Available online at http://scik.org
Commun. Math. Biol. Neurosci. 2022, 2022:16
https://doi.org/10.28919/cmbn/6908
ISSN: 2052-2541

PROPOSED TWO VARIABLE SELECTION METHODS FOR BIG DATA: SIMULATION AND APPLICATION TO AIR QUALITY DATA IN ITALY

AHMED A. EL-SHEIKH[1], MOHAMED R. ABONAZEL[1,*], MOHAMED C. ALI[2]

[1]Department of Applied Statistics and Econometrics, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza, Egypt

[2]Faculty of Business Administration, Deraya University, Minya, Egypt

Abstract: In this era of big data, considerable amounts of information data are produced daily with the rapid development of technology. In various fields, such as engineering, computer science, and finance, several statistical and machine learning methods are used to uncover useful information and patterns behind these enormous datasets. Neural networks (NN) and random forests (RF) are the common model selection (variable selection) methods in machine learning. The least absolute shrinkage and selection operator (LASSO) and principal component analysis are the statistical methods. In this study, we propose two methods: a combination of NN and LASSO and a combination of NN and RF. We use Monte Carlo simulation and a real data application (air quality data in Italy) to investigate the performance of the classical methods (ordinary least square and feed-forward NN) and two proposed methods by the goodness of fit criteria. The results showed that the proposed methods perform better than the classical methods.

Keywords: artificial neural network; random forests; least absolute shrinkage and selection operator; principal component analysis.

2010 AMS Subject Classification: 92B20, 62H25.

[*]Corresponding author
E-mail address: mabonazel@cu.edu.eg
Received October 18, 2021

1. INTRODUCTION

Variable selection methods in regression analysis held statistical value, especially for models with multiple independent variables and recent developments in model selection methods for extracting useful information from large databases (big data) in all fields. However, traditional statistical methods cannot manage big data. Extracting useful information from these complex and informative rules has become a major challenge. The commonly used machine learning methods are neural network (NN), random forests (RF), and statistical tools, such as robust least absolute shrinkage and selection operator (LASSO) and principal component analysis (PCA). This study proposes two selection methods by combining NN with LASSO and RF. We compared the performance of classical selection methods (ordinary least square (OLS) and feed-forward NN) with the proposed methods through simulation methods using and life data application. Finally, we concluded that the proposed methods perform better than the classical methods with a minimum error.

Wang et al. [1] used the idea of quantile regression and random LASSO in the case of highly correlated variables. Mansoor et al. [2] used a feed-forward Neural networks (FFNN) on a dataset concerning commercial buildings because of a possible demand response program application. They used the machine learning method that deserves more attention, i.e., the RF method, which dominates all other methods. The combination of machine learning methods, i.e., RF with NN and LASSO with NN, produces new powerful methods.

The remainder of the paper is organized as follows: the classical variable selection methods are introduced in Section 2. Section 3 presents the proposed methods. In Section 4, we present the Monte Carlo simulation. In Section 5, we discuss the application of the proposed methods. Finally, Section 6 presents the conclusion.

2. VARIABLE SELECTION METHODS

2.1. Feed-Forward Neural Networks

A big deal of hyperbole has been devoted to NNs in their first wave in around 1960 [3,4] and their renaissance in around 1985 (inspired by [5]). However, the biologically relevant ideas have been detracted from the essence of what is being discussed. They are irrelevant to practical applications in pattern recognition. Because NNs have become a popular subject, they have collected numerous methods loosely related and not biologically motivated. A formal definition of a feed-forward network is given in the glossary. They basically contained units that have one-way connections to other units; the units can be labeled from inputs (low numbers) to outputs (high numbers) to connect to units with higher numbers. The units can always be arranged in layers so that connections go from one layer to another. This can be best observed graphically (Fig. 1). Each unit sums its inputs and adds a constant (the “bias”) to form a total input x_j and applies a function $f_j$to $x_j$to output y_j. The links have weights w_{ij},which multiply the signals traveling with them by that factor. The input units are used to distribute the inputs, so have f ≡1. Thus, a network such as that given in Fig.1, represents the function

$$y_k = f_k\left(\alpha_k + \sum_{j\to k} w_{jk} f_j\left(\alpha_j + \sum_{i\to j} w_{ij} x_i\right)\right). \quad (1)$$

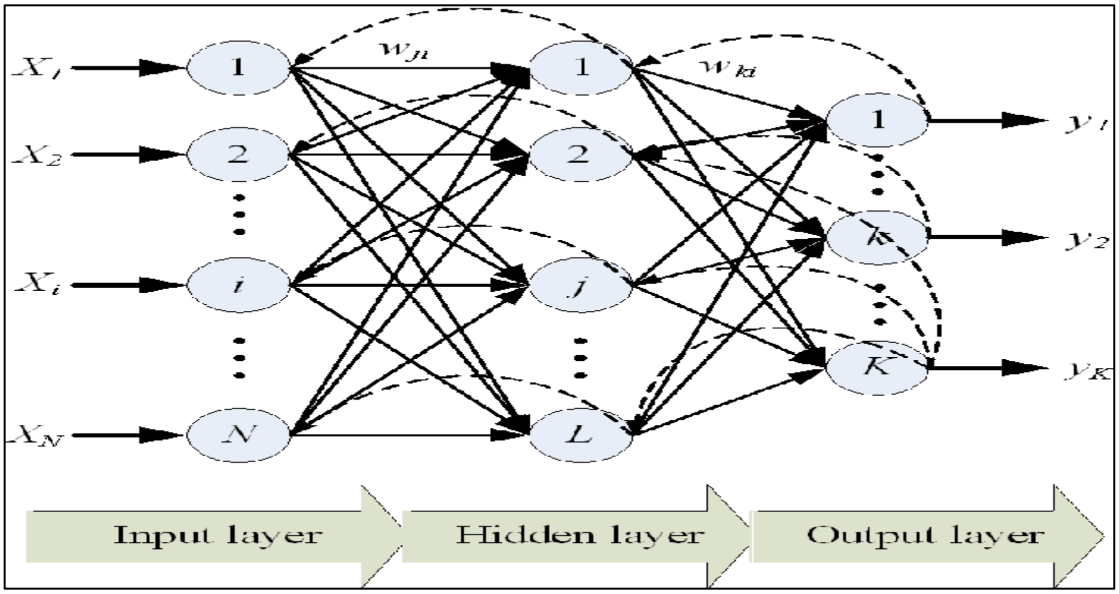

Fig. 1: Feed-forward neural network method Source: [7].

Fig. 1 shows a generic feed-forward network with a single hidden layer. To avoid overcrowding, bias units are shown for each layer; however, they can be the same unit from inputs to outputs. The functions f_j are almost invariably considered as linear, logistic with $f(x) = \ell(x) = e^x/(1 + e^x)$)or threshold functions with (with f(x) = I(x > 0)). A variant takes hyperbolic tangent units with f(x) = tan h(x) = $(e^x - 1)/(e^x + 1) = 2\,\ell(x) - 1$. However, it only introduces a linear transformation that can be absorbed into the weights, except at the output units. Only threshold units give a genuine multilayer extension of the perceptron, and such networks were considered by [4, 6]. The general definition allows more than one hidden layer, and it also allows “skip-layer” connections from input to output. If all units in a layer have the same functions f_h,f_0, we have

$$y_k = f_0\big(\alpha_k + \textstyle\sum_{i\to k} w_{ik}x_i + \sum_{j\to k} w_{jk} f_h\,\big(\alpha_j + \sum_{i\to j} w_{ij}x_i\big)\big). \tag{2}$$

The bias terms can be eliminated by introducing a new unit 0 (the bias unit), which is permanently at +1 and connected to all other units. We set $\mathrm{w}_{0\mathrm{j}} = \alpha_\mathrm{j}$. This is the same concept as incorporating the constant term in the design matrix of regression by including a column of 1’s. This is shown in Fig. 1. The general form is then given as

$$y_k = f_0\big(\textstyle\sum_{i\to k} w_{ik}x_i + \sum_{j\to k} w_{jk} f_h\,\big(\sum_{i\to j} w_{ij}x_i\big)\big). \tag{3}$$

Notably, if the hidden layer contains logistic units, adding skip-layer connections is not more general because we can add another unit per output in the hidden layer with input weights w_{jk}/G and output weight G to only unit k. Then, for large G, we only use the central, linear part of the range of the logistic function. However, skip-layer connections can be easier to implement and interpret. NN with a single logistic output unit is a nonlinear extension of the logistic regression. With several logistic output units, it corresponds to linked logistic regressions of each class vs. others. The terminology of NNs can be very confusing. Fig. 1 is sometimes considered to have three layers (which seems visually correct), two layers (as the input layer does nothing), and one hidden layer (as the states of the units in the central layer cannot be inspected from outside the “black box”). Che et al. [7] referred to the inputs, outputs, and hidden layer because we will always have only one hidden layer. We will extend our notation to allow every unit j to have an input $\mathrm{x}_\mathrm{j}$and output y_j. The inputs to the entire network are the inputs to the input units, and the outputs from the entire network are those of the output units. The signal paths through the network are determined using the following equation:

$$y_j = f_j(x_j),\, x_j = \textstyle\sum_{i\to j} w_{ij}y_i. \tag{4}$$

The conditions on the sum can be neglected by defining w_{ij} to be zero for all nonexistent links. When programming, numbering the units by layer is necessary, so that all units in the first layer precede those in the first hidden layer. Then, we know $\mathrm{w}_{\mathrm{ij}} = 0$, unless i< j. Che et al. [7] briefly consider how such functions were suggested and the theory that shows that they form large and flexible classes of functions. In practice, the main issues are how the parameters and weights should be selected and how the architecture (the number of layers and the number of units in each, and which connections to include) should be selected [6].

2.2. Principal Component Analysis with Neural Network (PCANN)

This method is based on the spectral analysis of the second-order moment matrix called a correlation matrix that statistically characterizes a random vector used by [8] and introduced by [6].

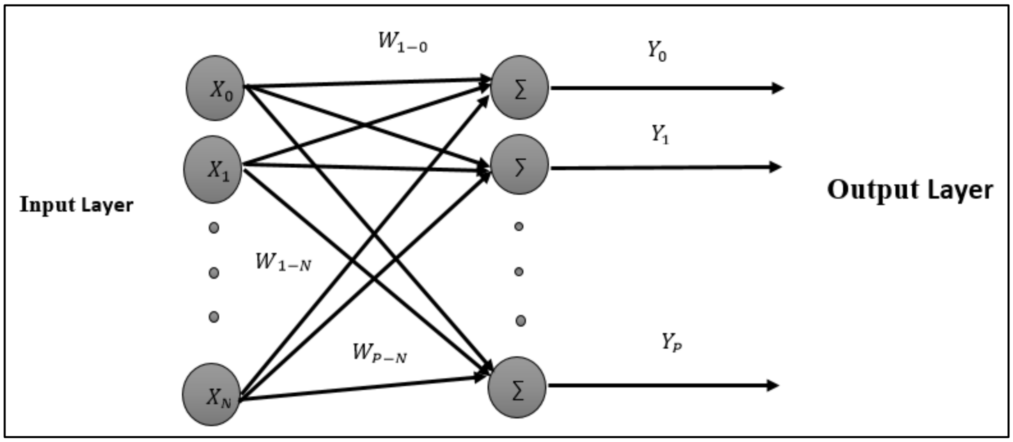

Fig. 2: Architecture of the PCA network. Source: [9]

The single-neuron model was extended from N to P feed-forward network model to extract the first P of PCs. Fig. 2 shows the architecture of the PCA network. The output of the network is given by

$$\text{y} = w^T\text{x}, \tag{5}$$

where y $= \left(y_1, y_2, \cdots, y_p\right)'$, x $= (x_1, x_2, \cdots, x_N)'$, W $= (w_1, w_2, \cdots, w_P)$, and $w_i = (w_{1i}, w_{2i}, \cdots, w_{Ni})'$; w_{jP} is the weight from the *j*th input to the *p*th neuron [10].

2.3. Least Absolute Shrinkage and Selection Operator

For a given collection of N predictor-response pairs $\{(x_i, y_i)\}_{i=1}^{N}$, LASSO obtains the solution $\left(\beta_0, \beta_j\right)$ to the following optimization problem:

$$\underset{\beta_0,\beta_j}{Min}\left\{\frac{1}{2N}\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{K}x_{ij}\,\beta_j\right)^2\right\},\quad \text{Subject to } \sum_{j=1}^{K}\left|\beta_j\right| \le \text{t}. \tag{6}$$

The constraint $\sum_{j=1}^{K}\left|\beta_j\right| \le t$ can be written more compactly as the l_1-norm constraint $\left\|\beta_j\right\| \le t$. Furthermore, (6) is usually represented using matrix-vector notation. Assuming $y = (y_1, \ldots, y_N)'$to be the N-vector of responses, X is a $n \times p$ matrix with $x_i \in \mathbb{R}^K$ in its i^{th} row, and β is a vector of β_j, then the optimization problem (6) can be re-expressed as follows:

$$\underset{\beta_0,\beta}{Min}\left\{\frac{1}{2N}\|y-\beta_0\mathbf{1}-X\beta\|_2^2\right\},\quad \text{Subject to } \|\beta\|_1 \le \text{t} \tag{7}$$

where **1** is the vector of N ones, and $\|.\|_2$ is the usual Euclidean norm on vectors. The bound t is a type of “budget”. It limits the sum of the absolute values of the parameter estimates. Because a shrunken parameter estimate corresponds to a more heavily constrained model, this budget limits how well we can fit the data [11, 12].

2.4. Random Forests

Similar to bagging, a large number of tree classification or regression trees are grown with bootstrap samples from training data. However, as each tree is grown, a random sample of predictors is taken before splitting each node. For example, if there are 20 predictors, a random five are selected as candidates for defining the split. Then, the best split is constructed, as usual, but it is selected only from the five chosen. This process is repeated for each prospective split without pruning. Thus, each tree is produced from a random sample of cases and a random sample of predictors at each split. The mean or proportion for each tree’s terminal nodes is determined similar to bagging. Finally, for each case, the over trees are averaged as in bagging, but only when that case is out-of-bag (OOB). Breiman [13] called such a procedure a “random forest”. This method was used by Liu et al. [14] to identify spatial poverty determinants in rural China. Ludwig et al. [15] used big data analytics for feature selection to forecast electricity prices using the LASSO and RF.

The RF algorithm is similar to the bagging algorithm. Assuming N to be the number of observations in the training data and assuming that the response variable is binary. The RF algorithm is designed through the following steps [16]:

Algorithm: Random Forests Method

Step 1. Taking a random sample of size N with replacement from the data.

Step 2. Taking a random sample without replacement of the predictors.

Step 3. Constructing the first recursive partition of the data as usual.

Step 4. Repeating Step 2 for each subsequent split until the tree is as large as desired. This usually leads to one observation in each terminal node. Prune is not done. Computing each terminal node proportion as usual.

Step 5. Dropping the OOB data down the tree. Storing the class assigned to each observation along with each observation’s predictor values.

Step 6. Repeating Steps 1–5 several times (tree = 500).

Step 7. Using only the class assigned to each observation when the observation is OOB, counting the number of times over trees so that the observation is classified into one category and the number of times over trees classified in the other category.
Step 8. Assigning each case to a category by a majority vote over the set of trees when the case is OOB. Thus, if 51% of the time over several trees for a given case is classified as 1, it becomes its estimated classification [16].

3. PROPOSED METHODS

In this section, we introduce a combination of RF, LASSO, and NN to obtain the two proposed methods (RFNN and LASSONN). We hope obtaining a more powerful goodness of fit compared with the classical methods (OLS and PCANN).

3.1. LASSONN

In this method, we combine NN and LASSO to obtain a new estimator with a better goodness fit. We can simplify this algorithm in the following steps:

Algorithm 1: LASSONN

Step 1. Starting with LASSO model
Step 2. Choosing the selected variables from the LASSO model
Step 3. Entering the selected variables to NN.

3.2. RFNN

In this method, we combine NN and RF to obtain a new estimator with a better goodness fit. Then, we randomly permute the values of X^J in OOB_t to obtain a perturbed sample denoted by $\widetilde{OOB}_t^J$ and calculate $err\widetilde{OOB}_t^J$, the error of the predictor t on the perturbed sample. Variable importance of X^J is expressed as follows:

$$\mathrm{VI}(X^J) = \frac{1}{ntree}\sum_t \left(err\widetilde{OOB}_t^J - errOOB_t\right). \tag{8}$$

where the sum is over all trees t of RF and *ntree* denotes the number of RF trees. Notably, we used this definition of importance and not the normalized one. Instead of considering that the raw VI are independent replicates, normalizing them and assuming the normality of these scores, we prefer a fully data-driven solution. This is a key point of our strategy: we prefer directly estimating the variability of importance across repetitions of forests instead of using normality when *ntree* is sufficiently large, which is valid only under specific conditions. We propose the following two-step procedure. The first one is common, whereas the second one depends on the objective.

Step 1. Preliminary elimination and ranking: Computing the RF scores of importance, cancelling the variables of small importance, and arranging the remaining variables in decreasing order of importance.

Step 2. Variable selection: For interpretation: constructing the nested collection of RF models involving the k first variables, for k = 1 to m, and selecting the variables involved in the model leading to the smallest OOB error. For prediction: starting from the ordered variables retained for interpretation, constructing an ascending sequence of RF models by invoking and testing the variables stepwise. The variables of the last model are selected. This is a sketch of the procedure, and more details are required for its effectiveness [17]. Then we can use the selected variables from RF as input variables in NN. We can simplify this algorithm through the following steps:

Algorithm 2: RFNN

Step1. Starting with RF method

Step2. Choosing the selected variables using RF method

Step3. Entering the selected variables to NN.

4. MONTE CARLO SIMULATION STUDY

In this section, we conduct a comparative study between the classical estimator (PCANN) and two proposed estimators (LASSONN and RFNN) via the Monte Carlo simulation. In our simulation study, we used different simulation factors (see Table 1) to investigate the performance of these estimators in different situations. R-software version 4 was used to perform the simulation study, see [18, 19].

We generate independent variables, as in [18, 20, 21] from a multivariate normal distribution $MVN(0, \Sigma_X)$, where diag$(\Sigma_X) = 1$ and off–diag $(\Sigma_X) = \rho_x$; ρ_x = 0.90 and 0.95 [22, 23], where ρ_x is the correlation coefficient between the independent variables. We also generate an error using a standard normal distribution with two outlier rates (OR) of 10% and 15% [19, 24, 25]. We used a simulation study with different sample sizes N = 75,150 and 300, and independent variables K = 10, 20, 30, 40, 60, and 70. The true regression parameter is 0.5 and 0.001 [26]. Then, we construct the LASSO, RF, PCANN, and proposed estimators for their comparison. Fig. 3 shows the simulation design flowchart.

Table 1: Simulation factors

Factor	**Values**		
$\boldsymbol{\rho_x}$	0.90		0.95
OR	0.10		0.15
N	75	150	300
K	10, 20, 30	10, 20, 30, 40, 60	10, 20, 70

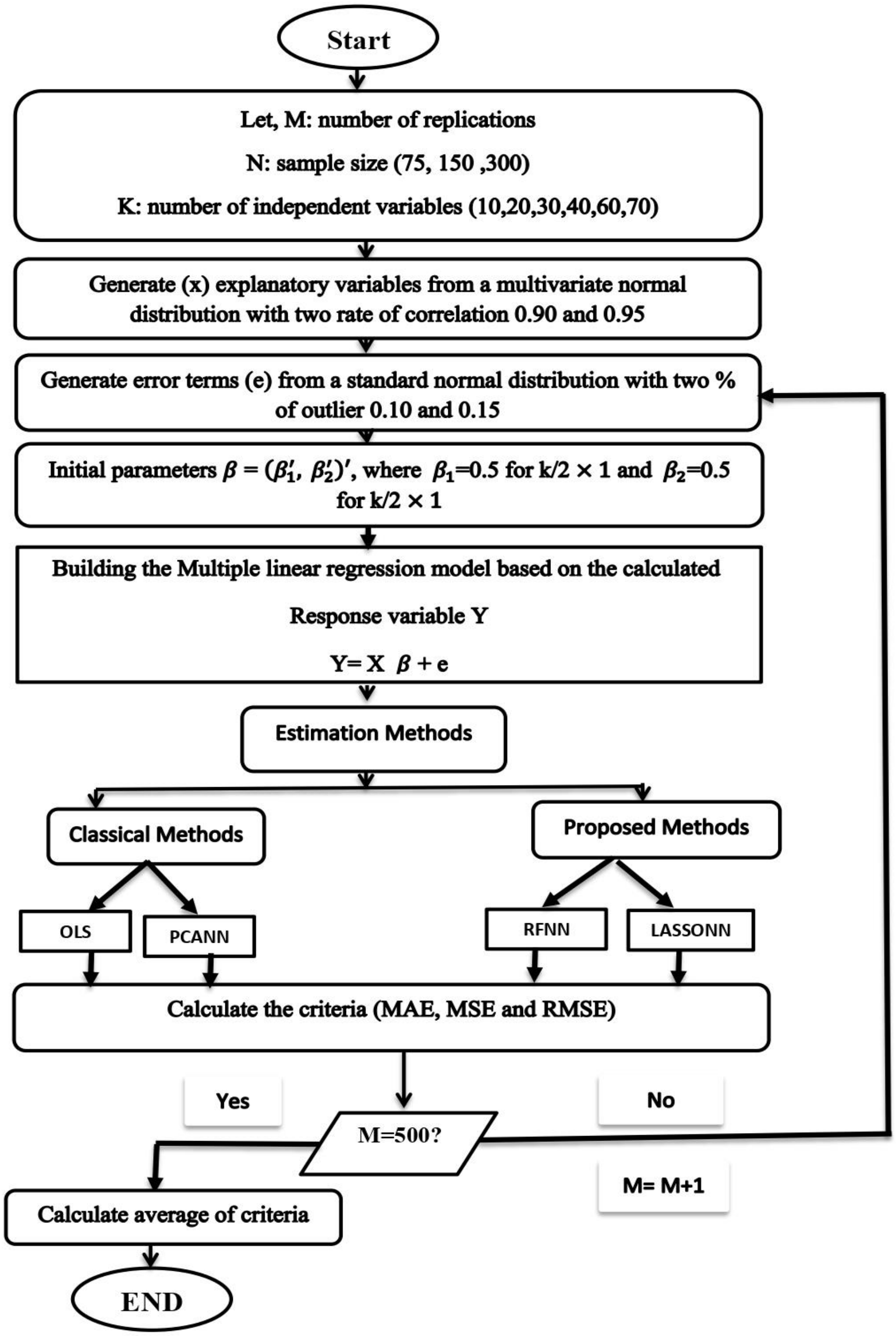
Start
Let, M: number of replications
N: sample size (75, 150 ,300)
K: number of independent variables (10,20,30,40,60,70)
Generate (x) explanatory variables from a multivariate normal distribution with two rate of correlation 0.90 and 0.95
Generate error terms (e) from a standard normal distribution with two % of outlier 0.10 and 0.15
Initial parameters $\beta = (\beta_1', \beta_2')'$, where β_1=0.5 for k/2 × 1 and β_2=0.5 for k/2 × 1
Building the Multiple linear regression model based on the calculated
Response variable Y
Y= X β + e
Estimation Methods
Classical Methods
Proposed Methods
OLS
PCANN
RFNN
LASSONN
Calculate the criteria (MAE, MSE and RMSE)
Yes
No
M=500?
M= M+1
Calculate average of criteria
END

Fig. 3: Simulation flowchart

In our study, we used three goodness of fit criteria for verification: mean square error (MSE), mean absolute error (MAE), and root mean square error (RMSE). Furthermore, we presented for each method the number of selected variables (#SVs) and the number of principal comments (#PCs). The simulation results are presented in Tables 2–13 for a different number of independent variables (K = 10, 20, 30, 40, 60, 70) and different sample sizes (N = 75, 150, 300). Two different rates of correlation and outliers are (0.9, 0.95), (0.10, 0.15), respectively. Notably, when the sample size (N) is 75, independent variables (K) are 10, 20, and 30; when N is 150, K is 10, 20, 30, 40, and 60; when N is 300, K is 10, 20, and 70.

Table 2: Simulation results when N = 75, ρ_x = 90%, and OR = 10%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.043	1.853	1.777	1.788
	MSE	8.191	7.193	6.796	6.880
	RMSE	2.861	2.681	2.606	2.622
	#SVs (#PCs)	10	10 (9)	3	4
20	**MAE**	2.813	2.334	2.212	2.203
	MSE	13.954	9.771	9.145	9.112
	RMSE	3.735	3.125	3.024	3.018
	#SVs (#PCs)	20	20 (14)	3	7
30	**MAE**	4.612	2.811	2.566	2.571
	MSE	37.051	13.632	12.097	12.078
	RMSE	6.086	3.692	3.478	3.475
	#SVs (#PCs)	30	30 (19)	4	10

Table 3: Simulation results when N=75, ρ_x= 90% and OR = 15%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.37	2.133	2.059	2.063
	MSE	10.182	8.675	8.193	8.258
	RMSE	3.19	2.945	2.862	2.873
	#SVs (#PCs)	10	10(9)	2	4

20	**MAE**	3.139	2.619	2.481	2.478
	MSE	16.821	11.918	10.993	11.01
	RMSE	4.101	3.452	3.315	3.318
	#SVs (#PCs)	20	20(14)	3	7
30	**MAE**	5.125	3.066	2.792	2.839
	MSE	45.571	15.56	13.575	13.963
	RMSE	6.75	3.944	3.684	3.736
	#SVs (#PCs)	30	30(19)	4	9

Table 4: Simulation results when N=75, ρ_x = 95% and OR = 10%

K	**Criteria**	**OLS**	**PCANN**	**RFNN**	**LASSONN**
10	**MAE**	2.119	1.824	1.795	1.8
	MSE	8.704	7.097	6.972	6.992
	RMSE	2.95	2.664	2.64	2.644
	#SVs (#PCs)	10	10(8)	2	4
20	**MAE**	2.805	2.247	2.183	2.193
	MSE	13.89	9.423	9.075	9.151
	RMSE	3.726	3.069	3.012	3.025
	#SVs (#PCs)	20	20(11)	3	6
30	**MAE**	4.531	2.64	2.493	2.524
	MSE	35.712	12.515	11.782	11.932
	RMSE	5.975	3.537	3.432	3.454
	#SVs (#PCs)	30	30(15)	4	8

Table 5: Simulation results when N = 75, ρ_x = 95%, and OR = 15%

K	**Criteria**	**OLS**	**PCANN**	**RFNN**	**LASSONN**
10	**MAE**	2.442	2.114	2.073	2.090
	MSE	10.795	8.745	8.457	8.581
	RMSE	3.285	2.957	2.908	2.929
	#SVs (#PCs)	10	10(8)	2	3

20	**MAE**	3.212	2.512	2.451	2.447
	MSE	17.706	11.364	10.881	10.869
	RMSE	4.207	3.371	3.298	3.296
	#SVs (#PCs)	20	20(11)	3	5
30	**MAE**	5.089	2.881	2.747	2.759
	MSE	45.324	14.258	13.459	13.530
	RMSE	6.732	3.775	3.668	3.678
	#SVs (#PCs)	30	30(15)	3	8

Table 6: Simulation results when N=150, ρ_x = 90% and OR = 10%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	1.891	1.899	1.830	1.834
	MSE	7.883	7.890	7.581	7.598
	RMSE	2.807	2.808	2.753	2.756
	#SVs (#PCs)	10	10 (9)	3	5
20	**MAE**	2.172	2.281	2.103	2.095
	MSE	9.389	10.034	9.087	9.061
	RMSE	3.064	3.167	3.014	3.01
	#SVs (#PCs)	20	20 (16)	4	8
30	**MAE**	2.528	2.677	2.360	2.340
	MSE	11.635	12.947	10.919	10.785
	RMSE	3.411	3.598	3.304	3.284
	#SVs (#PCs)	30	30 (23)	5	11
40	**MAE**	2.929	2.981	2.643	2.652
	MSE	14.917	16.103	13.826	13.809
	RMSE	3.862	4.012	3.718	3.716
	#SVs (#PCs)	40	40 (28)	6	14
60	**MAE**	4.825	3.892	3.415	3.511
	MSE	38.967	27.962	23.718	24.411
	RMSE	6.242	5.287	4.87	4.94
	#SVs (#PCs)	60	60 (34)	8	20

Table 7: Simulation results when N=150, ρ_x = 90% and OR = 15%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.304	2.296	2.211	2.212
	MSE	10.294	10.201	9.709	9.717
	RMSE	3.208	3.193	3.115	3.117
	#SVs (#PCs)	10	10 (9)	3	4
20	**MAE**	2.605	2.702	2.487	2.486
	MSE	12.372	12.989	11.505	11.555
	RMSE	3.517	3.604	3.391	3.399
	#SVs (#PCs)	20	20 (16)	4	7
30	**MAE**	2.937	3.047	2.666	2.68
	MSE	14.896	15.806	12.861	12.936
	RMSE	3.859	3.975	3.586	3.596
	#SVs (#PCs)	30	30 (23)	4	10
40	**MAE**	3.421	3.364	2.937	2.944
	MSE	19.601	19.483	15.873	15.876
	RMSE	4.427	4.413	3.984	3.984
	#SVs (#PCs)	40	40 (28)	5	13
60	**MAE**	5.304	4.124	3.606	3.703
	MSE	46.949	30.663	25.395	26.243
	RMSE	6.851	5.537	5.039	5.122
	#SVs (#PCs)	60	60 (33)	7	19

Table 8: Simulation results when N=150, ρ_x = 95% and OR = 10%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	1.911	1.854	1.822	1.821
	MSE	8.026	7.726	7.586	7.587
	RMSE	2.833	2.779	2.754	2.754
	#SVs (#PCs)	10	10 (8)	3	4
20	**MAE**	2.191	2.186	2.071	2.066
	MSE	9.519	9.552	8.969	8.974
	RMSE	3.085	3.09	2.994	2.995
	#SVs (#PCs)	20	20 (14)	4	7
30	**MAE**	2.518	2.474	2.296	2.295
	MSE	11.632	11.631	10.619	10.586
	RMSE	3.41	3.41	3.258	3.253
	#SVs (#PCs)	30	30 (19)	4	9
40	**MAE**	2.968	2.779	2.598	2.606
	MSE	15.3	14.835	13.761	13.78
	RMSE	3.911	3.851	3.709	3.712
	#SVs (#PCs)	40	40 (22)	5	12
60	**MAE**	4.839	3.528	3.348	3.45
	MSE	39.286	24.971	23.739	24.41
	RMSE	6.267	4.997	4.872	4.94
	#SVs (#PCs)	60	60 (25)	7	17

Table 9: Simulation results when N=150, ρ_x = 95% and OR = 15%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.4	2.219	2.182	2.197
	MSE	10.593	9.454	9.172	9.262
	RMSE	3.254	3.074	3.028	3.043
	#SVs (#PCs)	10	10 (8)	3	4
20	**MAE**	2.61	2.562	2.414	2.431
	MSE	12.252	11.975	10.995	11.102
	RMSE	3.5	3.46	3.315	3.331
	#SVs (#PCs)	20	20 (14)	3	6
30	**MAE**	2.932	2.871	2.640	2.654
	MSE	14.894	14.429	12.802	12.892
	RMSE	3.859	3.798	3.577	3.59
	#SVs (#PCs)	30	30 (19)	4	9
40	**MAE**	3.429	3.142	2.905	2.949
	MSE	19.688	17.746	15.931	16.233
	RMSE	4.437	4.212	3.991	4.029
	#SVs (#PCs)	40	40 (22)	5	11
60	**MAE**	5.477	3.815	3.596	3.683
	MSE	50.206	27.778	25.934	26.585
	RMSE	7.085	5.27	5.092	5.156
	#SVs (#PCs)	60	60(25)	7	16

Table 10: Simulation results when N=300, ρ_x = 90% and OR = 10%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	1.874	1.927	1.876	1.870
	MSE	8.347	8.584	8.344	8.315
	RMSE	2.889	2.929	2.888	2.883
	#SVs (#PCs)	10	10 (9)	3	5
20	**MAE**	2.010	2.138	2.045	2.026
	MSE	8.850	9.613	9.137	9.042
	RMSE	2.974	3.1	3.022	3.006
	#SVs (#PCs)	20	20 (17)	4	9
70	**MAE**	2.901	3.910	3.258	3.239
	MSE	15.045	28.705	21.150	21.061
	RMSE	3.878	5.357	4.598	4.589
	#SVs (#PCs)	70	70 (47)	9	28

Table 11: Simulation results when N=300, ρ_x = 90% and OR = 15%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.372	2.417	2.361	2.355
	MSE	11.233	11.522	11.144	11.11
	RMSE	3.351	3.394	3.338	3.333
	#SVs (#PCs)	10	10 (9)	3	5
20	**MAE**	2.521	2.639	2.511	2.505
	MSE	12.139	12.963	12.061	12.031
	RMSE	3.484	3.6	3.472	3.468
	#SVs (#PCs)	20	20(17)	4	8
70	**MAE**	3.427	4.372	3.574	3.573
	MSE	19.822	34.096	23.824	23.9
	RMSE	4.452	5.839	4.88	4.888
	#SVs (#PCs)	70	70 (48)	9	26

Table 12: Simulation results when N=300, ρ_x= 95% and OR = 10%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	1.869	1.887	1.852	1.849
	MSE	8.241	8.345	8.181	8.163
	RMSE	2.87	2.888	2.86	2.857
	#SVs (#PCs)	10	10 (8)	3	5
20	**MAE**	2.026	2.096	2.024	2.015
	MSE	9.056	9.557	9.186	9.156
	RMSE	3.009	3.091	3.03	3.025
	#SVs (#PCs)	20	20 (15)	4	8
70	**MAE**	2.87	3.334	3.17	3.167
	MSE	14.621	22.135	20.677	20.585
	RMSE	3.823	4.704	4.547	4.537
	#SVs (#PCs)	70	70 (37)	9	23

Table 13: Simulation results when N=300, ρ_x= 95% and OR = 15%

K	Criteria	OLS	PCANN	RFNN	LASSONN
10	**MAE**	2.362	2.378	2.341	2.337
	MSE	11.087	11.187	10.925	10.920
	RMSE	3.329	3.344	3.305	3.304
	#SVs (#PCs)	10	10 (8)	3	4
20	**MAE**	2.528	2.597	2.507	2.502
	MSE	12.361	12.887	12.231	12.240
	RMSE	3.515	3.589	3.497	3.498
	#SVs (#PCs)	20	20 (15)	4	7
70	**MAE**	3.451	3.732	3.531	3.516
	MSE	20.133	25.685	23.783	23.546
	RMSE	4.486	5.068	4.876	4.852
	#SVs (#PCs)	70	70 (37)	7	21

We obtain the following results from the simulation: From Table 2, we can concluded that RFNN method is more effective than PCANN and LASSONN methods because it has the least MSE, MAE, and RMSE and selects fewer variables compared with LASSONN. From Table 5, increasing the correlation between independent variables and outliers in the error term with the same sample size, OLS and PCANN selected all variables 10, 20, and 30, and PCANN had a number of PCs 8, 11, and 15. LASSONN selected variables 3, 5, and 8. RFNN selected 2, 3, and 3. As K increases, the error increases. RFNN method is more effective than PCANN and LASSONN methods because it had a minimum MSE, MAE, and RMSE and selected fewer variables than LASSONN. From Table 13, the increase in sample size, the correlation of 0.95 between independent variables, and percentage of outliers 0.15 in error term OLS and PCANN selected all variables 10, 20, and 70, and PCANN had a number of PCs 8, 15, and 37. LASSONN selected variables 4, 7, and 21. RFNN selected 3, 4, and 7. As K increases, the error increases. RFNN method is more powerful than PCANN and LASSONN methods because it had a minimum MSE, MAE, and RMSE and selected fewer variables than LASSONN.

Finally, the simulation results showed that the values of MSE, MAE, and RMSE for all methods increased when the sample size, the correlation between independent variables, and outliers in the error term are increasing. However, the proposed methods (RFNN and LASSONN) had a minimum MSE, MAE, and RMSE compared with the classical methods (OLS and PCANN). And the RFNN has minimum selected variables in all cases as well as minimum MSE, MAE, and RMSE.

5. APPLICATION: AIR QUALITY

In this section, we present the application to air quality dataset and compare the variable selection methods (FFNN and proposed methods LASSONN and RFNN).

The dataset contains 250 instances of hourly averaged responses from an array of five metal oxide chemical sensors embedded in an Air Quality Chemical Multisensory Device. Data were recorded from March 2004 to February 2005. The device was located on the field in a significantly polluted area, at road level, within an Italian city. The dataset comprises one response variable and 12 independent variables [27] (see Table 14).

The aim of our analysis is to underline the independent variables that are most relevant for predicting response variable. Thus, we use the model selection (variable selection) methods.

However, we first analyze the dataset for a better understanding of the data. As presented in Table 15, some correlations between variables are stronger than the other. From Table 15, the correlation coefficients indicate that there are strong relationships (more than 0.8) between some independent variables. Then, we obtained variance inflation factors (VIF) to ensure the existence of multicollinearity, which is not normal for some variables greater than 10. It means that a multicollinearity problem exists between independent variables.

Table 14: Variables description

Variable	Description
$\mathbf{Y}$	True hourly averaged concentration CO in mg/m^3 (reference analyzer)
$\mathbf{X_1}$	PT08.S1 (tin oxide) hourly averaged sensor response (nominally CO targeted)
$\mathbf{X_2}$	True hourly averaged overall Non-Methane HydroCarbons concentration in micro g/m^3 (reference analyzer)
$\mathbf{X_3}$	True hourly averaged Benzene concentration in micro g/m^3 (reference analyzer)
$\mathbf{X_4}$	PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted)
$\mathbf{X_5}$	True hourly averaged NOx concentration in ppb (reference analyzer)
$\mathbf{X_6}$	PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NOx targeted)
$\mathbf{X_7}$	True hourly averaged NO_2 concentration in micro g/m^3 (reference analyzer)
$\mathbf{X_8}$	PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO_2 targeted)
$\mathbf{X_9}$	PT08.S5 (indium oxide) hourly averaged sensor response (nominally O_3 targeted)
$\mathbf{X_{10}}$	Temperature in Â°C
$\mathbf{X_{11}}$	Relative Humidity (%)
$\mathbf{X_{12}}$	AH Absolute Humidity

In this section, we compare three variable selection methods: PCA with FFNN and two proposed methods (LASSO with FFNN and RF with FFNN). Table 16 shows the goodness of fit measures for variable selection methods.

From Table 16, we conclude that OLS and PCANN consider all independent variables and PCANN had 7 PCs. The proposed methods (RFNN and LASSONN) perform better than the classical methods (OLS and PCANN) with a minimum MSE, MAE, and RMSE. RFNN is better than all methods that exhibited minimum MSE, MAE, and RMSE but LASSONN selected less independent variables compared with all methods.

Table 15: Correlation matrix and variance inflation factor (VIF) values

	$\mathbf{X}_1$	$\mathbf{X}_2$	$\mathbf{X}_3$	$\mathbf{X}_4$	$\mathbf{X}_5$	$\mathbf{X}_6$	$\mathbf{X}_7$	$\mathbf{X}_8$	$\mathbf{X}_9$	$\mathbf{X}_{10}$	$\mathbf{X}_{11}$	$\mathbf{X}_{12}$
$\mathbf{X}_1$	1											
$\mathbf{X}_2$	0.491	1										
$\mathbf{X}_3$	0.938	0.521	1									
$\mathbf{X}_4$	0.953	0.484	0.984	1								
$\mathbf{X}_5$	0.809	0.478	0.802	0.816	1							
$\mathbf{X}_6$	−0.87	−0.30	−0.84	−0.91	−0.73	1						
$\mathbf{X}_7$	0.570	0.280	0.534	0.576	0.879	−0.55	1					
$\mathbf{X}_8$	0.945	0.481	0.979	0.971	0.801	−0.88	0.519	1				
$\mathbf{X}_9$	0.913	0.557	0.896	0.914	0.770	−0.89	0.537	0.900	1			
$\mathbf{X}_{10}$	0.400	0.122	0.415	0.462	0.289	−0.49	0.348	0.368	0.405	1		
$\mathbf{X}_{11}$	−0.38	−0.21	−0.42	−0.46	−0.30	0.403	−0.36	−0.32	−0.38	−0.92	1	
$\mathbf{X}_{12}$	0.120	−0.23	0.042	0.043	0.012	−0.27	−0.01	0.187	0.158	0.074	0.280	1
VIF	**18.70**	**2.77**	**188.71**	**314.54**	**29.77**	**35.18**	**11.88**	**132.44**	**14.63**	**24.32**	**30.52**	**11.32**

Table 16: Goodness of fit measures for variable selection methods

Criteria	**OLS**	**PCANN**	**RFNN**	**LASSONN**
MAE	26.668	19.214	16.654	17.979
MSE	2594.126	2203.824	1774.062	2178.518
RMSE	50.932	46.944	42.119	46.674
#SVs (#PCs)	12	12 (7)	4	3

6. CONCLUSIONS

We investigated the efficiency of the model selection (variable selection) methods for a higher multicollinearity and outlier effect. The proposed methods include LASSONN and RFNN. We used the Monte Carlo simulation study and application to compare the performance of the proposed and classical methods. We summarized the main results of the simulation study as follows: 1) The proposed estimators have powerful goodness fit than the classical methods OLS and PCANN in all cases. 2) RFNN is better than PCANN and LASSONN and has the least values of MSE, MAE, and RMSE. 3) PCANN and OLS have selected all variables but selected more variables than RFNN in the case of the proposed LASSONN.

We used classical variable selection methods (OLS and PCANN) and proposed methods (LASSONN and RFNN) in the real dataset (air quality). We computed the correlation matrix and obtained a higher correlation between independent variables and a higher correlation between the independent variables. Thus, we had multicollinearity and computed the variance inflation factor. To ensure multicollinearity, we obtained some variables greater than 10. Finally, we applied all methods to the dataset and summarized some results of the application: 1) OLS and PCANN selected all independent variables (K) in the analysis (full model). 2) OLS and PCANN have higher values of MSE, MAE, and RMSE. 3) The proposed methods perform better than classical methods (PCANN and OLS). 4) RFNN method is better than all methods with the least values of MSE, MAE, and RMSE. 5) We recommend using the two proposed methods (RFNN and LASSONN) because they had a less MSE, RMSE, and MAE compared with the classical methods (OLS and PCANN).

In future work, we can study another variable selection method for handling multicollinearity and outliers problems in different regression models. Also, we can study another estimation method for handling multicollinearity and outlier together without selection, such as [28, 29], and extend this estimation to the case of high-dimensional data (when the number of independent variables is greater than the sample size).

CONFLICT OF INTERESTS

The author(s) declare that there is no conflict of interests.

REFERENCES

[1] Y. Wang, Y. Jiang, J. Zhang, Z. Chen, B. Xie, C. Zhao, Robust variable selection based on the random quantile LASSO, Commun. Stat. – Simul. Comput. 51 (2022), 29–39.

[2] M. Mansoor, F. Grimaccia, S. Leva, M. Mussetta, Comparison of echo state network and feed-forward neural networks in electrical load forecasting for demand response programs, Math. Computers Simul. 184 (2021), 282–293.

[3] B. Widrow, M.E. Hoff, Adaptive switching circuits, In: IRE WESCON Conv. Record, Part 4, (1960), pp. 96–104.

[4] M. Rosenblatt, D. Slepian, Nth order Markov chains with every N variables independent, J. Soc. Ind. Appl. Math. 10 (1962), 537–549.

[5] J.L. McClelland, D.E. Rumelhart, The PDP Research Group: Parallel Distributed Processing. In: Psychological and Biological Models, vol. 2, The MIT Press, Cambridge, MA (1986).

[6] B.D. Ripley, Pattern recognition and neural networks, Cambridge University Press, Cambridge, (1996).

[7] Z.G. Che, T.A. Chiang, Z.H. Che, Feed-forward neural networks training: a comparison between genetic algorithm and back-propagation learning algorithm, Int. J. Innov. Comput. Inform. Control, 7 (2011), 5839-5850.

[8] A.H. Sahoolizadeh, B.Z. Heidari, C.H. Dehghani, A new face recognition method using PCA, LDA and neural network, Int. J. Computer Sci. Eng. 2 (2008), 218-223.

[9] M. Mrówczyńska, Analysis of principal components used for modelling changes in glacitectonically disturbed areas, J. Water Land Develop. 39 (2018), 119–123.

[10] J. Qiu, H. Wang, J. Lu, B. Zhang, K.-L. Du, Neural network implementations for PCA and its extensions, ISRN Artif. Intell. 2012 (2012), 847305.

[11] T. Hastie, R. Tibshirani, M. Wainwright, Statistical learning with sparsity: the lasso and generalizations, CRC Press, Boca Raton, 2015.

[12] V. Fonti, E. Belitser, Feature selection using lasso. In: VU Amsterdam Research Paper in Business Analytics, (2017), pp 1–25.

[13] L. Breiman, Random forests, Mach. Learn. 45 (2001), 5-32.

[14] M. Liu, S. Hu, Y. Ge, G.B.M. Heuvelink, Z. Ren, X. Huang, Using multiple linear regression and random forests to identify spatial poverty determinants in rural China, Spatial Statistics. 42 (2021), 100461.

[15] N. Ludwig, S. Feuerriegel, D. Neumann, Putting big data analytics to work: feature selection for forecasting electricity prices using the LASSO and random forests, J. Decision Syst. 24 (2015), 19–36.

[16] R.A. Berk, Classification and regression trees (CART), in: Statistical Learning from a Regression Perspective, Springer New York, New York, NY, 2008: pp. 1–65.

[17] R. Genuer, J.M. Poggi, C. Tuleau-Malot, VSURF: An r package for variable selection using random forests, The R Journal, R Foundation for Statistical Computing, 7 (2015), 19-33.

[18] M.R. Abonazel, A practical guide for creating Monte Carlo simulation studies using R, Int. J. Math. Comput. Sci. 4 (2018), 18-33.

[19] M.R. Abonazel, Handling outliers and missing data in regression models using R: Simulation examples. Acad. J. Appl. Math. Sci. 6 (2020), 187-203.

[20] M.R. Abonazel, R.A. Farghali, Liu-type multinomial logistic estimator, Sankhya B. 81 (2019), 203-225.

[21] M.R. Abonazel, I.M. Taha, Beta ridge regression estimators: simulation and application, Commun. Stat. – Simul. Comput. (2021), 1–13. https://doi.org/10.1080/03610918.2021.1960373.

[22] Z.Y. Algamal, M.R. Abonazel, Developing a Liu ‐ type estimator in beta regression model, Concurrency Comput. Pract. Exp. 34 (2022), e6685.

[23] M.R. Abonazel, Z.Y. Algamal, F.A. Awwad, I.M. Taha, A New Two-Parameter Estimator for Beta Regression Model: Method, Simulation, and Application, Front. Appl. Math. Stat. 7 (2022), 780322.

[24] M.R. Abonazel, O.M. Saber, A comparative study of robust estimators for Poisson regression model with outliers. J. Stat. Appl. Probab. 9(2020), 279-286.

[25] M.R. Abonazel, S.M. El-Sayed, O.M. Saber, Performance of robust count regression estimators in the case of overdispersion, zero inflated, and outliers: simulation study and application to German health data, Commun. Math. Biol. Neurosci. 2021 (2021), Article ID 55.

[26] H. Binder, W. Sauerbrei, P. Royston, Comparison between splines and fractional polynomials for multivariable model building with continuous covariates: a simulation study with continuous response, Stat. Med. 32 (2013), 2262–2277.

[27] S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia, On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario, Sensors Actuators B: Chem. 129 (2008), 750–757.

[28] I. Dawoud, M.R. Abonazel, Robust Dawoud–Kibria estimator for handling multicollinearity and outliers in the linear regression model, J. Stat. Comput. Simul. 91 (2021), 3678–3692.

[29] F.A. Awwad, I. Dawoud, M.R. Abonazel, Development of robust Özkale–Kaçiranlar and Yang–Chang estimators for regression models in the presence of multicollinearity and outliers, Concurrency Comput. Pract. Exp. (2021), e6779. https://doi.org/10.1002/cpe.6779.