A Machine Learning Approach to Smart Farming

[1]Prof. Shailaja Udtewar, [2]Prajakta Dagade, [3]Piyush Khatpe, [4]Rohit Shembekar

[1]Professor, [2,3,4]Student
Electronics and Telecommunication Engineering,
Xavier Institute of Engineering
Mumbai, India.

***Abstract*- Five popular machine learning algorithms are thoroughly compared in this study: Random Forest, Decision Tree, Naive Bayes, K-Nearest Neighbors (KNN), and Logistic Regression. The objective of the study is to assess and compare these algorithms' performance on various datasets, including factors like accuracy, precision, recall, and F1 score. All algorithms were built, adjusted, and trained in accordance with standard protocols, and experimentation was used to identify performance metrics. The outcomes demonstrate the algorithms' performance. High-dimensional, complex data sets work well for Random Forest, but Decision Trees are easier to grasp. When analyzing categorical data, Naive Bayes is robust against extraneous characteristics. When local patterns are significant, KNN works well; nevertheless, for binary data, logistic regression becomes an invaluable tool.**

***Index Terms*- Machine Learning, Random Forest, Decision Tree, Naive Bayes, K-Nearest Neighbors, Logistic Regression, Comparative Analysis, Performance Metrics.**

I. INTRODUCTION

Machine learning has emerged as a transformational force in recent years, altering how it tackles complicated issues in a variety of disciplines. This technological revolution is powered by a broad range of machine learning algorithms, each designed to extract patterns, make predictions, and uncover insights from vast datasets. From image recognition to natural language processing, these algorithms have exhibited unrivaled powers, allowing machines to learn and adapt without explicit programming. This research paper will delve into the complex environment of machine learning algorithms, providing a thorough understanding of their fundamental concepts, applications, and the growing approaches that drive their efficacy. As our reliance on data-driven decision-making grows, academics, practitioners, and enthusiasts must grasp the complexities of machine learning algorithms.

In the ever-changing field of artificial intelligence, machine learning algorithms have developed as strong tools for extracting useful insights and making predictions from large datasets. This study analyzes five widely used machine learning algorithms: Random Forest, Decision Tree, k-Nearest Neighbours (KNN), Naive Bayes, and Logistic Regression. Each algorithm represents a unique approach to problem solving, with distinct benefits and drawbacks in different applications. This research aims to provide practitioners and researchers with insights into the strengths, weaknesses, and optimal use cases of these machine learning algorithms through a thorough examination of these algorithms. Understanding the complexities of these algorithms allows stakeholders to make informed decisions when choosing the best model for their specific applications.

II. RELATED WORK

Varun Prakash R et al [1], have proposed a system to get appropriate crop recommendations, and pre-trained models are employed. Among various Random Forests, XGBOOST, and Naıve Bayes have greater accuracy rates—roughly 99 percent—than other trained models. Furthermore, the optimal decision is made utilizing a polling system because different Machine Learning (ML) models yield varied proposal outcomes.

M Chandraprabha et al [2], this research has analyzed different types of soil suitable for a particular crop. Algorithms like Naive Bayes, Random Forest, Instance-bases learning with parameter k (Ibk), and Bayes Net were used. Among these, the accuracy of Ibk and Random Forest was the highest at around 97 percent.

Kasara Sai Pratyush Reddy [3], the author has expressed the need of a supervised learning algorithm in which the machine is trained with previously labeled data whose answers are known in order to identify patterns in the data. It examines the many types of data, the solutions to every issue, and looks for patterns in them.The author has used the decision tree learning algorithm, which belongs to the class of supervised machine learning algorithms, analyzes real-time data, produces a yes/no answer, and emails the farmer with the outcome. By making this choice, a farmer can select for themselves to water the crop just when necessary and prevent water waste.

Arun Kumar et al [4], in this work crop yield categorization, has been carried out utilizing artificial neural networks to group based on yield productivity. Additionally, the productivity range will be specified. To determine the real crop production and expected cost, regression analysis will be done.

Sachee Nene et al [5], suggested two machine learning algorithms, the Back Propagation Network and the Kohonen Self-Organizing Map, which were compared to see which produced the most accurate results. It examines the nutrients in the soil as well as crop productivity specific to the area. Additionally, it suggests the right fertilizers to use with a specific crop.

Sujata Terdal et al [6], the author has employed three distinct supervised learning methods, including decision trees, K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). It offers guidance on how to use data analysis on the sugar cane crop data collection. There are three types of data: sets related to soil, rainfall, and crop production. This data set includes a range of criteria that can be used to classify information into different categories, determine crop status, and undertake supervisory training on data sets gathered from the agriculture domain. With a 99% accuracy rate and an extremely low mean square error, the decision tree method performs better. This approach will assist in lessening the issues that farmers confront.

P.Priya et al [7], the author demonstrates how the Random Forest algorithm can be used to estimate agricultural yields with accuracy.With the fewest models, the Random Forest algorithm produces the most crop yield models.It is appropriate for large-scale agricultural planning's forecast of crop yield. This encourages farmers to choose the best crop to grow, which leads to the development of creative ideas in the agricultural industry.

V.Sathya Narayanan et al [8], the prepared dataset was trained and tested using all five of the machine learning techniques that the author employed. The best-fitting model that characterizes a dependent variable based on a set of independent variables is found using a logistic regression model. The Bayes theorem's probability is calculated using the Naive Bayes algorithm. Every independent variable influences the likelihood of the result. Support Vector Machine is an algorithm with the greatest possible distance between the border points of each class, this algorithm determines the ideal hyperplane to divide two classes. The decision tree is an algorithm that represents a set of rules that result in a class or value using an analogy to a tree structure. Similar to DT, Random Forest is an algorithm that builds numerous little trees. When compared to other algorithms, the Random Forest Regression Algorithm yields 98.1818%, as demonstrated in the study.

Sk Al Zaminur Rahman et al [9], research has been carried out to classify the soil types with the help of a decision tree classifier and naive Bayes classifier to produce good yield and profit. The accuracy attained by Naive Bayes was 98% which is found to be the highest value.

Kale, Shivani S et al [10], this model suggests using the gradient descent and Rectified linear activation function (RELU) activation function, the backpropagation model that has been suggested aims to lower the mean square error (MSE). Every layer's learning rate, which is 0.001, is maintained constant. The error will decrease as the number of epochs increases. This outcome is taken into consideration when determining the crop's success rate relative to other crops. Depending on the weather and district, the farmer will be advised to plant the best crop.

Haque, Fatin Farhan et al [11], in the proposed system Support vector regression and linear regression, are the supervised models that are taken into account for yield prediction. Regression models are regarded as a component of the non-linear data that is being built to assess how well various categories' yield estimates perform. The accuracy of the assumption is determined by the hyperplane built in both models to balance the test sets.

Aswathi Malanthra et al [12],the primary objective of this research project is to forecast agricultural production for the upcoming growing season by leveraging historical data on crop yields and weather patterns. Many machine learning (ML) techniques are used in the analysis, such as the Decision Tree (DT) study that came from publicly available data sets that comprised historical data from multiple crop kinds over several years. The study's findings demonstrate that machine learning models predict agricultural production for the following season with a level of accuracy that seems promising. It looks at the challenges and disadvantages of utilizing machine learning to forecast crops, as well as the analysis required for additional research.

III. PROPOSED WORK

1) Import Dataset: [https://github.com/deepaksaipendyala/CropPrediction-Machine-Learning-model] 2200 entries with 7 parameters like temperature, humidity, pH, N, K, P and soil moisture along with the labels(Crops).

2) Train the machine learning model: DeThe algorithm evaluation is carried out using the Python compiler. Decision trees, Logistic Regression, Na¨ıve Bayes, Random Forest, and KNN are the algorithms used here. Python compilers are used to implement these algorithms.Decision Trees algorithm is one of the generally used supervised learning algorithms due to its understand-ability, accuracy detection and cost. All the data has to be numerical for a decision tree. Feature columns must be separated from the target columns. In the decision tree the data-set is divided into smaller subsets. Initial nodes represent the entire population of data, then consequently splitting of data is done at each node. Imported dataset and divided it into training and testing (70-30) and applied the decision tree classifier and calculated its accuracy for different depths (no. of splits) and plotted graph for training accuracy and testing accuracy concerning

its depth. Calculated the confusion matrix and found the True Positive (TP), True Negative (TN), False Positive(FP) and False Negative (FN) and found some parameters like Precision, Recall and F1 score.
3) Similarly, all the other algorithm models are trained.
4) Finally, plot the graph of all the parameters of all the algorithms as shown in Fig. 19,20,21,22.

IV. DATASET

GitHub Dataset link: https://github.com/deepaksaipendyala:
The dataset was retrieved from GitHub. The temperature, humidity, pH sensor, soil moisture, and nitrogen (N), phosphorus (P) and potassium (K) values are all included as column vectors in the dataset. Moreover, it has 2200 reading rows with labels from 22 different classes. The appropriate crops are labeled for each of these values. The information provides requirements for various crops in terms of temperature, humidity, pH sensor, soil moisture, and nitrogen (N), phosphorus (P) and potassium (K) values.
Figure 1 shows the given dataset information used.

```
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   N            2200 non-null   int64
 1   P            2200 non-null   int64
 2   K            2200 non-null   int64
 3   temperature  2200 non-null   float64
 4   humidity     2200 non-null   float64
 5   ph           2200 non-null   float64
 6   rainfall     2200 non-null   float64
 7   label        2200 non-null   object
dtypes: float64(4), int64(3), object(1)
```

Fig. 1 Dataset Information

V. SOFTWARE COMPONENTS

The algorithm evaluation is carried out using the Python compiler. Decision trees, Logistic Regression, Naıve Bayes, Random Forest, and KNN are the algorithms used here. Python compilers are used to implement these algorithms.
Google Colaboratory: Rich text, LaTeX, pictures, and executable code can all be included in a single document using Colab notebooks. Your Google Drive account is where your custom Colab notebooks are kept after you create them. Colleagues or friends may readily view and even edit your Colab notebooks, leaving comments on your work.

VI. MACHINE LEARNING ALGORITHMS

A. DECISION TREE

Decision trees are prediction models that, based on the most significant attributes, recursively split a dataset into smaller groupings. These trees split at each node according to the feature that most effectively divides the data, starting with a root node that represents the entire dataset. The process keeps going, creating branches and nodes until a stopping condition is met, like reaching a particular depth or stopping when splitting doesn't get any better. Every route that leads from the root to a leaf node denotes a choice point or rule. In order to create predictions, fresh data follow these paths according to the feature values they contain, finally arriving at a leaf node that yields the expected result. Decision trees are useful tools for predictive analysis in a variety of industries because they are excellent at interpreting complicated relationships within data.
A decision tree is a graphic depiction of a set of options and potential consequences. It resembles a tree, with each node representing a decision and each branch reflecting the outcome of that option. The leaves of the tree represent final judgments or projections. Decision trees are built by repeatedly splitting data into smaller subsets depending on a given trait, maximizing the information acquired from each partition.
Essential Terminologies to Implement Decision Tree:

- Root Node: The root node of the decision tree represents the complete dataset.
- Branch Nodes: Internal nodes that represent the decision points where a specific attribute is used to split the data.
- Leaf Nodes: The last classification or terminal nodes that represent predictions.
- Decision Node: Nodes formed by the division of root nodes.
- Sub-Tree: A subtree is a subsection of a decision tree, just as a sub-graph is a subset of a graph. It stands for a given area of the decision tree.
- Decision rules: Specify how data is split at each branch node.
- Attribute selection: Choosing the property that contains the most information for each split.
- Splitting Criteria: To determine the best split, metrics are used.

Basic Concept to use Decision Tree:

- The entire training set was initially considered and taken as ”the root.”
- For information gain, qualities are assumed to be categorical and continuous.
- Records are scattered recursively according to attribute values.
- For classifying attributes as internal or root nodes, it uses statistical methods.

Working:

- Starting at the Root: The algorithm begins with the ”root node,” which represents the entire dataset.
- Posing the Best Questions: It looks for the key feature or query that splits the data into the most relevant subgroups. This is similar to posing a question at a tree fork.
- Branching out: It generates new branches by breaking the data into smaller portions based on the response to the query. Every branch of the tree represents a potential path.
- Repeating the process: The algorithm keeps asking questions and dividing the data at each branch until it reaches the final ”leaf nodes,” which stand for the anticipated conclusions or classifications.
- Stop Splitting: It use the max depth option to specify the maximum depth of our decision tree. The higher the max depth value, the more intricate your tree will get.

Advantages:

- Interpretability: Decision trees are simple to grasp and interpret, making them an invaluable tool for both professionals and non-experts. A decision tree’s graphical representation is understandable and matches human decision making processes.
- Decision trees don’t assume anything about how the data will be distributed. When dealing with numerical and categorical data, they don’t require a lot of data preparation.
- Handles Non-Linearity: Decision trees can model complex relationships between features and the target variable, including non-linear relationships. They are capable of capturing patterns that linear models might miss.
- Importance of Features: Decision trees offer details on the significance of various features in forming predictions. Selecting features and comprehending the underlying dynamics of the data can both benefit from this.
- Ease of Handling Missing Values: Decision trees can handle datasets with missing values. They can still make predictions for instances where some features have missing values without requiring imputation.

Disadvantages:

- Overfitting: Decision trees are prone to overfitting, particularly when they are large and complex. A deep tree may perfectly suit the training data but fail to generalize to new, previously unknown data.
- High Variance: Variations in the training data have an impact on decision trees. Variations in the training set might lead to significant variation since small changes can produce diverse tree architectures.

Why to use Decision Tree: A decision tree’s function is to use historical data to inform future decisions or forecasts. Determining the most important characteristics that influence the choice at hand and comprehending the connections between the input factors and their results are beneficial.

Figure 2 shows the Decision Tree Algorithm Classifier.

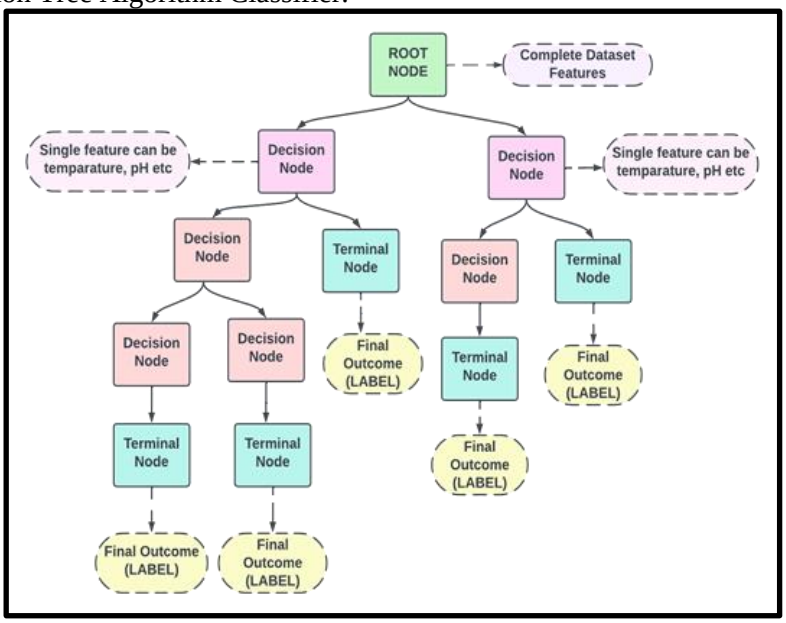

Fig. 2 Decision Tree Classifier

B. RANDOM FOREST

During training, Random Forest, an ensemble learning technique based on decision tree classifiers, generates a huge number of decision trees. It operates by choosing a random selection of attributes for each split and training multiple decision trees on a random subset of the dataset (bootstrap samples). The trees are formed independently and simultaneously during training. The forecasts from each tree are used by the Random Forest to create a final prediction when making predictions.Regression involves averaging the outputs from each tree, while classification typically requires a majority vote among the trees. Using an ensemble approach improves forecast accuracy, robustness, and generalization to new data while decreasing overfitting in individual decision trees. Because of its adaptability and widespread use across multiple fields, Random Forests are especially useful for handling noisy and high-dimensional datasets and for revealing the relevance of individual features.

Basic Concept to use Random Forest:

- Because the random forest combines many trees to estimate the dataset's class, some decision trees may predict the correct output while others may not. However, when all trees are joined, each predicts the correct outcome.
- There should be some actual values in the dataset's feature variable so that the classifier may predict correct outcomes rather than a predicted result.
- The predictions from each tree must be exceedingly low in correlation.

Working:

- The two stages of Random Forest's operation are the creation of the random forest from the combination of N decision trees and the prediction of each tree generated in the first phase.
- From the training set, choose K data points at random.
- Create decision trees for the selected data points (subsets).
- Select the number N for the decision trees you wish to construct.
- Carry out Steps 1 and 2.
- Find the predictions made by each decision tree for the new data points, then assign them to the side that got the majority of votes.

Advantages:

- High Accuracy: Random Forest is often quite accurate in both classification and regression problems. It combines the predictions of numerous decision trees to reduce overfitting and improve generalization to new data.
- Reduced Overfitting: Random Forest reduces overfitting by combining numerous trees' predictions, unlike individual decision trees. This improves its robustness and ability to generalize to previously unexplored data.
- Feature Importance:Random Forest calculates a feature importance score, which indicates the contribution of each feature to the model's predictions. This information can help with feature selection and understanding how different variables affect the outcome.
- Handles Missing Values and Outliers: Random Forest can accept missing numbers and is less susceptible to outliers. It is noise-resistant and can make accurate predictions even when given incomplete or noisy data.
- Parallelization: Since individual trees can be trained independently, Random Forest is easily parallelizable. It is hence appropriate for large datasets and computationally efficient.

Disadvantages:

- Lack of Interpretability: While Random Forest provides high accuracy, the ensemble nature of the model can make it less interpretable than a single decision tree. It might be challenging to explain the logic behind the combined predictions of multiple trees.
- Computational Complexity: To train a large number of trees causes high computational cost. This might limit its practicality in real-time applications or on devices with resource constraints.

Why to use Random Forest: In comparison to other algorithms, it requires less training time. It operates effectively even with a large dataset and predicts output with high accuracy. Even in situations where a sizable amount of data is missing, it can nevertheless remain accurate.

Figure 3 shows the Random Forest Algorithm Classifier.

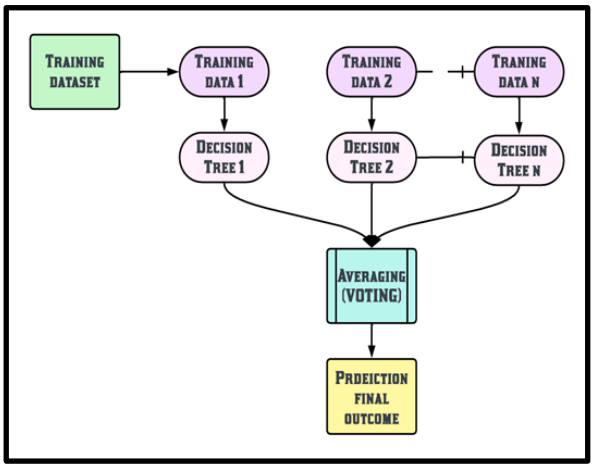

Fig. 3 Random Forest Classifier

C. NAIVE BAYES

Naive Bayes is a probabilistic machine learning technique based on Bayes' theorem that is well-suited for classification tasks. It works under the assumption of feature independence, which means that each characteristic contributes independently to the probability of a specific class. This algorithm calculates the likelihood that an input belongs to a specific class by multiplying the individual conditional probabilities of each characteristic within that class. During training, Naive Bayes computes the likelihood of each feature occurring in each class, as well as the prior probability of each class in the dataset. When making predictions, the class with the highest probability is chosen as the predicted class for the given input.Despite its simplistic assumption of feature independence, Naive Bayes excels in a variety of applications, including text classification, spam filtering, and sentiment analysis, thanks to its computational efficiency and effectiveness on tiny datasets. Naive Bayes classifiers are a family of algorithms based on Bayes' Theorem.

Basic Concept to use Naive Bayes:

- Naive: It gets its name from the naive assumption that one feature's occurrence stands alone from the occurrence of other features.
- Bayes: Its reliance on the Bayes' Theorem is the basis for its moniker.
- Feature independence: Depending on the class name, the data's characteristics are conditionally independent of one another.
- Features that are continuous have a normal distribution: It is presumed that a feature is normally distributed within each class if it is continuous.
- Multinomial distributions are found in discrete features: Within each class, a discrete feature is presumed to have a multinomial distribution.
- Each attribute is thought to contribute equally to the class label prediction, making them all equally significant.
- No omitted information The data should not contain any missing values.
- Bayes' Theorem: The Bayes theorem, sometimes referred to as Bayes' law or Rule, is a tool for estimating a hypothesis's likelihood given available information. The conditional probability determines this.

$$P(A|B) = P(B|A)P(A) / P(B) \quad (1)$$

Where,

In posterior probability represents the likelihood of hypothesis A regarding the observed event B.
Likelihood probability is the likelihood that a hypothesis will come true based on the evidence.
Prior Probability, or P(A), is the probability of a hypothesis before the evidence is observed.
The probability of evidence, or marginal probability, is P(B).

Working:

- Frequency tables are created using the given dataset.
- Create a likelihood table by determining the odds of the supplied features.
- Now, determine the posterior probability using the Bayes theorem.

Advantages:

- Simplicity and Speed: Naive Bayes is a simple and computationally efficient algorithm. It is easy to implement and works well with large datasets. Training and making predictions with Naive Bayes are generally fast processes.
- Effective for Text Classification: Naive Bayes is particularly effective for text classification tasks, such as spam filtering or sentiment analysis. Its ability to handle high dimensional data and the assumption of independence between features makes it suitable for such applications.
- Requires Less Training Data: Naive Bayes can operate efficiently with a small amount of training data.

● Handles Categorical Features Well: Naive Bayes is ideal for categorical data. It can handle numerical and categorical features, making it useful in a variety of datasets.
● Probabilistic Framework: Naive Bayes makes probabilistic predictions. It not only forecasts the class label, but also the likelihood of the prediction, which can help with decision-making and understanding the model's confidence.

Disadvantages:

● Sensitivity to Feature Scaling: Naive Bayes is sensitive to the size of numerical features. If features have various scales, the performance may be unsatisfactory. Feature scaling (e.g., normalization) may be essential.
● Cannot Handle Non-Numeric Data Directly: While Naive Bayes can handle categorical features, it requires numerical values for computations. For non-numeric data, preprocessing steps such as encoding or transformation are needed.

Why to use Naive Bayes: Particularly, naive Bayes classifiers perform admirably in spam filtering and document classification, even with their oversimplified assumptions. In real world situations, their effectiveness, quickness, and capacity to operate with little data make them valuable, making up for their naive independence assumption.
Figure 4 shows the Naive Bayes Algorithm Classifier.

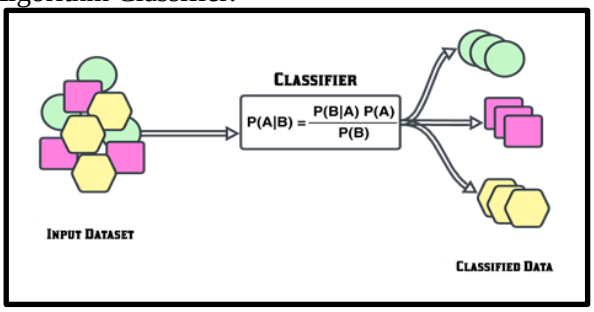

Fig. 4 Naive Bayes Classifier

D. LOGISTIC REGRESSION

A statistical method for binary classification that forecasts whether a result falls into a particular class is called logistic regression. It functions by imitating the relationship that exists between one or more independent variables and the dependent binary variable. Through the use of a logistic function on a linear combination of input features, the approach produces an output (a number between 0 and 1) that indicates the likelihood that the input falls into a specific class. The algorithm adjusts the model parameters during training in order to minimize the difference between the actual class labels and the anticipated probability. In order to categorize input into one of two groups while predicting new data, logistic regression first calculates the probability and then sets a threshold, usually 0.5. Because of its simplicity, effectiveness, and capacity to produce probability estimates, logistic regression is a widely used method in many domains, especially when dealing with binary classification problems. When a decision threshold is included, logistic regression turns into a classification method. A crucial component of logistic regression is threshold setting, which is contingent on the nature of the classification problem.The precision and recall levels have a significant influence on the threshold value determination. Although it is ideal for both precision and recall to be 1, this is rarely the case.

Essential Terminologies to Implement Logistic Regression:

● Independent variables: The predictor elements or input qualities that are used to make predictions about the dependent variable. Dependent variable: The variables attempting to predict in a logistic regression model.
● The formula that illustrates the relationship between the independent and dependent variables is called the logistic function. The logistic function transforms the input variables into a probability value between 0 and 1, indicating the likelihood that the dependent variable will be either 0 or 1.
● Odds: The ratio of something happening versus something not happening.
● Log-odds: The natural logarithm of the odds is the logodds, sometimes referred to as the logit function. The dependent variable's log chances are described in logistic regression as a linear mixture of the independent variables and the intercept.
● Coefficient: The estimated parameters of the logistic regression model indicate the relationship between the independent and dependent variables.
● Intercept: The log odds when all independent variables are equal to zero are represented by the constant factor.
● Maximum likelihood estimation: The process of estimating the logistic regression model's coefficients that maximizes the chance of observing the data given the model.

Basic Concept to use Logistic Regression:

● Separate observations: Every observation stands alone from the others. indicating that no relationship exists between any of the input variables.
● Dependent variables that can only take two values: It is assumed that the dependent variable must be binary, also known as dichotomous. Softmax functions are utilized for more than two categories.

● Relationship between log odds and independent factors: The log odds of the dependent variable and the independent variables should be linear.
● No outliers: The dataset shouldn't contain any outliers.
● Large-scale sample size: There is enough data in the sample.

Working:

● The outcome of a categorical dependent variable is predicted using logistic regression. The outcome must therefore be a discrete or category value. The probabilistic values, which range from 0 to 1, are provided in place of the exact values.
● In logistic regression, two maximum values are predicted by fitting a "S" shaped logistic function, as opposed to fitting a regression line (0 or 1).
● The logistic function curve displays the likelihood of a number of variables, like whether the cells are malignant or not, if a mouse is fat according to its weight, and more.

Advantages:

● Interpretability: It is easy to interpret the results of a logistic regression. The model's coefficients show the impact of each variable, providing a clear picture of degree and direction of the association between each outcome and the predictor variable. Probabilistic Predictions: Logistic Regression produces probabilities, which are valuable in situations where understanding the likelihood of an event is critical. This makes it ideal for binary classification challenges.
● Works well with Linearly Separable Data: When the data is linearly separable, logistic regression tends to perform well. It is capable of modeling linear relationships between the log odds of the outcome and the features with effectiveness.
● Low Computational Requirements: Logistic Regression is computationally efficient, especially when dealing with large datasets. Training and prediction times are generally lower compared to more complex models, making it suitable for real-time applications.
● Less Prone to Overfitting: Logistic Regression is less susceptible to overfitting, particularly when the number of features is minimal. It eliminates the complexities of more advanced models, making it more reliable when dealing with small amounts of data.

Disadvantages:

● Assumes Linearity: The assumption of logistic regression is that the independent variables and the outcome's log odds have a linear relationship. In cases where the underlying relationship is nonlinear, the model's representation of it may not be precise.
● Limited Modeling of Complex Relationships: Logistic Regression may struggle with capturing complex relationships in the data, especially if interactions or nonlinearities are present. More flexible models might be needed for such scenarios.

Why to use Logistic Regression: Training, interpreting, and implementing logistic regression is simpler. It swiftly classifies unknown records. The dataset performs best when it is linearly separable. Model coefficients can be used to determine how important a feature is.

Figure 5 shows the Logistic Regression Algorithm Function

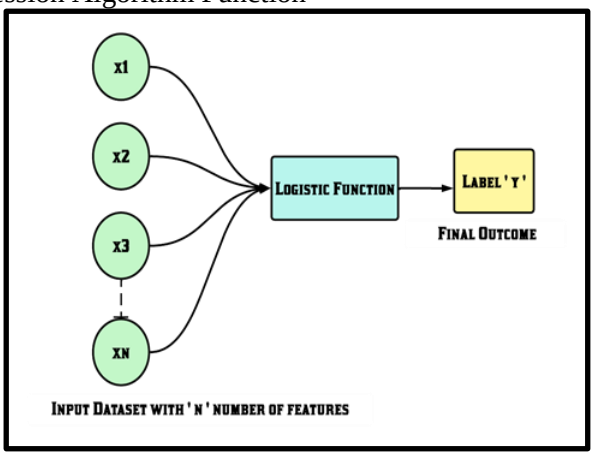

Fig. 5 Logistic Regression Function

E. K-Nearest Neighbors (KNN)

It is a simple yet powerful regression and classification method. It is predicated on the idea that comparable instances are located close to one another in a feature space. When a new data point needs to be forecasted in a KNN, the algorithm finds the K nearest neighbors using a distance metric (such the Euclidean distance) among the attributes of the training data. In order to classify, KNN counts the K neighbors' class labels and predicts using the majority class. Regression

uses the average of the K nearest neighbors' goal values to forecast a continuous variable.The performance of the algorithm is determined by the K value: larger K values smooth out the decision boundary but may increase bias, whereas smaller K values are more sensitive to noise but have lower bias. KNN requires calculating the distances to every training sample. However, it is intuitive, easy to implement, and suitable for small to medium-sized datasets.

Basic Concept to use KNN:

- KNN does not make any assumptions about the underlying data because it is a non-parametric algorithm.
- Because it saves the dataset and acts on it while classifying, it is also known as a lazy learner algorithm. This is because it does not immediately pick up knowledge from the training set.
- When new data is received, the KNN algorithm categorizes it into a category that is quite similar to the previously stored dataset, which is all that is needed during the training phase.
- When defining the number of neighbors in the KNN algorithm, the value of k is highly important. In the k nearest neighbors (k-NN) procedure, the input data should be used to determine the value of k.
- A greater value of k would be preferable if the input data contained more noise or outliers. Selecting an odd value for k is advised in order to prevent ties in the classification.
- The best k value for the given dataset can be chosen with the use of cross-validation techniques.

Working:

- The K-Nearest Neighbours (KNN) approach predicts the label or value of a new data point using the similarity principle and the labels or values of its K nearest neighbors in the training dataset.
- Calculating K's Ideal Value: K denotes the number of neighbors to be considered when making a prediction.
- Distance calculation: The Euclidean distance indicates how similar the target and training data points are. Every data point in the collection is judged to be a certain distance from the objective.
- Locate the Closest Neighbors: The closest neighbors are the k data points that have the least distances to the target point.
- Selecting a Classification by Voting or Regression by Taking the Average: The majority vote is used to determine the class labels in the categorization problem. The predicted class for the target data point is the one with the highest frequency among the neighbors. The class label in the regression issue is determined by averaging the K nearest neighbors' goal values. The calculated average value becomes the expected output for the target data point.

Advantages:

- Simple and Intuitive: KNN is a simple and easy-to understand algorithm. It does not make strong assumptions about the underlying data distribution, making it suitable for a wide range of scenarios.
- Non-Parametric: KNN is a non-parametric algorithm, which means it makes no assumptions about the form of the data distribution. This makes it adaptable and capable of processing data with complicated patterns.
- Adaptability to Local Patterns: KNN can adapt well to local trends in data. It is especially beneficial in cases where the decision boundaries are not linear and the relationship between features and the target variable is complex.
- Absence of a Training Phase: KNN lacks a training phase. The model only keeps the training data, and its predictions are dependent on how close new samples are to the existing data points. When working with dynamic datasets, this is helpful. • Effective for Small Datasets: KNN can perform well with small datasets, and it tends to be robust in situations where there is a finite supply of training data.

Disadvantages:

- Computational Complexity: Calculating the distance between the query instance and all training instances can be computationally expensive, particularly as the dataset grows in size. This can affect the algorithm's efficiency.
- Sensitive to Irrelevant Features: KNN is sensitive to features that are irrelevant or redundant. Including irrelevant information may introduce noise and have an impact on predictive accuracy.

Why to use KNN: Using a distance metric such as Euclidean distance, the K-NN algorithm finds the K nearest neighbors to a given data point. The class or value of the data item is then determined by the majority vote or the average of the K neighbors. By using this technique, the algorithm can adapt to different patterns and anticipate results based on the local structure of the data.

Figure 6 shows the KNN Algorithm.

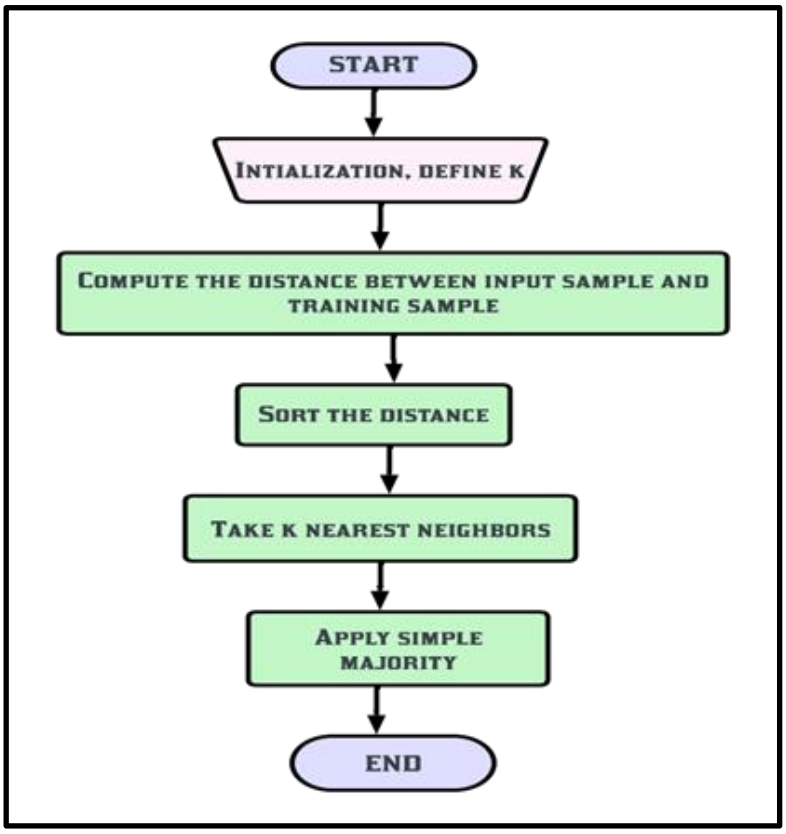

Fig. 6 K-Nearest Neighbor

VII. RESULT

Confusion Matrix:

A confusion matrix gives a summary of how well a machine learning model performed on a set of test data. It serves as a means of displaying the number of based on the model's predictions, both precise and im- precise. It is often used to evaluate the effectiveness of categorization models, which aim to give each input instance a categorical label. The matrix displays the number of instances that the model produced using the test data.

- When a positive data point is correctly predicted by the model, this is known as a true positive (TP).
- When a negative data point is correctly predicted by the model, this is known as a true negative (TN).
- When the model predicts a positive data point inaccurately, it results in true positives (FP).
- When a negative data point is mispredicted by the model, it results in false negatives (FN).

Figure 7 shows the Confusion Matrix of Decision Tree Algorithm.

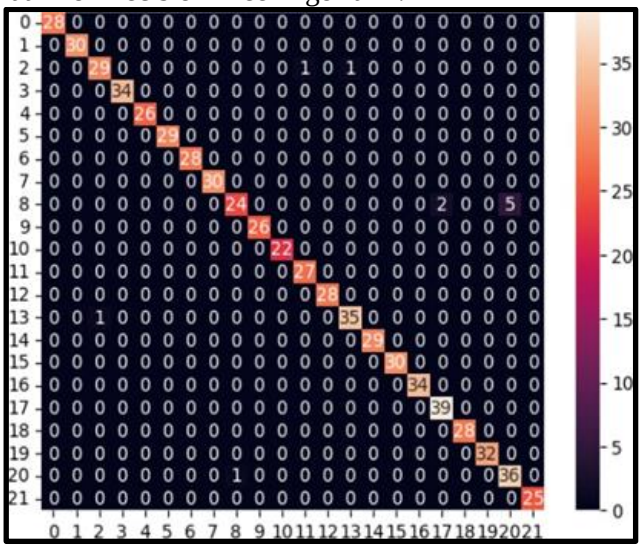

Fig. 7 Confusion Matrix of Decision Tree Algorithm.

Figure 8 shows the Confusion Matrix of Random Forest Algorithm.

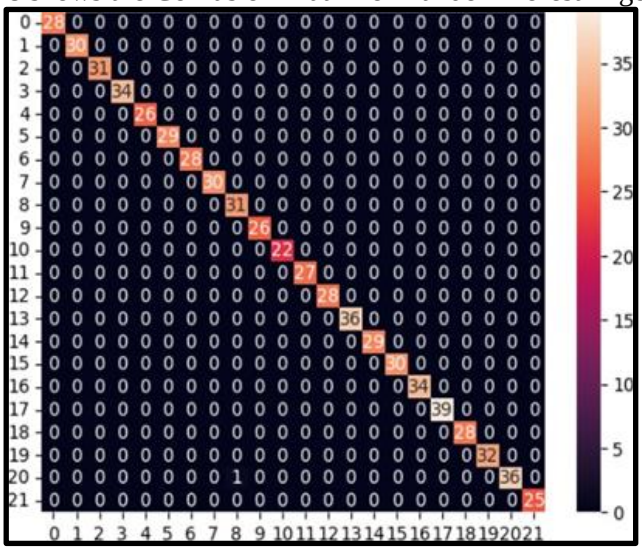

Fig. 8 Confusion Matrix of Random Forest Algorithm

Figure 9 shows the Confusion Matrix of Naive Bayes Algorithm.

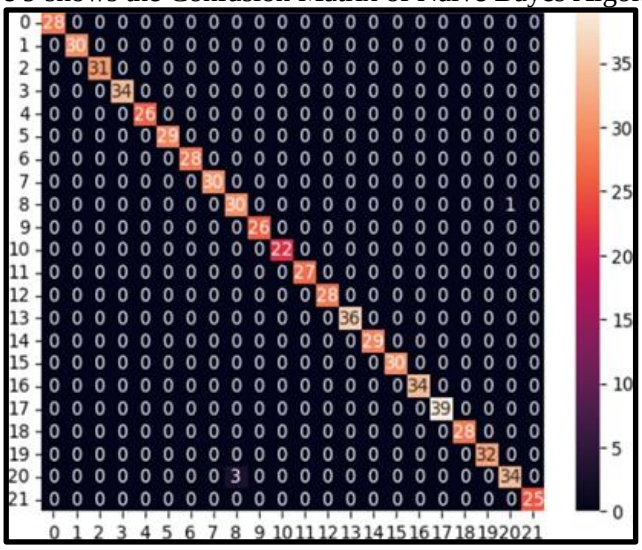

Fig. 9 Confusion Matrix of Naive Bayes Algorithm.

Figure 10 shows the Confusion Matrix of Logistic Regression Algorithm.

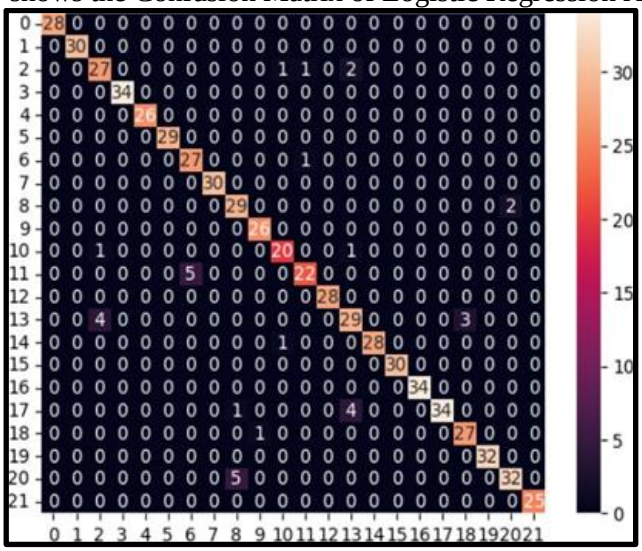

Fig. 10 Confusion Matrix of Logistic Regression Algorithm.

Figure 11 shows the Confusion Matrix of KNN Algorithm.

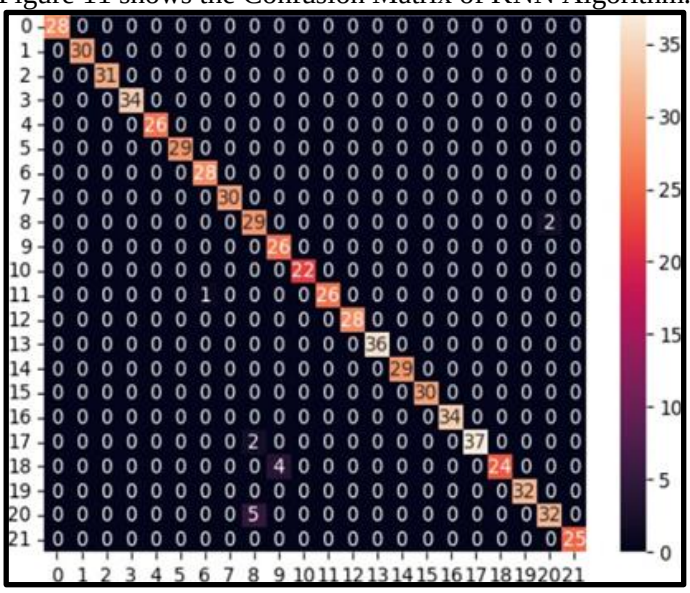

Fig. 11 Confusion Matrix of K-Nearest Neighbor Algorithm.

Accuracy: The model's accuracy is used to gauge its performance. It is calculated as the proportion of all correct instances to all instances.

Accuracy = TP+TN/TP+TN+FP+FN

- The accuracy score achieved using the Decision Tree is: 98.33
- The accuracy score achieved using the Random Forest is: 99.85
- The accuracy score achieved using the Naive Bayes is: 99.39
- The accuracy score achieved using the Logistic Regres- sion is: 95.0
- The accuracy score achieved using the KNN is: 97.88

Precision:: Precision: The precision of a model's positive predictions is gauged by its precision. Its definition is the proportion of actual positive forecasts to all of the model's positive predictions.

Precision = TP/TP+FP

- The precision score achieved using the Decision Tree is: 98.6
- The precision score achieved using the Random Forest is: 99.86
- The precision score achieved using the Naive Bayes is: 99.46
- The precision score achieved using the Logistic Regres- sion is: 95.0
- The precision score achieved using the KNN is: 98.09

Recall: A classification model's recall gauges how well it can locate each pertinent instance within a dataset. It is the proportion of true positive (TP) cases to the total of false negative (FN) and true positive (TP) cases.

Recall = TP/TP+FN

- The recall score achieved using the Decision Tree is: 98.43
- The recall score achieved using the Random Forest is: 99.88
- The recall score achieved using the Naive Bayes is: 99.48
- The recall score achieved using the Logistic Regression is: 95.3
- The recall score achieved using the KNN is: 98.04

F1 Score: A classification model's overall performance is assessed using the F1-score. It is the precision and recall harmonic mean.

F1 Score = 2*Precision*Recall/Precision+Recall

- The F1 score achieved using the Decision Tree is: 98.52
- The F1 score achieved using the Random Forest is: 99.87
- The F1 score achieved using the Naive Bayes is: 99.47
- The F1 score achieved using the Logistic Regression is: 95.15
- The F1 score achieved using the KNN is: 98.06

Table 1 Performance Parameters Of Different Algorithms

Name of Algorithm	Accuracy	Precision	Recall	F score
Decision Tree	98.33 %	98.6 %	98.43 %	98.52 %
Logistic Regression	95 %	95 %	95.3 %	95.50 %
Naive Bayes	99.39 %	99.46 %	99.48 %	99.47 %
Random Forest	99.85 %	99.86 %	99.88 %	99.87 %
KNN	97.88 %	98.0 9%	98.04 %	98.06 %

The table shows the different performance parameter values for each algorithm. From the above algorithms, RANDOM FOREST has the highest accuracy (99.85%) ,highest preci- sion (99.86%). highest recall (99.88%) and highest F score (99.87%).The figure below shows the graphical comparison of all the parameters.

Figure 12 shows Decision Tree Evaluation Parameters Line Plot

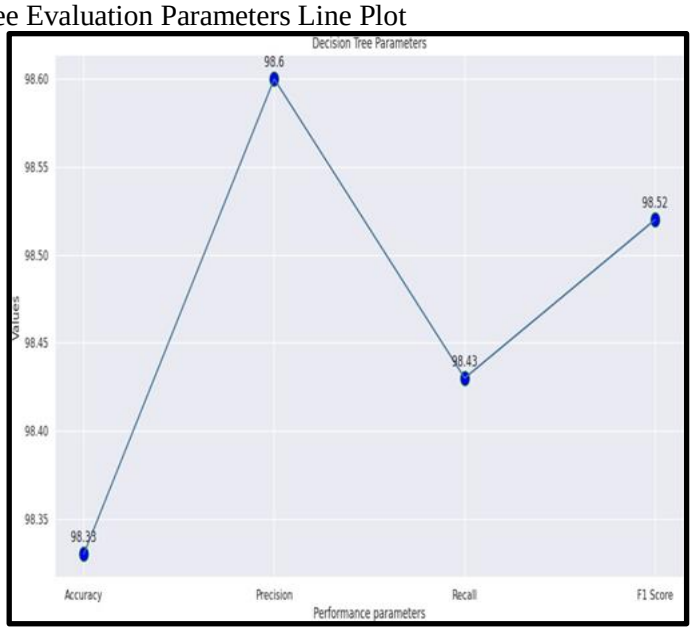

Fig. 12 Decision tree Evaluation Parameters

Figure 13 shows Random Forest Evaluation Parameters Line Plot

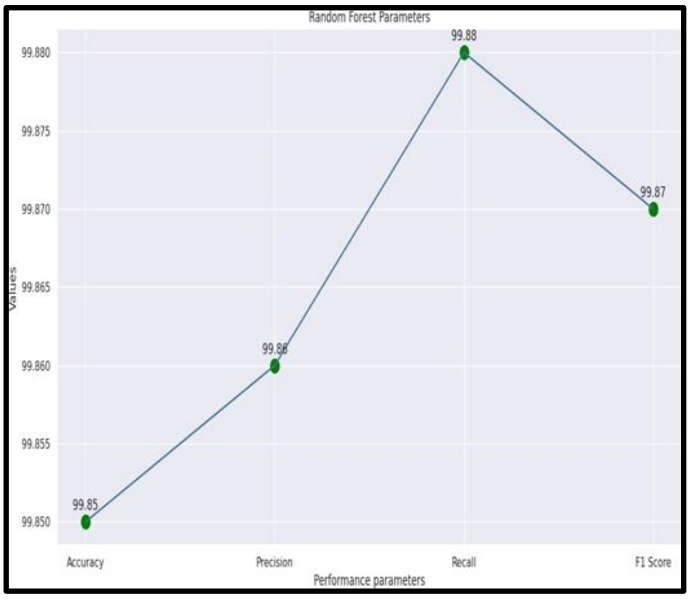

Fig. 13 Random Tree Evaluation Parameter

Figure 14 shows Naive Bayes Evaluation Parameters Line Plot

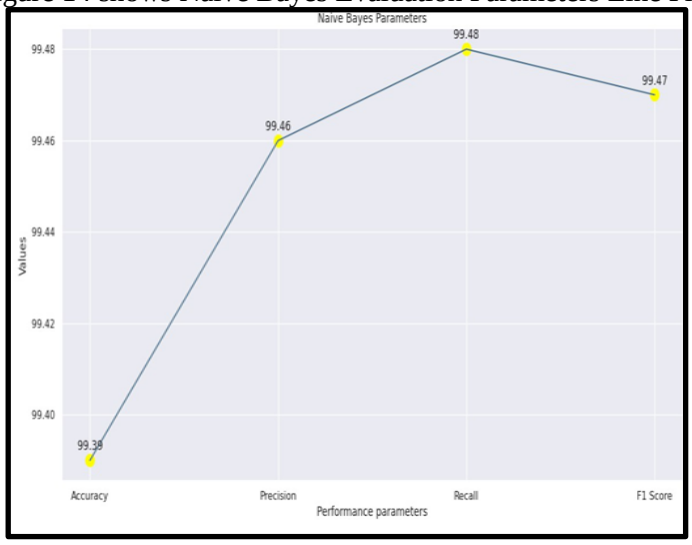

Fig. 14 Naive Bayes Evaluation Parameters

Figure 15 shows Logistic Regression Evaluation Parameters Line Plot

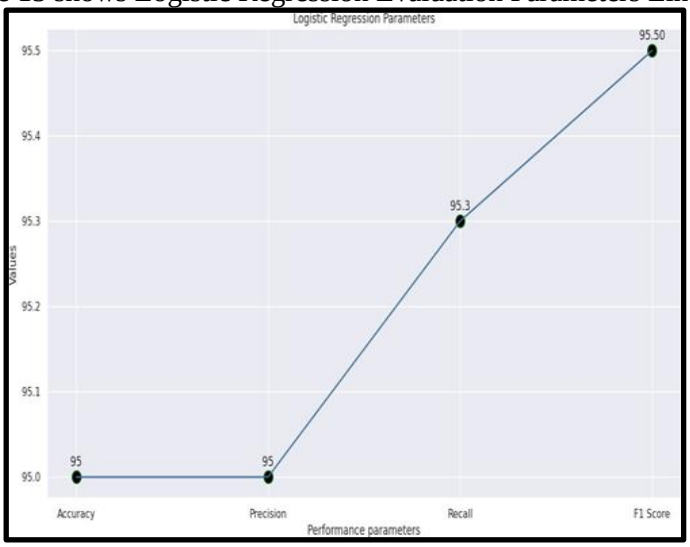

Fig. 15 Logistic Regression Evaluation Parameters

Figure 16 shows K-Nearest Neighbors Evaluation Parameters Line Plot

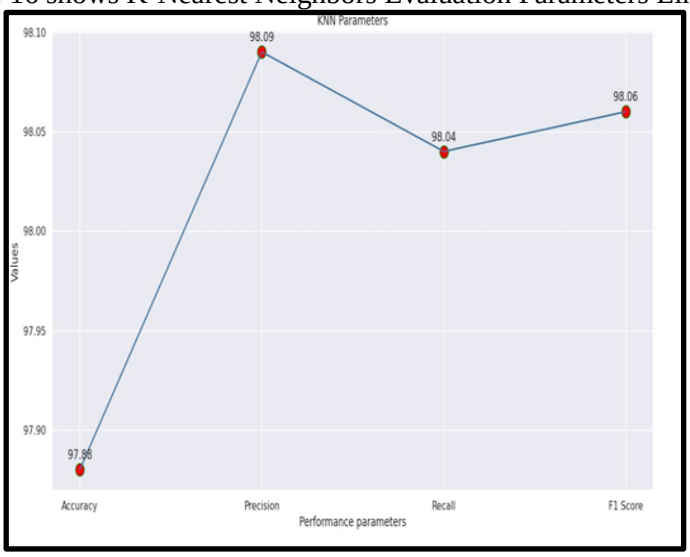

Fig. 16 K-Nearest Neighbor Evaluation Parameters

Comparison Of Evaluation Parameters

Figure 17 shows Accuracy graph of all algorithm

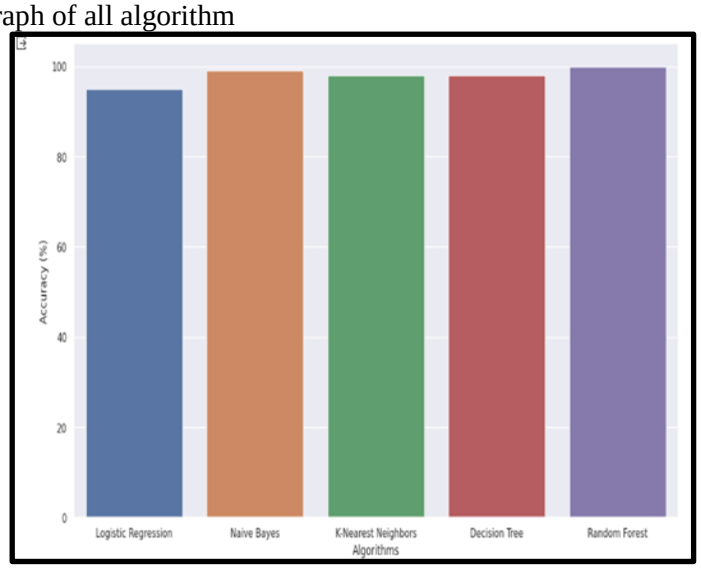

Fig. 17 Accuracy Graph

Figure 18 shows Precision graph of all algorithm

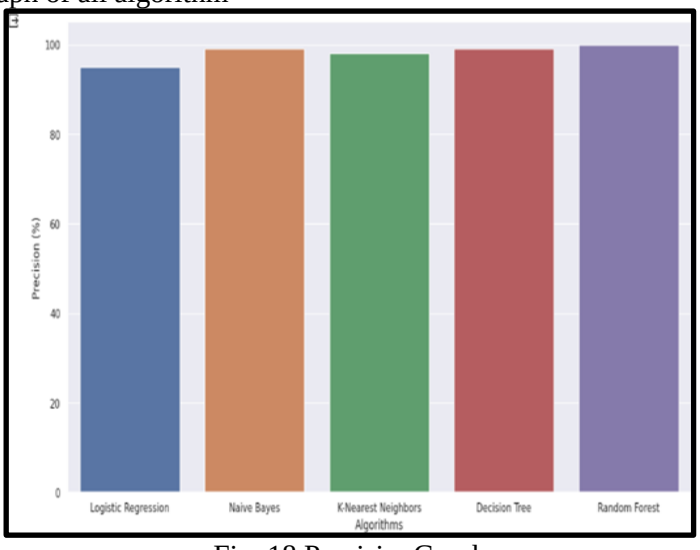

Fig. 18 PrecisionGraph

Figure 19 shows Recall graph of all algorithm

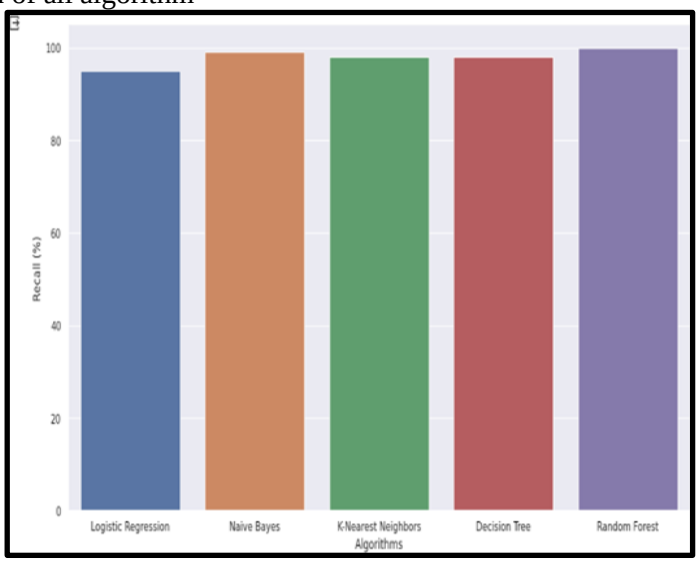

Fig. 19 RecallGraph

Figure 20 shows F1-scoregraph of all algorithm

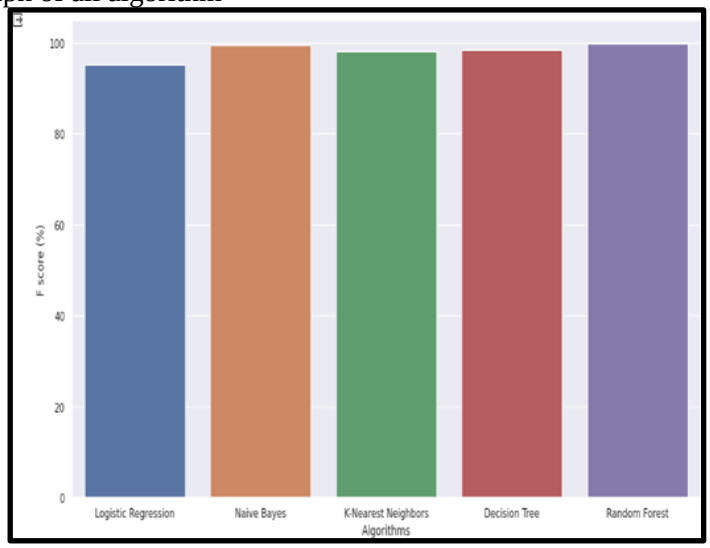

Fig. 20 F1-Score Graph

VIII. CONCLUSION

As a result, the suggested system contrasts the outcomes of various machine learning algorithms. Decision trees, Logistic regression, Naive Bayes, (KNN) K-Nearest Neighbors, and Random Forest. Each classifier receives the dataset along with its accuracy, precision, recall, and F score. The system determines that Random Forest yields the best result and can be used for prediction by comparing the outputs of all the algorithms. This work can be further enhanced by testing it on various datasets and taking into account additional parameters. When choosing a machine learning algorithm, the knowledge acquired helps to make well-informed decisions based on the task objectives and the properties of the data.

REFERENCES:

1. R Varun Prakash M Mohamed Abrith, S Pandiyarajan, ” Machine Learning Based Crop Suggestion System,” 6th International Conference on Intelligent Computing and Control Systems(ICICCS),pp. 103- 107,doi:10.1109/ ICICCS53718.2022.
2. M. Chandraprabha, Rajesh Kumar Dhanaraj,” Soil Based Prediction for Crop Yield using Predictive Analytics”,2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N),978-1-6654-3811-7/21/31.00,2021.
3. Kasara Sai Pratyush Reddy, Y Mohana Roop, Kovvada Rajeev L N, Y Mohana Roopa, Narra Sai Nandan, ”IoT based Smart Agriculture using Machine Learning ”, Proceedings of the Second International Conference on Inventive Research in Computing Applications (ICIRCA2020).

4. Arun Kumar, Naveen Kumar and Vishal Vats, “Efficient Crop Yield Prediction Using Machine Learning Algorithms", International Research Journal of Engineering and Technology (IRJET), vol. 05, no. 06, June 2018.
5. Sachee Nene and Priya. R “Prediction of Crop yield using Machine Learning”, International Research Journal of Engineering and Technology (IRJET), vol. 05, no. 02, Feb 2018.
6. Renuka and Sujata Terdal, ”Evaluation of Machine Learning Algorithms for Crop Prediction,” in International Journal of Engineering and Advanced Technology (IJEAT), vol. 8, Aug 2019.
7. P.Priya, U.Muthaiah, M.Balamurugan. “ Predicting yield of the crop using machine learning Algorithm”, IJESRT et al., 7(4): April 2018, pp 2277-2284.
8. V.Sathya Narayanan, Kavinraj N2, Kishore Kumar S3, Manoj Kumar K4, ”SOIL AND WEATHER MONITORING SYSTEM WITH CROP PREDICTION FOR FARMERS USING IOT AND MACHINE LEARNING”, October 22, 2021.
9. Sk Al Zaminur Rahman, Kaushik Chandra Mitra, S.M.Mohidul Islam, ”Soil Classification using Machine Learning Methods and Crop Suggestion Based on Soil Series”, 21-23 December 2018.
10. Kale, Shivani S., Patil, Preeti S., ”A Machine Learning Approach to Predict Crop Yield and Success Rate”,2019 IEEE Pune Section International Conference (PuneCon) - 1–5. doi:10.1109/PuneCon46936.2019.9105741.
11. Fatin Farhan Haque, Abdelgawad, Ahmed, Yanambaka, Venkata Prasanth, Yelamarthi, Kumar, ” Crop Yield Analysis Using Machine Learning Algorithms”, 2020 IEEE 6th World Forum on Internet of Things (WF-IoT) , doi:10.1109/WF-IoT48130.2020.9221459.
12. Aswathi Malanthra; Vaidehi Jadhav; Sakshi Gupta; Ritika Varpe; Vitthal S. Gutte, ”Crop Prediction Analysis using Machine Learning Techniques”, 2023 International Conference on Sustainable Communication Networks and Application (ICSCNA), 15-17 November 2023, 10.1109/ICSCNA58489.2023.10370674.