COMPUTING AND ALGORITHM INSIGHT
Volume 1, Issue 1, June 2023 | pp. 45-70
https://doi.org/00.0000.000/xxxxx

OPEN ACCESS
Check for updates

OPTIMIZING FIRE DETECTION ACCURACY USING GENETIC PROGRAMMING SYMBOLIC CLASSIFIER (GPSC) AND SMOTE-BALANCED SENSOR FUSION DATA

Bambang Setiawan[1] ✉, Arief Hidayat[1], and Dedi Kurniawan [2]

Department of Electrical Engineering, Islamic University of Lamongan (UNISLA), Lamongan, East Java, Indonesia[1]; Department of Informatics, Galuh University, Ciamis, West Java, Indonesia[2]

ABSTRACT

Conventional fire detection systems primarily rely on smoke and heat sensors, yet environmental parameters like air temperature, humidity, and pressure offer critical supplementary data.1 This study employs a sensor fusion approach using a Genetic Programming Symbolic Classifier (GPSC) to generate a transparent AI model in the form of a symbolic expression. To address significant class imbalance, the investigation evaluates multiple resampling techniques, including Random Sampling, Near Miss-1, ADASYN, SMOTE, and Borderline SMOTE. These datasets were processed using a custom random hyperparameter search and 5-fold cross-validation to ensure model robustness. Results indicate that the SMOTE-balanced dataset yielded superior performance, achieving an accuracy of $0.998 \pm 4.79 \times 10^{-5}$ alongside consistently high values for AUC and F1-score. The paper concludes by presenting the optimal symbolic expression and validating its efficacy on the original dataset.

Keywords: *Fire Detection; Genetic Programming Symbolic Classifier (GPSC); Sensor Fusion; Imbalanced Datasets; SMOTE.*

CORRESPONDING AUTHOR:
Bambang Setiawan
Department of Electrical Engineering, Islamic University of Lamongan (UNISLA), Lamongan, East Java, Indonesia m
contact: *b.setiawan@unisla.ac.id*

ARTICLE HISTORY
Received : April 20, 2023
Final Revised : May 30, 2023
Accepted : June 17, 2023
Published : June 30, 2023

© 2023 The Author(s) | Published by Integra Academic Press | This article is an open access article and distributed under the terms and conditions of the Creative Commons Attribution (CC BY-SA) license.

1. | INTRODUCTION

Fire detection [1] is fundamentally based on sensing combustion byproducts, including smoke, heat, gases, and radiation (infrared or ultraviolet). These physical indicators are monitored using specialized heat, smoke, flame, or gas detectors. In residential environments, detection typically relies on cost-effective, standalone smoke detector units, whereas commercial and non-domestic infrastructure utilizes comprehensive fire alarm systems.

Smoke detectors, which identify smoke as a primary precursor to fire, operate primarily on two distinct principles: photoelectric (optical) and ionization. Photoelectric detectors [2] function by housing a light source (such as an LED or incandescent bulb producing IR/UV/visible light), a lens, and a receiver within an airflow chamber. In projected-beam configurations, a light beam traverses the air to a photosensor or reflector. When smoke, dust, or other airborne particulates obstruct this path, the light intensity is attenuated. If the intensity falls below a calibrated threshold, the system triggers an alarm.

Conversely, ionization detectors [3] utilize a small radioactive source positioned between two electrically charged electrodes. This source ionizes the air within the gap, facilitating a constant electrical current. The intrusion of smoke particles disrupts this ion flow, altering the current and subsequently activating the alarm mechanism.

A review of contemporary literature indicates that research on fire detection via smoke analysis predominantly employs image or video processing techniques. These studies extensively utilize Deep Convolutional Neural Networks (DCNN), including architectures such as ResNet [4], Inception [5,6], and YOLO [7]. While there is a limited number of studies focusing solely on the integration of standard smoke detectors with Artificial Intelligence (AI), there is a significant body of research investigating fire detection systems that leverage AI algorithms in conjunction with sensor fusion methodologies.

1.1. AI-Based Fire Detection Using Conventional Detectors

The integration of Artificial Intelligence (AI) with traditional detection hardware has been extensively explored to enhance fire identification reliability. In [8], an Adaptive Neuro-Fuzzy Inference System (ANFIS) achieved 100% accuracy in verifying fire incidents by analyzing the rates of change in smoke, temperature, and humidity, processing data through fuzzy logic principles. Furthermore, guidelines for optimizing sensor combinations for residential applications were proposed in [9]. This study also compared the efficacy of Neural Networks (NN) and Naive Bayes classifiers, reporting maximum accuracies of 97% and 100%, respectively. Similarly, [10] employed an Artificial Neural Network (ANN) utilizing smoke density, temperature, and carbon monoxide inputs, achieving a detection accuracy of 98.3%.

Alternative approaches focus on olfactory signatures; for instance, a Probabilistic Neural Network (PNN) analyzed data from an electronic nose sniffing vapors from materials scorched in a vacuum oven, yielding a mean accuracy of 94.18% [11]. In larger-scale experiments within a 240 m^3 facility, a gas sensor array was evaluated for distinguishing between smoldering/plastic fires and nuisance sources [12]. By applying Partial Least Squares-Discriminant Analysis (PLS-DA) and Support Vector Machines (SVM), the study found that while PLS-DA offered 100% specificity, its sensitivity was limited to 85%, indicating difficulties in differentiating plastic fire signatures from non-fire nuisances.

1.2. Sensor Fusion and AI in Fire Detection

Sensor fusion [13] is defined as the synergistic combination of data from multiple sources to reduce uncertainty beyond what is achievable by individual sensors. This methodology necessitates equipment with at least two distinct sensing modalities.

Research in [14] developed a mobile robot equipped with smoke, flame, and temperature sensors. Designed for autonomous pathfinding and obstacle avoidance, the robot utilized sensor fusion to locate fire sources, achieving 92% accuracy—surpassing single-sensor systems.

Several studies have focused on stationary arrays combining temperature, smoke, and CO detection. A three-layer fusion architecture (signal, characteristic, and decision layers) utilizing neural networks and fuzzy inference demonstrated an exceptionally low error rate of 10^{-4} [15]. Another approach utilized iterative weight adjustment of sensor inputs to generate a final decision, reporting high classification accuracy [16].

Complex arrays utilizing eight distinct sensors (including MOX, NDIR CO_2, and electrochemical CO sensors) were analyzed using PLS-DA in [17]. While the system achieved 97% sensitivity, it suffered from a high false alarm rate of 35%. Conversely, a simpler fusion of smoke, light, and temperature sensors processed via Arduino and Raspberry Pi achieved consistent accuracies of 97–98% across day and night conditions [18]. Additionally, incorporating Back-Propagation (BP) neural networks into indoor fusion systems has been shown to reduce detection times by 32% [19].

Comprehensive smart home ecosystems have been proposed, integrating wearable motion sensors for gesture control and positioning with multi-sensor fire detection modules [20]. Data processed via PNNs for safety monitoring achieved 98.81% accuracy. To address signal noise in Wireless Sensor Networks (WSN), ANN-based target localization was investigated in [21], demonstrating improved consistency across 11 training functions. Furthermore, data mining and ant colony optimization were applied to WSNs for early forest fire detection, optimizing node deployment [22].

Sensor fusion combines inputs from diverse modalities (cameras, radar, lidar) to construct unified environmental models, a technique prevalent in autonomous driving [23] and increasingly adopted across engineering disciplines [24,25]. However, challenges persist, such as data quality and missing values. Rahate et al. [27] demonstrated that multi-task modality fusion remains robust even with 90% missing data, challenging the assumption of "clean" datasets. Additionally, rapid processing requirements have led to innovations like FERNN for fusing RF and electro-optical data [28], and analytical approaches using Neighborhood Component Analysis (NCA) to streamline data for faster SVM processing without performance loss [29]. Singh et al. [26] further illustrated the utility of sensor fusion in IoT-enabled safety devices for mining environments. Collectively, current research underscores the critical role of AI and statistical methods in advancing sensor fusion technologies.

1.3. Novelty, Hypotheses, and Scientific Contribution

While prior studies utilizing AI/ML methods (specifically neural networks) demonstrate high accuracy, they often entail significant computational and storage overhead. This complexity hinders direct implementation on resource-constrained microcontrollers typical of multi-sensor systems, often necessitating additional hardware. To address this, this paper presents a novel application of the Genetic Programming Symbolic Classifier (GPSC) to generate a simple, interpretable symbolic expression for fire detection. Unlike "black-box"

models, the resulting symbolic expression requires minimal storage and is easily integrable into standard microcontrollers while maintaining high classification accuracy.

GPSC is an evolutionary algorithm that iteratively refines a population of symbolic expressions to fit a specific task, akin to supervised learning. This study applies GPSC to a publicly available dataset [30] comprising aerosol, humidity, temperature, pressure, and gas measurements across various scenarios (e.g., indoor/outdoor, wood/coal/gas fires).

Research Questions:

1. Can GPSC generate a symbolic expression that achieves high accuracy in fire detection?
2. Does balancing class distribution via undersampling/oversampling influence the accuracy of the resulting expressions?
3. Can a random hyperparameter search combined with 5-fold cross-validation yield optimal GPSC performance?
4. Will the best symbolic expression derived from a balanced dataset perform effectively on the original, imbalanced dataset?

Scientific Contributions:

1. Evaluation of GPSC applicability for fire detection using public multi-sensor data.
2. Analysis of the impact of data balancing techniques on symbolic model accuracy.
3. Validation of a random hyperparameter search framework with 5-fold cross-validation for GPSC optimization.
4. Verification of the optimal symbolic model's performance on original, real-world data distributions.

The remainder of this paper is organized as follows: Section 2 details the materials and methods, including the dataset, balancing techniques, GPSC algorithm, and evaluation metrics. Section 3 presents the results of the hyperparameter search and cross-validation, followed by a discussion of the final model's performance. Section 4 provides concluding remarks derived from the findings and hypotheses.

2. | MATERIALS AND METHODS

This section delineates the experimental framework employed in the study, encompassing the dataset characteristics, data balancing techniques, and the implementation of the Genetic Programming Symbolic Classifier (GPSC). Furthermore, it details the validation protocols, specifically the random hyperparameter optimization and 5-fold cross-validation methods, alongside the selected evaluation metrics and computational infrastructure.

2.1. Research Methodology

The schematic workflow of the research methodology utilized in this investigation is illustrated in Figure 1.

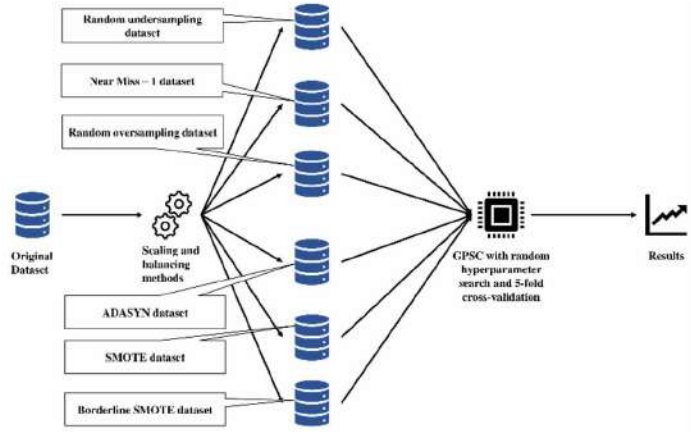


Figure 1. Schematic representation of the research methodology.

As illustrated in Figure 1, the experimental framework involved the application of six distinct data balancing techniques to the original dataset:

1. Random Undersampling;
2. Random Oversampling;
3. Near Miss-1;
4. Adaptive Synthetic (ADASYN);
5. Synthetic Minority Over-sampling (SMOTE);
6. Borderline Synthetic Minority Over-sampling (Borderline SMOTE).

Prior to resampling, the raw dataset was normalized using the Standard Scaler method. Consequently, six unique variations of the dataset were generated. Each variant was partitioned into training and testing subsets utilizing a 70:30 ratio. The training subset (70%) was subjected to a 5-fold cross-validation procedure coupled with a random search for GPSC hyperparameters. A performance threshold was established for this phase; if the resulting symbolic expressions yielded evaluation metrics exceeding 0.99, the model proceeded to the final training stage using the identified hyperparameters on the full training set. Conversely, if the metrics fell below 0.99, the process re-initialized with a new random selection of hyperparameters.

Upon completion of the final training, the resulting symbolic expression was evaluated against both the training and testing datasets to compute the mean and standard deviation of the performance metrics. A successful termination of the optimization loop required all metric values to remain above 0.99; otherwise, the hyperparameter search was restarted. Ultimately, the optimal symbolic expression was selected based on a trade-off between maximizing classification accuracy and minimizing expression complexity (size). The efficacy of the selected model was subsequently validated against the original, unbalanced dataset.

2.2. Dataset Description

This study utilizes a publicly accessible dataset available via Kaggle [30], originally compiled for a project focusing on AI-driven sensor fusion for fire detection. Sensor fusion is defined here as the aggregation of data from disparate sources to reduce uncertainty compared to individual sensor readings. The data acquisition hardware included a specific suite of sensors: the Sensirion SPS30 [31] for particulate matter (PM 1.0, PM 2.5) and number concentration (NC 0.5, NC 1.0, NC 2.5); the BME688 [32] for humidity and gas; the SHT31 [33] for temperature; the BMP390 [34] and BMP388 [35] for atmospheric pressure; and the

SPG30 [36] for gas sensing. All peripheral sensors were arranged around the central SPS30 smoke detector, and data collection was managed via a Nicla Sense ME board [37].

According to the dataset documentation [30], the project aimed to leverage diverse environmental parameters to enhance fire detection reliability. While various fusion algorithms (e.g., non-linear functions, thresholds) were considered, the original authors selected linear regression-based fusion to manage the high volume of correlated sensor readings effectively. Data collection encompassed diverse environmental conditions and fire sources, including normal indoor/outdoor baselines, indoor wood and gas fires, and outdoor scenarios involving grills (wood, coal, gas) and high humidity.

The initial dataset comprises 60,000 samples collected at a sample rate of 1 Hz, with no missing values. The raw data included 15 variables:

Timestep UTC [s];

Air Temperature [°C];

Air Humidity [%];

Total Volatile Organic Compounds (TVOC) [ppb];

CO_2 equivalent concentration [ppb];

Raw H_2 (molecular hydrogen);

Raw Ethanol Gas;

Air Pressure [hPa];

Particulate Matter sizes (PM 1.0 and PM 2.5) [μm];

Number Concentration of Particulate Matter (NC 0.5, NC 1.0, and NC 2.5);

Sample Counter (CNT);

Fire Alarm (Target: 0 for no alarm, 1 for activated alarm).

For the purpose of this study, administrative variables (UTC and CNT) were excluded to focus solely on physical sensor inputs. Consequently, the final dataset utilized 12 input features—Temperature, Humidity, TVOC, eCO_2, Raw H_2, Raw Ethanol, Pressure, PM1.0, PM2.5, NC0.5, NC1.0, and NC2.5—to predict the "Fire Alarm" target. The statistical descriptive analysis of these variables is presented in Table 1.

Table 1. Initial statistical summary and GPSC variable mapping for the original dataset.

Variable Name	Count	Mean	Std	Min	Max	GPSC Variable Representation
Temperature [C]	62,630	15.97042	14.35958	-22.01	59.93	X_0
Humidity [%]		48.5395	8.865367	10.74	75.2	X_1
TVOC [ppb]		1942.058	7811.589	0	60,000	X_2
eCO2 [ppm]		670.0211	1905.885	400	60,000	X_3
Raw H2		12,942.45	272.4643	10,668	13,803	X_4
Raw Ethanol		19,754.26	609.5132	15,317	21,410	X_5
Pressure [hPa]		938.6276	1.331344	930.852	939.861	X_6
PM1.0		100.5943	922.5242	0	14,333.69	X_7
PM2.5		184.4678	1976.306	0	45,432.26	X_8
NC0.5		491.4636	4265.661	0	61,482.03	X_9
NC1.0		203.5865	2214.739	0	51,914.68	X_{10}
NC2.5		80.04904	1083.383	0	30,026.44	X_{11}
target		0.714626	0.451596	0	1	y

To provide a deeper understanding of the feature distribution relative to the target variable, the data was partitioned into two distinct subsets corresponding to the non-activation and activation of the fire alarm. The statistical breakdown for these respective categories is presented in Table 2 and Table 3.

Table 2. Statistical summary of observations recorded during non-alarm conditions.

Variable Name	Count	Mean	Std	Min	Max
Temperature [C]	17,873	19.6948031	14.9829319	-22.01	59.93
Humidity [%]		42.9300767	11.9628544	10.74	75.2
TVOC [ppb]		4596.58725	14,255.5756	0	60,000
eCO2 [ppm]		962.5872552	1921.74993	400	39,185
Raw H2		12,896.3168	432.44162	10,668	13,803
Raw Ethanol		20,082.8235	956.339624	15,317	21,410
Pressure [hPa]		938.101383	1.23795718	931.131	939.861
PM1.0		261.982706	1439.7256	0	13,346.69
PM2.5		450.034639	2828.77478	0	41,262.98
NC0.5		1356.28382	7155.12266	0	61,482.03
NC1.0		493.872027	3150.39016	0	47,089.598
NC2.5		178.982259	1446.59579	0	26,916.836

Table 3. Statistical summary of observations recorded during alarm conditions.

Variable Name	Count	Mean	Std	Min	Max
Temperature [C]	44,757	14.4831516	13.8255854	-22.01	41.41
Humidity [%]		50.7795337	5.93723882	13.36	70.28
TVOC [ppb]		882.013071	548.606072	0	18,062
eCO2 [ppm]		553.189356	1275.26098	400	60,000
Raw H2		12,960.878	167.385665	10,939	13,637
Raw Ethanol		19,623.0504	307.123385	17,809	21,109
Pressure [hPa]		938.837806	1.3090303	930.852	939.771
PM1.0		36.1464057	590.458583	0.15	14,333.69
PM2.5		78.4178419	1493.57607	0.16	45,432.26
NC0.5		146.111337	2144.94205	1.06	60,442.71
NC1.0		87.6655491	1689.24266	0.165	51,914.68
NC2.5		40.5416272	895.171419	0.004	30,026.438

A comparative analysis of Table 2 and Table 3 reveals distinct variations in the mean values of input variables across the two distinct classes. Regarding environmental conditions, the mean temperature for the non-alarm state is 19.69 °C, whereas the alarm-triggering state averages 14.48 °C. Conversely, humidity levels display an inverse trend, rising from a mean of 42.93% in non-alarm conditions to 50.78% during alarm activation.

Significant disparities are also evident in gas concentrations. The Total Volatile Organic Compound (TVOC) levels drop significantly from a non-alarm mean of 4596.587 ppb to 882 ppb during alarm events. Similarly, the mean eCO_2 concentration decreases from 962.58 ppm (non-alarm) to 553.19 ppm (alarm). In terms of raw sensor data, raw H_2 averages 12,896.31 ppm in non-alarm states compared to 12,960.87 ppm in alarm states, while raw ethanol decreases from 20,082.82 ppm to 19,623.05 ppm, respectively. Atmospheric pressure shows a

marginal variance, averaging 938.1 hPa in non-alarm conditions versus 938.8 hPa when the alarm is triggered.

Particulate matter (PM) and Number Concentration (NC) metrics consistently exhibit higher mean values in the non-alarm dataset compared to the alarm dataset. Specifically, PM 1.0 averages 261.98 ppm (non-alarm) versus 36.14 ppm (alarm), while PM 2.5 averages 450.03 ppm versus 78.4 ppm. A similar pattern is observed for NC values: NC 0.5 (1356.35 vs. 146.1 ppm), NC 1.0 (493.8 vs. 87.7 ppm), and NC 2.5 (178.98 vs. 40.5 ppm).

A critical observation derived from these tables is the substantial class imbalance. The dataset contains 17,873 samples representing the non-alarm state, compared to 44,757 samples for the alarm-activated state. To mitigate this disparity, the balancing methodologies previously outlined will be employed to equalize class distribution.

Furthermore, Tables 1 through 3 highlight the heterogeneity in the dynamic ranges of the input variables. While variables such as temperature, humidity, and pressure operate within relatively narrow intervals, nine of the twelve variables span extensive ranges. Although preliminary experiments with the GPSC algorithm yielded promising classification accuracies (0.90–0.97) using raw data, the disparity in feature magnitudes necessitated normalization. Consequently, the Standard Scaler method [38] was implemented to standardize the input variables by centering the mean to zero and scaling to unit variance. The standard score is calculated utilizing the following expression:

$$z = \frac{x - u}{s}, \tag{1}$$

where u denotes the mean of the input variable samples, s represents the standard deviation, and x corresponds to the individual input sample value. It is important to note that this normalization procedure was exclusively applied to the input feature set, leaving the target output variable in its original format.

To quantify the linear relationship between each input feature and the target variable, Pearson's correlation analysis [39] was conducted. The resulting correlation coefficients range from -1 to 1, where a value of 0 indicates the absence of a linear relationship. A coefficient of -1 signifies a perfect negative correlation, implying an inverse relationship where an increase in the independent variable corresponds to a decrease in the dependent variable. Conversely, a coefficient of 1 indicates a perfect positive correlation, where both variables move in the same direction. Generally, coefficient values falling within the ranges of [-1, -0.5] and [0.5, 1] are considered indicative of strong associations. Values residing between -0.5 and 0.5 suggest weak correlations, which can theoretically pose challenges during the machine learning training phase; however, low linear correlation does not necessarily preclude effective predictive modeling. The visual representation of these correlations is provided in the heatmap in Figure 2.

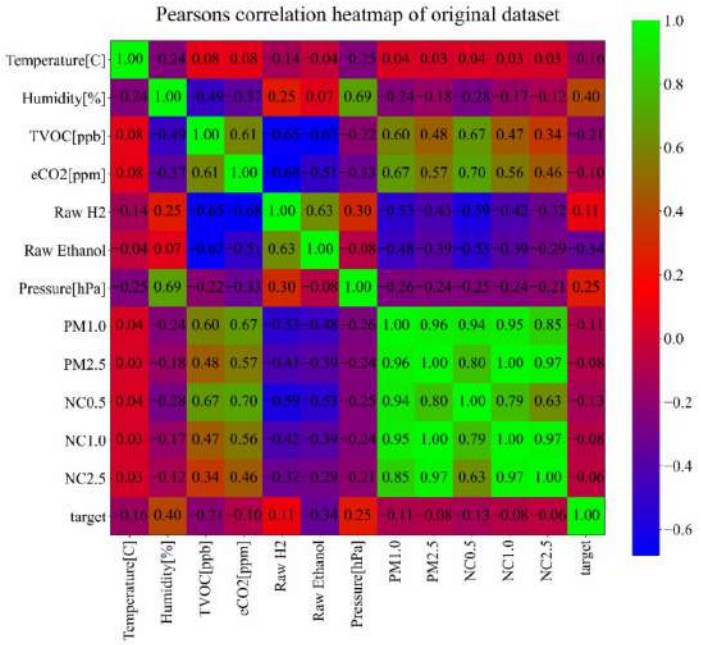


Figure 2. Heatmap illustrating the Pearson correlation coefficients derived from the original dataset.

As illustrated in Figure 2, only Humidity, Pressure, and Raw Ethanol demonstrate notable correlations with the target variable, whereas the remaining nine predictors exhibit relatively weak associations. It is also noteworthy that the parameters derived from the Sensirion SPS30 sensor display high multi-collinearity, with inter-correlation coefficients ranging between 0.63 and 0.99. Despite these disparities, this investigation retains the comprehensive set of input variables for the GPSC modeling process. This strategy allows the algorithm to autonomously determine feature relevance and select those variables that yield symbolic expressions with the highest classification accuracy. To further elucidate the relationship between the input features and the binary target (fire/no fire alarm), the correlation coefficients have been ranked in ascending order, as presented in Figure 3.

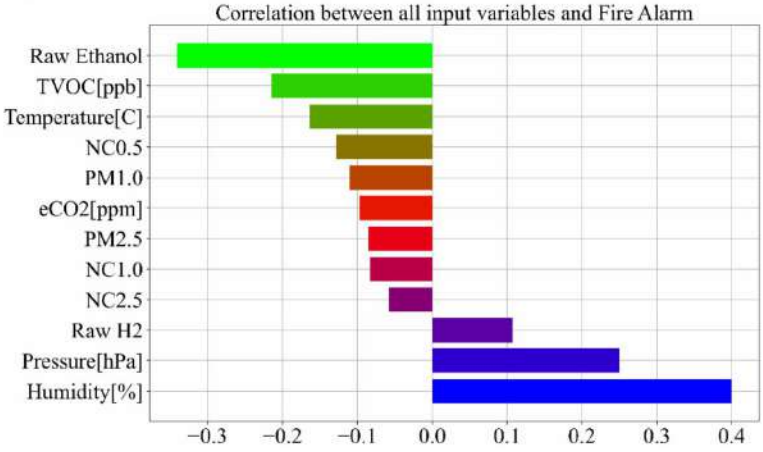


Figure 3. Correlation analysis between input variables and the target class.

As illustrated in Figure 3, the input variables exhibiting the strongest correlation with the Fire Alarm target are Raw Ethanol, Air Pressure, and Humidity. Conversely, particulate matter

variables demonstrate the weakest correlations, suggesting a low probability of their inclusion in the final symbolic expressions generated during the GPSC execution.

2.3. Data Balancing Methods

The original dataset presents a significant class imbalance, comprising 17,873 samples for the non-fire class (Class 0) and 44,757 samples for the fire class (Class 1). As documented in [40], such disparities can severely degrade the performance of supervised machine learning algorithms. To mitigate this issue, this study employs various undersampling and oversampling techniques to equalize the sample counts between the minority class (fewer samples) and the majority class (more samples). Specifically, the methods utilized include Random Undersampling, Near Miss-1, Random Oversampling, ADASYN, SMOTE, and Borderline SMOTE. Brief descriptions of these algorithms follow.

Random Undersampling [41] achieves dataset balance by arbitrarily selecting a subset of observations from the majority class. The process continues until the number of selected majority samples equals the total count of the minority class.

The Near Miss framework [42] encompasses a family of undersampling strategies that select majority class samples based on their Euclidean distance to minority samples in feature space. While variations exist—such as NearMiss-2 (based on distance to the furthest minority neighbors) and NearMiss-3 (based on minimum distance to each minority sample)—this investigation utilizes NearMiss-1. This specific variant retains majority class samples that exhibit the minimum average distance to the three closest minority class neighbors.

Random Oversampling [43] addresses imbalance by augmenting the minority class through the random duplication of existing samples. This iterative process continues until the minority class size matches that of the majority class, thereby achieving a balanced distribution.

The Adaptive Synthetic (ADASYN) algorithm [44] begins by quantifying the degree of class imbalance on a scale of 0 to 1. If a significant imbalance is detected, the algorithm computes the total number of synthetic samples required. Subsequently, for each minority class sample, the K nearest neighbors are identified in n-dimensional space using Euclidean distance to facilitate the generation of synthetic data based on the calculated ratio:

$$r_i = \frac{\Delta_i}{K} \quad i = 1, \ldots, m_s, \tag{2}$$

where Δ_i denotes the count of majority class examples found within the K nearest neighbors of a specific minority sample, and m_s represents the total population of the minority class. The calculated ratio r_i falls within the interval [0, 1]. Subsequently, r_i is normalized to establish a density distribution, which dictates the requisite number of synthetic samples to be generated for each minority instance. The generation process continues until the minority class count is equivalent to that of the majority class.

The Synthetic Minority Over-sampling Technique (SMOTE) addresses class imbalance by generating artificial data points for the underrepresented class rather than simply duplicating existing ones. As detailed by Chawla et al. [45], the algorithmic procedure involves three primary steps:

1. Calculate the difference between the feature vector of a target sample and its nearest neighbor;
2. Scale this difference by a random coefficient between 0 and 1;

3. Add the resulting value to the original feature vector to create a new synthetic sample along the line segment joining the two points.

Borderline SMOTE [46] refines the standard oversampling approach by specifically targeting minority examples located at the decision boundary between the majority and minority classes. The algorithm first identifies these critical "borderline" samples and subsequently generates synthetic data points exclusively from them, thereby reinforcing the separation between classes.

The statistical distribution of the dataset following the application of these various undersampling and oversampling methodologies is summarized in Table 4.

Table 4. Sample distribution per class resulting from the application of diverse undersampling and oversampling techniques.

Dataset Balancing Method	Class 0	Class 1	Total
Random Undersampling	17,873	17,873	35,746
Near Miss-1	17,873	17,873	35,746
Random Oversampling	44,757	44,757	89,514
SMOTE	44,757	44,757	89,514
Borderline SMOTE	44,757	44,757	89,514
ADASYN	44,759	44,757	89,516

2.4. Genetic Programming Symbolic Classifier

Genetic Programming (GP) is an evolutionary computation methodology that begins with an initial population of stochastic, task-agnostic candidates. Through the iterative application of genetic operators—specifically crossover and mutation—these candidates are refined to address a specific problem statement. The process initiates by synthesizing a population of naive symbolic expressions, constructed via the random selection of components (functions, constants, and variables) from a pre-defined primitive set. A defining characteristic of GP is the representation of population members as syntax tree structures. This topological representation is critical, as the complexity of a symbolic expression is quantified not merely by its linear length, but significantly by its tree depth.

To ensure structural diversity, the "ramped half-and-half" technique is employed for population initialization. This approach mandates that 50% of the initial population is generated using the 'full' method, while the remaining half utilizes the 'grow' method. A primary advantage of this strategy is the constraint of tree depths within a specific range (typically 3 to 12), which introduces greater initial diversity compared to exclusive reliance on either the full or grow methods. Within the GPSC architecture, the magnitude of the population and the geometric constraints of the trees are governed by the population_size and init_depth hyperparameters, respectively.

Following initialization, the fitness of each candidate must be assessed. The evaluation protocol within GPSC proceeds as follows:

1. Compute the raw output of the symbolic expression using the input variables from the dataset;
2. Transform this raw output by using it as the argument for the Sigmoid activation function, which is expressed as:

$$S(x) = \frac{1}{1+e^{-x}} \quad (3)$$

3. Compute the logarithmic loss by assessing the divergence between the sigmoid-transformed predictions and the actual ground truth labels for each respective instance.

Following the fitness evaluation of the population, the tournament selection mechanism is employed to identify candidates for reproduction. In this process, a subset of individuals is randomly sampled from the population, and the candidate exhibiting the optimal (lowest) fitness score is selected as the winner. The size of this sampling pool is governed by the tournament_size hyperparameter. These selected individuals subsequently undergo genetic modifications via four distinct operators: crossover, subtree mutation, hoist mutation, and point mutation.

Crossover requires two tournament winners (parents). A random subtree within the first parent is identified and replaced by a random subtree extracted from the second parent, thereby generating an offspring for the subsequent generation. Conversely, the three mutation operators act upon a single parent:

1. Subtree Mutation: A random subtree within the parent is replaced by a completely new, randomly generated subtree derived from the primitive set.
2. Hoist Mutation: This operator functions as a pruning mechanism. A random subtree is selected within the parent, and then a smaller sub-branch within that subtree is chosen to replace the larger original subtree, effectively reducing the program size.
3. Point Mutation: Individual nodes within the tree are randomly selected and substituted. Variables are replaced with other variables, constants with different constants, and functions with alternative functions of the same arity (argument count).

The probabilities for these operations are controlled by the hyperparameters p_crossover, p_subtree_mutation, p_hoist_mutation, and p_point_mutation. Ideally, the sum of these probabilities should approximate or equal 1. If the sum is less than 1, certain individuals will propagate to the next generation unchanged.

The termination of the GPSC algorithm is dictated by two primary hyperparameters: generations and stopping_criteria. The generations parameter establishes the maximum number of evolutionary cycles. The stopping_criteria defines a specific fitness threshold (typically near zero); if a population member achieves this value, execution halts. In practice, however, since the ideal fitness is rarely attained, the algorithm typically terminates upon reaching the maximum generation count.

Numeric constants integrated into the symbolic expressions are drawn from a specific interval defined by the const_range hyperparameter. Additionally, the max_samples parameter dictates the number of data points sampled from the training set for fitness evaluation.

To maintain model efficiency, a parsimony_coefficient is utilized to mitigate the "bloat phenomenon"—a scenario where symbolic expressions grow excessively large without yielding significant fitness improvements. This coefficient applies a penalty to larger trees, reducing their selection probability and thereby preventing memory overflows and promoting concise models.

Prior to implementing the random hyperparameter search, preliminary experiments were conducted to define effective operational ranges. This step was particularly critical for the parsimony coefficient, as inappropriate values can lead to either uncontrolled tree growth or the stagnation of evolution resulting in overly simplistic models. The established hyperparameter ranges utilized in this study are detailed in Table 5.

Table 5. Hyperparameter search space and range constraints utilized for the GPSC algorithm.

Hyperparameter Name	Lower Bound	Upper Bound
Population_size	500	2000
number_of_generations	200	300
tournament_size	100	500
init_depth	(3,7)	(7,12)
crossover	0.95	1
subtree_mutation	0.001	0.1
point_mutation	0.001	0.1
hoist_mutation	0.001	0.1
stopping_criteria	$1e^{-7}$	$1e^{-6}$
max_samples	0.99	1
constant_range	-100,000	100,000
parsimony_coeff	$1e^{-5}$	$1e^{-4}$

2.5. Random Hyperparameter Search

As outlined in the preceding section, the GPSC algorithm relies on a specific set of hyperparameters to govern its evolutionary process. To optimize performance, a random search strategy was implemented wherein hyperparameter values are stochastically selected from predefined intervals prior to each execution. These intervals were established through an initial trial-and-error empirical testing phase. This randomization process was operationalized via a dedicated function within the experimental script, ensuring a unique configuration for each run. The complete search space and constraints for these hyperparameters are detailed in Table 5.

2.6. 5-Fold Cross-Validation

To ensure the generation of robust symbolic expressions capable of high-accuracy fire detection, a 5-fold cross-validation protocol was adopted. Standard train/test splits often carry the risk of overfitting, where models perform exceptionally well on training data but fail to generalize to unseen data. Cross-validation effectively mitigates this issue, a benefit well-documented in various applications [47–49].

To rigorously assess model stability and rule out overfitting, the mean and standard deviation were computed for all evaluation metrics across the training and testing datasets. In this context, a high standard deviation would indicate a significant performance disparity between the training and testing phases. However, the results of this investigation demonstrated minimal variance, suggesting robust generalization capabilities. The integrated workflow combining random hyperparameter search with 5-fold cross-validation is structured as follows:

1. Initialization: Randomly sample GPSC hyperparameters from the ranges defined in Table 5.
2. Validation: Execute 5-fold cross-validation to calculate the mean values for Accuracy, Area Under the Receiver Operating Characteristic Curve (AUC), Precision, Recall, and F1-Score.
3. Thresholding (Stage 1): If the obtained metric values exceed 0.99, proceed to the final analysis. If not, the system resets, selects new random hyperparameters, and re-initiates the cross-validation loop.
4. Final Model Training: Upon satisfying the first threshold, the GPSC model is retrained on the full training partition using the successful hyperparameter configuration. The

resulting symbolic expression is then evaluated on both train and test sets to derive the final mean and standard deviation statistics for all performance metrics.

5. Thresholding (Stage 2): If the final mean evaluation metrics remain above 0.99, the process is deemed complete. Otherwise, the procedure restarts from the hyperparameter selection phase.

A flowchart illustrating this iterative methodological sequence is presented in Figure 4.

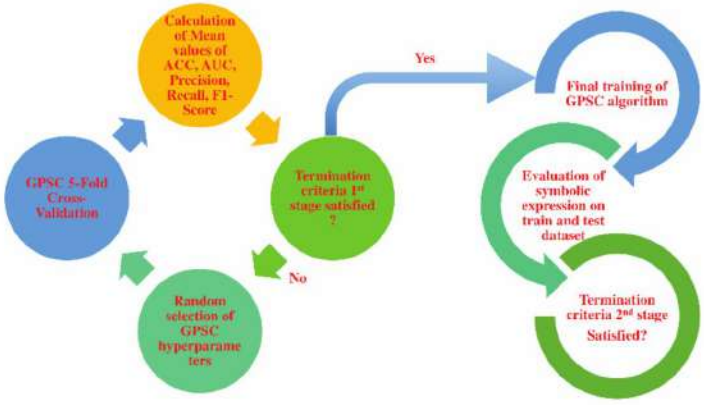


Figure 4. Schematic diagram illustrating the GPSC framework integrated with random hyperparameter optimization and 5-fold cross-validation.

2.7. Evaluation Metrics and Methodology

The efficacy of the symbolic expressions generated via GPSC was evaluated using a comprehensive suite of performance metrics: Accuracy, Area Under the Receiver Operating Characteristic Curve (AUC), Precision, Recall, and F1-Score. Rather than relying solely on test set performance, this study reports the mean values and standard deviations for these metrics across both training and testing datasets. This dual-dataset approach facilitates a rigorous assessment of the model's stability and its ability to generalize to unseen data.

As previously established, the quantitative assessment relies on Accuracy (ACC), AUC, Precision, Recall, and F1-Score. These metrics are derived from the fundamental components of the confusion matrix: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) [50]. In this context, TP and TN represent instances where the machine learning model correctly identifies positive and negative classes, respectively. Conversely, misclassifications are categorized as FP (incorrect prediction of the positive class) or FN (incorrect prediction of the negative class). These four parameters constitute the confusion matrix and serve as the foundational elements for calculating ACC, AUC, Precision, Recall, and F1-Score.Accuracy is defined as the proportion of total predictions that the model classifies correctly [51]. It is calculated using the following expression:

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \tag{4}$$

The AUC score [52] quantifies the model's discriminative capability by computing the integral area situated beneath the Receiver Operating Characteristic (ROC) curve.

Precision [53] serves as a performance metric designed to assess the accuracy of positive class predictions. Specifically, it determines the proportion of correctly identified positive instances relative to the total number of instances classified as positive. Mathematically, precision is defined as the ratio of True Positives (TP) to the aggregate of True Positives and False Positives (FP), expressed as follows:

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

Recall [53] characterizes the capacity of a machine learning model to successfully identify all actual positive instances within a dataset. Mathematically, it is defined as the ratio of True Positives (TP) to the total count of actual positives (the sum of TP and False Negatives, FN), and is formulated as follows:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

The F1-score [54] serves as the harmonic mean of precision and recall, effectively balancing the contributions of both metrics into a single performance indicator. Mathematically, the F1-score is computed using the following expression:

$$F1 - Score = 2\frac{Precision .Recall}{(Precision+Recall)} \quad (7)$$

The performance metrics employed in this study—ACC, AUC, Precision, Recall, and F1-Score—are normalized within the unit interval [0, 1], where a value of 1 represents optimal classification performance and 0 indicates complete model failure.

The evaluation framework operates on a strict, multi-stage validation protocol involving an initial 70:30 dataset partition for training and testing, respectively. The process iterates through the following stages:

1. Cross-Validation and Hyperparameter Search (Stage 1): The training subset undergoes 5-fold cross-validation utilizing randomly selected GPSC hyperparameters. A termination criterion is applied wherein the mean values for all evaluation metrics (ACC, AUC, Precision, Recall, F1-Score) must exceed a threshold of 0.99. If this performance benchmark is not met, the hyperparameters are discarded, and the search process re-initializes.
2. Final Model Evaluation (Stage 2): Upon satisfying the Stage 1 criteria, the GPSC algorithm is retrained on the full 70% training partition using the optimized hyperparameters to generate a symbolic expression. This derived expression is subsequently evaluated against both the training and the reserved 30% testing datasets.
3. Stability Verification: The final acceptance of the model is contingent upon a secondary termination criterion: the mean metric values must remain above 0.99, and the standard deviation across evaluations must be less than 10^{-3}. Failure to meet these stability and accuracy requirements triggers a complete restart of the hyperparameter search.

This methodology essentially bifurcates the validation into two distinct steps:

1. Step One: Execution of 5-fold cross-validation to filter for hyperparameter configurations that yield mean metric values exceeding the defined threshold (noted as >0.97 in preliminary screening, tightening to >0.99 for final acceptance).
2. Step Two: Final training and testing of the approved model to derive the ultimate mean and standard deviation statistics, ensuring the symbolic expression generalizes well to unseen data.

2.8. Computational Resources

All experimental investigations were executed on a workstation equipped with an AMD Ryzen 5 Mobile 5500U processor (6 cores, 12 threads) and 16 GB of DDR4 RAM. The software environment was established using the Python programming language (version 3.9.1). Data balancing operations utilized the imblearn library (version 0.9.1), while statistical and correlation analyses were conducted via pandas (version 1.0.5). The GPSC implementation

relied on the gplearn library (version 0.4.1), with matplotlib (version 3.4.3) employed for data visualization. It is important to note that the algorithms for the random hyperparameter search and the 5-fold cross-validation were custom-developed specifically for this research.

3. | RESULTS AND DISCUSSION

This section delineates the experimental findings derived from the investigation. Initially, the performance metrics obtained via the GPSC algorithm—augmented with random hyperparameter optimization and 5-fold cross-validation—are reported. Subsequently, the optimal symbolic expression, selected for achieving maximum classification accuracy, is presented. The section concludes with an evaluation of this specific expression against the original, unprocessed dataset.

A preliminary analysis was conducted using the GPSC on the original dataset, excluding the random hyperparameter search and cross-validation protocols. The primary objective of this phase was to empirically establish effective hyperparameter boundaries for the subsequent randomized search. This calibration was particularly critical for the parsimony coefficient, a sensitive parameter where excessive magnitude can stifle evolutionary progress (resulting in sub-optimal accuracy), while insufficient magnitude risks inducing the "bloat phenomenon." Additionally, the stopping criteria were set to an infinitesimal value to ensure the algorithm executed through the maximum allocation of generations, thereby minimizing the fitness function value. This preliminary phase also revealed that higher crossover coefficient values significantly contributed to fitness reduction and the generation of highly accurate symbolic expressions.

As indicated by the descriptive statistics in Table 1, the input variables exhibit significant heterogeneity in their dynamic ranges, with some possessing narrow intervals and others spanning extensive magnitudes. This disparity adversely affected the initial classification performance, capping mean evaluation metrics at approximately 0.97. Consequently, to enhance predictive accuracy, all input variables were normalized utilizing the Standard Scaler method.

3.1. Results Achieved with GPSC Using Random Hyperparameter Search Method and 5-Fold Cross-Validation

Each variation of the balanced dataset was subjected to the GPSC workflow, incorporating the random hyperparameter search and 5-fold cross-validation. The specific hyperparameter configurations that yielded the optimal symbolic expressions for each dataset variation are summarized in Table 6.

Table 6. Optimal randomly selected hyperparameters yielding the highest classification accuracy for the generated symbolic expressions.

Dataset Type	GPSC Hyperparameters (Population_Size, Number_of_Generations, Tournament_Size, Initial_Depth, Crossover, Subtree_Mutation, Hoist_Mutation, Point_Mutation, Stopping_Criteria, Max_Samples, Constant_Range, Parsimony_Coefficient)
Random Undersampling	1477, 221, 406, (5, 12), 0.96, 0.013, 0.013, 0.012, 7.82e-7, 0.99, (-13467.47, 36155.63), 5.45e-5
Near Miss-1	1422, 173, 290, (6, 10), 0.96, 0.0059, 0.0075, 0.021, 9e-6, 0.99, (-46197.99, 30568.98), 2.76e-5

Random Oversampling	654, 250, 383, (7, 11), 0.96, 0.023, 0.013, 0.0019, 6.79e-7, 0.99, (-76506.62, 63083.63), 5.5e-5
ADASYN	952, 252, 290, (7, 8), 0.96, 0.0051, 0.0016, 0.024, 1.56e-7, 0.99, (-47945.94, 94095.29), 3.34e-5
SMOTE	1111, 217, 190, (6, 11), 0.97, 0.005, 0.01, 0.0085, 8.12e-7, 0.99, (-12456.11, 25100.79), 9.7e-5
Borderline SMOTE	1194, 108, 180, (5, 8), 0.95, 0.014, 0.011, 0.013, 5e-6, 0.99, (-87036.2, 28148.73), 3.65e-5

Analysis of Table 6 reveals that the optimal symbolic expressions for the majority of dataset variations were generated utilizing relatively large population sizes. The notable exception is the Random Oversampling method, where the population size converged near the lower boundary defined in Table 5. As anticipated, given the pre-defined hyperparameter range of 0.95–1.0, the crossover coefficient emerged as the dominant genetic operator across all experimental scenarios. Furthermore, the stopping criteria were intentionally established at negligible thresholds (10^{-7}–10^{-6}) to ensure the GPSC algorithm executed through the full duration of the allocated generations rather than terminating prematurely. Regarding model complexity, the parsimony coefficient was maintained at a minimal value; while this successfully prevented the "bloat phenomenon," it is observed that several of the highest-performing models still retained significant structural size, as detailed in Table 7. The aggregate performance results, comprising mean evaluation metrics and their corresponding standard deviations (error bars), are illustrated in Figure 5.

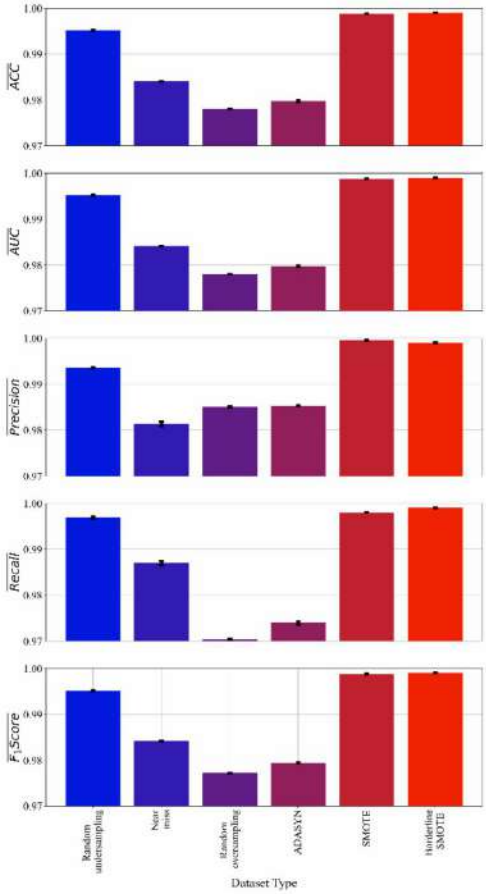


Figure 5. Mean evaluation metrics with standard deviation error bars achieved by the GPSC algorithm for each balanced dataset variant.

Table 7. Numerical representation of the mean and standard deviation for ACC, AUC, Precision, Recall, and F1-Score metrics.

Dataset Balancing Method	ACC ± SD	AUC ± SD	Precision ± SD	Recall ± SD	F1-Score ± SD	Avg. CPU Time [min]	Length/Depth
Random Undersampling	0.9952 ± 8.65e-5	0.9952 ± 8.67e-5	0.9936 ± 6.53e-5	0.996 ± 2.39e-4	0.995 ± 8.67e-5	360	188/32
Near Miss-1	0.984 ± 2.68e-5	0.984 ± 2.57e-5	0.981 ± 4.8e-4	0.987 ± 4.34e-4	0.984 ± 2.57e-5	360	460/51
Random Oversampling	0.978 ± 1.58e-5	0.978 ± 1.98e-5	0.985 ± 1.28e-4	0.97 ± 1.82e-4	0.977 ± 4.71e-5	600	83/23
ADASYN	0.979 ± 1.4e-4	0.979 ± 1.34e-4	0.985 ± 1.11e-4	0.973 ± 3.5e-4	0.979 ± 1.26e-4	600	727/38
SMOTE	0.998 ± 4.79e-5	0.998 ± 4.79e-5	0.999 ± 5.32e-5	0.998 ± 4.26e-5	0.998 ± 4.79e-5	600	140/25
Borderline SMOTE	0.999 ± 2.92e-5	0.999 ± 2.92e-5	0.999 ± 9.15e-5	0.999 ± 3.07e-5	0.999 ± 3.04e-5	600	450/43

As illustrated in Figure 5, the most favorable evaluation metrics were observed in models derived from datasets balanced via the SMOTE and Borderline SMOTE methods. To determine the optimal solution, a trade-off analysis between classification accuracy and model complexity (size) was conducted. Given that the standard deviations in Figure 5 are negligible, detailed performance statistics—including mean values, standard deviations, computational duration, and symbolic expression dimensions—are provided in Table 7.

A comparative assessment of Table 7 reveals that while both SMOTE and Borderline SMOTE yielded similarly high classification accuracies, the structural complexity of the resulting models differed significantly. The symbolic expression generated using the SMOTE dataset is notably more compact in terms of both length and depth compared to its Borderline SMOTE counterpart. Consequently, based on the principle of parsimony, the SMOTE-derived model was identified as the superior candidate.

Regarding computational expenditure, the average CPU execution time was directly correlated with the dataset size resulting from the specific balancing technique employed. Undersampling methods (Random Undersampling and Near Miss-1), which reduced the total sample count, required approximately 60 minutes per fold during cross-validation and an additional 60 minutes for final training, totaling 360 minutes. Conversely, oversampling

techniques (Random Oversampling, ADASYN, SMOTE, and Borderline SMOTE) expanded the dataset, thereby increasing the processing time to approximately 100 minutes per fold plus final training, resulting in a total execution time of 600 minutes.

In summary, all dataset variations produced symbolic expressions with classification accuracies exceeding 0.97. The lowest relative performance was observed in the Near Miss-1, Random Oversampling, and ADASYN variants, while the highest accuracy was shared by the SMOTE and Borderline SMOTE models. However, the excessive structural complexity of the Borderline SMOTE model (length/depth of 450/43) rendered it less desirable than the SMOTE model (140/25). Therefore, the symbolic expression obtained via SMOTE was selected as the optimal model for final evaluation.

3.2. The Final Evaluation of the Best Symbolic Expression

As established in the preceding analysis, the symbolic expression derived from the SMOTE-balanced dataset represents the optimal balance between predictive performance and structural complexity. This mathematical model is expressed as follows:

$$
\begin{aligned}
y_1 &= 5.8\log\left(0.26\sqrt{X_0 - 17.1}\right) + 0.63\sqrt{X_1 - 46.86} + 0.86\log\left(0.098(X_1 - 46.86)\right) \setminus \\
&+0.86\log\left(0.43\log\left(\log\left(\log\left(\log\left(9.6\times10^{-5}(X_2 - 2788.37)\right)\right)\right)\right) + \frac{0.048\sqrt{X_5 - 19849.2}}{X_6 - 938.471}\right) \setminus \\
&+2\log\Big(1.44\log\Big(0.43\log\Big(\log\Big(1.44\log\Big(0.036\sqrt{X_5 - 19849.2}\csc\Big(1.44\log\Big(2.3\log\Big(1.44\log\Big(0.43\log\Big(0.43\log\Big(\log\Big(\log\Big(9.6\times10^{-5}(X_2 \\
&\quad - 2788.37)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big) \\
&+2\csc\left(0.75(X_6 - 938.471)\right)\sqrt{\log\left(1.44\log\left(\frac{10380.5\log\left(0.75(X_6 - 938.471)\right)}{X_2 - 2788.37}\right)\right)} \setminus \\
&+2\sqrt{\log\left(1.44\log\left(1.44\log\left(\log\left(\log\left(9.6\times10^{-5}(X_2 - 2788.37)\right)\right)\right)\right)\right)}\csc\left(0.75(X_6 - 938.471)\right) \setminus \\
&+2\log\left(1.44\log\left(\log\left(1.44\log\left(\log\left(\log\left(9.6\times10^{-5}(X_2 - 2788.37)\right)\right)\right)\right)\right)\right) \setminus \\
&+0.86\log\left(\log\left(1.44\log\left(1.44\log\left(\log\left(0.43\log\left(0.43\log\left(\log\left(\log\left(9.6\times10^{-5}(X_2 - 2788.37)\right)\right)\right)\right)\right)\right)\right)\right)\right) \setminus \\
&+0.86\log\left(\log\left(1.44\log\left(1.44\log\left(\log\left(1.44\log\left(0.43\log\left(0.43\log\left(\log\left(\log\left(9.6\times10^{-5}(X_2 - 2788.37)\right)\right)\right)\right)\right)\right)\right)\right)\right)\right) \setminus \\
&+2.8\log\Big(1.44\log\Big(\log\Big(1.44\log\Big(0.43\log\Big(0.43\log\Big(0.43\log\Big(0.43\log\Big(0.43\log\Big(\log\Big(\log\Big(9.6\times10^{-5}(X_2 - 2788.37)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big)\Big) \\
&\quad \setminus \\
&+2.8\log\left(\log\left(0.75(X_6 - 938.471) - 0.003(X_4 - 12926.3)\right)\right) \setminus \\
&+2\sqrt{\log\left(1.44\log\left(0.003(X_4 - 12926.3)\right)\right)}\csc\left(\sin\left(\sin\left(\sin\left(\sin\left(0.75(X_6 - 938.471)\right)\right)\right)\right)\right) \setminus \\
&+0.006(X_4 - 12926.3) \setminus \\
&+2.4\sqrt{\log\left(0.43\log\left(\log\left(\log\left(1.44\log\left(0.003(X_4 - 12926.3)\right)\right)\right)\right)\right)} \setminus \\
&+0.86\log\left(\log\left(1.44\log\left(0.036\sqrt{X_5 - 19849.2}\csc\left(1.44\log\left(\frac{0.084\sqrt{X_5 - 19849.2}}{\log\left(\log\left(0.75(X_6 - 938.471)\right)\right)}\right)\right)\right)\right)\right) \setminus \\
&+0.036\sqrt{X_5 - 19849.2} + 6\log\left(1.44\log\left(\sin\left(0.75(X_6 - 938.471)\right)\right)\right) \setminus \\
&+2.4\sqrt{\log\left(0.43\log\left(\log\left(0.43\log\left(\log\left(1.44\log\left(0.43\log\left(\log\left(\sin\left(0.75(X_6 - 938.471)\right)\right)\right)\right)\right)\right)\right)\right)\right)}\csc\Big(\sin\Big(\sin\Big(\sin\Big(0.75(X_6 \\
&\quad - 938.471)\Big)\Big)\Big)
\end{aligned}
\tag{8}
$$

The optimal symbolic expression, presented in Equation (8), incorporates a specific subset of input variables: Temperature (X_0), Humidity (X_1), TVOC (X_2), Raw H_2 (X_4), Raw Ethanol (X_5), and Air Pressure (X_6). Notably, the particulate matter variables (PM1.0, PM2.5, NC0.5, NC1.0, and NC2.5) were excluded from the final model. This exclusion is theoretically consistent with the preliminary correlation analysis (see Figure 2), which demonstrated that while these variables exhibit high inter-correlation (0.64 to 1.0), they possess a negligible correlation with the target "Fire Alarm" variable.

To validate the model's efficacy, a final evaluation was conducted by applying the symbolic expression to the complete, original dataset. Performance was quantified using Accuracy, Area Under the Curve (AUC), Precision, Recall, and F1-Score. The computational procedure for this evaluation involves the following sequential steps:

1. Raw Output Calculation: The raw input values from the original dataset are processed through the symbolic expression to generate an initial numerical output.
2. Activation: This output is subsequently passed as the argument to the Sigmoid function (defined in Equation (3)) to normalize the result.
3. Binarization: The continuous output from the Sigmoid function is rounded to the nearest integer to yield a definitive binary classification (0 or 1).

It is crucial to highlight that external pre-processing via the Standard Scaler method is not required for this final deployment. The scaling parameters (mean and standard deviation) detailed in Equation (1) have been intrinsicially integrated into the mathematical structure of Equation (8), allowing it to operate directly on the raw original data. The quantitative results of this final evaluation are summarized in Table 8.

Table 8. Performance metrics achieved by applying the derived equation y_1 to the original dataset.

Evaluation Metric	Value
ACC	0.9984
AUC	0.9986
Precision	0.9997
Recall	0.998
F1-Score	0.9988

As evidenced by Table 8, the performance metrics demonstrate a marginal decrement when compared to the evaluation values obtained from the SMOTE-balanced dataset detailed in Table 7.

This subsection provides a comparative assessment of the proposed method against the state-of-the-art approaches previously delineated in the Introduction. Furthermore, a synthesis of the critical observations derived from the conducted experiments is presented at the conclusion of this section. The quantitative comparison between the results achieved on the original dataset in this study and those reported in the literature is summarized in Table 9.

Table 9. Comparative assessment of the results achieved in this study against the prior research reviewed in the Introduction.

References	Method	Fire or Smoke	Results
[8]	ANFIS	Fire	ACC: 100%
[9]	ANN, Naive Bayes	Smoke/Fire	ACC: 97–100%
[10]	ANN	Fire/Smoke	ACC: 98.3%
[11]	PNN	Fire	ACC: 94.8%
[17]	PLS-DA	Fire	Sensitivity: 97%
[14]	PLS	Fire	ACC: 92%
[15]	Neural Network	Fire	classification error 10^-4
[16]	BP Neural Network	Fire	ACC: 98%
[18]	Dempster-Shafer Theory	Fire	ACC: 97–98%
[20]	PNN	Fire	ACC: 98.81%
This investigation	GPSC	Fire	ACC: 99.84%

As illustrated in Table 9, previously published research in fire and smoke detection reports classification accuracies ranging from 92% to 100%. Consequently, the results achieved in this study surpass the majority of existing benchmarks. While studies [8, 9] report perfect accuracy (100%) utilizing ANFIS and ANN algorithms, these methods generally demand significantly

greater computational resources compared to the symbolic expressions derived in this work. Furthermore, it is important to note that the referenced studies benefited from utilizing inherently balanced datasets for training. While acquiring a naturally balanced dataset is ideal, the use of dataset balancing techniques remains a highly effective alternative for initializing machine learning training when such data is unavailable.

Based on the conducted investigation, several key observations can be synthesized:

1. GPSC is an effective tool for generating symbolic expressions capable of detecting fire via sensor fusion data. However, achieving optimal classification accuracy necessitates rigorous hyperparameter tuning, which was successfully accomplished here via a random hyperparameter search.
2. The random hyperparameter search method is highly effective for identifying optimal configuration sets. However, it is computationally intensive and time-consuming, particularly given the magnitude of the datasets involved.
3. Due to the severe class imbalance in the raw data, direct analysis was unfeasible. The application of balancing methods was essential, providing a viable foundation that ultimately led to the generation of high-accuracy symbolic models.
4. The most effective symbolic expressions were derived from datasets balanced via SMOTE and Borderline SMOTE. Although both yielded comparable accuracy, the model generated using Borderline SMOTE was approximately three times larger in structural length. Consequently, the SMOTE-derived model was selected as the optimal solution due to its superior efficiency-to-accuracy ratio.
5. The validation of the final model against the original dataset confirms the efficacy of the proposed workflow: balancing the training data to optimize the learner, followed by evaluation on the original, unmodified distribution.

4. | CONCLUSIONS

This study demonstrates the successful application of the Genetic Programming Symbolic Classifier (GPSC)—augmented by random hyperparameter search and 5-fold cross-validation—to publicly available datasets for the generation of robust fire detection models.

The findings confirm that GPSC is capable of producing symbolic expressions with high classification accuracy. The integration of resampling techniques (ADASYN, SMOTE, and Borderline SMOTE) proved critical, influencing the performance significantly and yielding accuracies between 0.97 and 0.999. Notably, the optimal symbolic expression retained high performance even when deployed on the original, imbalanced dataset. Furthermore, feature selection occurred naturally during the evolutionary process; the best model excluded particulate matter variables entirely, relying solely on Temperature, Humidity, TVOC, Raw H_2, Raw Ethanol, and Air Pressure.

Advantages of the Proposed Method:

1. Computational Efficiency: Unlike complex "black-box" ML algorithms, the resulting symbolic expressions are mathematically concise and require minimal computational power for execution.
2. Effective Initialization: The use of data balancing provided a stable starting point, enabling the GPSC to converge on high-performance solutions.
3. Robustness: The combination of random hyperparameter search and 5-fold cross-validation ensured the generation of statistically robust models.

Disadvantages and Limitations:

1. Hyperparameter Sensitivity: The random search requires predefined bounds derived from preliminary testing. Parameters such as population size, generation count, and tournament size directly correlate with execution time. The parsimony coefficient is particularly sensitive; values that are too low risk code bloat, while excessively high values can stifle evolutionary growth, leading to underfitting.
2. Training Latency: Oversampling techniques significantly increase the dataset size, thereby extending the computational time required for the GPSC training phase.

This approach illustrates that a relatively simple GPSC algorithm, when applied to sensor fusion data, can yield a robust mathematical model for fire detection. A significant benefit of this symbolic approach over traditional ML models is its low memory and processing footprint, allowing for easy integration into resource-constrained microcontrollers like Arduino to control multi-sensor systems.

Future research will focus on the development of a physical multi-sensor hardware system to collect a naturally balanced dataset. Once a comprehensive dataset is established, various machine learning methods will be benchmarked to derive models with maximal accuracy. These models will subsequently be deployed on microcontrollers to empirically measure detection latency across diverse fire scenarios.

REFERENCES

1. Khan, F.; Xu, Z.; Sun, J.; Khan, F.M.; Ahmed, A.; Zhao, Y. Recent Advances in Sensors for Fire Detection. Sensors 2022, 22, 3310.
2. Seo, B.K.; Nam, S.G. Study of the improvement of false fire alarms in analog photoelectric type smoke detectors. Fire Sci. Eng. 2016, 30, 108–115.
3. Newman, J.S. Modified theory for the characterization of ionization smoke detectors. Fire Saf. Sci. 1994, 4, 785–792.
4. Mohammed, K.R. A real-time forest fire and smoke detection system using deep learning. Int. J. Nonlinear Anal. Appl. 2022, 13, 2053–2063.
5. Zheng, X.; Chen, F.; Lou, L.; Cheng, P.; Huang, Y. Real-Time Detection of Full-Scale Forest Fire Smoke Based on Deep Convolution Neural Network. Remote Sens. 2022, 14, 536.
6. Ryu, J.; Kwak, D. A Study on a Complex Flame and Smoke Detection Method Using Computer Vision Detection and Convolutional Neural Network. Fire 2022, 5, 108.
7. Masoom, S.M.; Zhang, Q.; Dai, P.; Jia, Y.; Zhang, Y.; Zhu, J.; Wang, J. Early Smoke Detection Based on Improved YOLO-PCA Network. Fire 2022, 5, 40.
8. Sarwar, B.; Bajwa, I.S.; Jamil, N.; Ramzan, S.; Sarwar, N. An intelligent fire warning application using IoT and an adaptive neuro-fuzzy inference system. Sensors 2019, 19, 3150.
9. Muhammad, K.; Khan, S.; Palade, V.; Mehmood, I.; De Albuquerque, V.H.C. Edge intelligence-assisted smoke detection in foggy surveillance environments. IEEE Trans. Ind. Inform. 2019, 16, 1067–1075.

10. Choueiri, S.; Daoud, D.; Harb, S.; Achkar, R. Fire and Smoke Detection Using Artificial Neural Networks. In Proceedings of the 2020 IEEE 14th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 16–17 December 2020; pp. 1–6.
11. Andrew, A.; Shakaff, A.; Zakaria, A.; Gunasagaran, R.; Kanagaraj, E.; Saad, S. Early stage fire source classification in building using artificial intelligence. In Proceedings of the 2018 IEEE Conference on Systems, Process and Control (ICSPC), Melaka, Malaysia, 14–15 December 2018; pp. 165–169.
12. Solórzano, A.; Eichmann, J.; Fernandez, L.; Ziems, B.; Jiménez-Soto, J.M.; Marco, S.; Fonollosa, J. Early fire detection based on gas sensor arrays: Multivariate calibration and validation. Sens. Actuators Chem. 2022, 352, 130961.
13. Sasiadek, J.Z. Sensor fusion. Annu. Rev. Control. 2002, 26, 203–228.
14. Sucuoglu, H.S.; Bogrekci, I.; Demircioglu, P. Development of mobile robot with sensor fusion fire detection unit. IFAC-PapersOnLine 2018, 51, 430–435.
15. Chen, S.; Bao, H.; Zeng, X.; Yang, Y. A fire detecting method based on multi-sensor data fusion. In Proceedings of the SMC'03 Conference Proceedings, 2003 IEEE International Conference on Systems, Man and Cybernetics, Conference Theme-System Security and Assurance (Cat. No. 03CH37483), Washington, DC, USA, 8 October 2003; Volume 4, pp. 3775–3780.
16. Liang, Y.H.; Tian, W.M. Multi-sensor fusion approach for fire alarm using BP neural network. In Proceedings of the 2016 International Conference on Intelligent Networking and Collaborative Systems (INCoS), Ostrawva, Czech Republic, 7–9 September 2016; pp. 99–102.
17. Solórzano, A.; Fonollosa, J.; Fernández, L.; Eichmann, J.; Marco, S. Fire detection using a gas sensor array with sensor fusion algorithms. In Proceedings of the 2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN), Montreal, QC, Canada, 28–31 May 2017; pp. 1–3.
18. Ting, Y.Y.; Hsiao, C.W.; Wang, H.S. A data fusion-based fire detection system. IEICE Trans. Inf. Syst. 2018, 101, 977–984.
19. Wu, L.; Chen, L.; Hao, X. Multi-sensor data fusion algorithm for indoor fire early warning based on BP neural network. Information 2021, 12, 59.
20. Hsu, Y.L.; Chou, P.H.; Chang, H.C.; Lin, S.L.; Yang, S.C.; Su, H.Y.; Chang, C.C.; Cheng, Y.S.; Kuo, Y.C. Design and implementation of a smart home system using multisensor data fusion technology. Sensors 2017, 17, 1631.
21. Jondhale, S.R.; Sharma, M.; Maheswar, R.; Shubair, R.; Shelke, A. comparison of neural network training functions for rssi based indoor localization problem in WSN. In Handbook of Wireless Sensor Networks: Issues and Challenges in Current Scenario's; Springer: Berlin, Germany, 2020; pp. 112–133.
22. Farooqui, N.A.; Tyagi, A. Data Mining and Fusion Techniques for Wireless Intelligent Sensor Networks. In Handbook of Wireless Sensor Networks: Issues and Challenges in Current Scenario's; Singh, P.K., Bhargava, B.K., Paprzycki, M., Kaushal, N.C., Hong, W.C., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 592–615.
23. Butt, F.A.; Chattha, J.N.; Ahmad, J.; Zia, M.U.; Rizwan, M.; Naqvi, I.H. On the Integration of Enabling Wireless Technologies and Sensor Fusion for Next-Generation Connected and Autonomous Vehicles. IEEE Access 2022, 10, 14643–14668.

24. Yusuf, A.; Sulaiman, T.A.; Alshomrani, A.S.; Baleanu, D. Optical solitons with nonlinear dispersion in parabolic law medium and three-component coupled nonlinear Schrödinger equation. Opt. Quan. Electron. 2022, 54, 1–13.
25. Ren, X.; Li, C.; Ma, X.; Chen, F.; Wang, H.; Sharma, A.; Gaba, G.S.; Masud, M. Design of multi-information fusion based intelligent electrical fire detection system for green buildings. Sustainability 2021, 13, 3405.
26. Singh, N.; Gunjan, V.K.; Chaudhary, G.; Kaluri, R.; Victor, N.; Lakshmanna, K. IoT enabled HELMET to safeguard the health of mine workers. Comput. Commun. 2022, 193, 1–9.
27. Rahate, A.; Mandaokar, S.; Chandel, P.; Walambe, R.; Ramanna, S.; Kotecha, K. Employing multimodal co-learning to evaluate the robustness of sensor fusion for industry 5.0 tasks. Soft Comput. 2022, 2022, 1–17.
28. Vakil, A.; Liu, J.; Zulch, P.; Blasch, E.; Ewing, R.; Li, J. A survey of multimodal sensor fusion for passive RF and EO information integration. IEEE Aerosp. Electron. Syst. Mag. 2021, 36, 44–61.
29. Mian, T.; Choudhary, A.; Fatima, S. A sensor fusion based approach for bearing fault diagnosis of rotating machine. Proc. Inst. Mech. Eng. Part O J. Risk Reliab. 2022, 236, 661–675.
30. Contractor, D. Kaggle: Smoke Detection Dataset. 2022. Available online: https://www.kaggle.com/datasets/deepcontractor/smoke-detection-dataset (accessed on 22 November 2022).
31. Sensirion Company. Sensor for HVAC and Air Quality Applications SPS30 Datasheet. Available online: https://cdn.sparkfun.com/assets/2/d/2/a/6/Sensirion_SPS30_Particulate_Matter_Sensor_v0.9_D1__1_.pdf (accessed on 22 November 2022).
32. Bosch. BME688 4-in-1 Air Quality Breakout (Gas, Temperature, Pressure, Humidity) Datasheet. Available online: https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme688/ (accessed on 22 November 2022).
33. ±2% (0–100%RH) Digital Humidity and Temperature Sensor. Available online: https://sensirion.com/products/catalog/SHT31DISB/ (accessed on 22 November 2022).
34. Bosch BMP390 Barometric Pressure Sensor. Available online: https://eu.mouser.com/new/bosch/bosch-bmp390-pressure-sensor/ (accessed on 22 November 2022).
35. Available online: https://www.bosch-sensortec.com/products/environmental-sensors/pressure-sensors/bmp388/ (accessed on 22 November 2022).
36. Multi-Gas (VOC and CO2eq) Sensor. Available online: https://sensirion.com/products/catalog/SGP30/ (accessed on 22 November 2022).
37. Arduino Officia: NICLA Sense Me. Available online: https://store.arduino.cc/products/nicla-sense-me (accessed on 22 November 2022).
38. Raju, V.G.; Lakshmi, K.P.; Jain, V.M.; Kalidindi, A.; Padma, V. Study the influence of normalization/transformation process on the accuracy of supervised classification. In Proceedings of the 2020 IEEE Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 20–22 August 2020; pp. 729–735.
39. Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In Noise Reduction in Speech Processing; Springer: Berlin, Germany, 2009; pp. 1–4.

40. Jeni, L.A.; Cohn, J.F.; De La Torre, F. Facing imbalanced data–Recommendations for the use of performance metrics. In Proceedings of the 2013 IEEE Humaine Association Conference on Affective Computing and Intelligent Interaction, Geneva, Switzerland, 2–5 September 2013; pp. 245–251.
41. Prusa, J.; Khoshgoftaar, T.M.; Dittman, D.J.; Napolitano, A. Using random undersampling to alleviate class imbalance on tweet sentiment data. In Proceedings of the 2015 IEEE International Conference on Information Reuse and Integration, San Francisco, CA, USA, 13–15 August 2015; pp. 197–202.
42. Babikir, M. Imbalanced Data Classification Enhancement Using SMOTE and NearMiss Sampling Techniques. Ph.D. Thesis, Sudan University of Science & Technology, Khartoum, Sudan, 2021.
43. Pang, Y.; Chen, Z.; Peng, L.; Ma, K.; Zhao, C.; Ji, K. A signature-based assistant random oversampling method for malware detection. In Proceedings of the 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), Rotorua, New Zealand, 5–8 August 2019; pp. 256–263.
44. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–8 June 2008; pp. 1322–1328.
45. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. J. Artif. Intell. Res. 2002, 16, 321–357.
46. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In Proceedings of the International Conference on Intelligent Computing, Springer, Shenzhen, China, 12–15 August 2005; pp. 878–887.
47. Lorencin, I.; Baressi Šegota, S.; Anđelić, N.; Mrzljak, V.; Ćabov, T.; Španjol, J.; Car, Z. On urinary bladder cancer diagnosis: Utilization of deep convolutional generative adversarial networks for data augmentation. Biology 2021, 10, 175.
48. Lorencin, I.; Anđelić, N.; Mrzljak, V.; Car, Z. Genetic algorithm approach to design of multi-layer perceptron for combined cycle power plant electrical power output estimation. Energies 2019, 12, 4352.
49. Anđelić, N.; Lorencin, I.; Glučina, M.; Car, Z. Mean Phase Voltages and Duty Cycles Estimation of a Three-Phase Inverter in a Drive System Using Machine Learning Algorithms. Electronics 2022, 11, 2623.
50. Sammut, C.; Webb, G.I. Encyclopedia of Machine Learning; Springer Science & Business Media: Berlin, Germany, 2011.
51. Sturm, B.L. Classification accuracy is not enough. J. Intell. Inf. Syst. 2013, 41, 371–406.
52. Hand, D.J.; Till, R.J. A simple generalisation of the area under the ROC curve for multiple class classification problems. Mach. Learn. 2001, 45, 171–186.
53. Flach, P.; Kull, M. Precision-recall-gain curves: PR analysis done right. Adv. Neural Inf. Process. Syst. 2015, 28, 1–9.
54. Chicco, D.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genom. 2020, 21, 6.