Green Intelligent Systems and Applications

Research Article
Volume 3 (2), 2023, 86–97
https://doi.org/10.53623/gisa.v3i2.313

Utilization of Machine Learning Algorithms for Precision Agriculture: Enhancing Crop Selection

Suhas Kakade[1]*, Rohan Kulkarni[2], Somesh Dhawale[3], Muhammed Fasil C[4]

[1]Department of Electrical Engineering COEP Tech. University, Pune, India.
[2]Department of Electrical Engineering, KJCOEMR, Pune, Maharashtra, India.
[3]Department of Electrical Engineering COEP Tech. University, Pune, Maharashtra, India.
[4]Department of Applied Electronics and Instrumentation Engineering, GEC Kozhikode, Kerala, India.

*Correspondence: smk.elec@coeptech.ac.in

SUBMITTED: 13 September 2023; REVISED: 17 October 2023; ACCEPTED: 21 October 2023

ABSTRACT: Agriculture stands as a crucial economic driver, playing a pivotal role in fostering economic progress. Understanding the dynamics of the agricultural system is imperative for ensuring food security. Even as technological strides like vertical farming emerge, conventional farming practices and beliefs continue to hold sway. This study delves into fundamental aspects such as soil composition, pH levels, humidity, and rainfall, employing a range of machine learning models including kernel naive Bayes, Gaussian naive Bayes, linear support vector machine (SVM), quadratic discriminant analysis, and quadratic SVM. The primary objective is to provide insightful crop recommendations to farmers. Accurate crop forecasting is paramount for optimizing agricultural methodologies and maintaining a consistent food supply. By leveraging historical weather trends, soil quality, and crop production data, machine learning algorithms proficiently anticipate crop yields. The outcomes of this investigation have the potential to refine crop management practices and reinforce food security measures.

KEYWORDS: Machine learning; agriculture; crop prediction; soil quality; classification learner algorithm.

1. Introduction

Agriculture plays an indispensable role in sustaining human life by generating vital resources such as food, fiber, and fuel. The projected global population of 9.7 billion by 2050 underscores the urgency to significantly augment food production. Precise crop forecasting, anchored in soil quality assessment, becomes pivotal for fine-tuning agricultural strategies, guaranteeing food security, and mitigating economic setbacks stemming from crop loss. Conventional approaches to crop prediction are burdened by lengthy processes, labor intensiveness, and susceptibility to inaccuracy due to the unpredictability of climatic conditions and other ecological variables. The application of machine learning algorithms in agriculture has gained considerable traction in recent times. These algorithms proficiently anticipate crop yields by swiftly and accurately analyzing extensive datasets comprising historical weather patterns, soil quality, and crop production data. This analytical prowess aids farmers in enhancing their crop

management practices. Numerous research endeavors have indicated that these algorithms hold promise for amplifying production and operational efficiency within the agricultural domain.

This research paper directs its focus towards the integration of machine learning within the realm of agriculture, particularly in predicting crops based on soil attributes. Despite the strides in agricultural technology, conventional methods and beliefs still wield considerable influence within the agricultural sector. The precision of crop prediction remains paramount for the optimization of farming techniques and the assurance of a consistent food supply. This study thoroughly examines pivotal factors encompassing soil composition, pH levels, humidity, and rainfall. A diverse array of machine learning models, including kernel naive Bayes, Gaussian naive Bayes, linear SVM, quadratic discriminant analysis, and quadratic SVM, are presented to analyze these aspects. The comparative assessment of these algorithms underscores the remarkable performance of kernel naive Bayes and Gaussian naive Bayes in the domain of crop prediction. Capitalizing on the insights gleaned from these algorithmic outcomes, the paper advocates for precise crop recommendations to be provided to farmers. Furthermore, the research delves into the potential implications of these algorithms in refining crop management practices and fortifying food security measures. The approach places significant emphasis on harnessing historical weather trends, soil quality, and crop production data to derive meticulous forecasts of forthcoming crop yields.

2. Literature Survey

Sharma et al. [1] implemented several machine learning models, such as decision tree, Gaussian Naive Bayes, logistic regression, random forest, and XGBoost, to recommend the appropriate crop based on soil composition, pH value, humidity, and rainfall. However, they did not utilize temperature as an input parameter, which affects plant growth. Ngozi Clara and Eli-Chukwu [2] discussed the uncertainties and challenges of farming, emphasizing the importance of soil, crop, disease, and weed management. The authors also highlighted the potential of artificial intelligence (AI) in addressing these issues. Aruna Devi et al. [3] elaborated on a study that uses machine learning techniques to select and predict crops based on agricultural parameters. They used the random forest algorithm as a classifier and predictor, outperforming support vector machine (SVM) and multivariate regression in terms of accuracy.

Vamsi Krishna et al. [4] designed a method based on five features: season, area, temperature, rainfall, and crop name. They obtained an accuracy of 67% for the linear regression technique, 75% for the random forest algorithm, and 97% for the k-nearest neighbor approach. Janhavi Babber et al. [5] developed a system that uses the WEKA tool to analyze soil, considering variables like temperature and humidity to forecast the most lucrative crops. The goal is to determine the best crop to grow in each soil type to maximize yield. The study highlights the potential of using machine learning tools and techniques to optimize agricultural practices and improve crop management. Apoorva Chaudhari et al. [6] employed data mining techniques and web scraping to create a platform for crop suggestion and optimal pricing, aiming to benefit farmers by providing recommendations for crop selection and optimal purchasing of crop seeds.

Pande et al. [7] designed a system that discussed the limitations of current yield prediction systems and proposed a mobile application for farmers. Utilizing different machine learning methods like Random Forest, Artificial neural networks (ANN), SVM, multiple linear regression (MLR), and K-Nearest Neighbors (KNN), it has a predictor and recommender

system. According to the results, Random Forest has a 95% accuracy rate. The model also suggests timely usage of fertilizers.

Sabbir Ahmed et al. [8] constructed a system in which data was pre-processed from a Bangla book series on land and soil resources. The resulting machine-learning model achieved 97% accuracy in crop yield prediction. Sachin D. Shingade et al. [9] proposed a system that uses machine learning and data analytics to predict the best agricultural seed. The results, with an accuracy of 95.12%, demonstrate that the random forest classifier is the most successful in the suggestion. M. Chandraprabha et al. [10] designed a system that analyzes soil, crop codes, and production data to predict suitable soils for rice, cumbu, and raagi crops. Ibk, bayes net, naive bayes, and random forest algorithms are used, and Ibk and random forest show the highest accuracy of 97.31% and 97.4%, respectively. It is suggested that ensemble techniques could improve accuracy for larger datasets.

Numerous researchers have leveraged machine learning (ML) techniques for crop selection prediction, as evidenced by a range of studies [12–15]. In addition, several papers have delved into crop yield estimation through the utilization of ML algorithms [16–20], aiming to enhance accuracy. This paper is centered on conducting a comparative analysis of diverse ML algorithms employed in the context of crop selection.

3. Proposed System

By considering geographical attributes including rainfall, humidity, soil pH, and the ratios of nitrogen, phosphorus, and potassium, this study employs a comprehensive set of ten distinct machine learning algorithms: kernel naive Bayes, Gaussian naive Bayes, linear SVM, quadratic discriminant analysis, quadratic SVM, SVM kernel, coarse tree, fine Gaussian SVM, coarse KNN, ensemble-RUS Boosted trees, and medium trees. The primary objective is to determine suitable crop choices for cultivation on various farms. The algorithms undergo a thorough evaluation, encompassing key metrics such as accuracy, error rate, training time, and execution time. The overarching aim of this framework is to equip farmers with insightful guidance, enabling them to make informed decisions about crop selection based on the specific conditions of their individual farms.

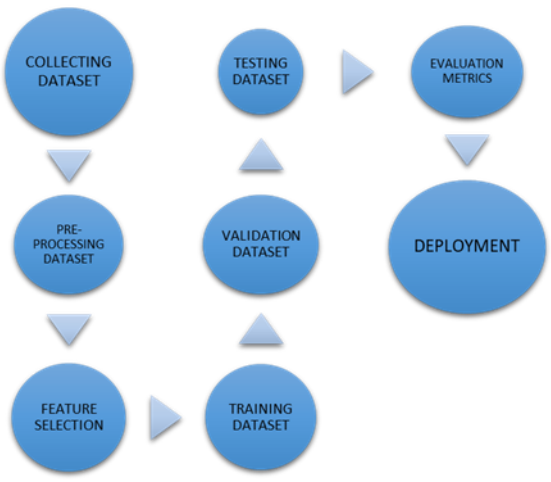

Figure 1. Flow diagram for the proposed system.

3.1. System setup.

The 'Classification Learner App' within MATLAB's Statistics and Machine Learning toolbox serves as the tool for crafting and evaluating the algorithms. By using this app, you can automatically train multiple supervised machine learning classifiers using input data and predefined groups. Leveraging MATLAB's prediction function, trained classifiers can seamlessly be integrated to make predictions for new input data. Notably, the app now works with 30 different types of classifiers in the latest version (R2022b) of MATLAB, giving you a huge range of options for analysis.

3.2. Data collection.

In this research, we utilized the dataset cited in [11]. This dataset was constructed to encompass 22 different crops, documented annually across various states. The data collection was facilitated through the installation of sensors on different farmlands. The dataset consisted of 2200 rows and eight columns, encompassing attributes like crop type, rainfall, humidity, pH level, potassium, nitrogen, and phosphorus, as demonstrated in Table 1. This data serves as the foundational material for training machine learning models tailored for predicting crop yields, offering valuable insights to aid farmers in making well-informed decisions.

Table 1. Snippet of crop prediction dataset [11].

	Croprecommendation							
	N	P	K	temperature	humidity	ph	rainfall	label
	Number	Number	Number	Number	Number	Number	Number	Categorical
1	N	P	K	temperature	humidity	ph	rainfall	label
2	90	42	43	20.87974371	82.00274423	6.502985292	202.9355362	rice
3	85	58	41	21.77046169	80.31964408	7.038096361	226.6555374	rice
4	60	55	44	23.00445915	82.3207629	7.840207144	263.9642476	rice
5	74	35	40	26.49109635	80.15836264	6.980400905	242.8640342	rice
6	78	42	42	20.13017482	81.60487287	7.628472891	262.7173405	rice
7	69	37	42	23.05804872	83.37011772	7.073453503	251.0549998	rice
8	69	55	38	22.70883798	82.63941394	5.70080568	271.3248604	rice
9	94	53	40	20.27774362	82.89408619	5.718627178	241.9741949	rice
10	89	54	38	24.51588066	83.5352163	6.685346424	230.4462359	rice
11	68	58	38	23.22397386	83.03322691	6.336253525	221.2091958	rice
12	91	53	40	26.52723513	81.41753846	5.386167788	264.6148697	rice
13	90	46	42	23.97898217	81.45061596	7.50283396	250.0832336	rice
14	78	58	44	26.80079604	80.88684822	5.108681786	284.4364567	rice
15	93	56	36	24.01497622	82.05687182	6.98435366	185.2773389	rice

3.3. Pre-processing step.

The initial phase of the implemented methodology centers on data pre-processing. This step entails addressing missing values, verifying accurate data formatting, and extracting essential features. The correct formatting of data is imperative for ensuring precise analysis. The insights collected during this stage will be imported into MATLAB to generate the intended results.

3.4. Feature extraction.

In the realm of machine learning, feature extraction plays a vital role in minimizing the data required to effectively represent a substantial dataset. Through the identification of the most influential predictive features, the overall scale of the training dataset can be diminished. These features commonly encompass soil attributes, crop variety, and atmospheric conditions in the

prediction of crop yields. The correlation matrix is often employed as a guiding tool for feature selection, with attributes displaying higher correlation values being frequently prioritized as potent yield predictors.

3.5. Data prediction.

The crop recommendation dataset is divided into two segments: 80% is allocated for training and validation, while the remaining 20% is reserved for testing. This partitioning is conducted randomly. MATLAB's Classification Learner App provides pre-defined validation methods to assess the effectiveness of the trained model's predictions. Among these strategies are "no validation", "holdout", and "cross-validation". The "no validation" approach employs all input data for training and employs the same data for constructing the confusion matrix. "holdout validation" divides the input data into two distinct sets: one for training the model and the other for validating it. Alternatively, in "cross-validation", the data is divided into subsets, with one set used for validation and the rest for training. The default option for validation is typically set to q = 5. Irrespective of the chosen validation method, the ultimate predictive model is trained using the complete dataset. The system incorporates a range of ten diverse machine-learning prediction models.

3.5.1. Gaussian naive Bayes (GNB).

Built upon the principles of Bayes' theorem, this method operates under the assumption of attribute independence. Through analysis of input data, the projected outcome for each class is computed, ultimately designating the class with the highest probability as the favored outcome. This algorithm is notable for its simplicity and efficiency, particularly excelling in scenarios involving high-dimensional data with continuous attributes. Nonetheless, its efficacy relies on the fulfillment of the feature independence assumption. When presented with a novel input example characterized by feature values *x = (x1, x2, ..., xn),* the Gaussian Naive Bayes (GNB) classifier evaluates the posterior probability of each class label *c*, based on the provided evidence *x*. This computation follows the prescribed equation:

$$p(c|x) = p(x|c) * p(c)/p(x) \ldots\ldots\ldots (1)$$

In this context, *p(c)* represents the prior probability of class *c*, while *p(x)* signifies the marginal likelihood of the evidence *x, a* constant across all classes. The posterior probability of class *c*, given the evidence *x*, is denoted as $p(c|x)$. The likelihood of the evidence *x* given class *c* is expressed as $p(x|c)$, and the likelihood of the evidence *x* given class *c* is also represented as $p(c|x)$.

3.5.2. Kernel naive Bayes.

This variation of the Naive Bayes algorithm employs kernel methods to address scenarios where classes are not linearly separable. It operates on the fundamental premise that, within a given class, features exhibit conditional independence. By utilizing a kernel function, the algorithm initiates by mapping input features to a higher-dimensional space. Subsequently, it calculates the likelihood of each class for a provided input vector, relying on the distances between the mapped points and the class centroids. This approach adeptly handles complex high-dimensional feature spaces and finds application in tasks like image and text

classification. In the context of a training dataset comprising *N* instances, each encompassing a *d*-dimensional input vector *x* and *a* corresponding class label *y*:

(a) Apply a kernel function *k* to the input vectors to map them into a higher-dimensional space:

$$\varphi(x) = [k(x, x_1), k(x, x_2), \ldots, k(x, x_N)] \ldots\ldots\ldots (2)$$

where *k (x, x_1)* is the kernel function applied to the input vector *x* and the i^{th} training example *x_i*. The steps to implement the algorithms are as follows:

(b) Estimate the class prior probabilities *p(y)* and the class-conditional probabilities *p(φ(x)|y)* using maximum likelihood estimation:

$$p(y) = \frac{N_y}{N}$$

$$p(\varphi(x)|y) = (1/N_y) * \Sigma i: y_i = y k(\varphi(x), \varphi(x_i)) \ldots\ldots\ldots (3)$$

where *Ny* is the number of training examples with class label *y*, and *Σ_{i:y_i=y}* denotes the sum of all training examples with class label *y*.

(c) Calculate the posterior probabilities for each class using Bayes' theorem:

$$p(y|x) = p(y) * p(\varphi(x)|y) / p(\varphi(x)) \ldots\ldots (4)$$

where *p(y)* is the prior probability of class *y*, *p(φ(x)|y)* is the class-conditional probability of the transformed input vector *φ(x)* given class *y*, and *p(φ(x))* is the marginal probability of the transformed input vector, which can be calculated as:

$$p(\varphi(x)) = \Sigma y p(\varphi(x)|y) * p(y) \ldots\ldots\ldots (5)$$

3.5.3. Linear SVM.

This distinct machine learning algorithm functions by identifying the most effective hyperplane to distinguish between various classes within a dataset. The optimal hyperplane is the one that maximizes the separation margin between the classes during classification. Subsequently, this hyperplane is employed to categorize new data points, assigning them to specific classes based on their position relative to the hyperplane. The descriptor "linear" in Linear SVM indicates the algorithm's assumption of a straight-line decision boundary between classes. The key equations underpinning the operations of a linear SVM are as follows:

Given a training set of *N* examples, each consisting of a *d*-dimensional input vector *x* and *a* corresponding binary class label *y (+1 or -1)*, the goal is to find a hyperplane with weights *w* and bias *b* that separates the positive and negative examples with the maximum margin:

$$w^T x_i + b \geq +1 \; if \; y_i = +1$$

$$w^T x_i + b \leq -1 \; if \; y_i = -1 \ldots\ldots (6)$$

3.5.4. Quadratic Discriminant Analysis (QDA).

It is like Linear Discriminant Analysis (LDA), but instead of assuming that all classes have the same covariance of predictors, it makes a different assumption. QDA models are fitted using maximum likelihood estimation, and a quadratic equation determines the decision boundary between classes. QDA can be helpful when the covariance matrices of the predictors are not equal across classes, and it can potentially result in better classification accuracy than LDA in such cases. The key equations for QDA are as follows:

(a) Given a training set of *N* examples, each consisting of a *d*-dimensional input vector *x* and a corresponding binary class label *y (+1 or -1)*, the goal is to estimate the class-conditional densities *p(x|y)* for each class using multivariate Gaussian distributions:

$$p(x|y=+1) = N(x|\mu_1, \Sigma_1)$$

$$p(x|y=-1) = N(x|\mu_2, \Sigma_2) \ldots\ldots\ldots (7)$$

where $N(x \mid \mu, \Sigma)$ denotes the probability density function of a *d*-dimensional Gaussian distribution with mean μ and covariance matrix Σ.

(b) The prior probabilities *p(y)* is estimated based on the relative frequency of the classes in the training set:

$$p(y=+1) = N_1/N$$

$$p(y=-1) = N_2/N \ldots\ldots\ldots\ldots\ldots\ldots (8)$$

where *N1* and *N2* are the numbers of positive and negative examples, respectively, and $N = N1 + N2$ is the total number of examples.

3.5.5. Quadratic SVM (QSVM).

This SVM algorithm maps data points into a higher-dimensional feature space using a quadratic kernel function. In this feature space, the decision boundary between different data classes is a quadratic curve instead of a straight line, as in linear SVM. This allows the QSVM to capture more complex patterns in the data and achieve higher classification accuracy, especially when dealing with non-linearly separable data. However, QSVM can be computationally expensive for large datasets, and the kernel function and parameter selection can significantly influence its performance. The key equations for QSVM are as follows:

Given a training set of *N* examples, each consisting of a d-dimensional input vector *x* and corresponding binary class label *y (+1 or -1)*, the goal is to find a quadratic decision boundary in the form:

$$f(x) = w^T x + b + x^T M x \ldots\ldots\ldots\ldots (9)$$

where *w* is a *d*-dimensional weight vector, *b* is a bias, and *M* is a *d x d* symmetric positive definite matrix.

3.5.6. Coarse tree.

A Coarse Tree approach simplifies decision tree structures by limiting their depth and complexity, favoring interpretability and overfitting avoidance. The strategy stops tree growth at a predetermined depth, resulting in larger leaf nodes for more generalized predictions. This emphasis on interpretability enhances the model's transparency, making it well-suited for extracting insights and generalizing to new data. However, the trade-off is a potential sacrifice in predictive accuracy on the training data, as intricate patterns might not be captured. Coarse Trees strike a balance between interpretability and predictive power, making them suitable when a clear understanding of the model's decisions is valued over maximal accuracy.

3.5.7. Fine Gaussian.

The Fine Gaussian approach extends the Gaussian Naive Bayes (GNB) algorithm by allowing for correlations between features using multivariate Gaussian distributions.

$$p(x \mid c) = \frac{1}{(2\pi)^{d/2}|\Sigma c|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu c)^T \Sigma_c^{-1}(x-\mu c)\right) \ldots\ldots\ldots (10)$$

Explanation:

- $p(x|c)$ represents the probability of observing feature vector x given class c.
- μc is the mean vector for class c.
- Σc is the covariance matrix for class c.
- $|\Sigma c|$ denotes the determinant of the covariance matrix.
- The equation uses the multivariate Gaussian distribution formula to compute the likelihood of x belonging to class c. This accounts for the correlations between features, allowing for more flexible modelling.

3.5.8. Coarse KNN.

Coarse KNN is a variation of the KNN algorithm designed to enhance efficiency. It achieves this by dividing the training data into clusters and focusing computations on the most relevant clusters for classifying new data. By doing so, it reduces the computational load while still providing reasonably accurate predictions, making it suitable for handling large datasets and scenarios where speed is crucial.

Distance Calculation:

$$\text{Distance}(xi, xj) = \sqrt{\sum_{k-1}^{d}(xi, k - xj, k)^2} \ldots\ldots\ldots\ldots (11)$$

Explanation:

- Distance Calculation: The distance between two data points xi and xj is calculated using the Euclidean distance formula. This helps determine how close or similar the points are in a d-dimensional space.
- Majority Voting: For classification, the class of a query point is decided by the majority class among its k nearest neighbours. The algorithm finds the k nearest neighbours based on their distances to the query point.

3.5.9. Ensemble-RUS boosted tree.

Ensemble-RUS Boosted Tree combines Random Under-Sampling (RUS) and Boosted Trees to address class imbalance while boosting predictive performance. The approach is designed to tackle scenarios where one class significantly outweighs the others. Initially, weak classifiers are set up and example weights are established. These weak classifiers, often decision trees, are trained using a weighted training set, giving more importance to misclassified examples. Random Under-Sampling (RUS) is then employed to reduce the number of majority class instances, creating a more balanced dataset. As the process iterates, weights are updated based on the misclassification rates of examples, giving more emphasis to harder-to-classify instances. Boosting ensues, with the ensemble combining the weak classifiers and granting more weight to those with lower misclassification rates. Ultimately, the ensemble's prediction is determined through weighted majority voting, producing a more robust prediction by leveraging the strengths of both RUS and Boosted Trees.

3.5.10. Medium tree.

A Medium Tree represents a type of decision tree that finds a middle ground between shallow and deep trees. It aims to capture a mix of intricate details and broader patterns in the data. Shallow trees are simple but might miss complex relationships, while deep trees can overfit and memorize noise. Medium trees strike a balance by capturing intermediate-level patterns, improving generalization to new data while still preserving relevant information. This approach is valuable when seeking a trade-off between complexity and interpretability, and it often requires experimentation to determine the optimal tree depth.

$$\text{Impurity(node)} = 1 - \sum_{i-1}^{K} {p_i}^2 \ldots\ldots\ldots (12)$$

Explanation:

- Impurity Measure: The equation represents the impurity of a node in the decision tree. It quantifies how mixed the class labels are within the node. A lower impurity indicates that the node contains predominantly examples from a single class.
- K is the number of classes.
- p_i is the proportion of examples belonging to class i in the node.
- Decision: The tree splits data by selecting the feature and threshold that minimize impurity across child nodes. Medium Trees aim to strike a balance between overly simplistic and overly complex trees.

4. Result and Analysis

After each model has been trained, it is evaluated by feeding the test dataset to it and comparing its output with the respective labels. Various metrics, such as accuracy and ROC (Receiver Operating Characteristic) curve, are used to compare the models. The ROC curve illustrates the trade-off between true and false positive rates for various threshold values. Accuracy is defined as the proportion of adequately predicted instances among all instances. The performance of the models can be assessed and contrasted by evaluating these metrics. All the models were evaluated, and their obtained accuracies are mentioned in Table 2.

Table 2. Experimental outcomes.

Sr No	Algorithm	Accuracy (validation)	Total cost (validation)	Prediction speed (obs/sec)	Training time (sec)
1	Gaussian Naive Bayes	99.5	2	28000	0.75635
2	Kernel Naive Bayes	99.5	1	5400	1.1389
3	Linear SVM	99.3	5	3800	6.9315
4	Quadratic Discriminant	99.3	3	59000	1.1792
5	Quadratic SVM	99.3	6	1900	5.4965
6	Coarse tree	22.7	340	170000	0.42124
7	Fine Gaussian	77	84	1200	6.2402
8	Coarse KNN	77.3	103	13000	1.1464
9	Ensemble-RUS Boosted tree	85.7	42	9100	2.9227
10	Medium tree	85.7	42	220000	1.3979

Table 2 shows the performance of ten different machine learning algorithms on a specific dataset. Each algorithm is ranked based on its accuracy on the validation dataset, which is the percentage of correct predictions made by the model. The table also includes information on the model's total cost on the validation dataset, which includes factors such as computational resources or time required to train the model. The prediction speed of each algorithm is measured in observations per second, which indicates how quickly the model can make predictions on new data once it has been trained. Finally, the training time for each algorithm is also included, which measures the time required to train the model on the dataset. By comparing these parameters for each algorithm, one can determine the best-suited algorithm for crop prediction. From Table 2, gaussian naive bayes gives the optimum solution for the used dataset.

Table 3. Comparison of different algorithm's accuracy used on a similar dataset.

Sr. No.	Algorithms	Accuracy	Ref.
1	Linear Regression	67%	[4]
2	Random Forest	75%	[4]
3	k-Nearest Neighbor	97%	[4]
4	Random Forest	95%	[7]
5	Random Forest Classifier	95.12%	[9]
6	Instance-based Learner (Ibk)	97.31%	[10]
7	Random Forest	97.4%	[10]
8	Gaussian Naive Bayes	99.5%	This paper
9	Kernel Naive Bayes	99.5%	This paper

From Table 2, it is observed that kernel naive bayes, gaussian naive bayes, linear SVM, quadratic discriminant, and quadratic SVM showed superior accuracy; while other models, such as coarse KNN, fine Gaussian SVM and coarse tree performed poorly. Figures 3 and 4 show the confusion matrices for Gaussian naive bayes and kernel naive bayes respectively. Table 3 depicts the comparison of various algorithms applied on a similar dataset. It shows that the ML techniques suggested by us have higher accuracy than the techniques used in the literature.

5. Conclusions

Based on several variables like soil type, and weather, this system employs various machine learning algorithms to assist farmers in selecting the crops to grow. The results show that it is an effective tool for improving agricultural productivity and profitability. We achieved 99.5% accuracy as compared to the results stated in the literature. This system allows farmers to optimize their yields and profitability by making data-driven decisions about what crops to plant. The application of ML-based systems in agriculture has the potential to transform the

industry and contribute to supplying the world's expanding food needs. The system enables the planting of crops in a way that maximizes the use of soil components while minimizing the wastage of resources.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Sharma, A.; Bhargava, M.; Khanna, A.V. (2021). AI-Farm: A crop recommendation system. International Conference on Advances in Computing and Communications (ICACC), pp. 1–7. https://doi.org/10.1109/ICACC-202152719.2021.9708104.

[2] Eli-Chukwu, N.C. (2019). Applications of artificial intelligence in agriculture: A review. Engineering, Technology & Applied Science Research, 9, 4377–4383. https://doi.org/10.48084/etasr.2756.

[3] Devi, M.A.; Suresh, D.; Jeyakumar, D.; Swamydoss, D.; Lilly Florence, M. (2022). Agriculture Crop Selection and Yield Prediction using Machine Learning Algorithms. Second International Conference on AI and Smart Energy (ICAIS), pp. 510–517. https://doi.org/10.1109/ICAIS53314.2022.9742846.

[4] Krishna, V.; Reddy, T.; Harsha, S.; Ramar, K.; Hariharan, S.; Bhanuprasad, A (2022). Analysis of Crop Yield Prediction using Machine Learning Algorithms. 2nd International Conference on Innovative Sustainable Computational Technologies (CISCT), pp. 1–4. https://doi.org/10.1109/CISCT55310.2022.10046581.

[5] Babber, J.; Malik, P.; Mittal, V.; Purohit, K.C. (2022). Analyzing Supervised Learning Algorithms for Crop Prediction and Soil Quality. 6th International Conference on Computing Methodologies and Communication (ICCMC), pp. 969–973. https://doi.org/10.1109/ICCMC53470.2022.9753798.

[6] Chaudhari, A.; Beldar, M.; Dichwalkar, R.; Dholay, S. (2020). Crop recommendation and its optimal pricing using shopbot. International Conference on Smart Electronics and Communication (ICOSEC), pp. 36–41. https://doi.org/10.1109/ICOSEC49089.2020.9215411.

[7] Pande, S.M.; Ramesh, P.K.A.; Aishwarya, B.R.; Rohilla, K.; Shaurya, K. (2021). Crop recommender system using machine learning approach. 5th International Conference on Computing Methodologies and Communication (ICCMC), pp. 1066–1071. https://doi.org/10.1109/ICCMC51019.2021.9418351.

[8] Ahmed, S.; Yesmin, S.; Saha, L.R.; Sadat, A.K.M.; Khan, M.H.A. (2022). Major Cropping Pattern Prediction in Bangladesh from Land, Soil and Climate Data Using Machine Learning Techniques. 25th International Conference on Computer and Information Technology (ICCIT), pp. 988–993. https://doi.org/10.1109/ICCIT57492.2022.10056051.

[9] Shingade, S.D.; Mudhalwadkar, R.P.; Masal, K.M. (2022). Random Forest, DT and SVM Machine Learning Classifiers for Seed with Advanced WSN Sensor Node. International Conference on Automation, Computing and Renewable Systems (ICACRS), pp. 321–326. https://doi.org/10.1109/ICACRS55517.2022.10029310.

[10] Chandraprabha, M.; Dhanaraj, R.K. (2021). Soil Based Prediction for Crop Yield using Predictive Analytics. 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 265–270. https://doi.org/10.1109/ICAC3N53548.2021.9725758.

[11] Kalimuthu, M.; Vaishnavi, P.; Kishore, M. (2020). Crop prediction using machine learning. Third International Conference on Smart Systems and Inventive Technology (ICSSIT), pp. 926–932. https://doi.org/10.1109/ICSSIT48917.2020.9214190.

[12] Vanarase, V.; Mane, V.; Bhute, H.; Tate, A.; Dhar, S. (2021). Crop Prediction Using Data Mining and Machine Learning Techniques. Third International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1764–1771. https://doi.org/10.1109/ICIRCA51532.2021.9544724.

[13] Somasundaram, R.S.; Nagamani, K.; Swamydoss, D. (2022). Estimation and Prediction of Crop Yielding Rate Using Machine Learning Techniques. International Conference on Computer Communication and Informatics (ICCCI), pp. 1–5. https://doi.org/10.1109/ICCCI54379.2022.9741032.

[14] Devikarani, A.; Jyothi, G.V.D.; Sri, J.D.L.; Devi, B.S.K. (2022). Towards Smart Agriculture using Machine Learning Algorithms. International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), pp. 837–842. https://doi.org/10.1109/ICCCIS56430.2022.10037619.

[15] Vashisht, S.; Kumar, P.; Trivedi, M.C. (2022). Improvised Extreme Learning Machine for Crop Yield Prediction. 3rd International Conference on Intelligent Engineering and Management (ICIEM), pp. 754–757. https://doi.org/10.1109/ICIEM54221.2022.9853054.

[16] Elavarasan, D.; Vincent, P.M.D. (2020). Crop yield prediction using deep reinforcement learning model for sustainable agrarian applications. IEEE Access, 8, 86886–86901. https://doi.org/10.1109/ACCESS.2020.2992480.

[17] Ranjani, J.; Kalaiselvi, V.K.G.; Sheela, A.; Janaki, G. (2021). Crop Yield Prediction Using Machine Learning Algorithm. 4th International Conference on Computing and Communications Technologies (ICCCT), pp. 611–616. https://doi.org/10.1109/ICCCT53315.2021.9711853.

[18] Sharma, A.; Tamrakar, A.; Dewasi, S.; Naik, N.S. (2022). Early Prediction of Crop Yield in India using Machine Learning. IEEE Region 10 Symposium (TENSYMP), pp. 1–6. https://doi.org/10.1109/TENSYMP54529.2022.9864490.

[19] Paudel, S.; Nakarmi, R.; Giri, P.; Karki, S.B. (2022). Prediction of Crop Yield Based-on Soil Moisture using Machine Learning Algorithms. 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS), pp. 491–495. https://doi.org/10.1109/ICTACS56270.2022.9988186.