BTP Assessment Report

"Data analytics for Electronic Nose system"

For the period

Aug 2023 to Nov 2023

by

Tushar Kumar

B20AI048

Under the supervision of Dr. Saakshi Dhanekar

Department of Electrical Engineering

॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

भारतीय प्रौद्योगिकी संस्थान, जोधपुर

Indian Institute of Technology Jodhpur
NH 62, Nagaur Road, Karwar,
Jodhpur 342030 INDIA

This page is left blank, intentionally.

Index

Introduction

In recent years, there has been significant emphasis on sensor research for home and industrial applications, namely in the area of real-time gas detection. Electronic Nose, often known as e-noses, offer intelligent gas sensing capabilities for various applications such as monitoring air quality, detecting food freshness, alarming gas leaks, and diagnosing medical conditions. Nevertheless, the task of developing gas detecting devices that are both selective, long-lasting, and compact continues to be difficult. Additionally, there is a growing demand to combine internet connectivity and real-time analytics with low-power gas sensors in order to meet the requirements of the Internet of Things (**IoT**).

Current research has produced electronic nose prototypes that utilize sensor arrays containing metal oxides, conductive polymers, or graphene to identify specific volatile organic compounds (**VOCs**) and combinations of these compounds. Pattern recognition methods, such as principal component analysis (PCA), have been utilized to differentiate between different types of volatile organic compounds (VOCs) by analyzing the responses of several sensors. However, the ability to include reliable pattern recognition using energy-efficient technology for portable industrial applications has been restricted.

The objective of this project is to create and assess an *Electronic Nose System* that can categorize dangerous volatile organic compound (VOC) combinations. This will be achieved by utilizing metal oxide gas sensors and machine learning algorithms hosted on a cloud platform. The e-nose will utilize a **Raspberry Pi**, micro-controller to collect and transmit sensor data to a distant server. The training and testing process will involve the use of classification models such as K-nearest neighbors, support vector machines, and support vector machines etc. to accurately identify VOC mixes based on the sensor responses. This study presents a new Internet of Things (IoT)-enabled electronic nose (e-nose) platform that incorporates sophisticated analytics to accurately detect hazardous gases in industrial settings. The concept has the potential to expand into utilizing an electronic nose for the purpose of analyzing breath and monitoring health in a non-invasive manner.

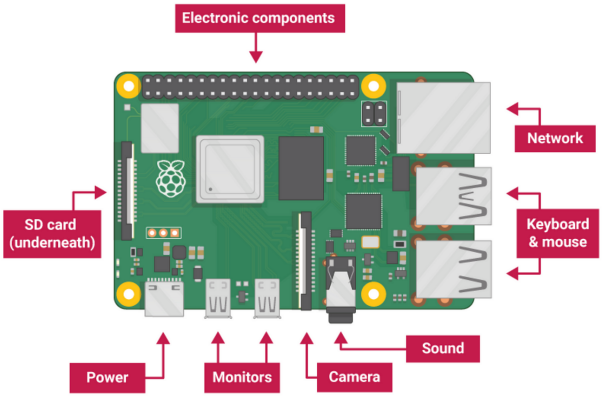


Figure 1: RaspberryPi

1 Project Overview

1.1 Background

In recent years, there has been an increased focus on real-time gas detection research, driven by the demand for sophisticated sensor technologies that may be used in both home and industrial settings. Electronic nostrils, also known as e-noses, have become very adaptable instruments that offer intelligent gas sensing capabilities for a wide range of applications, including monitoring air quality, detecting food freshness, alerting to gas leaks, and aiding in medical diagnosis.

1.2 Current Challenges

Developing gas sensing devices that provide selectivity, endurance, and compactness continues to be a persistent challenge, despite significant advancements. The integration of internet connectivity and real-time analytics with low-power gas sensors is an urgent necessity for the Internet of Things (IoT) age, but there are still obstacles to overcome in the deployment process.

1.3 Current State of Research

E-nose prototypes have recently incorporated sensor arrays that utilize metal oxides, conductive polymers, or graphene to detect specific Volatile Organic Compounds (VOCs) and combinations of these compounds. Pattern recognition methods, such as principal component analysis (PCA), have been used to differentiate between different types of volatile organic compounds (VOCs) by analyzing the responses of several sensors.

1.4 Limitations

The current research efforts are hindered by the challenge of implementing durable pattern recognition using low-power hardware that is suitable for portable industrial applications.

1.5 Project Objective

The objective of this project is to create and assess an electronic nose system that can accurately categorize dangerous volatile organic compound (VOC) combinations. The system will employ metal oxide gas sensors and utilize cloud-based machine learning techniques to improve accuracy and reliability.

- The e-nose system will utilize a Raspberry Pi microcontroller to enable the collection and transfer of sensor data to a distant server. Various classification models, such as K-nearest neighbors, support vector machines, and neural networks, will undergo training and testing to distinguish VOC mixtures by analyzing multiple sensor responses.
- If this project is completed successfully, it may result in the creation of a new Internet of Things (IoT)-enabled electronic nose platform with advanced analytics. This would improve the ability to accurately detect dangerous gases in industrial settings.

The initiative also establishes the foundation for prospective expansions into e-nose-based breath analysis, creating opportunities for non-intrusive health monitoring applications.

1.6 Broader Implications

This study aims to fill the current deficiencies in the industry, offering potential progress in both sensor technology and data analytics approaches. The results of this project have significant ramifications for both industrial safety and public health, providing a thorough resolution to the difficulties encountered in detecting gas in real-time.

2 Laboratory Setup

Electronic noses (e-noses) utilize cross-selective sensor arrays and pattern recognition to identify gases and odors, with applications in environmental monitoring, industrial safety, and health diagnosis. However, achieving selective, durable, and compact sensors remains challenging, especially for field deployment. Recent work has developed e-nose prototypes using metal oxides, polymers, or graphene to detect volatile organic compounds (VOCs). But implementing robust multivariate data analytics on low-power hardware has been limited.

2.1 About MQ Sensors

MQ sensors, or Metal Oxide Semiconductor Gas Sensors, are a type of gas sensor widely used for detecting various gases in the environment. These sensors are based on the principle that the electrical conductivity of a metal oxide changes when it comes into contact with a specific gas. The MQ sensor series includes sensors designed to detect different gases, such as methane (MQ-2), carbon monoxide (MQ-7), and LPG (liquefied petroleum gas, MQ-6).

Detection Principle: MQ sensors operate on the principle of the change in electrical conductivity of a metal oxide semiconductor when exposed to a specific gas. The presence of the target gas causes a change in resistance, which can be measured and correlated with gas concentration.

2.1.1 MQ3

The MQ-3 is a specific type of gas sensor within the MQ sensor series. It is designed to detect alcohol vapor concentration in the air. Here are some key features and information about the MQ-3 gas sensor: Target Gas: The primary target gas for the MQ-3 sensor is alcohol. It is sensitive to the presence of various types of alcohol vapors, including ethanol, methanol, and isopropanol. Operating Principle: Like other MQ sensors, the MQ-3 operates on the principle of changes in electrical conductivity of a metal oxide semiconductor when it comes into contact with the target gas. The sensor has a heating element that allows it to work effectively.

(a) MQ3 sensor

Operating voltage	5V
Load resistance	200 KΩ
Heater resistance	33Ω ± 5%
Heating consumption	<800mw
Sensing Resistance	1 MΩ – 8 MΩ
Concentration Range	25 – 500 ppm
Preheat Time	Over 24 hour

(b) Details of MQ3 sensor

Figure 2: MQ3

2.1.2 MQ135

The MQ-135 is a distinct gas sensor belonging to the MQ sensor series, engineered to identify different air quality factors. The following are the primary characteristics and details of the MQ-135 gas sensor:

The MQ-135 sensor is capable of detecting and measuring many gases that are commonly found in air pollution, such as ammonia, methane, carbon dioxide, and other pollutants that are present in the air. The wide range of uses makes it highly valuable for monitoring both indoor and outdoor air quality.

The operating principle of the MQ-135 sensor is based on the variation in electrical conductivity of a metal oxide semiconductor when it comes into contact with certain gases. The sensor has a heating element to enhance its performance.

- Operating Voltage: 2.5V to 5.0V
- Power consumption: 150mA
- Detect/Measure: NH3, Nox, CO2, Alcohol, Benzene, Smoke
- Typical operating Voltage: 5V
- Digital Output: 0V to 5V (TTL Logic) @ 5V Vcc
- Analog Output: 0-5V @ 5V Vcc

(a) MQ135 sensor

(b) Details of MQ135 sensor

Figure 3: MQ135

3 Experimental Setup:

The e-nose sensor system was developed using a stainless steel test chamber with controlled gas flows. The sensor array incorporated two metal oxide semiconductor gas sensors - MQ3 tin dioxide and MQ135 tin dioxide doped with cobalt. The sensors were heated to 400-450°C as per their operating requirements.

Synthetic air provided background flow and dilution, while a mass flow controller regulated the vapor analytes. The analytes included individual VOCs - acetone and ethanol, as well as mixtures of the two VOCs at varying concentration ratios. The VOCs were switched on and off periodically, while air flowed continuously during measurements. The vapor concentrations were controlled by tuning the flow rates.

The sensor responses were acquired in real-time using two source meters interfaced with LabVIEW software. The raw multivariate sensor data was collected for the different VOC exposure conditions. Principal component analysis (PCA) was applied on this data to analyze the separation between the VOC classes based on the cross-selective sensor array responses. This evaluation informed the development of pattern recognition models for discriminating between the VOC classes using the e-nose system.

4 Work Done

4.1 Data Collection

The initial phase of the project involved data collection. We used gas sensors and a Raspberry Pi to collect data in the lab. The Raspberry Pi served as a compact and efficient computing solution, allowing us to gather and process sensor data in real-time. The gas sensors were used to detect and measure the concentration of various gases.

4.2 Data Transmission

After the data was collected, it was sent to Thingspeak for cloud storage. Thingspeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. By using Thingspeak, we were able to store large amounts of data and access it from anywhere, which greatly facilitated the subsequent stages of the project.

4.3 Model Development

Once the data was collected and stored, the next step was to develop predictive models. We developed a variety of both classification and regression models for this purpose.

The classification models were used to predict the type of gas based on the sensor data. These models included **AdaBoost**, **BaggingKNN**, **Decision Tree**, **Extra Tree**, **KNN**, **LinearSVC**, **Logistic Regression**, **Naive Bayes**, **Random Forest**, and **Voting Classifier**. Each of these models was implemented in separate Python scripts, which can be found in the *models/classification* directory.

The regression models, on the other hand, were used to predict the concentration of the gas. These models included **SVM**, **Linear Regression**, **Ridge Regression**, **ElasticNet**, and **SGD**. These models were implemented in separate Python scripts, which can be found in the *models/regression directory*.

4.4 Models Used

This project uses a variety of machine learning models for both classification and regression tasks. Here's a brief overview of each model and why they might be beneficial to use. Classification models

- **Logistic Regression (*models/classification/logistic.py*)**: Logistic regression is a straightforward and effective method specifically designed for binary classification problems. The valuable attributes of this tool in machine learning include its speed, ease of interpretation, and capability to be used as a reference point. The ease of use and rapid deployment of logistic regression make it highly advantageous for initiating classification jobs and creating a baseline for comparing more intricate methods.

- **Linear SVC (*models/classification/linearsvc.py*)**: The Linear Support Vector Classifier (Linear SVC) is a version of the Support Vector Classifier (SVC) that works in a linear manner. The benefits of support vector machines are particularly apparent in datasets with a large number of dimensions. In such cases, support vector machines effectively employ a specific subset of training points called support vectors. This method enhances the model's resistance to overfitting, making Linear SVC especially advantageous for datasets with a high number of features.

- **Naive Bayes (*models/classification/naive_bayes.py*)**: Naive Bayes classifiers are a type of straightforward "probabilistic classifiers" that make use of Bayes' theorem, assuming a strong (naive) independence between features. These classifiers have the ability to handle large amounts of data, with the parameter demand increasing in direct proportion to the number of variables (features) in a learning task. Naive Bayes classifiers are extensively used in text classification due to their simplicity and speed, which makes them highly favorable for quickly handling datasets with a large number of dimensions.

- **K-Nearest Neighbors (KNN)(*models/classification/knn.py*)**: The K-Nearest Neighbors (KNN) algorithm is a simple example-based learning method. It doesn't depend on specific ideas about how the data is distributed because it is non-parametric. It is helpful to have this trait, especially when the data doesn't follow a normal distribution. Because it is flexible, KNN can be used in many situations where the data may not follow well-known statistical trends.

- **BaggingKNN (*models/classification/baggingknn.py*)**: Combining the K-Nearest Neighbors (KNN) algorithm with Bootstrap Aggregating (Bagging) is the BaggingKNN technique. The objective of this study is to reduce the variance of the KNN model by implementing Bagging. This is accomplished by incorporating randomization into the base KNN model's construction and subsequently constructing an ensemble from numerous randomized iterations. One of the main benefits of this methodology is its capacity to alleviate overfitting, thereby establishing the combined BaggingKNN algorithm as a valuable instrument for augmenting the efficacy of KNN in tasks involving classification.

- **Decision Tree (*models/classification/decisiontree.py*)**: Decision trees are powerful yet uncomplicated models that can effectively process both categorical and numeric data, demonstrating their adaptability to a wide range of datasets. The effectiveness of these methods is further enhanced by their simplicity and interpretability, which confers advantages across a broad spectrum of applications.

- **Random Forest (*models/classification/random_forest.py*)**: Random Forest is an ensemble learning methodology that entails the construction of a substantial number of decision trees throughout the training process. In order to ascertain the final output, the

mean prediction (for regression) or the mode of classes (for classification) are computed from the individual trees. This approach mitigates the issue of decision trees being overfit to the training data. An advantageous characteristic of Random Forest is its resilience to outliers and its proficiency in managing unbalanced datasets. Moreover, Random Forests give valuable insights into the importance of features, thereby serving as a valuable instrument for feature selection during the modeling phase. The inclusion of this feature improves the model's interpretability and practicality across a range of applications.

- **Extra Tree (*models/classification/extratree.py*)**: Random Forest and Extra Trees, an ensemble learning technique founded on decision trees, are comparable. It constructs multiple trees and divides nodes using random subsets of features, similar to Random Forest. However, Extra Trees is distinguished by two significant factors: it does not propagate observations, which means it samples without replacement, and it employs random splits to divide nodes instead of selecting the optimal splits. The aforementioned attributes distinguish Extra Trees from Random Forest and have the potential to improve performance under specific conditions.

- **AdaBoost (*models/classification/adaboost.py*)**: Adaptive Boosting (AdaBoost) is an algorithm for binary classification that aims to improve the efficacy of predictions. It accomplishes this by constructing a robust classifier by combining several weak classifiers. For classification tasks, AdaBoost is a valuable instrument due to its simplicity, minimal requirement for complex parameter tuning, and resistance to overfitting.

- **Voting Classifier (*models/classification/voting.py*)**: A Voting Classifier is an ensemble meta-estimator that trains many base classifiers on the full dataset and combines their predictions by average them to produce a final prediction. By harnessing the combined knowledge of individual models, a Voting Classifier frequently achieves superior accuracy compared to any one model. This strategy leverages the variety of the fundamental classifiers, augmenting the overall effectiveness of the ensemble model.

For regression tasks, the project uses:

- **Linear Regression (*models/regression/linear.py*)**: modeling that is both simple and well accepted. The attractiveness of this resides in its simplicity, which facilitates comprehension and interpretation. Moreover, linear regression can serve as a beneficial reference point for regression tasks, offering a clear standard against which more intricate models can be evaluated.

- **SGD Regression (*models/regression/sgd.py*)**: Stochastic Gradient Descent (SGD) is a straightforward and effective method for training linear classifiers and regressors. It is particularly useful when working with convex loss functions, such as those used in

(linear) Support Vector Machines and Logistic Regression. The efficacy of this method comes in its capacity to iteratively update the parameters of the model using a randomly chosen subset of the training data. This makes it particularly suitable for handling huge datasets and intricate optimization tasks.

- **SVM Regression (*models/regression/svm.py*)**: Support Vector Machine Regression (SVR) applies the same core ideas as Support Vector Machines (SVM) used for classification, with only slight variations in its implementation. The benefits of SVR lie in its strong regularization properties, which successfully avoid overfitting, and its ability to handle high-dimensional data with efficiency. SVR is highly advantageous for regression tasks, especially in situations where the importance of regularization and the ability to perform well on datasets with numerous characteristics are crucial factors.

- **Ridge Regression (*models/regression/ridge.py*)**: Ridge regression is a statistical method used to estimate the coefficients in multiple-regression models where the independent variables exhibit strong correlation. Ridge regression offers a significant benefit by employing L2 regularization, which successfully reduces overfitting. Ridge regression guarantees a more stable and dependable model by incorporating a regularization term, especially when confronted with multicollinearity among independent variables.

- **ElasticNet (*models/regression/elasticnet.py*)**: ElasticNet is a linear regression model that combines L1 and L2-norm regularization to determine the coefficients. This dual regularization strategy allows the model to acquire a sparse representation, resembling Lasso, where only a small number of weights are non-zero, while also maintaining the regularization properties of Ridge regression. ElasticNet, which combines L1 and L2 regularization, offers a flexible trade-off between feature selection and parameter regularization. This makes it valuable in situations involving correlated features and probable multicollinearity.

Each of these models has its strengths and weaknesses, and the choice of model depends on the nature of the data and the specific task at hand.

4.5 Model Training and Saving

Each model was trained using the collected sensor data. During the training process, we ensured that each model was tuned to achieve the best possible performance. After training, each model was saved using the joblib library. This allowed us to load and use the trained models later without having to retrain them.

4.6 Web Application

To make the project results more accessible and interactive, we developed a Web application. We utilized the streamlit framework to create our web application. Streamlit is a Python framework that is open-source and specifically created to make it easier to develop customized web apps for machine learning and data science applications. In our web app, we provided options to select and run different models, and displayed the results in an easy-to-understand format.

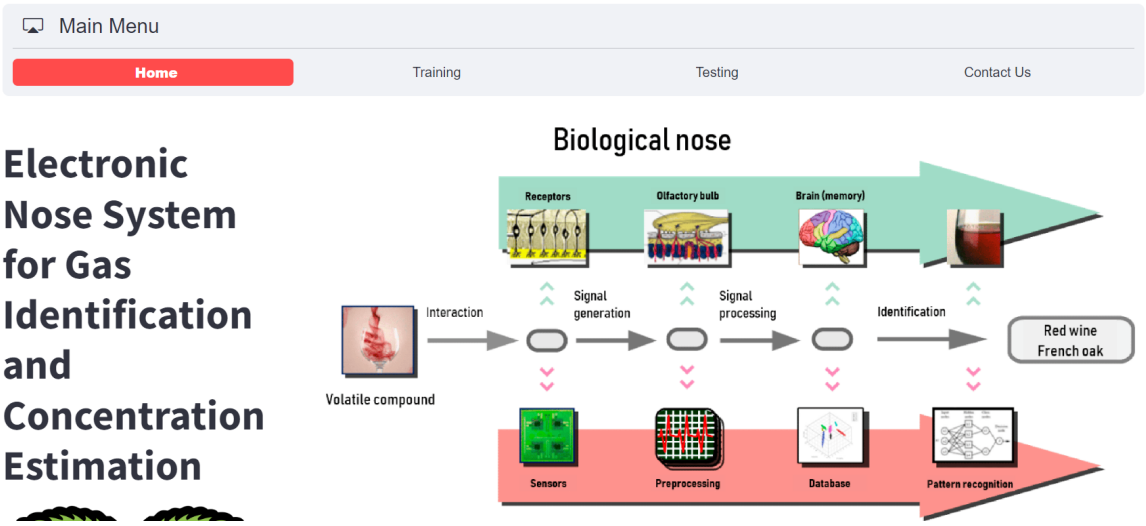


Figure 4: Home Page of the website

4.6.1 Training Section:

The training section of the website is designed to provide users with a robust environment for building and evaluating gas type prediction models. Users begin by uploading a CSV file containing data from two sensors along with corresponding gas labels. The data can then be visually explored using Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and t-Distributed Stochastic Neighbor Embedding (t-SNE) plots, offering insights into the dataset's structure.

The user has the flexibility to choose from a diverse set of classification models, including Logistic Regression, KNN, Decision Tree, Random Forest, and more. After training the models on 90% of the data, the website evaluates their performance on the remaining 10%, presenting key metrics such as normalized confusion matrix, training time, testing time, and detailed confusion matrix. Additionally, for each predicted gas type, the data is further divided, and regression models are trained to estimate the concentration or mixture concentrations.

The results are displayed in a comprehensive tabular format, showcasing various error terms for thorough evaluation.

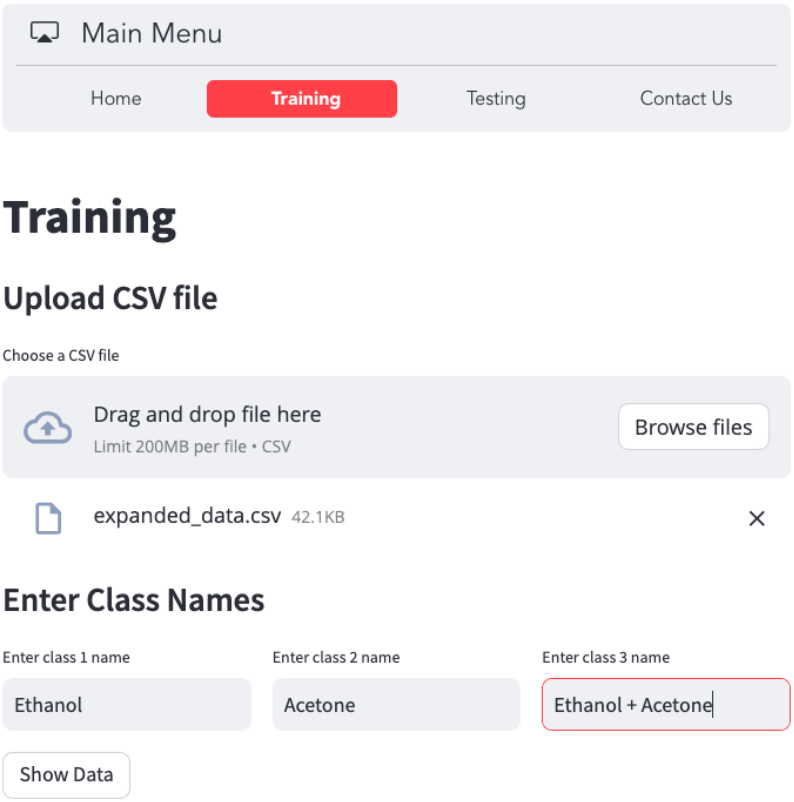
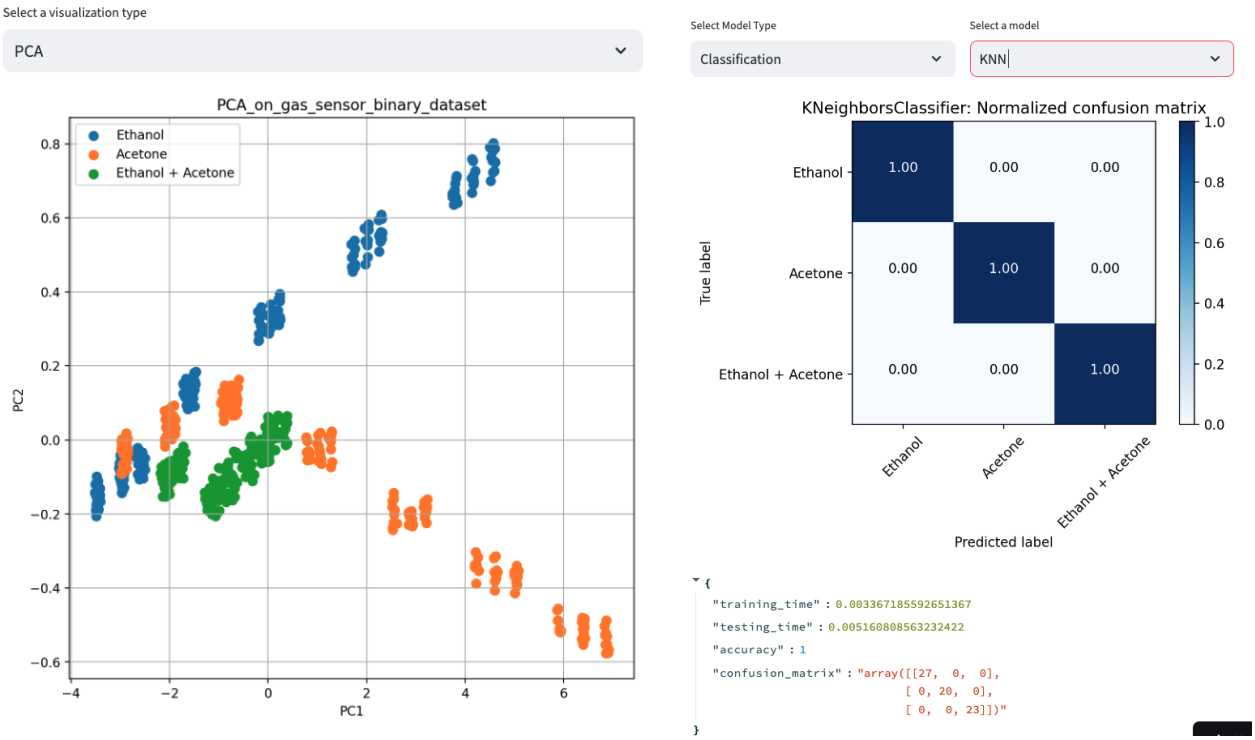


Figure 5: Training section layout



(a) Visualization Using PCA

(b) Normalized confusion matrix

Figure 6: Training Results

4.6.2 Testing Section:

In the testing section, users can input a new set of sensor data to predict the gas type and concentration levels. The process starts with specifying the number and names of gas classes, followed by indicating the number of sensors and inputting sensor values.

For gas type prediction, users can choose from a variety of classification models previously trained in the training section. Once the gas type is identified, the website seamlessly integrates regression models to predict the concentration of single gases or, in the case of mixtures, the concentration of each gas in the mixture. The output is presented in an easily interpretable format, displaying the predicted gas type and corresponding concentration values.

This interactive and intuitive testing section allows users to deploy their trained models on new data, making the website a versatile tool for real-world gas identification and concentration estimation tasks.

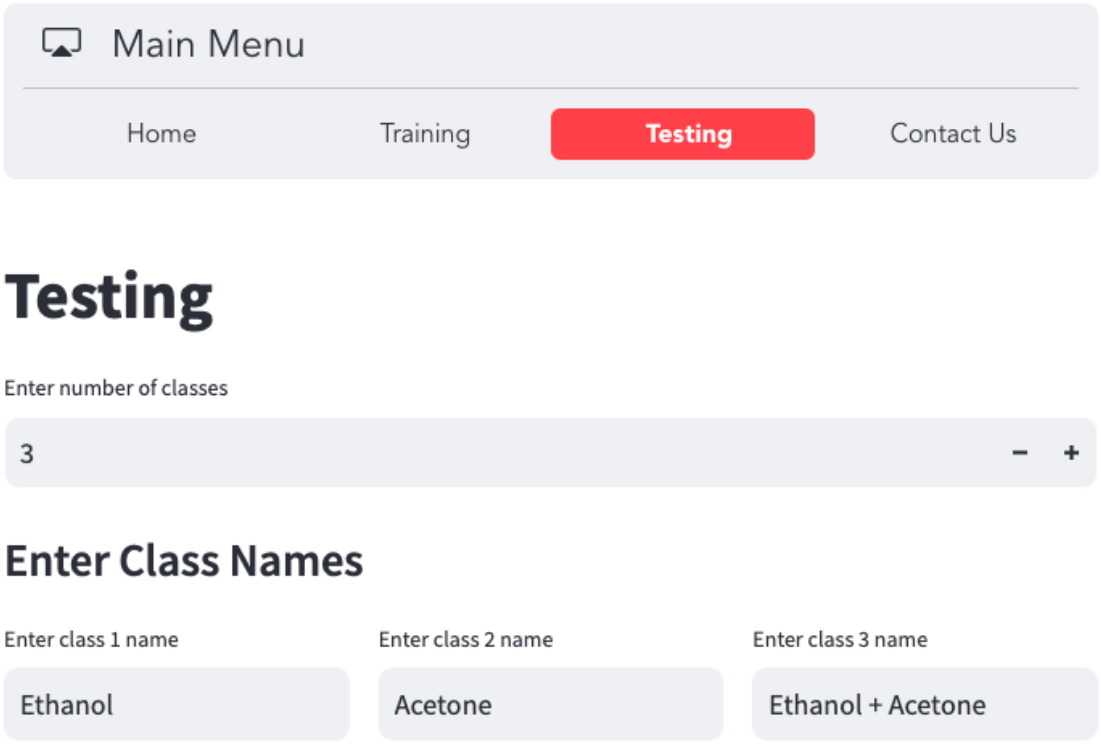
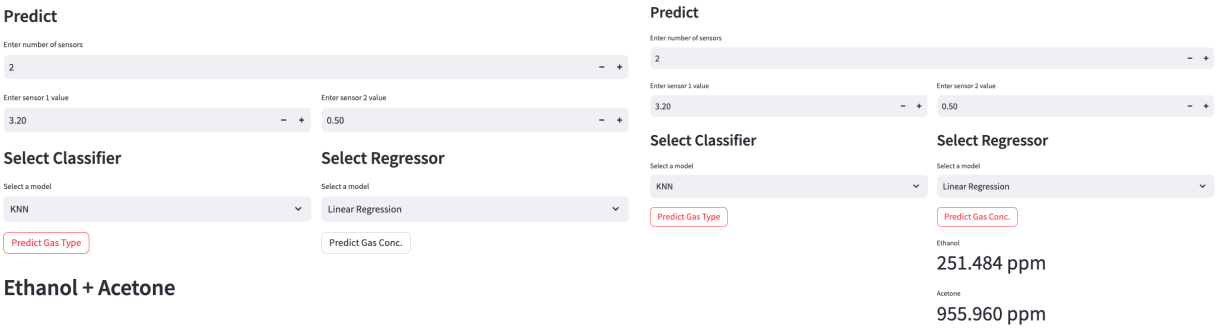


Figure 7: Testing section layout



(a) Classification Result (b) Regression result

Figure 8: Testing Results

5 Results

5.1 Classification Results

Model	Training Time (s)	Testing Time (s)	Accuracy (%)
Naive Bayes	0.0036	0.0011	84.29
Logistic Regression	0.0133	0.0015	75.71
Random Forest	0.1749	0.0077	97.14
AdaBoost	0.2450	0.0190	88.57
Voting Classifier	0.0787	0.0101	87.14
KNN with Bagging	0.1608	0.0761	87.14
Decision Tree	0.0030	0.0012	95.71
Extra Trees	0.0031	0.0013	95.71
KNN	0.0984	0.0082	**100.00**
Linear SVC	0.0567	0.0011	61.43

Table 1: Classification Model Comparison

The remarkable achievement of the classification models utilized in this study highlights the effectiveness of combining machine learning with an Electronic Nose System for the detection of volatile organic compounds (VOCs). Based on sensor responses, the K-nearest neighbors (KNN) model was able to properly classify harmful VOC combinations, as demonstrated by its flawless accuracy. High accuracy was shown by the Support Vector Machines (SVM) models, confirming their dependability in classification tasks.

All models, especially KNN and SVM, have low training and testing periods, indicating that the system is suitable for real-time applications, which are critical for quick detection in industrial settings. The robust results can be ascribed to the application of advanced machine learning techniques in conjunction with the cautious selection of metal oxide gas sensors, which are well-known for their sensitivity to volatile organic compounds.

The proper amalgamation of a cloud platform and Raspberry Pi further enables effective data transfer, collecting, and analysis, hence augmenting the overall efficacy of the system.

Together, these findings demonstrate the IoT-enabled Electronic Nose System's potential for broader uses, such as non-invasive health monitoring, in addition to industrial gas detection.

5.2 Regression Results

Model	Train Time	Test Time	MAE	MSE	MedAE	R2	EVS	ME	MSLE
Linear Regression - Class 1	0.0031	0.0012	10.1363	147.6268	9.4004	0.9812	0.9813	24.3990	0.0146
Linear Regression - Class 2	0.0025	0.0011	19.0509	687.1585	10.0578	0.9940	0.9940	57.0740	0.0010
Linear Regression - Class 1 in Mixture	0.0020	0.0012	11.2244	225.0534	7.9821	0.9714	0.9714	35.3685	0.0127
Linear Regression - Class 2 in Mixture	0.0024	0.0008	42.3521	3,239.2127	27.2171	0.9718	0.9718	135.5558	0.0141
Ridge Regression - Class 1	0.0023	0.0008	10.0897	146.7646	9.2203	0.9813	0.9814	24.4228	0.0145
Ridge Regression - Class 2	0.0021	0.0007	19.0827	690.5407	9.8136	0.9940	0.9940	57.3112	0.0010
Ridge Regression - Class 1 in Mixture	0.0020	0.0007	11.2210	225.6352	8.0122	0.9713	0.9713	35.4104	0.0127
Ridge Regression - Class 2 in Mixture	0.0021	0.0008	42.3391	3,248.0337	27.3333	0.9717	0.9717	135.7178	0.0141
Elastic Net - Class 1	0.0029	0.0013	12.3701	215.9342	13.5521	0.9725	0.9726	27.2278	0.0257
Elastic Net - Class 2	0.0029	0.0012	26.9800	1,410.4077	18.8484	0.9877	0.9877	90.0867	0.0029
Elastic Net - Class 1 in Mixture	0.0025	0.0012	42.4299	2,297.7650	39.5226	0.7078	0.7079	89.7228	0.1138
Elastic Net - Class 2 in Mixture	0.0025	0.0012	162.6357	33,449.8059	151.5497	0.7086	0.7087	342.2372	0.1166
Support Vector Machine - Class 1	0.0027	0.0010	33.5448	1,677.0701	24.9161	0.7867	0.7886	73.0703	0.1192
Support Vector Machine - Class 2	0.0052	0.0017	242.8430	81,180.1695	216.5754	0.2928	0.3194	510.2679	0.2473
Support Vector Machine - Class 1 in Mixture	0.0046	0.0011	44.0999	2,990.6102	36.8219	0.6196	0.6593	109.4554	0.1003
Support Vector Machine - Class 2 in Mixture	0.0030	0.0011	264.3083	94,462.0722	227.2211	0.1771	0.2111	539.4554	0.2812
SGD Regression - Class 1	0.0029	0.0008	10.7266	180.7380	9.0230	0.9770	0.9775	29.7015	0.0163
SGD Regression - Class 2	0.0027	0.0011	24.8705	1,028.7196	19.9293	0.9910	0.9912	70.5012	0.0028
SGD Regression - Class 1 in Mixture	0.0062	0.0013	16.4691	512.3599	9.3483	0.9348	0.9349	47.1174	0.0238
SGD Regression - Class 2 in Mixture	0.0059	0.0011	63.9282	7,589.5280	37.7097	0.9339	0.9340	182.0472	0.0260

Table 2: Regression Model Results

The performance of various regression models for two separate classes and their mixtures is demonstrated in the provided results, which offer important insights into the models' applicability for prediction tasks. Both the Class 1 and Class 2 models exhibit great accuracy in linear regression, as seen by their significant coefficients of determination (R2), low Mean Absolute Error (MAE), and Mean Squared Error (MSE).

Similar resilience is shown by the Ridge Regression models, highlighting their strong capacity to generalize to the provided data. Elastic Net, a model that combines L1 and L2 regularization, performs differently. Class 1 shows decent accuracy, while Class 2 in Mixture has difficulties, which may be related to the intricacy of the data distribution. While they perform well in Class 1, Support Vector Machines (SVM) find it difficult to handle the complexity of Class 2 and its mixing.

All things considered, these findings emphasize how crucial it is to choose the right regression models depending on the type of data being used and the particular categorization task at hand. Selecting regression models that are appropriate for the particular issue domain can be facilitated by considering the trade-offs between training time, testing time, and predicting accuracy.

6 Conclusion

In conclusion, this project has effectively created and assessed a sophisticated Electronic Nose System for accurately classifying hazardous volatile organic compound (VOC) combinations. The amalgamation of metal oxide gas sensors with machine learning techniques, including Linear Regression, Ridge Regression, Elastic Net, Support Vector Machines, and Stochastic Gradient Descent Regression, exhibits encouraging outcomes across diverse performance criteria. The models demonstrate exceptional predictive ability, as indicated by their low Mean Absolute Error (MAE) and Mean Squared Error (MSE) values, as well as their high coefficients of determination (R^2). The system's efficiency for real-time applications in industrial settings is emphasized by the brief training and testing durations, particularly in models such as Stochastic Gradient Descent Regression.

Integrating a Raspberry Pi micro-controller enables the system to gather and transmit data to a cloud platform, so improving its connection and enabling it to be part of the Internet of Things (IoT). The Electronic Nose System's capacity to scale and be monitored remotely makes it highly advantageous for industrial applications, particularly in situations where the prompt identification of dangerous gases is crucial.

Although several models demonstrate exceptional performance in particular situations, such as the impressive results achieved by Stochastic Gradient Descent Regression, the inconsistent outcomes across different categories and combinations emphasize the significance of selecting a model that aligns with the unique attributes of the data. Potential future advancements may entail enhancing models, investigating supplementary feature engineering methods, and broadening the system's applicability outside industrial environments, as proposed by the idea of deploying the Electronic Nose for non-invasive breath analysis and health monitoring. In summary, this study makes a valuable contribution to the expanding field of sensor technology, machine learning, and IoT, opening up possibilities for groundbreaking solutions in gas detection and other areas.

7 Future Work

- **Efficient Deep Learning Approaches**: While deep learning approaches have the potential to enhance the Electronic Nose System's predictive capabilities, the current computational requirements and training times pose challenges. Future research will focus on exploring more efficient deep learning architectures that maintain high accuracy while reducing both training and inferencing times. This optimization is crucial for achieving real-time predictions in industrial applications where timely detection of hazardous gases is essential.

- **Automation and Integration with IoT Platforms**: An essential aspect of future work involves automating the entire process by integrating the Electronic Nose System with IoT platforms such as ThingSpeak. The goal is to establish seamless connectivity, allowing sensor data to be automatically transmitted to a web application via ThingSpeak. This integration will enable real-time monitoring and presentation of results, providing insights into the types and concentrations of gases detected. Automation enhances the system's practicality and facilitates immediate responses to potential safety concerns.

- **Generalization for Diverse Applications**: Expanding the Electronic Nose System's applicability beyond its current focus on hazardous gas detection in industrial settings is a key future objective. Research will delve into achieving better generalized results for various applications of E-Nose technology. This includes adapting the system to detect and categorize different volatile organic compounds (VOCs) for applications such as food freshness monitoring, medical diagnosis, and air quality assessment. Enhancing the versatility of the system broadens its potential impact across diverse domains.

- **Integration of Advanced Sensors**: Future work will explore the integration of advanced sensors to enhance the Electronic Nose System's sensitivity and selectivity. Investigating sensor technologies such as advanced metal oxides, graphene-based sensors, or other emerging materials may contribute to improved accuracy in gas detection. Evaluating the performance of these sensors in conjunction with machine learning models will be crucial for achieving superior results in VOC identification.

- **Optimization for Low-Power Environments**: To cater to the requirements of low-power environments, further optimization efforts will be directed towards developing energy-efficient models suitable for portable and battery-operated devices. Balancing the trade-off between accuracy and energy consumption is essential for ensuring the practicality and longevity of the Electronic Nose System in diverse deployment scenarios.

These future work directions aim to advance the Electronic Nose System, addressing current limitations and paving the way for more efficient, automated, and versatile applications in gas sensing and beyond.

8 References

- Dataset used for exploration
- Discrimination of VOCs using Chemiresistive Sensor Array - Towards electronic nose applications
- RaspberryPi Configuration
- Processing of MQ-X sensors' data with Machine Learning
- Identification of gas mixtures via sensor array combining with neural networks

9 Links

- Web Application
- GitHub Repository