
Real-Time Gas Leak Detection in Logistics and Manufacturing Environments Using Support Vector Machine on IoT Gas Sensor Data



UNIVERSITY OF CHITTAGONG
DEPARTMENT OF MATHEMATICS

*A Project Report Submitted in Partial Fulfillment of the Requirements for the Degree of
M.S. in Mathematics (Applied Mathematics)*

Course Code: AMAT-527

Supervisor:

Dr. Mohammad Jalal Ahammad
Professor
Department of Mathematics
University of Chittagong

Submitted by:

Md. Ashfaque Rahman
ID No: 18203096
Session: MS 2023-2024

25 October 2026

Abstract

Gas leaks in factories, warehouses, cold-chain facilities and freight depots threaten workers, property and the environment, yet most low-cost monitoring systems still rely on a fixed threshold on a single sensor. Such alarms cannot tell a dangerous release from a harmless volatile, and they generate false alarms that erode trust. This project develops a mathematically transparent, real-time gas classification system based on a Support Vector Machine (SVM) operating on readings from an array of seven low-cost metal-oxide (MQ-series) gas sensors. The public MultimodalGasData dataset (6,400 samples; four balanced classes: No Gas, Perfume, Smoke and Mixture) was split into training (70%), validation (15%) and test (15%) sets. The complete theory of the classifier is derived, from the geometric margin and the soft-margin primal problem to the Lagrangian dual, the Karush–Kuhn–Tucker conditions, the radial basis function (RBF) kernel and one-vs-one multi-class voting. A linear SVM on raw readings served as the baseline and reached 80.31% test accuracy. After z-score standardisation and a five-fold cross-validated grid search over the regularisation parameter C and the kernel width γ (optimum $C = 100$, $\gamma = 10$), the RBF-SVM achieved 98.23% test accuracy, a macro F1-score of 0.9823 and a macro ROC-AUC of 0.9986, with perfect recognition of Smoke and Mixture. Viewed as a leak detector, it reached 99.03% sensitivity and 95.83% specificity. A single prediction took on average 1.10 ms on an ordinary laptop CPU, far below the 2 s sensor sampling interval, confirming suitability for real-time use. A chronological robustness test exposed the effect of sensor drift, which is discussed as the main limitation. The study shows that a carefully tuned, interpretable kernel classifier on an inexpensive sensor array can match heavier deep-learning approaches for industrial gas-leak monitoring.

Keywords: gas leak detection; support vector machine; RBF kernel; metal-oxide gas sensors; electronic nose; real-time monitoring; Internet of Things; manufacturing and logistics safety.

Acknowledgements

I, Md. Ashfaur Rahman, express my heartfelt gratitude to the Almighty for granting me the strength, patience and perseverance to complete this research project successfully.

I would like to express my sincere appreciation to my supervisor, Dr. Mohammad Jalal Ahammad, Professor, Department of Mathematics, University of Chittagong, for the invaluable guidance, encouragement and constructive feedback provided throughout this work. The supervisor's insight shaped both the mathematical framework and the direction of the experiments.

I am grateful to all the respected teachers of the Department of Mathematics for the knowledge and academic support they have given me throughout my studies, which laid the foundation for this research.

I also thank the creators of the MultimodalGasData dataset for making their measurements openly available, and the developers of the open-source scientific Python ecosystem used in this study.

Finally, I am deeply thankful to my family, friends and classmates for their continuous encouragement, patience and moral support. Their belief in me has been a constant source of motivation.

Certification of Completion

This is to certify that Md. Ashfaur Rahman, a student of the University of Chittagong, has successfully completed the project titled “**Real-Time Gas Leak Detection in Logistics and Manufacturing Environments Using Support Vector Machine on IoT Gas Sensor Data**”. This project was carried out under the guidance of Dr. Mohammad Jalal Ahammad, Professor, Department of Mathematics, who provided valuable support and supervision throughout the entire duration of the work.

Project Details:

Project Title: Real-Time Gas Leak Detection in Logistics and Manufacturing Environments Using Support Vector Machine on IoT Gas Sensor Data

Student: Md. Ashfaur Rahman

Student ID: 18203096

Session: MS 2023-2024

Course Code: AMAT-527

Institution: University of Chittagong

Supervisor: Dr. Mohammad Jalal Ahammad

This project work has been examined and evaluated, and it is deemed satisfactory and to meet the requirements for completion. The undersigned expresses appreciation for the effort and dedication demonstrated by the student in the successful execution of the project.

Supervisor's Name: Dr. Mohammad Jalal Ahammad

Signature: _____

Date: _____

Declaration

I hereby declare that the work contained in this *M.S. project report* is my own and has not been submitted in any form for another degree or diploma at any university or institution. All information derived from other published or unpublished sources has been properly acknowledged and referenced.

I further confirm that I have adhered to all ethical principles and guidelines required for this research, that the dataset used is publicly available under an open licence, and that any assistance received during the preparation of this report has been fully disclosed.

Signature: _____

Name: Md. Ashfaqur Rahman

Date: _____

Table of Contents

Abstract.....	i
Acknowledgements	ii
Certification of Completion.....	iii
Declaration.....	iv
Table of Contents	v
List of Tables.....	viii
List of Figures.....	ix
List of Abbreviations	x
CHAPTER 1: INTRODUCTION.....	1
1.1 Background of the Study	1
1.2 Hazardous Gases in Manufacturing and Logistics	2
1.3 Causes of Gas Leakage.....	3
1.4 Gas Sensing Technologies.....	4
1.5 Current Leak Detection Approaches	5
1.6 Real-Time Monitoring and Machine Learning.....	5
1.7 Research Objectives	6
1.8 Organisation of the Report	6
CHAPTER 2: LITERATURE REVIEW.....	8
2.1 Machine Learning for Gas-Sensor Arrays.....	8
2.2 Support Vector Machines for Gas Identification.....	9
2.3 IoT and Real-Time Gas Leak Monitoring	9
2.4 Deep Learning and Studies on the MultimodalGasData Dataset	10
2.5 Gas Leak Detection in Industrial and Logistics Settings.....	11
2.6 Sensor Drift and Cross-Sensitivity	12
2.7 Research Gap.....	12
CHAPTER 3: METHODOLOGY	14

3.1 Research Framework	14
3.2 Data Collection	14
3.2.1 Sample of the Sensor Data.....	16
3.2.2 Class Distribution	16
3.3 Exploratory Data Analysis	16
3.4 Data Preprocessing	20
3.4.1 Data Splitting.....	20
3.4.2 Feature Standardisation	20
3.5 Support Vector Machine: Mathematical Formulation.....	21
3.5.1 Problem Setting and the Separating Hyperplane.....	21
3.5.2 Hard-Margin SVM	22
3.5.3 Soft-Margin SVM.....	22
3.5.4 The Lagrangian and the Dual Problem.....	23
3.5.5 Karush–Kuhn–Tucker Conditions and Support Vectors	24
3.5.6 The Kernel Trick and the RBF Kernel	25
3.5.7 The Decision Function	26
3.5.8 Multi-Class Classification: One-vs-One Voting.....	26
3.5.9 Probability Estimates (Platt Scaling).....	27
3.5.10 Solving the Dual: Sequential Minimal Optimisation	27
3.5.11 Computational Complexity and Real-Time Suitability	28
3.6 Hyper-Parameter Tuning by Grid Search and Cross-Validation.....	29
3.7 Performance Evaluation Metrics	30
3.8 Real-Time Deployment Concept	31
3.9 Python Code for Training the SVM	31
CHAPTER 4: RESULTS AND DISCUSSION	37
4.1 Computational Environment and Tools.....	37
4.2 Baseline Model: Linear SVM on Raw Readings	37

4.3 Hyper-Parameter Tuning Results	38
4.4 Performance of the Tuned RBF-SVM.....	39
4.4.1 Overall Improvement.....	39
4.4.2 Confusion Matrix.....	40
4.4.3 Precision, Recall and F1-Score.....	41
4.4.4 ROC Curves.....	41
4.4.5 Learning Curve	42
4.5 Comparison of Kernel Functions.....	42
4.6 Visualisation of the Decision Regions.....	43
4.7 Performance as a Leak Detector	44
4.8 Real-Time Performance.....	44
4.8.1 Inference Latency	44
4.8.2 Simulated Real-Time Predictions with Confidence	45
4.9 Robustness Under Temporal Drift.....	46
4.10 Comparison with Previous Studies.....	47
4.11 Discussion.....	47
CHAPTER 5: CONCLUSION	49
5.1 Summary of the Study	49
5.2 Contributions	49
5.3 Limitations.....	50
5.4 Future Work.....	50
5.5 Concluding Remarks	50
REFERENCES	52

List of Tables

Table 1: Exposure and flammability limits of common industrial gases (NIOSH, n.d.-a-f; NOAA, n.d.-a, n.d.-b; OSHA, n.d.-a, n.d.-b). C = ceiling value.	2
Table 2: MQ-series gas sensors used in this study and their sensitive gases (Narkhede et al., 2022a; MQ-2 datasheet, n.d.).	4
Table 3: Reported results of previous studies on the MultimodalGasData dataset	11
Table 4: Summary of the dataset used in this study	15
Table 5: Sample rows of the sensor data file	16
Table 6: Descriptive statistics of the sensor readings (ADC counts)	16
Table 7: Class-wise mean sensor readings (ADC counts).....	17
Table 8: Stratified split of the dataset	20
Table 9: Interpretation of the KKT conditions	24
Table 10: Hyper-parameter search space.....	29
Table 11: Computational environment	37
Table 12: Performance before and after improvement	39
Table 13: Classification report of the tuned RBF-SVM on the test data.....	41
Table 14: Comparison of SVM kernels (all with standardised inputs)	42
Table 15: Binary leak-detection performance of the tuned RBF-SVM on the test data	44
Table 16: Inference latency of the tuned RBF-SVM (Intel Core i3-6006U).....	44
Table 17: Predicted class probabilities for sample readings	45
Table 18: Confusion matrix under the chronological split (accuracy 54.37%).....	46
Table 19: Comparison with previous results on the MultimodalGasData sensor readings.	47

List of Figures

Figure 1: Taxonomy of gas leak detection methods and the position of this study (adapted from Murvay & Silea, 2012; Adegboye et al., 2019)	5
Figure 2: Research framework of the study	14
Figure 3: Sample thermal images of the four classes recorded simultaneously with the sensor readings (Narkhede et al., 2022a); only the sensor readings are used for modelling	15
Figure 4: Class distribution of the dataset	16
Figure 5: Box plots of each sensor's readings by class.....	17
Figure 6: Sensor readings in time order for each class.....	18
Figure 7: Pearson correlation matrix of the seven gas sensors.....	19
Figure 8: Projection of the standardised data onto the first two principal components	20
Figure 9: Geometry of (a) the hard-margin and (b) the soft-margin SVM. Circled points are support vectors.....	22
Figure 10: Illustration of the kernel idea: a mapping to a higher-dimensional space makes the classes linearly separable.....	25
Figure 11: (a) RBF kernel value against distance for different γ ; (b)–(d) effect of γ on the decision boundary of an RBF-SVM	26
Figure 12: One-vs-one decomposition of the four-class gas problem into six binary SVMs with majority voting	27
Figure 13: Training and prediction algorithm of the SVM classifier	28
Figure 14: Stratified five-fold cross-validation used to score each (C, γ) pair.....	29
Figure 15: Real-time deployment pipeline of the proposed SVM gas detector	31
Figure 16: Confusion matrix of the baseline linear SVM on the test data	38
Figure 17: Five-fold cross-validation accuracy of the RBF-SVM for each (C, γ) pair.....	39
Figure 18: Confusion matrix of the tuned RBF-SVM on the test data.....	40
Figure 19: One-vs-rest ROC curves of the tuned RBF-SVM on the test data.....	41
Figure 20: Learning curve of the tuned RBF-SVM.....	42
Figure 21: Decision regions of an RBF-SVM in the plane of the first two principal components.....	43
Figure 22: Distribution of single-reading prediction times	45

List of Abbreviations

ADC	Analog-to-Digital Converter
AUC	Area Under the (ROC) Curve
CNN	Convolutional Neural Network
CV	Cross-Validation
EDA	Exploratory Data Analysis
FN / FP	False Negative / False Positive
FPR / TPR	False Positive Rate / True Positive Rate
IDLH	Immediately Dangerous to Life or Health
IoT	Internet of Things
KKT	Karush–Kuhn–Tucker (conditions)
LEL / UEL	Lower / Upper Explosive Limit
LIBSVM	Library for Support Vector Machines
LPG	Liquefied Petroleum Gas
LSTM	Long Short-Term Memory (network)
MOS / MOX	Metal-Oxide Semiconductor
NDIR	Non-Dispersive Infrared
OvO / OvR	One-vs-One / One-vs-Rest
OSHA	Occupational Safety and Health Administration
NIOSH	National Institute for Occupational Safety and Health
PCA	Principal Component Analysis
PEL / REL	Permissible / Recommended Exposure Limit
ppm	parts per million
QP	Quadratic Programming
RBF	Radial Basis Function
ROC	Receiver Operating Characteristic
SMO	Sequential Minimal Optimization
SV	Support Vector
SVM	Support Vector Machine
TinyML	Tiny Machine Learning
TN / TP	True Negative / True Positive
VOC	Volatile Organic Compound

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

Modern manufacturing and logistics depend heavily on gases. Liquefied petroleum gas (LPG) and natural gas fuel boilers, furnaces and forklifts; anhydrous ammonia is the standard refrigerant of industrial cold stores and food-processing plants; hydrogen, oxygen and other industrial gases are produced, cylinder-filled and shipped through warehouses and container depots; and countless volatile organic compounds (VOCs) are released by solvents, paints and chemical stock. Every link of this chain, from storage tanks and pipework to cylinders in transit, is a potential source of leakage. Because most of these gases are colourless and many are odourless, human senses cannot be relied on to detect them (Narkhede et al., 2021).

The human and economic cost of hazardous exposure at work is very large. The International Labour Organization estimates about 2.78 million work-related deaths and 374 million non-fatal work-related injuries and illnesses every year, with lost working days costing almost 4% of global gross domestic product (International Labour Organization [ILO], 2019). More than one billion workers are exposed to hazardous substances, including vapours and fumes, every year (ILO, 2021). An analysis of chemical-process accidents found that piping caused about 25% of equipment-related accidents and reactors and storage tanks about 14% each, with technical design and human–technical interface failures dominating the contributing causes (Kidam & Hurme, 2013).

History offers sobering examples. In December 1984 more than 40 tonnes of methyl isocyanate escaped from a pesticide plant in Bhopal, India, killing at least 3,800 people within hours and exposing more than 500,000 (Broughton, 2005). In May 2020 about 3 tonnes of styrene vapour leaked from the LG Polymers plant in Visakhapatnam, killing 12 people; the official probe found that alarms and sirens at 36 locations were not functioning (Down To Earth, 2020). Cold-chain logistics has its own record: an ammonia leak at a seafood cold store in Shanghai killed at least 15 workers in 2013 (Al Jazeera, 2013), and a hydraulic shock at Millard Refrigerated Services in Alabama released more than 32,000 lb of anhydrous ammonia in 2010, sending 32 people to hospital (U.S. Chemical Safety and Hazard Investigation Board [CSB], 2015).

Bangladesh, and the Chattogram region in particular, has suffered repeatedly. The fire and explosions at the BM Container Depot in Sitakunda on 4 June 2022, traced to hydrogen-peroxide containers, killed 51 people including firefighters and injured more than 200 (United News of Bangladesh [UNB], 2022). On 4 March 2023 an explosion at an oxygen plant in Sitakunda killed 7 and injured 32 (The Business Standard, 2023). Three days later an explosion at Siddique Bazar in Dhaka, which investigators linked to gas accumulating from a leaking line in an unapproved basement, killed 26 people (Dhaka Tribune, 2025). In September 2020 gas from a leaking underground pipeline ignited inside a mosque in Narayanganj and killed 34 worshippers (The Daily Star, 2020), while an ammonia tank at the DAP fertiliser factory in Anwara, Chattogram, exploded in 2016 and about 250 people fell ill (The Daily Star, 2016). Gas-cylinder fires are also frequent: the Fire Service recorded 953 such incidents in 2018 alone (Dhaka Tribune, 2021).

These incidents share a pattern: the leak began long before the disaster, and either no detection system existed or the existing one failed to give a timely, trustworthy warning. Reliable, fast and affordable gas-leak detection is therefore a critical safety requirement for manufacturing and logistics facilities, and it is the motivation for this project.

1.2 Hazardous Gases in Manufacturing and Logistics

The gases encountered in industrial and logistics settings differ in toxicity and flammability. Table 1 summarises occupational exposure limits and flammability ranges for the most common ones. The lower explosive limit (LEL) is the minimum concentration in air that can ignite; occupational limits (OSHA PEL, NIOSH REL) are the concentrations to which a worker may be exposed over a shift; and the IDLH value is the concentration that is immediately dangerous to life or health.

Table 1: Exposure and flammability limits of common industrial gases (NIOSH, n.d.-a-f; NOAA, n.d.-a, n.d.-b; OSHA, n.d.-a, n.d.-b). C = ceiling value.

Gas	Typical source	OSHA PEL	NIOSH REL	IDLH	LEL–UEL (% vol)
LPG (propane/butane)	fuel, forklifts, cylinders	1,000 ppm	1,000 ppm	2,000 ppm	≈2.1–9.5
Methane	natural gas lines	—	—	asphyxiant	5–15
Carbon monoxide	combustion, furnaces	50 ppm	35 ppm	1,200 ppm	12.5–74
Hydrogen sulfide	waste, chemical process	20 ppm (C)	10 ppm (C)	100 ppm	4.0–44.0
Ammonia	cold-chain refrigeration	50 ppm	25 ppm	300 ppm	15–28
Hydrogen	industrial gas, batteries	—	—	asphyxiant	4–75
Benzene (VOC)	solvents, fuels	1 ppm	0.1 ppm	500 ppm	1.2–7.8

Two features of the table matter for detection design. First, several gases become toxic at a few tens of ppm, far below their flammable range, so an early warning must react to low concentrations. Second, many harmless substances (cleaning agents, perfumes, solvents) also contain VOCs that excite the same low-cost sensors; a useful detector must therefore *discriminate* between gases, not merely sense that *something* is present.

1.3 Causes of Gas Leakage

Gas leaks in industrial and logistics environments arise from technical, operational and human causes. The main ones reported in the literature and incident investigations are:

- **Equipment and design failure:** failure of piping, valves, seals, flanges, reactors and storage tanks, which together account for a large share of chemical-process accidents (Kidam & Hurme, 2013).
- **Corrosion and ageing:** internal and external corrosion and material fatigue weaken pipes and cylinders over time (Adegboye et al., 2019; Murvay & Silea, 2012).
- **Storage and temperature-control failure:** loss of temperature control can raise pressure until a valve fails, as at LG Polymers (Down To Earth, 2020), or produce hydraulic shock in refrigeration lines (CSB, 2015).
- **Hazardous-goods handling in logistics:** uncertified containers and prolonged storage of dangerous goods at depots (UNB, 2022), and expired or poorly maintained LPG cylinders, valves and hoses (Dhaka Tribune, 2021).
- **Distribution-network defects and illegal connections:** leaking underground lines and improperly joined commercial and residential supplies (The Daily Star, 2020; Dhaka Tribune, 2025).
- **Human error and non-functional safety systems:** unskilled operation (The Business Standard, 2023) and disabled or unmaintained alarms (Down To Earth, 2020).

Because many of these causes develop gradually and invisibly, continuous automated monitoring is the most effective safeguard.

1.4 Gas Sensing Technologies

Gas sensors convert the presence of a gas into an electrical signal. Following Liu et al. (2012) and Feng et al. (2019), the main families are:

- **Metal-oxide semiconductor (MOS) sensors.** A heated layer of a metal oxide such as SnO_2 adsorbs atmospheric oxygen, which traps conduction electrons and raises the resistance. Reducing gases react with the adsorbed oxygen, release the electrons and lower the resistance, so the conductance rises with gas concentration (Wang et al., 2010; Dey, 2018). MOS sensors are small, cheap, fast and long-lived, but they are poorly selective and drift with temperature, humidity and age.
- **Electrochemical sensors,** which generate a current proportional to the concentration of toxic gases such as CO , H_2S and NH_3 (Stetter & Li, 2008).
- **Catalytic (pellistor) sensors,** which measure the heat of combustion of flammable gases on a catalyst bead (Feng et al., 2019).
- **Optical and non-dispersive infrared (NDIR) sensors,** which measure the absorption of infrared light at gas-specific wavelengths (Hodgkinson & Tatam, 2013).

The popular low-cost MQ-series modules are MOS sensors with an integrated heater and an analogue output voltage that rises with gas concentration. Each is designed for a target gas but responds to several others. Table 2 lists the seven MQ sensors used in this study.

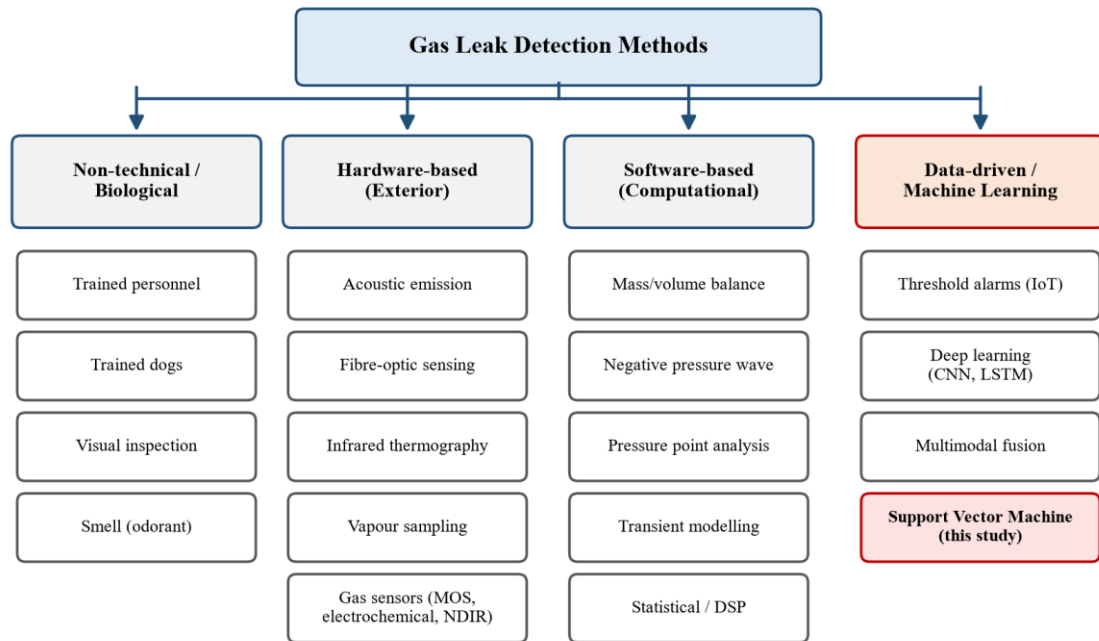
Table 2: MQ-series gas sensors used in this study and their sensitive gases (Narkhede et al., 2022a; MQ-2 datasheet, n.d.).

Sensor	Principal target gases	Type
MQ2	LPG, butane, methane, smoke	MOS (SnO_2)
MQ3	alcohol, ethanol, smoke	MOS
MQ5	LPG, natural gas	MOS
MQ6	LPG, butane	MOS
MQ7	carbon monoxide	MOS
MQ8	hydrogen	MOS
MQ135	air quality (NH_3 , benzene, smoke)	MOS

Because each MOS sensor responds to several gases (cross-sensitivity), a single sensor cannot identify a gas. An *array* of partially selective sensors, however, produces a distinctive response pattern for each gas, an idea known as the electronic nose (e-nose). Recognising these patterns is a classification problem, which is where machine learning enters (Ye et al., 2021).

1.5 Current Leak Detection Approaches

Leak detection methods are commonly grouped into non-technical (biological and visual), hardware-based and software-based approaches (Murvay & Silea, 2012; Adegboye et al., 2019). In the last decade a fourth, data-driven group has emerged, in which machine-learning models interpret sensor data. Figure 1 summarises this taxonomy and locates the present study.



Adapted from Murvay & Silea (2012) and Adegboye et al. (2019), with a data-driven branch added.

Figure 1: Taxonomy of gas leak detection methods and the position of this study (adapted from Murvay & Silea, 2012; Adegboye et al., 2019)

Most installed industrial gas detectors, and almost all low-cost Internet of Things (IoT) prototypes, raise an alarm when the reading of one sensor exceeds a fixed threshold (Praveen Sharma et al., 2023; Nipo et al., 2025). Threshold alarms are simple but have well-known weaknesses: they cannot tell which gas is present, they are triggered by harmless VOCs, and their behaviour depends on drifting sensor baselines, which leads to false alarms that operators learn to ignore (Rahman et al., 2017). Pattern-recognition approaches that use the whole sensor array address these weaknesses.

1.6 Real-Time Monitoring and Machine Learning

A real-time system is one whose correctness depends not only on the logical result of a computation but also on the time at which the result is produced (Shin & Ramanathan, 1994). For a gas alarm the deadline is set by physics: in a confined space a toxic gas can

reach its IDLH concentration within minutes, so each new reading must be classified well before the next one arrives. Industry 4.0 connects sensors, controllers and cloud services into cyber-physical production systems (Lasi et al., 2014), which makes continuous, data-driven monitoring practical. The classifier at the heart of such a system must therefore be accurate, fast, light enough for embedded hardware and, for safety certification, understandable.

Deep neural networks have achieved high accuracy on gas-sensor data, but they need large training sets and considerable computation, and they behave as black boxes. The Support Vector Machine (SVM) (Cortes & Vapnik, 1995) offers an attractive alternative: it rests on a rigorous convex-optimisation theory, works well with small and medium-sized data sets, and its prediction cost depends only on a small set of *support vectors*. This project develops and analyses such a classifier for gas-leak detection.

1.7 Research Objectives

The primary objectives of this research are:

1. To formulate the gas identification problem mathematically and to derive the Support Vector Machine classifier, including the soft-margin primal problem, its Lagrangian dual, the Karush–Kuhn–Tucker conditions and the kernel extension.
2. To develop an SVM-based multi-class gas detection model using the readings of a seven-sensor MQ array, starting from a linear baseline and improving it through feature standardisation, the RBF kernel and cross-validated hyper-parameter tuning.
3. To evaluate the model with accuracy, precision, recall, F1-score, confusion matrices and ROC analysis, and to interpret it as a leak / no-leak detector.
4. To assess the real-time suitability of the model by measuring per-reading inference latency, and to examine its robustness under temporal sensor drift.

1.8 Organisation of the Report

The remainder of this report is organised as follows. Chapter 2 reviews the literature on machine learning for gas sensing, SVM-based gas identification, IoT leak monitoring and studies on the dataset used here, and identifies the research gap. Chapter 3 describes the data and develops the mathematical methodology, from preprocessing and the SVM derivation to tuning and evaluation metrics, followed by the implementation. Chapter 4

presents and discusses the results, and Chapter 5 concludes the report with limitations and directions for future work.

CHAPTER 2: LITERATURE REVIEW

Research on automated gas detection spans sensor physics, signal processing, machine learning and embedded systems. This chapter reviews the work most relevant to this project in six themes: machine learning for gas-sensor arrays, Support Vector Machines for gas identification, IoT and real-time leak monitoring, deep-learning and multimodal studies on the dataset used here, industrial and logistics applications, and sensor drift. It closes by identifying the research gap that this project addresses.

2.1 Machine Learning for Gas-Sensor Arrays

The use of pattern recognition to interpret arrays of partially selective sensors is well established. Feng et al. (2019) reviewed smart gas sensing and described the standard pipeline of preprocessing, feature extraction, classification (PCA, LDA, SVM, ANN) and networked alarm; however, being a review, it reported no comparative benchmark and predates recent multimodal work. Karakaya et al. (2020) surveyed electronic-nose hardware and algorithms, grouping classifiers into statistical, kernel and neural families, but treated gas-safety applications only briefly and did not consider real-time latency. Ye et al. (2021) argued that feature extraction, modelling and drift compensation jointly determine the accuracy and long-term stability of an e-nose; however, their review said little about the mathematical derivation of classifiers or their embedded inference cost. Chowdhury and Oehlschlaeger (2025) recently surveyed artificial intelligence in gas sensing for industrial safety, environmental monitoring and medicine, concentrating mainly on deep learning and giving little attention to closed-form, mathematically transparent classifiers.

Several experimental studies have used low-cost sensor arrays similar to the one in this project. Viciano-Tudela et al. (2023) built an array of seven MQ-family sensors to classify essential oils and compared discriminant analysis, SVM, k-nearest neighbours (kNN), a neural network and naive Bayes; kNN reached 100% accuracy, although on static laboratory headspace samples rather than dynamic leaks. Zeng et al. (2024) combined principal component analysis (PCA), synthetic oversampling (SMOTE) and SVM or kNN to classify ethanol, acetone and air, reaching 91.7% accuracy and showing that PCA raised the AUC of the SVM by 25.2%; however, the study used only three classes from a single custom sensor.

2.2 Support Vector Machines for Gas Identification

Since its introduction by Cortes and Vapnik (1995), the SVM has been a standard tool for e-nose classification. Song et al. (2011) designed a wireless e-nose in which a Fe_2O_3 sensor array and a digital signal processor ran a least-squares support vector regression that identified CH_4 , H_2 and their mixture and estimated concentration in real time; it was more accurate than artificial neural networks, but covered only two gases and reported no latency. Laref et al. (2019) studied how C , ϵ and the RBF width affect SVM performance on chemical sensor arrays and proposed generalised pattern search as a faster alternative to exhaustive grid search; their work confirms that cross-validated tuning of these hyper-parameters is decisive, but it addressed regression rather than multi-class classification.

Boubin and Shrestha (2019) trained an SVM on a PC and ported it to an 8-bit ATmega microcontroller to classify breath-VOC sensor readings, achieving 97.1% accuracy and demonstrating that a trained SVM is light enough for low-end hardware; however, the task was binary and inference time was not reported. Bakar et al. (2020) used PCA and SVM in an e-nose for pre-entry testing of industrial confined spaces and obtained 95.83% test accuracy on diesel, gasoline, petrol and thinner vapours, although the kernel and tuning procedure were not specified. Huang et al. (2021) built a portable ten-sensor MOX e-nose read by an STM32 microcontroller and classified wine and oil samples with an SVM, achieving sensitivity and specificity of about 96–99%, but classification ran offline. Zhang and Han (2022) tuned an SVM with the sparrow search algorithm to identify mixed gases and predict their concentrations, outperforming random forest and neural-network baselines; the metaheuristic search, however, adds stochastic cost and the paper provided no runtime measurement.

2.3 IoT and Real-Time Gas Leak Monitoring

Low-cost IoT detectors are now common. Praveen Sharma et al. (2023) combined an Arduino, a Wi-Fi module, an MQ-2 sensor and GPS to send cloud alerts when a leak was detected, but the system used a single-sensor threshold and reported no accuracy or false-alarm rate. In Bangladesh, Nipo et al. (2025) developed an ESP32-based smart warehouse with MQ-135, MQ-2 and temperature–humidity sensors that triggered ventilation, fire suppression and emergency exits within three seconds; however, detection was again threshold-based, so a harmless VOC and a dangerous leak could raise the same alarm. A systematic review of IoT ammonia monitoring by da Silva et al. (2025) found that most

systems use MQ sensors with ESP32 or Raspberry Pi controllers and fixed thresholds, and identified integration with artificial intelligence as an open problem.

Tiny machine learning (TinyML) moves inference onto the device itself. Abadade et al. (2023) surveyed TinyML and argued that on-device inference reduces latency, energy use and cloud dependence, though the survey focused almost entirely on neural networks. Tsoukas et al. (2023) built a standalone TinyML leak detector that reached F1-scores of only 0.77 for smoke and 0.70 for ammonia. El Barkani et al. (2024) deployed MobileNetV1 and EfficientNet-B0 networks on an Arduino Nano 33 BLE Sense using the thermal images of the MultimodalGasData set, reaching 91.73% accuracy with an inference time of 1,414 ms per image; however, they ignored the seven gas-sensor channels, and more than one second per prediction is slow for a safety alarm.

2.4 Deep Learning and Studies on the MultimodalGasData Dataset

Deep networks have been applied to generic e-nose data with considerable success. Peng et al. (2018) designed a 38-layer convolutional network, GasNet, that reached about 95.2% accuracy, compared with 79.9% for an SVM, although the SVM baseline appears to have been only lightly tuned. Pan et al. (2019) combined convolutional and recurrent layers to classify gases from the early transient response, reaching 98.28% after 4 s; Zhao et al. (2019) used a multi-label one-dimensional CNN to identify gas mixtures with 96.30% accuracy; and Zhou and Liu (2021) reached up to 96% with a convolutional LSTM. These models, however, require dense high-rate sampling or large training sets, and none reported embedded inference cost.

The public MultimodalGasData dataset used in this project was introduced by Narkhede et al. (2021, 2022a). In the original study, an LSTM network trained on the seven MQ sensor readings achieved 82% test accuracy, a CNN on thermal images 93%, and early or late fusion of both about 96%; the authors attributed the weak sensor-only result to the unreliability of individual low-cost sensors, but did not test well-tuned classical classifiers. Attallah (2023) converted the sensor vectors into heat-map images, extracted features with three pretrained CNNs and classified them with a Bi-LSTM, obtaining 93.27% from the sensors alone and 99.25% with multitask fusion; the pipeline is, however, computationally heavy for a seven-value input. Sharma et al. (2024) reached 93.39% on the sensor data with a BiLSTM-dense network and 99.7% with multimodal federated learning. Jadhav et al. (2025) super-resolved the thermal images with a generative network and fused them

with the sensor data in an EfficientNet-based model, achieving 98.55%, but gave no sensor-only baseline.

Classical machine learning has also been tried on this dataset. Arya et al. (2024) concatenated normalised sensor readings with VGG16 image features and reported 96.41% for random forest, 94.53% for kNN and 94.22% for SVM, all on the fused feature space. Lalithadevi and Krishnaveni (2025) used only the seven sensor channels and obtained 98.67% with a tuned, explainable random forest; notably, the SVM in their comparison reached only **88.36%**, with no kernel, scaling or hyper-parameter search reported. Table 3 summarises the reported results.

Table 3: Reported results of previous studies on the MultimodalGasData dataset

Study	Input data	Model	Accuracy
Narkhede et al. (2021)	Sensors only	LSTM	82%
Narkhede et al. (2021)	Sensors + thermal	Early fusion CNN + LSTM	≈96%
Attallah (2023)	Sensors only (as heat maps)	CNN features + Bi-LSTM	93.27%
Attallah (2023)	Sensors + thermal	Multitask fusion	99.25%
Sharma et al. (2024)	Sensors only	BiLSTM-Dense	93.39%
Arya et al. (2024)	Sensors + VGG16 features	SVM / RF	94.22% / 96.41%
El Barkani et al. (2024)	Thermal only (MCU)	EfficientNet-B0	91.73% (1,414 ms)
Jadhav et al. (2025)	Sensors + thermal	EfficientNetB2 late fusion	98.55%
Lalithadevi & Krishnaveni (2025)	Sensors only	SVM (untuned)	88.36%
Lalithadevi & Krishnaveni (2025)	Sensors only	Tuned random forest	98.67%

2.5 Gas Leak Detection in Industrial and Logistics Settings

Machine learning is increasingly applied to industrial leak data. Aljameel et al. (2022) compared random forest, SVM, kNN, gradient boosting and decision trees for anomaly detection in oil and gas pipelines and found that the SVM gave the best accuracy (97.4%), supporting its suitability for industrial leak data; the task, however, was binary and used process measurements rather than gas-sensor arrays. Argirusis et al. (2024) used networked IoT sensors and supervised learning to discriminate refrigerant gases relevant to cold-storage and HVAC systems, but considered pure gases only. Krayden et al. (2025) developed a TinyML temporal network that compensates drift in real time for ethylene monitoring in food-transport containers and cold rooms, keeping the error below 1 ppm, though it does not classify gases. Xi et al. (2026) analysed 56 verified leak events from hazardous-chemical enterprises and identified four leak patterns by clustering, and Hwang

et al. (2026) proposed a class-imbalance-aware thermal-image detector for manufacturing; both depend on either unsupervised analysis or costly thermal cameras.

2.6 Sensor Drift and Cross-Sensitivity

MOS sensors drift over time because of ageing, poisoning and changes in temperature and humidity. Vergara et al. (2012) recorded 13,910 measurements from 16 MOS sensors over three years and showed that a classifier trained once degrades steadily; they proposed an ensemble of SVMs trained on different time batches, which requires labelled data from every period. Ziyatdinov et al. (2010) removed a common drift direction with common principal component analysis, assuming linear drift shared by all classes. Liu and Tang (2013) weighted SVM ensembles dynamically, Zhang and Zhang (2015) used domain-adaptation extreme learning machines, Jiang et al. (2021) aligned source and target distributions, and Dong et al. (2021) developed online SVM updating with memory and forgetting that kept update time below 300 ms. Even dedicated drift methods, however, achieve below 90% on the long-term drift benchmark (Rehman et al., 2021), and Huerta et al. (2016) showed that humidity and temperature must be modelled explicitly for continuous monitoring. Drift therefore remains a fundamental limitation of low-cost sensor arrays and is examined experimentally in Chapter 4.

2.7 Research Gap

The literature reveals three strands that have not been brought together. First, low-cost IoT leak detectors are cheap and responsive but almost always rely on single-sensor thresholds, so they cannot tell which gas is present and are prone to false alarms (Praveen Sharma et al., 2023; Nipo et al., 2025; da Silva et al., 2025). Second, deep and multimodal models reach 93–99% accuracy on the MultimodalGasData benchmark (Narkhede et al., 2021; Attallah, 2023; Sharma et al., 2024; Jadhav et al., 2025), but only by adding thermal cameras or heavy networks, and their real-time cost is rarely measured; the only reported on-device latency on this dataset is about 1.4 s (El Barkani et al., 2024). Third, the sensor array on its own has been inconsistently benchmarked: sensor-only results range from 82% to 98.67%, and the only sensor-only SVM reported scored 88.36% with no description of kernel, scaling or tuning (Lalithadevi & Krishnaveni, 2025).

Although SVMs are well established for e-nose discrimination, embedded deployment and industrial leak detection, no study has provided a complete mathematical treatment of a multi-class support vector classifier for low-cost MQ arrays (covering the primal and dual

problems, the KKT conditions, the kernel, the multi-class decomposition and principled hyper-parameter selection) together with measured per-reading latency and an honest assessment of temporal robustness. This project fills that gap by developing, deriving and evaluating a lightweight, interpretable RBF-SVM for four-class gas discrimination on the seven MQ channels of MultimodalGasData, positioned as a practical alternative to threshold alarms and heavy multimodal networks in manufacturing and logistics environments.

CHAPTER 3: METHODOLOGY

This chapter describes the data and the complete methodology. It begins with the overall research framework, the dataset and its exploratory analysis, and the preprocessing steps. It then develops the mathematical theory of the Support Vector Machine in detail, explains hyper-parameter selection and the evaluation metrics, and ends with the real-time deployment concept and the implementation.

3.1 Research Framework

The study follows the workflow in Figure 2. Raw readings of the seven gas sensors are first explored statistically and visually, then split into training, validation and test sets and standardised. A linear SVM on raw readings provides a baseline. The model is then improved with the RBF kernel, whose hyper-parameters are selected by five-fold cross-validation on the training set. The final model is evaluated once on the untouched test set, its real-time latency is measured, and its robustness to temporal drift is examined.

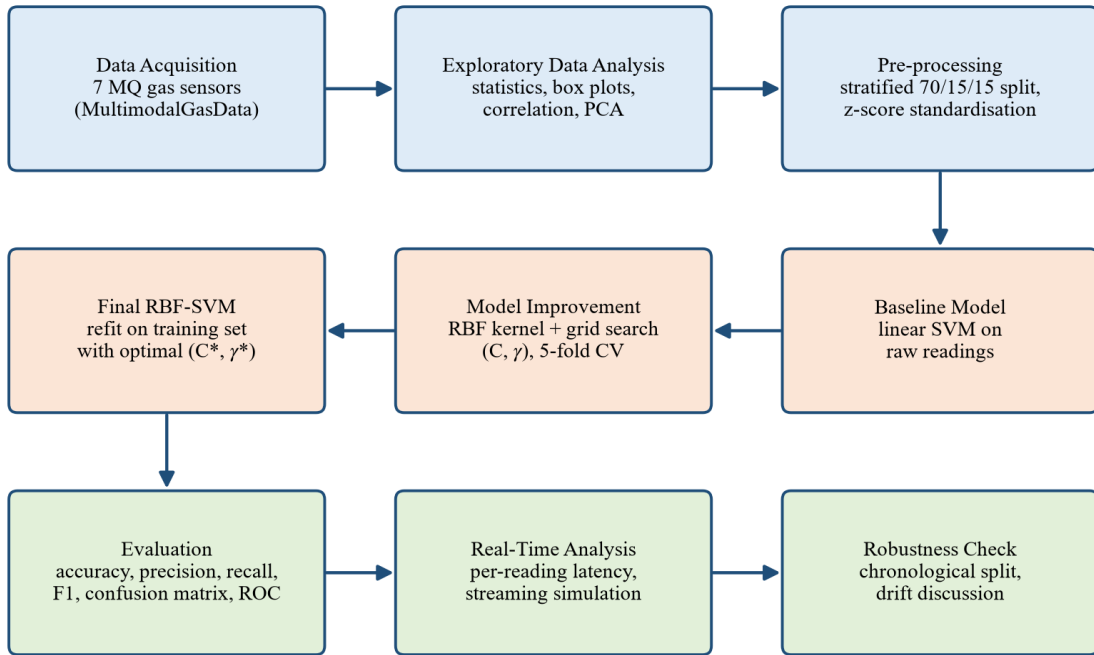


Figure 2: Research framework of the study

3.2 Data Collection

This study uses the public **MultimodalGasData** dataset (Narkhede et al., 2022a), archived on Mendeley Data (Narkhede et al., 2022b; <https://doi.org/10.17632/zkwgkjkjn9.2>) under the Creative Commons Attribution 4.0 licence. It was recorded at the Symbiosis Institute

of Technology, Pune, India, with seven MQ-series metal-oxide gas sensors (MQ2, MQ3, MQ5, MQ6, MQ7, MQ8 and MQ135) mounted close together, and a Seek Compact thermal camera recording simultaneously. The analogue sensor outputs were digitised by a 10-bit analogue-to-digital converter (ADC), so every reading is an integer between 0 and 1023 that increases with gas concentration (Narkhede et al., 2021, 2022a).

Two gas sources were used: a deodorant containing about 95% alcohol (the *Perfume* class) and incense sticks (the *Smoke* class, mainly carbon monoxide, carbon dioxide and other combustion products). Releasing both together produced the *Mixture* class, and readings in a neutral environment formed the *NoGas* class. Data were logged every 2 s while the gas was released at controlled intervals (every 15 s, 30 s and then 45 s in successive 30-minute periods). Only the gas-sensor measurements are used in this project, because the aim is a low-cost detector that does not need a thermal camera. Table 4 summarises the dataset and Figure 3 shows one thermal image per class for illustration.

Table 4: Summary of the dataset used in this study

Property	Value
Source	MultimodalGasData (Narkhede et al., 2022a), Mendeley Data, CC BY 4.0
File used	Gas_Sensors_Measurements.csv (330 KB)
Number of samples	6,400
Features	7 sensor readings: MQ2, MQ3, MQ5, MQ6, MQ7, MQ8, MQ135 (10-bit ADC counts)
Classes	NoGas, Perfume, Smoke, Mixture (1,600 samples each)
Sampling interval	2 seconds
Missing values	None

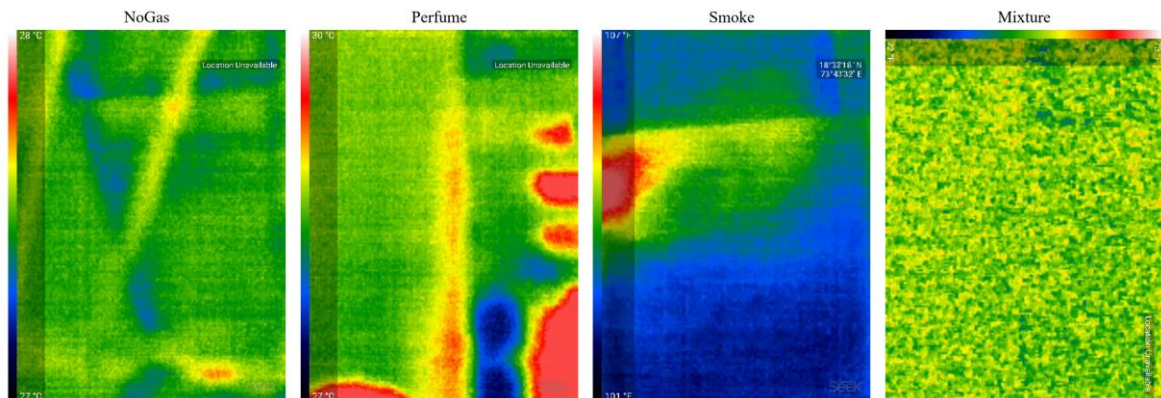


Figure 3: Sample thermal images of the four classes recorded simultaneously with the sensor readings (Narkhede et al., 2022a); only the sensor readings are used for modelling

3.2.1 Sample of the Sensor Data

The data are stored in a comma-separated values (CSV) file with ten columns: a serial number, the seven sensor readings, the class label (Gas) and the name of the corresponding thermal image. Table 5 shows two rows from each class.

Table 5: Sample rows of the sensor data file

Serial No.	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135	Gas	Image name
0	555	515	377	338	666	451	416	NoGas	0_NoGas
1	555	516	377	339	666	451	416	NoGas	1_NoGas
0	592	517	370	341	632	419	389	Perfume	0_Perfume
1	801	516	450	415	660	507	428	Perfume	1_Perfume
0	518	341	365	400	572	572	332	Smoke	0_Smoke
1	519	341	365	401	571	598	342	Smoke	1_Smoke
0	506	392	344	311	395	222	302	Mixture	0_Mixture
1	564	392	346	311	402	222	314	Mixture	1_Mixture

3.2.2 Class Distribution

Each class contains exactly 1,600 samples (Figure 4). The dataset is therefore perfectly balanced, which means that accuracy is a meaningful summary measure and no resampling is required. This is an important improvement over datasets in which one class dominates.

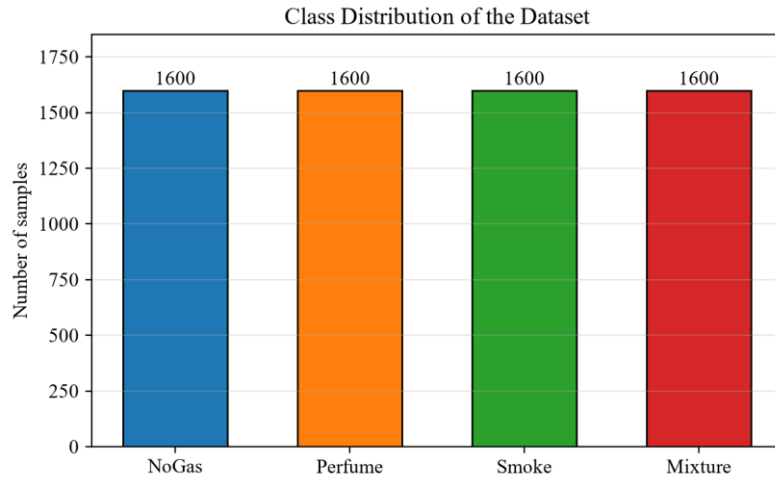


Figure 4: Class distribution of the dataset

3.3 Exploratory Data Analysis

Before modelling, the statistical properties of the sensor readings were examined. Table 6 gives the descriptive statistics of the seven sensors over all 6,400 samples.

Table 6: Descriptive statistics of the sensor readings (ADC counts)

Statistic	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135
Mean	677.59	462.02	404.58	399.76	565.95	542.47	416.73

Statistic	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135
Std. deviation	92.91	70.28	55.67	45.09	83.13	151.02	76.68
Minimum	502	337	291	311	361	220	275
1st quartile	591	405	366	366	524	447	354
Median	701	486	400	393	576	576	437
3rd quartile	756	529	443	426	629	642	473
Maximum	824	543	596	524	796	794	589

The sensors have quite different scales: the standard deviation of MQ8 (151.0) is more than three times that of MQ6 (45.1). Because distance-based methods such as the RBF-SVM are sensitive to scale, this motivates the standardisation described in Section 3.4. Table 7 shows the mean reading of each sensor within each class.

Table 7: Class-wise mean sensor readings (ADC counts)

Class	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135
NoGas	748.4	528.9	430.7	425.4	606.3	637.1	474.5
Perfume	745.3	527.7	439.9	429.0	617.2	633.8	480.9
Smoke	623.2	372.4	340.2	368.2	580.2	583.7	308.1
Mixture	593.4	419.1	407.5	376.5	460.1	315.2	403.4

Several patterns are visible. Smoke lowers the readings of MQ3, MQ5 and MQ135 markedly, while the Mixture class shows the lowest MQ7 and MQ8 values. In contrast, the NoGas and Perfume classes have almost identical mean readings on every sensor, which suggests that these two classes will be the hardest to separate. Figure 5 confirms this with box plots: the Smoke and Mixture distributions are clearly shifted on several sensors, whereas the NoGas and Perfume boxes overlap almost completely.

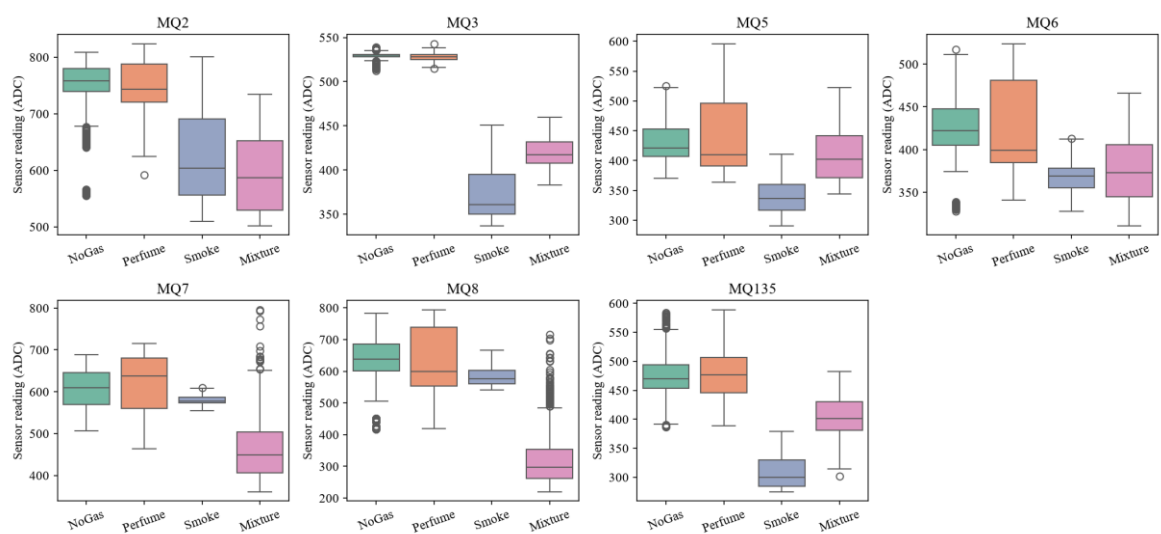


Figure 5: Box plots of each sensor's readings by class

Figure 6 plots the readings of all sensors in time order for each class. The periodic rises and slow decays reflect the repeated release and dispersion of gas. The responses are therefore not independent snapshots but a time series, with neighbouring readings highly similar; this property is important when interpreting the results and is examined in the robustness analysis of Chapter 4.

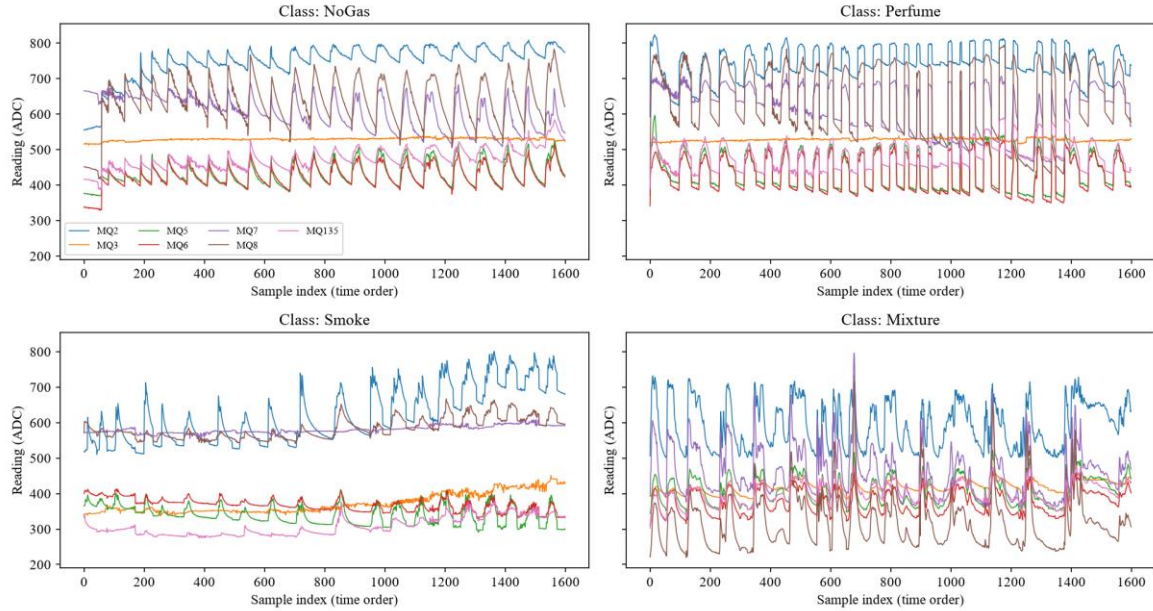


Figure 6: Sensor readings in time order for each class

The Pearson correlation matrix (Figure 7) shows strong positive correlation between several sensors, for example MQ5 and MQ6 ($r = 0.91$) and MQ5 and MQ135 ($r = 0.85$), because metal-oxide sensors respond to overlapping families of gases. The sensors nevertheless carry partly complementary information, which is what allows an array to recognise patterns that no single sensor can.

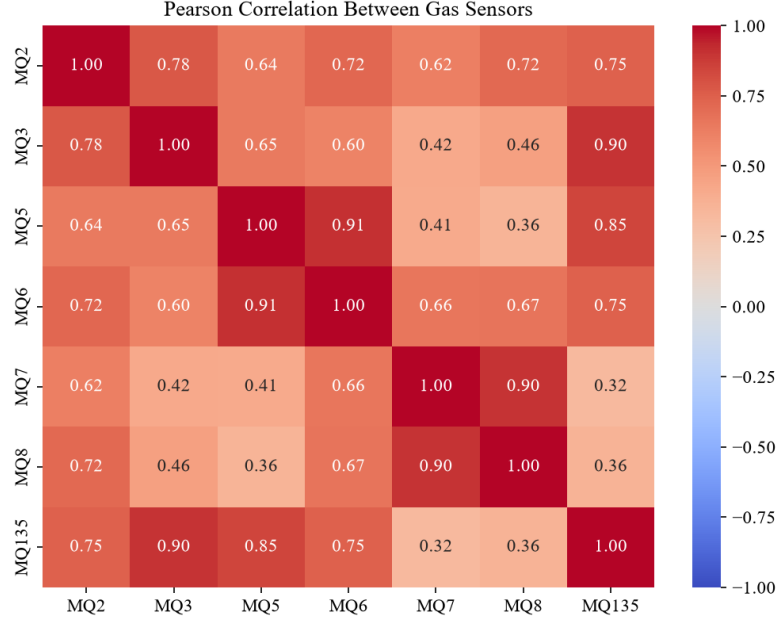


Figure 7: Pearson correlation matrix of the seven gas sensors

To visualise the seven-dimensional data, principal component analysis (PCA) was applied to the standardised training data (Jolliffe & Cadima, 2016). PCA finds orthogonal directions of maximum variance as the eigenvectors of the sample covariance matrix $\mathbf{S}$:

$$\mathbf{S}\mathbf{v}_k = \lambda_k \mathbf{v}_k, \quad \text{explained variance ratio}_k = \frac{\lambda_k}{\sum_{j=1}^7 \lambda_j} \quad (3.1)$$

The first two principal components explain 70.2% and 17.6% of the variance (87.8% together). In the projection of Figure 8, Smoke forms a compact, isolated cluster and Mixture is largely separate, whereas NoGas and Perfume overlap heavily. A straight line cannot separate these overlapping classes, which anticipates the need for a non-linear kernel. PCA is used here for visualisation only; the classifier uses all seven standardised features.

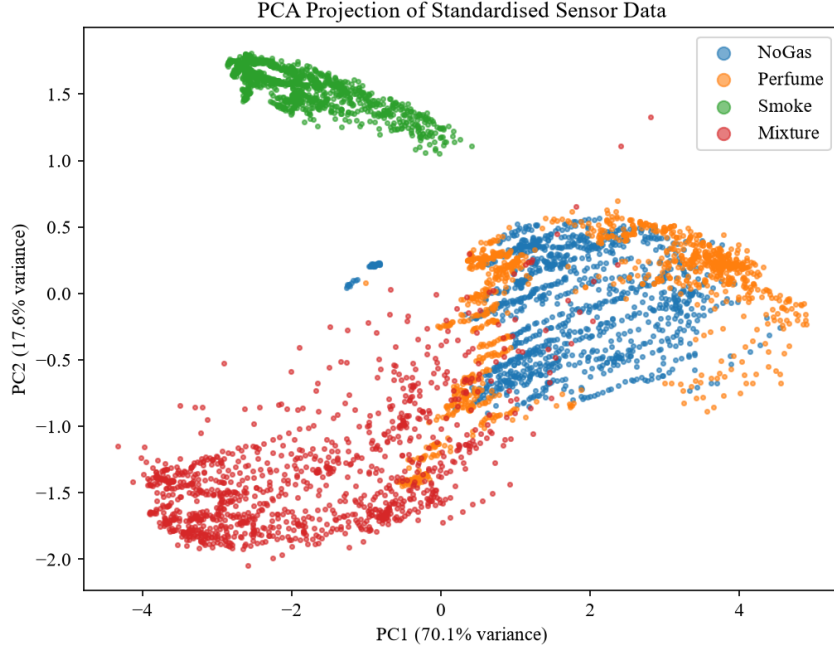


Figure 8: Projection of the standardised data onto the first two principal components

3.4 Data Preprocessing

3.4.1 Data Splitting

The 6,400 samples were divided by stratified random sampling into a training set (70%), a validation set (15%) and a test set (15%), preserving the 1:1:1:1 class ratio in each part (Table 8). The training set is used to fit the model and, through cross-validation, to choose its hyper-parameters; the validation set monitors generalisation; and the test set is used only once, for the final evaluation. A fixed random seed (42) makes the split reproducible. The 228 rows whose readings repeat an earlier reading were kept, because they are genuine consecutive measurements of a slowly changing signal.

Table 8: Stratified split of the dataset

Subset	Samples	Share	Per class	Purpose
Training	4,480	70%	1,120	model fitting and 5-fold CV tuning
Validation	960	15%	240	monitoring generalisation
Test	960	15%	240	final, one-time evaluation
Total	6,400	100%	1,600	

3.4.2 Feature Standardisation

Let $\mathbf{x}_i = (x_{i1}, \dots, x_{id})^\top \in \mathbb{R}^d$ denote the vector of $d = 7$ sensor readings of sample i , $i = 1, \dots, n$. Each feature j is transformed to zero mean and unit variance (z-score

standardisation) using the mean and standard deviation computed from the **training set only**:

$$\mu_j = \frac{1}{n} \sum_{i=1}^n x_{ij}, \quad \sigma_j = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_{ij} - \mu_j)^2} \quad (3.2)$$

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}, \quad j = 1, \dots, d \quad (3.3)$$

The same μ_j and σ_j are applied to validation, test and future real-time readings, so no information leaks from the evaluation data into training. Standardisation is essential for the RBF kernel introduced in Section 3.5.6, which depends on the squared Euclidean distance $\|\mathbf{x} - \mathbf{z}\|^2 = \sum_j (x_j - z_j)^2$; without it, high-variance sensors such as MQ8 would dominate the distance and low-variance sensors such as MQ6 would be practically ignored.

3.5 Support Vector Machine: Mathematical Formulation

The Support Vector Machine (Boser et al., 1992; Cortes & Vapnik, 1995; Vapnik, 1995) is a supervised classifier grounded in statistical learning theory and convex optimisation. Its central idea is to separate two classes by the hyperplane that has the *largest margin*, that is, the greatest distance to the nearest training points. This section develops the theory for two classes and then extends it to non-linear boundaries and to the four-class gas problem.

3.5.1 Problem Setting and the Separating Hyperplane

Consider a binary training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with $\mathbf{x}_i \in \mathbb{R}^d$ and labels $y_i \in \{-1, +1\}$. A hyperplane in $\mathbb{R}^d$ is the set

$$H = \{\mathbf{x} \in \mathbb{R}^d: \mathbf{w}^\top \mathbf{x} + b = 0\} \quad (3.4)$$

where $\mathbf{w} \in \mathbb{R}^d$ is the normal vector and $b \in \mathbb{R}$ the bias. A new point is classified by the sign of $\mathbf{w}^\top \mathbf{x} + b$. To measure how far a point lies from H , write any $\mathbf{x}$ as $\mathbf{x} = \mathbf{x}_H + r\mathbf{w}/\|\mathbf{w}\|$ with $\mathbf{x}_H \in H$. Since $\mathbf{w}^\top \mathbf{x}_H + b = 0$, we obtain $\mathbf{w}^\top \mathbf{x} + b = r\|\mathbf{w}\|$, and hence the distance of $\mathbf{x}$ from H is

$$\text{dist}(\mathbf{x}, H) = \frac{|\mathbf{w}^\top \mathbf{x} + b|}{\|\mathbf{w}\|} \quad (3.5)$$

3.5.2 Hard-Margin SVM

If the classes are linearly separable, infinitely many hyperplanes separate them. Because $(\mathbf{w}, b)$ and $(t\mathbf{w}, tb)$ describe the same hyperplane for any $t > 0$, we fix the scale (canonical form) by requiring $\min_i |\mathbf{w}^\top \mathbf{x}_i + b| = 1$. Correct classification of every point with this scaling means

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \quad i = 1, \dots, n \quad (3.6)$$

The closest points then lie on the two parallel hyperplanes $\mathbf{w}^\top \mathbf{x} + b = \pm 1$, each at distance $1/\|\mathbf{w}\|$ from H by (3.5). The **margin**, the width of the empty band between them, is therefore

$$\rho = \frac{2}{\|\mathbf{w}\|} \quad (3.7)$$

Maximising $2/\|\mathbf{w}\|$ is equivalent to minimising $\frac{1}{2}\|\mathbf{w}\|^2$, a smooth, strictly convex function. The hard-margin SVM is thus the quadratic programme

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \quad i = 1, \dots, n \quad (3.8)$$

The geometry is illustrated in Figure 9(a). The points lying exactly on the margin hyperplanes are the **support vectors**; as shown below, they alone determine the solution.

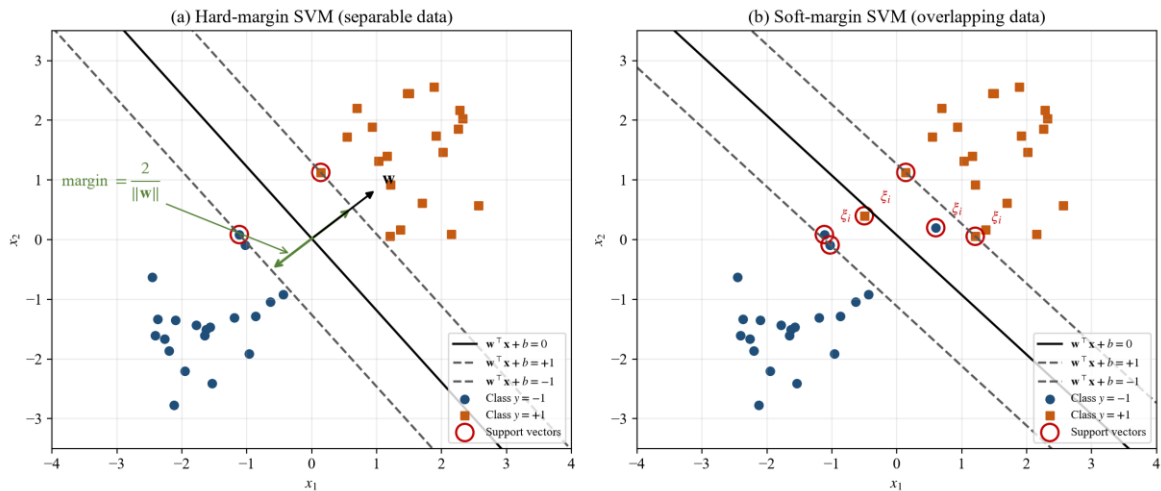


Figure 9: Geometry of (a) the hard-margin and (b) the soft-margin SVM. Circled points are support vectors

3.5.3 Soft-Margin SVM

Real sensor data overlap (as the NoGas and Perfume classes do), so a perfectly separating hyperplane may not exist. Cortes and Vapnik (1995) introduced non-negative **slack**

variables ξ_i that allow a point to violate the margin at a cost. The soft-margin (C-SVM) primal problem is

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \quad (3.9)$$

$$\text{subject to } y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n \quad (3.10)$$

Here ϕ is a feature map, introduced in Section 3.5.6 (for a linear SVM, $\phi(\mathbf{x}) = \mathbf{x}$). A point with $\xi_i = 0$ lies on or outside its margin; $0 < \xi_i \leq 1$ means it is inside the margin but correctly classified; and $\xi_i > 1$ means it is misclassified. Hence $\sum_i \xi_i$ is an upper bound on the number of training errors. The **regularisation parameter** $C > 0$ controls the trade-off: a large C penalises violations heavily and gives a narrow margin that fits the training data closely (risking over-fitting), whereas a small C gives a wide margin that tolerates more errors (risking under-fitting). At the optimum $\xi_i = \max\{0, 1 - y_i f(\mathbf{x}_i)\}$, so the problem is equivalent to minimising the regularised **hinge loss**

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \max\{0, 1 - y_i f(\mathbf{x}_i)\}, \quad f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b \quad (3.11)$$

3.5.4 The Lagrangian and the Dual Problem

The primal problem is solved through its dual. Introducing Lagrange multipliers $\alpha_i \geq 0$ for the margin constraints and $\mu_i \geq 0$ for $\xi_i \geq 0$, the Lagrangian is

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \xi, \alpha, \mu) = & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ & - \sum_{i=1}^n \alpha_i [y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) - 1 + \xi_i] - \sum_{i=1}^n \mu_i \xi_i \end{aligned} \quad (3.12)$$

At the optimum the Lagrangian is stationary with respect to the primal variables:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{0} \Rightarrow \mathbf{w} = \sum_{i=1}^n \alpha_i y_i \phi(\mathbf{x}_i) \quad (3.13)$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \sum_{i=1}^n \alpha_i y_i = 0 \quad (3.14)$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \Rightarrow C - \alpha_i - \mu_i = 0 \Rightarrow 0 \leq \alpha_i \leq C \quad (3.15)$$

Substituting (3.13)–(3.15) back into $\mathcal{L}$, the terms in b and ξ_i cancel and the inner products $\phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$ appear. Writing them as a kernel $K(\mathbf{x}_i, \mathbf{x}_j)$, we obtain the **dual problem**

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \quad (3.16)$$

$$\text{subject to } \sum_{i=1}^n \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, \dots, n \quad (3.17)$$

In matrix form this is $\min_{\alpha} \frac{1}{2} \alpha^\top Q \alpha - \mathbf{e}^\top \alpha$ with $Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{e} = (1, \dots, 1)^\top$.

When the kernel is positive semi-definite, Q is positive semi-definite and the dual is a convex quadratic programme. Since the primal constraints are affine and feasible, Slater's condition holds, so strong duality holds and the optimal values of the primal and dual coincide (Boyd & Vandenberghe, 2004). The dual has only n variables bounded in a box, and the data enter it only through kernel values, which is what makes the kernel trick possible.

3.5.5 Karush–Kuhn–Tucker Conditions and Support Vectors

For this convex problem the Karush–Kuhn–Tucker (KKT) conditions are necessary and sufficient for optimality (Boyd & Vandenberghe, 2004). Besides primal feasibility, dual feasibility ($\alpha_i, \mu_i \geq 0$) and stationarity (3.13)–(3.15), they include the **complementary slackness** conditions

$$\alpha_i [y_i f(\mathbf{x}_i) - 1 + \xi_i] = 0, \quad (C - \alpha_i) \xi_i = 0 \quad (3.18)$$

These conditions classify every training point into one of three groups (Table 9).

Table 9: Interpretation of the KKT conditions

Case	Consequence	Interpretation
$\alpha_i = 0$	$y_i f(\mathbf{x}_i) \geq 1$	not a support vector; no influence on the model
$0 < \alpha_i < C$	$\xi_i = 0, y_i f(\mathbf{x}_i) = 1$	free support vector, exactly on the margin
$\alpha_i = C$	$y_i f(\mathbf{x}_i) \leq 1$	bounded support vector: inside the margin or misclassified

By (3.13), $\mathbf{w}$ is a combination of the points with $\alpha_i > 0$ only, the support vectors, whose index set we denote by $\mathcal{S}$. This **sparsity** is the key to fast prediction. The bias b follows

from any free support vector s , for which $y_s f(\mathbf{x}_s) = 1$ and $y_s^2 = 1$; for numerical stability it is averaged over the set $\mathcal{F}$ of all free support vectors (Chang & Lin, 2011):

$$b = \frac{1}{|\mathcal{F}|} \sum_{s \in \mathcal{F}} (y_s - \sum_{i \in \mathcal{S}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_s)) \quad (3.19)$$

3.5.6 The Kernel Trick and the RBF Kernel

A linear boundary is too simple for overlapping gas classes. The SVM obtains non-linear boundaries by mapping the data into a higher-dimensional feature space $\mathcal{H}$ through $\phi: \mathbb{R}^d \rightarrow \mathcal{H}$ and separating them linearly there. Figure 10 shows a simple example: points that cannot be separated by any threshold on the real line become linearly separable after the map $\phi(x) = (x, x^2)$.

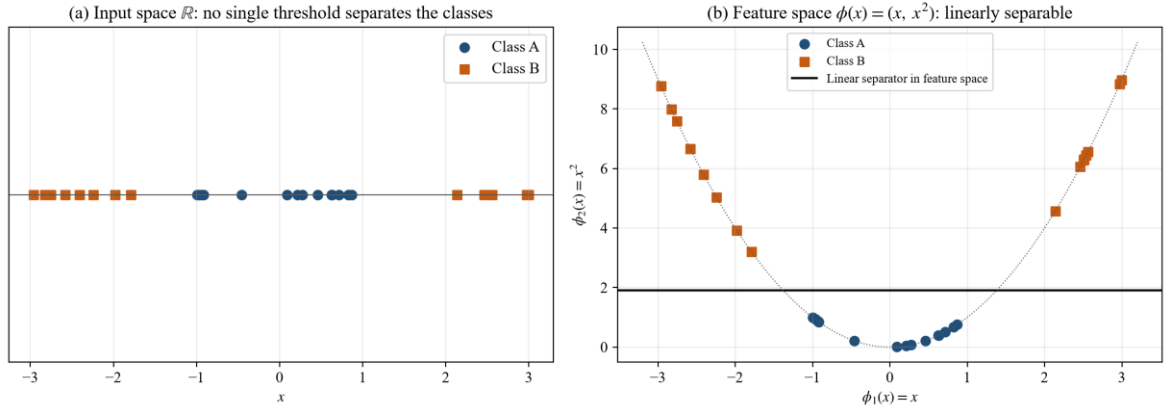


Figure 10: Illustration of the kernel idea: a mapping to a higher-dimensional space makes the classes linearly separable

Because the dual (3.16) and the decision function use the data only through inner products, ϕ never needs to be computed explicitly; it suffices to evaluate a **kernel** $K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle_{\mathcal{H}}$. This is the *kernel trick* (Boser et al., 1992). By **Mercer's theorem** (Mercer, 1909), a continuous symmetric function K is a valid kernel if and only if it is positive semi-definite; equivalently, for every finite set of points the Gram matrix $G_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ satisfies $\mathbf{c}^T G \mathbf{c} \geq 0$ for all $\mathbf{c} \in \mathbb{R}^n$ (Schölkopf & Smola, 2002). This property guarantees that the dual remains convex. The three kernels considered in this study are

$$\text{Linear: } K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z} \quad (3.20)$$

$$\text{Polynomial: } K(\mathbf{x}, \mathbf{z}) = (\gamma \mathbf{x}^T \mathbf{z} + r)^p \quad (3.21)$$

$$\text{RBF (Gaussian): } K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2), \gamma > 0 \quad (3.22)$$

The radial basis function (RBF) kernel corresponds to an infinite-dimensional feature space, satisfies $0 < K \leq 1$ with $K(\mathbf{x}, \mathbf{x}) = 1$, and yields a strictly positive definite Gram matrix for distinct points. The parameter γ controls how far the influence of a single training point reaches. A small γ gives a wide kernel and a smooth, almost linear boundary (under-fitting); a large γ gives a narrow kernel, a highly flexible boundary and many support vectors (over-fitting). Figure 11 illustrates this on a two-dimensional example. Because C and γ interact, they must be tuned jointly (Section 3.6).

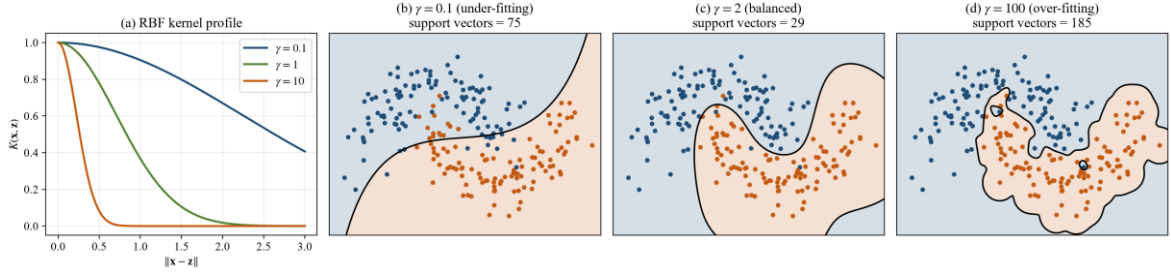


Figure 11: (a) RBF kernel value against distance for different γ ; (b)–(d) effect of γ on the decision boundary of an RBF-SVM

3.5.7 The Decision Function

Substituting (3.13) into $f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b$ and applying the kernel gives the decision function of the trained binary SVM:

$$f(\mathbf{x}) = \sum_{i \in \mathcal{S}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b, \quad \hat{y} = \text{sign}(f(\mathbf{x})) \quad (3.23)$$

The magnitude $|f(\mathbf{x})|$ is proportional to the distance from the decision boundary in feature space and therefore serves as a confidence score. Only the support vectors contribute to the sum.

3.5.8 Multi-Class Classification: One-vs-One Voting

The gas problem has $k = 4$ classes. The SVM implementation used here (LIBSVM, through scikit-learn) uses the **one-vs-one** (OvO) strategy (Kreßel, 1999; Hsu & Lin, 2002): one binary SVM f_{pq} is trained for every pair of classes (p, q) , using only the samples of those two classes. The number of binary classifiers is

$$N_{\text{OvO}} = \frac{k(k-1)}{2} = \frac{4 \times 3}{2} = 6 \quad (3.24)$$

A new reading is presented to all six classifiers; each casts a vote for one of its two classes, and the class with the most votes is predicted:

$$v_p(\mathbf{x}) = \sum_{q \neq p} \mathbb{1}[f_{pq}(\mathbf{x}) \text{ votes for } p], \quad \hat{c}(\mathbf{x}) = \underset{p \in \{1, \dots, k\}}{\operatorname{argmax}} v_p(\mathbf{x}) \quad (3.25)$$

Compared with one-vs-rest, each OvO sub-problem is smaller and balanced, which makes training efficient (Hsu & Lin, 2002). The procedure is shown in Figure 12.

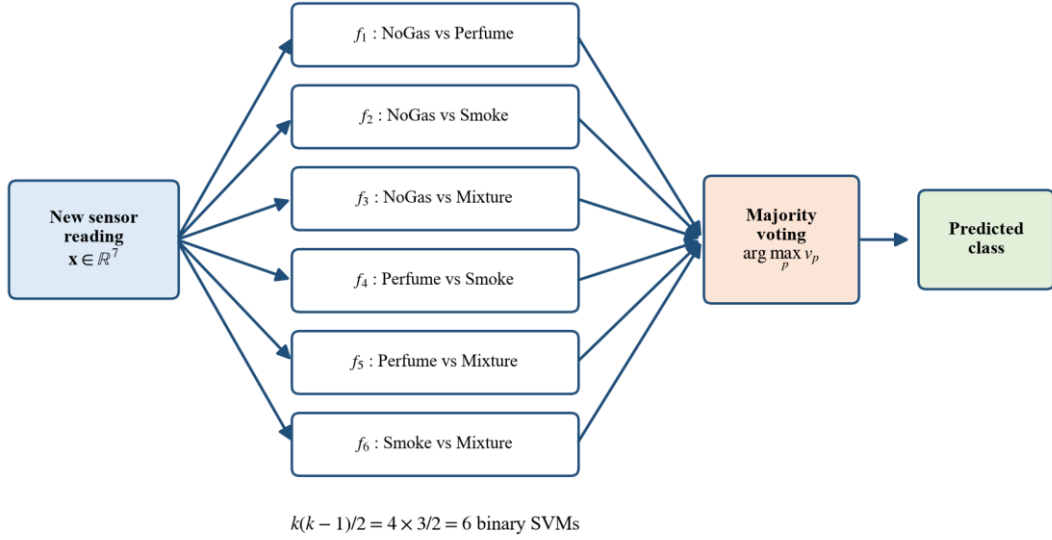


Figure 12: One-vs-one decomposition of the four-class gas problem into six binary SVMs with majority voting

3.5.9 Probability Estimates (Platt Scaling)

ROC analysis and confidence reporting require class probabilities rather than votes. Platt (2000) proposed fitting a sigmoid to the decision values of a binary SVM,

$$P(y = 1 \mid \mathbf{x}) \approx \frac{1}{1 + \exp(Af(\mathbf{x}) + B)} \quad (3.26)$$

where A and B minimise the cross-entropy on held-out decision values obtained by internal cross-validation (Lin et al., 2007). For several classes, the pairwise probabilities are combined by pairwise coupling (Wu et al., 2004), which yields a probability for each of the four gas classes.

3.5.10 Solving the Dual: Sequential Minimal Optimisation

The dual (3.16) is solved with an SMO-type decomposition algorithm (Platt, 1998; Chang & Lin, 2011). Because of the equality constraint (3.14), at least two multipliers must change together; SMO repeatedly selects a pair (α_i, α_j) that violates the KKT conditions,

solves the resulting two-variable problem analytically, clips the solution to the box $[0, C]$, and stops when all KKT conditions hold within a tolerance. The complete training and prediction procedure is summarised in Figure 13.

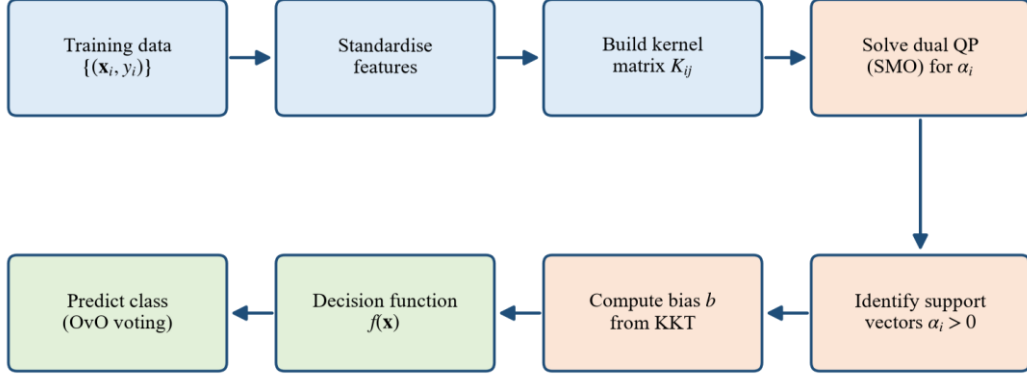


Figure 13: Training and prediction algorithm of the SVM classifier

3.5.11 Computational Complexity and Real-Time Suitability

Prediction with (3.23) requires one kernel evaluation per support vector, each costing $O(d)$ operations (a squared distance and an exponential). For the OvO model, each distinct support vector's kernel value is computed once and reused by the pairwise classifiers, so the cost of classifying one reading is

$$T_{\text{predict}} = O(n_{SV}d) + O(n_{SV}(k-1)) + O(k^2) \approx O(n_{SV}d) \quad (3.27)$$

where n_{SV} is the total number of support vectors. The cost depends on n_{SV} , not on the size of the training set, and with $d = 7$ and around 10^3 support vectors a prediction needs only about 10^4 floating-point operations. Training, which scales between $O(dn^2)$ and $O(dn^3)$, is carried out offline once. This analysis predicts that the SVM is fast enough for real-time use; the claim is verified experimentally in Chapter 4.

3.6 Hyper-Parameter Tuning by Grid Search and Cross-Validation

The RBF-SVM has two hyper-parameters, C and γ , that are not learned by the optimisation and must be chosen. Following the practical guide of Hsu et al. (2003), they were selected by an exhaustive **grid search** with **stratified five-fold cross-validation** on the training set (Kohavi, 1995). The training set is split into five folds $\mathcal{D}_1, \dots, \mathcal{D}_5$ with equal class proportions; for each candidate $\theta = (C, \gamma)$ the model is trained on four folds and scored on the fifth, and the five scores are averaged (Figure 14):

$$CV(\theta) = \frac{1}{5} \sum_{m=1}^5 \text{Acc}(\hat{f}_{\theta}^{(-m)}; \mathcal{D}_m), \quad \theta^* = \underset{\theta \in \Theta}{\operatorname{argmax}} CV(\theta) \quad (3.28)$$

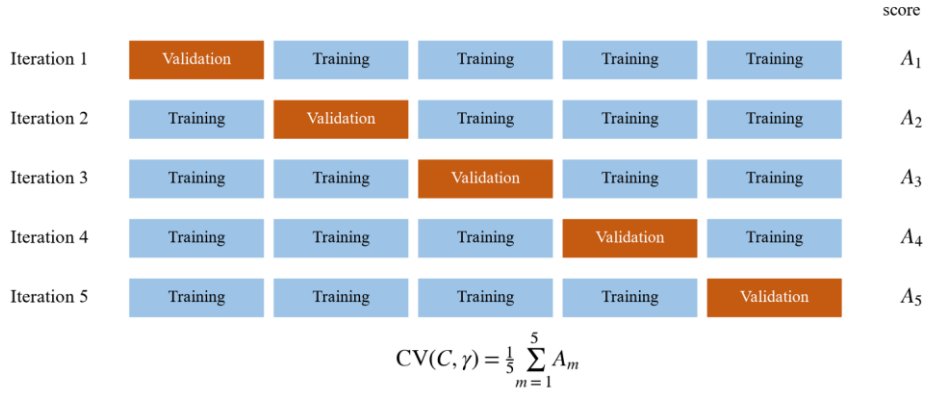


Figure 14: Stratified five-fold cross-validation used to score each (C, γ) pair

Standardisation is fitted inside each training fold (as a pipeline), so the validation fold never influences the scaling. The search grid is listed in Table 10; it spans six orders of magnitude for each parameter, giving 42 combinations and 210 model fits. The best pair θ^* is then refitted on the whole training set.

Table 10: Hyper-parameter search space

Hyper-parameter	Candidate values	Role
C	0.1, 1, 10, 100, 1000, 10000	penalty on margin violations
γ	0.001, 0.01, 0.1, 1, 10, 30, 100	width of the RBF kernel
Folds	5 (stratified, shuffled, seed 42)	cross-validation
Scoring	accuracy (classes are balanced)	selection criterion

For comparison within the SVM family (a kernel ablation), the linear kernel was tuned over $C \in \{0.1, 1, 10, 100\}$ and the polynomial kernel over the same C values with degree $p \in \{2, 3\}$.

3.7 Performance Evaluation Metrics

Let $M \in \mathbb{N}^{k \times k}$ be the **confusion matrix**, whose entry M_{pq} counts the samples of true class p predicted as class q . For class p , treated as positive against the rest, the true positives, false positives and false negatives are

$$TP_p = M_{pp}, \quad FP_p = \sum_{q \neq p} M_{qp}, \quad FN_p = \sum_{q \neq p} M_{pq} \quad (3.29)$$

and $TN_p = N - TP_p - FP_p - FN_p$, where N is the number of evaluated samples. The following measures are used (Sokolova & Lapalme, 2009):

$$\text{Accuracy} = \frac{\sum_{p=1}^k M_{pp}}{N} \quad (3.30)$$

$$\text{Precision}_p = \frac{TP_p}{TP_p + FP_p}, \quad \text{Recall}_p = \frac{TP_p}{TP_p + FN_p} \quad (3.31)$$

$$F1_p = \frac{2 \cdot \text{Precision}_p \cdot \text{Recall}_p}{\text{Precision}_p + \text{Recall}_p}, \quad \text{Macro F1} = \frac{1}{k} \sum_{p=1}^k F1_p \quad (3.32)$$

Precision measures how trustworthy an alarm for class p is, recall measures how many true cases of class p are caught, and F1 is their harmonic mean. Macro averages weight all classes equally. For safety, recall of the gas classes is critical, because a false negative is a missed leak.

The **receiver operating characteristic (ROC)** curve of class p plots the true positive rate against the false positive rate as a decision threshold τ on the predicted probability is varied (Fawcett, 2006):

$$\text{TPR}_p(\tau) = \frac{TP_p(\tau)}{TP_p(\tau) + FN_p(\tau)}, \quad \text{FPR}_p(\tau) = \frac{FP_p(\tau)}{FP_p(\tau) + TN_p(\tau)} \quad (3.33)$$

The area under the curve (AUC) equals the probability that a randomly chosen sample of class p receives a higher score than a randomly chosen sample of another class; 1 is perfect and 0.5 is chance. Finally, to interpret the system as a **leak detector**, the classes are collapsed into *no leak* (NoGas) and *gas present* (Perfume, Smoke, Mixture), and the sensitivity (detection rate) and specificity are computed:

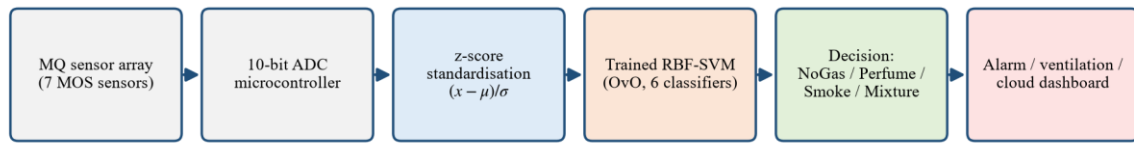
$$\text{Sensitivity} = \frac{TP}{TP + FN}, \quad \text{Specificity} = \frac{TN}{TN + FP} \quad (3.34)$$

$$\text{False alarm rate} = \frac{FP}{FP + TN} = 1 - \text{Specificity}$$

Real-time performance is measured as the wall-clock **latency** of classifying a single reading, including standardisation, averaged over 500 consecutive test readings fed one at a time.

3.8 Real-Time Deployment Concept

Figure 15 shows how the trained model would operate in a manufacturing or logistics facility. Every 2 s the microcontroller reads the seven sensors through its ADC, standardises the vector with the stored training means and standard deviations, and evaluates the six pairwise SVMs. The predicted class (and its probability) drives local actions, such as an alarm or ventilation, and is logged to a cloud dashboard. Because the deployed model consists only of the support vectors, their coefficients, the six biases and the 14 scaling constants, it can be exported to embedded hardware (Boubin & Shrestha, 2019).



every 2 s a new reading arrives; SVM inference takes about 1 ms

Figure 15: Real-time deployment pipeline of the proposed SVM gas detector

3.9 Python Code for Training the SVM

The experiments were implemented in Python using scikit-learn (Pedregosa et al., 2011), whose SVC class wraps LIBSVM (Chang & Lin, 2011), together with NumPy (Harris et al., 2020), pandas (McKinney, 2010), Matplotlib (Hunter, 2007) and Seaborn (Waskom, 2021). The complete script is listed below. It runs unchanged in Google Colab after uploading the dataset file to /content.

```

"""
Real-Time Gas Leak Detection Using Support Vector Machine
Dataset : MultimodalGasData (Narkhede et al., 2022) - gas sensor measurements only
Model   : Support Vector Machine (baseline linear SVM -> tuned RBF-kernel SVM)

Run in Google Colab or locally with Python 3 and scikit-learn
Outputs figures to /content/figures and numeric results to /content/results
"""

```

```

# STEP 0: IMPORT LIBRARIES
import os
import json
import time
import numpy as np
import pandas as pd
import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.colors import ListedColormap
import warnings
warnings.filterwarnings("ignore")

from sklearn.model_selection import train_test_split, GridSearchCV, StratifiedKFold,
learning_curve
from sklearn.preprocessing import StandardScaler, label_binarize
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.metrics import (accuracy_score, precision_recall_fscore_support,
confusion_matrix,
                             classification_report, roc_curve, auc, roc_auc_score)

DATA = "/content/Gas_Sensors_Measurements.csv" # upload the dataset file to Colab
FIG = "/content/figures"
RES = "/content/results"
os.makedirs(FIG, exist_ok=True)
os.makedirs(RES, exist_ok=True)

SEED = 42
SENSORS = ["MQ2", "MQ3", "MQ5", "MQ6", "MQ7", "MQ8", "MQ135"]
CLASSES = ["NoGas", "Perfume", "Smoke", "Mixture"]
plt.rcParams.update({"font.family": "Times New Roman", "font.size": 11, "figure.dpi": 150})
results = {}

# STEP 1: LOAD THE DATASET
df = pd.read_csv(DATA)
print(df.head())
print(df["Gas"].value_counts())
results["n_samples"] = int(len(df))
results["class_counts"] = df["Gas"].value_counts().to_dict()

X = df[SENSORS].values.astype(float)
y = df["Gas"].map({c: i for i, c in enumerate(CLASSES)}).values

# STEP 2: EXPLORATORY DATA ANALYSIS
desc = df.groupby("Gas")[SENSORS].agg(["mean", "std"]).round(2)
desc.to_csv(os.path.join(RES, "class_wise_statistics.csv"))
df[SENSORS].describe().round(2).to_csv(os.path.join(RES, "descriptive_statistics.csv"))

# Box plots of each sensor by class
fig, axes = plt.subplots(2, 4, figsize=(14, 6.5))
for ax, s in zip(axes.flat, SENSORS):
    sns.boxplot(data=df, x="Gas", y=s, order=CLASSES, ax=ax, palette="Set2", hue="Gas",
legend=False)
    ax.set_title(s); ax.set_xlabel(""); ax.set_ylabel("Sensor reading (ADC)")
    ax.tick_params(axis="x", rotation=20)
axes.flat[-1].axis("off")
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_boxplots.png")); plt.close()

# Sensor signal over time (sample index) for each class
fig, axes = plt.subplots(2, 2, figsize=(13, 7), sharey=True)
for ax, c in zip(axes.flat, CLASSES):
    sub = df[df["Gas"] == c]
    for s in SENSORS:
        ax.plot(sub["Serial Number"].values, sub[s].values, lw=0.8, label=s)
    ax.set_title(f"Class: {c}"); ax.set_xlabel("Sample index (time order)");
ax.set_ylabel("Reading (ADC)")
axes.flat[0].legend(ncol=4, fontsize=8)

```

```

plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_signals.png")); plt.close()

# Correlation heat map
plt.figure(figsize=(7, 5.5))
sns.heatmap(df[SENSORS].corr(), annot=True, fmt=".2f", cmap="coolwarm", vmin=-1, vmax=1)
plt.title("Pearson Correlation Between Gas Sensors")
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_correlation.png")); plt.close()
results["correlation"] = df[SENSORS].corr().round(3).to_dict()

# STEP 3: TRAIN / VALIDATION / TEST SPLIT (70 / 15 / 15, stratified)
X_train, X_tmp, y_train, y_tmp = train_test_split(X, y, test_size=0.30, stratify=y,
random_state=SEED)
X_val, X_test, y_val, y_test = train_test_split(X_tmp, y_tmp, test_size=0.50, stratify=y_tmp,
random_state=SEED)
results["split"] = {"train": len(y_train), "validation": len(y_val), "test": len(y_test)}
print("Split:", results["split"])

# STEP 4: PCA VISUALISATION (on standardised training data)
sc_vis = StandardScaler().fit(X_train)
pca = PCA(n_components=7).fit(sc_vis.transform(X_train))
results["pca_explained_variance_ratio"] = pca.explained_variance_ratio_.round(4).tolist()
Z = pca.transform(sc_vis.transform(X_train))[:, :2]
plt.figure(figsize=(7, 5.5))
for i, c in enumerate(CLASSES):
    plt.scatter(Z[y_train == i, 0], Z[y_train == i, 1], s=6, alpha=0.6, label=c)
plt.xlabel(f"PC1 ({pca.explained_variance_ratio_[0]*100:.1f}% variance)")
plt.ylabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.1f}% variance)")
plt.title("PCA Projection of Standardised Sensor Data"); plt.legend(markerscale=3)
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_pca.png")); plt.close()

def evaluate(model, Xs, ys, name):
    """Return metrics dictionary for a fitted model on a data split."""
    yp = model.predict(Xs)
    p, r, f, _ = precision_recall_fscore_support(ys, yp, average="macro", zero_division=0)
    return {"split": name, "accuracy": accuracy_score(ys, yp), "precision_macro": p,
            "recall_macro": r, "f1_macro": f}

# STEP 5: BASELINE MODEL - LINEAR SVM ON RAW (UNSCALED) READINGS
baseline = SVC(kernel="linear", C=1.0, random_state=SEED, max_iter=200000)
t0 = time.perf_counter(); baseline.fit(X_train, y_train); t_base = time.perf_counter() - t0
results["baseline"] = {"train_time_s": t_base,
                        "metrics": [evaluate(baseline, X_train, y_train, "train"),
                                    evaluate(baseline, X_val, y_val, "validation"),
                                    evaluate(baseline, X_test, y_test, "test")],
                        "n_support_vectors": int(baseline.n_support_.sum())}
print("Baseline:", results["baseline"]["metrics"])

# STEP 6: IMPROVED MODEL - STANDARDISATION + RBF SVM WITH GRID SEARCH (5-FOLD CV)
pipe = Pipeline([("scaler", StandardScaler()),
                 ("svm", SVC(kernel="rbf", probability=True, random_state=SEED))])
param_grid = {"svm__C": [0.1, 1, 10, 100, 1000, 10000],
              "svm__gamma": [0.001, 0.01, 0.1, 1, 10, 30, 100]}
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=SEED)
grid = GridSearchCV(pipe, param_grid, cv=cv, scoring="accuracy", n_jobs=-1,
return_train_score=True)
t0 = time.perf_counter(); grid.fit(X_train, y_train); t_grid = time.perf_counter() - t0
best = grid.best_estimator_
results["grid"] = {"best_params": grid.best_params_, "best_cv_accuracy": grid.best_score_,
                  "search_time_s": t_grid}
print("Best params:", grid.best_params_, "CV acc:", grid.best_score_)

# Grid-search heat map (mean CV accuracy for each C, gamma)
cvres = pd.DataFrame(grid.cv_results_)
heat = cvres.pivot_table(index="param_svm__C", columns="param_svm__gamma",
values="mean_test_score")
heat.to_csv(os.path.join(RES, "grid_search_cv_accuracy.csv"))
heat.index = [f"{v:g}" for v in heat.index]
heat.columns = [f"{v:g}" for v in heat.columns]

```

```

plt.figure(figsize=(7, 5))
sns.heatmap(heat, annot=True, fmt=".3f", cmap="viridis")
plt.xlabel(r"Kernel parameter $\gamma$"); plt.ylabel("Regularisation parameter C")
plt.title("5-Fold Cross-Validation Accuracy (RBF-SVM Grid Search)")
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_grid_heatmap.png")); plt.close()

# Also compare kernels at their own best settings (all SVM - kernel ablation)
kernel_rows = []
for kern, grid_k in {"linear": {"svm_C": [0.1, 1, 10, 100]},
                    "poly": {"svm_C": [0.1, 1, 10, 100], "svm_degree": [2, 3]},
                    "rbf": param_grid}.items():
    pk = Pipeline([("scaler", StandardScaler()), ("svm", SVC(kernel=kern,
random_state=SEED))])
    gk = GridSearchCV(pk, grid_k, cv=cv, scoring="accuracy", n_jobs=-1).fit(X_train, y_train)
    kernel_rows.append({"kernel": kern, "best_params": str(gk.best_params_),
                      "cv_accuracy": gk.best_score_,
                      "test_accuracy": accuracy_score(y_test, gk.predict(X_test))})
pd.DataFrame(kernel_rows).to_csv(os.path.join(RES, "kernel_comparison.csv"), index=False)
results["kernel_comparison"] = kernel_rows

# STEP 7: EVALUATE THE IMPROVED MODEL
results["improved"] = {"metrics": [evaluate(best, X_train, y_train, "train"),
                                  evaluate(best, X_val, y_val, "validation"),
                                  evaluate(best, X_test, y_test, "test")],
                      "n_support_vectors": int(best.named_steps["svm"].n_support_.sum()),
                      "support_vectors_per_class":
best.named_steps["svm"].n_support_.tolist())
print("Improved:", results["improved"]["metrics"])

y_pred = best.predict(X_test)
report = classification_report(y_test, y_pred, target_names=CLASSES, digits=4,
output_dict=True)
results["classification_report_test"] = report
with open(os.path.join(RES, "classification_report_test.txt"), "w") as fh:
    fh.write(classification_report(y_test, y_pred, target_names=CLASSES, digits=4))

# Binary leak-detection view: NoGas = no leak (0); Perfume, Smoke, Mixture = gas present (1)
yb_true, yb_pred = (y_test != 0).astype(int), (y_pred != 0).astype(int)
tn, fp, fn, tp = confusion_matrix(yb_true, yb_pred).ravel()
results["binary_leak_view"] = {"TP": int(tp), "TN": int(tn), "FP": int(fp), "FN": int(fn),
                              "sensitivity": tp / (tp + fn), "specificity": tn / (tn + fp),
                              "false_alarm_rate": fp / (fp + tn), "missed_leak_rate": fn /
(fn + tp),
                              "accuracy": (tp + tn) / len(yb_true)}
print("Binary leak view:", results["binary_leak_view"])

for name, model in [("baseline", baseline), ("improved", best)]:
    cm = confusion_matrix(y_test, model.predict(X_test))
    results[f"confusion_matrix_{name}"] = cm.tolist()
    plt.figure(figsize=(6, 5))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", cbar=False, xticklabels=CLASSES,
yticklabels=CLASSES)
    plt.xlabel("Predicted Label"); plt.ylabel("True Label")
    plt.title(f"Confusion Matrix on Test Data ({'Linear SVM' if name == 'baseline' else 'RBF-
SVM'})")
    plt.tight_layout(); plt.savefig(os.path.join(FIG, f"fig_confusion_{name}.png"));
plt.close()

# One-vs-rest ROC curves
y_score = best.predict_proba(X_test)
y_bin = label_binarize(y_test, classes=range(4))
plt.figure(figsize=(6.5, 5.5))
roc_auc = {}
for i, c in enumerate(CLASSES):
    fpr, tpr, _ = roc_curve(y_bin[:, i], y_score[:, i])
    roc_auc[c] = auc(fpr, tpr)
    plt.plot(fpr, tpr, lw=2, label=f"{c} (AUC = {roc_auc[c]:.4f})")
plt.plot([0, 1], [0, 1], "k--", label="Chance level")
plt.xlabel("False Positive Rate"); plt.ylabel("True Positive Rate")
plt.title("One-vs-Rest ROC Curves (RBF-SVM, Test Data)"); plt.legend(loc="lower right")

```

```

plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_roc.png")); plt.close()
results["roc_auc_per_class"] = roc_auc
results["roc_auc_macro"] = roc_auc_score(y_bin, y_score, average="macro", multi_class="ovr")

# Learning curve
sizes, tr_sc, va_sc = learning_curve(best, X_train, y_train, cv=cv, n_jobs=-1,
                                     train_sizes=np.linspace(0.1, 1.0, 8), scoring="accuracy")

plt.figure(figsize=(7, 5))
plt.plot(sizes, tr_sc.mean(1), "o-", label="Training accuracy")
plt.plot(sizes, va_sc.mean(1), "o-", label="Cross-validation accuracy")
plt.fill_between(sizes, va_sc.mean(1) - va_sc.std(1), va_sc.mean(1) + va_sc.std(1), alpha=0.2)
plt.xlabel("Number of training samples"); plt.ylabel("Accuracy"); plt.title("Learning Curve of RBF-SVM")
plt.legend(); plt.grid(alpha=0.3)
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_learning_curve.png")); plt.close()
results["learning_curve"] = {"sizes": sizes.tolist(), "train": tr_sc.mean(1).tolist(), "cv": va_sc.mean(1).tolist()}

# STEP 8: DECISION REGIONS IN 2-D PCA SPACE (illustration of the SVM boundary)
vis = Pipeline([("scaler", StandardScaler()), ("pca", PCA(n_components=2)),
                ("svm", SVC(kernel="rbf", C=grid.best_params_["svm_C"],
                           gamma="scale"))]).fit(X_train, y_train)
Ztr = vis[0].transform(X_train)
xx, yy = np.meshgrid(np.linspace(Ztr[:, 0].min() - 1, Ztr[:, 0].max() + 1, 400),
                     np.linspace(Ztr[:, 1].min() - 1, Ztr[:, 1].max() + 1, 400))
zz = vis.named_steps["svm"].predict(np.c_[xx.ravel(), yy.ravel()]).reshape(xx.shape)
plt.figure(figsize=(7.5, 6))
plt.contourf(xx, yy, zz, alpha=0.20, cmap=ListedColormap(["C0", "C1", "C2", "C3"]),
             levels=[-0.5, 0.5, 1.5, 2.5, 3.5])
for i, c in enumerate(CLASSES):
    plt.scatter(Ztr[y_train == i, 0], Ztr[y_train == i, 1], s=5, label=c)
plt.xlabel("Principal Component 1"); plt.ylabel("Principal Component 2")
plt.title("RBF-SVM Decision Regions in 2-D PCA Space"); plt.legend(markerscale=3)
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_decision_regions.png")); plt.close()

# STEP 9: REAL-TIME INFERENCE LATENCY (one reading at a time, like a live sensor stream)
stream = X_test[:500]
for m in (best,):
    m.predict(stream[:1]) # warm-up
lat = []
for row in stream:
    t0 = time.perf_counter(); best.predict(row.reshape(1, -1));
lat.append((time.perf_counter() - t0) * 1000)
lat = np.array(lat)
t0 = time.perf_counter(); best.predict(X_test); batch_ms = (time.perf_counter() - t0) * 1000 / len(X_test)
results["latency_ms"] = {"single_mean": lat.mean(), "single_median": float(np.median(lat)),
                        "single_p95": float(np.percentile(lat, 95)), "single_max": lat.max(),
                        "batch_per_sample": batch_ms}

plt.figure(figsize=(7, 4.5))
plt.hist(lat, bins=40, color="steelblue", edgecolor="white")
plt.axvline(lat.mean(), color="red", ls="--", label=f"Mean = {lat.mean():.3f} ms")
plt.xlabel("Prediction time per reading (ms)"); plt.ylabel("Frequency")
plt.title("Single-Reading Inference Latency of RBF-SVM"); plt.legend()
plt.tight_layout(); plt.savefig(os.path.join(FIG, "fig_latency.png")); plt.close()

# Simulated real-time stream: walk through a test sequence and record predictions
sim_rows = []
for c in CLASSES:
    sub = df[df["Gas"] == c].sort_values("Serial Number").iloc[:200].head(2)
    for _, r in sub.iterrows():
        x = r[SENSORS].values.astype(float).reshape(1, -1)
        prob = best.predict_proba(x)[0]
        sim_rows.append({"sample": f"{r['Serial Number']}_c", "true": c, "predicted":
                        CLASSES[int(prob.argmax())],
                        **{f"P({k})": round(float(v), 4) for k, v in zip(CLASSES, prob)}})
pd.DataFrame(sim_rows).to_csv(os.path.join(RES, "sample_predictions.csv"), index=False)
results["sample_predictions"] = sim_rows

# STEP 10: ROBUSTNESS CHECK - CHRONOLOGICAL (BLOCK) SPLIT

```

```

# Readings are a time series; random splits put neighbouring readings in train and test.
# Here the first 70% of each class (in time order) trains the model and the last 30% tests it.
tr_idx, te_idx = [], []
for c in CLASSES:
    idx = df.index[df["Gas"] == c].values
    cut = int(0.7 * len(idx)); tr_idx += list(idx[:cut]); te_idx += list(idx[cut:])
chron = Pipeline([("scaler", StandardScaler()),
                  ("svm", SVC(kernel="rbf", C=grid.best_params_["svm_C"],
                             gamma=grid.best_params_["svm__gamma"])]))
chron.fit(X[tr_idx], y[tr_idx])
yc = chron.predict(X[te_idx])
p, r, f, _ = precision_recall_fscore_support(y[te_idx], yc, average="macro", zero_division=0)
results["chronological_split"] = {"accuracy": accuracy_score(y[te_idx], yc),
                                  "precision_macro": p,
                                  "recall_macro": r, "f1_macro": f,
                                  "confusion_matrix": confusion_matrix(y[te_idx],
                                yc).tolist())
print("Chronological split:", results["chronological_split"])

with open(os.path.join(RES, "results.json"), "w") as fh:
    json.dump(results, fh, indent=2, default=lambda o: o.item() if hasattr(o, "item") else
    str(o))
print("Done. Results in", RES)

```

CHAPTER 4: RESULTS AND DISCUSSION

This chapter reports the experimental results. It first describes the computing environment, then presents the baseline model, the hyper-parameter search and the performance of the final RBF-SVM. The model is then examined as a leak detector, in terms of real-time latency and under a chronological robustness test, and finally compared with previous studies.

4.1 Computational Environment and Tools

All experiments were run on an ordinary laptop without a graphics processor, deliberately, to show that the method does not require specialised hardware. Table 11 lists the environment. The same script also runs in Google Colab.

Table 11: Computational environment

Item	Specification
Processor	Intel Core i3-6006U, 2.0 GHz, 2 cores (no GPU used)
Memory	8 GB RAM
Operating system	Microsoft Windows 10 Pro
Language	Python 3.14
Machine learning	scikit-learn 1.9.1 (SVC, backed by LIBSVM)
Numerical / data	NumPy 2.5.2, pandas 3.0.6, SciPy 1.18.1
Visualisation	Matplotlib 3.11.1, Seaborn 0.13.2
Reproducibility	fixed random seed 42 for splitting, cross-validation and SVM probability fitting

4.2 Baseline Model: Linear SVM on Raw Readings

The baseline is a linear SVM ($C = 1$) trained directly on the unscaled ADC readings. It reached a test accuracy of 80.31% and a macro F1-score of 0.8023, with similar training (77.81%) and validation (78.65%) accuracy, which indicates under-fitting rather than over-fitting: a linear boundary is simply too rigid for this problem. The confusion matrix in Figure 16 shows where the errors occur.



Figure 16: Confusion matrix of the baseline linear SVM on the test data

Smoke and Mixture are recognised perfectly even by the linear model, because their sensor signatures are distinct (Section 3.3). All errors are between NoGas and Perfume: 79 NoGas readings were labelled Perfume and 110 Perfume readings were labelled NoGas, so these two classes were recognised with only 67.1% and 54.2% recall. This agrees with the exploratory analysis, in which the two classes had nearly identical means and overlapping PCA clusters. Separating them requires a non-linear boundary.

4.3 Hyper-Parameter Tuning Results

The grid search evaluated 42 combinations of C and γ with five-fold cross-validation (210 fits, about 9 minutes on the laptop). Figure 17 shows the mean cross-validation accuracy of every combination. The best pair was $C = 100$ and $\gamma = 10$, with a cross-validation accuracy of 97.63%.

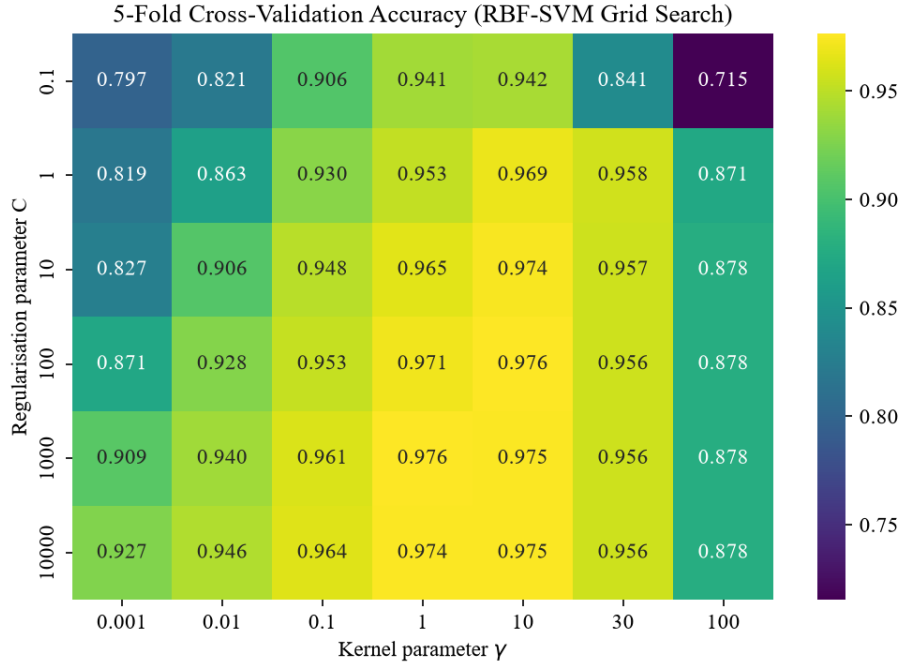


Figure 17: Five-fold cross-validation accuracy of the RBF-SVM for each (C, γ) pair

The heat map confirms the theory of Section 3.5.6. With a very small γ (0.001) the kernel is so wide that the model is nearly linear, and accuracy stays at 80–93%. Accuracy rises as γ increases towards 1–10, where the boundary becomes flexible enough to separate NoGas from Perfume. Beyond $\gamma = 10$ accuracy falls again (to about 88% at $\gamma = 100$), because each support vector influences only a tiny neighbourhood and the model memorises noise. For a fixed γ , increasing C helps up to about 100, after which the gain saturates. Because the optimum lies in the interior of the grid, the search range was adequate.

4.4 Performance of the Tuned RBF-SVM

4.4.1 Overall Improvement

Refitted on the full training set with $C = 100$ and $\gamma = 10$, the RBF-SVM achieved 98.23% accuracy on the unseen test set, an improvement of 17.92 percentage points over the baseline. Table 12 compares the two models.

Table 12: Performance before and after improvement

Metric	Baseline (linear SVM, raw data)	Improved (RBF-SVM, standardised)
Training accuracy	77.81%	99.98%
Validation accuracy	78.65%	98.12%
Test accuracy	80.31%	98.23%
Test macro precision	0.8040	0.9823
Test macro recall	0.8031	0.9823

Metric	Baseline (linear SVM, raw data)	Improved (RBF-SVM, standardised)
Test macro F1-score	0.8023	0.9823
Number of support vectors	1,047	1,304

The validation (98.12%) and test (98.23%) accuracies agree closely, and both are close to the cross-validation estimate (97.63%), which shows that the model generalises consistently to unseen data from the same recording sessions. The training accuracy of 99.98% is higher, as expected for a flexible kernel model, but the gap to the test accuracy is under two percentage points. The number of support vectors rose from 1,047 to 1,304, reflecting the more detailed boundary needed between NoGas and Perfume; even so, only 29.1% of the 4,480 training readings are support vectors, so the model remains compact.

4.4.2 Confusion Matrix

Figure 18 shows the confusion matrix of the tuned model on the 960 test readings.

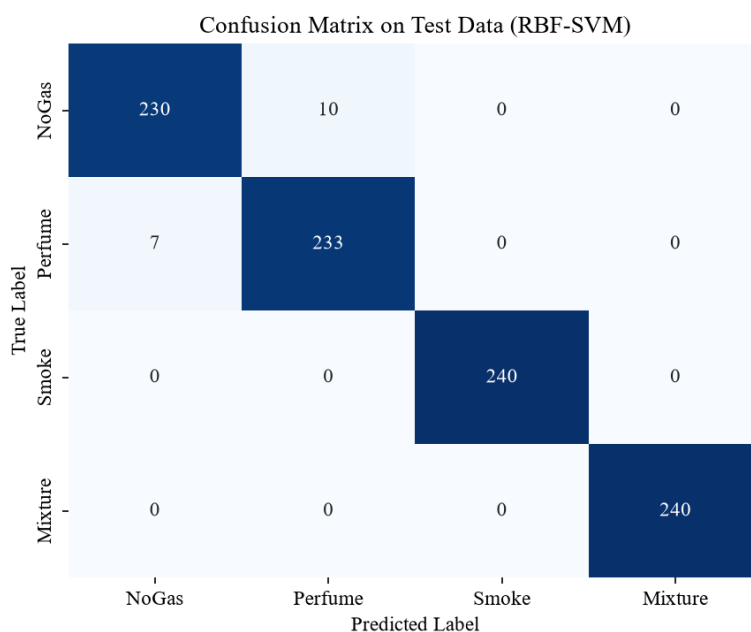


Figure 18: Confusion matrix of the tuned RBF-SVM on the test data

Smoke and Mixture remain perfectly classified (240 of 240 each). The NoGas–Perfume confusion, which caused 189 errors in the baseline, fell to only 17 errors (10 NoGas readings predicted as Perfume and 7 Perfume readings predicted as NoGas). No gas-containing reading was ever confused with Smoke or Mixture, and every misclassification involves the two chemically weakest conditions.

4.4.3 Precision, Recall and F1-Score

Table 13 gives the per-class precision, recall and F1-score on the test set.

Table 13: Classification report of the tuned RBF-SVM on the test data

Class	Precision	Recall	F1-score	Support
NoGas	0.9705	0.9583	0.9644	240
Perfume	0.9588	0.9708	0.9648	240
Smoke	1.0000	1.0000	1.0000	240
Mixture	1.0000	1.0000	1.0000	240
Macro average	0.9823	0.9823	0.9823	960
Accuracy			0.9823	960

Perfume has the highest recall of the two difficult classes (0.9708), and NoGas the highest precision (0.9705). Since the classes are balanced, macro and weighted averages coincide. The macro F1-score of 0.9823 indicates excellent and balanced performance across all classes.

4.4.4 ROC Curves

Figure 19 shows the one-vs-rest ROC curves computed from the Platt-scaled class probabilities. All four curves hug the top-left corner.

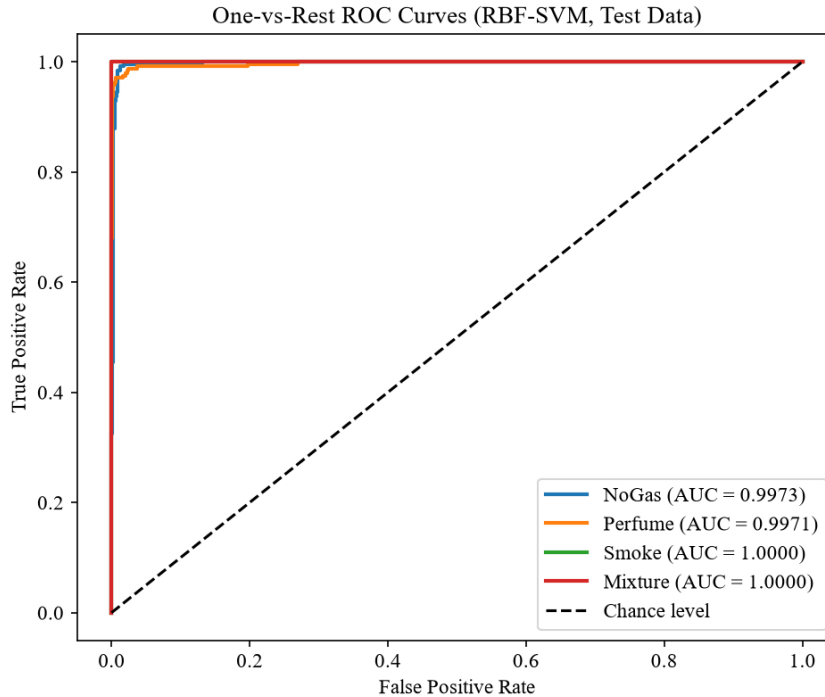


Figure 19: One-vs-rest ROC curves of the tuned RBF-SVM on the test data

The AUC is 1.0000 for Smoke and Mixture, 0.9973 for NoGas and 0.9971 for Perfume, giving a macro-average AUC of 0.9986. An AUC this close to 1 means that, for almost

every pair of a positive and a negative reading, the model assigns the higher probability to the positive one; the classifier therefore ranks gas conditions correctly at virtually any decision threshold, which leaves freedom to tune the alarm threshold for a desired false-alarm rate.

4.4.5 Learning Curve

Figure 20 shows how training and cross-validation accuracy change as the training set grows.

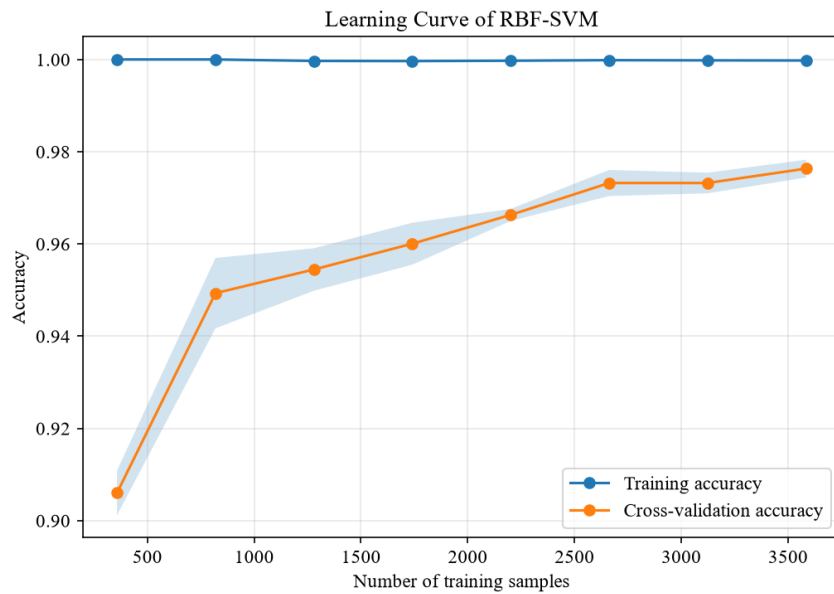


Figure 20: Learning curve of the tuned RBF-SVM

With only 358 training samples the cross-validation accuracy is 90.6%; it rises steadily to 97.6% with 3,584 samples, while the training accuracy stays near 100%. The narrowing gap and the still-rising validation curve indicate that the model is not over-fitting badly and that additional data would yield further, though diminishing, gains.

4.5 Comparison of Kernel Functions

To justify the choice of the RBF kernel, linear and polynomial kernels were tuned with the same cross-validation procedure (Table 14).

Table 14: Comparison of SVM kernels (all with standardised inputs)

Kernel	Best parameters	CV accuracy	Test accuracy
Linear	$C = 100$	85.31%	84.79%
Polynomial	$C = 100$, degree = 3	94.84%	95.52%
RBF (Gaussian)	$C = 100$, $\gamma = 10$	97.63%	98.23%

Even after standardisation and tuning, the linear kernel reached only 84.79%, and the cubic polynomial kernel 95.52%, whereas the RBF kernel reached 98.23%. The locality of the Gaussian kernel lets it carve the tight, curved boundaries needed between the NoGas and Perfume clusters. Comparing the tuned linear kernel (84.79%) with the raw-data baseline (80.31%) also isolates the contribution of standardisation and tuning alone.

4.6 Visualisation of the Decision Regions

To visualise how the RBF-SVM partitions the input space, a model with the same C was trained on the first two principal components of the standardised data and its decision regions were plotted (Figure 21). Because two dimensions keep only about 88% of the variance, this is an illustration rather than the actual seven-dimensional model.

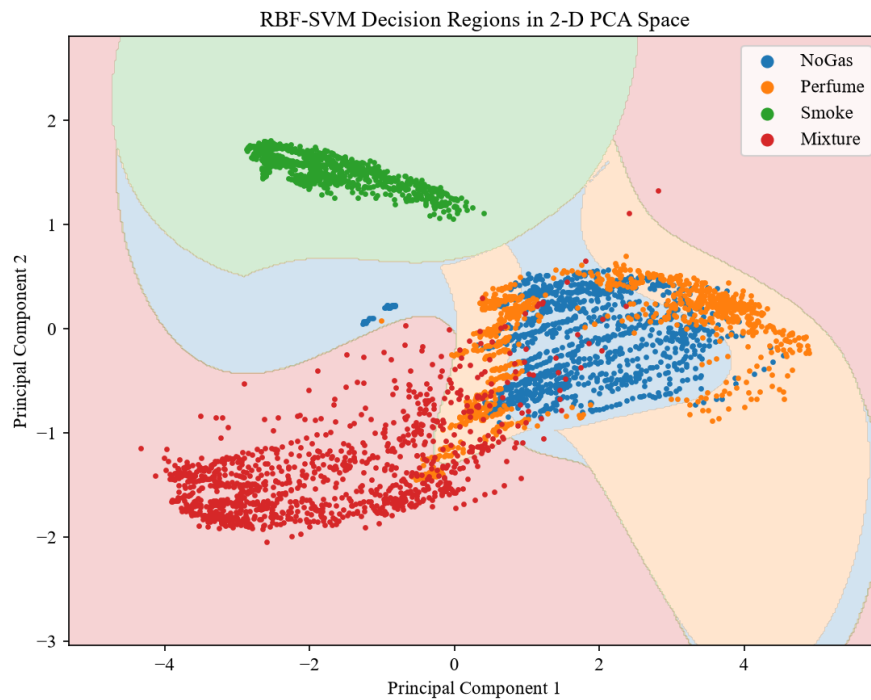


Figure 21: Decision regions of an RBF-SVM in the plane of the first two principal components

The figure shows the non-linear, curved boundaries characteristic of the RBF kernel. Smoke occupies a compact, isolated region and Mixture a separate elongated region, while the NoGas and Perfume regions interlock where the two classes overlap; in the full seven-dimensional space these classes are far better separated than this two-dimensional projection suggests.

4.7 Performance as a Leak Detector

In practice, the first question for a safety system is whether *any* gas is present. Collapsing the predictions into *no leak* (NoGas) and *gas present* (Perfume, Smoke or Mixture) gives the binary results in Table 15.

Table 15: Binary leak-detection performance of the tuned RBF-SVM on the test data

Measure	Value	Meaning
True positives (TP)	713	gas readings correctly flagged
True negatives (TN)	230	clean-air readings correctly passed
False positives (FP)	10	false alarms
False negatives (FN)	7	missed gas readings
Sensitivity (detection rate)	99.03%	$TP / (TP + FN)$
Specificity	95.83%	$TN / (TN + FP)$
False alarm rate	4.17%	$FP / (FP + TN)$
Missed-leak rate	0.97%	$FN / (FN + TP)$
Binary accuracy	98.23%	$(TP + TN) / N$

The model detected 99.03% of all gas-containing readings, and every missed reading was a Perfume reading, the least hazardous condition; no Smoke or Mixture reading was missed. The false alarm rate of 4.17% is low, and because a real leak produces many consecutive readings, an alarm rule such as *alarm when two consecutive readings indicate gas* would suppress most isolated false alarms while delaying a true alarm by only one sampling interval (2 s).

4.8 Real-Time Performance

4.8.1 Inference Latency

To emulate a live sensor stream, 500 test readings were classified one at a time, and the wall-clock time of each prediction (including standardisation) was recorded (Table 16, Figure 22).

Table 16: Inference latency of the tuned RBF-SVM (Intel Core i3-6006U)

Latency statistic	Value
Mean time per reading	1.099 ms
Median	0.989 ms
95th percentile	1.573 ms
Maximum	2.880 ms
Batch prediction (per reading)	0.209 ms
Sensor sampling interval	2,000 ms
Time budget used per reading	0.055%

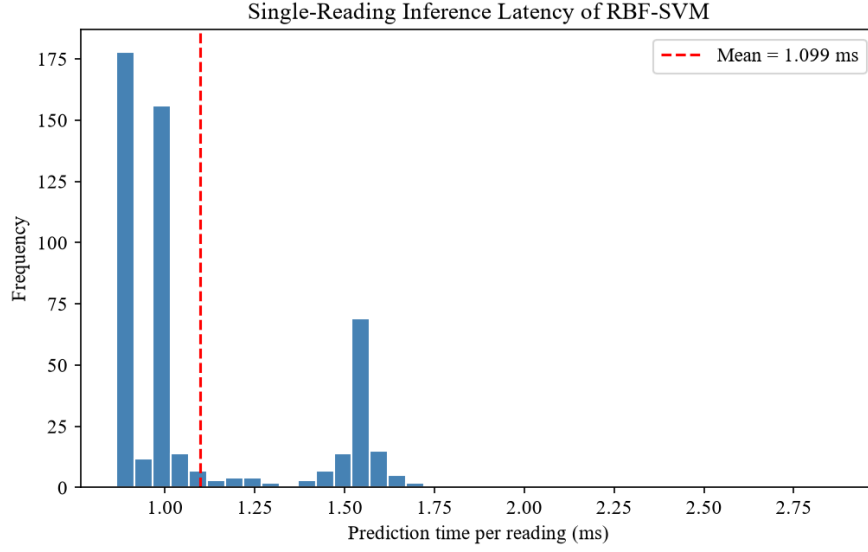


Figure 22: Distribution of single-reading prediction times

The average prediction took 1.10 ms, and even the slowest of the 500 predictions took only 2.88 ms. Since the sensors deliver a new reading every 2,000 ms, the classifier uses about 0.05% of the available time, leaving ample margin for communication and alarm logic. This is consistent with the complexity analysis of Section 3.5.11 ($n_{SV} = 1,304$, $d = 7$) and is more than a thousand times faster than the 1,414 ms reported for a thermal-image CNN on a microcontroller (El Barkani et al., 2024). Even allowing for a microcontroller that is one to two orders of magnitude slower than the laptop CPU, the model would comfortably meet the real-time requirement.

4.8.2 Simulated Real-Time Predictions with Confidence

Table 17 shows the predictions and class probabilities for eight readings taken at different times (two from each class), as they would appear on a monitoring dashboard.

Table 17: Predicted class probabilities for sample readings

Reading (true class)	Prediction	P(NoGas)	P(Perfume)	P(Smoke)	P(Mixture)	Confidence
0 (NoGas)	NoGas	0.9401	0.0458	0.0058	0.0084	Very high
200 (NoGas)	NoGas	0.9661	0.0265	0.0015	0.0059	Very high
0 (Perfume)	Perfume	0.0885	0.8908	0.0038	0.0170	High
200 (Perfume)	Perfume	0.1019	0.8977	0.0002	0.0001	High
0 (Smoke)	Smoke	0.0020	0.0032	0.9912	0.0036	Very high
200 (Smoke)	Smoke	0.0000	0.0000	0.9999	0.0001	Very high
0 (Mixture)	Mixture	0.0000	0.0000	0.0000	1.0000	Very high
200 (Mixture)	Mixture	0.0000	0.0000	0.0000	1.0000	Very high

All eight readings were classified correctly. Smoke and Mixture readings received probabilities close to 1, reflecting their distinct signatures, while NoGas and Perfume

readings, whose signatures are similar, were still assigned the correct class with clear majorities. Reporting these probabilities, rather than a bare label, allows an operator to treat low-confidence predictions with caution.

4.9 Robustness Under Temporal Drift

The results above use a random split, in which readings recorded seconds apart can fall into both the training and test sets. To test the model more strictly, a **chronological split** was also performed: within each class, the first 70% of readings in time order were used for training and the last 30% for testing, so the model had to predict readings recorded later than anything it had seen.

Table 18: Confusion matrix under the chronological split (accuracy 54.37%)

True \ Predicted	NoGas	Perfume	Smoke	Mixture
NoGas	289	48	0	143
Perfume	0	225	0	255
Smoke	0	0	50	430
Mixture	0	0	0	480

Under this protocol accuracy fell to 54.37% (macro F1 0.5187). The confusion matrix shows a systematic pattern: 430 of the 480 later Smoke readings, 255 Perfume readings and 143 NoGas readings were classified as Mixture. Figure 6 explains why: during each recording session the sensor baselines rose or fell steadily (for example, the MQ3 reading in the Smoke session climbed from about 350 to 430), so later readings moved into regions of the feature space that the model associated with another class. This is the well-known **drift** problem of metal-oxide sensors (Vergara et al., 2012; Rehman et al., 2021).

This finding does not contradict the main results, but it qualifies them. The random-split accuracy measures how well the model recognises gas signatures under the conditions it was trained on; the chronological result shows that a deployed system must periodically recalibrate or retrain the model, or use drift-compensation techniques (Dong et al., 2021; Krayden et al., 2025). Reporting this limitation honestly is itself a contribution, since previous studies on this dataset evaluated only random splits.

4.10 Comparison with Previous Studies

Table 19 compares the proposed model with published results on the same dataset that use the sensor readings.

Table 19: Comparison with previous results on the MultimodalGasData sensor readings

Study	Input	Model	Accuracy	Latency reported
Narkhede et al. (2021)	Sensors	LSTM	82%	No
Lalithadevi & Krishnaveni (2025)	Sensors	SVM (untuned)	88.36%	No
Attallah (2023)	Sensors (as images)	3 CNNs + Bi-LSTM	93.27%	No
Sharma et al. (2024)	Sensors	BiLSTM-Dense	93.39%	No
Arya et al. (2024)	Sensors + image features	SVM	94.22%	No
Lalithadevi & Krishnaveni (2025)	Sensors	Tuned random forest	98.67%	No
This study	Sensors	Tuned RBF-SVM	98.23%	Yes (1.10 ms)

The tuned RBF-SVM (98.23%) improves substantially on the only previously reported sensor-only SVM (88.36%), demonstrating that the earlier result reflected missing standardisation and tuning rather than a weakness of the SVM itself. It also clearly exceeds the original sensor-only LSTM (82%) and the deep sequence models (about 93%), and it is comparable to the best reported tuned random forest (98.67%), using a single, mathematically transparent model on an inexpensive laptop. Unlike all previous studies, it reports measured real-time latency. Because the studies used different splits (70/30 or 70/15/15) and the reported figures come from single runs, differences of about one percentage point should not be over-interpreted.

4.11 Discussion

Three main findings emerge. First, **preprocessing and tuning matter as much as the choice of model**. Standardising the inputs and tuning (C, γ) turned an 80% linear baseline into a 98% classifier; the same SVM family that appeared weak in earlier work proved highly competitive once its mathematics was respected. Second, **the remaining errors are physically meaningful**. All misclassifications lie between NoGas and Perfume. The deodorant is mostly alcohol, to which only MQ3 is specifically sensitive, and the MQ3 readings of the two classes are nearly identical (Table 7); the perfume plume is weak and dissipates quickly, so many Perfume readings are taken when little vapour reaches the sensors. Hazardous conditions (Smoke and Mixture) were never misclassified. Third, **real-time operation is easily achievable**. With about 1 ms per reading on an entry-level CPU

and a model defined by 1,304 seven-dimensional support vectors, the classifier is suitable for edge deployment in factories, warehouses and cold stores.

The chronological experiment adds an important caveat: the excellent random-split results hold for the recording conditions represented in the training data. In a real facility, sensor drift, humidity, temperature and new interfering gases would require periodic retraining or adaptive models. The SVM is well suited to this, since retraining on a few thousand readings takes seconds, and incremental and ensemble SVM methods for drift compensation already exist (Vergara et al., 2012; Dong et al., 2021).

CHAPTER 5: CONCLUSION

5.1 Summary of the Study

This study developed a real-time gas leak detection model for manufacturing and logistics environments based on a Support Vector Machine operating on the readings of seven low-cost MQ-series metal-oxide gas sensors. The work addressed a practical weakness of most existing low-cost detectors, which rely on single-sensor thresholds and cannot tell a hazardous release from a harmless volatile. The complete mathematical theory of the classifier was derived, from the geometric margin and the soft-margin primal problem through the Lagrangian dual and the Karush–Kuhn–Tucker conditions to the RBF kernel, one-vs-one voting and probability calibration, and every step of the methodology was implemented and evaluated on the public MultimodalGasData dataset.

A linear SVM on raw readings reached 80.31% test accuracy and confused the NoGas and Perfume classes heavily. After z-score standardisation and a cross-validated grid search, the RBF-SVM with $C = 100$ and $\gamma = 10$ achieved 98.23% test accuracy, a macro F1-score of 0.9823 and a macro ROC-AUC of 0.9986. Smoke and Mixture, the hazardous conditions, were recognised without a single error. As a leak detector the model reached 99.03% sensitivity and 95.83% specificity, and it classified a reading in about 1.1 ms on an entry-level laptop processor, about two thousand times faster than the 2 s sampling interval.

5.2 Contributions

- A complete, self-contained mathematical derivation of the multi-class kernel SVM, presented in the context of gas-sensor data and linked explicitly to the implementation.
- A tuned RBF-SVM that raises the best reported sensor-only SVM accuracy on MultimodalGasData from 88.36% to 98.23%, comparable to the best published sensor-only model, without deep learning or a thermal camera.
- The first measured real-time inference latency for a sensor-based model on this dataset, supported by a complexity analysis.
- An interpretation of the multi-class model as a binary leak detector with sensitivity, specificity and false-alarm rate, the measures that matter to safety engineers.

- An honest chronological robustness test that quantifies the effect of sensor drift, a limitation not examined by previous studies on this dataset.

5.3 Limitations

- The dataset was recorded in a single laboratory setting with two benign gas sources (deodorant and incense), not with industrial gases such as LPG, ammonia or carbon monoxide.
- The chronological test reduced accuracy to 54.37%, showing that sensor drift and changing baselines would degrade a fixed model over time.
- Readings are raw ADC counts without calibration to concentrations, so the model identifies gas types but cannot estimate ppm levels or compare them with exposure limits.
- Latency was measured on a laptop processor; it was argued, but not measured, that a microcontroller would also meet the real-time requirement.

5.4 Future Work

- Collect data with industrially relevant gases (LPG, methane, ammonia, CO) in real warehouses, cold stores and factories, over long periods and varying temperature and humidity.
- Add drift compensation, for example periodic recalibration, incremental or ensemble SVMs, or domain adaptation, and evaluate the model with time-based validation.
- Extend the classifier to support vector regression to estimate gas concentration and relate it to OSHA and NIOSH exposure limits.
- Deploy the trained model on an ESP32 or similar microcontroller and measure on-device latency and energy consumption in a live IoT alarm system.
- Use temporal features, such as the rate of change of each sensor, to capture leak dynamics and further separate weak sources from clean air.

5.5 Concluding Remarks

The results show that a carefully formulated and tuned Support Vector Machine, a model whose behaviour can be explained entirely through convex optimisation and kernel theory, can discriminate gases from an inexpensive sensor array with high accuracy and in real

time. Such a classifier offers a practical, affordable and transparent alternative to threshold alarms and heavy deep-learning systems, and can help make manufacturing and logistics facilities, in Bangladesh and elsewhere, safer from gas leaks.

REFERENCES

- Abadade, Y., Temouden, A., Bamoumen, H., Benamar, N., Chtouki, Y., & Hafid, A. S. (2023). A comprehensive survey on TinyML. *IEEE Access*, 11, 96892–96922. <https://doi.org/10.1109/ACCESS.2023.3294111>
- Adegboye, M. A., Fung, W.-K., & Karnik, A. (2019). Recent advances in pipeline monitoring and oil leakage detection technologies: Principles and approaches. *Sensors*, 19(11), 2548. <https://doi.org/10.3390/s19112548>
- Al Jazeera. (2013, August 31). *Deadly chemical leak in China's Shanghai*. <https://www.aljazeera.com/news/2013/8/31/deadly-chemical-leak-in-chinas-shanghai>
- Aljameel, S. S., Alomari, D. M., Alismail, S., Khawaher, F., Alkhudhair, A. A., Aljubran, F., & Alzannan, R. M. (2022). An anomaly detection model for oil and gas pipelines using machine learning. *Computation*, 10(8), 138. <https://doi.org/10.3390/computation10080138>
- Argirusis, N., Konstantaras, J., Argirusis, C., Dimokas, N., Thanopoulos, S., & Karvelis, P. S. (2024). A machine learning approach for the classification of refrigerant gases. *Applied Sciences*, 14(14), 6230. <https://doi.org/10.3390/app14146230>
- Arya, G., Bagwari, A., Agarwal, S., Aneja, J. K., & Rodriguez, C. (2024). A comparative analysis of early fusion architectures for multimodal gas detection using machine learning models. *Instrumentation Mesure Métrologie*, 23(4), 297–306. <https://doi.org/10.18280/i2m.230405>
- Attallah, O. (2023). Multitask deep learning-based pipeline for gas leakage detection via e-nose and thermal imaging multimodal fusion. *Chemosensors*, 11(7), 364. <https://doi.org/10.3390/chemosensors11070364>
- Bakar, M. A. A., Abdullah, A. H., Mustafa, W. A., Razali, Z. B., Saidi, S. A., Kader, M. M. M. A., & Aman, M. N. S. B. S. (2020). Electronic nose testing for confined space application utilizes principal component analysis and support vector machine. *IOP Conference Series: Materials Science and Engineering*, 932(1), 012072. <https://doi.org/10.1088/1757-899X/932/1/012072>
- Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (pp. 144–152). ACM. <https://doi.org/10.1145/130385.130401>
- Boubin, M., & Shrestha, S. (2019). Microcontroller implementation of support vector machine for detecting blood glucose levels using breath volatile organic compounds. *Sensors*, 19(10), 2283. <https://doi.org/10.3390/s19102283>
- Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press. <https://doi.org/10.1017/CBO9780511804441>
- Broughton, E. (2005). The Bhopal disaster and its aftermath: A review. *Environmental Health*, 4, 6. <https://doi.org/10.1186/1476-069X-4-6>

- Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), Article 27. <https://doi.org/10.1145/1961189.1961199>
- Chowdhury, M. A. Z., & Oehlschlaeger, M. A. (2025). Artificial intelligence in gas sensing: A review. *ACS Sensors*, 10(3), 1538–1563. <https://doi.org/10.1021/acssensors.4c02272>
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. <https://doi.org/10.1007/BF00994018>
- da Silva, A. H. M. C., da Silva, M. K., Santos, A., & Gómez Malagón, L. A. (2025). A systematic review for ammonia monitoring systems based on the Internet of Things. *IoT*, 6(4), 66. <https://doi.org/10.3390/iot6040066>
- Dey, A. (2018). Semiconductor metal oxide gas sensors: A review. *Materials Science and Engineering: B*, 229, 206–217. <https://doi.org/10.1016/j.mseb.2017.12.036>
- Dhaka Tribune. (2021, February 2). *LPG cylinder blasts ring alarm bells as consumption skyrockets*. <https://www.dhakatribune.com/bangladesh/nation/237473/lpg-cylinder-blasts-ring-alarm-bells-as>
- Dhaka Tribune. (2025, March 8). *Siddique Bazar explosion: Investigation still incomplete after two years*. <https://www.dhakatribune.com/bangladesh/375700/siddique-bazar-explosion-investigation-still>
- Dong, X., Han, S., Wang, A., & Shang, K. (2021). Online inertial machine learning for sensor array long-term drift compensation. *Chemosensors*, 9(12), 353. <https://doi.org/10.3390/chemosensors9120353>
- Down To Earth. (2020, July 7). *Visakhapatnam gas leak: LG Polymers neglected safety norms, finds govt probe*. <https://www.downtoearth.org.in/pollution/visakhapatnam-gas-leak-lg-polymers-neglected-safety-norms-finds-govt-probe-72169>
- El Barkani, M., Benamar, N., Talei, H., & Bagaa, M. (2024). Gas leakage detection using tiny machine learning. *Electronics*, 13(23), 4768. <https://doi.org/10.3390/electronics13234768>
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), 861–874. <https://doi.org/10.1016/j.patrec.2005.10.010>
- Feng, S., Farha, F., Li, Q., Wan, Y., Xu, Y., Zhang, T., & Ning, H. (2019). Review on smart gas sensing technology. *Sensors*, 19(17), 3760. <https://doi.org/10.3390/s19173760>
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hodgkinson, J., & Tatam, R. P. (2013). Optical gas sensing: A review. *Measurement Science and Technology*, 24(1), 012004. <https://doi.org/10.1088/0957-0233/24/1/012004>

- Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). *A practical guide to support vector classification* [Technical report]. Department of Computer Science, National Taiwan University. <https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>
- Hsu, C.-W., & Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2), 415–425. <https://doi.org/10.1109/72.991427>
- Huang, Y., Doh, I.-J., & Bae, E. (2021). Design and validation of a portable machine learning-based electronic nose. *Sensors*, 21(11), 3923. <https://doi.org/10.3390/s21113923>
- Huerta, R., Mosqueiro, T., Fonollosa, J., Rulkov, N. F., & Rodríguez-Luján, I. (2016). Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. *Chemometrics and Intelligent Laboratory Systems*, 157, 169–176. <https://doi.org/10.1016/j.chemolab.2016.07.004>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Hwang, G., Kim, J., Choi, J., Han, G., Kim, G., & Lim, S. (2026). Class-imbalanced gas thermal image detection in manufacturing using thermal region masking and density-based reweighting. *Applied Soft Computing*, 186, 114186. <https://doi.org/10.1016/j.asoc.2025.114186>
- International Labour Organization. (2019). *Safety and health at the heart of the future of work: Building on 100 years of experience*. ILO. <https://www.ilo.org/publications/safety-and-health-heart-future-work-building-100-years-experience-0>
- International Labour Organization. (2021). *Exposure to hazardous chemicals at work and resulting health impacts: A global review*. ILO. <https://www.ilo.org/publications/exposure-hazardous-chemicals-work-and-resulting-health-impacts-global>
- Jadhav, P., Sairam, V. A., Bhojane, N., Singh, A., Gite, S., Pradhan, B., Bachute, M., & Alamri, A. M. (2025). Multimodal gas detection using e-nose and thermal images: An approach utilizing SRGAN and sparse autoencoder. *Computers, Materials & Continua*, 83(2), 3493–3517. <https://doi.org/10.32604/cmc.2025.060764>
- Jiang, Z., Xu, P., Du, Y., Yuan, F., & Song, K. (2021). Balanced distribution adaptation for metal oxide semiconductor gas sensor array drift compensation. *Sensors*, 21(10), 3403. <https://doi.org/10.3390/s21103403>
- Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A*, 374(2065), 20150202. <https://doi.org/10.1098/rsta.2015.0202>
- Karakaya, D., Ulucan, O., & Turkan, M. (2020). Electronic nose and its applications: A survey. *International Journal of Automation and Computing*, 17(2), 179–209. <https://doi.org/10.1007/s11633-019-1212-9>
- Kidam, K., & Hurme, M. (2013). Analysis of equipment failures as contributors to chemical process accidents. *Process Safety and Environmental Protection*, 91(1–2), 61–78. <https://doi.org/10.1016/j.psep.2012.02.001>

- Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (Vol. 2, pp. 1137–1145). Morgan Kaufmann.
- Krayden, A., Avraham, M., Ashkar, H., Blank, T., Stolyarova, S., & Nemirovsky, Y. (2025). TinyML-based real-time drift compensation for gas sensors using spectral–temporal neural networks. *Chemosensors*, 13(7), 223.
<https://doi.org/10.3390/chemosensors13070223>
- Kreßel, U. H.-G. (1999). Pairwise classification and support vector machines. In B. Schölkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in kernel methods: Support vector learning*. MIT Press. <https://doi.org/10.7551/mitpress/1130.003.0020>
- Lalithadevi, B., & Krishnaveni, S. (2025). ExAIRFC-GSDC: An advanced machine learning-based interpretable framework for accurate gas leakage detection and classification. *International Journal of Computational Intelligence Systems*, 18, 16.
<https://doi.org/10.1007/s44196-025-00742-6>
- Laref, R., Losson, E., Sava, A., & Siadat, M. (2019). On the optimization of the support vector machine regression hyperparameters setting for gas sensors array applications. *Chemometrics and Intelligent Laboratory Systems*, 184, 22–27.
<https://doi.org/10.1016/j.chemolab.2018.11.011>
- Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business & Information Systems Engineering*, 6(4), 239–242. <https://doi.org/10.1007/s12599-014-0334-4>
- Lin, H.-T., Lin, C.-J., & Weng, R. C. (2007). A note on Platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3), 267–276. <https://doi.org/10.1007/s10994-007-5018-6>
- Liu, H., & Tang, Z. (2013). Metal oxide gas sensor drift compensation using a dynamic classifier ensemble based on fitting. *Sensors*, 13(7), 9160–9173.
<https://doi.org/10.3390/s130709160>
- Liu, X., Cheng, S., Liu, H., Hu, S., Zhang, D., & Ning, H. (2012). A survey on gas sensing technology. *Sensors*, 12(7), 9635–9665. <https://doi.org/10.3390/s120709635>
- McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61).
<https://doi.org/10.25080/Majora-92bf1922-00a>
- Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A*, 209, 415–446. <https://doi.org/10.1098/rsta.1909.0016>
- MQ-2 semiconductor sensor for combustible gas [Datasheet]. (n.d.).
<https://www.pololu.com/file/0J309/MQ2.pdf>
- Murvy, P.-S., & Silea, I. (2012). A survey on gas leak detection and localization techniques. *Journal of Loss Prevention in the Process Industries*, 25(6), 966–973.
<https://doi.org/10.1016/j.jlp.2012.05.010>

- Narkhede, P., Walambe, R., Chandel, P., Mandaokar, S., & Kotecha, K. (2022a). MultimodalGasData: Multimodal dataset for gas detection and classification. *Data*, 7(8), 112. <https://doi.org/10.3390/data7080112>
- Narkhede, P., Walambe, R., Chandel, P., Mandaokar, S., & Kotecha, K. (2022b). *MultimodalGasData: Multimodal dataset for gas detection and classification* (Version 2) [Data set]. Mendeley Data. <https://doi.org/10.17632/zkwgkjkjn9.2>
- Narkhede, P., Walambe, R., Mandaokar, S., Chandel, P., Kotecha, K., & Ghinea, G. (2021). Gas detection and identification using multimodal artificial intelligence based sensor fusion. *Applied System Innovation*, 4(1), 3. <https://doi.org/10.3390/asi4010003>
- National Institute for Occupational Safety and Health. (n.d.-a). *NIOSH pocket guide to chemical hazards: Ammonia*. <https://www.cdc.gov/niosh/npg/npgd0028.html>
- National Institute for Occupational Safety and Health. (n.d.-b). *NIOSH pocket guide to chemical hazards: Carbon monoxide*. <https://www.cdc.gov/niosh/npg/npgd0105.html>
- National Institute for Occupational Safety and Health. (n.d.-c). *NIOSH pocket guide to chemical hazards: n-Butane*. <https://www.cdc.gov/niosh/npg/npgd0068.html>
- National Institute for Occupational Safety and Health. (n.d.-d). *NIOSH pocket guide to chemical hazards: Hydrogen sulfide*. <https://www.cdc.gov/niosh/npg/npgd0337.html>
- National Institute for Occupational Safety and Health. (n.d.-e). *NIOSH pocket guide to chemical hazards: Propane*. <https://www.cdc.gov/niosh/npg/npgd0524.html>
- National Institute for Occupational Safety and Health. (n.d.-f). *NIOSH pocket guide to chemical hazards: Benzene*. <https://www.cdc.gov/niosh/npg/npgd0049.html>
- National Oceanic and Atmospheric Administration. (n.d.-a). *CAMEO Chemicals: Hydrogen*. <https://cameochemicals.noaa.gov/chemical/8729>
- National Oceanic and Atmospheric Administration. (n.d.-b). *CAMEO Chemicals: Methane, refrigerated liquid*. <https://cameochemicals.noaa.gov/report?key=CH3872>
- Nipo, A., Islam, R. A., Afrin, N. N., Khandakar, S. N., & Islam, M. E. (2025). IoT-enabled smart warehouse monitoring and safety system for real time hazard detection and automated mitigation. In *2025 IEEE 7th International Conference on Sustainable Technologies for Industry 5.0 (STI)* (pp. 1–6). IEEE. <https://doi.org/10.1109/STI69347.2025.11367508>
- Occupational Safety and Health Administration. (n.d.-a). *Liquefied petroleum gas (L.P.G.)* [Chemical data]. <https://www.osha.gov/chemicaldata/484>
- Occupational Safety and Health Administration. (n.d.-b). *Ammonia refrigeration*. <https://www.osha.gov/ammonia-refrigeration>
- Pan, X., Zhang, H., Ye, W., Bermak, A., & Zhao, X. (2019). A fast and robust gas recognition algorithm based on hybrid convolutional and recurrent neural network. *IEEE Access*, 7, 100954–100963. <https://doi.org/10.1109/ACCESS.2019.2930804>
- 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, É. (2011). Scikit-learn: Machine learning in

- Python. *Journal of Machine Learning Research*, 12, 2825–2830.
<https://www.jmlr.org/papers/v12/pedregosa11a.html>
- Peng, P., Zhao, X., Pan, X., & Ye, W. (2018). Gas classification using deep convolutional neural networks. *Sensors*, 18(1), 157. <https://doi.org/10.3390/s18010157>
- Platt, J. C. (1998). *Sequential minimal optimization: A fast algorithm for training support vector machines* (Technical Report MSR-TR-98-14). Microsoft Research.
<https://www.microsoft.com/en-us/research/publication/sequential-minimal-optimization-a-fast-algorithm-for-training-support-vector-machines/>
- Platt, J. C. (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. J. Smola, P. Bartlett, B. Schölkopf, & D. Schuurmans (Eds.), *Advances in large margin classifiers*. MIT Press.
<https://doi.org/10.7551/mitpress/1113.001.0001>
- Praveen Sharma, V., Dugyala, R., Padmavathi, V., & Gurram, V. R. (2023). IoT based gas leakage detection system using GPS. *E3S Web of Conferences*, 391, 01064.
<https://doi.org/10.1051/e3sconf/202339101064>
- Rahman, M., Charoenlarnopparut, C., Suksompong, P., Toochinda, P., & Taparugssanagorn, A. (2017). A false alarm reduction method for a gas sensor based electronic nose. *Sensors*, 17(9), 2089. <https://doi.org/10.3390/s17092089>
- Rehman, A. U., Belhaouari, S. B., Ijaz, M., Bermak, A., & Hamdi, M. (2021). Multi-classifier tree with transient features for drift compensation in electronic nose. *IEEE Sensors Journal*, 21(5), 6564–6574. <https://doi.org/10.1109/JSEN.2020.3041949>
- Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press.
<https://doi.org/10.7551/mitpress/4175.001.0001>
- Sharma, A., Khullar, V., Kansal, I., Chhabra, G., Arora, P., Popli, R., & Kumar, R. (2024). Gas detection and classification using multimodal data based on federated learning. *Sensors*, 24(18), 5904. <https://doi.org/10.3390/s24185904>
- Shin, K. G., & Ramanathan, P. (1994). Real-time computing: A new discipline of computer science and engineering. *Proceedings of the IEEE*, 82(1), 6–24.
<https://doi.org/10.1109/5.259423>
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437.
<https://doi.org/10.1016/j.ipm.2009.03.002>
- Song, K., Wang, Q., Liu, Q., Zhang, H., & Cheng, Y. (2011). A wireless electronic nose system using a Fe₂O₃ gas sensing array and least squares support vector regression. *Sensors*, 11(1), 485–505. <https://doi.org/10.3390/s110100485>
- Stetter, J. R., & Li, J. (2008). Amperometric gas sensors: A review. *Chemical Reviews*, 108(2), 352–366. <https://doi.org/10.1021/cr0681039>
- The Business Standard. (2023, March 15). *Unskilled operators to blame for Sitakunda oxygen plant blast: Police*. <https://www.tbsnews.net/node/599954>

- The Daily Star. (2016, August 23). *Gas disaster averted in Ctg.*
<https://www.thedailystar.net/frontpage/gas-disaster-averted-ctg-1274602>
- The Daily Star. (2020, September 22). *Another Narayanganj blast victim dies, death toll now 34.* <https://www.thedailystar.net/country/news/another-narayanganj-blast-victim-dies-death-toll-now-34-1965525>
- Tsoukas, V., Gkogkidis, A., Boumpa, E., Papafotikas, S., & Kakarountas, A. (2023). A gas leakage detection device based on the technology of TinyML. *Technologies*, 11(2), 45.
<https://doi.org/10.3390/technologies11020045>
- United News of Bangladesh. (2022, July 7). *Probe body finds owners, monitoring authority responsible for Sitakunda depot fire.* <https://unb.com.bd/category/Bangladesh/probe-body-finds-owners-monitoring-authority-responsible-for-sitakunda-depot-fire/96437>
- U.S. Chemical Safety and Hazard Investigation Board. (2015). *Millard Refrigerated Services ammonia release.* <https://www.csb.gov/millard-refrigerated-services-ammonia-release/>
- Vapnik, V. N. (1995). *The nature of statistical learning theory.* Springer.
<https://doi.org/10.1007/978-1-4757-2440-0>
- Vergara, A., Vembu, S., Ayhan, T., Ryan, M. A., Homer, M. L., & Huerta, R. (2012). Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, 166–167, 320–329. <https://doi.org/10.1016/j.snb.2012.01.074>
- Viciano-Tudela, S., Parra, L., Navarro-Garcia, P., Sendra, S., & Lloret, J. (2023). Proposal of a new system for essential oil classification based on low-cost gas sensor and machine learning techniques. *Sensors*, 23(13), 5812. <https://doi.org/10.3390/s23135812>
- Wang, C., Yin, L., Zhang, L., Xiang, D., & Gao, R. (2010). Metal oxide gas sensors: Sensitivity and influencing factors. *Sensors*, 10(3), 2088–2106.
<https://doi.org/10.3390/s100302088>
- Waskom, M. L. (2021). seaborn: Statistical data visualization. *Journal of Open Source Software*, 6(60), 3021. <https://doi.org/10.21105/joss.03021>
- Wu, T.-F., Lin, C.-J., & Weng, R. C. (2004). Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5, 975–1005.
<https://www.jmlr.org/papers/v5/wu04a.html>
- Xi, J., Guan, L., Zhu, X., Zong, K., & Yan, W. (2026). Pattern recognition of hazardous gas leak monitoring data based on field sensors. *Processes*, 14(1), 108.
<https://doi.org/10.3390/pr14010108>
- Ye, Z., Liu, Y., & Li, Q. (2021). Recent progress in smart electronic nose technologies enabled with machine learning methods. *Sensors*, 21(22), 7620.
<https://doi.org/10.3390/s21227620>
- Zeng, X., Shahzeb, M., Cheng, X., Shen, Q., Xiao, H., Xia, C., Xia, Y., Huang, Y., Xu, J., & Wang, Z. (2024). An enhanced gas sensor data classification method using principal component analysis and synthetic minority over-sampling technique algorithms. *Micromachines*, 15(12), 1501. <https://doi.org/10.3390/mi15121501>

- Zhang, H., & Han, Y. (2022). A new mixed-gas-detection method based on a support vector machine optimized by a sparrow search algorithm. *Sensors*, 22(22), 8977. <https://doi.org/10.3390/s22228977>
- Zhang, L., & Zhang, D. (2015). Domain adaptation extreme learning machines for drift compensation in E-nose systems. *IEEE Transactions on Instrumentation and Measurement*, 64(7), 1790–1801. <https://doi.org/10.1109/TIM.2014.2367775>
- Zhao, X., Wen, Z., Pan, X., Ye, W., & Bermak, A. (2019). Mixture gases classification based on multi-label one-dimensional deep convolutional neural network. *IEEE Access*, 7, 12630–12637. <https://doi.org/10.1109/ACCESS.2019.2892754>
- Zhou, K., & Liu, Y. (2021). Early-stage gas identification using convolutional long short-term neural network with sensor array time series data. *Sensors*, 21(14), 4826. <https://doi.org/10.3390/s21144826>
- Ziyatdinov, A., Marco, S., Chaudry, A., Persaud, K., Caminal, P., & Perera, A. (2010). Drift compensation of gas sensor array data by common principal component analysis. *Sensors and Actuators B: Chemical*, 146(2), 460–465. <https://doi.org/10.1016/j.snb.2009.11.034>