ISSN: 2583-9292 (ONLINE)
DOI: 10.21917/ijdsml.2025.0186

AN INTELLIGENT CROP ADVISORY SYSTEM USING WO-XGBOOST CLASSIFICATION FOR SUSTAINABLE FARMING

P. Nithya[1], A.M. Kalpana[2] and P. Tharani[3]

[1,3]Department of Computer Science and Engineering, Government College of Engineering, Salem, India

[2]Department of Computer Science and Engineering, Government College of Engineering, Dharmapuri, India

Abstract

Agriculture is extremely vital to India's economy. Farmers' main concern is that they do not choose the proper crop based on factors such as soil nutrients, humidity, water level, moisture, and season. As a result, they are experiencing a major drop in productivity. Machine learning algorithms are employed in modern farming operations to select the best crops based on soil types, weather, and climatic conditions. In this study, a model is developed to forecast feasible crops based on the soil physicochemical properties, climate and crop rotation factors. An optimized XGBoost classifier model is developed using the Whale Optimization (WO) algorithm which would choose appropriate hyperparameters for the boosting tree classifier. Further, feature selection algorithm is employed which would improves the accuracy by selecting optimal feature subset by eliminating insignificant and redundant features. The crop recommendation using WO based XGBoost achieves the accuracy of 95.7 % which is significantly higher than the XGBoost and Grid Search-XGBoost.

Keywords:

Machine Learning, Classifier, Optimization, Feature Selection, Crop Recommendation

1. INTRODUCTION

Agriculture is important for global food security and economic prosperity, but the issues of sustainable crop production and effective land use remain major concerns. The physicochemical features of the soil, such as pH, nutrient availability, organic matter content, soil texture, and water retention capacity, naturally influence crop choices in a given region. These characteristics are critical in affecting crop growth, yield, and resilience [1] Traditional crop selection methods, which frequently rely on broad guidelines or empirical knowledge, may fail to account for site-specific differences, resulting in poor agricultural practices [2].

The emergence of machine learning (ML) provides a disruptive way to tackling these issues. Machine learning algorithms have showed the ability to model complicated, non-linear connections between soil qualities, climatic conditions, and crop requirements. Using massive datasets, these algorithms can find patterns and provide predictions that outperform traditional methods in terms of accuracy and efficiency [3]. ML applications in agriculture include soil

categorization, yield prediction, insect detection, and, more recently, crop suitability analysis [4].

Predictive models based on machine learning can combine a wide range of datasets, including physicochemical soil parameters, remote sensing data, and historical crop performance records, to select crops that are appropriate for local conditions. These models improve precision agriculture, reduce environmental impact, and improve decision-making for farmers and policymakers [5].

These study aims to create an ML-based framework for predicting viable crops based on soil physicochemical parameters. The study uses methods such as random forests, support vector machines, and deep learning techniques to increase crop recommendation precision and dependability. This strategy aims to bridge the gap between data-driven insights and practical agricultural applications, resulting in more efficient and sustainable farming systems.

2. LITERATURE SURVEY

A multi-objective crop model to maximize crop net return wee proposed. The CSA-PSO model combines the Crow Search Algorithm (CSA) and PSO to solve the multi-objective problem is proposed. The performance of the proposed algorithm is evaluated using the CEC 2009 benchmark function. The proposed algorithm is analyzed by considering eight crops to find the optimal cropping pattern that increases annual net yield while reducing fertilizer use[7]. A MH based decision support system to help farmers for making optimal strategic decisions in the plant planning system. Analysis and modelling of decision-making processes in crop planning are attractive for production and formalized knowledge of cultivation plans on farms. Formalizing the decision-making process is critical in developing decision-support systems that exceed the limitations of previously developed systems' prescriptive approach[8].

A hybrid model using ACO to optimize DCNN (Deep Convolution Neural Networks) and Long Short TermMemory (LSTM) networks to forecast the suitable crop. DCNNs generally achieve high accuracy but requires computation. Since the computation complexity is mainly based on the number of layers, ACO optimizes training hyperparameters to reduce complexity. An Automatic Crop Yield Prediction using the Chaotic Political Optimizer with DL (ACYP-CPODL) model. The ACYP-CPODL technique involves different processes: preprocessing, prediction and parameter optimization were projected. In addition, a hybrid CNN and LSTM technique is also designed for the prediction process. Moreover, the CPO algorithm performs the hyperparameter tuning of the CNN-LSTM approach. As a result, the proposed ACYP-CPODL technique achieved effective results with an MSE of 0.031 and an R2 score of 0.936, whereas the LSTM model achieved near-optimal results. To validate the improved predictive power of the ACYP-CPODL method, extensive simulations were performed on benchmark datasets. Comparative results highlight the improvement of the ACYP-CPODL method over current methods[9].

An Automatic Crop Yield Prediction using the Chaotic Political Optimizer with DL (ACYP-CPODL) model. The ACYP-

CPODL technique involves different processes: preprocessing, prediction and parameter optimization. In addition, a hybrid CNN and LSTM technique is also designed for the prediction process. Moreover, the CPO algorithm performs the hyperparameter tuning of the CNN-LSTM approach. As a result, the proposed ACYP-CPODL technique achieved effective results with an MSE of 0.031 and an R2 score of 0.936, whereas the LSTM model achieved near-optimal results. To validate the improved predictive power of the ACYP-CPODL method, extensive simulations were performed on benchmark datasets. Comparative results highlight the improvement of the ACYP-CPODL method over current methods[10].

A model has been presented based on various geographical and farm parameters using a softmax classifier and a nature-inspired optimization algorithm. A hybrid technique was developed using two naturally-inspired algorithms, bio-geographical optimization and Plate Tectonics Optimization. The Adam optimization algorithm was then merged with PBO to develop a hybrid PBO/Adam algorithm. This methodology uses classifiers to predict suitable crops using various parameters. The classifier weights are optimized using the hybrid algorithm[11].

3. MATERIALS AND METHODS

The materials and the methodologies used in this work are discussed in this section. The proposed system architecture is shown in Fig.1. In this research, a recommendation system is proposed to predict suitable crop using optimized XGBoost (Extreme Gradient Boosting) classifier. In order to increase the predictive power of the proposed model, feature selection algorithm is adopted to eliminate the redundant and irrelevant attributes. Proper hyper-parameter tuning is necessary for any classifier's successful application. To optimize the hyper-parameters of XGBoost, Whale optimization algorithm (WO) is used in this research work. Chaos strategy is adopted to improve the distribution quality of the initial population. The efficacy of the proposed model is evaluated on the collected data set and the result is compared with Grid Search-XGBoost and conventional XGBoost models. Four different evaluation metrics: accuracy, precision, recall, F1-score were used for performance evaluation. The experimental result shows its validity and efficiency in the prediction of suitable crop. In addition, proposed model has better performance compared to the previously suggested models. Moreover, the proposed method reaches the high prediction accuracy of 95.7%.

XGBoost algorithm is the advanced implementation of gradient boosting algorithm and has been effectively applied to some studies [6]. It is capable of handing regularization and over fitting - under fitting issues. The efficiency of a classifier derived from this model depends greatly on the number of parameters to be modified by the user; these are normally referred to as hyper-parameters and their values can significantly affect a classifier's performance. Hyper-parameter modification for a machine learning algorithm needs familiarity with the method, practice, and typically trial and error. However, this task can be presented as an optimization problem in order to obtain the best possible solution systematically and effectively, given an appropriate objective function capturing the classifier's performance in terms of hyper-parameter configurations.

Several approaches, such as Manual, Grid Search (GS) and Random Search (RS) have been effective in solving the problem of hyper-parameter tuning[7]. Manual tuning of hyper-parameters by the user is ineffective and unhopeful approach, if there are substantial number hyper-parameters values to be taken into consideration. Grid and random searches take a long time to complete because they waste time assessing unproductive parts of the search space. These methods barely rely on any information learned by the model during previous optimizations.

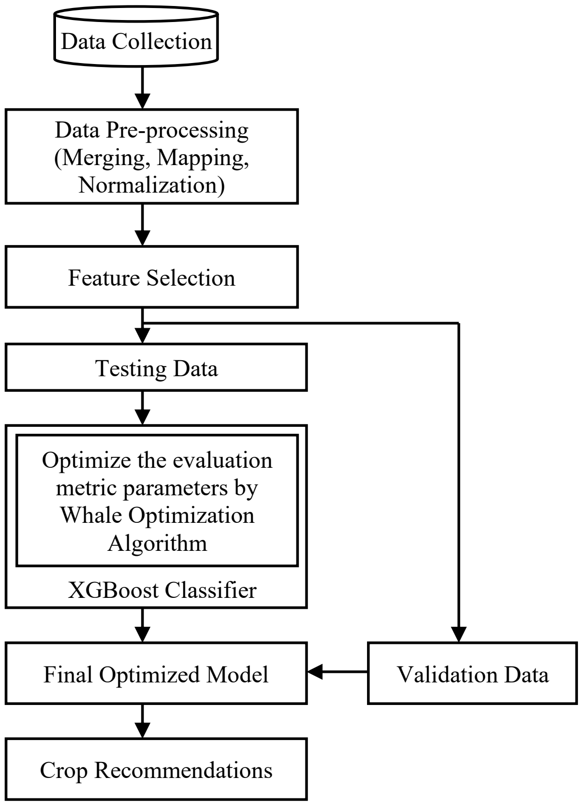
Data Collection
Data Pre-processing (Merging, Mapping, Normalization)
Feature Selection
Testing Data
Optimize the evaluation metric parameters by Whale Optimization Algorithm
XGBoost Classifier
Final Optimized Model
Validation Data
Crop Recommendations

Fig.1. Proposed crop recommendation using WO-XGBoost

3.1 DATA COLLECTION

The standard dataset with the required parameters is not readily available for this research. Hence a dataset is created from scratch, which is one of the significant contributions. Agro professionals, farmers, agricultural departments, government websites, and office records were the sources of information for data collection. The study area was Salem district in the state of Tamil Nadu, India. This district was selected because the agricultural crops like rice, ragi, green gram, black gram, red gram, sorghum, tapioca, ground nut and sesame are the major economic crops in these areas. The required data for this study were collected from various blocks in Salem district. Data related to weather, soil, water level and crop in this work are obtained from randomly selected farming sites in the study area. The data set consist of the following parameters related to climate, crop and soil.

3.1.1 Data Description

- ***Climate Parameters:*** Precision farming aims to improve the productivity by selecting appropriate crops based on the climate factors such as temperature, ambient humidity, soil moisture, solar radiance, and rainfall. Evaluation of such parameters helps the farmers to select appropriate climate adapted crops and varieties, and plan their agricultural activities.

- ***Site Specific Parameters:*** The soil site suitability evaluation helps in identifying the potential of the soils to produce different crops on a sustainable basis without degrading land. The most suitable crop selection based on site specific parameters not only improves yield but also aids in lowering the unnecessary application of fertilizers, which ultimately reduces soil quality and crop yield. Altitude, Drainage, Erosion and stoniness are the site specific parameters collected for analysis.
- ***Soil Characteristics:*** The soil and site characteristics are used as parameters for assessing the suitability of land for crop selection. The parameters related to soil characteristics are soil type, soil texture, lime status, moisture retention, bulk density and soil depth
- ***Soil Fertility:*** Proper nutrition is necessary for satisfactory crop growth and production. Each piece of land will have a unique combination of minerals, living things, and inorganic substances, which determines the type of plant which grows successfully. Cultivating the crop that best fits the soil will decrease the need for soil treatment, reducing the costs and potential environmental damages. Parameters related to soil are potential of Hydrogen (pH), Electrical Conductivity (EC), Organic carbon (OC), Nitrogen (N), Phosphorus (P), Potassium (K), Sulphur (S), Zinc (Z), Boron (B), Iron (Fe), Manganese (Mn) and Copper (Cu).

Crop Data: In terms of nutrients, light, water, temperature, and air, different plant species may have distinct requirements. The plant will not grow correctly if one of these fundamental requirements is not supplied. Selection of the right crop and variety is a very important factor to obtain maximum profit. Planting several crops in succession on the same piece of land in order to enhance soil health, maximize nutrient content, and reduce insect and weed. Crop type, Crop predecessor and season are the parameters related to crop.

The dataset was created crop-wise in consultation with agriculture experts. Data such as zone name, district name, seasons, soil physicochemical properties, soil fertility, minimum, maximum, optimal temperature, crop name and crop predecessor were collected. An appropriate values/range of values was collected from experts. Total 1250 instances with 31 attributes for 10 crops have been created. Among the data collected 70% datasets are used for training and 30% for testing respectively.

3.2 PREPROCESSING

In this research work, preprocessing includes various activities such as cleaning and integration, feature extraction, purity check, classification, transformation, and discretization. Data cleaning is the process of filling missing values, removing data noise, and fixing data inconsistencies. Data transformation is the process of normalizing data so that it can be used for ML. Raw data includes incomplete, noise and inconsistent data. Since the data is collected from different sources and format, quality issues arise when processing with the ML algorithms.

In this work, data from various sources are collected and integrated into a single dataset in excel format. The data cleansing step is performed to remove irrelevant attributes and missing data. The Table.1 shows the variable and its type used in this research

There are two types of variables in the dataset, depending on the type of value stored, such as numerical and categorical. Numeric variables represent the data in numbers, and nominal variables contain categorical data that correspond to label values instead of numbers. The Table.1 shows variable description and unit measures.

Table.1. Variable description and unit measure

Attributes	Type	Units
Average Rainfall	Numeric	mm
Maximum Temperature	Numeric	°C
Minimum Temperature	Numeric	°C
Optimum temperature	Numeric	°C
Altitude	Numeric	
Drainage	Nominal	-
Stoniness	Nominal	-
Soil Colour	Nominal	-
Soil Texture	Nominal	-
Soil Depth	Numeric	cm
CaCO3	Numeric	-
pH (potential of Hydrogen)	Numeric	-
EC (Electrical Conductivity)	Numeric	dS/M
OC (Organic Carbon)	Numeric	%
N (Nitrogen)	Numeric	Kg/Acer
P (Phosphorus)	Numeric	ppm
K(Potassium)	Numeric	Meq/100g
S (Sulphur)	Numeric	ppm
Z (Zinc)	Numeric	ppm
B (Boron)	Numeric	ppm
Fe (Iron)	Numeric	ppm
Mn (Manganese)	Numeric	ppm
Cu (Copper)	Numeric	ppm
Na(Sodium)	Numeric	ppm
Ca(calcium)	Numeric	ppm
Crop type	Nominal	-
Irrigation Type	Nominal	-
Base Saturation	Numeric	%
CEC	Numeric	cmol/kg
Crop type	Nominal	-
Crop predecessor	Nominal	-

After the data was collected, it was then entered into a Microsoft Excel spreadsheet for further processing. For the simplicity of classification, the target classes (output variable) were determined using four labels: HR, ModR, MR and NR. Finally, the target classes were assigned to each instance to quantify the degree of suitability. The real-world dataset contains missing data, inconsistent results, and noise. ML algorithms would not produce high-quality results when applied to noisy data because they cannot find the patterns successfully. Therefore, data processing is crucial to improve the overall level of data quality. In the real world, dataset inputs are generally on widely different scales.

To build the ANN forecast models which are insensitive to units, all input variables and targets are normalized. In the experimental dataset, all the categorical parameters considered for analysis are converted into numerical values using One-hot encoding and Label encoding methods. In this work, Min-Max normalization is used to rescale the inputs from one range of values to a new field. Generally, the variables are rescaled to be within a range of 0 to 1 or from -1 to 1. The rescaling is done by using the eq.(1),

$$x_{\text{scaled}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{1}$$

where, $x_{\min}$, $x_{\max}$ are the minimum and maximum absolute value of x and y respectively. x_{scaled} is the new value of each entry in data. x is the old value of each entry in data.

Many ML algorithms cannot handle categorical data directly. Therefore, data encoding is performed to convert categorical data to numeric values. There are many ways to encode data, such as label encoding, one-hot, leave-one-out, probability ratio, encoding, ordinal and M-estimator, etc. Among all these techniques, one-hot encoding and label encoding are used in this proposed research work. The categorical data is mapped to a vector containing 1 representing presence and 0 as the absence of the feature using one-hot encoding method. The number of vectors is mainly based on the number of categories of the feature, for example: the parameter "season" is categorized as kharif, rabi, and summer.

Table.2. One- hot encoding representation of season attribute

Season		
Kharif	**Summer**	**Rabi**
1	0	0
0	1	0
0	0	1

The Table.2 shows the outcome of applying one hot encoding and Table.3 shows the outcome of label encoding methods.

Table.3. Label encoding representation of class label attributes

Label	Value
HR	0
MR	1
ModR	2
NR	3

Data discretization refers to converting continuous values into a range of attribute intervals. The Table. 4 shows mapped values for variable Rainfall.

Table.4. Mapping of variable rainfall

	Values Range from	Values range to	Values mapped to
Rainfall (mm)	200	485	1
Rainfall (mm)	485	610	2
Rainfall (mm)	610	745	3
Rainfall (mm)	745	900	4
Rainfall (mm)	900	1200	5

3.3 FEATURE SELECTION

Feature selection technique is used in this work to reduce the number of input variables by eliminating irrelevant and redundant features. Adding redundant features reduces the generalization ability of the model and also reduces the overall accuracy of a classifier [8]. Furthermore, a model's total complexity rises as more variables are added to it. In this work, a feature selection method is proposed by combining feature clustering and filter approaches. The clustering method discovers the nature of features and eliminates irrelevant features. The filter method selects only the relevant and non redundant features. *K*-means clustering method is used to understand the structure of data and form the cluster. Initially '*K* 'features were randomly selected from original dataset *D*, as initial cluster centers. Based on the distance between the features and cluster mean, the most similar feature is assigned to the cluster. New mean value is computed for each cluster. The process is repeated until there is no redistribution of features in any cluster [9]. In this work, *K* value is chosen as 2. Hence two clusters where formed for the given data set. The similarity between two features is calculated using Euclidean distance function as follows

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \tag{2}$$

where *d* denotes the distance between the vector *x* and *y*.

Feature selection Algorithm

```
Input: $D=(f_1,f_2,f_3,\ldots f_n,)$ // $f_c$: training data set, N: number of cluster
Output: Obest //Optimal feature subset
Procedure:
Step 1: Start
Step 2: Choose arbitrarily initial cluster center k
Step 3: Compute distance from each feature to each cluster k, using squared Euclidean distance function by Eq.(1) and Eq.(2)
Step 4: Assign each feature to the closest cluster based on cluster mean and similarity
Step 5: Calculate the new mean for each cluster
Step 6: Repeat step 2 to 4 until there is no reassignments in the cluster center
Step 7: Remove the irrelevant feature which does not fit to any of the cluster.
Step 8: Compute correlation between features
 for (i=1;i<n;i++)
```

```
{
    $C = \rho(X,C) = \frac{E(XC) - E(X)E(C)}{\sqrt{\sigma^2(X)}\sqrt{\sigma^2(C)}}$
If($C$>0.5) remove one of the feature;
Obest=$f_i$ //add the feature to the optimum feature subset
 }
Step 9: End
```

Once the clusters are formed, the feature, which does not belong to any clusters are eliminated. The next task is to remove redundant features from each cluster. This can be done by computing correlation between features using Eq.(3).For feature X with values x and classes C with values c, the Pearson's linear correlation coefficient is defined as:

$$C = \rho(X,C) = \frac{E(XC) - E(X)E(C)}{\sqrt{\sigma^2(X)}\sqrt{\sigma^2(C)}} \tag{3}$$

where *X, C* are considered as random variables. If $\rho(X,C)$ value is 0 then features are uncorrelated and if it is greater than 0.5 then the features are highly correlated. The value is less than - 0.5 if both the features are negatively correlated (Schober et al. 2018)

3.4 XGBOOST CLASSIFIER

XGBoost is an ensemble tree method which uses the gradient descent architecture to boost weak learners. Boosting is a strategy that promotes the fusion of strong and base classifiers. First, the base classifier is trained with the initial training set and then the weight of the training samples in the subsequent base classifier is adjusted based on performance so that the samples with classification errors obtain more attention. This process is repeated until the condition has been met. The trained multiple classifiers are weighted and combined finally.

Gradient boosting is a stage-wise additive modeling method in which a weak classifier is fit to the dataset and then fits another weak classifier to enhance the performance of current model, without making modification in the previous classifier, and this process continues. Each new classifier has to judge where the previous classifiers were not performing well.

General Boosting algorithm flow is shown in Fig.2. First, estimate y1 by fitting the data to a decision tree, and the second tree is fitted based on the residual from the earlier step which is y-y1 and this process continues. The algorithm error can be minimized efficiently by analogy.

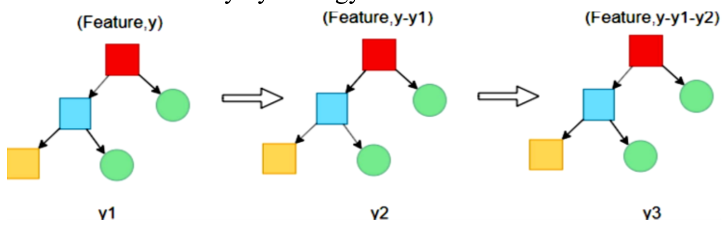

Fig.2. Boosting Classifier

With the use of gradient boosting, XGBoost approach speeds up and improves the computation process for the objective function. Parallel computing is automatically achieved during the training phase to quickly and accurately address data science challenges. XGBoost's fundamental assumption is to learn new features by including a tree structure, fitting the residuals of the final prediction, and calculating the sample score. The sample's final prediction score can be calculated by aggregating the scores of each tree. Let D be the data set and it is given by,

$$D = \{x, y\}, \quad |D| = n, \quad x \in \mathrm{R}^m, y \in \mathrm{R} \tag{4}$$

where *n* represents the number of samples, *m* be the number of features, *x* denote the features and *y* denote the target variable of the dataset. The dataset contains *n*=965 observations and *m*=32 features. In Gradient Boosting (GB) for dataset *D*, *k*-trees computed scores is the final prediction result which is calculated by a function called k additive function, as shown below

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in F \tag{5}$$

where, $\hat{y}_i$ denotes the prediction of the *i*th instance at the *k*th boost and x_i represent the *i*th instance sample of the training dataset. The value of *k*th tree is $f_k(x_i)$ and all the values of the decision trees are represented by the function *f*. GB minimizes the loss function which is defined as

$$L_k = \sum_{i=1}^{n} L(y_i, \hat{y}_i) \tag{6}$$

Since GB and XGBoost are decision tree-based algorithms, multiple tree-related hyper-parameters, such as subsample and max depth, are used to minimize the overfitting problem and enhance the model performance [10]. In addition, the learning_rate determines the trees weighting which are added to the model and it is also used to reduce the model's rate of adaptation to the training data.

XGBoost objective function has a regularization idea that helps to decide predictive functions and control the model's complexity. The objective function of the XGBoost is obtained by combining the loss function and regularization term together. Loss function regulates the model's ability to forecast, while regularization term regulates the model's simplicity. The objective function of the XGBoost as shown in Eq.(7) as follows,

$$J = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{i=1}^{k} R(f_i) \tag{7}$$

where y_i is the *i*th target's actual value; $\hat{y}_i$ is the *i*th target's predicted value; $L(y_i, \hat{y}_i)$ is the difference between y_i and $\hat{y}_i$; *n* is the sample size; *R*(*f*) is accountable for penalizing the complexity of the functions of the training tree. It also manages the overfitting problem. The function of the tree *f*(*x*) is defined as,

$$F = \{f(x) = w_{q(x)} \mid q : \mathrm{R}^m \to T, w \in \mathrm{R}^T\} \tag{8}$$

where, *w* represents the leaves scores vector, *q* denotes a mapping function which maps data instances to the corresponding leaf and *T* denotes the number of leaves. *R*(*f*) is accountable for penalizing the complexity of the functions of the training tree. It also manages the overfitting problem. It is defined as,

$$R(f) = \gamma T + \alpha \|w\| + \frac{1}{2}\lambda \|w\|^2 \tag{9}$$

where *γ* and *λ* are the hyper parameters .Each leaf value is denoted by *γ* and the total number of leaves in the tree is represented by *T*. $\|w\|^2$ represents the L2-norm of the weight of the leaf controls by *λ* term and $\|w\|$ denotes the L1 norm of the weight of the leaf controls by α term. L2 regularization (controlled by the reg_lambda term) encourages the weights to be small, whereas L1 regularization (controlled by the reg_alpha term) encourages scarcity. The hyper-parameter γ (gamma) for further partition computes the minimum loss reduction. The hyper parameter min_child_weight() controls the depth of the tree.

In a supervised learning model, an objective function is utilized to optimize the model. XGBoost employs gradient descent to optimize the objective function. The model developed is an additive model in which it includes a tree in the model every time the result of the prediction is equal to the combined outcome of the previous tree and the new tree. So, at the tth step, among these equations, the objective at every step is calculated by Eq.(9) and a f_t is selected to minimize the objective function, which is to reduce the error between the actual outcome and predicted outcome after adding f_t.

The objective function's iterative result in time is as follows:

$$J^{-(t)} \approx \sum_{i=1}^{n}\left[L\left(y_i,\hat{y}_i^{(t-1)}\right)+f_t(x_i)\right]+R(f_i)+C \tag{10}$$

where $f_t(x_i)$ is the decision tree's complexity in the tth iteration when the variable x_i is calculated; and c is a constant.

If the loss function is expanded to a second order Taylor expansion and the loss function is set to the m error, the objective function is

$$J^{(t)} \approx \sum_{i=1}^{n}\left[L\left(y_i,\hat{y}_i^{(t-1)}\right)+g_i f_t(x_i)+\frac{1}{2}h_i f_t^2(x_i)\right]+\Omega(f_t) \tag{11}$$

where g_i and h_i are the m error loss function's first and second derivatives respectively.

The tree structure is recognized by computing the leaf scores, regularization, and objective function at every level since it is not possible to compute all combinations of trees at the same time. This tree structure will be reused in subsequent iterations, which will considerably reduce the computational complexity. Furthermore, the gain of each feature is computed in the node splitting procedure. It finds the best splitting spot recursively until it reaches to the maximum depth. Then, it removes the nodes in a bottom-up order, which results in a loss. In this way the XGBoost goes deep into trees and classify the data. XGBoost has many hyper-parameters which can be used to perform some tasks as desired by the model. In this work, the XGBoost classifier output is obtained by setting the parameter values as shown in Table.5.

Table.5. Parameters values used in the XGBoost Classifier

Parameter	**Value**	**Description**
learning_rate (eta)	0.3	Shrink the weights on each step
n_estimators	100	Number of trees to fit
objective	Multi:softprob	Multiclass classification
Num_classes	4	Number of output classes
Evaluation metric	merror	Evaluation metric used
booster	gbtree	Select the model for each iteration
nthread	max	Input the system core number
min_child_weight	1	Minimum sum of weights
max_depth	0	Maximum depth of a tree
gamma	0	The minimum loss reduction needed for splitting
subsample	1	Control the sample's proportion
colsample_bytree	1	Column's fraction of random samples
reg_lambda	1	L2 regularization term on weights
reg_alpha	0	L1 regularization terms on weights

XGBoost will have less accuracy in the first iteration because the model will be primitive [11]. However, as the number of iterations increase, the model will use the Gradient Descent technique to optimize the loss function. This approach is repeated until the model reaches a point where it can no longer be optimized. As a result, when number of iterations increase, the model's accuracy improves [12].

XGBoost classifier Algorithm

Step 1: Construct a single leaf tree.
Step 2: For the first tree, calculate the average of target variable as prediction and calculate the residuals using the desired loss function. For succeeding trees the residuals come from forecast made by previous tree.
Step 3: Compute the similarity score using the following formula:
$\text{Similarity score} = \frac{\text{Gradient}^2}{\text{Hessian}+\lambda}$ (12)
Where, Hessian is equal to number of residuals; Gradient2 is the squared sum of residuals and λ is a regularization hyper parameter.
Step 4: With similarity score select the appropriate node. Higher the similarity score more the homogeneity.
Step 5: By using the similarity score, compute the information gain. Information gain provides the difference between previous similarity and new similarity and it is given by,
Information gain=left similarity+ right similarity –similarity for the root (13)
Step 6: Construct the tree of desired length using the above process. Pruning and regularization would be performed through the regularization hyper parameter.
Step 7: Predict the residual values using the constructed Decision Tree.
Step 8: The new set of residuals is calculated using the following formula:
new residuals= old residuals +$\rho\sum$predicted residuals (14)
where ρ is the learning rate.
Step 9: Go back to step 1 and repeat the process for all the trees.

In this work, XGBoost classifier is developed with 10 fold cross validation technique. Fig.3 demonstrates the tenfold cross-validation procedure used to evaluate the model. The appropriate model is built using the training dataset after the hyperparameter set values have been added to the XGBoost model. By randomly partitioning the training dataset into ten different subsets, a ten-fold cross-validation strategy was used in this study to increase the training accuracy. The developed XGBoost model was subsequently trained and evaluated ten times, choosing nine subsets for training and one for evaluation each time. Finally, the averages of ten evaluation ratings (E) were obtained.

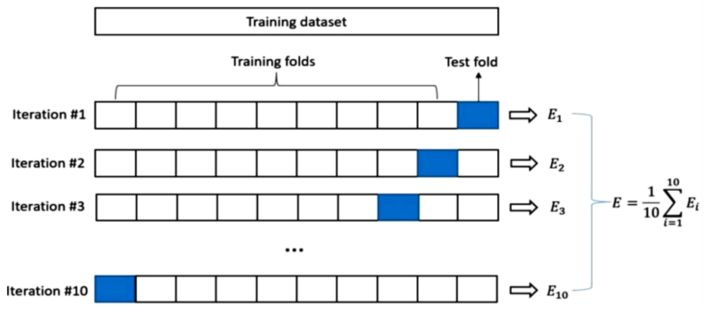

Fig.3.Tenfold cross-validation procedure

Following model construction, the developed XGBoost model's accuracy was evaluated using merror evaluation metric as follows

$$\text{merror} = \frac{\#(\text{wrong cases})}{\#(\text{all cases})} \tag{15}$$

In machine learning, the objective of hyper-parameter optimization is to identify the optimum hyper-parameters for a given machine learning algorithm that produce the best results when tested on a validation set. The following equation represents hyper-parameter optimization:

$$x^* = \arg\min_{x \in X} f(x) \tag{16}$$

where $f(x)$ denotes an objective score to minimize which is evaluated in the validation set. x^* represents a set of hyper parameters that yields the lowest score value and x can take any value in the domain x. Determining the model hyper-parameters that give the greatest score on the validation set metric.

The problem with hyper-parameter optimization is that it is tremendously costly to estimate the objective function to find the score. A model has to be trained on the training data, make predictions on the validation data, and then compute the validation metric each time with different hyper-parameters. This method cannot be carried out manually because to the high number of hyper-parameters involved and models like ensembles or deep neural networks, which can take more time to train.

3.5 HYPERPARAMETER OPTIMIZATION USING GRID SEARCH

Grid search is a comprehensive search based on subsets, and its hyperparameters are determined by utilising a lower limit, an upper limit, and the number of steps. Grid search works by preparing a grid which contain all possibilities and then be evaluated to obtain the best value among all the grids. The working mechanisms of grid search are as follows:

1. Initialize value for all the parameters.
2. Iterate through the combination of all parameter values
3. Carry out the training using XGBoost classifier on training data
4. Evaluate the resultant classifications with test data
5. Storing the finest value from the classification result and the optimum parameter value combination.

In this work, using the GridsearchCV method, the initial XGBoost model was created with 10-fold cross validation procedure and the hyperparameters were adjusted using gridsearch. The technique approves a search in a recognized range of hyperparameters and defines the preferred results leading to the best outcomes of the evaluation criteria. This can be performed using GridSearchCV( ) method in scikit-learn library. This method simply computes the score of cross validation for all hyperparameters integrated with a particular search. The Table.6 shows the upper bound and lower bound of each hyperparameters for model fine tuning.

Table.6. Hyperparameter ranges for model improvement

Parameter	Range of values
learning_rate (eta)	0.01-0.3
max_depth	1-10
n_estimators	100-600
reg_gamma	0-0.05
Subsample	0-1

The hyperparameters such as max_ depth, learning rate, n_estimators, gamma and subsample are selected for tuning. Further, the test dataset has been applied to the optimized model to assess its prediction accuracy and the final best estimator and its hyperparameter were obtained.

3.6 WHALE OPTIMIZATION ALGORITHM

The Whale Optimization Algorithm (WO) is a meta heuristic optimization algorithm that simulates humpback whale hunting mechanism. Roundup, net bubble attack, and prey search are the three key components of the WO algorithm. The whale hunts in a single or multidimensional space by changing its position vector in this algorithm. The whale represents the candidate solution, and the location or value of the potential answer indicates the problem's parameters.

In the WO method, each individual whale starts at a random place and subsequently changes its position in line with the appropriate individual whale position determined after each iteration or an individual whale that was randomly chosen. The procedure is as follows,

Encircling prey:

$$\vec{D} = |\vec{C} \cdot \vec{X}^*(t) - \vec{X}(t)| \tag{17}$$

$$\vec{X}(t+1) = \vec{X}_{\text{rand}} - \vec{A} \cdot \vec{D} \tag{18}$$

In the above formula, t denotes the current iteration number, $\vec{X}^*$ denotes the best individual position of the whale currently acquired, and $\vec{A}$ and $\vec{C}$ denotes the coefficient vectors, which can be represented as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{19}$$

$$\vec{C} = 2\vec{r} \tag{20}$$

In the range of [0, 2], the value of $\vec{a}$ declines linearly as the number of iterations increases, and the value of $\vec{r}$ is randomly produced number in the range of [0, 1].

3.6.1 Spiral Updating Position:

Using the method given in the following formula, a new position that is appropriate with the spiral motion can be determined between the whale's initial position and the prey's present position:

$$\vec{X}^{(t+1)} = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^{*}(t) \tag{21}$$

where b is an internal parameter, $\vec{D}'$ denotes the distance from the ith whale to the intended prey, and b is a random value generated between [0, 1]. The whale is randomly chosen to move in a circle or in a spiral path with a probability of 50% in order to imitate simultaneous activities. The procedure for creating a new position for the whale can be expressed using the following formula:

$$\vec{X}(t+1) = \begin{cases} \vec{X}^{*}(t) - \vec{A} \cdot \vec{D}, & p < 0.5 \\ \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}(t), & p \geq 0.5 \end{cases} \tag{22}$$

where, p is randomly generated number.

3.6.2 Prey Searching:

In order to maintain balance, a whale from the present population will be chosen at random as the current best option for the population as a whole, and other whales will migrate about to fill the void left by that whale. The following is the mathematical expression:

$$\vec{D} = | \vec{C} \cdot \vec{X}_{\text{rand}} - \vec{X} | \tag{23}$$

$$\vec{X}(t+1) = \vec{X}_{\text{rand}} - \vec{A} \cdot \vec{D} \tag{24}$$

In the WO method, each individual whale starts at a random place and subsequently changes its position in accordance with the best individual whale position discovered after each iteration or an individual whale that was randomly chosen.

WO algorithm
Input: Number of whales (**N**), Maximum number of iterations (***T*max**), Max number of episodes (**Emax**), Max number of steps **(t)**
Output : Optimized set of parameter values
Step 1: Start the search agents with a random initialization and select the first best solution.
Step 2: Update the position depending on the value of p and A in every step as follows
Step 2.1: Utilize spiral movement by updating current position by Eq.(18)
Step 2.2: Begin exploration or exploitation phase by updating the current position by Eq.(19) or (21) respectively
Step3: Verify the search space constraints and update finest solution if there is a better one found
Step 4: Finally, terminate the loop when maximum number of iteration is reached.

3.7 PROPOSED WO- XGBOOST MODEL

In this work whale optimization algorithm is proposed to select suitable hyperparameters for the XGBoost classifier because it provides good convergence speed in many optimization problems and the local optima avoidance mechanism is devised to avoid the searches from trapping into local optimal solution easily. The work flow of WO-XGBoost model is shown in Fig.4.

The global optimal solutions convergence efficiency and quality are significantly influenced by the diversity of the initial population. The randomly initialized population can't assurance the uniform distribution in the search space. By using chaos strategy method of population initialization, the distribution quality of the initial population in the search space can be improved. Once the distribution quality is improved, the convergence efficiency of the algorithm also improved. In this work, logistic chaotic mapping strategy is proposed to initialize the initial population.

$$a_{i+1} = \begin{cases} 4\mu a_i(0.5 - a_i), & 0 \leq a_i < 0.5 \\ 1 - 4\mu a_i(0.5 - a_i)(1 - a_i), & 0.5 \leq a_i \leq 1 \end{cases} \tag{25}$$

where $3.569946 \leq \mu \leq 4$, $\mu = 4$, a_0 belongs to a random number between 0 and 1, chaotic sequences are generated and then mapped to search space to create initial populations.

WO algorithm
Step 1: Preprocess the collected data.
Step 2: Initialize the WO algorithm using chaos strategy with the method's initial settings.
Step 3: Determine the top and lower bounds of the XGBoost algorithm parameters that must be optimized to generate the initial population of whales, ensuring that each whale's position is within a reasonable range.
Step 4: Calculate the fitness of each whale position based on the obtained whale population and the XGBoost model.
Step 5: Sort the collected fitness values to determine the best whale position of the current whale population, which will be saved as the current global best position.
Step 6: Using Eq.(22)–(24), update the position of the whale in each subgroup.
Step 7: Enter iterative optimization mode and execute steps 2–4 again. Stop the loop when the number of iterations hits the limit and get the best max depth, learning rate, n_ estimators, gamma and subsample parameters from the final best whale position.
Step 8: After training, add the best parameters such as max depth, learning rate, n_ estimators, gamma and subsample to the XGBoost model to build the best classification model.

The Python toolkit XGBoost is chosen to optimize five significant parameters in the XGBoost classifier: learning rate (learning_rate, ETA for short), maximum depth of the tree (max_depth), n_estimators, gamma and subsample.

Learning_rate and gamma has a strong influence in on the model to get the optimal value. The learning rate improves the model's stability and resilience, and over-fitting is managed with the parameter max depth.

Then, max_depth influence the depth of the tree and subsample handles regularization. A number of additional significant hyper-parameters are assigned with default value and Multi:softmax serves as the objective hyper-parameter.

With several hyper-parameters, manual search is infeasible. Grid search are superior to manual searches because they can automatically repeat the train-predict-evaluate cycle on a predetermined grid of model hyper-parameters. Even these approaches, though, are somewhat inefficient because they do not consider previous evaluations when choosing the subsequent hyper-parameters to evaluate. As a result, many times they spend a considerable amount of time evaluating the wrong set of hyper-parameters. From the result it is observed that the proposed Whale Optimization approach of XGBoost classifier achieves better accuracy when compared to the grid search.

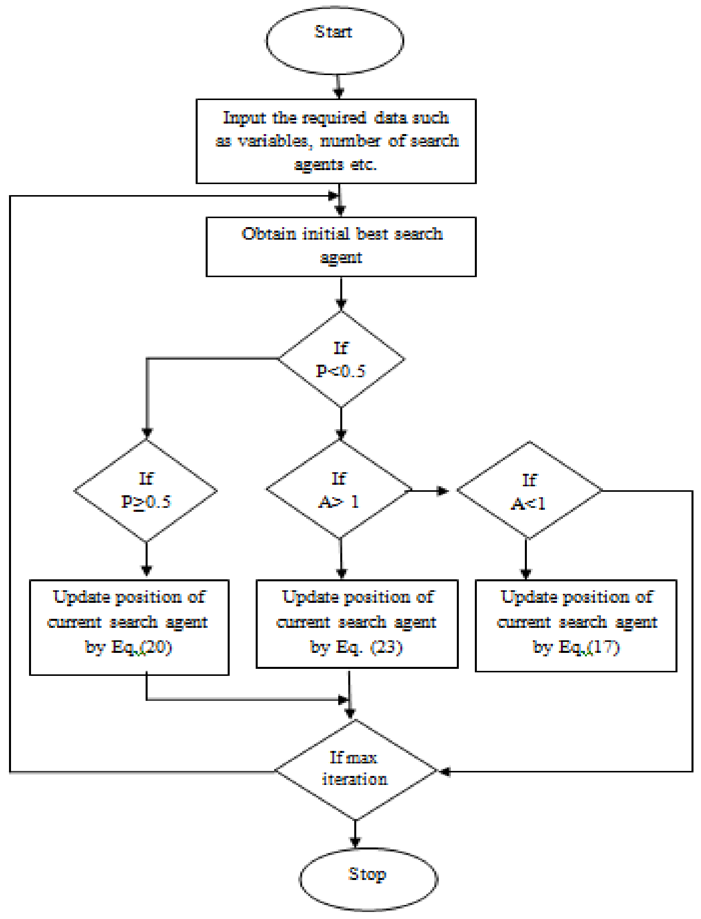
Start
Input the required data such as variables, number of search agents etc.
Obtain initial best search agent
If P<0.5
If P≥0.5
If A> 1
If A<1
Update position of current search agent by Eq.(20)
Update position of current search agent by Eq. (23)
Update position of current search agent by Eq.(17)
If max iteration
Stop

Fig.4.Flow of WO-XGboost

4. RESULT AND DISCUSSION

The experiments were performed using XGBClassifier() function from scikit-learn python package for performing the classification task. XGBoost classifier model is developed using 10-fold cross validation method. Then the model is optimized using Gridsearch optimization method and also with WO method. Based on the WO results, the best combination of hyperparameters produced was learning_rate=0.126, gamma=0.1, max_depth=5, subsample=0.8 and colsample_bytree=0.6. The Gridsearch resulted in learning_rate=0.184, gamma=0.3, maxdepth=9, subsample=0.7 and colsample_bytree=0.831.

Table.7. Comparisons of the proposed method with conventional methods in terms of various performance metrics

Classification Method	Accuracy	Precision	Recall	f1-Score
XGBoost	91.6	91.6	91.5	91.6
Grid Search-XGBoost	93.1	93.2	93.1	93.1
WO-XGBoost	95.7	95.7	95.8	95.8

The Table.7 and Fig.5 have demonstrated the comparison analysis of the proposed classification methods with the conventional method on crop recommendation in terms of accuracy, precision, recall, and F-measure.

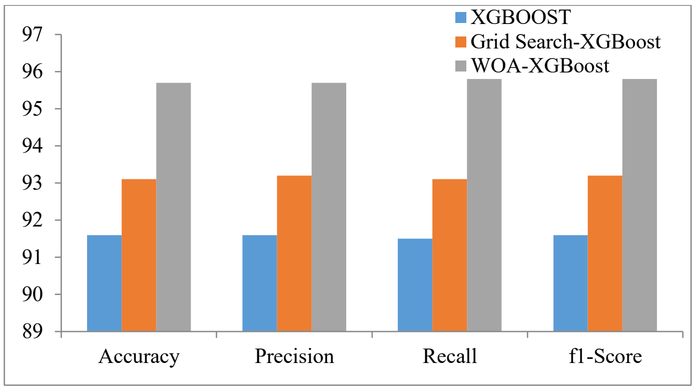

Fig.5. Comparisons of the proposed method with conventional methods in terms of various performance metrics

From Fig.5, it is observed that the proposed WO-XGBoost classifier has achieved high percentages of precision, recall, and f1-score such as 95.71%, 95.7%, and 95.73% compared to XGBoost and GridSearch-XGBoost classifiers. The GridSearch-XGBoost classifier achieves an accuracy of 93.12%, precision of 93.61%, recall of 93.36% and f1 score of 93.3%.The XGBoost classifier achieves an accuracy of 91.6%, precision of 91.7%, and recall of 91.6% and f1 score of 91.6%.

Table.8. Confusion Matrix for the proposed XGBoost Classifier

	Predicted HS	Predicted ModS	Predicted MS	Predicted NS
Actual HS	216	8	5	6
Actual ModS	6	213	5	6
Actual MS	5	8	230	7
Actual NS	8	8	9	225

Table.8 shows the confusion matrix for the XGBoost classifier. Out of 965 instances, 884 (i.e. 91.6%) were found to be correctly classified while 81 instances (i.e. 8.4%), where incorrectly classified. The model predicts 216 instances as highly suitable, 213 instances as moderately suitable, 230 instances as marginally suitable and 225 instances as not suitable out of 235, 230, 250 and 250 respectively.

Table.9. Confusion Matrix for proposed Gridsearch -XGBoost

	Predicted HS	Predicted ModS	Predicted MS	Predicted NS
Actual HS	225	6	3	1
Actual ModS	4	220	4	2
Actual MS	8	6	227	8
Actual NS	5	10	8	227

The Table.9 shows the confusion matrix for the whale optimization based XGBoost classifier. Out of 965 instances, 899 (i.e. 93.1%) were found to be correctly classified while 66 instances (i.e. 6.9 %) where incorrectly classified. The model predicts 225 instances as highly suitable, 220 instances as moderately suitable, 227 instances as marginally suitable and 227 instances as not suitable out of 235,230,250 and 250 respectively.

Table.10. Confusion Matrix for the proposed WO-XGBoost Classifier

	Predicted HS	Predicted ModS	Predicted MS	Predicted NS
Actual HS	225	6	3	1
Actual ModS	4	220	4	2
Actual MS	8	6	226	8
Actual NS	5	10	8	227

The Table.10 shows the confusion matrix for the WO-XGBoost classifier. Out of 965 instances, 924 (i.e. 95.7%) were found to be correctly classified while 41 instances (i.e. 4.3%) where incorrectly classified. The model predicts 225 instances as highly suitable, 220 instances as moderately suitable, 226 instances as marginally suitable and 227 instances as not suitable out of 235,230,250 and 250 respectively.

Table.11. Crop data set learning result

	XGBoost	GridSearch-XGBoost	WO-XGBoost
Learning Iteration	450	240	210
Error convergence	8.4	6.8	0.43
Correct classification	91.6	93.1	95.7

The Table.11 shows the crop data set learning result of the proposed method with conventional methods in terms of error convergence.

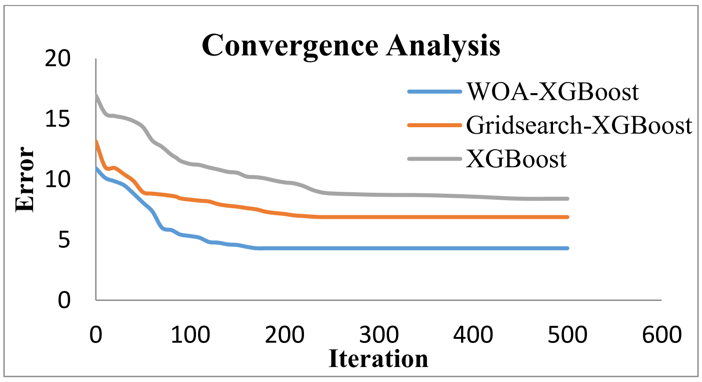

Fig.6. Convergence rates of crop data set

The Fig.6 shows that the WO-XGBoost manages to converge using minimum error at iteration 210, while Gridsearch-XGBoost at 240 and XGBoost at 450 iterations respectively.

5. CONCLUSION

In this work, WO algorithm is proposed to choose appropriate hyperparameters for the boosting tree classifier. WO exhibits brilliant convergence speed and the local optima avoidance mechanism is designed to prevent searches from easily trapping into suboptimal solutions. The experimental result shows that WO not only behaves much more stable but also outperforms Gridsearch-XGboost and XGBoost model in terms of accuracy and convergence rate. The crop recommendation using XGBoost classifier achieves an accuracy of 91.6% and Grid Search-XGBoost achieves an accuracy of 93%.The WO based XGBoost achieves the accuracy of 95.7 % which is significantly high when compared to other methods. The WO-XGBoost manages to converge using minimum error at iteration 210, while Gridsearch-XGBoost at 240 and XGBoost at 450 iterations respectively.

REFERENCES

[1] P. Singh, B. Jagyasi, N. Rai and S. Gharge, "Decision Tree based Mobile Crowd Sourcing for Agriculture Advisory System", *Proceedings of IEEE International Conference on India*, pp. 1-6, 2014.

[2] N.C. Brady and R.R. Weil, "*The Nature and Properties of Soils*", Pearson Education, 2016.

[3] R. Singh, K.K. Tiwari and A.K. Singh, "Soil Health and Crop Productivity: A Comprehensive Review", *Agricultural Reviews*, Vol. 41, No. 3, pp. 190-197, 2020.

[4] Y. Zhang, X. Zhao and Q. Liu, "Machine Learning in Agriculture: A Review of Applications in Crop Management", *Computers and Electronics in Agriculture*, Vol. 162, pp. 104-118, 2019.

[5] X. E. Pantazi, D. Moshou and A.A. Tamouridou, "Automated Leaf Disease Detection in Different Crop Species through Image Features Analysis and Support Vector Machines", *Computers and Electronics in Agriculture*, Vol. 113, pp. 251-260, 2016.

[6] S. Jain, R. Dharavath and D. Bhattacharya, "A Multi-Objective Algorithm for Crop Pattern Optimization in Agriculture", *Applied Soft Computing*, Vol. 112, pp. 107772-10787, 2021.

[7] O. Adekanmbi and P. Green, "A Meta-Heuristics based Decision Support System for Optimal Crop Planning", *Mediterranean Journal of Social Sciences*, Vol. 5, No. 20, pp. 359-368, 2014.

[8] K. Mythili and R. Rangaraj, "Crop Recommendation for Better Crop Yield for Precision Agriculture using Ant Colony Optimization with Deep Learning Method", *Annals of the Romanian Society for Cell Biology*, Vol. 25, No. 4, pp. 4783-4794, 2021.

[9] G. Sunitha, M. N. Pushpalatha, A. Parkavi, P. Boyapati and R. Walia, "Modeling of Chaotic Political Optimizer for Crop Yield Prediction", *Intelligent Automation and Soft Computing*, Vol. 34, No. 1, pp. 423-437, 2022.

[10] L. Goel, A. Jindal and S. Mathur, "Design and Implementation of a Crop Recommendation System using Nature-Inspired Intelligence for Rajasthan, India", *Proceedings of IEEE International Conference on Deep Learning for Sustainable Agriculture*, pp. 1-5, 2022.

[11] Kamilaris and F.X. Prenafeta Boldu, "Deep Learning in Agriculture: A Survey", *Computers and Electronics in Agriculture*, Vol. 147, pp. 70-90, 2018.

[12] L. Xia, G. Liu and Z. Xiao, "A Novel Approach to Crop Yield Prediction based on Machine Learning and Soil Parameters", *Agricultural Systems*, Vol. 157, pp. 59-73, 2017.

[13] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization", *Journal of Machine Learning Research*, Vol. 13, No. 2, pp. 281-305, 2012.

[14] Y. Pristyanto, H. Murase and T. Suharyanto, "Development of a Machine Learning-based Crop Selection System using Soil and Climate Data", *Agricultural Informatics*, Vol. 10, No. 1, pp. 15-25, 2019.

[15] K.P. Sinaga, M.S. Yang and M. Jameel, "Unsupervised Machine Learning for Crop Classification: A Review", *Pattern Recognition Letters*, Vol. 128, pp. 59-65, 2020.

[16] T. Santhanam, S. Sundararajan and S. Palaniappan, "Machine Learning Techniques for Crop Yield Prediction: A Review", *International Journal of Advanced Research in Computer Science*, Vol. 8, No. 9, pp. 156-162, 2017.

[17] W. Chen, X. Zhang and S. Li, "A Machine Learning Approach for Crop Yield Prediction using Environmental and Soil Data", *Computers and Electronics in Agriculture*, Vol. 121, pp. 15-22, 2016.

[18] J. Yu, H. Liu, W. Zhang and X. Li, "Predicting Crop Yields using Machine Learning Models: A Comprehensive Review", *Agricultural Systems*, Vol. 178, pp. 102746-102759, 2020.

[19] S. Rajeswari and K. Suthendran, "C5.0: Advanced Decision Tree (ADT) Classification Model for Agricultural Data Analysis on Cloud", *Computers and Electronics in Agriculture*, Vol. 156, pp. 530-539, 2019.

[20] S. Pande and B. Jagyasi, "Late Blight Forecast using Mobile Phone Based Agro-Advisory System", *Proceedings of IEEE International Conference on Computing, Communication and Automation*, pp. 1-5, 2015.

[21] A.K. Mishra, S. Kumar and P. Singh, "Mobile Crowdsourcing in Crop Production for Farmers in Rural Areas of India", *Proceedings of International Conference on Computing and Sustainable Societies*, pp. 1-5, 2020.

[22] Kamilaris and F.X. Prenafeta Boldu, "Deep Learning in Agriculture: A Survey", *Computers and Electronics in Agriculture*, Vol. 147, pp. 70-90, 2018.

[23] X.E. Pantazi, D. Moshou and A.A. Tamouridou, "Automated Leaf Disease Detection in Different Crop Species through Image Features Analysis and Support Vector Machines", *Computers and Electronics in Agriculture*, Vol. 113, pp. 251-260, 2015.

[24] A. Kamilaris, M. Tomasevic and F.X. Prenafeta Boldu, "Agri-Food Data Visualization and Analytics with the Use of Internet of Things (IoT) and Machine Learning (ML) Technologies", *Sustainable Computing: Informatics and Systems*, Vol. 20, pp. 100374-100395, 2018.

[25] A. Jain, N. Nandakumar and A.P. James, "Machine Learning in Precision Agriculture: A Review", *IEEE Access*, Vol. 9, pp. 48692-48712, 2021.

[26] G. Singh, R. Kaur and H. Kaur, "Crop Recommendation System using Hybrid Ensemble Learning Model", *Procedia Computer Science*, Vol. 167, pp. 2201-2210, 2020.

[27] R. Chlingaryan, S. Sukkarieh and B. Whelan, "Machine Learning Approaches for Crop Yield Prediction and Nitrogen Status Estimation in Precision Agriculture: A Review", *Computers and Electronics in Agriculture*, Vol. 151, pp. 61-69, 2018.

[28] M.M. Rahman, M.F. Rahman, M.A. Rahman and M.A.M. Azad, "Crop Prediction using Machine Learning Approach", *Proceedings of International Conference on Robotics, Electrical and Signal Processing Techniques*, pp. 400-403, 2019.

[29] H.R. Abdulsalam, M.A. Sulaiman and M.F. Hassan, "Soil Fertility Prediction using Machine Learning Algorithms", *Journal of King Saud University - Computer and Information Sciences*, Vol. 34, No. 6, pp. 3137-3146, 2022.

[30] A.K. Srivastava and A.K. Singh, "Smart Farming: IoT based Smart Sensors Agriculture Stick for Live Temperature and Moisture Monitoring using Arduino, Cloud Computing and Solar Technology", *Computational Intelligence in Data Mining*, Vol. 712, pp. 179-189, 2018.

[31] R.S. Mulla, "Twenty Five Years of Remote Sensing in Precision Agriculture: Key Advances and Remaining Knowledge Gaps", *Biosystems Engineering*, Vol. 114, No. 4, pp. 358-371, 2013.

[32] R. Behera, A. Jena and S. Nayak, "A Deep Learning Approach for Crop Disease Prediction using Hybrid CNN-LSTM Model", *Proceedings of International Conference on Computer Science, Engineering and Applications*, pp. 1-6, 2020.