# AI-assisted Distributed Cloud Services Framework for Enhanced Safety in Urban Smart Cities Environment

*Niveshitha Niveshitha, *Student Members, IEEE;*
*Fathi Amsaad, †Ahmed Sherif, and ‡Noor Zaman Jhanjhi, *Senior Members, IEEE*

***Abstract*—Smart cities have emerged to tackle life critical challenges that can thwart the overwhelming urbanization process, such as expensive health care, increasing energy demand, traffic jams, and environmental pollution. This paper proposes efficient and high-quality cloud-based machine-learning solutions for safe urban smart city environment. For that, supervised ML-based models, i.e., regression and classification, are developed utilizing cloud-based solutions to forecast high performance in execution time and enhanced quality of the solution in terms of the accuracy of the implemented cloud-based ML solution. To predict AQI, i.e. air quality index, ML models utilize pollutants in the air data sets. The mean absolute error, mean squared error, root means the squared error, R2 score are used to validate and test the designed models. As classification models, we perform the support vector machine and random forest algorithms, which are measured using the accuracy score and confusion matrix. Execution times and accuracy of the developed models are computed and contrasted with the times for the cloud-based versions of these models. The results show that among the regression algorithms, lasso regression has an r2 score of 80 percent, while linear regression has an r2 score of 75 percent. Furthermore, among the classification models, the random forest algorithm performs better with an accuracy of 99 percent than the support vector machine approach with 95 percent accuracy. In conclusion, our findings demonstrate that run-time is minimized when models are executed on a cloud platform compared to a desktop machine. Moreover, the accuracy of our models is maintained with reduced execution time.**

***Index Terms*—Urban Cities Smart Safety and Security, Distributed Cloud Services, Artificial Intelligence (AI), Machine Learning (ML), Random Forest (RF), Lasso Regression (LASSO), Liner Regression (LINER), Support Vector Machine (SVM).**

## I. INTRODUCTION

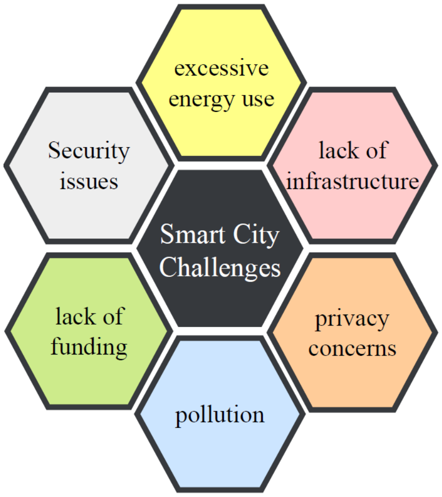

Fig. 1: Challenges in an urban smart city environment

SMART cities have several definitions and cannot be wrapped into a single phrase. The principal purpose of this approach is to enable sustainable life for the masses through digital interconnections in a more efficient and environmentally friendly urban area. To provide individuals with a good standard of living, it manages the resources and services of a city using data from sensors and electrical gadgets. Smart cities use ICT (information and communication technology) to reduce costs and resource consumption [1]. Artificial intelligence, IoT (Internet of Things), extensive data analysis, and machine learning are some techniques implemented in smart cities. Smart city applications rely on four critical features: sustainability, comfort, quality of life, urbanization, and intelligence [2]. Cities must overcome several obstacles during the design and development phases to incorporate these features. AI-assisted distributed computing developments can help to efficiently address the research challenges in urban smart cities environment listed in Figure 1.

This study aims to address the safety issues in smart cities environment focusing on smart air pollution for a safe and clean urban intelligent city environment. Air pollution is the contamination of the air with gases, biological molecules, and poisonous particles. Natural calamities like volcanic eruptions and manufactured activities, including automobile emissions and industrial by-products, are the fundamental causes of air pollution [3].

The authors would like to express their sincere gratitude for the technical and financial support provided by the Air Force Research Lab (AFRL) under the Assured and Trusted Digital Microelectronics Ecosystem (ADMETE) grant, BAA-FA8650-18-S-1201. Additionally, partial support for this work was received from the National Security Agency (NSA) funding to Wright State University in Dayton, Ohio, USA. This project adhered to CAGE Number: 4B991 and DUNS Number: 047814256.

Niveshitha Niveshitha and Fathi Amsaad, ***Corresponding Authors***, are affiliated with the Department of Computer Science and Engineering, Wright State University, 3640 Colonel Glenn Hwy, Dayton, OH 45435, USA. Emails: (ghimire.18, fathi.amsaad)@wright.edu

Ahmed Sherif is with the School of Computing Sciences and Computer Engineering, University of Southern Mississippi, Hattiesburg, Mississippi, USA (e-mail: ahmed.sherif@usm.edu

Noor Zaman Jhanjhi is associated with the School of Computer Science Faculty of Innovation and Technology, Taylor's University, 47500 Subang Jaya, Selangor, Malaysia. Email: noorzaman.jhanjhi@taylors.edu.my

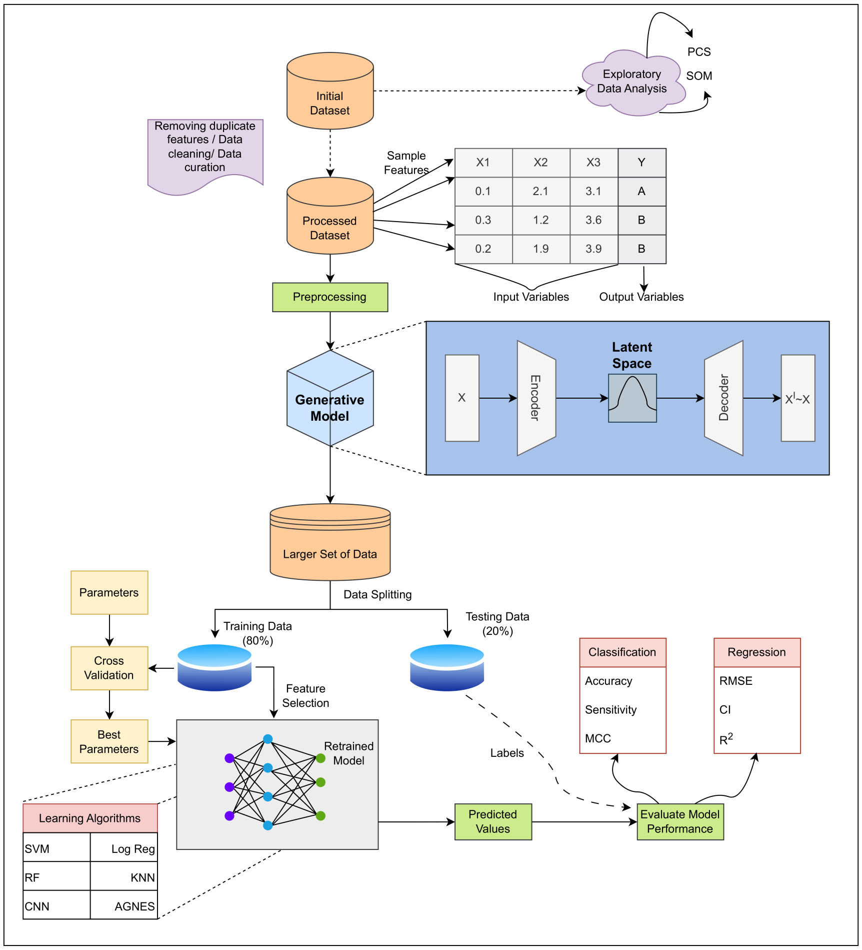

Fig. 2: Overview of the development of machine learning models

These causes provoke severe threats for urban cities, like chronic diseases in human bodies, i.e., lung cancer, skin infections, bronchitis, asthma, etc. They may also harm animals, plants, and other species [4]. Air pollutants are gases, chemicals, and other polluting elements in the atmosphere. These include nitric oxide, sulfur dioxide, toluene, ozone, nitric x-oxide, particulate matter, ammonia, xylene, nitrogen dioxide, benzene, and carbon monoxide [5]. As a result, one of the primary problems affecting intelligent urban cities is the issue of safety and pollution of the air. For that, the quality of the air index (target feature), calculated based on atmospheric pollutants, can be used to determine air pollution [6].

Smart cloud services devices including IoT devices are strategically deployed throughout urban areas to collect massive amounts of data [7]. Because analyzing and preprocessing this data takes more time, computing plays a vital role. Cities are already adopting high-performance computing (HPC) to help them achieve objectives such as resource conservation, social safety, and an overall better quality of life. Distributed computing, cloud computing, parallel processing, edge computing, and fog computing are high-performance computing (HPC) [8]. In this work, we employ cloud machine learning. Machine learning accuracy relies heavily on development robust model with big data training. Therefore, the largest the dataset, the better the ML model accuracy for the prediction, optimization, and classification applications. Cloud computing can enhance the execution time and the quality of the solution of the developed ML models using the cloud.

The next sections of this paper are organized as follows. Section II covers the previous papers dealing with this field of research, analyzing these papers to improve this work. Section III shows the previous articles related to this research field and analyzes these papers to enhance this work. Section IV provides information about data collection, data pre-processing of the data, and a detailed description of the development of the ML models and the experimental setup. Section V examines developed ML cloud computing models' measure, evaluation, and validation. Experimental results are shown and explained in Section VI. Finally, Section VI discusses the finding and lays out the conclusions and future work.

## II. BACKGROUND

### A. Pollution of the Air in Urban Smart Cities

Pollution of the Air is a severe problem in various cities, especially smart cities. Based on World Health Organization (WHO) research, air contamination has developed as a leading danger to human health, accounting for one fatality out of every eight in a year [9]. The difficulty of regulating air pollution still exists in intelligent cities, despite using technology to manage infrastructure and resources effectively. Smart cities can handle air pollution in various ways [10]. One approach uses sensors' data and processing to measure air quality and detect pollution sources. By using this information, appropriate policies and initiatives can be developed to decrease pollution levels. Air pollution in smart urban centers must be controlled through a combination of technical solutions, policy initiatives, and public outreach [11]. In this paper, we mainly focus on developing efficient distributed ML-assisted solutions that can be utilized to estimate the pollution of the air in smart cities. Because the data is time-based, the model should be constructed with a shorter processing time. As a result, we are utilizing a cloud platform.

*1) Air Quality Index Calculation:* Another name for the air quality index (AQI) in several nations is air pollution index (API) [12]. The number of pollutants varies according to the area or location based on the computed AQI. For that, common pollutants factors include specific matter, nitric x-oxide, ammonia, carbon monoxide, benzene, nitric oxide, sulfur dioxide, toluene, xylene and nitrogen dioxide found in the atmosphere [13]. With the concentration of the pollutants, the AQI is computed with the use of the below formula 1 [14].

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - I_{low}) + I_{low} \tag{1}$$

Where,
I = AQI
C = Pollutant Concentration
$C_{low}$ = the break-point concentration $<$ C
$C_{high}$ = the concentration break-point $>=$ C
$I_{low}$ = the break-point index related to $C_{low}$
$I_{high}$ = the break-point index related to $C_{high}$

The one with the highest AQI value among all pollutants is considered the final AQI. This computed value becomes the target feature for our regression model. There is no specific upper limit for the AQI; however, it may be categorized, and this classification varies depending on the country, and we are considering the AQI range used in India [15]. The AQI category includes the AQI range, where AQI values are divided into specific classes. The AQI range is the target variable of the classification model, represented in our datasets by the AQI bucket.

### *B. Machine Learning*

In machine learning, strategies are utilized for training machines to discover patterns and relationships in datasets. Machines produce predictions based on these patterns [16]. We have both supervised and unsupervised machine learning models. If there are no labels for the data, unsupervised machine learning is applied, which can process raw data. For labeled data, supervised models are preferable [17]. We have two significant subcategories of supervised machine learning: classification and regression. Regression yields a continuous numerical value, whereas classification produces an output with defined labels, a value belonging to a predefined group.

Figure 2 depicts a general machine learning model. The model explains how available machine learning models work. We can see the steps required to make predictions from the input data. Also, different algorithms are listed in the figure, along with the evaluation metrics needed for testing these models. The optimal splitting of the data frame into testing and training sets is given in the diagram [18]. We use all these steps to make predictions in our work which are explained in detail in the below chapters.

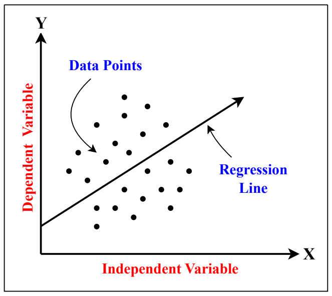

Fig. 3: The notion of ML linear regression models

*1) Regression:* The procedure of identifying a relationship between a specific dependent parameter and numerous independent parameters is referred to as "regression." In plainer terms, it implies fitting a function from a specified group of tasks to the collected input data under some error function [19]. Regression is a technique that is generally utilized in the areas of finance and economics. Some of the best applications of this technique include estimating housing costs, stock market prices, and an employee's salary. The most prevalent regression techniques include linear regression, support vector regressor, decision trees regressor, lasso regression, and random forest. Regression would be suitable for this work because the target variable (Air Quality Index) is a constant numeric value [20]. We are implementing lasso regression and linear regression techniques.

- **Lasso Regression:** LASSO is an acronym for Least Absolute Selection Shrinkage Operator. In general, shrinkage is described as a limitation on features or variables. This algorithm functions by identifying and implementing a constraint on the model parameters that causes the regression coefficients for certain features to shrink until they are equal to zero. The model does not include characteristics such as a regression coefficient of zero. Due to this reason, lasso regression modeling is essentially a shrinkage and feature-choosing approach, and it aids in identifying the essential predictors. While it prevents over-fitting, it will only choose one attribute from a collection of correlated features, and the selected attribute may be strongly biased.
- **Linear Regression:** This regression algorithm belongs to supervised machine learning that predicts a dependent variable using various independent variables. [21] It is less complicated to implement than other regression methods. To predict air pollution, the AQI is the dependent attribute, and the independent characteristics are the sub-indices of the pollutants nitric oxide, particulate matter, ammonia, nitrogen dioxide, toluene, ozone, xylene, sulfur dioxide, nitric x-oxide, benzene, and carbon

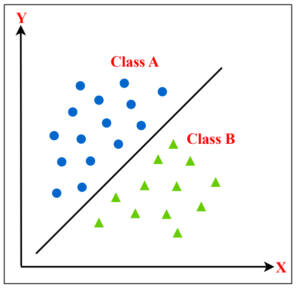

Fig. 4: The principle of ML classification models

monoxide. As illustrated in Figure 3, this regression model provides a relationship between the dependent parameter termed as y and various independent parameters termed as series of x, and this relationship is linear.

*2) Classification:* The classification strategy is a supervised machine learning strategy employed to recognize the class of new findings based on training data. When classifying data, a software program first learns from the original datasets or observations before categorizing new results into a range of clusters or categories, such as zero or one and yes or no [22]. Categories are sometimes known as labels, targets, or groups. In contrast to regression, the target parameter of classification is the name of a category rather than a value. The classification algorithm uses labeled input data; this implies that it includes input with appropriate output because it is a supervised learning process. The best applications of classification algorithms are diabetes detection, email spam detection, emotion prediction, and cat breed detection [23].

The classification algorithm's primary objective is to identify the class of the original data, and these techniques are primarily utilized to predict the outcome for categorical variables. With the help of Figure 4 below, classification techniques can be clearly explained. There are two groups in the diagram: Group A and Group B. These groups have attributes related to one another but not other groups. Classification can be further divided into two: binary and multi-class [24]. When it comes to multi-class classification, the algorithm predicts more than one category as opposed to binary classification, where it just predicts yes or no, 0 or 1.

- **Support Vector Machine (SVM):** SVM technique's fundamental objective is to discover a hyperplane with n dimensions (where n denotes the total range of parameters) which distinctly labels the data items. Numerous feasible planes could be utilized to segregate the two groups of data items. The plane which has the highest margin, that is, the longest distance needed in both the classes to separate the points, is what we are aiming for. To enhance ML classification accuracy, in terms of the data points, the margin distance should be maximized. For classifying the data points, hyperplanes serve as decision boundaries. Data points on both ends of the plane can be assigned to distinct categories.
  In addition, the size of the hyperplane is determined by the count of features. When the count of input parameters is two, the hyperplane becomes just a line. When the count of input parameters is increased to three, the hyperplane changes into a plane with two dimensions. When there are more than three features, it is not easy to imagine a hyperplane. Data points not far from the hyperplane or close to the two dimensions plane are known as support vectors. These support vectors impact both the orientation and the position of the plane, and by using these support vectors, the classifier's margin is increased. When the support vectors are deleted, the plane will change its orientation.
- **Random Forest (RF):** A smaller decision tree that work collaboratively to complete tasks is referred to as a "random forest." This Algorithm can deal with the input data comprising either continuous variables, concerning regression, or categorical variables, concerning classification; this is considered one of its most significant features [25]. Classification problems produce valuable outcomes. Understanding the ensemble technique is essential before learning how the random forest algorithm functions. Ensemble refers to the combination of various models. In Ensemble, we have bagging and boosting. As Figure 5 shows, random forest works on the bagging principle. In this principle, different subsets are obtained using the original training data set as the replacement, and the final output is decided by voting [26]. As a result, rather than a single model, a set of models is employed to generate predictions. A category prediction is provided by every individual tree in the random forest. The category with the highest majority becomes the model's final prediction. The crucial factor is the low correlation of the models. Uncorrelated models have the ability to provide ensemble predictions that are more precise when compared to any of the particular predictions, similar to the assets with weak correlations merging to form a collection that is larger than the sum of the parts of this collection. The trees protect each other from their different individual errors, which results in this impressive effect.

### *C. Cloud Computing*

The concept of offering widespread, accessible, on-demand system access to a distributed pool of customizable computing services is known as "cloud computing." It allows for rapid deployment and release of these resources with much less organizational operations or service operator involvement [27]. A transition in the computing framework occurs when the responsibility of computing is transferred from personal desktop computers, single server applications, or personal data centers to a cloud of computers. The fundamental mechanisms of how computing is performed in the cloud are concealed, so clients are concerned only about the computing function being requested.

The majority of cloud-based computing services are categorized into four major classes: serverless, infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). Since the architecture of these servers show that they are stacked on top of each other, it is typically known as the "cloud computing stack" architecture. It is simpler

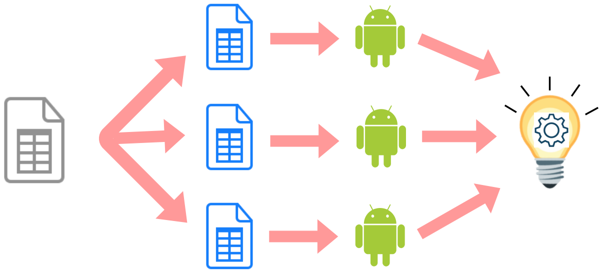

Fig. 5: The notion of RF bagging model

to achieve the business's goals when one knows what these categories are and how they differ.

*1) Infrastructure as a Service:* The IaaS offers a computing platform as a service upon request. In an entirely outsourced, on-demand framework, the user buys software data storage, network infrastructure, or servers and rents those resources. These resources are supplied as a service, and flexible scalability is enabled. Multiple users are usually handled on an individual hardware platform. Choosing resources efficiently and in accordance with needs is entirely up to the consumer. It also facilitates billing management.

*2) Software as a Service:* The SaaS is a type of software delivered like a hosting service and obtained across Output Rephrased or Re-written Text on the internet. It also refers to a method of software delivery in which software and its relevant information are hosted securely and retrieved by their user, generally a web browser, through the internet. The development and implementation of present applications leverage SaaS services. A reliable internet connection and a browser permit access to software and its functionalities from any location. Users from diverse backgrounds can also access an application through the internet because it is centrally hosted.

*3) Platform as a Service:* The PaaS is a cloud deployment platform for projects that consist of resources controlled by a third party. It offers elastic scaling for the application, allowing programmers to develop programs and services through the internet, with deployment methods including private, public, and hybrid. Simply put, it refers to a platform where a third-party enterprise offers cloud computing access to hardware and software tools. Developers use the software and hardware tools that are offered. An alternative name for PaaS is "application PaaS." With its support, we can manage and handle valuable applications and services. It is less expensive than IaaS and includes a robust management platform.

## III. RELATED WORK

### A. *Machine Learning-based Smart City Applications*

In the majority of earlier studies, machine learning is used for forecasting air quality in smart cities. Deep machine learning and regression models are the most commonly utilized machine learning techniques. At the same time, some studies also employ the support vector machine approach because these models function better than the other machine learning methods [28]. The previous section contains a complete overview of regression models and the support vector machine technique. AI/ML algorithms have grown more significant in a variety of industries [29].

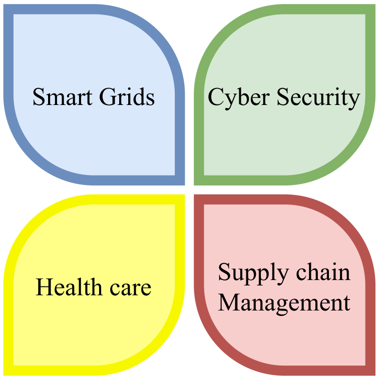

Fig. 6: Example of ML-based applications in smart cities

The safety in urban cities can be improved by utilizing AI in IoT applications [30]. The main emphasis is on enhancing the urban infrastructure to improve living standards. It emphasizes the significance of machine learning in fields such as supply chain management, cybersecurity, smart grids, and healthcare associated with smart cities, as illustrated in Figure **??** [31]. This research highlights the relevance of machine learning (ML) in various essential enabling technologies, such as logistics management, smart metering, medical care, and intelligent transportation. In addition, this paper evaluates different data intrusions for smart cities and outlines the primary elements impacting the development of urban smart cities.

A broader class of machine learning strategies based on representation learning and artificial neural networks is called deep learning [32]. Deep learning integrates different layers to extract highly complex features from the raw input data gradually. In deep learning, every level gains the capability to convert its input data into a composite form that is a little more abstract [33]. The learning process can determine how to place the features at different levels. For many years, DL has made essential advancements in artificial intelligence by utilizing machine learning techniques to address significant issues [34]. The ANNs are popular neural systems that are composed of neurons triggered by connections that are weighted based on prior activation [35]. Deep Neural Network (DNN) architecture processes ML data sets and more profoundly and appropriately classy or predicate the data set in complex applications.

*1) Regression models:* Regression is one of the best machine-learning techniques for predicting air pollution. This is because the variable to be predicted is a continuous value. In a study by Saba Ameer et al., they compare four distinct sophisticated regression techniques (Gradient Boosting, Random Forest, Decision Tree, and Ann Multi-Layer perceptron) to examine which technique is most accurate at estimating air quality given the amount of information and the time required for processing the data [36]. The model's conclusions suggest that random forest regression is the most accurate approach for predicting pollution levels for datasets that differ in size, location, and other properties.

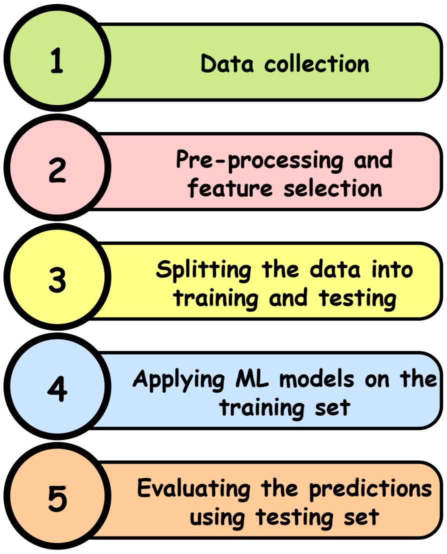
1
Data collection
2
Pre-processing and feature selection
3
Splitting the data into training and testing
4
Applying ML models on the training set
5
Evaluating the predictions using testing set

Fig. 7: General ML Model Development Steps

TABLE I: Features of the datasets with description.

| FEATURES | DESCRIPTION |
|---|---|
| AQI | Air quality index |
| AQI bucket | Air quality index bucket |
| PM2.5 | Particulate matter 2.5-micrometer in ug / m3 |
| PM10 | Particulate matter 10-micrometer in ug / m3 |
| NO | Nitric oxide in ug / m3 |
| Benzene | Benzene in ug / m3 |
| CO | Carbon monoxide in mg / m3 |
| Toluene | Toluene in ug / m3 |
| NOx | Nitric x-oxide in ppb |
| Xylene | Xylene in ug / m3 |
| SO2 | Sulphur dioxide in ug / m3 |
| O3 | Ozone in ug / m3 |
| NH3 | Ammonia in ug / m3 |
| NO2 | Nitric dioxide in ug / m3 |
| City | Cities of India |

The computation time was remarkably lower than that of both gradient boosting techniques and multi-layer feedforward neural networks, according to the results. However, this study is limited to regression techniques, although other machine learning algorithms also perform better at predicting air quality. The research in [37] addresses the before-mentioned limitation by comparing multi-layer convolution and random forest methods on a Malaysian air pollution dataset. The findings suggest that Random Forest outperformed MLP in forecasting the PM2.5 air pollution index in Malaysia's smart city. Another study presented in [38] demonstrates how support vector regression (SVR) enables precise forecasts of hourly pollution concentrations in California using a radial basis function kernel (RBF). Though the outputs achieved are satisfactory, this paper lacks a concise explanation of the methodology. The authors hope to advance their work in the future by optimizing the SVR parameters.

*2) Classification Model:* There has been very little research done to predict air pollution in smart cities using deep machine-learning classification models. Support vector machines are a common technique used in studies that use classification models. In their research [39], the authors forecast the AQI using neural networks and support vector machines. For different decision functions, other kernel functions may be utilized, and by combining multiple kernel functions, more complicated types of planes can be generated. Two vectors can be applied using the kernel function, and a transformation can be used to map each point into a high-dimensional space. Results demonstrated that in the case where the ”MG-SVM” function is employed, the best accuracy of 97.3% is achieved. In our work, we are using more classification models as a result of the small contributions made in this sector.

### *B. Deep Learning Approaches*

The authors of paper [40] proposed an IoT-based air quality prediction and evaluation procedure that measures pollution using a machine learning method known as a recurrent neural network (RNN). Online monitoring is continuously performed on the components necessary to predict pollution levels using a recurrent neural network. All sensor readings are uploaded to a cloud server. A DHT11 sensor is used in this investigation to collect continuous digital temperature and humidity data. The system collects this information utilizing air detectors, which is then transmitted to a microcontroller. The microcontroller uploads the data to a web service after it has been gathered.

A new approach known as Long Short-Term Memory (LSTM) is proposed [41]. It is found that the approach is unstable to build an accurate ML prediction model. The approach enables rapid convergence while reducing training times with high-precision RNNs. However, they commonly experience disappearing and shattering variations that cause the training model to steady down or terminate all at once. In processing the time, standard LSTM and RNN frequently overlook the future information, whereas Bi-LSTM has the ability to make the best of future information.

A new method is proposed to create a heuristic method based on CNN and LSTM and use it to anticipate PM2.5 levels in Beijing's urban area [42]. Consequently, they choose to employ the CNN, CNN-LSTM, GRU, LSTM, and Bi-LSTM to predict the PM2.5 concentrations. The key conclusions of this research are based on effectiveness and comparison of results: the model effectively identifies the spatial and temporal features of the information by employing CNN and LSTM, which have high precision and stability. A related study is documented in the paper [43], however, it is confined to one technique, whereas the article discussed above compares several techniques. When compared to other existing learning methodologies, the LSTM model (APNet) offers the highest predicting efficiency. This technology may help improve air pollution estimation in smart cities.

Air quality in Delhi is forecasted utilizing machine learning approaches which is provided in the research conducted in [39]. The authors employ neural networks and support vector

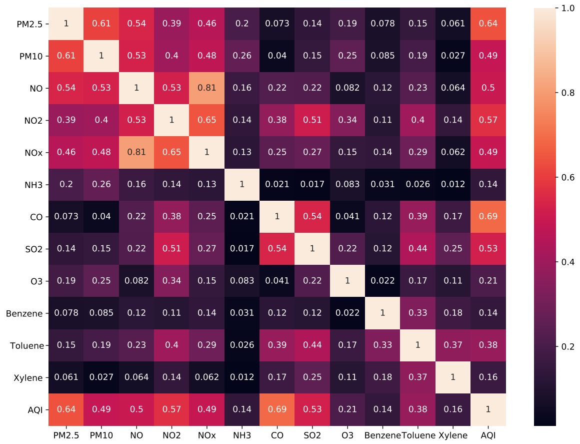

Fig. 8: Correlation Matrix of the Applied Datasets

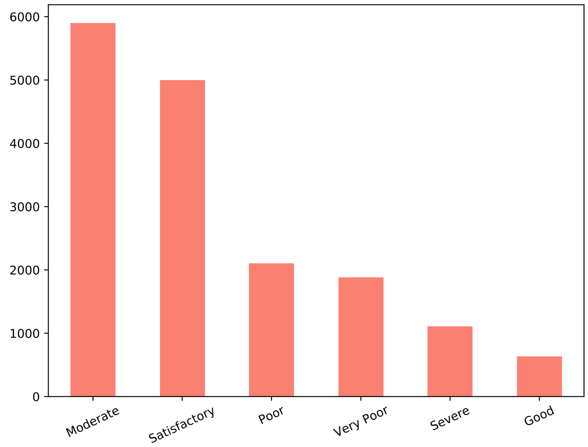

Fig. 9: Barplot for AQI Range

machine approaches in this project. Depending on the training database, the neural network in this study classifies samples as severe, very poor, poor, satisfactory, or good levels of air pollution. According to the results, both SVMs and neural networks are effective at accurately predicting the air quality index, with the former having an efficiency of 91.62 percent and the latter of 97.3 percent.

A deep learning model is proposed to help mitigate air pollution in South Carolina [44]. According to the results of the experiments, the authors initially configured the system with the appropriate hyperparameters. Following that, the proposed algorithm is trained/tested/validated utilizing the popular RMSE and MAEE error function metrics [44]. Overall outputs suggest that regardless of the basic network structure, an LSTM-based forecasting model can enhance prediction performance by memorizing large amounts of historical data. Authors aim to work on more complex models that employ a variety of DL techniques to better analyze IoT data in the long run [45].

### C. *Cloud Enabled AI Approaches*

The research [46] described a cloud-enabled technique of measuring emissions related to vehicular flow, and the outputs were analysed by adding up the emissions from each individual car. The fluctuations in the emission lines are identified by analysing the recurring data obtained by the sensor units strategically placed throughout the area, as well as the raw CCTV footage of the site. The pollution problem in the city of Manipal, Karnataka, has been taken into account in this proposed work. The data from the Transportation Department has been uploaded to the cloud. The proposed effort might be utilized as well to examine the contaminants in the air in residential areas, public spaces, and commercial buildings.

A new method is proposed to build an intelligent predictor of airborne pollutant measures for the following two days with the help of DL techniques, using the RNN approach [47]. For that PSO technique is used to determine the ideal value for the developed model. The smart air quality prediction model (SAQPM) is a predictor related to intelligent computation that relies on unsupervised machine learning, that is, LSTM and development, that is, particle swarm optimization. Big data technology is used to analyze data obtained through transactions, interactions, and observations to discover patterns and draw conclusions.

## IV. DATA AND EXPERIMENTAL SETUP

The experiments are carried out on a personal laptop as well as on the Amazon cloud platform. To conduct the experiments using the Amazon Cloud Platform, we are utilizing the Amazon SageMaker service. We are building two models, one for regression and another for classification. Figure 7 shows the flowchart for our machine learning model.

### A. *Data Collection*

The datasets we utilized for this research were obtained from Kaggle, which consists of air pollution data from India. We're using three separate datasets from India, each for a different period: 2015–2019, 2020, and 2021–2022. We are dividing the datasets due to the fact that the year 2020 was locked down, and the pollutant levels were deficient at that time, which are considered outliers and affect the predictions. As a result, using three different datasets from 2015 to 2022 would yield better predictions. The datasets consist of atmospheric pollutants data and the Air Quality Index (AQI) at hourly and day-to-day levels for different positions across several municipalities in India.

The data is made public by the Central Pollution Control Board, the official website of the Government of India. The cities included in the datasets are Hyderabad, Jorapokhar, Delhi, Amaravati, Bengaluru, Jaipur, Amritsar, Guwahati, Ahmedabad, Chennai, Aizawl, Gurugram, Bhopal, Talcher, Thiruvananthapuram, Brajrajnagar, Kolkata, Chandigarh, Shillong, Coimbatore, Mumbai, Ernakulam, Kochi, Patna, Lucknow, and Visakhapatnam. Table I lists the key features considered in the model, along with their descriptions.

### B. Data Pre-processing

For cleaning the unprocessed data, several techniques are available known as pre-processing approaches. We use exploratory data analysis (EDA) to clean the raw data and prepare it for training. The datasets contain values of the integer, float, and object types. The data has null or missing values, denoted as NaN, in the datasets. This data cannot be processed because the system generates an error as invalid input. These empty spaces are handled by filling these null spaces with the mean of the entire feature. After managing the invalid data, box plots are used to find anomalies. An outlier is a data point that varies from the entire set of other data points in a feature by a large margin. A few outliers in these datasets can be identified and treated using the interquartile range.

- The 25th percentile is represented by Q1.
- The 50th percentile is represented by Q2.
- The 75th percentile is represented by Q3.

The datasets are divided into four equal portions by the three quartiles (Q1, Q2, and Q3). The interquartile range (IQR) is the difference between Q3 and Q1. The lower bound is 1.5 times the IQR subtracted from Q1, whereas the upper bound is 1.5 times the IQR added to Q3. Outliers are values that are outside of this threshold. Instead of dropping these values, we replace them with the lower and higher bound values in our work.

### C. Feature Selection

This selection strategy is a critical procedure in which features from the dataset strongly correlated to the target feature are selected. The correlation values generated by the correlation matrix are used in this process. The closer the values are to ”1”, the stronger the relationship between them. In Figure 8, a heat map depicts the correlation values for the features selected in our research. We can observe that the target variable has a positive value for all the components used in this work because these attributes are air contaminants utilized to compute the air quality index (the target variable).

### D. Splitting Data

The train_test_split() is a function that divides the dataset into testing and training data in a 80:20 proportion. This ratio is usually regarded as the best ratio for partitioning the datasets, but it can be altered depending on the number of records in the dataframe. The training set is operated to create the model, and the testing set is employed to estimate the model's accuracy. After partitioning the datasets into training and testing, we take the X and y sets, where y is the AQI in case of regression. In case of classification, the AQI bucket is created based on AQI values. This y is the target or the dependent variable, and X consists of all the pollutants known as the independent variables.

### E. Balancing the Data

The values of the target variable (AQI Range) in the datasets are not uniform or balanced. This unbalanced dataset decreases the functioning of the model and affects its accuracy. Using the bar plots, we have depicted the imbalanced nature as shown in Figure 9.

To balance the target variable, we are using SMOTE approach. Synthetic Minority Over-sampling Method (SMOTE) is a commonly used data augmentation strategy that is employed to balance class distributions in unbalanced records for a classification algorithm. These unbalanced records have considerably more values in one class than others, resulting in inaccurate model predictions favoring the majority class. SMOTE creates new data examples among current minority class examples to generate synthetic instances of the minority category. SMOTE aims to normalize the category labels in the dataset by creating synthetic illustrations, which decreases the probability of a model preferring the majority class.

### F. Machine Learning Techniques

Machine learning algorithms are applied upon the completion of the pre-processing of the dataset. This study uses regression models (linear regression and lasso regression) and classification models (support vector machine and random forest). The parameters are further divided into X and y after the data has been split into testing and training sets. In total, we now have four data frames: X_test, X_train, y_test, and y_train.

*1) Linear Regression:* Linear regression is utilized to estimate a dependent feature (y) related to a set of independent features (X). Thus, the term ”linear regression” was proposed. The line that fits the model the best is the regression line.

**Hypothesis function:**

$$y = \theta_1 + \theta_2 X \tag{2}$$

Where,
$\theta_1$ = intercept.
$\theta_2$ = coefficient of X.

**Cost Function:**

$$J = \frac{1}{n} \sum_{i=1}^{n} \left( y^{(i)} - h(x^{(i)}) \right)^2 \tag{3}$$

Where,
n = complete number of training samples in the dataset.
h(x(i)) = hypothetical function for prediction.
y(i) = value of target variable.

By determining the best $\theta_1$ and $\theta_2$ values, the model obtains the best regression fit line. The model is focused on predicting the y value with the best-fit regression line so that the error dissimilarity between the forecasted and actual values is as low as possible. So, it is necessary to update the $\theta_1$ and $\theta_2$ values in order to discover the value that decreases the difference between the anticipated y value and the true y value. The model uses gradient descent to adjust $\theta_1$ and $\theta_2$ to lower the cost function and get the line that fits the best. The concept is to iteratively modify the values until they are close to the minimum cost, starting with random values for $\theta_1$ and $\theta_2$

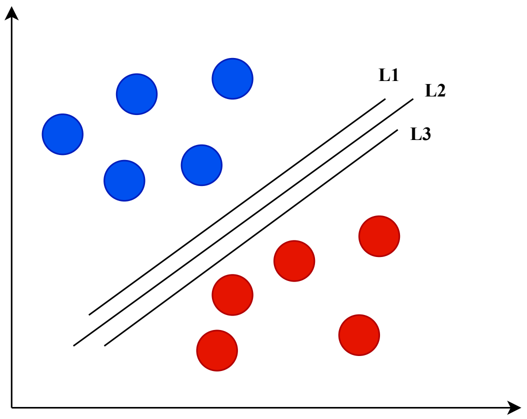

Fig. 10: Support Vector Machine

*2) Lasso Regression:* Lasso regression is another linear model obtained from linear regression that employs the same hypothetical function for prediction. The linear regression model treats all parameters as equally crucial for prediction. When the dataset contains many features, even when some are irrelevant to the predictive model, this complicates the model and results in an inaccurate prediction on the test set (or over-fitting). A high-variance model of this kind cannot be generalized to the new data [48]. For solving this problem, Lasso regression is used, which includes an L1 penalty in the linear regression cost function.

$$J = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - h(x^{(i)})\right)^2 + \lambda\sum_{j=1}^{m} w_j \tag{4}$$

Where,
$w_j$ = weight of jth feature.
$\lambda$ = regularisation strength.
m = number of features in dataset.

The additional l1 penalty during gradient descent optimization shrunk weights to nearly zero or zero. The features in the hypothetical function are eliminated when the weights are shrunk to zero. As a result, irrelevant features are excluded from the prediction model. The penalization of weights simplifies the hypothesis, which favors sparsity. The intercept remains unaffected when it is added. Bias increases with increasing lambda, whereas variance increases with decreasing lambda. A feature of the model is eliminated as lambda increases because more and more weights are reduced to zero.

*3) Support Vector Machine:* The fundamental objective of this technique is to discover a hyperplane with n dimensions (where n denotes the total range of parameters) which distinctly labels the data items. Numerous feasible planes could be utilized to segregate the two groups of data items. The optimal hyperplane is the one with the most significant distance or margin between all the classes, which is shown in Figure 10.

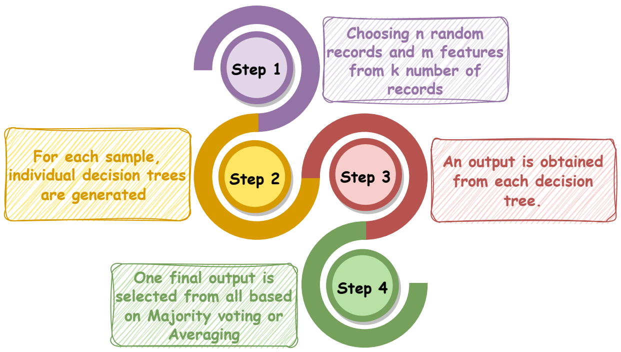
Step 1
Choosing n random records and m features from k number of records
Step 2
For each sample, individual decision trees are generated
Step 3
An output is obtained from each decision tree.
Step 4
One final output is selected from all based on Majority voting or Averaging

Fig. 11: Steps in Random Forest Algorithm

*4) Random Forest:* The random forest strategy is a bagging method extension that employs both bagging and attribute randomness to develop an uncorrelated forest of decision trees. Steps involved in the random forest algorithm are shown in Figure 11. Random forests select only a subset of the available feature splits, whereas decision trees involve all. As the model grows the trees, a random forest adds more randomness. When splitting a node, instead of looking for the essential parameter, it seeks out the best feature among the random selection of attributes. As a result, there is a greater variety, which leads to a better model. The procedure for dividing a node in a random forest only assumes a random subset of the parameters.

An overview of Distributed Cloud Services in Smart Cities is depicted in Figure 12 (a), while the Implementation and Testing of ML-assisted Distributed Framework using the Amazon SageMaker Platform is illustrated in Figure 12 (b). As seen in Figures our AI-assisted cloud machine learning service termed Amazon SageMaker allows users to develop, train, and use machine learning techniques. It comes with various machine-learning algorithms that programmers can use on their data. Amazon SageMaker Studio extends the capabilities of JupyterLab with new tools that enhance machine learning operations by leveraging AWS's computational capacity. Amazon SageMaker has high scalability and can easily handle large datasets and modeling strategies. As a result, it is feasible to analyze vast amounts of data more efficiently, which can significantly minimize processing time. Amazon SageMaker provides high-performance computing instances for tasks related to machine learning. These instances are equipped with powerful CPUs and GPUs, which can considerably speed the implementation of machine learning models. As shown in the Figure 12, IoT devices of the smart city, such as smart buildings, IoT/health devices, and smart vehicles, etc., are connected to the cloud using wireless connections, i.e., Wi-Fi, to provide efficient cloud services. the developed ML cloud architecture includes data gathering and storage, data pre-processing, training the model, and model deployment for air quality prediction using machine learning, as shown in Figure **??**. As seen in the model Amazon has a big market share among cloud computing service providers because of its low-cost structure and flexible toolsets. It offers services ranging from infrastructure technologies like storage, computation, and databases to emerging technologies like machine learning, artificial intelligence, and IoT. It has been designed as a flexible and safe cloud computing environment. The central infrastructure has been constructed to meet the security standards of the military, significant banks, and other

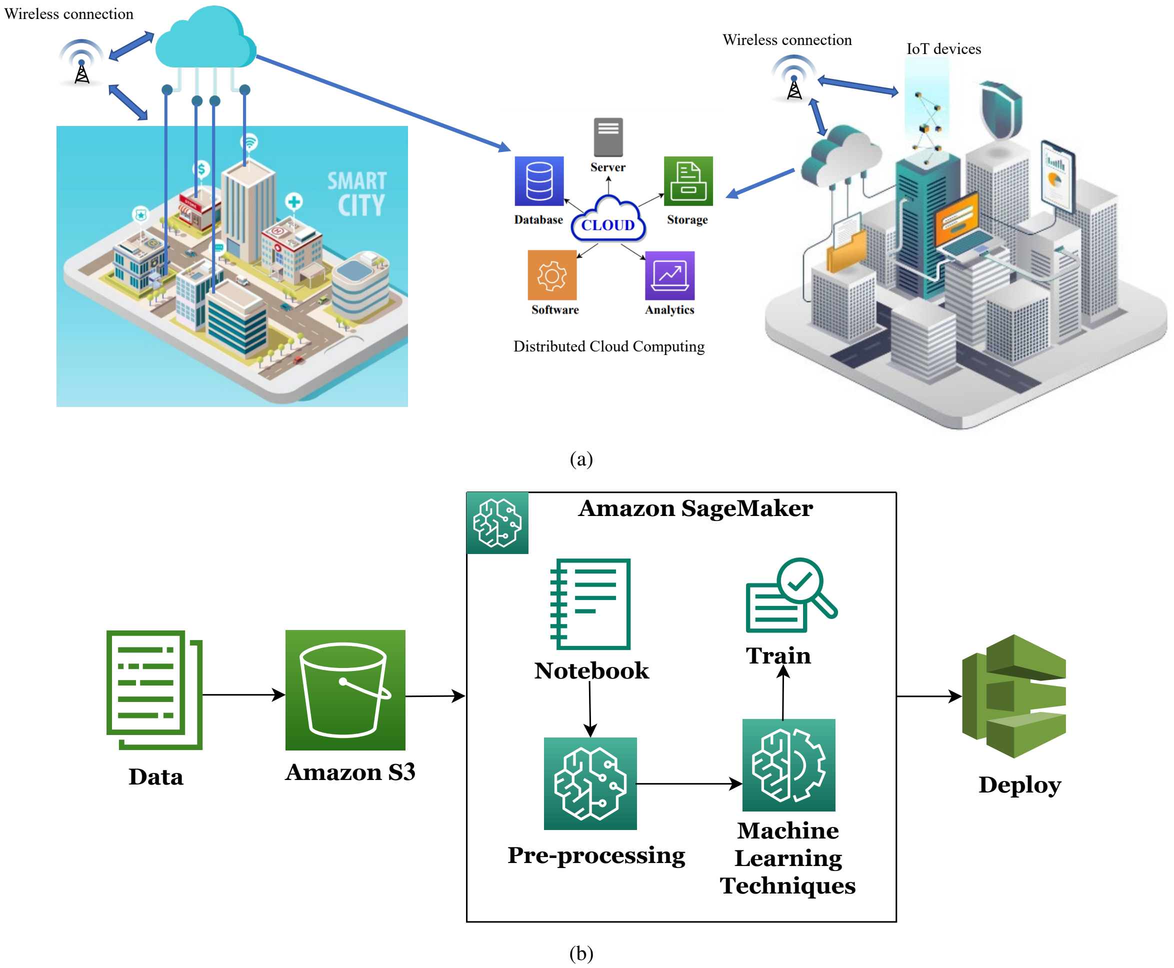

Fig. 12: The Proposed AI-assisted Distributed Cloud Services Framework for Enhanced Safety in Urban Smart Cities Environment: a) Overview of Distributed Cloud Services in Smart Cities, b) Implementation and Testing of ML-assisted Distributed Framework using the Amazon SageMaker Platform.

susceptible organizations. We use the Amazon SageMaker Service to deploy our machine-learning models from all the provided services. Individuals without much knowledge of machine learning can also use this service to execute their models, which is a significant advantage.

*5) Amazon SageMaker:* A cloud platform for machine learning called Amazon SageMaker enables programmers to build, train, and implement machine learning models in the cloud. SageMaker delivers pre-trained machine learning algorithms that may be deployed as-is at the highest degree of abstraction [49]. Furthermore, SageMaker includes a number of built-in machine learning techniques that programmers can use on their data. Moreover, SageMaker offers maintained instances of Tensor Flow and Apache MXNet so that programmers can build custom machine learning algorithms from scratch [50]. We don't need to manage servers because it includes an integrated Jupyter Notebook instance for quick access to the datasets for analysis and exploration. It also contains standard machine learning methods that have been designed to perform efficiently on massive datasets in a distributed context.

Amazon SageMaker Studio is a machine learning integrated development environment (IDE) that allows you to construct, train, debug, deploy, and monitor machine learning models [51]. SageMaker Studio has everything users need to move their models from data pre-processing to experimenting to deployment while increasing productivity.

Amazon SageMaker Studio expands JupyterLab's capabilities with custom resources that can enhance the machine learning operation by leveraging AWS compute capacity. JupyterLab 1 and JupyterLab 3 are now supported by Studio, where JupyterLab 3 is the default JupyterLab version in Studio.

*6) SageMaker Architecture:* This SageMaker Architecture consists of built-in features and abilities make it a highly flexible and adaptable platform for creating and implementing machine learning models. The following aspects would be included in the architecture for forecasting air quality in smart cities using Amazon SageMaker:

- The datasets collected from kaggle which are air quality data from India from 2015 to 2022 can be stored in Amazon S3 bucket.

- We use the SageMaker's Jupyter Notebook to build the models.
- To make the collected data suitable for machine learning, pre-processing is required. For this purpose, it is necessary to clean the data and prepare it for applying the machine learning techniques.
- After pre-processing the data, we apply our four models on the training set.
- We evaluate these models using the testing set.

Furthermore, for this research, we embarked on an in-depth exploration of [52]–[58] employing a comprehensive approach that encompassed research methodologies and algorithms.

## V. EVALUATION METRICS

The concept of developing machine learning or artificial intelligence models is based on the principle of constructive feedback. We develop a model, get insights from metrics, improve, and repeat until we obtain the needed accuracy. Evaluation standards explain the model's performance. The capability of evaluation metrics to distinguish between model outputs is an important segment. In this field, we evaluate machine learning instances using a variety of measures. The evaluation metric chosen is entirely dependent on the kind of model and the model's implementation strategy. When we speak of predictive models, we are referring to either a classification approach (binary result) or a regression standard (continuous outcome). Each of these measures uses a different set of evaluation metrics.

### *A. Regression*

When estimating a numeric value, such as a length or a sales price, we are not interested in knowing if the model anticipated the value precisely (this may be impractical in practice); rather, we would like to determine how closely the predictions matched the actual data values. Error handles this directly and highlights, in general, how close forecasts were to their predicted values. There are four evaluation measures that are commonly used for assessing and presenting the performance of a regression algorithm as listed below. Though these are the four most frequently utilized regression metrics, there are many others.

- Mean Absolute Error
- Mean Squared Error
- Root Mean Squared Error
- R2 Score

*1) Mean Absolute Error (MAE):* Mean absolute error is a simple and direct metric that computes the fundamental difference between the actual and estimated measures. Now, we must determine our standard's mean absolute error, a misconception caused by the method and regarded as an error. Then, we will estimate the difference between the true and estimated measures; this would be an absolute mistake, though we must determine the mean absolute of the whole testing set. Therefore, add each error and split those by the complete number of instances. Since this is a loss, we must get the lowest mean fundamental error possible.

$$MAE = \frac{1}{N} \sum \left| y - \hat{y} \right| \tag{5}$$

Where,
y = actual values.
$\hat{y}$ = predicted values.
N = total data values.
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test,y_pred)

The MAE value you received belongs to the identical type as the result parameter. It is the most invulnerable to outliers, but the chart of mean absolute error cannot be distinguishable, so we must utilize different techniques, such as gradient descent, which is distinguishable.

*2) Mean Square Error:* MSE is a prominent error measure for regression challenges. Additionally, it is a significant loss function for techniques that fit or optimize regression problems that use the least squares framework. In this context, least squares reduce the mean square error between true and estimated values. The average or mean of the squared differences between a dataset's calculated and actual numerical values is used to calculate the MSE. It is a regularly used and extremely simple statistic that changes the mean absolute error. Calculating the squared difference between the true and estimated weight is necessary according to the mean squared error. Hence, we discovered the fundamental difference above and the squared difference here. It denotes the squared difference between the real and expected measurements.

To the benefit of MSE, we square terms in order to prevent the cancellation of negative words. The graph of mean square error is distinguishable; hence, it can be utilized as a loss function. The result of calculating the mean square error is a squared measure of output. If the collected data includes outliers, they are penalized more than the null values, and the evaluated mean square error is high. To outline, the model is not potent to outliers that were advantageous in mean absolute error.

$$MSE = \frac{1}{n} \sum (y - \hat{y})^2 \tag{6}$$

Where,
y = actual values.
$\hat{y}$ = predicted values.
n = total data values.

from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test , y_pred)

*3) Root Mean Squared Error:* As the name indicates, it is basically the square root of the mean squared error. It is simple to comprehend a failure since the outcome we receive falls into the identical unit as the required outcome feature. In contrast to the mean absolute error, it is less robust to outliers. Root mean squared error is commonly employed as a testing measure.

$$RMSE = \sqrt{MSE} \tag{7}$$

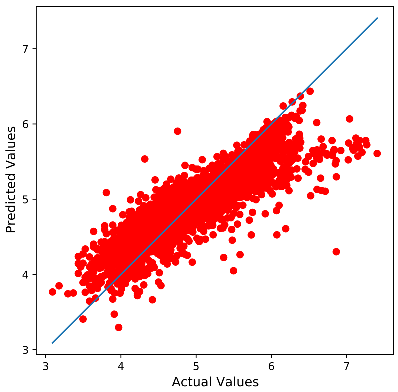

Fig. 13: Actual vs Predicted graph for Air Quality in India(2015-2019) dataset using linear regression

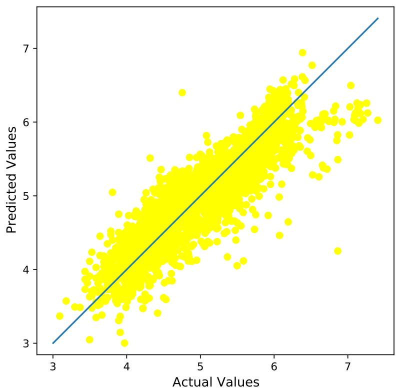

Fig. 14: Actual vs Predicted graph for Air Quality in India(2015-2019) dataset using lasso regression

$$RMSE \; = \; \sqrt{\frac{1}{n} \sum \; (y - \hat{y})^2} \tag{8}$$

rmse = np.sqrt(mean_squared_error(y_test,y_pred))

*4) R2 Score:* The R2 value is a measure that indicates the effectiveness of the prototype rather than the complete loss. As seen above, MSE and MAE are dependent on context, while the R2 value is independent of context. Hence, by employing R2, we can resemble baseline models, something that neither of the other measures enables. In classification problems, we have what is known as a fixed threshold of 0.5. This evaluation metric estimates how considerably better the line of regression stands than a mean line. It is hence correspondingly referred to as the coefficient of determination or, alternatively, as the goodness of fit.

$$R2 \; Score \; = \; 1 \; - \; \frac{SSr}{SSm} \tag{9}$$

Where,
SSr = Squared sum error of regression line.
SSm = Squared sum error of mean line.

The R2 score increases as the regression line approaches perfection and the prototype's implementation enhances. The common situation exists when the R2 score is between zero and one, like 0.8, implying that the instance can justify 80 percent of the variance in the information.
from sklearn.metrics import r2_score
r2 = r2_score(y_test,y_pred)

### B. Classification

The primary measures used to examine a classification model are F1 score, accuracy, recall, and precision. The percentage of accurate predictions for the testing data is known as accuracy. It is simple to calculate by splitting the number of correct forecasts by the entire number of forecasts. The accuracy measure contains no information on false positives or false negatives. As a result, there is a significant loss of data, as some may help evaluate and improve our model.

*1) Confusion Matrix:* This matrix measures the effectiveness of a classification model and is just a N x N matrix (here, N refers to the numeral of labels). For a confusion matrix, every row signifies a real category, whereas every column signifies an anticipated category. The confusion matrix() method from the SkLearn library can be used to create this matrix instantly.

- True positive (TP) denotes that the real and forecasted values are correct.
- False positive (FP) means that the true value must be false whereas the predicted value is true, indicating an error.
- True negative (TN) implies that both true and predicted values, are untrue.
- False negative (FN) signifies that the real value must be true, while the predicted value is false.

*2) Precision:* Precision is described as the ratio of the technique's true positive value and its overall true positive value. Precision could be simply computed using the SkLearn library's precision score() function. Precision will not be enough because a model can only make one true positive prediction while returning the others as negative. As a result, the precision would be $1/(1 + 0) = 1$. Precision, in combination with another measure known as "recall," is required.

$$Precision \; (P) \; = \; \frac{TP}{(TP + FP)} \tag{10}$$

*3) Recall:* This metric is referred to as the proportion of the digit of true positives to the complete number of actual positives. The terms "true positive rate" and "sensitivity" are other names for recall. The recall score() procedure in the SkLearn library makes it simple to compute recall.

$$Recall \; (R) \; = \; \frac{TP}{(TP + FN)} \tag{11}$$

*4) F1 Score:* Another classification measure that combines recall and accuracy is the recall-precision metric which is known as the f1 score. It is the harmonic average of recall and precision. The harmonic average is much more vulnerable to low values, so the f1 score will be great only after precision and recall are both high. The SkLearn library's f1 score() function makes it simple to compute the f1 score.

$$F_1 \ Score \ (R) \ = \ \frac{PR}{P+R} \tag{12}$$

Where,
P = Precision.
R = Recall.

## VI. RESULTS AND DISCUSSION

We use regression and classification models to forecast air quality in Indian smart cities. The f1 score, recall, accuracy, and precision are used to assess the SVM and random forest algorithms. The mean square error, R2 score, mean absolute error, and root mean square error are utilized to estimate linear regression and Lasso regression models. We are also calculating the execution time for each model separately. The process is repeated on the Amazon SageMaker service, and the results are compared in order to determine which method predicts air quality in smart cities better.

The outcomes of air quality prediction employing a regression algorithm would be determined by the model used and the data collected. The regression analysis results could provide insight into the elements most strongly connected with fluctuations in air quality. This research could inform decisions on public policy and other actions that enhance air quality. In general, the outcomes of regression-based air quality prediction can offer valuable data for monitoring and improving air quality, in addition to guiding public health decision-making and policy. The results for all three datasets are provided in the section below. We are presenting the outputs in the form of graphs and tables.

The actual vs. predicted graph for the Air Quality in India(2015-2019) dataset using linear regression is depicted in Figure 13, and the same graph for lasso regression is shown in Figure 14. In these scatter plots, the actual values in the dataset are graphed against the model's predicted values. These graphs are considered one of the richest forms of data visualization. The diagonal blue line is the line of regression, and the yellow and red dots are the data points. The line of regression is determined from the regression function. This dataset has more data than the other two datasets.

Hence, we can see more points in these two graphs. Comparing these two figures demonstrates that the data points in the lasso model are closer to the regression line than the linear model. These illustrations mean the lasso model has better performance in our work. The R2 value is higher for the lasso method because the nearer the points are to the diagonal line, the greater is the R2 score.

The actual vs. predicted graph for the Air Quality in India(2020) dataset using linear regression is depicted in Figure 15, and the same graph for lasso regression is shown in Figure 16. We plot the true values that we have in the dataset against the predictions made by our model. The blue line denotes the regression line, and the orange and brown dots represent the data points. The regression function determines the line of regression. The points in these two graphs are fewer when compared to the above two graphs. The reason for this is the amount of data we have in the dataset.

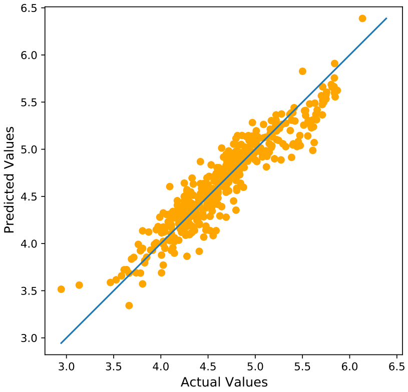

Fig. 15: Actual vs Predicted graph for Air Quality in India(2020) dataset using linear regression

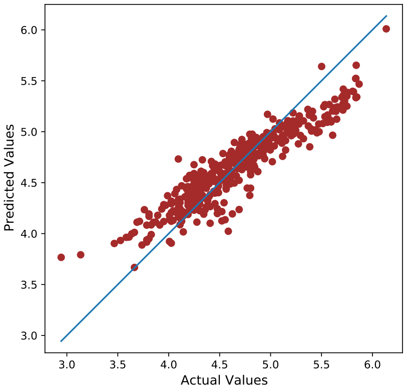

Fig. 16: Actual vs Predicted graph for Air Quality in India(2020) dataset using lasso regression

With the lasso model, the points are located nearer the line of regression than in the linear model, as seen when we analyze these two graphs. This analysis implies that the lasso method operates better in our work. The R2 score for the lasso model is higher since the points are nearer to the regression line.

For all three datasets, the depictions are similar, and the Figures 17 and 18 depict the actual vs predicted charts for Air quality in India(2021-2022) dataset using linear and lasso regression respectively. Purple and green dots are the data points, and the blue diagonal line represents the regression line. This dataset has very few values displayed in the figures above. In the linear model, the points are almost in the form of a horizontal line. These demonstrations can be because the model is under-fitting. The results of this dataset are the same as the other two datasets. That is, lasso regression outshines linear regression.

The evaluation measures used for all three datasets for testing regression models are listed in three tables. Lasso

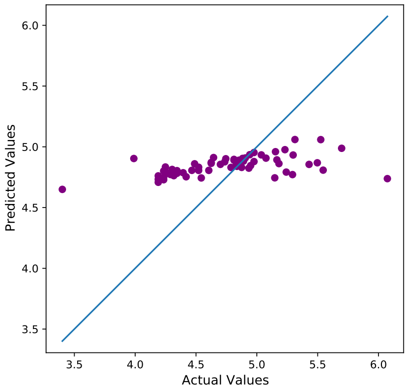

Fig. 17: Actual vs Predicted graph for Air Quality in India(2021-2022) dataset using linear regression

TABLE II: Evaluation of Regression models for India(2015-2019) dataset.

| **Evaluation Metrics** | **MAE** | **MSE** | **RMSE** | **R2 Score** |
|---|---|---|---|---|
| **Lasso Regression** | 0.2218 | 0.0866 | 0.2944 | 80.66 |
| **Linear Regression** | 0.2502 | 0.1110 | 0.3332 | 75.22 |

regression performs better than linear regression according to the evaluation metrics also. Among all the metrics, we consider the R2 score in most cases. For all our datasets, the R2 scores for lasso models are higher than for linear models. All the other measure values should be low for a good model, and in most situations, lasso regression does that; hence it is considered a better approach for estimating air quality in smart cities.

The first dataset's evaluation results, the Indian Air Quality from 2015-2019 dataset, are given in Table II. We are getting an R2 score of around 80 percent for the lasso model, more significant than the value obtained for linear regression. All the other measures are higher for the linear model than the lasso model. These measurements indicate that for this dataset, lasso regression does well than linear regression. Above Table III lists the evaluation metrics for the Air Quality in India(2020) dataset. In this table, the values have less difference for lasso and linear approaches. These results are similar to the dataset above.

The evaluation metrics for the Air quality in India(2021-2022) dataset are given in Table IV. Here, the R2 score for the lasso algorithm is greater than the linear algorithm, though the values have a minimal difference. But we can see that the mean square error is more impressive for lasso regression along with the root mean square error, which differs from the other two datasets. This difference is due to the amount of data that we have in the dataset.

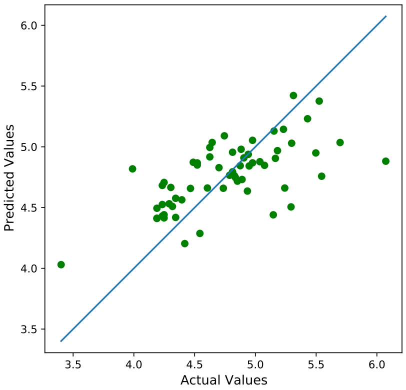

Fig. 18: Actual vs Predicted graph for Air Quality in India(2021-2022) dataset using lasso regression

TABLE III: Evaluation of Regression models for India(2020) dataset.

| **Evaluation Metrics** | **MAE** | **MSE** | **RMSE** | **R2 Score** |
|---|---|---|---|---|
| **Lasso Regression** | 0.1462 | 0.0343 | 0.1854 | 87.56 |
| **Linear Regression** | 0.1806 | 0.0522 | 0.2285 | 81.09 |

TABLE IV: Evaluation of Regression models for India(2021-2022) dataset.

| **Evaluation Metrics** | **MAE** | **MSE** | **RMSE** | **R2 Score** |
|---|---|---|---|---|
| **Lasso Regression** | 0.2730 | 0.1292 | 0.3595 | 74.25 |
| **Linear Regression** | 0.3458 | 0.1158 | 0.3403 | 71.42 |

Lasso regression is a linear regression variant that integrates regularization to avoid over-fitting and enhance the model's predictive performance. Employing lasso regression is particularly beneficial when there are several variables, and a few of them might not be significant. While predicting air quality, numerous possible variables, including all air pollutants, can influence the results. In this case, lasso regression is preferable to simple linear regression. The reasons for this are:

- The most significant variables can be found using lasso regression, whereas irrelevant or minor factors can be disregarded. This can result in a more compact and interpretable model.
- Regularization is used in Lasso regression to reduce over-fitting and enhance model generalization ability by shrinking the coefficients of less significant variables towards zero.
- The accuracy of the model can be increased by adding polynomial or interaction variables in lasso regression to identify non-linear interactions between variables and outcomes.

In Figure 19, we use a bar graph to compare the R2 scores obtained by linear regression and lasso regression techniques for all three datasets. The green bars indicate linear model values, and the pink bars indicate lasso model scores. The graph shows that for all the datasets, the lasso model surpasses the linear approach.

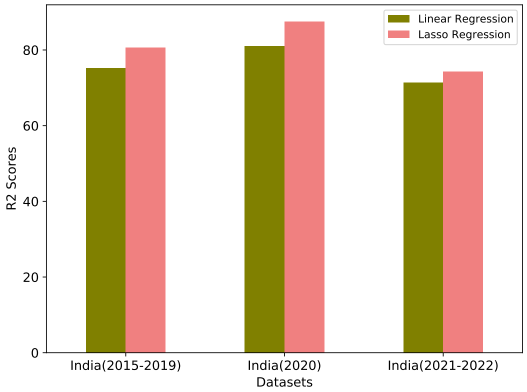

Fig. 19: Comparison of R2 Scores for all Datasets

## A. *Classification*

Classification is a different way to predict air quality than regression analysis. Classification is the process of categorizing or classifying data based on its characteristics, whereas regression includes estimating a set of continuous numeric values. The classification model's accuracy would be measured using evaluation measures such as f1 score, recall, accuracy, and precision. Here, the model's accuracy is computed as the amount of correct forecasts divided by the complete amount of estimations. In contrast, other metrics are computed using the values from the confusion matrix. These measurements indicate how effectively the model can correctly classify air quality into various categories. However, it is crucial to note that the classification algorithm's accuracy relies on the data quality utilized to construct the model, the features used for classification, and the classification algorithm utilized.

A confusion matrix gives a prediction summary for a model. It shows the amount of incorrect and correct predictions for every class separately. The precision, F1 scores, and recall are measured using the values obtained from the confusion matrix. Figure 20 depicts the confusion matrix of the random forest approach. Here, we have six columns and six rows because the classes we have in our work are six. As we can see in this diagram, the y label illustrates the true or actual values in the dataset. In contrast, the x label represents the values the random forest model predicted.

Figure 21 shows the confusion matrix of the support vector machine model. In this, we have six columns and six rows because the classes we have in our work are six. As depicted in this diagram, the y label represents the true or actual values in the dataset, whereas the values predicted by the random forest model are represented by the x label.

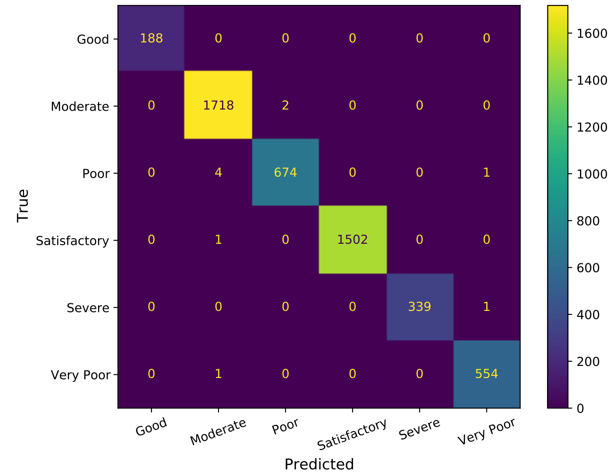

Fig. 20: Confusion Matrix of Random forest model

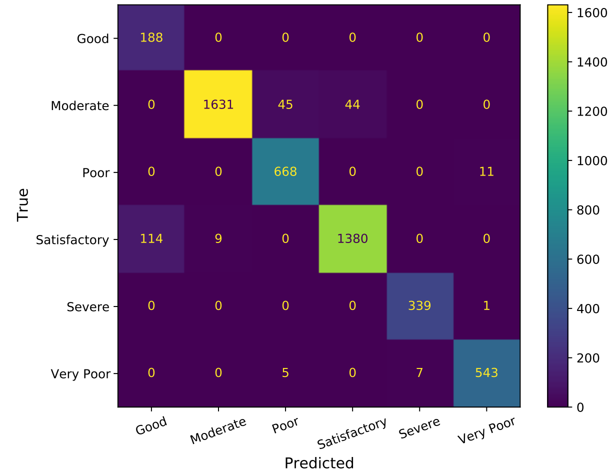

Fig. 21: Confusion Matrix of Support vector machine model

TABLE V: Evaluation of Classification models for India(2015-2019) dataset.

| **Evaluation Metrics** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
|---|---|---|---|---|
| **Random Forest** | 99.8595 | 99.8604 | 99.8672 | 99.8638 |
| **Support Vector Machine** | 95.4262 | 91.5197 | 97.2247 | 93.6382 |

The evaluation outcomes for the first dataset, the Indian Air Quality from 2015-2019 dataset, are given in Table V. The random forest algorithm outperforms the SVM algorithm, as seen in the table. All four measures used for testing the model give higher values for the random forest method than for the support vector machine method. We have more relevant results for this dataset than the other two datasets because the amount of data here is more, which helps the models understand the trends in the dataset more and, in turn, helps them generate accurate predictions.

Table VI lists the results for the Air Quality in India(2020) dataset where classification models are evaluated using some metrics. The results are similar to the above dataset, which means the random forest approach has better values than the SVM approach. The difference in the values for both

TABLE VI: Evaluation of Classification models for India(2020) dataset.

| Evaluation Metrics | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| **Random Forest** | 98.8580 | 99.0174 | 90.7186 | 93.4667 |
| **Support Vector Machine** | 91.3539 | 88.1005 | 87.5075 | 85.1661 |

TABLE VII: Evaluation of Classification models for India(2021-2022) dataset.

| Evaluation Metrics | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| **Random Forest** | 93.5483 | 91.4814 | 94.4421 | 92.4836 |
| **Support Vector Machine** | 93.5483 | 72.2214 | 70.8334 | 71.5079 |

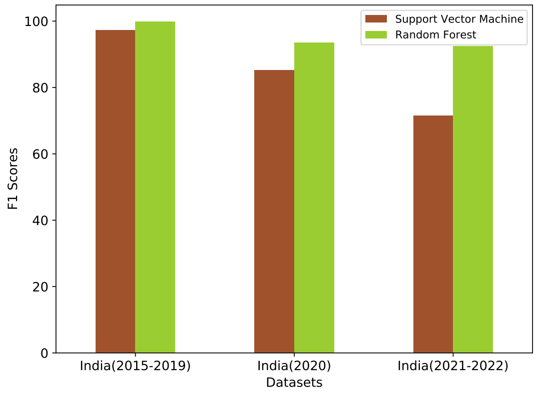

Fig. 22: Comparison of F1 Scores for all Datasets

techniques is noticeable in this dataset. The accuracy and precision values are very high for the random forest algorithm compared to the recall and F1 scores, but this is not considered problematic as the difference is inadequate.

Evaluation results of the classification models for the Air quality in India (2021-2022) dataset are given in Table VII. The outcomes are similar to the other two datasets, but the values are lower here because of the data we have. This dataset has fewer records than the other two datasets; therefore, the models get to learn less. The accuracy is not calculated using the values obtained from the confusion matrix. Hence, it is much higher than the support vector model's recall, precision, and F1 score values.

TABLE VIII: Execution Times.

| Models | Execution Time in secs | |
|---|---|---|
| | Local Machine | Cloud Platform |
| **Linear Regression** | 0.0877 | 0.0301 |
| **Lasso Regression** | 0.0649 | 0.0228 |
| **Regression with EDA** | 15.9758 | 5.6914 |
| **Random Forest** | 2.0584 | 1.7560 |
| **Support Vector Machine** | 22.3402 | 17.8313 |
| **Classification with EDA** | 33.6759 | 21.3631 |

The ability of random forests to handle high-dimensional records, like datasets with numerous attributes, is advantageous for predicting air quality. In contrast, SVMs may encounter difficulties with high-dimensional data. For predicting air quality, there can be complicated relationships between the attributes and the target variables, which a random forest algorithm can handle better than an SVM model. Compared to SVMs, random forests are less vulnerable to outliers in the dataset, which can be helpful when there are sometimes extreme values for specific pollutant concentrations that might affect the data.

Generally, from all the measures we have, the F1 score is preferable. The results are listed in the form of a table for each dataset separately. Additionally, a bar graph is generated, which compares the F1 scores of all datasets. In Figure 22, we compare the F1 scores obtained by the random forest and the SVM algorithms for the three datasets using a bar graph. The green bars indicate random forest model scores, and the brown bars indicate the values of the SVM model. Observations reveal that random forest outshines the support vector machine technique for all the datasets.

### *B. Cloud Platform*

The above-mentioned four regression and classification models are deployed on the cloud platform. We are using Amazon SageMaker service to deploy the models on the cloud. The Amazon SageMaker provides a Jupyter Notebook where we can execute the same Python code. The results reveal that the cloud platform helps reduce the execution time for all four models, and the time required for data pre-processing can also be reduced. The execution times using personal computer Jupyter Notebook and SageMaker's Jupyter Notebook are given in Table VIII.

Amazon SageMaker offers high scalability, which means it can efficiently address large datasets and modeling techniques. As a result, it is possible to process vast amounts of data more efficiently, which can drastically decrease the processing time. For projects related to machine learning, Amazon SageMaker offers high-performance computing instances. These instances are equipped with powerful CPUs and GPUs, which can considerably speed the implementation of machine learning models, resulting in rapid predictions.

Prediction time can be significantly decreased by running inference on large datasets parallelly using Amazon SageMaker batch transform. This method is especially beneficial when estimating air quality in smart cities, where a massive amounts of information must be analysed. Overall, Amazon SageMaker delivers a number of capabilities and features that can help reduce the execution time for estimating air quality in smart cities. By utilizing these capabilities, we can produce precise air quality predictions quickly and economically.

A graph is plotted for the execution times in seconds of all the models as shown in Figure 23. The x-axis represents the machine learning models, and the y-axis represents the time in seconds. The blue line portrays the execution on the personal computer, and the pink defines execution in Amazon SageMaker. We can see that the execution times are reduced while using a cloud platform which is the key finding of this work. Air quality data is a time series, meaning the values change hourly. A reduced runtime will be beneficial for this type of data. It is also challenging to manage the time series data collected from smart cities because smart cities generate numerous data that require processing time. Cloud computing plays a significant role in situations like this, as it shortens the runtime and aids in storing the vast amounts of data acquired.

## VII. Conclusion

Smart cities aim to improve our lifestyles and solve many problems for different applications. The primary goal of this research is to design a model that can forecast air quality in smart cities utilizing cloud computing and machine learning. To determine air quality, regression, and classification techniques are employed. As regression models, both linear regression and lasso regression are used. We apply the support vector machine and random forest techniques as classification models. We also compute the execution timings for each of the four models and compare them to the times for their cloud-deployed versions. The results demonstrate that lasso regression outshines linear regression among regression techniques, and the random forest algorithm outperforms the support vector machine approach among classification models.

Furthermore, these findings show that when models are executed on the Amazon SageMaker rather than a desktop computer, run-time is reduced. Additionally, accuracy is maintained while execution time is reduced. Amazon SageMaker offers a distributed computing architecture that includes many compute instances that collaborate in a distributed manner, which enables faster processing of large datasets than personal computers. For example, to improve performance and shorten execution times, use a cloud-based solution, Amazon SageMaker, that can dynamically scale the computing resources utilized according to the size of the dataset.

In conclusion, ML could provide an efficient and flexible solution to easily modify the machine learning framework and match their specific requirements despite having pre-defined algorithms and built-in tools. Also, instance type selection and the development of custom training algorithms are easily attainable. We can minimize execution time and improve quality by adapting distributed cloud-based infrastructure for a safe and sustainable smart cities environment.

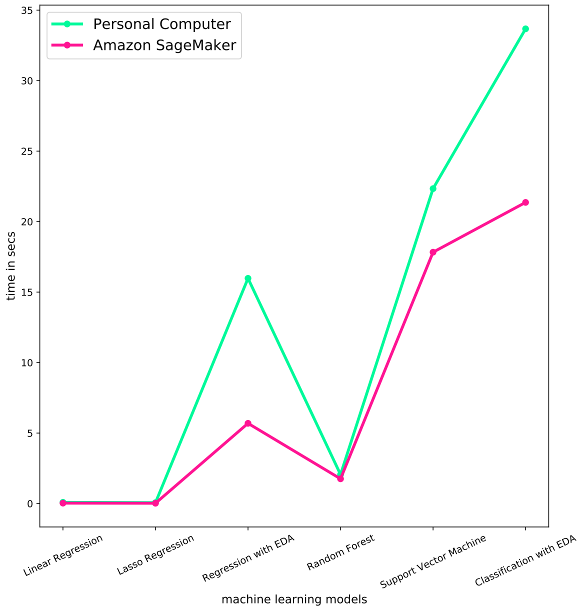

Fig. 23: Execution Time in seconds

## VIII. Future Work

As a future work, we will apply regression and classification models and deploy them on other cloud-based platform. We will employee other cloud based service to deploy secure and privacy preserving efficient ML models on other cloud platforms in addition to Amazon cloud. We aim to build secure and efficient cloud platform to helps reduce the security risks and enhance the performs, i.e. execution time for the developed ML models and the time required for data pre-processing can also be reduced.

## References

[1] H. Chourabi, S. Walker, J. R. Gil-Garcia, and H. J. Scholl, "Understanding smart cities: An integrative framework," *2012 45th Hawaii International Conference on System Sciences*, pp. 2289–2297, 2012.

[2] T. Singh, A. Solanki, S. K. Sharma, A. Nayyar, and A. Paul, "A decade review on smart cities: Paradigms, challenges and opportunities," *IEEE Access*, vol. 10, pp. 68 319–68 364, 2022.

[3] S. R. Garzon, S. Walther, S. Pang, B. Deva, and A. Küpper, "Urban air pollution alert service for smart cities," in *Proceedings of the 8th International Conference on the Internet of Things*, ser. IOT '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3277593.3277599

[4] T. M. Ghazal, M. K. Hasan, M. T. Alshurideh, H. M. Alzoubi, M. Ahmad, S. S. Akbar, B. Al Kurdi, and I. A. Akour, "Iot for smart cities: Machine learning approaches in smart healthcare—a review," *Future Internet*, vol. 13, no. 8, 2021. [Online]. Available: https://www.mdpi.com/1999-5903/13/8/218

[5] N. Shahid, M. A. Shah, A. Khan, C. Maple, and G. Jeon, "Towards greener smart cities and road traffic forecasting using air pollution data," *Sustainable Cities and Society*, vol. 72, p. 103062, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210670721003462

[6] K. Mehmood, Y. Bao, Saifullah, W. Cheng, M. A. Khan, N. Siddique, M. M. Abrar, A. Soban, S. Fahad, and R. Naidu, "Predicting the quality of air with machine learning approaches: Current research priorities and future perspectives," *Journal of Cleaner Production*, vol. 379, p. 134656, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652622042287

[7] D. Iskandaryan, F. Ramos, and S. Trilles, "Air quality prediction in smart cities using machine learning technologies based on sensor data: A review," *Applied Sciences*, vol. 10, no. 7, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/7/2401

[8] L. Zhihan, C. Dongliang, L. Ranran, and W. Qingjun, “Intelligent edge computing based on machine learning for smart city,” *Future Generation Computer Systems*, vol. 115, pp. 90–99, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20306889

[9] X. Jiang, R. Wang, T. Chang, Y. Zhang, K. Zheng, R. Wan, and X. Wang, “Effect of short-term air pollution exposure on migraine: A protocol for systematic review and meta-analysis on human observational studies,” *Environment International*, p. 107892, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0160412023001654

[10] M. Y. Thu, W. Htun, Y. L. Aung, P. E. E. Shwe, and N. M. Tun, “Smart air quality monitoring system with lorawan,” in *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, 2018, pp. 10–15.

[11] N. M. Kumar, S. Goel, and P. K. Mallick, “Smart cities in india: Features, policies, current status, and challenges,” in *2018 Technologies for Smart-City Energy Security and Power (ICSESP)*, 2018, pp. 1–4.

[12] S. Duangsuwan, A. Takarn, R. Nujankaew, and P. Jamjareegulgarn, “A study of air pollution smart sensors lpwan via nb-iot for thailand smart cities 4.0,” in *2018 10th International Conference on Knowledge and Smart Technology (KST)*, 2018, pp. 206–209.

[13] L. Bai, J. Wang, X. Ma, and H. Lu, “Air pollution forecasts: An overview,” *International Journal of Environmental Research and Public Health*, vol. 15, no. 4, 2018. [Online]. Available: https://www.mdpi.com/1660-4601/15/4/780

[14] W. contributors, “Air quality index — Wikipedia, the free encyclopedia,” 2023. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Air_quality_index

[15] M. Bisht and K. R. Seeja, “Air pollution prediction using extreme learning machine: A case study on delhi (india),” in *Proceedings of First International Conference on Smart System, Innovations and Computing*. Singapore: Springer Singapore, 2018, pp. 181–189.

[16] T. E. Karakasidis, F. Sofos, and C. Tsonos, “The electrical conductivity of ionic liquids: Numerical and analytical machine learning approaches,” *Fluids*, vol. 7, no. 10, 2022. [Online]. Available: https://www.mdpi.com/2311-5521/7/10/321

[17] Y. Rybarczyk and R. Zalakeviciute, “Machine learning approaches for outdoor air quality modelling: A systematic review,” *Applied Sciences*, vol. 8, no. 12, 2018. [Online]. Available: https://www.mdpi.com/2076-3417/8/12/2570

[18] Z. Qi, T. Wang, G. Song, W. Hu, X. Li, and Z. Zhang, “Deep air learning: Interpolation, prediction, and feature analysis of fine-grained air quality,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 12, pp. 2285–2297, 2018.

[19] E. Gambhir, R. Jain, A. Gupta, and U. Tomer, “Regression analysis of covid-19 using machine learning algorithms,” in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 2020, pp. 65–71.

[20] G. E. Kulkarni, A. A. Muley, N. K. Deshmukh, and P. U. Bhalchandra, “Autoregressive integrated moving average time series model for forecasting air pollution in nanded city, maharashtra, india,” 2018, pp. 1435–1444. [Online]. Available: https://doi.org/10.1007/s40808-018-0493-2

[21] S. Kavitha, S. Varuna, and R. Ramya, “A comparative analysis on linear regression and support vector regression,” in *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, 2016, pp. 1–5.

[22] M. Carney, B. Webster, I. Alvarado, K. Phillips, N. Howell, J. Griffith, J. Jongejan, A. Pitaru, and A. Chen, “Teachable machine: Approachable web-based tool for exploring machine learning classification,” in *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, ser. CHI EA ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–8. [Online]. Available: https://doi.org/10.1145/3334480.3382839

[23] P. C. Sen, M. Hajra, and M. Ghosh, “Supervised classification algorithms in machine learning: A survey and review,” in *Emerging Technology in Modelling and Graphics*, J. K. Mandal and D. Bhattacharya, Eds. Singapore: Springer Singapore, 2020, pp. 99–111.

[24] J. Wiley and L. Sons, *Introduction to Machine Learning*, 2019, ch. 1, pp. 1–25. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119556749.ch1

[25] Rubal and D. Kumar, “Evolving differential evolution method with random forest for prediction of air pollution,” *Procedia Computer Science*, vol. 132, pp. 824–833, 2018, international Conference on Computational Intelligence and Data Science. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918308263

[26] N. U. Khan, M. A. Shah, C. Maple, E. Ahmed, and N. Asghar, “Traffic flow prediction: An intelligent scheme for forecasting traffic flow using air pollution data in smart cities with bagging ensemble,” *Sustainability*, vol. 14, no. 7, 2022. [Online]. Available: https://www.mdpi.com/2071-1050/14/7/4164

[27] Q. Liu, J. Gu, J. Yang, Y. Li, D. Sha, M. Xu, I. Shams, M. Yu, and C. Yang, *Cloud, Edge, and Mobile Computing for Smart Cities*. Springer Singapore, 2021, pp. 757–795. [Online]. Available: https://doi.org/10.1007/978-981-15-8983-6_41

[28] Y.-S. Chang, H.-T. Chiao, S. Abimannan, Y.-P. Huang, Y.-T. Tsai, and K.-M. Lin, “An lstm-based aggregated model for air pollution forecasting,” *Atmospheric Pollution Research*, vol. 11, no. 8, pp. 1451–1463, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1309104220301215

[29] D. Schürholz, S. Kubler, and A. Zaslavsky, “Artificial intelligence-enabled context-aware air quality prediction for smart cities,” *Journal of Cleaner Production*, vol. 271, p. 121941, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652620319880

[30] Y. Qian, D. Wu, W. Bao, and P. Lorenz, “The internet of things for smart cities: Technologies and applications,” *IEEE Network*, vol. 33, no. 2, pp. 4–5, 2019.

[31] S. Mehta, B. Bhushan, and R. Kumar, *Machine Learning Approaches for Smart City Applications: Emergence, Challenges and Opportunities*. Springer International Publishing, 2022, pp. 147–163. [Online]. Available: https://doi.org/10.1007/978-3-030-90119-6_12

[32] W. contributors, “Deep learning — Wikipedia, the free encyclopedia,” 2023, [Online; accessed 22-March-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Deep_learning&oldid=1145954968

[33] B. S. Freeman, G. Taylor, B. Gharabaghi, and J. Thé, “Forecasting air quality time series using deep learning,” *Journal of the Air & Waste Management Association*, vol. 68, no. 8, pp. 866–886, 2018. [Online]. Available: https://doi.org/10.1080/10962247.2018.1459956

[34] S. Du, T. Li, Y. Yang, and S.-J. Horng, “Deep air quality forecasting using hybrid deep learning framework,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 6, pp. 2412–2424, 2021.

[35] S. M. Cabaneros, J. K. Calautit, and B. R. Hughes, “A review of artificial neural network models for ambient air pollution prediction,” *Environmental Modelling Software*, vol. 119, pp. 285–304, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364815218306352

[36] S. Ameer, M. A. Shah, A. Khan, H. Song, C. Maple, S. U. Islam, and M. N. Asghar, “Comparative analysis of machine learning techniques for predicting air quality in smart cities,” *IEEE Access*, vol. 7, pp. 128 325–128 338, 2019.

[37] R. Murugan and N. Palanichamy, “Smart city air quality prediction using machine learning,” in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2021, pp. 1048–1054.

[38] M. Castelli, F. M. Clemente, A. Popovič, S. Silva, and L. Vanneschi, “A machine learning approach to predict air quality in california,” p. 8049504, 2020. [Online]. Available: https://doi.org/10.1155/2020/8049504

[39] U. Mahalingam, K. Elangovan, H. Dobhal, C. Valliappa, S. Shrestha, and G. Kedam, “A machine learning model for air quality prediction for smart cities,” in *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 2019, pp. 452–457.

[40] T. W. Ayele and R. Mehta, “Air pollution monitoring and prediction using iot,” pp. 1741–1745, 2018.

[41] B. Ghose and Z. Rehena, “A deep learning approach for predicting air pollution in smart cities,” in *Computational Intelligence and Machine Learning*. Singapore: Springer Singapore, 2021, pp. 29–38.

[42] A. Bekkar, B. Hssina, S. Douzi, and K. Douzi, “Air-pollution prediction in smart city, deep learning approach,” *Journal of Big Data*, vol. 8, p. 161, 2021. [Online]. Available: https://doi.org/10.1186/s40537-021-00548-1

[43] C.-J. Huang and P.-H. Kuo, “A deep cnn-lstm model for particulate matter (pm2.5) forecasting in smart cities,” *Sensors*, vol. 18, no. 7, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/7/2220

[44] I. Kök, M. U. Şimşek, and S. Özdemir, “A deep learning model for air quality prediction in smart cities,” in *2017 IEEE International Conference on Big Data (Big Data)*, 2017, pp. 1983–1990.

[45] H. Samih, “Smart cities and internet of things,” *Journal of Information Technology Case and Application Research*, vol. 21, no. 1, pp. 3–12, 2019. [Online]. Available: https://doi.org/10.1080/15228053.2019.1587572

[46] Y. Mehta, M. Manohara Pai, S. Mallissery, and S. Singh, “Cloud enabled air quality detection, analysis and prediction - a smart city application for smart health,” in *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, 2016, pp. 1–7.

[47] S. Al-Janabi, M. Mohammad, and A. Al-Sultan, “A new method for prediction of air pollution based on intelligent computation,” *Soft Computing*, vol. 24, pp. 661–680, 2020. [Online]. Available: https://doi.org/10.1007/s00500-019-04495-1

[48] D. Zhu, C. Cai, T. Yang, and X. Zhou, “A machine learning approach for air quality prediction: Model regularization and optimization,” *Big Data and Cognitive Computing*, vol. 2, no. 1, 2018. [Online]. Available: https://www.mdpi.com/2504-2289/2/1/5

[49] A. V. Joshi, *Amazon’s Machine Learning Toolkit: Sagemaker*. Cham: Springer International Publishing, 2020, pp. 233–243. [Online]. Available: https://doi.org/10.1007/978-3-030-26622-6_24

[50] E. Liberty, Z. Karnin, B. Xiang, L. Rouesnel, B. Coskun, R. Nallapati, J. Delgado, A. Sadoughi, Y. Astashonok, P. Das, C. Balioglu, S. Chakravarty, M. Jha, P. Gautier, D. Arpin, T. Januschowski, V. Flunkert, Y. Wang, J. Gasthaus, L. Stella, S. Rangapuram, D. Salinas, S. Schelter, and A. Smola, “Elastic machine learning algorithms in amazon sagemaker,” in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 731–737. [Online]. Available: https://doi.org/10.1145/3318464.3386126

[51] N. Rauschmayr, V. Kumar, R. Huilgol, A. Olgiati, S. Bhattacharjee, N. Harish, V. Kannan, A. Lele, A. Acharya, J. Nielsen, L. Ramakrishnan, I. Bhatt, K. Chia, N. Dodda, Z. Li, J. Gu, M. Choi, B. Nagarajan, J. Geevarghese, D. Davydenko, S. Li, L. Huang, E. Kim, T. Hill, and K. Kenthapadi, “Amazon sagemaker debugger: A system for real-time insights into machine learning model training,” in *Proceedings of Machine Learning and Systems*, A. Smola, A. Dimakis, and I. Stoica, Eds., vol. 3, 2021, pp. 770–782.

[52] M. Zıa Ur Rahman, B. V. Vardhan, L. Jenith, V. Rakesh Reddy, S. Surekha, and P. Srinivasareddy, “Adaptive exon prediction using maximum error normalized algorithms,” in *Proceedings of 2nd International Conference on Artificial Intelligence: Advances and Applications*. Springer Nature Singapore, 2022, pp. 511–523. [Online]. Available: https://doi.org/10.1007/978-981-16-6332-1

[53] V. E. Adeyemo, A. Abdullah, N. JhanJhi, M. Supramaniam, and A. O. Balogun, “Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: an empirical study,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 9, 2019.

[54] G. Ghosh, S. Verma, N. Jhanjhi, M. Talib *et al.*, “Secure surveillance system using chaotic image encryption technique,” in *IOP conference series: materials science and engineering*, vol. 993, no. 1. IOP Publishing, 2020, p. 012062.

[55] Z. A. Almusaylim, N. Zaman, and L. T. Jung, “Proposing a data privacy aware protocol for roadside accident video reporting service using 5g in vehicular cloud networks environment,” in *2018 4th International conference on computer and information sciences (ICCOINS)*. IEEE, 2018, pp. 1–5.

[56] H. Shahid, H. Ashraf, H. Javed, M. Humayun, N. Jhanjhi, and M. A. AlZain, “Energy optimised security against wormhole attack in iot-based wireless sensor networks,” *Comput. Mater. Contin*, vol. 68, no. 2, pp. 1967–81, 2021.

[57] S. Sennan, R. Somula, A. K. Luhach, G. G. Deverajan, W. Alnumay, N. Jhanjhi, U. Ghosh, and P. Sharma, “Energy efficient optimal parent selection based routing protocol for internet of things using firefly optimization algorithm,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 8, p. e4171, 2021.

[58] S. J. Hussain, U. Ahmed, H. Liaqat, S. Mir, N. Jhanjhi, and M. Humayun, “Imiad: intelligent malware identification for android platform,” in *2019 International Conference on Computer and Information Sciences (ICCIS)*. IEEE, 2019, pp. 1–6.

**Niveshitha Niveshitha** (Student Member, IEEE) is a Graduate Research Assistant working under the supervision of Dr. Fathi Amsaad, an Assistant Professor of Computer Science and Engineering at Wright State University in Dayton, Ohio, USA. She received a Master of Science degree in Computer Science from Wright State University in Spring 2023. Her research interests include Machine Learning, Parallel Programming, and Distributed Computing.

**Fathi Amsaad** (Senior Member, IEEE) is an Assistant Professor of Computer Science and Engineering at Wright State University, Dayton, Ohio, USA. He received the Bachelor’s degree in Computer Science from the University of Benghazi, Libya, in 2002. He received a dual Master’s degree in Computer Science/ Computer Engineering from the University of Bridgeport, CT, USA, in 2011/ 2012. He received a Ph.D. degree in Engineering with emphases in Computer Science and Engineering from the University of Toledo, OH, USA, in 2017. His research interest include AI Hardware Security, Machine Learning and Trusted AI. Both government and industry fund Dr. Amsaad’s research including NSF, AFRL, AFOSR, Intel, NSA, and Ohio department of education. He has served as an Organizer, Program Chair, Technical Program Committee member, Gust Editor, and on the Reviewer Board for several international conferences and journals. In addition to his research activities, Dr. Amsaad has established teaching experience in hardware security, IoT and embedded systems security, distributed computing, digital systems, and network administration and security curriculum.

**Ahmed Sherif** (Senior Member, IEEE) received the M.Sc. degree in computer science and engineering from the Egypt-Japan University of Science and Technology (E-JUST), in 2014, and the Ph.D. degree in electrical and computer engineering from Tennessee Tech University, Cookeville, TN, USA, in August 2017. He is currently an Assistant Professor with the School of Computing Sciences and Computer Engineering, The University of Southern Mississippi (USM), USA. He is the author of numerous papers published in major IEEE conferences and journals, such as IEEE International Conference on Communications (IEEE ICC), IEEE Vehicular Technology Conference (IEEE VTC), the IEEE Transactions on Dependable and Secure Computing, and the IEEE Internet of Things Journal. His research interests include cybersecurity, security and privacy-preserving schemes in autonomous vehicles (AVs), vehicular ad hoc networks (VANETs), the Internet of Things (IoT) applications, and smart grid advanced metering infrastructure (AMI) networks. He served as a Reviewer for several journals and conferences, such as the IEEE Transactions on Vehicular Technology, the IEEE Internet of Things Journal, and the journal of Peer-to-Peer Networking and Applications.

**Noor Zaman Jhanjhi** (N.Z Jhanjhi) is currently working as a Professor in Computer Science (Cybersecurity), Program Director for the Postgraduate Research Degree Programmes in computer science, Acting Program Director for Master of Applied Computing MAC, Director of the Center for Smart Society (CSS5) at the School of Computer Science at Taylor's University, Malaysia. He has been nominated as the world's top 2% research scientist globally, the top researcher in computer science in Malaysia by Scopus in terms of publications and nominated as an Outstanding faculty member by the MDEC Malaysia for the year 2022. He has highly indexed publications in WoS/ISI/SCI/Scopus, and his collective research Impact factor is more than 900 plus points. His Google Scholar H index is 46, and I-10 Index is close to 202, and his Scopus H index is 36, with more than 550 publications on his credit. He has several international Patents on his account, including Australian, German, and Japanese. He edited/authored over 40 research books published by world-class publishers, including Springer, Taylors and Frances, Willeys, Intech Open, IGI Global USA, etc. He has excellent experience supervising and co-supervising postgraduate students, and more than 33 Postgraduate scholars graduated under his supervision. Prof. Jhanjhi serves as Associate Editor and Editorial Assistant Board for several reputable journals, such as PeerJ Computer Science, CMC Computers, Materials & Continua, Computer Systems Science and Engineering CSSE and Frontier in Communication and Networks. He received Outstanding Associate Editor for IEEE ACCESS. Active reviewer for a series of top-tier journals has been awarded globally as a top 1% reviewer by Publons (Web of Science). He is an external Ph.D./Master thesis examiner/evaluator for several universities globally and evaluated 50 plus theses. He has completed more than 40 internationally funded research grants successfully. He has served as a Keynote/Invited speaker for more than 60 international conferences and chaired international conference sessions internationally. He has vast experience in academic qualifications, including ABET, NCAAA, and NCEAC, for 10 years. His research areas include Cybersecurity, IoT security, Wireless security, Data Science, Software Engineering, and UAVs.