Webology (ISSN: 1735-188X)
Volume 19, Number 2, 2022

Enhanced K-Means ++ Using Probability Theory For Efficient Prediction Of Air Pollution

Chetan Shetty , Sowmya BJ , Seema S , Nagasri K

Department of Computer Science and Engineering, M S Ramaiah Institute of Technology,
MSRIT Post, Bangalore 560054

Abstract

Increasing concentrations of air pollutants is a global concern as it is a major underlying cause for other serious issues like premature deaths, global warming, increased susceptibility to heart diseases, lung disorders and skin disorders. Exposure to particulate pollutants increases vulnerability to Covid-19 and risk of succumbing to the virus. Air pollution analysis is a widely undertaken study by government officials and research scholars. K-means is a frequently used algorithm to understand the condition of the atmosphere from massive sensor generated data. The algorithm however comes with its drawbacks. Random initialization of the initial centroids can lead to bad clustering, an alternative, K-means++ does away with this, however, takes more execution time and iterations which is not ideal. We propose an advanced K-means++ initialization algorithm which incorporates an oversampling factor for smarter initialization of centroids using probability theory and weight assignment. We also propose a probability based convergence algorithm as opposed to the regular convergence algorithm to smartly select a portion of the data points to recompute the centroids. This will ensure a faster formation of final clusters. Real time Bengaluru, India air pollution data is scraped, pre-processed and clustered using the proposed technique. All the variants of K-means under study are compared over parameters of execution time, iterations and performance metrics. This work is also extended to tackle future air data points using a fast ensemble model. The solution proposed is better in terms of being reliable, fast and helps with better clustering, which leads to better air quality analysis, which leads to better air quality prediction, which leads to taking apt precautions to mitigate and regulate the air pollution.

1. Introduction

Air quality monitoring is key in ensuring public health as it can reduce the effect of pollution on animal life, plant life and the earth. Air quality forecasting of concentration of main pollutants like O3, NO2, NOx and NMHC(Benzene) can reduce the effect of this dreadful pollution. Air pollution analysis is widely undertaken by government officials and research scholars. The K-means algorithm is used frequently to analyse massive datasets by clustering, since it has few drawbacks, K-means ++ is used to overcome those drawbacks.

In the proposed solution K-means++ advanced algorithm with probability fit function, which is an enhancement of K-means++ algorithm, for smarter initialization of centroids and formation of clusters using probability theory and weight assignment technique is applied on real time Bengaluru air data. Future air data points are also handled using a fast ensemble model. This proposed solution is better in terms of being reliable, fast and helps with better clustering, which leads to better air quality analysis, which leads to better air quality prediction that can predict the intensity of pollution with more accuracy and less execution time.

To use improved K-means++ algorithm for smarter initialization of centroids and formation of clusters by using probability theory and weight assignment on real time Bengaluru air data, scraped and preprocessed to derive better efficiency and accuracy and to extend the work to handle future air data using a fast ensemble model. The outcome is shown to be better on the basis of execution time, iterations and performance metrics. Consequently, enhancing the prediction of air pollutants.

The Objectives of the work are

- Obtain data scraped from Central Pollution Control Board (CPCB) using selenium module to obtain real time Bengaluru air data from 2018-2021.
- Apply various preprocessing steps to the data to improve data quality and to make it suitable to be input to the model.
- Implement pre-existing algorithms: Random initialization, K-means++ initialization and Regular clustering algorithm from scratch.
- **Advanced K-means++ initialization algorithm, Weighted clustering algorithm and Probability-based clustering algorithm**: Our proposed improved versions are implemented, tested and contrasted to algorithms in 3 on parameters: Execution time, Iterations, Speed Up obtained, Silhouette score, Davies Bouldin score and Calinski harabasz score.
- Cluster the air pollution data into sets of points with similar characteristics of air pollutants and use the clustered model to predict the intensity of air pollution
- Extend to predict AQI category of future air data points on the basis of the patterns found by the clustering algorithm and evaluate on prediction time and accuracy.
- Visualization and snapshots of results obtained and comparison of the same to provide evidence of the improvements made by our proposed work.
- Improve the **efficiency and accuracy** of the algorithm by improving the initialization algorithm of kmeans++ and the convergence algorithm.

Air pollutants being one of the major concerns of today, every world authority has invested in prediction techniques to take adequate precautionary and preventive measures which if implemented correctly can save lives and prevent environment exploitation. To achieve this goal, it is important to provide the predictions at the proper time, involving lots of analysis. Our project is capable of analyzing huge amounts of data and provides the result of air quality with decent accuracy, which allows us to take our preventive measures.

The project has a major deliverable, that is, an improved K Means algorithm i.e., Advanced Kmeans++ and an improved convergence algorithm - probability fit. The algorithm will be better in terms of execution time, iterations and performance metrics, when compared to traditional Kmeans. Our choice of dataset is the Air Pollution dataset, which is scraped from CPCB website. The system analyses air pollution dataset, which has quantities of various pollutants like CO, NOx, NO2,SO2 and O3, and predicts the extent of air pollution. The system also has a preprocessing module which handles missing values and visualises data. It uses the IQR method to remove outliers, handles skewness and uses normalization so that one attribute does not dominate. Once clustering is done using the proposed techniques, the work is further extended to handle future data points using a fast ensemble model.

With the growing need of ML Algorithms in critical parts of the industry, the flexibility of the algorithm to adapt to various patterns of data is necessary. The Advanced Kmeans++ algorithm and the probability based convergence algorithms proposed in this project is the first step in improving the traditional Kmeans to adapt to a variety of data. Air pollutants in our atmosphere have been a growing concern with the side effects of the same being exponentially alarming. Our government has been taking increased measures to handle the same by prediction techniques so that precautionary and preventive measures can be taken well in hand to safeguard the wellbeing and health of the citizens. Timely knowledge is invaluable and many researchers and scientists are constantly searching to develop effective and efficient ways to handle real-time air pollution data. Hence, our choice of dataset is the air pollution dataset.

2. Literature Survey

Air pollution kills approximately seven million people worldwide every year. World Health Organization data shows that 90 percent of people breathe air containing high levels of pollutants. The combined effects of outdoor and household air pollution cause about seven million premature deaths every year. Hence undoubtedly air pollution is becoming a growing concern globally. There has been increased efforts by authorities to scan and analyze the urban air in order to implement adequate control measures. Short-term prediction of air quality is a necessity in order to take preventive and evasive action. Instead of traditional deterministic modeling, researchers usually employed statistical methods to analyze and forecast air quality. Lately, many researchers have started to use various Machine Learning and Big Data Analytics approaches as there is a large availability of environmental sensing networks and sensor data available. A number of linear methods have been applied to time series data for air pollutants, especially ozone and NO2 prediction including comparisons with Artificial Neural Networks (ANNs). One of the most popular effective clustering methods widely applied to the air pollution problem is the K-Means clustering.

An incremental K-means algorithm (Sanjay Chakraborty.et.al, 2014) is proposed for periodic bulk updates in databases and the authors define a threshold that determines the point of database change upto which the proposed method performs much better. It’s aim is for better time, cost and effort in the use case of dynamic databases. This is applied on the West Bengal

Air pollution dataset and it is compared with the traditional K-means. Evaluation of the proposed methodology with tradition K-means is done on metrics of time and threshold. It is observed that upto 57% threshold incremental K-means does better than regular K-means.

In dynamic databases, data keeps getting appended with time. The proposed methodology reduces time as the number of scans over the dataset is reduced. However, after a certain % of change in data, the incremental approach suggested does not outperform the traditional approach. Future work can include parameters that increase this %.

This work proposes (Sujatha, S.et.al, 2013) a technique to assign the initial centroids to improve clustering performance of regular K-means. The authors suggest the use of Initial Cluster Centers Derived from Data Partitioning along the Data Axis with the Highest Variance to assign for cluster centroid. The goal of these modifications is to achieve better accuracy and faster convergence. The methodology is evaluated on popularly used datasets: Wine, Iris, Glass and Leukemia on the metrics of iterations, cluster distance and elapsed time. It was observed that the proposed methodology reduced the number of iterations consequently decreasing execution time. It also gave the maximum cluster distance in comparison to the traditional approach which is an indicator of better accuracy. The future work will be to increase the classification accuracy of the proposed approach. Moreover, the time taken by the proposed approach should also be considered. The time taken for classification should be very less with high accuracy.

This paper investigates (Shou-Qiang Wang.et.al, 2008) the approximate algorithm for the k-means clustering by means of selecting the k initial points from the input point set. An expected 2-approximation algorithm is presented in this paper. Meanwhile, an efficient algorithm for selecting the initial points is also proposed. At last some experimental results are given to test the validity of these algorithms. Three famous datasets: Iris, RuspIni and Spath Postal Zone are used. The proposed algorithm is run and the results compared with the optimal results that could be obtained for different values of K. The cost and average running time of all the algorithms under comparison is graphically depicted. It is observed that it performed better than regular K-means. The new algorithm cannot break through the bounding that it relies on the initial point and the result of this algorithm still be local minimum. It is standard practice to run the k-means algorithm multiple times, and then keep only the best clustering found. This area is future work as there has been research done to avoid this trap.

In this work, (G. R. Kingsy.et.al, 2016) an enhanced K-Means clustering algorithm is proposed to analyze the air pollution data. The correlation coefficient is calculated from the real time monitored pollutant datasets. The Air Quality Index value is calculated from the correlation coefficient to determine the air pollution level in a particular place. The proposed enhanced K-Means clustering algorithm is compared with the Possibility Fuzzy C-Mean(PFCM) clustering algorithm in terms of accuracy and execution time. Accuracy and execution time are the performance metrics taken. Five different datasets are used for testing the work proposed. On dataset 1: 86.5% accuracy with the proposed method and 75.69% accuracy using PFCM. Execution time was 140s for enhanced versions while PFCM took 152s. Also in other datasets the proposed method performed better than PFCM. Future work is to build a distributed version of the K-means Clustering algorithm where data or computational power is distributed. Efficiency can also be improved by using variable clusters instead of a constant 'K' number of clusters.

This study (Haraty, Ramzi.et.al, 2015) assessed the performance (Shahriar, Shihab A.et.al, 2021) of the application of hybrid models, that is, Autoregressive Integrated Moving Average (ARIMA)-Artificial Neural Network (ANN), ARIMA-Support Vector Machine (SVM) and Principal Component Regression (PCR) along with Decision Tree (DT) and CatBoost deep learning model to predict the ambient $PM_{2.5}$ concentrations.The data from January 2013 to May 2019 with 2342 observations were utilized in this study. Eighty percent of the data was used as training and the rest of the dataset was employed as testing. The performance of the models was evaluated by R^2, RMSE and MAE values. Among the models, CatBoost performed best for predicting $PM_{2.5}$ for all the stations. The RMSE values during the test period were 12.39 μg m^{-3}, 13.06 μg m^{-3} and 12.97 μg m^{-3} for Dhaka, Narayanganj and Gazipur, respectively. The authors note that ARIMA-ANN and catboost deep learning models prove to show potential for better accuracy and generalization capability and hence it is future work is to investigate and obtain concrete results to attest to the same.

The proposed methodology (Verma, Vikas.et.al, 2016) uses G-means algorithm, which utilizes a greedy approach to produce the preliminary centroids and then takes k or lesser passes over the dataset to adjust these centre points. The system implements a new technique that can cluster very large data which sometimes k-means could not, while minimizing the number of reads of the entire dataset. The experimental results, which were used in an increasing manner on the same dataset, showing that G-means outperforms k-means in terms of entropy and F-scores. In terms of the coefficient of variance and the execution time the experiments also yield better results for G-means. The drawback with this is that the initial centroids selections are not going to change, yet the change is going to reside in the second step where the centroids are being compared to the constant k. This shows that the algorithm faces a dilemma where the chances are the same for both of the points.

The proposed methodology (Sadiq .et.al, 2017) discussed designing a system of air quality management based on a distributed and adaptive problem-solving approach. The proposed system provides an intelligible tool for the air quality data collection and analysis, based on the use of K-means algorithm and Hadoop framework: HBase for data storage and Map Reduce process for data processing. A data warehouse is integrated with the Hadoop engine to exploit its MapReduce parallel processing power, the implementation of the K-means clustering algorithm over distributed data gathered from different air quality monitoring stations and managed using a Hadoop based system. This provides a robust and efficient system for processing big volumes of data. K-means algorithm and Hadoop framework in air quality analysis data results in fast data loading, fast query processing, highly efficient storage utilization. This provides a robust and efficient system for processing big volumes of data.

The proposed methodology (Chakraborty.t.al, 2011) uses the Incremental K-means and DBSCAN algorithms. These algorithms are efficient compared to their existing algorithms with respect to time, cost and effort. Using an air pollution database, the performance evaluation of incremental DBSCAN clustering algorithm is compared with the performance of incremental K-means clustering algorithm. The characteristics of these two algorithms based on the changes of the data in the database and some logical differences between the two when they are applied on real time dynamic databases is explained. As a result, with respect to time analysis the incremental K-means clustering performs better than the incremental DBSCAN clustering. After the comparison of Incremental K-means and Incremental DBSCAN Algorithms, the result, with respect to time analysis the incremental K-means clustering performs better than the incremental DBSCAN clustering.

The methodology (Gómez-Losada Á.et.al, 2018) uses clustering statistical techniques such as agglomerative hierarchical clustering, hidden Markov models (hmm) and k-means as a modelling approach to characterize this pollution regime while deriving reliable information such as estimates of exposures related to background pollution as its mean, acuteness and time incidence values in the ambient air for all the air pollutants and sites studied.

Four well-known clustering techniques(fmm, hc, hmm and km) were compared under the same probabilistic framework. As a result, outperformed with respect to the rest of clustering techniques studied. The information obtained from hmm when analysing background pollution may result in interest for epidemiological research in that it provides a full characterization of the background pollution. Mean , standard deviation and representation of background pollution may be used as estimates of exposure to this fraction of pollution in ambient air, and hence to better understand the implications of background pollution on the population's health.

The methodology (Rauf.et.al, 2012) proposed for K-means Clustering calculates initial centroids instead of random selection. The process is divided into two phases. In phase one, cluster size is fixed and it outputs initial clusters. In phase two, cluster size is varied and its output is the final clusters. The output of the first phase is input to the second phase. The proposed algorithm is then compared against the basic K-means algorithm and also against other enhanced algorithms. The performance of the proposed K-mean clustering algorithm in terms of number of iterations and time complexity is improved with respect to its basic version.

Future scope can be to extend this method to text-based clustering as for now integer data is used.

The methodology (Bahman Bahmani.et.al, 2012) proposed is an initialization algorithm k-means|| which obtains a nearly optimal solution after a logarithmic number of passes and then it is shown that in practice a constant number of passes is sufficient. The simple and parallel algorithm involves sampling and reclustering points to obtain initial centres for Lloyd's iterations which admits easy realization in any parallel computational model like MapReduce. A non-trivial analysis shows that k-means achieves a constant factor approximation to the optimum. Experimental results on large real-world datasets (on which many existing algorithms for k-means take long to execute) show that k-means has a better clustering cost and runs in less iterations and demonstrates scalability of k-means. Since there are many modifications to the basic k-means algorithm for specific purposes, future scope can be to efficiently parallelize these modifications.

The methodology (Moustris, Kostas .et.al, 2010) proposed uses Artificial Neural Network (ANN) to forecast the maximum daily value of the European Regional Pollution Index (ERPI) as well as the number of consecutive hours, during the day, with at least one of the pollutants above a threshold concentration, 24 to 72 hours ahead in the Greater Athens Area (GAA), Greece. The model predicts reliably 3 days ahead (for year 2005), the excesses or non-excesses days ($ERPI\geq50$), with success index ranging between 84.6% and 92.2%. Also predicts reliably the days with eight consecutive hours with $ERPI\geq50$ with success index between 86.1% and 99.7%. Coefficients of determination between real and predicted values of the ERPI and consecutive hours during the day with $ERPI\geq50$ are found to be statistically significant ($p<0.01$). In the future, to obtain more reliable forecasts for the air quality in the GAA, available input data and quality (no blank days) can be increased which are important for the ANN model training.

The methodology (Saksena.et.al, 2002) proposed is a hierarchical agglomerative clustering algorithm using the average linkage between groups method and the Euclidean distance metric. Spatial patterns of air pollution in Delhi over a ten-year period using three criteria pollutants was studied. This method can be used for an exploratory study of spatial patterns and/or data quality issues of air pollution in Indian cities using the National Ambient Air Quality Monitoring System data in the absence of the huge data sets needed for more sophisticated space-time modelling. Robust and consistent results were obtained with log-transformed data. The results provided evidence of a systematic difference between measurements made by the two agencies involved in monitoring air quality (NEERI and CPCB). This was more apparent in the case of SO2 and NO2 measurements, than SPM measurements (a comparatively simpler pollutant to measure). Future work can be to do similar analysis on PM10 data when it becomes available. Also, cluster analysis can be based on daily data (not yet available in the public domain in India) and to do this seasonally.

The proposed methodology (Vitolo, Claudia .et.al, 2018) identifies the probabilistic dependence structure of the environment-health relationship using multivariate Bayesian Networks and Big Data technologies. The heterogeneous data consists of environmental factors (weather), exposure levels (pollutants), and health outcomes (mortality rates). Expectation- Maximization (EM) algorithm was used to include air pollution data set's observations with missing values, to generate the Directed Acyclic Graph (DAG) – graphical model of dependencies. Results show that for pollution and weather variables the model does well using the training set but also has good predictive power on the testing set. The model generalizes well to new regions and time periods. EM algorithm is found to be of great use as availability of input data like air pollution is usually scarce. Future works can play with train-test set splits. Also, can implement simulated scenario modelling and test the accuracy and effectiveness of model predictions of the environmental and weather variables when data quality is not very good.

First check if the given data set contains the negative value attributes or not. If the data set contains the negative value attributes, (Madhu, Yedla.et.al, 2010) then transform all data points in the data set to the positive space. Next, for each data point calculate the distance from origin. Then, the original data points are sorted. After sorting the partition, the sorted data points into k equal sets. In each set take the middle points as the initial centroids. Then, K-Means is applied to all data points and centroids are updated in each iteration by taking the mean of data points in the cluster. The proposed algorithm has more accuracy with less computational time compared to the original k-means clustering algorithm. The sorting method used determines the time complexity to find initial centroids. Since the proposed enhanced method uses heap sort, its overall time complexity becomes O(nlogn) in both average and worst case.

The proposed algorithm is computational intensive. Hence, future work can work on parallelising the computationally - intensive part of the algorithm. Also, automating the determination of the value of k is suggested as a future work.

K-Means Clustering algorithm (Navjot Kaur.et.al, 2012) is an idea, in which there is a need to classify the given data set into K clusters, the value of K (Number of clusters) is defined by the user which is fixed. This algorithm consists of four steps: Initializing, Classification, Centroid Recalculation, Convergence Condition. Search of relevant records or similar data search is a most popular function of databases to obtain knowledge. There are certain similar records that we want to fall in one category or form one cluster. That's why, we need to rank the more relevant student marks by a ranking method and to improve search effectiveness. In order to reduce the execution time, the Ranking Method is used. And also show how clustering is performed in less execution time as compared to the traditional method. This work makes an attempt at studying the feasibility of K-means clustering algorithm in data mining using the Ranking Method. The proposed ranking based K-means algorithm produces better results than that of the existing k-means algorithm. In case of clustering the marks of students from different databases are considered by using the concept of Query redirection. By using the Query redirection approach we can easily cluster the large amount of data from a distributed environment as from different databases. If this approach is considered, then the performance of the K-means clustering algorithm is improved.

The study focused on the pattern recognition (Azid.et.al, 2014) of Malaysian air quality based on the data obtained from the Malaysian Department of Environment(DOE). Eight air quality parameters in ten monitoring stations in Malaysia for 7 years (2005–2011) were gathered. Principal component analysis (PCA)in the environmetric approach was used to identify the sources of pollution in the study locations. The combination of PCA and artificial neural networks (ANN) was developed to determine its predictive ability for the air pollutant index(API). The PCA has identified that CH4, NmHC, THC, O3, and PM10 are the most significant parameters. The PCAANN showed better predictive ability in the determination of API with fewer variables, with R2 and root mean square error (RMSE) values of 0.618 and 10.017, respectively. The work has demonstrated the importance of historical data in sampling plan strategies to achieve desired research objectives, as well as to highlight the possibility of determining the optimum number of sampling parameters, which in turn will reduce costs and time of sampling. The future work for this paper includes applying other Machine learning or deep learning techniques on the historical data sample collected and comparing it against each other to search for the best method.

The Euclidean distance formula of K-Means algorithm (Bansal.et.al, 2017) is enhanced to increase the cluster quality. The enhancement will be based on normalization. First, calculate normal distance metrics on the basis of normalization. Second, functions will be clustered on the basis of majority voting. The process flow is as follows: First, data inputs, generated by sigma and random functions are given to the system. When all data has been generated, simple k-means is applied. After applying normalization on that data, we give scatter data. Now, we read text file data that generates data after that hierarchy k-means is applied before normalization. After this, the iteration process starts. This process is continued until a nearest point to an accurate position with generating data is obtained. There is a big variation between execution time of existing KMeans algorithm and proposed algorithm. And with this proposed algorithm the accuracy level gets almost double. So it can be said that the proposed modification in the existing k-means algorithm will give a huge improvement to the clustering techniques. The future work for this paper may involve parallelizing techniques for further improving the execution time.

The approach builds models (Mauro Castelli.et.al, 2020) for hourly air quality forecasting for the state of California, using one of the most powerful existing machine learning (ML) approaches, namely, a variant of support vector machines (SVMs), called support vector regression (SVR). The proposal is to build an SVR model for the prediction of each pollutant and particulate measurement on an hourly basis and an SVR model to predict the hourly air quality index (AQI) for the state of California. SVM has three hyperparameters that need to be user-defined: the kernel type function, the regularization constant C, and the maximum allowed deviation ε. Time-series split combined with random grid search was used to obtain the optimal numbers for both C and ε. This approach achieved results as good as the ones achieved by the grid-search algorithm. The pollutants used in the dataset are CO, NO2, SO2, ozone, and PM2.5. Both PCA SVR-RBF and SVR-RBF achieved similar performance in forecasting the AQI. Nevertheless, there is an underestimation of the pollution levels when the highest AQI values were registered by PM2.5, which would imply that pollution alerts would not be sent to the affected population groups in those cases. Investigate the usage of SVR to forecast air quality through the following topics: Dataset and variable selection—considering a large dataset with more parameters and measurements. SVR parameter optimization—as SVR model performance is greatly influenced by kernel function selection and the penalty parameter C, it’s interesting to explore other methods for hyperparameter optimization such as genetic algorithms or particle swarm optimization.

3. Design and Implementation

System is designed to be a stand-alone application. Any authorized user of an underlying operating system shall be able to launch the system on the software interface using input commands. System provides a framework for determining problems, learning and providing solutions with big data and cost effective computing power.

With our work we improve the K-means++ for smarter initialization of centroids and formation of clusters, by utilizing probability theory and weight assignment. Accuracy of the clustering is improved in our algorithm by dealing with the initialization problem that the original K-means faced. The algorithm is applied on real-time air pollution data which is processed effectively and efficiently. Our algorithm also aims to reduce the number of iterations normally taken by the original K-means, making the total execution time much less and realistic which is much required to deal with real-time data.

Application of our work to the available air pollution data will help us draw valuable precise conclusions about the air quality that can be used wisely to positively impact the society.

The system collects required dataset (here,air pollution dataset) from CPCB (Central Pollution Control Board), using a data scraping process in the Data Collection stage. The dataset is passed through a pre-processing module of the system. This module is responsible for cleaning the dataset (like removing outliers, replacing missing data, normalisation) in order to improve the processing and train the system with good data. Feature engineering is done on the dataset, which implies using domain knowledge to extract features from raw data. So the dataset will consist of randomly sampled values of various parameters which are known to be the major pollutants. Then the dataset is given to the K-means++ advanced module, which applies the algorithm that is coded to the dataset and puts each datapoint available in the set to one of the ‘K ‘clusters based on the similarity of the datapoint with the cluster. The system also focuses on minimising WCSS (Within Cluster Sum of Squares) value, so that each cluster is as distinguishable as possible. Probability fit function of the algorithm improvises clustering of the dataset using probability theory and weight assignment. This smartly converges thereby maintaining accuracy and at the same takes much less execution time. Each cluster is labelled with a class name (say, as less polluted, moderately polluted and highly polluted) based on AQI (Air Quality Index) calculated for each cluster. So, given a new datapoint, it is assigned to one of the clusters and hence predicts the level of pollution. Further incoming data is classified based on the patterns found by the clustering done previously using CatBoost-A fast ensemble model selected to reduce prediction time as much as possible while still giving good accuracy.

Our work aims to enhance KMeans algorithm by improving its Clustering capabilities. Here, we perform clustering on the air pollution dataset, display and analyze the results accurately and efficiently using improved K-means (K-means++ Advanced) which uses the Probability Fit function. The major attributes used to cluster by the algorithm are identified through feature engineering. Preprocessing functionality deals with outliers, missing values and duplicate values. Normalization of the attributes is done so that one attribute does not dominate. To build our prototype input data is collected from the Central Pollution Control Board (CPCB). Our work is implemented in the python language with Jupyter Notebook as our platform.

The K-means++ algorithm functions uses an initialization algorithm that deals away with the drawback of bad clustering due to the original way of choosing random initial centroids. Improvement of the K-means++ is done with Advanced K-means++, this deals with reducing the number of iterations in the former method. As the dataset grows larger the regular convergence algorithm becomes time consuming, hence we also suggest an improvement of using a probability based convergence algorithm that smartly converges thereby reducing the execution time and maintaining good performance metrics. Our overall proposed clustering algorithms will provide better efficiency and accuracy. To extend this work to real time, a fast ensemble model is further used to predict the AQI category of new data points on the basis of the patterns found by the K-means algorithm.

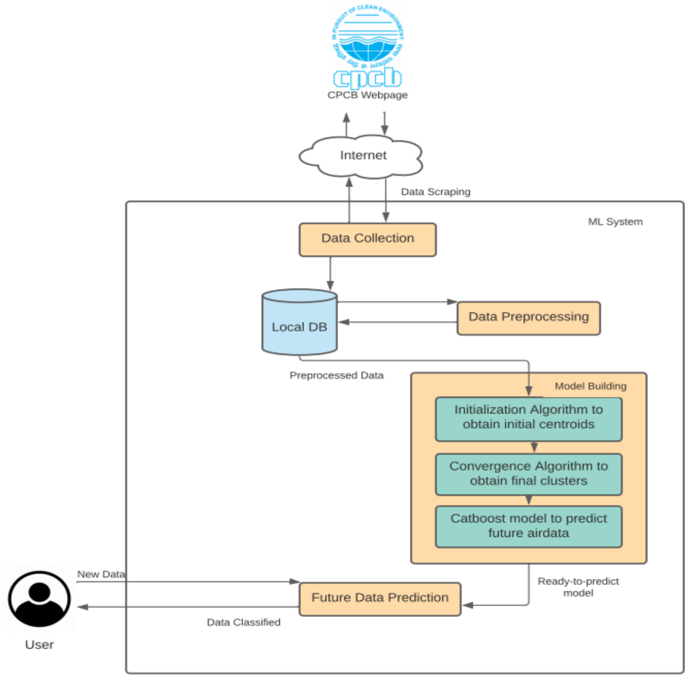


Figure 1: Architecture Design Diagram for the proposed methodology

Fig 1 shows the architecture design diagram that outlines the major project modules and the relationship between them. Data is scraped from the CPCB web portal via the internet and is then transferred to the local database. This is queried by the data preprocessing module which takes the raw data and transforms it into a format suited to the chosen models and writes it back to the local storage. Again from the local storage the data is consumed by the model building module that encompasses three distinct algorithms: Initialization Algorithms, Convergence Algorithms and the Future Data Prediction Algorithms. The trained and final model is then used to predict the AQI category of future data points as and when new data is inserted into the system by the User. This is overall architecture followed by the proposed methodology.

The aim of our work's design as seen above is to use improved k means (using k means++ initialization algorithm and probability fit function) as our technique on air pollution dataset on which was collected using data scraping and data preprocessing techniques have been applied in order to derive better efficiency and accuracy of the algorithm. Therefore enhancing the prediction of air pollutants.

Air pollution is visibly becoming a critical issue that needs immediate and adequate attention and solutions to assure public wellbeing. Researchers are constantly trying to develop more accurate prediction models. Hence, using our improvised algorithm to predict air pollution is one more advantage of the project.

The implementation begins with the Data Collection step. In order to develop a machine learning model, data in proper format is a paramount requirement. To collect data, we employ

Data Scraping, which is the process of extracting information from a website into a spreadsheet or local file saved on our system. Data Collection ends with storing the data in a proper tabular format so that it is easier to apply further machine learning steps.

Next we do Data Cleaning. In order to refine the crude data, it is subjected to standard preprocessing steps to clean up and convert it to the format which is required by our K-Means algorithm. These preprocessing steps involve dropping unwanted columns, handling missing data, removing rows that have NaN(not a number) in all columns, imputing missing values, visualization of data, removing outliers and handling skewness in the data and normalization of data.

The K-means clustering algorithm, though simple to implement, has its own drawbacks. It takes many iterations to converge, very sensitive to initializations. The initialization algorithm, to obtain the initial centroids to start clustering, is evolved as follows:

- Random initialization algorithm involves random selection of data points as centroids. The disadvantage is that two centers can be produced in the same logical cluster.
- K-means++ initialization algorithm on the other hand combines randomness and probability. It uses probability to check the likelihood of a point being selected as centroid and randomness to choose such centroid. This method suffers the disadvantage of requiring ‘K’ passes over the data to complete its task and hence, is very slow on huge data.
- K-means++ advanced initialization algorithm uses weights along with randomness and probability. A new feature called **Oversampling Factor** is used which chooses more centroids than needed and then merges those centroids that are close to one another. Also the centroids are weighted according to the number of data points around it, in order to increase the likelihood of its selection. This method produces good clusters faster solving the problems of the previous methods.
  Next, the convergence algorithm, to obtain the final clusters, is evolved as follows:
- Regular K-means convergence algorithm iteratively calculates minimum distances to form the clusters and then update the centroids from the clusters formed per iteration, if reassignment does not happen at some stage then the clustering is complete.
- Probability K-means convergence algorithm computes probabilities and uses the pbound and raceToCenter factors to smartly select the data points used to recompute the new centroids. This improvement makes the formation of clusters more efficient especially as the dataset size gets larger and still maintains good accuracy. Execution ends once the convergence criterion is met.

Finally we use Catboost - A fast ensemble model to classify the AQI category of a new datapoint on the basis of the patterns previously discovered by the clustering algorithm.

Information about the implementation of Modules

Data Collection

Data collection for machine learning involves gathering and measuring information from several different sources. In order to develop machine learning solutions, it is necessary to collect and store data in such a way that it can be subjected to further machine learning steps like preprocessing and model building. Here we use 2 steps to collect data:

- **Data Scraping**
    - Data Scraping is a technique where a computer program is employed to extract information from a website and store it in our local system.
    - The main advantage of Data Scraping includes automation of the process and data accuracy due to less window for human error.
    - Python module Selenium is used for this purpose. Selenium is a tool for controlling web browsers through programs and performing browser automation.
    - The Selenium module creates a new webdiver and spans a new browser window, in which it opens the CPCB URL provided.
    - In the website landing page it fills up all the details asked - Station name, Parameters to be selected, Criteria, Start and End Date, Start and End Time.
    - Once everything is filled, the form is submitted and the website loads the corresponding data. Finally, the webdriver loads the data into a spreadsheet.
    - To fill the necessary details and download the dataset by clicking appropriate buttons, we make use of Xpaths or XML paths, which allows us to navigate through the HTML structure of the page.
- **Arranging data in tabular format**
    - We made an observation that the raw data downloaded from the CPCB website using Data Scraping had columns split and appended vertically instead of horizontal arrangement.
    - Hence, various slice and join operations are performed to render the data in a proper tabular format which is used in further steps i.e., Data Cleaning and Model Building.

Data Cleaning

Data Cleaning / Preprocessing involves applying various techniques on raw data in order to make it presentable in the format required by the model building module. The preprocessing is done in 6 steps:

- **Step1: Drop Missing Columns and rows**

  The columns containing all NaN values or more than half NaN values are dropped from the dataset. Also drop rows that have NaN or None values in all the columns.

- **Step2: Handle Missing Values:**

The remaining missing values are filled using the mean-before-after method. In this method, given an empty cell, we fill it with the mean of values present in the cell before it and after it.

- **Step3: Visualisation**

  The easiest way in understanding the structure of a big dataset is to visualize it. Python provides various visualization libraries to achieve this goal. The library made use of here is Seaborn. Seaborn heatmap plots the correlation matrix between the various features whereas the pairplot gives an overview of data distribution of each feature against all other features.

- **Step4: Outlier Analysis**

  An Outlier is a point that deviates significantly from the other points in the dataset. Keeping such points in the dataset distorts the model in the wrong direction. Hence, to avoid this we use InterQuartile Range (IQR) method to remove the outliers. The IQR method to remove outliers is as below:

  - We define Q3 which represents the 75th percentile of data and Q1 which represents the 25th percentile of data.
  - We then define IQR = Q3 - Q1
  - Drop all data points which are less than (Q1 - 1.5 * IQR) or greater than (Q3 + 1.5 * IQR)

- **Step5: Handle Skewness**

  Skewness in data is the measure of asymmetry in the probability distribution of the dataset. K-means expects symmetric distribution of data. Unhandled skewness might result in all data clumped into one cluster. Hence, handling skewness is necessary. To do this we use the log transformation function provided by numpy Python library - numpy.log (). This will transform the data into normal distribution.

- **Step 6: Normalization:**
  Normalization is often used to change values of numeric columns in a dataset to use a common scale. K-means clustering is ‘isotropic’ in all directions of space and therefore, produces more or less round clusters. Hence, unequal variances put more weight on points with low variances. Normalization is done by transforming data as below:
  - data = data - mean(data)
  - data = data / standard_deviation(data)

Model Building

The Model Building module takes in the preprocessed data from the previous module and applies the different types of initializations and clustering algorithms explained in section 7.4. All the necessary functions are defined in the class “OurKmeans”.

- The WCSS() function computes Within-Cluster-Sum-of-Squares (WCSS) which is the sum of squares of the distances of each data point in all clusters to their respective centroids. WCSS is used in finding the ideal number of centroids for clustering.

- **Initialization Algorithms:**
    - The method random_init() implements the basic initialization algorithm. It uses rd.randint() to get random centroids(explained in point 1 of section 7.4.1).
    - The kmeanspp_init() implements K-means++ initialization algorithm (explained in point 2 of section 7.4.1). All the data structures are managed using the numpy library using various functions like numpy.array(), numpy.append(), numpy.sum(), numpy.argmin().
    - The kmeans_adv_init() is one of the core methods that implements K-means++ advanced initialization algorithm (explained in point 3 of section 7.4.1). The libraries and data structure used to code it are very similar to the ones used for K-means++ implementation.

- **Clustering algorithms:**
    - The fit() method implements the Regular clustering algorithm (explained in point 1 of section 7.4.2). We feed the centroids obtained using any of the initialisation algorithms and cluster the remaining data points around these centroids.
    - The prob_fit() method implements the proposed improvement-the probability based clustering algorithm. The improvements suggested brought out by the hyperparameters and the method of computing and recomputing probabilities and centroids respectively. This is the implementation of the algorithm elaborated in section 7.4.2.

- The get_cluster_aqi() method is used to assign a particular Air Quality Index (AQI) level to each cluster. This is done with respect to the centroid of each cluster. The features of each centroid represent the concentration levels of the pollutants i.e. NO, NO2, O3, SO2 and CO. The worst concentration level reflects overall AQI. The method get_aqi_level() is used to map the maximum pollutant concentration to respective AQI category namely “Good” (0-50), “Satisfactory” (51-100), “Moderate” (101-200), “Poor” (201-300), “Very Poor” (301-400) and “Severe” (401 above) with sub levels namely 1st level, 2nd level and so on. This helps us in defining a label for each cluster.

- **Prediction algorithms:**
  Five select algorithms namely: Catboost, Adaboost, Light GBM, XG Boost and Random Forest are implemented using their respective library versions: CatBoost Classifier, Ada Boost Classifier, XGB Classifier, LGBM Classifier and Random

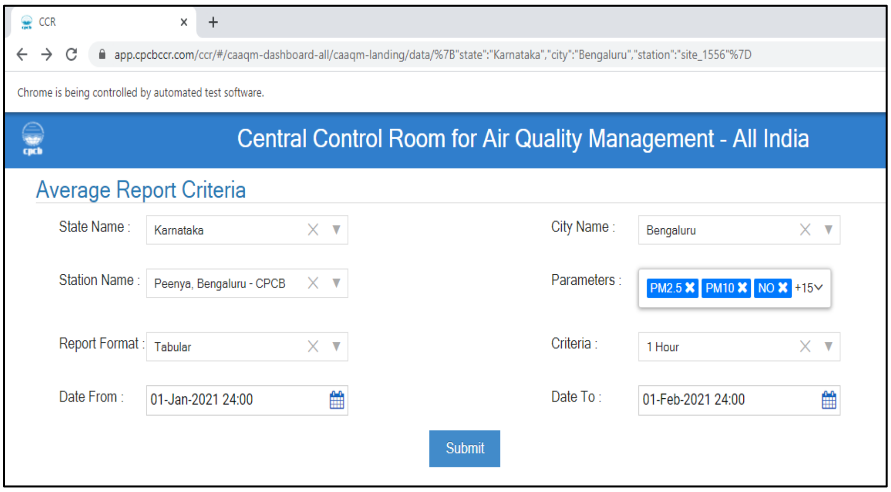
Forest Classifier and metrics from sklearn are used to evaluate their performance. Below are the functions used on each classifier:

  - model.fit() which trains the classifier model using the historical data patterns found by the K-means algorithm. The data fed into this phase consists of the air data columns and the AQI category column which signifies the AQI value of a particular cluster found previously.
  - model. predict() is used to predict the AQI Category of a future test datapoint.
  - accuracy_score() is then used to measure numerically the correctness of the predictions made.

- **Performance Comparison:**
  - For evaluating the quality and performance of clustering, we use Silhouette score, Davies Bouldin index and Calinski Harabasz index since the ground truth labels are not known. These evaluations are performed using the estimated labels and the model itself in the score_eval() method.
  - Silhouette score is calculated for each instance by the formula, Silhouette score = (x-y)/ max(x,y) where x is the mean nearest cluster distance and y is the mean intra cluster distance. This is computed using the find_sil_score() function.
  - Davies Bouldin index signifies the average ‘similarity’ between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves. This is computed using sklearn metrics inbuilt function.
  - The Calinski Harabasz index is the ratio of the sum of between-clusters dispersion and of inter-cluster dispersion for all clusters (where dispersion is defined as the sum of distances squared). This is also computed using sklearn metrics inbuilt function.
  - Silhouette score closer to 1, Davies Bouldin index closer to 0 and higher Calinski Harabasz index indicate better clustering.

4. Results and Inference

In this section we show and evaluate the results obtained starting from data scraping, data preprocessing, model building, all the way to comparison of the pre-existing method with our proposed work.



CCR

app.cpcbccr.com/ccr/#/caaqm-dashboard-all/caaqm-landing/data/%7B"state":"Karnataka","city":"Bengaluru","station":"site_1556"%7D

Chrome is being controlled by automated test software.

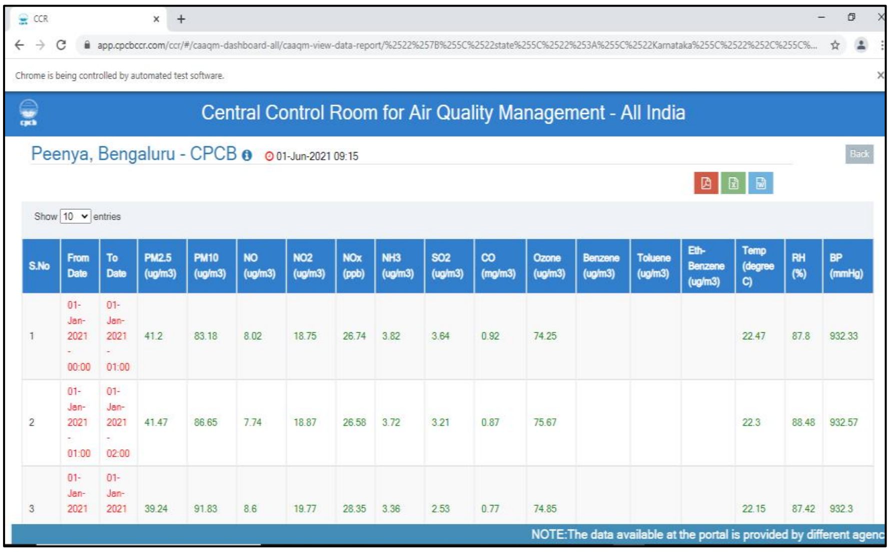
Central Control Room for Air Quality Management - All India

Average Report Criteria

State Name :	Karnataka	City Name :	Bengaluru
Station Name :	Peenya, Bengaluru - CPCB	Parameters :	PM2.5 ✖ PM10 ✖ NO ✖ +15
Report Format :	Tabular	Criteria :	1 Hour
Date From :	01-Jan-2021 24:00	Date To :	01-Feb-2021 24:00

Figure 2: (Central Pollution Control Board) Website

In Figure 2, after spanning the new browser and visiting the CPCB website, the Selenium webdriver fills up all the necessary details in the form displayed, automatically. The details required in the form are:- State Name (Karnataka), City Name (Bengaluru), Station Name (Peenya), Report Format (Tabular), Date From (01-Jan-2021 24:00), Date To (01-Feb-2021 24:00), Parameters (All - PM2.5, NO, N02, NOx, CO, SO2, Ozone, Benzene, Toluene, Eth-Benzene etc), Criteria (1 hour - intervals between the readings)



CCR

app.cpcbccr.com/ccr/#/caaqm-dashboard-all/caaqm-view-data-report/%2522%257B%255C%2522state%255C%2522%253A%255C%2522Karnataka%255C%2522%252C%255C%...

Chrome is being controlled by automated test software.

Central Control Room for Air Quality Management - All India

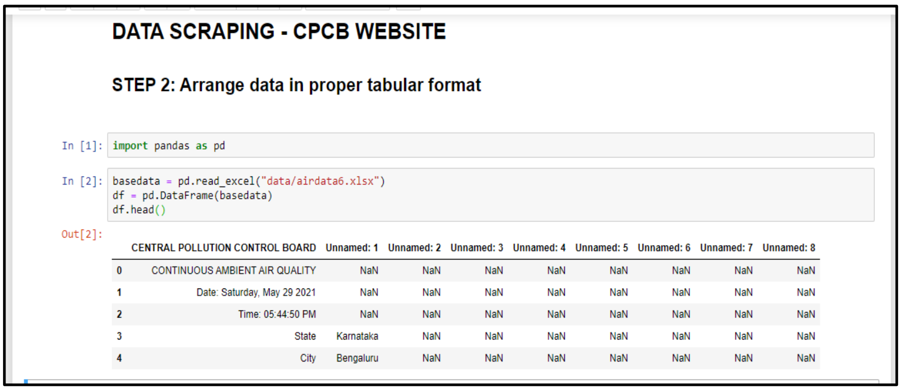
Peenya, Bengaluru - CPCB 01-Jun-2021 09:15

S.No	From Date	To Date	PM2.5 (ug/m3)	PM10 (ug/m3)	NO (ug/m3)	NO2 (ug/m3)	NOx (ppb)	NH3 (ug/m3)	SO2 (ug/m3)	CO (mg/m3)	Ozone (ug/m3)	Benzene (ug/m3)	Toluene (ug/m3)	Eth-Benzene (ug/m3)	Temp (degree C)	RH (%)	BP (mmHg)
1	01-Jan-2021 - 00:00	01-Jan-2021 - 01:00	41.2	83.18	8.02	18.75	26.74	3.82	3.64	0.92	74.25				22.47	87.8	932.33
2	01-Jan-2021 - 01:00	01-Jan-2021 - 02:00	41.47	86.65	7.74	18.87	26.58	3.72	3.21	0.87	75.67				22.3	88.48	932.57
3	01-Jan-2021	01-Jan-2021	39.24	91.83	8.6	19.77	28.35	3.36	2.53	0.77	74.85				22.15	87.42	932.3

NOTE:The data available at the portal is provided by different agenc

Figure 3: Dataset loaded for the parameters filled.

After submitting the form, the website loads the appropriate dataset, as shown in Figure 3. The Selenium webdriver automatically downloads the dataset. This is repeated to download datasets of different time periods, which are finally appended together to get a large dataset.



DATA SCRAPING - CPCB WEBSITE

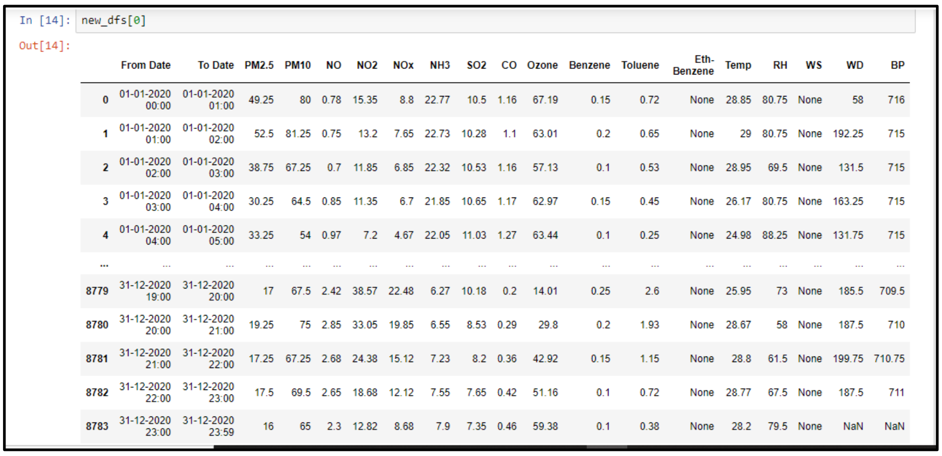
STEP 2: Arrange data in proper tabular format

```
In [1]: import pandas as pd

In [2]: basedata = pd.read_excel("data/airdata6.xlsx")
        df = pd.DataFrame(basedata)
        df.head()

Out[2]:
```

	CENTRAL POLLUTION CONTROL BOARD	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
0	CONTINUOUS AMBIENT AIR QUALITY	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Date: Saturday, May 29 2021	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	Time: 05:44:50 PM	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	State	Karnataka	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	City	Bengaluru	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 4: Dataset before arranging in tabular format

The data, though downloaded in tabular format, is not arranged properly, with few columns cut and appended at the end of all the rows i.e., the data is cut vertically and appended horizontally. It is seen in Figure 4. This is not efficient, since preprocessing steps which require legitimate rows and columns.

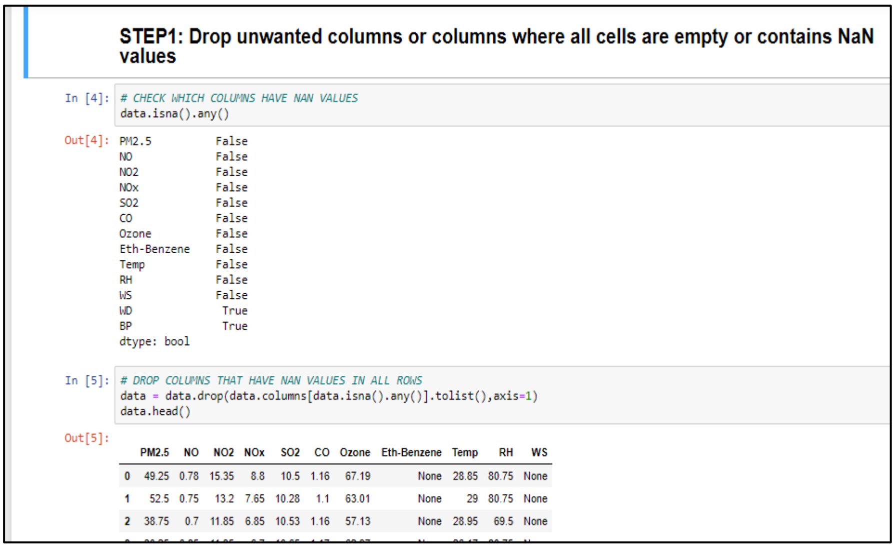


```
In [14]: new_dfs[0]

Out[14]:
```

	From Date	To Date	PM2.5	PM10	NO	NO2	NOx	NH3	SO2	CO	Ozone	Benzene	Toluene	Eth-Benzene	Temp	RH	WS	WD	BP
0	01-01-2020 00:00	01-01-2020 01:00	49.25	80	0.78	15.35	8.8	22.77	10.5	1.16	67.19	0.15	0.72	None	28.85	80.75	None	58	716
1	01-01-2020 01:00	01-01-2020 02:00	52.5	81.25	0.75	13.2	7.65	22.73	10.28	1.1	63.01	0.2	0.65	None	29	80.75	None	192.25	715
2	01-01-2020 02:00	01-01-2020 03:00	38.75	67.25	0.7	11.85	6.85	22.32	10.53	1.16	57.13	0.1	0.53	None	28.95	69.5	None	131.5	715
3	01-01-2020 03:00	01-01-2020 04:00	30.25	64.5	0.85	11.35	6.7	21.85	10.65	1.17	62.97	0.15	0.45	None	26.17	80.75	None	163.25	715
4	01-01-2020 04:00	01-01-2020 05:00	33.25	54	0.97	7.2	4.67	22.05	11.03	1.27	63.44	0.1	0.25	None	24.98	88.25	None	131.75	715
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8779	31-12-2020 19:00	31-12-2020 20:00	17	67.5	2.42	38.57	22.48	6.27	10.18	0.2	14.01	0.25	2.6	None	25.95	73	None	185.5	709.5
8780	31-12-2020 20:00	31-12-2020 21:00	19.25	75	2.85	33.05	19.85	6.55	8.53	0.29	29.8	0.2	1.93	None	28.67	58	None	187.5	710
8781	31-12-2020 21:00	31-12-2020 22:00	17.25	67.25	2.68	24.38	15.12	7.23	8.2	0.36	42.92	0.15	1.15	None	28.8	61.5	None	199.75	710.75
8782	31-12-2020 22:00	31-12-2020 23:00	17.5	69.5	2.65	18.68	12.12	7.55	7.65	0.42	51.16	0.1	0.72	None	28.77	67.5	None	187.5	711
8783	31-12-2020 23:00	31-12-2020 23:59	16	65	2.3	12.82	8.68	7.9	7.35	0.46	59.38	0.1	0.38	None	28.2	79.5	None	NaN	NaN

Figure 5: Dataset after arranging in tabular format

The data is processed and arranged in such a way that all the columns are in a single line, as shown in Figure 5. This completes Data Scraping/ Collection. The dataset was cut wherever column names were found and were appended vertically to the column axis. All the metadata about the dataset were dropped.



STEP1: Drop unwanted columns or columns where all cells are empty or contains NaN values

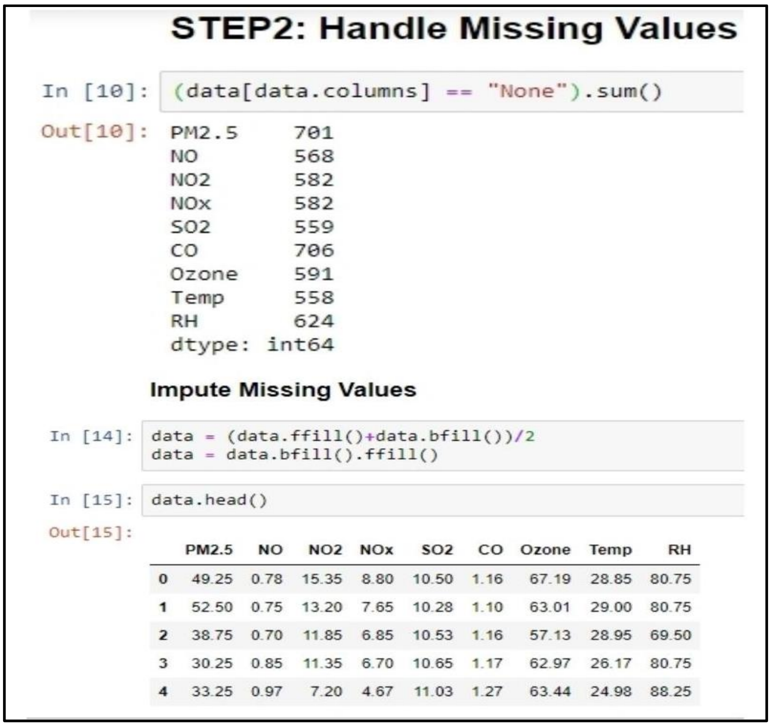
```
In [4]:  # CHECK WHICH COLUMNS HAVE NAN VALUES
         data.isna().any()

Out[4]:  PM2.5          False
         NO             False
         NO2            False
         NOx            False
         SO2            False
         CO             False
         Ozone          False
         Eth-Benzene    False
         Temp           False
         RH             False
         WS             False
         WD              True
         BP              True
         dtype: bool

In [5]:  # DROP COLUMNS THAT HAVE NAN VALUES IN ALL ROWS
         data = data.drop(data.columns[data.isna().any()].tolist(),axis=1)
         data.head()

Out[5]:
```

	PM2.5	NO	NO2	NOx	SO2	CO	Ozone	Eth-Benzene	Temp	RH	WS
0	49.25	0.78	15.35	8.8	10.5	1.16	67.19	None	28.85	80.75	None
1	52.5	0.75	13.2	7.65	10.28	1.1	63.01	None	29	80.75	None
2	38.75	0.7	11.85	6.85	10.53	1.16	57.13	None	28.95	69.5	None

Figure 6: Dropping unwanted columns

Data Preprocessing Step 1: Unwanted columns in Figure 6 mean:

a. Where the majority of the rows or all the rows have NaN (Not a Number) values.
b. Which were not present in the majority of other datasets downloaded.
c. Which are not really necessary for air quality prediction (like From and To Date).



STEP2: Handle Missing Values

```
In [10]:  (data[data.columns] == "None").sum()

Out[10]:  PM2.5     701
          NO        568
          NO2       582
          NOx       582
          SO2       559
          CO        706
          Ozone     591
          Temp      558
          RH        624
          dtype: int64
```

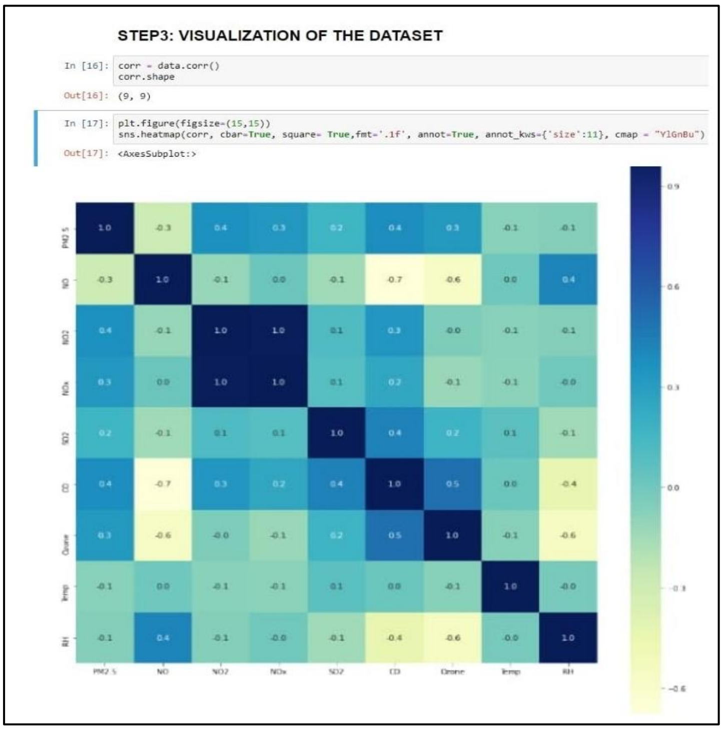
Impute Missing Values

```
In [14]:  data = (data.ffill()+data.bfill())/2
          data = data.bfill().ffill()

In [15]:  data.head()

Out[15]:
```

	PM2.5	NO	NO2	NOx	SO2	CO	Ozone	Temp	RH
0	49.25	0.78	15.35	8.80	10.50	1.16	67.19	28.85	80.75
1	52.50	0.75	13.20	7.65	10.28	1.10	63.01	29.00	80.75
2	38.75	0.70	11.85	6.85	10.53	1.16	57.13	28.95	69.50
3	30.25	0.85	11.35	6.70	10.65	1.17	62.97	26.17	80.75
4	33.25	0.97	7.20	4.67	11.03	1.27	63.44	24.98	88.25

Figure 7: Handling missing values

Data Preprocessing Step 2: Missing values are handled by imputing them using the mean-before-after method. First, all the values are converted to numeric values and Empty or 'None' values are set to numpy.nan. Then they are imputed This is a necessary step since all the further preprocessing and Model building require data in numeric non-empty format, as shown in Figure 7.



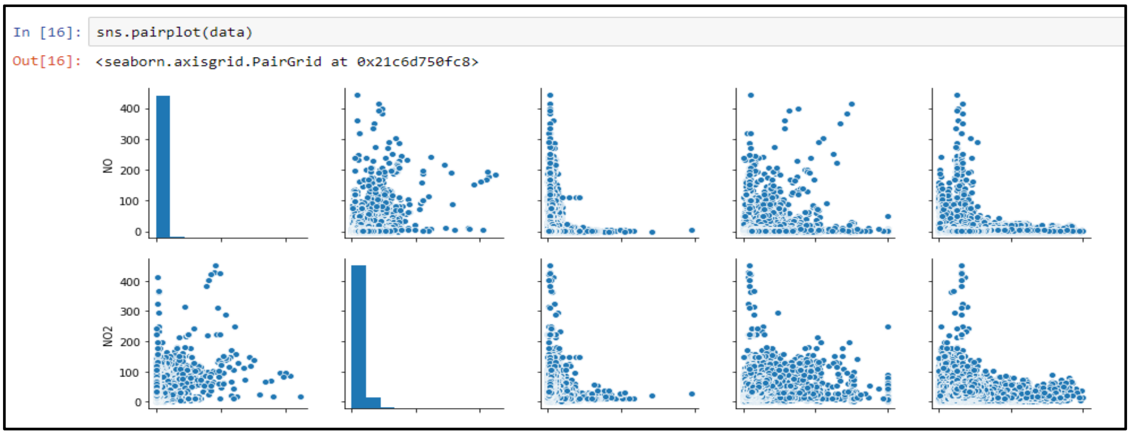
STEP3: VISUALIZATION OF THE DATASET

```
In [16]: corr = data.corr()
         corr.shape
Out[16]: (9, 9)

In [17]: plt.figure(figsize=(15,15))
         sns.heatmap(corr, cbar=True, square= True,fmt='.1f', annot=True, annot_kws={'size':11}, cmap = "YlGnBu")
Out[17]: <AxesSubplot:>
```

PM2.5 1.0 -0.3 0.4 0.3 0.2 0.4 0.3 -0.1 -0.1
NO -0.3 1.0 -0.1 0.0 -0.1 -0.7 -0.6 0.0 0.4
NO2 0.4 -0.1 1.0 1.0 0.1 0.3 -0.0 -0.1 -0.1
NOx 0.3 0.0 1.0 1.0 0.1 0.2 -0.1 -0.1 -0.0
SO2 0.2 -0.1 0.1 0.1 1.0 0.4 0.2 0.1 -0.1
CO 0.4 -0.7 0.3 0.2 0.4 1.0 0.5 0.0 -0.4
Ozone 0.3 -0.6 -0.0 -0.1 0.2 0.5 1.0 -0.1 -0.6
Temp -0.1 0.0 -0.1 -0.1 0.1 0.0 -0.1 1.0 -0.0
RH -0.1 0.4 -0.1 -0.0 -0.1 -0.4 -0.6 -0.0 1.0
PM2.5
NO
NO2
NOx
SO2
CO
Ozone
Temp
RH
0.9
0.6
0.3
0.0
-0.3
-0.6

Figure 8: Visualisation of the data using heatmap.

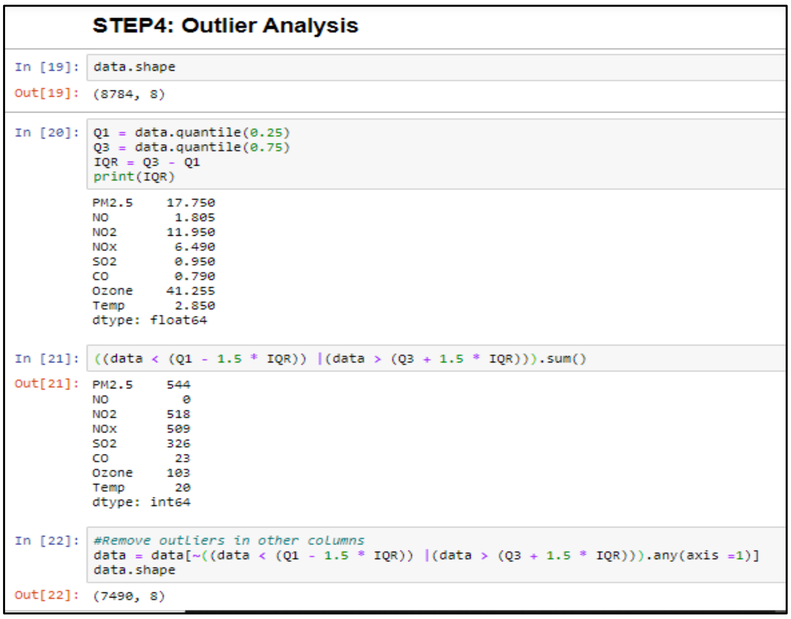
Data Preprocessing Step 3a: The vital tool in understanding big data is visualisation. The heatmap shows the correlation between any two features of the dataset. If the features are highly correlated, then the information provided by them is considered redundant. Hence, only one of the highly correlated features is retained. Like in Fig 8, we can see that NO2 and NOx are highly correlated. Hence, NOx is dropped.



```
In [16]: sns.pairplot(data)
Out[16]: <seaborn.axisgrid.PairGrid at 0x21c6d750fc8>
```

Figure 9: Pairplot of the features of the dataset.

Data Preprocessing step 3b: Pairplots are pairwise relationships in a dataset. Each feature is plotted against all the features of the dataset to show the bivariate distribution of the dataset. It is clear from Figure 9, the distribution of data is not uniform or is uneven.

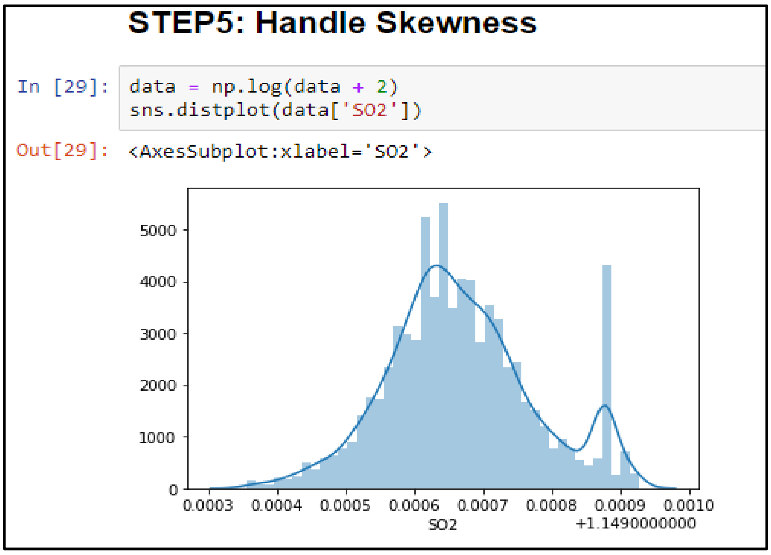


STEP4: Outlier Analysis

```
In [19]: data.shape
Out[19]: (8784, 8)

In [20]: Q1 = data.quantile(0.25)
         Q3 = data.quantile(0.75)
         IQR = Q3 - Q1
         print(IQR)

         PM2.5     17.750
         NO         1.805
         NO2       11.950
         NOx        6.490
         SO2        0.950
         CO         0.790
         Ozone     41.255
         Temp       2.850
         dtype: float64

In [21]: ((data < (Q1 - 1.5 * IQR)) |(data > (Q3 + 1.5 * IQR))).sum()
Out[21]: PM2.5     544
         NO          0
         NO2       518
         NOx       509
         SO2       326
         CO         23
         Ozone     103
         Temp       20
         dtype: int64

In [22]: #Remove outliers in other columns
         data = data[~((data < (Q1 - 1.5 * IQR)) |(data > (Q3 + 1.5 * IQR))).any(axis =1)]
         data.shape
Out[22]: (7490, 8)
```

Figure 10: Outlier analysis using IQR (InterQuartile Range) Method.

Data Preprocessing step 4: Since outliers distort or bias the model towards particular direction, it's necessary to identify and drop them. As seen in Fig 11 there are 8784 rows of data before outlier analysis. The quantiles for each feature are displayed. All rows that don’t meet the quantile specification of all columns are dropped. Hence, we finally have 7490 rows of data i.e.1294 rows of outliers are dropped.

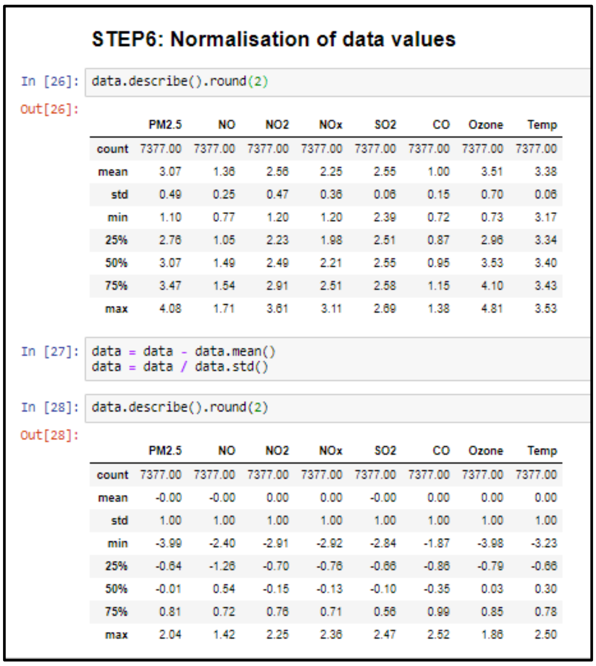


STEP5: Handle Skewness

```
In [29]: data = np.log(data + 2)
         sns.distplot(data['SO2'])
Out[29]: <AxesSubplot:xlabel='SO2'>
```

Figure 11: Handling the skewness of the data

Data Preprocessing Step 5: Due to Kmeans assumptions about distribution of features i.e. each attribute is spherical, it is necessary to have normal distribution of attributes of the dataset. This is done using the numpy log function. Though the graph shown in Figure 9.1.10 does not depict a perfect normal distribution, it is better than before.



STEP6: Normalisation of data values

```
In [26]: data.describe().round(2)
Out[26]:
```

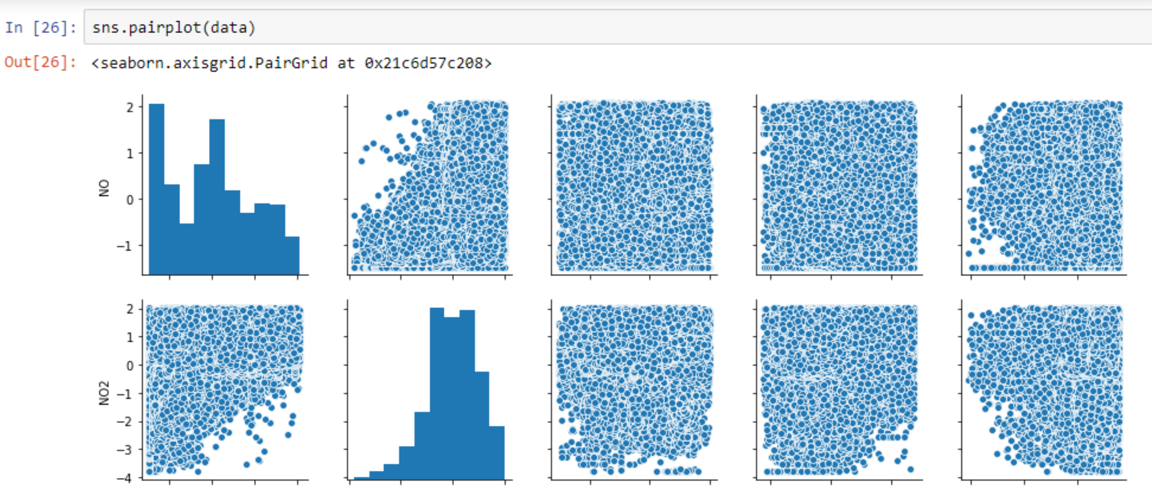
	PM2.5	NO	NO2	NOx	SO2	CO	Ozone	Temp
count	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00
mean	3.07	1.36	2.56	2.25	2.55	1.00	3.51	3.38
std	0.49	0.25	0.47	0.36	0.06	0.15	0.70	0.06
min	1.10	0.77	1.20	1.20	2.39	0.72	0.73	3.17
25%	2.76	1.05	2.23	1.98	2.51	0.87	2.96	3.34
50%	3.07	1.49	2.49	2.21	2.55	0.95	3.53	3.40
75%	3.47	1.54	2.91	2.51	2.58	1.15	4.10	3.43
max	4.08	1.71	3.61	3.11	2.69	1.38	4.81	3.53

```
In [27]: data = data - data.mean()
         data = data / data.std()

In [28]: data.describe().round(2)
Out[28]:
```

	PM2.5	NO	NO2	NOx	SO2	CO	Ozone	Temp
count	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00	7377.00
mean	-0.00	-0.00	0.00	0.00	-0.00	0.00	0.00	0.00
std	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
min	-3.99	-2.40	-2.91	-2.92	-2.84	-1.87	-3.98	-3.23
25%	-0.64	-1.26	-0.70	-0.76	-0.66	-0.86	-0.79	-0.66
50%	-0.01	0.54	-0.15	-0.13	-0.10	-0.35	0.03	0.30
75%	0.81	0.72	0.76	0.71	0.56	0.99	0.85	0.78
max	2.04	1.42	2.25	2.36	2.47	2.52	1.86	2.50

Figure 12:Normalization of data

Data Preprocessing step 6: To satisfy the assumption of K-means that all attributes have the same variances, normalisation of data is necessary. This is done using mean and standard deviation values of the dataset and is shown in Figure 12.



```
In [26]: sns.pairplot(data)
Out[26]: <seaborn.axisgrid.PairGrid at 0x21c6d57c208>
```

Figure 13: Pairplot of the features of the dataset

The pairplot in fig 13 is plotted after applying all the preprocessing steps. It is clear that the distribution of data is more uniform or has even distribution.

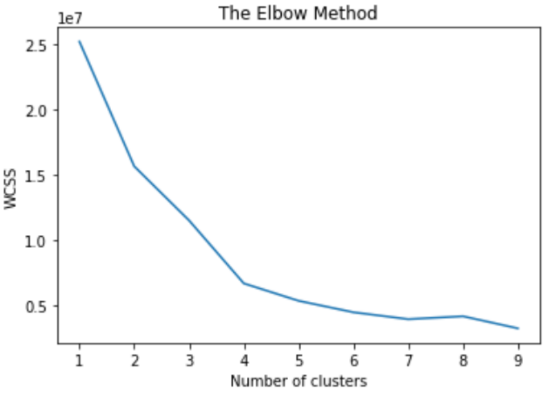
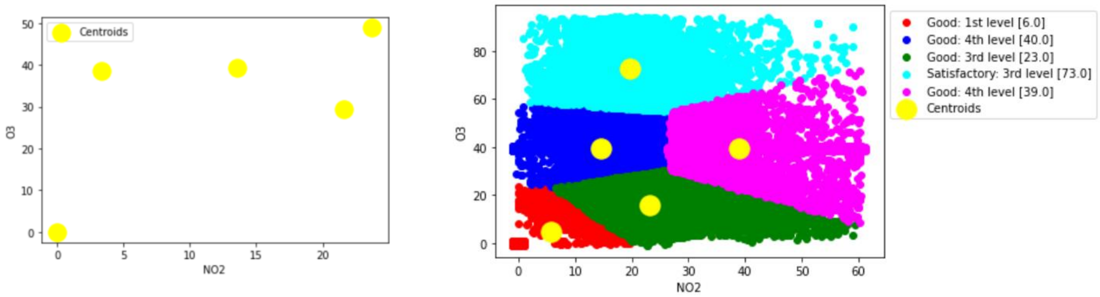
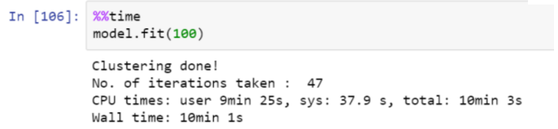


Figure 14: The Elbow Method: Number of Clusters vs WCSS .

This plot in Figure 14 is used to find the ideal number of centroids. For different numbers of centroids, K-means++ initialization and Regular clustering is applied and WCSS is calculated. The Elbow rests at 5.

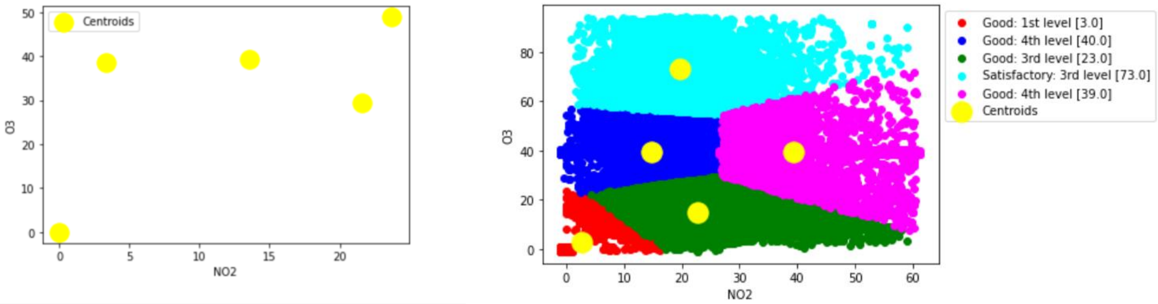


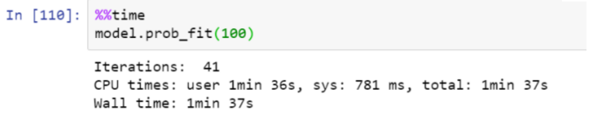


```
In [106]: %%time
          model.fit(100)

          Clustering done!
          No. of iterations taken :  47
          CPU times: user 9min 25s, sys: 37.9 s, total: 10min 3s
          Wall time: 10min 1s
```

Figure 15: Plot of initial centroids (Random Initialization) and Plot of clusters obtained from Regular Clustering

Figure 15 shows the plot of initial centroids obtained through Random Initialization. Here, the dataset used is of size 1 lakh. After applying Regular Clustering, cluster convergence happens at 10min 1s. Figure 9.1.15 shows the clusters obtained along with its respective AQI categories and sublevels. The AQI categories obtained here are Good (with AQI range 0-50) and Satisfactory (with AQI range 51-100). Both the categories are further divided into sublevels like 1st level, 2nd level and so on with range 10.

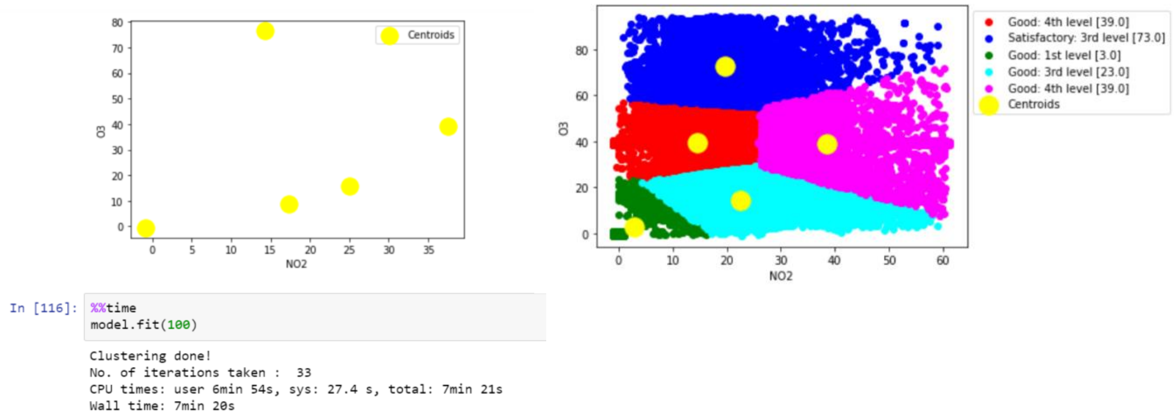




```
In [110]: %%time
          model.prob_fit(100)

          Iterations:  41
          CPU times: user 1min 36s, sys: 781 ms, total: 1min 37s
          Wall time: 1min 37s
```

Figure 16: Plot of initial centroids (Random Initialization) and Plot of clusters obtained from Probability-Based Clustering

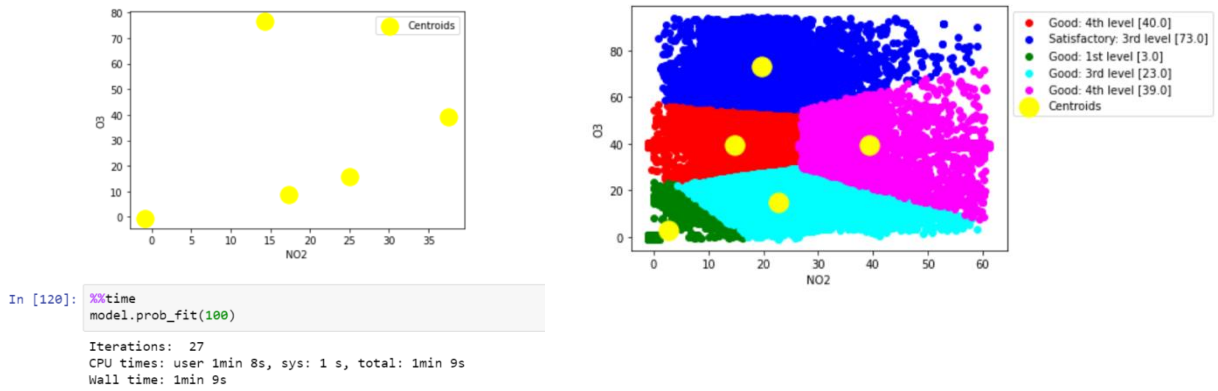
Figure 16 shows the plot of the same initial centroids obtained through Random Initialization in Figure 14. After applying Probability-Based Clustering, cluster convergence happens at 1min 37s which is faster than Regular Clustering in Figure 9.1.14. Figure 9.1.17 shows the clusters obtained along with its respective AQI categories and sublevels which is very similar to the one in Figure 15.



```
In [116]: %%time
          model.fit(100)

          Clustering done!
          No. of iterations taken :  33
          CPU times: user 6min 54s, sys: 27.4 s, total: 7min 21s
          Wall time: 7min 20s
```

Figure 17: Plot of initial centroids (K-means++ Initialization) and Plot of clusters obtained from Regular Clustering

Figure 17 shows the plot of initial centroids obtained through K-means++ Initialization. It can be seen that there is not much improvement in centroid spacing. After applying Regular Clustering, cluster convergence happens at 7min 20s. Andalso shows the clusters obtained along with its respective AQI categories and sublevels i.e., Good and Satisfactory with sublevels 1st level, 2nd level and so on.



```
In [120]: %%time
          model.prob_fit(100)

          Iterations:  27
          CPU times: user 1min 8s, sys: 1 s, total: 1min 9s
          Wall time: 1min 9s
```

Figure 18: Plot of initial centroids (K-means ++ Initialization) and Plot of clusters obtained from Probability-Based Clustering

Figure 18 shows the plot of the same initial centroids obtained through K-means++ Initialization in Figure. After applying Probability-Based Clustering, cluster convergence happens at 1min 9s which is faster than Regular Clustering in Figure 16. Figure 18 shows the clusters obtained along with its respective AQI categories and sublevels which is very similar to the one in Figure 17 and also similar to the ones in Figure 16 and 17. Hence, it is seen that the quality of clustering has not improved much.

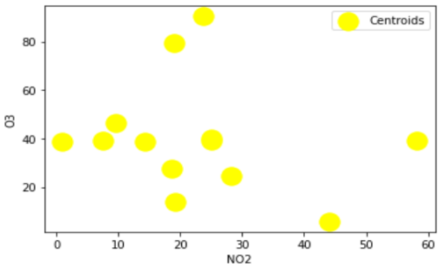
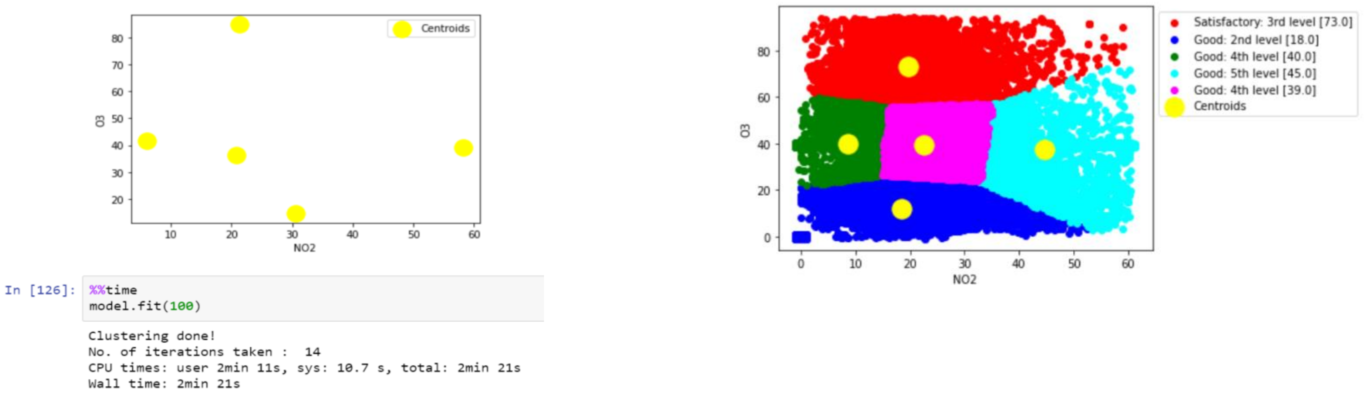


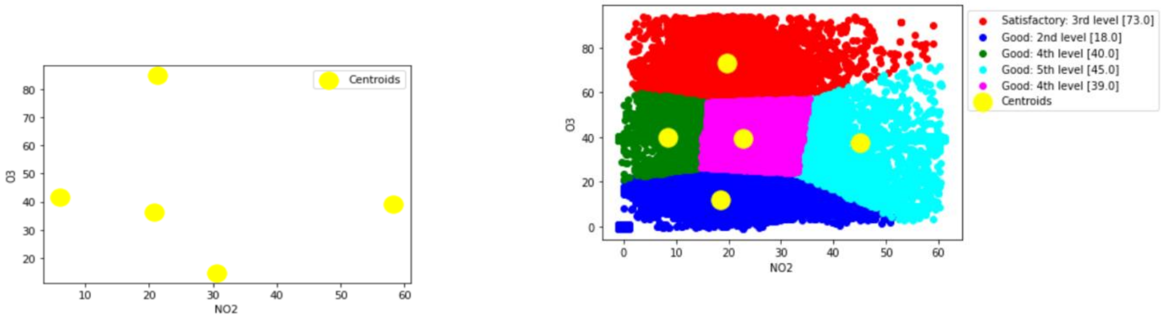
Figure 19: Plot of oversampled centroids (K-means++ Advanced Initialisation)

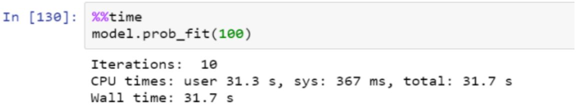


```
In [126]: %%time
          model.fit(100)

          Clustering done!
          No. of iterations taken :  14
          CPU times: user 2min 11s, sys: 10.7 s, total: 2min 21s
          Wall time: 2min 21s
```

Figure 20: Plot of initial centroids (K-means++ Advanced Initialization) and Plot of clusters obtained from Regular Clustering

Figure 20 shows the plot of the oversampled centroid candidates which is a crucial part of the K-means++ advanced initialization algorithm. After applying Weighted Clustering, initial centroids are obtained as shown in Figure 20. It can be seen that from K-means++ Advanced Initialization, centroids are much spaced out. After applying Regular Clustering, cluster convergence happens at 2min 21s faster than both Random and K-means++ Initialization with Regular Clustering. Figure 19 shows the clusters obtained along with its respective AQI categories and sublevels i.e., Good and Satisfactory with sublevels 1st level, 2nd level and so on.

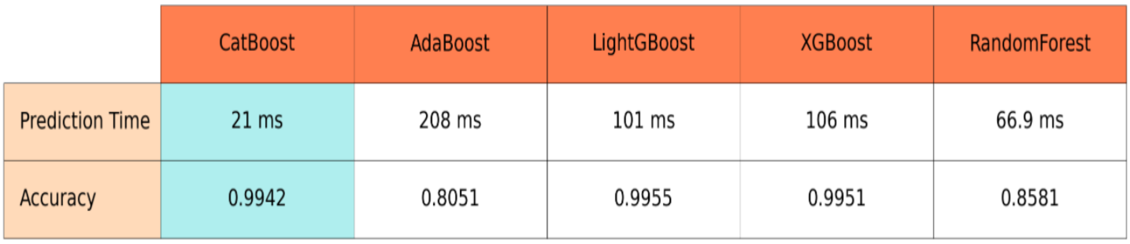




```
In [130]: %%time
          model.prob_fit(100)

          Iterations:  10
          CPU times: user 31.3 s, sys: 367 ms, total: 31.7 s
          Wall time: 31.7 s
```

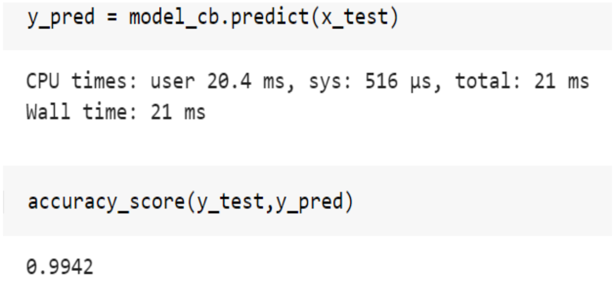
Figure 21: Plot of initial centroids (K-means ++ Advanced Initialization) and Plot of clusters obtained from Probability-Based Clustering

Figure 21 shows the plot of the same initial centroids obtained through K-means++ Advanced Initialization. After applying Probability-Based Clustering, cluster convergence happens at 31.7s which is the fastest. Figure 21 shows the clusters obtained along with its respective AQI categories and sublevels which is very similar to the one in Figure 19. The clusters obtained are much more distinct with better quality of clustering.



	CatBoost	AdaBoost	LightGBoost	XGBoost	RandomForest
Prediction Time	21 ms	208 ms	101 ms	106 ms	66.9 ms
Accuracy	0.9942	0.8051	0.9955	0.9951	0.8581

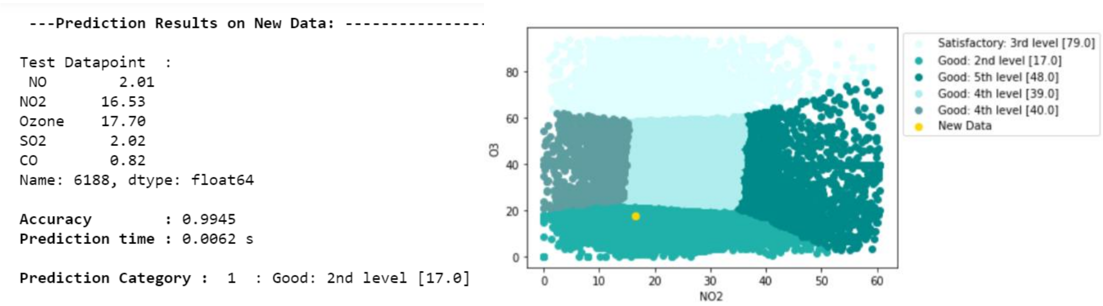
Table 1: Tabular Comparison of Performance of prediction algorithms under criteria of accuracy + prediction time.



```
y_pred = model_cb.predict(x_test)

CPU times: user 20.4 ms, sys: 516 µs, total: 21 ms
Wall time: 21 ms

accuracy_score(y_test,y_pred)

0.9942
```

Figure 22: Catboost performance snapshot in criteria of prediction time and accuracy.

Five ensemble models were implemented on our chosen use case of Bengaluru air data namely, Catboost, Adaboost, Light GBM, XG Boost and Random Forest. The input to the prediction algorithms were solely our patterns found by the previously discussed clustering algorithm and the target variable was the AQI category of cluster the data point fell into. The ideal algorithm needs to minimize prediction time to as small as possible while keeping good accuracy. Table 1 summarizes that Catboost attained 5x to 10x speedup over other competitors maintaining accuracy of 0.9942. Therefore, Catboost was used finally as the prediction algorithm in our work. Figure 22 is the snapshot of the prediction time and accuracy obtained for Catboost.



```
 ---Prediction Results on New Data: ---------------

Test Datapoint  :
 NO          2.01
NO2        16.53
Ozone      17.70
SO2          2.02
CO           0.82
Name: 6188, dtype: float64

Accuracy        : 0.9945
Prediction time : 0.0062 s

Prediction Category :  1  : Good: 2nd level [17.0]
```

Figure 23: Predicting AQI Category for New Datapoint and Visualizing the prediction

After the most optimal prediction algorithm is chosen-Catboost, it was subsequently trained and applied to an incoming testpoint. The results shown in Figure 9.1.29 and 9.1.30 bring out the prediction category obtained for the test datapoint: [NO-2.01, NO2-16.53, O3-17.70, SO2-2.02, CO-0.82] namely: AQI category-Good, 2nd level with AQI value 17.00 and show how the new datapoint lies with respect to the previous patterns discovered. This work has thus efficiently and accurately clustered historical air data and has also been extended to tackle future air data points in a realistic time as well.

Clustering Algorithms	**Scores**	**Initialization Algorithms**		
		Random	K-means ++	K-means ++ Advanced
Regular	Silhouette	0.508	0.512	0.443
	Davies Bouldin	0.759	0.688	0.699
	Calinski harabasz	79807.817	79893.269	96544.518
Probability -based	Silhouette	0.519	0.519	0.444
	Davies Bouldin	0.680	0.679	0.698
	Calinski harabasz	79902.799	79902.629	96533.693

Table 2: Tabular Comparison of Accuracy in terms of Performance Scores for Initialization + Clustering Algorithms for 1 Lakh dataset

Table 2 shows the comparison of the three Initialization algorithms along with the two Clustering algorithms on the 1 lakh dataset with respect to the quality of clusters obtained. Three scores are used, namely, Silhouette, Davies Bouldin and Calinski Harabasz scores which are explained in Section 7.5.4. From the table, among the Initialization algorithms, K-means ++ Advanced has better Davies and Calinski scores, i.e., two out of the three parameters and the Calinski score has improved by a large margin compared to the pre-existing methods of Random and K-means++. Among the Clustering algorithms, Probability-based has either similar or better scores compared to Regular. Hence, K-means ++ Advanced initialization algorithm improves the quality of clustering and Probability-based Clustering maintains the accuracy.

Dataset Size	**Clustering Algorithms**	**Initialization Algorithms**		
		Random	K-means ++	K-means ++ Advanced
30000	Regular	37.8 s	35 s	21.2 s
	Probability-based	21.9 s	16.2 s	13.2 s
50000	Regular	49.6 s	57.7 s	35.8 s
	Probability-based	38 s	18.2 s	16.7 s
70000	Regular	5 min 17 s	3 min 54 s	1 min 50 s
	Probability-based	1 min 23 s	47.9 s	36.6 s
100000	Regular	10 min 1 s	7 min 20 s	2 min 21 s
	Probability-based	1 min 37 s	1 min 9 s	**31.7 s**

Table 3: Tabular Comparison of Execution times of Initialization + Clustering Algorithms with varying dataset size

Table 3 compares the execution times of the three Initialization algorithms along with the two Clustering algorithms where the dataset sizes of 30000, 50000, 70000 and 1 lakh are considered. The highlighted cells show that for each dataset size, K-means++ Advanced initialization and Probability-based Clustering took the minimum time among the Initialization and Clustering algorithms respectively. Taking both K-means ++ Advanced

Initialization and Probability-based Clustering together, the fastest time can be achieved with upto 19x speedup.

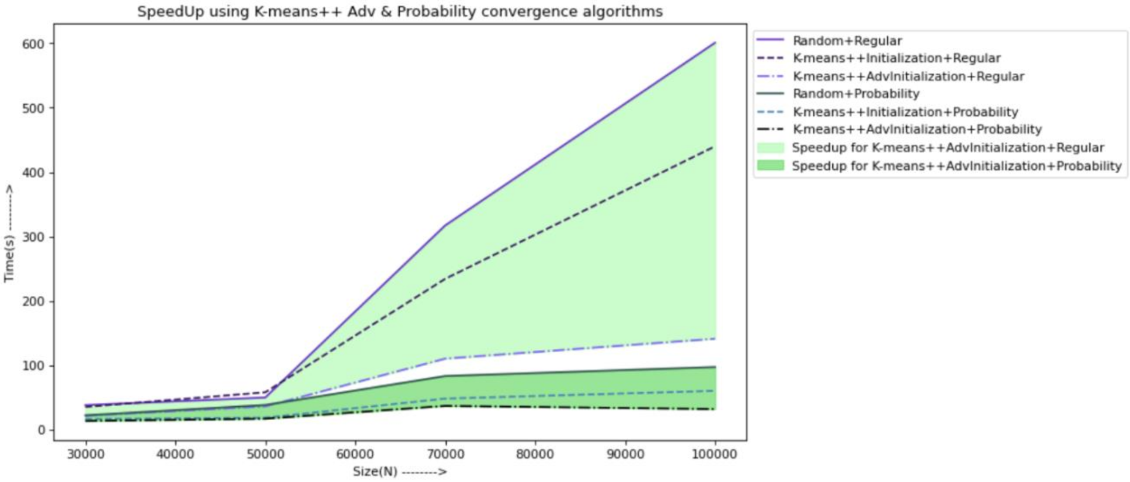


Figure 24: Size Of Data(N) vs Time in seconds for the various Initialization and Clustering algorithms

To better grasp the improvement made by the proposed methodology, Figure 24 shows the speedup obtained by the various algorithms we have studied and tested in this work. The X-axis is the size of the dataset as a function of N and the Y-axis is time in seconds. The top 3 lines are the plottings for all the 3 initialization algorithms studied (Random, K-means++ and K-means++ Advanced respectively) along with the regular convergence algorithm and the bottom 3 are the plottings for the initialization algorithms and probability based convergence algorithm. From observing the top 3 lines we can see that only using K-means++ Advanced provided a speedup of 4x (shown by the light green shaded region). The bottom 3 lines show that using only the probability based convergence algorithm has provided a significant speedup over just the regular convergence algorithm. Using K-means++ advanced along with probability based convergence gave the lowest line which testifies to a speedup of 19x obtained over the Random initialization along with the regular convergence which is the most typically used combination. Hence this work has achieved a much better efficiency and better accuracy.

Conclusion

Our work aims in predicting the air quality index (AQI) by clustering the air pollutants. Air pollution dataset has to be preprocessed as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn. It is important to handle missing values and outliers before feeding it into our Kmeans clustering algorithm. Finding the ideal number of centroids for clustering is crucial, so we apply the elbow method to do the same. Selecting the initial centroids is critical for quality clustering. Random initialization can take many iterations to converge and can lead to poor clustering because of incorrect centroids. To overcome this, we use K-means++ initialization. This algorithm ensures a smarter initialization of the centroids so that distinct clusters are formed. The issue found with Kmeans ++ is that it requires K passes over the data and therefore initialization takes time. To solve this problem, we use K-means ++ advanced initialization. This algorithm doesn’t require K passes, it samples more than one data point in one iteration thereby reducing the total number of iterations. The oversampled data points, which are centroid candidates, are assigned weights depending on how many data points are close to it and then are clustered using Weighted K-means. The resultant initial centroids converge faster and form better quality clusters.

Once the initial centroids are selected the remaining data points are clustered using the regular clustering algorithm. This method, although simple, does not age well with increasing dataset size. As the size increases the execution time also drastically increases. Hence we propose a probability based convergence algorithm that smartly selects a portion of the data for the step of recomputation of centroids on the basis of a probability metric and on the basis of the value of the cutoff probability-pbound and the raceToCenter and Stiffness hyperparameters. This way of formation of clusters reduces execution time and thereby obtains a significant speedup.The combination of the K-means++ Advanced Initialization algorithm and the probability based convergence are shown to give the lowest time+good accuracy, which was what we were targeting with this work. Further the work is also extended to identify the AQI category of the future data points on the basis of the patterns discovered by the clustering algorithm. Hence, To conclude this work has successfully delivered its major objective to improve the K-means initialization and clustering algorithm.

REFERENCES

1. Sanjay Chakraborty, , and N. K. Nagwani. "Performance Evaluation of Incremental K-means Clustering Algorithm." (2014).
2. Sujatha, S. and A. Sona. “New Fast K-Means Clustering Algorithm using Modified Centroid Selection Method.” International journal of engineering research and technology 2 (2013): n. Pag.
3. Shou-Qiang Wang and Da-Ming Zhu, "Research on selecting initial points for k-means clustering," 2008 International Conference on Machine Learning and Cybernetics, 2008, pp. 2673-2677, doi: 10.1109/ICMLC.2008.4620860.
4. G. R. Kingsy, R. Manimegalai, D. M. S. Geetha, S. Rajathi, K. Usha and B. N. Raabiathul, "Air pollution analysis using enhanced K-Means clustering algorithm for real time sensor data," 2016 IEEE Region 10 Conference (TENCON), 2016, pp. 1945-1949, doi: 10.1109/TENCON.2016.7848362.
5. Shahriar, Shihab A.; Kayes, Imrul; Hasan, Kamrul; Hasan, Mahadi; Islam, Rashik; Awang, Norrimi R.; Hamzah, Zulhazman; Rak, Aweng E.; Salam, Mohammed A. 2021. "Potential of ARIMA-ANN, ARIMA-SVM, DT and CatBoost for Atmospheric PM2.5 Forecasting in Bangladesh" Atmosphere 12, no. 1: 100. https://doi.org/10.3390/atmos1201010
6. Haraty, Ramzi A.; Dimishkieh, Mohamad; Masud, Mehedi (2015). An Enhanced k - Means Clustering Algorithm for Pattern Discovery in Healthcare Data. International

Journal of Distributed Sensor Networks, 2015(), 1–11. doi:10.1155/2015/615740

7. Verma, Vikas; Bhardwaj, Shaweta; Singh, Harjit (2016). A Hybrid K-Mean Clustering Algorithm for Prediction Analysis. Indian Journal of Science and Technology, 9(28), –. doi:10.17485/ijst/2016/v9i28/98392

8. Sadiq, Abderrahmane & Fazziki, A. & Ouarzazi, J. & Sadgal, Mohammed. (2017). Air quality analysis based on MapReduce and K-Means: A Decision Making System. International Journal of Advances in Soft Computing and its Applications. 9. 140-153.

9. Chakraborty, Sanjay & Nagwani, Naresh & Dey, Lopamudra. (2011). Performance Comparison of Incremental K-means and Incremental DBSCAN Algorithms. International Journal of Computer Applications (0975 – 8887). 27. 975-8887.

10. Gómez-Losada Á, Pires JCM, Pino-Mejías R. Modelling background air pollution exposure in urban environments: Implications for epidemiological research. Environ Model Softw. 2018 Aug;106:13-21. doi: 10.1016/j.envsoft.2018.02.011. PMID: 30078988; PMCID: PMC6018063.

11. Rauf, Azhar & Sheeba, & Mahfooz, Saeed & Khusro, Shah & Javed, Huma. (2012). Enhanced K-Mean Clustering Algorithm to Reduce Number of Iterations and Time Complexity. Middle-East Journal of Scientific Research. 12. 959-963. 10.5829/idosi.mejsr.2012.12.7.1845.

12. Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. 2012. Scalable k-means++. Proc. VLDB Endow. 5, 7 (March 2012), 622–633. DOI:https://doi.org/10.14778/2180912.2180915

13. Moustris, Kostas & Ziomas, Ioannis & Paliatsos, Athanasios. (2010). 3-Day-Ahead Forecasting of Regional Pollution Index for the Pollutants NO2, CO, SO2, and O3 Using Artificial Neural Networks in Athens, Greece. Water, Air, & Soil Pollution. 209. 10.1007/s11270-009-0179-5.

14. Saksena, S., V. Joshi and R. S. Patil. “Determining spatial patterns in Delhi's ambient air quality data using cluster analysis.” (2002).

15. Vitolo, Claudia & Scutari, Marco & Ghalaieny, Mohamed & Tucker, Allan & Russell, Andrew. (2018). Modeling Air Pollution, Climate, and Health Data Using Bayesian Networks: A Case Study of the English Regions. Earth and Space Science. 5. 10.1002/2017EA000326.

16. Madhu, Yedla & Pathakota, Srinivasa & Srinivasa.T.M,. (2010). Enhancing K-means Clustering Algorithm with Improved Initial Center. International Journal of Computer Science and Information Technologies. 1.

17. Navjot Kaur, Jaspreet Kaur Sahiwal and Navneet Kaur. Efficient K-means clustering algorithm using ranking method in data mining. International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 3, May 2012

18. Azid, Azman & Juahir, Hafizan & Toriman, Mohd & Kamarudin, Mohd khairul amri & Mohd Saudi, Ahmad Shakir & Hasnam, Che & Aziz, Nor & Azaman, Fazureen & Latif, Mohd Talib & Zainuddin, Syahrir & Osman, Romizan & Yamin, Mohammad. (2014). Prediction of the Level of Air Pollution Using Principal Component Analysis and Artificial Neural Network Techniques: a Case Study in Malaysia. Water Air and Soil Pollution. 225. 10.1007/s11270-014-2063-1.

19. Bansal, Arpit & Sharma, Mayur & Goel, Shalini. (2017). Improved K-mean Clustering Algorithm for Prediction Analysis using Classification Technique in Data Mining. International Journal of Computer Applications. 157. 35-40. 10.5120/ijca2017912719.

20. Mauro Castelli, Fabiana Martins Clemente, Aleš Popovič, Sara Silva, Leonardo Vanneschi, "A Machine Learning Approach to Predict Air Quality in California", Complexity, vol. 2020, Article ID 8049504, 23 pages, 2020. https://doi.org/10.1155/2020/8049504

21. Jirat Boonphun, Chalat Kaisornsawad, Papis Wongchaisuwat."Machine learning algorithms for predicting air pollutants" E3S Web Conf. 120 03004 (2019). DOI: 10.1051/e3sconf/201912003004

22. Rybarczyk, Yves; Zalakeviciute, Rasa. 2018. "Machine Learning Approaches for Outdoor Air Quality Modelling: A Systematic Review" Appl. Sci. 8, no. 12: 2570. https://doi.org/10.3390/app8122570

23. Yin Zhao & Yahya Abu Hasan, 2013. "Comparison of Three Classification Algorithms for Predicting Pm2.5 in Hong Kong Rural Area," Journal of Asian Scientific Research, Asian Economic and Social Society, vol. 3(7), pages 715-728, July.

24. Liang, Yun-Chia; Maimury, Yona; Chen, Angela H.-L.; Juarez, Josue R.C. 2020. "Machine Learning-Based Prediction of Air Quality" Appl. Sci. 10, no. 24: 9151. https://doi.org/10.3390/app10249151

25. Doreswamy, Hosahalli & Harishkumar, K S & Km, Yogesh & Gad, Ibrahim. (2020). Forecasting Air Pollution Particulate Matter (PM2.5) Using Machine Learning Regression Models. Procedia Computer Science. 171. 2057-2066. 10.1016/j.procs.2020.04.221.

26. X. Miao and J. S. Heaton, "A comparison of random forest and Adaboost tree in ecosystem classification in east Mojave Desert," 2010 18th International Conference on Geoinformatics, 2010, pp. 1-6, doi: 10.1109/GEOINFORMATICS.2010.5567504.

27. Anna Veronika Dorogush and Andrey Gulin and Gleb Gusev and Nikita Kazeev and Liudmila Ostroumova Prokhorenkova and Aleksandr Vorobev (2017). Fighting biases with dynamic boosting. CoRR, abs/1706.09516.