in terms of mean absolute error and mean square error. In terms of percentage bias, FRFN.ANN-LM is also suggested.
- When the dataset is large in terms of the number of observations and features, use of FRFN.ANN-LM is preferable over FRFN.ANN-BR due to the computational time requirements.

Overall, we challenged 36 ML methods including ANN, SVM, FRFN implementations with 5-fold cross-validation under the existence of outliers in the sample. The fuzzy c-means clustering with a noise cluster provided FRF method with a considerably higher robustness against the existence of outliers than the other considered ML methods. In the general sense, ANN implementations have higher robustness than SVMs. For ANNs, BR and LM kernels produce promising results. For SVMs, the best performing kernels are the Gaussian and the polynomial kernels. These results are also supported by our numerical study with 15 benchmarking datasets.

The data generation strategy for this study includes the generation of independent features from skewed and symmetric distributions such as gamma, truncated-normal, and uniform using a linear link function. Also, the dependent feature is generated in the interval scale. This implies the generalizability of the results for the regression problem to the cases where the true data generation mechanism is composed of percentages, extreme values, and normally distributed features with an interval scale dependent feature. Therefore, in real applications, the practitioners can rely on our results if the source of their data generation mechanism includes multiple independent features with different characteristics. However, our results should be used cautiously when there is a straightforward relationship between two features, the existence of a nonlinear relationship between the dependent and exploratory features is known, or the dependent feature is measured in ordinal, nominal or ratio scales.

The main advantage of the proposed FRFN approach is that it improves the robustness of the ANNs and SVMs when the data is either uncontaminated or contaminated by outliers with different magnitudes. Its main disadvantage is the increased computation time due to the addition of another fuzzy layer to the implementation. Since we do not include any violations of assumptions to the synthetic data generation, one of the limitations of the simulation study is that the inferences are drawn under the assumption that the classical regression assumptions hold. However, we observe similar results to those of the simulation study when we consider 15 real datasets including a variety of sample sizes and number of features. Investigation of the cases where any of the regression assumptions do not hold is a future direction. Since the efficiency of using the concept of noise cluster along with FRFs is clearly observed in this study, another possible future direction is to implement deep learning algorithms for the estimation of fuzzy cluster parameters.

CRediT authorship contribution statement

Srinivas Chakravarty: Conceptualization, Methodology, Software, Formal analysis, Data curation, Writing - original draft, Writing - review & editing. **Haydar Demirhan:** Conceptualization, Methodology, Software, Formal analysis, Writing - original draft, Writing - review & editing. **Furkan Baser:** Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing.

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.

Acknowledgments

The authors would like to thank three reviewers for detailed comments that considerably improved clarity and quality of the article. SC is supported by an Australian Government Research Training Program Scholarship through RMIT University, Australia.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.asoc.2020.106535.

References

[1] Y. Li, P. Zhou, An outlier detection method and its application to multicore-chip power estimation, in: Proceedings of International Conference on ASIC, 2018, pp. 460–463.
[2] L. Yang, H. Dong, Robust support vector machine with generalized quantile loss for classification and regression, Appl. Soft Comput. J. 81 (2019) 105483.
[3] N.J. Shah, H.A. Patil, A novel approach to remove outliers for parallel voice conversion, Comput. Speech Lang. 58 (2019) 127–152.
[4] E. Egrioglu, U. Yolcu, E. Bas, A. Dalar, Median-Pi artificial neural network for forecasting, Neural Comput. Appl. 31 (2019) 307–316.
[5] O. Panagopoulos, P. Xanthopoulos, T. Razzaghi, O. Şeref, Relaxed support vector regression, Ann. Oper. Res. 276 (2019) 191–210.
[6] K.S. Kula, A. Apaydin, Fuzzy robust regression analysis based on the ranking of fuzzy sets, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 16 (2008) 663–681.
[7] K.Ş. Kula, A. Apaydın, Hypotheses testing for fuzzy robust regression parameters, Chaos Solitons Fractals 42 (2009) 2129–2134.
[8] G. Xu, Z. Cao, B.-G. Hu, J.C. Principe, Robust support vector machines based on the rescaled hinge loss function, Pattern Recognit. 63 (2017) 139–148.
[9] H.-J. Xing, M. Ji, Robust one-class support vector machine with rescaled hinge loss function, Pattern Recognit. 84 (2018) 152–164.
[10] W. Zhu, Y. Song, Y. Xiao, A new support vector machine plus with pinball loss, J. Classification 35 (2018) 52–70.
[11] H. Yan, D.J. Yu, Short-term traffic condition prediction of urban road network based on improved SVM, in: 2017 International Smart Cities Conference, ISC2 2017, 2017.
[12] S. Suzumura, K. Ogawa, M. Sugiyama, I. Takeuchi, Outlier path: A homotopy algorithm for robust SVM, in: 31st International Conference on Machine Learning, ICML 2014, 2014, pp. 2797–2806.
[13] M. Singla, K.K. Shukla, Robust statistics-based support vector machine and its variants: a survey, Neural Comput. Appl. (2019) 1–22.
[14] C.A. Araújo Júnior, P.D.d. Souza, A.L.d. Assis, C.D. Cabacinha, H.G. Leite, C.P.B. Soares, A.A.L.D. Silva, R.V.O. Castro, Artificial neural networks, quantile regression, and linear regression for site index prediction in the presence of outliers, Pesqui. Agropecu. Bras. 54 (2019).
[15] U. Yolcu, E. Bas, E. Egrioglu, C.H. Aladag, A new multilayer feedforward network based on trimmed mean neuron model, Neural Netw. World 25 (2015) 587–602.
[16] C. Aladag, E. Egrioglu, U. Yolcu, Robust multilayer neural network based on median neuron model, Neural Comput. Appl. 24 (2014) 945–956.
[17] E. Bas, V.R. Uslu, E. Egrioglu, Robust learning algorithm for multiplicative neuron model artificial neural networks, Expert Systems Appl. 56 (2016) 80–88.
[18] Y.-L. Lin, J.-G. Hsieh, J.-H. Jeng, W.-C. Cheng, On least trimmed squares neural networks, Neurocomputing 161 (2015) 107–112.
[19] S.H. Liao, J.Y. Chang, C.T. Lin, Study on least trimmed absolute deviations artificial neural network, in: 2013 International Conference on Fuzzy Theory and its Applications, 2013, pp. 156–160.
[20] G. Beliakov, A. Kelarev, J. Yearwood, Derivative-free optimization and neural networks for robust regression, Optimization 61 (2012) 1467–1490.
[21] A. Khamis, Z. Ismail, K. Haron, A. Tarmizi Mohammed, The effects of outliers data on neural network performance, J. Appl. Sci. 5 (2005) 1394–1398.
[22] Y. Oi, Y. Endo, Kernel fuzzy c-regression based on least absolute deviation with modified huber function, J. Adv. Comput. Intell. Intell. Inform. 23 (2019) 571–576.
[23] M.G. Akbari, G. Hesamian, A partial-robust-ridge-based regression model with fuzzy predictors-responses, J. Comput. Appl. Math. 351 (2019) 290–301.
[24] E.d.A. Lima Neto, F.d.A.T. de Carvalho, An exponential-type kernel robust regression model for interval-valued variables, Inform. Sci. 454–455 (2018) 419–442.
[25] J. Li, W. Zeng, J. Xie, Q. Yin, A new fuzzy regression model based on least absolute deviation, Eng. Appl. Artif. Intell. 52 (2016) 54–64.
[26] R.N. Davé, Characterization and detection of noise in clustering, Pattern Recognit. Lett. 12 (1991) 657–664.
[27] Z. Ghafoori, S.M. Erfani, S. Rajasegarar, J.C. Bezdek, S. Karunasekera, C. Leckie, Efficient unsupervised parameter estimation for one-class support vector machines, IEEE Trans. Neural Netw. Learn. Syst. 29 (2018) 5057–5070.
[28] J. Chachi, M. Roozbeh, A fuzzy robust regression approach applied to bedload transport data, Comm. Statist. Simulation Comput. 46 (2017) 1703–1714.
[29] M. Maniruzzaman, M. Rahman, M. Al-MehediHasan, H. Suri, M. Abedin, A. El-Baz, J. Suri, Accurate diabetes risk stratification using machine learning: Role of missing value and outliers, J. Med. Syst. 42 (2018) 1–17.
[30] I.B. Türkşen, Fuzzy functions with LSE, Appl. Soft Comput. J. 8 (2008) 1178–1188.
[31] A. Celikyilmaz, I. Burhan Turksen, Enhanced fuzzy system models with improved fuzzy clustering algorithm, IEEE Trans. Fuzzy Syst. 16 (2008) 779–794.
[32] A. Çelikyılmaz, I. Burhan Türkşen, Fuzzy functions with support vector machines, Inform. Sci. 177 (2007) 5163–5177.
[33] A. Celikyilmaz, I.B. Turksen, Uncertainty modeling of improved fuzzy functions with evolutionary systems, IEEE Trans. Syst. Man Cybern. B 38 (2008) 1098–1110.
[34] J.C. Bezdek, Objective function clustering, in: Pattern Recognition with Fuzzy Objective Function Algorithms, Springer, 1981, pp. 43–93.
[35] D. Dua, E. Karra Taniskidou, UCI Machine Learning Repository, School of Information and Computer Science, University of California, Irvine, CA, 2017, in, 2019.
[36] V. Arel-Bundock, Rdatasets, An archive of datasets distributed with R, 2020, https://vincentarelbundock.github.io/Rdatasets/datasets.html, Accessed: 28/04/2020.
[37] L. Scott, 10 open datasets for linear regression, 2020, https://lionbridge.ai/datasets/10-open-datasets-for-linear-regression/, Accessed: 28/04/2020.
[38] C. Sammut, G.I. Webb, Encyclopedia of Machine Learning, Springer US, Boston, MA, Boston, MA, 2010.
[39] J. Kacprzyk, W. Pedrycz, Springer Handbook of Computational Intelligence, first ed., Springer Berlin Heidelberg, Berlin, Heidelberg, Berlin, Heidelberg, 2015.
[40] G. Arulampalam, A. Bouzerdoum, A generalized feedforward neural network architecture for classification and regression, Neural Netw. 16 (2003) 561–568.
[41] M.T. Hagan, H.B. Demuth, M.H. Beale, Neural Network Design, PWS Pub., Boston, 1995.
[42] D.W. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, J. Soc. Ind. Appl. Math. 11 (1963) 431–441.
[43] D.J.C. MacKay, Bayesian interpolation, Neural Comput. 4 (1992) 415–447.
[44] F. Dan Foresee, M.T. Hagan, Gauss-Newton approximation to Bayesian learning, in, vol. 1933, 1997, pp. 1930–1935.
[45] P.E. Gill, W. Murray, M.H. Wright, Practical Optimization, Academic Press, New York, 1981.
[46] R. Battiti, First- and second-order methods for learning: Between steepest descent and Newton's method, Neural Comput. 4 (1992) 141–166.
[47] M. Riedmiller, H. Braun, A direct adaptive method for faster backpropagation learning: the RPROP algorithm, in, vol. 581, 1993, pp. 586–591.
[48] M. Møller, M.F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Netw. 6 (1993) 525–533.
[49] M.J.D. Powell, Restart procedures for the conjugate gradient method, Math. Program. 12 (1977) 241–254.
[50] H.W. Sorenson, Comparison of some conjugate direction procedures for function minimization, J. Franklin Inst. B 288 (1969) 421–441.
[51] B. Lammers, ANN2: Artificial neural networks for anomaly detection, 2020, R package version 2.3.3. https://CRAN.R-project.org/package=ANN2.
[52] MathWorks®, Support Vector Machines for Binary Classification, in, MathWorks®, 1994.
[53] C. Campbell, Y. Ying, Learning with Support Vector Machines, Morgan & Claypool, San Rafael, Calif. (1537 Fourth Street, San Rafael, CA 94901 USA), 2011.
[54] MathWorks®, Support Vector Machine (SVM), in, MathWorks®, 1994.
[55] A. Kowalczyk, Support Vector Machines Succintly, SyncFusion® Inc., 2017.
[56] T. Hofmeister, Qrsvm: SVM quantile regression with the pinball loss, 2017, R package version 0.2.1. https://CRAN.R-project.org/package=qrsvm.

[57] F. Baser, H. Demirhan, A fuzzy regression with support vector machine approach to the estimation of horizontal global solar radiation, Energy 123 (2017) 229–240.
[58] R.N. Davé, S. Sen, Robust fuzzy clustering of relational data, IEEE Trans. Fuzzy Syst. 10 (2002) 713–727.
[59] C. Subbalakshmi, R. Sayal, H.S. Saini, Cluster validity using modified fuzzy silhouette index on large dynamic data set, in: Computational Intelligence in Data Mining, Springer, 2020, pp. 1–14.
[60] M.B. Ferraro, P. Giordani, A. Serafini, Fclust: An R package for fuzzy clustering, R J. 11 (1) (2019) https://journal.r-project.org/archive/2019/RJ-2019-017/RJ-2019-017.pdf.
[61] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[62] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Amer. Statist. Assoc. 32 (1937) 675–701.
[63] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, third ed., Chapman & Hall/CRC, Boca Raton, 2004.
[64] R.L. Iman, J.M. Davenport, Approximations of the critical region of the fbietkan statistic, Comm. Statist. Theory Methods 9 (1980) 571–595.
[65] P.B. Nemenyi, Distribution-Free Multiple Comparisons (doctoral dissertation, princeton university, 1963), Vol. 25, Dissertation Abstracts International, 1963, p. 1233.
[66] W. Ma, J. Duan, H. Zhao, B. Chen, Chebyshev functional link artificial neural network based on correntropy induced metric, Neural Process. Lett. 47 (2018) 233–252.