

“Smart Farming Crop Yield Prediction using Machine Learning”

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

By

G.Manoj Reddy(112015054)

Rao Suraj Rao(112015108)

Uppara Vinod Kumar(112015158)

V. Bharadwaj(112015160)

Under the Supervision of:Prof. Ritu Tiwari

Semester: 5th Semester



Name of Department: Department of Computer Science and Engineering

Indian Institute of Information And Technology, Pune

(An Institute of National Importance by an Act of Parliament)

DECEMBER 2022

BONAFIDE CERTIFICATE

This is to certify that the project report entitled “**Smart Farming Crop Yield Prediction** ” submitted by **G.Manoj Reddy** bearing the **MIS No: 112015054**, **Rao Suraj Rao** bearing the **MIS No: 112015108**, **U Vinod Kumar** bearing the **MIS No:112015158**, **V.Bharadwaj** bearing the **MIS No.112015160** in completion of his/her project work under the guidance of **Prof.Ritu Tiwari** is accepted for the project report submission in partial fulfillment of the requirements for the award of the degree of **Bachelor/Master of Technology** in the **Department of Computer Science and Engineering**, Indian Institute of Information Technology, Pune (IIIT Pune), during the academic year **2022-23**.

Supervisor’s Name

Prof.Ritu Tiwari

Professor

Department of CSE

IIIT Pune

HoD Name

Sanjeev Sharma

Assistant Professor

Department of CSE

IIIT Pune

Project Viva-voce held on

15-12-2022

Undertaking for Plagiarism

We “G.Manoj Reddy(112015054) ,Rao Suraj Rao(112015108) ,Uppara Vinod Kumar(112015158) ,V.Bharadwaj(112015160)” solemnly declare that research work presented in the **report/dissertation** titled “**Smart Farming Crop Yield Prediction using Machine Learning and Deep Learning**” is solely **our** research work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete report/dissertation has been written by **us**. I understand the zero tolerance policy of **Indian Institute of Information Technology Pune** towards plagiarism. Therefore **we** declare that no portion of my **report/dissertation** has been plagiarized and any material used as reference is properly referred/cited. I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of the degree, the Institute reserves the right to withdraw/revoke my **B.Tech/M.Tech** degree.

Students'/ Student's Name and Signature with Date

G. Manoj Reddy (112015054)
Manoj:-

R. Suraj Rao (112015108)
Suraj Rao:-

U. Vinod Kumar (112015158)
U. Vinod Kumar. 1/12/2022

V. Bharadwaj (112015160).
Bharadwaj

Conflict of Interest

Manuscript title: Smart Farming Crop Yield Prediction using Machine Learning

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

Students'/ Student's Name and Signature with Date

Problem Statement: In this study, we use the data and trends related to agriculture of the previous years for the prediction of the best crop to cultivate. The objective of model is to predict We experiment with various machine learning and deep learning models namely Random Forest Regression model, Support Vector Regression and Neural Network. The objective is to create a model/classifier that can predict Crop yield by taking different parameters like soil type, temperature, PH values etc..

Modern machine learning classifiers use conventional feature representation techniques, however the suggested deep learning classifier uses the proper number of layers, an ideal set of parameters, and gives output in optimized time with good accuracy.

Objectives:

1. To increase the accuracy of crop yield prediction by using different machine learning and deep learning algorithms.
2. To help farmers to choose a suitable crop to cultivate in order to maximize the yield.
3. To give farmers a generic idea on how external factors (like temperature, rainfall etc.) can influence their yield.

ACKNOWLEDGEMENT

This project would not have been possible without the help and cooperation of many. I would like to thank the people who helped me directly and indirectly in the completion of this project work.

First and foremost, I would like to express my gratitude to our honorable Director, **Prof. O.G. Kakde**, for providing his kind support in various aspects. I would like to express my gratitude to my project guide **Prof.Ritu Tiwari, Department of CSE**, for providing excellent guidance, encouragement, inspiration, constant and timely support throughout this **B.Tech Project**. I would like to express my gratitude to **Tanmoy Hazra, Head of Department of CSE** for providing his kind support in various aspects. I would also like to thank all the faculty members in the **Department of CSE** and my classmates for their steadfast and strong support and engagement with this project.

Abstract

For most developing countries, agriculture is the primary source of income. Modern agriculture is an ever-increasing access to agricultural advances and farming techniques. It becomes challenging for farmers to meet the evolving demands of our planet and the expectations of traders, customers, etc. Food production and forecasts are depleting due to unnatural climate change, which will adversely affect farmers' economy by getting bad yield and also help farmers to remain less aware predicting future crops.

This research work helps the beginning farmers in such a way to guide them to sow sensible crops by deploying machine learning, one of the advanced technologies in crop prediction. Some of the challenges farmers are faced with (i) Coping with climate change due to soil erosion and industrial emissions (ii) A lack of nutrients in the soil, caused by a lack of key minerals such as potassium, nitrogen and phosphorus can result in reduced crop growth. (iii) Farmers make a mistake by growing the same crops year after year without experimenting with different varieties. They add fertilizers randomly without understanding inferior quality or quantity.

Machine learning (ML) is a game-changer for the agricultural sector. Machine learning is part of artificial intelligence, emerged together with big data technologies and high-performance computing new opportunities for data-intensive science in the multidisciplinary field of agricultural technology. IN For example, machine learning in agriculture is not some mysterious trick or magic, it is a set of well-defined models that collect specific data and use specific algorithms to achieve expected results.

The aim of the work is to discover the best model for crop prediction that can help farmers to decide on the type of crop to be grown based on climatic conditions and nutrients present in the soil. This project compares popular algorithms like Polynomial Support Vector Machine, Random Forest Algorithm, XGBooster and Neural networks. The results show that Random Forest provides the highest accuracy of the three.

Keywords: Random Forest ,Support Vector Machine,Neural Networks,XGBooster

TABLE OF CONTENTS

Abstract	i
(i) List of Figures/Symbols/Nomenclature	iv
(ii) List of Tables	v
1 Introduction	11
1.1 Overview of work	11
1.2 Motivation of work	11
1.3 Literature Review	12
1.4 Research Gap.	14
2 Problem Statement	
2.1 Research Objectives	15
2.2 Methodology of work	15
3 Analysis And Design	29
4 Results and Discussion	39
5 Conclusion and Future Scope	44
6 References	45

List of Figures / Symbols/ Nomenclature

Figure No:	Figure Name	Page No:
2.1	Random Forest	21
2.2	Support Vector Machine	22
2.3	Neural Network	24
2.4.1	Bagging	25
2.4.2	Boosting	26
2.4.3	XGBoost 1	27
2.4.2	XGBoost 2	27
3	Correlation Matrix	29
3.1	System Architecture	30
3.2	Random Forest Flow Chart	31
3.3	Flow chart of crop yield prediction	31
3.4.1	UML Diagram - I	32
3.4.2	UML Diagram - II	33
3.4.3	Sequence Diagram	34
4.1.1	Mean Absolute Error	40
4.1.2	Model Loss	40
4.2	User Interface Code	42
4.3	User Interface	43
4.4	Result/ Prediction	43

List of Tables

Table No:	Table Name	Page No:
4.1	Results of Random Forest	39
4.2	Results of Polynomial SVM	39
4.3	Results of XGBooster	40

Chapter 1

Introduction

1.1 Overview of Work

1.1.1 Project Overview

Agriculture is one of the major sectors that are affected by various sources such as climate change, soil properties, seasonal changes, etc. Crop yield prediction [1] is based on different kinds of data collected and extracted using data mining techniques [2], such as machine learning techniques[3] various resources that are useful for crop growth. It is the art of predicting the harvest and the amount of the yield in advance, that is, even before the harvest actually takes place. Predicting crop yield can be extremely useful for farmers. If they have an idea of the amount of yield they can expect, they can withdraw the crop before harvest, often securing a more competitive price than if they waited until after harvest. The involvement of experts in crop yield prediction leads to problems such as lack of knowledge about natural events, negation of personal perception and fatigue, etc. These problems can be overcome by using machine learning models and decision tools for crop yield prediction. information regarding parameters that consider yield such as temperature, rainfall, moisture, soil content, etc. and predict crop yield. This can help farmers decide which crop to grow based on the profit they get from it. Similarly, industry can benefit from revenue forecasting by better planning the logistics of their business.

In this project we use machine learning algorithms like ANN, Polynomial SVM, XGBooster and Random Forest Algorithm to predict the yield and calculate the accuracy of each model.

1.2 Motivation of the work

The project aims to use machine learning algorithms and techniques to predict the crop yield based on various factors. Farmers may only benefit from the ultimate yield if they harvest the crop that is appropriate for the field's circumstances. This may be achieved by carefully examining the soil and climate data.

We require a lot of soil content and nature estimations and predictions. Machine learning and deep learning algorithms may be used to easily make these predictions based on vast datasets. compare various supervised learning algorithms like ANN, Polynomial SVM, XGBooster and Random Forest Algorithm on the dataset containing 16 features.

The performance of the algorithms ANN, Polynomial SVM,XGBooster and Random Forest Algorithms are measured using accuracy measures and scores.

1.3 Literature Review

Many systems and techniques were suggested and published in the past for the prediction of crop yield using machine learning algorithms.

- 1) **“A Survey on crop Prediction Using Machine Learning”** by Sriram Rakshith.K, Dr. Deepak.G, Rajesh M, Sudarshan K S, Vasanth S & Harish Kumar

Research has mainly focused on crop prediction using techniques such as artificial neural network (ANN), information fuzzy network and other data mining techniques.

This paper mainly focuses on the techniques and measures taken to improve agriculture by inculcating technical knowledge and development to make the agricultural sector more reliable and easier for farmers by predicting the suitable crop using machine learning techniques by sensing parameters such as - soil, weather and market trends. The considered parameters are PH, the content of nitrogen, phosphate and potassium in the soil, temperature, precipitation and humidity. They consider artificial neural networks, information fuzzy networks and other data mining techniques.

- 2) **“Machine learning approach for forecasting crop yield based on climatic parameters”** by S.Veenadhari, Dr. Bharat Misra & Dr. CD Singh.

The research mainly focuses on the Machine learning approach for forecasting crop yield based on climatic parameters.

In this paper, the study was aimed at developing a website to find out the effect of climatic parameters on crop production in selected areas of Madhya Pradesh. The selection of areas was made on the basis of the area where a particular crop is grown. Based on this criterion, the first five best districts in which the selected crop area is maximum were selected. The crops selected in the study were based on the prevailing crops in the selected district. Crops selected included: soybeans, corn, paddy and wheat. The yields of these crops were tabulated for continuous 20 years by collecting information from secondary sources. Similarly, climatic parameters such as precipitation, maximum and minimum temperature, potential evapotranspiration, cloudiness, frequency of rainy days were also collected from secondary sources for the corresponding years. The methodology adopted for the analysis involves that values above the threshold were considered as one child and the remaining as another child. It also handles missing attribute values. In the pseudocode, there is a general algorithm for creating decision trees: For each attribute a : find the normalized information gain from partitioning on a , let a_{best} be the attribute with the highest normalized information gain, Create a decision node to partition on a_{best} , Recursion on the sublists obtained by partitioning on a_{best} and add those nodes as child nodes. In this relevance analysis approach, they calculated the information gain for each of the attributes defining the samples in S . The attribute with the highest information gain was considered the most discriminating attribute of the set. By calculating the information gain for each attribute, they obtained the rank of the attributes. This rating can be used for relevance analysis to select attributes to be used in the concept description. The web software was developed in C# on the .net platform. The backend used is sql server 2008. The findings were that out of 20 years of data, the predictions were correct in 18 years and incorrect in two years, showing that the developed model's prediction accuracy is 90% for soybean in Dewas district. The prediction accuracy of the developed model ranged from 76 to 90% for selected crops and selected districts. Based on these observations, the overall prediction accuracy of the developed model is 82.00%. This article focuses on analyzing the relevance approach to ensure accurate prediction. This is achieved by calculating the information

gain of each attribute and comparing them. However, it does not include analysis of other supervised machine learning algorithms such as random forest and linear regression.

3) **“Smart Crop Prediction using IoT and Machine Learning”** by Rushika Ghadge, Juilee Kulkarni, Pooja More, Sachee Nene, Priya R L .

The research focuses mainly on the study of soil and fertilizers. It also returns information about the given fertilizer and also recommends one after calculating requirements based on soil properties and location. This article states that most existing systems are hardware based, which makes them expensive and difficult to maintain and lack accurate results. Some systems suggest a crop order based on yield and market price. In this paper, the proposed system attempts to overcome these drawbacks and predicts crops using structured data analysis. As it is a completely software solution, it does not allow much consideration of maintenance factors. Also, the level of accuracy would be high compared to hardware solutions as components like soil composition, soil type, pH value, weather conditions all come into picture during the prediction process. This can be achieved using unsupervised and supervised learning algorithms such as Kohonen Dept of CSE, CMRIT 2019-2020 Weather-Based Crop Yield Prediction Using Self-Organizing Machine Learning Map (Kohonen's SOM) and BPN (Back Propagation Network). The dataset will then be trained by learning networks. It compares the accuracy obtained by different network learning techniques and the most accurate result will be delivered to the end user. This paper proposes a system that will monitor soil quality and predict crop yield accordingly and provide fertilizer recommendations when needed depending on soil quality. The system takes input pH value and position from the user and the processing of the results is done by two controllers. The results of controller 1 and controller 2 are compared to a predefined "nutrient" data store. These compared results are supplied to the control unit 3 where a combination of the above results and a predefined set of data present in the crop data store are compared. Finally, the results are displayed in the form of bar graphs along with the percentage accuracy.

4) **“Predicting yield of the crop using Machine Learning Algorithms”** by P.Priya, U.Muthaiah & M.Balamurugan.

Research focuses mainly on random forest classifiers and tools used for statistics, data analysis. This article uses R programming with machine learning techniques. R is a leading tool for statistics, data analysis and machine learning. It's more than a stat pack; it's a programming language, so you can create your own objects, functions, and packages. It is platform independent so it can be used on any operating system and it is free. R programs explicitly document the steps of our analysis and make it easy to reproduce and/or update the analysis, meaning they can quickly test many ideas and/or fix problems. All datasets used in the research were sourced from publicly available records of the Government of India. It was obtained for the years 1997 to 2013 for different seasons like Kharif and Rabi rice production. From the extensive initial data set, only a limited number of important factors that have the greatest influence on agricultural yield were selected for this research. The dataset contains the following parameters: precipitation, season and temperature, and crop production. This article also compares two machine learning algorithms: decision trees and Random Forest.

1.4 Research Gap

Existing System

Other than blogging sites that provide information about agriculture and farming accessories, there is no specific website to predict crop yield depending on history in that particular geographic area.

Proposed System

We have collected temperature, rainfall, crop yield and other datasets for the state of Maharashtra from various sources like India Meteorological Department (IMD), KSNDMC and Agriculture website department. We implement machine learning algorithms such as Random Forest, Neural Networks, XGBooster, multi-linear regression algorithm, clustering Algorithm, SVM algorithm to predict crop yield based on factors such as temperature, rainfall and pressure. Using Flask, HTML, CSS, database frontend and the backend they are designed for. The user interface obtains the crop yield prediction result and displays it to the user.

Advantages

Because people use this application, the dataset may be stored in this particular area.

The stored data can help other farmers in the future to control the harvest of the crop well in advance of yield.

Adding more data will make the forecast even more accurate.

Chapter 2

Problem Statement

In this study, we use the data and trends related to agriculture of the previous years for the prediction of the best crop to cultivate. The objective of model is to predict We experiment with various machine learning and deep learning models namely Random Forest Regression model, Support Vector Regression and Neural Network . The objective is to create a model/ classifier that can predict Crop yield by taking different parameters like soil type, temperature, PH values etc.. and also predict the type of disease the plant has on specific crops. Modern machine learning classifiers use conventional feature representation techniques, however the suggested deep learning classifier uses the proper number of layers, an ideal set of parameters, and gives output in optimized time with good accuracy.

2.1. Research Objectives

The main objectives of this research is:

- To increase the accuracy of crop yield prediction by using different machine learning and deep learning algorithms..
- To get a generic idea on how external factors (temperature,rainfall,humidity,soil content etc.)can influence the yield of the crop.
- To understand how various machine learning algorithms are implemented on a complex data set which contains weather conditions and soil conditions.

2.2. Methodology of the Work

Predicting the crop yield from the given parameters is done by using various machine learning algorithms involving some preliminary steps. They are:

Step1: Import datasets-

Collect datasets and merge these datasets in a structured form.

Step 2:Data Preprocessing -

It is done to remove inaccurate, incomplete and unreasonable data which results in an increase in quality of the data and hence the overall productivity.

Step 3:Feature Extraction-

It is the process of reducing the raw data into manageable groups (features) for processing it. Beginning with an initial set of raw data it builds up derived values (features) which results in an informative and non-redundant data.

Step 4:Train-Test Splitting-

Divide the analyzed crop data into training and testing sets and train the model using the training data to predict the crop yield for given inputs.

Step 5:Algorithms-

Compare results of various algorithms by passing the analyzed dataset through them and calculating the error rate and accuracy for each. Choose the machine learning algorithm with the highest accuracy and lowest error rate.

Step 6:Testing- Test the implemented system to check for accuracy and failures.

The above mentioned steps of machine learning are performed using the programming language called Python.

Python:

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

Python consistently ranks as one of the most popular programming languages.

Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like:

- 1) Pandas
- 2) Numpy
- 3) Matplotlib
- 4) Seaborn
- 5) Sklearn
- 6) TensorFlow
- 7) Keras
- 8) Pytorch
- 9) Scikit-learn.

1)Pandas:

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the data set must be prepared before training.

In this case, Pandas comes handy as it was developed specifically for data extraction and preparation.

It provides high-level data structures and a wide variety of tools for data analysis.

It provides many inbuilt methods for grouping, combining and filtering data.

It is the library we use in our methodology to import datasets.

2) Numpy:

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions.

It is very useful for fundamental scientific computations in Machine Learning.

It is particularly useful for linear algebra, Fourier transform, and random number capabilities.

High-end libraries like TensorFlow use NumPy internally for manipulation of Tensors.

3) **Matplotlib:**

Matplotlib is a very popular Python library for data visualization.

Like Pandas, it is not directly related to Machine Learning.

It particularly comes in handy when a programmer wants to visualize the patterns in the data.

It is a 2D plotting library used for creating 2D graphs and plots.

A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc.

It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar charts, etc.

4) **Seaborn:**

Seaborn is a library mostly used for statistical plotting in Python.

It is built on top of Matplotlib.

It provides beautiful default styles and color palettes to make statistical plots more attractive.

5) **Sklearn:**

The Sklearn Library is mainly used for modeling data and it provides efficient tools that are easy to use for any kind of predictive data analysis.

The main use cases of this library can be categorized into 6 categories which are the following:

- Preprocessing
- Regression
- Classification
- Clustering
- Model Selection
- Dimensionality Reduction

6) **TensorFlow:**

TensorFlow is a very popular open-source library for high performance numerical computation developed by the Google Brain team in Google.

As the name suggests, Tensorflow is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications.

TensorFlow is widely used in the field of deep learning research and application.

7) **Keras:**

It provides many inbuilt methods for grouping, combining and filtering data.

Keras is a very popular Machine Learning library for Python.

It is a high-level neural networks API capable of running on top of

TensorFlow, CNTK, or Theano.

It can run seamlessly on both CPU and GPU.

Keras makes it really helpful for ML beginners to build and design a Neural Network.

One of the best thing about Keras is that it allows for easy and fast prototyping.

8) PyTorch:

PyTorch is a popular open-source Machine Learning library for Python based on Torch, which is an open-source Machine Learning library which is implemented in C with a wrapper in Lua.

It has an extensive choice of tools and libraries that supports on Computer Vision, Natural Language Processing(NLP) and many more ML programs.

It allows developers to perform computations on Tensors with GPU acceleration and also helps in creating computational graphs.

9) Scikit-learn:

Scikit-learn is one of the most popular ML libraries for classical ML algorithms.

It is built on top of two basic Python libraries, viz., NumPy and SciPy.

Scikit-learn supports most of the supervised and unsupervised learning algorithms.

Scikit-learn can also be used for data-mining and data-analysis, which makes it a great tool for those starting out with ML.

After importing the libraries,the next step i.e. data preprocessing is done on the data.

The various techniques which are employed in data pre-processing are:

1)Handling Null Values —

In any real-world dataset there are always few null values. There are various ways for us to handle this problem. The easiest way to solve this problem is by dropping the rows or columns that contain null values. However, it is not the best option to remove the rows and columns from our dataset as it can lead to loss of valuable information.

If you have 300K data points then removing 2–3 rows won't affect your dataset much but if you only have 100 data points and out of which 20 have NaN values for a particular field then you can't simply drop those rows.

In real-world datasets it can happen quite often that you have a large number of NaN values for a particular field.

Example: Suppose we are collecting the data from a survey, then it is possible that there could be an optional field which let's say 20% of people left blank. So when we get the dataset then we need to understand that the remaining 80% data is still useful, so rather than dropping these values we need to somehow substitute the missing 20% values.

Sometimes, filled by most frequent data

2) Standardization

It is another integral preprocessing step. In Standardization, we transform our values such that the mean of the values is 0 and the standard deviation is 1.

Consider a data frame having 2 numerical values: Age and Weight. They are not on the same scale as Age is in years and Weight is in Kg and since Weight is more likely to be greater than Age. Therefore, our model will give more weightage to Weight, which is not the ideal scenario as Age is also an integral factor here. In order to avoid this issue, we perform Standardization.

We just calculate the mean and standard deviation of the values and then for each data point we just subtract the mean and divide it by standard deviation.

$$z = \frac{x_i - \mu}{\sigma}$$

3) Normalization

In most cases, normalization refers to the re-scaling of data features between 0 and 1, which is a special case of Min-Max scaling.

In the given equation, subtract the min value for each feature from each feature instance and divide by the spread between max and min.

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{min}}{x_{max} - x_{min}}$$

4) Handling Categorical Variables

Categorical variables are basically the variables that are discrete and not continuous. Ex — color of an item is a discrete variable whereas its price is a continuous variable.

Categorical variables are further divided into 2 types —

Ordinal categorical variables — These variables can be ordered. Ex — Size of a T-shirt. We can say that M<L<XL.

Nominal categorical variables — These variables can't be ordered. Ex — Color of a T-shirt. We can't say that Blue<Green as it doesn't make any sense to compare the colors as they don't have any relationship.

Integer Encoding- Assign numerical values for ordinal data (S-1, M-2, L-3 etc.)

One Hot Encoding-One Hot Encoding is used to convert numerical categorical variables into binary vectors. Before implementing this algorithm. Make sure the categorical values must be label encoded as one hot encoding takes only numerical categorical values.

After data preprocessing is done, various machine learning algorithms are used to predict the crop yield. They are:

1) **Random Forest Algorithm:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both classification and regression problems in ML. It is based on the concept of ensemble learning, which is the process of combining multiple classifiers to solve a complex problem and improve model performance. A larger number of trees in the forest leads to higher accuracy and avoids the overfitting problem.

Some assumptions must be made before implementing the Random Forest algorithm. They are:

There should be some actual values in the dataset feature variable so that the classifier can predict exact results rather than an estimated result.

The predictions from each tree must have very low correlations.

Why do we use the Random Forest Algorithm?

Some of the reasons for using the Random Forest algorithm are:

Compared to other algorithms, training takes less time.

It predicts the output with high accuracy even for a large data set that runs efficiently.

It can also maintain accuracy when a large portion of data is missing.

The steps involved in random forest regression are:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

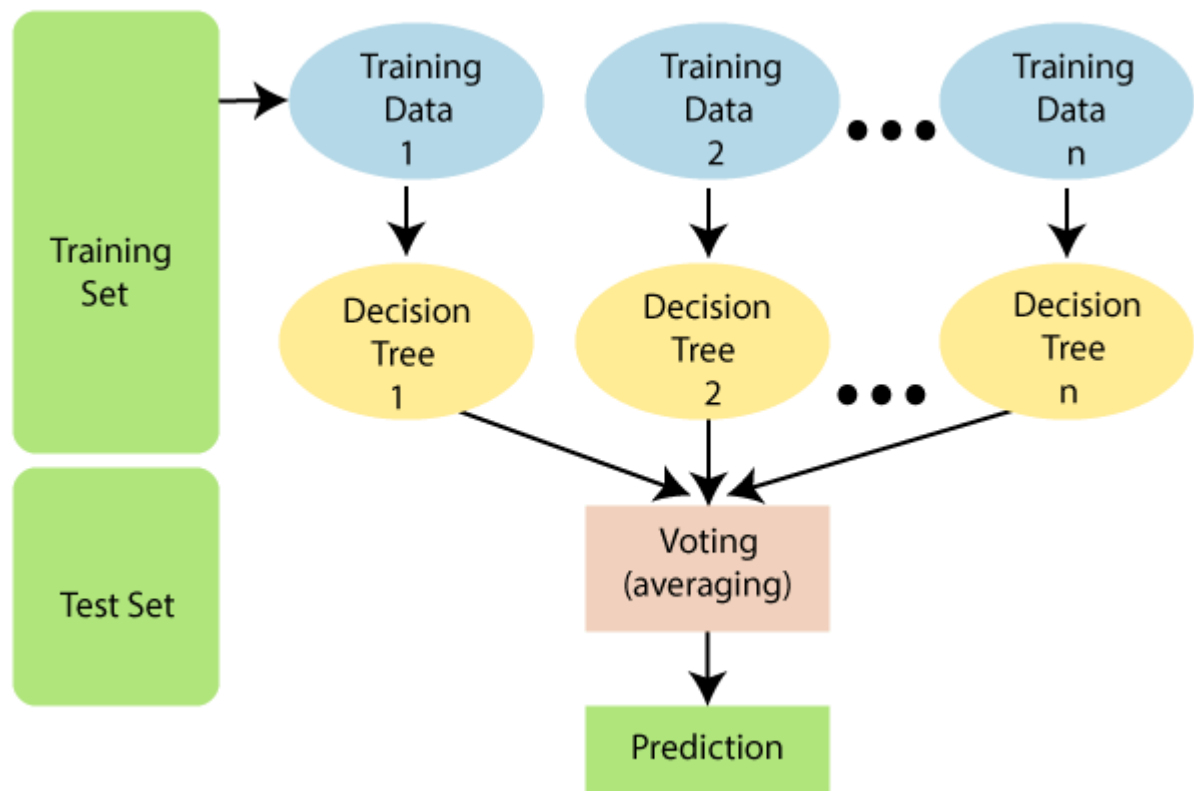


Fig 2.1 Random Forest

javatpoint.com

2)Support Vector Machine(SVM) Algorithm:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine.

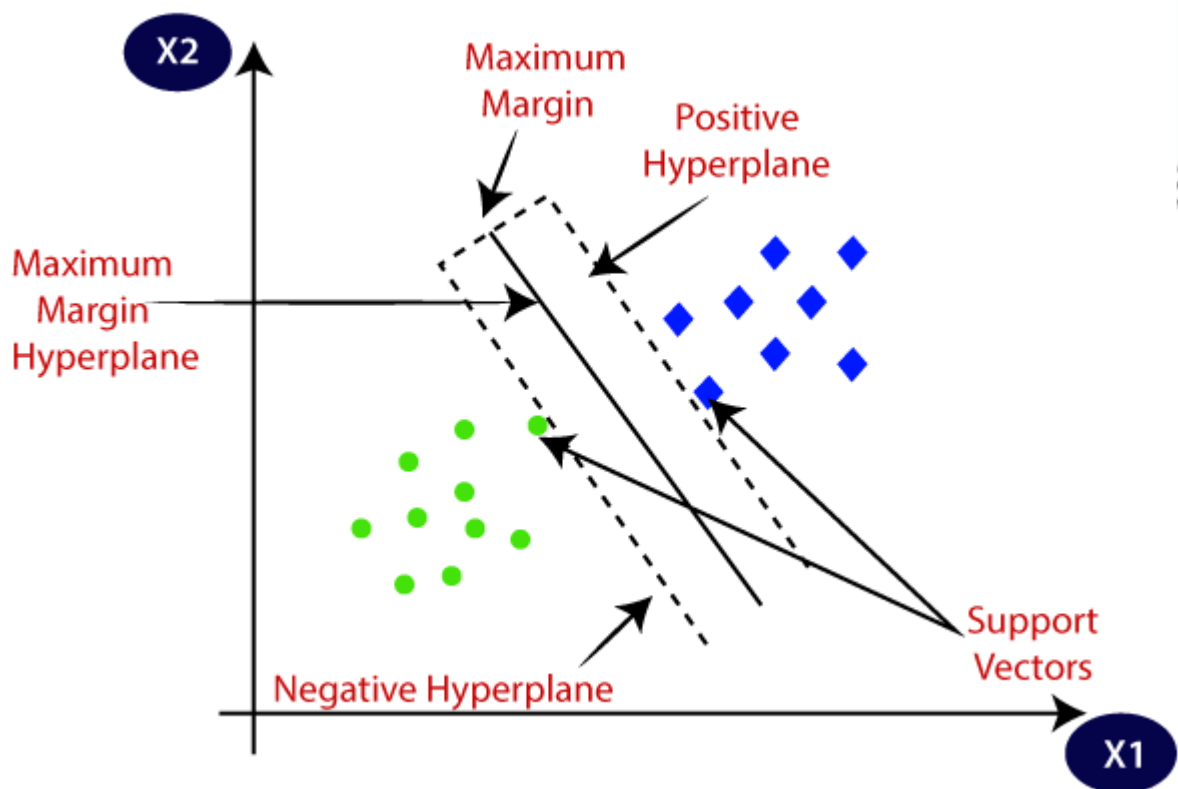


Fig 2.2 Support Vector Machine

The above mentioned method is applied when our data is linearly separable. If the data is not linearly separable, then we use various kernel functions to modify our data into higher feature space so that they can be linearly separable. Some of the kernel functions used are:

- **Linear kernel:** $K(x_i, x_j) = x_i^T x_j$
- **Polynomial kernel:** $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
- **RBF kernel :** $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \gamma > 0$
- **Sigmoid kernel:** $K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$

Here, γ , r and d are kernel parameters.

1. **The Kernel:** The kernel is selected based on the type of data and also the type of transformation. By default, the kernel is the Radial Basis Function Kernel (RBF).
2. **Gamma :** This parameter decides how far the influence of a single training example reaches during transformation, which in turn affects how tightly the decision boundaries end up surrounding points in the input space. If there is a small value of gamma, points farther apart are considered similar. So more points are grouped together and have smoother decision boundaries (maybe less accurate). Larger values of gamma cause points to be closer together (may cause overfitting).

In this project, we use polynomial SVM, i.e. our kernel function is a polynomial kernel.

Polynomial features are derived features from given features in the data set. For example, we have a data set with a single feature x and we want to find polynomial features with degree 3, then polynomial features will be x, x^2, x^3 . If we have other features each of them would be converted similarly. We require more polynomial features if our dataset is complex, leading to slow model training as the feature increases. Fortunately, when using SVMs you can apply an almost miraculous mathematical technique called the kernel trick (explained briefly). It makes it possible to get the same result as if you added many polynomial features, even with very high-degree polynomials, without actually having to add them. So there is no combinatorial

explosion of the number of features since you don't actually add any features. This trick is implemented by the SVC class.

3. Neural Networks:

An artificial neural network is one of the new data mining techniques that are based on the biological neural processes of the human brain. According to this technique, once a neural network is trained, it can predict crop yield in a similar way even if the past data contains some errors. Even if the data is complex, multidimensional, non-linear, this network provides accurate results and also without any underlying principles, the relationship between them is extracted. Artificial Neural Networks (ANN) networks in which each node represents a somatic cell and each link represents a method 2 somatic cell. Each somatic cell performs straightforward tasks, while a network representing the work of all its neurons is ready to perform other complex tasks. A neural network is a connected set of input/output units where each association contains a weight associated with it. The ANN classification process can generally be outlined as follows:

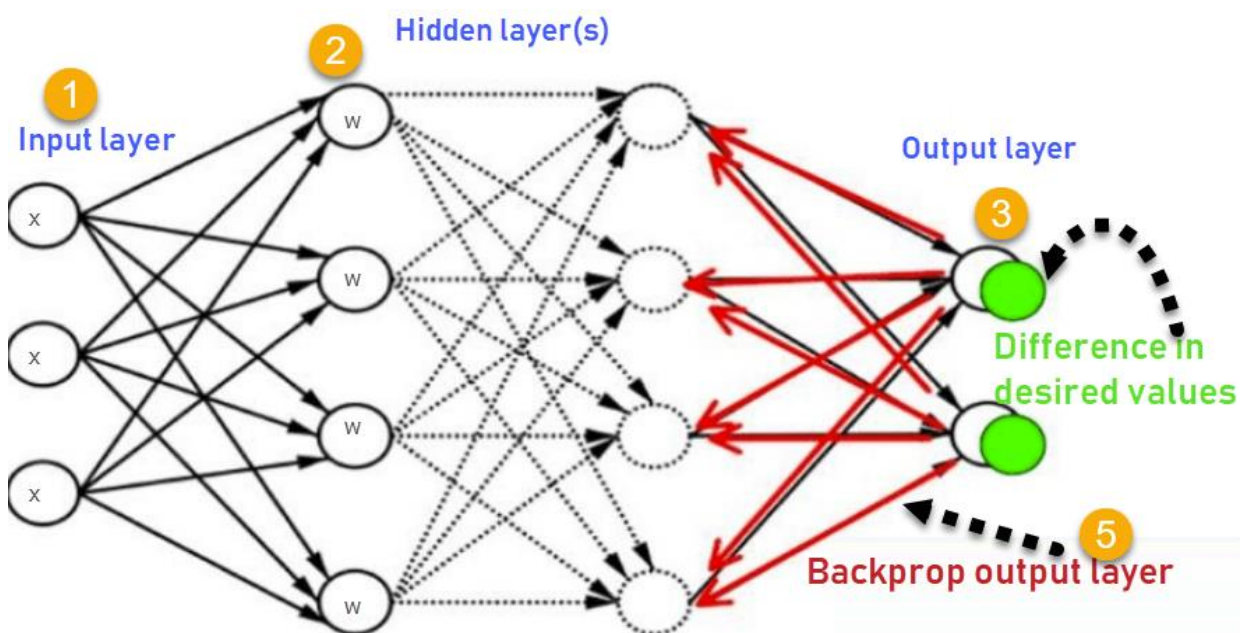


Fig 2.3 Neural Network

www.guru99.com

1. Inputs X, arrive through the preconnected path

2. Input is modeled using real weights W . The weights are usually randomly selected.
3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
4. Calculate the error in the outputs

$$\text{Error}(B) = \text{Actual Output} - \text{Desired Output}$$

5. Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

4)Extreme Gradient Booster(XGBooster):

XGBoost is a powerful approach for building supervised regression models. To understand XGBoost we need to first understand bagging, boosting and gradient boosting.

Bagging:

A bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregates their individual predictions (either by voting or averaging) to produce a final prediction. Each base classifier is trained in parallel with a training set that is generated by randomly drawing, with replacement, N examples (or data) from the original training dataset, where N is the size of the original training set. The training set for each of the basic classifiers is independent of each other. Many of the original data may be repeated in the resulting training set, while others may be omitted.

Random Forest has several decision trees as basic learning models. We randomly sample rows and sample elements from the data set forming the sample data sets for each model. This part is called Bootstrap.

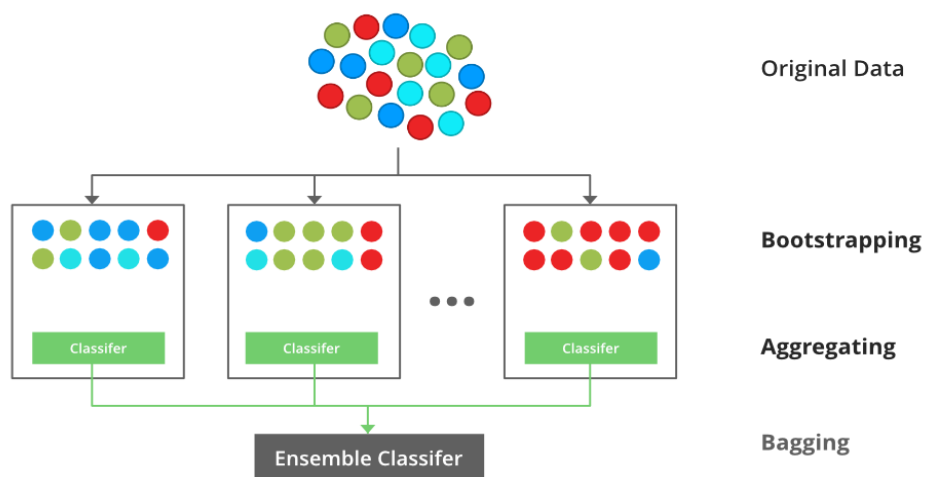


Fig 2.4.1 Bagging

[geeksforgeeks.org](https://www.geeksforgeeks.org)

Boosting:

Boosting is an ensemble modeling technique that attempts to create a strong classifier from a number of weak classifiers. This is done by building the model using weak models in series. First, a model is created from the training data. A second model is then built that tries to correct the errors present in the first model. This procedure continues and models are added until the full set of training data is correctly predicted or until the maximum number of models is added.



Fig 2.4.2 Boosting Diagram

geeksforgeeks.org

Gradient Boosting:

Gradient Boosting is a popular boosting algorithm. In gradient boosting, each predictor corrects the error of its predecessor. Unlike Adaboost, the weights of the training instances are not tuned, instead each predictor is trained using the residual errors of the predecessor as labels.

XGBoost

XGBoost is an implementation of transition-boosted decision trees. In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all independent variables, which are then fed into a decision tree that predicts outcomes. The weight of the variables that the tree predicted incorrectly is increased and these variables are then fed into the second decision tree. These individual classifiers/predictors are then grouped together to provide a stronger and more accurate model. It can work on regression, classification, ranking, and user-defined prediction problems.

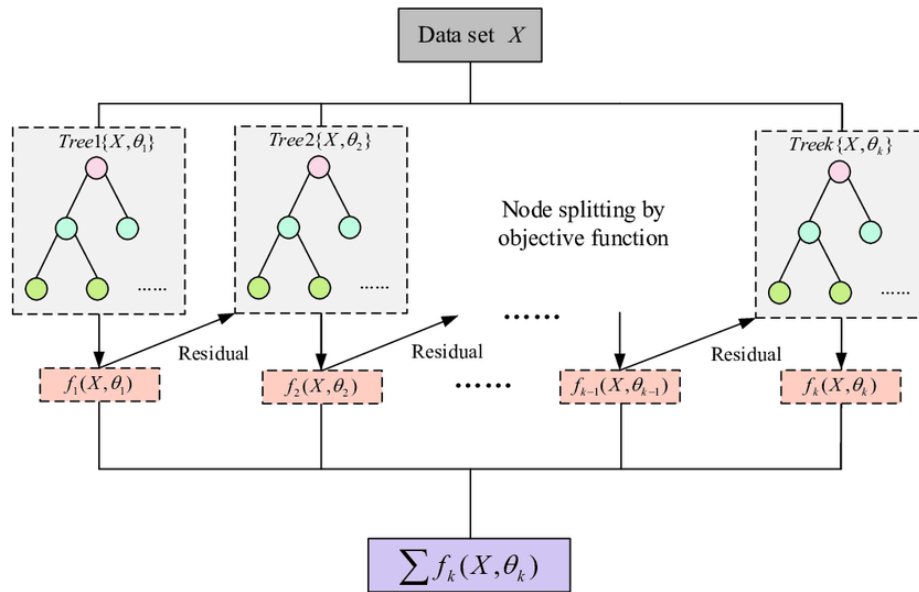


Fig 2.4.3 XGBoost 1

www.researchgate.net

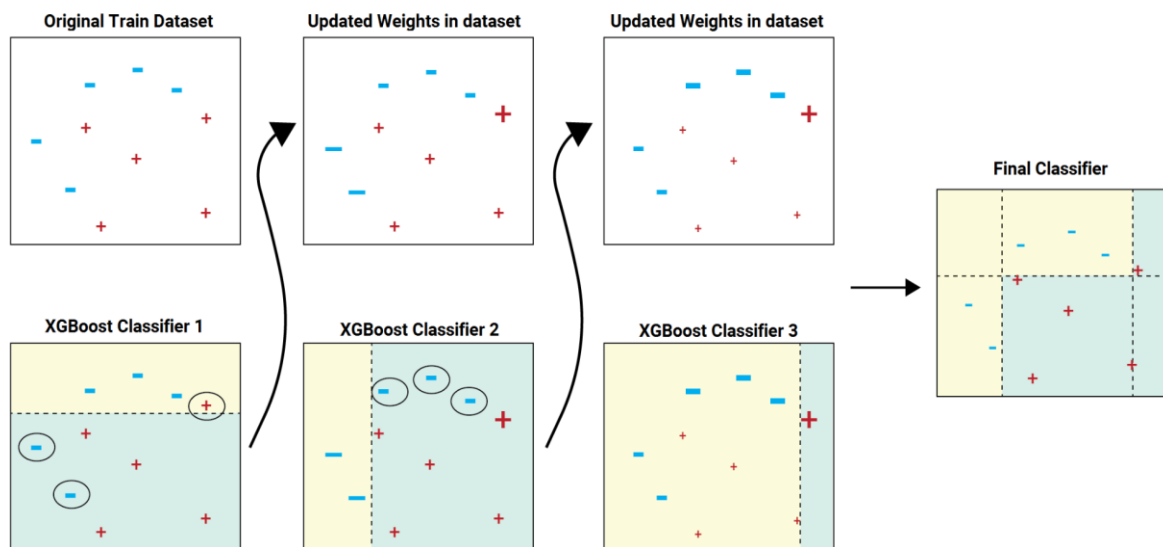


Fig 2.4.4 XGBoost 2

blog.quantinsti.com

Hyperparameters involved in the XG-Boost algorithm

To learn any machine learning algorithm, we must know the different factors that can affect the performance of our model when we try to fit that algorithm into our dataset. In XG-Boost, these factors are:

1. `n_estimators`: Number of trees we want to grow sequentially.
2. `max_depth`: Maximum depth allowed for a tree.
3. `learning_rate`: Slow learners are said to be better learners, and to control the update step in residuals, we need to tune the learning rate.
4. `subsample`: To control the factor of row sampling.

5. `colsample_bytree`, `colsample_bylevel`: To control the factor of column sampling.

In addition to these, XG-Boost provides the functionality where we can define the grid of values (multiple values of these parameters in arrays), and the in-built cross-validation technique will find the best set of parameters.

Chapter 3

Analysis and Design

System analysis is the process of gathering and interpreting facts, identifying problems, and decomposing a system into its components. System analysis is performed to study a system or its parts in order to identify its objectives. Analysis specifies what the system should do.

System design is the process of planning a new business system or replacing an existing system by defining its components or modules to meet specific requirements. System design focuses on how to achieve the goal of the system.

Data Analysis:

Using various built-in functions, we can get an overview of the number of values in each column, which can give us information about null values or duplicate data. We can also find the mean, standard deviation, minimum value and maximum value. To get a better overview of the data we are using, we can plot graphs like the correlation matrix, which is one of the most important concepts that gives us a lot of information about how the variables (columns) are related to each other and what impact each one has on the other.

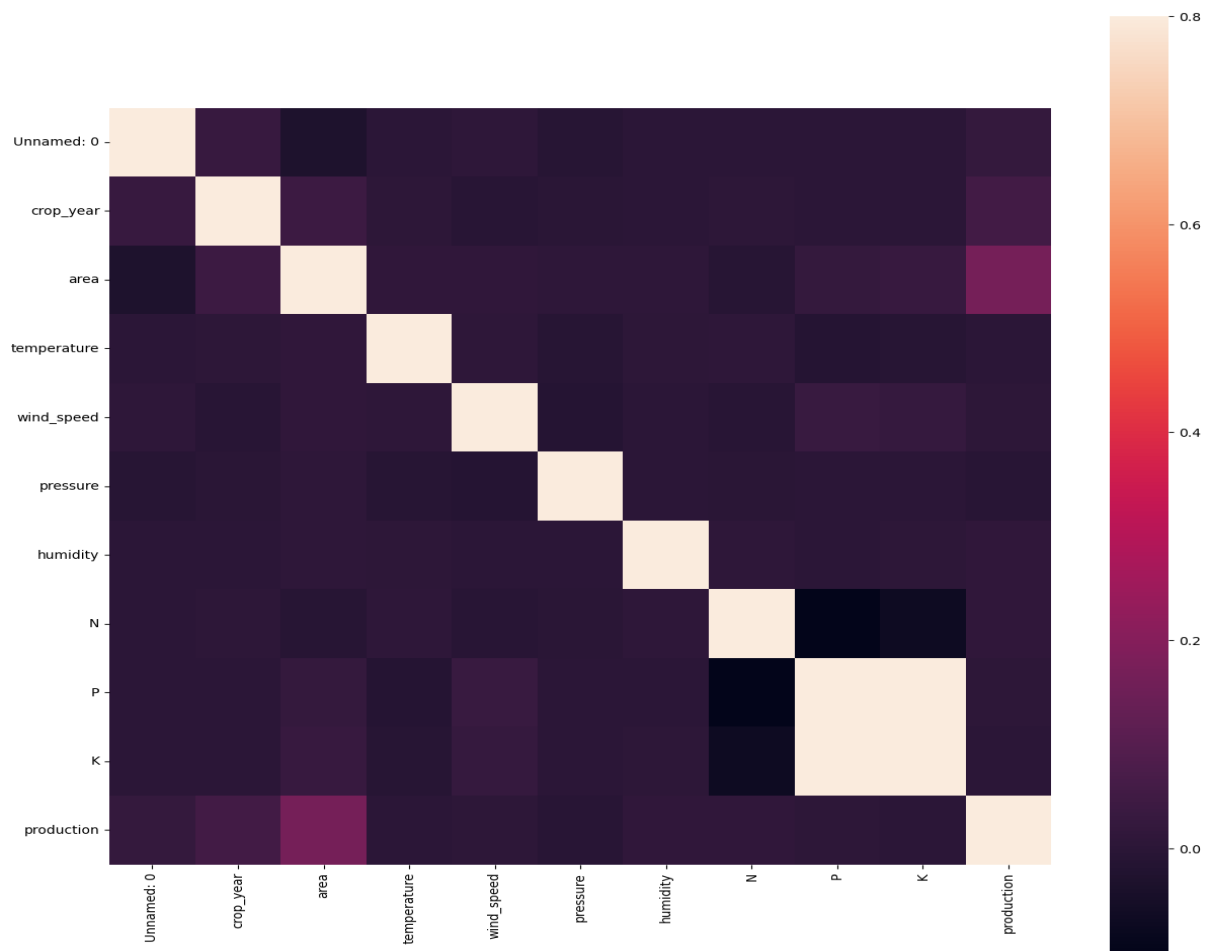


Fig 3 Correlation Matrix

System Architecture:

Architecture diagrams can help system designers and developers visualize the high-level, overall structure of their system or application for the purpose of ensuring the system meets their users' needs.

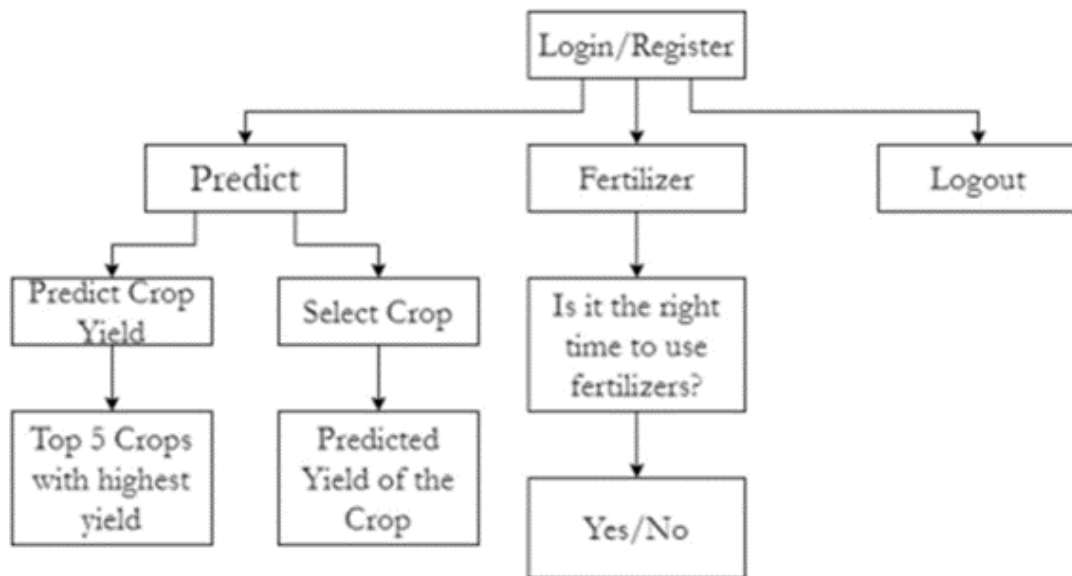


Fig 3.1 System Architecture

The first step is to log in or register in the application. In the next step, three options are available. i.e. Prediction, Fertilizer and Logout. The user can choose one of the three options and proceed further. In the Predict section, the system offers two options, depending on whether the user knows what to plant or whether the crop is still being decided. In both cases the inputs are taken from the user and the predicted value is passed to the user. When the fertilizer module is selected, the user will get a popup message saying whether he can use the fertilizer or not and it may or may not rain for the next 15 days. The last one is Logout, which logs the user out and back to the login/registration page.

Flowchart:

A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes

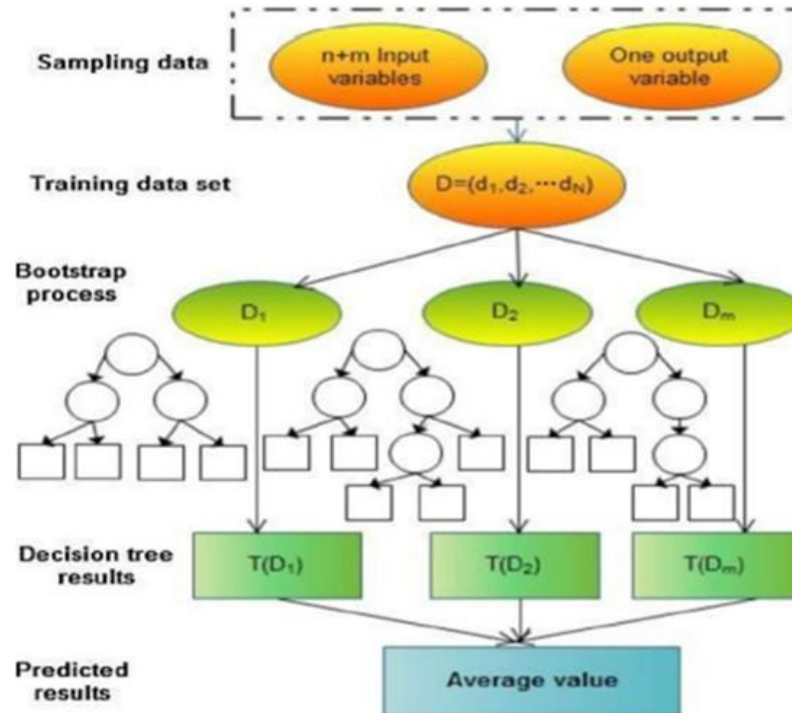


Fig 3.2 Flowchart of Random Forest algorithm
www.researchgate.net

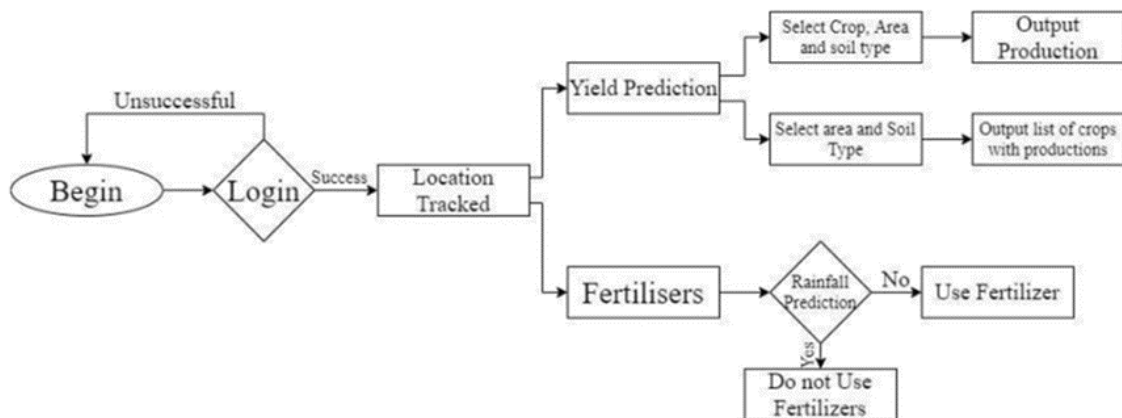


Fig 3.3 Flowchart of Crop Yield Prediction

Use Case Diagram:

A use case is a methodology used in systems analysis to identify, clarify, and organize system requirements. A use case is made up of a set of possible sequences of interactions between systems

and users in a specific environment and related to a specific goal.

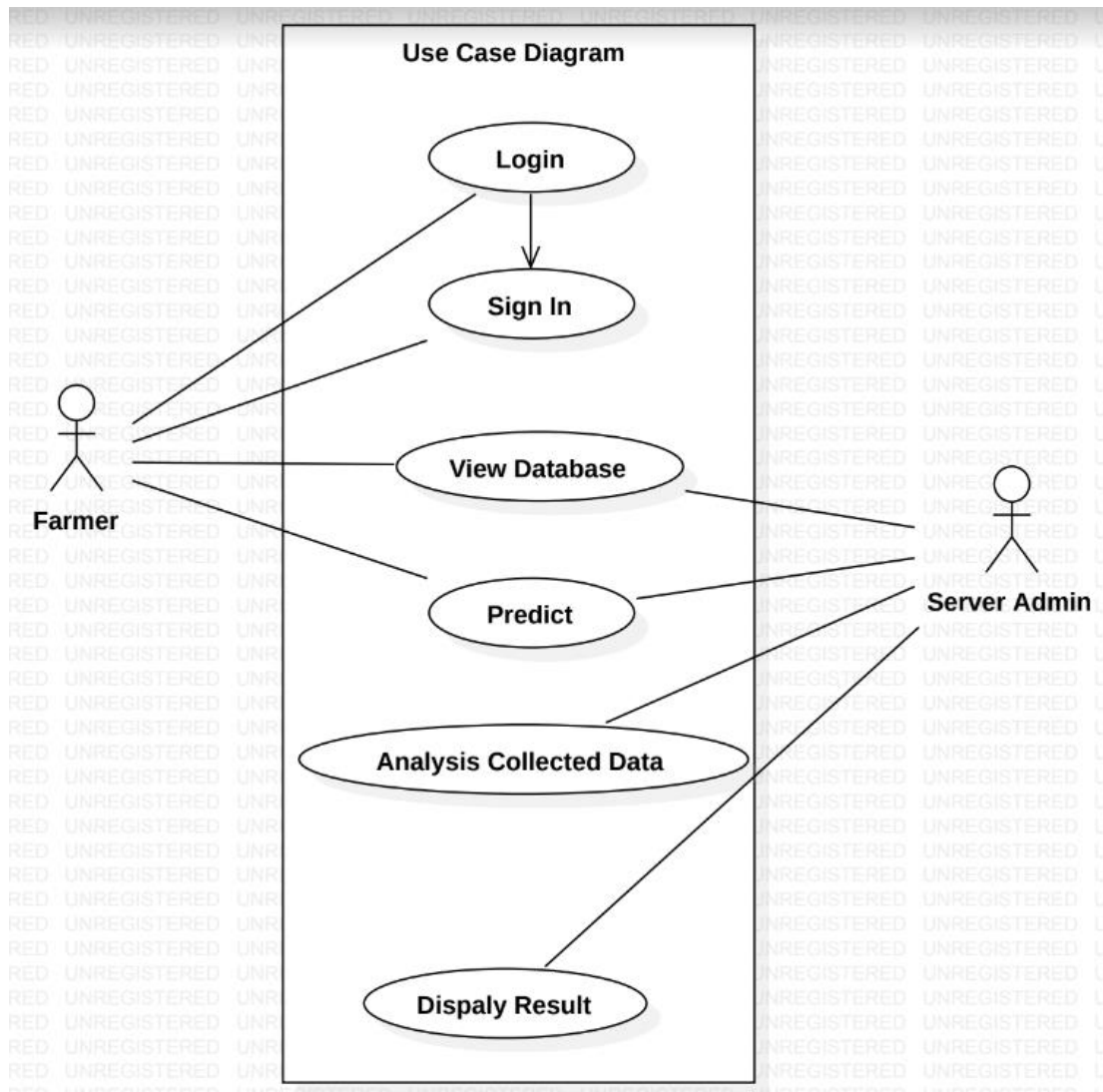


Fig 3.4.1 UML Diagram - I

The above figure represents the actors (users) and their functional requirements provided by the system. The system includes two actors - the end user (farmer) and the administrator. Functions provided by the system are shown in ovals. Arrows represent dependencies and feature visibility. The user is only allowed to access a few functions such as Login, Login, View Database, Prediction Results, while the administrator has access to two other functions which are analysis of collected data and prediction results.

Activity Diagram:

An activity diagram is another important diagram in UML for describing the dynamic aspects of a system. It is essentially a flowchart that shows the flow from one activity to another. The activity can be described as the activity of the system.

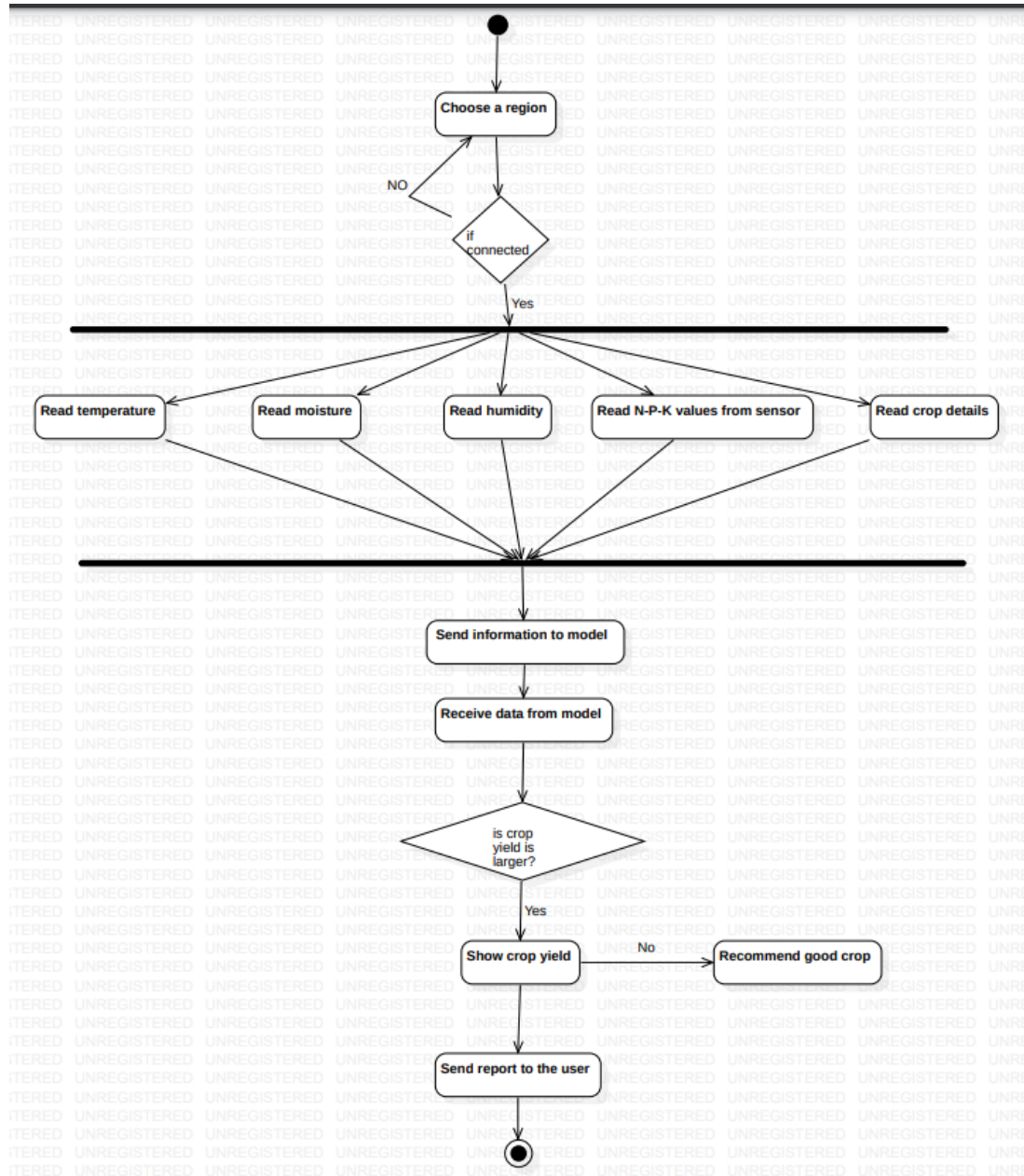


Fig 3.4.2 UML Diagram - II

The above diagram represents the flow of operations in the system. As seen in the diagram, the first activity is to select an area and then the activity diagram is branched into five different functions, i.e. reading temperature, reading moisture, reading humidity, reading N-P-K values and reading crop details. The model gives us a forecast of crop yield and will recommend a good crop along with its yield based on weather and soil conditions and suggest whether it is the ideal time to apply fertilizer.

Sequence Diagram:

A sequence diagram simply shows the interaction between objects in sequential order, i.e. the order in which these interactions take place. A sequence diagram simply shows the interaction between objects in sequential order, i.e. the order in which these interactions take place.

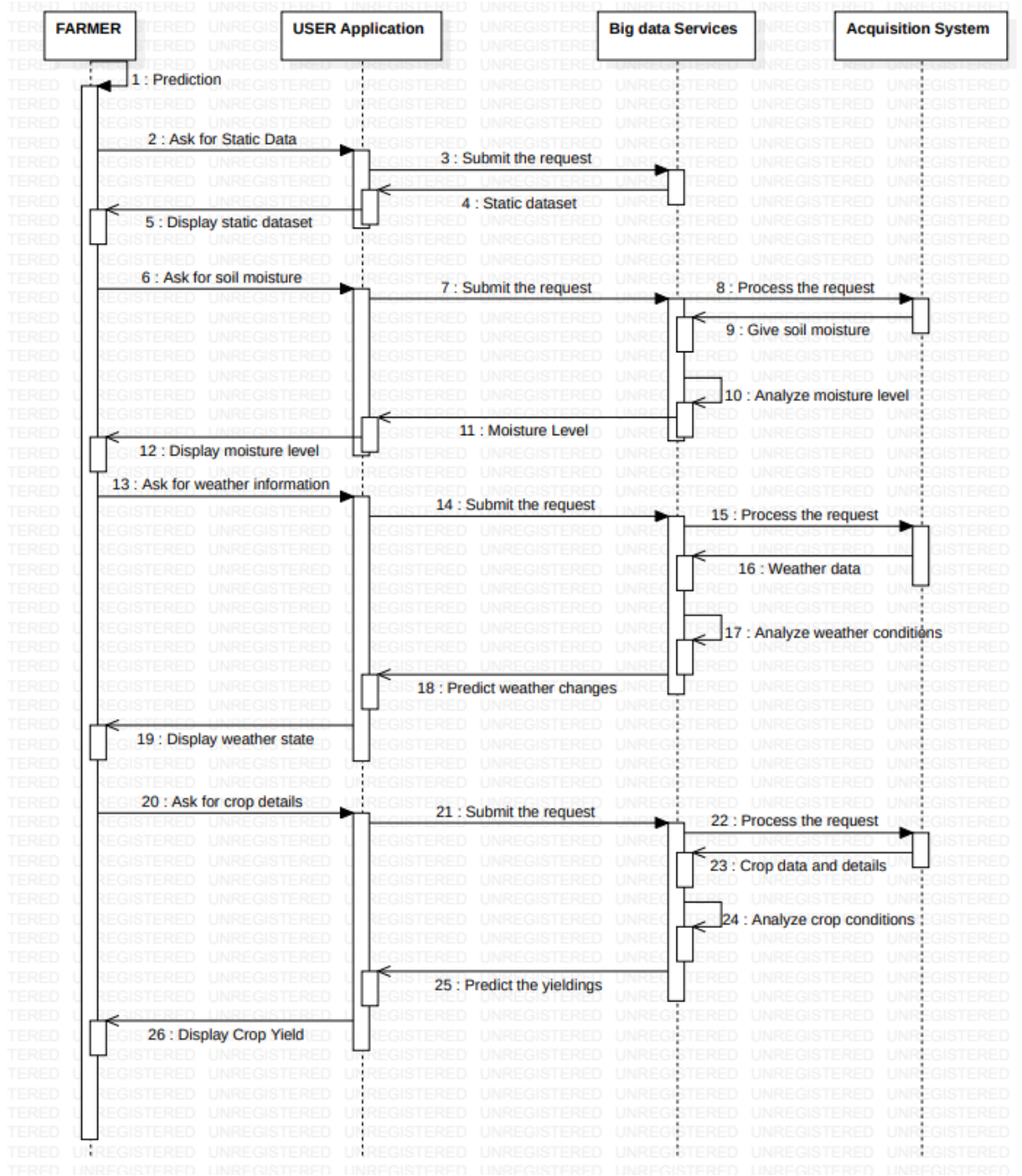


Fig 3.4.3 Sequence Diagram

The figure above represents the various interactions between the user and the objects involved in the system. The various objects or modules involved in the system are the user interface, location tracking, yield forecasting functionality, weather forecasting functionality, and precipitation forecasting API calls. Here are various modules of user interface, location tracking, yield forecast, weather forecast and rainfall forecast. The user interacts with each of the modules to provide corresponding results.

Implementation:

```
import pandas as pd
df = pd.read_csv('C:/Users/VinodKumar/Desktop/SEM-V/CYP/finalised_dataset.csv',na_values='')
df
df=df.drop('Yield', axis = 1)
df.info()
df.columns
df = df[df['state_names'] == "Maharashtra"]
df.info()
df.isnull().sum()
import matplotlib.pyplot as plt
import seaborn as sb
C_mat = df.corr()
fig = plt.figure(figsize = (15,15))
sb.heatmap(C_mat, vmax = .8, square = True)
plt.show()
df = df[df['crop_year']>=2011]
df
df = df.join(pd.get_dummies(df['district_names']))
df = df.join(pd.get_dummies(df['season_names']))
df = df.join(pd.get_dummies(df['crop_names']))
df = df.join(pd.get_dummies(df['state_names']))
df = df.join(pd.get_dummies(df['soil_type']))
df
df['Yield'] = df['production']/df['area']
df
df = df.drop('production', axis=1)
df=df.drop('district_names', axis=1)
df = df.drop('season_names',axis=1)
df = df.drop('crop_names',axis=1)
df = df.drop('state_names', axis=1)
df = df.drop('soil_type', axis=1)
df

from sklearn import preprocessing
# Create x, where x the 'scores' column's values as floats
x = df[['area']].values.astype(float)
x
# Create a minimum and maximum processor object
min_max_scaler = preprocessing.MinMaxScaler()
# Create an object to transform the data to fit minmax processor
x_scaled = min_max_scaler.fit_transform(x)
# Run the normalizer on the dataframe
#df_normalized = pd.DataFrame(x_scaled)
x_scaled
df['area'] = x_scaled
df

df = df.fillna(df.mean())
from sklearn.model_selection import train_test_split
a=df
```

```

b = df['Yield']
c = df.drop('Unnamed: 0', axis = 1)
a=c.drop('Yield', axis = 1)
len(a.columns)
a.columns
features_list=['crop_year', 'area', 'temperature', 'wind_speed', 'pressure',
              'humidity', 'N', 'P', 'K', 'AHMEDNAGAR', 'AKOLA', 'AMRAVATI',
              'AURANGABAD', 'BEED', 'BHANDARA', 'BULDHANA', 'CHANDRAPUR',
              'DHULE',
              'GADCHIROLI', 'GONDIA', 'HINGOLI', 'JALGAON', 'JALNA', 'KOLHAPUR',
              'LATUR', 'NAGPUR', 'NANDED', 'NANDURBAR', 'NASHIK', 'OSMANABAD',
              'PALGHAR', 'PARBHANI', 'PUNE', 'RAIGAD', 'RATNAGIRI', 'SANGLI',
              'SATARA', 'SINDHUDURG', 'SOLAPUR', 'THANE', 'WARDHA', 'WASHIM',
              'YAVATMAL', 'Kharif', 'Rabi', 'Summer', 'Whole Year',
              'Arhar/Tur', 'Bajra', 'Castor seed', 'Cotton(lint)', 'Gram',
              'Groundnut', 'Jowar', 'Linseed', 'Maize', 'Moong(Green Gram)',
              'Niger seed', 'Other Rabi pulses', 'Other Cereals & Millets',
              'Other Kharif pulses', 'Ragi', 'Rapeseed & Mustard', 'Rice', 'Safflower',
              'Sesamum', 'Soyabean', 'Sugarcane', 'Sunflower', 'Tobacco', 'Urad',
              'Wheat', 'other oilseeds', 'Maharashtra', 'chalky', 'clay', 'loamy',
              'peaty', 'sandy', 'silt', 'silty']
features_list123=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
len(features_list123)

a=df[features_list]
a.head()

a_train, a_test, b_train, b_test = train_test_split(a, b, test_size = 0.3, random_state = 42)
print(a_train)
print(a_test)
print(b_train)
print(b_test)

import numpy as np
import matplotlib.pyplot as plt
import seaborn as seabornInstance
from sklearn.linear_model import LinearRegression
from sklearn import metrics
%matplotlib inline

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
a_train = sc.fit_transform(a_train)
a_test = sc.transform(a_test)

from sklearn.ensemble import RandomForestRegressor
regr = RandomForestRegressor(max_depth=2, random_state=0, n_estimators=100)
regr.fit(a_train, b_train)
b_pred = regr.predict(a_test)

```

```

from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import mean_absolute_error as mae
from sklearn.metrics import r2_score

print('MSE =', mse(b_pred, b_test))
print('MAE =', mae(b_pred, b_test))
print('R2 Score =', r2_score(b_pred, b_test))
b_pred

from sklearn.svm import SVR
regressorpoly=SVR(kernel='poly',epsilon=1.0)
regressorpoly.fit(a_train,b_train)
pred=regressorpoly.predict(a_test)
print(regressorpoly.score(a_test,b_test))
print(r2_score(b_test,b_pred))

from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error
XGBModel = XGBRegressor()
XGBModel.fit(a_train,b_train , verbose=False)

# Get the mean absolute error on the validation data :
XGBpredictions = XGBModel.predict(a_test)
MAE = mean_absolute_error(b_test , XGBpredictions)
print('XGBoost validation MAE = ',MAE)
XGBpredictions

print(r2_score(b_test , XGBpredictions))

import pickle
# Dump the trained SVM classifier with Pickle
SVM_pkl_filename = 'xgboost_yield_prediction_final.pkl'
# Open the file to save as pkl file
SVM_Model_pkl = open(SVM_pkl_filename, 'wb')
pickle.dump(XGBpredictions, SVM_Model_pkl)
# Close the pickle instances
SVM_Model_pkl.close()

import joblib
# Save the model as a pickle in a file
joblib.dump(XGBModel, 'xgboost_yield_prediction_final.pkl')
# Load the model from the file
knn_from_joblib = joblib.load('xgboost_yield_prediction_final.pkl')

from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation, Flatten
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error

```

```

from matplotlib import pyplot as plt
import seaborn as sb
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import warnings
from tensorflow.keras.callbacks import History
warnings.filterwarnings('ignore')
warnings.filterwarnings('ignore', category=DeprecationWarning)

NN_model = Sequential()

# The Input Layer :
NN_model.add(Dense(128, kernel_initializer='normal',input_dim = a_train.shape[1],
activation='relu'))
# The Hidden Layers :
NN_model.add(Dense(256, kernel_initializer='normal',activation='relu'))
NN_model.add(Dense(256, kernel_initializer='normal',activation='relu'))
NN_model.add(Dense(256, kernel_initializer='normal',activation='relu'))
# The Output Layer :
NN_model.add(Dense(1, kernel_initializer='normal',activation='linear'))
# Compile the network :
NN_model.compile(loss='mean_absolute_error', optimizer='adam',
metrics=['accuracy','mean_absolute_error'])
NN_model.summary()

from tensorflow.keras.callbacks import History
history = History()
History=NN_model.fit(a_train, b_train, epochs=50, batch_size=500, validation_split = 0.2,
callbacks=[history])

print(history.history.keys())
plt.plot(History.history['mean_absolute_error'])
plt.ylabel('mean_absolute_error')
plt.xlabel('epoch')

plt.plot(History.history['mean_absolute_error'])
plt.plot(History.history['val_mean_absolute_error'])
plt.title('mean_absolute_error')
plt.ylabel('mean_absolute_error')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

```

Chapter 4

Results and Discussion

The different machine learning algorithms used in the project produced different kinds of results with different accuracy scores.

For any system, its efficiency and accuracy are important. Similarly, in our system, accuracy is a key property to judge the correctness of the model. In our model, we considered the Maharashtra region with average minimum and maximum temperature, average rainfall and average minimum and maximum pressure datasets. We have considered the data points of all of the above

parameters for the years 2007 to 2021. In these years 2007 to 2019, the data points are considered as training sets and

2019-2021 data points as test sets. Using these algorithms “Random Forest, SVM, XGBooster and Neural Networks” we evaluated the accuracy for rice, ragi and sugarcane crops. We observed the accuracy for seasonal crops (rice and ragi) using our model as follows:

The following are the results for the Random Forest Algorithm:

Result Type	Value
Mean Squared Error(MSE)	7.671048879964048
Mean Average Error(MAE)	0.8953650873829122
R2 Score	0.9589614680509005

Table 4.1 Results of Random Forest

The following are the results for the Polynomial Support Vector Machine(SVM) Algorithm:

Result Type	Value
Polynomial SVM Score	0.6312485850825429
R2 Score	0.9598856437260073

Table 4.2 Results of Polynomial SVM

The following are the results for theXGBooster Algorithm:

Result Type	Value
XGBoost validation MAE	0.6670485576475581
R2 Score	0.9654928330252374

Table 4.3 Results of XGBooster Algorithm

The following are the results of the Neural Network Algorithm:

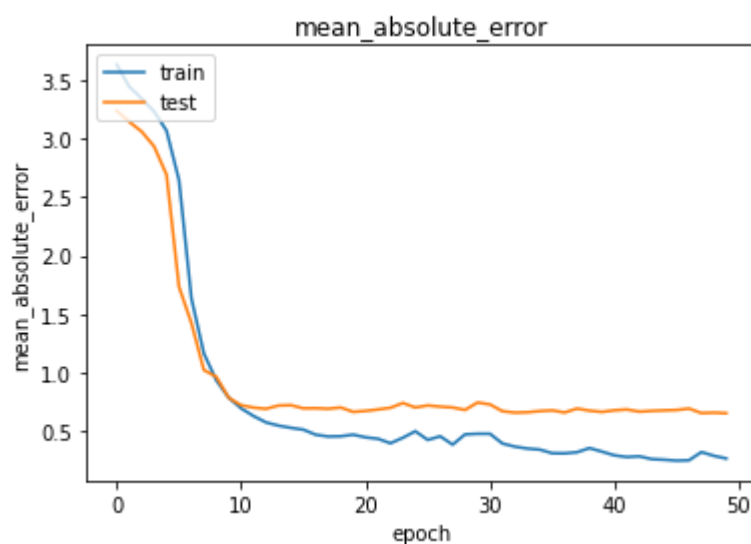


Fig 4.1.1 Mean Absolute Error

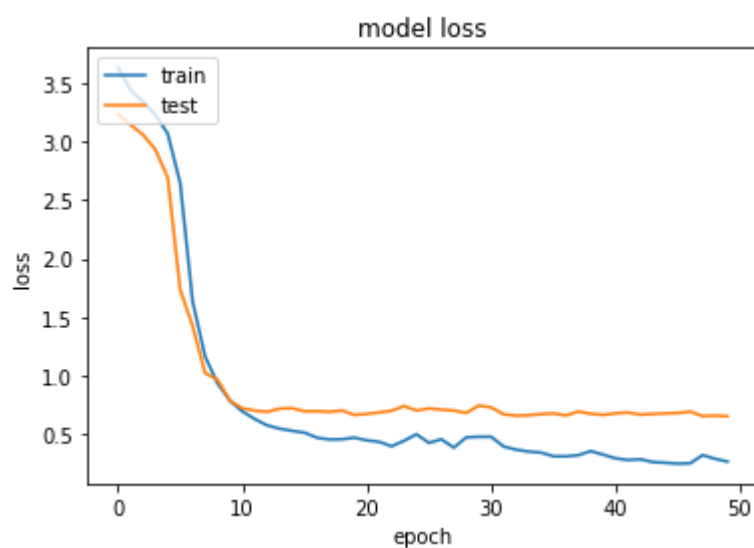


Fig 4.1.2 Model loss

Where

Mean Squared Error(MSE):

$$MSE = \frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2$$

Σ : Greek symbol for summation

Y_i : Actual value for the i th observation

\hat{Y}_i : calculated value for the i th observation

N : Total number of observations

Mean Absolute Error (MAE):

$$MAE = (1/n) * \sum |y_i - x_i|$$

Σ : Greek symbol for summation

y_i : Actual value for the i th observation

x_i : Calculated value for the i th observation

n : Total number of observation

R2 Score

$$R^2 = 1 - SS_{res} / SS_{tot}$$

SS_{res} = the sum of squares of the residual errors.

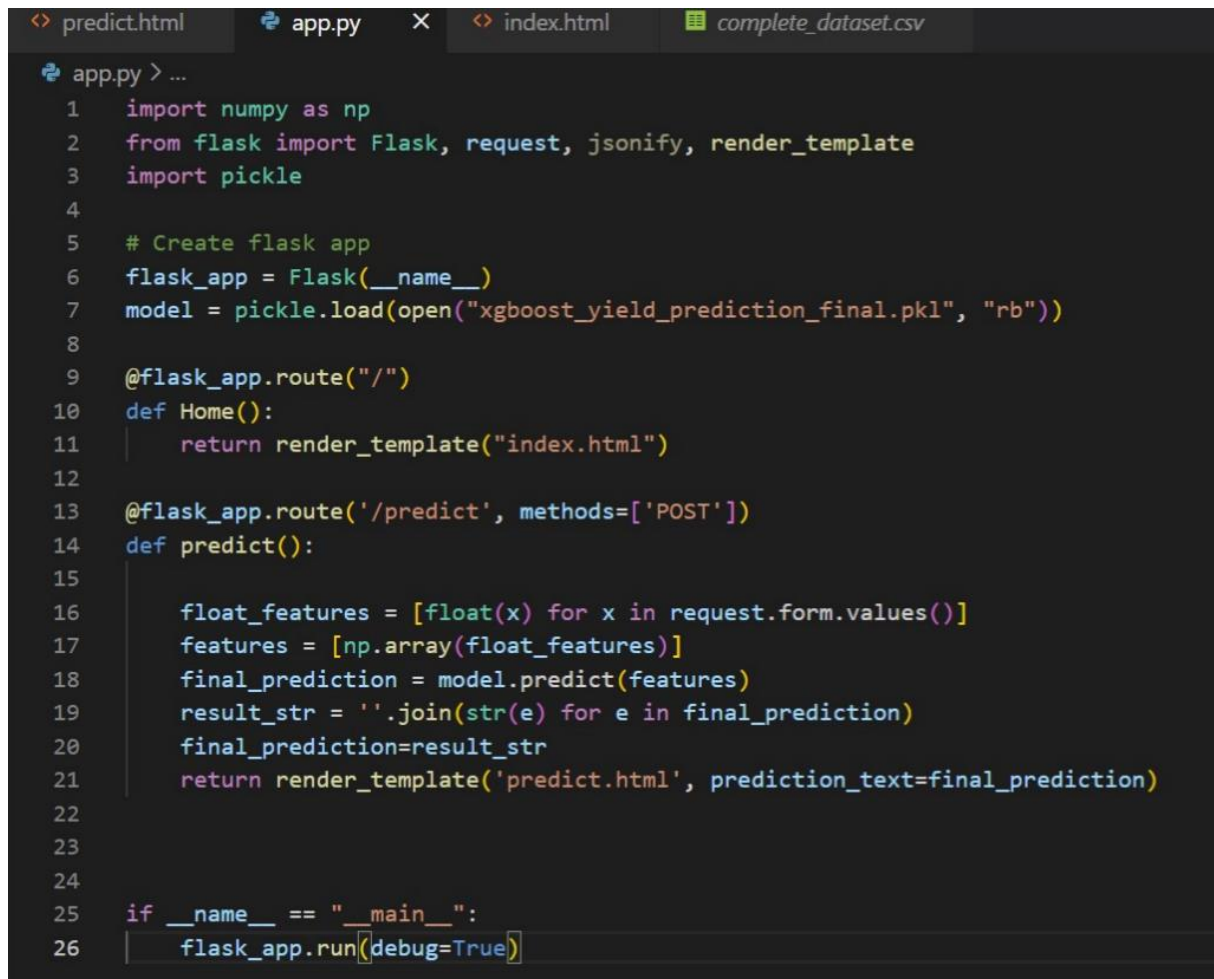
SS_{tot} = the total sum of the errors

Discussion of Results:

From the given error values, we can understand that Extreme Gradient Booster (XGBooster) is the algorithm that gives the highest R2 Score. This is because in XGBoster, after the model is trained on a part of the dataset, the model is tested. After the testing is done, the weights of the misclassified points are increased so that the probability of their selection increases. In this way, we ensure that the misclassified points are trained more so that the machine learning algorithm can correctly classify these points in the future.

Because of this regular updation of weights and repeated training, XGBooster algorithm gives the highest accuracy while predicting.

SnapShots:



```
<> predict.html  app.py  X  <> index.html  complete_dataset.csv
app.py > ...
1  import numpy as np
2  from flask import Flask, request, jsonify, render_template
3  import pickle
4
5  # Create flask app
6  flask_app = Flask(__name__)
7  model = pickle.load(open("xgboost_yield_prediction_final.pkl", "rb"))
8
9  @flask_app.route("/")
10 def Home():
11     return render_template("index.html")
12
13 @flask_app.route('/predict', methods=['POST'])
14 def predict():
15
16     float_features = [float(x) for x in request.form.values()]
17     features = [np.array(float_features)]
18     final_prediction = model.predict(features)
19     result_str = ''.join(str(e) for e in final_prediction)
20     final_prediction=result_str
21     return render_template('predict.html', prediction_text=final_prediction)
22
23
24
25 if __name__ == "__main__":
26     flask_app.run(debug=True)
```

Fig 4.2 User Interface Code

Crop Yield Prediction

Select state ▾	
Select season ▾	Haryana ▾
Select Crop ▾	Kharif ▾
temperature	Wheat ▾
wind speed	30
precipitation	2
humidity	1010
Select soil type ▾	64
Nitrogen value	clay ▾
Phosphorus value	8
Potassium value	18
	20
Predict	Predict

Fig 4.3 User Interface

Crop Yields 4.0793695 quintal for an acre

Fig 4.4 Result/ Prediction

Chapter 5

Conclusion and Future Scope

This scheme is designed to tackle the rising suicide rate of farmers and help them become financially stronger. The Crop Recommender system helps farmers predict the yield of a given crop and also helps them decide which crop to grow. In addition, it also informs the user about the right time to use the fertilizer. Appropriate datasets were collected, studied and trained using machine learning tools. The system tracks the user's location and, based on the location, gets the necessary information from the backend. So the user needs to provide limited information such as soil type and area. This system contributes to the field of agriculture. One of the most important and novel contributions of the system is to suggest to the user the right time to apply fertilizer, by forecasting the weather for the next 14 days. The system also provides a list of crops with their productions based on climatic conditions.

Future work will focus on providing a sequence of crops to grow depending on soil and weather conditions and update the datasets from time to time to make accurate predictions. The Future Work focuses on a fully automated system that will do the same. We also aim to anticipate a crisis situation in advance, such as the recent increase in onion prices. In the future, this model can be implemented across India by adding data points for all regions. Our system can be integrated with a messaging module so that registered farmers can receive forecast notifications directly to their registered mobile numbers.

References

1. Anakha Venugopal, Aparna S, Jinsu Mani, Rima Mathew, Vinu Williams Crop Yield Prediction using Machine Learning Algorithms, 2021
2. Dhivya Elavarasan, P. M. Durairaj Vincent “Crop Yield Prediction Using Deep Reinforcement Learning Model for Sustainable Agrarian Applications”, IEEE Access, 2020
3. Thomas van Klompenburg, Ayalew Kassahun, Cagatay Catal “Crop yield prediction using machine learning: A systematic literature review” Computers and Electronics in Agriculture, Volume 177, October 2020
4. Sangeeta, Shruthi G, “Design And Implementation Of Crop Yield Prediction Model In Agriculture”, 2020
5. YongKang Xing, JiaPeng Huang, YongYao Lai “Research and Analysis of the Front-end Frameworks and Libraries in E-Business Development”, Proceedings of the 2019 11th International Conference on Computer and Automation Engineering - ICCAE 2019, 2019.
6. Ranjini B Guruprasad, Kumar Saurav, Sukanya Randhawa, “Machine Learning Methodologies for Paddy Yield Estimation in India: A CASE STUDY”, 2019
7. P.Priya, U.Muthaiah & M.Balamurugan Predicting Yield of the Crop Using Machine Learning Algorithm, International Journal of Engineering Sciences & Research Technology (IJESRT), April, 2018
8. S. Pavani, Augusta Sophy Beulet P “Heuristic Prediction of Crop Yield using Machine Learning Technique”, International Journal of Engineering and Advanced Technology (IJEAT) Volume-9, December 2019
9. Sriram Rakshith.K, Dr. Deepak.G, Rajesh M, Sudarshan K S, Vasanth S & Harish Kumar, “A Survey on Crop Prediction using Machine Learning Approach” International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 7, Issue IV, Apr 2019
10. Charoen-Ung, P., Mittrapiyanuruk Sugarcane Yield Grade Prediction Using Random Forest with Forward Feature Selection and Hyper-parameter Tuning, 2019
11. Senthil Kumar Swami Durai, Mary Divya Shamili “Smart farming using Machine Learning and Deep Learning techniques” Decision Analytics Journal, Volume 3, June 2022, 100041
12. Madhuri Shripathi Rao¹, Arushi Singh¹, N.V. Subba Reddy¹ and Dinesh U Acharya¹ Agriculture Crop Selection and Yield Prediction using Machine Learning Algorithms Conference paper Feb 2022
13. Chlingaryan, A., Sukkarieh, S., & Whelan, B. (2018). Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. Computers and electronics in agriculture, 151, 61-69.
14. Madhuri Shripathi Rao¹, Arushi Singh¹, N.V. Subba Reddy¹ and Dinesh U Acharya, AICECS 2021 Journal of Physics: Conference Series, **2161**(2022) 012033, IOP Publishing doi:10.1088/1742-6596/2161/1/012033
15. <https://www.agricultureinformation.com/forums/threads/agriculture-in-india.101537/>
16. <https://www.data.gov.in/>
17. <https://openweathermap.org/>