

Smart Soil-Based Crop Recommendation System Using Real-Time Soil and Climate Data with Machine Learning

Sudharsan S

Master of Computer Applications, Nehru Institute of Information Technology and Management, Tamil Nadu, India

Reg. No: 721524622053 | Batch: 2024–2026 –

MENTOR –Mrs. N. Anandha Priya MCA, Asst Professor [Department of computer Applications]

ABSTRACT

In contemporary precision agriculture, selecting the optimal crop for a given piece of land requires detailed knowledge of soil nutrient composition and local climate conditions—data that is frequently unavailable to smallholder farmers without expensive laboratory testing or IoT hardware. This paper presents a **Smart Soil-Based Crop Recommendation System**, an intelligent web-based application developed using Python and Flask that automatically detects the user's GPS location, retrieves real-time soil nutrient data (Nitrogen, Phosphorus, Potassium, and pH) from the Mannvalam Geoportal API maintained by the Tamil Nadu government, and fetches live and historical climate data (temperature, humidity, and rainfall) from Open-Meteo APIs. These inputs are analyzed by a trained Random Forest classification model to predict the top 5 crops most suitable for cultivation from a catalogue of 56 crop types grown across Tamil Nadu. A fertilizer recommendation module compares detected nutrient levels against crop-specific optimal ranges and provides actionable fertilizer product guidance. A Tamil language voice output module powered by Google Text-to-Speech and Google Translate ensures accessibility for farmers with limited digital literacy. The system achieves a model accuracy of 93.2%, eliminates the need for physical sensors or manual data entry, and provides real-time, location-specific recommendations at zero hardware cost. Results validate that integrating government-maintained API data with ensemble machine learning delivers practical, scalable, and inclusive agricultural decision support.

Index Terms—Machine learning, Random Forest, crop recommendation, soil nutrients, real-time API, Flask, precision agriculture, Tamil Nadu, voice output, fertilizer planning, Mannvalam Geoportal, Open-Meteo, GPS, gTTS, Google Translate.

I. INTRODUCTION

Agriculture constitutes the backbone of rural livelihoods across India, with Tamil Nadu alone supporting over 6 million farming households. Despite rapid advances in agricultural science, the majority of small and marginal farmers continue to make crop selection decisions based on personal experience, regional tradition, and generational knowledge rather than objective, data-driven analysis [1]. This reliance on informal methods is increasingly inadequate in the context of climate change, shifting rainfall patterns, and variable soil health, often resulting in suboptimal crop choices, reduced yields, and significant economic loss [2].

Machine learning has emerged as a transformative technology for agricultural decision support. Supervised classification models trained on soil and climate parameters can identify complex, non-linear relationships between input features and crop suitability with high accuracy [3]. Research has consistently demonstrated that ensemble methods, particularly Random Forest, achieve superior predictive performance on multi-class crop recommendation tasks compared to single classifiers such as Decision Tree, Naive Bayes, and Support Vector Machines [4][5].

However, most existing crop recommendation systems suffer from a critical limitation: they depend on static historical datasets sourced from public repositories such as Kaggle, which do not reflect the real, current conditions of a specific farm location. Systems that do collect real-time data typically require expensive IoT sensors, Raspberry Pi hardware, and continuous network connectivity—barriers that make them impractical for the majority of rural farmers [6]. Furthermore, existing systems rarely support local language output or voice narration, limiting their accessibility for users with limited formal education or digital literacy [7].

This paper presents a Smart Soil-Based Crop Recommendation System that addresses all of these limitations through four integrated innovations: (1) automatic GPS-based location detection that identifies the user's assembly constituency and district without manual input; (2) real-time soil nutrient data retrieval from the Mannvalam Geoportal, a government-maintained API providing district-level N, P, K, and pH data for Tamil Nadu; (3) live and historical climate data from Open-Meteo APIs, covering temperature, humidity, and rainfall; and (4) a trained Random Forest classifier that predicts crop suitability scores across 56 crop types and ranks recommendations with a traditional crop boost mechanism that integrates regional agricultural knowledge with data-driven prediction.

The system further includes a fertilizer planning module that computes nutrient deficiencies and excesses relative to crop-specific optimal ranges and recommends specific fertilizer products with dosage guidance. Tamil voice output generated through Google Text-to-Speech and Google Translate ensures that recommendations are accessible to farmers who cannot read digital text. The entire system is implemented in Python and Flask and requires no physical hardware beyond a GPS-enabled smartphone, making it immediately deployable across Tamil Nadu.

II. RELATED WORK

A. Predictive Analytics in Precision Agriculture

The application of machine learning to agricultural decision support has attracted substantial research attention. Sharafat et al. [1] demonstrated a real-time crop prediction system using sensor data and weather APIs with edge computing, establishing that API-sourced climate data can substitute for hardware sensors in prediction pipelines. Dey et al. [2] applied supervised machine learning to soil and climate data for crop recommendation, achieving accuracy above 90% using Random Forest on multi-class classification tasks.

Bhargavi and Srinivasan [4] conducted a comparative study of machine learning algorithms for crop recommendation, finding that Random Forest consistently outperformed Decision Tree, SVM, and Logistic Regression across accuracy, precision, recall, and F1-score metrics. Jaya Deepthi et al. [5] further validated Random Forest superiority in a smart agriculture context, combining crop recommendation with yield prediction and reporting a classification accuracy of 94.7%.

B. IoT and Sensor-Based Systems

Sawant et al. [12] developed an IoT-driven machine learning system for crop recommendation that combines physical soil sensors with a cloud-based Random Forest model. While this approach achieves high accuracy by using direct field measurements, it requires significant hardware investment—Raspberry Pi boards, soil probes, and network infrastructure—that is inaccessible to most smallholder farmers. The authors acknowledge that API-based data sourcing represents a viable lower-cost alternative for deployment at scale [12].

Kumar et al. [11] investigated the effect of geographic location features on crop recommendation accuracy, finding that location-aware systems outperform generic models by up to 8.3% in F1-score. Their work directly motivates the GPS-based, location-specific architecture adopted in the proposed system.

C. Fertilizer and Soil Management Systems

Afzal et al. [10] published a comprehensive study integrating soil information with machine learning for crop recommendation and fertilizer planning, reporting that nutrient deficiency analysis improves farmer acceptance of recommendations by providing actionable soil preparation guidance. Their findings support the design of the fertilizer recommendation module in the proposed system.

Nair et al. [9] developed a smart crop selection decision support system using machine learning, incorporating fertilizer guidance based on N-P-K comparisons. Their work demonstrated that combined crop and fertilizer recommendations are significantly more useful to farmers than crop prediction alone, an insight that directly influenced the dual-output design of the proposed system.

D. Language Accessibility and Voice Output

A consistent finding across agricultural technology adoption studies is that language and literacy barriers significantly impede uptake of digital decision support tools among rural farming communities [7][8]. Gupta and Lekhana [8] specifically identified the absence of local language support as a primary barrier to adoption in their study of crop

recommender systems in South India. The integration of Tamil voice output via gTTS and Google Translate in the proposed system directly addresses this identified gap.

III. SYSTEM ARCHITECTURE

A. Five-Layer Web Architecture

The proposed system is implemented as a five-layer web application deployed on a local Flask server, with all computation performed server-side to ensure compatibility with low-end client devices.

The Presentation Layer consists of a browser-based HTML/CSS frontend that captures the user's GPS coordinates via the browser's Geolocation API and renders the recommendation results in a farmer-friendly card layout with color-coded nutrient status indicators.

The API Integration Layer handles asynchronous communication with three external data sources: (1) the Mannvalam Geoportal API (Mann1 endpoint for crop history, Mann3 endpoint for soil nutrients); (2) the Open-Meteo Forecast API for live weather; and (3) the Open-Meteo Archive API for 12-month historical climate averages. All API calls implement timeout handling and fallback default values to ensure graceful degradation when connectivity is limited.

The Data Processing Layer preprocesses raw API responses into normalized feature vectors. Soil nutrient ratios are computed from proportional soil category data (high, medium, low counts per nutrient), and climate aggregates are derived from historical monthly records. A location matching module maps GPS coordinates to the nearest assembly constituency using geodesic distance calculation against a pre-loaded constituency database.

The Machine Learning Layer applies the trained Random Forest classifier to the processed feature vector to generate probability scores across all 56 crop classes, applies the traditional crop boost mechanism, normalizes the probability distribution, and selects the top 5 recommendations.

The Output Generation Layer compiles recommendation results, fertilizer plans, and voice narration. The gTTS module generates a Tamil audio file from a Google Translate-translated summary string, which is served to the client as a playable audio element.

B. End-to-End System Workflow

The system workflow proceeds through eight stages: (1) The user opens the web application and grants GPS location permission; (2) the frontend sends latitude and longitude to the Flask backend via the /recommend endpoint; (3) the backend identifies the nearest assembly constituency and district; (4) the Mann1 and Mann3 Mannvalam APIs are queried for the detected district, returning soil nutrient distribution data and traditional crop history; (5) the Open-Meteo Forecast API returns live weather conditions and the Archive API returns 12-month historical climate statistics; (6) feature extraction computes N, P, K, pH, temperature, humidity, and monthly rainfall values; (7) the Random Forest model predicts crop suitability probabilities, the boost mechanism is applied, and top 5 crops are selected; (8) fertilizer plans are generated for each recommended crop and Tamil voice output is synthesized and returned to the client.

IV. DATASET DESCRIPTION

The system uses a custom training dataset of 280 labeled records covering 56 crop types cultivated in Tamil Nadu. Each record contains seven input features: Nitrogen (N), Phosphorus (P), Potassium (K), temperature, humidity, pH, and monthly rainfall. Feature ranges are calibrated to Tamil Nadu's agro-climatic conditions rather than generic national averages, improving model relevance for the target deployment region.

TABLE I. DATASET FEATURE SUMMARY

Feature	Min	Max	Mean	Unit
Nitrogen (N)	10	130	58.4	kg/ha
Phosphorus (P)	10	130	53.6	kg/ha
Potassium (K)	10	170	60.2	kg/ha
Temperature	15.0	43.7	28.3	°C

Humidity	14.3	99.9	63.8	%
pH	3.5	9.9	6.47	—
Rainfall	20.2	298.6	103.4	mm/month

The dataset is constructed from 5 samples per crop type, representing typical growing conditions across major agro-climatic zones of Tamil Nadu. For crops with wider growing condition ranges, additional samples were included to capture variability. The dataset was split into 80% training (224 records) and 20% testing (56 records) with stratified sampling to maintain class distribution balance.

V. MACHINE LEARNING MODEL: RANDOM FOREST

A. Algorithm Selection Rationale

Random Forest is selected as the prediction model based on its demonstrated superiority for multi-class classification on tabular agricultural data [4][5]. Its ensemble architecture—aggregating predictions across multiple decorrelated decision trees—provides robustness against overfitting on small-to-medium datasets, tolerance of feature correlation (as exists between temperature and humidity), and built-in feature importance estimation that supports model interpretability.

B. Mathematical Formulation

A Random Forest classifier $F(x)$ is constructed from an ensemble of T decision trees $\{h_1(x), h_2(x), \dots, h_T(x)\}$, where each tree $h_i(x)$ is trained on a bootstrap sample of the training data using a random subset of features at each split node. For a given feature vector $x = [N, P, K, T, H, \text{pH}, R]$, the predicted crop class C^* is determined by majority vote:

$$C^* = \underset{c}{\operatorname{argmax}} \sum_{i=1}^T \mathbb{1}[h_i(x) = c]$$

where $\mathbb{1}[\cdot]$ is the indicator function and c ranges over all 56 crop classes. For crop probability estimation, the predicted probability for class c is:

$$P(C = c | x) = (1/T) \sum_{i=1}^T P_i(C = c | x)$$

where $P_i(C = c | x)$ is the fraction of training samples of class c in the leaf node reached by x in tree i . The Gini impurity criterion is used for node splitting:

$$\text{Gini}(t) = 1 - \sum_c p(c|t)^2$$

The traditional crop boost mechanism modifies the raw probability distribution by applying a boost factor $\beta = 0.35$ to crops known to be traditionally cultivated in the detected district:

$$P'(c) = P(c) + (1 - P(c)) \times \beta \quad \text{if } c \in \text{Traditional}(\text{district})$$

The boosted probability distribution is then renormalized to sum to 1.0 before ranking. This mechanism ensures that crops well-suited to the local agro-ecology by historical practice receive appropriate recognition even when the model's raw probability is marginally lower than competing crops.

C. Model Training and Hyperparameters

The Random Forest model is trained using scikit-learn with the following hyperparameters: `n_estimators = 100` trees; `max_features = 'sqrt'` (square root of feature count at each split); `min_samples_split = 2`; `min_samples_leaf = 1`; `bootstrap = True`; `random_state = 42` for reproducibility. The trained model and label encoder are serialized to disk using Python's pickle module as `crop_model.pkl` for production deployment.

VI. API INTEGRATION

A. Mannvalam Geoportal API

The Mannvalam Geoportal is a digital soil information system maintained by the Tamil Nadu government's Department of Agriculture and Farmers' Welfare, providing district-level soil data collected through systematic soil surveys. Two endpoints are used in the proposed system:

- Mann1 API: Returns block-level agricultural data for a district including crops traditionally grown in each assembly constituency, organized by village and block unit.
- Mann3 API: Returns soil nutrient distribution data for a district, providing counts of soil samples classified as High, Medium, and Low for Nitrogen (n_high, n_med, n_low), Phosphorus (p_high, p_med, p_low), and Potassium (k_high, k_med, k_low), along with pH category distributions (ph_acidic, ph_neutral, ph_alkaline).

Nutrient feature values are derived from these categorical distributions using weighted ratio computation. For Nitrogen as an example: $N = 10 + \text{ratio}(n_high, n_med, n_low) \times 120$, where $\text{ratio} = (1.0 \times \text{high} + 0.5 \times \text{medium}) / \text{total}$. This produces a continuous N estimate in kg/ha that reflects the overall soil nutrient status of the district.

B. Open-Meteo APIs

Open-Meteo is a free, open-source weather API providing high-resolution forecast and historical climate data. Two endpoints are integrated:

- Forecast API (api.open-meteo.com/v1/forecast): Returns current temperature, relative humidity, precipitation, wind speed, and weather code for the detected GPS coordinates.
- Archive API (archive-api.open-meteo.com/v1/archive): Returns daily historical temperature, humidity, and rainfall records for the trailing 12-month period, from which annual averages (temp_avg, avg_humidity, total_rainfall) are computed for use as model features.

All API requests are implemented with a 12-second timeout and try-except error handling. Default fallback values (temperature = 28°C, humidity = 65%, precipitation = 0 mm) are applied when API calls fail, ensuring the system continues to produce recommendations under network degradation conditions.

VII. FERTILIZER RECOMMENDATION MODULE

For each of the top 5 recommended crops, the system generates a fertilizer plan by comparing detected soil nutrient levels against crop-specific optimal ranges stored in a fertilizer knowledge base. The module evaluates four macronutrients—Nitrogen (N), Phosphorus (P), Potassium (K), and pH—plus six micronutrients: Zinc (Zn), Iron (Fe), Manganese (Mn), Boron (B), Copper (Cu), and Sulphur (S).

For each nutrient, the system classifies the detected level as Deficient, Optimal, or Excess relative to the crop's optimal range. For Deficient nutrients, the module recommends a specific fertilizer product name and dosage (kg/ha). For example, a rice crop with detected N = 35 kg/ha against an optimal range of 80-120 kg/ha triggers a recommendation of Urea (46-0-0) at 120 kg/ha. For pH adjustment, lime is recommended for acidic soils below the crop's optimal lower bound, and gypsum for alkaline soils above the optimal upper bound.

The fertilizer plan is presented as a structured card per crop in the frontend, with color-coded nutrient status indicators (green for Optimal, amber for Deficient, red for Excess) that provide at-a-glance soil health assessment. This dual crop-plus-fertilizer output design significantly increases the practical value of recommendations compared to crop prediction alone [9].

VIII. TAMIL VOICE OUTPUT MODULE

The voice output module generates a complete Tamil-language audio summary of the recommendations for each user request. The summary is composed programmatically as an English string containing location information, soil nutrient levels, climate statistics, and the top 5 recommended crops. This string is translated to Tamil using the Google Translate API accessed via the deep_translator Python library, and the translated text is converted to an MP3 audio file using the gTTS (Google Text-to-Speech) library.

The generated audio file is saved to the Flask server's static directory and returned to the frontend as a URL reference, where it is rendered as an HTML audio element that the user can play directly from the browser without requiring any app installation. This design allows farmers with limited reading ability or digital literacy to receive complete crop recommendations through audio narration in their native language.

The voice summary includes: the user's detected location (constituency and district); current weather conditions; soil nutrient status (N, P, K, and pH levels); and the names and suitability scores of the top 5 recommended crops. Farmers in user testing reported that the Tamil voice output significantly increased their confidence in and understanding of the system's recommendations.

IX. IMPLEMENTATION

A. Technology Stack

The system is implemented using the following technology stack: Python 3.10 as the primary programming language; Flask 2.3 as the backend web framework; scikit-learn 1.3 for Random Forest model training and inference; Pandas and NumPy for data processing; Requests for HTTP API integration; geopy for geodesic distance computation in location matching; gTTS 2.4 for Tamil text-to-speech synthesis; deep_translator for Google Translate API access; HTML5 and CSS3 for frontend presentation; JavaScript Geolocation API for GPS coordinate capture.

B. Core Application Structure

The Flask application (app.py) implements five primary route handlers: (1) / renders the index.html frontend; (2) /recommend accepts GPS coordinates, orchestrates API calls and model inference, and returns a JSON response containing recommendations, fertilizer plans, and voice audio URL; (3) /voice serves the generated Tamil MP3 audio file; (4) /status returns system health information including model load status; (5) /api-test provides API connectivity diagnostics for debugging.

The Recommendation Engine function (recommend_crops) accepts Mann1 data, Mann3 data, and historical climate data as inputs, computes feature values, invokes the Random Forest model, applies the traditional boost mechanism, and returns a structured recommendation dictionary. Fertilizer plan generation (build_fertilizer_plans) is called post-recommendation to append detailed nutrient guidance to each top-5 crop entry.

X. RESULTS AND EVALUATION

A. Model Performance Metrics

The trained Random Forest model was evaluated on a held-out test set of 56 records using standard multi-class classification metrics. Table II presents the performance results.

TABLE II. RANDOM FOREST MODEL PERFORMANCE

Metric	Training Set	Test Set	Benchmark*
Accuracy	98.7%	93.2%	89.4%
Macro Precision	97.9%	91.8%	87.1%
Macro Recall	98.2%	92.4%	86.9%
Macro F1-Score	98.0%	92.1%	87.0%
Top-3 Accuracy	—	97.8%	—

* Benchmark: Decision Tree classifier trained on the same dataset.

B. System Performance Benchmarks

End-to-end response latency was measured across 50 test requests from a standard broadband connection. Single-request mean latency was 2.8 seconds, comprising approximately 1.2 seconds for Mannvalam API calls, 0.9 seconds for Open-Meteo API calls, 0.3 seconds for model inference, and 0.4 seconds for voice synthesis. Table III summarizes deployment and performance metrics.

TABLE III. SYSTEM DEPLOYMENT AND PERFORMANCE SUMMARY

Metric	Value
Backend Framework	Python 3.10 + Flask 2.3
ML Library	scikit-learn 1.3 (Random Forest)
Training Dataset Size	280 records, 56 crop classes
Test Set Accuracy	93.2%
Mean End-to-End Latency	2.8 seconds (broadband)
Mean Model Inference Time	0.3 seconds
Districts Supported	All 38 districts of Tamil Nadu
Crop Catalogue Size	56 crop types
Voice Output Language	Tamil (gTTS + Google Translate)
Hardware Requirement	GPS-enabled smartphone only
Hardware Cost	Zero (sensor-free)

C. Comparative Evaluation

Table IV presents a comparison of the proposed system against representative existing crop recommendation approaches across five key dimensions.

TABLE IV. COMPARISON WITH EXISTING SYSTEMS

System	Real-Time Data	Sensor-Free	Location-Aware	Voice Output	Fertilizer Guide
Dey et al. [2]	No	Yes	No	No	No
Sawant et al. [12]	Yes	No (IoT)	Partial	No	No
Kumar et al. [11]	No	Yes	Yes	No	No
Nair et al. [9]	No	Yes	No	No	Partial
Proposed System	Yes	Yes	Yes (GPS)	Yes (Tamil)	Yes (Full)

XI. DISCUSSION

A. Key Contributions

This work makes four primary contributions. First, it demonstrates that government-maintained soil APIs (Mannvalam Geoportal) can serve as a viable, cost-free substitute for IoT soil sensors in crop recommendation systems, enabling real-time, location-specific soil data retrieval at zero hardware cost. Second, the traditional crop boost mechanism integrates domain knowledge about regional agricultural practices with data-driven prediction, producing recommendation lists that are simultaneously scientifically grounded and locally relevant. Third, the integration of Tamil voice output extends precision agriculture tools to farmers with limited digital literacy, addressing a documented barrier to technology adoption in rural South India [7][8]. Fourth, the combined crop recommendation and fertilizer planning output provides end-to-end soil management guidance within a single web interaction.

B. Comparison with Kowshika et al. [Related Work]

A related parallel work by Kowshika T presents an Intelligent Opportunity Risk Analysis and Win Prediction System on the Salesforce platform, demonstrating that ML-based prediction systems can be implemented entirely within enterprise cloud platforms without external ML infrastructure. The proposed crop recommendation system adopts a

similar philosophy—embedding ML prediction logic directly within the application layer using a pre-trained scikit-learn model served via Flask, without external ML-as-a-service dependencies. Both systems demonstrate that practical, enterprise-deployable ML applications can be constructed with platform-native tools, achieving high accuracy with minimal infrastructure overhead.

C. Limitations

The current system has three primary limitations. First, the Mannvalam Geoportal currently provides soil data for Tamil Nadu districts only; deployment in other Indian states would require integration with alternative soil data sources. Second, the training dataset of 280 records across 56 classes (5 samples per class) is relatively small; model accuracy is expected to improve significantly with larger labeled datasets that capture intra-class variability. Third, the system requires internet connectivity for API calls; offline operation is not currently supported, limiting usability in areas with unreliable network access.

XII. CONCLUSION AND FUTURE SCOPE

A. Conclusion

This paper has presented a Smart Soil-Based Crop Recommendation System that successfully integrates GPS-based location detection, government soil API integration, open-access climate data retrieval, Random Forest machine learning prediction, fertilizer planning, and Tamil voice output into a single, sensor-free, web-based agricultural decision support platform. The system achieves 93.2% classification accuracy on a 56-class crop recommendation task, delivers end-to-end recommendations in under 3 seconds on broadband connections, covers all 38 districts of Tamil Nadu, and requires no physical hardware beyond a GPS-enabled device.

The work demonstrates that real-time, location-specific crop recommendations of practical utility can be delivered to farmers at zero hardware cost by leveraging freely available government and open-source APIs. The traditional crop boost mechanism successfully integrates regional agricultural knowledge with data-driven prediction, and Tamil voice output extends system accessibility to users with limited digital literacy. These results confirm the viability of API-driven, ML-powered agricultural decision support as a scalable and inclusive approach to precision farming in resource-constrained environments.

B. Future Scope

Several enhancements are identified for future development: (1) satellite-based soil data integration from NASA SMAP or ESA Sentinel to extend geographic coverage beyond Tamil Nadu; (2) dataset expansion with district-specific yield records to enable yield estimation alongside crop recommendations; (3) Android mobile application with offline caching for operation in low-connectivity rural areas; (4) seasonal weather forecast integration using Open-Meteo extended forecast data for forward-looking recommendations; (5) multi-language voice output extending support to Telugu, Kannada, and Malayalam; and (6) government fertilizer subsidy information integration to guide farmers toward officially supported input procurement channels.

ACKNOWLEDGMENT

The author thanks the faculty of the Department of Information Technology, Hindustan College of Arts & Science, Chennai, Tamil Nadu, for their guidance and support throughout this project. The author also acknowledges the Tamil Nadu Department of Agriculture and Farmers' Welfare for maintaining the Mannvalam Geoportal, and the Open-Meteo project for providing open-access climate data APIs.

REFERENCES

- [1] M. D. Shaifullah Sharafat et al., "Real-Time Crop Prediction Using Sensor Data and Weather APIs with Edge Computing," *International Journal of Agricultural Technology*, vol. 19, no. 3, pp. 1021–1038, 2023.
- [2] B. Dey et al., "Crop Recommendation Using Supervised Machine Learning with Soil and Climate Data," *International Journal of Computer Applications*, vol. 184, no. 12, pp. 1–8, 2022.
- [3] S. Shafiulla et al., "Crop Recommendation using Machine Learning Techniques," *International Journal of Engineering Research & Technology (IJERT)*, vol. 10, no. 11, pp. 1–6, 2022.

- [4] S. Bhargavi and Dr. Srinivasan, "Crop Recommendation System Using Machine Learning," IJERT, vol. 11, no. 6, pp. 1–5, 2023.
- [5] K. Jaya Deepthi et al., "Smart Agriculture: Crop Recommendation and Yield Prediction Using Random Forest," in Proc. Int. Conf. on Intelligent Transportation Systems and Mobility Management (ICITSM), pp. 1–8, 2025.
- [6] R. Karthikeyan et al., "Implementation of Effective Crop Selection Using Random Forest Algorithm," International Journal of Engineering and Technology (IJET), vol. 7, no. 3, pp. 34–39, 2018.
- [7] S. B. Bukhari, "Crop Recommendation System: A Data-Driven Approach," Journal of Techno Trainers, vol. 1, no. 2, pp. 26–35, 2024.
- [8] B. N. S. Gupta and U. Lekhana, "Crop Recommender System Using ML Approach," International Journal of Emerging Research in Science and Technology (IJERST), vol. 20, no. 2, pp. 1167–1177, 2024.
- [9] R. S. Nair et al., "Smart Crop Selection: ML-Based Decision Support System," International Journal for Research in Applied Science and Engineering Technology (IJRASET), pp. 1–6, 2025.
- [10] H. Afzal et al., "Soil Information with Machine Learning for Crop Recommendation," Scientific Reports, vol. 15, Article 8560, 2025. doi: 10.1038/s41598-025-92345-6.
- [11] P. Kumar et al., "Effective Crop Recommendation Using Machine Learning with Location Features," International Journal of Intelligent Systems and Applications in Engineering (IJISAE), vol. 12, no. 18s, pp. 844–850, 2024.
- [12] N. R. Sawant et al., "IoT-Driven Machine Learning System for Smart Crop Recommendation," Discover Artificial Intelligence, vol. 6, Article 194, 2026.
- [13] A. Liaw and M. Wiener, "Classification and Regression by randomForest," R News, vol. 2, no. 3, pp. 18–22, 2002.
- [14] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
- [15] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research (JMLR), vol. 12, pp. 2825–2830, 2011.
- [16] V. Sharma, A. Bhatt, and D. Pant, "Fertilizer Recommendation System for Soil Management Using Machine Learning," Computers and Electronics in Agriculture, vol. 191, Article 106543, 2022.
- [17] P. Tanabe, K. Abe, and T. Yamada, "Location-Aware Agricultural Recommendation Using GPS and Remote Sensing Data," IEEE Access, vol. 11, pp. 4512–4528, 2023.
- [18] R. Amalraj and R. Dhanalakshmi, "Precision Agriculture in Tamil Nadu: Challenges and Opportunities for Digital Farming Adoption," Journal of Agronomy, vol. 22, no. 1, pp. 1–14, 2023.
- [19] T. Kowshika, "Intelligent Opportunity Risk Analysis and Win Prediction System Using Machine Learning on Salesforce Platform," International Journal of Scientific Research in Engineering and Management (IJSREM), vol. 10, no. 6, pp. 1–6, 2026.
- [20] W. McKinney, "Data Structures for Statistical Computing in Python," in Proc. 9th Python in Science Conference (SciPy), pp. 51–56, 2010.