Computers, Materials & Continua
DOI:10.32604/cmc.2022.021968
Article

Tech Science Press

SMOTEDNN: A Novel Model for Air Pollution Forecasting and AQI Classification

Mohd Anul Haq*

Department of Computer Science, College of Computer and Information Sciences, Majmaah University Almajmaah, 11952, Saudi Arabia
*Corresponding Author: Mohd Anul Haq. Email: m.anul@mu.edu.sa
Received: 22 July 2021; Accepted: 23 August 2021

Abstract: Rapid industrialization and urbanization are rapidly deteriorating ambient air quality, especially in the developing nations. Air pollutants impose a high risk on human health and degrade the environment as well. Earlier studies have used machine learning (ML) and statistical modeling to classify and forecast air pollution. However, these methods suffer from the complexity of air pollution dataset resulting in a lack of efficient classification and forecasting of air pollution. ML-based models suffer from improper data preprocessing, class imbalance issues, data splitting, and hyperparameter tuning. There is a gap in the existing ML-based studies on air pollution due to improper data handling and optimization. The present investigation aims to bridge these gaps and aid in effective air pollution classification and forecasting. Five ML models were developed, including one novel model named SMOTEDNN (Synthetic Minority Oversampling Technique with Deep Neural Network) to address air pollution classification. All five models utilized efficient data pre-processing and rigorous hyperparameter optimization. Three forecasting models were developed to forecast air pollution for one step-index based on statistical autoregression. All developed models in present investigation showed higher accuracy. Significantly, the novel model SMOTEDNN achieved an accuracy of (99.90%) higher than the other models from the current investigation and previous studies.

Keywords: Air pollution; smote; dnn; classification; autoregression

1 Introduction

Globally, due to rapid industrialization and urbanization, air pollution is increasing. Air pollution poses severe risks for the environment, creating health-related hazards and worsening climate change. Notably, smog from waste products such as carbon, nitric oxide (NO), carbon monoxide (CO), hydrocarbons, and synthetic nectar from cellular sources affect the environment [1]. The threat of air pollutants, especially particulate matter (PM), is serious enough to cause a higher rate of mortality, as advised by the World Health Organization (WHO) [2,3]. Additionally, an increasing number of vehicles are responsible for increasing pollutants such as NO_2, CO, NH_3, PM2.5, and PM10, whereas pollutants such as and SO_2, CO, O_3, B (Benzene), T (Toluene), and

X (Xylene) are coming from industrial sources. India stands at the second place in terms of severity of pollution after Kuwait, based on higher Particulate Matter (PM) concentrations [4–6].

Previous studies have used statistical models, mathematical models, and Machine Learning (ML) models to classify and forecast air pollution. Reference [7] used Recurrent Neural Networks (RNN) for classification of ozone and achieved an accuracy of 81%. Reference [8] also used RNN to analyze PM 2.5 and PM10 with 95% accuracy [9] used Support Vector Regression and ensemble model to classify PM10 with an accuracy of 96%. Reference [10] used multiple ML models to classify air pollution and found that logistic regression provides an accuracy of 93%. However, parameter tuning was missed. Majority of studies used ANN [9–12] hybrid ANN [13–15] and ML [9,10], [16–18] to forecast the air quality. However, due to the complexity of the dataset due to trend and seasonality, most models lack efficient classification and forecasting of air pollution [19]. Given the learning ability and complex data handling capacity of ML, the use of ML models has rapidly increased [20]. However, critical issues such as data pre-processing, class imbalance issues, data splitting, and hyper-parameter tuning have been poorly addressed to optimize the performance of the models. Specifically, most studies showed high accuracy for the class with more observations and low accuracy for the class with less observation; clearly, illusory accuracy has been achieved due to all these issues. ML models can provide output to almost any given input based on training; however, proper data pre-processing and hyperparameter tuning can improve the model in terms of accuracy, sensitivity, and stability. There is a gap in the collective findings of the existing ML-based air pollution studies due to improper data handling and optimization [6,21]. The present investigation aims to bridge the gaps for more effective air pollution classification and forecasting. Five ML models were developed, including one novel model named SMOTEDNN to address air pollution classification. All five models utilized efficient data pre-processing and rigorous hyperparameter optimization. Three forecasting models were developed to forecast air pollution for one step-index based on statistical autoregression.

1.1 SMOTEDNN

Previous studies focused on getting higher accuracy values, with lesser attention on the illusory accuracy due to class imbalance in the dataset. Classification on imbalanced dataset showed less accuracy for minority class (fewer observations) and high accuracy for majority class (more observations). SMOTE was integrated with DNN in the current investigation to overcome the issue of class imbalance. It oversampled the values of the minority class based on duplicating minority class values. These new values do not add new information to the algorithm; however, new values were synthesized from existing values. SMOTE randomly selects a minor class and creates a new value based on k nearest neighbors randomly. The SMOTE was combined with DNN (SMOTEDNN) to classify air pollution.

Neural Network (NN) belongs to the feedforward artificial neural network (ANN); consists of three layers: an input, a hidden, and an output layer. The higher number of layers for ANN can be defined as DNN. The node of DNN mainly uses a non-linear activation function, excluding the input nodes. The primary benefit of automatic feature extraction in DL-based models makes it a widespread choice for classification [7,8]. The main components of the DNN have been delineated in the ensuing sub-sections.

1.1.1 Activation Layer

A neural network requires an activation function to make predictions. The rectifier activation function (ReLU) is one of the default activation functions for DNN-based applications; it adds

nonlinearity to the network. ReLU output 0 for negative value and output the same value for non-negative values. Softmax is an output layer function used in the output layer for classification in a neural network. Softmax predicts a multinomial probability distribution with more than two classes.

1.1.2 Dense Layer

The dense layer in a neural network is deeply connected. Each neuron in the dense layer receives input from all neurons of its preceding layer. It uses a linear operation function to map every input with every output.

1.1.3 Training

Models such as neural networks use learning algorithms to minimize errors. For ANN, one of the leading learning algorithms is backpropagation, which computes the gradient of a function to fine-tune the network parameters for error minimization.

1.2 XGBoost

XGBoost (eXtensive Gradient Boosting) is a decision-tree-based ensemble ML model or optimized gradient boosting algorithm based on gradient descent algorithm. It includes parallelization, efficient handling of missing data, tree pruning, and regularization to prevent overfitting. XGBoost is one of the best algorithms in processing time and performance when compared to other models. XGBoost uses parallelized implementation approach to process the sequential trees. XGBoost and Random Forest (RF) are both decision tree-based models, and the difference between them is that XGBoost can minimize errors where RF cannot.

1.3 Random Forest

Random forest is a supervised classification algorithm. RF produces decision trees on data samples and utilizes each tree for prediction based on an ensemble of voting or bagging. Bagging allows producing various subsets of the data randomly from the training data used to train the decision trees. RF is based on the classifiers $c(a|\Theta_1),\ldots, c(a|\Theta_k)$ related to classification tree with parameters Θ_k selected randomly from a model random vector Θ. The final classification f(a) uses an ensemble of $\{c_k\ (a)\}$ where the best fit classification calculates based on voting from each tree for input a. The dataset $d = \{(ai, bi)\}_{i=1}^{n}$ is trained on the collection of classifiers $\{c_k\ (a)\}$.

1.4 Support Vector Machine (SVM)

SVM is an ML algorithm that carries out classification using an optimal hyperplane. Generally, SVM classifiers are non-linear, aiming to find a higher margin to separate the classes in feature space [22]. SVM can be defined as [22,23]: (i) Suppose a set of training vectors T, where

$$T = \{(a_1, b_1), (a_2, b_2), (a_3, b_3), \ldots, (a_n, b_n)\} \tag{1}$$

Here, xi $\in \mathbb{R}^n$ where *i= 1, 2, 3,…, n.* (ii) T can be defined as $b_i = \{-1, 1\}$ for two classes. (iii) T is a linearly separable hyperplane using the function $d(x)$ as:

$$d(x) = < w, a > + c = \sum_{i=1}^{n} (w_i a_i) + b = 0 \tag{2}$$

where a is an independent variable, w is weight calculated using the model, and c is a constant.

SVM algorithm maximizes the margin of the hyperplane from the training vectors as

$$\underset{w,b}{min}\frac{1}{2}||w||^2 \tag{3}$$

SVM calculates the cost function as

$$©(w,c;\alpha)=\frac{1}{2}||w^2||-\sum_{i=1}^{n}\alpha_i(b_i[w.a_1+c]-1) \tag{4}$$

Here, the set $\alpha=(\alpha_1,\alpha_2,\ldots,\alpha_n)^T\in\mathbb{R}_+^n$ is the Lagrangian multiplier.

To overcome the finding optimum hyperplane, the penalty parameter 'C' and slack variables ξ_i were introduced [23,24]:

$$b_i((w.a_i)+c)\geq 1-\xi_i \tag{5}$$

The equation used to maximize the margin for the optimal hyperplane is given as:

$$min\frac{1}{2}||w||^2+C\sum_{i=1}^{n}\xi_i \tag{6}$$

1.5 KNN

The KNN algorithm is a supervised ML algorithm, and it assumes that a value of a point is similar to the values that exist in the neighbors. KNN is based on the principle of obtaining the value of a point using neighbor points in the dataset based on distance. Thus, it works on the principle of choosing the value of K(neighbors) near to the point of interest and voting for the most frequent class. The high number of K reduces the noise, and local anomalies add more error towards the decision boundaries.

The primary issue with KNN is that it become slows as the data grows. The Euclidean distance (d) for two points (a_1-a_2) and (b_1-b_2) can be obtained using

$$d=((a_1-b_1)^2+(a_2-b_2)^2)^{\frac{1}{2}} \tag{7}$$

For n-dimensional space, d can be obtained as

$$\mathrm{d}_e=\left[\sum_{i=0}^{n}(a_i-b_i)^2\right]^{\frac{1}{2}} \tag{8}$$

2 Dataset

The National Air Quality Monitoring Program (NAMP) is a nationwide program from the Central Pollution Control Board (CPCB) of India. It aims to monitor the levels of air pollutants from 793 active stations spreading over 344 cities from 29 states of India. The present study used a dataset released under the NAMP program (http://www.cpcbenvis.nic.in/air_quality_data.html) from Jan 01, 2015, to July 07, 2020. The air pollutants analyzed in the present investigation are NOx, Nitrogen Oxide (NO), Nitrogen dioxide (NO_2), Particulate Matter (PM2.5, and PM10), Sulphur Dioxide (SO_2), Carbon monoxide (CO), Ammonia (NH_3), Ozone (O_3) Benzene (B), Toluene

and Xylene(X) (Fig. 1). In addition, the dataset contains an air quality index (AQI) parameter, which is an indication used by government authorities to categorize the pollution in terms of its severity. The six AQI values and their ambient concentrations with health consequences are given in Fig. 2. (Source: https://app.cpcbccr.com/AQI_India/).

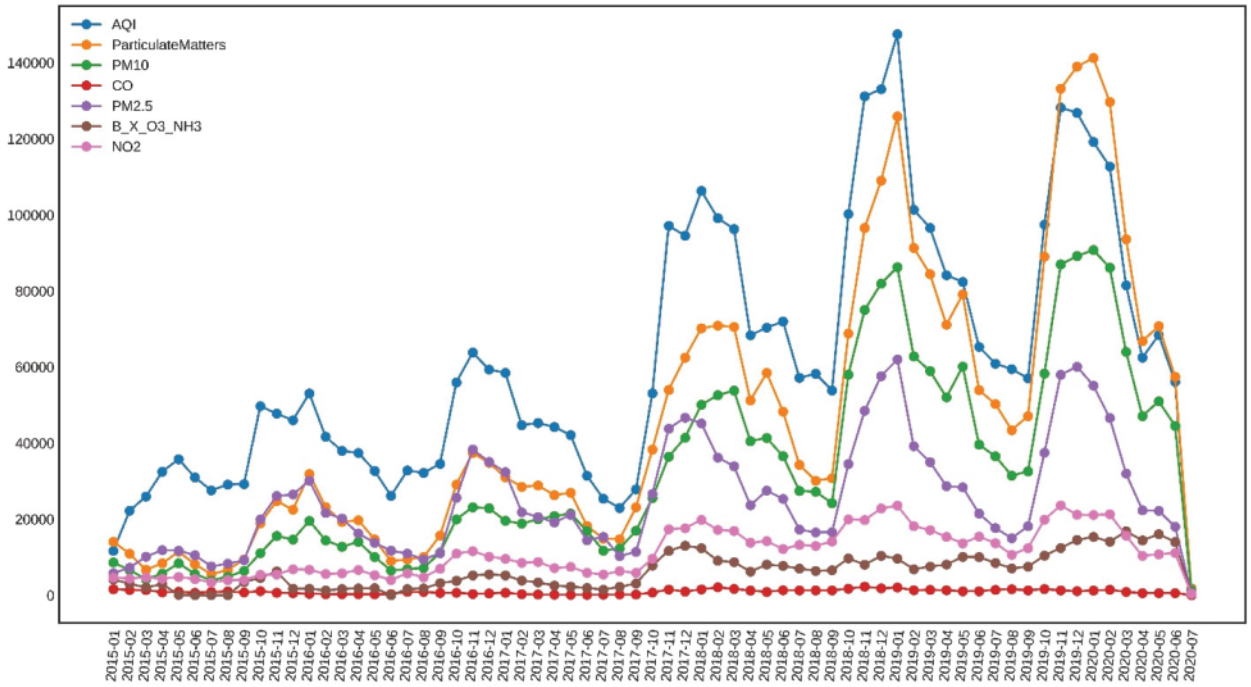


Figure 1: Air pollution parameters from 793 stations across India from Jan 2015 to July 2020 daily

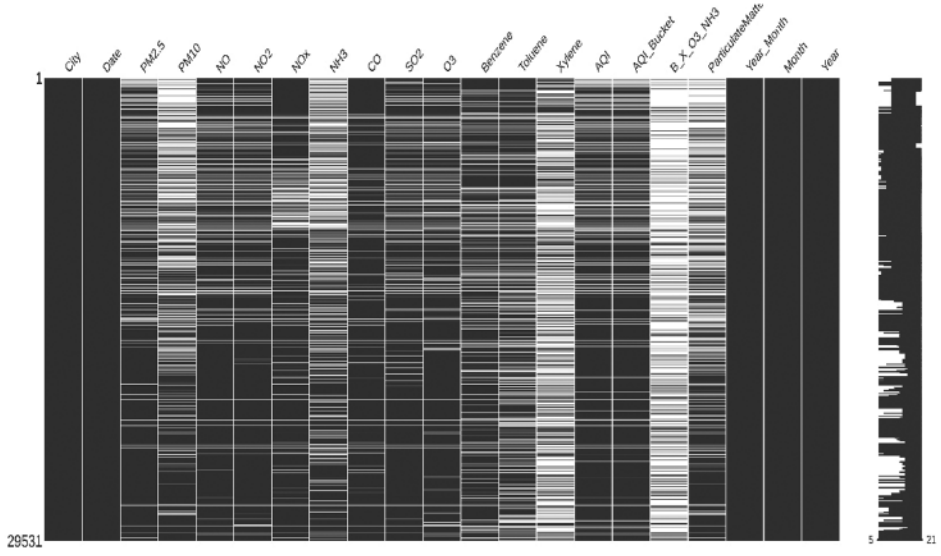
AQI	Remark	Color Code	Possible Health Impacts
0-50	Good		Minimal impact
51-100	Satisfactory		Minor breathing discomfort to sensitive people
101-200	Moderate		Breathing discomfort to the people with lungs, asthma and heart diseases
201-300	Poor		Breathing discomfort to most people on prolonged exposure
301-400	Very Poor		Respiratory illness on prolonged exposure
401-500	Severe		Affects healthy people and seriously impacts those with existing diseases

Figure 2: Six AQI values and their ambient concentrations with health consequences

3 Methodology

3.1 Data Pre-Processing

To improve the understanding of the data, handling missing values, and making data ready for modeling, the raw data underwent the data cleaning process. The primary step was to understand the missing values in the dataset (Fig. 3). It was clear from Fig. 3 that B, X, T, O_3, and NH_3 are among the highest missing values. On the other hand, the less missing value pollutants were CO, NO, NO_2, SO_2, O_3, NOx, PM 2.5, and AQI values. The missing values were removed using Pandas dropna function, where any NA values are present in any row/column. The fields city, date, Year_Month, and AQI_Bucket were removed based on their substitute fields in the dataset to avoid redundancy.



	# of Null	Percentage
B_X_O3_NH3	22788.0	77.2
Xylene	18109.0	61.3
ParticulateMatters	11899.0	40.3
PM10	11140.0	37.7
NH3	10328.0	35.0
Toluene	8041.0	27.2
Benzene	5623.0	19.0
AQI_Bucket	4681.0	15.9
AQI	4681.0	15.9
PM2.5	4598.0	15.6
NOx	4185.0	14.2
O3	4022.0	13.6
SO2	3854.0	13.1
NO2	3585.0	12.1
NO	3582.0	12.1
CO	2059.0	7.0
Date	0.0	0.0
Year	0.0	0.0
Month	0.0	0.0

Figure 3: The missing values and % of pollutants from the dataset

3.2 Data Analysis

The data pertaining to various air pollutants were analyzed to understand city-wise concentrations of various pollutants (Fig. 4a). Delhi showed the highest levels for PM values. The average levels of the pollutants from 25 cities are given in Fig. 5. Ahmedabad showed very high concentrations for SO_2 and NO_3 values. Delhi showed high values for SO_2 and NO values, whereas Kochi showed the highest level of NO. The AQI and CO values for Ahmedabad were the highest among all cities for the observation period, followed by Delhi, which showed AQI and PM10 values on the higher side. The correlation between all air pollution parameters is given in Fig. 4b. The correlation values between pollutants were statistically significant emphasizing the high pollution levels and interrelationships between pollutants. It was essential to understand the AQI interrelationships with each pollutant, to understand which ones were contributing most significantly towards the index. Interestingly, this study found that AQI values were highly related to PM 2.5, NO, and NO_2, based on a correlation value of 0.8 (CI = 95%). SO_2 and O_3 followed next, based on a correlation value of 0.7 (CI = 95%). Rest of the pollutants showed correlation values between 0.2 to 0.5.

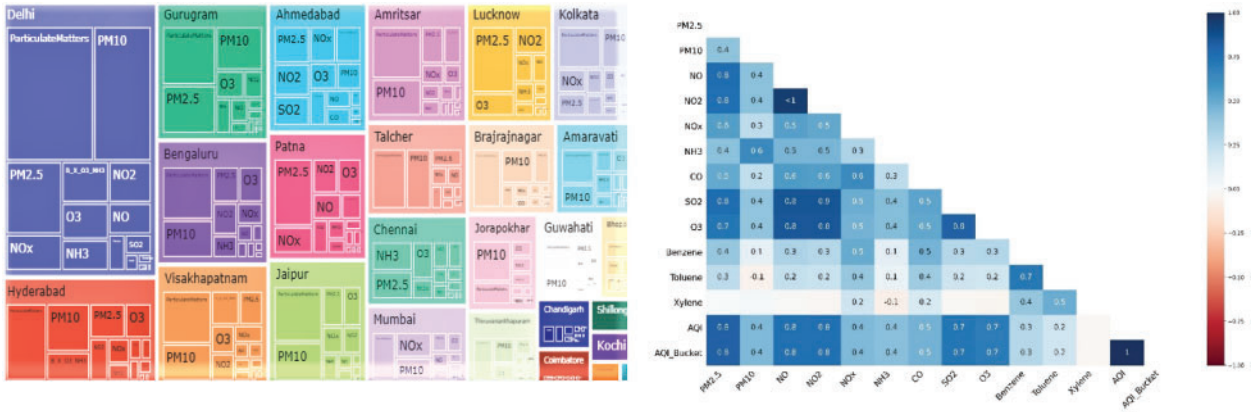
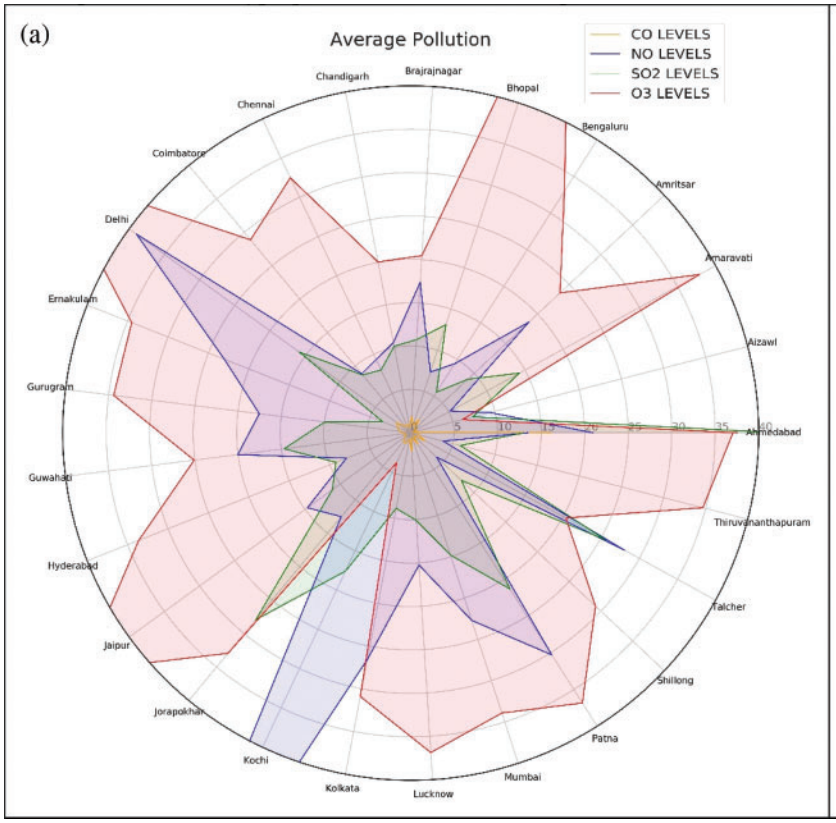


Figure 4: (a) The city-wise proportion of pollution; (b) Correlation heatmap between all air pollution parameters



(b)

City	AQI	PM10	CO
Aizawl	34.765766	23.352857	0.283628
Shillong	53.795122	42.199170	0.239675
Coimbatore	73.023256	37.982137	0.949321
Thiruvananthapuram	75.878327	52.863659	0.951378
Ernakulam	92.359477	48.335633	1.627848
Bengaluru	94.318325	83.243287	1.838634
Amaravati	95.299643	76.362020	0.627904
Chandigarh	96.498328	85.656546	0.631349
Kochi	104.284810	67.335432	1.296667
Mumbai	105.352258	96.745059	0.568362
Hyderabad	109.207447	91.931532	0.590780
Chennai	114.502654	62.834404	1.067349
Visakhapatnam	117.269855	105.951354	0.736739
Amritsar	119.920959	115.202599	0.551301
Bhopal	132.827338	119.324321	0.880393
Jaipur	133.679159	123.483509	0.800894
Guwahati	140.111111	116.604900	0.735269
Kolkata	140.566313	115.629499	0.799251
Brajrajnagar	150.280505	124.217373	1.798274
Jorapokhar	159.251621	149.657942	0.663049
Talcher	172.886819	165.772708	1.847722
Lucknow	217.973059	NaN	2.130328
Gurugram	225.123882	191.504432	1.257516
Patna	240.782042	126.747958	1.533591
Delhi	259.487744	232.809229	1.976053
Ahmedabad	452.122939	114.584029	22.193407

Figure 5: (a) Average pollution from 25 major cities of India from Jan 2015 to July 2020 on daily (b) Average levels of pollution in 26 cities

3.3 SMOTEDNN Model Development for AQI Classification

The present investigation developed a novel model SMOTEDNN to classify air pollution. In order to assess the model performance, it was compared with four state-of-the-art ML models-XGBoost, Random Forest, SVM, and KNN to classify the AQI into six classes shown in Fig. 2. The performance in terms of accuracy, stability, and time complexity of ML/DL models depend on optimizing the hyperparameters. In the present investigation, hyperparameters for all the developed models were optimized rigorously to avoid illusory accuracy.

3.3.1 SMOTEDNN

SMOTE was integrated to overcome the issue of class imbalance; it oversampled the values of the minority class based on duplicating the minority class values. These new values did not add new information to the algorithm; however, new values were synthesized from the existing values. SMOTE randomly selects a minor class and creates a new value based on k nearest neighbors randomly. The counter object was used to summarize the number of points in each class to confirm that the dataset was created correctly. SMOTE performed well to overcome class imbalance issues and resultant pollution classes containing a balanced number of occurrences in each class (i.e., 129277).

The novel SMOTEDNN model was developed with five layers each in the present study to classify AQI based on the air pollution data. Python 3.8, Keras 2.3.0 API, Tensorflow 2.0 back-end, NumPy, pandas, os, sklearn, matplotlib, and DateTime libraries were used in this research (Fig. 6). Data pre-processing and SMOTE were applied to the raw dataset so that the output of both steps could be utilized with the developed DNN models. We used five neural network layers that operated on all the six classes of air quality. The rectifier activation function (ReLU) was utilized for starting four neural network layers. The ReLU activation function can be defined as Eq. (9). ReLU acts as a linear function for all positive values and provides zero for all negative values.

$$y = max(0, x) \tag{9}$$

The fifth layer used the kernel initializer followed by the dense layer with softmax function for classification in the sixth layer. The softmax function can be given as Eq. (10).

$$\sigma(\vec{z})_i = \frac{e^{Zi}}{\Sigma_{j=1}^{n} e^{Zj}} \tag{10}$$

where n=no of classes, and $e^{zi,}$ and e^{zj} are input and output vector function, respectively.

Early stopping was used to reduce the learning rate through Keras callbacks function to prevent overfitting. The number of epochs was automatically chosen using the early stopping of Keras callback functions based on validation loss, minimum delta value, and patience. In DNN model training, the number of parameters such as iterations, learning rate, batch size, and the activation function was obtained using GridSearchCV. Deep learning models such as DNN might be complex, and data splitting is also a significant issue while tuning the parameters. There was a total 11,990 number of parameters, and all parameters were trainable using SMOTEDNN.

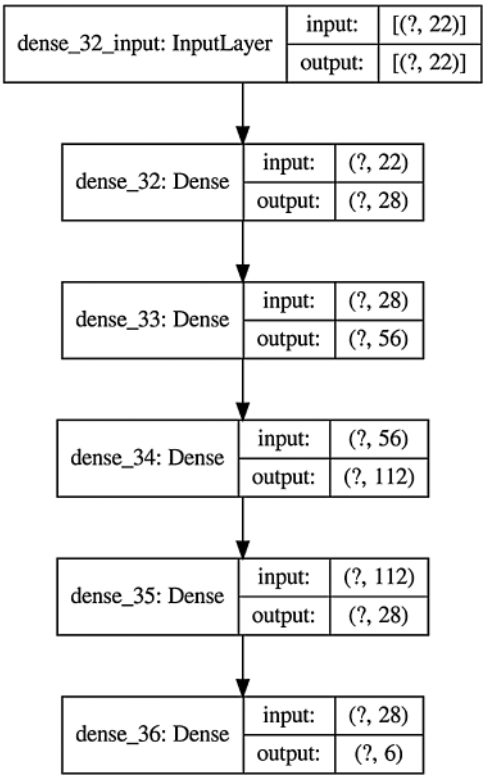


Figure 6: Developed SMOTEDNN model to classify the air quality for six classes

SMOTEDNN model was compiled after defining the model. The Adam optimizer was used with a decay of 1e-3; the learning rate was selected automatically and dynamically, using a call-back monitor. The present problem was multiclass classification; therefore, the loss was measured based on categorical cross-entropy. It was used to compute the error rate between the actual and the m values for classification, as in Eq. (11).

$$\text{Loss} = -\frac{1}{\text{O}_\text{s}} \sum_{\text{i}=0}^{\text{O}_\text{s}} \text{a}_\text{i} \cdot \log \hat{t}_\text{i} + (1 - \text{a}_\text{i}) \cdot \log(1 - \hat{t}_\text{i}) \tag{11}$$

where O_s, a_i and $\hat{t}_i$ are the output size, target, and output values, respectively.

The optimization of hyperparameters for the developed SMOTEDNN model is given in Tab. 1.

3.3.2 XGBoost

The XGBoost model was developed using XGBClassifier class within the 'XGBoost' module in the sklearn (scikit-learn) package in Python. The XgBoost algorithm was applied to the pre-processed data (in Section 3.1) to classify the AQI based on pollutants data. The hyperparameters were tuned based on RandomizedSearchCV with the following parameters (Tab. 2).

Table 1: Hyperparameters optimization for SMOTEDNN

Parameter	Value	Optimized parameter	Remark
Activation function	identity, logistic, tanh, relu	relu	Activation function for the hidden layer
Size of hidden layers	Different values	22, 28, 56, 112, 28	Neurons in the hidden layer
Optimizer	lbfgs, sgd, adam	adam	Used for error optimization
Alpha	0.00001, 0.0001, 0.001, 0.01	0.001	It refers to the regularization parameter
Learning rate	0.0001, 0.001, 0.01	0.001	Step-size controller to update the weights

Table 2: Hyperparameters optimization for XGBoost

Parameter	Value	Optimized parameter	Remarks
Learning rate	0.05, 0.10, 0.15, 0.20, 0.25, 0.30	0.20	Loss minimization for each iteration
max_depth	[3, 4, 5, 6, 8, 10, 12, 14, 16]	4	Maximum depth for a tree
min_child_-weight	[1, 3, 5, 7, 9, 11]	7	Minimum combined weights of all observations for a child node
gamma	[0.0, 0.1, 0.2, 0.3, 0.4, 0.5]	0.0	Refers to the minimum loss minimization need to do a split.
colsample_-bytree	[0.3, 0.4, 0.5 , 0.6, 0.7]	0.5	Column fraction that samples randomly for every tree
n_estimators	10, 20, 50, 100, 200	100	Number of trees

3.3.3 Random Forest

Random Forest Classifier from the sklearn.ensemble module of the sklearn package [25] was used to develop the RF model. In addition, the Gini impurity function was used in the present investigation as it requires less computation [26]. The optimized hyperparameters for RF are given in Tab. 3.

Table 3: Hyperparameters optimization for RF

Parameter	Value	Optimized parameter	Remark
n_estimators	10, 20, 50, 100, 200	100	Number of trees
n_jobs	1, −1	−1	Number of processors −1 means no restrictions, and 1 means only one is allowed
random_state	10, 20, 30, 40, 50, 60, 100	50	For replication of same results with same models
min_samples_-leaf	10, 20, 30, 40, 50, 60, 100	60	End node of the decision tree

3.3.4 SVM

The SVM model was developed using the svm module of the sklearn package [25]. The regularization parameters (C) control the SVM model performance with radial basis function (RBF) kernel. The RBF kernel can be represented as:

$$K(x_i, x_j) = e^{(-\gamma||x_i - x_j||^2)} \tag{12}$$

Here, γ is the kernel width. The optimized hyperparameters for the SVM model used in the present investigation are given in Tab. 4.

Table 4: Hyperparameters optimization for SVM

Parameter	Value	Optimized parameter	Remark
C	10, 20, 50, 100, 200	100	Regularization parameter
kernel	poly, linear, rbf, sigmoid,	rbf	Kernel type used
degree	1, 2, 3, 4, 5	none	Used only in polykernel
gamma	scale, float, auto	auto	Kernel coefficient

3.3.5 KNN

KNN model was developed using KNeighborsClassifier class in sklearn package in Python. One of the most critical parameters for the KNN is neighbors, which determines that the unknown value can be obtained from how many neighbors have known values. The optimized hyperparameters for the KNN model used in the present investigation are given in Tab. 5.

3.4 AQI Forecasting Using Linear Regression

For AQI time series forecasting, the New Delhi city was selected based on high pollutant levels for the observation period. The primary step was to understand which particular pollutant affected New Delhi AQI values the most. The correlation values with AQI for different pollutants and the percentage of null values are given in Tab. 6. It was evident from Tab. 6 that particulate matter (PM 2.5, PM 10), B, NO_2, and NO showed a significantly high correlation with AQI. The combined value of pollutants B_X_O_3_NH_3 showed a higher correlation but more null values due to the non-availability of X pollutant data.

Table 5: Hyperparameters optimization for KNN

Parameter	Value	Optimized parameter	Remark
N_neighbors	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	10	Number of neighbors
n_jobs	1, −1	−1	Number of processors −1 means no restrictions, 1 means only 1 is allowed
weights	uniform, distance	distance	Weight function used in prediction
algorithm	auto, ball_tree, kd_tree, brute	Kd_tree	The algorithm used to compute the NN

Table 6: Correlation of pollutants with AQI and % of null values for New Delhi for observation period

Pollutant/Variable	Correlation value	% of Null values
Particulate matters	0.92	3.50
PM10	0.88	3.50
PM2.5	0.88	0.10
Benzene	0.67	0.00
NO2	0.67	0.10
NO	0.64	0.10
B_X_O3_NH3	0.63	38.60
NOx	0.56	0.00
NH3	0.52	0.40
SO2	0.41	5.10
O3	0.33	3.80
CO	0.28	0.00
Toluene	0.28	0.00
Xylene	0.23	38.60
Month	0.06	0.00
Year	−0.28	0.00

The city, date, Year_Month, and AQI_Bucket columns were not required and were deleted. The frequency distribution of AQI for New Delhi is not a normal distribution. Therefore, seasonality involvement did exist. AD Fuller statistical test was performed to check the time series behavior (i.e., stationary or non-stationary). A value of −3.351 was obtained based on the AD Fuller test with a p-value of 0.01263; it was found based on the test that the dataset was non-stationary. Since the data behavior of time series was non-stationary, the original data could not forecast the value through an autoregressive model (Fig. 7a). However, one-step prediction based on the previous step's modeling was possible if there was autocorrelation in the dataset. Figs. 7b–7d shows a significantly higher correlation for one-step forecasting based on the previous step value. The first time series forecasting model used the previous step-index to generate the next step-index (Fig. 8a).

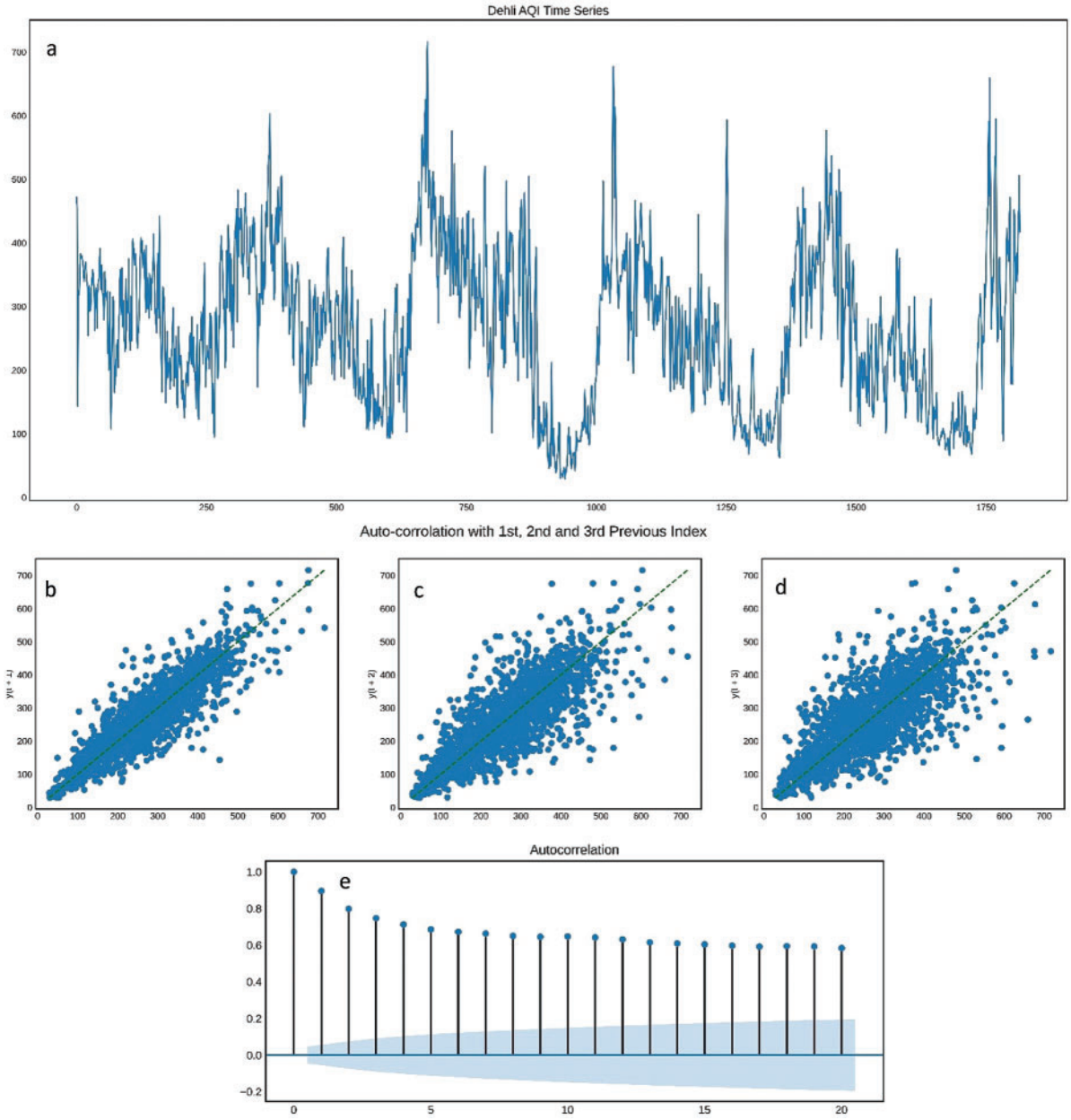


Figure 7: (a) AQI time series for New Delhi, (b) Autocorrelation for first previous index, (c) Autocorrelation for second previous index, (d) Autocorrelation for third previous index, (e) Autocorrelation for the time series

4 Results and Discussions

4.1 Accuracy Assessment

We evaluated the performance of all the developed models based on accuracy, error, sensitivity, specificity, false-positive rate, and false-negative rate. These metrics are defined as follows:

Accuracy of a method on a test dataset is the percentage used to correctly identify the test occurrences, and it was computed as Eq. (12). The error rate was obtained using Eq. (13). The uncertainty of the 'sensitivity' and 'specificity' was used to obtain the model's strength and stability Eqs. (14)–(15). False-positive ratio (FPR) and false-negative ratio (FNR) were obtained using Eqs. (16)–(17).

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{FP} + \text{TN} + \text{FN}) \tag{13}$$

$$\text{Error} = 1 - \text{Accuracy} \tag{14}$$

$$\text{Sensivitiy} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100 \tag{15}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \times 100 \tag{16}$$

$$\text{FPR} = \frac{\text{FP}}{\text{TP} + \text{FP}} \tag{17}$$

$$\text{FNR} = \frac{\text{FN}}{\text{TN} + \text{FN}} \times 100 \tag{18}$$

TP, TN, FP, and FN represent true positive, true negative, false positive, and false negatives, respectively.

4.2 Air Quality Classification Models

In this section, the results of air quality classification models are given. The performance of the developed SMOTEDNN model was assessed, with the unforeseen data kept separately during the training process for proper assessment and evaluation of the developed models. An attempt was made to see if the SMOTEDNN model was overfitted. It was evident that the variance between validation loss and training loss was almost negligible; therefore, overfitting did not exist. As mentioned earlier, the hyperparameters of SMOTEDNN were tuned using GridsearchCV, and callbacks were utilized to select the optimal number of epochs to prevent overfitting automatically. The SMOTEDNN model-optimized results were obtained using 17 epochs, which consumed fewer computing resources and lesser time (Fig. 8). The present investigation utilized Tensor Processing Units (TPUs) v 2–8. These TPUs are Google's application-specific circuits, which accelerate the training workflows of AI models. There were eight cores and 64GiB memory in the TPU v2-8 used in the present investigation. SMOTEDNN took 34 s and 17 ms with 17 epochs on an average. It was evident from Fig. 8 that the accuracy and loss were optimized for selecting the number of epochs through callbacks.

Figure 8: The performance of the SMOTEDNN model based on accuracy and loss

Based on the performance metrics given in Tab. 7, it was observed that SMOTEDNN outperformed all the other models, though the other models did fairly well. The reason behind better performance of Model 3 was the efficient data pre-processing and rigorous tuning of the hyperparameters.

Table 7: Confusion matrix for all five developed Models to classify air quality index

SMOTEDNN	Good	Satisfactory	Moderate	Poor	Very poor	Severe
Good	3365	0	0	0	0	0
Satisfactory	0	16845	0	6	0	0
Moderate	0	0	4816	0	0	0
Poor	0	0	0	13408	0	0
Very poor	0	0	1	0	997	1
Severe	0	1	0	0	2	3220
XGBoost	**Good**	**Satisfactory**	**Moderate**	**Poor**	**Very poor**	**Severe**
Good	3360	0	5	0	0	0
Satisfactory	0	16845	0	6	0	0
Moderate	0	2	4814	0	0	0
Poor	0	0	0	13408	0	0
Very Poor	0	0	1	0	984	14
Severe	0	0	20	0	4	3199
RF	**Good**	**Satisfactory**	**Moderate**	**Poor**	**Very poor**	**Severe**
Good	3285	0	3	77	0	0
Satisfactory	0	16818	25	8	0	0
Moderate	0	26	4717	0	0	73
Poor	55	113	0	13240	0	0
Very poor	0	2	0	0	974	23
Severe	0	0	7	0	12	3204
SVM	**Good**	**Satisfactory**	**Moderate**	**Poor**	**Very poor**	**Severe**
Good	3288	0	0	77	0	0
Satisfactory	0	16818	25	8	0	0
Moderate	0	26	4717	0	0	73
Poor	55	113	0	13240	0	0
Very poor	0	0	0	0	976	23

(Continued)

Table 7: Continued

SVM	Good	Satisfactory	Moderate	Poor	Very poor	Severe
Severe	0	0	7	0	12	3204
KNN	**Good**	**Satisfactory**	**Moderate**	**Poor**	**Very poor**	**Severe**
Good	3195	0	0	170	0	0
Satisfactory	1	16568	81	201	0	0
Moderate	0	211	4504	0	0	101
Poor	245	778	0	12385	0	0
Very poor	0	0	0	0	946	53
Severe	0	0	147	0	58	3018
Models	**SMOTEDNN**	**XGBoost**	**RF**	**SVM**	**KNN**	
Accuracy	99.9	99.2	99.1	99.01	95.2	
Sensitivity	98.76	96.54	95.42	95.23	91.87	
Specificity	99.13	97.65	96.74	96.58	93.46	
Error rate	0.1	0.8	0.9	0.99	4.8	
FDR	2.17	1.97	2.73	1.96	6.49	
FOR	0.08	0.16	0.47	0.73	5.82	

4.3 Air Quality Forecasting Models

Three air quality forecasting models were developed in the current investigation. The first linear model used one previous step as input to forecast the output of the following step (Fig. 9a). The second linear model used seven previous steps to forecast the following step's output (Fig. 9b). The third autoregressive model (Model 3) used tuning of the optimized number of steps required to forecast the air quality for New Delhi (Fig. 9c).

Model 1 showed a correlation value of 0.9 when comparing the forecasted value with the actual value with a root mean square error (RMSE) of 29.17 (Fig. 9a). Model 1 can be used for forecasting based on the high correlation value (Tab. 8). Model 2 was developed to forecast a one-step model based on n previous steps; initially, we chose a value of k was one week (Fig. 9b). The RMSE value for Model 2 was 53.21, larger than Model 1 (Tab. 9). Model 3 was based on autoregression. However, as already mentioned, the time series pattern was non-stationary and unsuitable for autoregression. Therefore, the trend and seasonality from the time series were removed based on the approach given by [27]. The AR package was used from the StatsModel library using Python to develop Model 3. The tuning of n, trend parameter, and the seasonal parameter was crucial to obtain the optimized forecasting model. The k values ranging from 1 to 365 days were given to model using a loop to obtain the optimized autoregression model. The optimized value for k was 14 steps or previous days value (Fig. 9c). The trend parameter and seasonal parameter were n and True, respectively. The RMSE for Model 3 was 15.48 with an R-value of 0.93, which was significantly better than Models 1 and 2, it was also indicated through Fig. 9c that the third developed model shown better fitting between actual and forecasted AQI values.

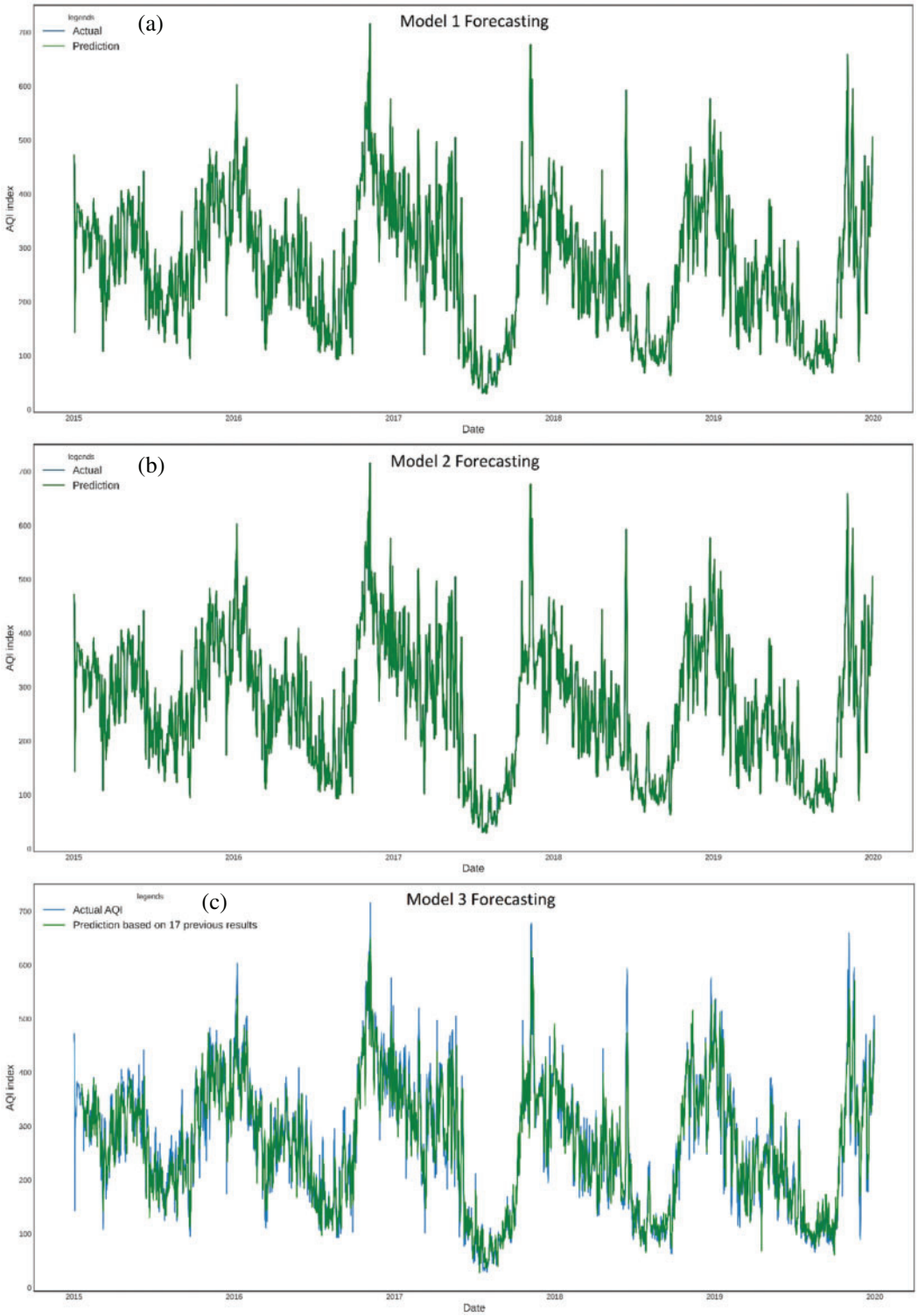


Figure 9: (a) AQI forecasting based on Model 1, (b) AQI forecasting based on Model 2, (c) AQI forecasting based on Model 3

Table 8: Forecasting performance of Model 1 with k = 1 step

CM	actual	pred	r
actual	1	0.8981	0.9
pred	0.8989	1	

Table 9: Forecasting performance of model 2 with k = 7 steps

CM	actual	pred	r
1	384	372.427	0.4
2	340	353.787	
3	372	346.997	
4	425	343.519	
5	455	338.261	
6	506	332.855	
7	417	328.223	

4.4 Comparison with Other Studies

The present investigation developed five models to classify air pollution severity based on different pollutants. The models developed in the present investigation were compared with other studies to assess the performance of the developed models (see Tab. 10). Overall, compared to the other models, the SMOTEDNN model produced the highest classification accuracy. Similarly, the autoregression-based Model 3 for forecasting yielded higher accuracy compared to other studies (Tab. 10).

Table 10: Comparison with other studies

AQI Classification models performance comparison			
Author	Method	Accuracy %	Remarks
[7]	Recurrent neural networks	81.00	Used for O_3 pollutant
[8]	Recurrent neural networks	95.00	Only for PM2.5 and PM10
[9]	SVR+wavelet, Ensemble	96.00	Only for PM10
[10]	LR	93.00	Lack of hyperparameters tuning
[10]	RF	86.00	Lack of hyperparameters tuning

(Continued)

Table 10: Continued

AQI Classification models performance comparison			
Author	Method	Accuracy %	Remarks
[28]	LR	68.74	Lack of hyperparameters tuning
[28]	SGD	66.23	Lack of hyperparameters tuning
[28]	RFR	72.22	Lack of hyperparameters tuning
[28]	DTR	71.08	Lack of hyperparameters tuning
[28]	MLP	70.43	Lack of hyperparameters tuning
[28]	SVR	70.97	Lack of hyperparameters tuning
[28]	GBR	74.91	Lack of hyperparameters tuning
[28]	ABR	49.63	Lack of hyperparameters tuning
Current study	SMOTEDNN	99.90	Novel model with rigorous hyperparameters tuning
Current study	XGBoost	99.20	Rigorous hyperparameters tuning
Current study	RF	99.10	Rigorous hyperparameters tuning
Current study	SVM	99.01	Rigorous hyperparameters tuning
Current study	KNN	95.20	Hyperparameters tuning
AQI Forecasting Models Performance Comparison			
[9]	ANN	0.91	Forecasting for PM10, R-value, lack of parameters tuning

(Continued)

Table 10: Continued

AQI Classification models performance comparison			
Author	Method	Accuracy %	Remarks
[10]	SARIMA	20.69	Forecasting of AQI for 2019, RMSE
[10]	SARIMA	43.95	Forecasting of AQI for 2020, high RMSE
[10]	Facebook-Prophet	22.81	Forecasting of AQI for 2019, RMSE
[11]	ANN	0.88	O_3 peak forecasting, lack of parameters tuning
[12]	ANN	0.89	Air pollution forecasting, lack of parameters tuning
[13]	Hybrid ANN	0.70–0.83	Forecasting of PM10, CO, NO, NO_2, lack of parameters tuning
[14]	PCA+ANN	0.27–0.75	Weak data splitting and lack of parameters tuning
[15]	MLP+ANN	0.78–0.82	Detailed investigation
[16,17]	Decision tree and Naive based	0.91	Time series forecasting
[18]	SVR+RBF	−0.67	Used one station data
[18]	SVR+Linear	−0.61	Used one station data
[18]	SVR+Poly	−0.79	Used one station data
[18]	RNN+LSTM	0.48	Used one station data
[29]	Urban Airshed Model	0.69	Forecasting for O_3, R-value
[30]	RNN	0.35–0.36	Only forecast PM2.5
[31]	StackedLSTM	10.65–21.44	RMSE for 12 h forecasting for CO, O_3, NO_2, SO_2, and PM, used data only from 2 sensor locations
[32]	GRU+SGD	0.82	PM 2.5 forecasting with GRU, SGD and RNN
[33]	CNN+LSTM	0.43	Weak parameter tuning
[34]	LSTM+PSO	18	MAPE
Current study	Linear regression model 1	0.90	Pre-processing and statistical operations were performed with one step-index forecasting
Current study	Linear regression model 2	0.40	Trial and error model for seven step-index forecasts, low accuracy
Current study	Auto regression model 3	0.93	Tuned number of k for autoregression, higher accuracy

5 Conclusions

The increasing rate of industrialization and urbanization is the main reason for worsening air pollution status, especially for developing nations. Previous studies that used ML and statistical modeling to classify and forecast the air pollution suffered heavily due to the dataset's nature and complexity, resulting in a lack of efficient classification and forecasting of air pollution. Especially, ML-based models have shown improper data handling, class imbalance issues, data division for training and testing, and, most importantly, inaccurate hyperparameter tuning. The current investigation contributed toward bridging the identified gaps for both aspects of air pollution analysis, i.e., classification and forecasting. Five ML models were developed, including one novel model named SMOTEDNN to address the air pollution classification. All five models utilize efficient data pre-processing and rigorous hyperparameter optimization. All the developed models showed excellent performance based on accuracy, precision, sensitivity, and specificity. Significantly, the novel model SMOTEDNN showed higher accuracy (99.90%) than the other models from the current investigation and previous studies. The primary reason for this exceptional performance was rigorous data pre-processing and intense hyperparameter tuning. The performance of the two forecasting models (Model 1 and Model 3) was good; however, Model 2 was not efficient enough. The study indicated that air pollution in India, during Jan 2015 to Jul 2020, showed severity of pollution trends with two weeks of index data as a baseline. The future scope of present investigation to include more datasets through IoT based pollution dataset to real time air quality assessment and forecasting.

Acknowledgement: The author is thankful to CPCB for air pollution data.

Funding Statement: Mohd Anul Haq would like to thank Deanship of Scientific Research at Majmaah University for supporting this work under Project No. R-2021-202.

Conflicts of Interest: The author declare that they have no conflicts of interest to report regarding the present study.

References

[1] P. M. Mannucci and M. Franchini, "Health effects of ambient air pollution in developing countries," *International Journal of Environmental Research and Public Health,* vol. 14, no. 9, pp. 1–8, 2017.

[2] D. Fang, B. Chen, K. Hubacek, R. Ni, L. Chen *et al.*, "Clean air for some: Unintended spillover effects of regional air pollution policies," *Science Advances,* vol. 5, no. 4, pp. 4707–4731, 2019.

[3] D. A. Glencross, T. R. Ho, N. Camiña, C. M. Hawrylowicz and P. E. Pfeffer, "Air pollution and its effects on the immune system," *Free Radical Biology and Medicine,* vol. 151, pp. 56–68, 2020.

[4] J. Miao, X. Zhou and T. Z. Huang, "Local segmentation of images using an improved fuzzy C-means clustering algorithm based on self-adaptive dictionary learning," *Applied Soft Computing Journal,* vol. 91, pp. 1–15, 2020.

[5] Z. Ghaemi, A. Alimohammadi and M. Farnaghi, "LaSVM-based big data learning system for dynamic prediction of air pollution in Tehran," *Environmental Monitoring and Assessment,* vol. 190, no. 5, pp. 1–17, 2018.

[6] S. Chen, K. Mihara and J. Wen, "Time series prediction of CO2, TVOC and HCHO based on machine learning at different sampling points," *Building and Environment,* vol. 146, pp. 238–246, 2018.

[7] M. A. Esfandani and H. Nematzadeh, "Predicting air pollution in Tehran: Genetic algorithm and back propagation neural network," *Journal of Artificial Intelligence and Data Mining,* vol. 4, no. 1, pp. 49–54, 2016.

[8] F. Biancofiore, M. Busilacchio, M. Verdecchia, B. Tomassetti, E. Aruffo *et al.*, "Recursive neural network model for analysis and forecast of PM10 and PM2.5," *Atmospheric Pollution Research,* vol. 8, no. 4, pp. 652–659, 2017.

[9] K. Siwek and S. Osowski, "Improving the accuracy of prediction of PM 10 pollution by the wavelet transformation and an ensemble of neural predictors," *Engineering Applications of Artificial Intelligence,* vol. 25, no. 6, pp. 1246–1258, 2012.

[10] R. Mangayarkarasi, C. Vanmathi, M. Z. Khan, A. Noorwali, R. Jain *et al.*, "Covid19: Forecasting air quality index and particulate matter (pm2.5)," *Computers, Materials and Continua,* vol. 67, no. 3, pp. 3363–3380, 2021.

[11] A. L. Dutot, J. Rynkiewicz, F. E. Steiner and J. Rude, "A 24-h forecast of ozone peaks and exceedance levels using neural classifiers and weather predictions," *Environmental Modelling and Software,* vol. 22, no. 9, pp. 1261–1269, 2007.

[12] N. H. A. Rahman, M. H. Lee, M. T. Latif and S. Suhartono, "Forecasting of air pollution index with artificial neural network," *Jurnal Teknologi (Sciences and Engineering),* vol. 63, no. 2, pp. 59–64, 2013.

[13] A. Russo and A. O. Soares, "Hybrid model for urban air pollution forecasting: A stochastic spatio-temporal approach," *Mathematical Geosciences,* vol. 46, no. 1, pp. 75–93, 2014.

[14] A. Azid, H. Juahir, M. T. Latif, S. M. Zain and M. R. Osman, "Feed-forward artificial neural network model for air pollutant index prediction in the southern region of peninsular Malaysia," *Journal of Environmental Protection,* vol. 04, no. 12, pp. 1–10, 2013.

[15] L. Bai, J. Wang, X. Ma and H. Lu, "Air pollution forecasts: An overview," *International Journal of Environmental Research and Public Health,* vol. 15, no. 4, pp. 1–44, 2018.

[16] A. J. Cohen, M. Brauer, R. Burnett, H. R. Anderson, J. Frostad *et al.*, "Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: An analysis of data from the global burden of diseases study 2015," *Lancet,* vol. 389, no. 10082, pp. 1907–1918, 2017.

[17] R. W. Gore and D. S. Deshpande, "An approach for classification of health risks based on air quality levels," in *Proc.-1st Int. Conf. on Intelligent Systems and Information Management, ICISIM*, 2017, Aurangabad, India, vol. 2017-Jan, pp. 58–61, 2017.

[18] K. S. Rao, G. L. Devi and N. Ramesh, "Air quality prediction in visakhapatnam with lstm based recurrent neural networks," *International Journal of Intelligent Systems and Applications,* vol. 11, no. 2, pp. 18–24, 2019.

[19] S. Ameer, M. A. Shah, A. Khan, H. Song, C. Maple *et al.*, "Comparative analysis of machine learning techniques for predicting air quality in smart cities," *IEEE Access,* vol. 7, pp. 128325–128338, 2019.

[20] K. Gu, Y. Zhou, H. Sun, L. Zhao and S. Liu, "Prediction of air quality in shenzhen based on neural network algorithm," *Neural Computing and Applications,* vol. 32, no. 7, pp. 1879–1892, 2020.

[21] D. Sahoo, S. C. H. Hoi and B. Li, "Large scale online multiple kernel regression with application to time-series prediction," *ACM Transactions on Knowledge Discovery from Data,* vol. 13, no. 1, pp. 33, 2019.

[22] V. N. Vapnik, "*The Nature of Statistical Learning Theory,*" New York, USA: Springer. 1995.

[23] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks,* vol. 10, no. 5, pp. 988–999, 1999.

[24] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning,* vol. 20, pp. 273–297, 1995.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research,* vol. 12, pp. 2825–2830, 2011.

[26] L. E. Raileanu and K. Stoffel, "Theoretical comparison between the gini index and information gain criteria," *Annals of Mathematics and Artificial Intelligence,* vol. 41, no. 1, pp. 77–93, 2004.

[27] M. Sultan, N. C. Sturchio, S. Alsefry, M. K. Emil, M. Mohamed *et al.*, "Assessment of age, origin, and sustainability of fossil aquifers: A geochemical and remote sensing-based approach," *Journal of Hydrology,* vol. 576, no. May, pp. 325–341, 2019.

[28] C. Srivastava, S. Singh and A. P. Singh, "Estimation of air pollution in Delhi using machine learning techniques," in *2018 Int. Conf. on Computing, Power and Communication Technologies, GUCON 2018*, no. 2018, pp. 304–309, 2019.

[29] M. E. Chang and C. Cardelino, "Application of the urban airshed model to forecasting next-day peak ozone concentrations in Atlanta," *Journal of the Air & Waste Management Association,* vol. 50, no. 11, pp. 2010–2024, 2000.

[30] B. Liu, S. Yan, J. Li, Y. Li, J. Lang and G. Qu, "A spatiotemporal recurrent neural network for prediction of atmospheric PM2.5: A case study of Beijing," *IEEE Transactions on Computational Social Systems,* vol. 8, no. 3, pp. 578–588, 2021.

[31] V. Chaudhary, A. Deshbhratar, V. Kumar and D. Paul, "Time series based lstm model to predict air pollutant's concentration for prominent cities in India," in *Proc: Udm'18*, London, pp. 1–9, 2018.

[32] C. J. Masinde, J. Gitahi and M. Hahn, "Training recurrent neural networks for particulate matter concentration prediction," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives,* vol. 43, no. b2, pp. 1575–1582, 2020.

[33] K. Tripathi and P. Pathak, "Deep learning techniques for air pollution," in *Proc. Int. Conf. on Computing, Communication, and Intelligent Systems (ICCCIS)*, no. 2021, pp. 1013–1020, 2021.

[34] A. Heydari, N. M. Majidi, D. A. Garcia, F. Keynia and L. D. Santoli, "Air pollution forecasting application based on deep learning model and optimization algorithm," *Clean Technology and Environment Policy,* pp. 1–15, 2021.