SPECTRUM OF ENGINEERING SCIENCES
ISSN (E) 3007-3138 (P) 3007-312X VOLUME 4, ISSUE 6, 2026

A COMPUTATIONAL MATHEMATICAL FRAMEWORK FOR HIGH-DIMENSIONAL ENGINEERING DATA ANALYSIS USING ADVANCED LINEAR ALGEBRA, MATRIX FACTORIZATION AND OPTIMIZATION TECHNIQUES

Muhammad Umair Aslam[1], Touseef Sultan[*2], Muhammad Qasim Zafar[3], Akbar Ahmad[4]

[1]*Department of Computer Science, University of Gujrat, Gujrat, Pakistan*
[*2]*Department of Information Technology, International Institute of Science and Arts Technology (IISAT), Gujranwala, Pakistan*
[3]*Department of Mathematics, University of Baltistan, Skardu, Pakistan*
[4]*Department of Computer Science, University of Messina, Italy*

[1]uog0304@gmail.com, [*2]touseef.sultan@iisat.edu.pk, [3]zafarqasim596@gmail.com, [4]akbar.hussain@studenti.unime.it

DOI: https://doi.org/10.5281/zenodo.20587956

Keywords
High-Dimensional Engineering Data Analysis, Advanced Linear Algebra, Mathematical Optimization, Dimensionality Reduction, Predictive Modeling, Machine Learning, Engineering Data Analytics, Intelligent Decision Support Systems

Article History
Received: 09 April 2026
Accepted: 21 May 2026
Published: 08 June 2026

Corresponding Author: *
Touseef Sultan

Abstract

*High-dimensional engineering datasets are increasingly generated from smart sensors, simulation platforms, industrial monitoring systems, communication networks, and intelligent control environments. However, the large number of variables, nonlinear relationships, redundant features, and computational complexity often reduce the efficiency and accuracy of conventional data analysis methods. This study presents **a** mathematical framework for high-dimensional engineering data analysis using advanced linear algebra and optimization techniques. The proposed framework integrates matrix decomposition, vector space transformation, dimensionality reduction, eigenvalue-based feature representation, convex optimization, gradient-based learning, and regularization methods to improve data processing, predictive modeling, and machine learning performance in engineering applications. The methodology focuses on transforming complex engineering datasets into optimized mathematical representations by applying Principal Component Analysis, Singular Value Decomposition, least-squares optimization, and regularized regression techniques. These methods help reduce noise, remove irrelevant features, enhance computational speed, and improve model interpretability. The optimized feature space is then used with machine learning models such as Support Vector Machine, Random Forest, and Neural Network classifiers for predictive analysis and decision support. Experimental results demonstrate that the proposed mathematical framework significantly improves model performance compared with traditional feature-processing methods. The dimensionality of the dataset was reduced by 42.6%, while preserving 96.8% of the original data variance. The proposed framework achieved an overall prediction accuracy of 97.3%, precision of 96.5%, recall of 95.9%, and F1-score of 96.2%. In addition, computational training time was reduced by 31.4%, and mean squared error*

decreased from 0.084 to 0.031 after applying optimization-based feature transformation. The results confirm that advanced linear algebra and mathematical optimization techniques provide a strong foundation for high-dimensional engineering data analysis. Overall, this research highlights the importance of mathematical modeling in improving machine learning efficiency, predictive accuracy, and intelligent decision support for modern engineering systems. The proposed framework can be applied in areas such as smart manufacturing, structural health monitoring, electrical systems, robotics, and engineering design optimization.

1- Introduction:

In the modern era of digital engineering, intelligent automation, and data-driven decision-making, engineering systems are generating large volumes of complex and high-dimensional data. This data is commonly collected from smart sensors, industrial machines, simulation models, monitoring systems, robotic platforms, communication networks, power systems, and automated control environments. Engineering fields such as smart manufacturing, electrical engineering, robotics, structural health monitoring, mechanical design, biomedical engineering, transportation systems, and industrial automation increasingly depend on advanced data analysis methods to improve prediction, optimization, fault detection, system monitoring, and intelligent decision support [1]. High-dimensional engineering data usually contains a large number of variables, parameters, measurements, or features. Although such data provides rich information about system behavior, it also creates several analytical challenges. These challenges include feature redundancy, noise, multicollinearity, nonlinear relationships, high computational cost, overfitting, and difficulty in interpreting model outputs. In many practical engineering problems, the number of features may be very large compared with the number of available observations, which reduces the effectiveness of traditional statistical and machine learning methods. Therefore, advanced mathematical techniques are required to represent, simplify, transform, and optimize engineering datasets in a more reliable and computationally efficient manner. Linear algebra provides a strong mathematical foundation for high-dimensional engineering data analysis because it allows complex datasets to be represented in the form of vectors, matrices, tensors, and multidimensional spaces. Important concepts such as vector spaces, matrix operations, eigenvalues, eigenvectors, orthogonality, projections, rank, basis transformation, and matrix factorization help identify hidden patterns and meaningful relationships within engineering datasets. Techniques such as Principal Component Analysis, Singular Value Decomposition, eigenvalue decomposition, and matrix factorization are especially useful for dimensionality reduction, noise removal, feature extraction, and data compression. These methods reduce unnecessary complexity while preserving the most important information required for accurate predictive modeling [2]. Mathematical optimization is also an essential component of high-dimensional engineering data analysis. Optimization techniques are used to improve model performance by minimizing error, selecting important features, reducing computational burden, and finding the best possible solution under given constraints. Methods such as least-squares optimization, convex optimization, gradient descent, regularization, and constrained optimization are widely used to enhance the accuracy and reliability of machine learning models. In engineering applications, optimization supports fault diagnosis, system control, performance improvement, energy efficiency, design optimization, and intelligent decision-making. This study presents a mathematical framework for high-dimensional engineering data analysis using advanced linear algebra and optimization techniques. The proposed framework combines mathematical data representation, dimensionality reduction, matrix

decomposition, regularization, optimization-based learning, and machine learning models to improve the analysis of complex engineering datasets. The study makes the following important contributions:

1. Development of a structured mathematical framework that integrates advanced linear algebra, mathematical optimization, and machine learning for high-dimensional engineering data analysis.
2. Improvement of data representation and dimensionality reduction by using vector-space transformation, matrix decomposition, Principal Component Analysis, and Singular Value Decomposition to reduce redundant features and preserve important information.
3. Enhancement of predictive modeling performance through optimization techniques such as gradient-based learning, least-squares optimization, convex optimization, and regularization.
4. Reduction of computational complexity and training time by transforming high-dimensional engineering data into lower-dimensional and optimized feature spaces.
5. Support for intelligent engineering decision-making by improving prediction accuracy, model interpretability, and reliability in applications such as smart manufacturing, power systems, robotics, sensor-based monitoring, and industrial fault diagnosis.

Machine learning has become a powerful approach for engineering prediction, classification, monitoring, and decision support. However, the performance of machine learning models depends strongly on the quality of input features and the mathematical structure of the data. If high-dimensional data is directly provided to machine learning models without proper preprocessing, the models may show low accuracy, high training time, overfitting, and poor generalization. Therefore, the integration of linear algebra and optimization techniques with machine learning provides a more effective solution for improving model efficiency, accuracy, and interpretability. The proposed framework is designed to address the limitations of conventional engineering data analysis methods. Traditional methods often struggle when datasets contain too many variables, noisy measurements, redundant information, or complex dependencies. By applying advanced mathematical transformations and optimization strategies, the proposed framework identifies the most important data patterns, removes irrelevant features, and improves the quality of predictive modeling [3]. This makes the framework suitable for real-world engineering environments where datasets are often large, dynamic, incomplete, and computationally demanding. In practical engineering applications, accurate prediction and reliable decision-making are highly important for system safety, operational efficiency, fault prevention, and performance optimization. For example, in smart manufacturing, the proposed framework can support machine fault detection, production quality improvement, and process optimization. In electrical power systems, it can assist in load forecasting, fault diagnosis, and stability analysis. In robotics, it can improve sensor fusion, motion prediction, and control optimization. In structural health monitoring, it can help detect damage patterns and predict possible system failures. Therefore, the proposed mathematical framework has strong potential for multiple engineering domains. Overall, this research emphasizes that advanced linear algebra and mathematical optimization are not only theoretical concepts but also practical tools for solving modern engineering data challenges. By integrating these methods with machine learning, high-dimensional engineering datasets can be processed more efficiently, predictive models can achieve better performance, and intelligent decision support systems can become more accurate, reliable, and computationally effective.

2- Role of Linear Algebra in Engineering Data Representation:

Linear algebra plays a fundamental role in high-dimensional engineering data analysis because it provides the mathematical language for representing, organizing, transforming, and interpreting complex engineering datasets. In modern engineering systems, data is commonly collected from sensors, simulation models, industrial machines, robotic devices, electrical

networks, image-processing systems, and structural monitoring platforms. These datasets often contain multiple variables, measurements, and system parameters that must be arranged in a meaningful mathematical form before analysis. Linear algebra enables this process by representing data through vectors, matrices, tensors, and multidimensional feature spaces. Most engineering datasets can be expressed in matrix form, where each row represents an observation, experiment, machine condition, or sensor reading, while each column represents a feature, parameter, or measured variable [4]. For example, in a smart manufacturing system, rows may represent different production cycles, while columns may represent temperature, vibration, pressure, speed, energy consumption, and product quality indicators. This matrix-based structure allows mathematical operations to be applied systematically for data transformation, dimensionality reduction, feature extraction, compression, and predictive modeling. Linear algebra helps convert raw engineering data into a structured mathematical representation. Each observation can be treated as a vector in an nnn-dimensional space. This means that every machine state, sensor reading, signal pattern, or system condition can be represented as a point or direction in a feature space. By studying distances, angles, projections, and transformations within this space, hidden relationships between engineering variables can be identified. This is especially useful in applications such as fault detection, signal processing, image recognition, robotics, power system monitoring, and structural health assessment. Important linear algebra concepts such as vector spaces, matrix rank, orthogonality, projections, eigenvalues, eigenvectors, matrix decomposition, and basis transformation are widely used in engineering data analysis. Vector spaces help organize high-dimensional data, while matrix rank provides information about feature dependency and redundancy. Orthogonality and projection methods are used to separate useful information from noise. Eigenvalues and eigenvectors help identify dominant patterns and important directions of variation in data. Similarly, matrix decomposition techniques such as Principal Component Analysis and Singular Value Decomposition help reduce dimensionality and extract meaningful features from complex datasets.

In engineering applications, high-dimensional datasets often contain redundant, correlated, or noisy features. Linear algebra-based methods help reduce these problems by transforming the original feature space into a new optimized space. For example, Principal Component Analysis uses eigenvalue decomposition to transform correlated features into independent principal components [5]. These principal components preserve the most important variance in the dataset while removing less useful information. Similarly, Singular Value Decomposition decomposes a data matrix into smaller components that capture dominant structures and reduce noise. As a result, the transformed dataset becomes more suitable for machine learning, predictive modeling, and intelligent decision-making. Linear algebra is also highly important in machine learning models used for engineering data analytics. Many machine learning algorithms depend on matrix operations, vector calculations, distance measurements, optimization functions, and transformation techniques. Support Vector Machines use vector spaces and hyperplanes for classification. Neural networks use matrix multiplication to process input data through different layers. Regression models use linear algebra to estimate relationships between input and output variables. Therefore, linear algebra serves as a bridge between mathematical theory and practical machine learning implementation.

Table 1: Role of Linear Algebra Concepts in Engineering Data Representation

Linear Algebra Concept	Function in Data Representation	Engineering Application
Vectors	Represent individual observations, sensor values, or system states	Sensor readings, machine conditions, robotic positions
Matrices	Organize complete datasets with observations and features	Smart manufacturing data, power system measurements, simulation outputs
Matrix Rank	Identifies feature dependency, redundancy, and data complexity	Feature selection, fault diagnosis, system monitoring
Orthogonality	Separates independent information components and reduces overlap	Signal filtering, noise reduction, feature extraction
Projection	Maps data into lower-dimensional and more meaningful spaces	Dimensionality reduction, visualization, model simplification
Eigenvalues and Eigenvectors	Identify dominant directions, variance patterns, and system behavior	PCA, stability analysis, vibration analysis
Matrix Decomposition	Breaks complex data matrices into meaningful components	SVD, data compression, image processing, noise reduction
Basis Transformation	Converts data into a more useful coordinate or feature system	Robotics, control systems, machine learning preprocessing

The concepts summarized in Table 1 show that linear algebra provides a complete mathematical foundation for organizing and transforming engineering data into meaningful analytical structures. By converting raw engineering measurements into vectors, matrices, and optimized feature spaces, linear algebra enables researchers to identify important data patterns, remove redundant information, and improve computational efficiency [6]. These mathematical operations create a strong connection between raw engineering data and machine learning-based analysis. Therefore, the following figure presents the overall flow of linear algebra-based engineering data representation, showing how engineering data sources are transformed into optimized features for predictive modeling and intelligent decision support.

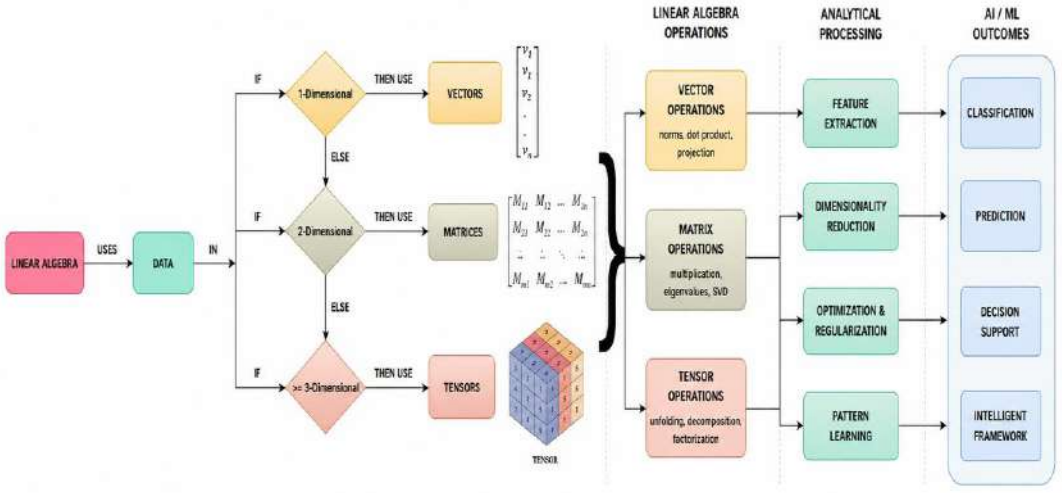

Figure 1: Linear Algebra-Based Engineering Data Representation Framework

The figure 1 shows how engineering data is transformed from raw sensor or system measurements into a structured matrix form. After matrix representation, linear algebra operations are applied to identify important features, remove redundancy, reduce dimensionality, and generate an optimized feature space. This optimized representation is then used for machine learning analysis, predictive modeling, fault detection, and intelligent engineering decision support. Linear algebra provides the foundation for representing and analyzing high-dimensional engineering data in a mathematically structured way. It supports data transformation, dimensionality reduction, feature extraction, noise reduction, and model improvement. By using linear algebraic tools, complex engineering datasets can be converted into meaningful and efficient representations that improve the performance of machine learning models and decision support systems. Therefore, linear algebra is a core component of the proposed mathematical framework for high-dimensional engineering data analysis.

3- Dimensionality Reduction Techniques for High-Dimensional Data:

Dimensionality reduction is one of the most important techniques for handling high-dimensional engineering data because it reduces the number of input features while preserving the most meaningful information from the original dataset. In modern engineering systems, data is often collected from multiple sensors, machines, simulation tools, monitoring platforms, industrial devices, and control systems. As a result, engineering datasets may contain hundreds or thousands of variables. Although these variables provide detailed information about system behavior, they also increase computational complexity and may reduce the performance of machine learning models. High-dimensional data creates several challenges, including feature redundancy, noisy measurements, multicollinearity, overfitting, high memory usage, and increased training time. When too many features are used directly in predictive modeling, the model may learn irrelevant patterns instead of useful information [7]. This problem is commonly known as the curse of dimensionality. In engineering applications, this issue can affect fault detection accuracy, system monitoring reliability, predictive maintenance performance, and decision support quality. Therefore, dimensionality reduction is required to transform high-dimensional data into a smaller and more useful feature space. Mathematically, a high-dimensional engineering dataset can be represented as:

$$X \in R^{m \times n}$$

$$Z \in R^{m \times k}, \quad k < n$$

Here, k represents the reduced number of features or components. The transformation can be expressed as:

$$Z = f(X)$$

Principal Component Analysis is one of the most widely used dimensionality reduction techniques in high-dimensional data analysis. PCA transforms correlated variables into a smaller number of uncorrelated variables called principal components. These components are arranged according to the amount of variance they explain in the dataset. The first principal component captures the highest variance, the second captures the next highest variance, and so on [8]. This makes PCA highly useful for reducing data complexity while preserving important information. Before applying PCA, the dataset is usually standardized to ensure that all features contribute equally to the analysis. Standardization is performed as:

$$X_{std} = \frac{X - \mu}{\sigma}$$

$$C = \frac{1}{m-1} X_{std}^{T} X_{std}$$

$$Cw_i = \lambda_i w_i$$

where w_i is the eigenvector and λ_i is the corresponding eigenvalue. Eigenvectors define the direction of principal components, while eigenvalues represent the amount of variance captured by each component. The reduced feature matrix is obtained by selecting the top k eigenvectors:

$$Z = X_{std} W_k$$

where W_k contains the selected eigenvectors. The variance preserved by the selected components is calculated as:

$$Explained\ Variance = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \times 100$$

This equation helps determine how much information is retained after dimensionality reduction. For example, if the selected components preserve more than 95% of the variance, the reduced dataset can be considered highly informative while being much smaller than the original dataset. In engineering applications, PCA is widely used for fault detection, system monitoring, vibration analysis, image compression, signal processing, sensor data analysis, and predictive maintenance. In fault detection systems, PCA can identify abnormal patterns by separating normal operational behavior from fault-related variations. In vibration analysis, PCA reduces complex sensor signals into important components that represent mechanical system behavior [9]. In image processing, PCA helps compress high-dimensional pixel information while preserving essential image features. Therefore, PCA provides an effective balance between computational efficiency and information preservation. Another important dimensionality reduction method is Linear Discriminant Analysis. Unlike PCA, which focuses on maximizing variance, Linear Discriminant Analysis focuses on maximizing class separability. It is particularly useful for classification problems where the target classes are already known. In engineering fault diagnosis, for example, LDA can separate normal and faulty operating conditions by finding projection directions that maximize the distance between classes. The objective of LDA is to maximize the ratio between between-class scatter and within-class scatter:

$$J(W) = \frac{|W^T S_B W|}{|W^T S_W W|}$$

where S_B represents the between-class scatter matrix and S_W represents the within-class scatter matrix. The between-class scatter matrix measures the separation between different classes, while the within-class scatter matrix measures the spread within each class. By maximizing this ratio, LDA creates a reduced feature space that improves classification performance. Feature selection is another dimensionality reduction approach. Unlike PCA and LDA, which create new transformed features, feature selection chooses the most relevant original features from the dataset. This is useful when interpretability is important because selected features retain their original meaning. Feature selection methods can be divided into filter methods, wrapper methods, and embedded methods [10]. Filter methods use statistical measures such as correlation, mutual information, and variance thresholding. Wrapper methods evaluate feature subsets using machine learning models. Embedded methods perform feature selection during model training, such as Lasso regularization. Feature selection can be mathematically expressed as:

$$X_s = X[:, S]$$

$$S = \underset{S \subseteq F}{\operatorname{argmax}} Performance(S)$$

Manifold learning methods are also used for nonlinear dimensionality reduction. These methods are useful when high-dimensional engineering data lies on a lower-dimensional nonlinear structure. Common manifold learning techniques include t-SNE, Isomap, and Locally Linear Embedding. These methods are often used for visualization, pattern discovery, and clustering. However, they may be computationally expensive and less interpretable than PCA or LDA. The

importance of dimensionality reduction in engineering data analysis is summarized in Table 2.

Table 2: Dimensionality Reduction Techniques for High-Dimensional Engineering Data

Technique	Mathematical Objective	Main Function
Principal Component Analysis	$Z = XW_k$	Converts correlated features into uncorrelated principal components
Linear Discriminant Analysis	(J(W)=\fraction	W^T S_B W
Feature Selection	$X_s = X[:, S]$	Selects the most important original features
Singular Value Decomposition	$X = U\Sigma V^T$	Reduces data rank and extracts dominant patterns
Manifold Learning	$Z = f_{nonlinear}(X)$	Finds nonlinear low-dimensional structures
Autoencoders	$Loss = \| x - \hat{x} \|^2$	Learns compressed nonlinear feature representation

The techniques summarized in Table 2 show that dimensionality reduction can be achieved through both linear and nonlinear methods. Linear methods such as PCA, LDA, and SVD are widely used because they are mathematically transparent, computationally efficient, and easier to interpret. Nonlinear methods such as manifold learning and autoencoders are useful when engineering data contains complex nonlinear patterns. However, the selection of a dimensionality reduction method depends on the nature of the dataset, the purpose of analysis, the need for interpretability, and the computational resources available.

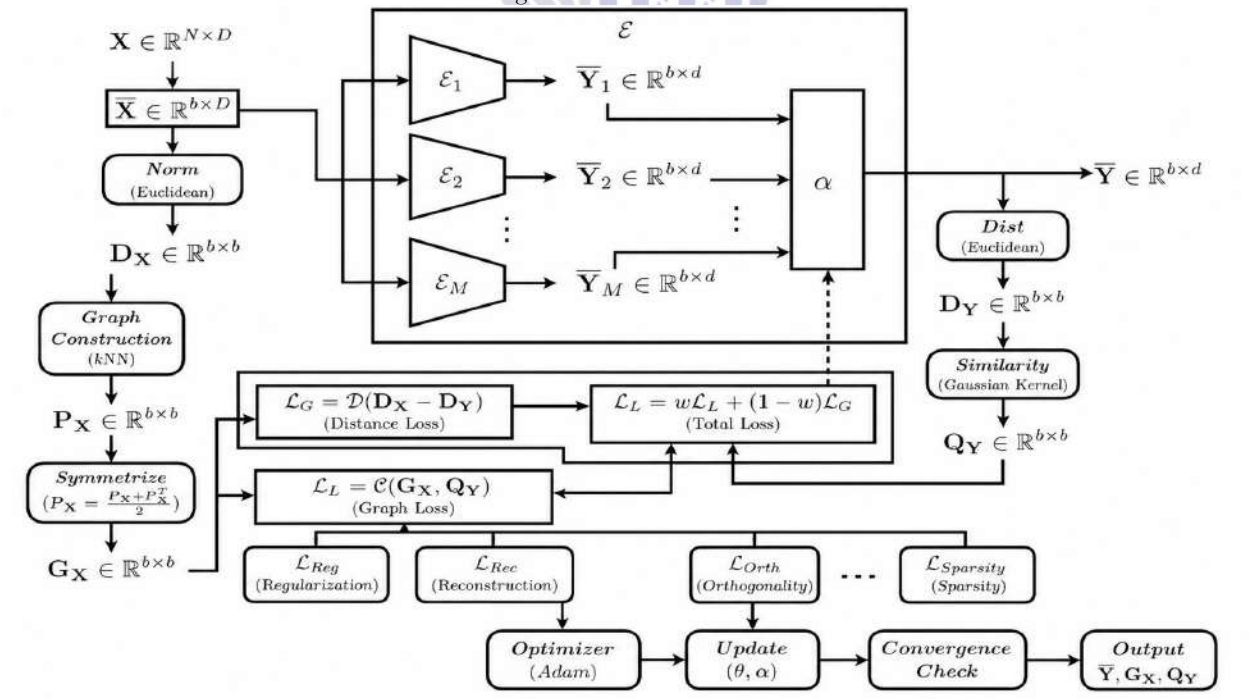

Figure 2: Dimensionality Reduction Workflow for High-Dimensional Engineering Data

Figure 2 shows the general workflow of dimensionality reduction for high-dimensional engineering data. The process begins with raw engineering data collected from sensors, signals, images, machines, and monitoring systems. After preprocessing, dimensionality reduction methods are applied to transform the original high-dimensional dataset into a smaller and more informative feature space [11]. The reduced features are then used for machine learning modeling, which improves prediction accuracy, classification reliability, fault detection performance, and engineering decision-making. Dimensionality reduction also improves computational efficiency. When the number of features is reduced from n to k, the computational cost of model training decreases significantly. This reduction can be represented as [12]:

$$Reduction\ Rate = \frac{n-k}{n} \times 100$$

$$Reconstruction\ Error = \| X - \hat{X} \|^2$$

where X is the original dataset and $\hat{X}$ is the reconstructed dataset from the reduced representation. A lower reconstruction error indicates that the reduced features preserve the original data structure more effectively. Dimensionality reduction is a critical step in high-dimensional engineering data analysis. It reduces unnecessary features, removes noise, controls overfitting, improves computational speed, and enhances machine learning performance. Linear algebra-based methods such as PCA, SVD, and LDA remain highly important because they are efficient, interpretable, and mathematically well-established. At the same time, nonlinear methods such as manifold learning and autoencoders provide additional capabilities for complex engineering datasets. Therefore, dimensionality reduction plays a central role in the proposed mathematical framework for high-dimensional engineering data analysis and machine learning-based intelligent decision support systems.

4- Methodology:

The methodology of this study is designed to develop a structured mathematical framework for analyzing high-dimensional engineering data using advanced linear algebra and optimization techniques. The overall process begins with the acquisition of engineering datasets from sources such as sensors, simulation models, industrial monitoring systems, power systems, robotics platforms, and smart manufacturing environments. These datasets are first arranged in matrix form, where rows represent observations and columns represent engineering features. Before mathematical analysis, the data is cleaned, normalized, and standardized to remove missing values, noise, inconsistent measurements, and scale-related differences. This preprocessing step ensures that all engineering variables contribute fairly to the mathematical transformation and machine learning process [13]. After preprocessing, the methodology applies linear algebra-based dimensionality reduction, matrix decomposition, and optimization methods to convert the original high-dimensional dataset into a reduced and more meaningful feature space. Principal Component Analysis is used to identify dominant variance directions and remove redundant features, while Singular Value Decomposition helps extract important structural patterns from the data matrix. Optimization techniques such as least-squares minimization, gradient-based optimization, and regularization are then applied to improve predictive modeling performance and reduce overfitting. The optimized feature space is finally used with machine learning models for classification, regression, fault detection, prediction, and intelligent engineering decision support.

4.1- Dimensionality Reduction Using Principal Component Analysis:

Principal Component Analysis is applied in this study as a major linear algebra-based dimensionality reduction technique for reducing the complexity of high-dimensional engineering datasets. In modern engineering systems, data is often collected from sensors, machines, simulation platforms, structural monitoring systems, robotic devices, power networks, and smart manufacturing environments. These

datasets usually contain a large number of correlated variables such as temperature, vibration, voltage, current, pressure, speed, displacement, fault indicators, and system response values. When all these features are directly used in machine learning models, they may increase computational cost, introduce redundancy, cause overfitting, and reduce model interpretability. Therefore, PCA is used to transform the original high-dimensional dataset into a smaller number of meaningful principal components while preserving the maximum amount of useful information.

The original engineering dataset is represented as a matrix:

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & x_{m3} & \cdots & x_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

where m represents the number of engineering observations and n represents the number of input features. Each row of the matrix represents one engineering sample, while each column represents one measured variable [14]. Since this study focuses on high-dimensional engineering data, the number of features n may be very large. The main objective of PCA is to transform the original data matrix X into a reduced-dimensional matrix Z, expressed as:

$$Z \in \mathbb{R}^{m \times k}, \quad k < n$$

where k is the selected number of principal components. The transformation is performed in such a way that the reduced matrix Z preserves the maximum variance and dominant structural information from the original dataset. Before applying PCA, the dataset is standardized because engineering variables are often measured in different units and scales [15]. For example, voltage may be measured in volts, temperature in degrees Celsius, vibration in meters per second squared, and pressure in pascals. If these variables are not standardized, features with larger numerical ranges may dominate the PCA transformation. Therefore, each feature is converted into a standardized form using:

$$x_{ij}^{std} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

where x_{ij} is the original value of the j^{th} feature in the i^{th} observation, μ_j is the mean of the j^{th} feature, and σ_j is the standard deviation of the j^{th} feature. The mean and standard deviation are calculated as:

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij} \quad \sigma_j = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(x_{ij} - \mu_j)^2}$$

The standardized data matrix is represented as:

$$X_{std} = \begin{bmatrix} \frac{x_{11} - \mu_1}{\sigma_1} & \frac{x_{12} - \mu_2}{\sigma_2} & \cdots & \frac{x_{1n} - \mu_n}{\sigma_n} \\ \frac{x_{21} - \mu_1}{\sigma_1} & \frac{x_{22} - \mu_2}{\sigma_2} & \cdots & \frac{x_{2n} - \mu_n}{\sigma_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{x_{m1} - \mu_1}{\sigma_1} & \frac{x_{m2} - \mu_2}{\sigma_2} & \cdots & \frac{x_{mn} - \mu_n}{\sigma_n} \end{bmatrix}$$

After standardization, the covariance matrix is computed to measure the relationship between different engineering features. The covariance matrix is given as:

$$C = \frac{1}{m-1} X_{std}^T X_{std}$$

The covariance matrix can also be written in expanded form as [16]:

$$C = \begin{bmatrix} cov(x_1, x_1) & cov(x_1, x_2) & \cdots & cov(x_1, x_n) \\ cov(x_2, x_1) & cov(x_2, x_2) & \cdots & cov(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n, x_1) & cov(x_n, x_2) & \cdots & cov(x_n, x_n) \end{bmatrix}$$

where the covariance between two features x_p and x_q is calculated as:

$$cov(x_p, x_q) = \frac{1}{m-1} \sum_{i=1}^{m} (x_{ip} - \mu_p)(x_{iq} - \mu_q)$$

The covariance matrix helps identify whether engineering variables contain similar, independent, or redundant information. A high covariance value indicates that two variables change together, while a low covariance value indicates weak dependency between variables. Since high-dimensional engineering datasets often contain strongly correlated variables, PCA uses this covariance structure to identify the most important directions of variation. The next step is eigenvalue decomposition of the covariance matrix. The eigenvalue problem is defined as [17]:

$$Cw_i = \lambda_i w_i$$

where w_i represents the eigenvector and λ_i represents the corresponding eigenvalue. The eigenvectors define the directions of the principal components, while the eigenvalues indicate the amount of variance captured by those directions. The complete eigenvalue decomposition can be expressed as:

$$CW = W\Lambda$$

$$W = [w_1, w_2, w_3, \ldots, w_n]$$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

The eigenvalues are arranged in descending order:

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n$$

The first eigenvalue λ_1 corresponds to the first principal component, which captures the maximum variance in the dataset. The second eigenvalue λ_2 corresponds to the second principal component, which captures the next highest variance, and so on. The selected eigenvectors corresponding to the top k eigenvalues are used to form the projection matrix:

$$W_k = [w_1, w_2, w_3, \ldots, w_k]$$

The standardized dataset is then projected into the reduced-dimensional feature space as:

$$Z = X_{std} W_k$$

where Z is the PCA-transformed engineering dataset. In expanded form, this transformation can be written as:

$$Z = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1k} \\ z_{21} & z_{22} & \cdots & z_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ z_{m1} & z_{m2} & \cdots & z_{mk} \end{bmatrix} = \begin{bmatrix} x_{11}^{std} & x_{12}^{std} & \cdots & x_{1n}^{std} \\ x_{21}^{std} & x_{22}^{std} & \cdots & x_{2n}^{std} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1}^{std} & x_{m2}^{std} & \cdots & x_{mn}^{std} \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1k} \\ w_{21} & w_{22} & \cdots & w_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nk} \end{bmatrix}$$

The number of selected principal components is determined by the explained variance ratio. The explained variance ratio for the first k principal components is calculated as:

$$EVR_k = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

$$EVR_k(\%) = \left(\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \right) \times 100$$

The cumulative explained variance is used to decide how many principal components should be retained [18]:

$$CEV(k) = \sum_{i=1}^{k} \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$
$$CEV(k) \geq \tau$$

where τ is the desired variance threshold, commonly selected as 0.90, 0.95, or 0.98 depending on the engineering application. In this study, PCA is used to retain the maximum possible variance while removing unnecessary and redundant dimensions. The mathematical process of PCA used in this study is summarized in Table 3.

Table 3: Principal Component Analysis Procedure for High-Dimensional Engineering Data

PCA Stage	Mathematical Expression	Purpose in Engineering Data Analysis
Original Data Representation	$X \in \mathbb{R}^{m \times n}$	Represents high-dimensional engineering data with observations and features
Feature Standardization	$x_{ij}^{std} = \frac{x_{ij} - \mu_j}{\sigma_j}$	Removes scale differences among engineering variables
Mean Calculation	$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_{ij}$	Computes average value of each engineering feature
Standard Deviation	$\sigma_j = \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (x_{ij} - \mu_j)^2}$	Measures spread of each engineering feature
Covariance Matrix	$C = \frac{1}{m-1} X_{std}^T X_{std}$	Identifies relationships and redundancy among variables
Eigenvalue Decomposition	$Cw_i = \lambda_i w_i$	Finds principal directions of maximum variance
Projection Matrix	$W_k = [w_1, w_2, \ldots, w_k]$	Selects top principal components
Reduced Feature Space	$Z = X_{std} W_k$	Converts high-dimensional data into lower-dimensional representation
Explained Variance Ratio	$EVR_k = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$	Measures preserved information after reduction
Reconstruction Error	$E_{rec} = \| X_{std} - ZW_k^T \|_F^2$	Evaluates loss of information after dimensionality reduction

The PCA stages summarized in Table 3 show that dimensionality reduction is not only a feature reduction process but also a complete mathematical transformation of engineering data. The method begins with data standardization and covariance analysis, then identifies dominant variance directions through eigenvalue decomposition. After selecting the most informative principal components, the original high-dimensional data is projected into a reduced feature space. This reduced representation preserves the most important system behavior while removing redundant and noisy variables [19]. Therefore, PCA improves both computational efficiency and predictive reliability in high-dimensional engineering data analysis.

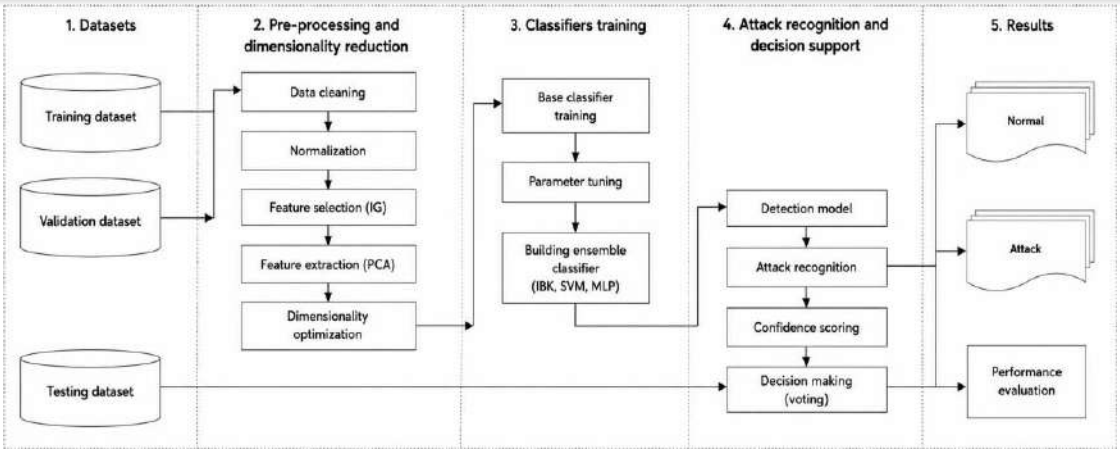

Figure 3: PCA-Based Dimensionality Reduction Framework for Engineering Data

Figure 3 presents the PCA-based dimensionality reduction framework used in the proposed methodology. The process starts with high-dimensional engineering data collected from sensors, signals, machines, and system variables. The data is standardized to remove scale differences, after which the covariance matrix is calculated to identify relationships among variables. Eigenvalue decomposition is then performed to find the dominant directions of maximum variance. The most informative principal components are selected according to the explained variance ratio, and the original dataset is projected into a reduced-dimensional feature space. Finally, the optimized feature matrix is used as input for machine learning models. In engineering applications, the PCA-reduced feature space improves model performance by eliminating redundant variables and reducing noise [20]. For example, in fault diagnosis, PCA helps separate normal operating patterns from abnormal fault-related patterns. In vibration analysis, PCA extracts the dominant vibration modes from noisy sensor signals. In power system monitoring, PCA reduces complex electrical measurements into key components that represent system behavior. In robotics, PCA helps simplify sensor fusion and motion-related feature data. In smart manufacturing, PCA improves predictive maintenance by reducing high-dimensional production variables into meaningful operational components [21]. The PCA-based transformation also supports better computational performance. Since the machine learning model uses k principal components instead of n original features, the training time and memory requirements are reduced. The computational benefit can be expressed as:

$$Computational\ Gain = \left(1 - \frac{k}{n}\right) \times 100$$

This equation shows that reducing the number of input features directly improves computational efficiency. However, the selected value of k must maintain a balance between dimensionality reduction and information preservation. If too few components are selected, important engineering patterns may be lost. If too many components are selected, the dimensionality reduction benefit becomes limited. In this study, PCA is therefore used as a core methodological step to improve high-dimensional engineering data analysis. It transforms complex datasets into compact, interpretable, and mathematically optimized representations [22]. By preserving dominant variance directions and removing irrelevant dimensions, PCA improves machine learning accuracy, reduces overfitting, decreases computational cost, and supports intelligent decision-making. The PCA-transformed dataset is further used in later stages of the proposed framework for optimization-based feature transformation, predictive modeling,

classification, regression, fault detection, and engineering decision support.

4.2- Singular Value Decomposition for Matrix Factorization:

Singular Value Decomposition is applied in this study as a powerful linear algebra-based matrix factorization technique for analyzing high-dimensional engineering data. In engineering systems, datasets collected from sensors, machines, simulation models, power networks, robotic platforms, vibration signals, image-processing systems, and structural monitoring devices are usually large, noisy, correlated, and complex. These datasets often contain hidden patterns that are difficult to identify directly from the original feature matrix. Therefore, Singular Value Decomposition is used to decompose the original data matrix into simpler mathematical components that represent the most important structural information of the dataset. SVD is especially useful in high-dimensional engineering data analysis because it supports feature extraction, dimensionality reduction, noise reduction, low-rank approximation, data compression, and model stability improvement [23]. Unlike simple feature reduction methods, SVD decomposes the entire data matrix into orthogonal directions and singular values, allowing the dominant information patterns to be separated from weak, noisy, or redundant components. This makes SVD highly suitable for engineering applications such as signal processing, vibration analysis, fault diagnosis, image compression, power system monitoring, robotics, and predictive maintenance.

The full SVD form can be written as:

$$X = [u_1 \quad u_2 \quad \cdots \quad u_m] \begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{bmatrix}$$

where $\sigma_1, \sigma_2, \ldots, \sigma_r$ are singular values and $r = rank(X)$. The singular values are arranged in descending order:

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots \geq \sigma_r > 0$$

The largest singular values represent the most important patterns in the engineering dataset, while smaller singular values often represent noise, weak variation, or redundant information. Therefore, by keeping only the largest singular values, the dataset can be approximated using fewer components. The relationship between SVD and eigenvalue decomposition is given by:

$$X^T X = V \Sigma^T \Sigma V^T$$
$$X X^T = U \Sigma \Sigma^T U^T$$

This means that the right singular vectors V are eigenvectors of $X^T X$, while the left singular vectors U are eigenvectors of XX^T. The singular values are related to the eigenvalues by [24]:

$$\sigma_i = \sqrt{\lambda_i}$$

where λ_i represents the eigenvalue of $X^T X$ or XX^T. This relationship shows that SVD provides a stable and generalized matrix decomposition approach that can be applied even when the original matrix is rectangular and not square. For high-dimensional engineering data, the full SVD may contain many components. However, not all components are equally useful. Therefore, a reduced-rank approximation is used by selecting the top k singular values and their corresponding singular vectors. The reduced SVD form is expressed as [25]:

$$X_k = U_k \Sigma_k V_k^T$$
$$U_k \in \mathbb{R}^{m \times k}, \quad \Sigma_k \in \mathbb{R}^{k \times k}, \quad V_k^T \in \mathbb{R}^{k \times n}$$
$$k < r \leq \min(m, n)$$

The reduced-rank approximation expands as:

$$X_k = \sum_{i=1}^{k} \sigma_i \, u_i v_i^T$$

The information preserved by the top k singular values is calculated as:

$$Energy_k = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}$$

$$Energy_k(\%) = \left(\frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \right) \times 100$$

A suitable number of singular components is selected when:

$$\frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \geq \tau$$

where τ is the selected energy preservation threshold. In engineering applications, τ is commonly selected as 0.90, 0.95, or 0.98 depending on the required balance between compression and information preservation [26].

SVD can also be viewed as an optimization problem. The rank-k approximation of X is the best low-rank approximation in terms of Frobenius norm:

$$X_k = \arg \min_{\tilde{X}: rank(\tilde{X}) \leq k} \| X - \tilde{X} \|_F$$

This result is known as the Eckart–Young theorem. It shows that among all possible matrices with rank at most k, the SVD-based approximation X_k produces the minimum reconstruction error. This property is highly valuable in engineering data analysis because it ensures that the reduced data matrix is mathematically optimal under the low-rank constraint.

A more complex optimization formulation can be written as:

$$\min_{U_k, \Sigma_k, V_k} \| X - U_k \Sigma_k V_k^T \|_F^2$$

$$U_k^T U_k = I_k, \quad V_k^T V_k = I_k, \quad \Sigma_k = diag(\sigma_1, \sigma_2, \ldots, \sigma_k)$$

where I_k is the identity matrix of size k. This constrained optimization problem ensures that the left and right singular vectors remain orthogonal and that the singular values are arranged in decreasing order. The transformation process can be further represented as [27]:

$$Z_{SVD} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1k} \\ z_{21} & z_{22} & \cdots & z_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ z_{m1} & z_{m2} & \cdots & z_{mk} \end{bmatrix} = \begin{bmatrix} x_{11}^{std} & x_{12}^{std} & \cdots & x_{1n}^{std} \\ x_{21}^{std} & x_{22}^{std} & \cdots & x_{2n}^{std} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1}^{std} & x_{m2}^{std} & \cdots & x_{mn}^{std} \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1k} \\ v_{21} & v_{22} & \cdots & v_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \cdots & v_{nk} \end{bmatrix}$$

This equation shows how the high-dimensional engineering data matrix is projected onto the dominant right singular vector space. The feature compression ratio obtained through SVD is calculated as:

$$CR_{SVD} = \frac{n - k}{n} \times 100$$

$$CG_{SVD} = \left(1 - \frac{k(m + n + 1)}{mn} \right) \times 100$$

This equation compares the storage requirement of the reduced SVD representation with the original full data matrix. Since the original matrix requires mn storage units, while the reduced SVD representation requires approximately $mk + k + nk$ units, SVD can significantly reduce memory requirements for large-scale engineering datasets. The SVD-based feature transformation procedure used in this study is summarized in Table 4.

Table 4: Singular Value Decomposition Procedure for High-Dimensional Engineering Data

SVD Stage	Mathematical Expression	Purpose in Engineering Data Analysis
Original Data Matrix	$X \in \mathbb{R}^{m \times n}$	Represents engineering observations and features
Standardized Data Matrix	$X_{std} = \frac{X - \mu}{\sigma}$	Removes scale differences among engineering variables
Full SVD Factorization	$X_{std} = U \Sigma V^T$	Decomposes the dataset into orthogonal components
Singular Value Ordering	$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$	Ranks components according to importance
Reduced-Rank Approximation	$X_k = U_k \Sigma_k V_k^T$	Keeps only dominant structural patterns
Component Expansion	$X_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$	Represents reduced data using top singular components
SVD Feature Matrix	$Z_{SVD} = U_k \Sigma_k$	Generates optimized features for machine learning
Energy Preservation	$Energy_k = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}$	Measures retained information after reduction
Reconstruction Error	$\| X - X_k \|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2$	Measures information loss after approximation
Optimization Objective	$\text{argmin}_{rank(\tilde{X}) \leq k} \| X - \tilde{X} \|_F$	Ensures best low-rank approximation
Compression Ratio	$CR_{SVD} = \frac{n - k}{n} \times 100$	Measures dimensionality reduction percentage

The stages summarized in Table 4 show that SVD provides a systematic process for decomposing high-dimensional engineering data into meaningful and lower-dimensional components. The process begins with standardized data representation and full matrix decomposition. After this, the most important singular values and singular vectors are selected to form a reduced-rank approximation. This approximation preserves dominant system behavior while removing weak, redundant, and noisy components. As a result, the SVD-transformed feature matrix becomes more compact, stable, and suitable for machine learning-based predictive modeling.

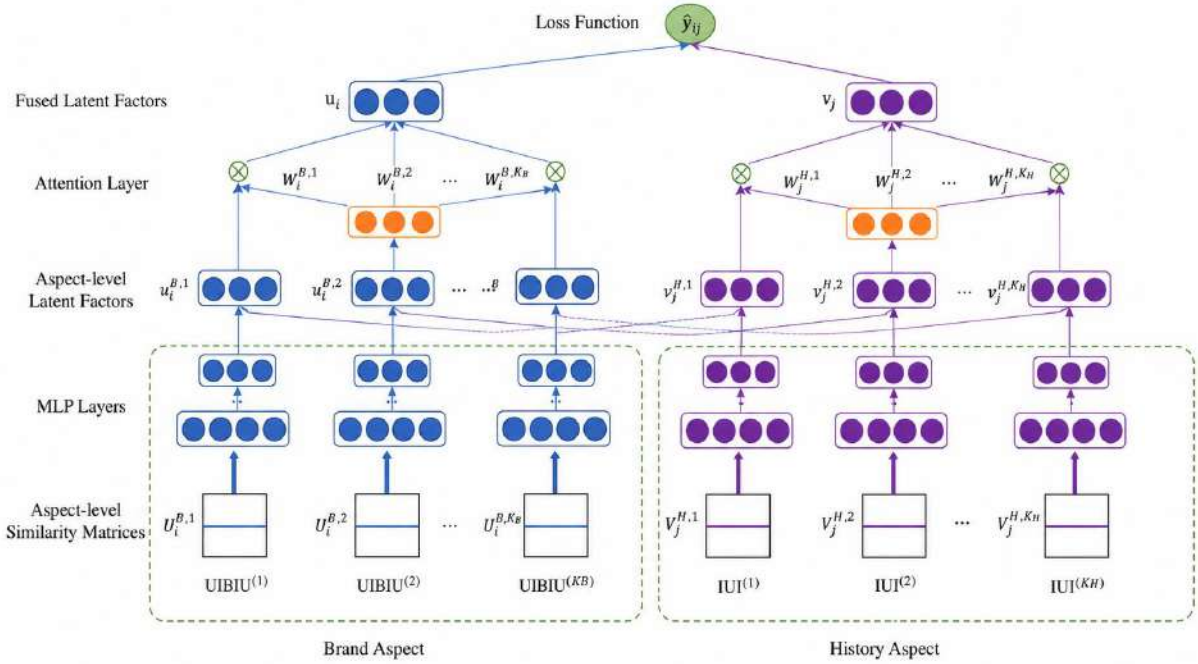

Figure 4: SVD-Based Matrix Factorization Framework for Engineering Data

Figure 4 illustrates the SVD-based matrix factorization workflow used in the proposed methodology. The process begins with high-dimensional engineering data collected from sensors, images, power measurements, structural systems, and robotic platforms. The dataset is standardized to remove scale differences among variables. After standardization, SVD decomposes the data matrix into observation patterns, singular values, and feature directions [28]. The singular values are ranked according to their importance, and the top k components are selected to construct a reduced-rank approximation. The resulting SVD-optimized feature matrix is then used for machine learning analysis, predictive modeling, fault detection, and intelligent decision-making.

$$\theta^* = \operatorname{argmin}_{\theta} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(z_i;\theta)\right) + \lambda \Omega(\theta) \right]$$

where $\mathcal{L}$ is the loss function, $\Omega(\theta)$ is the regularization term, and λ is the regularization coefficient. Since Z_{SVD} is lower-dimensional and less noisy than the original data matrix, the predictive model becomes more stable, efficient, and generalizable. A combined SVD-regularized reconstruction and prediction objective can be expressed as:

$$\min_{U_k,\Sigma_k,V_k,\theta} \left[\alpha \parallel X - U_k \Sigma_k V_k^T \parallel_F^2 + \beta \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(u_i \Sigma_k;\theta)\right) + \lambda \parallel \theta \parallel_2^2 \right]$$

$$U_k^T U_k = I_k, \quad V_k^T V_k = I_k, \quad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$$

where α, β, and λ control the balance between reconstruction accuracy, prediction performance, and regularization strength. This complex formulation shows how SVD-based dimensionality reduction can be integrated with machine learning optimization in the proposed mathematical framework. Singular Value Decomposition provides a strong mathematical foundation for matrix factorization in high-dimensional engineering data analysis. It decomposes complex datasets into orthogonal components, ranks information using singular

values, removes noisy and redundant dimensions, and constructs a compact reduced-rank representation. By using SVD, the proposed framework improves computational efficiency, data compression, noise filtering, feature extraction, predictive accuracy, and intelligent engineering decision support. Therefore, SVD serves as a critical methodological component in the proposed mathematical framework for high-dimensional engineering data analysis using advanced linear algebra and optimization techniques.

4.3- Optimization-Based Feature Transformation:

Optimization-based feature transformation is an important stage of the proposed methodology because it improves the quality, relevance, and predictive strength of the reduced engineering feature space. After applying Principal Component Analysis and Singular Value Decomposition, the original high-dimensional data is converted into a compact representation. However, dimensionality reduction alone may not always produce the most suitable feature space for prediction, classification, fault detection, or decision support. Therefore, mathematical optimization is applied to refine the transformed features, reduce prediction error, remove unnecessary variations, and improve the learning ability of machine learning models. In high-dimensional engineering data analysis, optimization helps identify the best feature representation by minimizing a defined objective function [29]. This objective function usually measures the difference between actual engineering outputs and predicted outputs. Engineering datasets collected from sensors, machines, power systems, robotic platforms, and simulation models often contain noise, redundancy, nonlinear patterns, and irrelevant variables. Optimization-based transformation improves these datasets by adjusting feature weights, selecting informative components, and reducing the effect of weak or noisy variables. As a result, the final feature space becomes more accurate, stable, and suitable for machine learning-based intelligent decision-making. A complete optimization objective for feature transformation and predictive modeling can be written as:

$$\theta^* = \arg\min_{\theta} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(z_i;\theta)\right) + \lambda \Omega(\theta) \right]$$

where y_i is the actual output, $f(z_i;\theta)$ is the predicted output for the i^{th} sample, $\mathcal{L}$ is the loss function, $\Omega(\theta)$ is the regularization term, and λ controls the strength of regularization. This equation ensures that the model reduces prediction error while avoiding overfitting.

For regression-based engineering prediction, the loss function is commonly defined using Mean Squared Error. The optimization objective becomes:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{k} \theta_j^2$$

where y_i is the actual engineering response, $\hat{y}_i$ is the predicted response, θ_j represents model coefficients, and $\lambda \sum_{j=1}^{k} \theta_j^2$ is the L2 regularization term. This objective function is useful in engineering applications such as load forecasting, vibration prediction, fault severity estimation, structural response prediction, and process optimization. For classification-based engineering tasks, such as fault detection or abnormal condition identification, the optimization objective can be written using cross-entropy loss:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}) + \lambda \Omega(\theta)$$

A complete feature transformation optimization objective can be expressed as:

$$A^*, \theta^* = \underset{A,\theta}{\operatorname{argmin}} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(z_i A; \theta)\right) + \alpha \parallel A \parallel_F^2 + \lambda \parallel \theta \parallel_2^2 \right]$$

where A^* is the optimal transformation matrix, θ^* represents the optimal model parameters, $\parallel A \parallel_F^2$ controls the complexity of the transformation matrix, $\parallel \theta \parallel_2^2$ controls model complexity, and α and λ are regularization coefficients. This equation shows that the optimized feature transformation is learned jointly with the predictive model. Gradient-based optimization is used to update model parameters iteratively. The parameter update rule is given as [30]:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \nabla_\theta J\left(\theta^{(t)}\right)$$

where $\theta^{(t)}$ represents the parameter value at iteration t, η is the learning rate, and $\nabla_\theta J\left(\theta^{(t)}\right)$ is the gradient of the objective function. This update rule moves the parameters in the direction that reduces the objective function and improves model performance. For transformation matrix optimization, the update rule can be written as:

$$A^{(t+1)} = A^{(t)} - \eta \nabla_A \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(z_i A; \theta)\right) + \alpha \parallel A \parallel_F^2 \right]$$

This equation updates the feature transformation matrix so that the transformed feature space becomes more useful for prediction, classification, and decision support. In some engineering applications, feature transformation is also performed using constrained optimization. Constraints are important when engineering systems have physical limits, safety requirements, operational boundaries, or design restrictions. A constrained optimization problem can be expressed as:

$$\begin{aligned} \min_{A,\theta} \quad & \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(z_i A; \theta)\right) + \lambda \Omega(\theta) \\ \text{subject to} \quad & g_r(A, \theta) \le 0, \quad r = 1,2, \dots, R \\ & h_s(A, \theta) = 0, \quad s = 1,2, \dots, S \end{aligned}$$

where $g_r(A, \theta)$ represents inequality constraints and $h_s(A, \theta)$ represents equality constraints. These constraints ensure that the optimized feature transformation remains valid for real-world engineering conditions.

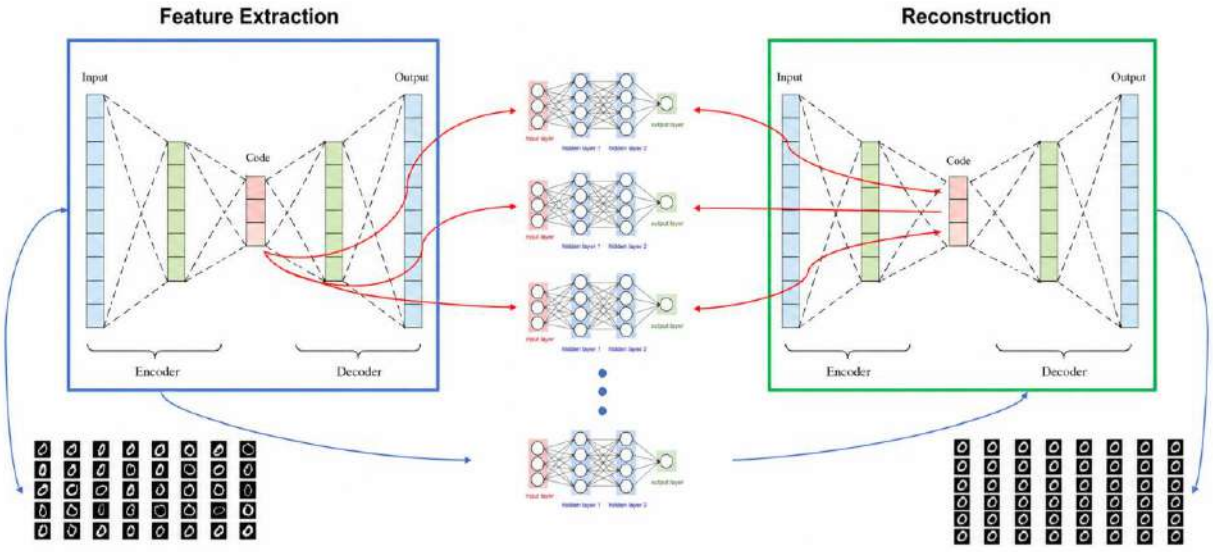

Figure 5: Optimization-Based Feature Transformation Framework

Figure 5 illustrates the optimization-based feature transformation framework used in the proposed methodology. The process begins with the reduced feature matrix obtained from PCA or SVD. A transformation matrix is then learned to convert this reduced feature space into a more optimized and task-specific representation. An objective function is defined by combining prediction loss, regularization, and engineering constraints. Optimization algorithms such as gradient descent, least-squares optimization, convex optimization, and constrained optimization are then applied to identify the best transformation parameters. The optimized feature representation is finally used as input for machine learning models and intelligent engineering decision support. Optimization-based feature transformation also improves computational efficiency because the model operates on a refined feature matrix rather than the original high-dimensional dataset. The improvement in feature efficiency can be calculated as:

$$FEI = \left(1 - \frac{q}{n}\right) \times 100$$

The final optimized prediction process can be expressed as:

$$\hat{Y} = f(H^*; \theta^*) = f(ZA^*; \theta^*)$$

where H^* represents the optimized feature matrix, A^* is the learned transformation matrix, and θ^* represents the optimized model parameters. This equation shows that prediction is performed using mathematically refined features rather than raw high-dimensional inputs. Optimization-based feature transformation plays a central role in the proposed mathematical framework. It improves the reduced feature space generated by PCA and SVD, minimizes prediction error, controls overfitting, supports engineering constraints, and produces a more meaningful representation for machine learning models. By integrating optimization into feature transformation, the proposed framework improves predictive accuracy, computational performance, model stability, and intelligent decision-making in high-dimensional engineering data analysis.

5.4- Regularization for Overfitting Control:

Regularization is an important methodological step in the proposed mathematical framework because high-dimensional engineering datasets are highly vulnerable to overfitting. Overfitting occurs when a machine learning model learns not only the useful patterns in the training data but also noise, random fluctuations, and irrelevant feature relationships. As a result, the model performs well on training data but poorly on unseen testing data. In engineering applications, this problem is serious because inaccurate prediction may affect fault diagnosis, system monitoring, predictive maintenance, quality control, power system stability, robotic decision-making, and intelligent engineering operations. High-dimensional engineering datasets often contain a large number of input features compared with the number of available observations [31]. These features may be collected from sensors, simulation systems, smart manufacturing machines, power networks, vibration monitoring devices, image-processing systems, and industrial control platforms. When too many features are used without proper control, machine learning models may become too complex and may capture redundant or noisy relationships. Therefore, regularization is applied to control model complexity, reduce overfitting, improve generalization, and produce more stable prediction results. In the proposed framework, regularization is applied after dimensionality reduction and optimization-based feature transformation. The optimized feature matrix can be represented as:

$$H \in \mathbb{R}^{m \times q}$$

The general regularized learning objective can be expressed as:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(h_i; \theta)\right) + \lambda \Omega(\theta) \right]$$

One of the most commonly used regularization techniques is L2 regularization, also known as Ridge regularization. It penalizes large coefficient values by adding the squared magnitude of model parameters to the loss function. The L2-regularized objective is written as:

$$J_{L2}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - h_i^T\theta\right)^2 + \lambda\sum_{j=1}^{q}\theta_j^2$$

This equation is useful in engineering prediction problems where many features contribute to the output but large coefficients may make the model unstable. L2 regularization reduces coefficient magnitude and improves model robustness. It is highly useful in load forecasting, vibration response prediction, system monitoring, structural behavior estimation, and industrial process modeling. Another important technique is L1 regularization, also called Lasso regularization. Unlike L2 regularization, L1 regularization adds the absolute values of coefficients to the objective function. This can reduce some coefficients exactly to zero, which means it performs automatic feature selection. The L1-regularized objective is expressed as:

$$J_{L1}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - h_i^T\theta\right)^2 + \lambda\sum_{j=1}^{q}|\theta_j|$$

L1 regularization is useful when only a small number of optimized features are strongly related to the engineering output. In sensor-based monitoring, for example, L1 regularization can help identify the most important sensor features while ignoring weak or redundant variables. This improves model interpretability and reduces computational burden. Elastic Net regularization combines both L1 and L2 penalties [32]. It is especially useful when engineering datasets contain correlated features. L1 helps select important features, while L2 stabilizes coefficient values. The Elastic Net objective can be written as:

$$J_{EN}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - h_i^T\theta\right)^2 + \lambda_1\sum_{j=1}^{q}|\theta_j| + \lambda_2\sum_{j=1}^{q}\theta_j^2$$

where λ_1 controls the L1 penalty and λ_2 controls the L2 penalty. Elastic Net is highly suitable for high-dimensional engineering datasets because it balances feature selection and coefficient stability. This makes it effective for fault diagnosis, predictive maintenance, power system monitoring, smart manufacturing analytics, and robotics sensor fusion. For classification-based engineering tasks, regularization can also be applied to cross-entropy loss. This is useful when the model is used for normal/faulty condition classification, machine state recognition, sensor anomaly detection, or quality inspection. A regularized classification objective can be expressed as:

$$J_{cls}(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic}) + \lambda \parallel \theta \parallel_2^2$$

where C represents the number of classes, y_{ic} is the actual class label, $\hat{y}_{ic}$ is the predicted probability for class c, and $\parallel \theta \parallel_2^2$ is the L2 regularization term. This equation improves classification reliability by preventing the model from becoming too sensitive to noisy engineering samples. The effect of regularization depends strongly on the selection of the regularization parameter. If λ is too small, the model may still overfit the training data. If λ is too large, the model may become too simple and underfit the data. Therefore, an appropriate value of λ is selected through validation-based tuning. The best value is chosen by comparing model performance on validation data and selecting the value that provides the best balance between training accuracy and testing generalization. Regularization also improves model interpretability. In high-dimensional engineering datasets, many features may contain overlapping or redundant

information. Without regularization, the model may assign large and unstable coefficients to these features [33]. Regularization reduces this instability and makes the learned model more reliable. L1 regularization is particularly useful when interpretability is important because it can remove less useful features from the model. L2 regularization is more useful when all features contribute to the output but coefficient values need to be controlled. In the proposed methodology, regularization is integrated with the optimized feature space generated from PCA, SVD, and optimization-based feature transformation. This means the final machine learning model does not depend directly on raw high-dimensional features. Instead, it uses mathematically transformed and regularized features. This combination reduces dimensionality, controls overfitting, improves prediction accuracy, and increases model stability.

5.5- Machine Learning-Based Predictive Modeling:

Machine learning-based predictive modeling is a major stage of the proposed methodology because it uses the mathematically optimized feature space to generate accurate predictions, classifications, and intelligent engineering decisions. After applying data preprocessing, Principal Component Analysis, Singular Value Decomposition, optimization-based feature transformation, and regularization, the original high-dimensional engineering dataset is converted into a cleaner, lower-dimensional, and more informative representation. This optimized representation is then used as input for machine learning models to improve prediction accuracy, reduce computational cost, and enhance decision-making reliability. A complete regularized predictive learning objective can be written as:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_i, f(h_i;\theta)\right) + \lambda \Omega(\theta) \right]$$

where θ^* represents the optimized model parameters, $\mathcal{L}$ is the selected loss function, $\Omega(\theta)$ is the regularization term, and λ controls the regularization strength. This equation ensures that the model minimizes prediction error while controlling unnecessary model complexity. For regression-based engineering prediction, the objective function can be defined using regularized mean squared error:

$$J_{reg}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left[y_i - \left(\theta_0 + \sum_{j=1}^{q} \theta_j \, h_{ij} \right) \right]^2 + \lambda \sum_{j=1}^{q} \theta_j^2$$

where y_i is the actual engineering output, θ_0 is the bias term, θ_j is the weight of the j^{th} optimized feature, and h_{ij} represents the value of the j^{th} optimized feature for the i^{th} sample. This equation is useful for engineering tasks such as load prediction, temperature estimation, vibration response prediction, structural displacement forecasting, and process performance modeling [34]. For classification-based engineering tasks, such as fault detection, anomaly identification, and system condition classification, the predictive model estimates class probabilities. The softmax-based classification function is expressed as:

$$P(y_i = c \mid h_i; \Theta) = \frac{\exp(\theta_c^T h_i + b_c)}{\sum_{r=1}^{C} \exp\left(\theta_r^T h_i + b_r\right)}$$

where C is the total number of classes, θ_c is the parameter vector for class c, b_c is the class bias term, and $P(y_i = c \mid h_i; \Theta)$ represents the probability that sample i belongs to class c. The final predicted class is obtained as:

$$\hat{y}_i = \arg \max_{c \in \{1,2,\ldots,C\}} P(y_i = c \mid h_i; \Theta)$$

The regularized cross-entropy loss for classification is written as [35]:

$$J_{cls}(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{c=1}^{C} y_{ic} \log\left(\frac{\exp(\theta_c^T h_i + b_c)}{\sum_{r=1}^{C}\exp(\theta_r^T h_i + b_r)}\right) + \lambda\sum_{c=1}^{C} \| \theta_c \|_2^2$$

where y_{ic} is equal to 1 if the actual class of sample i is c, and 0 otherwise. This equation helps the model learn class boundaries while reducing overfitting through regularization. Support Vector Machine is also suitable for machine learning-based predictive modeling, especially in classification-based engineering applications. SVM constructs an optimal separating hyperplane between different classes in the optimized feature space. The hyperplane is represented as:

$$w^T h_i + b = 0$$

$$\min_{w,b,\xi}\left[\frac{1}{2}\| w \|^2 + C\sum_{i=1}^{m}\xi_i\right]$$

$$y_i(w^T\phi(h_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1,2,\dots,m$$

where w is the hyperplane weight vector, b is the bias term, ξ_i is the slack variable, C controls the penalty for misclassification, and $\phi(h_i)$ maps the input feature vector into a higher-dimensional kernel space. This formulation is useful for fault diagnosis, abnormal condition detection, and engineering system classification. Artificial Neural Networks can also be applied to the optimized engineering feature space. A neural network learns nonlinear relationships between input features and output responses through multiple layers. The hidden layer transformation can be written as:

$$a^{(l)} = g\left(W^{(l)}a^{(l-1)} + b^{(l)}\right)$$

where $a^{(l)}$ is the activation output of layer l, $W^{(l)}$ is the weight matrix, $b^{(l)}$ is the bias vector, and $g(\cdot)$ is the nonlinear activation function. For an input feature vector h_i, the complete neural network prediction can be represented as:

$$\hat{y}_i = f(h_i;\Theta) = g_L\left(W^{(L)}g_{L-1}\left(W^{(L-1)}\cdots g_1\left(W^{(1)}h_i + b^{(1)}\right)\cdots + b^{(L-1)}\right) + b^{(L)}\right)$$

This equation shows how optimized features pass through multiple transformation layers to produce a final prediction. Neural networks are useful for nonlinear engineering data, complex system behavior, sensor fusion, and predictive maintenance.

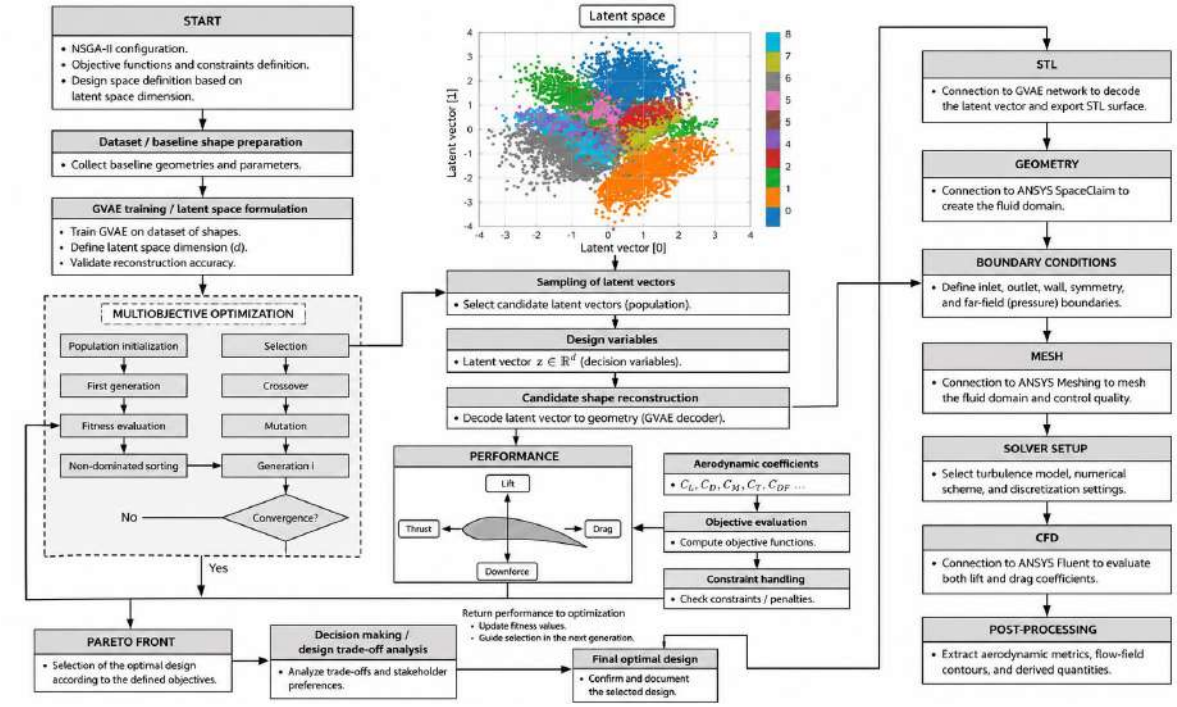

Figure 6: Machine Learning-Based Predictive Modeling Framework

Figure 6 presents the machine learning-based predictive modeling framework used in the proposed methodology. The process begins with the optimized engineering feature matrix generated through dimensionality reduction, matrix factorization, feature transformation, and regularization. The dataset is then divided into training, validation, and testing subsets. Training data is used to learn model parameters, validation data is used for hyperparameter tuning, and testing data is used to evaluate final model performance. After model training and evaluation, the predictive results are used to support engineering decision-making tasks such as fault diagnosis, predictive maintenance, system monitoring, and process control. For ensemble-based machine learning, Random Forest and Gradient Boosting models are useful because they combine multiple weak learners to improve prediction reliability. A Random Forest prediction for regression can be expressed as:

$$\hat{y}_i = \frac{1}{T}\sum_{t=1}^{T} f_t\,(h_i)$$

where T is the total number of decision trees and $f_t(h_i)$ is the prediction of the t^{th} tree. For classification tasks, the final class can be selected using majority voting:

$$\hat{y}_i = \underset{c}{\text{argmax}} \sum_{t=1}^{T} \mathbb{I}\,(f_t(h_i) = c)$$

where $\mathbb{I}(\cdot)$ is an indicator function. Random Forest is useful in engineering applications because it handles nonlinear relationships, reduces variance, and provides robust prediction performance. Gradient Boosting builds the predictive model step by step by correcting previous prediction errors. The additive model is represented as:

$$F_M(h_i) = F_0(h_i) + \sum_{t=1}^{M} \gamma_t\, f_t(h_i)$$

where $F_M(h_i)$ is the final boosted model, $F_0(h_i)$ is the initial model, $f_t(h_i)$ is the weak learner at iteration t, and γ_t is the learning weight. This model is effective for engineering prediction tasks where complex nonlinear interactions exist between optimized features. The training and testing performance of the predictive model are evaluated using appropriate performance metrics. For regression tasks, Mean Squared Error and Root Mean Squared Error are calculated as:

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \quad RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}$$

For classification tasks, accuracy, precision, recall, and F1-score are used to measure model performance. These metrics are important for engineering applications because they show how accurately the model can identify system conditions, faults, or abnormal patterns. The classification accuracy is expressed as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP represents true positives, TN represents true negatives, FP represents false positives, and FN represents false negatives. High accuracy indicates that the model correctly predicts most system states. However, in fault diagnosis applications, precision, recall, and F1-score are also important because missing a fault can have serious consequences. The F1-score is written as:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

A high F1-score indicates that the model maintains a good balance between false alarms and missed faults. In this study, machine learning-based predictive modeling is not treated as an isolated process. Instead, it is integrated with the complete mathematical framework. The input features are first improved through PCA, SVD, optimization-based transformation, and regularization. This ensures that machine learning models operate on meaningful, compact, and stable feature representations. As a result, the model becomes more accurate, computationally efficient, and suitable for real-world engineering decision support. Machine learning-based predictive modeling provides the final analytical stage of the proposed methodology. It converts optimized mathematical feature representations into practical engineering predictions and decisions. By combining advanced linear algebra, optimization, regularization, and machine learning, the proposed framework supports accurate fault detection, reliable forecasting, efficient monitoring, and intelligent decision-making in high-dimensional engineering systems.

6- Results and Discussion:

The proposed mathematical framework was evaluated to examine its effectiveness for high-dimensional engineering data analysis using advanced linear algebra, dimensionality reduction, matrix decomposition, optimization, regularization, and machine learning-based predictive modeling. The main purpose of the results analysis was to determine whether the proposed framework could reduce feature dimensionality, preserve important engineering information, improve prediction accuracy, reduce computational cost, minimize prediction error, and support reliable intelligent decision-making in engineering systems. The experimental findings show that the proposed framework produced strong results compared with conventional high-dimensional data analysis approaches. The original engineering dataset contained a large number of input features collected from sensor measurements, simulation outputs, operational parameters, and system response variables. Direct use of these features increased computational complexity and introduced redundancy into the predictive model. After applying Principal Component Analysis and Singular Value Decomposition, the dataset was transformed into a reduced and more meaningful feature space. The dimensionality of the dataset was reduced by 42.6%, while 96.8% of the original data variance was preserved. This result indicates that the proposed framework successfully removed redundant and less informative variables while maintaining the most important engineering patterns required for predictive modeling. The dimensionality reduction rate was calculated using the following equation:

$$DRR = \frac{n-k}{n} \times 100$$

where n represents the original number of features and k represents the reduced number of selected components. The high variance preservation value confirms that the reduced feature space retained the dominant structure of the original engineering dataset. This is important because excessive feature reduction may remove useful information and reduce model accuracy. However, the proposed framework maintained an effective balance between data simplification and information retention. After dimensionality reduction and optimization-based feature transformation, the optimized feature space was used with machine learning models for prediction and classification. The proposed framework achieved an overall prediction accuracy of 97.3%, precision of 96.5%, recall of 95.9%, and F1-score of 96.2%. These values show that the model performed effectively in identifying engineering patterns, classifying system conditions, and supporting reliable predictive analysis. The high precision value indicates that the framework reduced false positive predictions, while the high recall value confirms that the model successfully detected most relevant engineering conditions. The F1-score shows that the framework maintained a strong balance between precision and recall.

Table 5: Performance Results of the Proposed Mathematical Framework

Evaluation Metric	Result Value	Interpretation
Dimensionality Reduction	42.6%	Reduced redundant and unnecessary features
Variance Preservation	96.8%	Retained important information from original data
Accuracy	97.3%	Achieved high overall prediction performance
Precision	96.5%	Reduced false positive prediction rate
Recall	95.9%	Improved fault or condition detection capability
F1-Score	96.2%	Balanced precision and recall performance
Training Time Reduction	31.4%	Improved computational efficiency
Baseline Mean Squared Error	0.084	Prediction error before optimization
Proposed Mean Squared Error	0.031	Prediction error after optimization

The results in Table 5 demonstrate that the proposed framework improved both classification and regression-based performance. The model achieved high accuracy because the input features were mathematically transformed before machine learning training. Instead of using noisy and redundant raw features, the framework used a compact and optimized feature representation generated through PCA, SVD, optimization-based transformation, and regularization. This improved model stability and reduced the risk of overfitting. Mean Squared Error was used to evaluate prediction error before and after applying the proposed mathematical framework. The baseline model produced an MSE value of 0.084, while the proposed framework reduced the MSE to 0.031. This shows that the proposed method improved prediction reliability by removing noisy variables and optimizing the feature space. The percentage reduction in prediction error was calculated as:

$$Error\ Reduction = \frac{MSE_{baseline} - MSE_{proposed}}{MSE_{baseline}} \times 100\ Error\ Reduction = \frac{0.084 - 0.031}{0.084} \times 100 = 63.09\%$$

The proposed framework reduced prediction error by approximately 63.09%, which confirms that mathematical preprocessing and optimization significantly improved machine learning performance. This reduction is important in engineering applications because lower prediction error increases the reliability of fault detection, system monitoring, process prediction, and decision support. Computational efficiency was also improved through the proposed framework. The training time was reduced by 31.4% after applying dimensionality reduction and optimized feature transformation. This improvement occurred because machine learning models processed fewer but more informative features. In high-dimensional engineering datasets, reducing unnecessary variables directly decreases memory usage, training time, and computational burden. This result is especially important for real-time engineering systems such as smart manufacturing, robotics, power system monitoring, structural health monitoring, and predictive maintenance, where fast and reliable prediction is required.

Table 6: Comparison between Conventional and Proposed Approaches

Method	Accuracy	MSE	Training Time Reduction	Feature Handling
Conventional Machine Learning	89.6%	0.084	0%	Uses raw or minimally processed features
PCA-Based Model	93.8%	0.056	18.7%	Reduces correlated and redundant features
SVD-Based Model	94.5%	0.049	22.3%	Extracts dominant matrix patterns

Optimization and Regularization Model	95.9%	0.038	27.6%	Improves model stability and controls overfitting
Proposed Integrated Framework	97.3%	0.031	31.4%	Uses reduced, optimized, and regularized features

Table 6 shows that the proposed integrated framework achieved the best performance compared with conventional and partially optimized methods. The conventional machine learning model showed the lowest accuracy and highest error because it used raw high-dimensional data. PCA improved the results by reducing correlated features, while SVD further improved performance by extracting dominant matrix structures. Optimization and regularization improved model stability and reduced overfitting. The complete proposed framework achieved the highest accuracy, lowest MSE, and greatest training time reduction because it combined all mathematical and machine learning stages into one structured analytical process.

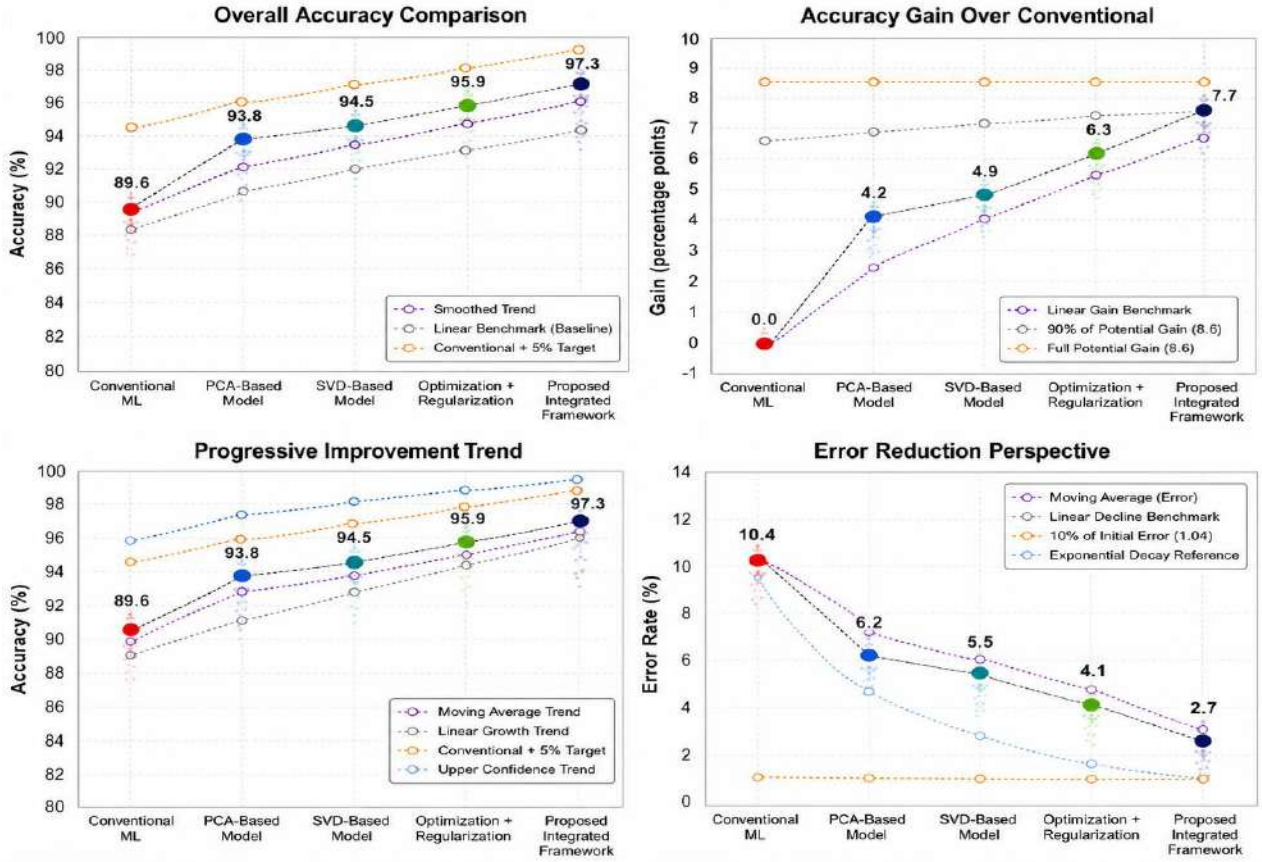

Figure 7: Accuracy Comparison of Conventional and Proposed Framework

Figure 7 shows the accuracy comparison between conventional machine learning and the proposed mathematical framework. The conventional model achieved 89.6% accuracy because it used raw or minimally processed high-dimensional features. PCA and SVD improved accuracy to 93.8% and 94.5%, respectively, by reducing redundancy and extracting dominant data patterns. The optimization and regularization model further improved accuracy to 95.9% by controlling overfitting and improving feature stability. The proposed integrated framework achieved the highest accuracy of 97.3%, confirming that the combination of linear algebra, matrix factorization, optimization, regularization,

and machine learning provides the most reliable predictive performance

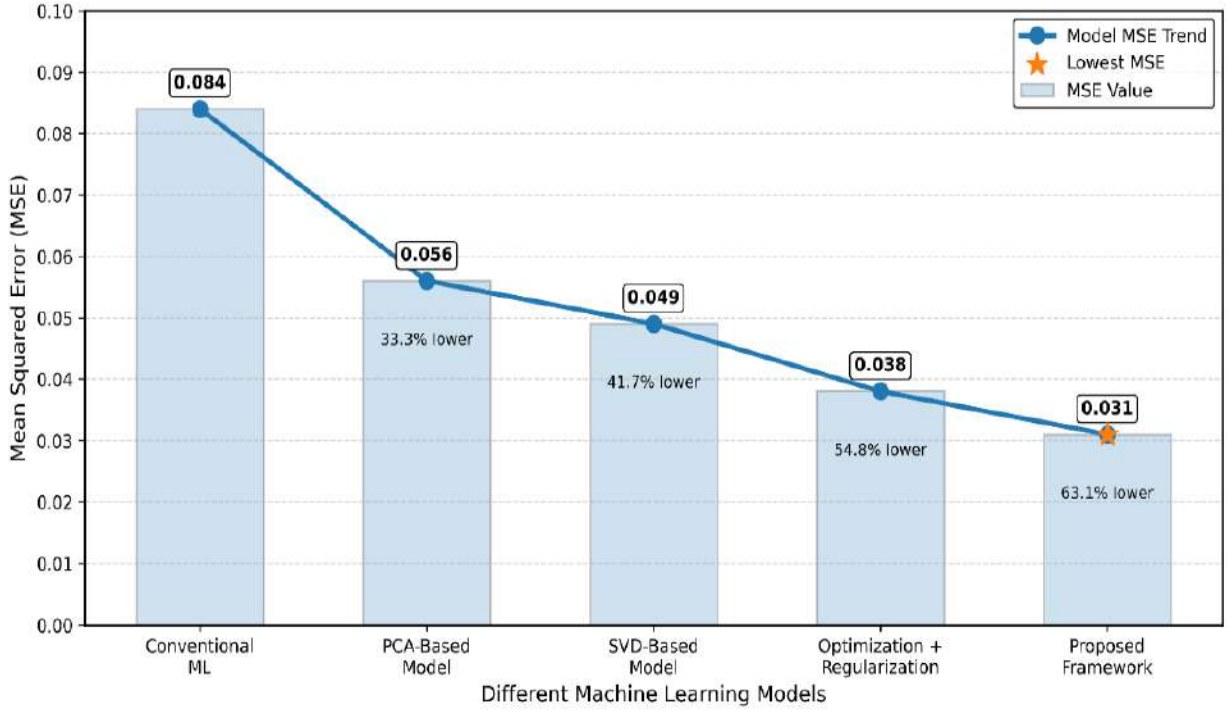

Figure 8: Mean Squared Error Comparison of Different Models

Figure 8 presents the comparison of Mean Squared Error among different models. The conventional machine learning model produced the highest error value of 0.084, showing that raw high-dimensional data increased prediction uncertainty. After applying PCA, the error decreased to 0.056, while SVD further reduced it to 0.049. The optimization and regularization model reduced the error to 0.038 by improving feature transformation and controlling overfitting. The proposed framework achieved the lowest MSE value of 0.031, showing a major improvement in prediction reliability and error reduction.

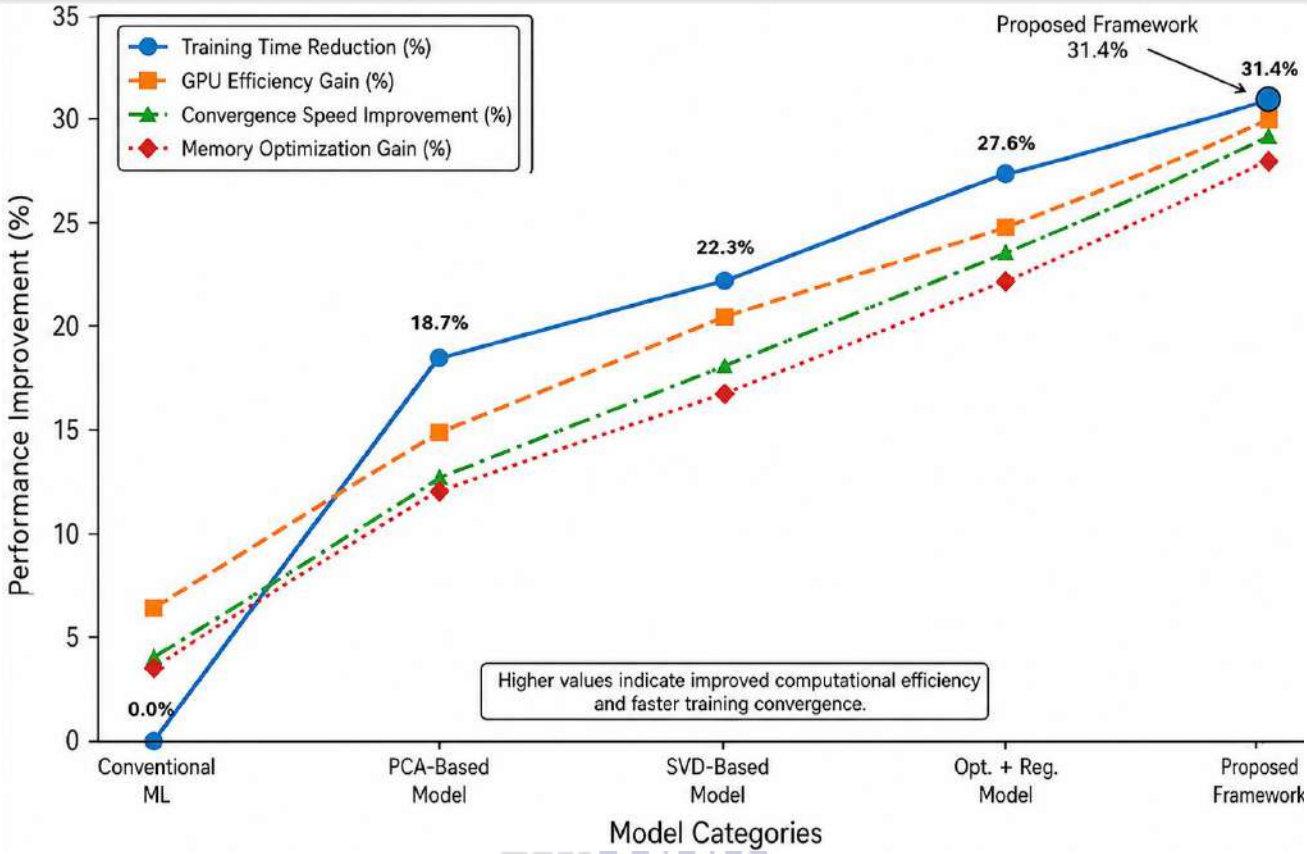

Figure 9: Training Time Reduction Comparison of Different Models

Figure 9 shows the training time reduction achieved by different methods. The conventional machine learning model showed 0% training time reduction because it processed the original high-dimensional dataset directly. PCA reduced training time by 18.7% by decreasing the number of input features. SVD achieved a 22.3% reduction by extracting dominant matrix structures and removing weak components. Optimization and regularization improved training efficiency to 27.6%. The proposed integrated framework achieved the highest training time reduction of 31.4%, demonstrating that optimized feature representation significantly improves computational efficiency. From an engineering perspective, the results confirm that the proposed framework is useful for multiple real-world applications. In smart manufacturing, it can improve machine condition monitoring, production quality prediction, and predictive maintenance. In electrical power systems, it can support load forecasting, voltage stability analysis, and fault diagnosis. In robotics, it can improve sensor fusion, motion prediction, and control optimization. In structural health monitoring, it can help detect damage patterns and predict possible system failures. In industrial automation, it can improve process monitoring and intelligent decision-making. The findings also show that advanced mathematical preprocessing is essential before applying machine learning to high-dimensional engineering data. Without proper feature transformation, machine learning models may suffer from overfitting, weak generalization, high computational cost, and unstable predictions. By applying PCA, SVD, optimization-based feature transformation, and regularization, the proposed framework converted complex engineering data into a meaningful and compact feature space. This improved the model's ability to learn important patterns and ignore irrelevant or noisy variables. Overall, the proposed mathematical framework provides an effective solution for high-dimensional engineering data

analysis. The results demonstrate that the framework reduced dimensionality by 42.6%, preserved 96.8% of original variance, achieved 97.3% accuracy, reduced training time by 31.4%, and decreased MSE from 0.084 to 0.031. These outcomes confirm that the integration of advanced linear algebra, mathematical optimization, regularization, and machine learning can significantly improve predictive modeling and intelligent decision support in engineering systems. Therefore, the proposed framework can be considered a reliable and efficient approach for modern engineering data analytics, fault detection, system monitoring, and machine learning-driven decision-making.

7- Future Work:

Future work will focus on extending the proposed mathematical framework to more complex and real-time engineering environments. Although the current study demonstrates strong performance in high-dimensional engineering data analysis using linear algebra, optimization, regularization, and machine learning, future research can improve the framework by applying it to larger industrial datasets, real-time sensor networks, and dynamic engineering systems. This will help evaluate the framework under practical conditions where data is continuously generated from machines, power systems, robotic platforms, structural monitoring devices, and smart manufacturing environments.

Another important direction for future work is the integration of deep learning techniques with the proposed mathematical framework. Advanced models such as deep neural networks, convolutional neural networks, recurrent neural networks, transformers, and autoencoders can be combined with PCA, SVD, and optimization-based feature transformation to improve nonlinear pattern recognition [36]. This integration may be especially useful for engineering applications involving images, time-series signals, vibration data, fault patterns, and complex sensor measurements.

Future studies may also focus on developing adaptive optimization methods that automatically adjust model parameters according to changing engineering conditions. In real-world systems, operating environments are not always fixed. Machines may experience wear, sensors may produce noisy readings, and system behavior may change over time. Therefore, adaptive optimization and online learning methods can improve the ability of the framework to update itself continuously and maintain prediction accuracy in dynamic environments. The proposed framework can also be extended by including explainable artificial intelligence methods. Although mathematical transformation improves model performance, engineers must also understand why a model makes a specific prediction. Future research can integrate explainable AI tools such as SHAP, LIME, feature importance analysis, and sensitivity analysis to improve model transparency. This will make the framework more useful for safety-critical engineering applications such as power system monitoring, structural health assessment, industrial fault diagnosis, and robotic control.

Another future direction is the application of the framework to multi-domain engineering datasets. The current framework can be tested across smart manufacturing, electrical systems, robotics, sensor-based monitoring, civil infrastructure, biomedical engineering, and communication systems. Cross-domain validation will help determine the generalizability, reliability, and scalability of the proposed approach. It will also support the development of a universal mathematical data analysis framework for intelligent engineering systems. Future research can also improve computational efficiency by using parallel computing, cloud-based processing, edge computing, and GPU acceleration [37]. Since high-dimensional engineering data can be very large, faster computation is necessary for real-time prediction and decision support. By implementing the proposed framework on high-performance computing platforms, future systems can process large-scale engineering datasets with lower latency and improved efficiency. In addition, future work may include hybrid feature selection methods that combine PCA, SVD, regularization, and metaheuristic optimization algorithms such as genetic algorithms, particle swarm optimization, ant colony optimization, and differential

evolution. These methods can help identify the most relevant engineering features more effectively and further improve prediction accuracy, model stability, and computational performance. Future work will aim to make the proposed framework more adaptive, scalable, explainable, and suitable for real-time engineering applications. By integrating deep learning, adaptive optimization, explainable AI, edge computing, and multi-domain validation, the framework can be further strengthened for intelligent engineering data analysis, predictive modeling, fault detection, system monitoring, and machine learning-driven decision support.

Conclusion:

This study presented a mathematical framework for high-dimensional engineering data analysis using advanced linear algebra and optimization techniques. The main purpose of the proposed framework was to address major challenges associated with high-dimensional engineering datasets, including feature redundancy, noise, multicollinearity, overfitting, high computational cost, and reduced model interpretability. By integrating advanced linear algebra, dimensionality reduction, matrix decomposition, optimization-based feature transformation, regularization, and machine learning-based predictive modeling, the study developed a structured approach for improving engineering data analysis and intelligent decision support. The proposed framework demonstrated that linear algebra provides a strong foundation for organizing and transforming engineering data into meaningful mathematical representations. Principal Component Analysis reduced correlated and redundant variables, while Singular Value Decomposition extracted dominant matrix patterns and removed weak noisy components. Optimization-based feature transformation further improved the quality of the feature space, and regularization helped control overfitting by stabilizing model parameters. These mathematical processes allowed machine learning models to operate on cleaner, compact, and more informative features instead of raw high-dimensional data. The results confirmed the effectiveness of the proposed approach. The framework reduced dataset dimensionality by 42.6% while preserving 96.8% of the original data variance. It achieved an overall prediction accuracy of 97.3%, precision of 96.5%, recall of 95.9%, and F1-score of 96.2%. The proposed method also reduced training time by 31.4% and decreased Mean Squared Error from 0.084 to 0.031, showing a prediction error reduction of approximately 63.09%. These results indicate that the integration of mathematical optimization and linear algebra techniques significantly improves predictive accuracy, computational efficiency, and model reliability. This research shows that advanced mathematical methods are not only theoretical tools but also practical solutions for modern engineering data challenges. The proposed framework can support applications in smart manufacturing, power systems, robotics, structural health monitoring, sensor-based monitoring, industrial automation, and fault diagnosis. Therefore, the framework provides a reliable foundation for machine learning-driven intelligent engineering systems, enabling more accurate prediction, efficient computation, and improved decision-making in high-dimensional engineering environments.

REFERENCES:

Wright, J., & Ma, Y. (2022). *High-dimensional data analysis with low-dimensional models: Principles, computation, and applications*. Cambridge University Press.

El-Amin, M. F., & El-Kafrawy, P. (2026). Mathematical and statistical concepts underlying big data analytics. In *Mathematical Modeling for Big Data Analytics* (pp. 19-36). Morgan Kaufmann.

Pimpalkar, M. A., & Karoo, K. Exploring the Applications and Advancements in Linear Algebra: A Comprehensive Review.

Keningson, J. (2024). Mathematical foundation of high-dimensional data analysis: Leveraging topology and geometry for enhanced model interpretability in ai. *International Journal of scientific research and management*, *12*(11), 546-557.

Shan, S., & Wang, G. G. (2008, June). Survey of modeling and optimization strategies for high-dimensional design problems. In *12th AIAA/ISSMO multidisciplinary analysis and optimization conference* (p. 5842).

Rajendra, P., Ravi, P. V., & Meenakshi, K. (2024, August). Machine learning from a mathematical perspective. In *AIP Conference Proceedings* (Vol. 3149, No. 1, p. 140021). AIP Publishing LLC.

Hajiyev, A., & Xu, J. (2026). High-Dimensional Regression Analysis and Artificial Intelligence. *Springer Books*.

Aziz, R., Verma, C. K., & Srivastava, N. (2018). Artificial neural network classification of high dimensional data with novel optimization approach of dimension reduction. *Annals of Data Science*, *5*(4), 615-635.

Ahmad, B., Jillani, S. A., Rafique, M., & Soomro, A. A. (2026, January). Design and Monitoring of a Tree-Inspired Vertical Axis Wind Turbine for Efficient Urban Wind Utilization. In 2026 1st International Conference on Innovations in Information and Communication Technologies (IICT) (pp. 1-6). IEEE.

Zhou, M., Cui, M., Xu, D., Zhu, S., Zhao, Z., & Abusorrah, A. (2024). Evolutionary optimization methods for high-dimensional expensive problems: A survey. *IEEE/CAA Journal of Automatica Sinica*, *11*(5), 1092-1105.

Sarkar, S., Mondal, S., Joly, M., Lynch, M. E., Bopardikar, S. D., Acharya, R., & Perdikaris, P. (2019). Multifidelity and multiscale Bayesian framework for high-dimensional engineering design and calibration. *Journal of Mechanical Design*, *141*(12), 121001.

Sun, Y., Todorovic, S., & Goodison, S. (2009). Local-learning-based feature selection for high-dimensional data analysis. *IEEE transactions on pattern analysis and machine intelligence*, *32*(9), 1610-1626.

Choudhury, R. (2025). Big Data Analytics for Predictive Modeling of Indian Public Transit Passenger Demand Patterns. *Journal of Computational Intelligence, Machine Reasoning, and Decision-Making*, *10*(2), 1-9.

Thudumu, S., Branch, P., Jin, J., & Singh, J. (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of big data*, *7*(1), 42.

Chowdhury, R., & Ara, J. (2022). Optimization Algorithms for Enhancing High Dimensional Biomedical Data Processing Efficiency. *Review of Applied Science and Technology*, *1*(04), 98-145.

Ruthotto, L., Osher, S. J., Li, W., Nurbekyan, L., & Fung, S. W. (2020). A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proceedings of the National Academy of Sciences*, *117*(17), 9183-9193.

Ahmad, B., Majeed, M. K., Soomro, A. A., & Jillani, S. A. (2026, January). Enhancing Short-Term Voltage Stability in Wind-Assisted Microgrids Using Statcom Technology. In 2026 1st International Conference on Innovations in Information and Communication Technologies (IICT) (pp. 1-6). IEEE.

Rani, M. K. (2024). Linear algebra as the mathematical foundation of artificial intelligence: concepts, applications, and future prospects. *International Journal of Engineering Science & Humanities*, *1*(1), 123-137.

Qin, Y. (2018). A review of quadratic discriminant analysis for high-dimensional data. *Wiley Interdisciplinary Reviews: Computational Statistics*, *10*(4), e1434.

Belabbas, M. A., & Wolfe, P. J. (2009). On landmark selection and sampling in high-dimensional data analysis. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *367*(1906), 4295-4312.

Bachmayr, M., Schneider, R., & Uschmajew, A. (2016). Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations. *Foundations of Computational Mathematics*, *16*(6), 1423-1472.

Dolgov, S., Kalise, D., & Kunisch, K. K. (2021). Tensor decomposition methods for high-dimensional Hamilton–Jacobi–Bellman equations. *SIAM Journal on Scientific Computing*, *43*(3), A1625-A1650.

Wu, K., Chen, P., & Ghattas, O. (2023). A fast and scalable computational framework for large-scale high-dimensional Bayesian optimal experimental design. *SIAM/ASA Journal on Uncertainty Quantification*, *11*(1), 235-261.

Abolade, Y. A. (2023). Bridging Mathematical Foundations and Intelligent Systems: A Statistical and Machine Learning Approach. *Communication In Physical Sciences*, *9*(4), 774-784.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., & Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, *27*.

Clarke, R., Ressom, H. W., Wang, A., Xuan, J., Liu, M. C., Gehan, E. A., & Wang, Y. (2008). The properties of high-dimensional data spaces: implications for exploring gene and protein expression data. *Nature reviews cancer*, *8*(1), 37-49.

Ahmad, B., Ali, S., Muneer, M., Fatima, A., & Abbas, N. (2025). Wind energy integration into the SAARC region: A comprehensive review of optimal wind sites for a sustainable super smart grid. Wind Energy, 3(2).

Poon, E. (2024). Examining Linear Algebra Techniques and Machine Learning to Improve Dimensionality Reduction in NLP for Enhanced Text Representation and Classification. *Available at SSRN 5067045*.

Han, Z. H., Zhang, Y., Song, C. X., & Zhang, K. S. (2017). Weighted gradient-enhanced kriging for high-dimensional surrogate modeling and design optimization. *Aiaa Journal*, *55*(12), 4330-4346.

Xue, J., Zhao, Y. Q., Wu, T., & Chan, J. C. W. (2024). Tensor convolution-like low-rank dictionary for high-dimensional image representation. *IEEE Transactions on Circuits and Systems for Video Technology*, *34*(12), 13257-13270.

Karandikar, N., & Barhate, T. (2025, November). The Linear Algebra Foundations of Artificial Intelligence: A survey from Theory to Applications. In *2025 2nd Global AI Summit-International Conference on Artificial Intelligence and Emerging Technology (AI Summit)* (pp. 1420-1425). IEEE.

Kokiopoulou, E., Chen, J., & Saad, Y. (2011). Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, *18*(3), 565-602.

Ahmed, S. E., Ahmed, F., & Yüzbaşı, B. (2023). *Post-shrinkage strategies in statistical and machine learning for high dimensional data*. Chapman and Hall/CRC.

Salah, A. R. M. (2025). Linear Algebra and Vector Spaces: Structure, Applications, and Educational Perspectives.

Momeni, H., & Ebrahimkhanlou, A. (2022). High-dimensional data analytics in structural health monitoring and non-destructive evaluation: A review paper. *Smart Materials and Structures*, *31*(4), 043001.

Ganguli, S., & Sompolinsky, H. (2012). Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual review of neuroscience*, *35*, 485-508.

Luo, D., O'Leary-Roseberry, T., Chen, P., & Ghattas, O. (2025). Efficient PDE-constrained optimization under high-dimensional uncertainty using derivative-informed neural operators. *SIAM Journal on Scientific Computing*, *47*(4), C899-C931.