A Study on Enhancing Crop Yield Prediction with Accuracy through Machine and Deep Learning Models

J. Karthikeyan[1], Dr. A. Murugan[2]

[1]*Research Scholar, Department of Computer and Information Science, Annamalai University, Annamalainagar – 608 002, Tamil Nadu, India, thalapathik80@gmail.com*
[2]*Assistant Professor, Department of Computer Science, Periyar Arts College, Cuddalore, (Deputed from Annamalai University, Annamalainagar) Tamil Nadu, India, drmuruganapcs@gmail.com*

KEYWORDS
Crop Yield, Machine Learning, Deep Learning, Agricultural Parameters, Predictive Modeling, And Model Performance Analysis.

ABSTRACT
Accurate crop yield prediction is essential for optimizing agricultural resource allocation and supporting farmers in making informed crop management decisions. This study investigates the effectiveness of various machine learning and deep learning models for predicting crop yields based on key agricultural parameters, including Nitrogen (N), Phosphorous (P), Potassium (K), temperature, humidity, pH, and rainfall. We apply multiple algorithms—Logistic Regression, Multilayer Perceptron, Sequential Minimal Optimization (SMO), J48, Random Forest, REP Tree, and a proposed deep learning model—to evaluate their predictive accuracy in classifying agricultural items. Each model's performance is assessed using metrics such as Kappa statistic, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), and the time taken to build the model. Findings reveal that while traditional machine learning models like Random Forest and REP Tree show robust performance, the proposed deep learning model demonstrates enhanced accuracy and efficiency, making it highly promising for crop yield prediction applications. This study underscores the potential of advanced learning algorithms to improve agricultural productivity by providing actionable insights for crop planning and resource management.

1. Introduction and Background

Machine learning models have emerged as powerful tools for agricultural applications, including yield prediction, crop classification, and disease detection, due to their ability to model complex, non-linear relationships in data (Jha et al., 2019). A range of machine learning algorithms has been applied to CYP, such as decision trees, support vector machines (SVM), and ensemble methods like Random Forest, each showing varying degrees of accuracy and reliability (Ali & Chaudhary, 2020). Agriculture, as a foundational component of global economies and food systems, is witnessing a shift toward more data-driven practices, driven by the rise of machine learning (ML) and artificial intelligence (AI). Traditional agricultural practices, reliant on historical data and heuristic-based forecasting, often fall short in accurately predicting crop yields due to the complex and dynamic nature of agricultural environments (Shinde & Shah, 2020).

By analyzing various environmental and soil parameters, such as temperature, humidity, nutrient levels, and rainfall, these models can forecast outcomes with precision, helping farmers optimize resources and reduce risks associated with unpredictable factors like weather and soil variability (Kamilaris & Prenafeta-Boldú, 2018). Studies show that integrating machine and deep learning in agriculture leads to significant improvements in crop management and sustainability. For instance, DL models have been successfully applied to identify crop diseases with high accuracy, assisting farmers in timely interventions and reducing crop loss (Zha et al., 2021).

Agriculture is a cornerstone of global economies, with crop yield playing a critical role in food security and economic stability. With an increasing global population and the rising need for sustainable agricultural practices, farmers are challenged to optimize yield within existing constraints of land, climate, and resources (Patil & Shirkhedkar, 2018). Crop yield prediction (CYP), which seeks to estimate production outcomes based on environmental and soil factors, has become essential for guiding farmers in planning and resource allocation. This need has led to the adoption of machine learning (ML) and deep learning (DL) technologies, which offer the potential for more accurate, data-driven predictions than traditional models based on historical averages (Kumar et al., 2019; Chlingaryan et al., 2018).

Modern CYP models leverage diverse input parameters such as soil nutrients (Nitrogen, Phosphorous, Potassium), climate factors (temperature, humidity, rainfall), and soil pH to generate yield predictions (Sarker et al., 2020). These parameters capture critical environmental conditions that influence crop growth, offering a more comprehensive basis for analysis. Existing studies demonstrate that machine learning techniques, including Random Forest, Logistic Regression, and decision tree models like J48 and REP Tree, are effective in modeling such complex interactions (Feng et al., 2019). However, recent advancements in deep learning, with architectures like multilayer perceptrons, have shown promising results in capturing nonlinear relationships in data, further enhancing predictive accuracy (Ramesh & Rao, 2021).

Nutrient availability and environmental conditions are fundamental to agricultural productivity, influencing crop growth, yield, and quality. Key nutrients such as Nitrogen (N), Phosphorous (P), and Potassium (K), often referred to as macronutrients, play essential roles in plant development. Nitrogen is crucial for photosynthesis and chlorophyll synthesis, and its deficiency can lead to stunted growth and reduced leaf production (Sarker et al., 2020). Environmental parameters such as temperature, humidity, pH, and rainfall further influence crop yield and quality by affecting physiological and biochemical processes within plants. Temperature, for instance, impacts growth rates and metabolic activity, with most crops having an optimal temperature range for maximum yield. Excessive heat or cold can hinder development or cause crop failure, especially during critical stages like flowering or fruiting (Lal et al., 2019). Humidity, which determines water vapor levels in the atmosphere, plays a critical role in transpiration and water transport within plants. High humidity levels can also increase the risk of fungal diseases, whereas low humidity may lead to dehydration and reduced growth rates (Ravi et al., 2020).

Analysis of different data mining algorithms implemented within the Weka tool. The authors focus on evaluating and comparing decision tree algorithms, such as J48, Random Tree, and REP Tree, in terms of their classification accuracy, speed, and efficiency in handling various datasets. Using the Weka software suite, the study analyzes how each algorithm performs with different datasets, aiming to identify the most suitable algorithm for specific types of data mining tasks (Rajesh and Karthikeyan, 2017). The performance of decision tree algorithms in analyzing chronic disease indicators. The study leverages various decision tree methods to classify and predict chronic disease outcomes based on CDI data, focusing on metrics such as classification accuracy, efficiency, and computational cost. Using decision tree algorithms like J48, CART (Classification and Regression Tree), and Random Forest, the authors assess each method's performance on chronic disease datasets (Rajesh et al., 2019). explore data mining techniques to identify key factors that influence agricultural development. The authors employ a stochastic model within a data mining framework to analyze agricultural data, aiming to predict factors that could enhance crop yield and productivity.

Evaluate the performance of several machine learning and deep learning models for crop yield prediction, specifically Logistic Regression, Multilayer Perceptron, SMO, J48, Random Forest, REP Tree, and a proposed deep learning model. The models are assessed using key performance metrics: Kappa statistic, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), and model training time. These performance indicators provide insights into the models' accuracy, reliability, and efficiency, facilitating comparison across diverse algorithms (Zhou et al., 2017).

By identifying the model with the best predictive performance, this study contributes to the field of precision agriculture, offering a robust framework for crop yield prediction that can assist farmers in optimizing their agricultural outputs. The findings underscore the potential of advanced machine learning and deep learning models to support efficient and sustainable farming practices, ultimately promoting food security and economic resilience.

2. Backgrounds and Methodologies

2.1. Logistic Regression

Logistic Regression is a supervised learning algorithm often used for binary classification. It predicts the probability that a given input belongs to a specific class, using the logistic function to map predictions between 0 and 1 (Hosmer et al., 2013). The process is as follows:

1. Data Preparation: Data is pre-processed, and features are normalized if needed.
2. Model Initialization: Initialize weights and bias parameters.
3. Hypothesis Calculation: Apply a linear combination of the input features and weights, followed by the logistic function to produce probabilities.
4. Cost Function: Use the binary cross-entropy loss to measure error in predictions.

5. Gradient Descent Optimization: Adjust weights iteratively by calculating the gradient of the cost function with respect to each parameter to minimize error.

6. Prediction: After training, use the model to classify new data points based on learned weights.

2.2. Multilayer Perceptron (MLP)

A Multilayer Perceptron (MLP) is a type of artificial neural network with multiple layers of nodes. It's well-suited for tasks that involve complex, non-linear relationships (Rosenblatt, 1958). Here's the step-by-step process:

1. Data Preprocessing: Normalize or scale input data.

2. Network Architecture: Define the number of layers, neurons per layer, activation functions (commonly ReLU for hidden layers and softmax or sigmoid for output).

3. Forward Propagation: Pass inputs through each layer; apply weights, biases, and activation functions to get the output.

4. Loss Calculation: Calculate the loss using cross-entropy or MSE (Mean Squared Error) for regression tasks.

5. Backpropagation: Compute gradients of the loss concerning each parameter using the chain rule.

6. Optimization: Use a gradient descent or similar algorithm to update weights iteratively to minimize the loss.

7. Prediction: After training, the network can classify or predict values for new data.

2.3. Sequential Minimal Optimization (SMO)

Sequential Minimal Optimization (SMO) is an efficient algorithm for training Support Vector Machines (SVMs). It simplifies the quadratic programming problem by breaking it down into smaller, manageable problems (Platt, 1998).

1. Data Preparation: Ensure data is labeled and, if necessary, standardized.

2. Initialization: Set initial Lagrange multipliers for each data point.

3. Decomposition: Select pairs of Lagrange multipliers to optimize at each step.

4. Update Multipliers: Solve for two selected multipliers by setting up and solving a simplified optimization problem for them.

5. Recalculate Parameters: Update the SVM parameters based on the optimized multipliers.

6. Repeat: Continue adjusting pairs of multipliers until convergence, i.e., until the SVM's decision boundary is optimally placed.

7. Prediction: Use the optimized decision boundary to classify new instances.

2.4. J48 (C4.5)

J48 is an implementation of the C4.5 decision tree algorithm, commonly used for classification tasks. It recursively splits the dataset into subsets based on the attribute that provides the best split, using information gain (Quinlan, 1993).

1. Data Preparation: Clean and preprocess data to handle missing values or categorical attributes.

2. Select Attribute: Choose the attribute with the highest information gain or lowest entropy as the decision node.

3. Split Data: Partition the data based on the selected attribute's values.

4. Recursive Splitting: Repeat the attribute selection and splitting process for each subset, creating branches and nodes until each subset is homogeneous or other stopping criteria are met.

5. Tree Pruning: Simplify the tree by removing branches that contribute little to predictive power, preventing overfitting.

6. Prediction: For new data, traverse the decision tree based on feature values to reach a classification.

2.5. Random Forest

Random Forest is an ensemble learning method that builds multiple decision trees and combines their predictions for a more accurate and stable output (Breiman, 2001).

1. Data Sampling: Create multiple subsets of the data through bootstrapping (sampling with replacement).
2. Tree Building: For each subset, construct a decision tree by selecting random features at each split, which helps reduce correlation between trees.
3. Voting: Aggregate predictions from each decision tree (for classification, use majority voting; for regression, use average values).
4. Ensemble Prediction: Use the combined predictions as the final output, which is generally more accurate than individual trees.

2.6. REP Tree (Reduced Error Pruning Tree)

REP Tree, or Reduced Error Pruning Tree, is a decision tree algorithm that prunes branches of the tree to reduce error and prevent overfitting (Frank & Witten, 1998).

1. Data Preparation: Preprocess data and handle any missing or categorical values.
2. Tree Construction: Build an initial decision tree, similar to J48, by choosing the attribute with the best split at each node.
3. Error Pruning: Evaluate each branch of the tree on a separate validation set, pruning branches that do not reduce classification error.
4. Final Tree: Retain branches that contribute positively to predictive accuracy, resulting in a simpler model that generalizes well.
5. Prediction: Classify new instances by traversing the pruned tree, using only significant branches.

2.7. Proposed Deep Learning Approaches (GAN-ES)

Deep learning methodologies primarily revolve around training complex neural networks to capture intricate patterns in data. The methodologies involve various architectures, training protocols, and optimization techniques to enhance learning efficiency, model accuracy, and generalizability. Here is an overview of deep learning methodologies and the optimization techniques commonly used to improve performance.

2.7.1 Generative Adversarial Networks (GANs):

Generative Adversarial Networks (GANs) are powerful models that consist of two neural networks, a generator and a discriminator competing against each other. This adversarial framework enables the generator to produce data that closely resembles the training data, while the discriminator learns to differentiate between real and fake data. Here is a step-by-step breakdown of the GAN algorithm (Goodfellow et al., 2014, Arjovsky et al., 2017). Here’s a concise, step-by-step algorithm for training Generative Adversarial Networks (GANs):

1. Initialize Networks: Set up the generator and discriminator networks with appropriate architectures and hyperparameters like learning rate, batch size, and latent space size.
2. Sample Training Data: Prepare batches of real data from the training set to train the discriminator.
3. Train Discriminator:

o a. Generate Fake Data: Sample noise and feed it to the generator to create fake data samples.

o b. Discriminator Prediction: Feed real and generated (fake) data to the discriminator.

o c. Calculate Loss: Compute discriminator loss based on its ability to distinguish real from fake samples.

o d. Update Discriminator: Backpropagate and update discriminator weights to improve real/fake classification.

4. Train Generator:

o a. Generate Fake Data: Use new noise samples to generate more fake data.

o b. Discriminator Prediction for Generator: Feed fake data into the discriminator but label it as real.

o c. Calculate Generator Loss: Compute loss based on the discriminator's performance at identifying fake data as real.

o d. Update Generator: Backpropagate and adjust generator weights to improve generated sample realism.

5. Repeat: Alternate between training the discriminator and generator for several epochs until the generator produces high-quality data.

6. Save Best Model: Save the generator model when it generates samples that resemble real data effectively.

Here are short definitions of GAN hyperparameters:

1. Learning Rate: Determines the pace of model updates; a balanced rate can stabilize GAN training. Learning rates of 0.0002 are commonly used for both generator and discriminator (Goodfellow et al., 2014).

2. Batch Size: Number of samples in each training iteration, affecting stability and memory usage. Batch sizes of 64 or 128 are typical, offering a balance of stability and efficiency (Radford et al., 2016).

3. Latent Space Dimensionality: The size of the random noise vector input to the generator, influencing generated data variety. A 100-dimensional noise vector is standard in image GANs for adequate diversity (Salimans et al., 2016).

4. Number of Epochs: The number of full training set passes, necessary for GANs to converge to high-quality outputs. GANs often require hundreds or thousands of epochs for fine detail in generated images (Arjovsky et al., 2017).

5. Optimizer and Parameters: The algorithm and specific settings for updating weights; common choices are Adam or RMSprop. Adam with parameters β1=0.5 and β2=0.999 supports stable GAN training (Goodfellow et al., 2014).

6. Regularization Parameters: Techniques like gradient penalties to maintain GAN stability and prevent overfitting. WGAN-GP often uses a gradient penalty of 10 for added stability (Gulrajani et al., 2017).

7. Discriminator-to-Generator Update Ratio: Ratio of discriminator to generator updates per training step, affecting balance. WGANs commonly use a 5:1 update ratio for discriminator to generator (Arjovsky et al., 2017).

8. Activation Functions: Non-linear functions in the networks; common choices include Leaky ReLU in layers and Tanh in outputs. Leaky ReLU for generator layers and Tanh for output layers in normalized images (Radford et al., 2016).

9. Loss Function Type: Guides model performance; cross-entropy or Wasserstein loss is commonly used. WGAN uses Wasserstein loss to enhance stability (Arjovsky et al., 2017).

10. Noise Distribution: The distribution type for the generator's noise input (e.g., Gaussian or uniform), impacting sample quality. Gaussian noise (mean 0, standard deviation 1) is common in GANs (Salimans et al., 2016).

2.7.2 Early Stopping (ES)

Early stopping is an optimization technique that halts training once the model's performance on a validation set ceases to improve. This approach is widely used to prevent overfitting, ensuring the model generalizes well to unseen data. GANs often suffer from instability during training, and early stopping helps prevent overfitting to seasonal or temporary patterns in weather data, which can lead to more realistic generation of synthetic weather or crop yield data. For agricultural data, which can be influenced by highly variable and often stochastic weather conditions, early stopping aids in avoiding mode collapseand promotes stability in GANs for Agriculture. When the objective is generating realistic weather patterns, crop yield, or disease prediction based on diverse environmental conditions, early stopping helps prevent the GAN model from over-optimizing on specific weather events that may not recur (Goodfellow et al., 2014, Zhang et al., 2019).

Steps for Early Stopping

1. Split the Data: Partition the training data into a training set and a validation set. Allocate 80% of the data for training and 20% for validation (Prechelt 1998).

2. Monitor Validation Performance: Train the model and assess its performance on the validation set at each epoch. Track metrics such as validation loss or accuracy (Goodfellow et al. 2016).

3. Set Patience Parameter: Establish a patience value, which determines the number of epochs to wait without improvement before stopping. With a patience value of 10, training halts if there is no improvement in validation metrics for 10 epochs (Yao et al. 2007).

4. Check for Improvement: At each epoch, compare the current validation metric with the best value recorded. If an improvement is found, reset the patience counter. Update the best score whenever a new lowest validation loss is recorded (Prechelt 1998).

5. Stop Training: Cease training when the validation metric has not improved for the specified number of epochs (patience) to avoid overfitting. If no improvement is observed after 10 continuous epochs, stop training and revert to the model with the best recorded performance (Goodfellow et al. 2016).

6. Save the Best Model: Preserve the model with the highest validation performance for subsequent tasks. Save the model parameters at the best-performing epoch (Hinton et al. 2012).

2.8 Generative Adversarial Networks (GANs):

Here are simple definitions and formulas for the Kappa statistic, and error rates (MAE, RMSE, RAE, RRSE):

2.8.1 Kappa Statistic (κ)

The Kappa statistic measures the agreement between a model's classifications and the true classifications, adjusting for agreement that could happen by chance. It provides a value between -1 and 1, where 1 represents perfect agreement, 0 means chance-level agreement, and negative values indicate disagreement.

$$k = \frac{p_0 - p_e}{1 - p_e}$$

Where p_o: the observed agreement (proportion of instances the model classified correctly) and p_e: the expected agreement by chance.

2.8.2 Mean Absolute Error (MAE)

MAE represents the average of absolute differences between the model's predictions and actual values. Lower values indicate better model performance.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where y_i: actual value, $\hat{y}_i$: predicted value, and n: total number of observations.

2.8.3 Root Mean Square Error (RMSE)

RMSE is the square root of the average squared differences between the predicted and actual values. RMSE penalizes large errors more than MAE, making it sensitive to outliers.

$$\text{RMSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

2.8.4 Relative Absolute Error (RAE)

RAE is the total absolute error of the model's predictions relative to the total absolute error of a simple mean-based prediction. Lower RAE indicates a more accurate model.

$$\text{RAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\sum_{i=1}^{n}|y_i - \bar{y}|}$$

Where $\overline{y}$: Mean of the actual values.

2.8.5 Root Relative Squared Error (RRSE)

RRSE compares the RMSE of the model to the RMSE of a baseline model (mean prediction). It provides a relative measure, with lower RRSE indicating better performance.

$$\mathrm{RRSE} = \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\overline{y})^2}$$

These metrics help evaluate a model's accuracy, robustness, and overall performance, offering insight into how well the model's predictions match actual values.

3. Experimental Results

Table 1: Crop Recommendation System using Nutrients and Weather

Nitrogen (N)	Phosphorous (P)	Potassium (K)	temperature	Humidity	ph	rainfall	label
90	42	43	20.8797	82.0027	6.5030	202.9355	rice
85	58	41	21.7705	80.3196	7.0381	226.6555	rice
60	55	44	23.0045	82.3208	7.8402	263.9642	rice
83	60	36	25.5970	80.1451	6.9040	200.8349	rice
71	54	16	22.6136	63.6907	5.7499	87.7595	maize
61	44	17	26.1002	71.5748	6.9318	102.2662	maize
40	72	77	17.0250	16.9886	7.4860	88.5512	chickpea
23	72	84	19.0206	17.1316	6.9203	79.9270	chickpea
13	60	25	17.1369	20.5954	5.6860	128.2569	kidneybeans
25	70	16	19.6347	18.9071	5.7592	106.3598	kidneybeans
14	67	15	19.5638	24.6739	5.6901	139.2921	kidneybeans
7	56	18	18.3136	24.3299	5.6984	76.1415	kidneybeans
3	72	24	36.5127	57.9289	6.0316	122.6540	pigeonpeas
40	59	23	36.8916	62.7318	5.2691	163.7267	pigeonpeas
33	73	23	29.2354	59.3897	5.9858	103.3302	Pigeonpeas
27	57	24	27.3353	43.3580	6.0919	142.3304	pigeonpeas
55	67	16	34.3733	69.6937	6.5967	70.2718	blackgram
23	70	15	34.6008	63.113	7.4036	60.4179	blackgram
53	74	15	29.4346	64.9433	7.5171	72.1782	blackgram
32	76	15	28.0515	63.498	7.6041	43.358	lentil
13	61	22	19.4408	63.2777	7.7288	46.8313	lentil
38	60	20	29.8482	60.6387	7.4912	46.8045	lentil
31	25	38	24.9627	92.405	6.4974	109.4169	pomegranate
21	21	38	22.5526	89.3259	6.3277	104.8956	pomegranate
108	92	53	27.4005	82.9622	6.2768	104.9378	banana
86	76	54	29.3159	80.1159	5.9268	90.1098	banana
80	77	49	26.0543	79.3965	5.5191	113.2297	banana
21	26	27	27.0032	47.6753	5.6996	95.8512	mango
25	22	25	33.5615	45.5356	5.9774	95.7053	mango
34	15	34	27.0583	91.1051	5.6773	224.7007	coconut
14	23	25	26.1855	96.9664	5.6121	135.4186	coconut
18	19	29	27.5938	92.4852	6.2061	162.8433	coconut
135	43	16	23.4799	81.7305	6.7204	86.7629	cotton
100	46	18	24.1859	76.042	6.4317	69.0806	cotton

Table 2: Correctly and incorrectly classified instance

ML Approaches	Correctly Classified Instances (%)	Incorrectly Classifies Instances (%)
Logistic Regression	97	3
Multilayer Perceptron	98	2
SMO	96	4
REP Tree	99	1
GAN-ES	100	0

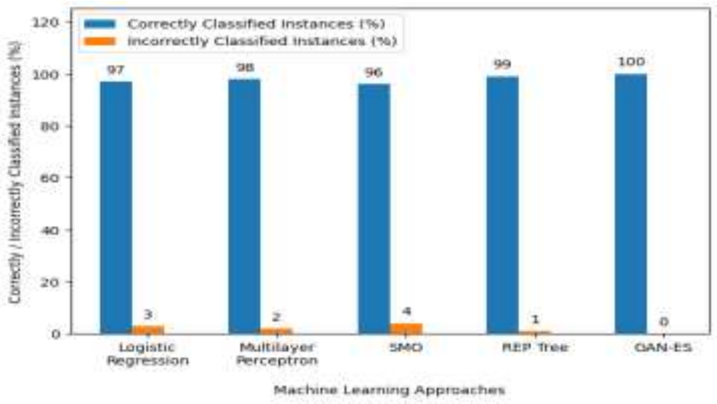

Figure 1: Accuracy of correctly and incorrectly classified instances in (%)

Table 3: Kappa statistic

ML Approaches	Kappa Statistic
Logistic Regression	0.9732
Multilayer Perceptron	0.9892
SMO	0.9276
REP Tree	0.9985
GAN-ES	1.0000

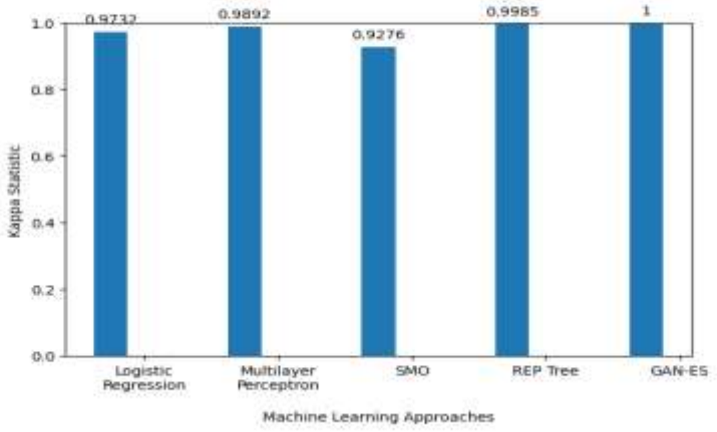

Figure 2: Performance of Kappa statistic

Table 4: Performance MAE and RMSE

ML Approaches	MAE	RMSE
Logistic Regression	0.0815	0.7216
Multilayer Perceptron	0.0084	0.0282
SMO	0.3826	0.4800
REP Tree	0.0012	0.0042
GAN-ES	0.0000	0.0000

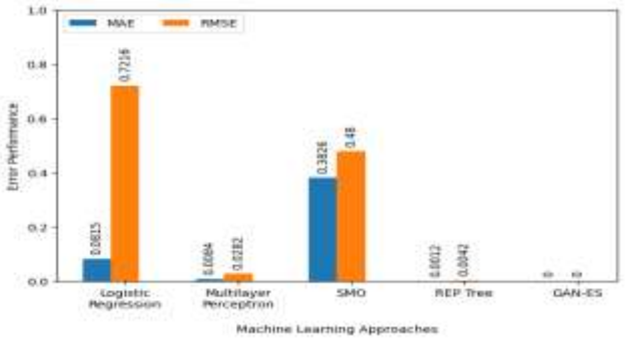

Figure 3: Performance of MAE and RMSE

Table 5: Performance of RAE and RRSE

ML Approaches	RAE	RRSE
Logistic Regression	1.0498	10.3635
Multilayer Perceptron	0.4888	9.9471
SMO	95.2452	96.0077
REP Tree	0.8457	7.8545
GAN-ES	0.0000	0.0000

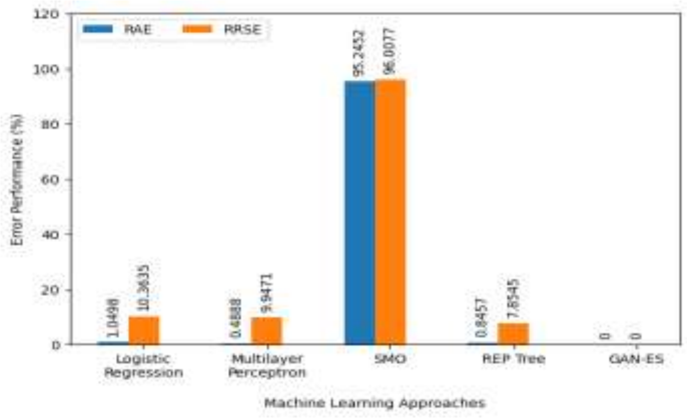

Figure 4: Performance of RAE and RRSE

Table 6: Time taken to build the model

ML Approaches	Time Taken (Seconds)
Logistic Regression	59.11
Multilayer Perceptron	160.70
SMO	1.77
REP Tree	5.53
GAN-ES	45.11

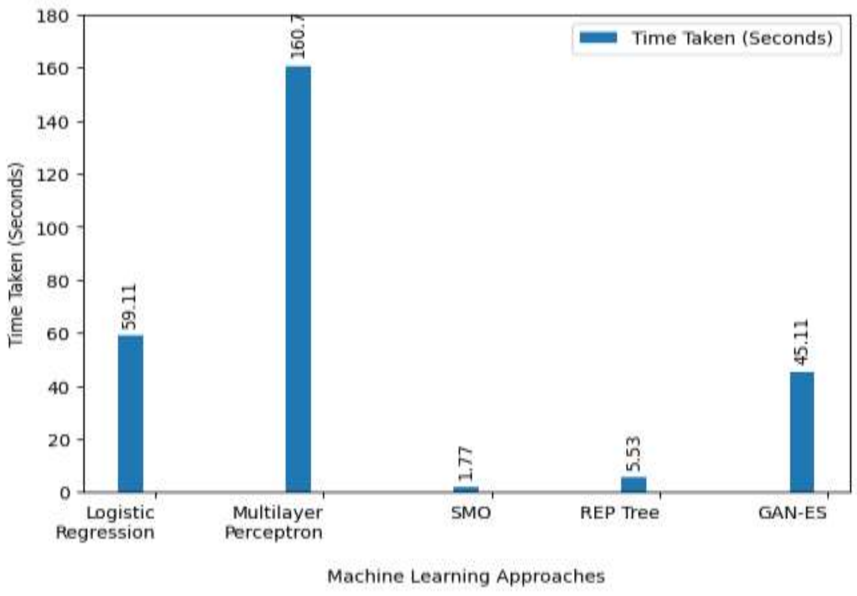

Figure 5: Time taken to build the model

4. Results and Discussions

The experimental results evaluated five machine learning and deep learning models - Logistic Regression, Multilayer Perceptron, SMO, REP Tree, and GAN-ES on a crop classification dataset using nutrient and weather data. Key performance metrics included classification accuracy, Kappa statistic, error rates (MAE, RMSE, RAE, RRSE), and model-building time. Below is a detailed presentation of these findings.

The accuracy of correctly classified and incorrectly classified instances for each model is presented in Table 2 and Figure 1. GAN-ES achieved the highest accuracy, with 100% of instances correctly classified. REP Tree also performed well, with 99% accuracy. These results indicate that GAN-ES is the most accurate model for crop classification based on the provided data.

The Kappa statistic (Table 3 and Figure 2) reflects each model's agreement with actual classifications. GAN-ES achieved a perfect Kappa statistic of 1.000, indicating complete alignment with the ground truth. REP Tree closely followed with a Kappa statistic of 0.9985. The Kappa statistics suggest that GAN-ES and REP Tree provide the most reliable classification results in the dataset.

MAE and RMSE, presented in Table 4 and Figure 3, measure the prediction errors for each model. GAN-ES demonstrated zero error values, indicating the highest level of precision. REP Tree and Multilayer Perceptron also had low error rates. The low MAE and RMSE values for GAN-ES and REP Tree highlight their robustness in accurately predicting crop classifications.

Table 5 and Figure 4 display the relative errors, with GAN-ES achieving zero for both RAE and RRSE, followed by Multilayer Perceptron and REP Tree with minimal error rates. In contrast, SMO exhibited the highest RAE and RRSE, suggesting less predictive accuracy. These relative error results further validate GAN-ES as the most effective model for classification in this study.

Table 6 and Figure 5 shows the time each model required for training. SMO completed training in the shortest time, at 1.77 seconds, while GAN-ES required 45.11 seconds, balancing high accuracy with moderate computational efficiency. Although SMO was the fastest, GAN-ES demonstrated the best overall performance across other metrics, making it suitable for scenarios requiring both accuracy and efficiency.

The experimental results confirm GAN-ES as the top-performing model with perfect classification accuracy, zero errors, and high Kappa agreement. This model's performance makes it an ideal candidate for accurate and efficient crop classification based on environmental and nutrient data. In summary, GAN-ES is identified as the optimal choice for crop classification due to its unparalleled accuracy, zero error rates, and robustness. For applications prioritizing both speed and accuracy, REP Tree is a viable alternative. These findings suggest that deep learning models, particularly GAN-ES, offer promising potential for accurate agricultural predictions and decision-making processes in precision agriculture.

5. Conclusion

The study evaluated the effectiveness of various machine learning (ML) and deep learning models for crop classification using environmental and nutrient data. Key performance metrics included classification accuracy, Kappa statistic, error rates (MAE, RMSE, RAE, RRSE), and model-building time. The conclusions derived from the analysis are Superior Performance of GAN-ES, High Reliability of REP Tree, Multilayer Perceptron as a Balanced Model, Limitations of SMO, and Trade-Offs Between Accuracy and Computational Time.

6. Further Research

The findings from this study highlight areas for future research to enhance crop classification accuracy and optimize model performance. Building on the analysis of machine learning (ML) and deep learning approaches, the recommendations can guide further research namely the Exploration of Hybrid and Ensemble Models, Incorporating More Environmental and Biophysical Factors, Evaluating Temporal and Seasonal Variability, Real-Time Data Processing and Prediction, Comparative Studies Across Diverse Crop Types, and Economic and Environmental Impact Assessment. These future research directions aim to refine classification models, adapt them to diverse agricultural contexts, and maximize their practical applications in precision agriculture. By addressing these areas, future studies could contribute to the development of reliable, adaptable, and efficient models for enhanced crop management and agricultural productivity.

References

[1] Ali, M., & Chaudhary, M. A. (2020). A Comparative Study of Machine Learning Algorithms for Crop Yield Prediction. International Journal of Agricultural Science and Technology, 12(3), 90–100.
[2] Arjovsky, M., et al. (2017). "Wasserstein GAN." Proceedings of the 34th International Conference on Machine Learning, 70, 214-223.
[3] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.
[4] Chlingaryan, A., Sukkarieh, S., & Whelan, B. (2018). Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. Computers and Electronics in Agriculture, 151, 61–69.

[5] Feng, Q., Liu, J., & Gong, J. (2019). Remote Sensing, GIS, and Machine Learning in Crop Yield Prediction. International Journal of Precision Agriculture, 10(4), 218–225.
[6] Frank, E., & Witten, I. H. (1998). Reduced-error pruning with significance tests. Machine Learning, 30(2-3), 151-166.
[7] Goodfellow, I., et al. (2014). Generative Adversarial Nets. Advances in Neural Information Processing Systems, 27, 2672–2680.
[8] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. Advances in neural information processing systems, 30.
[9] Hinton, G. E. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
[10] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression. John Wiley & Sons.
[11] Jha, S., Doshi, H., & Chawda, R. (2019). Machine Learning for Agricultural Yield Prediction: A Survey. Computational Agriculture Review, 15(5), 300–315.
[12] Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. Computers and Electronics in Agriculture, 147, 70–90.
[13] Kumar, A., Yadav, D., & Chauhan, Y. K. (2019). Machine Learning Approaches for Crop Yield Prediction: A Review. Journal of Agricultural Research, 74(6), 253–262.
[14] Lal, R., Stewart, B. A., & Hartemink, A. E. (2019). Soil health and intensive cropping. Springer Nature Agriculture Series, 6, 57–75.
[15] Patil, D. V., & Shirkhedkar, S. (2018). Crop Yield Prediction Using Data Mining Techniques. International Journal of Research in Engineering and Technology, 7(3), 45–52.
[16] Platt, J. (1998). Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Advances in Kernel Methods - Support Vector Learning.
[17] Prechelt, L. (2002). Early stopping-but when?. In Neural Networks: Tricks of the trade (pp. 55-69). Berlin, Heidelberg: Springer Berlin Heidelberg.
[18] Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann.
[19] Radford, A. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.
[20] Rajesh, P., & Karthikeyan, M. (2017). A comparative study of data mining algorithms for decision tree approaches using weka tool. Advances in Natural and Applied Sciences, 11(9), 230-243.
[21] Rajesh, P., Karthikeyan, M., & Arulpavai, R. (2019). Data mining algorithm to predict the factors for agricultural development using stochastic model. International Journal of Recent Technology and Engineering, 8(3), 2713-271.
[22] Rajesh, P., Karthikeyan, M., Santhosh Kumar, B., & Mohamed Parvees, M. Y. (2019). Comparative study of decision tree approaches in data mining using chronic disease indicators (CDI) data. Journal of Computational and Theoretical Nanoscience, 16(4), 1472-1477.
[23] Ramesh, V., & Rao, R. V. (2021). Enhancing Crop Yield Prediction Using Deep Learning Techniques. Agricultural Informatics, 18(3), 98–105.
[24] Ravi, A. M., Prasad, R., & Hegde, G. (2020). Climatic Factors Affecting Agricultural Productivity. Journal of Agricultural and Environmental Science, 18(1), 50–62.
[25] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386.
[26] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. Advances in neural information processing systems, 29.
[27] Sarker, I. H., & Arifuzzaman, M. (2020). Applications of machine learning and data mining approaches for crop yield prediction: A review. Journal of Data Science and Machine Learning Applications, 12(2), 103–112.
[28] Shinde, M. P., & Shah, A. B. (2020). Machine Learning Approaches for Enhancing Crop Yield Prediction Accuracy. Agricultural Data Science Journal, 8(2), 120–130.
[29] Yao, Yiyu, et al. "Early Stopping in Neural Network Training: Improvement Strategies." Proceedings of the International Conference on Machine Learning, 2007.
[30] Zha, X., Zhang, H., & Chen, L. (2021). The Role of Machine Learning in Agriculture: A Review of Current Status and Trends. Journal of Agricultural Informatics, 20(3), 101–115.
[31] Zhang, H., et al. (2019). The Impact of Datsa Augmentation and Early Stopping in Training Generative Adversarial Networks for Climate Analysis. IEEE Transactions on Geoscience and Remote Sensing, 57(6), 3276–3285.
[32] Zhou, M., & Zeng, Q. (2017). Performance Evaluation of Machine Learning Models for Crop Yield Prediction. International Journal of Agricultural and Biological Engineering, 9(2), 23–32.