Hindawi
Journal of Engineering
Volume 2024, Article ID 2893021, 16 pages
https://doi.org/10.1155/2024/2893021

Research Article

A Machine Learning Approach for Environmental Assessment on Air Quality and Mitigation Strategy

Chetan Shetty,[1] S. Seema,[1] B. J. Sowmya,[2] Rajesh Nandalike,[3] S. Supreeth,[4] Dayananda P.,[5] Rohith S.,[6] Vishwanath Y.,[7] Rajeev Ranjan,[8] and Venugopal Goud[9]

[1]*Department of Computer Science and Engineering, Ramaiah Institute of Technology, Bengaluru 560054, India*
[2]*Department of Artificial Intelligence and Data Science, Ramaiah Institute of Technology, Bengaluru 560054, India*
[3]*Department of Electronics & Communication Engineering, Nitte Meenakshi Institute of Technology, Bengaluru 560064, India*
[4]*School of Computer Science and Engineering, REVA University, Bengaluru 560064, India*
[5]*Department of Information Technology, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Manipal 576104, India*
[6]*Department of Electronics & Communication Engineering, Nagarjuna College of Engineering & Technology, Bengaluru 562110, India*
[7]*Presidency University, School of Computer Science & Engineering, Bengaluru 560064, India*
[8]*Government Engineering College, West Champaran 845450, India*
[9]*G. Pulla Reddy Engineering College, Kurnool 518007, India*

Correspondence should be addressed to Dayananda P.; dayananda.p@manipal.edu

Received 4 December 2023; Revised 9 January 2024; Accepted 8 February 2024; Published 26 February 2024

Academic Editor: Mehmet Şükrü Adin

Air pollution has a significant impact on environment resulting in consequences such as global warming and acid rain. Toxic emissions from vehicles are one of the primary sources of pollution. Assessment of air pollution data is critical in order to assist residents in locating the safest areas in the city that are ideal for life. In this work, density-based spatial clustering of applications with noise (DBSCAN) is used which is among the widely used clustering algorithms in machine learning. It is not only capable of finding clusters of various sizes and shapes but can also detect outliers. DBSCAN takes in two important input parameters—Epsilon (Eps) and Minimum Points (MinPts). Even the slightest of variations in the parameter values fed to DBSCAN makes a big difference in the clustering. There is a need to find Eps value in as minimum time as possible. In this work, the goal is to find the Eps value in less time. For this purpose, a search tree technique is used for finding the Eps input to the DBSCAN algorithm. Predicting air pollution is a complex task due to various challenges associated with the dynamic and multifaceted nature of the atmosphere such as meteorological variability, local emissions and sources, data quality and availability, and emerging pollutants. Extensive experiments prove that the search tree approach to find Eps is quicker and efficient in comparison to the widely used KNN algorithm. The time reduction to find Eps makes a significant impact as the dataset size increases. The input parameters are fed to DBSCAN algorithm to obtain clustering results.

1. Introduction

In this section, an introduction to air pollution is given. The release of toxins into the air which are detrimental to humans and the atmosphere is called air pollution. If toxic or unsustainable concentrations of chemicals, such as gases, particulates, and biological molecules, are released into the environment, this leads to air pollution. It has the potential to cause infections, illnesses, and even death of humans; it also has the potential to affect other living species such as animals and food crops, as well as degrade the natural or constructed environment. Air pollution can be caused by both manmade and natural factors. Air pollution is not a constant. Various compounds react with

one another, resulting in ever-changing compositions of compounds.

An air pollutant is a substance in the atmosphere that can cause damage to humans and the environment in [1]. This can include solid substance, water droplets, or gases as the constituents. Pollutants or contaminants may be either naturally occurring or man-made. Pollutants are divided into two categories: primary and secondary. Primary contaminants are normally formed by natural processes, such as volcano activity. Carbon monoxide gas emitted by automobiles or sulphur dioxide emitted by factories are two such examples. Secondary contaminants are not explicitly emitted. However, they form in the air as a result of the reaction or interaction of primary contaminants. Ozone at the ground level is a good indicator of a secondary contaminant. Some contaminants are both primary and secondary in nature, meaning they are released directly and produced by other primary contaminants.

Some of the air pollutants [2] are described below.

Smog: The following two forms of pollution are very common—Smog and Soot. Smog, also known as "ground-level ozone," is produced when contaminants from fossil fuels combine with sunlight. Soot, also called as "particulate matter," is made up of small fragments of chemicals, dirt, ash, dust, or allergens which are borne in the air as gas or solids.

Sulphur dioxide (SO_2): The largest pollutant in the air is sulphur dioxide. SO_2 is primarily generated by burning sulphur-bearing stone fuels, and coal and petroleum usually contain between 1% and 5% sulphur.

Nitrogen oxide (NO_x): Among oil, natural gas, and coal, coal comprises of a higher amount of nitrogen. At high temperatures, nitrogen and oxygen in the atmosphere react even more, resulting in significant emissions from transportation and power generation.

Carbon particles and noncarbon particles: The utilization of biomass energy from fossil fuels, such as diesel and gasoline generators, is the predominant contributor to carbon particles in the form of carbon elements and low volatile carbon compounds. Moving dust is a significant contributor of carbon-free particles. It is mostly emitted from fossil fuels.

Carbon monoxide (CO): CO is a part of vehicle exhaust that is emitted primarily as a result of insufficient combustion.

Volatile organic compounds (VOC): In the atmosphere and water vapour, VOCs are made up of hydrocarbons, halogenates, and oxygen-containing compounds. Some sources comprise pressurised system leakage including natural gas and nitrogen and liquid fuel volatilization such as fuel tanks.

Ozone: Ozone, an active oxidant is a chief constituent of photochemical smog, is produced when nitrogen oxides and sunlight react, which is a photochemical reaction. It is formed in the atmosphere by gases released by tailpipes, smokestacks, and a variety of several other influences.

Particulate matter: Particles suspending in the atmosphere are called aerosol particles or particulate matter (PM). It floats in the air and stays there for weeks or even months, depending on their size. Dust is a very big, type of particulate matter, being 10 micron or greater and approximately of thickness of a hair.

PM2.5: Particles less than a diameter of 2.5 microns are called PM2.5. When such tiny particles are inhaled, they are carried into the lungs, causing health concerns.

PM10: The particles of diameter 10 microns or smaller are called PM10. Dust and pollen form PM10 which are either solid or liquid course particles suspended in air.

Some of the other factors that affect air pollution are wind speed (WS), wind direction (WD), air temperature (AT), and sunlight [3].

Wind speed: Pollutants are likely to add up when the wind speed is not greater than 10 kmph in calm situations. 15 kmph wind speeds or higher favour a pollution dispersion, which would clear the air. The greater the speed of wind, the greater the pollutants dispersion and the lower their quantities in air. But strong winds can also produce dust—a concern in dry windy regions.

Wind direction: When the wind blows from an industrial region into a metropolitan area, the amount of pollution in the urban area would be higher than when wind blows from other direction, like open land. For the pollutants in the valleys, the wind affects the directions of pollutant movement that increases the levels of downwind pollutants over upwind.

Air temperature: Air temperature influences air circulation and hence air pollution migration. Since the Sun absorbs energy from the surface of the earth, the air closer to the ground is warmer than the air in the troposphere. On the surface, the warmer, lighter air rises; and in the upper troposphere, the cooler and heavier air sink.

Sunlight: There is an effect of sunlight on air pollution. The Sun is a significant source of energy. The electrons represent everything when it refers to chemical activity and transition. In the sun's presence, the electrons in a chemical substance are neutral, whereas when they are subjected to the appropriate wavelength of radiation, the compound absorbs it. This excites the electrons, which opens up a whole new universe of possibilities for how the chemical can react. Photochemistry refers to how light and its ability to excite electrons play a part in allowing a chemical transformation to occur. As a result, the energy is quantized and contained in the electrons going stepwise.

Figure 1 shows the percentage of air pollutants. Chemical reactions involving primary air contaminants such as nitrogen oxides which are produced by power plants and oxygen are triggered by sunlight and elevated temperatures, resulting in the formation of ozone. More ozone is formed when the sun is brighter and the day is hotter. Primary particles are often transformed by heat and sunlight into

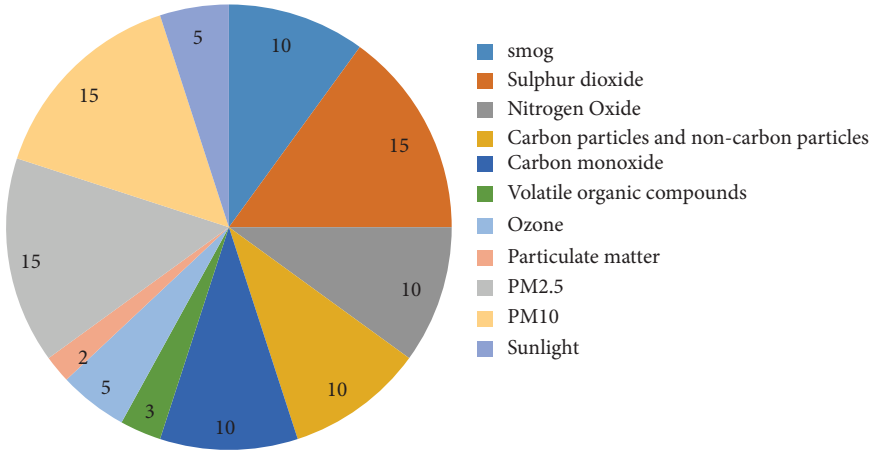

Figure 1: The percentage of air pollutants.

secondary, smaller particles which are potentially more harmful.

Air pollution remains at the ground level due to high ambient pressure, causing the level to rise. Since high pressure causes a humid atmosphere, heat waves and poor quality of air always go together. Contaminants do not get filtered from the air even when there are mild winds and no moisture, but they accumulate just above ground level.

Predicting air pollution using machine learning techniques aligns with various objectives and motivations, leveraging the capabilities of advanced computational models. Here are some key objectives and motivations for employing machine learning in predicting air pollution:

1.1. Enhanced Accuracy and Precision. Machine learning models can capture complex patterns and nonlinear relationships within large datasets. Using these techniques can improve the accuracy and precision of air pollution predictions compared to traditional modeling approaches, leading to more reliable information for decision-making.

(i) Real-time and short-term prediction:

Machine learning enables the development of models that can provide real-time or short-term predictions of air pollution levels. This capability is valuable for implementing timely interventions, issuing warnings, and responding to rapidly changing atmospheric conditions.

(ii) Integration of diverse data sources:

Machine learning allows the integration of diverse data sources, including meteorological data, satellite imagery, and sensor networks. By considering a wide range of variables, machine learning models can provide a more comprehensive understanding of the factors influencing air quality.

(iii) Adaptability to dynamic environments:

Air quality is influenced by dynamic and evolving environmental factors. Machine learning models can adapt to changing conditions and learn from new data, making them well-suited for handling the dynamic nature of atmospheric processes and pollution sources.

(iv) Optimization of monitoring resources:

Machine learning can help optimize the deployment of monitoring resources. By identifying key variables and influential factors, machine learning models can guide the strategic placement of monitoring stations, ensuring efficient data collection for accurate predictions.

(v) Identification of novel patterns and trends:

Machine learning techniques excel at identifying complex patterns and trends within datasets. This capability is particularly valuable for uncovering hidden relationships and novel insights into the sources and dynamics of air pollution, contributing to a deeper understanding of the problem.

(vi) Personalized health risk assessment:

Machine learning models can be used to assess personalized health risks based on individual exposure patterns and susceptibility factors. This information can be valuable for public health interventions and personalized recommendations for vulnerable populations.

(vii) Optimization of emission reduction strategies:

Machine learning can assist in optimizing strategies for reducing emissions by identifying the most effective interventions and their potential impact on air quality. This can aid policymakers and industries in making informed decisions to mitigate pollution.

(viii) Continuous model improvement:

Machine learning models can be continuously updated and improved as new data become available. This adaptive nature allows for ongoing refinement, ensuring that prediction models stay relevant and accurate over time.

(ix) Public engagement and awareness:

Machine learning-based air quality predictions can be communicated in a more accessible and engaging manner to the public. Visualizations and user-friendly interfaces can enhance public awareness and understanding of air quality issues.

In summary, the use of machine learning techniques for predicting air pollution aims to improve accuracy, enable real-time predictions, leverage diverse data sources, adapt to dynamic conditions, and provide valuable insights for decision-makers, ultimately contributing to better air quality management and public health outcomes. The paper is organized into Abstract, Introduction, Literature Survey Design and Implementation, Results and Inferences, and References.

2. Literature Survey

Some of the research works to find the input parameters for DBSCAN algorithm are described below.

A DBSCAN—*K* Nearest Neighbor—Genetic Algorithm (DBSCAN-KNN-GA) is proposed by Mu et al. for performing DBSCAN on different density-level datasets to find the Eps range and MinPts automatically in [4]. Iris, Aggregation, t5.8k and TLC taxi trip record datasets are used for comparison of the put forth approach with the original DBSCAN for evaluating the performance. Instead of the original concept of the *k*-dist graph which considers the distance from a point to all its *k*-nearest neighbors, this approach calculates the mean of the distance from one point to all its *k*-nearest neighbors. The mean gives a smooth curve thereby removing noise, which makes it easy to find the threshold of the density levels. While the actual DBSCAN makes use of global Eps and MinPts which makes the result of clustering the dataset with varying density inaccurate, this paper introduces the use of KNN algorithm for determining a range of Eps values. On this the genetic algorithm is applied to find the most accurate Eps and MinPts which are further fed to the DBSCAN algorithm as input parameters. Experimental results show that this algorithm has a better accuracy in automatic detection of parameters in datasets having varying densities. For huge datasets, this algorithm will be expensive to calculate the distance matrix and lacks efficiency in such a case. The future research of this paper is to use a parallel or distributed computing for reducing the execution time of the algorithm and increasing the efficiency.

An algorithm to find Eps automatically is proposed by Akbari and Unland, thereby eliminating human interaction in [5]. Comparison between the original DBSCAN and the proposed algorithm is done on normal distribution artificial datasets. The method uses the 3 Sigma rule (an empirical or statistical technique) using the concept of standard deviation for detecting the outliers and finding the Eps value. MinPts is chosen as 4 based on history for 2 dimensional dataset. The *k*-dist values are calculated for each point and sorted in ascending order. The border points might cause a negative effect for detecting the Eps value, hence the *k*-dist value of border points is replaced by the *k*-dist value of its nearest core point's *k*-dist value. Mean and standard deviation (SD) of the newly calculated *k*-dist values are obtained. The Eps value is calculated using Mean + 3 ∗ SD formula. The time complexity of the algorithm is $O(n2)$ to calculate the *k*-dist value. $O(n)$ time is needed to calculate the mean and SD. An Eps value is found based on outlier detection for normal distribution dataset. The points lying beyond the 3 sigma value is considered an outlier or noise. The negative effect of border points is removed by replacing the *k*-dist value of the border point with its nearest core point's *k*-dist value. There is no need to find knee which is a challenging task. Nested clusters can also be detected which the normal DBSCAN cannot detect. This study focus primarily on using an empirical rule for outlier identification in normally distributed data. Their future work aims at using Chebyshev's inequality for nonnormal distribution datasets.

Starczewski et al. put forth a new method for determining the Eps and MinPts for DBSCAN for different kinds of clusters in [6]. Artificial datasets are used for the evaluation having different clusters and different dataset shapes and size of two dimensions and three dimensions. The basis of the approach is to find sharp distance increases obtained using *k*-dist function that calculates distances between every element in the dataset with its *k*-th nearest neighbor. Here, the steepest increase is used for selecting a distance that can determine the most appropriate value for Eps. Accurate determining of the knee in the sorted distances is very essential to choose Eps. MinPts is selected empirically according to different datasets. Based on several experiments, the formulae proposed to find the input parameters have good efficiency. The number noise points or outliers after clustering are small in the proposed approach.

An algorithm to determine multiple Eps based on different density levels in the dataset is proposed in [7]. The evaluation are done using various datasets with same and different density levels dataset on different *k* and an appropriate *k* value is found based on experiments. A comparison between automatic generation of Eps for DBSCAN (AGED) and original DBSCAN algorithm is done in terms of accuracy, average silhouette width, Dunn Index and Pearson gamma. First, the algorithm finds the influence function (Euclidean distance) for each point to its *k*-nearest neighbors. The local density which is the sum of influences of a point to its KNN for each point is found and average is taken. A min-max normalization is performed on the average and it is put into respective bins having similar average local density. If a bin contains elements less than *k*, it is considered as an outlier. Otherwise, the average value of each bin having similar local densities is found and a reverse min-max normalization is applied. The value in each bin gives all the possible Eps values. MinPts is chosen using trial and error method. The algorithm works well on varied and nonvaried density datasets to calculate multiple Eps values.

Esmaelnejad et al. proposed a new algorithm for determining a suitable Eps for DBSCAN clustering method. In [8], Eps is replaced with a new parameter named ρ (Noise ratio of the data set). The evaluation is done on three datasets obtained from a dataset named Chameleon. This approach does not minimize the number of parameters, but it is easier than the method to set the Eps since in certain cases, the

programmer priorly knows the data set's noise ratio. This method introduces a heuristic chart to find Eps. The x axis is the number of nodes and y axis is a function given for the dataset and plotted for MinPts 6. The number of noise points is obtained by multiplying ρ and size of data set. MinDist for each point is calculated based on a proposed formula. If there are points p and q such that MinDist(p) ≥ MinDist(q) where p is noise, but q is not, those points lying on the right of the selected x axis value (x) are the noise points. Eps is obtained by using $(x/\sqrt{2})$. The introduction of the parameter ρ is advantageous because (1) in some applications, the users will know the noise ratio of the dataset in advance or it can be easily computed. (2) ρ is a relative (and not absolute) measure which is dependent on the distance function. Though noise ratio is easily identifiable parameter, epsilon cannot be absolutely omitted because it was replaced with the noise ratio. This method primarily depends on the value of MinPts; to find an optimum value of MinPts is a challenge.

A data-driven system for outlier flight identification was proposed by Jiang et al., focusing on the landing approach of the plane, which has a higher probability of fatalities; first, three checking points and landing approach output parameters are selected and retrieved from the quick access record (QAR) data in the flight operational quality assurance (FOQA) station in [9]. Second, if discrete parameters are involved in the training datasets, density-based spatial clustering of applications with noise (DBSCAN) algorithm is introduced for detection of outliers, and if discrete parameters are not involved in the training datasets, one-class support vector machine (SVM) model is applied. Lastly, aircrafts that deviate from the group's characteristics known as outlier planes are identified. Outlier planes are filtered across the checking points of different heights, offering essential guidance for landing safety management. The DBSCAN algorithm is used by choosing different Eps values keeping the MinPts constant and fixed at 5 in their approach, thereby identifying the outlier planes in each case aiding in reducing the aviation accidents.

Hao et al. presents a new approach for detecting outlier data based on DBSCAN algorithm, and the support vector data definition (SVDD) model to increase the accuracy of actual datasets by removing outliers in [10]. To begin with, the data were clustered and noise data were removed using the DBSCAN technique. Through adding the SVDD algorithm, the outliers in each class, the method considerably decreases the clustering error samples of DBSCAN clustering output. Dismissing the DBSCAN-identified outliers in the first step and the outlier clusters within a class of DBSCAN core points in the second step will substantially increase clustering accuracy. The Eps, MinPts, the kernel parameter, and the penalty parameter C are the four parameters that must be chosen in their approach. These parameters have a big impact on the experimental outcomes, and the optimal parameters for various datasets differ as well. These parameters were manually set and finding an optimal way for automatic selections of parameters is a challenge.

Dharni and Bnasal proposed an improved DBSCAN algorithm which works by splitting the dataset instead of using it as a whole in [11]. The datasets used in the experiment are taken from the UCI repository. The efficiency of clusters of the proposed method is effective in clustering results and more reliable, according to various experiments using different datasets in attribute relation file format with the aid of the WEKA software. The following conclusions can be drawn from the experiment findings and analysis of the model: For large datasets, the suggested technique will efficiently analyze the cluster. It splits the dataset instead of running over the entire dataset; therefore, the enhanced algorithm is more scalable than the density-based approach. The future work would be to propose an advanced DBSCAN algorithm for speeding up the model by using parallel programming as well as finding the appropriate values for Eps and MinPts for big datasets.

Ranjith et al. introduces novel anomaly detection-DBSCAN (NAD-DBSCAN), an unsupervised clustering method for detecting outliers in traffic video surveillance in [12]. This method groups the trajectories of moving vehicles of different sizes and shapes. If an incident never fits the trained model, the trajectory is considered to be irregular. To find the sum of clusters for a data point dynamically, Eps and MinPts are the necessary parameters. The proposed work uses the trajectories of driving objects to identify anomalies such as violation of rules, accidents, haphazard driving, and other questionable events. The nonreachable feature of the DBSCAN method is used to track abnormal trajectory directions. On a small traffic dataset, the tests are evaluated, and an accuracy of 68.70 percent is obtained. The development of trajectory simplification methodologies to improve irregular detection accuracy is their future work.

To detect the varying degree of investment capability according to different areas, DBSCAN is the clustering algorithm used in this analysis by Nabarian et al. [13]. The findings revealed that seven aspects of regional investment capability seemed to have a correlation of greater than 50% with investment realization in Indonesian provinces. Provinces were divided into three groups, by choosing Eps as 6.8 and MinPts as 3, for potential investment and 0.85 and 2, respectively, for realization investment. According to the cluster similarities, one approach for growing investment in Indonesia is to raise the province's or region's investment potential and capability. It was concluded that increasing the investment potential has a positive effect on investment realization by analyzing the clustering outcomes obtained by DBSCAN.

A cattle detection method is introduced in the article by Ismail et al. using a pre-trained fast-region based convolutional neural networks and the Caffe architecture [14]. Making use of the clustering technique on the performance of cow identification is suggested for the classification of cows. To compare the clustering approaches to conclude which is a better technique at identifying herds and outliers, researchers used K-means and DBSCAN algorithms. In addition, Euclidean and Manhattan distance were used in the experiment to study the impact of clustering with these two types of distance metrics. DBSCAN outperforms K-means with respect to herd trend and outlier identification in live-stock tracking, according to the findings. Euclidean

distance produces a greater classification of herds in *K*-means method. Manhattan distance, on the other hand, provides improved herd and outlier identification in DBSCAN algorithm. It is believed that the distance metric method determines identifying herds and outliers are different in the two clustering techniques used. Finally, it is concluded that in conjunction with Smart Farming 4.0, DBSCAN clustering is best suited for livestock tracking for local ranchers.

Ghanbarpour and Minaei propose an extension of DBSCAN (EXDBSCAN) in [15]. EXDBSCAN is a DBSCAN-based approach for covering multidensity datasets. This approach just needs the user to input a single parameter. It can not only detect clusters of multiple densities but can also identify the outliers correctly. The outcomes of a comparison of this system's final clusters with those of two other clustering techniques on certain multidensity datasets suggest that their method performs better in such cases. The approach suggested in the work will identify clusters with various densities and resolve the issue of density-based clustering methods such as DBSCAN and OPTICS when multiple densities clusters are taken because of using specific Eps for each of the cluster. This approach employs a greedy algorithm to find a cluster density and then extend the cluster using the discovered parameter. However, calculating a density for every cluster requires time, so their approach takes longer time to run when compared to DBSCAN which is one of the drawbacks of this method. Whereas, the comparison of results of their methodology with two well-known techniques DBSCAN & Make Density-Based Cluster in identifying clusters and outliers demonstrates the benefits of EXDBSCAN approach over the other two.

Du et al. suggest a novel approach for accurate local outlier identification using statistical parameters that combines clustering-based concepts when using large data sets in [16]. To begin with, the approach uses the 3 sigma standard to find certain density peaks in the dataset. Second, the other data items in the dataset are classified to the very same cluster just like its higher-density nearest neighbor. Ultimately, they classify local outliers of every category using Chebyshev's inequality and density peak reachability. The outcomes indicate that the proposed approach is effective and reliable at detecting both global and local outliers. Furthermore, the approach has been shown to be more stable than traditional outlier detection approaches like Local Outlier Factor and DBSCAN. The approach is not only correlated with a single statistical parameter which could be calculated without a priori domain information, but it is also tolerant to parameter change, ensuring that this method is stable under a variety of conditions. The experiment is carried out on both synthetic and benchmark datasets. The future work of the paper is to use the latest local outlier tracking technique to find irregular GPS trajectories in taxicab drivers. Additionally, their idea is to integrate the new method into the Hadoop architecture to increase its performance even more.

To address the DBSCAN algorithm's limitation of high time expense, an updated DBSCAN method is introduced by Meng'Ao et al. on the basis of grid cells that enhances DBSCAN's very time-consuming area query mechanism and avoids many unwanted query operations by splitting data space into grid cells in [17]. The effect of the grid cell dividing approach on the algorithm is then studied. By selecting the best dividing process, it will improve the algorithm's performance. The accuracy and time complexity of the DBSCAN algorithm based on grid cells have been demonstrated experimentally which shows a higher accuracy in less time. The improved algorithm retains the benefits of the original method while also incorporating grid technologies for reducing the number of distance computations and increasing performance. The experiments suggest that this modified algorithm performs better than the original approach.

Thang and Kim propose a new approach called DBSCAN-MP for evaluating the input parameters of DBSCAN in [18]. In this method, every cluster can have differing Eps and MinPts. As the network environment evolves over time, they suggest a method for modifying normal behavior by adjusting cluster size or generating new clusters. In contrast to similar clustering techniques, the findings suggest that the efficiency has improved in the proposed approach. It is appropriate for detecting anomalies in network traffic of various sorts. Their findings demonstrate that it has a higher detection rate and a lower false positive rate when compared to techniques which make the same dataset assumptions and make use of the same dataset, KDD Cup 1999.

For identifying outliers from real-time monitoring trajectories, Dai et al. propose a trajectory outlier identification method using DBSCAN as well as velocity entropy in [19]. First, unusual subtrajectories are identified using an updated DBSCAN method that detects trajectory outliers with local features. Second, a comparison of the velocity entropy of the trajectories is made from a global perspective and the trajectory outliers are identified. The principle of trajectory confidence is suggested as a way to assess the efficacy of trajectory outlier identification performance, which decreases the rate of false detection in addition to increasing the accuracy. Ultimately, a test is carried, demonstrating that the system outperforms TRACLUS method. The future work of the paper is to identify trajectory outliers from low-confidence trajectories and assess the irregularity of the trajectories from different perspectives, in order to increase identification accuracy and extract more details from the incomplete trajectories.

The above presented noteworthy works clearly picturizes the problems being faced in finding the Epsilon parameter and MinPts and how some of the authors have used different techniques to tackle it. However, these works do not talk about the time taken to find the input parameters which is an important aspect.

The AQI [20] of New Delhi, Bangalore, Kolkata, and Hyderabad has been calculated using three different techniques: support vector regression (SVR), random forest regression (RFR), and CatBoost regression (CR). Random forest regression yields lower root mean square error (RMSE) values in Bangalore (0.5674), Kolkata (0.1403), and Hyderabad (0.3826) and higher accuracy in comparison to

SVR and CatBoost regression for Kolkata (90.9700%) and Hyderabad (78.3672%), while CatBoost regression yields lower RMSE values in New Delhi (0.2792) and the highest accuracy in New Delhi (79.8622%) and Bangalore (68.6860%). These findings were made after comparing the results of imbalanced datasets. Using the synthetic minority oversampling technique (SMOTE) algorithm on the given dataset, it can be observed that random forest regression yields the lowest root mean square error (RMSE) values in Kolkata (0.0988) and Hyderabad (0.0628), as well as higher accuracy values (93.7438%) and Hyderabad (97.6080%) when compared to SVR and CatBoost regression. On the other hand, CatBoost regression yields the highest accuracy values in New Delhi (85.0847%) and Bangalore (90.3071%). This unequivocally showed that datasets treated with the SMOTE method yielded improved accuracy. The most innovative aspect of this work is that the top regression models were selected by careful consideration and accuracy analysis. Furthermore, SMOTE is used for dataset balance, in contrast to the majority of similar articles.

This study [21] delves into six years of air pollution data from 23 Indian cities, aiming to conduct an in-depth analysis and prediction of air quality. The dataset has undergone thorough preprocessing, with key features meticulously selected based on correlation analysis. Exploratory data analysis has been employed to unveil hidden patterns within the dataset, pinpointing pollutants directly influencing the air quality index. Notably, a substantial decline in almost all pollutants is evident in the pandemic year, 2020. Addressing the issue of data imbalance, a resampling technique has been applied. The study utilizes five machine learning models for air quality prediction, with the outcomes compared against standard metrics. Notably, the Gaussian Naive Bayes model attains the highest accuracy, while the Support Vector Machine model demonstrates the lowest accuracy. Rigorous evaluation and comparison of these models are conducted using established performance parameters. Among the models employed, the XGBoost model emerges as the most effective, showcasing superior performance and achieving the highest linearity between predicted and actual data.

3. Design and Methodology

In conclusion, the importance of existing methodologies in avoiding air pollution lies in their capacity to regulate emissions, promote cleaner technologies, encourage sustainable practices, and engage communities. Combining these methodologies with advanced prediction techniques, such as machine learning, can enhance our ability to monitor, understand, and proactively address air pollution challenges. Cluster analysis are among the most powerful unsupervised learning techniques frequently used in data mining and machine learning, aimed at finding important and prospective information in the dataset. DBSCAN is a typical representative built on a clustering method that could group clusters of any shape and helps in identifying data for noisy samples. The data points that are dense and close together are grouped into a cluster. Furthermore, DBSCAN has an important benefit, since while clustering it does not need cluster information. In many applications it is commonly used since it is not noise-sensitive and because high-dimensional data overlook the data shape and size. These benefits make it an approach that is becoming extremely popular [16, 22]. The clustering performance relies on the value of the parameter. DBSCAN technique have been used to determine if an object is core or not by use of two significant input parameters, Eps and MinPts. DBSCAN can have varying values of MinPts and Eps in various data sets.

3.1. Dataset Description for Air Pollution Prediction

(i) Data sources:

The dataset comprises a combination of sources, including air quality monitoring stations, meteorological data from weather stations, satellite imagery, and land-use information.

(ii) Variables:

The key variables in the dataset include hourly measurements of air pollutants such as PM2.5, PM10, nitrogen dioxide (NO_2), sulphur dioxide (SO_2), ozone (O_3), and carbon monoxide (CO). Meteorological parameters include temperature, humidity, wind speed, and atmospheric pressure. Land-use data include information on urban density and green spaces.

(iii) Temporal and spatial resolution:

The data are collected at hourly intervals over a span of two years (2019–2021). The spatial resolution covers a metropolitan area, with a grid of monitoring stations spaced at approximately 1-kilometer intervals.

(iv) Geographical coverage:

The dataset covers the metropolitan area of Delhi city, encompassing both urban and suburban regions, with a focus on areas with significant industrial and vehicular activities.

3.2. Training Details

(i) Data preprocessing:

Preprocessing involves handling missing values through interpolation, outlier detection using statistical methods, and normalization of numerical variables. Additionally, categorical variables are one-hot encoded.

(ii) Feature engineering:

Features are engineered to include lagged values of air pollutant concentrations to capture temporal patterns. Interaction terms between meteorological variables are also created to account for potential synergies or antagonistic effects.

(iii) Model selection:

A machine learning [23] ensemble model, combining gradient boosting and random forests, is

chosen for air pollution prediction due to its ability to handle nonlinear relationships and capture complex interactions.

(iv) Training set, validation set, and test set:

The dataset is split into a training set (70%), a validation set (15%), and a test set (15%). The training set is used for model training, the validation set is used for hyperparameter tuning, and the test set is used for final model evaluation.

(v) Hyperparameter tuning:

Hyperparameters are tuned using a randomized search approach, optimizing for mean squared error on the validation set.

(vi) Cross-validation:

A 5-fold cross-validation is employed during hyperparameter tuning to assess the model's performance across different subsets of the training data.

(vii) Training duration and hardware:

The model is trained on a high-performance computing cluster with GPU support, and the training process takes approximately 24 hours.

3.3. Performance Parameters

(i) Evaluation metrics:

The model's performance is evaluated using metrics such as mean squared error (MSE), mean absolute error (MAE), and R-squared on both the validation and test sets.

(ii) Validation results:

The model achieves an MSE of 20 on the validation set, indicating good performance in capturing the variability of air pollutant concentrations.

(iii) Test set results:

On the test set, the model maintains a similar level of performance, with an MSE of 22. This suggests that the model generalizes well to new, unseen data.

(iv) Comparison with baseline models:

The developed model is compared with baseline models, including a naive persistence model and a linear regression model, demonstrating a significant improvement in predictive accuracy.

3.4. Limitations and Challenges. Limitations include potential data gaps in satellite imagery during adverse weather conditions and the inherent difficulty in predicting rare, extreme pollution events. The model may also be sensitive to changes in monitoring station locations.

Some important terms and properties of the DBSCAN algorithm are described below [10].

(i) Eps-neighborhood of a point: The Eps-neighborhood for a point m also called neighborhood radius is defined as the neighborhood in a given Eps radius for an object.

(ii) Directly density-reachable: If point m is in the Eps-neighborhood of point n, and n is a core point, this means m is directly density-reachable from object n.

(iii) Density-reachable: A point m is density-reachable from a point n with respect to Eps and MinPts if there exists a chain of objects $m1, \ldots, mz$, $m1 = n$, $mz = m$ such that $mi + 1$ is directly density-reachable to mi.

(iv) Density-connected: A point m is density-connected to a point n with respect to Eps and MinPts if there exists an object o such that both, m, and n are density-reachable from o. In other words, if there exists an object such that both m as well as n are density-reachable, then the object m is said to be density-connected to point n.

There are three kinds of points, which are core points, border points, and noise [11].

(i) Core point: A core point in a data set is that point where the number of points is greater than the MinPts in the Eps-neighborhood of the point. These would be the points inside the cluster. In other words, if the object's Eps-neighborhood includes at least MinPts number of objects, then the object is referred to be the core point.

(ii) Border point: In the given radius (Eps), a border point contains lesser than the MinPts, yet it is within the core point's neighborhood. In other words, if a point is not a core point and lies in the Eps-neighborhood of a core point, it is called a border point.

(iii) Noise point: Noise is regarded to be those objects which are not included in a cluster. Any point which is neither a core point nor a boundary point is an outlier or a noise point.

According to these definitions, a density-based cluster is a group of density-connected objects which is maximum with respect to density reachability.

By examining the Eps neighborhood of each object in the dataset, DBSCAN determines the cluster. In case the number of points in the Eps neighborhood is larger than MinPts, a cluster is created, and p would be the core. This technique applies iteratively to every p-object still not categorized. Figure 2 shows the three types of objects [15].

The DBSCAN Algorithm is shown below [5] (see Algorithm 1).

In DBSCAN, the heuristic approach for determining Eps and MinPts parameters of the smallest cluster in the dataset is easy yet efficient. Every point mapped to the distance to its k-th nearest neighbor is called the k-distance (k-dist) function. The graph of such a function provides a few details concerning the dataset's density distribution while sorting the data points in the decreasing order of the k—dist values. This graph is known as k-dist plot. In this graph, the first point could be the threshold in the first valley and the maximum MinPts—dist value is deemed to be noise and rest points belong to any of the clusters. The DBSCAN algorithm flowchart is shown in Figure 3.

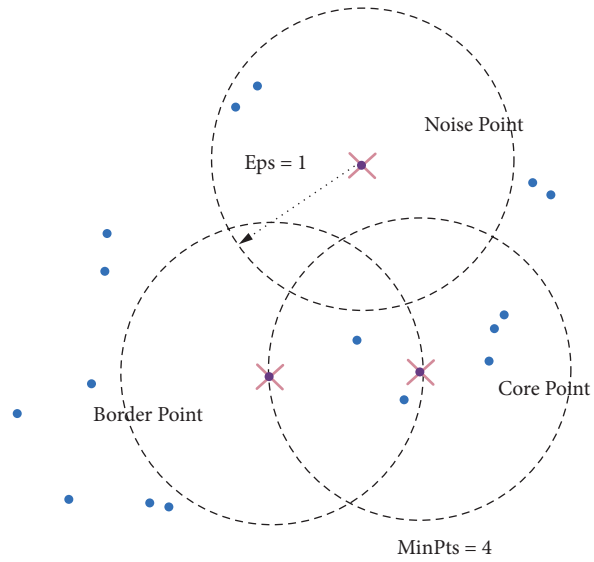

Figure 2: Core points, border points, and noise points.

Input: DS, Eps, MinsPts
Output: Clusters based on the density of data points
(1) While (DS has a point which is undetermined as noise or not)
(2) Choose a random undetermined point *P*.
(3) If *P* is not a core point,
(4) mark it as an outlier.
(5) Else
(6) retrieve all the density-reachable points from (*P*) building a cluster
(7) having (*P*) and mark all the points of this cluster as determined.
(8) End While

Algorithm 1: DBSCAN algorithm.

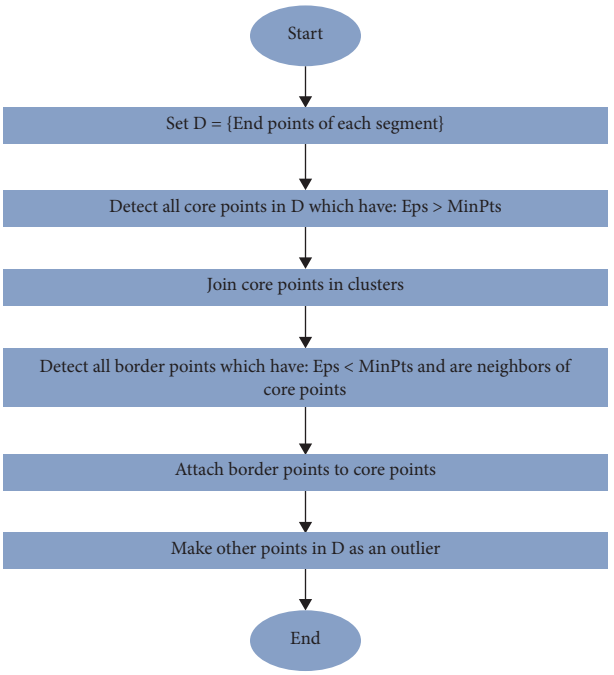

Figure 3: DBSCAN algorithm flowchart.

To find the Eps parameter to DBSCAN, a k-dist plot is drawn using the search tree algorithm and KNN algorithm. The search tree algorithm works in the following way. It constructs a binary tree and at each node data points are split. In the first layer, the first axis is used and in the second layer, the second axis is used; i.e. considering the tree is starting at layer 1, at every odd level/layer of the tree, the X axis is used to compare and at every even layer, Y axis is used. The algorithm is explained in detail with the help of an example.

Suppose there are a few data points (5, 4), (2, 6), (13, 3), (8, 7), (3, 1). For simplicity, 2 Dimensional data are considered. As shown in Figure 4, the nodes at each layer are X aligned and Y aligned alternatively. Now suppose a new data point, say (10, 2) comes. In the first layer, since it must X aligned, the corresponding X values for the existing node and the new data points are compared. If the latter value is greater than the former, the traversal further is to the right of the tree. But if it was the other way round, the traversal would be to the left sub tree. And finally, the exact position at which the new node must be inserted into the search tree is found.

The search tree can also be represented in a X-Y graph as shown in Figure 5. The vertical double headed arrow represents the splitting of X axis and the horizontal arrow represents the splitting of Y axis. The small balls indicate the data points.

Suppose the nearest neighbor of a target data point, say (9, 4) is to be found. Traversing down the tree happens using the insertion method and it finds that (8, 7) is a candidate nearest neighbor with a distance r, the line joining the target data point and the candidate nearest neighbor makes some random angle with respect to the horizontal axis as shown in Figure 6. But there could be a possibility of a closer neighbor. This is because while traversing down the tree, Y in the first node and X in the next node and so on were ignored. This inconsistency would be overcome while recursing back in the tree.

If the line (with a distance r') that is directly perpendicular to the section that that was not visited when traversing the tree is shorter than the line with distance r, then there is a possibility that, this section might have a point closer than the previously found candidate nearest neighbor. Figure 7 shows the recursion direction to traverse to the other section which was not visited to find the nearest distance to the target data point.

When the tree is recursed back, it checks for the section that was not visited. Let the distance between the unvisited section and the target data point be r'. If that distance is smaller than the best-found distance so far (i.e. $r' < r$), then traversing continues and the process is repeated until the condition becomes false (i.e., stops when $r' < r$). But during recursing back, every node will not be visited. If the height of the tree is H, then only 2H nodes will be visited which makes it a logarithmic search with respect to number of nodes in the tree. Since the distance to every node need not be calculated, unlike KNN, the search tree technique takes a lesser time to find the distance to its k-nearest neighbors

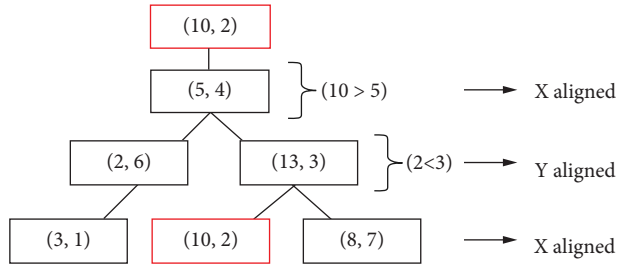

Figure 4: Insertion of a data point into the tree.

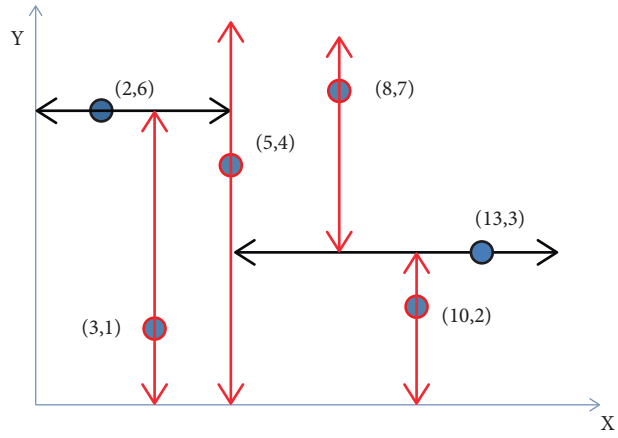

Figure 5: X-Y graph of data points with splits.

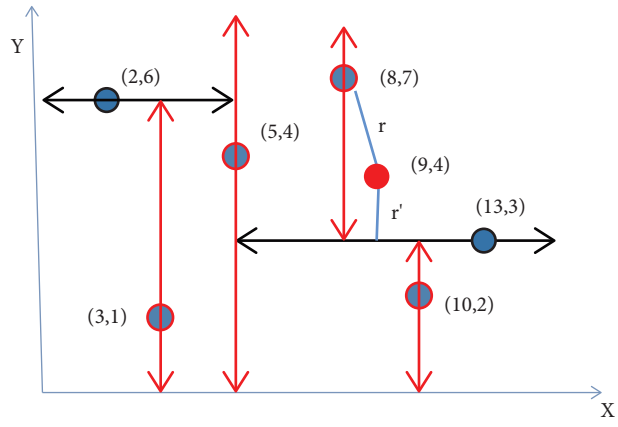

Figure 6: X-Y graph with a target data point.

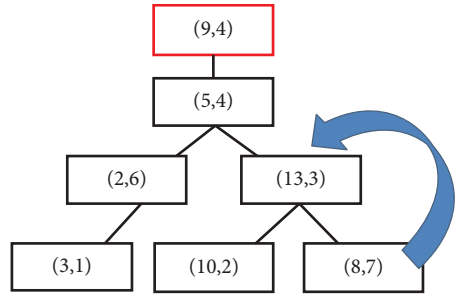

Figure 7: Recursing back the tree.

(where k = MinPts). Both search tree and KNN techniques calculate the distance between each point and its k-nearest neighbors and plots the k-distances graph of distances v/s sorted data points. The optimal value for Eps is the point at which the graph has the highest slope. The search tree technique works equally well for higher dimension data. In this case too, every node will have two branches, i.e., a binary tree will be constructed, but the number of values stored in a node could be arbitrarily large. The system architecture is illustrated in Figure 8.

In the first phase, data are collected and understood. In this work, an air pollution dataset is considered [24]. There are five attributes which are considered, PM2.5, AT, PM10, WS and WD. In the data preprocessing phase, the null values are updated and the data are cleaned and scaled. Feature reduction is done using Principal Component Analysis. The attributes are reduced to two principal components (PC1 and PC2) since it accounts to approximately 75% of the variance. In the next stage, the input parameter Eps is found using the search tree approach and KNN algorithm. The MinPts is set to 10, which is twice the number of dimensions considered. Next, the DBSCAN algorithm is modeled and finally the performance evaluation is done.

4. Results and Discussions

The following are the software and hardware requirements used for implementing the work:

(i) Software: The program is developed in Python language using Jupyter notebook. Scikit learn is used to import the necessary packages.

(ii) Hardware: 16 GB RAM, Intel i5 CPU, 64 bit OS.

First step is to understand the data. The air pollution data set obtained has 32413 rows and 7 attributes. The head of the dataset is shown in Table 1. The attributes' description is as follows:

(1) From Date: Gives a timestamp of the start of the hour at which values were collected

(2) To Date: Gives a timestamp of the end of the hour till which values were collected

(3) PM2.5: Particulate matter 2.5 (measured in micro grams/cubic meter—ug/m^3)

(4) PM10: Particulate matter 10 (ug/m^3)

(5) WS: Wind Speed (measured in meter/second)

(6) WD: Wind Direction (measured in degree)

(7) AT: Air Temperature (measured in degree Celsius)

The data types of the attributes are described in Table 2. The "From Date" and "To Date" are of type objects. The other attributes PM2.5, PM10, WS, WD, and AT are of floating point numbers which involves only numbers with decimal points. Float64 corresponds to float data type in Python.

The matrix visualization of missing values of the five attributes affecting air pollution in the dataset is shown in Figure 9. White line indicates the missing values. Most of the missing values are seen in PM10 attribute which has the most number of white lines. No specific pattern of missing values can be observed to indicate the correlation between missing values, for instance, no prediction can be made that such that if there is a missing values in attribute X, then there would be a missing value in attribute Y.

The null values can be graphically seen using a bar chart in Figure 10. The count of missing values can be represented with the bars where Y axis indicates the number of missing values and X axis indicates the attributes. The number of missing values are PM10 with 2160, PM2.5 with 875, AT with 589, WD with 313, and WS with 137 missing values.

As a next step of analysis, a complete description of variables is derived which provides various other useful statistical information about the data. Table 3 shows the statistical data description which in general shows the count, mean, standard deviation, minimum value, 25% quartile value, 50% quartile value, 75% quartile value, and the maximum value of each variable present in the dataset for each of the attributes.

A visualization of how the attributes values have changed over time is shown in Figure 11. The values of all the attributes over time, where the From Date on the X axis starts from January 1st 2017 and goes up till the 12th July 2021, the Y axis represents the values of all the attributes. Since the initial data are unscaled, there are differences in the range of each of the attributes. The legend shows the colors along with its attributes.

Because the attributes of the dataset will not be on the same scale, the whole dataset must be standardised, i.e., the scope and the range and magnitude of each attribute in the dataset are unique. A raise of 1 point in PM2.5 is not the same as a 1 point raise in PM10 and vice versa. If one attribute has greater fluctuation in its data, it will impact the distance computation significantly. With the scale of the attributes, all characteristics are averaged to zero and a standard deviation of one. The scaled data are shown in Table 4.

Once the data are scaled, the dimensionality reduction is done using principal component analysis. The proper number of principal components must be determined. PCA analysis is shown in Figure 12. The number of features that would account to approximately 75% of the variance must be the number of principal components. The horizontal red line indicates the line drawn corresponding to the 75% variance, the vertical red line indicates the number of features required in the PCA, which corresponds to 1.5 features, so after rounding off, it is decided that 2 principal components would be required.

The head of the data frame transformed after the creation of the two principal components is shown in Table 5. Once the data set is reduced from five features to two principal components as decided by the PCA analysis, the data will be transformed in terms of two principal components PC1 and PC2.

To find Eps, the k-dist plot is drawn using both search tree algorithm and the KNN algorithm. The knee point, which is the point at which a sharp increase in the curve in the k-dist graph is observed, is the Eps value to be fed to DBSCAN.

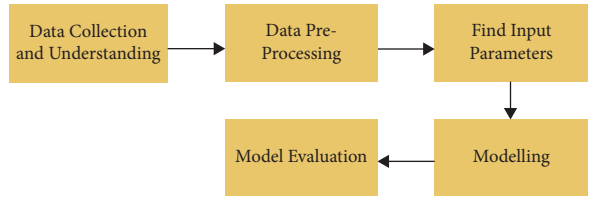

Figure 8: System architecture.

Table 1: Head of the dataset.

	From date	To date	PM2.5	PM10	WS	WD	AT
0	01-01-2017 00:00	01-01-2017 01:00	162	227	0.93	127.25	14.5
1	01-01-2017 01:00	01-01-2017 02:00	165.17	226.17	0.51	103.5	13.66
2	01-01-2017 02:00	01-01-2017 03:00	176.67	215.33	0.79	96.92	13.31
3	01-01-2017 03:00	01-01-2017 04:00	150.67	177.5	1.13	106.75	13.54
4	01-01-2017 04:00	01-01-2017 05:00	129.5	174.5	0.31	197.25	12.91
5	01-01-2017 05:00	01-01-2017 06:00	145	286	0.3	155.33	12.29

Table 2: Data type of attributes.

Attributes	Data type
From date	Object
To date	Object
PM2.5	Float64
PM10	Float64
WS	Float64
WD	Float64
AT	Float64
Dtype	Object

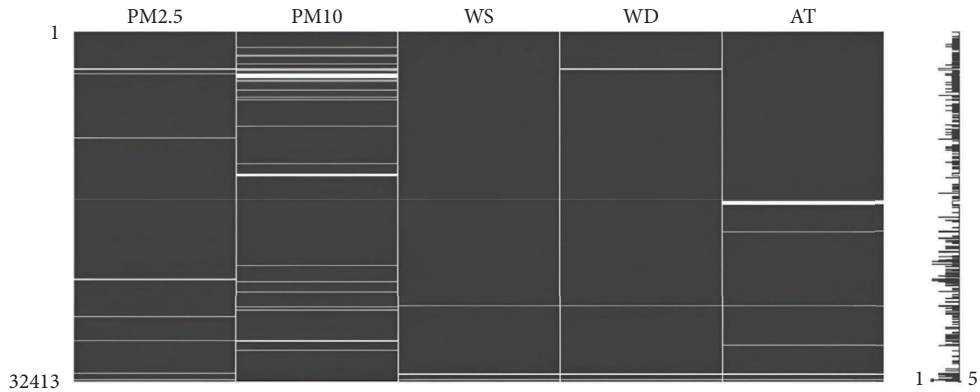

Figure 9: Matrix visualization of missing values.

Figure 13 shows the distances vs sorted data points k-dist graph for search tree algorithm. The time taken to find the distances and plot the graph is 161 milliseconds. The Eps value is found to be 0.05 which is the knee of the graph. The search tree algorithm takes a lesser time for every data point since for every data point the distance calculation need not be from one point to all other points.

Figure 14 shows the distances vs sorted data points k-dist graph for KNN algorithm. The time taken to find the distances and plot the graph is 208 milliseconds. The Eps value is found to be 0.05 which is the knee of the graph. The KNN algorithm takes more time since for every data point the distance calculation will be from one point to all other points.

As we can see from Figure 13, the time taken to find the distances and plot the graph to find the Eps parameter using the search tree approach is less than the time taken to find the epsilon using the broadly used KNN algorithm shown in Figure 14.

Since the DBSCAN algorithm's performance varies widely with respect to the Eps parameter, determining the Eps efficiently and in less time is very important. As the

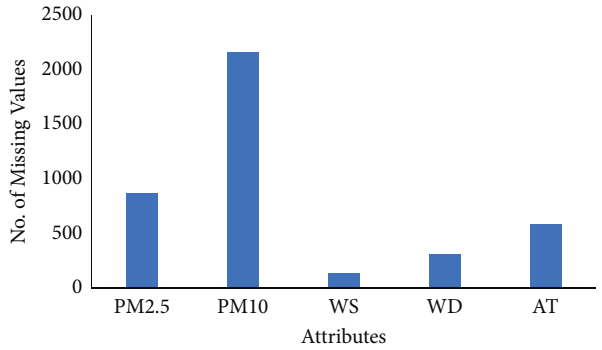

Figure 10: Bar chart depicting missing data.

Table 3: Statistical information of the dataset.

	PM2.5	PM10	WS	WD	AT
Count	32413.000000	32413.000000	32413.000000	32413.000000	32413.000000
Mean	144.382056	305.896467	0.925494	194.098125	26.235772
Std	132.115717	208.465311	0.901458	77.963890	8.575732
Min	0.200000	1.000000	0.120000	9.000000	1.300000
25%	54.750000	155.750000	0.300000	113.750000	19.520000
50%	100.000000	257.500000	0.500000	216.750000	26.910000
75%	189.526316	403.500000	1.250000	267.000000	32.950000
Max	985.000000	1000.000000	8.360000	338.250000	47.240000

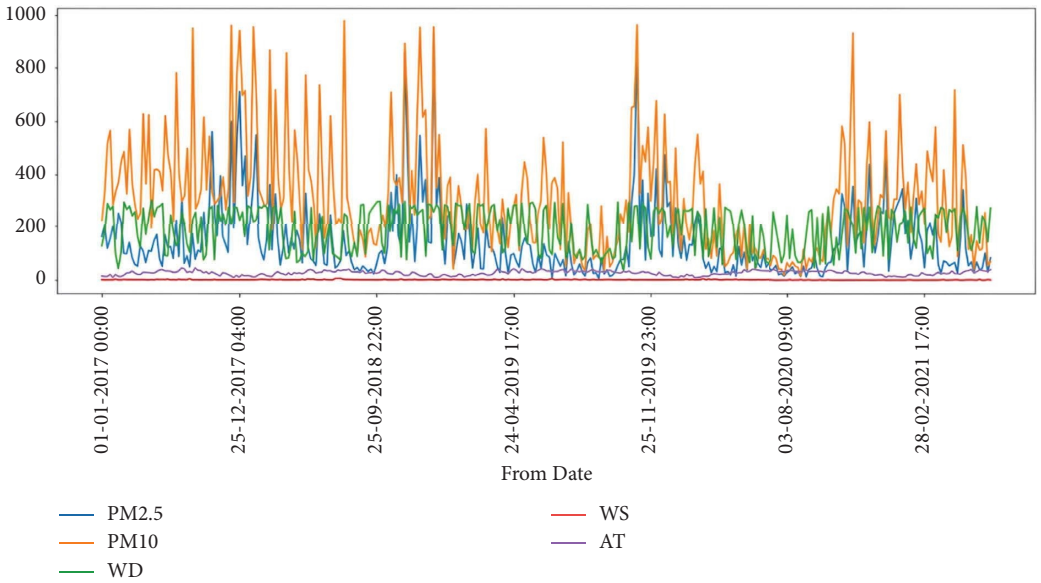

Figure 11: Graph of attributes over time.

Table 4: Scaled data.

	PM2.5	PM10	WS	WD	AT
0	0.133354	−0.378469	0.004999	−0.857437	−1.368508
1	0.157349	−0.382451	−0.460920	−1.162070	−1.466460
2	0.244395	−0.434451	−0.150308	−1.246470	−1.507273
3	0.047595	−0.615922	0.226865	−1.120384	−1.480453
4	−0.112646	−0.630313	−0.682787	0.040428	−1.553917

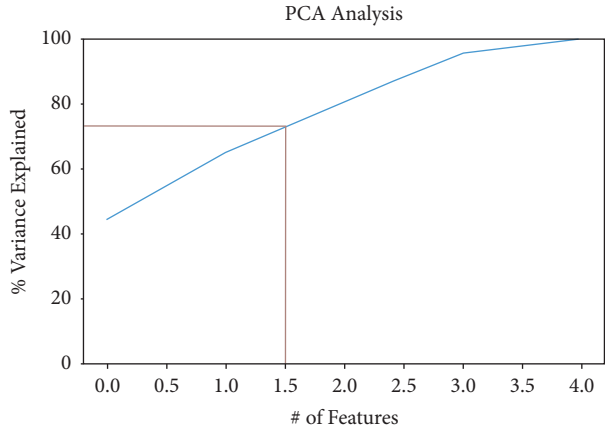

Figure 12: PCA analysis.

Table 5: Head of principle components.

PC1	PC2
0.316819	0.632772
0.488654	0.726437
0.403257	0.922526
0.059669	0.856911
0.506939	−0.52165

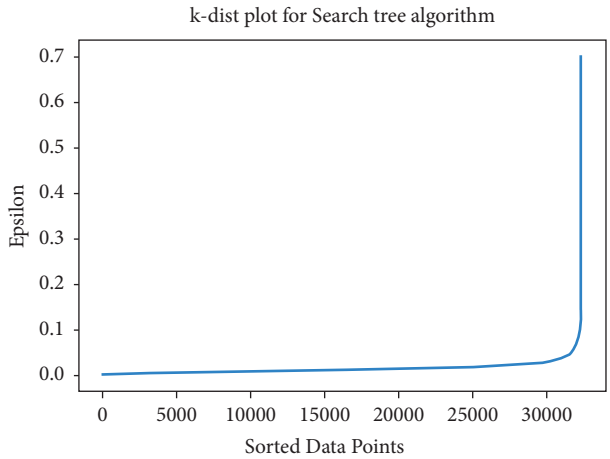

Figure 13: *K*-dist plot for search tree algorithm.

dataset size increases, the search tree approach to find Eps in less time is of great significance.

A scatter plot of the data points for the PC1 and PC2 before clustering is shown in Figure 15. The Eps value was found using the search tree approach and the KNN algorithm, and the MinPts are fed to the DBSCAN algorithm to find clusters and outliers and the scatter plot is plotted to obtain the clustering results.

The DBSCAN clustering result is shown in Figure 16. A scatter plot is drawn. Three clusters are formed. Cluster 0 represented by blue points, Cluster 1 in green and Cluster 2 in yellow. The purple points are the outliers. Most of the data points lie in the cluster 0, followed by cluster 1, then cluster 2. Remaining fall under noise points.

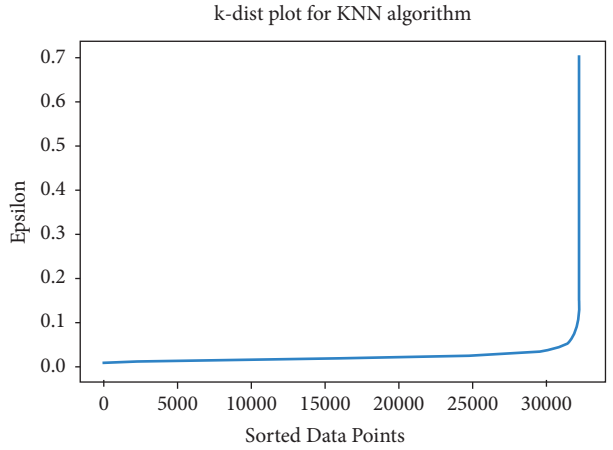

Figure 14: *K*-dist plot for KNN algorithm.

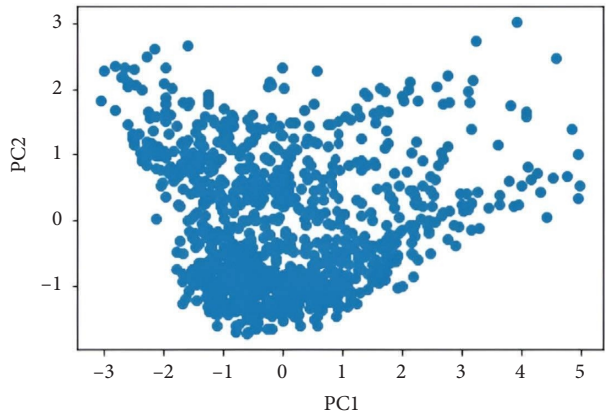

Figure 15: Scatter plot for data points.

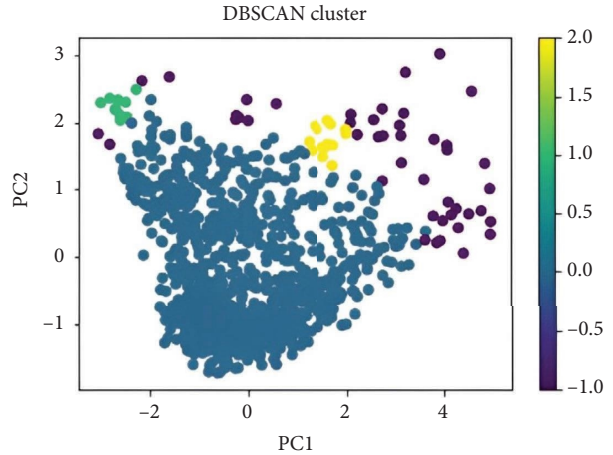

Figure 16: DBSCAN cluster.

Accuracy of a clustering algorithm can be measured using the Silhouette score performance evaluation metric which ranges from −1 to +1, −1 being the worst and +1 being the best score [25]. Other performance metrics for clustering algorithm, which are the Davies-Bouldin score, Calinski–Harabasz score

Table 6: Performance evaluation.

SI. No	Performance metric	Value obtained
1	Silhouette score	0.82
2	Davies-Bouldin score	0.72
3	Calinski–Harabasz score	0.78
4	Entropy	0.41

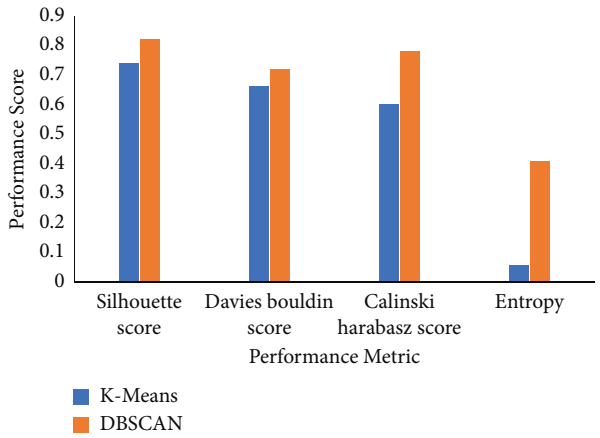

Figure 17: Performance evaluation for DBSCAN v/s K means.

and entropy lies in the range [0, 1], where higher the entropy, poorer the clustering.

The performance evaluation of the DBSCAN algorithm using 4 metrics is shown in Table 6. According to these values, the clustering done by the model by giving the Eps and MinPts values gives a satisfactory performance with all the performance metric values obtained being fairly good in terms of clustering evaluation.

Performance evaluation for DBSCAN v/s KMeans algorithms is shown in Figure 17.

The Silhouette score, Davies-Bouldin score, Calinski–Harabasz score of DBSCAN is greater than KMeans whereas entropy of DBSCAN is lesser than KMeans. This clearly shows that, DBSCAN has clustered the data better than KMeans clustering algorithm. DBSCAN does not require to be fed number of clusters as a parameter, detects clusters of any shape and size in addition to detecting outliers in comparison to KMeans algorithm. Therefore, based on experimental results of this work, DBSCAN has outperformed KMeans algorithm on this dataset.

4.1. Comparative Analysis

(i) Nature of task:

If the goal is to identify clusters of pollution sources or regions with similar pollution patterns, DBSCAN is useful.

(ii) Data characteristics:

If the data are noisy, and pollution sources are spatially clustered, DBSCAN might provide valuable insights.

(iii) Computational complexity:

DBSCAN has a lower computational complexity but may still require tuning of parameters which is done using hybridization.

(iv) Robustness:

DBSCAN is robust to outliers and noise due to its density-based nature.

In conclusion, the choice between DBSCAN and other algorithms depends on the specific goals of air pollution prediction task. If primarily interested in clustering spatial patterns, DBSCAN is useful.

5. Conclusions

DBSCAN is a clustering algorithm which helps in applications such as clustering the dataset into different levels of air pollution. Unlike other clustering algorithms, the ability of DBSCAN to find clusters and outliers for data of any shape and size makes it a broadly used clustering technique. The algorithm requires two input parameters Epsilon and Minimum points which are highly sensitive; in that, marginal changes of these values affects the DBSCAN clustering and performance. Finding out the Eps in less time is highly essential as the dataset size increases. In this work, a search tree technique and KNN algorithm are used to find the Eps parameter. It is found that, the time taken to find the Eps value using the search tree technique is less than the KNN algorithm. This is because unlike KNN, the search technique does not visit every node to find the distances to its k-nearest neighbors which makes it a better technique in saving time and improving the efficiency. A k-dist graph is plotted and the knee point of the k-dist plot is considered as Eps. The MinPts are taken to be twice the number of dimensions. These input parameters are fed to DBSCAN and the clustering results are obtained. The outcomes of this work are early warning systems, optimized environmental policies, public health interventions, data-driven urban planning, improved monitoring networks. The future scope is integration with IoT and sensor networks, multi-sensor fusion, health impact assessment, global collaboration.

Data Availability

The labelled datasets used to support the findings of this study can be obtained from the corresponding author upon request.

Conflicts of Interest

All authors declare that they have no conflicts of interest.

Acknowledgments

The authors acknowledge the support from Ramaiah Institute of Technology, REVA University, Manipal Institute of Technology, Nagarjuna College of Engineering & Technology, for the facilities provided to carry out the research.

References

[1] A. Fino, *Air Quality Legislation. Encyclopedia of Environmental Health*, Elsevier, Amsterdam, Netherlands, 2nd edition, 2018.

[2] Y. Chen, D. Chen, T. Song, and K. Song, "An intelligent and portable air pollution monitoring system based on chemical sensor array," in *Proceedings of the IEEE 4th International Conference on Frontiers of Sensors Technologies (ICFST)*, pp. 21–25, Shanghai, China, November 2020.

[3] P. Nejat, H. M. Hussen, F. Fadli, H. N. Chaudhry, J. Calautit, and F. Jomehzadeh, "Indoor environmental quality (IEQ) analysis of a two-sided windcatcher integrated with anti-short-circuit device for low wind conditions," *Processes*, vol. 8, no. 7, p. 840, 2020.

[4] B. Mu, M. Dai, and S. Yuan, "DBSCAN-KNN-GA: a multi Density-Level Parameter-Free clustering algorithm," *Institute of Physics Conference Series: Materials Science and Engineering*, vol. 715, no. 1, Article ID 012023, 2020.

[5] Z. Akbari and R. Unland, "Automated determination of the input parameter of DBSCAN based on outlier detection," *Artificial Intelligence Applications and Innovations*, pp. 280–291, 2016.

[6] A. Starczewski, P. Goetzen, and M. J. Er, "A new method for automatic determining of the DBSCAN parameters," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 10, no. 3, pp. 209–221, 2020.

[7] N. Soni and A. Ganatra, "AGED (automatic generation of Eps for DBSCAN)," *International Journal of Computer Science and Information Security*, vol. 14, 2016.

[8] J. Esmaelnejad, J. Habibi, and S. H. Yeganeh, "A novel method to find appropriate ε for DBSCAN," in *Intelligent Information and Database Systems. ACIIDS 2010. Lecture Notes in Computer Science*, N. T. Nguyen, M. T. Le, and J. Świątek, Eds., Springer, Berlin, Heidelberg, 2010.

[9] Y. Jiang, R. Liu, N. Le, and Y. Zheng, "A method for the outlier flights detection of the final approach based on FOQA data," in *Proceedings of the IEEE 1st International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, pp. 7–11, Kunming, China, October 2019.

[10] S. Hao, X. Zhou, and H. Song, "A new method for noise data detection based on DBSCAN and SVDD," in *Proceedings of the IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 784–789, Shenyang, China, June 2015.

[11] C. Dharni and M. Bnasal, "An improvement of DBSCAN Algorithm to analyze cluster for large datasets," in *Proceedings of the IEEE International Conference in MOOC, Innovation and Technology in Education (MITE)*, pp. 42–46, Jaipur, India, December 2013.

[12] R. Ranjith, J. J. Athanesious, and V. Vaidehi, "Anomaly detection using DBSCAN clustering technique for traffic video surveillance," in *Proceedings of the Seventh International Conference on Advanced Computing (ICoAC)*, pp. 1–6, Chennai, India, December 2015.

[13] T. Nabarian, S. Sutoto, N. Gusmawati, D. P. Trimaratus Sholehah, A. Nizar Hidayanto, and A. M. Sari, "Clustering of provinces in Indonesia based on regional investment capacity with density-based spatial clustering of applications with noise method," in *Proceedings of the 5th International Conference on Computing Engineering and Design (ICCED)*, pp. 1–6, Singapore, April 2019.

[14] Z. H. Ismail, A. K. K. Chun, and M. I. Shapiai Razak, "Efficient herd – outlier detection in livestock monitoring system based on density– based spatial clustering," *IEEE Access*, vol. 7, pp. 175062–175070, 2019.

[15] A. Ghanbarpour and B. Minaei, "EXDBSCAN: an extension of DBSCAN to detect clusters in multi-density datasets," in *Proceedings of the Iranian Conference on Intelligent Systems (ICIS)*, pp. 1–5, Bam, Iran, February 2014.

[16] H. Du, S. Zhao, D. Zhang, and J. Wu, "Novel clustering-based approach for local outlier detection," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 802–811, San Francisco, CA, USA, April 2016.

[17] L. Meng'Ao, M. Dongxue, G. Songyuan, and L. Shufen, "Research and improvement of DBSCAN cluster algorithm," in *Proceedings of the 7th International Conference on Information Technology in Medicine and Education (ITME)*, pp. 537–540, Huangshan, China, November 2015.

[18] T. M. Thang and J. Kim, "The anomaly detection by using DBSCAN clustering with multiple parameters," in *Proceedings of the International Conference on Information Science and Applications*, pp. 1–5, Jeju, Korea (South), April 2011.

[19] W. Dai, C. Zhang, X. Su, and S. Cao, "Trajectory outlier detection based on DBSCAN and velocity entropy," in *Proceedings of the International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, pp. 550–557, Rhodes, Greece, November 2020.

[20] N. Srinivasa Gupta, Y. Mohta, K. Heda, R. Armaan, B. Valarmathi, and G. Arulkumaran, "Prediction of air quality index using machine learning techniques: a comparative analysis," *Journal of Environmental and Public Health*, vol. 2023, Article ID 4916267, 26 pages, 2023.

[21] K. Kumar and B. P. Pande, "Air pollution prediction with machine learning: a case study of Indian cities," *International journal of Environmental Science and Technology*, vol. 20, pp. 5333–5348, 2023.

[22] W. Lai, "A new DBSCAN parameters determination method based on improved MVO," *IEEE Access*, vol. 7, pp. 104085–104095, 2019.

[23] M. Sowmya, S. A. Alex, A. Kanavalli, Supreeth, Shruthi, and Rohith, "Machine learning model for emotion detection and recognition using an enhanced convolutional neural network," *Journal of Integrated Science and Technology*, vol. 12, no. 4, 2024.

[24] Central control room for air quality management Delhi NCR: https://app.cpcbccr.com/ccr/.

[25] K. Shahapure and C. Nicholas, "Cluster quality analysis using silhouette score," in *Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, Sydney, NSW, Australia, October 2020.