Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://journal.nsps.org.ng/index.php/jnsps

J. Nig. Soc. Phys. Sci. 5 (2023) 1708

Journal of the Nigerian Society of Physical Sciences

Robust hybrid algorithms for regularization and variable selection in QSAR studies

Christian N. Nwaeme[a], Adewale F. Lukman[b,c,*]

[a]*African Institute for Mathematical Sciences, Mbour-Thies and BP. 1418, Senegal.*
[b]*University of Medical Sciences, Ondo State, PMB 536, Nigeria.*
[c]*University of North Dakota, Grand Forks, ND, USA*

Abstract

This study introduces a robust hybrid sparse learning approach for regularization and variable selection. This approach comprises two distinct steps. In the initial step, we segment the original dataset into separate training and test sets and standardize the training data using its mean and standard deviation. We then employ either the LASSO or sparse LTS algorithm to analyze the training set, facilitating the selection of variables with non-zero coefficients as essential features for the new dataset. Secondly, the new dataset is divided into training and test sets. The training set is further divided into k folds and evaluated using a combination of Random Forest, Ridge, Lasso, and Support Vector Regression machine learning algorithms. We introduce novel hybrid methods and juxtapose their performance against existing techniques. To validate the efficacy of our proposed methods, we conduct a comprehensive simulation study and apply them to a real-life QSAR analysis. The findings unequivocally demonstrate the superior performance of our proposed estimator, with particular distinction accorded to SLTS+LASSO. In summary, the two-step robust hybrid sparse learning approach offers an effective regularization and variable selection applicable to a wide spectrum of real-world problems.

DOI:10.46481/jnsps.2023.1708

Keywords: High dimension, QSAR, Multicollinearity, Regularization, outliers, Sparse least trimmed squares, Random forest, Support vector regression

Article History :
Received: 10 August 2023
Received in revised form: 10 November 2023
Accepted for publication: 16 November 2023
Published: 28 November 2023

Communicated by: B. J. Falaye

1. Introduction

The Quantitative Structure-Activity Relationship (QSAR) dates back to the nineteenth century and has since been employed in different fields for risk assessment, drug discovery, toxicity prediction, and regulatory decisions. QSAR models adopt supervised machine learning models, such as regression and classification and seek to predict a response variable, such as the biological activity of a chemical, with a set of predictors, such as the physicochemical properties of synthetic chemical drugs or theoretical molecular descriptors of chemicals [1–4]. Furthermore, mathematical and statistical QSAR models have proven to be the best computational methods in drug discovery, saving time and resources. As a result, QSAR research is becoming more prominent in finding new drugs [5]. QSAR models have also been used to deduce the activity of a chemical compound from its structural features.

*Corresponding author Tel.:+2347032328232
Email address: adewale.lukman@und.edu (Adewale F. Lukman)

Numerous studies exist on QSAR modelling. For instance, it has become an essential process in the pharmaceutical industry with certain limitations. QSAR data may include hundreds of thousands of chemicals (descriptors), leading to high-dimensional data. This high-dimensional data has become more common in computational chemistry studies where more molecules exist than molecular descriptors [6]. As a result, the significance of quantitative structure-activity relationship (QSAR) studies has increased in this field, concerning the structural characteristics of a group of chemical substances, with the goal of QSAR being to simulate various biological processes [2].

The commonly used fingerprints in QSAR modelling often result in correlated features and sparsity, with some values being zero. These issues make it challenging for QSAR-based models to achieve accurate predictions. The least squares method is not appropriate for QSAR models since X^TX matrix becomes non-invertible in high-dimensional data [7]. Therefore, more stable predictions in QSAR modelling are often achieved using machine learning models such as Bayesian neural networks and others [8, 9].

In high-dimensional modelling, an efficient dimension reduction method is essential to provide parsimonious models with strong prediction ability and interpretation. The availability of high-dimensional statistics in computational chemistry is increasing, but the selection of molecular descriptors remains a critical challenge in QSAR investigations. The significant variation of QSAR models generally leads to poor prediction performance. Therefore, it is necessary to improve prediction accuracy by selecting only the most critical molecular predictors. Other factors, such as the optimization of the chemical shape, the modelling technique, the risk of getting stuck in local minima, redundancy, and over-fitting, also greatly influence a QSAR model's ability to make suitable predictions.

Over the past decade, there has been an increased focus on big data as researchers seek to address critical issues with QSAR models such as redundancy, over-fitting, and being stuck in local minima [10]. Since 2015, deep learning architectures have gained preference over shallow learning models. These architectures have become popular as computational drug design tools because they can detect complex statistical patterns among the vast number of descriptors extracted from various compounds. Deep learning architectures used in QSAR applications include Artificial Neural Networks (ANN), Convolutional Neural Networks, Recurrent Neural Networks, and Support Vector Machines (SVM), which utilize multiple levels of linear and nonlinear techniques.

To increase prediction accuracy and address computational issues with high-dimensional data, the objective function of the regression can be modified by adding a penalty term to the regression coefficients. However, this strategy results in a trade-off between reduced variance and increased bias. Therefore, traditional statistical topics such as regularization and variable selection have received significant attention. Ridge regression [11] is an example of a regularization technique that reduces the residual sum of squares while maintaining a predetermined range for the L_2 norm of the coefficients. Ridge regression balances bias and variance to achieve optimal prediction performance but always includes all predictors in the model, failing to yield a parsimonious model. In contrast, [12] highlights that although best subset selection creates a sparse model, it is valuable due to its inherent discreteness.

Tibshirani proposed a promising approach called the lasso [13]. The lasso is a penalized least squares method that penalizes the regression coefficients by applying an L_1 penalty. The lasso performs continuous shrinkage and automatic variable selection simultaneously due to the properties of the L_1 penalty. When the lasso, ridge, and bridge regressions [14] were compared for prediction performance, Tibshirani and Fu found that none of them consistently outperformed the others [13, 15]. However, given the growing importance of variable selection in contemporary data analysis, the lasso is considerably more appealing because of its ability to produce a sparse representation. Despite its limitations, the lasso has been effective in many situations. Its limitations include:

(a) The lasso may select only one variable out of a set of highly correlated variables, making the selected variable somewhat arbitrary.

(b) When the number of predictors is much larger than the sample size, the lasso may select too many variables, which can lead to overfitting.

Zou and Hastie developed the elastic-net approach by combining L_2 and L_1 penalties on the regression coefficients [16]. Elastic net aims to group together strongly correlated variables, resulting in their inclusion or exclusion from the model. It performs best when there are high absolute values of pairwise correlations among the groups. In the case of correlated data, elastic net often outperforms lasso in terms of prediction error. However, since it does not reveal the underlying group structure in its solution, the elastic net may not perform well when the groups change and have only modest pairwise correlations. In QSAR studies, LASSO and Elastic-net have yielded fascinating results in terms of variable selection, estimation, and prediction [17–21].

Penalized regression techniques, such as Lasso, elastic nets, and others, are known to be sensitive to outliers or unusual observations, which are common problems in QSAR modelling [21]. It is essential to understand that these methods can become entirely untrustworthy with just one anomaly, which can negatively impact the prediction outcome. To address outliers in low-dimensional data, robust alternatives such as the Least Absolute Deviation (LAD) and Least Trimmed Squares (LTS) estimators are recommended [22, 23]. These estimators are effective in handling outliers in the y-direction but do not perform variable selection.

To address both outlier detection and variable selection, Wang et al. [23] developed the LAD-LASSO, which adds an L_1 norm to the LAD regression for robust prediction and variable selection. More recently, the Sparse Least Trimmed Squares regression was proposed by adding an L_1 penalty to the LTS regression, which combines outlier detection and variable selection in a robust way [24].

Deep learning models are powerful algorithms that have shown great promise in various research fields, including the pharmaceutical industry, for addressing regression and classification problems. However, deep learning algorithms also have some drawbacks, such as high computational time, over-fitting, and a requirement for a large amount of data and memory space [25].

This study will not focus on deep learning models, such as artificial neural networks, due to the nature of the adopted data and the need for faster computation. Instead, a variety of techniques have been developed to mitigate the core limitations of deep learning models, such as long-running times and high processing demands. Random Forest (RF), Bagging, and Support Vector Regression are some of the extensively utilized variable selection algorithms in computational drug design, as they offer criteria for obtaining the most crucial descriptors. Additionally, algorithms such as multivariate adaptive regression splines, Relief, and Boruta have also been used [26].

In recent studies, hybrid algorithms have been adopted to enhance prediction. For instance, Motamedi et al. [25] proposed LASSO-RF, which selects molecular descriptors using LASSO and predicts using random forest. Liu and Qin [27] developed a two-step approach by applying Lasso to the trained data and then performing regularization on the selected features using Elastic-net and Ridge regression. They concluded that the two-step algorithm produced more optimal models than LASSO alone. More recently, in a QSAR study, molecular descriptors were selected using LAD-LASSO and biological activity was predicted using artificial neural networks (ANN) [28].

This study aims to develop a new hybrid approach for selection and prediction. We selected molecular descriptors from the QSAR data using LASSO and Sparse LTS and predicted biological activity using random forest, support vector, and Ridge regressions. Additionally, we conducted a simulation study with high-dimensional data and contaminated the data with outliers in the response variable. Finally, we compared the performance of the algorithms using the root mean squared error, mean absolute deviation, and median absolute error. Section 2 provides an exhaustive exploration of established methodologies, while Section 3 introduces our novel approach. Section 4 focuses on the simulation studies and real-life analyses, and Section 5 gives the concluding remarks.

2. Literature Review: Concepts and Mathematical Model

In this section, we will briefly review two important concepts in regression analysis: multicollinearity and outliers. We will then delve into a detailed discussion of several popular estimators that were introduced in the previous section. These estimators include **ridge** regression, lasso regression, Random Forest, support vector regression, and sparse LTS. By the end of this section, you will have a comprehensive understanding of these estimators and their applications in regression analysis.

2.1. Regularization

To combat over-fitting, regularization is a technique that reduces generalization error while minimally affecting the training error. Overfitting often occurs when overly complex models are used to fit the training data, while underfitting happens when the model is too simple. Therefore, it is crucial to select an appropriate level of complexity for the model. However, this task is challenging as it cannot be determined solely from the provided training data. Thus, selecting the right model complexity for training requires careful consideration.

2.1.1. Different Types of regularisation techniques

There are different types of regularization techniques that affect the model very differently. Here are some of those;

2.1.2. Ridge Regression (L_2 Regularization)

Ridge regression addresses some of the limitations of linear regression. While linear regression can produce estimates with large magnitudes and high variance, ridge regression adds a constraint to the ordinary least squares (OLS) method to shrink the regression coefficients towards zero. This regularization reduces the variance of the estimates and the prediction error, without overly compromising bias. Specifically, ridge regression minimizes a penalized residual sum of squares (RSS), similar to OLS, but with a penalty term that depends on a tuning parameter, which controls the amount of shrinkage. As a result, the ridge regression estimates are biased but have less variance than the OLS estimates [11].

2.1.3. Mathematical Formulation

The coefficients for ridge regression are obtained by minimizing the residual sum of squares (RSS), subject to an additional constraint,

$$\hat{\delta}_{ridge} = \underset{\delta}{argmin} \sum_{i=1}^{N}(y_i - (\delta_0 + \sum_{i=1}^{N} \delta_i x_i))^2 + \alpha \sum_{i=1}^{N} \delta_i^2. \quad (1)$$

When α is the tuning parameter, which we explain in the following section, and $\sum_{i=1}^{N} \delta_i^2$ is the square of the vector δ norm, term $\alpha \sum_{i=1}^{N} \delta_i^2$ is referred to as a "shrinkage penalty." The L_2 norm is what is referred to as $\|\delta\|_2 = \sum_{i=1}^{N} \delta_i^2$. In other words, the ridge coefficients δ_{ridge} minimize a penalized RSS, which we refer to as an L_2 penalty as the penalty is determined by the L_2 norm [29].

To determine the δ_{ridge} parameters, we consider two critical assumptions of ridge regression. First, the intercept is not subjected to a penalty. Second, normalizing the predictors is essential. Unlike ordinary least squares (OLS) estimates, where scaling the coefficients inversely affects them proportionally, ridge coefficients can be significantly altered by multiplying them by a constant. Therefore, we standardize each predictor value by subtracting the mean and dividing the result by the standard deviation of the corresponding value in the training set, as recommended by Friedman et al. [30].

Next, we will discuss ridge regression using matrix algebra representation. The input consists of a centered n by p matrix (X) and a centered n-dimensional vector (Y). Both X and

Y have zero means, and X has a unity variance. We standardize the inputs before transforming the minimization problem in Equation (2.1.1) into an L_2 penalized problem using matrices.

$$\hat{\delta}_{ridge} = \underset{\delta}{argmin}\frac{1}{2}\|y - X\delta\|^2 + \alpha\|\delta^2\|. \tag{2}$$

The ridge coefficients transform into

$$\hat{\delta}_{ridge} = (X^T X + \alpha I)^{-1} X^T y, \tag{3}$$

such that I is an identity matrix.

2.1.4. LASSO (L_1 Regularization)

The Lasso regularization method, short for "Least Absolute Shrinkage and Selection Operator," is a technique that extends ridge regression by introducing two key features. In contrast to ridge regression, Lasso not only shrinks coefficients but can also reduce some of them to exactly zero, which is known as "sparsity." Another distinctive feature of Lasso is that it can identify and prioritize important variables by reducing some specific coefficients, a property known as "variable selection." Together, these properties allow Lasso to perform both regularization and variable selection simultaneously [13].

2.1.5. Mathematical Formulation

This section focuses on the lasso algorithm. The Lagrangian formulation of the lasso is as follows:

$$\hat{\delta}_{lasso} = \underset{\delta}{argmin} \sum_{i=1}^{N}[y_i - (\hat{\delta}_0 + \sum_{i=1}^{N}\hat{\delta}_i x_i)]^2 + \alpha \sum_{i=1}^{N}|\delta_i|, \tag{4}$$

where the shrinkage parameter is denoted by α. The shrinkage penalty $\sum_{i=1}^{N}|\delta_j|$, is actually provided by the vector's L_1 norm, δ defined as $\|\delta\|_1 = \sum|\delta_j|$. The predictors are normalized and the intercept, which is calculated as $\delta_0 = \bar{y}$, is not included in the model. Therefore, the main difference between lasso and ridge regression is that lasso uses an L_1 penalty whereas ridge regression uses an L_2 penalty. The difference between an L_1 penalty and an L_2 penalty is that the L_1 penalty has the effect of shrinking some coefficients exactly to zero [29]. The least angle regression (LAR) approach, for instance, is one of several strategies that can be used to solve this quadratic programming problem [13].

We investigate the matrix algebra formulation to elucidate the properties of the lasso estimations. In this instance, the solutions to an L_1 penalized issue are the lasso coefficients,

$$\hat{\delta}_{lasso} = \underset{\delta}{argmin}\frac{1}{2}\|y - X\delta\|^2 + \alpha\|\delta\|_1. \tag{5}$$

Contrary to ridge regression, the coefficients $\hat{\delta}_{lasso}$ lack a closed form since the L_1 penalty imposes an absolute value constraint that cannot be distinguished. Due to the non-smooth nature of the constraint, the solutions to the lasso issue are nonlinear in y_j [29].

2.2. Sparse Least Trimmed Squares (SLTS) models

Sparse Least Trimmed Squares (LTS) models are a modification of the LTS regression method, which is a robust regression technique that is effective in the presence of outliers. The goal of Sparse LTS is to identify a subset of the data that can produce the lowest sum of squared residuals, while at the same time, enforcing sparsity in the model. By introducing sparsity constraints, Sparse LTS can help identify the most relevant variables that contribute to the regression, leading to a more interpretable and efficient model. This approach is particularly useful when dealing with high-dimensional data, where many of the variables may not be relevant to the regression task at hand. Sparse LTS is widely used in various fields, including finance, biology, and engineering.

2.2.1. Mathematical Formulation

Let $\mathbf{x_i} = (x_1, x_2, \ldots, x_n)$ be the d-dimensional observations on the predictor variables where $i \in [1, n]$, and $\mathbf{y_i} = (y_1, y_2, \ldots, y_n)$ be the observations on the response, respectively. The linear regression model is examined

$$y_i = x_i'\delta + \varepsilon_i, \tag{6}$$

using a regression parameter $\delta = (\delta_1, \ldots, \delta_p)'$ with the error terms $\varepsilon_i \sim N(0, \gamma)$.

Applied statistics often face the challenge of outliers in the data, which can significantly impact the performance of penalized estimators like the lasso, ridge, and elastic net, which utilize the least squares loss function. To address this issue, several dependable alternatives have been proposed in the literature. One popular approach is to use penalized M-estimators, such as those proposed by Rosset and Zhu [31], Wang et al. [23], and Li et al. [32], which are designed to be resilient against outliers in the response variable but not necessarily in the predictor space.

However, to achieve robustness against outliers in the predictor space, one can regularize appropriate robust regression techniques, such as the least trimmed squares, as proposed by Rousseeuw and Van [33]. These techniques can effectively address the issue of outliers in both the response and predictor variables. Therefore, it is essential to carefully select an appropriate approach based on the nature of the data and the research question at hand to ensure accurate and reliable statistical inference.

The Least Trimmed Square (LTS) regression is a widely recognized and extensively studied method for dealing with outliers in regression analysis [22]. It is a commonly used robust regression model due to its simple specification and efficient computation. The squared vector of residuals can be denoted as $r^2(\delta) = (r_1^2, r_2^2, r_3^2, \ldots, r_n^2)'$, where $r_i^2 = (y_i - x_i'\delta)^2$. The LTS model is defined as a regression model that minimizes the sum of squared residuals, subject to a constraint on the proportion of data points to be included in the regression. Specifically, LTS regression minimizes the sum of squared residuals for a subset of the data that contains a specified proportion of the observations that have the smallest squared residuals. This subset is determined by minimizing the value of δ that satisfies the

constraint on the proportion of data points. The LTS model provides a balance between robustness and efficiency, making it a useful tool in many applications where outliers are of concern.

$$\hat{\delta}_{LTS} = \underset{\delta}{argmin} \sum_{i=1}^{h} (r^2(\delta))_{i:n}. \tag{7}$$

The squared residuals are typically ordered as $(r^2(\delta))_{1:n} \leq \cdots \leq (r^2(\delta))_{n:n}$, with $h \leq n$. The goal of LTS regression is to identify a subset of h observations with the least squares that results in the lowest sum of squared residuals. The choice of h determines the range of desirable observations within the statistics and can be used to model the subset length. Although LTS regression is a robust method, it may not produce estimates from sparse models. When $h \leq p$, it is impossible to compute the LTS model. To address this issue, an L_1 penalty can be applied to the objective function 7, with a penalty parameter α, resulting in a sparse and regularized LTS model, often referred to as Sparse LTS.

$$\hat{\delta}_{sparseLTS} = \underset{\delta}{argmin} \sum_{i=1}^{h} (r^2(\delta))_{i:n} + \alpha h \sum_{j=1}^{p} |\delta_j|. \tag{8}$$

The high breakdown point of sparse LTS has been demonstrated by Alfons et al. [24]. It is resistant to leverage points and vertical outliers. In addition to being very durable and functioning like LTS, sparse LTS

- increases prediction accuracy with the aid of decreasing variance whilst the pattern size is small in the assessment of the size.
- advanced interpretability is assured by the simultaneous model choice, and
- overcomes the computational problems of traditional robust regression strategies when managing high-dimensional data.

THEOREMS

1. The SLTS Model's Breakdown point: The replacement finite-sample breakdown point is the most commonly used measure of an estimator's robustness [22]. Where $N = (X, y)$ is the represented sample. The breakdown point for regression $\hat{\delta}$ is defined as

$$\varepsilon^*(\hat{\delta}; N) = min\left\{\frac{m}{n} : \underset{\tilde{N}}{Sup} \|\hat{\delta}(\tilde{N})\|_2 = \infty\right\}, \tag{9}$$

   $\tilde{N}$ is the corrupted data obtained from N by substituting arbitrary values form of the original n data points.

   Proof. By Alfons et al. [24].

2. A convex and symmetric loss function, $\varphi(x)$, with $\varphi(x) > 0$ and $\varphi(0) = 0$ for $x = 0$, is defined as $\varphi(x) := (\varphi(x_1), \ldots, \varphi(x_n))$. Consider the regression model with subset size $h > n$;

$$\hat{\delta} = \underset{\delta}{argmin} \sum_{i=1}^{h} (\varphi(y - X\delta))_{i:n} + n\alpha \sum_{j=1}^{p} |\delta_j|, \tag{10}$$

   given that $(\varphi(y - X\delta))_{1:n} \leq, \ldots, \leq (\varphi(y - X\delta))_{n:n}$ are the information order of the regression loss. The estimator $\hat{\delta}$ breakdown factor is then given by

$$\varepsilon^*(\hat{\delta}; N) = \frac{n - h + 1}{n}. \tag{11}$$

For any loss function φ that meets the assumptions, the breakdown point remains the same. In the SLTS estimator $\hat{\delta}_{SLTS}$ with subset length $h \leq n$, in which $\varphi(x) = x^2$, the breakdown point remains $\frac{n-h+1}{n}$. The breakdown point increases as h decreases. It is possible to have a breakdown point greater than 50% by taking h small enough [24].

2.3. Random Forest

Breiman's ideas in machine learning were significantly influenced by a number of pioneering methods including the early random subspace method of Ho [34], the geometric variable selection work of Amit and Geman [35], and the random split selection approach of Dietterich [36]. These techniques have since paved the way for more advanced methods such as boosting [37] and support vector machines, but none have been able to match the performance and versatility of random forests (RF). Random forests have proven to be highly effective in handling a large number of input variables without over-fitting, while also being simple and quick to implement, and producing highly accurate predictions. They are widely regarded as one of the most precise and reliable all-purpose learning methods available. For readers seeking a deeper understanding of random forests and related methods, the survey conducted by Genuer et al. [38] can provide valuable insights and a solid foundation for comprehension.

Random Forest (RF) is a powerful machine-learning technique that combines the results of multiple decision trees to produce robust and accurate predictions. In an RF model, each decision tree is built using a bootstrap sample of the training data and only a random subset of the available input features. Predictions are made by aggregating the individual tree predictions through either majority voting or averaging, depending on the task at hand.

In regression, the final predicted value is the average of the predicted values of each tree. The RF algorithm grows each tree using the entire training set as a bootstrap sample and uses an out-of-bag (OOB) set to estimate the model's generalization performance. The CART algorithm is used to choose the best split at each node among a random subset of the available input features. RF models do not perform pruning and have no tuning parameters.

The predictive ability of an RF model is evaluated using the Φ^2_{abs} determination coefficient on an external validation set.

RF models also offer useful features such as out-of-bag predictions for error estimation, natural closeness estimation of two substances, and variable importance metrics based on the difference in OOB error rate when a descriptor is permuted.

In conclusion, RF is a robust and effective technique for QSAR modelling, especially when the number of available input features is high. The ensemble nature of the method, along with its ability to handle noisy and correlated data, makes it a popular choice for many applications.

2.4. Support Vector Regression

Vladimir Vapnik is recognized as the pioneer of Support Vector Machines (SVMs), which are a type of supervised learning machine that can generalize well on a variety of learning patterns by using the structural risk minimization inductive principle. The structural risk minimization (SRM) approach aims to minimize both the empirical risk and the VC (Vapnik-Chervonenkis) dimension simultaneously. The theory was developed by Vapnik and his colleagues based on a separable bipartition problem. SVMs can recognize minor patterns in large volumes of data, making them an effective method for image reduction [39].

SVMs are divided into two categories: support vector classification (SVC) and support vector regression (SVR). SVMs are feature space-based learning techniques that operate in high dimensions, generating prediction functions based on a subset of support vectors. The SVM model for classification depends only on a subset of the training data since the cost function for constructing the model disregards any training points that are outside of the margin. Similarly, the SVR model depends only on a subset of the training data because the cost function for building the model rejects any training data that is close to or within a threshold, ε, of the model prediction. SVR uses kernels, sparse solutions, and VC control over the margin and number of support vectors, which is similar to classification.

Support Vector Regression (SVR) is the most common use of SVMs. The basic concepts of support vector machines for regression and function estimation were outlined by Vapnik et al., [40] and Smola et al. [41]. Furthermore, SVMs offer several training techniques for handling large datasets and quadratic or convex programming. The classic SV algorithm has been modified and extended with regularization and capacity control from an SV perspective. SVR is a supervised learning technique that uses a symmetrical loss function to penalize both high and low misestimates equally. To decrease the absolute values of errors, Vapnik's ε-insensitive method forms a flexible tube with a short radius symmetrically around the estimated function.

2.4.1. Mathematical Formulation

The input pattern space $\mathbf{R}^N$ is used to represent the training data, which have been taken as $\{(x_i, y_i), 1 \ldots n\} \subset \mathbf{R}^N \times \mathbf{R}$. The goal of ε-SV regression is to find a function $f(x)$ that is as flat as possible and that deviates from the objectives y_i for all of the training data by at most ε [39]. The description of the linear regression function f case is as follows:

$$f(x) = K^T \phi(x) + b. \tag{12}$$

Here, (x) converts the input x to a vector in f, and K is a vector in f. By resolving an optimization issue, the K and b in equation 12 are produced:

$$\min_{K,b} W = \frac{1}{2} K^T K + C \sum_{i=1}^{n} (\varphi_i + \varphi_i^*) \tag{13}$$

$$y_i - (K^T \phi(x) + b) \leq \varphi_i + \varphi_i^* \tag{14}$$

$$(K^T \phi(x) + b) - y_i \leq \varphi_i + \varphi_i^*$$
$$\varphi_i, \varphi_i^* \geq 0, i = 1 \ldots n.$$

When data points' y values depart from $f(x)$ by more than ε, the optimization criterion penalizes those data points. φ_i and φ_i^*, which stand for the size of the excess deviation for positive and negative deviations, respectively, are the slack variables [42].

We can express the equivalent Lagrangian by applying Lagrange multipliers σ, σ^*, ν, and ν^*.

$$L_W = \frac{1}{2} K^T K + C \sum_{i=1}^{n} (\varphi_i + \varphi_i^*) - \sum_{i=1}^{n} (\nu \varphi_i + \nu^* \varphi_i^*) - \sum_{i=1}^{n} \sigma_i (\varepsilon + \varphi_i + y_i$$

$$- K^T \phi(x_i) - b) - \sum_{i=1}^{n} \sigma_i^* (\varepsilon + \varphi_i - y_i + K^T \phi(x_i) + b), \tag{15}$$

such that $\sigma, \sigma^*, \nu, \nu^* \geq 0, i = 1 \ldots n$.

As a result, the dual optimization problem arises:

$$\min_{\sigma,\sigma^*} D_o = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} R_{ij} (\sigma_i - \sigma_i^*)(\sigma_j - \sigma_j^*) + \varepsilon \sum_{i=1}^{n} (\sigma_i + \sigma_i^*)$$

$$- \sum_{i=1}^{n} y_i (\sigma_i - \sigma_i^*), \tag{16}$$

with

$$0 \leq \sigma, \sigma^* \leq C, i = 1 \ldots n, \sum_{i=1}^{n} (\sigma_i - \sigma_i^*) = 0,$$

where $R_{ij} = \phi(x_i)^T \phi(x_j) = Q(x_i, x_j)$. $Q(x_i, x_j)$ is a kernel function [41]. Using the answer to equation 16, the regression function 12 can be expressed as

$$f(x) = \sum_{i=1}^{n} (\sigma_i - \sigma_i^*) Q(x_i, x_j) + b. \tag{17}$$

The following is how the equation's Lagrange formulation, 16, is portrayed:

$$L_{D_o} = \frac{1}{2}\sum_{i=1}^{n}(\sigma_i - \sigma_i^*)(\sigma_j - \sigma_j^*) + \varepsilon\sum_{i=1}^{n}(\sigma_i + \sigma_i^*) - \sum_{i=1}^{n} y_i(\sigma_i - \sigma_i^*)$$

$$-\sum_{i=1}^{n}(\xi_i\sigma_i + \xi_i^*\sigma_i^*) - \sum_{i=1}^{n}[r_i(\sigma_i - C) - r_i^*(\sigma_i^* - C)] + \lambda\sum_{i=1}^{n}(\sigma_i - \sigma_i^*). \tag{18}$$

The Lagrange multipliers are $\xi_i^{(*)}, r_i^{(*)}$ and λ. The Karush-Kuhn-Tucker (KKT) conditions are obtained by optimizing this Lagrangian,

$$\frac{\partial L_{D_o}}{\partial \sigma_i} = \sum_{j=1}^{n} R_{ij}(\sigma_i - \sigma_i^*) + \varepsilon - y_i + \lambda - \xi_i + r_i = 0,$$

$$\frac{\partial L_{D_o}}{\partial \sigma_i} = -\sum_{j=1}^{n} R_{ij}(\sigma_i - \sigma_i^*) + \varepsilon + y_i - \lambda - \xi_i^* + r_i^* = 0, \tag{19}$$

$$\xi_i^{(*)} \geq 0, \xi_i^{(*)}\sigma_i^{(*)} = 0,$$
$$r_i^{(*)} \geq 0, r_i^{(*)}(\sigma_i^{(*)} - C) = 0.$$

At optimality, λ in equation 19 equals b in equations 12 and 17, [43].

Only one of σ_i and σ_i^* will be nonzero, according to the KKT condition 19, and both of them can be nonnegative. Thus, the following is how a coefficient difference, μ_i, might be written:

$$\mu_i = \sigma_i - \sigma_i^*, \tag{20}$$

and μ_i determines σ_i and σ_i^* For the *ith* sample x_i, define a margin function $p(x_i)$ as follows;

$$p(x_i) = f(x_i) - y_i = \sum_{j=1}^{n} R_{ij}\mu_j - y_i + b. \tag{21}$$

Equations 19, 20, and 21 when combined give us,

$$\begin{cases} p(x_i) \geqslant \varepsilon, & \mu_i = -C \\ p(x_i) = \varepsilon, & -C < \mu_i < 0 \\ -\varepsilon \leqslant p(x_i) \leqslant \varepsilon, & \mu_i = 0 \\ p(x_i) = \varepsilon, & 0 < \mu_i < C \\ p(x_i) \leqslant -\varepsilon, & \mu_i = C. \end{cases} \tag{22}$$

Equation 22 compares the three conditions in support vector classification which has five conditions, but just like those conditions, the samples in training set T can be classified into them using three subsets, as they can in equation 22, [44]. And, two of the subsets (E_{sv} and M_{sv}), depending on the direction of the error $f(x_i) - y_i$, are each composed of two distinct components.

The E_{sv} Set: Error support vectors: $E_{sv} = \{i : |\mu_i| = C\}$; The M_{sv} Set: Margin support vectors: $M_{sv} = \{i : 0 < |\mu_i| < C\}$; and The R_s Set: Remaining samples: $R_s = \{i : \mu_i = 0\}$.

3. Methodology

Quantitative Structure-Activity Relationship (QSAR) prediction studies aim to discover new drug-like molecules that can be used as lead compounds. This is achieved by selecting appropriate molecular descriptors and using feature-selection algorithms to predict the biological activities of designed compounds. With the rise of Big Data, there has been increased interest in the use of deep learning models, and studies have shown the effectiveness of a robust hybrid algorithm proposed by Liu et al. [45, 46] .

This two-step approach involves dividing the original data set into a training and test set, scaling the training data using its mean and standard deviation, and analyzing the training set using the sparse LTS algorithm or Lasso. The variables or molecular descriptors that are shrunk to zero are eliminated, while the variables with non-zero coefficients are selected as features for the new data set. Next, the new data set is divided into a training and test set, and the training set is further divided into k folds. Sets of hyper-parameter values for various machine learning algorithms, such as Random Forest, Ridge, Lasso, and Support Vector Regression, are tuned, and the hyper-parameter with the optimal metric is selected as the final model. Finally, the test metric, such as root mean squared error, is obtained for the final model.

In summary, QSAR prediction studies use various techniques to discover new drug-like molecules, and the robust hybrid algorithm proposed in this study is an effective approach for handling Big Data and predicting the biological activities of designed compounds.

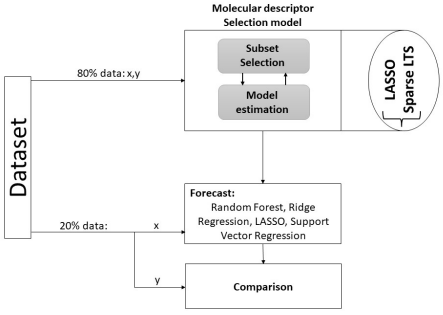

Figure 1. Schematic of the proposed model

4. Simulation Studies & Discussion

In this section, we have designed three distinct experiments to evaluate the performance of the proposed estimators and compared them [16, 47].

The simulation model is based on the linear regression framework:

$$y = X\beta + \sigma\varepsilon, \tag{23}$$

where $\varepsilon \sim N(0,1)$. Here, the response variable y is generated as a linear combination of the predictor variables X and the unknown coefficients β. The random error term ε follows a normal distribution with mean 0 and variance 1, and the predictors X are generated from a multivariate normal distribution with mean 0 and covariance matrix σ. The correlation between the predictors is specified by the parameter ρ.

To evaluate the performance of our proposed methods, we employed a standard approach of dividing each simulated dataset into three distinct parts: a training set, a validation set, and a test set. The data were split in a ratio of 60 percent for training, 20 percent for validation, and 20 percent for testing. We used the training set to fit the models and the validation set to tune the hyperparameters, which were chosen using a grid search. The test set was then used to provide an unbiased evaluation of the final model fit on the training data.

We conducted simulations under three distinct cases, each with varying degrees of dimensionality. In each case, we evaluated the estimators' performance using appropriate accuracy measures and compared their results. This approach allowed us to assess the effectiveness of our proposed method under different scenarios and make reliable conclusions about its performance. As per Alao et al. [48] and Lukman et al. [49–53] approach, the model was deliberately contaminated with outliers using the following equation:

$$y[i] = m * max(y) + y[i]. \tag{24}$$

Here, m represents the magnitude of the outlier, which was set to 10 in this study. 20 percent outlier was introduced to the response variable. The contamination allowed us to assess the robustness of the proposed methods and compare their performances in the presence of outliers.

a. In Case 1, 150 & 400 observations were split between 100 and 300 data sets. We set $\sigma = 5$ & 10 and, $\beta = (5, 10, -5, 10, 3, 10, -3, 10, 0, 40)$. $corr(i, j) = 5^{|i-j|}$ was chosen as the pairwise correlation between x_i and x_j.

b. Case 2 is similar to Case 1 with the exception that $corr(i, j) = 0.2$

c. In Case 3, 50 data sets with 150 observations each were simulated, and $\sigma = 3$ & 5.

The following steps were taken to create the predictors X:

$$\begin{cases} x_i = Z_1 + \varepsilon_i^*, & Z_1 \sim N(0,1) \\ x_i = Z_2 + \varepsilon_i^*, & Z_2 \sim N(0,1) \\ x_i = Z_3 + \varepsilon_i^*, & Z_3 \sim N(0,1) \end{cases} \tag{25}$$

$x_i \sim N(0,1)$, and where x_i is independent identically distributed.

To evaluate the performance of our models, we used various metrics that are commonly used to measure prediction accuracy. Specifically, we calculated the test root mean square error (RMSE), mean absolute deviation (MAD), and median absolute error (MAE) using the following formulas:

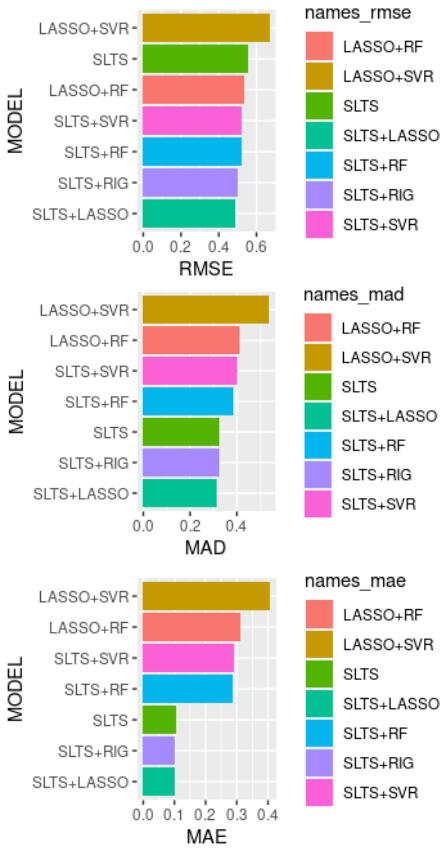

Figure 2. Model Performance Using RMSE, MAD and MAE

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \tag{26}$$

$$MAD = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{27}$$

$$MAE = median|y_i - \hat{y}_i| \tag{28}$$

Four levels of multicollinearity $\rho = 0.7, 0.9, 0.95$, and 0.99 were taken into consideration with the sample sizes (n) 50, 100, and 300, respectively. RStudio was used to conduct the simulation investigation.

4.1. Synthetic Simulation results and discussion

From Table 1, the study considered different estimators and parameter settings, specifically focusing on the impact of ρ, sample size (n), the number of predictors (p), and the standard deviation of the error term (σ) on three performance metrics: Root Mean Squared Error (RMSE), Mean Absolute Deviation (MAD), and Mean Absolute Error (MAE). The four scenarios evaluated are as follows:

- Scenario 1: $\rho = 0.9$, $n = 100$, $p = 150$, $\sigma = 5$

Table 1. Synthetic Simulation result for Case 1 when $\rho = 0.9$

Estimators	$\rho = 0.9$ n = 100; p = 150 $\sigma = 5$			$\rho = 0.9$ n = 100; p = 150 $\sigma = 10$			$\rho = 0.9$ n = 300; p = 400 $\sigma = 5$			$\rho = 0.9$ n = 300; p = 400 $\sigma = 10$		
	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE
LASSO+SVR	26.8430	21.7488	18.4440	28.7478	23.9014	20.8264	21.0754	16.8444	14.7899	23.4701	18.6448	16.2686
SLTS+SVR	22.2724	17.5132	15.0355	26.5552	21.1598	17.8926	12.6633	9.3135	7.3387	15.9820	12.6866	11.3594
LASSO+RF	35.5183	28.3968	25.1412	36.5823	29.6694	25.3498	33.4387	26.3257	22.7909	34.1354	26.8907	22.8692
SLTS+RF	28.5561	22.7934	19.6163	30.9533	24.6422	20.7138	23.2928	19.1172	17.5915	25.5744	21.0884	18.2711
SLTS	24.2746	16.2250	8.9803	27.1863	17.9567	8.7180	14.7946	11.1936	8.2332	17.0524	12.5492	8.6398
SLTS+RIG	17.0185	13.3692	11.1618	20.6295	16.6662	14.8956	6.95115	5.6712	4.8824	11.4487	9.2333	8.2059
SLTS+LASSO	16.6354	13.4070	11.9488	20.8337	16.6955	15.6771	7.0760	5.8659	5.3854	11.4745	9.2841	8.0862

Abbreviation: LASSO, Least Absolute Shrinkage and Selection Operator; SVR, Support Vector Regression; SLTS, Sparse Least Trimmed Squares; RF, Random Forest; RIG, Ridge Regression.

Table 2. Synthetic Simulation result for Case 1 when $\rho = 0.95$

Estimators	$\rho = 0.95$ n = 100; p = 150 $\sigma = 5$			$\rho = 0.95$ n = 100; p = 150 $\sigma = 10$			$\rho = 0.95$ n = 300; p = 400 $\sigma = 5$			$\rho = 0.95$ n = 300; p = 400 $\sigma = 10$		
	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE
LASSO+SVR	18.8437	15.0382	12.8898	21.9744	18.5746	15.9238	20.2904	16.5333	14.4936	22.6913	18.3954	16.4386
SLTS+SVR	19.9276	14.4614	9.3336	21.7505	17.4263	14.9174	11.7953	8.9305	7.4313	15.0384	11.8380	9.7362
LASSO+RF	25.2501	20.0724	15.9463	27.3645	22.3516	18.4902	28.1613	22.8860	20.2321	29.7132	24.3035	22.4155
SLTS+RF	22.2355	16.8777	12.2504	23.8792	18.8668	16.5866	18.8729	15.4873	13.7341	21.3918	17.1343	14.1497
SLTS	18.0306	12.2901	6.9412	20.4209	14.0493	8.02991	14.1634	10.5807	7.5748	16.6549	12.5130	8.9930
SLTS+RIG	13.2753	10.1151	7.8421	15.8780	12.5686	10.5520	8.03993	6.3708	5.1999	11.8304	9.4298	8.2578
SLTS+LASSO	13.1433	9.9631	7.4213	15.32505	12.2461	10.4511	7.9278	6.2571	5.1440	11.7514	9.3296	7.8722

Table 3. Synthetic Simulation result for Case 1 when $\rho = 0.99$

Estimators	$\rho = 0.99$ n = 100; p = 150 $\sigma = 5$			$\rho = 0.99$ n = 100; p = 150 $\sigma = 10$			$\rho = 0.99$ n = 300; p = 400 $\sigma = 5$			$\rho = 0.99$ n = 300; p = 400 $\sigma = 10$		
	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE
LASSO+SVR	14.7530	11.1688	8.9884	17.7964	13.4237	10.5611	12.8356	10.1985	8.7658	16.2932	13.2573	11.7498
SLTS+SVR	11.6338	8.8130	7.2373	16.3026	12.6488	9.8055	9.3921	7.4709	6.4415	13.3990	10.5959	9.0098
LASSO+RF	18.3331	14.1758	11.4683	20.7784	15.8149	12.5830	16.7843	13.5806	11.8525	18.8583	15.1867	13.3809
SLTS+RF	15.1037	11.7812	9.0266	18.1429	14.2824	11.2670	12.7014	10.0983	8.3977	15.6016	12.2684	9.7653
SLTS	10.2702	7.4963	4.74901	13.7753	9.7018	6.0962	9.3445	6.9557	4.8744	12.4994	9.2556	6.4781
SLTS+RIG	7.3247	5.8874	5.0630	11.5196	9.1739	8.1282	7.1093	5.6631	4.5268	11.4937	9.0484	7.5939
SLTS+LASSO	7.4749	5.9013	4.8593	11.4225	8.8145	7.2433	7.4587	5.7528	4.6000	11.4616	8.9406	7.2498

Table 4. Synthetic Simulation result for Case 2 & Case 3

Estimators	$\rho = 0.7$ Case 2 n = 50; p = 100 $\sigma = 3$			Case 3 n = 50; p = 150 $\sigma = 5$			$\rho = 0.9$ Case 2 n = 50; p = 100 $\sigma = 3$			Case 3 n=50; p = 150 $\sigma = 5$		
	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE
LASSO+SVR	18.0128	15.1143	13.8833	33.3547	25.7267	20.1642	12.4290	9.6161	7.2413	33.3547	25.7267	20.1642
SLTS+SVR	15.6207	12.2684	9.5411	10.4397	7.6970	5.5037	10.8582	8.7181	7.0332	10.4397	7.6970	5.5037
LASSO+RF	21.1271	17.3016	15.3856	38.1769	29.4782	21.0818	13.9574	11.2293	11.1138	38.1769	29.478	21.0818
SLTS+RF	18.3276	14.5703	11.5730	12.2648	9.7650	7.9109	12.0815	10.0769	9.3874	12.2647	9.7650	7.9109
SLTS	13.8578	8.3223	3.1139	7.3889	5.5422	3.9630	8.7448	5.6086	2.4571	7.3889	5.5422	3.9630
SLTS+RIG	18.0549	14.3016	11.8483	4.9899	3.7205	2.5877	7.9015	6.1895	4.3104	4.9899	3.7205	2.5877
SLTS+LASSO	17.6296	13.6973	11.5418	4.6998	3.4543	2.3944	7.7358	5.9558	4.2445	4.6998	3.4543	2.3944

- Scenario 2: $\rho = 0.9$, $n = 100$, $p = 150$, $\sigma = 10$
- Scenario 3: $\rho = 0.9$, $n = 300$, $p = 400$, $\sigma = 5$
- Scenario 4: $\rho = 0.9$, $n = 300$, $p = 400$, $\sigma = 10$

The following estimators were tested and their results are presented in the table: LASSO+SVR, SLTS+SVR, LASSO+RF, SLTS+RF, SLTS, SLTS+RIG and SLTS+LASSO

Key findings and observations (and the effect of σ):. The results indicate that as the standard deviation (σ) of the error term increases (comparing $\sigma = 5$ to $\sigma = 10$), the RMSE, MAD,

Table 5. Synthetic Simulation result for Case 2 & Case 3

Estimators	ρ = 0.95 Case 2 n = 50; p = 100 σ = 3			ρ = 0.95 Case 3 n = 50; p = 150 σ = 5			ρ = 0.99 Case 2 n = 50; p = 100 σ = 3			ρ = 0.99 Case 3 n=50; p = 150 σ = 5		
	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE	RMSE	MAD	MAE
LASSO+SVR	13.0255	10.7337	10.2178	33.3547	25.7267	20.1642	8.3043	6.3013	4.6266	33.3547	25.7267	20.1642
SLTS+SVR	11.5483	8.8556	6.0772	10.4397	7.6970	5.5037	7.6911	5.4824	3.6777	10.4397	7.6970	5.5037
LASSO+RF	13.0419	10.6005	9.4754	38.1769	29.4782	21.0818	11.4876	9.2508	7.6674	38.1769	29.4782	21.0818
SLTS+RF	10.9917	8.6037	7.3005	12.2648	9.7650	7.9109	8.6550	6.5675	5.3200	12.2648	9.7650	7.9109
SLTS	8.0537	5.4300	3.0799	7.3889	5.5422	3.9630	4.5487	3.3090	2.2189	7.3889	5.5422	3.9630
SLTS+RIG	6.5860	5.3516	4.5607	4.9900	3.7205	2.5877	4.4605	3.5403	2.7101	4.9899	3.7205	2.5877
SLTS+LASSO	6.4123	5.0102	3.8531	4.6998	3.4543	2.3944	4.3178	3.2641	2.5095	4.6998	3.4543	2.3944

Table 6. Hyper-parameter Values of the molecular descriptor for the Hybrid Methods

	RMSE	MAD	MAE
LASSO+SVR	0.6725	0.5382	0.4069
SLTS+SVR	0.5210	0.3993	0.2913
LASSO+RF	0.5343	0.4095	0.3117
SLTS+RF	0.5183	0.3837	0.2869
SLTS	0.5569	0.3233	0.1039
SLTS+RIG	0.5020	0.3231	0.1023
SLTS+LASSO	0.4852	0.3112	0.1007

and MAE generally exhibit higher values. This is expected, as higher σ introduces more variability in the data, leading to less accurate model predictions across all estimators.

Effect of Sample Size (n) and Predictor Count:. Comparing scenarios 1 (n = 100, p = 150) to scenario 3 (n = 300, p = 400), it is evident that larger sample sizes and predictor counts result in lower RMSE, MAD, and MAE. This suggests that larger datasets with more predictors tend to lead to improved model performance.

Estimator Performance:. In the scenario with σ = 5 and n = 100, SLTS+RIG and SLTS+LASSO outperform other estimators in terms of RMSE, MAD, and MAE, indicating their robustness in capturing the underlying relationships in the data. In the scenario with σ = 10 and n = 300, SLTS+RF demonstrates competitive results in terms of RMSE, MAD, and MAE, suggesting its effectiveness in high-dimensional data settings. SLTS stands out for its low MAE, especially when σ = 5, which implies that it is well-suited for situations where the absolute magnitude of errors is crucial.

SUMMARY:. From Tables 1 - 5, for the given dataset characteristics (n=50, 100, 300; p=100, 150, 400; σ=3, 5, 10; ρ=0.7, 0.9, 0.95, 0.99 respectively), SLTS+LASSO appears to be the most suitable estimation method, providing the lowest prediction errors across all three metrics. SLTS+RIG also performs exceptionally well. These results offer valuable guidance for researchers and practitioners in selecting the most appropriate modeling approach for similar datasets with high-dimensional predictors and multicollinearity.

Likewise, we can clearly observe that Tables 1, 2 & 3 present the results for Case 1, where the sampling generation technique used allowed for an accurate representation of the degree of multicollinearity through ρ. The results indicate that there is no discernible pattern in prediction error as ρ increases, demonstrating the robustness of the estimation strategies when dealing with multicollinearity using Sparse LTS. The best overall estimation method is still SLTS+LASSO, which provides a lower prediction error compared to SLTS+RIG. As expected, the error increases with higher σ and larger p due to the increase in the number of variables. Since sparsity was taken into account in this scenario, the MAE performance parameter performs better than the RMSE after MAD.

Next, in Cases 2 and 3, the sparsity level was still considered, and different signals were used. Tables 4 & 5 show that, unlike in Case 1, the prediction error increases as multicollinearity increases for all values of n, p, and σ. The LASSO+RF estimator performs the worst of all. SLTS+LASSO remains the superior method, providing a lower test MAE. When the grouping effect is present in Case 3, the estimators LASSO+RF and SLTS+RF produce lower errors than LASSO+SVR and SLTS.

4.2. Real-life Analysis

This study selected 65 imidazo[4,5-b]pyridine derivatives exhibiting anticancer activity from previously published research [54–56]. The biological activity of these compounds was measured using the IC50 value, which represents the concentration of the compound required to inhibit cell growth by 50 percent. To develop a quantitative structure-activity relationship (QSAR) model, the logarithmic scale of the IC50 values (pIC50 = log(IC50)) was used as the response variable. Molecular structures of the 65 compounds were created using CHEM3D software, optimized using the molecular mechanics (MM2) method and then by a molecular orbital package (MOPAC) module. Subsequently, 4885 molecular descriptors, including all 29 blocks based on the optimized molecular structures, were generated using DRAGON software (version 6.0) [2]. To ensure consistency and usefulness of the molecular descriptors, several preprocessing steps were carried out, including the exclusion of descriptors that had constant values for all compounds, the removal of descriptors in which 60 percent of their values were zeros, and the discarding descriptors that had zero values for all compounds. Ultimately, 2540 molecular descriptors were selected for evaluating the QSAR model.

The data were split in a ratio of 70 percent for training, and 30 percent for testing. We used the training set to fit the models and tune the hyperparameters, which were chosen using a grid search. We choose the tuning parameters that minimize the cross-validation error. The molecular descriptors that are shrunk to zero are eliminated, while the descriptors with non-zero coefficients are selected as features for the new data set. The new data set is divided into a training and test set, and the training set is further divided into five-folds. Sets of hyper-parameter values for various machine learning algorithms, such as Random Forest, Ridge, Lasso, and Support Vector Regression, are tuned, and the hyper-parameter with the optimal metric is selected as the final model. Finally, we obtained the root mean squared error, median absolute error and mean absolute error for the final model using the test.

The prediction result is presented in Table 6 and the prediction performance is displayed in Figure 2. It is obvious that selecting the molecular descriptors with Sparse LTS produced the most preferred prediction because the method is robust to outlying values. LASSO selected forty-eight descriptors, while Sparse LTS selected 15 active sets. Figure 2 serves as a visual representation of the estimator performances, specifically focusing on root mean squared error, mean absolute deviation and median absolute error metrics. It provides an intuitive way to assess how each estimator performs and complements the information presented in Table 6, offering a graphical perspective of their relative performance. Table 6 and Figure 2 demonstrate that SLTS+LASSO performs better in terms of the prediction metrics than the other six approaches. The results agree with the simulation study.

5. Conclusion

This study presents novel and robust methods for identifying potential drug compounds and predicting their biological activities. We utilized machine learning techniques to achieve this goal, specifically by using sparse LTS or Lasso algorithms to select important molecular descriptors. The selected descriptors were then divided into training and test sets, with further subdivision of the training set into training and validation sets. We employed various machine learning algorithms, including Random Forest, Ridge, Lasso, and Support Vector Regression, to tune hyper-parameter values for the final model. We evaluated the effectiveness of these algorithms using three standard metrics: root mean square error (RMSE), mean absolute deviation (MAD), and median absolute error (MAE).

To investigate the robustness of our methods, we conducted a simulation study exploring different scenarios. Our results demonstrated that the Sparse LTS or Lasso algorithms effectively handled multicollinearity and outliers. The SLTS+LASSO hybrid estimating approach was the most effective, followed by SLTS+RIG, due to their lower prediction errors. We found that MAE outperformed MAD and RMSE as performance metrics when sparsity was considered. We applied our methods to a QSAR example to validate our simulation results, and the results were consistent with the simulation study.

The findings of this study contribute to the field of high-dimensional data analysis and modeling with multicollinear and outlier data in linear models. Our methods have the potential to be used in drug discovery and development, as they can help identify potential drug compounds and predict their biological activities. Further research in this area is warranted to enhance our understanding of these methods and their potential applications.

Acknowledgment

The authors would like to express their gratitude to God Almighty and to AIMS Senegal for their support.

References

[1] M. Andrea, C. Viviana & R. Todeschini, *Molecular descriptors: handbook of computational chemistry*, Springer International Publishing, Milan, Italy 2017. pp. 2065–2093. https://doi.org/10.1007/978-3-319-27282-5_51.

[2] R. Todeschini, V. Consonni, A. Mauri, & M. Pavan, *Dragon Software: An easy approach to molecular descriptor calculations* (2006) Corpus ID: 13716008 https://api.semanticscholar.org/CorpusID:13716008.

[3] R. Todeschini & C. Viviana, "Molecular descriptors for chemoinformatics", ChemMedChem Journal **5** (2010) 306 https://doi.org/10.1002/cmdc.200900399.

[4] S. Zhong, J. Hu, X. Yu, & H. Zhang, "Molecular image-convolutional neural network (CNN) assisted QSAR models for predicting contaminant reactivity toward OH radicals: transfer learning, data augmentation and model interpretation", Chemical Engineering Journal **408** (2021) 127998 https://doi.org/10.1016/j.cej.2020.127998.

[5] G. Mohammad, D. Bieke, & V. Heyden-Yvan, "Feature Selection Methods in QSAR Studies", Journal of AOAC International **95** (2019) 636 https://doi.org/10.5740/jaoacint.sge_goodarzi.

[6] A. M. Al-Fakih, Z. Y. Algamal & M. H. Lee, "High-dimensional quantitative structure–activity relationship modeling of influenza neuraminidase A/PR/8/34 (H1N1) inhibitors based on a two-stage adaptive penalized rank regression", Journal of Chemometrics **30** (2012) 50. https://doi.org/10.1002/cem.2766.

[7] Z. Y. Algamal, M. H. Lee, A. M. Al-Fakih & M. Aziz, "High-dimensional QSAR prediction of anticancer potency of imidazo[4,5-b]pyridine derivatives using adjusted adaptive LASSO", Journal of Chemometrics **29** (2015) 547. https://doi.org/10.1002/cem.2741.

[8] J. M. Luco, & F. H. Ferretti, "QSAR based on multiple linear regression and PLS methods for the anti-HIV activity of a large group of HEPT derivatives", Journal of chemical information and computer sciences **37** (1997) 392. https://doi.org/10.1021/ci960487o.

[9] F. R. Burden & and D. A. Winkler, "Robust QSAR models using Bayesian regularized neural networks", Journal of Medicinal Chemistry **42** (1999) 3183. https://doi.org/10.1021/jm980697n.

[10] N. AlNuaimi, M. M. Masud, M. A. Serhani & N. Zaki, "Streaming feature selection algorithms for big data: A survey" **18** (2022) 113. https://doi.org/10.1016/j.aci.2019.01.001.

[11] A. Hoerl & R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems", Technometrics **12** (1970) 55. https://doi.org/10.2307/1267351.

[12] L. Breiman, "Heuristics of instability and stabilization in model selection.", The Annals of Statistics **30** (1996) 2350. https://doi.org/10.1214/aos/1032181158.

[13] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso", Journal of the Royal Statistical Society. Series B (Methodological) **58** (1996) 267. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x.

[14] I. Frank & J. Friedman, "A statistical view of some chemometrics regression tools", Technometrics **35** (1993) 109. https://doi.org/10.2307/1269656.

[15] W. Fu, "Penalized regression: The bridge versus the lasso", Journal of Computational and Graphical Statistics **7** (1998) 397. https://doi.org/10.1080/10618600.1998.10474784.

[16] H. Zou & T. Hastie, "Regularization and variable selection via the elastic net", Journal of the Royal Statistical Society **67** (2005) 301. https://doi.org/10.1111/j.1467-9868.2005.00503.x.

[17] M. Eklund, U. Norinder, S. Boyer, & L. Carlsson, "Benchmarking variable selection in QSAR", Molecular Informatics **31** (2012) 173. https://doi.org/10.1002/minf.201100142.

[18] Z. Li & M.J. Sillanpaa, "Overview of LASSO-related penalized regression methods for quantitative trait mapping and genomic selection", Theoretical and Applied Genetics **125** (2012) 419. https://doi.org/10.1007/s00122-012-1892-9.

[19] F. Ghasemi, "Deep neural network in QSAR studies using deep belief network", Applied Soft Computing Journal **62** (2018) 251. https://doi.org/10.1016/j.asoc.2017.09.040.

[20] S. Wacker & S. Y. Noskov, "Performance of machine learning algorithms for qualitative and quantitative prediction drug blockade of hERG1 channel", Computational Toxicology **6** (2018) 55. https://doi.org/10.1016/j.comtox.2017.05.001.

[21] Z. Mozafari, M. A. Chamjangali, M. Arashi, & N. Goudarzi, "Application of the LAD-LASSO as a dimensional reduction technique in the ANN-based QSAR study: Discovery of potent inhibitors using molecular docking simulation.", Chemometrics and Intelligent Laboratory Systems **222** (2022) 104510. https://doi.org/10.1016/j.chemolab.2022.104510.

[22] A. R. Maronna, R. D. Martin & V. J. Yohai, *Robust Statistics: Theory and Methods*, Wiley, New York, USA 2006, pp. 51-85. https://doi.org/10.1002/0470010940

[23] H. Wang, G. Li, & G. Jiang, "Robust regression shrinkage and consistent variable selection through the LAD-lasso", Journal of Business Economic Statistics **25** (2007) 347. https://doi.org/10.1198/073500106000000251.

[24] A. Alfons, C. Croux & S. Gelper, "Sparse Least Trimmed Squares Regression For Analyzing High-Dimensional Large Data Sets", The Annals of Applied Statistics **7** (2013) 226. https://doi.org/10.1214/12-AOAS575.

[25] F. Motamedi, H. Pérez-Sánchez, A. Mehridehnavi, A. Fassihi & F. Ghasemi, "Accelerating Big Data Analysis through LASSO-Random Forest Algorithm in QSAR Studies", Bioinformatics **38** (2022) 469. https://doi.org/10.1093/bioinformatics/btab659.

[26] V. I. Jurtz, A. J. Rosenberg, M. Nielsen, J. J. Almagro-Armenteros, H. Nielsen, C. K. Sonderby, O. Winther & S. K. Sonderby, "An Introduction to Deep Learning on Biological Sequence Data: Examples and Solutions", Bioinformatics **33** (2017) 3685. https://doi.org/10.1093/bioinformatics/btx531.

[27] Y. Liu & S. J. Qin, "A stable Lasso algorithm for inferential sensor structure learning and parameter estimation", Journal of Process Control **107** (2021) 70. https://doi.org/10.1016/j.jprocont.2021.10.005.

[28] Z. Mozafari, M. A. Chamjangali, & M. Arashi, "Combination of least absolute shrinkage and selection operator with Bayesian Regularization artificial neural network (LASSO-BR-ANN) for QSAR studies using functional group and molecular docking mixed descriptors", Chemometrics and Intelligent Laboratory Systems **200** (2020) 103998. https://doi.org/10.1016/j.chemolab.2020.103998.

[29] X. Yan, X. Su & World Scientific, *Linear Regression Analysis: Theory and Computing*, World Scientific Publication, Florida, USA, 2009. pp348. https://doi.org/10.1142/6986.

[30] J. Friedman, T. Hastie, & R. Tibshirani, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition", Springer Series in Statistics New York, California, USA, 2001, pp. 50-57. https://doi.org/10.1007/978-0-387-84858-7.

[31] B. Efron, T. Hastie, I. Johnstone & R. Tibshirani, "Discussion of Least angle regression", The Annals of Statistics **32** (2004) 407. https://doi.org/10.1214/009053604000000067.

[32] G. Li, H. Peng & L. Zhu "Nonconcave penalized M-estimation with a diverging number of parameters", Statistica Sinica **21** (2011) 391. https://www3.stat.sinica.edu.tw/sstest/oldpdf/A21n117.pdf.

[33] P. Rousseeuw & K. Driessen Van, "Computing LTS regression for large data sets", Data Mining and Knowledge Discovery **12** (2006) 3204. https://doi.org/10.1007/s10618-005-0024-4.

[34] T. K. Ho, "The random subspace method for constructing decision forests", EEE Transactions on Pattern Analysis and Machine Intelligence **30** (1998) 832. https://doi.org/10.1109/34.709601.

[35] Y. Amit & D. Geman, "Shape quantization and recognition with randomized trees", Neural Computation **9** (1997) 1545. https://doi.org/10.1162/neco.1997.9.7.1545.

[36] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization", Machine Learning **40** (2000) 139. https://doi.org/10.1023/A:1007607513941.

[37] Y. Freund & R. Shapire *Experiments with a new boosting algorithm. In L. Saitta, editor, Machine Learning*, Proceedings of the 13th International Conference, San Francisco, USA, 1996. pp 148-156. http://dl.acm.org/citation.cfm?id=3091696.3091715.

[38] R. Genuer, J. M. Poggi & C. Tuleau, "Random Forests: Random Forests: some methodological insights", Biomedical Signal Processing (2008) arXiv. https://doi.org/10.48550/arXiv.0811.3619.

[39] B. Debasish, P. Srimanta, & C. P. Dipak, "Support Vector Regression, Neural Information Processing", Statistics and Computing, **10** (2007) 203. https://tinyurl.com/mr46aykt.

[40] V. Vapnik, S. Golowich & A. Smola, "Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing", NIPS'96: Proceedings of the 9th International Conference on Neural Information Processing Systems, Cambridge, USA, 1997, pp. 281–287. https://dl.acm.org/doi/abs/10.5555/2998981.2999021.

[41] A. J. Smola & B. Scholkopf, "A tutorial on support vector regression", Statistics and Computing **14** (2004) 199. https://doi.org/10.1023/B:STCO.0000035301.49549.88.

[42] M. Junshui, T. James, & P. Simon, "Accurate On-line Support Vector Regression", Neural Computation 15, Massachusetts Institute of Technology **15** (2003) 2683. https://doi.org/10.1162/089976603322385117.

[43] C. C. Chang & C. J. Lin, "Training v-support vector regression: Theory and algorithms.", Neural Computation, **14** (2002) 1959. https://doi.org/10.1162/089976602760128081.

[44] G. Cauwenberghs & T. Poggio, "Incremental and decremental support vector machine learning. In T. K. Leen, T. G. Dietterich, and V. Tresp (Eds.), Advances in neural information processing systems", Proceedings 2001 IEEE International Conference on Data Mining. San Jose, CA, USA, 2001, pp 409—123. https://doi.org/10.1109/ICDM.2001.989589.

[45] K. Liu, "A new class of biased estimate in linear regression", Communications in Statistics Theory and Methods Journal **22** (1993) 393. https://doi.org/10.1080/03610929308831027.

[46] K. Liu, "Using Liu-Type estimator to combat collinearity", Communications in Statistics Theory and Methods Journal **32** (2003) 1009. https://doi.org/10.1081/STA-120019959.

[47] M. Arashi, A. F. Lukman & Z. Y. Algamal, "Liu regression after random forest for prediction and modeling in high dimension", Journal of Chemometrics **36** (2022) e3393. https://doi.org/10.1002/cem.3393.

[48] N. A. Alao, K. Ayinde & G. S. Solomon, "A Comparative Study on Sensitivity of Multivariate Tests of Normality to Outliers", ASM Science Journal **12** (2019) 65. https://rb.gy/08c9t.

[49] A. F. Lukman, M. Arashi & V. Prokaj, "Robust biased estimators for Poisson regression model: simulation and applications", Concurrency and Computation: Practice and Experience **35** (2023) e7594. https://doi.org/10.1002/cpe.7594.

[50] A. F. Lukman, E. Adewuyi, K. Månsson & G. B. M. Kibria, "A new estimator for the multicollinear poisson regression model: simulation and application.", Scientific Reports **11** (2021) 3732. https://doi.org/10.1038/s41598-021-82582-w.

[51] A. F. Lukman, A. C. Onate, K. Ayinde & S. Binuomote, "Modified ridge-type estimator to combat multicollinearity: Application to chemical data", Journal of Chemometrics **33** (2019) e3125. https://doi.org/10.1002/cem.3125.

[52] A. F. Lukman, K. Ayinde, S. L. Jegede, S. L. & G. B. M. Kibria, "Modified One-Parameter Liu Estimator for the Linear Regression Model", Modelling and Simulation in Engineering **2020** (2020) 427. https://doi.org/10.1155/2020/9574304.

[53] A. F. Lukman, K. Ayinde, B. Rasak & B. B. Aladeitan, " An unbiased estimator with prior information", Arab Journal of Basic and Applied Sciences **27** (2020) 45. https://doi.org/10.1080/25765299.2019.1706799.

[54] N. M. Ghanem, F. Farouk, R. F. George, S. E. S. Abbas, & O. M. El-Badry, "Design and synthesis of novel imidazo[4,5-b]pyridine based compounds as potent anticancer agents with CDK9 inhibitory activity", Bioorganic Chemistry **80** (2018) 565. https://doi.org/10.1016/j.bioorg.2018.07.006.

[55] Z. Y. Algamal, M. H. Lee, A. M. Al-Fakih, & M. Aziz, "High-dimensional QSAR modelling using penalized linear regression model

with $L_{1/2}$-norm", SAR and QSAR in Environmental Research **27** (2016) 703. https://doi.org/10.1080/1062936X.2016.1228696.

[56] L. Majdouline, C. Samir, A. Azeddine, H. Rachid, M. Bouachrine, & L. Tahar, "Anticancer activity of novel molecules based on Imidazo [4, 5-B] Pyridine. 3D-QSAR Study", International Journal of Advanced Research in Computer Science and Software Engineering **4** (2014) 34. https://doi.org/10.1080/1062936X.2016.1228696.