Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Intelligent Crop Recommendation System Including Plant-Based Diseases Detection

(Concise, technical, and outcome-oriented)

Rosy Pradhan
Assistant Professor, Department of Artificial Intelligence and Machine Learning,
St. John College of Engineering and Management,

Dinesh Joshi, Shivam Maurya, Rajat Gupta, Anirudh Pm
Student, Department of Artificial Intelligence and Machine Learning, St. John College of Engineering and Management,

Abstract - The economic and food security systems of developed countries, including India, are heavily dependent on agriculture which is mainly exposed to a set of challenges including climate variability, soil erosion and rapid outbreak of plant diseases epidemics. This paper presents the AgriGrow Analytics, an agricultural intelligence platform based on microservices that applies many machine learning and deep learning concepts to offer real-time, data-driven decision support. It comprises four main prediction modules, including; a random forest-based crop health classifier, Gradient Boosting ensemble crop recommender, an XGBoost regressor yield forecasting, and a Convolutional Neural Network (CNN) automated plant disease classifier using leaf images.

Conversational agronomic advice is available in a fifth module using the Google gemini 2.0 generative AI platform that incorporates a natural-language fallback engine. Classification algorithms such as Decision Tree, Gaussian Naive Bayes, Support Vector Machine (SVM), Logistic Regression, Random Forest, XGBoost and K-Nearest Neighbors(KNN) were then trained and benchmarked on a soil-environment dataset of 2200 records comprising 22 crop classes. The classification accuracy of Random Forest is the highest, 99.55. A CNN model of plant diseases was trained on 70,295 leaf images of 14 different plant species, 38 classes of disease and had a validation accuracy of over 95%. The execution of the system is done as an implementation of a FastAPI server with a Next.js 14 TypeScript dashboard and provides predictions on a RESTful API architecture. The viability and accurateness of this combined platform to sustainable, scalable, and available precision Agriculture has been shown experimentally.

***Index Terms* - Precision agriculture, crop recommendation, random forest, convolutional neural network, plant disease detection, xgboost, yield forecasting, generative AI, fastapi and microservices..**

I. INTRODUCTION

A. Background and Motivation

The agricultural sector of the world is experiencing convergence of a set of challenges like never before, a rising world population, which consumes more food, the increasing climatic change is changing the regional agro-climate zones, the soil is increasingly becoming unfertile because of over-cultivation, and the threat of spread of the plant diseases is ever increasing rapidly. Furthermore, almost half the rural population in India relies on agriculture to earn its main income; yet, loss of crops in a year based on improper choice of crops and even unknown plant disease would cost the country billions of dollars [1].

Conventional farming methods place a high premium on knowledge transmitted over the years through experience, which, though important, cannot withstand the current complexity of agriculture. When a farmer selects an incorrect crop to match a particular soil and weather conditions, he/she could face an almost complete production failure. Likewise, an unrecognized plant disease can cause the ruination of a field in a week when the crop is at its best condition. These facts propel the creation of machine-crafty, information based decision support systems that have the capacity to convert the years of agricultural research into actionable, surpassing actual guidance.

Recent developments in machine learning (ML) and deep learning (DL) have demonstrated that predictive settings, trained on environmental and spectral information, can be surveyed or surpass expert agronomists with numerous diagnostic problems [2]. Nevertheless, the current literature addresses these problems individually: crop recommendation systems are not including minor problems with detecting diseases, yield prediction models or tools are not taking into consideration crop health, and conversational advice is not often connected to empirical prediction backends.

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

B. Problem Statement

The main problem that is tackled in this work is that there has never been a unified, end-to-end, system in agricultural intelligence that is capable of performing crop health monitoring, crop selection recommendation, quantitative yield estimation, automated disease diagnosis and conversational agronomic advice, all, and all, in a single, production-ready system that a non-experts, end-user can access.

C. Contribution of This Work

This study offers the following main contributions:

1. Random Forest was also the most accurate at 99.55% in a competition of seven ML classifiers to predict soil-based crops.
2. A trained CNN network was able to over 95 percent validate its classification of plant diseases into 38 categories.
3. XGBoost regression model to forecast seasonal yields with historical state-level production data.
4. A microservices architecture links ML/DL inference backends with a TypeScript dashboard via a FastAPI RESTful layer.
5. Fallbracy engine (Continuous conversational agronomic assistance) integrates the use of rule-based fallback engine with integrated Google Gemini 2.0 generative AI.

The other parts of the work are planned to be presented in the following way: Section II will discuss pertinent literature. Section III shows a gap in research. Section IV discusses the system methodology. In section V, I elaborate on the system architecture. Section VI presents mathematical model. The experimental set-up is described in Section VII. Section VIII discusses the outcomes. In Section IX, future directions are summarized. A list of references is provided in Section X..

II. LITERATURE REVIEW

The advent of sensor technology, remote sensing and predictive modeling have come together to mature precision farming as a topic. In this section, the researcher examines the basic and modern sources that influenced the design of the AgriGrow Analytics.

A. Machine Learning for Crop Prediction

In early crop recommendation systems, agronomist knowledge was captured in rule-based expert systems so as to represent it as conditional logic trees. Although interpretable, these systems could not be used to generalize a wide range of situations in different regions. The transition to statistical machine learning led to flexible models, which have the capability of learning through observational data.

Rao and others [1] found that the en-block strategies consistently outperform individual model classifiers in crop prediction, with Decision Tree achieving 85-90 percent and random forensic reaching 97-99 percent based on mix of datasets. Pande et al. [10] trained a crop recommendation based on soil data of N-P-K soil data on Support Vector Machines and Naive Bayes. They found that high-dimensional feature spaces could not be modeled by linear models and instead were preferable to kernel-based and probabilistic models. Raju et al. [11] introduced CropCast, which is a stacking-based ensemble of many base learners through a meta-classifier which provides greater accuracy than all other methods since it utilizes the diversity of models. Similar results indicated that deep learning preprocessing layers provided with conventional ML were more effective in generalization on noisy agricultural data sets used by Jhajharia et al. [8].

B. Plant Disease Detection Using Deep Learning

Disease assessment of crop leaves using visual inspection is laborious and subject to fallibility. Deep learning, and in particular CNNs, has become the most trendy method of automating this process. Islam et al. [2] introduced DeepCrop, an example of a CNN-based disease prediction system, linked with a web application, and with an accuracy of more than 93 percent on a multi class datasets of plant diseases. In a comprehensive performance evaluation of AI-based solutions to crop disease, Tirkey et al. [5] determined that CNN-based models which utilized transfer learning through ImageNet pre-training had better performance on leaf disease datasets compared to shallow classifiers.

Bouguettaya et al. [7] experimented with deep learning algorithms to identify agricultural diseases under aerial images, it was found that CNNs trained on the view of ground-level leaves underperform when deployed to aerial imagery, a result attributed to the fact that leaf-level image classifying should be conducted. Rani et al. [4] proposed a hybrid CNN model to predict crop and fertilizer diseases concurrently and discovered that learning two-tasks jointly enhanced the precision of separate-task predictions by sharing discovered feature representations. Chauhan et al. [12] explored deep learning-based plant disease diagnosis as a form of smart agriculture and the importance of increasing data augmentation to prevent overfitting to small field-based data collections.

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

C. Yield Forecasting

Ramu and Priyadarsini [6] compared the yield prediction algorithms and discovered that gradient boosting algorithms such as XGBoost are always effective in structured tabular data with environmental and fertilizer and past production variables. They have discovered that the ensemble tree models were better than the linear regression benchmarks in mean absolute error by 15-20% in multi-crop yield data.

D. Web-Based Agricultural Platforms

Pande and colleagues point at the importance of easy delivery mechanisms, with availabilityof advanced models in the backend not being very useful when the interface needs technicalknowledge on part of the end user [10]. Past systems using Streamlit or Flask were only demonstrations of functionality but not as scalable, type safe or fast as production-grade frameworks.

III. RESEARCH GAP

In spite of important individual achievements in the field of crop recommendation, disease detection as well as yield forecasting, there does still exist a gap in research that is evidently lacking at the level of system-integration.

Gap 1 — Fragmentation: Existing solutions focus just on one type of agricultural intelligence. There is no recorded production-ready system using multi-crop recommendation with probability scores, quantitative yield regression, and multi-class disease vision classification on a single backend with crop health monitoring.
Gap 2 - Feature Dimensionality: Past systems of crop recommendation used a reduced input feature space, limited to the traditional N-P-K-Temperature- humidityPH- Rainfall (7 features). AgriGrow adds 23 parameters such as NDVI (Normalised Difference Vegetation Index) and SAVI (Soil Adjusted Vegetation Index), the chlorophyll content and soil moisture; parameters which are highly predictive and yet have not been utilized in commercial version of the recommendation systems.

Gap 3 — Architecture and Scalability In the past, solutions were based on monolithic Streamlit or Flask apps, which did not support simultaneous prediction requests or autonomous modules. One way to overcome this limitation is through a microservices architecture consisting of a RESTful FastAPI implementation and typed API contracts.
Gap 4: Gap 4 Conversational AI Integration No prior farm intelligence solution.
has added a generative AI advisory layer that softly degrades to a rule-based fallback, so the user can still be served by the API smoothly even when it is unavailable.

Gap 5 User Experience: Manually entering numbers in the form is friction and error-prone. AgriGrow interface will eliminate this obstacle because the range-slider-based interface allows non-technical farm operators to use the technology.

IV. METHODOLOGY

A. Dataset Description

Dataset 1- Crop Recommendation: The data set on crop classification consists of 2,200 records and eight key variables: nitrogen (N), phosphorus (P), potassium (K), temperature (o C), humidity (percentage), pH, rainfall (mm) and label (target class). The 22 list of crop groups on the label include: rice, maize, chickpeas, kidney beans, pigeonpeas, mothbeans, mungbean, blackgram, lentil, pomegranate, banana, mango, grapes, watermelon, muskmelon, apple, orange, papaya, coconut, cotton, jute, and coffee. The NPK values represent observation of soil content, and the temperature, humidity and rainfall are the average climatic conditions in the area.
Dataset 2 - Plant Disease Detection: The data set of disease detection is 70,295 leaf images organized into 38 different classes that represent 14 plant species and 26 different disease types with healthy-plant classes added as well. Bilinear interpolation to 128x128 pixels details images. This data is structurally similar to the benchmark on PlantVillage except that it has additional disease types.
Long Crop Health Characteristics: Crop health prediction module involves the use of 23 features such as, NDVI, SAVI, chlorophyll content, soil moisture, speed of wind, and solar radiation based on the synthetic sensor and remote sensation distribution that are correlated with observed field range.

B. Data Preprocessing

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

Before training the model, the crop recommendation data set was normalized using StandardScaler at the feature level to zero-center/unit-variance-scale all numerical features. LabelEncoder was used to perform categorical label encoding. Missing values were not present and null-count validation was used to verify the dataset.

Preprocessing pipeline Plant disease Image preprocessing pipeline involved reading raw images in directory structure with tf.keras.utils.image dataset from directory, and then resizing the images to 128x128 pixels, normalizing pixel values to [0,1] interval by denormalizing with this value, and random pixel horizontal flips and rotation in training to help to reduce overfitting.

C. Train-Test Split

The dataset with the crop recommendations (2,200 records) were divided based on the scikit-learn train test split () with stratified (80/20) proportion (1,760 samples of the training and 440 of the testing). Stratification ensured that each of the 22 crop classes were represented fairly in both subsets.

This plant disease image data (70,295 photographs) was split into the training set (80% approximately 56,236 photos) and the validation set (20% approximately 14,059 photos) by splitting by directory.

D. Model Training

In terms of the crop recommendation model, there were seven classification algorithms developed – Decision Tree (criterion: Gini, max_depth: None), Gaussian Naïve Bayes, Support Vector Machine (kernel='rbf', C=1.0), Logistic Regression (solver='lbfgs', max_iter=1000), Random Forest (n_estimators=100, max_features=sqrt), XGBoost Classifier (n_estimators=100, learning_rate=0.1), and K-Nearest Neighbors (n_neighbors=5, metric=Euclidean). All models were created using the scikit-learn library and the joblib package, then exported in .pkl format.

TensorFlow/Keras was used to build the CNN model for disease detection. It comprised four consecutive Convolutional blocks (Conv2D -> BatchNormalization -> MaxPooling2D), followed by the Flatten operation, two Dense layers (256 and 128 units, activation=relu), one Dropout layer (rate=0.5) for regularization, and finally one Softmax output layer with 38 units. Model creation entailed the following parameters: Adam optimizer (learning_rate=0.001), categorical_crossentropy loss, and accuracy as the main metric. There were 10 epochs and 32 batches during training.

For the yield prediction function, the XGBoost Regressor was used. Historical data on state-level yields, including acreage, annual rainfall, fertilizers (kg/ha), pesticides (kg/ha), and categorical values for states and crops, was utilized.

V. SYSTEM ARCHITECTURE

AgriGrow Analytics will be built as a 3-level microservice application comprising of machine learning inference service backend, access gateway (RESTful API) and a client-facing dashboard app.

Tier 1 ML Inference Engine (Python/FastAPI): FastAPI is the implementation of the backend, and Uvicorn is the ASGI server. In the configuration, all model artifacts (.pkl files of RF, XGBoost, encoders, and scalers;.h5 files of CNN) are gradually loaded into RAM, and once this is done, it can run at sub-millisecond inference latency. All prediction modules also have dedicated POST endpoints, and Pydantic schema validation will impose input constraints before any model is called.

Tier 2 - API Gateway and Routing Layer: API endpoints are five (5) in support of the five system modules.:

- POST /api/predict-health → Random Forest health classifier
- POST /api/recommend-crop → Gradient Boosting recommender returning top-3 crops with probability scores
- POST /api/predict-yield → XGBoost regressor returning production in metric tons
- POST /api/detect-disease → CNN image classifier with argmax class selection
- POST /api/chat → Gemini 2.0 generative AI with rule-based fallback

Tier 3 - Frontend Dashboard (Next.js 14 / TypeScript): built on React 18 with Next.js App Router, Tailwind CSS and a custom Premium Dark Glassmorphism system of design. Every module will have its own URL (/crop-health, /crop-recommend, /yield-forecast, /disease-detect, /chatbot). Input parameters are collected using range sliders with dynamically validated boundaries derived directly from the ML pipeline's categorical transformation metadata, hence preventing schema mismatch errors. API calls are handled via a type-safe Axios client (lib/api.ts) that is mapped to Pydantic model schemas.

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

```
[User Interface: Next.js Dashboard]
        │ HTTPS REST
        ▼
[FastAPI Backend (Uvicorn)]
  ├── /predict-health   → Random Forest (.pkl)
  ├── /recommend-crop   → Gradient Boosting (.pkl)
  ├── /predict-yield    → XGBoost (.pkl)
  ├── /detect-disease   → CNN (.h5, TF/Keras)
```

VI. MATHEMATICAL MODEL

A. Evaluation Metrics

Let TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives for a given class in a multi-class classification issue eva *Type equation here.* luated using a one-vs-rest technique. Precision estimates the fraction of expected positive events that are truly positive.

$$Precision = TP / (TP + FP) \quad \ldots\ldots (1)$$

Recall (Sensitivity) measures the fraction of actual positives correctly identified:

$$Recall = TP / (TP + FN) \quad \ldots\ldots (2)$$

F1-Score is the harmonic mean of Precision and Recall, penalizing extreme imbalance between the two:

$$F_1 = 2 \times (Precision \times Recall) / (Precision + Recall) \quad \ldots\ldots (3)$$

Classification Accuracy repre*Type equation here.*sents the proportion of all predictions that are correct:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad \ldots\ldots (4)$$

B. Random Forest Ensemble Model

Random Forest creates an ensemble of B decision trees, each trained on a bootstrapped subset of the training data. The final class prediction is established by a majority vote.

$$\hat{y} = \text{mode}\ \{h_b(x) \mid b = 1, 2, \ldots, B\} \ \ldots\ldots\ldots.(5)$$

where hb(x)h_b(\mathbf{x}) denotes the prediction of the b -th tree on input feature vector x\mathbf{x} .

C. XGBoost Yield Regression

XGBoost minimizes a regularized objective function combining a differentiable loss and a complexity penalty:

$$\mathscr{L}(\phi) = \Sigma_i\, l(y_i, \hat{y}_i) + \Sigma_k\, \Omega(f_k) \quad \ldots\ldots\ldots\ldots.(6)$$

where ll is the squared error loss for regression, y^i=∑kfk(xi)\hat{y}_i = \sum_k f_k(\mathbf{x}_i) is the additive ensemble prediction, and Ω(fk)=γT+12λ‖w‖2\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|\mathbf{w}\|^2 penalizes tree complexity (leaf count TT and leaf weight magnitude w\mathbf{w}).

D. CNN Softmax Output

The CNN's final layer applies the Softmax function to produce a probability distribution over 38 disease classes:

$$P(\hat{y} = c \mid x) = e^\wedge z^{c} / \Sigma^{\vee} e^\wedge z^{\vee} \quad \ldots\ldots (7)$$

The predicted class is selected as:

$$\hat{c} = argmax^{c}\ P(\hat{y} = c \mid x) \quad \ldots\ldots (8)$$

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

E. NDVI for Crop Health Assessment

The Normalized Difference Vegetation Index, which serves as an input feature for the crop health prediction, is calculated as follows:

$$NDVI = (\rho_{NIR} - \rho_{RED}) / (\rho_{NIR} + \rho_{RED}) \quad \ldots\ldots (9)$$

where ρNIR\rho_{\text{NIR}} and ρRED\rho_{\text{RED}} are the near-infrared and red band reflectance values respectively. NDVI values range from −1 to +1, with values above 0.5 typically indicating dense, healthy canopy.

VII. EXPERIMENTAL SETUP

All the model training and evaluation experiments were conducted within a development environment with the following criteria. The training of CNNs was done using Python 3.10, sci-kit-learn 1.3, XGBoost 1.7, and TensorFlow 2.13/Keras to operate the models quickly and with an NVIDIA GPU. The FastAPI server was run on Uvicorn on port 8000, and the frontend dashboard was run on Node.js 18 and Next.js 14 on port 3000. The API endpoints were tested through Swagger UI auto-documentation at /docs. The model artifacts were stored in a.pkl (scikit-learn/XGBoost) and.h5 (Keras CNN) file and were loaded at the start of the server.
In the CNN, the tf.keras.utils.image dataset from directory was used to construct dataset pipelines where automatic classification of the directory names is used as classifier, shuffle buffers, and prefetch optimization to enhance the throughput in training. Image processing was done by converting to BGR and then to RGB to enable OpenCV and scaling to [0,1]. We used Adam optimizer and default hyperparameters (1=0.9, 2=0.999) and a starting learning rate of 0.001. The training was 10 epochs long, and the validation loss was monitored early, and a patience of 3 epochs.

VIII. RESULTS AND DISCUSSION

A. Crop Recommendation — Classifier Comparison

Table I shows the classification accuracy of all seven algorithms for the 440-sample test split (20% of 2,200 data).

TABLE I: Classification Accuracy Comparison — Crop Recommendation Task

Algorithm	Accuracy (%)
Decision Tree	90.00
Gaussian Naïve Bayes	99.09
Support Vector Machine (SVM)	10.68
Logistic Regression	95.23
Random Forest	**99.55**
XGBoost Classifier	99.09
K-Nearest Neighbors (KNN)	97.50

Humanized Result
The low SVM accuracy (10.68) is unusual, and indicates that the default RBF kernel hyperparameters were not optimized to this feature distribution, or that feature scaling was not used before training SVM in this setup as kernel methods are known to be highly sensitive to input scaling. SVM performance will probably enhance significantly with a proper StandardScaler preprocessing. Random Forest achieves the highest accuracy of 99.55 which is in line with its strength of feature scale and shrinkage method by an ensemble. Gaussian Naivete Bayes and XGBoost have 99.09% accuracy, which shows the efficiency of probabilistic models and gradient-boosted trees in this classification problem. The accuracy of the Decision Tree is good, 90.00% and it illustrates the strength of tree-based splitting on categorical agricultural data.

The comparison of all seven algorithms as a bar chart is shown in Fig. 2 (mentioned in the original research), which indicates the superiority of the Random Forest. Figure 3 shows the per-crop accuracy of the Random Forest model and it is evident that the model has high generalizability with minimal inter-class volatility; it gives high accuracy in all 22 crop classes.
B. CNN Plant Disease Detection.
Using CNN trained on 70,295 pictures of illnesses in 38 classes yielded a validation accuracy of greater than 95 percent and a training accuracy of 97 percent by epoch 8. The training and validation loss curves (Fig. 5) depict a steady decrease with a small train-validation loss, and there is no severe overfitting of the model by the Dropout layer (rate=0.5) and data augmentation. The

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

probability distribution of class produced by the Softmax output layer can present the user with the projected disease class, as well as the relative confidence by the model, which can be used to support clinical-grade decisions.

TABLE II: CNN Plant Disease Detection — Model Summary

Parameter	Value
Input Image Size	128 × 128 × 3
Number of Disease Classes	38
Plant Species Covered	14
Total Training Images	~56,236
Total Validation Images	~14,059
Convolutional Blocks	4
Dropout Rate	0.5
Optimizer	Adam (lr=0.001)
Training Epochs	10
Validation Accuracy	>95%

C. System-Level Performance

TABLE III: System Module Overview and Technology Mapping

Module	Algorithm	Input Features	Output
Crop Health Predictor	Random Forest Classifier	23 (NDVI, SAVI, NPK, Weather, etc.)	Healthy / Stressed
Crop Recommender	Gradient Boosting Ensemble	7 (N, P, K, Temp, Humidity, pH, Rainfall)	Top-3 Crops + Confidence
Yield Forecaster	XGBoost Regressor	State, Crop, Area, Rainfall, Fertilizer	Yield (metric tons)
Disease Detector	CNN (TF/Keras)	Leaf Image (128×128)	Disease Class + Probability
GenAI Agronomist	Gemini 2.0 + Fallback NLP	Natural Language Query	Agronomic Advisory Text

D. Discussion

In terms of their applicability, it was revealed that tree ensembles, like Random Forest, are the more suitable options when performing classification based on soil parameters data, as they exhibit nearly 100% accuracy in an efficient way during inference time. Moreover, CNNs' ability to perform plant diseases recognition outperforms all other attempts [2][5] likely due to the increased dataset and the presence of the Dropout layer. As mentioned above, microservice architecture allows each component to be easily scaled, updated, or even replaced by another one without any influence on the rest of the system. The presented generative AI application for agriculture with two operating modes proves that the integration of large language models into other specific applications can also take place.

Undoubtedly, the biggest issue that has been discovered concerns the low performance of the SVM classifier before kernel tuning, which shows the importance of hyperparameter tuning algorithms' implementation. Therefore, future iterations should involve approaches like GridSearchCV or Optuna.

IX. CONCLUSION AND FUTURE SCOPE

In this research, we propose AgriGrow Analytics, a complete agriculture intelligence system suitable for deployment in production. It combines ensemble learning models, computer vision techniques using deep learning, gradient boosting regression algorithms, and generative AI techniques using microservice architecture. While evaluating seven classifiers on a soil environment dataset of 2,200 records, we found that Random Forest provided the best accuracy of 99.55% in recommending crops. In disease recognition, a customized convolutional neural network on a disease database of 70,295 images across 38 disease categories attained a validation

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

accuracy greater than 95%. This system incorporates FastAPI backend and Next.js 14 frontend and offers a scalable and secure user interface suitable for implementation in poor-resource farming societies.

Further, the following are some of the areas considered for future implementations: (i) inclusion of multi-spectral/hyper-spectral images using drones/satellites to add more capabilities into the crop health analysis feature set; (ii) use of reinforcement learning techniques to improve crop rotation and resource allocation advice for multiple years; (iii) use of TensorFlow Lite to implement edge inferencing, such that mobile phones with field camera will be able to do prediction offline; (iv) integration of weather APIs and providing recommendations according to weather predictions; (v) use of transfer learning on EfficientNetV2 or Vision transformers to expand the scope of disease detection model; (vi) building of collaborative/crowdsourced data layer.

For the farmer to earn the maximum income per acre, there is need for consideration of prices and costs associated with inputs along with subsidies.

REFERENCES

[1] M. Shripathi Rao, A. Singh, N. V. Subba Reddy, and D. U. Acharya, "Crop prediction using machine learning," *Journal of Physics: Conference Series*, vol. 2161, no. 1, p. 012033, Jan. 2022.

[2] Md. Manowarul Islam et al., "DeepCrop: Deep learning-based crop disease prediction with web application," *Journal of Agriculture and Food Research*, vol. 14, pp. 100764, Dec. 2023.

[3] A. Kar, N. Nath, U. Kemprai, and Aman, "Performance Analysis of Support Vector Machine (SVM) on Challenging Datasets for Forest Fire Detection," *International Journal of Communications, Network and System Sciences*, vol. 17, no. 2, pp. 11–29, Feb. 2024.

[4] G. Rani, E. T. Venkatesh, K. Balaji et al., "An automated prediction of crop and fertilizer disease using Convolutional Neural Networks (CNN)," *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Apr. 2022.

[5] D. Tirkey, K. K. Singh, and S. Tripathi, "Performance analysis of AI-based solutions for crop disease identification, detection, and classification," *Smart Agricultural Technology*, vol. 5, p. 100238, Oct. 2023.

[6] K. Ramu and K. Priyadarsini, "A Review on Crop Yield Prediction Using Machine Learning Methods," *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, Oct. 2021.

[7] A. Bouguettaya, H. Zarzour, A. Kechida, and A. M. Taberkit, "A survey on deep learning-based identification of plant and crop diseases from UAV-based aerial images," *Cluster Computing*, Aug. 2022.

[8] K. Jhajharia, P. Mathur, S. Jain, and S. Nijhawan, "Crop Yield Prediction using Machine Learning and Deep Learning Techniques," *Procedia Computer Science*, vol. 218, pp. 406–417, 2023.

[9] M. Shripathi Rao, A. Singh, N. V. Subba Reddy, and D. U. Acharya, "Crop prediction using machine learning," *Journal of Physics: Conference Series*, vol. 2161, no. 1, p. 012033, Jan. 2022.

[10] S. M. Pande, P. K. Ramesh, A. Anmol, B. R. Aishwarya, K. Rohilla, and K. Shaurya, "Crop Recommender System Using Machine Learning Approach," *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Apr. 2021.

[11] C. Raju, A. D. V., and A. P. B. V., "CropCast: Harvesting the future with interfused machine learning and advanced stacking ensemble for precise crop prediction," *Kuwait Journal of Science*, p. 100160, Dec. 2023.

[12] P. Chauhan, H. L. Mandoria, A. Negi, and R. S. Rajput, "Plant Diseases Concept in Smart Agriculture Using Deep Learning," *Advances in Environmental Engineering and Green Technologies Book Series*, pp. 139–153, Oct. 2020.

[13] R. Gupta, S. Maurya, D. Joshi, and A. PM, "AgriGrow Analytics — Advanced Agricultural Intelligence Platform," GitHub Repository, 2024. [Online]. Available: https://github.com/rajatrgupta/AgriGrow

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