Gas Leak Detection Traditional Classification Machine Learning Techniques: A Study with Random Forest and k-Nearest Neighbors

Hardo Triwahyu Septiadi

1. Introduction

Gas leaks pose a significant risk to human health, safety, and the environment, particularly in settings such as coal mines, chemical industries, and residential areas. The timely and accurate detection of harmful gases is essential for preventing accidents and mitigating potential hazards. However, many gaseous emissions are colourless, odorless, and tasteless, making them difficult to detect using human senses. Furthermore, relying on a single sensor for gas detection may not yield accurate results, as various real-world applications demand robust and reliable detection methods. Thus, there is a need for developing efficient gas detection systems that only utilize simple traditional classification machine learning techniques to enhance the accuracy and reliability of gas detection.

In this study, we focus on detecting and identifying gaseous emissions using an array of semiconductor gas sensors, specifically MQ2, MQ3, MQ5, MQ6, MQ7, MQ8, and MQ135. These sensors are sensitive to various gases such as LPG, butane, methane, smoke, ethanol, alcohol, natural gas, carbon monoxide, hydrogen, and air quality indicators like benzene. The motivation behind this study is to improve the performance of gas detection systems by using traditional classification machine learning techniques such as Random Forest (RF) and k-Nearest Neighbors (kNN), while also considering the simplicity of the model without incorporating computer vision-based methods.

The input for this problem is the readings from the seven gas sensors mentioned before. The dataset we have includes a serial number, sensor readings, and a corresponding gas class label. The output will be the prediction of the gas class, which can be one of four categories: Mixture, NoGas, Perfume, or Smoke. We want to make a model that can predict the gas class accurately based on the sensor readings, which will help in detecting dangerous gas emissions early and improve safety and environmental protection.

By looking closely at how well traditional classification methods like RF and kNN work for gas detection, this study hopes to give useful information for making more efficient, accurate, and reliable gas detection systems.

2. Related Work

In this section, we compare the methodology of our study with the multimodal AI-based fusion framework presented by Narkhede P., et al. Their work focuses on the reliable identification and detection of gases using both thermal camera images and an array of gas sensors. The data collected consists of 5200 samples with thermal images and gas sensor sequences of vector size (1 × 7) sensors. They employ Early and Late Fusion techniques for combining the data from these two modalities. Our study is based on the dataset and example machine learning approach provided by Narkhede P., et al., for gas detection and identification using multimodal artificial intelligence-based sensor fusion.

The key difference between the two methodologies is that Narkhede P., et al. utilize deep learning techniques, specifically Long Short-Term Memory (LSTM) networks for feature extraction from gas sensor values and Convolutional Neural Networks (CNN) for feature extraction from thermal camera images. After extracting features, they concatenate them to form a final feature vector, which is then used to train a classifier model. In contrast, our approach focuses on using traditional machine learning techniques, specifically Random Forest (RF) and k-Nearest Neighbors (kNN), for classification. Additionally, our study aims to simplify the model by not incorporating computer vision-based methods, as we do not use thermal camera images for gas detection.

The pros and cons of the two methodologies are as follows:

Methodology	**Pros**	**Cons**
Narkhede P., et al.	1. Higher accuracy and better performance with deep learning techniques.	1. Requires a large number of data samples for effective training.
	2. Enhanced robustness and reliability with two modalities.	2. Computationally expensive and may require dedicated hardware.
	3. Applicable to high-risk applications (e.g., leak detection in chemical plants, explosives).	
Our approach	1. Simpler and easier to implement with traditional machine learning techniques (RF and kNN).	1. Limited performance without incorporating thermal camera images.
	2. Less preprocessing and potentially faster training times.	2. May not achieve the same level of accuracy as deep learning-based techniques.
	3. Provides valuable insights into the effectiveness of traditional classification techniques.	

In conclusion, our study seeks to explore the effectiveness of traditional machine learning techniques, such as RF and kNN, for gas detection and identification, while simplifying the model by not using computer vision-based methods. We can utilize the dataset provided by Narkhede P., et al., and their example machine learning approach as a foundation for our research. By comparing the outcomes of our approach with their multimodal AI-based fusion framework, we aim to better understand the strengths and limitations of traditional classification techniques in the context of gas detection.

3. Dataset and Features

In this study, we utilize the dataset obtained from Narkhede P., et al.'s paper [(1)]. The dataset consists of 6400 samples, which are divided into training, testing, and validation sets using a ratio of 0.8:0.2:0.2, resulting in 3840 samples for training, and 1280 samples each for testing and validation. A sample of the data is shown below:

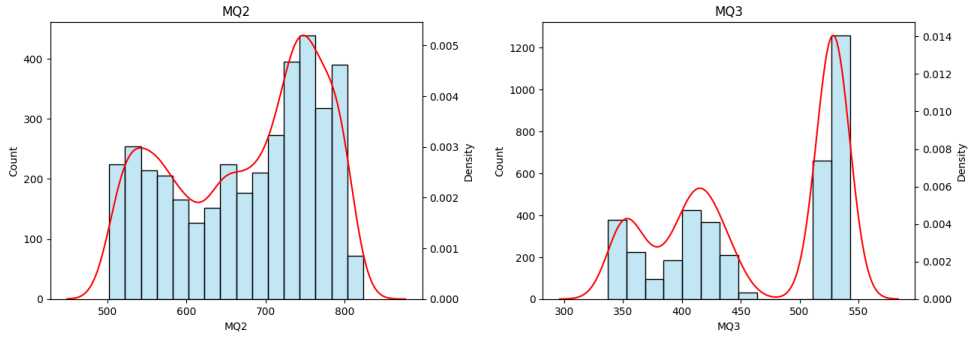
MQ2	**MQ3**	**MQ5**	**MQ6**	**MQ7**	**MQ8**	**MQ135**	**Gas**	**Image Name**
555	515	377	338	666	451	416	NoGas	0_NoGas
555	516	377	339	666	451	416	NoGas	1_NoGas
556	517	376	337	666	451	416	NoGas	2_NoGas
556	516	376	336	665	451	416	NoGas	3_NoGas

556	516	376	337	665	451	416	NoGas	4_NoGas

The dataset comprises values from seven Metal Oxide Semiconductor (MQ) gas sensors (MQ2, MQ3, MQ5, MQ6, MQ7, MQ8, and MQ135), with each sensor being sensitive to different gases. Each sensor's output is an integer value, with the minimum and maximum values ranging from 275 to 824 across all sensors. The target variable is the 'Gas' column, which consists of four classes representing different types of gases. Here is the lists the gas sensors used in the dataset and their corresponding sensitive gases:

Sensor	Sensitive Gas
MQ2	LPG, Butane, Methane, Smoke
MQ3	Smoke, Ethanol, Alcohol
MQ5	LPG, Natural Gas
MQ6	LPG, Butane
MQ7	Carbon Monoxide
MQ8	Hydrogen
MQ135	Air Quality (Smoke, Benzene)

Based on our exploratory data analysis, we found that the distribution of each feature in the dataset is not uniform. Some even have multiple peaks, which can be a major factor to consider when selecting a machine learning model later.



Our data preprocessing steps include data cleaning (outlier removal), data scaling (applying Standard Scaler to both train and test/validation data), and data label encoding (using Label Encoder). Outlier removal is applied due to kNN model's performance being sensitive to outlier data. The Standard Scaler is employed in this project because we use kNN, a classification technique highly sensitive to the scale of input features. Scaling the features ensures that all of them have equal weight when calculating the distance between data points in the kNN algorithm. The Label Encoder is utilized to convert the categorical gas labels into numerical values, making it easier for the classification algorithms to process the data.

In our study, we focus on using gas sensor values for gas detection and identification, excluding the thermal camera images provided in the original dataset. This decision was made to simplify the model and investigate the performance of traditional classification techniques such as RF and kNN when applied solely to the gas sensor data. By doing so, we aim to gain insights into the effectiveness of these techniques in the context of gas detection without relying on computer vision-based methods.

4. Methodology

In this study, we apply two traditional machine learning classification techniques, Random Forest (RF) and k-Nearest Neighbors (kNN), to detect and identify present gas type based on the data collected from gas sensors. We will choose which model performed the best and use that model as our basis in ML Process application (further report).

One of our major reason in choosing to use Random Forest (RF) and k-Nearest Neighbors (kNN) in this project is due to our finding in EDA which is the distribution of each feature in the dataset is not uniform. We will have to experiment with different machine learning algorithms that are capable of handling complex patterns and non-linear relationships between features, such as decision trees, random forests, or neural networks.

Random Forest (RF)

Random Forest is an ensemble learning method that constructs multiple decision trees during the training phase and outputs the majority vote of the individual trees for making predictions. By aggregating the results of several trees, the RF algorithm minimizes the overfitting problem commonly associated with single decision trees.

In classification case, the objective of the RF algorithm is to minimize the Gini impurity or the entropy of the decision trees. In mathematical notation, the Gini impurity for a node i is given by:

$$I_G = 1 - \sum_{j=1}^{c} p_j^2$$

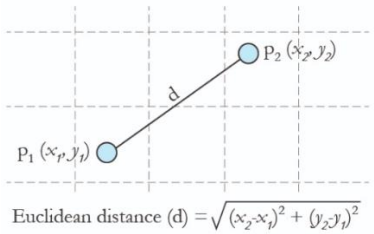
where p_j: proportion of the samples that belongs to class c for a particular node.

The RF algorithm works by creating a bootstrap sample of the training dataset for each tree and then constructing a decision tree based on that sample. During the construction of the decision tree, a random subset of features is considered at each node for splitting. This process introduces randomness and diversity among the trees, resulting in a more robust and accurate model.

k-Nearest Neighbors (kNN)

k-Nearest Neighbors is a non-parametric, instance-based learning algorithm that predicts the class label of a new data point based on the class labels of its k nearest neighbors in the feature space. The algorithm calculates the distance between the new data point and all the training data points using a distance metric, such as Euclidean distance, and then selects the k nearest points. The majority class among these k neighbors is assigned as the predicted class for the new data point.

Mathematically, the Euclidean distance between two data points x and y in an n-dimensional feature space is given by:



where x_i and y_i are the coordinates of points x and y in the i-th dimension.

The kNN algorithm works by finding the k training instances that are closest to the new data point based on the chosen distance metric. It then assigns the most frequent class (majority vote) among these neighbors as the prediction for the new data point. The value of k is a user-defined parameter that controls the trade-off between the model's bias and variance.

Random Forest aiming to reduce errors by increasing randomization (randomizing features) and correlating predictions from multiple trees. In contrast, kNN optimizes the selection of the number of nearest neighbors and the distance metric to maximize classification accuracy.

In this study, we apply both the Random Forest and k-Nearest Neighbors algorithms to the preprocessed gas sensor data to evaluate their effectiveness in detecting and identifying gaseous emissions. We analyze their performance in terms of accuracy, precision, recall, and F1-score and compare their results to understand the strengths and weaknesses of each technique in this particular application.

5. Experiments, Results and Discussion

5.1 Experiments

In this section, we conducted experiments using RF and kNN algorithms with various hyperparameters, used cross-validation to estimate model performance, and evaluated the models using several metrics. This comprehensive approach allows us to select the best model for gas detection and identification using the given dataset.

In our experiments, we use the Random Forest (RF) and k-Nearest Neighbors (kNN) algorithms with various hyperparameters to find the best combination for gas detection and identification. Hyperparameters play a crucial role in determining the performance of the models, and finding the optimal set of hyperparameters is essential for achieving high classification accuracy.

To find the best hyperparameter for each model, we used cross-validation with a 5-fold split to perform model grid search (GridSearchCV). A 5-fold cross-validation was employed, which means the dataset was split into five parts, and the model was trained and tested five times, using a different part for testing each time. This method helps to ensure that the model performance is consistent across different parts of the dataset and reduces the risk of overfitting.

Random Forest

We used a parameter grid for the Random Forest algorithm, which includes the following hyperparameters:

- *n_estimators*: The number of trees in the forest, with values [10, 50, 100, 200].
- *max_depth*: The maximum depth of the tree, with values [None, 10, 20, 30].
- *min_samples_split*: The minimum number of samples required to split an internal node, with values [2, 5, 10].
- *min_samples_leaf*: The minimum number of samples required to be at a leaf node, with values [1, 2, 4].

The optimal hyperparameters for the Random Forest model are to use 50 estimators, with no limit on the maximum depth of the trees, and minimum samples per leaf set to 1 and minimum samples per split set to 2.

k-Nearest Neighbors

We used a parameter grid for the kNN algorithm, which includes the following hyperparameters:

- *n_neighbors*: The number of neighbors to consider, with values [5, 10, 20].
- *weights*: The weight function used in prediction, with values ['uniform', 'distance'].
- *algorithm*: The algorithm to compute the nearest neighbors, with values ['ball_tree', 'kd_tree'].
- *leaf_size*: The leaf size passed to the BallTree or KDTree, with values [10, 30, 50].

The optimal hyperparameters for the kNN model are using a ball_tree algorithm with a leaf size of 10, 5 nearest neighbors, and distance-based weighting.

In this project, we chose accuracy as the primary evaluation metric because it quantifies the proportion of correctly classified instances out of the total instances. In scenarios where the classes are roughly balanced, as is the case with our gas type classification problem, accuracy provides a reasonable measure of overall model performance.

However, accuracy alone might not provide a comprehensive view of the model's performance across different classes, especially when dealing with imbalanced data or when the cost of misclassification varies among the classes. Therefore, we also considered the Classification Report, which includes precision, recall, and F1-score. These metrics provide insights into the model's ability to correctly identify each class while minimizing false positives and false negatives.

Lastly, we used the ROC AUC curve to assess the model's discriminative ability in a one-vs-rest (OvR) setup. This metric provides insights into the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) for each class. The ROC AUC curve enables us to visualize the overall performance of the model across a range of classification thresholds, making it a valuable tool for understanding how well the model distinguishes between different gas types in various operating conditions.

Formulas for the metrics used in this study:

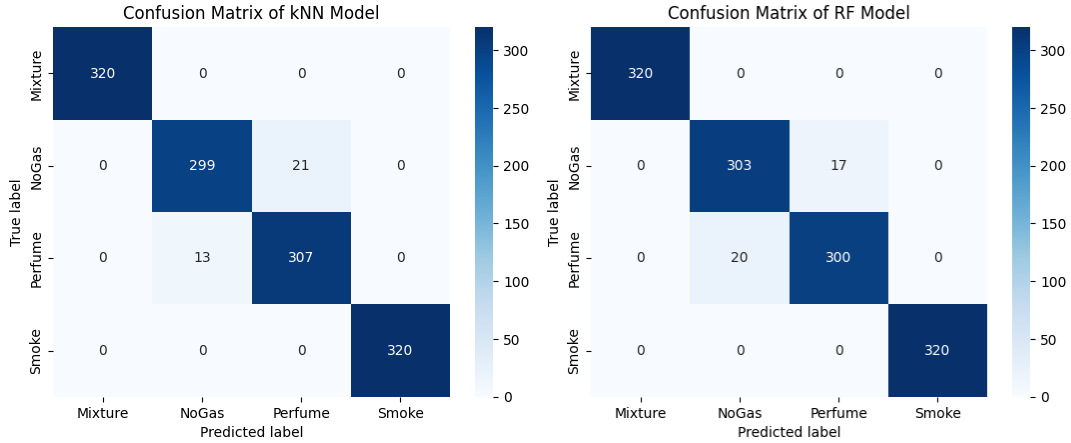
- Precision: TP / (TP + FP), where TP is True Positives, and FP is False Positives.
- Recall: TP / (TP + FN), where FN is False Negatives.
- F-1 Score: 2 * (Precision * Recall) / (Precision + Recall).
- Accuracy: (TP + TN) / (TP + TN + FP + FN)

5.2 Results

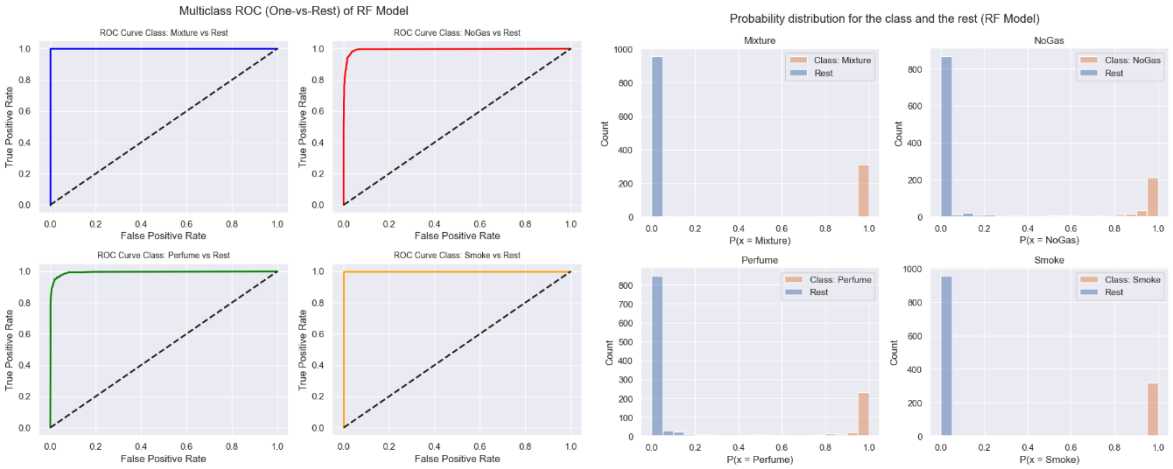
In evaluating two models, we find out that Random Forest model get accuracy 0.9703125 on validation set, with training time 0.263005 seconds. In other side, k-Nearest Neighbors (kNN) model get little better accuracy score 0.971875 on validation set, with much faster training time 0.004025 seconds. This means kNN training time about 65 times quicker than RF model.

The cross-validation scores for the Random Forest model are [0.971, 0.971, 0.959, 0.975, 0.970], with a mean cross-validation accuracy of 0.969. The classifier achieved an accuracy score of 1.0 on the training data and 0.970 on the validation set. For the k-Nearest Neighbors model, the cross-validation scores are [0.976, 0.971, 0.962, 0.964, 0.975], and the mean cross-validation accuracy is 0.970. The classifier also achieved an accuracy score of 1.0 on the training data and 0.972 on the validation set.

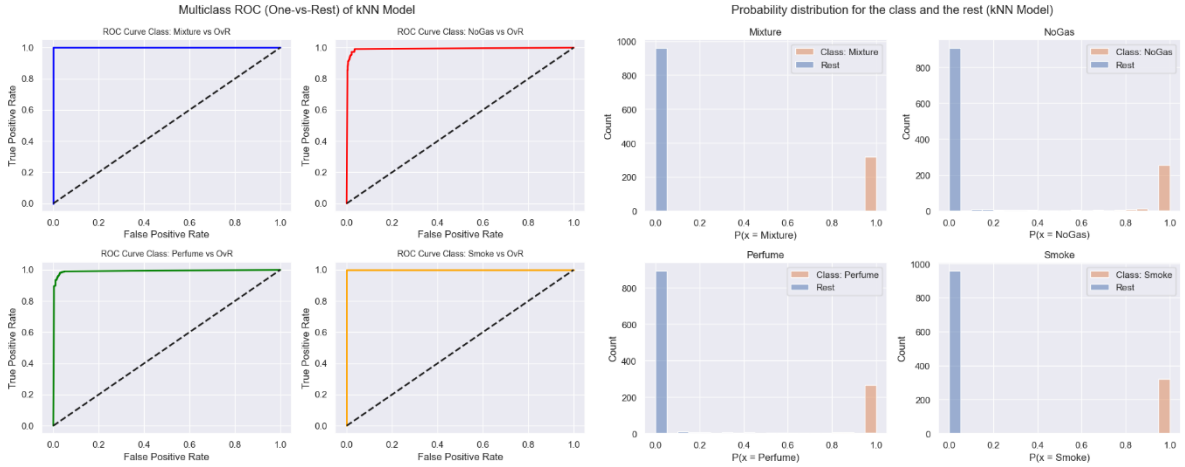
Here are the confusion matrix results based on our testing dataset for the two models under evaluation. On the left side, we have the confusion matrix for the Random Forest model, and on the right side, we present the confusion matrix for the k-Nearest Neighbors model.



Below, we present the ROC AUC analysis results for Random Forest model. As we are working with a multiclass classification problem, each analysis consists of four charts, representing the performance of the model for each gas type versus the rest (One-vs-Rest or OVR) approach. Alongside the ROC AUC charts, we also provide probability distribution plots on the right as supplementary information to help interpret the model's performance.

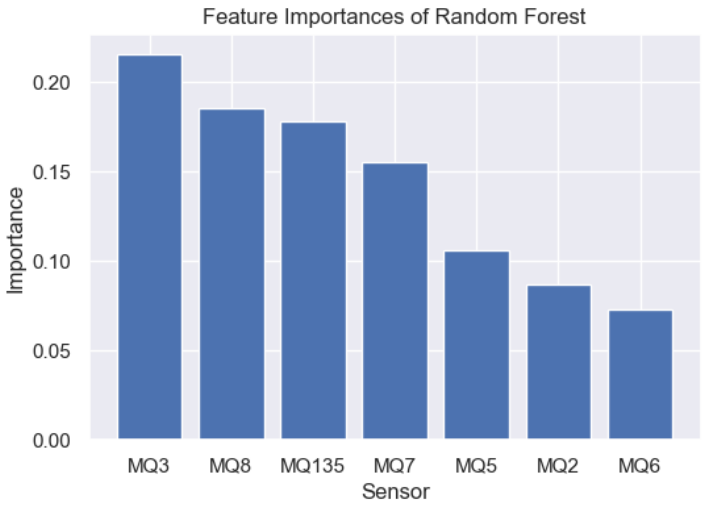


Similar to Random Forest model above, below the equivalent results for the k-Nearest Neighbors (kNN) model.



Next, we will display the feature importance analysis results. Since only the Random Forest model is capable of showcasing feature importance, this analysis is solely based on that model. The first part of the analysis presents the feature importance for each class (gas sensors). To obtain these results, we trained a OneVsRestClassifier. The outcomes are displayed below:

The second part of the analysis focuses on the overall feature importance across all classes, without separating them by individual classes. The results for this analysis are provided below:



5.3 Discussion

In Random Forest model, given the perfect accuracy score on the training data and the slightly lower score on the validation set, there might be a minor overfitting issue, which occurs when a model captures the noise in the data and becomes too complex, resulting in high variance. However, the model's performance on the validation set is still quite high, suggesting that it is generally well-fitted for this problem and bias is relatively low.

Similar to the Random Forest model, the perfect accuracy score on the training data of kNN model might indicate minor overfitting. However, the model's performance on the validation set is close to its cross-validation accuracy, which indicates that the kNN model is generally well-fitted for this classification problem and shows low bias.

In the confusion matrices, for the RF model, the matrix shows 320 correct predictions for class 0, 303 for class 1, 300 for class 2, and 320 for class 3, with some misclassifications between classes 1 and 2. In comparison, the kNN model accurately predicted 320 samples for class 0, 299 for class 1, 307 for class 2, and 320 for class 3, also exhibiting misclassifications between classes 1 and 2, but to a slightly lesser extent. Overall, both models display strong classification performance, with minor differences in their ability to distinguish between classes 1 and 2.

Based on the provided ROC AUC scores, both the k-Nearest Neighbors (kNN) and Random Forest (RF) models show strong performance across the four classes. Although both models exhibit excellent performance, the kNN model has slightly higher ROC AUC scores for classes 1 and 2 compared to the RF model. This suggests that the kNN model may be a better choice for this particular classification problem, as it demonstrates marginally superior performance in differentiating between the four gas types.

After analyzing the feature importance results using the One-vs-Rest approach, we observe the following relationships:

- Class 'Mixture' (Perfume + Smoke) is primarily influenced by MQ7 and MQ8 sensors.
- Class 'Smoke' is predominantly affected by MQ135 and MQ3 sensors, with some contribution from MQ5.

Ideally, Class 'Perfume' should be mainly influenced by MQ3 and MQ135 sensors, as MQ3 detects ethanol and alcohol (present in perfume) and MQ135 detects benzene (potentially found in perfume as VOCs).

On the other hand, Class 'NoGas' exhibits equal feature importance from all sensors, indicating a balanced contribution from each sensor in detecting the absence of gas.

Lastly, based on Random Forest (RF) feature importances data, the most influential sensors are MQ3 (0.2154), MQ8 (0.1853), and MQ135 (0.1782), followed by MQ7 (0.1553) and MQ5 (0.1061). The least impactful sensors are MQ6 (0.0728) and MQ2 (0.0869). These results provide insight into the sensors' contribution to the model's ability to distinguish between the different gas types, helping guide potential improvements or modifications in future iterations of the model or sensor configurations.

6. Conclusion

Considering balance between model performance and training time, we decide k-Nearest Neighbors (kNN) model best choice for this gas type classification project based on sensor values. kNN model not only give small higher accuracy but also much faster in training time compared to Random Forest model. This make kNN more efficient and effective for this case.

For live ML gas classification, kNN will be better solution. It important because faster training time make kNN more suitable when need quick response in real-time application.

Add without having to digest the thermal camera images data, our model still outperforms the REFERENCED model where they use a Neural Network and thermal camera images data.

Based on this research, we have Some recommendations for future work include:

1. **Feature selection**: Utilize various feature selection techniques to identify the most relevant sensors and eliminate redundant or noisy features.
2. **Sensor fusion**: Explore combining data from multiple sensors to create new features that better capture the underlying patterns in the data.
3. **Ensemble methods**: Experiment with ensemble learning techniques, such as stacking or bagging, to combine the strengths of multiple models.
4. **Alternative algorithms**: Investigate other machine learning algorithms that may be well-suited to this problem, such as Support Vector Machines (SVM) or deep learning models.

7. Bibliography

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. doi:10.1109/MCSE.2007.55

Narkhede, P.; Walambe, R.; Chandel, P.; Mandaokar, S.; Kotecha, K. MultimodalGasData: Multimodal Dataset for Gas Detection and Classification. Data 2022, 7, 112. https://doi.org/10.3390/data7080112

Narkhede P, Walambe R, Mandaokar S, Chandel P, Kotecha K, Ghinea G. Gas Detection and Identification Using Multimodal Artificial Intelligence Based Sensor Fusion. Applied System Innovation. 2021; 4(1):3. https://doi.org/10.3390/asi4010003

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.