

SUPPLEMENTARY MATERIALS

TABLE OF CONTENTS

	<u>Page</u>
Section A	2
Section B	3
Figure S1	3
Section C	4
Section D	6
Table S1	7

Section A

A brief description of the determination of k -NN hyperparameters:

- Number of neighbors (k)

Good care must be taken to ensure a proper selection of the k value as it greatly affects the accuracy of k -NN algorithm. One common way of choosing the optimal k value is to perform the cross-validation (CV) method, which is discussed later. An optimal value of k produces a minimum error using the validation dataset.

- Distance function

Distance function – considered the core of the k -NN algorithm – defines how to compute the distance of each observation in the testing dataset (called query observation) from all the observations in the training dataset. An observation is specified by a collection of $n + 1$ data points, $\{x_i, y\}_{i=1}^n$, where n represents the number of attributes known as input variables (x 's), and y represents the corresponding output. Several different types of distance functions have been used and researched for k -NN based models [36, 37]. The Euclidian distance function, often called Ruler distance which is an extension of Pythagoras' theorem, is the most popular, efficient, and commonly used function that is mathematically given by:

$$ED(A, B) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (S1)$$

where A and B are represented by feature vectors whose elements are in the sets $\{x_1, x_2, \dots, x_n\}$ and $\{y_1, y_2, \dots, y_n\}$, respectively, where n is the feature space dimensionality.

- Weighting function

The weighting function specifies how much each of the k observations in the training dataset contributes to the prediction of the query observation in the testing dataset. The standard k -NN algorithm used the same weight for each of the k observations, given by:

$$W(X, X_i) = \frac{1}{k} \quad (S2)$$

It is recommended to apply a non-uniform function in order to allocate smaller weights to the closer observations, and larger weights to the farther observations from the query observation, which means that the closer observations to the query observation will contribute more to the prediction, and vice versa. An example of a non-uniform weighting function is given by Eq. (S3) [38]. After assigning a weight value to each of the k nearest neighbors to the query observations, the output of the query observation (Y) is calculated according to Eq. (S4):

$$W(X, X_i) = \frac{\exp(-D(X, X_i))}{\sum_{i=1}^k \exp(-D(X, X_i))} \quad (S3)$$

where X and X_i are the query observation in the testing dataset and the i -th observation in the training dataset, respectively; $D(X, X_i)$ is the distance between X and X_i ; $W(X, X_i)$ represents the weight assigned to X_i based on the $D(X, X_i)$; k is the number of nearest neighbors to X .

$$Y = \sum_{i=1}^K W(X, X_i) \times Y_i \quad (S4)$$

where Y_i is the output corresponding to X_i .

Section B

An example of a q -fold cross-validation (CV) method:

A brief description of an example of a q -fold CV, graphically illustrated in Figure S1, is given below [39].

- i) The training subset was split into q equal-sized folds.
 - ii) For each k value (1 to 10), q runs were executed such that in each run, the model was trained on $(q-1)$ folds.
 - iii) The trained model was then tested against the remaining fold (called the validation fold) to determine the validation error.
 - iv) The average of errors on the validation folds was plotted versus the k values.
- The value of k that generated the least error in the trained model was selected as the optimal value.

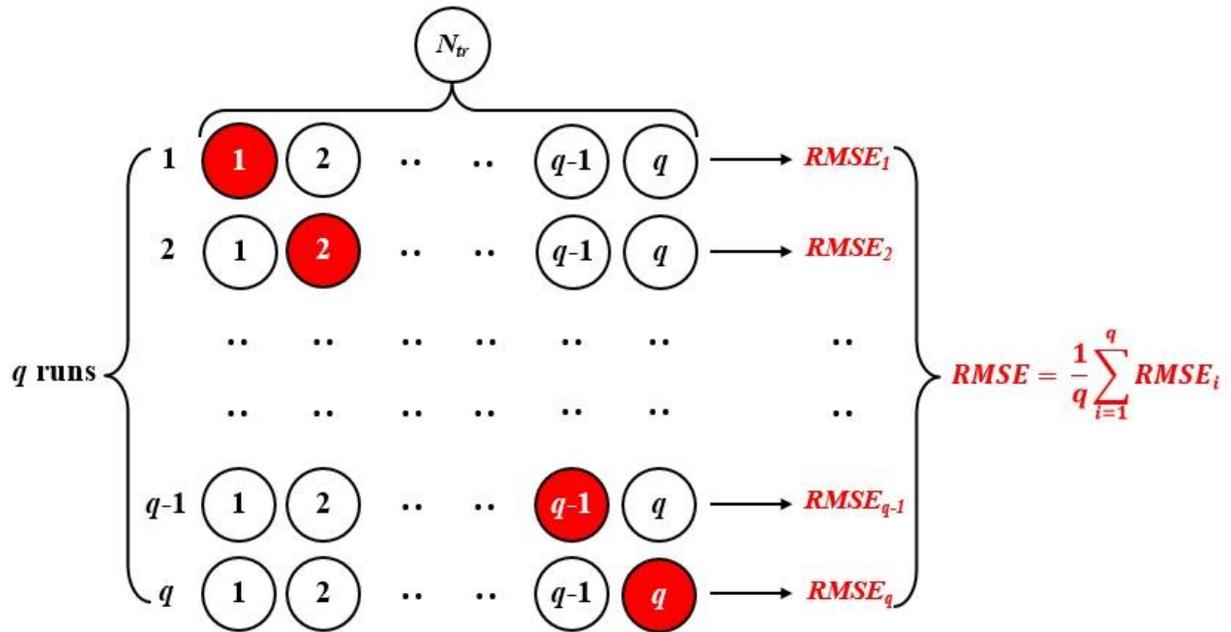


Figure S1. Schematic illustration of q -fold CV approach

Notes: N_{tr} stands for the size of the training subset, which is divided into q equal-sized subset; in each run, the white circles represent the training folds, whereas the red circle represents the validation fold; $RMSE$: root mean squared error.

Section C

To derive a linear SVM regression with the use of Lagrangian function and optimal constraints:

In order to derive a linear SVM regression, Eq. (6) should be optimized, which is generally solved in its dual formulation. To obtain the dual objective function, Lagrangian function (Eq. (S5)) from the primal objective function is constructed:

$$\begin{aligned}
L(w, \xi, \xi^*, b, \alpha, \alpha^*, \beta, \beta^*) &= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \xi_i^*) - \sum_{i=1}^n \alpha_i (\varepsilon + \xi_i - y_i + \langle w, x_i \rangle + b) \\
&\quad - \sum_{i=1}^n \alpha_i^* (\varepsilon + \xi_i + y_i - \langle w, x_i \rangle - b) - \sum_{i=1}^n \beta_i \xi_i - \sum_{i=1}^n \beta_i^* \xi_i^*
\end{aligned} \tag{S5}$$

where $\alpha_i, \alpha_i^*, \beta_i, \beta_i^* \geq 0$ are the Lagrange multipliers associated with the vector x_i , while the following partial derivatives of L with respect to w, b, ξ_i , and ξ_i^* must be equal to zero under the optimal conditions.

$$\partial_w L = 0 \Rightarrow w = \sum_{i=1}^n x_i (\alpha_i - \alpha_i^*) \tag{S6}$$

$$\partial_b L = 0 \Rightarrow \sum_{i=1}^n (\alpha_i^* - \alpha_i) = 0 \tag{S7}$$

$$\partial_{\xi_i} L = 0 \Rightarrow \sum_{i=1}^n \beta_i = C - \sum_{i=1}^n \alpha_i \tag{S8}$$

$$\partial_{\xi_i^*} L = 0 \Rightarrow \sum_{i=1}^n \beta_i^* = C - \sum_{i=1}^n \alpha_i^* \tag{S9}$$

Substituting Eqs. (S6-S9) into Eq. (S5) yields the dual objective function:

$$L(\alpha, \alpha^*) = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n (\alpha_i - \alpha_i^*) (\alpha_j - \alpha_j^*) \langle x_i, x_j \rangle + \varepsilon \sum_{i=1}^n (\alpha_i + \alpha_i^*) - \sum_{i=1}^n y_i (\alpha_i - \alpha_i^*) \tag{S10}$$

which should be minimized with respect to $\alpha_1, \alpha_1^*, \dots, \alpha_n$, and α_n^* , subject to the following constraints:

$$\text{subjected to } \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 0, \quad 0 \leq \alpha_i \leq C, \quad 0 \leq \alpha_i^* \leq C \tag{S11}$$

Equation (S10) can be solved using quadratic programming (QP) method. However, from the perspective of computational load, the use of QP method may be unfeasible because the Gramian matrix, often shortened to Gram matrix, may become too large to be stored in memory. Instead, the use of sequential minimal optimization (SMO) algorithm can reduce the computational load and prevent low memory problems. SMO, originally introduced by Platt [41] at Microsoft Research (MSR) (Redmond, Washington, United States), is the most common algorithm for solving SVM problems, which uses a two-point optimization technique. In other words, SMO breaks down the optimization problem into a series of sub-problems, each involving two Lagrange multipliers, which are then optimized analytically without

requiring a QP solver. For detailed information on SMO algorithm, the reader is referred to the study of Platt [41].

After solving Eq. (S10) to obtain $\alpha_1, \alpha_1^*, \dots, \alpha_n, \alpha_n^*$, substituting Eq. (S6) into Eq. (3) yields:

$$f(x) = \sum_{i=1}^n (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \quad (\text{S12})$$

which is the so-called linear SVM regression, where α_i , and $\alpha_i^* \geq 0$ are the Lagrange multipliers associated with the vector x_i .

All the vectors x_i for which $(\alpha_i - \alpha_i^*) \neq 0$ are called support vectors that are utilized to formulate the solution. The vectors for which $(\alpha_i - \alpha_i^*) = 0$ are not considered as support vectors, which means if these vectors are removed from training dataset prior to training the SVM model, the same solution could be achieved.

Section D

A solved example of how to use the developed SVM model in this study:

To apply the developed SVM model for estimating how much biogas can be produced for a given data on temperature (°C), C/N ratio, and retention time (d) over the range examined, the following equation can be used:

$$f(x) = \sum_{i=1}^{n_{sv}} \alpha \times K(x, x_i) + b$$

where,

- $\alpha = \alpha_i - \alpha_i^*$, where α_i and α_i^* are the Lagrange multipliers associated with x_i in the training dataset (see Table S1 for α values)
- x_i : the i -th observation in the form of a row vector; $x_i = [x_{1i} \ x_{2i} \ x_{3i}]$, where $x_1 = T$ (°C), $x_2 = C/N$ ratio, and $x_3 =$ retention time (d)
- x : a query observation (with the same format as x_i) whose output $f(x)$, cumulative biogas production (mL g VS⁻¹), needs to be estimated
- b : bias with an optimum value of 20.4169 (from Table 6)
- n_{sv} : the number of support vectors; note that all of the observations in the training dataset (90 observations) are considered as support vectors because $\alpha \neq 0$.

$$K(x, x_i) = \exp\left(-\frac{\|x - x_i\|^2}{\gamma^2}\right)$$

where, $\|x - x_i\|^2$, the squared Euclidean distance between the two vectors x and x_i , can be simplified as $(\|x\|^2 + \|x_i\|^2 - 2\langle x^T, x_i \rangle)$; $\langle x, x_i \rangle$ denotes as the dot product between the vectors x and x_i ; γ is the kernel scale (6.93) (from Table 6).

Let x be the query observation with the assumed values for x_1 (Temperature), x_2 (C/N ratio), and x_3 (retention time) as 35°C, 40, and 12 d, respectively.

$$\begin{aligned} f(x) = & (-1200) \times \exp\left(\frac{-(35^2 + 40^2 + 12^2) - (55^2 + 12^2 + 8^2) + 2 \times (35 \times 55 + 40 \times 12 + 12 \times 8)}{6.93^2}\right) + \\ & (-510.15) \times \exp\left(\frac{-(35^2 + 40^2 + 12^2) - (35^2 + 20^2 + 14^2) + 2 \times (35 \times 55 + 40 \times 20 + 12 \times 14)}{6.93^2}\right) + \\ & \vdots \\ & \vdots \\ & \vdots \\ & (-1200) \times \exp\left(\frac{-(35^2 + 40^2 + 12^2) - (55^2 + 20^2 + 9^2) + 2 \times (35 \times 55 + 40 \times 20 + 12 \times 9)}{6.93^2}\right) + 20.4169 \\ & = \mathbf{29.65 \text{ ml gVS}^{-1}} \text{ (The measured value was } \mathbf{29.58 \text{ ml gVS}^{-1}}) \end{aligned}$$

Table S1. α values for the support vectors

# SV	x_1	x_2	x_3	α	# SV	x_1	x_2	x_3	α
1	55	12	8	-1200.00	41	-55	40	2	-736.24
2	35	20	14	-510.15	42	35	12	11	391.22
3	35	40	6	-1200.00	43	35	12	5	-1200.00
4	35	40	7	1091.29	44	55	40	8	-1200.00
5	35	12	7	1200.00	45	55	12	10	1200.00
6	55	12	6	990.43	46	35	20	11	-87.40
7	55	20	3	-674.74	47	55	30	1	-908.06
8	55	20	8	-1200.00	48	35	20	5	-874.91
9	35	40	8	1200.00	49	55	30	4	-1200.00
10	35	12	9	-677.70	50	35	40	2	-154.47
11	35	40	10	-1200.00	51	55	40	4	-1200.00
12	35	30	14	-195.10	52	55	12	2	217.44
13	55	40	5	1200.00	53	55	30	13	83.77
14	55	20	11	1016.17	54	35	20	9	-166.40
15	35	12	6	-715.22	55	55	40	7	-1200.00
16	35	40	4	1200.00	56	35	30	10	314.03
17	55	20	6	1200.00	57	35	20	2	-540.69
18	55	20	14	592.24	58	55	30	8	1200.00
19	55	30	7	893.54	59	35	12	14	-209.33
20	35	20	10	-1200.00	60	55	12	5	1200.00
21	35	30	12	-1200.00	61	55	30	6	-1200.00
22	55	12	7	-1200.00	62	55	40	14	608.99
23	35	12	4	256.43	63	35	40	13	863.77
24	55	40	6	338.56	64	35	40	9	-151.26
25	35	12	2	1200.00	65	55	20	1	-492.69
26	35	30	8	1200.00	66	55	20	13	-1200.00
27	35	12	8	1200.00	67	35	20	3	1200.00
28	35	12	12	359.78	68	55	12	4	-1200.00
29	35	20	8	1200.00	69	35	30	2	-406.79
30	35	30	13	1016.69	70	55	40	3	1200.00
31	35	40	5	-1200.00	71	35	12	1	-795.91
32	55	40	13	-293.91	72	55	12	13	21.04
33	55	30	12	291.86	73	55	30	2	1200.00
34	55	12	11	-1200.00	74	55	20	7	1200.00
35	35	30	9	-119.36	75	35	30	3	590.94
36	55	12	14	135.39	76	55	12	9	1082.35
37	55	40	1	88.41	77	55	20	10	707.73
38	55	20	2	1200.00	78	35	20	6	-1200.00
39	55	40	10	1200.00	79	55	20	5	-1144.37
40	55	30	10	-1190.27	80	55	40	12	-1200.00

Table S1. α values for the support vectors (cont'd)

# SV	x_1	x_2	x_3	α
81	55	40	9	1200.00
82	35	20	7	1200.00
83	55	12	1	-43.84
84	35	12	10	-1200.00
85	35	30	7	-1200.00
86	35	40	14	-450.79
87	55	30	3	830.65
88	35	20	13	988.26
89	35	12	13	168.60
90	55	20	9	-1200.00

SV: support vector; x_1 : temperature; x_2 : carbon-to-nitrogen ratio; x_3 : retention time (d); $\alpha = \alpha_i - \alpha_i^*$ where α_i , and α_i^* are the Lagrange multipliers associated with the vector x_i whose elements are x_1 , x_2 , and x_3 ; α values were rounded off to two decimal places.