International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I3P106

Volume 74 Issue 3, 75-88, March 2026
© 2026 Seventh Sense Research Group®

Original Article

Predicting and Categorizing Urban Air Pollution: A Comparative Study of Regression and Classification Models with Vedic Mathematical Feature

Shital Tayade[1], Nalini Vaidya[2]

[1,2]Department of Mathematics, School of Science, G H Raisoni University, Saikheda, Madhya Pradesh, India.

[1]Corresponding Author : shital.tayade@raisoni.net

Received: 23 July 2025 Revised: 27 January 2026 Accepted: 29 January 2026 Published: 28 March 2026

Abstract - *Due to the adverse impacts on both humanity and the environment, severe air pollution issues have drawn attention from all over the world. The air quality in Delhi, the capital of India, has drastically declined. Delhi has the second-leading air pollution rate in the world, according to the 2024 Global Weather Pollution Review of an ecological surveillance organization. This work combines support vector approaches to learning, neural computing models, and Vedic mathematics for investigating Delhi's air quality using an enormous dataset. With a focus on computational performance and data efficiency during processing, we compare the proposed strategy to gradient descent-based optimization techniques like Adagrad, Adadelta, Adam, and RMSProp. The current investigation offers a thorough analytical approach for evaluating urban air quality through classification and regression processes. Based on airborne particle concentration data acquired in Delhi between 2020 and 2023, the solutions suggested utilize regression for accurately predicting Air Pollution Index (AQI) values and algorithmic classification for assessing the degree of air pollution. The modeling approach makes use of results from simulation to evaluate the effects of two Vedic mathematics sutras, Urdhva-Tiryakbhyam and Nikhilam Sutra. We evaluate the performance of models like SVM and ANN using feature transformations based on Vedic mathematics, like Urdhva-Tiryakbhyam. Furthermore, we contrast the Vedic Mathematics multipliers based on the Nikhilam Sutra and Urdhva-Tiryakbhyam with the conventional machine learning coefficient computations in SVM and ANN models, which use a variety of optimization techniques. Common metrics like Mean Squared Error (MSE) and R2 for regression and F1-score, precision, and recall for classification have been employed to evaluate the efficacy of the two approaches. These findings demonstrate how the Vedic Mathematics framework increases computing efficiency by enabling quicker and less expensive coefficient computation. On the other hand, the combined analytical approach offers a more profound understanding of environmental monitoring.*

Keywords - *SVM, ANN, ADAM, Adagrad, Adadelta, RTMSprop, UrdhvaTiryakbhyam, Nikhilam Sutra.*

1. Introduction

Industrial emissions, vehicular traffic, construction dust, and the combustion of agricultural residues are major contributors to Delhi's deteriorating air quality. Unfavourable winter meteorological conditions further trap pollutants in the atmosphere, significantly aggravating pollution levels. Rapid urbanisation, dense population, and increasing energy demands have intensified emissions, making Delhi one of the most polluted cities in the world, as well as one of the most environmentally degraded. Rising vehicular concentration, industrial expansion, and inefficient waste management practices further contribute to elevated carbon emissions and persistent air quality challenges. AQI data for Delhi spanning the period from 2020 to 2023 were collected from the Kaggle repository and used to develop the proposed forecasting models. The dataset contains hourly measurements of key air pollutants, including NO_2, O_3, SO_2, NH_3, CO, PM2.5, PM10, and NO, which are widely recognized as critical indicators of air pollution and pose serious potential harm to human health. During the months from March to September, Delhi's Air Quality Index (AQI) generally ranges from Good (0–50) to Moderate (101–200). However, due to factors such as road dust, vehicular emissions, stubble burning, and industrial activities, AQI levels often deteriorate to Poor (201–300), Very Poor (301–400), Severe (401–500), and occasionally Hazardous (>500). Based on historical trends and persistently elevated ambient PM2.5 concentrations exceeding citywide averages, environmental regulatory agencies at the state and national levels (DPCC & CPCB) identified 13 critically polluted locations in Delhi. These areas include R.K. Puram, Bawana, Mundka, Narela, Mayapuri, Wazirpur, Jahangirpuri, Anand Vihar, Vivek Vihar, Rohini, Dwarka, Okhla Phase-II, and Punjabi Bagh, which are consistently affected by high pollutant concentrations.

The following is a summary of the primary gaps in understanding identified across the entirety of current literature. Conventional machine learning or deep learning models are used in the majority of health prediction studies, with not much thought paid to computational efficiency and arithmetic optimization. Furthermore, the incorporation of Vedic mathematics into machine learning optimization procedures or coefficient computation mechanisms has not received much attention. Comprehensive comparison and contrast are restricted, considering previous studies seldom use an individual classifier and regression template with the identical atmosphere database and algorithmic optimization. Likewise, evolutionary actions and numerical productivity have received far fewer resources than accurate prediction, especially in real-time and real-world deployment scenarios. In an effort to address these gaps, the present investigation proposes a novel integration of the Vedic mathematical fundamentals, particularly the Nikhilam Sutra and the Urdhva-Tiryakbhyam Sutra, toward learning frameworks influenced by neurons and kernel-centered support vector methods for predicting urban air quality. Regression for precise AQI estimation and classification for pollution severity assessment are carried out concurrently in a unified analytical framework. Further, a consistent experimental setting serves for systematic assessment of four different adaptive improvement algorithms, such as Adagrad, Adadelta, RMSProp, and Adam. Research results show that mathematical operations grounded in classical mathematics enhanced computational effectiveness, all while conserving superior mathematical validity. The recorded strength of primary airborne contaminants, comprising PM2.5, PM10, NH_3, and SO_2, via more than stations for monitoring in the Delhi/NCR metropolitan territory is shown in Figure 1. Important pollution hotspots are highlighted by the visual analysis, which shows notable spatial variation in pollutant levels. Significantly, stations like Anand Vihar show remarkably high PM10 concentrations, suggesting serious localized pollution issues.

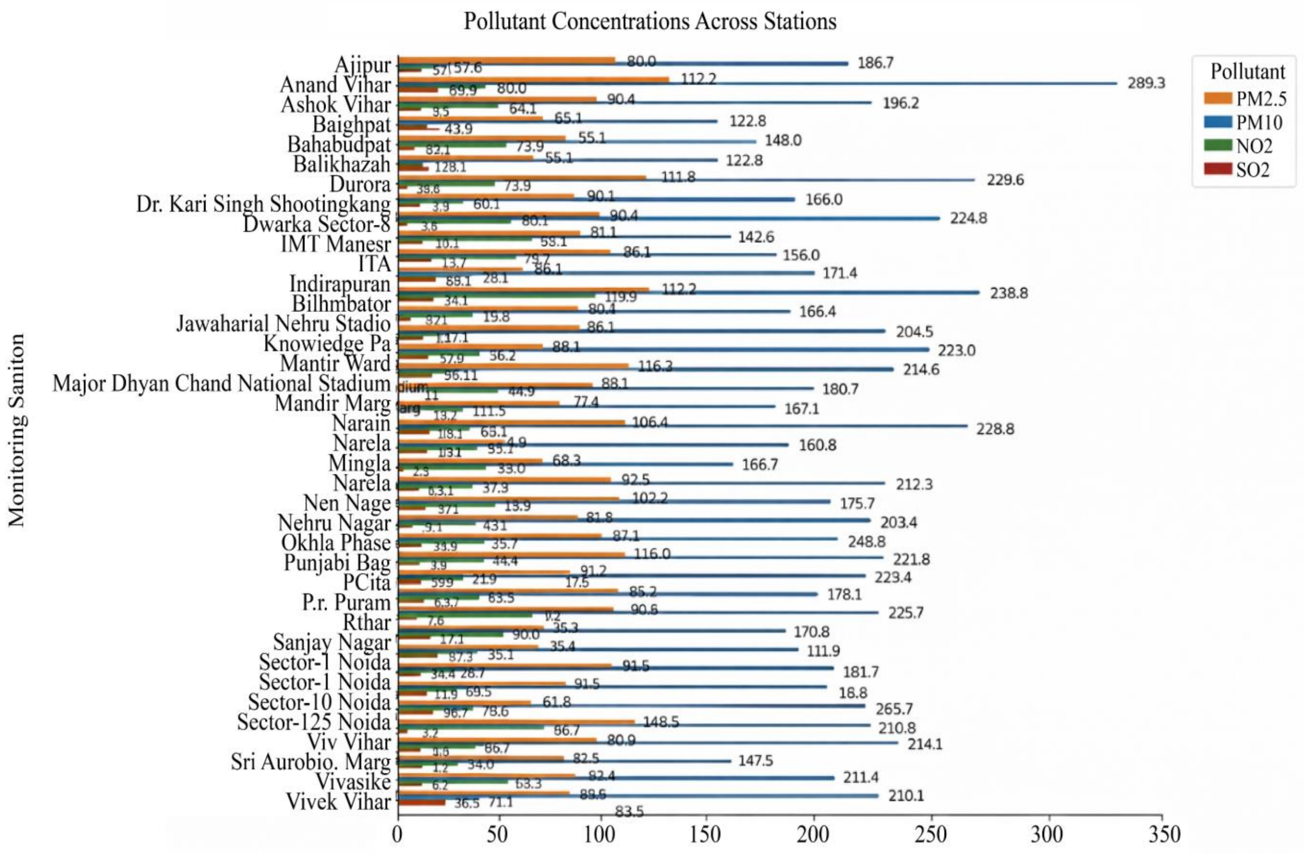


Fig. 1 Pollutant concentration across station

The measurements indicate extremely elevated levels of PM2.5 in the air. PM2.5 levels were seen in Pusa (124.0 µg/m³), Punjabi Bagh (120.1 µg/m³), and Indirapuram (101.6 µg/m³). Particles measuring PM2.5 micrometres are riskier as they have the ability to penetrate deeply into the lungs. The information shows that tiny bits of matter in the air, especially PM10 and PM2.5, are a big problem in the Delhi-National Capital Region (NCR). Numerous stations frequently recorded values significantly exceeding safe limits, particularly during the winter months. The level of residual pollution was minimal, indicating reduced emissions from both agricultural waste incineration and industrial activities.

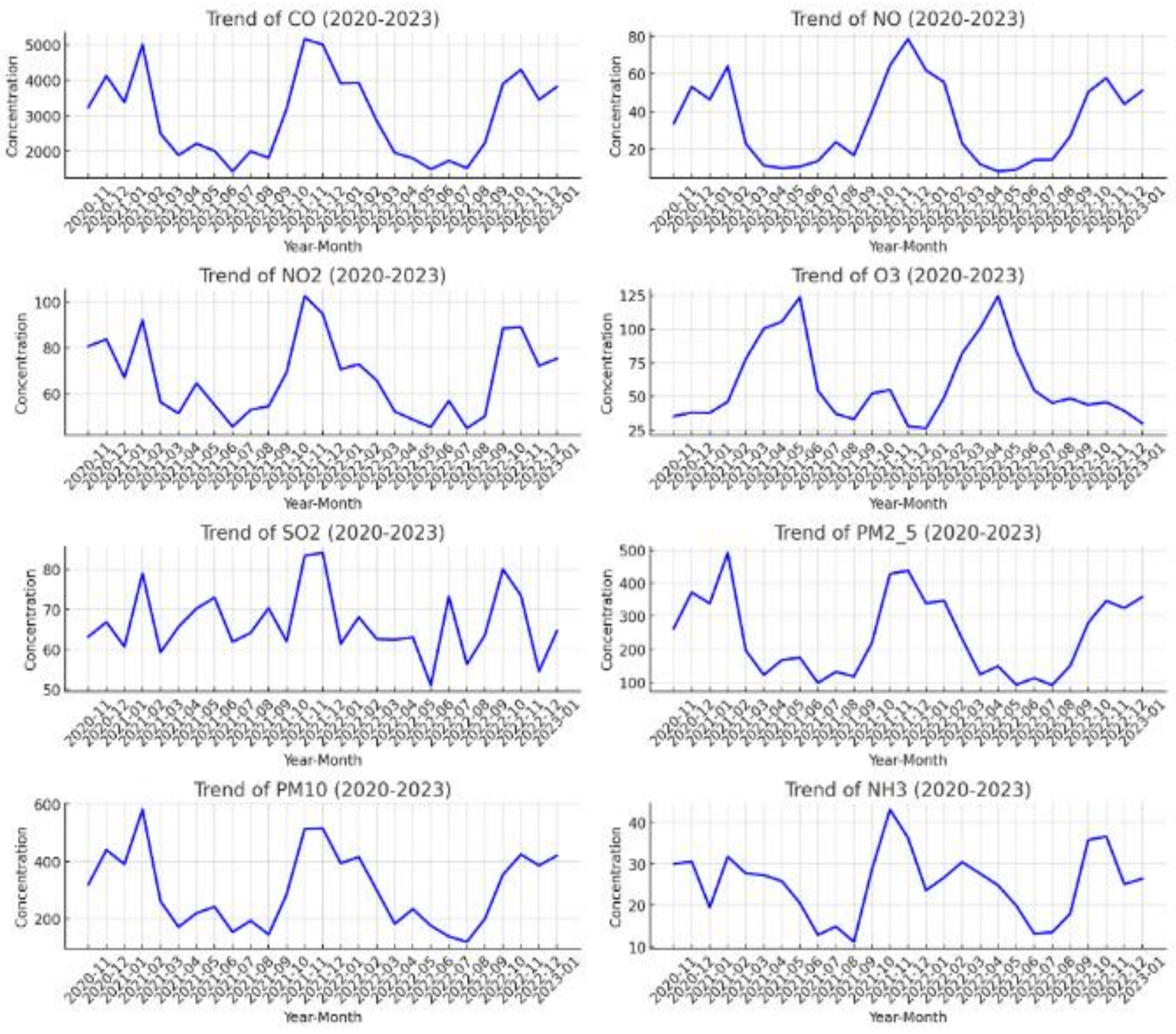
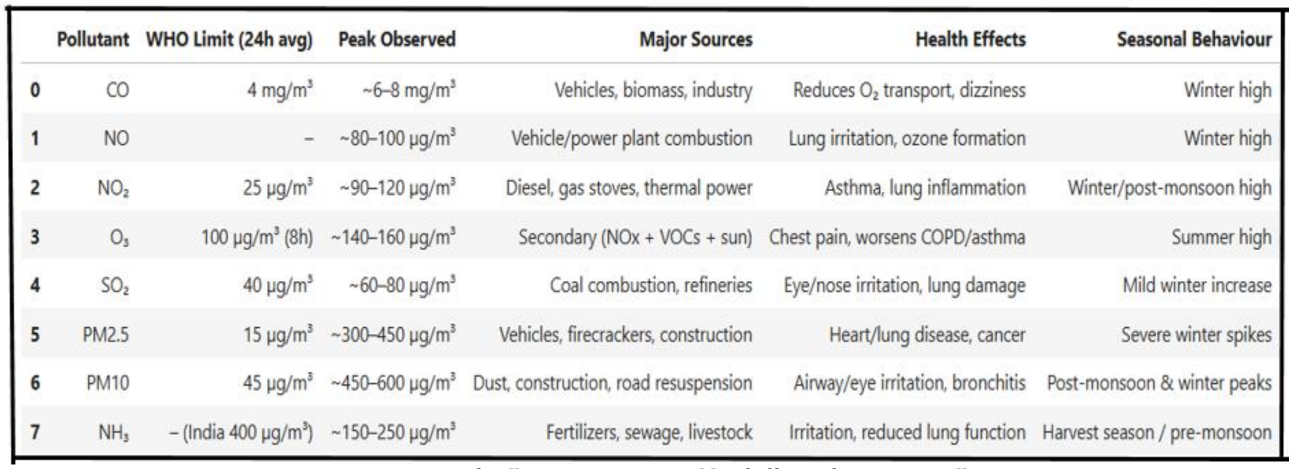


Fig. 2 Concentration of pollutant

Based on the observed pollution levels, carbon monoxide (CO) concentrations increased significantly during the early and late months of 2021. Nitric oxide (NO) levels also rose during the cooler periods of early and late 2021, most likely due to increased fossil fuel consumption for heating and transportation. On the contrary, quantities of NO decreased across the middle of 2021 and the middle of 2022.

The fluctuations in the level of nitrogen dioxide (NO_2) were not as evident compared to the rise in NO levels, but there were indeed several notable excursions in early 2021 as well as in mid-2022. The measured amounts of sulphur dioxide (SO_2) stayed mostly the same with limited modifications throughout the entirety of the investigation. At the beginning of 2022, the amounts of PM2.5 were somewhat higher than expected, which demonstrates that minor particle contamination was still on an upward trajectory. During the winter season, particularly towards the start and end of 2021, enormous quantities of PM10, which correspond to coarse fine particles, were regularly identified. The trend mentioned above shows that winter temperatures cause environmental damage regularly. Similar variations were observed in PM2.5, but the changes were more noticeable.

This suggests that there is a lot of particulate matter in the air, probably from burning things and other sources of pollution. The worst pollution happened in the winter and right after the monsoon season. In general, the levels of pollutants stayed pretty stable, with a few spikes that could be due to burning agricultural waste or industrial emissions. Below the corresponding graph, there is a summary table that shows how each pollutant affects the environment.



	Pollutant	WHO Limit (24h avg)	Peak Observed	Major Sources	Health Effects	Seasonal Behaviour
0	CO	4 mg/m³	~6–8 mg/m³	Vehicles, biomass, industry	Reduces O_2 transport, dizziness	Winter high
1	NO	–	~80–100 µg/m³	Vehicle/power plant combustion	Lung irritation, ozone formation	Winter high
2	NO_2	25 µg/m³	~90–120 µg/m³	Diesel, gas stoves, thermal power	Asthma, lung inflammation	Winter/post-monsoon high
3	O_3	100 µg/m³ (8h)	~140–160 µg/m³	Secondary (NOx + VOCs + sun)	Chest pain, worsens COPD/asthma	Summer high
4	SO_2	40 µg/m³	~60–80 µg/m³	Coal combustion, refineries	Eye/nose irritation, lung damage	Mild winter increase
5	PM2.5	15 µg/m³	~300–450 µg/m³	Vehicles, firecrackers, construction	Heart/lung disease, cancer	Severe winter spikes
6	PM10	45 µg/m³	~450–600 µg/m³	Dust, construction, road resuspension	Airway/eye irritation, bronchitis	Post-monsoon & winter peaks
7	NH_3	– (India 400 µg/m³)	~150–250 µg/m³	Fertilizers, sewage, livestock	Irritation, reduced lung function	Harvest season / pre-monsoon

Fig. 3 Concentration of pollutants: summary table of effects of major air pollutants

2. Related Work

China implemented the Air Quality Action Strategy from 2013 to 2017 to minimize PM2.5 pollution by imposing restrictions on emissions from manufacturing operations. Research results reveal that manufacturing exhaust and combustion were playing an essential part in dropping PM2.5 levels in the initial phase of the policy. Nevertheless, by 2017, the adverse impacts of such sources of contamination had declined, and the population density evolved into an extremely significant factor impacting PM2.5 pollution levels. The analysis implies that subsequent sustainability regulation needs to regulate growth in urban areas while simultaneously considering different variables that affect it [1]. Population growth has rendered the contamination of the air more severe, particularly in metropolitan areas and cities. As urban centres in the Yangtze River Delta region (YRDr) become increasingly interlinked, the emergence of factors makes the association between population growth and environmental degradation progressively more challenging. Studies involving geographical factors point out that residential construction typically prevents air pollution in the YRDr, even though growth in population elevates discharge levels. Urban growth additionally produces consequences for the financial sector, the community, and the environment.

As societies spread out, the emission trend progressively appears more severe, followed by turns that improve. This occurs due to individuals' wealth and manufacturing operations often. The findings indicate that specialized and region-specific regulations are vital for accomplishing equitable urban growth and improving the air [2]. For estimating and categorizing the quality of the air, many machine learning and statistical methods have been successfully employed. These involve both long and brief varying storage learning networks (LSTM), XGBoost, and SARIMAX regression for estimating, as well as the XGBoost classification system for classifying pollution levels. The XGBoost-based models for both regression and classification executed more successfully than all other models, with a regression coefficient score of 0.9519 and a classification performance of 0.9583, respectively [3]. Multidimensional adaptive learning traces have been applied to predicting yields of crops to enhance forecasting precision via powerful fine-tuning that minimizes root mean square error. A thorough performance evaluation using standard metrics like F1-score, RMSE, recall, accuracy, precision, and computational time shows that the proposed Multivariate Regressive Gradient Spiral Optimized Deep Belief Learning (MRGSODBL) method is more accurate, takes less time, and makes fewer mistakes than traditional deep learning methods [4]. Additionally, we explored Vedic Mathematics-based coefficients using performance indicators, which include mean relative error, area, speed, power consumption, and propagation delay. The Urdhva-Tiryakbhyam sutra had been discovered to carry out estimations more rapidly as well as take up fewer resources than the Ekadhikena Purvena sutra. In terms of energy conservation, Urdhva-Tiryakbhyam was the best sutra due to its ideal balance between dimension and interruption.

The Urdhva-Tiryakbhyam sutra was found to improve computational speed while reducing area compared to the Ekadhikena Purvena sutra. In terms of energy efficiency, Urdhva-Tiryakbhyam emerged as the most effective sutra, offering an optimal balance between area and delay. These findings highlight the scalability potential of Vedic multipliers and their suitability for parallel computing applications, particularly for handling large-scale biological data and complex computations [5]. Two Real multiplier designs inspired by the Urdhva-Tiryakbhyam Vedic algorithm (vertical and crosswise) sutra have been proposed, along with an analytical expression for routing loss in N × N complex multipliers. Complex multiplication was implemented using Vedic real multiplier architectures with minimized path delay. The VHDL-based designs for 32 × 32-bit complex multiplication were developed using Xilinx ISE 13.4, and simulation was performed using ModelSim 5.6. Comparative results demonstrate superior performance over conventional Booth and array multiplier-based approaches [6]. Support Vector Machine–based Regression approach (SVR) has been applied to air quality forecasting and demonstrated effective modelling of nonlinear interactions among air pollutants, resulting in improved prediction accuracy compared to traditional approaches [20]. Deep neural network-based models have also proven capable of capturing complex nonlinear patterns in air pollution data, thereby achieving higher predictive performance than conventional machine learning techniques [21]. The models developed by ARIMA serve as a conventional time-series framework for predicting significant ambient volatile pollutants such as O_3, NO, NO_2, and CO, while illustrating the inadequacies of linear statistical methodologies in recognizing intricate emissions fluctuations [22]. Neural network-driven environmental forecasting approaches exhibit that they can learn intricate nonlinear connections between the airborne contaminants and atmospheric factors, thus rendering their estimates far more precise and accurate. Predicting methods based on attribute inspection find significant information properties that elevate the predictability of predictions about air quality by identifying and utilizing the appropriate details [23]. Further, neural system-based verification processes have been profitable for assessing benzene levels in metropolitan locations, illustrating the utility of data-driven models for air pollution surveillance [24].

3. Methodology

3.1. Data Overview

The information set employed for the present investigation involves metropolitan air quality evaluations acquired through different monitoring locations in Delhi from 2020 to 2023. It retains periodic monitoring for significant airborne particles like PM2.5, PM10, NO_2, O_3, SO_2, CO, and NH_3. These particles are widely acknowledged as crucial evidence of pollutants in the air and health threats. It incorporates the AQI (Air Quality Index) in conjunction with the hazardous substance amount. The latter makes it attainable to employ linear regression techniques for predicting AQI and to evaluate certain levels of exhaust as severe. It displays sufficient variation over the course of time and space, which makes it possible for an objective assessment of the accuracy, reliability, and validity among various training frameworks and strategies for optimization.

3.2. Data Pre-processing and Normalisation

A strong preliminary processing system was implemented before training the models to ensure that the data being used was of excellent integrity and unbiased. To prevent errors in the procedure of learning, the system approach employed the proper filtering and imputation procedures for dealing with inadequate and uncorrelated parameters. It used the Z-score standardization to normalize all the various pollutant components to the identical extent. As well, a normalization approach centred on Vedic mathematical principles and the Nikhilam Sutra to make mathematical operations more straightforward and streamline computation. The initial results were, moreover, structured into AQI-driven environmental classifications to assist in the evaluation of classification while also making sure every level of pollutant was accurately reflected. The Nikhilam Sutra was used to restore tasks that seem normal and to alleviate mathematical equations instantly. The Urdhva-Tiryakbhyam Sutra was executed to make the matrix and weight multiplication operations in ANN (Artificial Neural Networks) and the kernel of distance computations in Support Vector Machines (SVM) more effective. The operations derived from the philosophy of Vedic mathematics tend to be less complex and far more effective than classical multiplier methods.

3.3. Experimental Configuration and Hyperparameters

The methodology of the study was intended to ensure equitable comparability and reproducible performance among various models and improvement algorithms. The set of data was broken down into three sections which were preparation, validation, and evaluation. The ratio of each part was 70%, 15%, and 15% in that sequential order. It adopted the radial basis function of the kernel to develop SVM algorithms and constructed artificial neural network designs with the appropriate hidden layers and activation procedures. Four adaptable optimization algorithms, namely Adagrad, Adadelta, RMSProp, and Adam. The training rates and optimal convergence parameters should be used with extreme caution. Every experiment was carried out under uniform circumstances, with various iterations that ensured accurate and consistent evaluations of performance.

3.4. The Nikhilam Sutra Technique (NST)

The Nikhilam Sutra (NST) is an elementary method of doing math in Vedic mathematical terms. The Nikhilam Sutra (NST) works especially well when multiplying and dividing figures that share a common base number, such as 10, 100, or 1000. The sutra is founded on the concept that establishing the starting point may render the calculations less complicated by enabling one to observe precisely how far off the results are from that base. This method is a systematic, effective way to facilitate simple math computations.

It becomes smoother as bigger numbers are used. Knowing the variance within the value and the nearest base allows for the method of multiplication to be far simpler, especially when dealing with large quantities. Using numbers that are close to powers of ten makes the Nikhilam multiplication method easier to do. To get the final result, you first figure out how far each number is from the base you chose, and then you do a series of simple math steps. This method not only speeds up calculations but also makes mental math easier by cutting down on the number of steps needed.

3.4.1. Subclasses of Nikhilam Sutra

a) The number is higher than the initial value

$A \times B = [B_0 + (a + b)] \times B_0 + (a \times b), B_0$ = Initial Value, deviation = a, b

If $A = 104 = 100 + 4$, $b = 105 = 100 + 5,$

$B_0 = 100, a = 4, b = 5$

$104 \times 105 = [100 + (4 + 5)] \times 100 + (4 \times 5) = 10920$

b) The number is lower than the initial value

$A \times B = [B_0 - (a + b)] \times B_0 + (a \times b)$

If $A = 95 = 100 - 5$, $b = 96 = 100 - 4,$

$B_0 = 100, a = 5, b = 4$

$95 \times 96 = [100 - (5 + 4)] \times 100 + (5 \times 4) = 9120$

c) The number is higher and lower than the initial value.

$A \times B = [A - b] \times B_0 - (a \times b)$

If $A = 102 = 100 + 2$, $b = 98 = 100 - 2,$

$B_0 = 100, a = 2, b = 2$

$102 \times 98 = [102 - 2] \times 100 - 4 = 9996$

3.4.2. Pseudo Code of Nikhilam Sutra

```
Step 1: Define $A, B \in N$
Step 2: Assign $B_0 = 10^n \ni B_0 \geq Max(A, B)$
Step 3: Assign $a = A - B_0$ , $b = B - B_0$
Step 4: $C = A + b, C = B + a$
Step 5: $M = a \times b$
Step 6: If $|M| < log_{10}(B_0)$:Include an additional zero at the
        start of $C$
Step 7: If $M \geq B_0$
Step 8: $C = C + \left[\frac{M}{B_0}\right]$
Step 9: $M = M|B_0|$
Result = $C \times B_0 + M$
```

3.5. Urdhva Tiryagbhyam (Optimize Matrix Multiplier)

One of the major multiplication methods in Vedic Mathematics is the Urdhva Tiryagbhyam Sutra, which corresponds to "In the vertical direction and Crosswise." The algebra rule of Urdhva Tiryagbhyam sutra (UTS) relates to the method of increasing polynomials.

UTS procedure can be connected to calculate the multiplier for an N by N matrix. This will frame 2N-1 cross items of differing sizes and produce (log2N + 1) littler increase outputs. To figure out the component outcomes, calculations are either upwards, downwards, or across the board.

This offers an intuitive and organized approach to applying multipliers that is highly efficient and performs adequately with regard to conventional and computational effectiveness. The principle holds true for every instance, regardless of how small or large the numbers are in fact. It is particularly beneficial for computer computations because it has integrated redundant information.

Let $\overline{X} = [X_0, X_1, X_2], \overline{Y}_{rev} = [Y_0, Y_1, Y_2]$ (1)

$\overline{X} \cdot \overline{Y}_{rev}$ = i) $M_{R0} = x_0 y_0 = \sum_{i-0}^{0} x_i y_{0-i}$ (2)

ii) $M_{R1} = x_0 y_1 + x_1 y_0 = \sum_{i-0}^{1} x_i y_{1-i}$ (3)

iii) $M_{R2} = x_0 y_2 + x_1 y_1 + x_2 y_0 = \sum_{i-0}^{2} x_i y_{2-i}$ (4)

iv) $M_{R3} = x_1 y_2 + x_2 y_1 = \sum_{i-0}^{3} x_i y_{3-i}$ (5)

v) $M_{R4} = x_2 y_2 = \sum_{i-0}^{4} x_i y_{4-i}$, (6)

since $i \in \{1,2\}$ correspondence to iv) & v)

3.5.1. Pseudo Code of Urdhva Triyakbhyam

```
Step 1: Multiply
$A(a_{n-1,}a_{n-2,} - - - - -, a_0)B(b_{n-1,}b_{n-2,} - - - - -, b_0)$
Step 2: Initialize $P$ $[2n]$ = array with zero
Step 3: Carry =0
Step 4: For $K = 0$ $to$ $2(n - 1)$:Total= 0
Step 5: Total

$= \sum_{i=0}^{k} \begin{cases} A[i] * b[k-i] & \text{if } i < n \text{ and } (k-i) < n \\ 0 & \text{otherwise} \end{cases}$

Step 6: Total =Total + Carry

Step 7: P[k] =Total %10 # to store last digit

Step 8: Carry =Total/10 # to keep the carry for next digit

    $P[2n-1]$ = Carry     # Also add final carry if exists

    Return P []
```

In order to render multiplication more rapid, parameters that perform effectively need to be produced. The Vedic efficiencies have been favored throughout other contemporary innovations that consume fewer resources, work more rapidly, and contribute to more efficient utilization of natural resources. The Framework of Air Quality Forecasting is as follows.

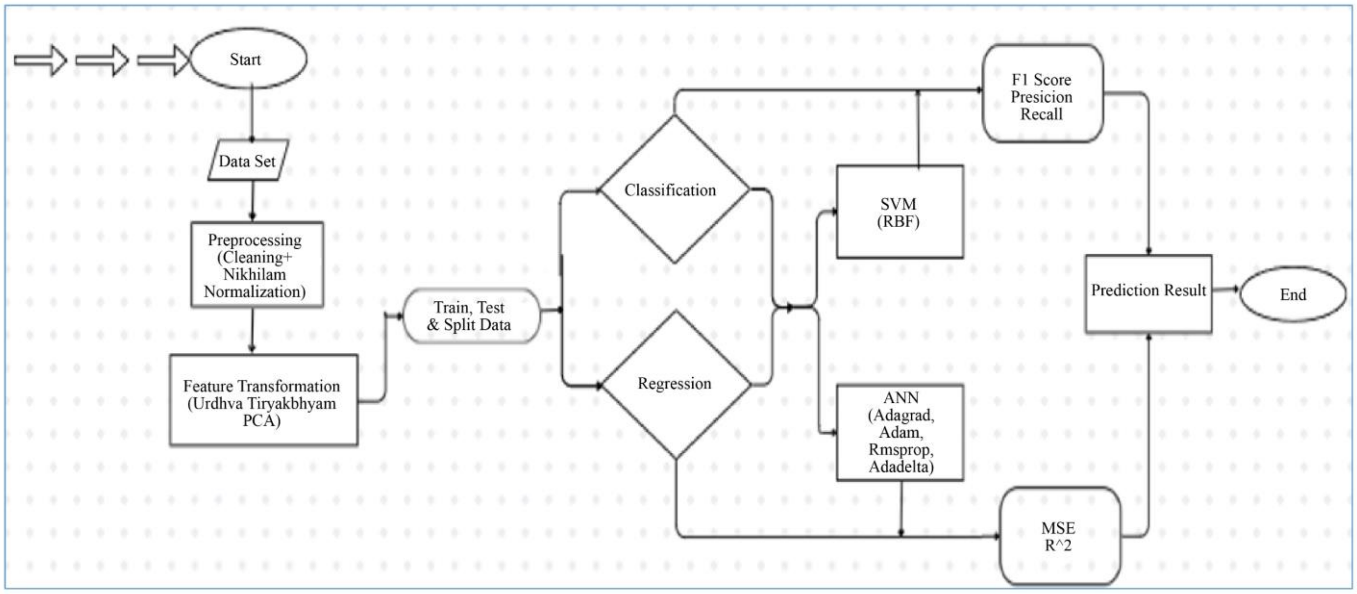

Start
Data Set
Preprocessing (Cleaning+ Nikhilam Normalization)
Feature Transformation (Urdhva Tiryakbhyam PCA)
Train, Test & Split Data
Classification
Regression
SVM (RBF)
ANN (Adagrad, Adam, Rmsprop, Adadelta)
F1 Score Presicion Recall
MSE R^2
Prediction Result
End

Fig. 4 Air Quality Forecasting Methodological Framework

3.6. Support Vector Machine (SVM)

SVM is a type of machine learning approach that recognizes insightful indications and trends in the data. The research investigation implements the SVM technique to organize the information points into numerous categories. The primary goal of a Support Vector Machine, or SVM, is to develop boundaries for decisions that promote classification distinctness.

In multidimensional feature spaces, this boundary is referred to as a hyperplane. The goal is to maximize the margin between classes, defined as the distance between the hyperplane and the nearest data points from each class, thereby improving class separability. Support Vector Machines are particularly effective for handling complex datasets that cannot be separated using simple linear boundaries. Through the use of kernel functions, nonlinear SVMs transform input samples into a higher-dimensional space, where linear separation becomes feasible. By maximizing the margin between classes, SVMs enhance generalization performance and improve the model's ability to classify previously unseen data accurately.

3.6.1. SVM Classification Analysis

To examine a problem using classification in binary form, with the unknown function $f(x): R_n \to \{-1,1\}$ assign each input vector $x \in R_n$ to a class label $y \in \{-1,1\}$ for the data set $D = (X, y)$ where,

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{bmatrix} \in R^{N\times n}, y = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_n \end{bmatrix} \in \{-1,1\} \qquad (7)$$

It is searching for a hyperplane that best differentiates the two segments so as to approximate the function $f(x)$. This is accomplished by optimizing the margin. For

Classifying air quality index (AQI) using SVM with Radial Basis function (RBF), the main decision-making function [7] is

$$f(X) = sign \sum_{i=}^{n} \alpha_i y_i e^{-\gamma \|x_i - x\|^2} + b \qquad (8)$$

Here, using the Nikhilam method can significantly speed up the squaring of numbers.

$$\|x_i - x\|^2 = \sum_k \left(x_{ik} - x_{jk}\right)^2 \tag{9}$$

For the k-th characteristic, this stands for the squared difference. This calculates the distance between two points in feature space; the greater the difference between their values, the greater the distance.

3.6.2. SVM Regression Analysis

Support Vector Regression (SVR), also referred to as SVM regression, is one machine learning method for analyzing regression. Instead of fitting a line to the data points, it finds the hyperplane that best fits the data points in a continuous dimension, which is how it varies from traditional linear regression techniques.

The SVR approach seeks to identify the hyperplane that, within a given margin, traverses as many data points as possible. By employing a kernel function, this method enables SVR to manage non-linear interactions between the variables being used and the target variable while also lowering the prediction error. Because of this, SVM regression is an effective technique for regression jobs in which the target variable and the input variables may have intricate interactions. Nonlinear Support Vector Regression (NSVR) requires mapping into a high-dimensional feature space (xi), where F is the same space as regular SV linear regression. The selected method is the kernel approach. Similar to linear SVR, our approach employs a loss function that overlooks minor inaccuracies. The target functions are shown as [9, 16].

$$Minimize_{w,b,\eta,\eta^*} \ \frac{1}{2}\ \|w\|^2 + C\sum_{i=1}^{l}(\eta_i{}^* + \eta_i) \tag{10}$$

$$subject\ to:\ \begin{cases} y_i - w^T\phi(x_i) - b \le \varepsilon + \eta_i \\ w^T\phi(x_i) + b - y_i \le\ \varepsilon +\ \eta_i{}^* \\ \eta_i, \eta_i{}^* \ge 0 \end{cases} \tag{11}$$

Following the implementation of a Lagrange function, determining the partial derivatives with respect to the primary variables, and establishing the value that results in zero derivatives. The solution can be found here. The Radial Basis Function (RBF), also known as the Gaussian kernel, defines inner products in an altered space. Alternatively, it determines similarity in the modified space.

The function is defined as:

$$k\left(x_i, x_j\right) = \exp(-\gamma * \|x_i - x\|^2) \tag{12}$$

Quadratic Programming is used to determine the regression function when addressing the dual problem is

$$f(x) = \sum_{i=1}^{l} \Delta\alpha_i \exp(-\gamma * \|x_i - x\|^2) + b \tag{13}$$

3.7. Adaptive Gradient Algorithm (AdaGrad)

It is an adaptable method for altering the rate of learning while developing the models to learn. The approach works most effectively while analyzing sparse databases or whenever features have substantially distinct scales. Adagrad never employs one rate for learning. alternatively, it modifies the training rates for all of the parameters based on the gradients that have developed upward over time. The technique makes the model acquire information more clearly by having it concentrate on the most significant aspects. The optimization process needs to do a calculation for each parameter, and this is very important because it helps the optimizer make changes that are effective. The accuracy and reliability of these gradients depend on the loss function and the data used.

The gradient in AdaGrad is determined through calculations as follows [12, 10].

$$g_t = \nabla\theta_t\, J(\theta_t) \tag{14}$$

After that, the subsequent SGD update for each parameter θ_t At each time step, t is as follows:

$$\theta_{t+1} = \theta_t - \eta g_t \tag{15}$$

Adagrad modifies each parameter's overall learning rate η, θ_t at each time step t in its update rule according to the gradients that have been calculated for θ_t in the past:

$$\theta_{t+1} =\ \theta_t - \frac{\eta}{\sqrt{v_t + \epsilon}} \cdot g_t \tag{16}$$

To stop zero from being divided, a factor ∈ is utilized. The usual unit of measurement for an exponential moving average, which is frequently the gradient average, is v_t.

3.8. Root Mean Square Propagation (RMSProp)

This is a proposed approach that adjusts how fast a learning model learns to improve its performance. It looks at the size of recent changes (gradients) for each part of the model and changes the learning speed based on that. This adaptability is essential for navigating the evolving targets and inconsistent difficulties that often present themselves in deep learning assignments. Estimated from the second-order moment of the gradients, the learning rate is reduced and eventually declines. Managing changing objectives is a common function of this optimizer, which is also used for training recurrent neural networks. The principles for updating the RMSProp algorithm include the following [10, 12, 13]:

$$v_t = \beta v_{t-1} + (1-\beta) g_t{}^2 \tag{17}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_{t+\epsilon}}} \cdot g_t \quad (18)$$

A decreasing factor for the moving average is β, and averaged over time, the mean of squared gradients is v_t .

3.9. Adaptive Moment Estimation (ADAM)

It is a widely used optimization technique that adjusts network weights step by step using training information. Adam merges the advantages of Momentum, which helps speed up gradients in the correct direction, and adaptive learning rates, such as AdaGrad and RMSprop. Therefore, Adam influences the rate of learning for every parameter and maintains an ongoing square gradient and an average of the gradient.

Two continuous averages are maintained by Adam: The average is the first moment estimate gradient average, and the second moment estimate is the uncentered variance gradient average. To update the parameter θ (weight, bias), the initial moment m_t is the average of the gradients, adjusted with an exponential weight [10, 14, 15]:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (19)$$

The average of the squared slopes is the second instance v_t , also adjusted with an exponential weight:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) {g_t}^2 \quad (20)$$

More precise estimates of the true first and second moments can be obtained by applying bias correction to m_t and v_t, particularly in the early training stages. Apply bias correction to m_t and v_t to get more accurate estimates of the true first and second moments, especially during the initial training steps:

$$m_t{}^\wedge = \left(\frac{m_t}{1-{\beta_1}^t}\right) \ , v_t{}^\wedge = \left(\frac{v_t}{1-{\beta_2}^t}\right) \quad (21)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t{}^\wedge + \epsilon}} m_t{}^\wedge \quad (22)$$

β Controls the average that fluctuates throughout time. The beta value falls between 0 and 1, with β = 0.9 being a normal number. It means that past gradients are ignored or discarded, and that just averages the gradients from the previous ten iterations. Taking into account more gradients in the averaging procedure when the beta value is large (e.g., β = 0.98).

3.10. Adaptive Delta (Adadelta)

Per-dimension learning rates are possible with the stochastic gradient-based optimization technique Adadelta. The goal of Adadelta, an Adagrad addition, is to lessen its aggressive, monotonically declining learning rate. Adadelta confines the area of cumulative past gradients to a predetermined size ω rather than adding up all prior squared gradients[10].

The sum of gradients is recursively defined as a decaying average of all prior squared gradients, as opposed to inefficiently holding ω previous squared gradients. As a result, the average of all times $E[g^2]_t$ at every step t depends exclusively on both the present gradient and the prior average [17-19].

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1 - \gamma) {g_t}^2 \quad (23)$$

$$\Delta\theta_t = -\eta . g_{t,i} \quad (24)$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \quad (25)$$

$$RMS[g]_t = \sqrt{E[g^2]_t + \epsilon} \quad (26)$$

$$\Delta\theta_t = -\frac{\eta}{RMS[g]_t} g_t \quad (27)$$

Following the application of the parameter modifications, the root mean squared error.

$$E[\Delta\theta^2]_t = \gamma E[\Delta\theta^2]_{t-1} + (1 - \gamma) {\Delta\theta_t}^2 \quad (28)$$

$$RMS[\Delta\theta]_t = \sqrt{E[\Delta\theta^2]_t + \epsilon} \quad (29)$$

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t \theta_{t+1} = \theta_t + \Delta\theta_t \quad (30)$$

4. Result

4.1. Resuhlt for Classification Analysis

The accuracy of air pollution categorization tasks is greatly influenced by the selection of the appropriate optimizer (Adagrad, RMSprop, Adam, Adadelta) for an Artificial Neural Network. Utilizing Vedic Mathematics in combination with optimizers such as Adadelta, RMSprop, and Adam significantly enhances prediction accuracy for certain pollutants and categories. They typically achieve better results than SVM. The grouped bar chart analysis shows that the choice of model and optimizer significantly affects the F1 Score for different air pollutants and pollution levels. For PM2.5, the RMSprop and Adam-based ANN consistently produce excellent F1 Scores for all classes; SVM also works well for Medium. When it comes to PM10, the Adadelta and SVM models outperform all others, attaining F1 scores of about 1.0 for the Low class and over 0.8 for the Medium and High classes. In account of NO2, SVM performs moderately for the Low class, whereas Adam and Adadelta show excellent results for the Medium and High classes. In favour of O3, RMSprop is also competitive, although Adagrad and Adam are particularly effective for Low and Medium courses.

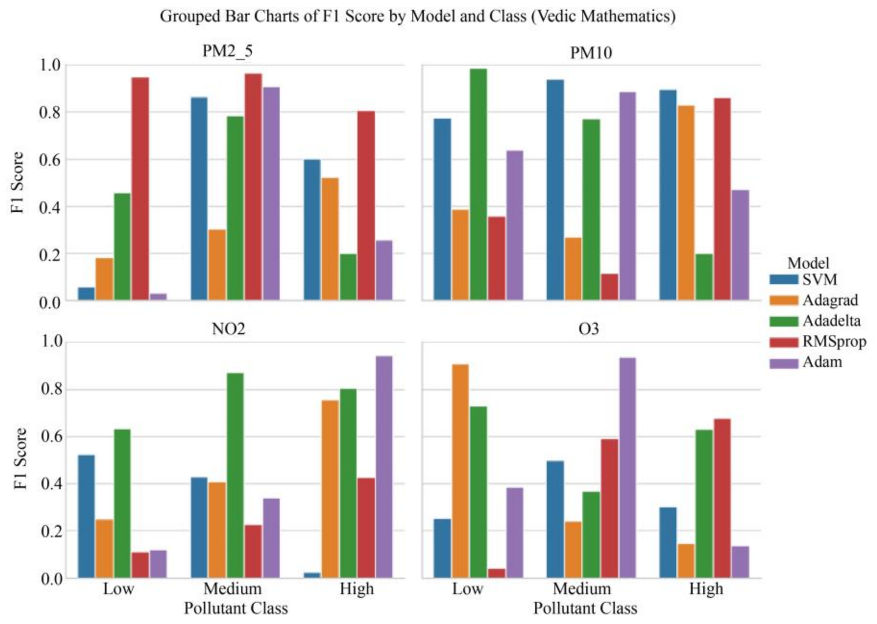


Fig. 5 Grouped bar chart of F1 score by model and class (vedic mathematics)

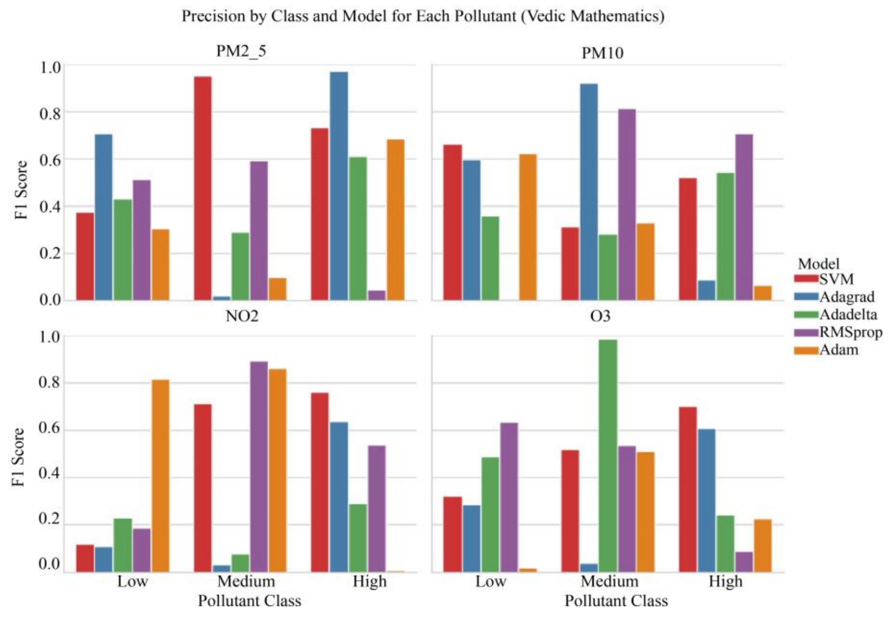


Fig. 6 Grouped bar chart of precision by model and class (vedic mathematics)

The Precision evaluation study reveals that artificial neural network optimization approaches like Adadelta, Adagrad, and Adam frequently generate accurate estimations instead of conventional approaches, notably for the moderate to serious emission classes of the four major pollutants (PM2.5, PM10, NO2, and O3). However, the model using the SVM continues to produce estimations with an outstanding extent of reliability. The outcome demonstrates how, once anticipating airborne contaminant segments, customized Artificial Neural Network (ANN) models can be more accurate than SVM. A comparison of recall scores across five machine learning models (SVM, Adagrad, Adadelta, RMSprop, and Adam) shows considerable variation in model performance based on pollutant type (PM2.5, PM10, NO_2, O_3) and pollution level (Low, Medium, High). For PM2.5, RMSprop had the highest recall in the Low class, whereas for PM10, Adagrad and Adadelta did well in the Low and Medium classes, with Adam performing strongly in the High class. Adagrad excelled for NO_2 in the Medium class, whereas Adadelta performed competitively at the Low level. For O_3, Adadelta and RMSprop had higher recall in Low and Medium classes, while recall scores declined for High levels across all models.

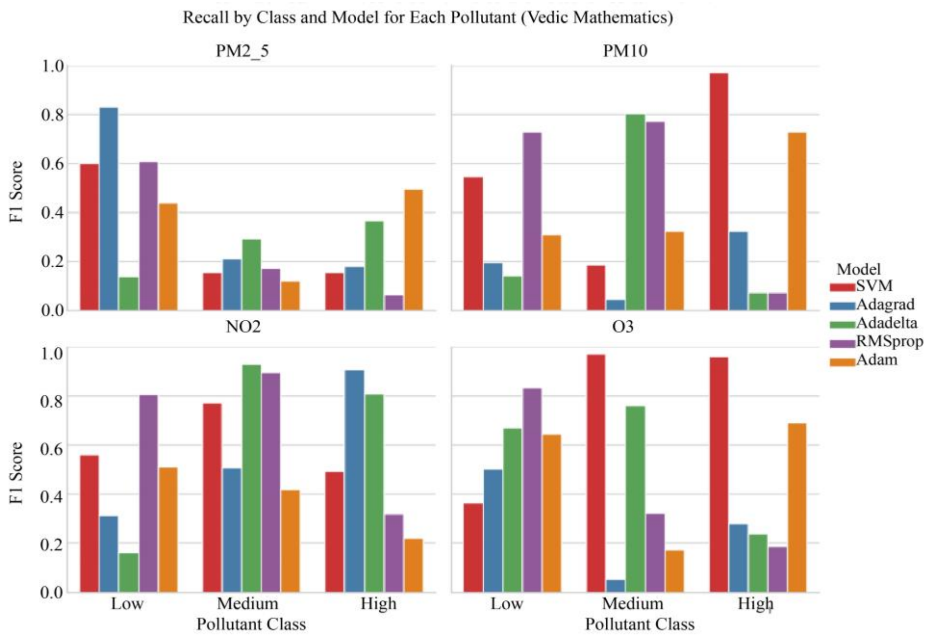


Fig. 7 Grouped bar chart of recall by model and class (vedic mathematics)

4.2. Result for Regression Analysis

The scatter plot presented in Figure 8 shows how well the SVM model with an RBF kernel predicts PM2.5 levels by comparing the real values with the predicted ones. The red the dashed line shows the perfect situation where predictions exactly match the real measurements. It can be seen that when PM2.5 levels are low, the predicted values are similar to the actual values. This shows that the model works well for low to moderate pollution levels. As the real PM2.5 levels go up, the predictions usually come in lower than expected, showing that Individuals typically underestimate the extent of PM2.5 concentrations. This result indicates that the SVM with an RBF kernel can usually find trends in PM2.5 data, but it has trouble predicting times when contamination is extremely high. Figure 9's line plot demonstrates the way actual and anticipated PM2.5 levels compare for the initial 100 samples tested. The blue line shows the real PM2.5 levels, which change a lot and sometimes spike quickly, which means that pollution levels go up swiftly. The model's predicted values are shown through the green line.

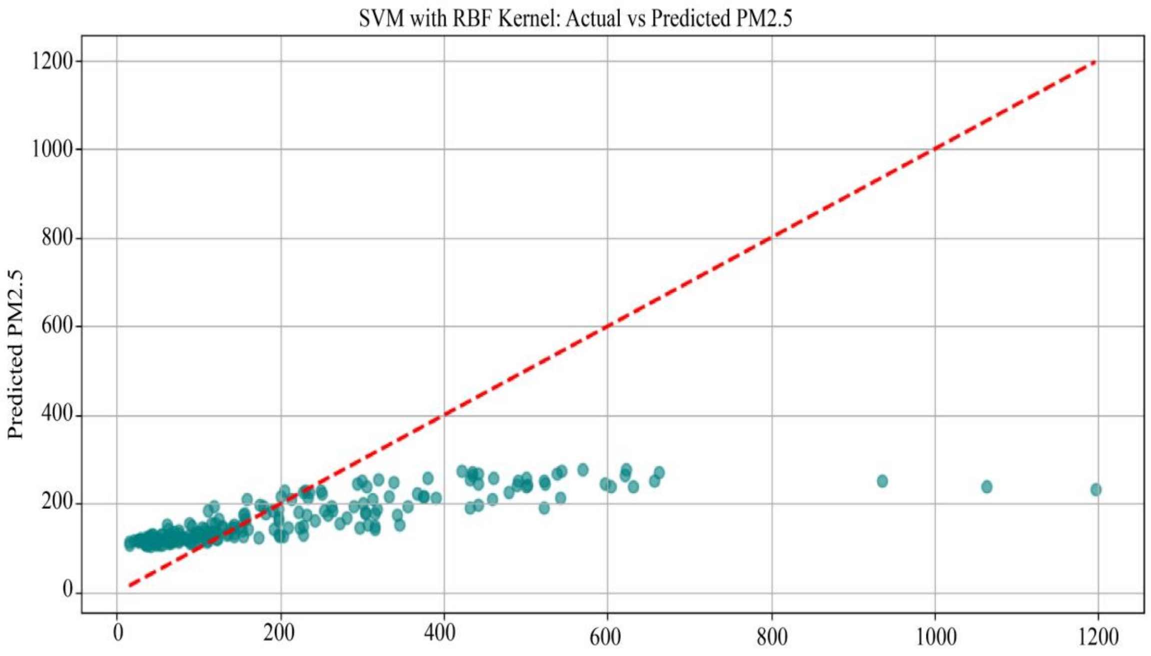


Fig. 8 Grouped bar chart Actual Vs Predicted (SVM with RBF kernel)

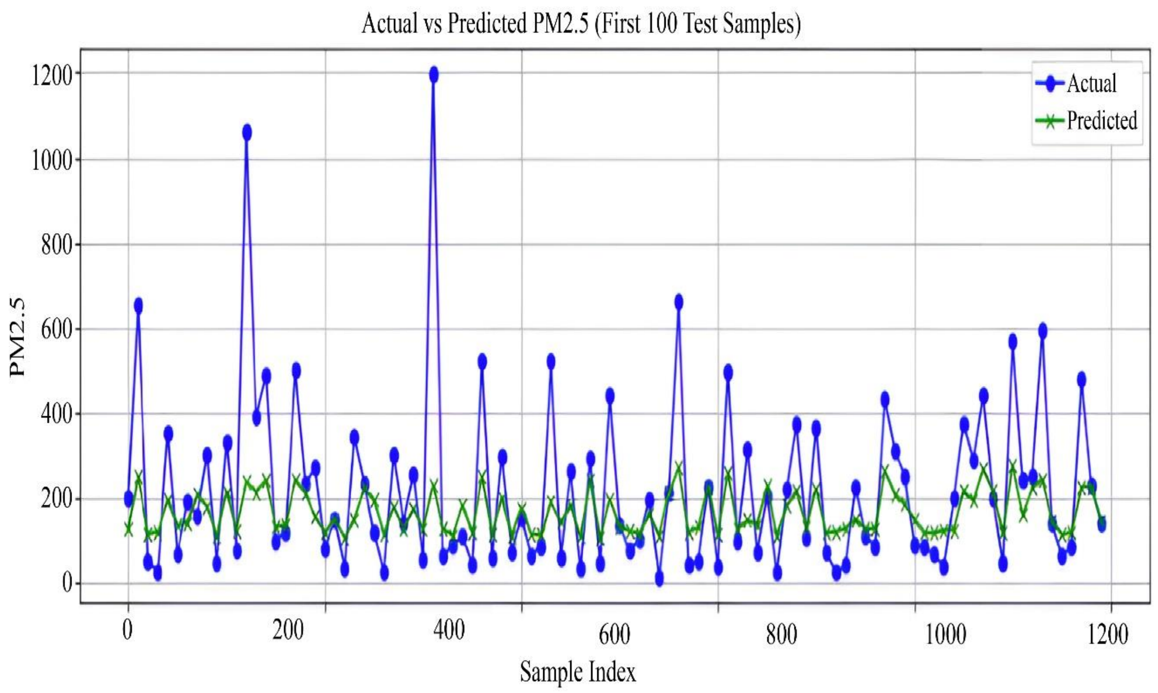


Fig 9 Actual Vs Predicted PM2.5 (SVM for 100 samples)

The predicted trend largely mirrors the real data's pattern, albeit smoother and devoid of significant jumps. This shows that the model is good at predicting the overall pattern of PM2.5 levels, but it often misses sudden times when pollution is very high.

The comparison charts clearly show how well different ANN optimizers work in predicting PM2.5 levels. The (Figure 10) scatter plots show that the RMSprop and Adam optimizers give predicted values that are very close to the real measurements. This is shown by the points grouping closely along the red ideal line. On the other hand, the Adagrad & Adadelta optimizer sees PM2.5 levels as lower than they really are. Most of its predictions are close to low values, and it does not do a good job of predicting higher levels. This pattern is clearly shown in the line graph (Figure 11) of the first 100 test samples. RMSprop and Adam closely match the real PM2.5 trend, including sudden increases and changes. In contrast, Adagrad & Adadelta continually predict lower values and miss the high points. These results show that RMSprop and Adam optimizers work much better than Adagrad, giving more dependable predictions for both medium and high pollution levels.

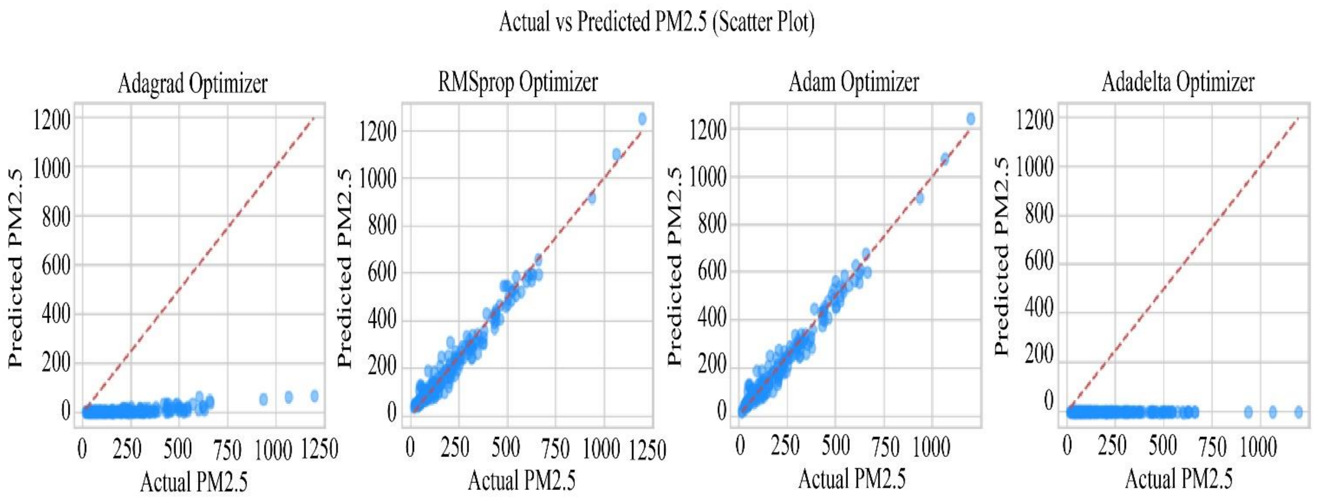


Fig. 10 Actual Vs Predicted PM2.5 (Adagrad, RMSprop, Adam,Adadelta)

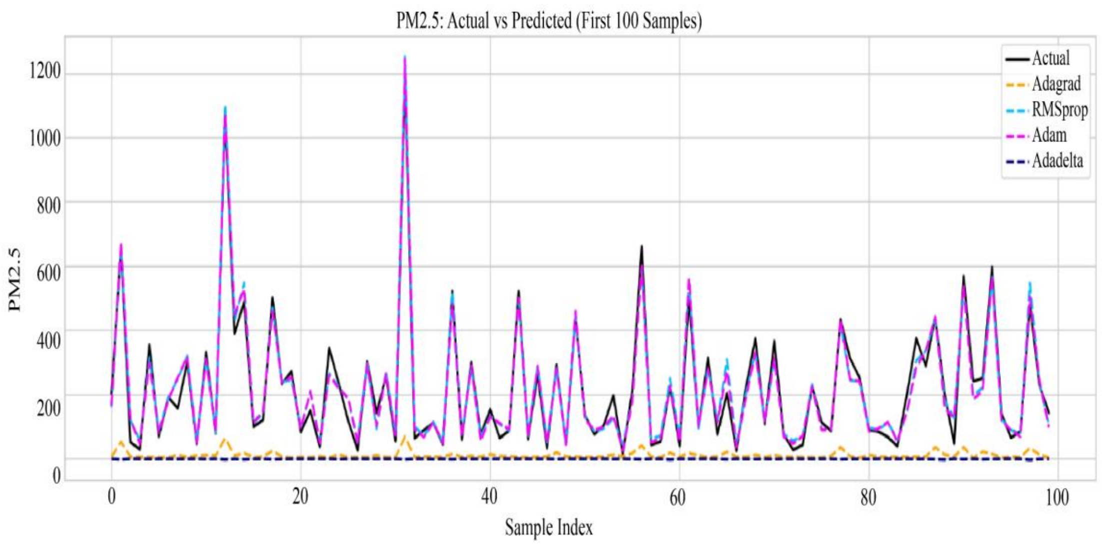


Fig. 11 Actual Vs. Predicted PM2.5 (Adagrad, RMSprop, Adam, Adadelta) for 100 samples

4.3. Statistical Investigation

Table 1. A statistical examination of methods of regression

Analysis	Mean MSE	Deviation from standard
Linear Regression	8746.47	±264.01
Support Vector Regression	263.49	±3.69
Multilayer Perceptron	113.62	±18.34

Table 2. A statistical analysis of numerous models of classification

Analysis	Mean F1	Deviation from standard
Logistic Regression	0.6919	±0.0063
Support Vector classification	0.7406	±0.0022
Multilayer Perceptron	0.9509	±0.0087

Table 3. Relevant Statistics

Comparison	t-statistic	p-value
MLP vs SVR (Regression)	-14.0864	0.000147

For assurance of scientific validity, the suggested design received assessments throughout several independent iterations. It used the average Mean Squared Error (MSE) to see whether or not the statistical model performed, and the macro-averaged F1-value to see the manner in which the method of classification was executed effectively. The multi-layer perceptron (MLP) model performed far better than SVR and regression models, with a mean MSE of 113.62 ± 18.33. The MLP model exhibited extremely accurate discrimination of classes with an enormous F1-score of 0.9509 ± 0.0087 for AQI severity identification. Statistical significance testing with paired t-tests showed that the suggested method was strong. The results showed that the MLP model's performance improvements over the SVR model were statistically significant ($p < 0.001$).

5. Conclusion

The selection of the optimizer, according to the comparison of actual and projected PM2.5 values, has a substantial influence on the model's performance. Both the RMSprop and Adam optimizers offer extremely precise predictions that accurately reflect the real-world PM2.5 values, successfully capturing abrupt peaks and variations. Despite this, the Adagrad optimizer consistently underestimates PM2.5 concentrations and is incapable of accounting for larger pollution volumes.

These results demonstrate that Adam and RMSprop are superior optimizers for air quality forecasting applications where the accurate prediction of extreme pollutant values is crucial. Therefore, optimizing for PM2.5 prediction models can improve the trustworthiness and versatility these designs illustrate. ANN, SVM, and Vedic mathematics (in particular, the multiplier) are highly significant innovations that possess an abundance of opportunities for enhancing the outcomes. The present research provides an innovative method for integrating these advancements for enhancing functionality and compares multiple optimizer types. Findings reveal that Vedic neural networks display exceptional agility in comparison with conventional neurons. The results from this research can be extended with the use of Vedic addition algorithms in neurons instead of standard adders, which is likely to enhance computational speed. Further study must be conducted to substantiate the functionality advantages resulting from Vedic neurons by collecting precise measurements and evidence. Statistical research indicates that the proposed models perform consistently throughout several runs. The significance test reveals that the observed performance increases are statistically significant and not attributable to random variation, verifying the experimental results' robustness and reliability.

References

[1] Yichen Wang et al., "Impacts of Natural and Socioeconomic Factors on $PM_{2.5}$ from 2014 to 2017," *Journal of Environmental Management*, vol. 284, pp. 1-9, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Biao Sun et al., "The Relationship Between Urbanization and Air Pollution Affected by Intercity Factor Mobility: A Case of the Yangtze River Delta Region," *Environmental Impact Assessment Review*, vol. 100, pp. 1-31, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Nikhil Vilasrao Deshmukh et al., "Predictive Analytics for Air Quality Classification: A Multi- City Study in India," *International Journal of Environmental Science*, vol. 11, no. 4, pp. 1508-1520, 2025. [Publisher Link]

[4] C. Karkuzhali, and R. Padmapriya, "Multivariate Regressive Gradient Spiral Optimized Deep Belief Learning for Crop Yield Prediction," *International Journal of Environmental Science*, vol. 11, no. 14s, pp. 327-345, 2025. [Publisher Link]

[5] R. Karthi Kumar, and S.P. Vimal, "Comparative Analysis of Vedic Multiplier using Vedic Sutras with Existing Multipliers in Biomedical Application," *Measurement: Sensors*, vol. 36, pp. 1-10, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[6] K. Deergha Rao, Ch. Gangadhar, and Praveen K. Korrai, "FPGA Implementation of Complex Multiplier using Minimum Delay Vedic Real Multiplier Architecture," *2016 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON)*, Varanasi, India, pp. 580-584, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ke-Lin Du et al., "Exploring Kernel Machines and Support Vector Machines: Principles, Techniques, and Future Directions," *Mathematics*, vol. 12, no. 24, pp. 1-58, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[8] Jing Geng et al., "Support Vector Machine Regression (SVR)-based Nonlinear Modeling of Radiometric Transforming Relation for the Coarse-Resolution Data-Referenced Relative Radiometric Normalization (RRN)," *Geo-Spatial Information Science*, vol. 23, no. 3, pp. 237-247, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[9] Xiaotong Hu, "*Support Vector Machine and its Application to Regression and Classification*," MSU Graduate Theses, Missouri State University, pp. 1-57, 2017. [Google Scholar] [Publisher Link]

[10] Sebastian Ruder, "An Overview of Gradient Descent Optimization Algorithms," *arXiv Preprint*, pp. 1-14, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[11] Diederik P. Kingma, and Jimmy Ba, "Adam: A Method for Stochastic Optimization," *arXiv Preprint*, pp. 1-15, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[12] Mohamed Reyad, Amany M. Sarhan, and M. Arafa, "A Modified Adam Algorithm for Deep Neural Network Optimization," *Neural Computing and Applications*, vol. 35, no. 23, pp. 17095-17112, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Tijmen Tieleman, "Lecture 6.5-RmsProp: Divide the Gradient by a Running Average of its Recent Magnitude, *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, 2012. [Google Scholar]

[14] Shiliang Sun et al., "A Survey of Optimization Methods from a Machine Learning Perspective," *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3668-3681, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[15] Ahmet Cevahir Cinar, and Narayanan Natarajan, "An Artificial Neural Network Optimized by Grey Wolf Optimizer for Prediction of Hourly Wind Speed in Tamil Nadu, India," *Intelligent Systems with Applications*, vol. 16, pp. 1-11, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[16] Alireza Askarzadeh, and Alireza Rezazadeh, "Artificial Neural Network Training using a New Efficient Optimization Algorithm," *Applied Soft Computing*, vol. 13, no. 2, pp. 1206-1213, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[17] Gulfraz Syed et al., "The Role of Artificial Neural Networks in Machine Learning," *Neuro Quantology*, vol. 20, no. 13, pp. 3239-3251, 2022. [Google Scholar]

[18] Faisal Mehmood, Shabir Ahmad, and Taeg Keun Whangbo, "An Efficient Optimization Technique for Training Deep Neural Networks," *Mathematics*, vol. 11, no. 6, pp. 1-22, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Matthew D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *arXiv Preprint*, pp. 1-6, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[20] Wentao Yang et al., "Prediction of Hourly $PM_{2.5}$ using a Space-Time Support Vector Regression Model," *Atmospheric Environment*, vol. 181, pp. 12-19, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[21] Xiang Li et al., "Deep Learning Architecture for Air Quality Predictions," *Environmental Science and Pollution Research International*, vol. 23, no. 22, pp. 22408-22417, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ujjwal Kumar, and V.K. Jain, "ARIMA Forecasting of Ambient Air Pollutants (O_3, NO, NO_2 and CO)," *Stochastic Environmental Research and Risk Assessment*, vol. 24, no. 5, pp. 751-760, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[23] Ying Zhang et al., "A Predictive Data Feature Exploration-based Air Quality Prediction Approach," *IEEE Access*, vol. 7, pp. 30732-30743, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[24] Saverio De Vito et al., "On Field Calibration of an Electronic Nose for Benzene Estimation in an Urban Pollution Monitoring Scenario," *Sensors and Actuators B: Chemical*, vol. 42, no. 2, pp. 750-757, 2008. [CrossRef] [Google Scholar] [Publisher Link]