ORIGINAL RESEARCH

Ensemble Learning with Extended Newton Support Vector Machines for Enhancing Gene Expression Classification

Huu-Hoa Nguyen[1] · Nguyen-Khang Pham[1]

Received: 3 April 2024 / Accepted: 7 May 2024

Abstract

Gene expression classification plays a crucial role in diagnosing diseases. In response to this critical challenge, the research community has developed a variety of methods. Among these, machine learning approaches, particularly those based on Support Vector Machine (SVM) algorithms, stand out for their effectiveness. However, these algorithms encounter major challenges due to the nature of gene expression datasets, which are characterized by extremely high dimensionality and a relatively small number of samples. This situation significantly challenges machine learning algorithms, as it increases the risk of overfitting and complicates the task of extracting meaningful patterns from a high-dimensional space with limited samples. To address these challenges, we propose an advanced ensemble framework based on SVM techniques. This framework begins with an extension of the Newton SVM, named NSVMX. Building on this foundation, we introduce an ensemble of NSVMX models, called E-NSVMX. We detail our methods through mathematical formulations and algorithmic procedures. Our comprehensive experiments across various gene expression datasets reveal that our proposed methods significantly outperform the LibSVM benchmark in terms of training speed. Moreover, they deliver competitive, and in certain instances, superior classification accuracy. These results make our methods particularly useful for applications that necessitate quick model updates or fast model retraining with new or augmented data. Beyond advancing theoretical knowledge, our research underscores the practical benefits, leading to more efficient and effective machine learning solutions for urgent real-world challenges.

Keywords Ensemble learning · Gene expression classification · High dimensionality · Support vector machine

Introduction

Exploring novel approaches in disease diagnosis through gene expression classification plays a vital role in biomedical research, aiming to broaden our understanding, enhance patient care and introduce new treatments. This endeavor requires an in-depth analysis of gene expression data to discover innovative therapeutic strategies, as highlighted by recent studies [1]. The scientific community widely recognizes the importance of advanced gene expression analysis for improving early anomaly detection and facilitating the development of effective medications [2]. This contributes to a deeper understanding of biological mechanisms, increases diagnostic accuracy, and leads to the discovery of potential therapies. Moreover, integrating cutting-edge computational techniques and machine learning into this process marks a significant advancement. This integration allows researchers to process large-scale datasets and reveal previously inaccessible insights. Such collaborative efforts result in key discoveries, facilitating the rapid translation of research findings into clinical practices that significantly enhance patient health [3].

High-dimensional gene expression datasets exemplify the vast informational depth found within the complex structures of biological systems. These datasets reflect intricate genetic and molecular interactions, posing substantial challenges to bioinformatics and computational biology. They require innovative solutions to access and interpret the extensive information they contain [4]. The ‘curse of dimensionality’, a term illustrating how adding dimensions can lead to

✉ Nguyen-Khang Pham
pnkhang@ctu.edu.vn

Huu-Hoa Nguyen
nhhoa@ctu.edu.vn

[1] College of Information and Communication Technology, Can Tho University, 3/2 Street, Can Tho City 900000, Vietnam

Published online: 31 May 2024

sparser information and reduced algorithm performance [5], represents a principal barrier in analyzing these datasets. This challenge extends beyond mere computational issues, pushing the boundaries of current analytical techniques and motivating the development of novel approaches to effectively handle the complexities of high-dimensional data [6, 7]. Gene expression analysis, with its direct implications for understanding health and disease, becomes even more critical [8]. Overcoming the obstacles presented by high-dimensional data is thus imperative for advancing precision medicine, identifying diverse disease subtypes and fostering early detection approaches. Consequently, addressing these challenges is not only crucial for improving analytical techniques but also for expanding the frontiers of genomics.

Among various strategies employed in machine learning to tackle the complex task of gene expression classification, Support Vector Machines (SVMs) stand out for their classification accuracy [9, 10]. Grounded in statistical learning theory, SVMs utilize kernel functions to effectively handle classification, regression and anomaly detection across multiple domains, including facial recognition, text analysis and bioinformatics [11]. Despite their proven effectiveness, SVM models face challenges when applied to large-scale, high-dimensional datasets like those found in gene expression analysis. The inherent complexity of these datasets, with their vast dimensionality and limited sample sizes, escalates the risk of overfitting and complicates the extraction of meaningful patterns. Additionally, the computational demands of training SVMs, particularly the need to solve quadratic programming (QP) problems, intensify these challenges, making traditional SVM approaches less practical for extensive datasets [10, 12].

To overcome these obstacles, we propose a novel ensemble framework based on SVM. Within this framework, we introduce NSVMX, an extension of the Newton SVM [13], aimed at accelerating the model training process while maintaining or enhancing classification accuracy. NSVMX uses the Sherman-Morrison-Woodbury formula [14] to ensure rapid and accurate classification, even in extremely high-dimensional spaces. We selected the Newton SVM, or NSVM, as our foundational model due to its computational efficiency, which bypasses the need for quadratic programming by directly solving linear equations. This efficiency is particularly noticeable in datasets with large but manageable dimensions, allowing NSVM to quickly classify data using standard computing resources [13].

To further enhance this framework, we introduce an ensemble methodology, E-NSVMX, which combines multiple NSVMX models to accelerate training times and improve classification accuracy. All proposed methods are detailed through meticulous mathematical formulations and precise algorithmic procedures. Comprehensive experimental testing across various gene expression datasets has confirmed our methods' effectiveness, surpassing established benchmarks like LibSVM [15], especially in training speed, while also maintaining or significantly enhancing classification accuracy. Our findings confirm E-NSVMX's superior performance over benchmarks such as LibSVM and highlight the practical benefits of our approach. These benefits include faster and more accurate solutions that are essential for tackling urgent biomedical challenges.

This paper is an extended version of our previous work, "Ensemble Learning with SVM for High-Dimensional Gene Expression Data", published in the proceedings of the International Conference on Intelligent Systems and Data Science (ISDS-2023) [16]. We have enhanced the foundational concepts with significant improvements to deepen and broaden our research. New sections provide a comprehensive exposition of the proposed algorithms. Additionally, we have conducted more rigorous experiments to validate the effectiveness of these algorithms by assessing their robustness and scalability under challenging scenarios. These enhancements not only boost the algorithms' performance but also enrich the reader's understanding of the methodologies and empirical validations.

The remainder of this paper is organized as follows. Section "Theoretical Foundations" delves into SVM and NSVM methodologies, providing a foundation for understanding the technical aspects of these approaches. Section "Methodology" elaborates on our innovative NSVMX and E-NSVMX strategies, detailing the theoretical underpinnings and practical applications of these enhancements. Section "Experimental Validation" is dedicated to the experimental validation of our methods across diverse gene expression contexts, offering empirical evidence of their effectiveness. Finally, Sect. "Conclusion and Future Directions" concludes the paper with a summary of our findings and outlines future research directions.

Theoretical Foundations

This section examines Support Vector Machines (SVMs) [9] and their advanced variant, Newton Support Vector Machines (NSVMs) [13], which are essential to machine learning and pattern recognition. Initially, we explore the SVM's theoretical foundations and operational mechanisms, focusing on how it determines an optimal hyperplane to maximize the margin between different classes. We then investigate the intricacies of NSVMs and explain how this methodology enhances the computational efficiency over traditional SVMs, especially for large-scale data challenges. By comparing these models, we aim to furnish readers with a comprehensive understanding of their applicability, including underlying mathematical formulations and algorithmic details, thereby offering thorough insights into these influential machine learning algorithms.

Support Vector Machines (SVM)

The Support Vector Machine (SVM) [9] is a critical machine learning algorithm extensively utilized for classification and regression tasks. SVM aims to find an optimal hyperplane that distinctly categorizes different classes within a multidimensional feature space by maximizing the margin of separation between these classes. This goal involves solving a quadratic programming problem to minimize an objective function subject to a set of constraints. Mathematically, the SVM optimization task for a linear binary classification scenario is detailed in Fig. 1 as follows.

Given a dataset with m data points x_i (where $i = 1, \ldots, m$ and each $x_i \in \mathbb{R}^n$) associated with their corresponding labels y_i, with $y_i \in -1, +1$, the aim is to find a hyperplane (defined by the normal vector $w \in \mathbb{R}^n$ and offset $b \in \mathbb{R}$) that maximizes the separation margin. This is achieved by maximizing the margin $2/|w|$ (where $|w|$ is the 2-norm of vector w) while minimizing the classification errors represented by the slack variables z_i, leading to the optimization formulation:

$$\min\ \Psi(w,b,z) = \frac{1}{2}|w|^2 + c\sum_{i=1}^{m} z_i\ s.t.\ :\ y_i(w \cdot x_i - b) + z_i \geq 1 z_i \geq 0.$$

Here, c is a regularization parameter that balances margin maximization and error minimization. The plane (w, b) is obtained by solving the quadratic programming problem in Eq. (). Finally, the decision/classification function for any new data point x is given by:

$$predict(x) = \mathrm{sign}(w \cdot x - b). \tag{1}$$

This framework supports linear classification and extends to nonlinear classification by applying kernel functions, effectively transforming linear separability issues into high-dimensional spaces where linear separation is possible. Some typical kernel functions include the Polynomial Function of Degree d, the Radial Basis Function and the Sigmoid Function. Additional insights into the standard SVM and other kernel-based learning methods are provided in [17].

Newton Support Vector Machines (NSVM)

In the scholarly article, Platt explains that the computational demand of SVM solutions, as outlined in Eq. (1), requires computational resources on the order of $O(m^2)$, where m represents the total number of training data points. This insight underscores the challenges of applying traditional SVM methods to large-scale, high-dimensional datasets [12].

Building upon this, the Newton Support Vector Machine (NSVM) methodology, introduced by Mangasarian [13], revises the SVM framework detailed in Eq. (1). This revision targets two primary objectives:

- Margin maximization: Achieved through the optimization process $\min \frac{1}{2}\|w\|^2$, aiming to expand the classification margin to its maximum feasible extent.
- Error minimization: Simultaneously, the methodology aims to minimize classification errors, indicated by the objective $\min \frac{c}{2}\sum |z_i|^2$, thus improving the model's classification accuracy.

Further elaboration involves substituting z with $[e - D(Aw - eb)]+$, where the operation $(x)+$ sets any negative components of the vector to zero, and e represents a column vector filled with ones. This substitution effectively transforms the objective function Ψ, as defined in the quadratic programming framework of Eq. (1), into an unconstrained minimization problem, now expressed in Eq. (2):

$$min\ \Psi(w,b) = (c/2)\|\ [e - D(Aw - eb)]_+\ \|^2 + (1/2)\|\ w, b\ \|^2. \tag{2}$$

To simplify further, let u represent the vector $[w_1, w_2, ...w_n, b]^T$ and let E stand for the matrix $[A \quad -e]$. This notation shift allows the re-expression of the unconstrained problem in Eq. (2) as:

To further simplify, let u represent the vector $[w_1, w_2, \ldots, w_n, b]^T$ and let E denote the matrix $[A \quad -e]$. This shift in notation enables us to re-express the unconstrained problem described in Eq. (2) as:

$$min\ \Psi(u) = (c/2)\|\ (e - DEu)_+\ \|^2 + (1/2)u^T u. \tag{3}$$

Mangasarian [13] advocates employing the finite-difference Newton method to robustly address this convex minimization challenge in Eq. (3).

The NSVM learning algorithm is mathematically detailed in Algorithm 1. Essentially, this algorithm enhances the traditional SVM learning framework by incorporating the Newton method, known for its fast convergence properties, particularly beneficial for processing large-scale datasets. The NSVM learning algorithm unfolds as follows:

- Input initialization: The algorithm starts by initializing input matrices A and D, representing the training features and the labels, respectively. It also sets a tuning constant c to balance classification accuracy and margin width.
- Matrix E construction: A crucial initial step is constructing matrix E by augmenting the feature matrix A with a negative bias unit vector $-e$, facilitating subsequent optimization processes.

- Iterative optimization: The algorithm focuses on an iterative process that refines the parameter vector u through the gradient and Hessian of the objective function Ψ, guiding parameter adjustments for more detailed updates.
- Gradient and Hessian computations: Each step involves computing the gradient and the generalized Hessian, utilizing these calculations to update u via the Newton–Raphson method, which takes full advantage of the Hessian's curvature for substantial, context-sensitive progress.
- Parameter updating and convergence: Parameters are iteratively refined until the gradient norm falls below a set threshold, indicating convergence. The final parameters from the converged vector u define the optimal separating hyperplane (w, b) for the SVM.

The NSVM algorithm marks a significant advancement in SVM training techniques, offering rapid convergence and effective scaling for large datasets. This ensures that an optimal solution is reached with greater computational efficiency. Mangasarian has shown that the sequence u_i in Algorithm 1 converges to the global minimum within a small number of iterations, typically between 5 and 8. As a result, NSVM only requires solving the linear equation 6 that involves $n + 1$ variables $(w_1, w_2, \ldots, w_n, b)$. This approach bypasses the more complex quadratic programming seen in Eq. 1. The efficiency improvement is especially prominent in scenarios with input space dimensionality under 10^3, even with datasets containing millions of data points. In such cases, NSVM can classify points within minutes on a standard personal computer, outperforming the traditional LibSVM [15] in speed while also delivering competitive accuracy.

Algorithm 1 Newton SVM learning algorithm

```
input :
        Training dataset represented by $A$ and $D$
        Constant $c > 0$ for tuning errors, margin size
output:
        $(w,b)$
Training:
 begin
   1. Create matrix $E = [A \quad -e]$
   2. Starting with $u_0 \in R^{n+1}$ and $i = 0$
   3. repeat
        3.1. The gradient of $\Psi$ at $u_i$ is:
             $$\bigtriangledown\Psi(u_i) = c(-DE)^T(e - DEu_i)_+ + u_i \qquad (4)$$
        3.2. The generalized Hessian of $\Psi$ at $u_i$ is:
             $$\partial^2\Psi(u_i) = c(-DE)^T diag([e - DEu_i]_*)(-DE) + I \qquad (5)$$
        with $diag([e - DEu_i]_*)$ denotes the $(n+1) \times (n+1)$ diagonal matrix
         whose $j^{th}$ diagonal entry is sub-gradient of the step function $(e - DEu_i)_+$
         and $I$ is the identity matrix of size $(n+1) \times (n+1)$.
        3.3. Updating
             $$u_{i+1} = u_i - \partial^2\Psi(u_i)^{-1} \bigtriangledown \Psi(u_i) \qquad (6)$$
        3.4. Increment $i = i + 1$
   until $\bigtriangledown\Psi(u_i) < tol$;
   4. Optimal plane $(w,b)$: $(w_1,\ w_2,\ \ldots,\ w_n,\ b)$ via $u_i$
end
```

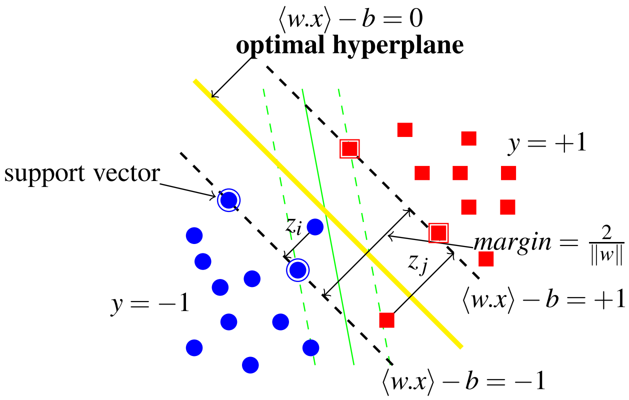
$\langle w.x\rangle - b = 0$
optimal hyperplane
$y = +1$
support vector
z_i
$margin = \frac{2}{\|w\|}$
z_j
$\langle w.x\rangle - b = +1$
$y = -1$
$\langle w.x\rangle - b = -1$

Fig. 1 Linear separation of data points into two classes

Methodology

This section introduces our proposed methods, detailing the development of ensemble learning algorithms in conjunction with the Newton Support Vector Machine (NSVM). These developments specifically target the demands of high-dimensional gene expression data analysis. First, we explore significant enhancements to the NSVM algorithm, leading to its evolved variant, NSVMX, which is proficient in handling the complexities of high-dimensional data environments. Following this, we introduce an ensemble learning strategy, E-NSVMX, that employs NSVMX to boost the training phase and enhance classification accuracy. The section is structured into two main parts: ‘Extension of NSVM foundations’, examining the modifications and enhancements to the NSVM algorithm, and ‘Ensemble learning with NSVMX’, elaborating on integrating these enhancements into an ensemble learning framework to effectively address the challenges of analyzing large-scale, high-dimensional datasets.

Extension of NSVM Foundations (NSVMX)

In the context of gene expression data, where dimensionality often vastly outnumbers available samples, the NSVM algorithm encounters computational hurdles due to the need to invert large $(n + 1) \times (n + 1)$ matrices. Addressing this challenge, we propose an extension of the NSVM algorithm, referred to as NSVMX, optimized for high-dimensional contexts.

To mitigate the computational demand associated with matrix inversion, we introduce an extension utilizing the Sherman-Morrison-Woodbury (SMW) formula. This approach transforms the computational burden into a more manageable $m \times m$ matrix inversion, where m represents a significantly smaller set of data points compared to the dimensionality n. By applying the SMW formula, we redefine the NSVM’s inverse matrix computation as follows:

- **Matrix inversion adaptation:** Using the SMW formula, the adjusted inverse computation is expressed as:

$$(A + UV^T)^{-1} = A^{-1} - A^{-1}U(I + V^TA^{-1}U)^{-1}V^TA^{-1}, \quad (4)$$

  where A and U are defined based on the dataset’s structure and dimensionality reduction requirements.
- **Algorithmic adaptation:** We specify $Q = \mathrm{diag}(\sqrt{c[e - DEu_i]_*})$ and $P = Q(-DE)$, enabling the concise representation of the NSVM’s inverse matrix, which is critical for the NSVMX algorithm. This representation is given by:

$$\partial^2\Psi(u_i)^{-1} = (I + P^TP)^{-1}. \quad (5)$$

  Therefore, we use the SMW formula (Eq. 4) on the right-hand side of Eq. 5, where the inverse matrix $\partial^2\Psi(u_i)^{-1}$ is represented as shown in Eq. 6:

$$\partial^2\Psi(u_i)^{-1} = I - P^T(I + PP^T)^{-1}P. \quad (6)$$

  This effectively reduces the computational burden associated with high-dimensional data analysis, as Eq. 6 involves inverting $(m) \times (m)$ matrices rather than large $(n + 1) \times (n + 1)$ matrices, where $m \ll n$ is typical for gene expression datasets.
- **Update rule:** The update rule for parameter u_i within the NSVM iterative process is consequently refined, offering enhanced computational efficiency, particularly beneficial for datasets where dimensionality extremely exceeds sample size.

By adopting this extension, we effectively scale the SVM framework for high-dimensional gene expression datasets, ensuring computational feasibility and maintaining the algorithm’s robustness against overfitting due to the high dimensionality. This extension not only preserves the intrinsic advantages of the NSVM, such as fast convergence and adaptability to large-scale datasets, but also broadens its applicability to prevalent scenarios in bioinformatics and computational biology.

The proposed NSVMX algorithm is mathematically detailed in Algorithm 2. The key ideas behind this algorithm are summarized as follows:

Algorithm 2 NSVMX algorithm

```
input :
        Training dataset represented by $A$ and $D$
        Constant $c > 0$ for tuning errors, margin size
output:
        $(w, b)$
Training:
 begin
   1. Create matrix $E = [A \quad -e]$
   2. Starting with $u_0 \in R^{n+1}$ and $i = 0$
   3. repeat
      3.1. The gradient of $\Psi$ at $u_i$ is:
           $\bigtriangledown\Psi(u_i) = c(-DE)^T(e - DEu_i)_+ + u_i$    (4)
      3.2. The generalized Hessian inversion of $\Psi$ at $u_i$ is:
           $\partial^2\Psi(u_i)^{-1} = I - P^T(I + PP^T)^{-1}P$    (9)
      with $Q = diag(\sqrt{c[e - DEu_i]_*})$ and $P = Q(-DE)$ and $I$ is the identity
        matrix of size $(n+1) \times (n+1)$.
      3.3. Updating
           $u_{i+1} = u_i - \partial^2\Psi(u_i)^{-1} \bigtriangledown \Psi(u_i)$    (10)
      3.4. Increment $i = i + 1$
   until $\bigtriangledown\Psi(u_i) < tol$;
   4. Optimal plane $(w, b)$: $(w_1, w_2, \ldots, w_n, b)$ via $u_i$
end
```

- Input preparation: The algorithm begins by defining its inputs: a training dataset represented by matrices A (attributes or features) and D (desired outputs or labels), along with a constant $c > 0$. This constant c is crucial as it balances the error margin and the size of the margin in the optimization problem, influencing the robustness and generalization of the model.
- Output definition: The expected output from this algorithm is a tuple (w, b), where w represents the weight vector defining the hyperplane, and b is the bias term, allowing the hyperplane to not necessarily pass through the origin.
- Matrix E construction: A new matrix E is constructed by augmenting matrix A with a negative bias vector $-e$. This step prepares for subsequent mathematical operations in the optimization process.
- Initialization and iterative optimization: The algorithm initializes a vector u_0 in the space $\mathbb{R}^{n+1}$ and an iteration counter $i = 0$. It then updates u_i iteratively until the gradient norm of the potential function $\Psi(u_i)$ is less than a tolerance level, indicating convergence.
- Gradient calculation: At each iteration, the gradient of the potential function Ψ at u_i is computed to guide parameter adjustment towards the optimal solution.
- Generalized Hessian computation: The generalized Hessian matrix, offering second-order curvature information of the potential function at u_i, is computed. Its inversion is simplified using the Sherman-Morrison-Woodbury formula for computational efficiency.
- Parameter update: Utilizing the Newton–Raphson method, u_i is updated by subtracting a portion of the Hessian's inverse multiplied by the gradient from u_i.
- Convergence check and iteration: The iteration counter is incremented after each update, repeating the process until the gradient norm meets the specified tolerance level for convergence.
- Optimal hyperplane extraction: Upon convergence, the final update of u_i yields the separating hyperplane parameters (w, b), defining the SVM's decision boundary in the feature space.

Ensemble Learning with NSVMX (E-NSVMX)

Building on the concepts introduced in Sect. "Extension of NSVM Foundations (NSVMX)", which detailed the NSVMX algorithm designed for handling high-dimensional data, this section presents an advanced ensemble learning framework. Our objective is to address the computational and memory demands posed by large-scale datasets, characterized by their extensive dimensions and the large number of data points. By combining advanced ensemble techniques with the foundational principles of the NSVMX algorithm, we aim to provide a comprehensive solution that enhances scalability and, potentially, classification accuracy in environments with massive datasets.

The standard NSVM algorithm faces significant hurdles with large-scale datasets, mainly due to the computational demands of managing and inverting large matrices-either $(m \times m)$ for NSVMX or $(n+1) \times (n+1)$ for NSVM. These operations require considerable resources and memory, affecting the efficiency of processing in large data contexts.

To address these challenges, we introduce an ensemble mechanism integrating Bagging [18] and Arc-x4 [19] techniques. This framework, named Ensemble of Extended Newton Support Vector Machines (E-NSVMX), aims to achieve two main goals: enhance scalability for effective management of large-scale datasets and improve classification accuracy in gene expression data settings.

E-NSVMX comprises an ensemble of 'weak' classifiers, each trained using either the NSVM or NSVMX algorithms (as detailed in Algorithms 1 and 2, respectively). This method iteratively refines the ensemble of weak classifiers over successive iterations. The choice between NSVM and NSVMX for each classifier training session is based on the specific characteristics of the dataset, particularly the ratio of its dimensionality to the number of data points:

- For datasets with high dimensionality, regardless of the number of data points, NSVMX (Algorithm 2) is utilized to train the ensemble's weak classifiers.
- Conversely, for datasets characterized by a larger number of data points relative to their dimensionality, the original NSVM (Algorithm 1) is selected for training the classifiers.

A key aspect of E-NSVMX is its dynamic weighting mechanism, particularly effective within the Arc-x4 framework. This mechanism dynamically adjusts the weights of data points after each iteration, giving more attention to those previously misclassified. This iterative process ensures subsequent classifiers focus on the more challenging data points, systematically boosting the model's accuracy and resilience. E-NSVMX combines scalability enhancements from ensemble techniques with accuracy improvements through iterative learning and adaptive weight adjustments. It aims to offer a robust classification solution for large-scale, high-dimensional datasets.

Our proposed E-NSVMX algorithm is mathematically formulated in Algorithm 3. Here is a breakdown of the algorithm's general ideas, structured to aid understanding:

- **Objective and input/output:**
  - Objective: The goal is to create a robust classification model capable of handling large-scale, high-dimensional datasets by using the strengths of ensemble learning combined with NSVMX.
  - Input: The algorithm takes a dataset comprising m data points, where each point x_i is a feature vector in $\mathbb{R}^n$, and y_i is its corresponding binary label. Additionally, a constant $c > 0$ adjusts the NSVMX's error margin and margin size, while *NumIt* determines the number of iterative training rounds.
  - Output: It produces the decision plane parameters (w, b), aggregated from various NSVMX model iterations.
- **Algorithm summary:**
  1. Initialization:
     - The algorithm starts by assigning equal initial weights to all data points, ensuring an unbiased beginning for the ensemble learning process.
  2. Iterative training:
     - For a specified number of iterations (*NumIt*), the algorithm conducts the following steps:
       - Data sampling: In each iteration t, it samples a subset of data points S_t based on the current probability distribution p_t, facilitating diverse and representative training sets over iterations.
       - Model training: A model $NSVMX_t$ is trained on the sampled subset S_t with the specified constant c, producing a weak classifier h_t.
       - Error evaluation and weight adjustment (Arc-x4 mode): If the Arc-x4 adjustment is applied, the algorithm calculates prediction errors ϵ_i for each data point across all classifiers. It then updates the weights to focus more on harder-to-classify data points in subsequent iterations. The updated weight for each data point increases with its classification error, ensuring adaptive learning that prioritizes challenging instances.

3. Model aggregation:

   - After iterating through training and updating, the algorithm aggregates all the weak classifiers h_t into a comprehensive model, represented by decision plane parameters (w, b). This final model encapsulates learned insights across diverse data segments and iterative refinements, aiming to achieve high classification accuracy on large-scale, high-dimensional datasets.

In summary, the E-NSVMX algorithm methodically executes a sequence of focused training phases, evaluates the accuracy of classifications and adjusts its learning focus based on these evaluations. This process of continuous refinement and adjustment enhances the algorithm's ability to classify data accurately. Ultimately, this iterative process equips the E-NSVMX algorithm with a robust decision-making capability, enabling it to effectively handle and interpret the intricacies of large and complex high-dimensional datasets.

Table 1 Summary of gene expression datasets

ID	Datasets	Classes	Data points	Dimensions
1	ALL-AML Leukemia	2	72	7129
2	Breast Cancer	2	97	24,481
3	Ovarian Cancer	2	253	15,154
4	Lung Cancer	2	181	12,533
5	Prostate Cancer	2	136	12,600
6	Ovarian Cancer NCI-QStar	2	216	373,410
7	Translation Initiation Sites	2	13,375	927

Algorithm 3 E-NSVMX algorithm

```
input :
        Training dataset with $m$ data points:
        $\{x_i, y_i\}_{i=1,m}$, $x_i \in R^n$ and $y_i \in \pm 1$
        Constant $c > 0$ for tuning errors, margin size
        Number of iterations $NumIt$
output:
        $(w, b)$
Training:
 begin
   1. Initial distribution of $m$ data points: $p_1(i) = 1/m$
   2. for $t \leftarrow 1$ to $NumIt$ do
      2.1. Sampling $S_t$ of data points using $p_t$
      2.2. Learning $NSVMX_t$ from $S_t$: $h_t = NSVMX_t(S_t, c)$
      2.3. if $Arc\text{-}x4$ then
         2.3.1. Computing predicting error for each data point $x_i$ with previous
            classifiers $h_t$:
            $\varepsilon_i = \sum_t h_t(x_i) \neq y_i$
            2.3.2. Updating distribution of $m$ data points:
            for $i \leftarrow 1$ to $m$ do
                  $p_{t+1}(i) = \frac{1+\varepsilon_i{}^4}{Z_t}$ (where $Z_t$ is the normalization factor)
            end
      end
   end
   3. Optimal plane $(w, b)$ is obtained by aggregating models $h_t$
end
```

Experimental Validation

In this section, we conduct a meticulous experimental validation to assess the effectiveness of our innovative ensemble learning algorithms, focusing specifically on their training time and classification accuracy over large-scale datasets. This thorough evaluation is segmented into four main parts: Experimental setup and implementation; Dataset descriptions; Parameter settings and evaluation protocols; and Comparative analysis of algorithm performance. Through this organized examination, we aim to provide detailed and comprehensive insights into the algorithms' performance and capabilities, highlighting their potential advantages and applicability in tackling challenges associated with large-scale data environments.

Experimental Setup and Implementation

Implementation Strategy

Our ensemble learning algorithms, including NSVMX, are implemented in the C/C++ programming language. This choice was made due to C/C++'s significant computational efficiency, essential for our experiments' complex demands. Additionally, we have integrated the ATLAS/Lapack linear algebra libraries [20, 21]. These libraries are well-known for their efficient and robust performance in extensive numerical computations. By integrating these resources, we aim to maximize the efficiency of our implementations.

Computational Environment

The computational analyses are conducted on a Linux Fedora 32 operating system, using an Intel Core i7-4790 CPU at 3.6 GHz and equipped with 32 GB of RAM. This configuration, akin to a high-end personal computing setup, provides a stable and balanced environment suitable for both the development and evaluation phases. It ensures that our experimental results are reflective of what can be expected in a typical high-performance computational setting, thus enhancing the relevance and generalizability of our findings.

Benchmarking Strategy

To thoroughly evaluate the performance and efficiency of our developed algorithms, we employ LibSVM [15] as a benchmark within our evaluation framework. Recognized as a benchmark in SVM implementations, LibSVM provides an essential comparative baseline, enabling a detailed analysis of our algorithms' performance in relation to this well-established standard. Benchmarking against LibSVM is crucial for uncovering the relative strengths and pinpointing potential areas for improvement in our proposed methodologies.

By integrating rigorous software practices with a robust computational infrastructure and a comprehensive benchmarking routine, we highlight the methodological rigor that forms the cornerstone of our experimental research. This detailed approach underscores our focus on delivering validated and reliable insights into the performance of our ensemble learning frameworks, especially within the challenging domain of complex gene expression data analyses.

Dataset Descriptions

This section details the seven high-dimensional gene expression datasets used in our experiments, as cited in [22]. Characterized by their significant dimensionality, these datasets serve as a rigorous testing ground for our algorithms, facilitating a thorough evaluation. A summary of these datasets is presented in Table 1, offering an overview of their primary characteristics for quick reference. In the following paragraphs, we offer detailed descriptions of each dataset to explain their unique characteristics and the challenges they present to our study.

- **ALL-AML Leukemia dataset** This dataset comprises gene expression profiles from patients diagnosed with Acute Lymphoblastic Leukemia (ALL) and Acute Myeloid Leukemia (AML), designed to facilitate the differentiation between these two types of leukemia. It includes 72 samples, each characterized by 7129 gene expression features. With a predefined division of 38 training samples and 34 testing samples, this dataset provides a

Table 2 Summary of evaluation protocols associated with gene expression datasets

ID	Dataset	Class	Data point	Dimension	Evaluation protocol
1	ALL-AML Leukemia	2	72	7129	38 trn - 34 tst
2	Breast Cancer	2	97	24,481	78 trn - 19 tst
3	Ovarian Cancer	2	253	15,154	Leave-one-out
4	Lung Cancer	2	181	12,533	32 trn - 149 tst
5	Prostate Cancer	2	136	12,600	102 trn - 34 tst
6	Ovarian Cancer NCI-QStar	2	216	373,410	Leave-one-out
7	Translation Initiation Sites	2	13,375	927	3-fold

well-organized framework for rigorous evaluation and analysis.

- **Breast cancer dataset** This dataset, comprising gene expression data from breast tissue samples, distinguishes between malignant and benign conditions. It includes 97 instances with 24,481 features, pre-divided into 78 training samples and 19 testing samples. This configuration makes it a comprehensive dataset for breast cancer analysis, providing a substantial basis for evaluating classification algorithms in distinguishing between these conditions.
- **Ovarian cancer dataset** With 253 samples and 15,154 features, the Ovarian Cancer dataset provides a comprehensive molecular examination of cancerous versus

Table 3 Training time performance of algorithms in seconds on gen expression datasets

ID	Dataset	LibSVM	E-NSVMX. Arc-x4	E-NSVMX. Bagging
1	ALL-AML Leukemia	9.14	*5.01*	**1.82**
2	Breast Cancer	269.66	*75.43*	**8.16**
3	Ovarian Cancer	403.60	*10.13*	**8.11**
4	Lung Cancer	20.80	*3.51*	**0.47**
5	Prostate Cancer	1.60	*0.70*	**0.51**
6	Ovarian Cancer NCI-QStar	158.95	*20.13*	**9.46**
7	Translation Initiation Sites	314.00	*50.27*	**35.72**

Bold values indicate the best performance of either our proposed algorithms or the benchmark algorithm within the table, highlighting the primary findings that are crucial for the empirical analysis and conclusions

Italic values indicate the second-best performance of either our proposed algorithms or the benchmark algorithm within the table, which provides additional context to the comparisons

Table 4 Classification accuracy (%) of algorithms on gene expression datasets

ID	Dataset	LibSVM	E-NSVMX. Arc-x4	E-NSVMX. Bagging
1	ALL-AML Leukemia	**97.06**	**97.06**	**97.06**
2	Breast Cancer	*73.88*	**84.36**	*76.28*
3	Ovarian Cancer	**100**	**100**	**100**
4	Lung Cancer	*98.63*	**98.78**	98.69
5	Prostate Cancer	73.58	*97.08*	**97.11**
6	Ovarian Cancer NCI-QStar	*97.65*	**97.72**	*97.65*
7	Translation Initiation Sites	**92.44**	*92.38*	*92.07*

Bold values indicate the best performance of either our proposed algorithms or the benchmark algorithm within the table, highlighting the primary findings that are crucial for the empirical analysis and conclusions

Italic values indicate the second-best performance of either our proposed algorithms or the benchmark algorithm within the table, which provides additional context to the comparisons

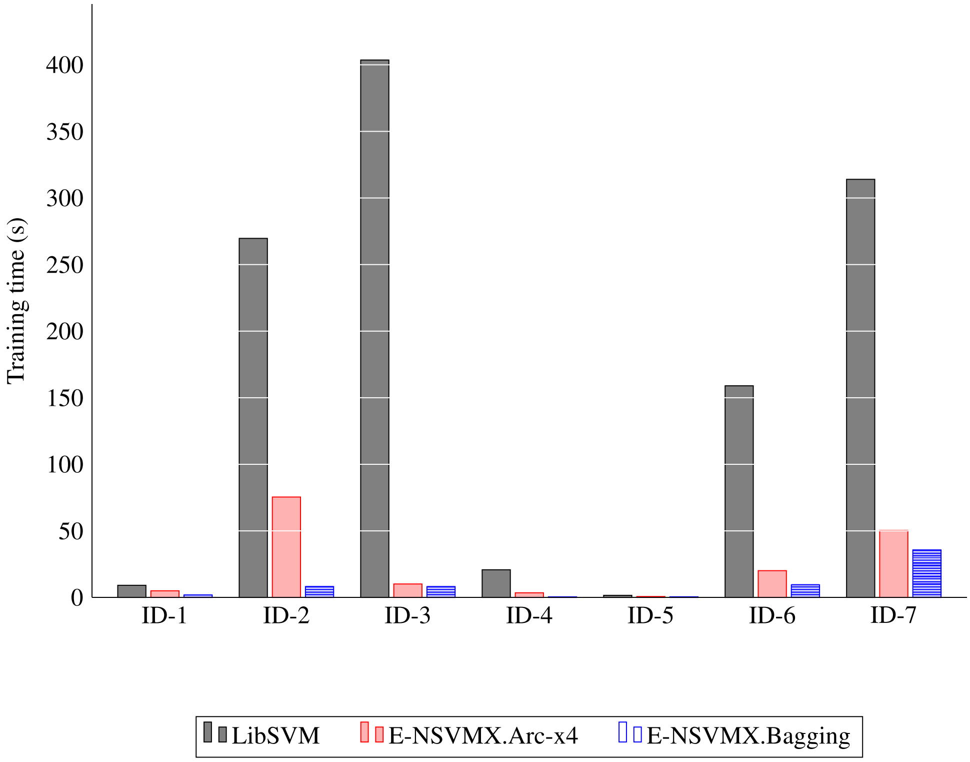

Fig. 2 Training time comparison in seconds across algorithms on gene expression datasets

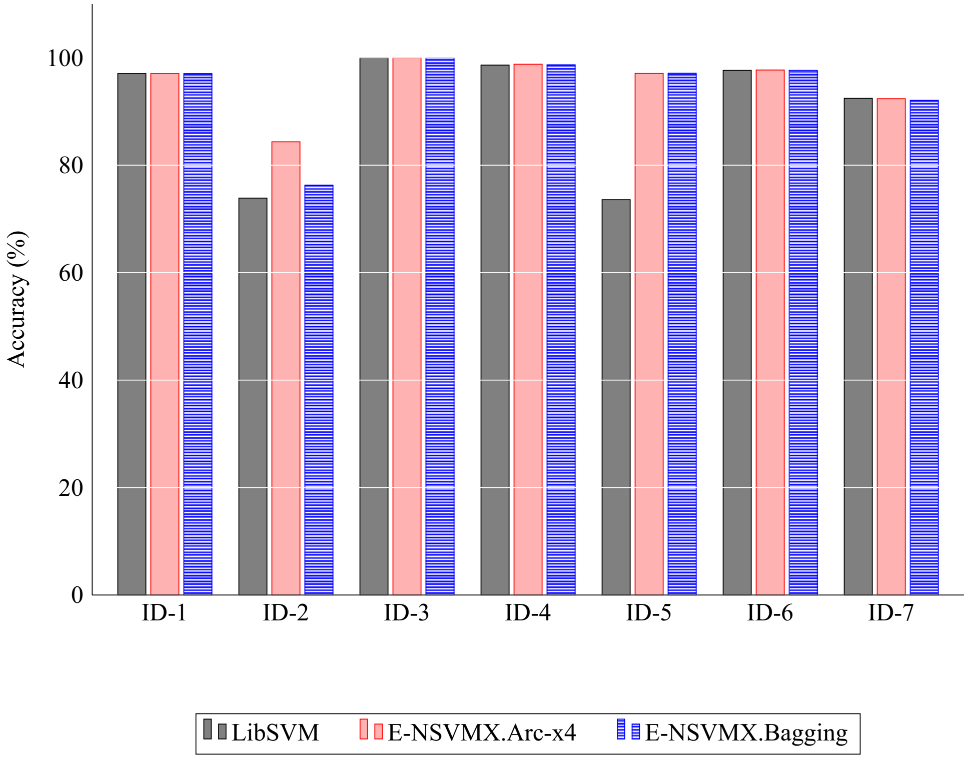

Fig. 3 Classification accuracy (%) comparison across algorithms on gene expression datasets

normal tissue expressions, aiding in unraveling the pathogenesis of ovarian cancer. This dataset has not been split in advance into training set and a testing set.

- **Lung cancer dataset** This dataset aids in analyzing gene expression differences between lung cancer and normal samples. It includes 181 samples and 12,533 features. This collection is crucial for identifying genetic markers relevant to lung cancer research. The dataset is predivided into 32 training and 149 testing samples, which facilitates a structured approach to evaluation.
- **Prostate cancer dataset** This dataset provides valuable insights into prostate cancer, encompassing 136 samples with 12,600 features. It compares gene expressions in tumor and normal tissues to explain the genetic basis of the disease. Pre-partitioned into 102 training and 34 testing samples, it facilitates targeted analysis and model evaluation.
- **Ovarian cancer NCI-QStar dataset** This dataset enables comprehensive molecular studies of ovarian cancer. It offers a vast collection of 373,410 gene expression features across 216 samples. The dataset stands out for its thorough representation of the molecular diversity inherent in the disease. It lacks a predetermined split into training and testing sets. This absence allows for flexible application in research.
- **Translation initiation sites dataset** Focused on identifying translation initiation sites, this dataset includes 13,375 samples. Each sample has 927 features crucial for examining gene regulation at the translation initiation phase. The dataset's significant sample size and comprehensive feature set are particularly remarkable. It does not come with a pre-specified split between training and testing sets. This design allows for flexible application in research.

Parameter Settings and Evaluation Protocols

This section outlines the customized approaches we have developed for algorithm tuning and evaluation. Each approach is meticulously adapted to the distinctive properties and structures of the gene expression datasets used in our study. Our methodologies are specifically designed to accommodate the datasets, varying based on the presence or absence of predefined training and testing sets.

Parameter Tuning in Predefined Dataset Structures

For datasets having predefined training and testing splits, such as the ALL-AML Leukemia, Breast Cancer, Lung Cancer and Prostate Cancer datasets, we undertake a thorough parameter tuning process. In this context, our algorithms are configured to train an ensemble of 50 weak NSVM classifiers.

This number, 50, is chosen based on empirical evidence. It effectively balances computational resource usage with improvements in classification accuracy. This balance allows for a detailed exploration of the datasets' decision boundary landscapes. It enhances the models' generalization capabilities without sacrificing computational efficiency.

For algorithms like LibSVM and NSVM, we carefully adjust the critical parameter c to find an optimal balance between error tolerance and margin width. We then evaluate our models' performance using the predefined testing sets.

Cross-Validation in Undefined Dataset Structures

For datasets lacking predetermined training and testing splits, such as the Ovarian Cancer, Ovarian Cancer NCI-QStar and Translation Initiation Sites datasets, we employ cross-validation to rigorously evaluate the algorithms' performance. This method is particularly effective for datasets without predefined splits, offering a robust evaluation framework.

With datasets comprising fewer than 300 data points, we utilize leave-one-out cross-validation (LOOCV) to make full use of the available data and ensure comprehensive model validation. In LOOCV, each data point is sequentially used as a single testing instance, while the model is trained on the remaining data. This technique allows for a detailed analysis of the model's classification accuracy on an individual level.

Conversely, for datasets larger than 300 data points, such as the Translation Initiation Sites dataset, we apply 3-fold cross-validation. This method divides the dataset into three equal parts, alternating each as the validation set and the remaining data for training. Repeating this process three times enables an aggregated evaluation of the model's generalization ability.

The final column of Table 2 concisely details these validation strategies. Here, 'trn' explicitly refers to the training set samples or data points, while 'tst' indicates the testing set samples, used to evaluate the model's performance.

Comparative Analysis of Algorithm Performance

This section analyzes algorithmic performance on high-dimensional gene expression datasets, emphasizing training time efficiency and classification accuracy. Results are presented in tables and figures for intuitive comparisons. Our E-NSVMX approaches, designated as E-NSVMX.Arc-x4 and E-NSVMX.Bagging, as introduced in Sect. "Ensemble Learning with NSVMX (E-NSVMX)", are thoroughly evaluated.

Training Time Analysis

Our analysis exhibits significant differences in training times between LibSVM, E-NSVMX.Arc-x4 and E-NSVMX.Bagging, as detailed in Table 3 and illustrated in Fig. 2. Specifically, LibSVM shows an average training time of 168.25 s, while E-NSVMX.Arc-x4 and E-NSVMX.Bagging demonstrate significantly shorter times of 23.60 s and 9.18 s, respectively.

These results indicate that LibSVM is approximately 7.13 times slower than E-NSVMX.Arc-x4 and 18.33 times slower than E-NSVMX.Bagging. Such marked differences in time efficiency underline the superior speed of our ensemble algorithms in processing a variety of gene expression datasets.

Accuracy Evaluation

The examination of classification accuracies demonstrates the effectiveness of LibSVM, E-NSVMX.Arc-x4 and E-NSVMX.Bagging across various gene expression datasets. These accuracies are presented as average percentages: 90.46% for LibSVM, 95.34% for E-NSVMX.Arc-x4 and 94.12% for E-NSVMX.Bagging, as detailed in Table 4 and illustrated in Fig. 3. Notably, E-NSVMX.Arc-x4 shows superior performance with 4 wins, 2 ties and 1 loss against LibSVM. Meanwhile, E-NSVMX.Bagging has 3 wins, 3 ties and 1 loss in a direct comparison with LibSVM.

These results underscore the capabilities of E-NSVMX.Arc-x4 and E-NSVMX.Bagging, especially for large-scale datasets like the Ovarian Cancer NCI-QStar and the Translation Initiation Sites datasets. Our proposed algorithms not only perform well in terms of training speed but also achieve commendable accuracy, demonstrating their suitability for complex gene expression datasets where both time efficiency and reliability are regarded as critical factors.

Combining insights from training speed and accuracy evaluations provides a comprehensive view of overall algorithm performance. While fast training enhances efficiency, accuracy remains the definitive benchmark of success. The analysis points out LibSVM's potential for accuracy, albeit slower in training. In contrast, E-NSVMX.Arc-x4 and E-NSVMX.Bagging stand out by offering both rapid training and high precision, proving their efficiency in scenarios that demand quick model updates with new data, thus facilitating the analysis of gene expression data.

Conclusion and Future Directions

In this research, we have introduced an advanced ensemble framework based on Support Vector Machine (SVM) techniques. This framework is specifically designed to tackle the complex challenge of classifying gene expressions, characterized by high dimensionality and sparse data points. Starting with the Newton Support Vector Machine (NSVM), a variant of SVM known for its rapid training time, we establish a foundation. We then enhance this foundation with NSVMX, an extended version of NSVM, to further increase training speed. Yet, our objectives go beyond speed. We strive to develop algorithms that stand out in both training efficiency and classification accuracy. This ambition leads to the creation of the NSVMX ensemble framework, termed E-NSVMX. Through detailed mathematical formulations and algorithmic implementations, we introduce two versions of E-NSVMX: NSVMX.Arc-x4 and E-NSVMX. Bagging.

Our comprehensive experiments across various gene expression datasets reveal that the proposed methods significantly surpass the benchmark set by LibSVM in terms of training time. Furthermore, they deliver competitive, and in certain instances, superior classification accuracy. These findings makes our methods particularly useful for applications that necessitate quick model updates or fast model retraining with new or augmented data. Our work contributes not only to the domain of gene expression classification but also holds promise for other bioinformatics applications, offering efficient and reliable solutions to critical biomedical challenges.

Future plans include broadening our research to cover a wider array of gene expression datasets, intensively evaluating our algorithms' flexibility and effectiveness across different bioinformatics contexts. Another aim is adapting our algorithmic framework to efficiently tackle multi-class classification problems. This adaption will further increase our methods' applicability within bioinformatics and related fields. Additionally, we plan to explore integrating our algorithms with other computational techniques to tackle new and complex bioinformatics challenges. Through these efforts, we seek to advance machine learning techniques for bioinformatics applications, contributing to the broader scientific community and its practical applications.

Author Contributions Huu-Hoa Nguyen contributed 90% and Nguyen-Khang Pham contributed 10% to this work.

Funding The authors declare no funding information.

Data Availibility Not applicable.

Declarations

Conflict of Interest The authors declare no conflict of interest.

Research Involving Human and/or Animals: Not applicable.

Informed Consent Not applicable.

References

1. Ensenyat-Mendez M, Orozco JIJ, Arias P, Munoz S, Baker JL, Salomon MP, Marti M, DiNome ML, Cortes J, Marzese DM. Construction and validation of a gene expression classifier to predict immunotherapy response in primary triple-negative breast cancer. Commun Med. 2023;3:93. https://doi.org/10.1038/s43856-023-00311-y.
2. Alharbi F, Vakanski A. Machine learning methods for cancer classification using gene expression data: a review. Bioengineering. 2023;10(2):173. https://doi.org/10.3390/bioengineering10020173.
3. Gupta S, Gupta MK, Shabaz M, Sharma A. Deep learning techniques for cancer classification using microarray gene expression data. Front Physiol. 2022;13: 952709.
4. Koul N, Manvi SS. Framework for classification of cancer gene expression data using bayesian hyper-parameter optimization. Med Biol Eng Comput. 2021;59:2353–71.
5. Lei C. Curse of dimensionality. Encyclopedia of Database Systems (2009)
6. Yang Q, Wu X. 10 challenging problems in data mining research. Int J Inf Technol Decis Mak. 2006;5(4):597–604.
7. Altman N, Krzywinski M. The curse(s) of dimensionality. Nat Methods. 2018;15:399–400. https://doi.org/10.1038/s41592-018-0019-x.
8. Gao F, Wang W, Tan M, Zhu L, Zhang Y, Fessler E, Vermeulen L, Wang X. Deepcc: a novel deep learning-based framework for cancer molecular subtype classification. Oncogenesis. 2019;8(9):44.
9. Vapnik V. The nature of statistical learning theory. Springer; 2013.
10. Tizi W, Berrado A. Machine learning for survival analysis in cancer research: a comparative study. Sci Afr. 2023;21:01880.
11. Lantz B. Machine learning with r: Learn techniques for building and improving machine learning models, from data preparation to model tuning, evaluation, and working with big data. Packt Publishing Ltd; 2023.
12. Platt J. Fast training of support vector machines using sequential minimal optimization. In: Schölkopf B, Burges C and Smola A (eds) Advances in Kernel Methods – Support Vector Learning. 1999;185–208.
13. Mangasarian O. A finite newton method for classification problems. Technical Report 01-11, Data Mining Institute, Computer Sciences Department, University of Wisconsin (2001)
14. Golub G, Loan CV. Matrix computations. 3rd ed. Balti-more: John Hopkins University Press; 1996.
15. Chang C-C, Lin C-J. LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol. 2011;2(3):1–27.
16. Do T-N, Tran-Nguyen M-T. Ensemble learning with svm for high-dimensional gene expression data. In: International Conference on Intelligent Systems and Data Science, 2023;29–40. Springer.
17. Cristianini N, Shawe-Taylor J. An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press; 2000.
18. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–40.
19. Breiman L. Arcing classifiers. Ann Stat. 1998;26(3):801–49.
20. Whaley R-C, Dongarra J. Automatically tuned linear algebra software. In: Ninth SIAM Conference on Parallel Processing for Scientific Computing (1999). CD-ROM Proceedings
21. Dongarra J, Pozo R, Walker D. LAPACK++: a design overview of object-oriented extensions for high performance linear algebra. In: proceeding of Supercomputing, 1993;62–171.

22. Jinyan L, Huiqing L. Kent ridge bio-medical dataset repository. School of Computer Engineering: Nanyang Technological University; 2004.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.