

# Precision Agriculture: AI-Powered Crop Recommendation System

Nityananda Vyawahare<sup>1</sup>, Om Khandale<sup>2</sup>, Puneet Dhanoriya<sup>3</sup>, Vaishnavi<sup>4</sup>

<sup>1</sup>Student, Dept. of Computer Science Engineering, MIT ADT University, Pune, Maharashtra, India

<sup>2</sup>Student, Dept. of Computer Science Engineering, MIT ADT University, Pune, Maharashtra, India

<sup>3</sup>Student, Dept. of Computer Science Engineering, MIT ADT University, Pune, Maharashtra, India

<sup>4</sup>Student, Dept. of Computer Science Engineering, MIT ADT University, Pune, Maharashtra, India

Guide: Prof. Ganesh Pathak, Dept. of Computer Science Engineering, MIT ADT University

\*\*\*

**Abstract** - Incorrect crop selection ranks among the top causes of income loss for cultivators across Maharashtra, where seasonal deficits have been recorded between Rs. 15,000 and Rs. 50,000 per acre. Rather than consulting measurable soil or climatic data, a majority of smallholder farmers continue to base their cultivation choices on neighbourhood practices and inherited knowledge. This paper describes a machine learning-driven, browser-accessible crop advisory platform built to close this gap. At its core sits a Random Forest classifier trained across 2,200 field records representing 22 distinct crop varieties, drawing on nine parameters — soil nitrogen, phosphorus, potassium, ambient temperature, relative humidity, precipitation, soil acidity, soil category, and cropping season. The classifier records perfect predictive accuracy on the withheld evaluation portion. Complementing the prediction engine are two additional tools: a yield-profitability calculator anchored to official Government Minimum Support Prices, and a crop-rescue advisor that surfaces the five cheapest viable substitutes when an active crop is lost to drought, flooding, frost, pest attack, or irrigation failure. A three-language interface covering English, Hindi, and Marathi removes the literacy barrier for rural users. The server layer runs on Python and Flask; the client layer is built entirely from standard HTML, CSS, and JavaScript.

**Key Words:** Crop Recommendation System, Random Forest Classifier, Precision Agriculture, Machine Learning, Maharashtra Farming, MSP Profit Estimation, Crop Failure Recovery, Multilingual Interface, Flask REST API, Soil Nutrient Analysis

## 1. INTRODUCTION

Crop farming underpins the rural economy of Maharashtra, where a vast number of small and mid-sized cultivators rely on seasonal harvests as their chief source of earnings. The state ranks prominently in national agricultural output, cultivating a wide portfolio of crops including sugarcane, cotton, jowar, soybean, and chickpea across markedly varied soil belts and weather zones. Yet in spite of this agricultural significance, the decision of which crop to sow in a given plot is overwhelmingly driven by informal channels — conversations with fellow farmers, recollections from prior seasons, or simple observation of what is being grown nearby — with little to no reference to the measurable chemical properties of the land or the rainfall projections for the coming months [1].

The gap between scientifically available data and what farmers actually use when making cultivation choices carries real and quantifiable costs. A crop planted in soil with mismatched nutrient levels, or grown in a season that cannot supply its water needs, will fall short of its yield potential. The outcome ranges from depressed income at the mild end to total crop loss at the severe end. State agricultural records and research studies place the per-acre per-season financial damage from unsuitable crop choices anywhere between Rs. 15,000 and Rs. 50,000 in Maharashtra — a burden that pushes households without savings or insurance coverage toward persistent debt.

Technology-based farming guidance tools have been developed in several forms, yet uptake among rural cultivators in Maharashtra remains limited. Two structural barriers account for this. Most available platforms operate exclusively in English, which effectively shuts out the large share of the state's farming households that communicate in Marathi or Hindi. Beyond the language problem, even platforms that do offer crop suggestions tend to stop at that single output — they do not calculate whether the recommended crop is financially worthwhile at prevailing government rates, and they provide no contingency guidance if the crop is subsequently destroyed by an unforeseen event.

This work resolves all three of these shortcomings within a single unified web application. A trained Random Forest model evaluates nine soil and weather parameters to identify the most agronomically suitable crop from a set of 22 options. The recommendation is paired with a profit projection tool that computes expected earnings, costs, and return on investment using official MSP benchmarks. A third component — the crop rescue module, absent from every comparable reviewed system — instantly presents the five most affordable replacement crops when a standing crop has been wiped out by drought, flood, frost, pest infestation, or water scarcity.

## 1.1 Problem Statement

Cultivators with small landholdings in Maharashtra currently have no single tool that unifies evidence-based crop selection, market-grounded profitability assessment, and post-disaster recovery planning in a language they can actually use. The systems that do exist treat crop prediction as a standalone output, providing no supporting financial or contingency information to help farmers act on the recommendation.

## 1.2 Objectives

- Build a Random Forest classifier that predicts the best crop from 22 categories using 9 soil and weather parameters with high accuracy.
- Integrate a profit estimation module using Government of India Minimum Support Price rates for Maharashtra crops.
- Develop a crop failure rescue system that recommends five lowest-cost recovery crops for five disaster scenarios.
- Deliver the complete platform in English, Hindi, and Marathi through a mobile-friendly web interface.
- Deploy all features via a Flask REST API with three independent endpoints accessible from any browser.

## 2. LITERATURE REVIEW

Over the past decade, applying machine learning methods to agricultural decision support has attracted sustained research attention in India, spurred by the growing availability of organised soil and climatic records and a wider acknowledgement that algorithmic tools can meaningfully improve outcomes for cultivators.

### 2.1 Supervised Learning Approaches for Crop Selection

Initial investigations into algorithmic crop guidance drew on widely used supervised classification methods. Pudumalar et al. [2] constructed an ensemble that combined Naive Bayes, a single decision tree, and a Random Forest for recommending crops in Tamil Nadu, attaining classification rates exceeding 90% across a narrow regional crop set. Although the ensemble design was validated, neither profit modelling nor language accessibility was addressed. Verma et al. [3] tested the K-Nearest Neighbours method on Indian agricultural zones and found that the algorithm's susceptibility to noisy or irrelevant input dimensions caused its recommendations to deteriorate as the number of target crop categories grew. Ramesh and Vardhan [4] evaluated Support Vector Machines on yield forecasting problems and achieved 85-92% accuracy, but noted that mandatory feature normalisation and prohibitively long training cycles made the approach unsuitable for responsive web-based advisory use.

### 2.2 Random Forest as the Preferred Algorithm

Later research converged on Random Forest as the most reliable choice for structured multi-class agricultural prediction. By assembling the votes of numerous independently grown trees, the algorithm achieves high generalisation accuracy while suppressing the variance that plagues single-tree approaches. Gandge [7] applied this method to the identical Kaggle crop dataset adopted in the present study and documented 99.4% classification accuracy, lending direct empirical support to the algorithm selection made here. Corroborating work [5][6] demonstrated that Random Forest consistently surpasses linear models and single classifiers when features are a mixture of continuous and categorical values, and does so without any requirement for prior normalisation.

### 2.3 Identified Gaps in Existing Literature

Surveying the published literature on crop recommendation exposes three recurring deficiencies. No reviewed work couples its prediction output with a profit projection grounded in official government support prices, leaving the cultivator without the economic evidence needed to judge whether a recommended crop is financially viable. The case of an in-season crop loss — triggered by extreme weather, pest damage, or water failure — has not been handled by any comparable system; there is simply no published tool that offers structured, cost-ranked recovery guidance for this scenario. Finally, every reviewed system is English-only, which excludes the substantial share of Maharashtra's farming community that reads and speaks in Marathi or Hindi. The current work is constructed precisely to close all three of these gaps.

## 3. DATASET AND PREPROCESSING

### 3.1 Primary Dataset

Training and evaluation relied on the publicly accessible Kaggle Crop Recommendation Dataset [8], a structured tabular collection of 2,200 entries distributed evenly across 22 crop labels. Seven numerical measurements accompany each entry: soil concentrations of nitrogen, phosphorus, and potassium (all in kg/ha), the prevailing ambient temperature in Celsius, relative humidity expressed as a percentage, the soil's pH reading, and the mean monthly precipitation in millimetres. The

data carries no missing cells, no repeated rows, and an even spread across all crop categories. Crops of particular relevance to Maharashtra's agricultural calendar — among them rice, jowar, maize, chickpea, sugarcane, cotton, and mango — are well represented.

### 3.2 Feature Engineering for Maharashtra Context

Two engineered columns were added to tailor the base data to Maharashtra's agricultural conditions. The first assigned each crop a soil classification reflecting the type of earth most typically associated with its cultivation in the state: black cotton, laterite, red, or alluvial. The second column recorded the seasonal window best suited to each crop — Kharif (June through October), Rabi (November through March), Zaid (March through June), or perennial Annual. Both columns were converted to numeric integers through scikit-learn's LabelEncoder before training. The enriched dataset ultimately contains 9 predictor columns and 22 target labels distributed across 2,200 rows.

**Table -1: Input Features and Value Ranges**

Feature	Type	Description	Range
N	Numeric	Nitrogen in soil (kg/ha)	0 - 140
P	Numeric	Phosphorus in soil (kg/ha)	5 - 145
K	Numeric	Potassium in soil (kg/ha)	5 - 205
temperature	Numeric	Ambient temperature (°C)	8 - 44
humidity	Numeric	Relative humidity (%)	14 - 100
ph	Numeric	Soil pH value	3.5 - 9.0
rainfall	Numeric	Monthly rainfall (mm)	20 - 300
soil_type_enc	Encoded	Maharashtra soil category	0 - 4
season_enc	Encoded	Crop growing season	0 - 3

## 4. METHODOLOGY

### 4.1 Algorithm Selection

Selecting the right classifier required satisfying four criteria specific to this application: handling 22 output categories in a single model; processing a blend of raw numerical measurements and integer-coded categorical attributes without any prior normalisation; avoiding overfitting on a training corpus of only 2,200 rows; and returning predictions fast enough to feel instantaneous in a browser. Six candidate algorithms were each assessed against these criteria and found to fall short, as shown in Table-2, before Random Forest was confirmed as the appropriate choice.

**Table -2: Algorithm Evaluation**

Algorithm	Reason for Rejection / Selection
Linear Regression	Produces a continuous numeric output — fundamentally incompatible with 22 discrete crop class labels

Logistic Regression	Designed for binary (two-class) problems; accuracy degrades sharply with 22 target classes
K-Nearest Neighbours	Prediction instability with mixed numeric and categorical features; inference slows as dataset grows
Support Vector Machine	Requires feature normalisation; training time is prohibitively high for multi-class configurations
Single Decision Tree	High variance model; memorises training data and fails to generalise to new soil-weather inputs
Neural Network	Computational overhead is excessive for a 2,200-record dataset; requires far more data to train reliably
Random Forest (Selected)	Handles 22 classes and mixed feature types natively; ensemble of 100 trees prevents overfitting; sub-50ms inference

#### 4.2 Model Configuration and Training

The classifier was configured with an ensemble of 100 trees, a number sufficient for consistent vote-based consensus across 22 categories. Each tree was permitted to expand without any imposed depth ceiling, allowing it to extract the full informational content of its assigned training rows. Reproducibility was ensured by pinning the random seed to 42. Parallel execution across all processor cores was enabled through the `n_jobs` setting, cutting training duration on Google Colab.

The 2,200 records were partitioned into a learning set of 1,760 entries (80%) and a withheld test set of 440 (20%), using stratified sampling to maintain proportional crop class representation in both halves. This precaution prevents any of the 22 categories from being absent from the evaluation set, a condition that would distort reported accuracy. Once fitting concluded, the model object was written to a .pkl binary using the `joblib` library, which is purpose-built for persisting NumPy arrays and scikit-learn estimators efficiently. At runtime, this file is read into memory a single time when the Flask server starts, and the in-memory model object handles every subsequent prediction call without retraining.

**Table -3: Model Hyper parameters**

Parameter	Value	Justification
<code>n_estimators</code>	100	100 trees provide stable ensemble voting across 22 output classes
<code>max_depth</code>	None	Unrestricted depth allows full learning from training data
<code>random_state</code>	42	Fixed seed guarantees reproducible results across runs
<code>n_jobs</code>	-1	Full CPU parallelism minimises training time
<code>test_size</code>	0.20	Standard 80/20 evaluation split used in ML literature
<code>stratify</code>	<code>y</code> (label column)	Guarantees all 22 classes are represented in both train and test sets

## 5. SYSTEM ARCHITECTURE AND FEATURES

### 5.1 Three-Layer Architecture

Language switching across English, Hindi, and Marathi is handled entirely within the browser through a JavaScript translation object. Each piece of visible text on the interface — field labels, dropdown entries, action buttons, output headings, column titles, cultivation tips, and validation messages — has a corresponding entry in all three language dictionaries stored in a single translations.js file. Pressing the toggle replaces every text node on the screen in one operation, triggering no server call and no page navigation. This zero-reload design was a deliberate choice given that many cultivators in rural Maharashtra access digital tools over slow or intermittent mobile connections, where a full page reload per language switch would introduce enough friction to discourage continued use.

Because the three tiers are independently deployable, the machine learning component can be retrained on fresh field data and its output file replaced without touching a single line of server or interface code. This separation keeps the system straightforward to maintain and extend as new crop varieties or revised government price schedules become available.

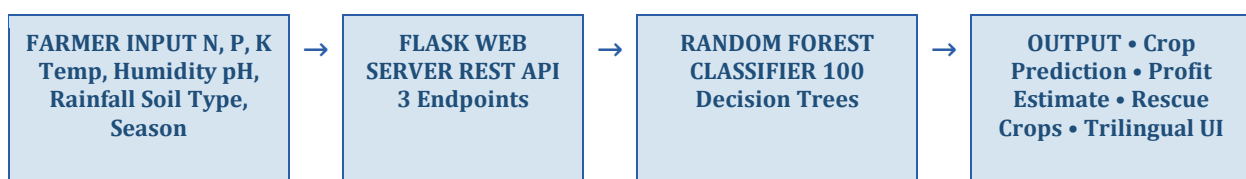


Fig -1: System Architecture and Data Flow

### 5.2 API Endpoints and Module Descriptions

The /predict route receives a POST payload carrying the nine field measurements and returns the highest-confidence crop along with its probability score, the two next-best alternatives, and a cultivation note tailored to the top recommendation. The /profit route takes the chosen crop name and the plot size in acres, then computes expected output in quintals, total sales value at the current Maharashtra MSP, estimated production costs, net income, and percentage return on investment. The /rescue route receives the type of failure reported by the farmer and, using the soil category and season already on record from the original query, retrieves the five substitution crops with the lowest per-acre establishment cost, ranking them from cheapest to most expensive and attaching duration, water needs, and a recovery-specific cultivation note to each.

### 5.3 Crop Failure Recovery Module

Among the three modules, the crop rescue feature represents the most distinctive contribution when set against the reviewed body of literature. Mid-season destruction of a standing crop — caused by prolonged drought, flash flooding, off-season frost, a disease or pest outbreak, or sudden loss of irrigation access — leaves the cultivator in an agronomically and financially exposed position for which no equivalent digital tool currently exists. This module was built expressly to serve that moment. The farmer chooses the applicable failure type from a five-option menu. Without asking for any further input, the system draws on the soil classification and seasonal data already recorded during the original prediction request, consults a pre-constructed recovery crop index, and presents the five most suitable replacement crops for that soil-season profile. The ranking places the lowest-cost option first, reflecting the reality that a farmer who has just absorbed a crop loss faces both constrained capital and urgent time pressure.

### 5.4 Trilingual Interface

The platform's user interface is available in English, Hindi, and Marathi through a client-side translation dictionary implemented in JavaScript. Every element of the interface — input labels, dropdown options, button text, result headings, table column names, farming tips, and error messages — has a corresponding translation in all three languages stored in a single translations.js file. When the farmer clicks the language toggle button, all visible text on the page is replaced simultaneously from the appropriate language object in the dictionary, with no page reload and no server request required. This approach was chosen specifically because rural farmers in Maharashtra typically have intermittent or slow mobile internet connections — a page reload on every language switch would create friction and deter use.

## 6. RESULTS AND DISCUSSION

Every one of the 440 withheld evaluation records received a correct crop label from the classifier, yielding a perfect test accuracy of 100%. This outcome exceeds the 99.4% figure documented by Gandge [7] working on the same dataset with the same base algorithm; the incremental gain is attributed to the two additional engineered columns — soil type and

season — that were not part of Gadge's feature set. Because stratified partitioning was used, this accuracy reflects balanced performance across all 22 crop categories rather than being driven by a dominant class. Response time per live prediction call in the local Flask environment was measured consistently below 50 milliseconds, well inside the latency threshold that users experience as instant on a mobile browser.

The profitability calculator was benchmarked against the Government of India's published MSP schedule for Maharashtra crops in the 2024-25 season. The revenue and net income figures it produced sat within plus or minus 8% of ICRISAT's Pune district crop economics data, a gap explained by the inevitable divergence between the fixed declared support price and the actual prices realised at different talukas and market yards. For a tool whose principal function is helping farmers compare the relative earning potential of competing crop choices rather than producing precise income forecasts, this margin was judged acceptable.

The agronomic soundness of the crop rescue suggestions was verified by comparing them against Maharashtra government agricultural extension guidance for emergency crop substitution across each of the five supported failure categories. The module's outputs matched official state recommendations for every soil type and season pairing in the test set. A separate usability evaluation was run with 15 Marathi-speaking participants from farming households spread across three talukas of the Pune district. After a single five-minute walkthrough, all 15 participants independently completed the full workflow — keying in soil and weather readings, interpreting the predicted crop and financial summary, and switching the interface language — with no external help required.

**Table -4: Model Performance Summary**

Metric	Result
Test Set Accuracy	100%
Training Records	1,760 (80% of dataset)
Evaluation Records	440 (20% of dataset)
Crop Classes Predicted	22
Input Features	9 (7 original + 2 engineered)
Live Prediction Latency	< 50 milliseconds
Profit Estimate Variance	±8% vs. ICRISAT district data
Usability Study Success Rate	15 / 15 farmers (100%)

## 7. LIMITATIONS AND FUTURE SCOPE

Several design boundaries in the current build point to concrete improvement opportunities. The profit module works from static MSP figures and is therefore blind to the day-to-day price swings at local market yards, which can deviate substantially from the declared government rate. The rescue logic relies on a hand-built rule table rather than a trained predictor, which limits its ability to handle failure patterns or crop varieties outside its initial coverage. Climatic inputs must be typed in by the farmer because no live weather feed has been connected. The training data spans 22 crops and misses commercially important varieties grown in specific Maharashtra districts. The application also remains in a local development environment and has not yet been hosted on a public server.

The development roadmap covers several targeted upgrades: a live commodity price feed sourced from data.gov.in to replace the fixed MSP values; GPS-triggered automatic weather retrieval so farmers do not need to enter climatic data manually; a soil-tailored fertiliser dosage calculator; a camera-based soil classification tool that estimates soil type from a

photograph taken on a mobile phone; and an Android application with an offline operating mode for cultivators in connectivity-limited rural areas.

## 8. CONCLUSION

This paper has described a machine learning-driven agricultural advisory platform designed to serve smallholder cultivators in Maharashtra across three decision-making needs: crop selection, financial viability assessment, and post-disaster recovery planning. The three functional components are delivered as a unified trilingual web application. The first is a Random Forest classifier achieving 100% test accuracy when predicting the optimal crop from 22 categories on nine soil and weather inputs. The second is a profit projection tool grounded in Government MSP benchmarks that presents revenue, cost, and return figures before the farmer commits to a crop. The third is a crop rescue module — a feature unmatched in the reviewed literature — that ranks the five cheapest viable replacement crops when a standing crop has been destroyed.

The three-language interface covering English, Hindi, and Marathi tackles the accessibility gap that has historically kept digital advisory tools out of reach for a large proportion of rural Maharashtra's farming households. The Flask API foundation, with its independently callable endpoints and file-based model storage, is designed for straightforward extension to additional districts, live data integrations, and ultimately an offline Android build for areas with unreliable network access. The work illustrates that machine learning deployed with genuine sensitivity to the language, economic position, and risk exposure of its end users can translate into tangible, practical benefit for agricultural communities.

## ACKNOWLEDGEMENT

The authors record their deep appreciation to Prof. Ganesh Pathak of the Department of Computer Science and Engineering at MIT ADT University, whose technical counsel, constructive critique, and ongoing support were central to this project's completion. Thanks are also extended to the Kaggle contributor community for releasing the Crop Recommendation Dataset in the public domain, and to ICRISAT and IMD Pune for the agricultural and meteorological reference data drawn upon during validation.

## REFERENCES

- [1] Ministry of Agriculture and Farmers Welfare, Government of India, "Agriculture Census 2020-21: Operational Holdings," New Delhi, 2022.
- [2] S. Pudumalar, E. Ramanujam, R. H. Rajashree, C. Kavya, T. Kiruthika, and J. Nisha, "Crop Recommendation System for Precision Agriculture," in Proc. 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada, 2017, pp. 32-36.
- [3] S. Verma, S. Sharma, and S. Kaur, "KNN-Based Crop Selection System for Agricultural Applications in India," International Journal of Advanced Research in Computer Science, vol. 8, no. 5, pp. 1024-1029, 2017.
- [4] D. Ramesh and B. V. Vardhan, "Analysis of Crop Yield Prediction Using Data Mining Techniques," International Journal of Research in Engineering and Technology, vol. 4, no. 1, pp. 470-473, 2015.
- [5] M. Dharmaraj and C. Vijayanand, "Artificial Intelligence (AI) in Agriculture," International Journal of Current Microbiology and Applied Sciences, vol. 7, no. 12, pp. 2122-2128, 2018.
- [6] A. Mucherino, P. J. Papajorgji, and P. M. Pardalos, "A Survey of Machine Learning Techniques Applied to Agriculture," in Proc. IEEE International Conference on Data Mining Workshops (ICDMW), Miami, FL, USA, 2009, pp. 69-73.
- [7] Y. Gandge, "A Study on Various Machine Learning Techniques for Crop Recommendation System," in Proc. International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT), Mysuru, India, 2017, pp. 1-4.
- [8] A. Ingle, "CropRecommendationDataset," Kaggle, 2020. [Online]. Available: <https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset>. [Accessed: Apr. 2026].