International Journal of Current Research and Techniques (IJCRT)
Volume 16, Issue 1, 2026; DOI: 10.5281/ZENODO.18576971
Original Research Article

AI- Driven Intrusion Detection System for Smart Agriculture in Remote and Disaster-Prone Areas

Kowshika D K, *PG Scholar, Department of Electronics and Communication Engineering, Government College of Engineering, Salem-11.*

Dr. D.Mary Sugantharathanam, *Professor and Head, Department of Electronics and Communication Engineering, Government College of Engineering, Salem-11.*

Manuscript Received: Jan 29, 2026; Revised: Feb 09, 2026; Published: Feb 10, 2026

Abstract: Smart agriculture integrates IoT devices, sensors and wireless communication to improve crop productivity and sustainable farming practices. In India, climate change has significantly affected major crops, resulting in unpredictable yields over recent decades. Accurate crop yield prediction before harvest is essential for effective planning and resource management. This work proposes an AI-driven Intrusion Detection System (IDS) for securing smart agriculture networks while enabling intelligent crop yield prediction. The system combines IoT-based agricultural data with machine learning techniques. A Random Forest algorithm is employed to predict crop yield with high accuracy. An interactive web-based platform is developed to provide a user-friendly interface for farmers and stakeholders. Standard datasets such as NSL-KDD are used to train and validate the IDS module. The IDS continuously monitors IoT network traffic to detect cyber threats. This ensures data integrity and reliable system operation. The integrated framework enhances network security and decision-making. It supports efficient resource allocation and farm planning. The proposed system improves overall agricultural productivity. It also strengthens cybersecurity in smart farming environments. The approach promotes sustainable and data-driven agriculture.

Keywords: Intrusion Detection System (IDS), Artificial Intelligence (AI), Machine Learning (ML), Internet of Things (IoT).

1.Introduction

Smart agriculture has gained significant attention as an effective solution to improve crop productivity, optimize resource utilization and promote sustainable farming practices through the use of Internet of Things (IoT) technologies[1]. In India, climate change and environmental variability over the past two decades have adversely affected the yield of major crops, creating uncertainty for farmers and policymakers[1][2][3]. Early and accurate crop yield prediction plays a vital role in supporting informed decisions related to resource management, storage and market planning.[4][5] However, the increasing reliance on IoT devices in agriculture exposes smart farming systems to cybersecurity threats, which can compromise data integrity and system reliability[5][6][7]. To address these challenges, this paper proposes an AI-driven Intrusion Detection System integrated with a crop yield prediction framework[7]. By employing machine learning techniques such as the Random Forest algorithm and utilizing standard datasets, the proposed system aims to enhance both agricultural productivity and network security in smart agriculture environments. An Intrusion Detection System (IDS) is a security mechanism that monitors network traffic and system activities to identify malicious behaviour and unauthorized access. In smart agriculture environments, where IoT devices and sensors are widely deployed, an IDS is essential for protecting sensitive agricultural data and ensuring uninterrupted system operation. The IDS analyses network patterns to detect cyber threats[7][8]. Modern IDS solutions utilize machine learning and artificial intelligence to distinguish normal and abnormal network behaviour and adapt to emerging threats[9]. By providing real-time alerts, the IDS enables timely response to security incidents [10]. Overall, integrating an IDS enhances the reliability, integrity and security of smart agriculture networks.

2.Need for Intrusion Detection System in Smart Agriculture

Smart agriculture relies heavily on Internet of Things (IoT) devices, wireless sensors, and cloud-based data systems to monitor and control various farming activities such as irrigation, soil moisture analysis, weather monitoring and crop

yield prediction. While these technologies enhance productivity and efficiency, they also expose agricultural networks to numerous cybersecurity threats. Cyberattacks such as Denial of Service (DoS), Distributed DoS (DDoS), data tampering and unauthorized access can disrupt communication between IoT devices, corrupt critical data and even cause system downtime, leading to severe economic losses and crop damage. An Intrusion Detection System (IDS) is therefore essential to monitor network traffic, detect malicious activities, and alert users in real time. It helps in maintaining data integrity, confidentiality and availability within smart farming systems. By integrating AI and machine learning techniques, an IDS can intelligently identify abnormal network behavior and adapt to new or evolving cyber threats. This proactive security mechanism ensures that smart agricultural operations remain uninterrupted, reliable and secure, especially in remote and disaster-prone regions where manual monitoring is difficult. Ultimately, implementing an IDS strengthens the overall resilience of smart farming infrastructures and safeguards the future of data-driven agriculture.

3.Implementation Methodology

The proposed work focuses on developing a machine learning-based system for crop prediction. The system's workflow begins with collecting a "Crop Dataset," which consists of relevant crop-related images or data. This raw data undergoes a pre-processing phase, where unnecessary noise is removed, and the dataset is enhanced for better accuracy in the subsequent steps. The pre-processing phase ensures that the data is clean and standardized, improving the model's performance.

3.1 Workflow of Crop Prediction System

Parallel to the crop dataset, the trained data, which has been previously processed and labelled, is also pre-processed. This trained data is essential for teaching the classifier how to make accurate predictions by learning from historical crop data. Both the crop dataset and trained data undergo "Feature Extraction," a process where relevant features such as colour, shape, texture, or spectral data are extracted to represent the data effectively. Feature extraction is crucial for reducing the complexity of the data while maintaining the most significant attributes for prediction.

Once the features are extracted, the data is passed to the "Classifier Algorithm." The classifier algorithm is a machine learning model, trained using the processed data. It uses the extracted features to classify the crop accurately. The classifier's role is to analyse the input data and predict the corresponding crop type based on patterns learned from the trained data.

In addition to generating crop predictions, the system evaluates the "Accuracy" of the classifier’s performance. This step involves comparing the predicted results with actual outcomes to measure the system's reliability and performance. The accuracy metric ensures that the classifier can make reliable predictions and provides feedback for improving the model.

The goal of this proposed work is to create an efficient, automated system for crop prediction that can aid farmers, agricultural researchers and crop management professionals in making data-driven decisions. By leveraging machine learning techniques, the system aims to accurately predict crops based on input data, improving agricultural productivity and planning. Furthermore, the system can be adapted to various environments and crop types, making it a versatile tool in the agricultural sector.

3.2 Block Diagram

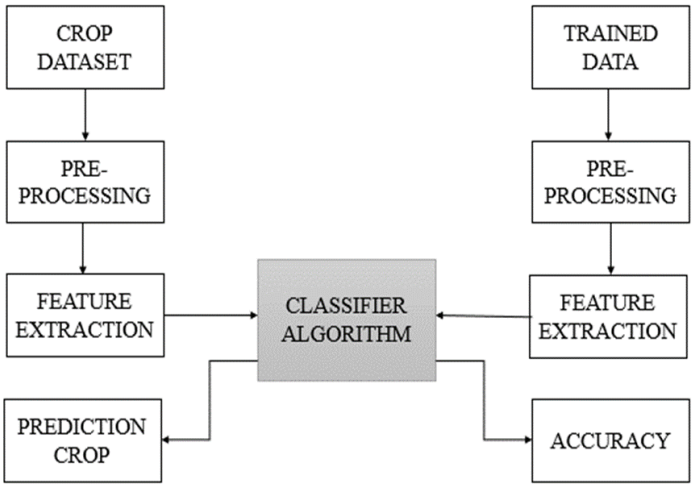

Fig 3.1: Block Diagram of Crop Prediction Model

3.3 Workflow

The block diagram Fig:3.1 demonstrates a systematic approach to intelligent crop prediction using machine learning. By integrating efficient data pre-processing, robust feature extraction and an accurate classifier, the proposed system enhances prediction accuracy, decision-making efficiency, and sustainable agricultural planning.

3.3.1 Data pre-processing

Data pre-processing is an essential stage in any data-driven project. It involves transforming the raw dataset into a structured, clean, and consistent format suitable for analysis and model training. The quality of data significantly affects the performance of machine learning models. In this study, several pre-processing steps were applied to the crop dataset to ensure high data quality, reliability, and readiness for feature extraction and classification. The steps are explained in detail below.

3.3.1.1 Importing Required Libraries

The Python libraries NumPy, Pandas, Matplotlib, Seaborn, and Plotly Express were imported to facilitate data handling, computation, and visualization. NumPy provides mathematical operations on arrays, while Pandas offers high-level data structures and tools for data manipulation. Matplotlib and Seaborn are used to generate static, publication-quality plots, and Plotly Express supports interactive visualizations. These libraries together provide a complete framework for data pre-processing and exploratory analysis, allowing for both numerical and graphical understanding of the dataset.

3.3.1.2 Loading the Dataset

The dataset was loaded into the Python environment using the Pandas function read.csv(). This command reads the comma-separated values file and converts it into a Data-Frame for easy data manipulation and inspection. The functions head () and shape were used to view the first few records and check the number of rows and columns respectively. This helps confirm that the dataset has been imported correctly without truncation or formatting errors. It also provides an initial understanding of the dataset's dimensions and structure, which is essential before proceeding to cleaning and analysis.

3.3.1.3 Handling Missing Values

The command df.isnull().sum() was used to check for missing or null values in each column. Missing data can occur due to human error during data entry or sensor malfunction during data collection. Such values can lead to inaccurate statistical measures and negatively impact model accuracy. Identifying and handling missing values ensures data

consistency and integrity. In this project, the output showed that there were no missing values, indicating that the dataset was complete and ready for further analysis without the need for imputation or record removal.

3.3.1.4 Descriptive Statistical Analysis

Descriptive statistics were computed using the df.describe () function, which provides information such as mean, standard deviation, minimum, maximum, and quartile values for each numerical feature. This step gives a general overview of the data distribution and central tendencies of attributes like Nitrogen, Phosphorus, Temperature, and Rainfall. It also helps to detect skewness and potential outliers that may affect model performance. Understanding these summary statistics aids in determining whether normalization, transformation, or scaling of features might be necessary in later stages.

3.3.1.5 Data Type and Structure Verification

The function df.info() was employed to examine the data types of each attribute and verify that all values were stored in appropriate formats. This function also reports the number of non-null entries, which is essential for detecting incomplete or incorrectly formatted columns. Correct data types ensure that numerical features can be used in mathematical operations and categorical features are processed properly during encoding. For instance, attributes like Nitrogen or Temperature should be numeric, whereas the "Label" column, representing crop type, should be of string or categorical type. Ensuring correct data types prevents computational errors and model mis-interpretation later.

3.3.1.6 Removing Duplicate Records

Duplicate entries in a dataset can introduce bias and affect the performance of predictive models. The command df.duplicated().sum() was used to check for any repeated rows. Duplicate records often occur due to repeated data collection or merging of multiple datasets without proper indexing. If present, they can cause overrepresentation of certain samples, leading to incorrect learning by the algorithm. In this dataset, the output confirmed that there were no duplicate entries, indicating that the dataset was unique and reliable. This verification step ensures the dataset's integrity and prevents redundancy.

3.3.1.7 Renaming Columns

Renaming columns improves readability and helps in maintaining consistency throughout the analysis. It provides clear, meaningful names for each feature, making it easier to interpret during visualization and modelling. This step also eliminates issues that may arise due to special characters or inconsistent naming conventions in the original dataset. Using uniform names ensures that the attributes can be accessed programmatically without confusion or errors during the model-building phase.

3.3.1.8 Exploratory Data Visualization

To understand the distribution and variability of each numerical feature, several visualizations were plotted using Seaborn and Matplotlib. Histograms with kernel density estimation (KDE) curves were used to visualize the frequency distribution and identify whether the data was normally distributed or skewed. Violin plots displayed both the distribution and density of the data, offering deeper insight into variability. Box plots were generated to detect the presence of outliers and visualize the median and quartile range. These visualizations help determine whether feature scaling or outlier removal is required before model training.

3.3.1.9 Grouping Data by Crop Type

The dataset was grouped based on the Label (crop type) column using the Pandas groupby() function, and the mean values of all numerical attributes were calculated for each crop. This grouping enables the comparison of average environmental and soil requirements for various crops. For instance, certain crops may require higher nitrogen levels, while others may thrive under specific temperature or rainfall conditions. This analysis provides valuable insights into the relationship between crop type and its environmental factors, which assists in developing a robust classification model.

3.3.1.10 Comparative Analysis Using Bar Plots

Finally, bar plots were generated to visually compare the mean values of all features for different crop categories. Each subplot represented one feature (e.g., Nitrogen, pH, or Rainfall) across all crop types. These visualizations provide a clear, comparative understanding of how different crops respond to varying soil nutrients and environmental parameters. Such comparisons are crucial in feature selection and in identifying which factors contribute most to crop differentiation. This step completes the exploratory phase and ensures that the data is well understood before feeding it into the classification algorithm.

4.Software Requiremets

4.1 Python Programming Language

Python is selected as the primary programming language for this project due to its simplicity, flexibility and strong support for data science and machine learning applications. It offers a rich set of libraries for data preprocessing, visualization, model development and performance evaluation. The clear syntax and open-source nature of Python make it well suited for academic and research use. Additionally, libraries such as Scikit-learn, TensorFlow and Keras enable efficient implementation of machine learning algorithms with reduced code complexity.

4.2 Jupyter Notebook

Jupyter Notebook is used as the development environment for this project due to its interactive and user-friendly nature. It enables step-by-step execution of Python code with immediate visualization of results. By combining code, outputs, and documentation in a single interface, Jupyter Notebook enhances workflow clarity and reproducibility.

4.2.1 Libraries and Packages Used

Several Python libraries were utilized to perform data handling, visualization and pre-processing operations efficiently. These include:

- NumPy: Used for performing numerical and mathematical operations on large datasets efficiently using multidimensional arrays.
- Pandas: A powerful data analysis library used for loading, cleaning, manipulating and organizing structured data into Data-Frames.
- Matplotlib: A plotting library used for creating static visualizations such as line charts, bar graphs, histograms, and scatter plots.
- Seaborn: Built on top of Matplotlib, Seaborn provides advanced statistical visualization tools and attractive default styles to represent data distributions and correlations.
- Plotly Express: Used to create interactive and dynamic visualizations for better data interpretation and presentation.

These libraries collectively provide a complete toolkit for handling the entire data science pipeline — from loading and pre-processing the dataset to building and evaluating machine learning models.

5. Results And Discussion

This presents a crop-wise analysis by grouping the dataset according to crop labels and computing the mean values of key soil and environmental features. This analysis highlights variations in nutrient availability and climatic conditions across different crops, providing essential insights that support accurate crop prediction and recommendation models.

5.1 Pre-Processing

5.1.1 Missing Value Detection

This step validates the integrity and reliability of the dataset, ensuring that the analytical results derived from it will be accurate and dependable.

```
In [4]: df.isnull().sum()

Out[4]: N              0
        P              0
        K              0
        temperature    0
        humidity       0
        ph             0
        rainfall       0
        label          0
        dtype: int64
```

Fig 5.1 Output for missing values

The output shows zero (0) missing values across all attributes such as N, P, K, Temperature etc., as shown in Fig 5.1. Since no data is missing, no imputation or correction is required, and the dataset can be directly used for further processing and analysis.

5.1.2 Descriptive Statistical Summary

This step provides an essential overview of dataset characteristics, ensuring that further data analysis and modelling steps are based on a clear understanding of value distribution and variation

```
In [5]: df.describe()
```

Out[5]:

	N	P	K	temperature	humidity	ph	rainfall
count	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000
mean	50.551818	53.362727	48.149091	25.616244	71.481779	6.469480	103.463655
std	36.917334	32.985883	50.647931	5.063749	22.263812	0.773938	54.958389
min	0.000000	5.000000	5.000000	8.825675	14.258040	3.504752	20.211267
25%	21.000000	28.000000	20.000000	22.769375	60.261953	5.971693	64.551686
50%	37.000000	51.000000	32.000000	25.598693	80.473146	6.425045	94.867624
75%	84.250000	68.000000	49.000000	28.561654	89.948771	6.923643	124.267508
max	140.000000	145.000000	205.000000	43.675493	99.981876	9.935091	298.560117

There are outliers present in the dataset

Fig 5.2 Output of Descriptive Statistical Summary

A large difference between the minimum and maximum values for attributes like N, P and K suggests the presence of outliers or extreme values in the dataset as shown in the Fig 6.2. The statistical summary aids in understanding data spread and identifying anomalies.

5.1.3 Duplicate Check

```
In [7]: # To check for duplicates
        df.duplicated().sum()
        #No duplicates present

Out[7]: 0
```

Fig 5.3 Output of Overall Interpretation

The output obtained is "0", which indicates that there are no duplicate entries present in the dataset as shown in the Fig 6.3.

5.1.4 Renaming Columns

Renaming columns is an important data pre-processing step to make the dataset more readable, organized and meaningful.

```
In [8]: #Renaming columns
        df.columns = ['Nitrogen','Phosphorus','Potassium','Temperature','Humidity','pH','Rainfall','Label']
```

Fig 5.4 Output of Renaming Columns

As shown in Fig 6.5 command df.columns = [...] is used to assign new names to all column headers in the dataset at once. Renaming the columns helps make each feature name **self-explanatory**, thereby improving the dataset's clarity.

5.1.5 Exploratory Data Visualization

The Exploratory Data Visualization which helps in understanding the underlying patterns, trends, and relationships within the dataset.

```
In [9]: plt.style.use('dark_background')
        sns.set_palette("Set2")
        for i in df.columns[:-1]:
            fig,ax=plt.subplots(1,3,figsize=(18,4))
            sns.histplot(data=df,x=i,kde=True,bins=20,ax=ax[0])
            sns.violinplot(data=df,x=i,ax=ax[1])
            sns.boxplot(data=df,x=i,ax=ax[2])
            plt.suptitle(f'Visualizing {i}',size=20)
```

Fig 5.5 Code for Exploratory Data Visualization

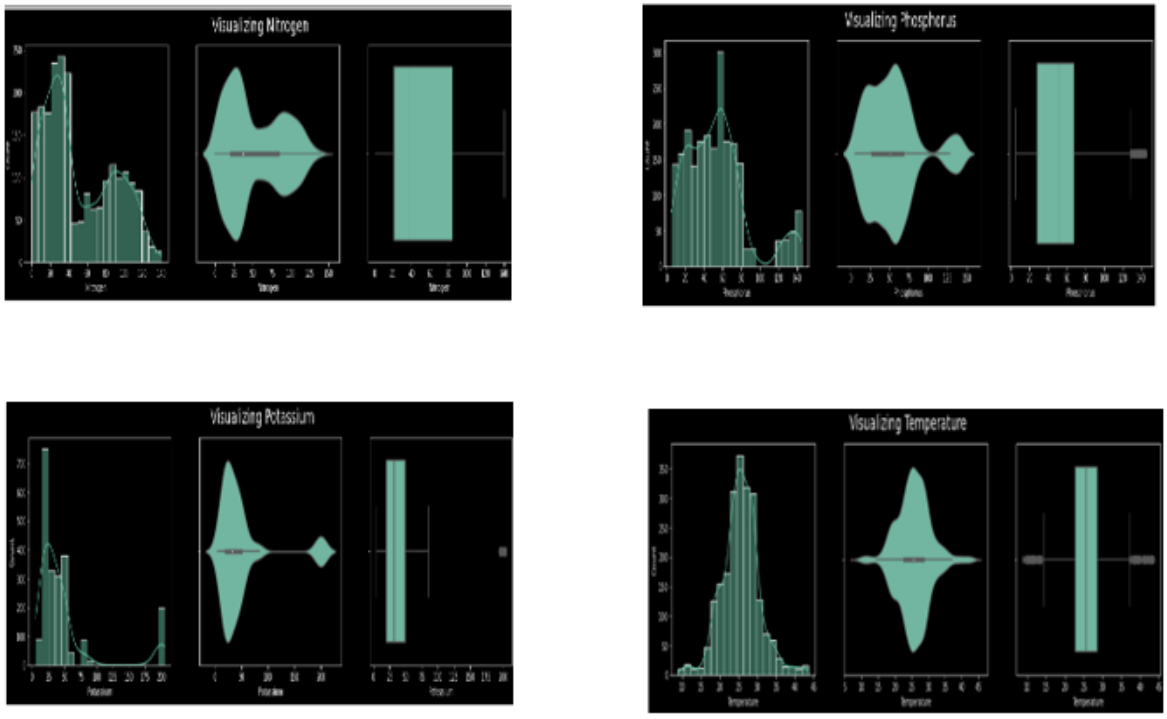

Fig 5.6 Visualization of Nitrogen Distribution across Crops

Histogram with KDE, Violin Plot and Box Plot are used to visualize feature distributions, central tendency, variability and skewness as shown in Fig 5.6. Together, these plots help identify data patterns and outliers, supporting informed pre-processing decisions such as normalization and feature scaling for improved model performance.

5.1.6 Grouping Data by Crop Type

Grouping helps in analysing trends and relationships between crops and their associated soil or environmental parameters.

```
In [11]: grouped
Out[11]:
```

	Label	Nitrogen	Phosphorus	Potassium	Temperature	Humidity	pH	Rainfall
0	apple	20.80	134.22	199.89	22.630942	92.333383	5.929663	112.654779
1	banana	100.23	82.01	50.05	27.376798	80.358123	5.983893	104.626980
2	blackgram	40.02	67.47	19.24	29.973340	65.118426	7.133952	67.884151
3	chickpea	40.09	67.79	79.92	18.872847	16.860439	7.336957	80.058977
4	coconut	21.98	16.93	30.59	27.409892	94.844272	5.976562	175.686646
5	coffee	101.20	28.74	29.94	25.540477	58.869846	6.790308	158.066295
6	cotton	117.77	46.24	19.56	23.988958	79.843474	6.912675	80.398043
7	grapes	23.18	132.53	200.11	23.849575	81.875228	6.025937	69.611829
8	jute	78.40	46.86	39.99	24.958376	79.639864	6.732778	174.792798
9	kidneybeans	20.75	67.54	20.05	20.115085	21.605357	5.749411	105.919778
10	lentil	18.77	68.36	19.41	24.509052	64.804785	6.927932	45.680454
11	maize	77.76	48.44	19.79	22.389204	65.092249	6.245190	84.766988
12	mango	20.07	27.18	29.92	31.208770	50.156573	5.766373	94.704515
13	mothbeans	21.44	48.01	20.23	28.194920	53.160418	6.831174	51.198487
14	mungbean	20.99	47.28	19.87	28.525775	85.499975	6.723957	48.403601
15	muskmelon	100.32	17.72	50.08	28.663066	92.342802	6.358805	24.689952
16	orange	19.58	16.55	10.01	22.765725	92.170209	7.016957	110.474969
17	papaya	49.88	59.05	50.04	33.723859	92.403388	6.741442	142.627839

Fig 5.7 Output of Grouping Data by Crop Type

The grouping process is performed to organize the dataset based on a common feature here, the Label column, which indicates the type of crop is shown in Fig 5.7

5.1.7 Comparative Analysis of Environmental Parameters for Different Crops

The comparative analysis of environmental parameters for different crops evaluates how factors like temperature, rainfall and humidity influence crop growth.

```
In [12]: fig,ax=plt.subplots(7,1,figsize=(25,25))
         for index,i in enumerate(grouped.columns[1:]):
             sns.barplot(data=grouped,x='Label',y=i,ax=ax[index])
             plt.suptitle("Comparision of Mean Attributes of various classes",size=25)
             plt.xlabel("")
```

Fig 5.8 Code for Comparative Analysis

This analysis supports precision agriculture, helping farmers choose crops best suited to their soil and climate. The summarized data also aids in crop prediction and recommendation systems for sustainable farming.

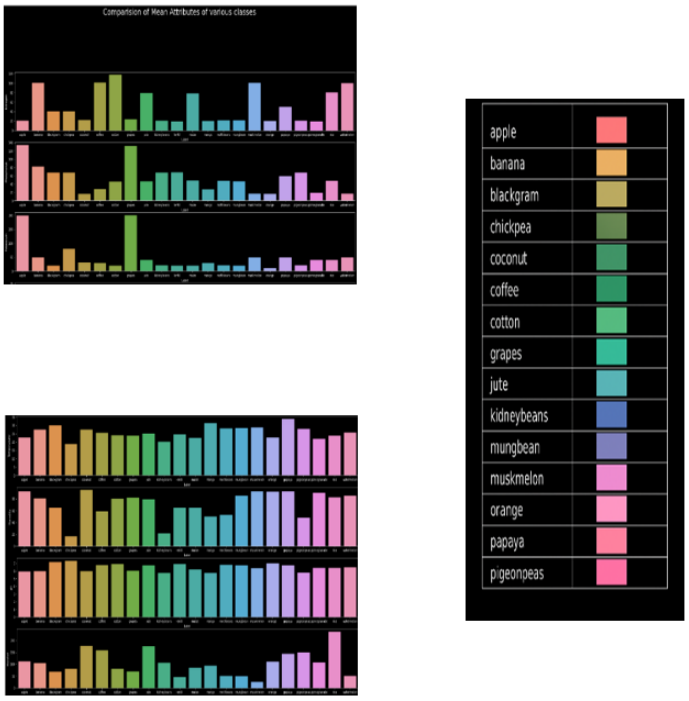

Fig 5.9 Comparison of Mean Soil Attributes for Different Crop Classes

Fig. 5.9 illustrates how average soil parameters vary among various crop types. The Fig 5.9 shows the mean soil and environmental parameters (Nitrogen, Phosphorus, Potassium, Temperature, Humidity, pH and Rainfall) across different crops.

6. Conclusion

The proposed system provides a comprehensive analysis of the relationship between soil nutrients and environmental conditions across different crops. Data preparation steps, including duplicate removal, column standardization and crop-wise grouping, ensured a clean and reliable dataset for analysis. Exploratory visualizations such as histograms, violin plots, and box plots facilitated an in-depth understanding of feature distributions, variability and outliers in key attributes such as Nitrogen, Phosphorus, pH, temperature and rainfall. Further, crop-wise grouping and mean-based comparative analysis revealed distinct nutrient and climatic preferences of individual crops, highlighting their specific growth requirements. These insights enabled accurate crop recommendation by identifying crops best suited to given soil and environmental conditions, thereby supporting precision agriculture, effective soil management, and sustainable yield optimization.

7. Future Work

In the future, this research can be extended by incorporating advanced feature extraction techniques to determine the most influential environmental and soil parameters that directly impact crop productivity. This will help in identifying the critical factors that govern crop growth and yield. Additionally, the implementation of Random Forest or other ensemble learning algorithms can be explored to enhance the accuracy, robustness, and generalization

capability of the predictive model. These algorithms are well-suited for handling complex, nonlinear relationships between features and crop types. Furthermore, the system can be expanded to include real-time agricultural data acquisition, enabling dynamic crop recommendation systems that can adapt to changing climatic conditions and soil properties. Integrating IoT-based sensors and cloud data processing would allow continuous monitoring and intelligent decision-making, ultimately supporting precision agriculture and sustainable farming practices.

8. References

[1] Jayveersinh Vansiya,Asha Chandi and Rashid A Khan (2025) "AI-Based Intrusion Detection & Prevention Models for Smart Home IoT Systems: A Literature Review", JCSTS- 982-996. (2025)

[2] Joel Paul "Integration of AI and ML in Intrusion Detection and Prevention Systems" ResearchGate (2025)

[3] Rafael Ferreira, Ivo Bispo, Carlos Rabadao, Leonel Santos, Rogerio Luís de C. Costa "Farm-flow dataset: Intrusion detection in smart agriculture based on network flows" ELSEVIER (2025)

[4] Abhishek Raghuvanshi, Harikumar Pallathadka,Umesh and Khongdet Phasinam "Intrusion Detection Using Machine Learning for Risk Mitigation in IoT-Enabled Smart Irrigation in Smart Farming" IEEE (2022)

[5] K.Subrahmanyam , Bandili Pavani, Meejuru Kishore, Malli Kumar, Koduru Suresh "AI-Driven Animal Intrusion Detection System for Agriculture: Real-Time Monitoring and Automated Response Using Deep Learning" IJIRT (2025)

[6] Gandhi,L.J.Armstrong,O.Petkar,andA.K.Tripathy,''Ricecropyield prediction in India using support vector machines,'' in Proc. 13th Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE), Jul. 2016, pp. 1–5.

[7] M.KhanandS.Noor, ''Irrigation runoff volume prediction using machine learning algorithms,'' Eur. Int. J. Sci. Technol., vol. 8, pp. 1–22, Jan. 2019.

[8] NIC IN. Annual Report 2020–21. Accessed: Jan. 20, 2023. [Online]. Available: https://agricoop.nic.in/Documents/annual-report 2020-21 (2023).

[9] Govt India. Crop Production Statistics Information System. [Online]. Available: https://aps.dac.gov.in/APY/Index.htm (2023).

[10] M. Khan and S. Noor, ''Performance analysis of regression-machine learning algorithms for predication of runoff time,'' Agrotechnology, vol. 8, no. 1, pp. 1–12, 2019.

[11] IBEF. Agriculture in India: Industry Overview, Market Size, Role in Development. Accessed: Jan. 20, 2023. India. [Online]. Available: https://www.india.gov.in/ (2023).

[12] Abdinasir H A, Lukman A, Adeb S, A, M. A., Rasheed, H., Ahmed, S., & Tahir, A. Security Control and Data Planes of SDN: A Comprehensive Review of Traditional, AI and MTD Approaches to Security Solutions. IEEE Access, 1–1. https://doi.org/10.1109/access.2024.3393548 (2024).

[13] Akmalbek A, Dusmurod K, Rashid N, Ilkhom R, & Cho, Y. I. Optimizing Smart Home Intrusion Detection with Harmony-Enhanced Extra Trees. IEEE Access, 1–1. https://doi.org/10.1109/access.2024.3422999 (2024).

[14] Albasir, A., Naik, K., & Manzano, R Towards Improving the Security of IoT and CPS Devices: An AI Approach. Digital Threats: Research and Practice. https://doi.org/10.1145/3497862 . (2022).

[15] Almaraz-Rivera, J. G., Perez-Diaz, J. A., Cantoral-Ceballos, J. A., Botero, J. F., & Trejo, L. A. Toward the Protection of IoT Networks: Introducing the LATAM-DDoS-IoT Dataset. IEEE Access, 10, 106909–106920. https://doi.org/10.1109/access.2022.3211513 (2022).

[16] Alrayes, F. S., Wafa M, Aljameel, S. S., Mashael M, Rizwanullah, M., & Salama, A. S. Improved Radial Movement Optimization With Fuzzy Neural Network Enabled Anomaly Detection for IoT Assisted Smart Cities. IEEE Access, 11, 143060–143068. https://doi.org/10.1109/access.2023.3342698 (2023).