International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

KrishiAI: A Rule-Based Organic Farming Recommendation and Farm Support Platform

[1] Thogiti Rishindra (237R1A6661),
Department of Computer Science and Engineering
(Artificial Intelligence and Machine Learning)
CMR Technical Campus, Medchal, Telangana-501401,
India

[3] Valaboju Ruthvik (237R1A6662),
Department of Computer Science and Engineering
(Artificial Intelligence and Machine Learning)
CMR Technical Campus, Medchal, Telangana-501401,
India

[2] Paranandi Sarayu (237R1A6646),
Department of Computer Science and Engineering
(Artificial Intelligence and Machine Learning)
CMR Technical Campus, Medchal, Telangana-501401,
India

[4] Bushra Tarannum
Assistant Professor, Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning)
CMR Technical Campus, Medchal, Telangana-501401,
India

Abstract - In the case of agricultural activities, decisions have to be taken on crop suitability, soil and water status, and management of irrigation, as well as on crop cultivation. These decisions may require consulting multiple sources (especially when they have to find it in different tools). In this paper, author's introduce KrishiAI: Web Farm Support platform to provide a centralized approach to multiple agriculture support functions. The project also has a backend developed with FastAPI and a relational database. The modules include farm profiles, soil data, crop suggestions, plantation tips, tracking, organic data, pest and disease data, harvest support data, market data, government schemes and a chat-bot assistant that guides farmers through the process of farming. Checking from the submitted implementation reveals that there is currently no verified model (TabPFN) included with the crop recommendation service, but instead predefined and score based crop records. Therefore, this paper presents the implemented system as a rule-based recommendation system, and discards measured machine-learning accuracy, yield improvement or profit maximization for these reasons. The paper explains the system structure, system flow, system developments and current system restrictions and practical recommendations for further development.

Keywords: KrishiAI; organic farming; crop recommendation; farm management; FastAPI; React; rule-based decision support; agricultural information system

I. INTRODUCTION

Soil properties, water availability, location climate, crop needs and the last stages of cultivation are factors which determine agricultural decisions. Knowledge related to these decisions is typically disbursed across a variety of sources making it challenging for farmers to have their information complete and to relate recommendations with their own farms. A digital farm support platform can allow for farm details to be written down and shared in one place, review crop guidance, and follow cultivation activities.

As part of this, the idea of KrishiAI, a web application to facilitate these tasks from a single point was conceived. The application combines farm and soil records, crop recommendation, plantation and harvest guidance, crop tracking, information about organic inputs, advice on pest and diseases, market related information, information on the government schemes and a conversational assistant. It aims at making farm information more accessible and easier to organize, and not to be an alternative to local agricultural skills.

One of these distinctions of documenting this project is that it is technology that has been designed and not the technology that is going to be implemented that will be inspected. The "TabPFN" tags mentioned in the previous project descriptions are not used in the backend recommendation service that was submitted; Instead of using TabPFN, the submitted backend makes use of a set of croplists and a set of predefined scoring rules. Therefore, the present paper describes the functionality implemented, but does not claim to be a validated TabPFN system. With this, it will be a technically correct report of the final product and will show what the requirements are in order to be able to make claims about the performance of the machine-learning techniques.

II. PROBLEM STATEMENT AND OBJECTIVES

Farmers will often need to switch between various resources or tools to view soil data, crop suitability, on-farm management activities, organic management and information for various schemes. It is also inconvenient to keep records on farms and make connections between records and advice when there isn't a particular farm, or the one farm doesn't have information about the other.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

The goal of KrishiAI is to create a single digital farm tool to access information and farm-related services. Specific objectives are to:

- To enable users to keep farm records and associated soil data on the farm;
- To deliver crop recommendations based on the existing rule based crop scoring logic;
- To plan on plantations, track and harvest cultivations;
- To share information on organic inputs, pests, sickness, markets and government schemes; To offer a 'chat' like capability for asking farming questions, depending on configuration and data; To provide application-specific access to the data in the back-end services using database models.
- clearly understanding technical and evaluation constraints to enable wise future planning of improvements.

III. RELATED WORK AND BACKGROUND

Typical functionalities of a digital agriculture system are management systems on the farm, crop guidance, information on the weather, markets, and agricultural extension material. There are a number of different approaches used to develop methods to recommend crops; from manually written rules to as many statistical models and machine learning classifiers. Rule based systems operate based on explicit conditions to the user inputs; they're easy to review, but require well formulated and complete rules to be useful. Machine-learning systems can only learn patterns from labelled data, which calls for appropriate datasets, procedure for training the system, and testing the system with data never used during the training.

Many structured prediction tasks using tabular data have been studied, for instance Tabular foundation models, TabPFN. But the performance results of a model published on a set of data should not be considered as an indicator of the performance results on another agricultural set of data. Any statement regarding a specific implementation of a model should be qualified by a statement of actual implementation in software and support for its predictions and with documentation of such predictions under a specified protocol.

The source code submitted indicates that KrishiAI is likely the most suitable approach as it is designed to be an integrated farm support application that includes modules for multiple pieces of farm support such as crop recommendation using rules. At this point, its part seems to be the introduction of these features within a standard workflow of an application. There could be a future research version by introducing and testing a machine-learning based classifier but it is a different research approach from the work described in this paper.

IV. SYSTEM ARCHITECTURE

An application architecture utilizing client–server. The frontend is competitively built using React, TypeScript and is powered by Vite as a build and development tool. The server (backend) is written in Python and FastAPI. The request and response structure is defined by schema models, and there are database operations defined by SQLAlchemy. Confirmed documentation: SQLite database is the default database for local development. Additional features in the project entail database migration support as well as API endpoints organised by functional region.

The Frontend displays the pages to be seen by the user and sends requests to Endpoints that exist in the Backend. The back end checks the requests, performs application logic, fetches stored records and responds with results. Current crop scoring logic is in the crop recommendation service. Functions like authentication, farms, soil, plantation, tracking, harvest, organic inputs, pest information, market information, schemes, and chat, are provided with separate services and endpoints. The "conversational" service is setup to call an external language-model API if the required key exists; the external API service and the outcome of the call to it is what determines if this service is available and how good the outcome of the call will be.

The project's "architectural" description is as follows:

1. Presentation layer: The user interface with React and TypeScript.
2. Domain layer: pydantic models representing the application's business logic.
3. Data layer: SQLAlchemy models and DB data.
4. Services (recommendation, conversational, weather, market and retrieval).
5. Persistence layer: SQLAlchemy models, and local configured sqlite shared database.

External integrations: services that require API credentials that are set up externally and access to the network. Presence of a service module or a user-interface section, does not mean that any feature has been validated with live, external data or field data.

V. IMPLEMENTED FUNCTIONAL MODULES

A. Farm and Soil Records

The application will have an endpoint for Farm profiles as well as soil-related information and data models. The records can be used to give context to other application functions. The more accurate and complete the information they contain, the more useful they will be to the user.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

B. Crop Recommendation

The inspected back-end recommendation service stores a few crop records, and performs a scoring based on rules that is calculated based on selected crop parameters, such as soil pH, organic matter, soil type, and irrigation parameters. The service sends out information about recommendations via the application workflow. This logic cannot be the same as that of a trained classifier—it cannot be expected to make statistical generalizations from a labeled training set and cannot be presented as validated model probabilities as the result of an experiment.

C. Plantation, Tracking, and Harvest Support

This is an app that has modules for plantation guidance, crop monitoring and harvest information. With these features users are anticipated to be able to plan cultivation activities and to keep track of progress. The specificity of their use will depend on the extent of information about their use of a specific crop and the user's ability to maintain their farm records up to date.

D. Organic Inputs and Pest/Disease Information

The application provides inputted information from the organic area and pest information and advice. Treatment recommendations need to be verified on accurate agricultural resources and appropriate for the crop stage, maturity and locale. It should be noted that all of the recommendations are not agronomically validated.

E. Market and Government Scheme Information

The application contains market/scheme modules. Values may differ over time or geographically in the market and on schemes, and should always include and/or refer to both the information source and the date of the information if applicable. The fact that these modules exist does not indicate that all of the information that is displayed is current, complete and independently verified information.

F. Conversational Farming Support

The project contains a chat endpoint and a service, which is set up for a conversational feature. This service can rely on an external API key. Usually, they will not take the place of local agricultural officers or verified crop specific instructions and may be useful for certain information to be explained. The external service is not available or not has been set up correctly, this can cause behaviour of the application to not match to the expected conversational workflow.

VI. CROP RECOMMENDATION WORKFLOW

The existing recommendation process for crops starts by inputs of farm or soil data via the application. The backend validates the request, and checks for attributes it needs and forwards them to the recommendation service. The service is used to compare value(s) provided with your predefined crop records and utilising your manually defined scoring rules. Applying the candidate crops and displaying them back in the application.

In the limited sense, rules can be examined directly and hence this process is interpretable. However, the scorecard only reflects the elements chosen, thresholds, cropping dates and assumptions. The number of the predefined crop records in the inspected service is limited, the paper does not pretend that rules are applicable for the variety of crops (and soil), regions, seasons, and crop cultivation found in actual farming.

Recommendation scores should therefore be referred to as a rule based scores, instead of as model confidence or prediction accuracy. Instead of applying these recommendations to their particular farms, the rules should be checked by agricultural experts and be applied to some typical local situations.

VII. IMPLEMENTATION AND TESTING

The project repository contains files of Backend, Frontend source code, API endpoints, database models, Dependency Definitions and automated test files. The coded local workflow includes the following tools: Python (backend) and Node.js (npm) (frontend). The frontend hosted using the Vite development workflow and the backend using its run script. The project includes a script to show how to interact with application endpoints.

There are automated test files in the repository. Their presence means that testing has been created, but this does not mean that all tests pass with the submissions. No documented test-run report and/or full evaluation log was provided along with the project materials examined for this paper. Therefore, no test pass percentage, benchmark for response-time, the percentage of classification success or score for user satisfaction are specified in this paper.

A full representation of the validation report should include software versions, configuration of the environment, test commands, results of the tests and failures. It is important to test valid and invalid requests, authentication and access control, database operations, system failure and missing configuration, as well as failures of external services. Any recommendation test also should include the effect of boundary values and incomplete information for the farm.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

VIII. RESULTS AND DISCUSSION

The results of the source inspection support the following observations: KrishiAI is built as a multi module Web application and the existing crop recommendation solution uses a rule-based scoring as defined in a specific set of crop records. The application consists of further routes and services for farm data as well as the delivery of some functions which support cultivation. These are implementation findings as opposed to evidence of the agricultural impact.

It was not possible to find a labelled subset of the data to use to suitably evaluate the model, a prediction file aside from the training set, a confusion matrix and/or a reproducible classification report in the provided data. Therefore there are no numbers to report when it comes to accuracy, precision, recall, F1 or a comparative model performance. Similarly, dashboard elements like Projected Yield or estimated profits should not be reported as an outcome without documentation on data sources, methods of calculation and how that is validated.

Main outcome of the present work is the development of an integrated application workflow to organize and present farm information, crop guidance (based on rules) and other farm support modules. While the recommendations developed by the system are a foundation to build on, the quality of the information given in the recommendations and their impact on the farm should be tested, documented, and validated using a representative sample of farms and expert reviews.

IX. LIMITATIONS AND ETHICAL CONSIDERATIONS

There are various restrictions which need to be taken into account. The crop recommendation service is created with only a small number of pre-stored crop data, and with a manual procedure to specify the scoring of the recommendation. It should not be offered as a proven TabPFN implementation, nor in the form of a model trained on a publicly documented agriculture specific data set. Second, the materials used in the project included do not contain a full set of experimental data or experimentally measured results for the model performance. Thirdly, external functions may require network access and API credentials which may influence the availability of these functions over deployments. Lastly, details regarding market prices, weather conditions, government programs, pests and pest management can get outdated or need to be localized.

Operational information that may be sensitive can be recorded on the farm. The system (once it's deployed) should provide protection for credentials, apply authorization uniformly, provide secure transport, and ensure proper backup. Assumptions should be clearly made when giving agricultural advice; if advice is given and important agricultural decisions are made, this should be followed by the verification process by checking with trusted people locally. Estimated yield and profit values are estimates, and should not be shown as guaranteed results.

X. FUTURE WORK

The rules and recommendations for crops could be extended and tested with experts on agriculture and case studies from the local community. When implementing machine-learning models, it should be explicitly identified and loaded into the model and should be documented what features goes into the training of that model and what data source is used to produce it and should also ensure that the maximum separation between the model outputs and post-processing as needed rule. The evaluation should be done on a held out test set and accuracy, balanced accuracy, macro precision, macro recall, macro F1, per-class results and confusion matrix should be reported. It is recommended to conduct an analysis of the baseline model (such as random forest) using the same dataset split.

In addition, more work needs to be done on the crop coverage, documentation of sources for agronomic advice, on dealing with missing and conflicting farm data, and on testing for failures of external services. The dates/place of origin of the market or scheme and the sources banners ought to list. Testing with the intended user and field testing in different locations and seasons would assist in finding out the understandability and usefulness of the application. These activities are future work, and not reported as done for the project in progress.

XI. CONCLUSION

The ability to access both a farm's and the soil's records, receive crop recommendation through rules, as well as enjoy the support and guidance of the cultivation modules - and all within one application on the web - is what KrishiAI is all about. It is created with a front end developed by React and TypeScript, a back end written using the FastAPI web framework and data models created with SQLAlchemy, and is localised with SQLite. The submitted implementation demonstrates that there is no evidence of measured TabPFN performance, and that it is implemented using a rule based approach.

Thus, the paper illustrates the actual system and not reported verified machine learning results. The platform offers the possibility of Agricultural information organization and its recommendations and wider effects needs to be validated. If the dataset, implementation and measured results are sufficiently documented, then a future version can optionally include a verified machine-learning based model, and a formal evaluation pipeline.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

REFERENCES

[1] H. Hollmann et al., “Accurate predictions on small data with a tabular foundation model,” Nature, vol. 637, pp. 319–326,2025, doi: 10.1038/s41586-024-08328-6. This reference provides background on TabPFN; it is not evidence that TabPFN is implemented or evaluated in the present project.

[2] B. Dey, J. Ferdous, and R. Ahmed, “Machine learning based recommendation of agricultural and horticultural crop farmingin India under the regime of NPK, soil pH and three climatic variables,” Heliyon, vol. 10, no. 3, article e25112, 2024, doi: 10.1016/j.heliyon.2024.e25112.

[3] KrishiAI project source repository and accompanying project documentation supplied by the project author, includingbackend services, API endpoints, data models, frontend source files, and test files. Accessed for implementation description; no public repository URL was provided for this paper.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)