iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: https://doi.org/10.22214/ijraset.2026.85509

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

AI-Based Drainage Safety Assistant Using Random Forest Risk Classification and a Flask Dashboard

K. Kavitha[1], Janani N[2], Kaliyamma G[3], Kavya C[4]
Department of Computer Science and Engineering, Adhiyamaan College of Engineering, Hosur, India
Assistant Professor and Project Supervisor, Adhiyamaan College of Engineering, Hosur, India

Abstract: ***Workers who inspect and clean drains and sewers face a combination of oxygen deficiency, toxic and flammable gases, heat and flooding, yet routine practice still depends on visual checks and instruments that report one quantity at a time. This paper describes a software prototype, the AI-Based Drainage Safety Assistant, that merges six environmental readings (O_2, H_2S, CH_4, temperature, humidity and water level) into one risk verdict of Safe, Warning or Danger. Readings are validated and placed in a fixed feature order, then classified by a Random Forest model built with Scikit-learn. A Flask dashboard shows each reading, the colour-coded verdict and a matching recommendation, and a buzzer or on-screen alarm is triggered for Danger. The model was trained and tested on a simulated dataset of 198 labelled records. On a stratified 80/20 hold-out split it reached 87.5% accuracy, and four-fold cross-validation gave 83.8% ± 1.5%, against 77.3% for a majority-class guess. Danger was recognised well (98.0% recall in pooled cross-validation), Warning poorly, and Safe could not be learned because it has only four examples; errors leaned towards over-warning. Seven other classifiers scored 80.8% to 86.4%, so no superiority of the Random Forest is claimed. Because the data are simulated, the findings demonstrate that the pipeline works and do not certify field safety; limits and a route towards calibrated sensors, real data and a rule-based safeguard are discussed.***

Keywords: drainage safety, confined space, hazardous gas monitoring, Random Forest, risk classification, Flask, occupational safety.

I. INTRODUCTION

Keeping drains and sewers working is a basic part of urban sanitation, but the people who do this work enter atmospheres that can change within minutes. Hydrogen sulphide (H_2S) is toxic at low concentrations, methane (CH_4) can displace air and ignite, biological activity can consume oxygen, and heavy rain can raise water levels rapidly [1]–[6]. Occupational-safety guidance therefore asks for atmospheric testing before and during entry into a confined space [2], [3].

In many municipal settings this guidance is applied with portable single-gas detectors and the worker's own judgement. Each device shows a number; deciding what several numbers mean together is left to the person at the manhole. A compact software layer that reads all available measurements and returns a single, plainly worded status could reduce that burden and make early warnings more visible. This work builds such a layer and reports on its design and first evaluation. The contributions are: (1) a six-feature representation of drainage conditions; (2) a Random Forest classifier [7] that assigns Safe, Warning or Danger; (3) a Flask [11] dashboard and alert path that turn the class into an action message; and (4) an honest evaluation on a simulated dataset, including the weaknesses it exposes. The assistant is meant to complement calibrated detectors, protective equipment, trained supervision and permit procedures, not to replace them. The remainder of the paper is organised as follows. Section II surveys related work and identifies the gap addressed here. Section III describes the system design, and Section IV the implementation. Section V reports the experimental evaluation, including comparisons with other classifiers, a feature ablation and an error analysis. Section VI discusses limitations and deployment issues, and Section VII concludes with directions for future work.

II. LITERATURE SURVEY

A. Hazard and Regulatory Background

Public-health and occupational agencies describe the dangers of sewer atmospheres in detail. WHO and NIOSH summarise the symptoms, exposure limits and precautions for H_2S [1], [4]; NIOSH likewise records the flammability of methane [5]. OSHA's permit-required confined-space rules and NIOSH's fatality-prevention and criteria documents explain why oxygen deficiency, toxic gas and engulfment are leading causes of confined-space deaths and stress testing, ventilation and supervised entry [2], [3], [6]. These documents define what must be controlled, but they are procedural; they do not supply an automated tool that fuses several readings.

©IJRASET: All Rights are Reserved

B. Learning and Software Foundations

Random Forests combine many decorrelated decision trees by majority vote and handle mixed-scale numeric inputs without feature scaling [7]. Support Vector Machines offer a margin-based alternative [8]. Standard references on data mining and applied machine learning describe cleaning, model selection and evaluation [9], [10], while Scikit-learn, Pandas and Flask provide practical building blocks for training, tabulating and serving a model [11]–[17]. All of these are generic and carry no drainage-specific thresholds, labels or alert logic.

C. Gap

The surveyed literature treats gas hazards, entry procedures, algorithms and web tooling in isolation (Table I). What is missing is a small, reproducible application that joins them for drainage work: multi-parameter input, learned classification, a readable dashboard and an alarm. The present prototype targets that gap.

TABLE I
SURVEYED SOURCES AND WHAT THEY LACK

Ref.	Contribution	Missing for drainage use
[1],[4]	H_2S effects and limits	No fusion with other readings
[5]	Methane hazard data	Single-gas view only
[2],[3],[6]	Confined-space rules and criteria	Procedural; no automated classifier
[7],[8]	Random Forest; SVM	General algorithms, no domain labels
[9]–[10]	Data mining; applied ML	No sensor-safety application
[11]–[17]	Flask, Python, Scikit-learn, Pandas	Tools only; no risk logic

D. Positioning of This Work

The present work does not propose a new learning algorithm. Its contribution is the integration of established components into one usable assistant, together with an honest measurement of how far a model trained on simulated labels can be trusted. Compared with procedural guidance [2], [3], the assistant produces an immediate status for every set of readings. Compared with single-gas displays [1], [4], [5], it considers oxygen, two gases, thermal conditions and water level jointly. Compared with general toolkits [11]–[17], it fixes the feature order, the class set and the response messages for one concrete application.

Two design choices follow from the survey. First, the output is deliberately coarse, three classes instead of a numeric score, because a worker at a manhole needs a decision aid and not another number to interpret. Second, the response messages are conservative and always defer to established entry procedures, in line with the confined-space literature [2], [3], [6].

III. SYSTEM DESIGN

A. Overview

The prototype is organised as five loosely coupled modules (Fig. 1): input, preprocessing, classification, presentation and alerting. Keeping them separate lets each be tested, replaced or extended, for example swapping simulated input for a serial or MQTT sensor feed without touching the model.

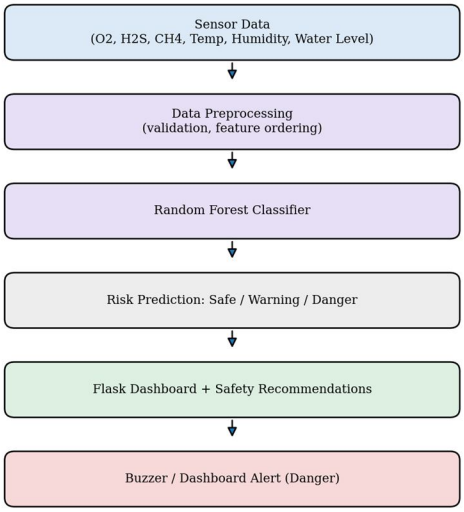


Fig. 1 Processing pipeline of the proposed assistant

B. Input Parameters

Six variables describe the drain atmosphere and flow: oxygen (% by volume), H_2S (ppm), CH_4 (ppm), temperature (°C), relative humidity (%) and water level (cm). In the prototype these are generated or typed in; in a deployed unit they would come from electrochemical O_2 and H_2S cells, a catalytic or infrared methane sensor, a temperature-humidity sensor and an ultrasonic level sensor.

C. Preprocessing

Incoming records are checked for missing fields and non-numeric values, cast to floating point and placed in the fixed order [O_2, H_2S, CH_4, temperature, humidity, water level]. Using exactly the training order at inference is essential, because the forest's splits are tied to column positions and names.

D. Classification

A Random Forest of 100 trees [13], [17] is fitted to labelled records. Each tree is grown on a bootstrap sample with a random subset of features considered at every split, and the predicted class is the majority vote. The trained model and the label encoder are serialised with Joblib so that the web service loads them once at start-up. Algorithm 1 summarises training and use.

Formally, let x be the preprocessed six-element feature vector and C = {Safe, Warning, Danger} the class set. Each tree h_t is grown on a bootstrap sample of the training records. At every node a random subset of the features is examined and the split that most reduces the Gini impurity is chosen, where the impurity of node n is

$$G(n) = 1 - \sum_{c \in C} p_c(n)^2 \qquad (1)$$

and $p_c(n)$ is the fraction of training records of class c reaching node n. With T = 100 trees, the forest assigns the class that receives the most votes:

$$\hat{y}(x) = \operatorname{argmax}_{c \in C} \sum_{t=1}^{T} I[h_t(x) = c] \qquad (2)$$

where $I[\cdot]$ equals 1 when its argument is true and 0 otherwise. The mean decrease in G over all splits that use a feature gives the feature importance reported in Section V.

Algorithm 1. Training and prediction

1. Load labelled CSV; encode labels.
2. Split data (stratified 80/20).
3. Fit Random Forest (100 trees).
4. Save model and encoder.
5. For each new reading: validate, order features, predict, decode label.
6. Map label to recommendation; if Danger, raise alarm.

E. Presentation and Alerting

Each predicted class is tied to a fixed message (Table II). The dashboard colours the verdict green, amber or red; for Danger a buzzer sounds when the hardware is fitted, otherwise a banner is shown. A Safe verdict only means that the model found no elevated risk in the supplied readings.

TABLE II
CLASSES AND RESPONSES

Class	Colour	Recommended response
Safe	Green	Continue routine monitoring
Warning	Amber	Proceed with caution; arrange further inspection
Danger	Red	Do not enter; alert safety personnel; sound alarm

F. Proposed Hardware Realisation

The prototype takes its readings from a file, but the same pipeline can be fed by a small sensing node suspended in the drain or placed at the manhole opening. Table III lists a plausible sensing element for each parameter. A low-power microcontroller would sample the sensors at a fixed interval, reject values outside each sensor's valid range, and pass the six readings to the Flask service over a wired serial link or a wireless protocol. The buzzer would be driven from a digital output and would sound whenever the predicted class is Danger.

TABLE III
INDICATIVE SENSING ELEMENTS FOR THE SIX PARAMETERS

Parameter	Typical element	Practical note
Oxygen	Electrochemical O_2 cell	Needs periodic calibration
H_2S	Electrochemical H_2S cell	Affected by temperature and humidity
Methane	Catalytic bead or infrared sensor	Infrared avoids poisoning
Temperature, humidity	Digital temperature-humidity sensor	Low cost, easy to interface
Water level	Ultrasonic or float sensor	Mount clear of splash

These elements are indicative and not a validated bill of materials. Because electrochemical and catalytic cells drift and show cross-sensitivity, routine bump tests against a reference gas would be part of any real deployment, and the assistant must never replace the calibrated personal detectors required by entry procedures.

IV. IMPLEMENTATION

The prototype is written in Python and developed in Visual Studio Code. Table IV lists the components.

TABLE IV
SOFTWARE COMPONENTS

Layer	Technology
Interface	HTML, CSS, JavaScript
Server	Python, Flask
Learning	Scikit-learn Random Forest
Data handling	Pandas, NumPy
Persistence	Joblib model files; JSON current reading

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

On the root route the Flask service reads the latest values from a JSON file, builds a one-row DataFrame, calls the model, decodes the label and renders the dashboard template with the readings and verdict. A login page protects access to the monitoring view. The dashboard displays one card per parameter with its unit and a prominent risk panel (Fig. 2).

V. EXPERIMENTAL EVALUATION

A. *Dataset*

No field measurements were available, so a simulated dataset of 198 records was used. Each record has the six features and one label. Value ranges are O_2 15.0–21.0 %, H_2S 0.1–49.4 ppm, CH_4 1.3–199.3 ppm, temperature 20.2–44.8 °C, humidity 40.1–99.7 % and water level 0.1–99.9 cm. The class distribution is strongly skewed (Table V).

TABLE V
CLASS DISTRIBUTION

Class	Records	Share
Danger	153	77.3 %
Warning	41	20.7 %
Safe	4	2.0 %
Total	198	100 %

The mean feature values by class (Table VI) show the structure available to the model. Danger records have the lowest mean oxygen (17.9 %) and the highest mean H_2S, methane and water level, whereas the four Safe records have the highest oxygen (19.9 %) and the lowest water level (23.1 cm). Oxygen alone is strongly informative: 139 of the 198 records have oxygen below 19.5 %, and none of these is labelled Safe (122 are Danger and 17 Warning), whereas among the 59 records at or above that level 31 are Danger, 24 Warning and 4 Safe.

TABLE VI
MEAN FEATURE VALUES BY CLASS

Class	O_2 (%)	H_2S (ppm)	CH_4 (ppm)	Temp. (°C)	Hum. (%)	Water (cm)
Danger	17.87	25.08	105.62	33.26	71.56	54.46
Warning	19.33	18.51	78.51	32.37	74.97	29.49
Safe	19.92	6.82	58.18	29.36	65.62	23.08

B. *Protocol*

The data were split 80/20 with stratification (random seed 42), giving 158 training and 40 test records. A Random Forest with 100 trees and otherwise default Scikit-learn settings was trained and scored on the held-out set. To gauge stability on such a small sample, four-fold stratified cross-validation was also run. A majority-class predictor (always Danger) serves as the baseline.

For each class c the precision, recall and F1 score are computed from the true positives (TP), false positives (FP) and false negatives (FN):

$$P_c = TP_c / (TP_c + FP_c), \quad R_c = TP_c / (TP_c + FN_c), \quad F1_c = 2P_cR_c / (P_c + R_c) \quad (3)$$

Macro averages weight the three classes equally, so the rare Safe class affects them as much as Danger; weighted averages follow class frequency. Both are reported because accuracy alone hides the behaviour on rare classes.

C. *Results*

The hold-out accuracy was 87.5 % (35 of 40 correct). Cross-validation folds scored 84.0, 84.0, 81.6 and 85.7 %, a mean of 83.8 % ± 1.5, both above the 77.3 % baseline. Per-class results are in Table VII and the confusion matrix in Table VIII.

TABLE VII
HOLD-OUT PER-CLASS METRICS

Class	Precision	Recall	F1	Support
Danger	0.91	0.97	0.94	31
Warning	0.71	0.62	0.67	8
Safe	0.00	0.00	0.00	1
Macro avg	0.54	0.53	0.53	40
Weighted avg	0.85	0.88	0.86	40

TABLE VIII
CONFUSION MATRIX (ROWS: TRUE, COLUMNS: PREDICTED)

	Safe	**Warning**	**Danger**
Safe	**0**	**1**	**0**
Warning	**0**	**5**	**3**
Danger	**0**	**1**	**30**

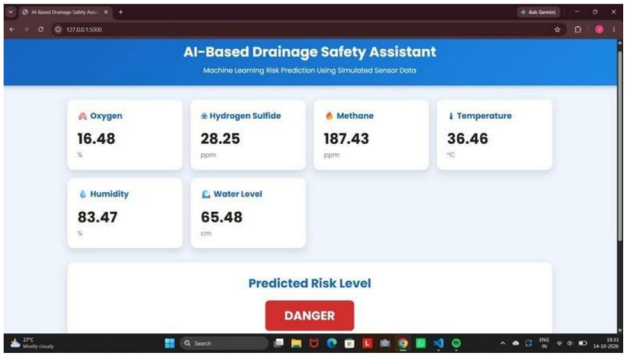


Fig. 2 Dashboard showing six readings and the predicted level

For the reading in Fig. 2 (O_2 16.48 %, H_2S 28.25 ppm, CH_4 187.43 ppm, 36.46 °C, 83.47 % humidity, 65.48 cm) the model returns Danger. This agrees with the low oxygen figure, which lies below the 19.5 % level commonly used as the lower limit for an acceptable atmosphere in confined-space practice [2].

The forest's feature importances (Fig. 3) show oxygen as the strongest cue, followed by water level, H_2S and CH_4, while temperature and humidity contribute least.

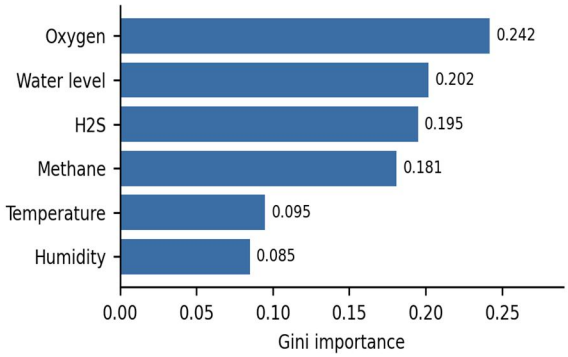


Fig. 3 Feature importance of the trained Random Forest

D. Comparison with Other Classifiers

To place the Random Forest in context, seven further classifiers were evaluated with the same four folds (Table IX, Fig. 4). Distance- and margin-based models were preceded by feature scaling. The support-vector machine and logistic regression reached the highest mean accuracy (86.4 %), followed by gradient boosting (85.9 %), while the Random Forest scored 83.8 % and Gaussian naive Bayes 80.8 %. The fold-to-fold standard deviations (1.5 to 4.5 points) are as large as the gaps between the leading models, so the data do not support a firm ranking. With only 198 records and four Safe examples, no model exceeded a macro-F1 of 0.58.

TABLE IX
FOUR-FOLD CROSS-VALIDATION OF CANDIDATE CLASSIFIERS

Model	Accuracy (%)	Macro-F1	Weighted-F1
Random Forest (100 trees)	83.8 ± 1.5	0.47	0.81
Random Forest (balanced)	83.8 ± 2.4	0.47	0.81
Decision tree	81.3 ± 1.8	0.58	0.82
k-NN (k = 5)	82.9 ± 4.5	0.52	0.79
SVM (RBF kernel)	86.4 ± 2.9	0.51	0.84
Logistic regression	86.4 ± 3.8	0.53	0.85
Gaussian naive Bayes	80.8 ± 2.1	0.43	0.78
Gradient boosting	85.9 ± 2.4	0.51	0.85

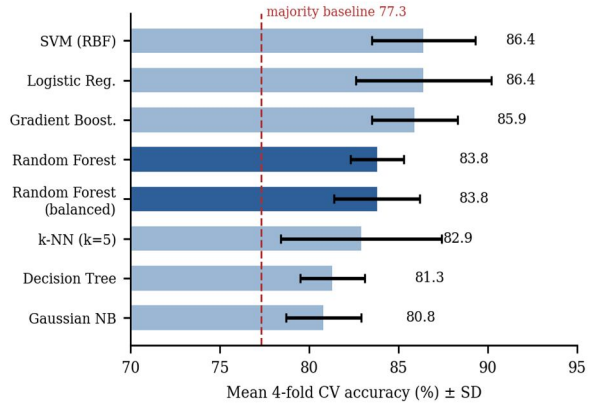


Fig. 4 Mean cross-validated accuracy of the candidate models

The Random Forest was kept for the prototype for practical reasons rather than because it won this comparison. It accepts unscaled inputs with very different ranges, models non-linear interactions among gases and water level, and yields feature importances that can be shown to a supervisor. Class weighting did not change its accuracy. A linear model is a reasonable alternative and the choice should be revisited once real measurements are available. Note that the macro-F1 values here are means over the four folds and are therefore not directly comparable with the single hold-out figure in Table VII.

E. Effect of the Number of Trees

Accuracy was essentially flat as the forest grew (Table X). Ten trees already reached 83.4 %, and 100, 200 and 300 trees all gave 83.8 %, so 100 trees is an adequate and inexpensive setting.

TABLE X
RANDOM FOREST ACCURACY VERSUS NUMBER OF TREES

Trees	Accuracy (%)
10	83.4 ± 2.9
50	82.8 ± 1.0
100	83.8 ± 1.5
200	83.8 ± 2.0
300	83.8 ± 1.3

F. Feature Ablation

Each feature was removed in turn and the Random Forest re-evaluated by cross-validation (Table XI). Removing oxygen cost 5.0 points of accuracy, water level 4.0 and methane 3.0, which agrees with the importance ranking in Fig. 3. Removing temperature or humidity did not reduce accuracy, and removing H_2S changed it by only 0.5 points. In this simulated data H_2S may carry information that is partly redundant with other variables; this should not be read as evidence that H_2S is unimportant in real drains, where it is a primary toxic hazard [1], [4]. Temperature and humidity were kept because they affect heat stress and sensor behaviour.

TABLE XI
CROSS-VALIDATED ACCURACY WITH ONE FEATURE REMOVED

Features used	Accuracy (%)
All six	83.8
Without oxygen	78.8
Without water level	79.8
Without methane	80.8
Without H_2S	83.3
Without temperature	84.3
Without humidity	84.8

G. Error Analysis

Pooling the out-of-fold predictions of the four folds gives the confusion matrix in Fig. 5. Of the 153 Danger records, 150 (98.0 %) were identified correctly and the remaining 3 were predicted as Warning; none was predicted as Safe. Warning recall was only 39.0 % (16 of 41), with 25 Warning records escalated to Danger, and all four Safe records were predicted as Warning. The errors therefore lean towards the more severe class: 29 records were over-estimated and only 3 under-estimated. For a safety aid this is the preferable failure direction, but a high rate of unnecessary Danger alarms would cause alarm fatigue, which is another reason to obtain better-balanced and better-labelled data.

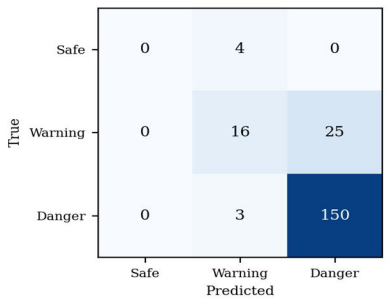


Fig. 5 Pooled out-of-fold confusion matrix of the Random Forest

VI. DISCUSSION AND LIMITATIONS

1) Class imbalance. With only four Safe records the model cannot learn that class; the single Safe test case was misclassified, and the macro F1 of 0.53 reflects this. Accuracy alone flatters the result, which is why per-class figures and a baseline are reported. Collecting more Safe and Warning examples, or using class weighting or resampling, is the first remedy.

2) Simulated and loosely defined labels. The labels were produced for the simulation and are not tied to published exposure limits. Some records with near-normal readings carry a Danger label, so part of the remaining error is label noise. Real use needs labels derived from recognised limits, such as the oxygen threshold above, the NIOSH ceiling and immediately-dangerous values for H_2S [4], and the lower explosive limit for methane [5], and validated by safety professionals.

3) Need for a safeguard. A learned model can misjudge inputs outside its training range. A sensible design is a hard-coded rule layer that forces Danger whenever any single reading crosses a regulatory limit, with the Random Forest refining the Safe/Warning distinction. This is proposed, not implemented.

4) Scope. Sensor calibration, drift, response time and cross-sensitivity are not modelled, and the buzzer path was not field-tested. The system must not be used as the sole basis for entering a drain.

5) Dataset scale versus regulatory scale. The simulated ranges differ from the levels that matter in practice (Table XII). Methane never exceeds 200 ppm in the dataset, below 0.5 % of its lower explosive limit, so a Danger label on such a record cannot be explained by flammability; it reflects the rules of the simulation. A field-ready model needs labels that respect recognised limits, and the values in Table XII should be checked against the current editions of [2], [4] and [5].

TABLE XII
DATASET RANGES COMPARED WITH COMMONLY CITED REFERENCE LEVELS

Parameter	Dataset range	Reference level
Oxygen	15.0–21.0 %	Below 19.5 % is oxygen-deficient [2]
H_2S	0.1–49.4 ppm	Ceiling 10 ppm; IDLH 100 ppm [4]
Methane	1.3–199.3 ppm	LEL about 5 % by volume (≈50,000 ppm) [5]

6) Threats to validity. Internally, the evaluation rests on 198 records and on one random seed for the hold-out split; cross-validation reduces but does not remove this sensitivity. The process that generated the labels is unknown to the model and may contain noise. Externally, no real drain data were used, so nothing here shows that the model generalises to field conditions. Regarding construct validity, accuracy on simulated labels is only a proxy for actual safety.

7) Deployment considerations. A deployed unit must behave safely when a sensor fails. If any reading is missing or outside its valid range, the assistant should report a sensor fault and treat the condition as unsafe instead of predicting from incomplete data. The model should be retrained whenever sensors are replaced or recalibrated, its output should be logged for audit, and workers should be trained to read the dashboard as one input among several. The very small input size makes inference fast on modest hardware, so a single low-cost computer can serve several nodes.

VII. CONCLUSION AND FUTURE WORK

The assistant shows that six drainage readings can be fused into a single colour-coded verdict with a Random Forest and delivered through a lightweight Flask dashboard. On simulated data it beat a majority-class baseline (87.5 % hold-out, 83.8 % cross-validated) and recognised Danger reliably (98.0 % recall in pooled cross-validation), with errors leaning towards over-warning. Other standard classifiers scored 80.8 % to 86.4 %, so the Random Forest is not claimed to be the best model on this data, and the evaluation exposes the need for balanced, realistic labels. Planned work: connect physical IoT sensors; collect and label field data with expert input; add the rule-based safeguard and sensor-fault handling; revisit the choice of classifier on real data; send SMS, e-mail or app notifications; store history for trend analysis; add flood forecasting from rainfall data; support GPS location of manholes; and offer a Tamil-language interface for sanitation workers.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

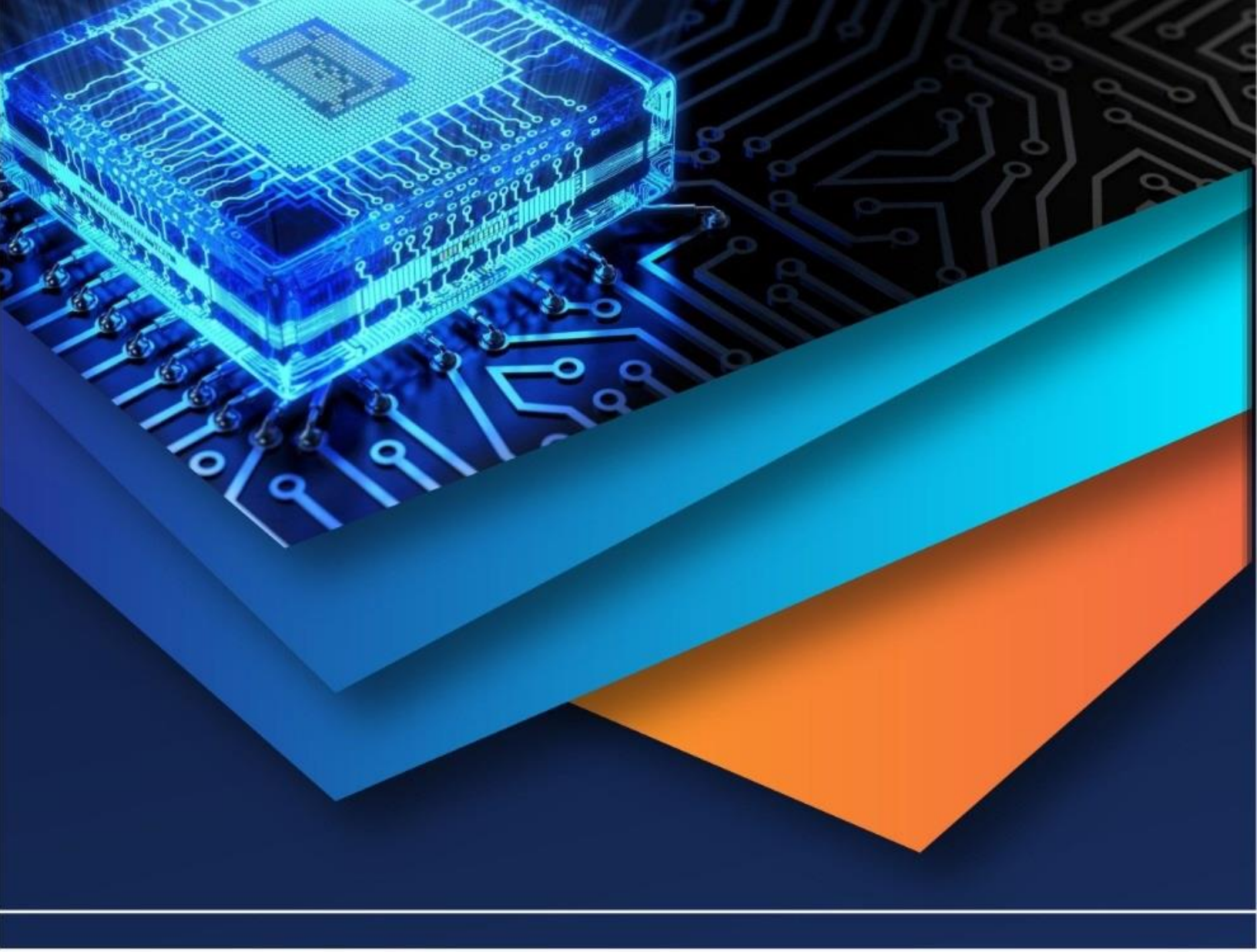
VIII. ACKNOWLEDGMENT

The authors thank the Principal, the Head of the Department of Computer Science and Engineering, the project coordinator and the faculty of Adhiyamaan College of Engineering, Hosur, for their guidance and facilities.

REFERENCES

[1] World Health Organization (WHO), "Hydrogen Sulfide," Public Health and Environmental Information.
[2] Occupational Safety and Health Administration (OSHA), "Permit-Required Confined Spaces," U.S. Department of Labor.
[3] NIOSH, "Preventing Occupational Fatalities in Confined Spaces."
[4] NIOSH, "NIOSH Pocket Guide to Chemical Hazards: Hydrogen Sulfide."
[5] NIOSH, "NIOSH Pocket Guide to Chemical Hazards: Methane."
[6] NIOSH, "Criteria for a Recommended Standard: Working in Confined Spaces."
[7] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.
[8] [8] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.
[9] [9] J. Han, M. Kamber and J. Pei, *Data Mining: Concepts and Techniques*, Morgan Kaufmann.
[10] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media.
[11] Pallets Projects, "Flask Documentation."
[12] Python Software Foundation, "Python Documentation."
[13] Scikit-learn Developers, "Random Forest Classifier Documentation."
[14] Scikit-learn Developers, "Data Preprocessing Documentation."
[15] W. McKinney, "Data Structures for Statistical Computing in Python," *Proc. 9th Python in Science Conf.*, pp. 56–61, 2010.
[16] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[17] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, McGraw-Hill.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)