Predicting the biodegradability of chemicals from QSAR data

Ameer S. H. Alwadiya*

***Abstract* — This paper explores the applications of supervised machine learning models in predicting the biodegradability of chemicals using Quantitative Structure-Activity Relationship (QSAR) data. The study aims to enhance our understanding of the complex relationships between chemical structures and their environmental effect. Through the use of advanced machine learning algorithms, specifically investigating logistic regression and comparing it with other machine learning methods such as the support vector classifier, and also a new logistic regression that is built from scratch, the research demonstrates the potential for accurate and efficient prediction of biodegradability. This contributes to the development of sustainable practices in chemical design and environmental management. The findings highlight the significance of integrating computational methods with experimental data to propel advancements in the field of chemical biodegradability assessment.**

***Keywords* — Quantitative Structure-Activity Relationship (QSAR); Machine learning; Logistic regression; Support vector classifier; Computational method.**

I. INTRODUCTION

IN the face of escalating environmental challenges, the application of machine learning technologies has become imperative for devising effective solutions.
This paper explores how machine learning models can be employed to understand the biodegradability of chemicals, a crucial factor in preventing their harmful accumulation and environmental dispersion. By analyzing QSAR dataset, which is a modeling technique used in medicinal chemistry, pharmacology, and environmental science to predict the biological or physicochemical activity of a molecule based on its chemical structure, machine learning helps us understand complex environmental dynamics and formulate effective solutions.

The objective of this work is to build a fully-validated, predictive model for biodegradability using the built-in models of sklearn, including logistic regression and support vector classifier, and compare them with a logistic regression model implemented from scratch. This will be accomplished using the Python programming language and the Jupyter Notebook IDE.

The task is a classification problem, which involves defining a model capable of predicting the class labels of new instances based on their feature values. With $X = (x_1, x_2, \ldots, x_n)$ representing the values of individual features. Let Y be the output variable representing the class label. In binary classification, Y takes values in $\{0,1\}$, where 0 and 1 represent the two classes. The model is defined by a hypothesis function $h(X)$ that maps input features X to predicted class labels Y'. The goal during training is to find the optimal parameters θ that minimize a predefined loss or cost function. The loss function measures the difference between predicted $Y'$and actual Y class labels.

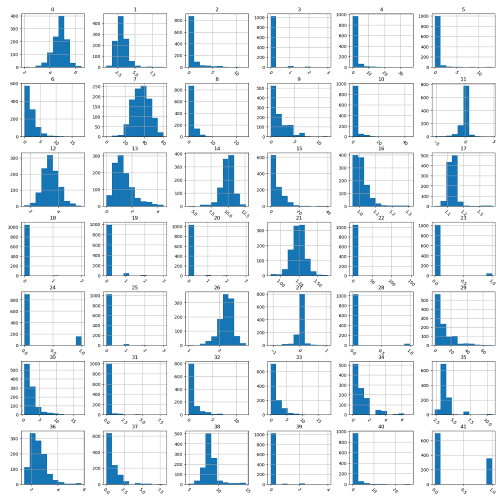
$$J(\theta) = Loss(h(X;\theta), Y) \tag{1}$$

Optimization algorithms, such as gradient descent, are used to minimize the loss function by adjusting the model parameters θ.

II. DATA PROCESSING

A dataset containing information on Quantitative Structure-Activity Relationship (QSAR) and the biodegradability status of 1,055 chemicals has been compiled. Each row in the dataset corresponds to a specific chemical, and the first 41 columns contain the 41 features associated with each chemical. The last column serves as a data label, with '1' indicating biodegradability and '0' indicating non-biodegradability.

The initial data has been converted from .mat format to Pandas using the scipy.io and Pandas libraries. As Pandas is a versatile library that simplifies data manipulation and analysis in Python.To better understand the data and clean it, we are going to generate a histogram for each individual feature. The hist() function from Matplotlib has been employed for this function, enabling us to visualize the distribution of our features.



*M. Alwadiya is a postgraduate from the Department of Automatic Control and Systems Engineering, The University of Sheffield. E-mail: asalwadiya2@sheffield.ac.uk

Fig. 1. Distributions of the features.

By observing the histogram for these features, we can see that features 3, 5, 18, 20, 22, 23, 25, 28, 31, and 39 are almost centered in a single class. Therefore, they may not have a significant impact on our classification. Redundant data can lead to overfitting, so we choose to delete them. The original shape was (1055, 42), and after deleting these specific features, the new shape becomes (1055, 32). We then check for any duplications in our data and remove them; after doing so, the shape becomes (1052, 32).

Now we are going to standardize our data. Standardizing data helps optimization algorithms converge faster and simplifies the process of hyper-parameter tuning. For data standardization, we are going to use Z-score normalization which is a statistical method used to standardize or rescale a distribution of values. This technique transforms the data in a way that the resulting distribution has a mean (average) of 0 and a standard deviation of 1. The formula for calculating the Z-score for a particular data point (x) in a dataset is given by

$$Z = (x - \mu)/\sigma \tag{2}$$

μ is the mean of the dataset, σ is the standard deviation of the dataset. This is the visualization of the data using PCA from the sklearn library before standardization and after standardization:

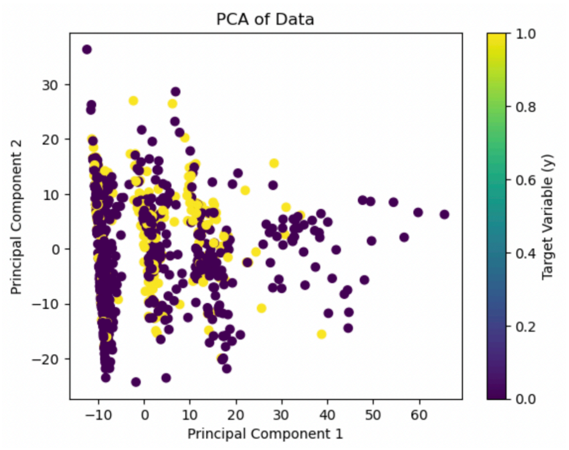


Fig. 2. Visualization of the data before standardization.

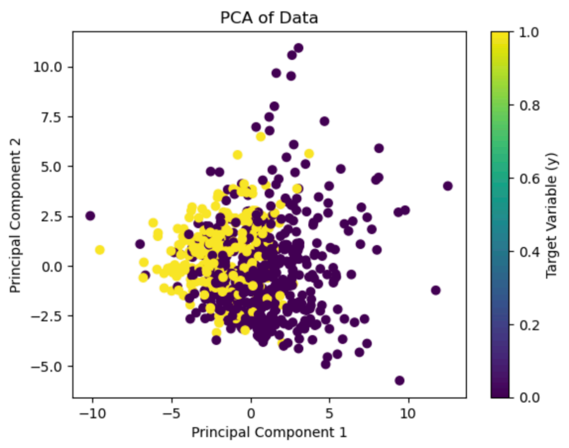


Fig. 3. Visualization of the data after standardization using Z-score.

III. METHODOLOGY

A. Logistic Regression using sklearn

For the first model, a logistic regression model is employed from the scikit-learn library. The logistic regression model is represented by the logistic function (also called the sigmoid function). The formula for the logistic function is

$$g(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

The linear combination z is calculated as

$$z = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + \ldots + b_n \cdot x_n \tag{4}$$

The loss function commonly used in logistic regression is the cross-entropy loss (also known as log loss or logistic loss). The cross-entropy loss measures the performance of a classification model whose output is a probability value between 0 and 1. It is defined as

$$loss = -\left(y \cdot \log(p) + (1 - y) \cdot \log(1 - p)\right) \tag{5}$$

The predicted probability p is given by the logistic function

$$p = \frac{1}{1 + e^{-z}} \tag{6}$$

The overall logistic regression cost function, which is the average of the log loss over all training examples, is often written as

$$J(b) = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} \cdot \log(p^{(i)}) + (1 - y^{(i)}) \cdot \log(1 - p^{(i)})\right) \tag{7}$$

Finding the optimal values of the parameters (b parameters or weights) in logistic regression involves the process of training the model. Gradient Descent is a widely used optimization algorithm for this purpose. Update the weights using the gradient of the cost function with respect to each weight as follow

$$b_j = b_j - \alpha \frac{\partial J(b)}{\partial b_j} \tag{8}$$

regularization is a technique used to prevent overfitting and improve the generalization of the model, Two common types of regularization used in logistic regression are 'L1' regularization and 'L2' regularization. The cost function with L1 regularization is

$$J(b) = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} \cdot \log(p^{(i)}) + (1 - y^{(i)}) \cdot \log(1 - p^{(i)})\right) + \lambda \sum_{j=1}^{n} |b_j| \tag{9}$$

The cost function with L2 regularization is

$$J(b) = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} \cdot \log(p^{(i)}) + (1 - y^{(i)}) \cdot \log(1 - p^{(i)})\right) + \lambda \sum_{j=1}^{n} b_j^2 \tag{10}$$

For tuning the hyperparameters of our model, we use Grid Search Cross-Validation, a method in scikit-learn that systematically tunes hyperparameters for a given model by evaluating various combinations of hyperparameter values.

B. Support Vector Classification (SVC) using sklearn

Support Vector Classification (SVC) involves finding a hyperplane that best separates the data into different classes. The equation for the hyperplane is

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{11}$$

the margin is the distance between the decision boundary (hyperplane) and the nearest data point from either class. The goal of SVM is to maximize this margin, as it generally leads to better generalization to unseen data.

$$M = \frac{1}{\|\mathbf{w}\|} \left| \mathbf{w} \cdot \mathbf{x}_i + b \right| \tag{12}$$

$$y_i \left(\frac{1}{\|\mathbf{w}\|} (\mathbf{w} \cdot \mathbf{x}_i + b) \right) \geq M \tag{13}$$

The objective function that SVMs aim to minimize is

$$\frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \qquad (13)$$

The first term is the regularization term that penalizes the magnitude of the weight vector to prevent overfitting. The second term is the sum of slack variables ξi that represent the degree of misclassification of each training example, C is the regularization parameter that controls the trade-off between achieving a wide margin and allowing misclassifications. The optimization problem is subject to the following constraints

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \tag{14}$$

The Lagrangian for the SVC optimization problem is formed by incorporating the constraints into the objective function

$$L\left(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}\right) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{N} \beta_i \xi_i \tag{15}$$

Grid Search Cross-Validation, a method in scikit-learn is also used here to find the optimal hyperparameters.

C. Logistic Regression from scratch

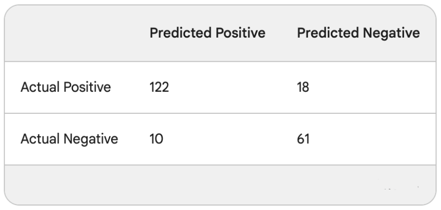
In this model, I have implemented a logistic regression class to gain a deeper understanding of its functionality and customize it for our specific task. The formal and mathematical aspects related to logistic regression are discussed in model A. In this logistic regression class, we have the option to include the L2 regularization term or not. I used gradient descent during the training process to determine the optimal parameters in the update method.

IV. MODEL ANALYSIS

Using train_test_split from sklearn, data is split into training and testing sets by setting test_size to 0.2. This means that 80% of the data will be used for training the model, and the remaining 20% will be used for testing the model.

A. Logistic Regression using sklearn

In this model, we used GridSearchCV to identify the optimal parameters for C, which controls the strength of regularization. Both L1 and L2 norms are employed in this model. After executing GridSearchCV on the set of C values (0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000), it suggests that C=0.1 is the optimal value among them. After training the model and testing it on the test data, we obtain the following confusion matrix



	Predicted Positive	Predicted Negative
Actual Positive	122	18
Actual Negative	10	61

Table 1. confusion matrix for model 1.

The following figure is the ROC (Receiver Operating Characteristic) for this model, which scores an AUC (Area Under the Curve) of 0.939 for the test data prediction, using auc() function from scikit-learn.

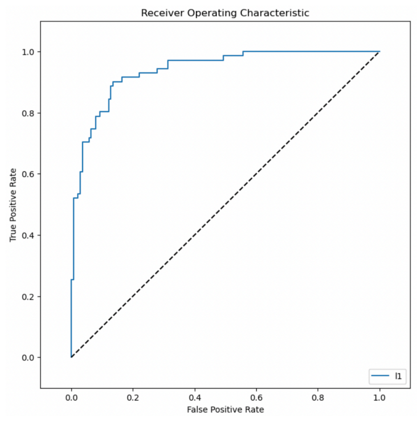


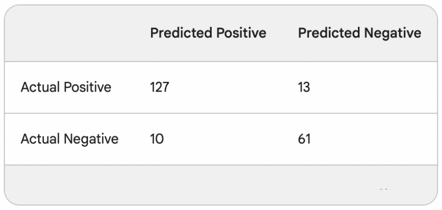
Fig. 4. ROC for model 1.

B. Support Vector Classification (SVC) using sklearn

As explained in equation (13), we need to find suitable values for 'C' for the objective function. Additionally, we must determine an appropriate value for 'gamma,' which is associated with the kernel function used—a mathematical technique that implicitly maps the input data into a higher-dimensional space. In this model RBF kernel function is used

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2) \tag{14}$$

To determine the optimal hyperparameters from a set of C values [0.001, 0.01, 0.1, 1, 10] and Gamma values [0.001, 0.01, 0.1, 1], we also used GridSearchCV. The results indicated that C=10 and gamma=0.01 are the optimal values. This is the confusion matrix for the test data prediction



	Predicted Positive	Predicted Negative
Actual Positive	127	13
Actual Negative	10	61

Table 2. confusion matrix for model 2.

The following figure depicts the ROC for this model, which achieves an AUC of 0.949 for the test data prediction

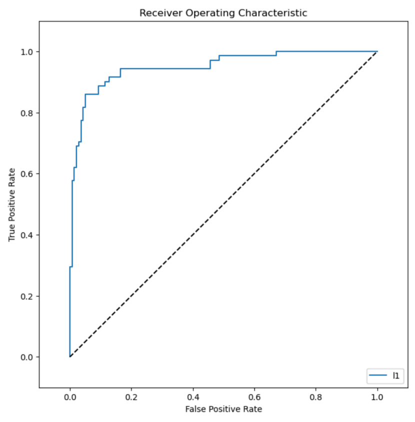
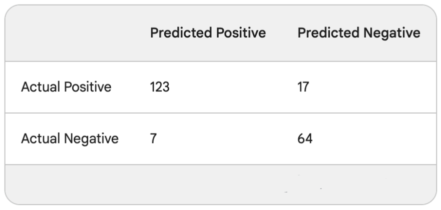


Fig. 4. ROC for model 2.

C. Logistic Regression from scratch

For this model, training is conducted with a learning rate of 0.02, and the number of iterations is set to 100,000 without considering the regularization term. After training the model and testing it on the test data, we obtain the following confusion matrix



	Predicted Positive	Predicted Negative
Actual Positive	123	17
Actual Negative	7	64

Table 3. confusion matrix for model 3.

And this is the ROC curve for this model, which achieves an AUC of 0.946 for the test data predictions

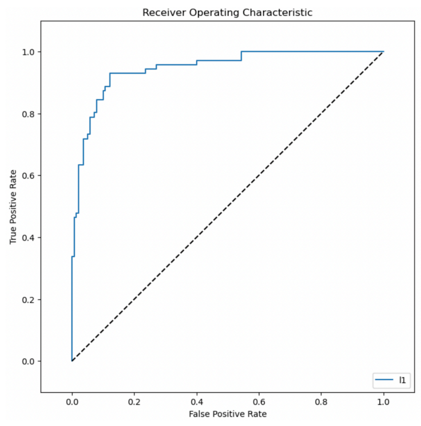


Fig. 5. ROC for model 3.

V. CONCLUSION AND RECOMMENDATION

As we can see from the performance of our models, we obtained an AUC of 0.939 for Model 1, an AUC of 0.949 for Model 2, and finally, an AUC of 0.946 for the last model. Therefore, the second method, which uses the support vector classifier, demonstrates the highest accuracy in predicting the biodegradability of chemicals. We noticed that our model has a higher AUC compared to the first model. This may be because, in our model, we have the option to choose the number of iterations and set the learning rate. In the sklearn model, we simply add regularization terms and control their strength using cross-validation. We let the model choose the learning rate and number of iterations, and it doesn't use gradient descent; instead, it employs many more advanced optimizers.

REFERENCES

1. K. Mansouri, T. Ringsted, D. Ballabio, R. Todeschini, and V. Consonni, "Quantitative structure–activity relationship models for ready biodegradability of chemicals," Journal of chemical information and modeling, vol. 53, no. 4, pp. 867–878, 2013

2. You can find the code files at this link on GitHub: https://github.com/ameer-alwadiya/binary-claasification-ml/blob/8bdb0f89df67be4f62f4931e3ce4afee629206a4/Predicting%20the%20Biodegradability%20of%20Chemicals.ipynb