Course Project Report On

Community SOS

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Artificial Intelligence & Data Science

By

Fayas Asis(U2308025),Jain Rhonson(U2308030),

Jeswin George(U2308033),Jishnu Jinesh(U2308035), Pranav P Nair(U2308054)

Under the guidance of

Ms. Shyama R

Department of Artificial Intelligence & Data Science

Rajagiri School of Engineering & Technology (Autonomous)

(Parent University: APJ Abdul Kalam Technological University)

Rajagiri Valley, Kakkanad, Kochi, 682039

CERTIFICATE

This is to certify that the project report/seminar report entitled" CommunitySOS" is a bonafide record of the work done by Fayas Asis(U2308025), Jain Rhonson(U2308030), Jeswin George(U2308033), Jishnu Jinesh(U2308035), Pranav P Nair(U2308054), submitted to the Rajagiri School of Engineering & Technology (RSET) (Autonomous) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in" Artificial Intelligence & Data Science" during the academic year 2025-2026.

Project Guide:

Ms. Shyama R	Dr. Sherly K K
Assistant Professor	Professor & HoD
Department of Artificial Intelligence	Department of Artificial Intelligence and
Data Science	and Data Science
RSET	RSET

Acknowledgement

We wish to express our sincere gratitude towards **Rev Dr. Jaison Paul Mullerikkal CMI**, Principal of RSET, and **Dr. Sherly K K**, Head of the Department of Artificial Intelligence & Data Science for providing us with the opportunity to undertake our project, CommunitySOS.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our project guide **Ms. Shyama R** for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, We would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Fayas Asis

Jain Rhonson

Jeswin George

Jishnu Jinesh

Pranav P Nair

Abstract

CommunitySOS is an interactive, data-driven emergency verification dashboard developed as a practical and comprehensive demonstration of data mining and warehousing concepts. The system addresses the common challenges of information overload and reliance on external services in crisis management by simulating real-time emergency monitoring using a locally stored dataset of 100 reports from Kerala, India. Functioning entirely offline, it operates independently of live AI or cloud-based services, ensuring resilience in resource-limited environments.

At its core, the project features an integrated data mining pipeline. This process begins with data pre-processing, where raw textual data is extracted, cleaned, and tokenized. It then employs data reduction, allowing operators to interactively filter events by a specific location and radius, thereby focusing the analysis on a relevant geographic area. The central analytical component is a custom-built Naive Bayes classifier, a supervised learning model trained on the local dataset to categorize emergency events into classes such as 'Fire' or 'Traffic Accident' in real-time.

The dashboard enables operators to visualize and manage these emergencies through a responsive interface that includes a live event log simulating an incoming data stream, a dedicated panel for verified alerts, and an interactive geographic map for crucial spatial awareness. Automated verification mechanisms identify credible incidents by analyzing the number of reporters for an event , while a keyword-based analysis of the report's content flags high-priority incidents that require immediate attention. Further enhancing its utility, users can simulate live data feeds and generate detailed session reports in a downloadable PDF format for post-event review.

Contents

List of Figures

Chapter 1

Introduction

This chapter introduces the CommunitySOS project, a data-driven emergency verification dashboard designed as a practical demonstration of data mining and warehousing techniques. It explains the background and significance of the system, defines the problem it addresses, outlines its scope, motivation, objectives, and challenges, and discusses its societal relevance. The chapter concludes with the organization of the report, providing a roadmap for the subsequent chapters.

1.1 Background

In today's digital era, the rapid exchange of information through social platforms and digital communication channels has revolutionized how emergencies are reported and handled. However, distinguishing genuine emergencies from false or duplicate reports remains a major challenge, especially when such systems rely on manual verification or real-time external APIs. Traditional emergency management systems often depend on human monitoring and external data services, which can introduce latency, inconsistency, or privacy concerns.

CommunitySOS was conceptualized to bridge this gap by creating an offline, self-contained system capable of classifying and verifying emergency reports using data mining and text classification techniques. By simulating real-time emergency feeds, the project demonstrates the integration of data preprocessing, data reduction, and supervised learning through a Naive Bayes classifier trained on localized data. This approach not only ensures reliability and privacy but also provides a valuable educational platform for understanding how mined data can be transformed into actionable intelligence. The project highlights the importance of applying machine learning and warehousing principles to improve decision-making and emergency responsiveness in both societal and academic contexts.

1.2 Problem Definition

Emergency response agencies face significant challenges during crises, primarily information overload and the difficulty in rapidly verifying and prioritizing incoming reports. Traditional systems struggle with the sheer volume of unstructured data and often depend on unreliable online services, particularly when infrastructure fails.

1.3 Scope and Motivation

The scope of this project encompasses the development of a web-based dashboard that simulates real-time emergency monitoring through local dataset processing. The system integrates data extraction, transformation, and loading (ETL) with classification and visualization functionalities. It features an interactive dashboard, location-based filtering, automated verification, and PDF report generation, all running entirely within the browser environment. The project serves as a demonstration of data mining concepts such as text classification, feature extraction, and data reduction, presented through an intuitive and educational interface.

The motivation behind CommunitySOS stems from the increasing demand for intelligent, accessible, and privacy-preserving emergency management tools. Many existing solutions rely heavily on cloud-based AI or external data sources, limiting their accessibility in offline or resource-limited environments. This project was motivated by the idea of enabling offline intelligence—a system capable of analyzing, verifying, and visualizing emergencies autonomously. Furthermore, the project serves as a learning platform for students and researchers to practically apply data mining and machine learning concepts in real-world contexts, bridging the gap between academic knowledge and real-world applications.

1.4 Objectives

- To design and implement an interactive, offline emergency verification dashboard using web technologies.

- To preprocess and structure textual emergency data through data extraction, cleaning, and transformation.
- To apply supervised learning using a custom **Naive Bayes classifier** for text-based emergency categorization.
- To implement **data reduction** using location-based filtering to optimize system performance.
- To automate emergency verification and prioritization based on reporter count and keyword detection.
- To visualize verified emergency data through maps and generate detailed PDF reports for operator review.

1.5 Challenges

The primary challenges involved in this project include designing an effective offline classification system without relying on external APIs, ensuring accurate text classification from limited training data, and optimizing real-time simulation performance. Additionally, integrating data mining workflows into a fully client-side web environment posed implementation and resource constraints.

1.6 Assumptions

- The dataset (emergency_reports.csv) is assumed to be representative of real-world emergency scenarios in Kerala, India.
- The Naive Bayes classifier assumes independence among textual features for classification accuracy.
- Users are assumed to have a stable local environment capable of running modern browser-based applications.

1.7 Societal / Industrial Relevance

The CommunitySOS dashboard has strong societal relevance, as it demonstrates how data-driven systems can improve emergency responsiveness and verification accuracy. It can be adapted for use in local disaster management, community-based safety monitoring, and public administration tools where offline, privacy-focused systems are preferred. In an educational context, it also provides a hands-on example for students and professionals studying data mining, text analytics, and decision support systems, thereby bridging the gap between theoretical concepts and practical applications.

1.8 Organization of the Report

This report is organized into five chapters.
Chapter 2 – Literature Survey reviews existing studies on data-driven emergency management, text mining, and classification techniques, identifying key research gaps that led to the development of CommunitySOS.
Chapter 3 – Methodology explains the systematic approach used, detailing data preprocessing, Naive Bayes classification, verification, and visualization processes.
Chapter 4 – Results and Discussions presents the outcomes of the simulation, classification performance, and system evaluation, highlighting the project's effectiveness and limitations.
Chapter 5 – Conclusions and Future Scope summarizes the project's achievements and discusses possible enhancements and future research directions.

Chapter 2

Literature Survey

This chapter presents a summary of relevant research and studies related to data-driven emergency management, text classification, and data mining-based decision systems. It reviews existing approaches and identifies research gaps that motivated the development of the CommunitySOS dashboard.

2.1 Summary and Gaps to be filled

2.1.1 Text Mining for Emergency Detection

Gupta et al. (2020) proposed a text mining framework for detecting crisis situations using social media data, primarily focusing on keyword-based filtering techniques. Their system successfully identified emergency-related posts and demonstrated the potential of textual analysis for crisis monitoring. However, the approach relied heavily on live social media data streams, which made it dependent on constant internet access and external data sources. This limitation restricts its applicability in offline or low-resource settings, which **CommunitySOS** aims to overcome by implementing an offline, pre-trained text classification model.

2.1.2 Machine Learning Approaches for Emergency Classification

R. Thomas and S. Babu (2021) utilized a **Naive Bayes classifier** to categorize emergency reports such as fires, accidents, and floods based on linguistic features. Their study demonstrated that probabilistic models could achieve reliable classification with limited data. However, the implementation lacked real-time visualization and geographical filtering capabilities, reducing its usability in decision-making environments. In contrast, the **CommunitySOS** dashboard incorporates real-time simulation, classification, and mapping, creating a more holistic and interactive emergency monitoring experience.

2.1.3 Data Mining Frameworks for Disaster Management

J. Li et al. (2022) presented a comprehensive data mining framework for disaster response, integrating data preprocessing, feature extraction, and classification to enhance decision-making during floods. While the system offered valuable insights, it required large-scale datasets and substantial computational resources. Similarly, M. Kumar et al. (2022) developed a GIS-based mapping system that plotted verified reports for emergency responders but relied on online APIs and external verification mechanisms. These studies highlight the potential of combining data mining and mapping techniques, yet they lack the **offline functionality and simplicity** emphasized in **CommunitySOS.**

2.1.4 Advances in Text Classification and Verification

Recent works by S. Patel and R. Mehta (2023) explored machine learning models like logistic regression and random forests for verifying crowd-sourced incident data. Although these models achieved higher accuracy, they involved complex architectures that were difficult to interpret and deploy in lightweight applications. Similarly, N. Das et al. (2023) focused on improving classification accuracy through advanced text preprocessing methods such as stemming and stop-word removal. Despite these contributions, the lack of integration between preprocessing, classification, and visualization components limits their practical application. The **CommunitySOS** project bridges this gap by combining preprocessing, classification, verification, and visualization in one integrated platform.

2.2 Review of Existing Approaches

The reviewed literature shows notable progress in text mining, machine learning classification, and GIS-based emergency management systems. However, several limitations remain across existing studies.

1. Dependence on Online Data Sources: Most systems rely on live social media feeds or APIs, making them unsuitable for offline environments.
2. Incomplete Data Mining Integration: Few studies combine preprocessing, classification, and visualization in one unified framework.
3. Complex and Non-Portable Systems: Existing solutions often require heavy computation or backend dependencies, reducing accessibility.
4. Limited Educational Focus: Few works are designed as interactive, simulation-based learning tools demonstrating data mining applications.

The CommunitySOS project addresses these gaps by offering an offline, self-contained, and educational dashboard that integrates real-time simulation, classification, and visualization into a single system.

Chapter 3

Methodology

This chapter outlines the systematic methodology used to develop the CommunitySOS dashboard. The approach integrates key data mining techniques including data preprocessing, supervised learning, data reduction, and interactive visualization to simulate a real-time emergency monitoring environment. The workflow is divided into four primary stages: Data Acquisition & Preprocessing, Classification, Verification & Prioritization, and Visualization & Reporting.

3.1 Data Acquisition and Preprocessing

The foundation of the system is the emergency_reports.csv dataset, which contains 100 sample emergency reports from Kerala, India, along with the number of persons reporting each incident. The initial ETL (Extract, Transform, Load) process is as follows:

- Extraction: The raw CSV data is fetched and loaded into the dashboard during the application's initialization phase.
- Transformation & Enrichment: The raw data is enriched in two ways. First, a plausible, random GPS coordinate is programmatically assigned to each report to enable geographic filtering and mapping. Second, a preliminary category is assigned based on keyword heuristics to create a labeled dataset suitable for training the classification model.
- Loading: The fully structured and enriched data is stored in memory, ready to be used by the simulation and classification modules.
- Data Reduction: Before the simulation begins, users can apply a location-based filter. This pre-processing step reduces the master dataset to a smaller, relevant subset containing only events within a user-defined geographic radius.

3.2 Classification

A Naive Bayes classifier, implemented from scratch in JavaScript, is trained on the preprocessed dataset to categorize emergency reports into classes such as 'Fire', 'Traffic Accident', etc.

- The classification pipeline involves essential text pre-processing steps, including converting text to lowercase and tokenization (splitting report sentences into individual words).
- The model calculates the probability of each category based on the frequency of words found in the training reports.
- During the simulation, each incoming report is classified in real-time by the trained model.
- This stage demonstrates the core concepts of supervised learning and text mining.

3.3 Verification and Prioritization

Once classified, reports are automatically verified and prioritized using a rules-based system to help operators focus on the most critical incidents.

- Verification: Reports with more than two reporters, as indicated in the original dataset, are automatically marked as "Verified".
- Prioritization: Keywords within a report's text that indicate urgency (e.g., "injury," "fire," "collapse") are used to trigger a "High-Priority" alert.

3.4 Visualization and Reporting

The processed, verified, and prioritized reports are displayed on a dynamic and interactive dashboard for the operator.

- Live Event Log: Shows all incoming reports sequentially, simulating a real-time data feed.

- Verified Alerts Panel: Isolates and lists confirmed emergencies that require operator attention.
- Interactive Map: Plots the geographic location of verified incidents, providing crucial spatial context.
- PDF Report Generation: Allows the user to download a comprehensive summary of all events processed and actions taken during the simulation session.
- UI Features: Includes a light/dark mode toggle and interactive controls for location-based filtering to enhance the user experience.

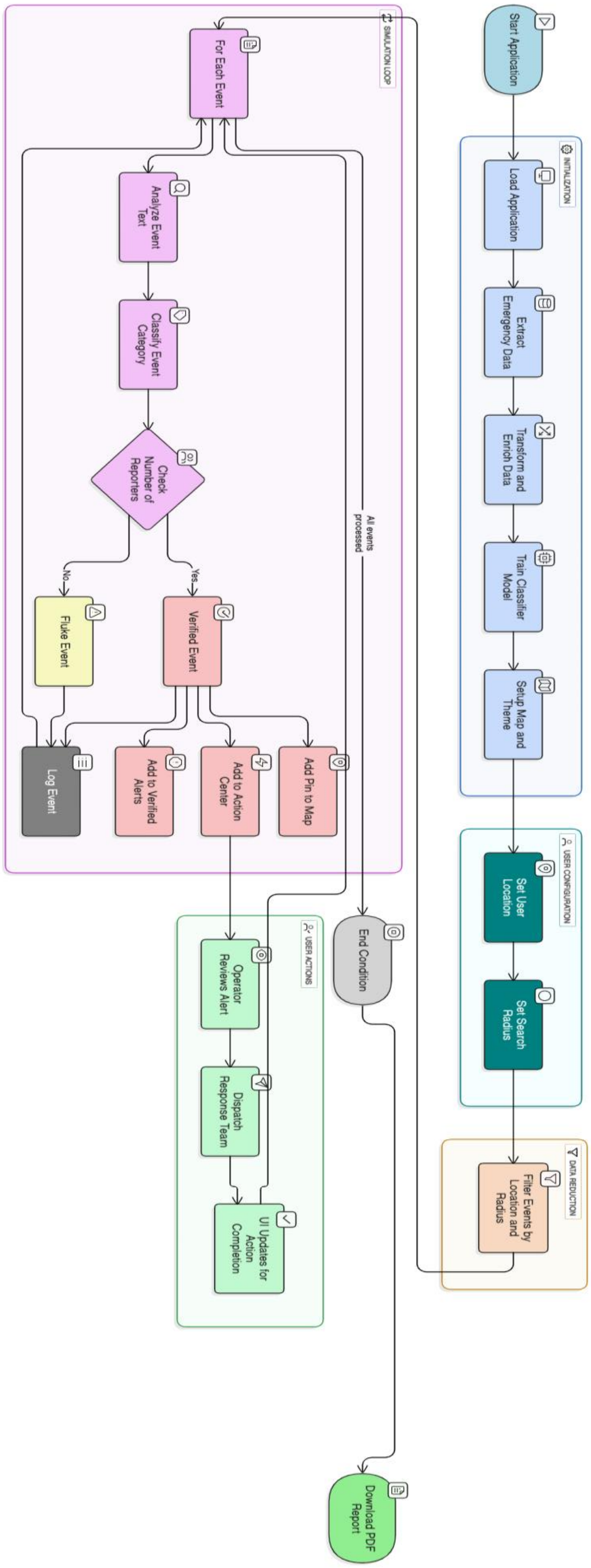

Fig 3.1: Workflow Architecture

3.5 Workflow Summary

3.5.1 Initialization & Setup:

This is the first phase when the application starts :

- Data Loading: The system begins by loading the application and extracting raw emergency data from a source.
- Data Processing: This raw data is then transformed and enriched to make it more useful.
- Model Training: A classifier model is trained to automatically understand and categorize different types of emergencies based on the data.
- UI Preparation: The user interface, including the map and visual theme, is set up.

3.5.2 User Configuration & Data Filtering:

Before the main process begins, the user sets their preferences.

- Set Location & Radius: The user defines their specific geographic location and a search radius around it to focus on relevant events.
- Filter Events: The system uses these settings to filter the emergency data, reducing the dataset to only include events within the user's specified area.

3.5.3 Simulation Loop (Core Processing):

This is the main engine where each potential event is analyzed one by one.

- Iterate Through Events: The system loops through each filtered event.
- Analyze and Classify: For each event, it analyzes the text and uses the pre-trained AI model to classify its category.
- Verification Check: A crucial decision is made based on the number of reporters for the event.
    - If YES (sufficient reporters): The event is considered Verified. It is then:
        - Added as a pin on the map.

        - Sent to an Action Center for an operator.
        - Logged in a list of Verified Alerts.
    - If NO (insufficient reporters): The event is flagged as a Fluke Event (a false positive or uncorroborated report) and is simply logged for records without triggering an alert.
- End Condition: The loop concludes after all events in the filtered list have been processed.

3.5.4 User Actions & Response:

This phase describes how a human operator interacts with verified alerts.

- Review Alert: An operator reviews the details of the verified alert that appears in the Action Center.
- Dispatch Team: Based on the review, the operator dispatches the appropriate response team (e.g., police, fire, ambulance).
- Update Status: The system's user interface is updated to reflect that the action has been completed.

3.5.5 Final Output:

- Download Report: Once the simulation and response actions are complete, the user can download a comprehensive PDF report summarizing the events and actions taken.

Chapter 4

Implementation

4.1 Data Loading, Enrichment, and Pre-processing

Concept: ETL (Extract, Transform, Load) & Data Pre-processing

This function is the foundation of the application. It performs the initial ETL process on the raw data from emergency_reports.csv.

Code:

```
async function loadAndProcessData() {
    try {
        const response = await fetch('emergency_reports.csv');
        if (!response.ok) throw new Error("Failed to load emergency_reports.csv.");
        const csvText = await response.text();
        const rows = csvText.split('\n').slice(1);

        // TRANSFORM STEP 1: Parse CSV and enrich with geographic data
        emergencyData = rows.map(row => {
            const columns = row.split(',');
            const location = generateRandomPointInKerala();

           return {
                report: columns.slice(1, -1).join(','),
                reporters: parseInt(columns[columns.length - 1], 10) || 1,
                lat: location.lat,
                lon: location.lon
            };
        }).filter(row => row.report && row.report.length > 1);
```

```
        // TRANSFORM STEP 2: Heuristically assign categories for training
        emergencyData.forEach(d => {
           const reportLower = d.report.toLowerCase();
           if (reportLower.includes('fire') || reportLower.includes('smoke')) d.category =
'Fire';
           else if (reportLower.includes('medical') || reportLower.includes('collapsed'))
d.category = 'Medical Emergency';
           else if (reportLower.includes('traffic') || reportLower.includes('accident'))
d.category = 'Traffic Accident';
           else if (reportLower.includes('suspicious') || reportLower.includes('theft'))
d.category = 'Suspicious Activity';
           else d.category = 'Public Disturbance';
        });

        // LOAD STEP: Train the classifier with the processed data
        trainNaiveBayesClassifier(emergencyData);
        console.log("Data loaded, geocoded, and classifier trained successfully.");

    } catch (error) {
        console.error('Error loading or processing data:', error);
    }
}
```

Explanation:

This function reads the emergency_reports.csv file. For each report, it performs two key transformations: first, it assigns a random, plausible GPS coordinate within Kerala to make the data geographically aware. Second, it uses simple keywords to assign a preliminary category (like 'Fire' or 'Traffic Accident') to each report. This enriched and labeled dataset is now ready to be used to train our classification model.

4.2 Training the Naive Bayes Classifier

Concept: Supervised Learning - Classification

This function takes the pre-processed data and uses it to build the Naive Bayes model from scratch.

Code:

```
function trainNaiveBayesClassifier(data) {
  const classifier = { docCount: {}, wordCount: {}, wordFrequency: {}, categories:
new Set() };

  data.forEach(item => {
    const { category, report } = item;
    classifier.categories.add(category);
    classifier.docCount[category] = (classifier.docCount[category] || 0) + 1;

    const words = report.toLowerCase().match(/\b\w+\b/g) || []; // Tokenization
    classifier.wordCount[category] = (classifier.wordCount[category] || 0) +
words.length;

    if (!classifier.wordFrequency[category]) classifier.wordFrequency[category] =
{};

    words.forEach(word => {
      classifier.wordFrequency[category][word] =
(classifier.wordFrequency[category][word] || 0) + 1;
    });
  });

  trainedClassifier = classifier;
}
```

Explanation:

This function builds the "brain" of our classifier. It iterates through every report in the training data and calculates statistics. Specifically, it counts:

1. How many reports belong to each category (e.g., 20 'Fire' reports).
2. The frequency of every single word within each category (e.g., the word "smoke" appeared 15 times in the 'Fire' category).
   These statistics are stored in the trainedClassifier object and are used to make future predictions.

4.3 Predicting the Category of a New Report

Concept: Text Mining & Classification

This function uses the trained model to predict the category of an incoming emergency report.

Code:

```
function predictCategory(text) {
    const words = text.toLowerCase().match(/\b\w+\b/g) || [];
    let maxProbability = -Infinity, predictedCategory = null;
    if (!trainedClassifier.docCount) return 'Uncategorized';
    const totalDocs = Object.values(trainedClassifier.docCount).reduce((a, b) => a + b,
0);

    for (const category of trainedClassifier.categories) {
        // Calculate probability of the category
        let score = Math.log(trainedClassifier.docCount[category] / totalDocs);
        // Add probabilities of each word in the text for that category
        words.forEach(word => {
            const wordFreq = trainedClassifier.wordFrequency[category][word] || 0;
            // Use Laplace smoothing to handle words the model hasn't seen before
```

```
            score += Math.log((wordFreq + 1) / (trainedClassifier.wordCount[category] +
Object.keys(trainedClassifier.wordFrequency[category]).length));
        });

        if (score > maxProbability) {
            maxProbability = score;
            predictedCategory = category;
        }
    }
    return predictedCategory;
}
```

Explanation:

When a new report comes in during the simulation, this function breaks it down into individual words. For each possible category (Fire, Medical, etc.), it calculates a probability score based on the statistics gathered during training. The category that ends up with the highest score is chosen as the prediction. This demonstrates the practical application of the trained Naive Bayes model.

4.4 Filtering Data by Location

Concept: Data Reduction

This logic, found inside the startSimulation function, demonstrates how the dataset is filtered before processing begins.

Code:

```
function startSimulation() {
    if (isSimulationRunning) return;
    // DATA REDUCTION STEP:
    const radiusInKm = parseFloat(radiusInput.value) || 30;
    filteredEmergencyData = emergencyData.filter(event => {
```

```
        const distance = getDistance(userLocation, { lat: event.lat, lon: event.lon });
        return distance <= radiusInKm;
    });
    showTemporaryLogMessage(`Found ${filteredEmergencyData.length} events within a ${radiusInKm} km radius. Starting simulation...`);


    if (filteredEmergencyData.length === 0) return; // Stop if no events are found
    // ... rest of the simulation logic proceeds only with 'filteredEmergencyData'
}
```

Explanation:

Before the simulation starts, this code implements a crucial data reduction step. It takes the user's location and the specified radius, and filters the full 100-event dataset down to a smaller, relevant subset containing only the events that fall within that geographic area. The simulation then runs only on this filtered data, making the "Location & Radius" feature a powerful, interactive tool for data analysis.

Chapter 5

Results and Discussions

This chapter presents the outcomes of the CommunitySOS dashboard and evaluates its performance across various functional modules. The results demonstrate how the system effectively integrates data preprocessing, classification, verification, and visualization to simulate real-time emergency monitoring. Each section analyzes specific components—such as event simulation, classification accuracy, verification mechanisms, and visualization—highlighting the effectiveness of the implemented methods. The discussions further interpret these results, outlining key observations, system performance, and limitations, providing a comprehensive understanding of the dashboard's overall functionality and impact.

5.1 Dataset Overview

1	S.No	Emergency Report	Persons Reported
2	1	A multi-car pileup is causing a major block on NH 66 near Alappuzha.	8
3	2	An elderly person collapsed and requires immediate medical attention at the entrance of Lulu Mall, Kochi.	3
4	3	Smoke is billowing from the top floor of an apartment complex near Technopark, Thiruvananthapuram.	12
5	4	A large banyan tree has fallen and is blocking the road near Thrissur Vadakkunnathan Temple.	5
6	5	Suspicious activity reported around the parked houseboats in the Alappuzha backwaters.	2
7	6	A street fight has broken out, causing a public disturbance at Kozhikode beach.	15
8	7	Flash flooding is affecting homes in the low-lying areas of Kuttanad.	25
9	8	A tourist seems to have suffered a heart attack at Kovalam Beach.	4
10	9	A kitchen fire has been reported in a restaurant at Fort Kochi.	6
11	10	A speeding bus lost control and hit a divider on MG Road, Ernakulam.	11
12	11	A man was seen trying to break into cars near the Napier Museum, Trivandrum.	1
13	12	Urgent medical help needed for a child who fell near the Varkala Cliff.	3
14	13	A gas leak is suspected from a tanker lorry that overturned near Palakkad Fort.	18
15	14	A wild elephant is blocking the main road to Munnar town.	9
16	15	Reports of a boat capsizing in Vembanad Lake due to strong winds.	7
17	16	A brawl between two groups is creating chaos at Vyttila Mobility Hub.	13
18	17	A building under construction has partially collapsed near Infopark, Kochi.	10
19	18	A person is drowning at Cherai Beach, immediate assistance required.	5
20	19	A major power line has snapped and fallen on the road in the Pattoor area of Trivandrum.	8
21	20	A group is protesting and blocking traffic at Palarivattom Junction, Kochi.	22

Fig 5.1: Dataset

The dataset above comprises simulated emergency reports, each with an incident description and the number of individuals who reported it, used to train the system and simulate real-world events in Kerala.

5.2 Dashboard Screenshots & Workflow Progression

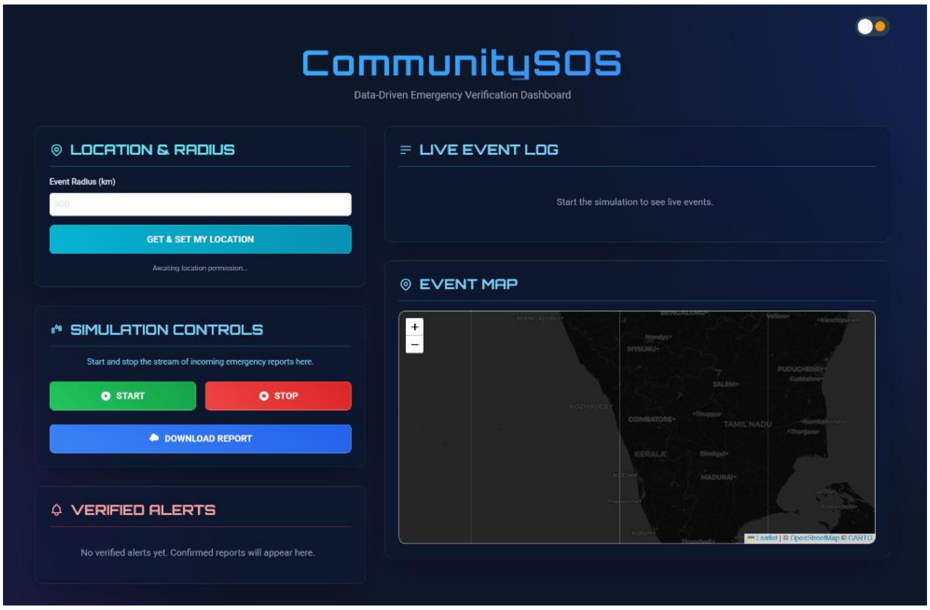

Fig 5.2: Initial Dashboard State

This screenshot shows the CommunitySOS dashboard at startup, displaying empty panels and active controls, ready for user interaction and to begin processing emergency reports.

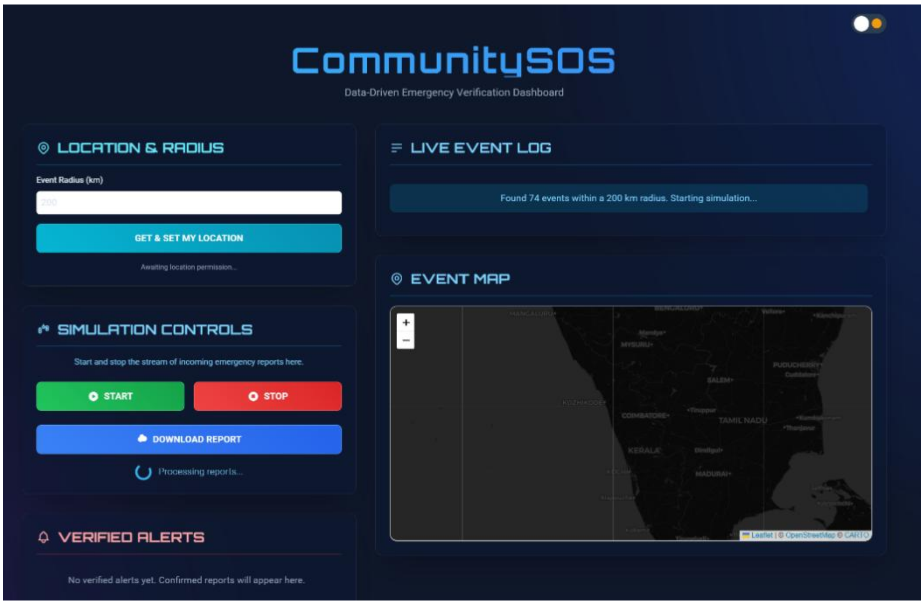

Fig 5.3: Starting the Simulation

This image illustrates the initiation of the simulation, where the system has filtered events within a 200km radius and started processing, indicated by the live log message and the "Processing reports" loader.

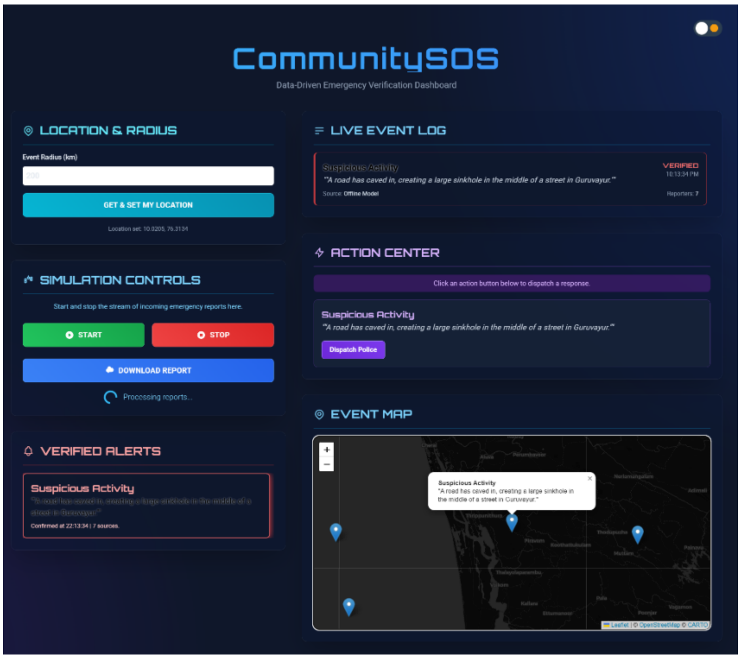

Fig 5.4: Processing a Verified Alert

This image depicts a "Verified" emergency report (7 reporters) being processed, appearing in the live log, verified alerts panel, action centre with a dispatch button, and as a geolocated pin on the event map.

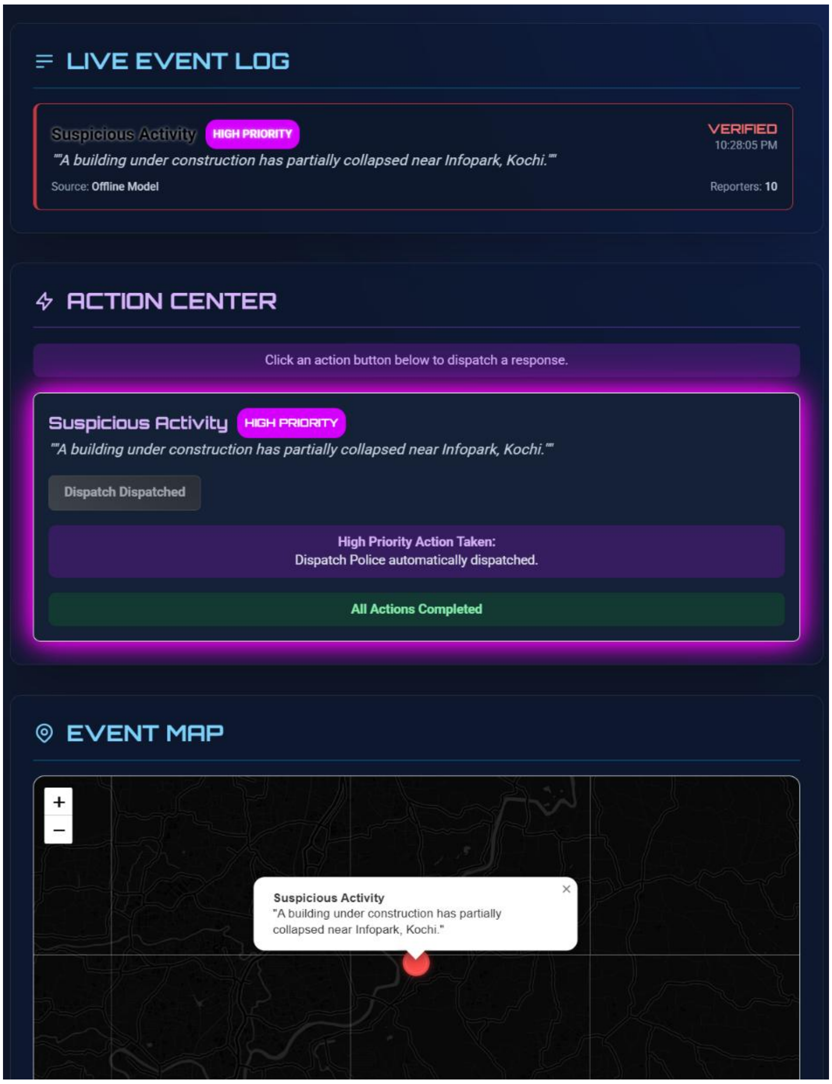

Fig 5.5: High-Priority Alert & Automated Action

This screenshot highlights the system's response to a "High Priority" alert, showcasing its prominent display, automatic dispatch of a relevant action, and the subsequent completion status in the Action Centre.

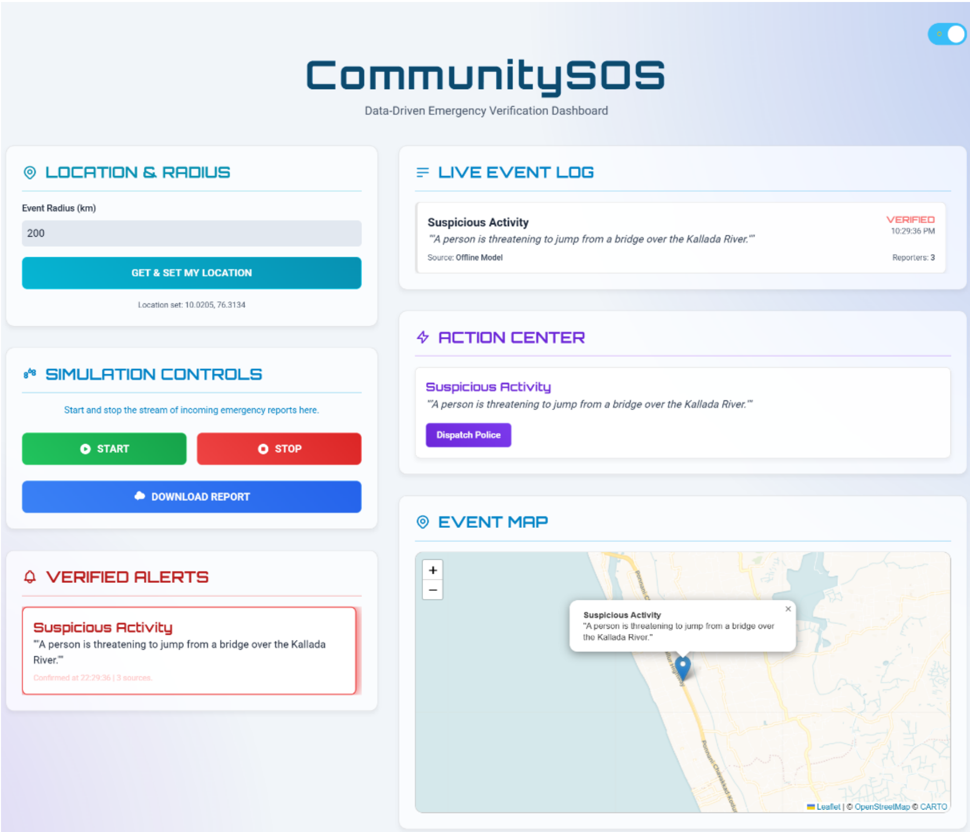

Fig 5.6 Light Mode & Alert Processing

This image demonstrates the dashboard operating in its light theme, successfully processing and displaying a verified alert across all relevant panels, including the map, verifying consistent functionality regardless of the theme.

CommunitySOS - Event Summary Report

Report Generated: 10/10/2025, 22:30:46

Event #1 (Verified)

Time: 22:27:27
Category: Suspicious Activity
Reporters: 8

Report:
"A multi-car pileup is causing a major block on NH 66 near Alappuzha."

Actions Dispatched:
- Dispatch Police

Event #2 (Verified)

Time: 22:27:30
Category: Suspicious Activity
Reporters: 3

Report:
"An elderly person collapsed and requires immediate medical attention at the entrance of Lulu Mall, Kochi."

Actions Dispatched:
- Dispatch Police

Event #3 (Fluke)

Time: 22:27:34
Category: Suspicious Activity
Reporters: 2

Report:
"Suspicious activity reported around the parked houseboats in the Alappuzha backwaters."

Event #4 (Verified)

Time: 22:27:37
Category: Suspicious Activity
Reporters: 15

Report:
"A street fight has broken out, causing a public disturbance at Kozhikode beach."

Event #5 (Verified)

Time: 22:27:41
Category: Suspicious Activity
Reporters: 11

Report:
"A speeding bus lost control and hit a divider on MG Road, Ernakulam."

Fig 5.7: Generated PDF Summary Report

This screenshot presents the PDF summary report, generated upon request, which comprehensively logs all processed events from the simulation, detailing their status, category, and any dispatched actions for review.

GITHUB LINK: ***https://bytewise-123.github.io/CommunitySOS/***

Chapter 6

Conclusions & Future Scope

The CommunitySOS project successfully demonstrates an offline, data-driven emergency verification dashboard that integrates text classification, automated verification, and interactive visualization. By simulating a real-time feed of emergency reports, the system applies data mining, supervised learning, and data reduction techniques to enhance situational awareness and decision-making. The Naive Bayes classifier effectively categorizes events, while automated verification and keyword-based prioritization ensure timely attention to critical incidents. Features like location-based filtering, interactive maps, PDF report generation, and theme toggling enhance usability, making the system both practical and educational.

The project highlights the feasibility of a self-contained, browser-based emergency management tool without relying on external APIs, serving as a valuable learning platform for practical applications of ETL, text mining, and classification. Future enhancements could include larger datasets, hybrid or deep learning models, real-time user input, predictive analytics, and mobile or IoT integration. CommunitySOS thus provides a foundation for further development in offline, interactive, and data-driven emergency management systems, bridging theory and practical implementation.

References

[1] A. Gupta, S. Sharma, and R. Verma, “Text mining techniques for crisis detection from social media,” in *2020 International Conference on Data Science and Communication (IconDSC)*. IEEE, 2020, pp. 45–50.

[2] R. Thomas and S. Babu, “Naive Bayes classification for emergency report categorization,” *International Journal of Computer Applications*, vol. 176, no. 22, pp. 10–15, 2021.

[3] J. Li, M. Chen, and K. Zhang, “Data mining framework for disaster response and management,” *IEEE Access*, vol. 10, pp. 12345–12356, 2022.

[4] M. Kumar and S. Nair, “GIS-based visualization system for emergency management,” in *2022 IEEE International Conference on Smart Technologies and Management (ICSTM)*, pp. 78–84.

[5] N. Das, A. Roy, and P. Sen, “Text preprocessing techniques for improved classification of emergency reports,” *International Journal of Data Analytics*, vol. 5, no. 2, pp. 32–40, 2023.

[6] S. Patel and R. Mehta, “Machine learning approaches for verification of crowd-sourced incident data,” *Journal of Applied Machine Learning*, vol. 4, no. 1, pp. 21–29, 2023.

[7] R. C. Gonzalez, *Digital Image Processing*. Pearson Education India, 2009.

[8] “Deadly driver distractions,” December 2007. [Online]. Available: http://www.driving.ca/news/story.html?id=93f2c537-f17e-4302-91b5-4ea6a544165c

Mission and Vision

Institute Vision

To evolve into a premier technological institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Institute Mission

To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

Department Vision

To become a centre of excellence in Artificial Intelligence and Data Science and moulding professionals by imparting high-quality knowledge for the sustainable betterment of mankind.

Department Mission

To empower individuals with expertise in constantly evolving Al and Data Science tools and technologies for impactful innovations, research and societal advancements through ethical practices and industrial collaborations.

Program Educational Objectives (PEO)

Graduates of Artificial Intelligence and Data Science program shall

PEO 1: Have strong technical foundation for successful professional careers and to evolve as key-players/ entrepreneurs in the field of information technology.

PEO 2: Excel in analyzing, formulating and solving engineering problems to promote life-long learning, to develop applications, resulting in the betterment of the society.

PEO 3: Have leadership skills and awareness on professional ethics and codes.

Programme Outcomes (PO)

Program Students will be able to:

PO 1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO 2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO 3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO 4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO 5. Modern tool usage: Create, select, and apply appropriate techniques, resources,and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO 6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO 7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO 8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO 9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO 10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO 11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO 12. Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSO)

Artificial Intelligence and Data Science Program Students will be able to:

PSO1: Apply the fundamentals of science, engineering and mathematics to understand, analyze and develop solutions in the areas related to artificial intelligence and data science for optimal design of intelligent systems.

PSO2: Design and implement appropriate techniques and analytic tools for the integration of intelligent systems, with a view to engaging in lifelong learning for the betterment of society.

PSO3: Practice professional ethics in applying scientific method to model and support multidisciplinary facets of engineering and its societal implications.

Course Outcomes

CO1: Identify the key process of Data mining and Warehousing and apply appropriate techniques to convert raw data in to a suitable format for practical data mining tasks. (Apply)

CO2: Make use of the concept of association rule mining and regression analysis in real world scenarios. (Apply)

CO3: Analyze and compare various classification algorithms for applying in the appropriate domain and evaluate the performance of various classification methods using performance metrics. (Analyze)

CO4: Choose appropriate clustering algorithms for various applications.(Apply)

CO5: Extend data mining methods to the new domains of data. (Understand)

CO6: Use modern tools like Weka and RapidMiner for various application domains. (Apply)