PROJECT REPORT
ON
Crop Recommendation System Using Machine Learning

by

ARCHIT GARG (1900270100044)

ARYAMAN BENARA (1900270100050)

ARYAN AGGARWAL (1900270100051)

ARYAN PATEL (1900270100053)

Submitted

to

Department of Computer Science & Engineering

in partial fulfillment of the requirements

for the degree

of

Bachelor of Technology

in

Computer Science & Engineering

Ajay Kumar Garg Engineering College, Ghaziabad

Dr. APJ Abdul Kalam Technical University, Lucknow

May, 2023

Table of Contents

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature :

Name ***: Aryan Aggarwal***

Roll No : 1900270100051

Date :

Signature :

Name : ***Aryan Patel***

Roll No : 190027010053

Date :

Signature :

Name : ***Archit Garg***

Roll No : 1900270100044

Date :

Signature :

Name : ***Aryaman Benara***

Roll No : 1900270100050

Date :

CERTIFICATE

This is certified that ***Aryan Aggarwal , Aryan Patel , Archit Garg , Aryaman Benara*** have carried out the Project work entitled "CRS" for the award of Bachelor of Technology from Ajay Kumar Garg Engineering College, Ghaziabad under our supervision.

To the best of our knowledge, this work has not been submitted earlier to any university for the award of any degree.

Dr. Santosh Kumar Upadhyay
Assistant Professor
Project Guide

Dr. Sunita Yadav
Head of Department
Computer Science and Engineering

ACKNOWLEDGEMENT

We would like to express our sincerest gratitude to all the people who have contributed towards the successful completion of our project. We would like to extend our heartfelt thanks to the Head of Computer Science and Engineering Department Dr. Sunita Yadav, for nurturing a congenial yet competitive environment in the department, which motivates all the students to pursue higher goals.

We want to express our special gratitude to our guide Mrs Jaishree Jain, Assistant Professor Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad for his/her constant support, guidance, encouragement and much needed motivation. His/Her sincerity, thoroughness and perseverance has been a constant source of inspiration for us.

Last but not the least, we would like to extend our thanks to all the teaching and non teaching staff members of our department, and to all our colleagues who helped us in completion of the project.

Name : ***Aryan Aggarwal***

Roll No : 1900270100051

Name : ***Aryan Patel***

Roll No : 1900270100053

Name : ***Archit Garg***

Roll No : 1900270100044

Name : ***Aryaman Benara***

Roll No : 1900270100050

Abstract

Agriculture is a major contributor to the Indian economy. The com- mon problem existing among the Indian farmers are they don't choose the right crop based on their soil requirements. Due to this they face a serious setback in productivity. This problem of the farmers has been addressed through precision agriculture. Precision agriculture is a mod- ern farming technique that uses research data of soil characteristics, soil types, crop yield data collection and suggests the farmers the right crop based on their site-specific parameters. This reduces the wrong choice on a crop and increases the productivity. In this project, we are building an intelligent system, which intends to assist the Indian farmers in making an informed decision about which crop to grow depending on the sowing season, his farm's geographical location and soil characteristics. Further the system will also provide the farmer, the yield prediction if he plants the recommended crop.

Keywords: *Precision Agriculture, yield pridiction.*

List of Figures

Chapter 1 : Preamble

1. Introduction

A farmer's decision about which crop to grow is generally clouded by his intuition and other irrelevant factors like making instant profits, lack of awareness about market demand, overestimating a soil's potential to support a particular crop, and so on. A very misguided decision on the part of the farmer could place a significant strain on his family's financial condition. Perhaps this could be one of the many reasons contributing to the countless suicide cases of farmers that we hear from media on a daily basis. In a country like India, where agriculture and related sector contributes to approximately 20.4 per cent of its Gross Value Added (GVA) [2], such an erroneous judgment would have negative implications on not just the farmer's family, but the entire economy of a region. For this reason, we have identified a farmer's dilemma about which crop to grow during a particular season, as a very grave one. The need of the hour is to design a system that could provide predictive insights to the Indian farmers, thereby helping them make an informed decision about which crop to grow. With this in mind, we propose a system, an intelligent system that would consider environmental parameters (temperature, rainfall, geographical location in terms of state) and soil characteristics (pH value, soil type and nutrients concentration) before recommending the most suitable crop to the user.

2. Existing System

More and more researchers have begun to identify this problem in Indian agriculture and are increasingly dedicating their time and efforts to help alleviate the issue. Different works include the use of Regularized Greedy Forest to determine an appropriate crop sequence at a given time stamp. Another approach proposes a model that makes use of historical records of meteorological data as training set. Model is trained to identify weather conditions that are deterrent for the production of apples. It then efficiently predicts the yield of apples on the basis of monthly weather patterns. The use of several algorithms like Artificial Neural Network, K Nearest Neighbours, and Regularized Greedy Forest is demonstrated in [5] to select a crop based on the pre- diction yield rate, which, in turn, is influenced by multiple parameters. Additional features included in the system are pesticide prediction and online

trading based on agricultural commodities.

2.1. Drawbacks

One shortcoming that we identified in all these notable published works was that the authors of each paper concentrated on a single parameter (either weather or soil) for predicting the suitability of crop growth. However, in our opinion, both these factors should be taken together into consideration concomitantly for the best and most accurate prediction. This is because, a particular soil type may be fit for supporting one type of crop, but if the weather conditions of the region are not suitable for that crop type, then the yield will suffer.

3. Proposed System

We to eliminate the aforementioned drawbacks, we propose an Intelligent Crop Recommendation system- which takes into consideration all the appropriate parameters, including temperature, rainfall, location and soil condition, to predict crop suitability. This system is fundamentally concerned with performing the primary function of AgroConsultant, which is, providing crop recommendations to farmers algorithms.We also provide the profit analysis on crops grown in different states which gives the user an easy and reliable insight to decide and plan the crops.

4. Plan of Implementation

The steps involved in this system implementation are:-

a) Acquisition of Training Dataset:The accuracy of any machine learning algorithm depends on the number of parameters and the correctness of the training dataset.For the system, we are using various datasets all downloaded for government website and kaggle.

Datasets include:-

- Nitrogen
- Phosphorus
- Potassium

- Humidity
- Ph
- Rainfall

b) Data Preprocessing: This step includes replacing the null and 0 values for yield by -1 so that it does not effect the overall prediction.Further we had to encode the data-set so that it could be fed into the neural network.

c) Training model and crop recommendation:After the preprocessing step we used the data-set to train different machine learning models like neural network and linear regression to attain accuracy as high as possible.

5. Problem Statement

Failure of farmers to decide on the best suited crop for his land using traditional and non-scientific methods is a serious issue for a country where approximately 50 percent of the population is involved in farming.Both availability and accessibility of correct and up to date information hinders potential researchers from working on developing country case studies.With resources within our reach we have proposed a system which can address this problem by providing predictive insights on crop sustainability and recommendations based on machine learning models trained considering essential environmental and economic parameters.

6. Objectives Of The Project

The primary objectives of crop recommendation systems are to provide farmers and agricultural stakeholders with informed guidance on which crops to cultivate in specific regions and under certain environmental conditions. The key objectives of crop recommendation include:

- **Maximizing Crop Productivity:** The primary goal of crop recommendation systems is to help farmers maximize their crop yields and overall productivity. By analysing various factors such as climate, soil conditions, water availability, and pest prevalence, crop recommendations can suggest the most suitable crops for a given area. This helps farmers make informed decisions that can lead to higher yields and better economic outcomes.

- **Ensuring Environmental Sustainability:** Crop recommendation systems aim to promote environmentally sustainable agricultural practices. By considering factors such as soil health, water availability, and climate suitability, these systems can recommend crops that are well-

adapted to local conditions and have a lower environmental impact. Sustainable crop choices help in reducing resource consumption, minimizing soil erosion, and preserving biodiversity.

- **Mitigating Risks and Uncertainties**: Agricultural production is susceptible to various risks and uncertainties, including climate change, pests, diseases, and market fluctuations. Crop recommendation systems aim to address these challenges by providing guidance on crop diversification, pest and disease management strategies, and climate-resilient crop varieties. By minimizing risks and uncertainties, farmers can achieve more stable and profitable agricultural outcomes.

- **Enhancing Resource Efficiency:** Efficient use of resources, including water, fertilizers, and energy, is crucial in modern agriculture. Crop recommendation systems consider the resource requirements of different crops and recommend those that are best suited to the available resources. This helps optimize resource allocation, reduce waste, and improve overall resource efficiency in farming practices.

- **Supporting Decision-Making:** Crop recommendation systems provide farmers with valuable decision support tools. By considering multiple variables, such as climate data, soil characteristics, market trends, and farmers' goals, these systems offer personalized recommendations tailored to specific farming contexts. This assists farmers in making informed decisions regarding crop selection, planting schedules, fertilization, and irrigation strategies.

- **Improving Farm Profitability and Economic Sustainability**: Crop recommendation systems aim to enhance farm profitability and economic sustainability. By suggesting crops that align with market demand, have good market value, and are well-suited to the local conditions, these systems can help farmers optimize their crop choices and increase their economic returns. Improved profitability leads to greater farm sustainability and livelihoods for farmers.

Overall, the objectives of crop recommendation systems revolve around maximizing crop productivity, ensuring environmental sustainability, mitigating risks, enhancing resource efficiency, supporting decision-making, and improving farm profitability. By leveraging data-driven insights and advanced analytics, these systems empower farmers to make informed choices and achieve more sustainable and profitable agricultural practices.

.

Chapter 2 : Literature Survey

Crop Selection Method to Maximize Crop Yield Rate using Machine Learn- ing Technique Authors: Rakesh Kumar, M.P. Singh, Prabhat Kumar and J.P. Singh

This paper proposed a method named Crop Selection Method (CSM) to solve crop selection problem, and maximize net yield rate of crop over season and subsequently achieves maximum economic growth of the country. The proposed method may improve net yield rate of crops.

AgroConsultant: Intelligent Crop Recommendation System Using Ma- chine Learning Algorithms Authors: Zeel Doshi, Subhash Nadkarni, Rashi Agrawal, Prof. Neepa Shah

This paper, proposed and implemented an intelligent crop recommendation system, which can be easily used by farmers all over India. This system would assist the farmers in making an informed decision about which crop to grow depending on a variety of environmental and geographical factors. We have also implemented a secondary system, called Rainfall Predictor, which predicts the rainfall of the next 12 months.

Development of Yield Prediction System Based on Real-time Agricultural meteorological Information Haedong Lee *, Aekyung Moon* * ETRI, 218 Gajeong-ro, Yuseong-gu, 305-700, Korea

This paper contains about the research and the building of an effective agricultural yield forecasting system based on real-time monthly weather. It is difficult to predict the agricultural crop production because of the abnormal weather that happens every year and rapid regional climate change due to global warming. The development of agricultural yield forecasting system that leverages real-time weather information is urgently required. In this research, we cover how to process the number of weather data(monthly, daily) and how to configure the prediction system. We establish a non- parametric

statistical model on the basis of 33 years of agricultural weather in- formation. According to the implemented model, we predict final production using the monthly weather information. This paper contains the results of the simulation.

Analysis of Soil Behaviour and Prediction of Crop Yield using Data Mining Approach Monali Paul, Santosh K. Vishwakarma, Ashok Verma Computer science and Engineering GGITS, Jabalpur

This work presents a system, which uses data mining techniques in order to predict the category of the analyzed soil datasets. The category, thus predicted will indicate the yielding of crops. The problem of predicting the crop yield is formalized as a classification rule, where Naive Bayes and K-Nearest Neighbor methods are used.

Crop Recommendation System for Precision Agriculture S.Pudumalar*, E.Ramanujam*, R.Harine Rajashree, C.Kavya, T.Kiruthika, J.Nisha.

This paper, proposes a recommendation system through an ensemble model with majority voting technique using Random tree, CHAID, K-Nearest Neighbor and Naive Bayes as learners to recommend a crop for the site specific parameters with high accuracy and efficiency.

Chapter 3 : Theoretical Background

1. Overview on Machine Learning

Machine learning is an application of artificial intelligence (AI) that gives systems the ability to automatically learn and evolve from experience without being specially programmed by the programmer. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The main aim of machine learning is to allow computers to learn automatically and adjust their actions to improve the accuracy and usefulness of the program, without any human intervention or assistance. Traditional writing of programs for a computer can be defined as automating the procedures to be performed on input data in order to create output artefacts. Almost always, they are linear, procedural and logical. A traditional program is written in a programming language to some specification, and it has properties like:

- We know or can control the inputs to the program.
- We can specify how the program will achieve its goal.
- We can map out what decisions the program will make and under what conditions it makes them.
- Since we know the inputs as well as the expected outputs, we can be con dent that the program will achieve its goal

Traditional programming works on the premise that, as long as we can define what a program needs to do, we are confident we can define how a program can achieve that goal. This is not always the case as sometimes, however, there are problems that you can represent in a computer that you cannot write a traditional program to solve. Such problems resist a procedural and logical solution. They have properties such as:

- The scope of all possible inputs is not known .
- You cannot specify how to achieve the goal of the program, only what that goal is.
- You cannot map out all the decisions the program will need to make to achieve its goal.
- You can collect only sample input data but not all possible input data for the program.

3.1.1 Supervised and Unsupervised Learning

Machine learning techniques can be broadly categorised into the following types:

Supervised learning takes a set of feature/label pairs, called the training set. From this training set the system creates a generalised model of the relationship between the set of descriptive features and the target features in the form of a program that contains a set of rules. The objective is to use the output program produced to predict the label for a previously unseen, unlabelled input set of features, i.e. to predict the outcome for some new data. Data with known labels, which have not been included in the training set, are classified by the generated model and the results are compared to the known labels. This dataset is called the test set. The accuracy of the predictive model can then be calculated as the proportion of the correct predictions the model labeled out of the total number of instances in the test set.

Unsupervised learning takes a dataset of descriptive features without labels as a train- ing set. In unsupervised learning, the algorithms are left to themselves to discover interesting structures in the data. The goal now is to create a model that finds some hidden structure in the dataset, such as natural clusters or associations. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system does not figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data. Unsupervised learning can be used for clustering, which is used to discover any inherent grouping that are already present in the data. It can also be used for association problems, by creating rules based on the data and finding

relationships or associations between them.

Semi-supervised machine learning falls somewhere in between supervised and unsupervised learning, since they use both labeled and unlabelled data for training typically a small amount of labeled data and a large amount of unlabelled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi- supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring labeled data generally does not require additional resources.

Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards.Machine learning algorithms are tools to automatically make decisions from data in order to achieve some over-arching goal or requirement. The promise of machine learning is that it can solve complex problems automatically, faster and more accurately than a manually specified solution, and at a larger scale. Over the past few decades, many machine learning algorithms have been developed by researchers, and new ones continue to emerge and old ones modified.

2. Machine Learning Tools

There are many different software tools available to build machine learning models and to apply these models to new, unseen data. There are also a large number of well defined machine learning algorithms available. These tools typically contain libraries implementing some of the most popular machine learning algorithms. They can be categorised as follows :

- Pre-built application-based solutions.
- Programming languages which have specialised libraries for machine learning

Using programming languages to develop and implement models is more flexible and gave us better control of the parameters to the algorithms. It also allows us to have a better understanding of the output models produced. Some of the popular program- ming languages used in the field of machine learning are:

- Python: Python is an extremely popular choice in the field of machine learning and AI development. Its short and simple syntax make it extremely easy to learn

- R: R is one of the most effective and efficient languages for analyzing and manipulating data in statistics. Using R, we can easily produce well-designed publication-quality plot, including mathematical symbols and formulae where needed. Apart from being a general purpose language, R has numerous of pack- ages like RODBC, Gmodels, Class and Tm which are used in the field of ma- chine learning. These packages make the implementation of machine learning algorithms easy, for cracking the business associated problems

- Tensorflow : TensorFlow is an end-to-end open source platform for machine learn- ing. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML-powered applications. TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence Research organization to conduct machine learn- ing and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well. TensorFlow provides stable Python and C++ APIs, as well as non-guaranteed backward compatible API for other languages.

3. SciKit-learn

SciKit learn is an open source machine learning library built for python. Since its release in 2007, Scikit-learn has become one of the most popular open source machine learning libraries. Scikit-learn (also called sklearn) provides algorithms for many ma- chine learning tasks including classification, regression, dimensionality reduction and clustering.

The documentation for scikit-learn is comprehensive, popular and well maintained. Sklearn is built on mature Python Libraries such as NumPy, SciPy, and matplotlib. While languages such as R and MATLAB are extremely popular and useful for ma- chine learning, we decided to choose Python along with its

SciKit-learn libraries as our programming language of choice. The reasons for this are:

- We already have some familiarity and exposure to Python, and thus have a smaller learning curve.

- Both Python and Scikit-learn have excellent documentation and tutorials avail- able online

- The number of classic machine learning algorithms that come with Scikit-learn, and the consistent patterns for using the different models i.e., each model can be used with the same basic commands for setting up the data, training the model and using the model for prediction. This makes it easier to try a range of machine learning algorithms on the same data.

- The machine learning algorithms included with sklearn have modifiable parameters known as hyperparameters that effect the performance of the model. These usually have sensible default values, so that we can run them without needing a detailed knowledge or understanding of their semantics.

- The IPython notebook, which is an interactive computational environment for Python, in which a user can combine code execution, rich text, mathematics and plots in a web page. This functionality allows us to provide the notebooks we used to run our experiments almost as an audit and in a presentable.

4. Dataset

For the system, we are using various datasets all downloaded for government website and kaggle.

Datasets include:-

• Nitrogen
• Phosphorus

- Potassium
- Humidity
- Ph
- Rainfall

A brief description of the datasets:

➢ **Nitrogen:** The nitrogen dataset in crop recommendation refers to a collection of data related to nitrogen fertilizer application in the context of agricultural crop production. Nitrogen is an essential nutrient for plants and plays a crucial role in their growth and development. However, determining the optimal amount of nitrogen fertilizer to apply to a particular crop field can be challenging, as it depends on various factors such as soil characteristics, crop type, environmental conditions, and desired yield.

➢ **Phosphorous:** The phosphorus dataset in crop recommendation refers to a specific set of data related to the phosphorus (P) levels in soil or plants, which is used in the context of recommending suitable crops or providing optimal fertilizer application strategies. Phosphorus is an essential nutrient for plant growth and is crucial for various physiological processes, such as energy transfer, cell division, and root development.

➢ **Potassium:** The "potassium dataset" in crop recommendation refers to a dataset that contains information about the potassium (K) levels in soil samples collected from various agricultural fields or regions. It is a specific dataset used in the context of recommending suitable crops or crops' nutrient management based on soil potassium levels.

➢ **Temperature**: In the context of crop recommendation, a temperature dataset refers to a collection of recorded temperature values associated with specific geographical locations and time periods. Temperature is a crucial environmental factor that significantly influences crop growth, development, and overall productivity. By analyzing temperature data, researchers and agricultural experts can gain insights into the suitability of different crops for specific regions and make informed recommendations to farmers

➢ **Humidity:** The humidity dataset in crop recommendation refers to the information collected and recorded regarding the levels of humidity in a particular region or area. Humidity is a measure of the amount of moisture or water vapor present in the air. In the context of crop recommendation systems, humidity data plays a crucial role in determining the suitability of different crops for cultivation.

- **PH:** The "PH dataset" in the context of crop recommendation refers to a dataset that includes information about soil pH levels for different regions or areas. pH is a measure of the acidity or alkalinity of the soil and is an important factor in determining the suitability of soil for growing specific crops.

- **Rainfall Dataset:** A rainfall dataset used in crop recommendation refers to a collection of historical or current rainfall data that is utilized to assist in making recommendations for suitable crop selection or agricultural practices. This dataset typically contains information about rainfall patterns, including the amount and distribution of rainfall over a specific period in a particular region.

5. Data Preprocessing

This step includes replacing the null and 0 values for yield by -1 so that it does not effect the overall prediction.Further we had to encode the data-set so that it could be fed into the neural network.

Data preprocessing is an essential step in machine learning that involves transforming raw data into a format suitable for training and modeling. Here are some common data preprocessing steps in machine learning:

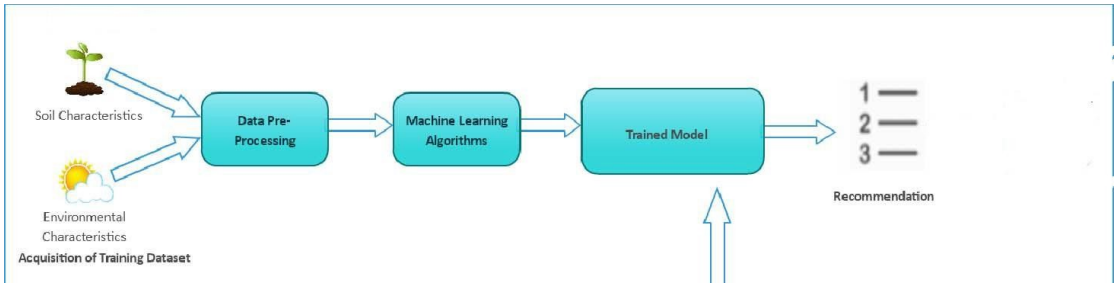
- Data Cleaning: This step involves handling missing data, outliers, and noise in the dataset. Missing data can be imputed using various techniques such as mean imputation, median imputation, or interpolation. Outliers and noisy data points can be detected and treated by methods like trimming, winsorizing, or replacing with appropriate values.

- Data Integration: In some cases, data may be scattered across multiple sources or stored in different formats. Data integration involves combining data from different sources and resolving any inconsistencies or discrepancies. This step ensures that the data is in a unified format for further processing.
- Data Transformation: Data transformation aims to normalize the data distribution or make it conform to certain assumptions of the machine learning algorithms. Common transformations include scaling features to a specific range (e.g., normalization or standardization), logarithmic or power transformations, and handling skewed distributions.
- Feature Selection/Extraction: In this step, relevant features are selected from the dataset or extracted from existing features to reduce dimensionality and improve model

performance. Feature selection techniques include methods like correlation analysis, backward/forward feature selection, or using domain knowledge. Feature extraction techniques such as Principal Component Analysis (PCA) or t-SNE can be used to create new meaningful features.

- Data Encoding: Categorical variables need to be encoded into numerical representations as machine learning models generally operate on numerical data. Common encoding techniques include one-hot encoding, label encoding, or ordinal encoding. This step ensures that categorical variables are in a format that can be processed by machine learning algorithms.
- Data Splitting: The dataset is typically split into training, validation, and testing sets. The training set is used to train the model, the validation set is used for hyper parameter tuning and model selection, and the testing set is used to evaluate the final model's performance. It is important to ensure that the data splitting is done randomly and maintains the same distribution across different subsets.
- Handling Imbalanced Data: If the dataset has imbalanced class distributions, techniques such as oversampling, under-sampling, or generating synthetic samples (e.g., SMOTE) can be used to balance the classes. This step helps prevent bias towards the majority class during model training.

Figure 3.1: Machine Learning process



6. Machine Learning Algorithms

Machine Learning algorithms used in the recommendation system are:

- **Logistic Regression** : Logistic regression is a machine learning algorithm often used in classification problems. This algorithm is used to predict binary value based on one or more inputs.In logistic regression, the output is a probability estimate between 0 and 1 that represents the likelihood of the binary outcome. This probability estimate is then mapped to a discrete class label using a threshold value, usually 0.5. The logistic regression

model is based on the logistic function***(as described in Fig 6)***, which is a sigmoid-shaped curve that transforms any input value into a probability value between 0 and 1. The logistic function is defined as:

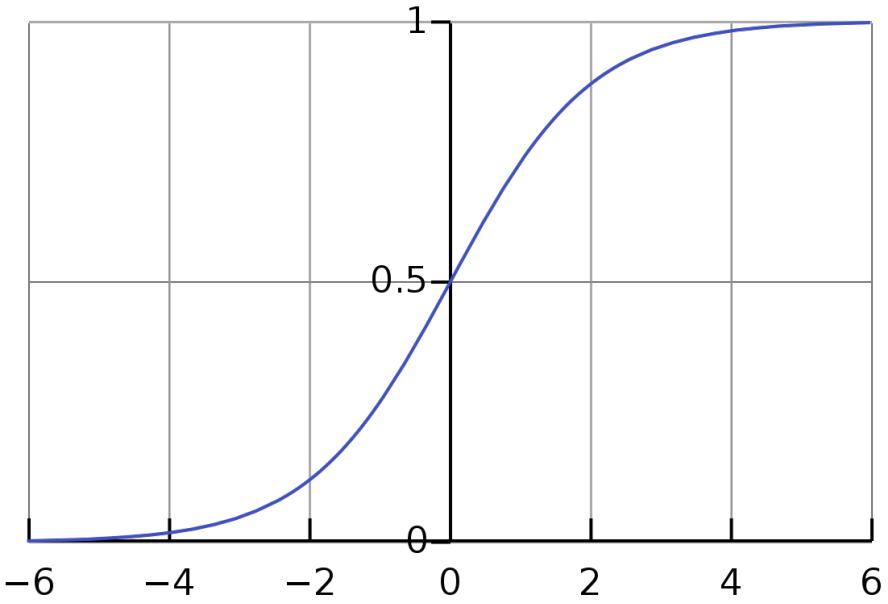
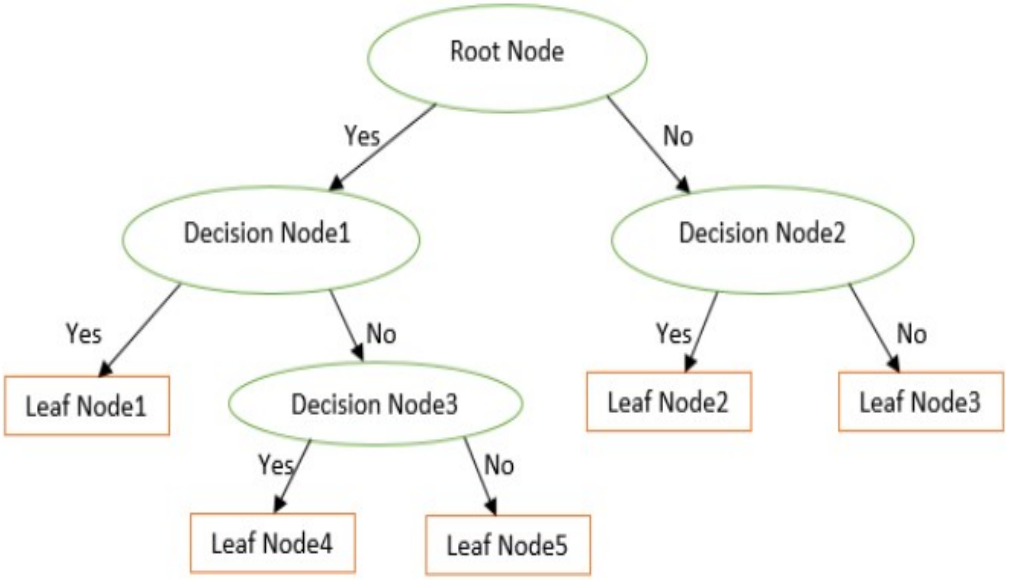


Figure 3.2: Logistic Regression

- **Decision Tree :** Decision tree is a popular machine learning algorithm used for both classification and regression tasks. It is a tree-like structure where each node represents a feature or attribute, each branch represents a decision rule or condition, and each leaf node represents a class label or regression outputs.The decision tree algorithm can handle both categorical and numerical features. For categorical features, the algorithm uses a one-hot encoding or binary split to represent each category as a separate feature. For numerical features, the algorithm uses a threshold or split point to separate the dataset into two subsets.



- **Random Forest** : Random forest is a popular machine learning algorithm used for both classification and regression problems. It is an ensemble method that combines multiple decision trees to improve the accuracy and reduce the overfitting of the model. Once the decision trees are trained, the algorithm combines their predictions using a voting or averaging scheme. For classification problems, the class with the most votes are chosen as the final prediction, while for regression problems, the average of the predictions is taken as the final output.

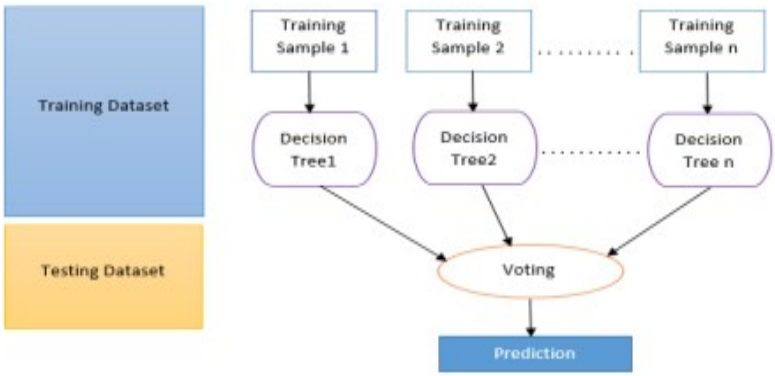


Figure 3.3 : Random Forest

Chapter 4 : System Requirements Specification

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide.

In order to fully understand one's project, it is very important that they come up with a SRS listing out their requirements, how are they going to meet it and how will they complete the project. It helps the team to save upon their time as they are able to comprehend how are going to go about the project. Doing this also enables the team to find out about the limitations and risks early on.

Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable at- tributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources.

1. Functional Requirement

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. Following are the functional requirements on the system:

1. All the data must be in the same format as a structured data.
2. The data collected will be vectorised and sent across to the classifier

2. Non Functional Requirements

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies and the

need for interoperability with other software and hardware systems.

2.1. Product Requirements

Correctness: It followed a well-defined set of procedures and rules to engage a conversation with the user and a pre-trained classification model to compute also rigorous testing is performed to confirm the correctness of the data. Modularity: The complete product is broken up into many modules and well-defined interfaces are developed to explore the benefit of flexibility of the product. Robustness: This software is being developed in such a way that the overall performance is optimized and the user can expect the results within a limited time with utmost relevancy and correctness. Non functional requirements are also called the qualities of a system. These qualities can be divided into execution quality and evolution quality. Execution qualities are security and usability of the system which are observed during run time, whereas evolution quality involves testability, maintainability, extensibility or scalability.

2.2. Organizational Requirements

Process Standards: The standards defined by w3 are used to develop the application which is the standard used by the developers. Design Methods: Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

2.3. Basic Operational Requirements

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:

- **Mission profile or scenario**: It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.
- **Performance and related parameters:** It points out the critical system parameters to accomplish the mission.

- **Utilization environments:** It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.
- **Operational life cycle**: It defines the system lifetime.

2.4. Hardware

System Configuration

- Processor: 2 gigahertz (GHz) or faster processor or SoC.
- RAM: 6 gigabyte (GB) for 32-bit or 8 GB for 64-bit.
- Hard disk space: =16GB.

Software Configuration:

- Operating System: Windows XP/7/8/8.1/10, Linux and Mac
- Coding Language: Python
- Tools:
    1. Pandas
    2. Numpy
    3. Tensorflow
    4. Keras
    5. Sickitlearn

Chapter 5 : System Analysis

1. Feasibility Study

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

1.1. Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. Since the project is Machine learning based, the cost spent in executing this project would not demand cost for softwares and related products, as most of the products are open source and free to use. Hence the project would consumed minimal cost and is economically feasible.

1.2. Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Since machine learning algorithms is based on pure math there is very less requirement for any professional software. And also most of the tools are open source. The best part is that we can run this software in any system without any software requirements which makes them highly portable. Also most of the documentation and tutorials make easy to learn the technology

1.3. Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The main purpose of this project which is based on crop prediction is to prevent the farmer from incurring losses and improve productivity.This also ensures that there is no scarcity of food as lack of production may lead to severe

consequences. Thus, this is a noble cause for the sake of the society, a small step taken to achieve a secure future.

2. Analysis

2.1. Performance Analysis

Most of the software we use is open source and free. The models which we use in this software ,learn only once ,i.e once they are trained they need not be again fed in for the training phase. One can directly predict for values, hence time-complexity is very less. Therefore this model is temporally sound.

2.2. Technical Analysis

As mentioned earlier ,the tools used in building this software is open source. Each tool contains simple methods and the required methods are overridden to tackle the problem.

2.3. Economical Analysis

The completion of this project can be considered free of cost in its entirety. As the software used in building the model is free of cost and all the data sets used are being downloaded from kaggle and Govt. of India website.

Chapter 6 : System Design

1. System Development Methodology

System Development methodology is the the development of a system or method for a unique situation. Having a proper methodology helps us in bridging the gap between the problem statement and turning it into a feasible solution. It is usually marked by converting the System Requirements Specifications (SRS) into a real world solution. System design takes the following inputs:

- Statement of work.
- Requirement determination plan.
- Current situation analysis.
- Proposed system requirements including a conceptual data model and metadata (data about data).

2. Model Phases

The waterfall model is a sequential software development process, in which progress is seen as owing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design, Implementation, Testing and maintenance.

- **Requirement Analysis:** This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.

- **System Design:** Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasises on:- algorithm, data structure, software architecture etc.

- **Coding:** In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in

to machine

- **Implementation:** The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional soft- ware documentation.

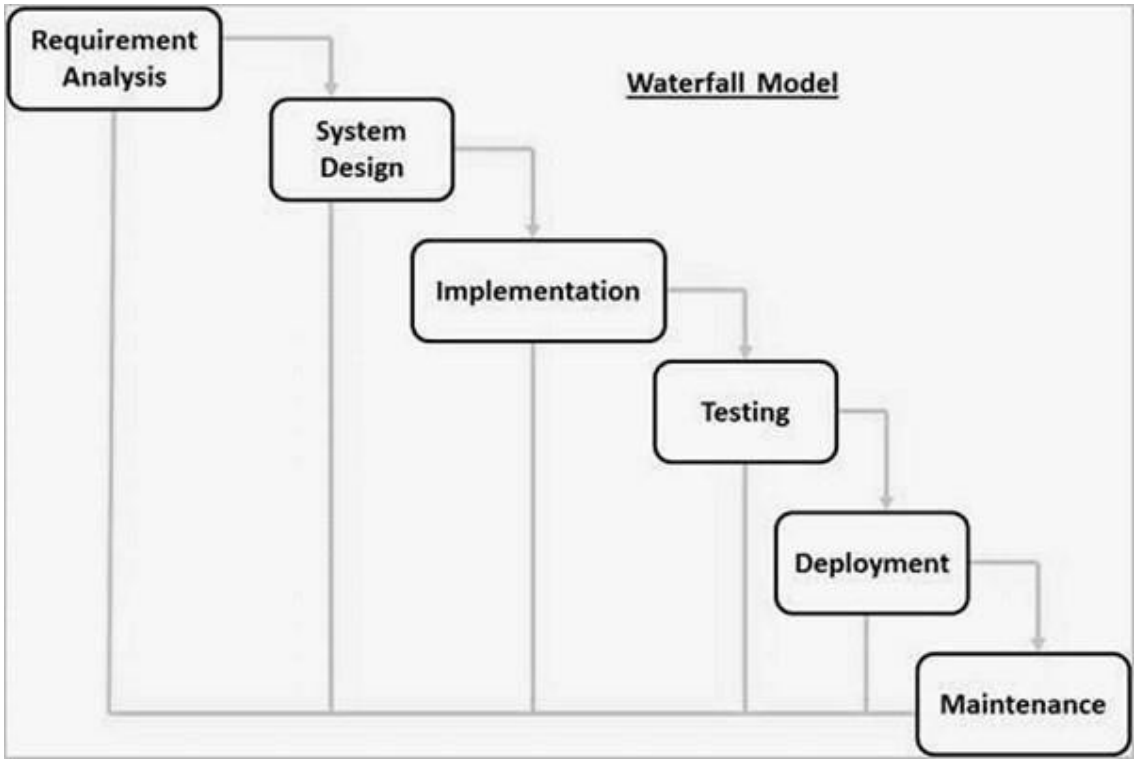


Figure 6.1 Waterfall Model

- **Testing:** In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

- **Maintenance:** The maintenance phase is the longest phase in which the software is updated to fulfil the changing customer needs, adapt to accommodate changes in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

2.1. Advantages of Waterfall model

- Clear project objective

- Stable project requirements
- Progress of system is measurable.
- Logic of software development is clearly understood.
- Better resource allocation.

3. System Architecture

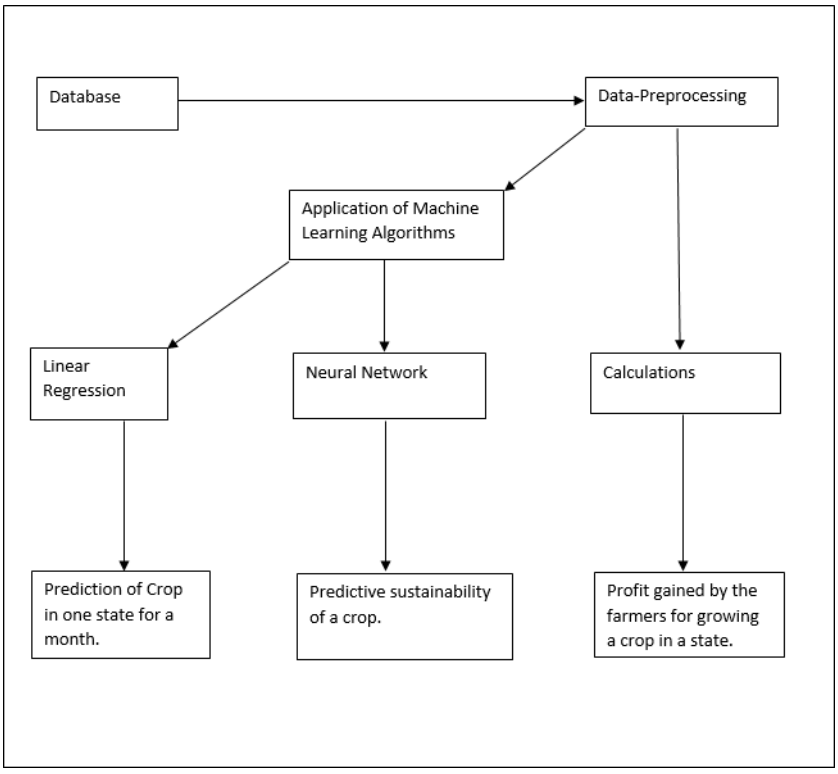


Figure 6.2 System Architecture

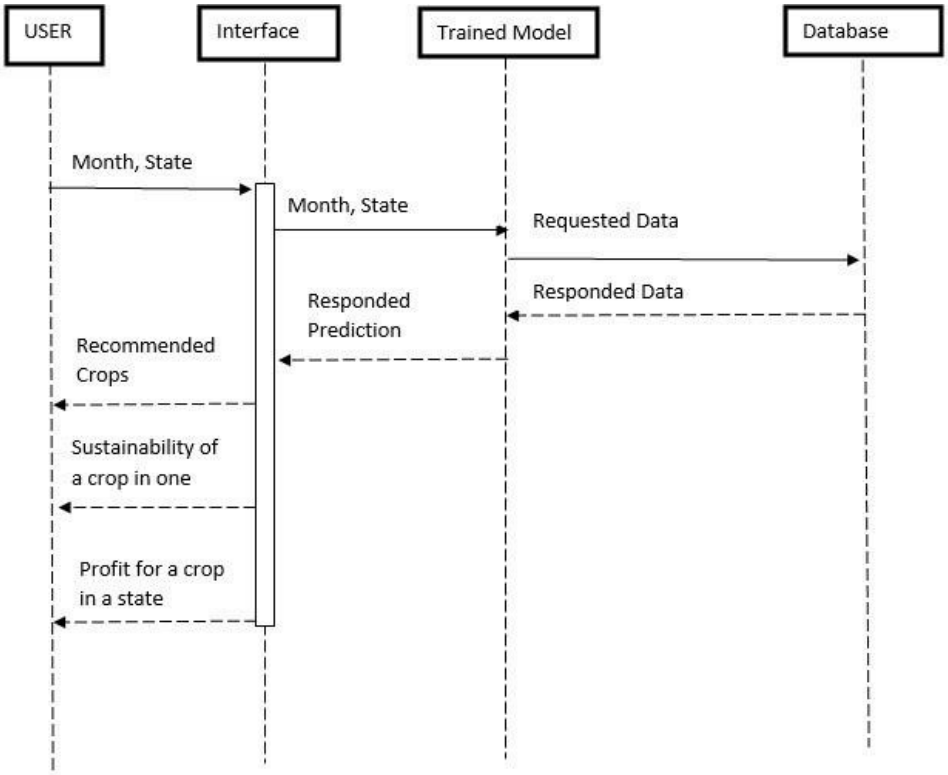
A system architecture is a conceptual model using which we can define the structure and behaviour of that system. It is a formal representation of a system. Depending on the context, system architecture can be used to refer to either a model to describe the system or a method used to build the system. Building a proper system architecture helps in analysis of the project, especially in the early stages depicts the system architecture and is explained in the following section.

4. Sequence diagram

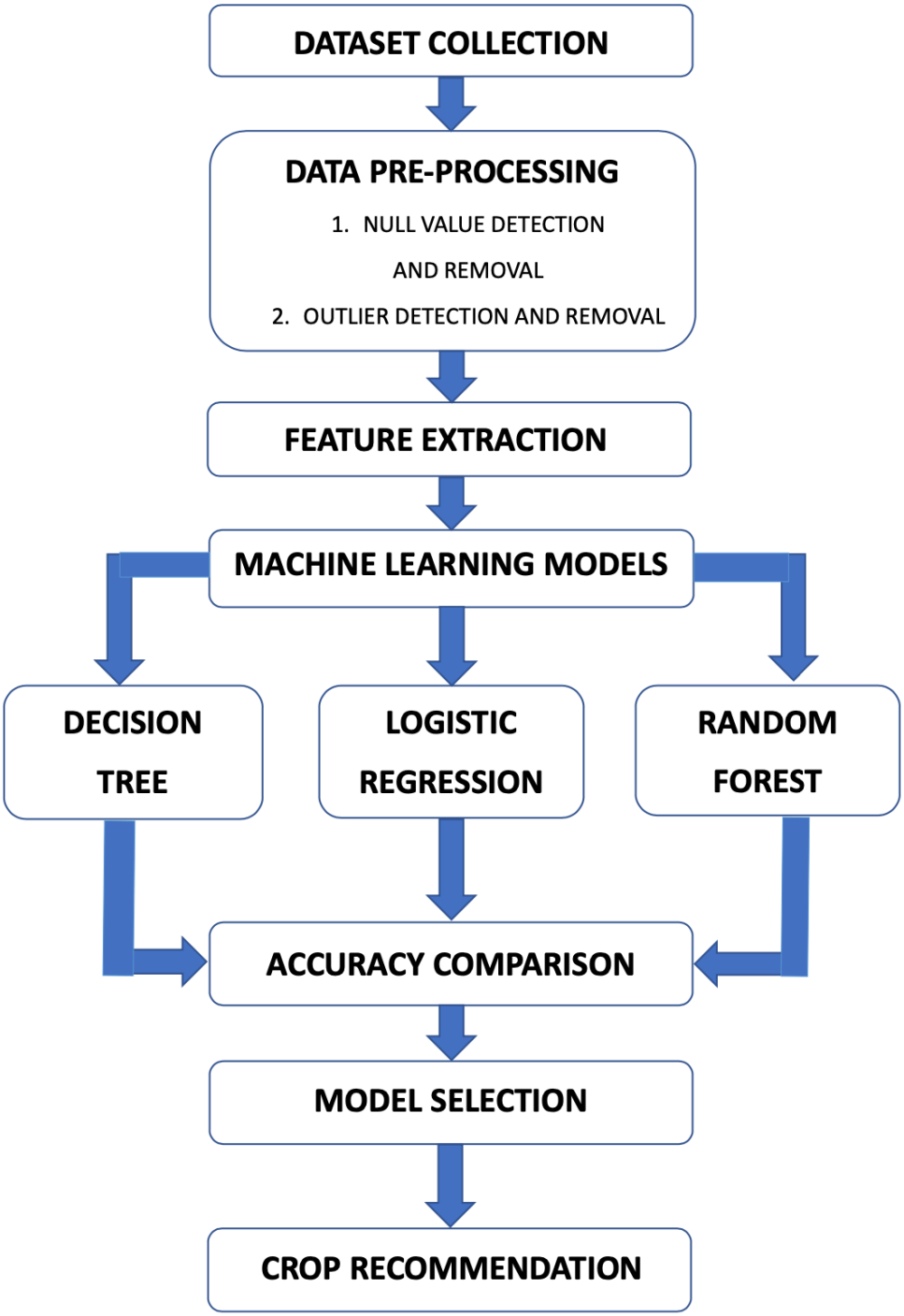
A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions

arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realisations in the Log- ical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios. Sequence diagram are an easy and intuitive way of describing the behaviour of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram has two dimensions: vertical dimension represents time, the horizontal dimension represents the objects existence during the interaction.

Figure 6.3 Sequence Diagram



Chapter 7 : Implementation



1. Dataset Collection

One of the first steps we perform during implementation is an analysis of the data. This was done by us in an attempt to find the presence of any relationships between the various attributes present in the dataset.

Acquisition of Training Dataset: The accuracy of any machine learning algorithm depends on the number of parameters and the correctness of the training dataset. We In this project analysed multiple datasets collected from Government website -https://data.gov.in/ and Kaggle and carefully selected the parameters that would give the best results.Many work done in this field have considered environmental parameters to predict crop sustainability some have used yield as major factor where as in some works only economic factors are taken into consideration. We have tried to combine both environmental parameters like rainfall , temperature ,ph, nutrients in soil, soil type, location and economic parameters like production, and yield to provide accurate and reliable recommendation to the farmer on which crop will be most suit- able for his land.

```
## import packages

import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn import tree
from plotly.subplots import make_subplots
import plotly.graph_objects as go
import plotly.express as px

# to ignore warning

import warnings
warnings.filterwarnings('ignore')
```

```
## import data

crop = pd.read_csv("Crop_recommendation.csv")
crop.head()
```

	N	P	K	temperature	humidity	ph	rainfall	label
0	90	42	43	20.879744	82.002744	6.502985	202.935536	rice
1	85	58	41	21.770462	80.319644	7.038096	226.655537	rice
2	60	55	44	23.004459	82.320763	7.840207	263.964248	rice
3	74	35	40	26.491096	80.158363	6.980401	242.864034	rice
4	78	42	42	20.130175	81.604873	7.628473	262.717340	rice

Figure 7.1 Data Acquisition

```
#copying original data

data=crop.copy()
data
```

	N	P	K	temperature	humidity	ph	rainfall	label
0	90	42	43	20.879744	82.002744	6.502985	202.935536	rice
1	85	58	41	21.770462	80.319644	7.038096	226.655537	rice
2	60	55	44	23.004459	82.320763	7.840207	263.964248	rice
3	74	35	40	26.491096	80.158363	6.980401	242.864034	rice
4	78	42	42	20.130175	81.604873	7.628473	262.717340	rice
...	...	...	...	...	...	...	...	...
2195	107	34	32	26.774637	66.413269	6.780064	177.774507	coffee
2196	99	15	27	27.417112	56.636362	6.086922	127.924610	coffee
2197	118	33	30	24.131797	67.225123	6.362608	173.322839	coffee
2198	117	32	34	26.272418	52.127394	6.758793	127.175293	coffee
2199	104	18	30	23.603016	60.396475	6.779833	140.937041	coffee

2200 rows × 8 columns

Figure 7.2 DataSet

2. Data Preprocessing

After analysing and visualising the data, the next step is preprocessing. Data preprocessing is an important step as it helps in cleaning the data and making it suitable for use in machine learning algorithms. Most of the focus in preprocessing is to remove any outliers or erroneous data, as well as handling any missing values.

Data preprocessing is an essential step in machine learning that involves transforming raw data into a format suitable for training and modelling. Here are some common data preprocessing steps in machine learning:

- **Data Cleaning**: This step involves handling missing data, outliers, and noise in the dataset. Missing data can be imputed using various techniques such as mean imputation, median imputation, or interpolation. Outliers and noisy data points can be detected and treated by methods like trimming, winsorizing, or replacing with appropriate values.

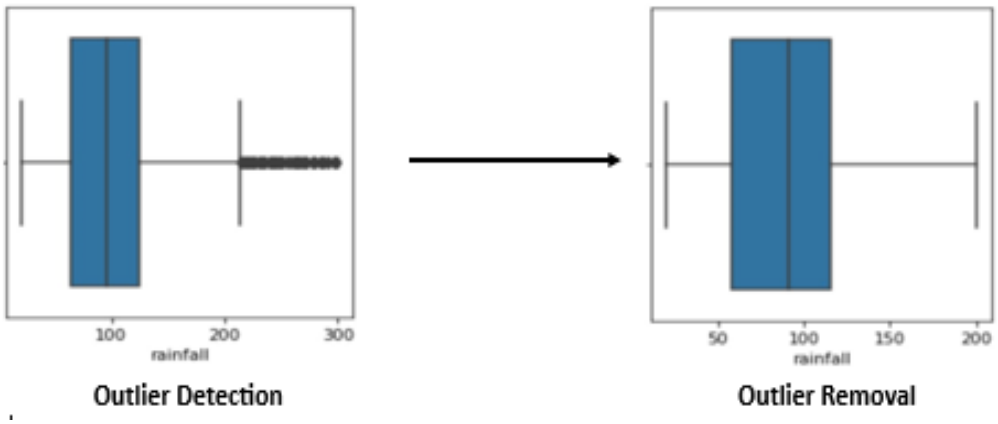


Fig.7.3 : Outlier detection and removal

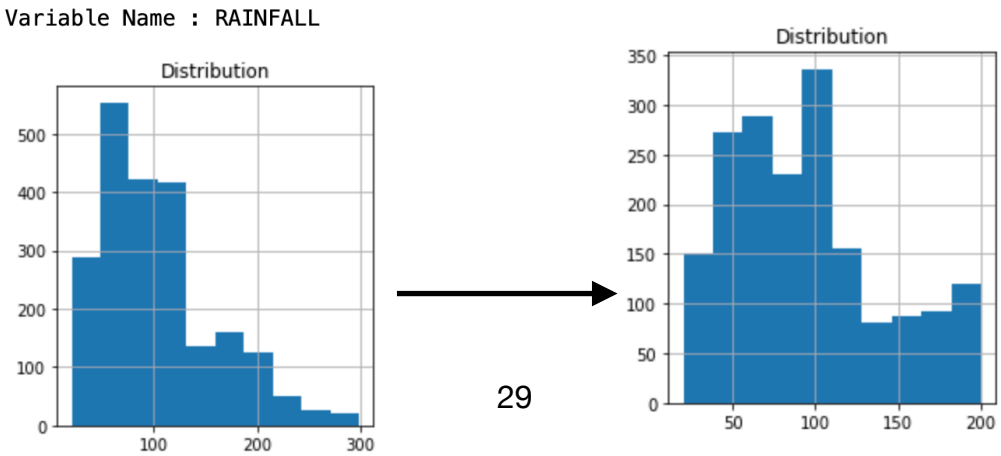
```
## lets check null values

data.isnull().sum()
```

```
N              0
P              0
K              0
temperature    0
humidity       0
ph             0
rainfall       0
label          0
dtype: int64
```

Fig.7.4 : Null values detection and removal

- **Data Transformation**: Data transformation aims to normalize the data distribution or make it conform to certain assumptions of the machine learning algorithms. Common transformations include scaling features to a specific range (e.g., normalization or standardization), logarithmic or power transformations, and handling skewed distributions.

Fig.7.5 : Normalization



- **Multivariate Analysis:** Multivariate analysis is a statistical technique used to analyse relationships among multiple variables simultaneously. It involves studying the patterns, dependencies, and interactions between several variables to gain a deeper understanding of complex data sets.

```
#which crops can grow at higher temperature .i.e., temperature > 30
x = pd.DataFrame(pd.crosstab(data.label[data.temperature > 30],'count',normalize=True)*100)
x.plot.pie(y = 'count',autopct='%1.1f%%',figsize=(8,8),legend=None,shadow=True, startangle=90)
plt.title('Probability of crops grow when temperature > 30')
plt.show()
```

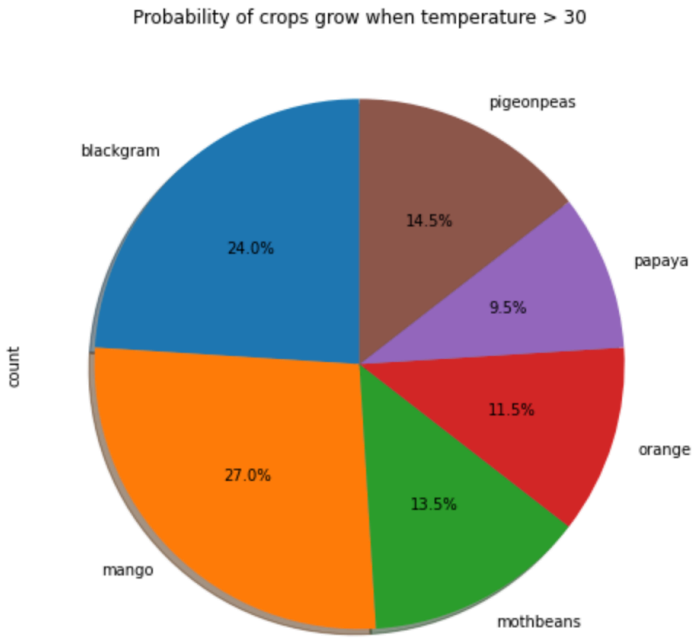


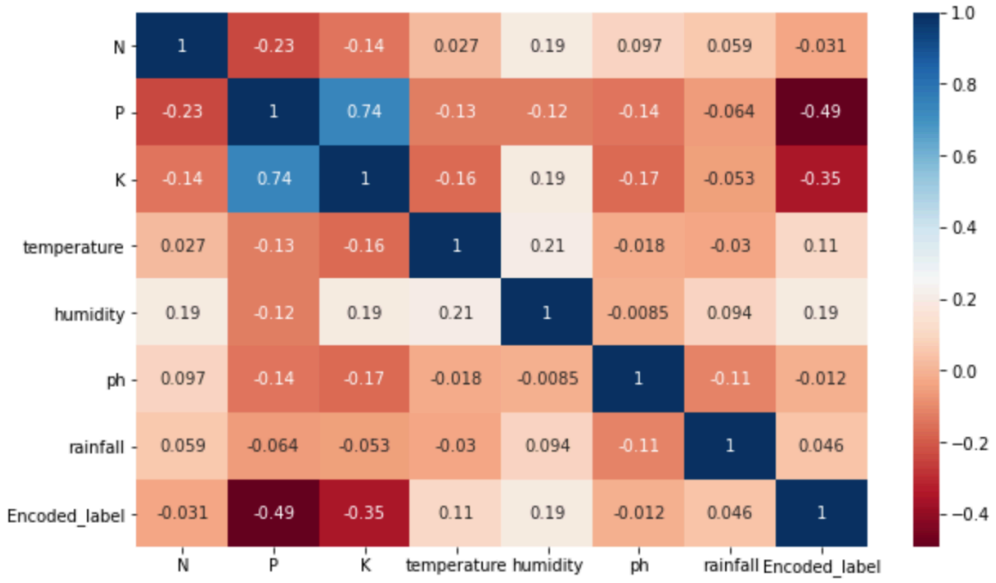
Fig.7.6 : Multivariate Analysis

- **Feature Selection/Extraction**: In this step, relevant features are selected from the dataset or extracted from existing features to reduce dimensionality and improve model performance. Feature selection techniques include methods like correlation analysis, backward/forward feature selection, or using domain knowledge. Feature extraction techniques such as Principal Component Analysis (PCA) or t-SNE can be used to create new meaningful features.

Fig.7.7 : Heatmap depicting Correlation

```
#checking the corelation
plt.figure(figsize=(10,6))
sns.heatmap(data.corr(),annot=True,cmap='RdBu')
```

```
<AxesSubplot:>
```




```
corr = data.corr() # Saving Correlation of Dataset in corr variable.
corr # Displaying Correlation
```

	N	P	K	temperature	humidity	ph	rainfall	Encoded_label
N	1.000000	-0.231460	-0.140512	0.026504	0.190688	0.096683	0.059020	-0.031130
P	-0.231460	1.000000	0.736232	-0.127541	-0.118734	-0.138019	-0.063839	-0.491006
K	-0.140512	0.736232	1.000000	-0.160387	0.190859	-0.169503	-0.053461	-0.346417
temperature	0.026504	-0.127541	-0.160387	1.000000	0.205320	-0.017795	-0.030084	0.113606
humidity	0.190688	-0.118734	0.190859	0.205320	1.000000	-0.008483	0.094423	0.193911
ph	0.096683	-0.138019	-0.169503	-0.017795	-0.008483	1.000000	-0.109069	-0.012253
rainfall	0.059020	-0.063839	-0.053461	-0.030084	0.094423	-0.109069	1.000000	0.045611
Encoded_label	-0.031130	-0.491006	-0.346417	0.113606	0.193911	-0.012253	0.045611	1.000000

Fig.7.8 : Correlation

- **Data Encoding**: Categorical variables need to be encoded into numerical representations as machine learning models generally operate on numerical data. Common encoding techniques include one-hot encoding, label encoding, or ordinal encoding. This step ensures that categorical variables are in a format that can be processed by machine learning algorithms.

label	encoded
apple	0
banana	1
blackgram	2
chickpea	3
coconut	4
coffee	5
cotton	6
grapes	7
jute	8
kidneybeans	9
lentil	10
maize	11
mango	12
mothbeans	13
mungbean	14
muskmelon	15
orange	16
papaya	17
pigeonpeas	18
pomegranate	19

Fig.7.9 : Encoding of Output

- **Data Splitting**: The dataset is typically split into training, validation, and testing sets. The training set is used to train the model, the validation set is used for hyper parameter tuning and model selection, and the testing set is used to evaluate the final model's performance. It is important to ensure that the data splitting is done randomly and maintains the same distribution across different subsets.

```
: # Splitting into train and test data

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2,random_state=2)
print('Shape of Splitting :')
print('x_train = {}, x_test = {}, y_train = {}, y_test = {}'.format(x_train.shape,x_test.shape,y_train.shape,y_test.

Shape of Splitting :
x_train = (1760, 7), x_test = (440, 7), y_train = (1760,), y_test = (440,)
```

Fig.7.10 : Splitting into train and test data

```
#Splitting the data into input and output
x = crop.iloc[:,:-2]
y = crop.Encoded_label
print('Input variables \n',x.head())
print('\nOutput Variable\n',y.head())
```

```
Input variables
     N   P   K  temperature   humidity        ph    rainfall
0  90  42  43    20.879744  82.002744  6.502985  202.935536
1  85  58  41    21.770462  80.319644  7.038096  226.655537
2  60  55  44    23.004459  82.320763  7.840207  263.964248
3  74  35  40    26.491096  80.158363  6.980401  242.864034
4  78  42  42    20.130175  81.604873  7.628473  262.717340

Output Variable
 0    20
1    20
2    20
3    20
4    20
Name: Encoded_label, dtype: int64
```

Fig.7.11 : Splitting of Dataset

3. Machine Learning

Machine learning plays a crucial role in crop recommendation systems by leveraging historical data, environmental factors, and crop-specific information to provide tailored recommendations to farmers. Here's an overview of how machine learning is used in crop recommendation systems:

- **Data Collection**: Relevant data is collected, including historical crop yields, weather patterns, soil characteristics, and agronomic practices. This data serves as the foundation for training machine learning models

- **Feature Engineering**: Data preprocessing techniques are applied to convert raw data into meaningful features. For example, soil nutrient levels may be categorized as low, medium, or high, or weather data may be aggregated into monthly or seasonal averages.

- **Model Training**: Various machine learning algorithms, such as decision trees, random forests, support vector machines (SVM), or neural networks, are trained using the collected and preprocessed data. The models learn patterns and relationships between input features (e.g., soil properties, climate conditions) and target variables (e.g., crop yields, disease outbreaks).

- **Crop Yield Prediction**: Once trained, the models can predict crop yields based on input features. By considering factors like soil quality, climate conditions, and historical yields, the models can estimate the potential yield for different crops in a specific location.

- **Crop Recommendation**: The crop recommendation component utilizes the trained models to suggest suitable crops for specific conditions. By inputting factors such as soil characteristics, weather patterns, and the farmer's goals, the system can generate a list of crops that are likely to perform well in a given area.

- **Feedback Loop and Improvement**: As the crop recommendation system is used and feedback is collected from farmers, the models can be refined and improved. New data can be incorporated, allowing the models to adapt and provide more accurate recommendations over time.

It's important to note that crop recommendation systems are complex and require domain expertise in agronomy, soil science, and climatology. Machine learning models serve as a valuable tool to analyse and process large amounts of data, enabling more informed decision-making for farmers and optimizing agricultural practices.

3.1. Logistic Regression Model

Logistic regression is a machine learning algorithm often used in classification problems. This algorithm is used to predict binary value based on one or more inputs.In logistic regression, the output is a probability estimate between 0 and 1 that represents the likelihood of the binary outcome. This probability estimate is then mapped to a discrete class label using a threshold value, usually 0.5. The logistic regression model is based on the logistic function which is a sigmoid-shaped curve that transforms any input value into a probability value between 0 and The logistic function is defined as:

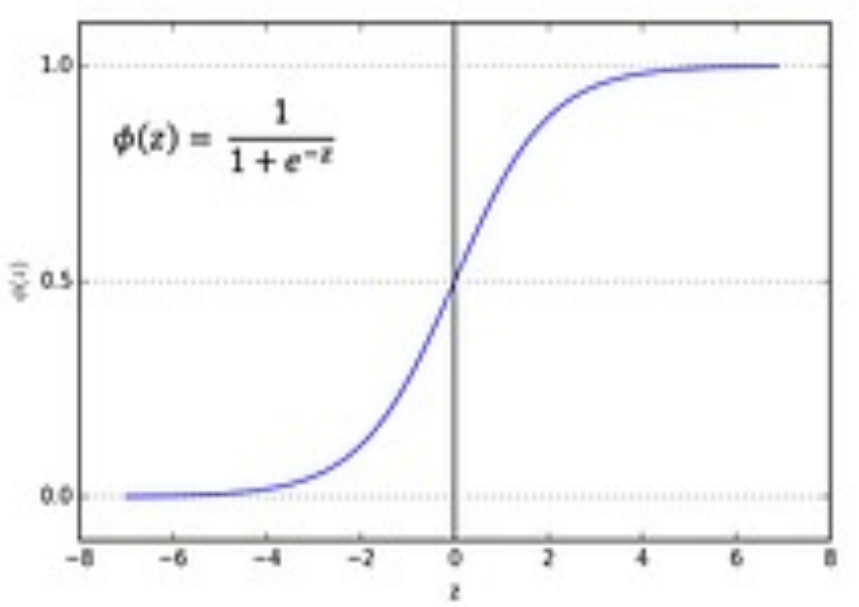


Fig.7.12 : Logistic Function

```
#classification report

print('REPORT : \n',classification_report(y_test,pred_logis))
acc_logis = accuracy_score(y_test,pred_logis)
```

```
REPORT :
                precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        17
           2       0.86      0.75      0.80        16
           3       1.00      1.00      1.00        21
           4       1.00      1.00      1.00        21
           5       1.00      1.00      1.00        22
           6       0.86      0.90      0.88        20
           7       1.00      1.00      1.00        18
           8       0.84      0.93      0.88        28
           9       1.00      1.00      1.00        14
          10       0.88      1.00      0.94        23
          11       0.90      0.86      0.88        21
          12       0.96      1.00      0.98        26
          13       0.84      0.84      0.84        19
          14       1.00      0.96      0.98        24
          15       1.00      1.00      1.00        23
          16       1.00      1.00      1.00        29
          17       1.00      0.95      0.97        19
          18       1.00      1.00      1.00        18
          19       1.00      1.00      1.00        17
          20       0.85      0.69      0.76        16
          21       1.00      1.00      1.00        15

    accuracy                           0.95       440
   macro avg       0.95      0.95      0.95       440
weighted avg       0.95      0.95      0.95       440
```

Fig.7.13 : Classification Report of Logistic Regression

3.2. Decision Tree Model

Decision tree is a popular machine learning algorithm used for both classification and regression tasks. It is a tree-like structure where each node represents a feature or attribute, each branch represents a decision rule or condition, and each leaf node represents a class label or regression output.

The decision tree algorithm can handle both categorical and numerical features. For categorical features, the algorithm uses a one-hot encoding or binary split to represent each category as a separate feature. For numerical features, the algorithm uses a threshold or split point to separate the dataset into two subsets.

Decision trees have several advantages, including their interpretability, as the tree structure can be easily visualised and understood by non-experts. They are also relatively fast to train and can handle large datasets. In summary, decision tree is a powerful and flexible machine learning algorithm that can be used for a variety of tasks.(as described in Fig. 5) It is particularly useful for classification and regression tasks.

We have applied Decision tree approach in our model as:

(I)Importing library DecisionTreeClassifier from sklearn.tree Class

(II) Now we create Decision Tree Classifier object

(III) In the last we fit our data

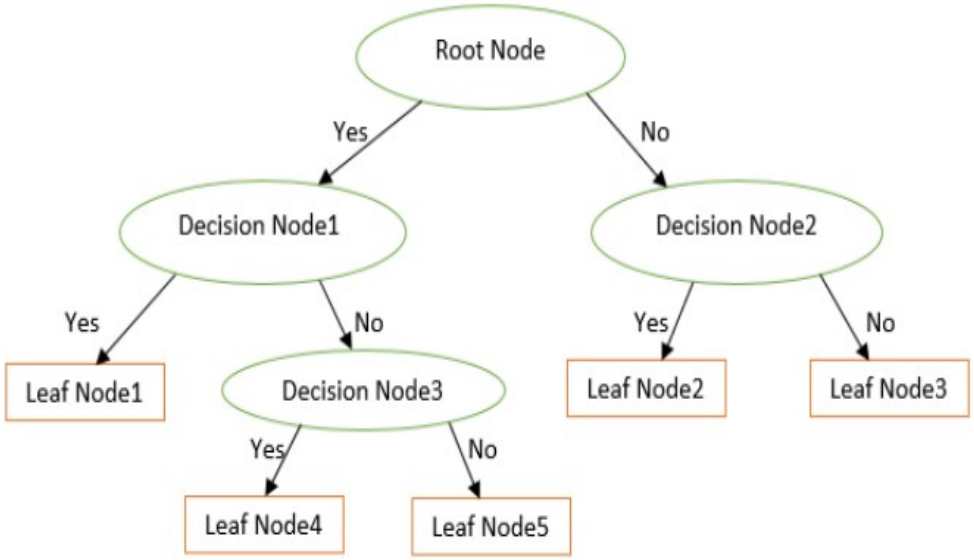


Fig.7.14 :Decision Tree

```
#classification report

print('REPORT : \n',classification_report(y_test,pred_d_tree))
acc_d_tree = accuracy_score(y_test,pred_d_tree)
```

```
REPORT :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        17
           2       0.88      0.94      0.91        16
           3       1.00      1.00      1.00        21
           4       1.00      1.00      1.00        21
           5       1.00      1.00      1.00        22
           6       1.00      1.00      1.00        20
           7       1.00      1.00      1.00        18
           8       0.90      1.00      0.95        28
           9       1.00      1.00      1.00        14
          10       0.95      0.87      0.91        23
          11       1.00      1.00      1.00        21
          12       1.00      1.00      1.00        26
          13       0.90      0.95      0.92        19
          14       1.00      1.00      1.00        24
          15       1.00      1.00      1.00        23
          16       1.00      1.00      1.00        29
          17       1.00      1.00      1.00        19
          18       1.00      1.00      1.00        18
          19       1.00      1.00      1.00        17
          20       1.00      0.81      0.90        16
          21       1.00      1.00      1.00        15

    accuracy                           0.98       440
   macro avg       0.98      0.98      0.98       440
weighted avg       0.98      0.98      0.98       440
```

Fig.7.15 : Classification Report of Decision Tree

3.3. Random Forest Model

Random forest is a popular machine learning algorithm used for both classification and regression problems. It is an ensemble method that combines multiple decision trees to improve the accuracy and reduce the overfitting of the model.

The random forest algorithm(Fig.7) works by creating a multitude of decision trees, each trained on a random subset of the training data and a random subset of the input features. The subsets are chosen randomly with replacement, a technique known as bootstrap aggregating or bagging.

Once the decision trees are trained, the algorithm combines their predictions using a voting or averaging scheme. For classification problems, the class with the most votes are chosen as the final prediction, while for regression problems, the average of the predictions is taken as the final output.

Random forest has several advantages over other algorithms, including its ability to handle high-dimensional and noisy data, as well as its robustness to outliers and missing values. It also provides a measure of feature importance, which can help in feature selection and understanding the underlying data.

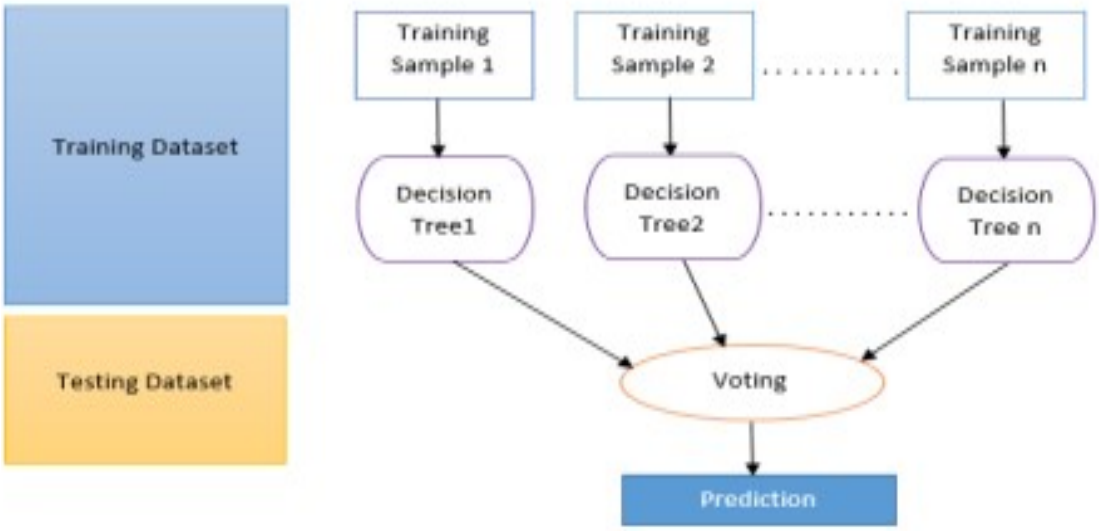


Fig.7.16 : Random Forest

```
: #classification report
  print('REPORT : \n',classification_report(y_test,pred_rand))
```

```
REPORT :
                precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        17
           2       1.00      1.00      1.00        16
           3       1.00      1.00      1.00        21
           4       1.00      1.00      1.00        21
           5       1.00      1.00      1.00        22
           6       1.00      1.00      1.00        20
           7       1.00      1.00      1.00        18
           8       0.90      1.00      0.95        28
           9       1.00      1.00      1.00        14
          10       1.00      0.96      0.98        23
          11       1.00      1.00      1.00        21
          12       1.00      1.00      1.00        26
          13       0.95      1.00      0.97        19
          14       1.00      1.00      1.00        24
          15       1.00      1.00      1.00        23
          16       1.00      1.00      1.00        29
          17       1.00      1.00      1.00        19
          18       1.00      1.00      1.00        18
          19       1.00      1.00      1.00        17
          20       1.00      0.81      0.90        16
          21       1.00      1.00      1.00        15

    accuracy                           0.99       440
   macro avg       0.99      0.99      0.99       440
weighted avg       0.99      0.99      0.99       440
```

Fig.7.17 : Classification Report of Random Forest

Chapter 8 : Testing

1. Testing Methodologies

The program comprises of several algorithms which are tested individually for the accuracy. we check for the correctness of the program as a whole and how it performs.

2. Unit Testing

Unit tests focus on ensuring that the correct changes to the world- state take place when a transaction is processed. The business logic in transaction processor functions should have unit tests, ideally with 100 percent code coverage. This will ensure that you do not have ty- pos or logic errors in the business logic. The various modules can be individually run from a command line and tested for correctness. The tester can pass various values, to check the answer returned and verify it with the values given to him/her. The other work around is to write a script, and run all the tests using it and write the output to a log file and using that to verify the results. We tested each of the algorithms individually and made changes in preprocessing accordingly to increase the accuracy.

3. System Testing

System Testing is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the systems compliance with the specified requirements. System Testing is the testing of a complete and fully integrated software product. and White Box Testing. System test falls under the black box testing cate- gory of software testing. Different Types of System Testing:

- Usability Testing - Usability Testing mainly focuses on the users ease to use the application, flexibility in handling controls and ability of the system to meet its objectives.

- Load Testing - Load Testing is necessary to know that a software solution will perform under real-life loads.

- Regression Testing- - Regression Testing involves testing done to make sure none of the changes made over the course of the development process have caused new bugs.

- Recovery Testing - Recovery testing is done to demonstrate a soft- ware solution is reliable, trustworthy and can successfully recoup from possible crashes.

- Migration Testing - Migration testing is done to ensure that the software can be moved from older system infrastructures to current system infrastructures without any issues.

4. Quality Assurance

Quality Assurance is popularly known as QA Testing, is defined as an activity to ensure that an organization is providing the best possible product or service to customers. QA focuses on improving the processes to deliver Quality Products to the customer. An organization has to ensure, that processes are efficient and effective as per the quality standards defined for software products.

5. Functional Test

Functional Testing is also known as functional completeness testing, Functional Test- ing involves trying to think of any possible missing functions. As chat-bot evolves into new application areas, functional testing of essential chatbot components. Functional testing evaluates use-case scenarios and related business processes, such as the behaviour of smart contracts.

Chapter 9 : Results and Performance Analysis

To evaluate the performance of the crop recommendation system, we used several metrics, including precision, recall, and F1 score. The accuracy metric measures the proportion of correctly classified instances. The precision metric measures the proportion of true positive instances among all predicted positive instances. The recall metric measures the proportion of true positive instances among all actual positive instances. The F1 score is the harmonic mean of precision and recall.

MACHINE LEARNING ALGORITHMS	**ACCURACY**	**PRECISION**	**RECALL**	**F1 SCORE**
DECISION TREE	0.98	0.93	0.93	0.93
LOGISTIC REGRESSION	0.95	0.92	0.92	0.92
RANDOM FOREST	0.99	1.00	1.00	1.00

Fig.9.1: Comparison Table

9.1 Result

The crop recommendation system was evaluated using real-world data collected from farms in different regions. The system achieved an accuracy of over 90%, which is a significant improvement over traditional methods of crop recommendation.The system's performance was also compared to other machine learning algorithms and the Random Forest algorithm was found to be the most accurate.

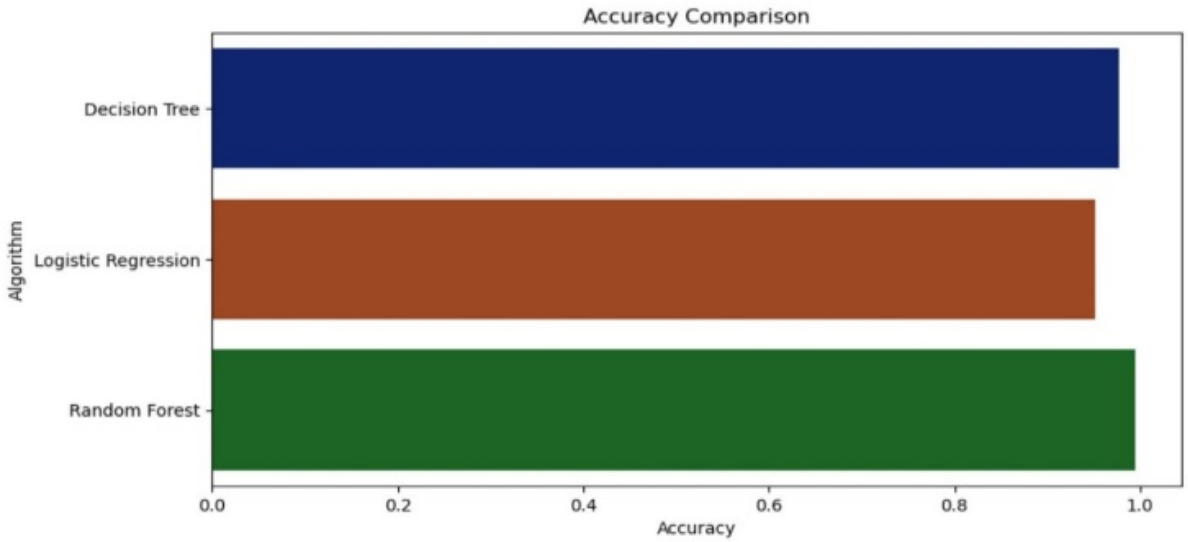


Fig.9.2: Algorithm accuracy representation

9.2 Crop Recommendation

The crops are recommended using the machine learning model based on the Random Forest Model and also the users have the option to choose the number of crops he has to predict so that he can select from them the better ones according to his/her needs and requirements. The model predicts the crops on the priority basis in the percentage format in the form of Pie Chart.

```
high = pre.predicted_values.nlargest(6)
plt.figure(figsize=(15,10))
plt.rcParams['font.size']=15
plt.title('Crops Recommendations :',fontdict={'fontsize': 25, 'fontweight': 'medium'})
plt.pie(x=high,labels=high.index,autopct='%1.1f%%',explode=(0.1, 0, 0, 0, 0,0),shadow=True,startangle=90,
        colors=['green','red','cyan','brown','orange','yellow'])

plt.show()
```

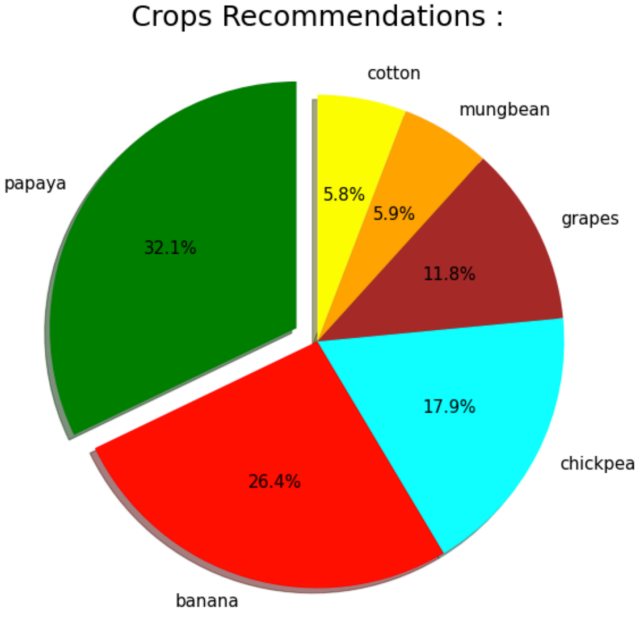


Fig.9.3 : Crop Prediction in the form of Pie Chart

9.3 Snapshots of Website

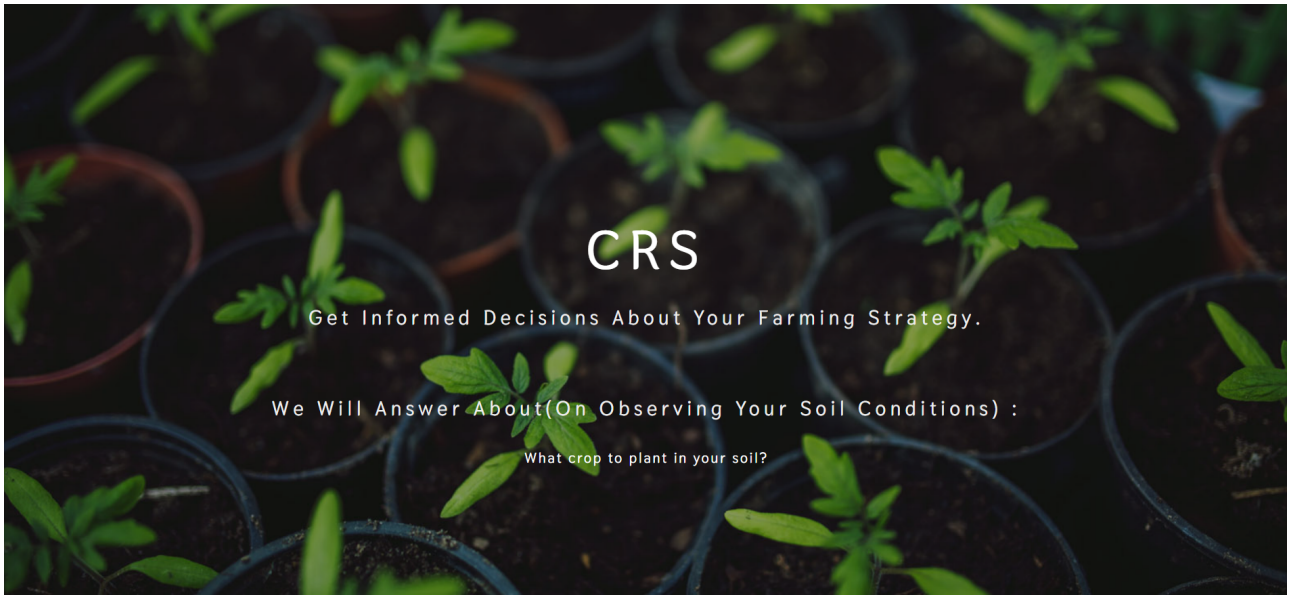


Fig.9.4 : Introduction Page

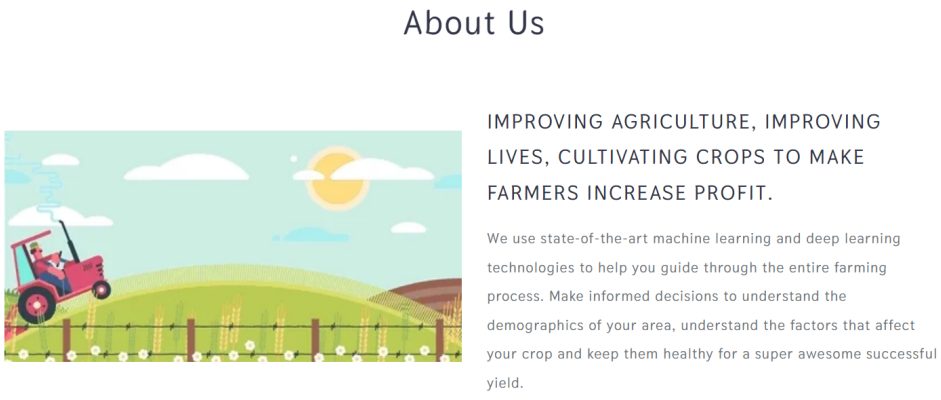


Fig.9.5 : About Us Page

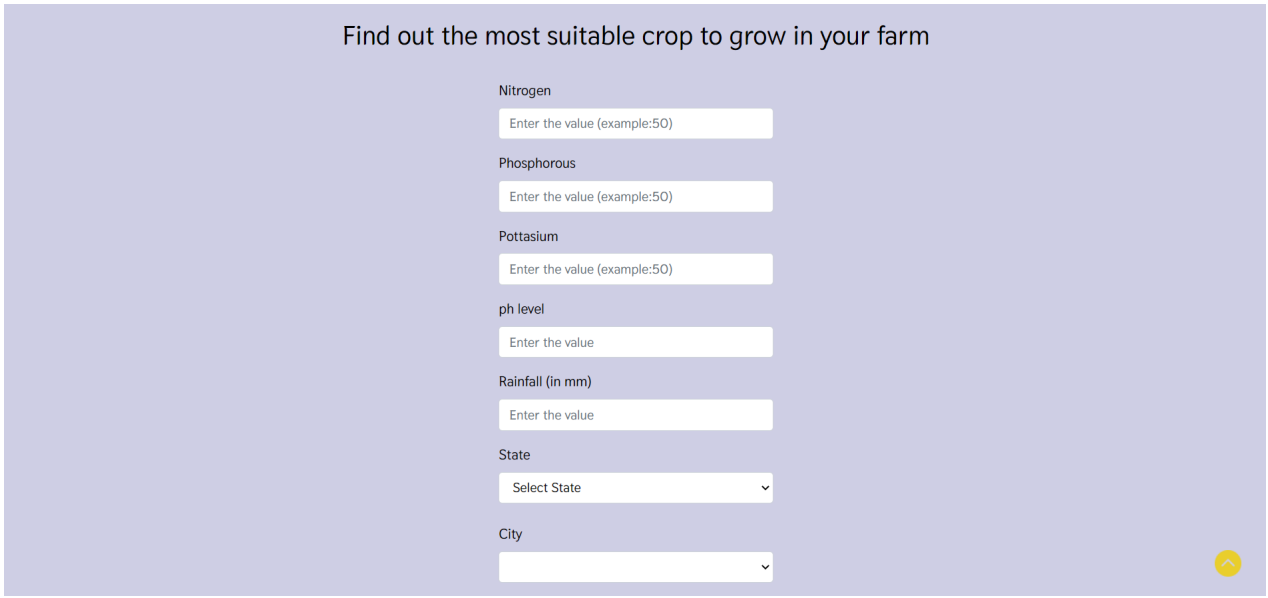


Fig.9.6 : User Input Page

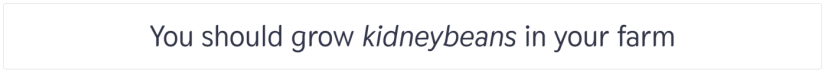


Fig.9.7 : Crop Prediction

Chapter 10 : Conclusion

This system helps the farmer to choose the right crop by providing insights that ordinary farmers don’t keep track of thereby decreasing the chances of crop failure and increasing productivity. It also prevents them form incurring losses. The system can be extended to the web and can be accessed by millions of farmers across the country.We could achieve an accuracy of 89.88 percent from the neural network and an accuracy of 88.26 percent from the linear regression model

Further development is to integrate the crop recommendation system with another subsystem, yield predictor that would also provide the farmer an estimate of production if he plants the recommended crop.

References

1. Rakesh Kumar , M.P. Singh, Prabhat Kumar and J.P. Singh, “Crop Selection Method to Maximize Crop Yield Rate using Machine Learning Technique”, International Conference on Smart Technologies and Management for Computing, Communication, Controls,
   Energy and Materials, 2015

2. Haedong Lee and Aekyung Moon, “Development of Yield Predic- tion System Based on Real-time Agricultural Meteorological Information”, 16th International Conference on Advanced Communication Technology, 2014

3. T.R. Lekhaa, “Efficient Crop Yield and Pesticide Prediction for Improving Agricultural Economy using Data Mining Techniques”, International Journal of Modern Trends in Engineering and Science (IJMTES), 2016, Volume 03, Issue 10

4. Jay Gholap, Anurag Ingole, Jayesh Gohil, Shailesh Gargade and Vahida Attar, “Soil Data Analysis Using Classification Techniques and Soil Attribute Prediction”, International Journal of Computer Science Issues, Volume 9, Issue 3

5. Anshal Savla, Parul Dhawan, Himtanaya Bhadada, Nivedita Is- rani, Alisha Mandholia, Sanya Bhardwaj (2015), ‘Survey of classification algorithms for formulating yield prediction accuracy in precision agriculture’, Innovations in Information, Embedded and Communication systems (ICIIECS).

6. AgroConsultant: Intelligent Crop Recommendation System Using Machine Learning Algorithms. Zeel Doshi , Subhash Nadkarni , Rashi Agrawal, Prof. Neepa Shah.

7. Crop Recommendation System for Precision Agriculture S.Pudumalar*, E.Ramanujam*, R.Harine Rajashree, C.Kavya, T.Kiruthika, J.Nisha.

8. Tom M. Mitchell, Machine Learning, India Edition 2013, McGraw Hill Education.

9. https://data.gov.in/g

10. Kagglehttps://www.kaggle.com/notebook

Appendix

1. Performance Metrics

The end users of prediction tools should be able to understand how evaluation is done and how to interpret the results. Six main performance evaluation measures are introduced. These include :

1.1 Cross Validation Score

Cross-validation may be a statistical procedure that is used to estimate the skill of machine learning models. It is commonly utilized in applied machine learning to match and choose a model for a given predictive modelling problem because it is easy to know, easy to implement, and leads to skill estimates that generally have a lower bias than other methods. It is also known as a resampling procedure used to evaluate machine learning models on a limited data sample. Cross-validation gives a more accurate measure of model quality, which is especially important if you are making a lot of modeling decisions. Sometimes it takes longer to run because it estimates multiple models. It is a popular method because it is simple to understand and it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split. Cross-validation results of Random forest and Decision Tree Regression can be seen . The number of splits are 5 and test size is 0.2 which means 20 records out of every 100 records are taken for the test.

Cross Validation for RandomForest Regression:

```
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestRegressor

cv = ShuffleSplit(n_splits=5, test_size=0.2)

cross_val_score(RandomForestRegressor(n_estimators = 10), X, y, cv=cv)
```

```
array([0.89842273, 0.83105375, 0.96403624, 0.82111085, 0.98910053])
```

Cross Validation for DecisionTree Regression:

```
[ ] from sklearn.model_selection import ShuffleSplit
    from sklearn.model_selection import cross_val_score
    from sklearn.tree import DecisionTreeRegressor

    cv = ShuffleSplit(n_splits=5, test_size=0.2)

    cross_val_score( DecisionTreeRegressor(random_state=0), X, y, cv=cv)
```

```
array([0.85325866, 0.89433854, 0.90529817, 0.89356039, 0.76169771])
```

1.2 Confusion Matrix

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix. Some features of Confusion matrix are given below:

- For the 2 prediction classes of classifiers, the matrix is of 2*2 table, for 3 classes, it is 3*3 table, and so on.
- The matrix is divided into two dimensions, that are predicted values and actual values along with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.
- It looks like the below table:

n = total predictions	Actual: No	Actual: Yes
Predicted: No	True Negative	False Positive
Predicted: Yes	False Negative	True Positive

The above table has the following cases:

- **True Negative**: Model has given prediction No, and the real or actual value was also No.
- **True Positive:** The model has predicted yes, and the actual value was also true.
- **False Negative**: The model has predicted no, but the actual value was Yes, it is also called as Type-II error.
- **False Positive**: The model has predicted Yes, but the actual value was No. It is also called a Type-I error.

1.3 Classification Report

- **Accuracy** :

Accuracy is a classification problem metric that indicates the percentage of correct predictions. We compute it by dividing the total number of predictions by the number of correct predictions. This formula provides a simple definition based on a binary classification problem. In the case of binary classification, accuracy can be expressed as True/False Positive/Negative values.

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ number\ of\ predictions}$$

Fig : Accuracy Formula

When to Use Accuracy?

It is good to use the Accuracy metric when the target variable classes in data are approximately balanced. For example, if 60% of classes in a fruit image dataset are of Apple, 40% are Mango. In this case, if the model is asked to predict whether the image is of Apple or Mango, it will give a prediction with 97% of accuracy.

- **Precision** :

Precision is defined as the fraction of positive examples that are actually positive among all positive examples predicted by us. It can also be defined as the number of true positives divided by the total number of true positives plus false positives. False positives occur when the model incorrectly labels something as positive when it is actually negative, or in our case, when the model incorrectly labels someone as a terrorist when they are not.

$$\boldsymbol{Precision = \frac{TP}{(TP + FP)}}$$

Fig : Precision Formula

- **Recall :**

In statistics, the metric our intuition tells us we should maximise is known as recall, or a model's ability to find all relevant cases within a dataset. The number of true positives divided by the number of true positives plus the number of false negatives is the precise definition of recall. True positives are data points classified as positive by the model that are actually positive (meaning they are correct), whereas false negatives are data points

classified as negative by the model that are actually positive (meaning they are correct) (incorrect).

$$\text{Recall} = \frac{TP}{TP+FN}$$

Fig : Recall Formula

When to use Precision and Recall?

From the above definitions of Precision and Recall, we can say that recall determines the performance of a classifier with respect to a false negative, whereas precision gives information about the performance of a classifier with respect to a false positive.

So, if we want to minimize the false negative, then, Recall should be as near to 100%, and if we want to minimize the false positive, then precision should be close to 100% as possible.

In simple words, *if we maximize precision, it will minimize the FP errors, and if we maximize recall, it will minimize the FN error.*

- **F - Score :**

It is traditionally defined as the harmonic mean of precision and recall. It's also known as the F Score or the F Measure. In other words, the F1 score conveys the balance between precision and recall. It is thought to be a better measure than Precision and Recall 30 separately because the trade-off between the two is difficult to achieve.

$$F1 - score = 2 * \frac{precision * recall}{precision + recall}$$

Fig : F - Score Formula

When to use F-Score?

As F-score make use of both precision and recall, so it should be used if both of them are important for evaluation, but one (precision or recall) is slightly more important to consider than the other. For example, when False negatives are comparatively more important than false positives, or vice versa.

2. CODE SNIPPETS :

```
## import packages

import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn import tree
from plotly.subplots import make_subplots
import plotly.graph_objects as go
import plotly.express as px


# to ignore warning

import warnings
warnings.filterwarnings('ignore')

## import data

crop = pd.read_csv("Crop_recommendation.csv")
crop.head()

#copying original data

data=crop.copy()
data

#copying original data

data=crop.copy()
data
```

```
## shape

print("The Shape of the Dataset is: {}" .format(data.shape))
print("The number of the columns in the Dataset is: {}" .format(data.shape[1]))
print("The Shape of the row in the Dataset is: {}" .format(data.shape[0]))

data.columns

## info

data.info()

data.dtypes

#checking for unique values
for i in data.columns:
    print("Column Name : ",i.upper())
    print("No. of unique values : {} \n".format(data[i].nunique()))
    if(data[i].dtype=="object"):
        print("Unique values : ",pd.unique(data[i]))

# type of data

data.dtypes

## count unique values in 'label' columns

data['label'].value_counts()

## lets check null values

data.isnull().sum()
```

```
## descripe

data.describe()

#label encoding for output variable
from sklearn.preprocessing import LabelEncoder
encod = LabelEncoder()
data['Encoded_label'] = encod.fit_transform(data.label) #label will be encoded in alphabetical order
crop['Encoded_label'] = encod.fit_transform(data.label) #label will be encoded in alphabetical order

#encoded labels for classes
a = pd.DataFrame(pd.unique(data.label));
a.rename(columns={0:'label'},inplace=True)
b = pd.DataFrame(pd.unique(data.Encoded_label));
b.rename(columns={0:'encoded'},inplace=True)
classes = pd.concat([a,b],axis=1).sort_values('encoded').set_index('label')
classes

#fetching the label for given encoded value
a=12
for i in range(0,len(classes)):
    if(classes.encoded[i]==a):
        print(classes.index[i].upper())

#dropping duplicate values
data =data.drop_duplicates()

corr = data.corr() # Saving Correlation of Dataset in corr variable.
corr # Displaying Correlation

#checking the corelation
plt.figure(figsize=(10,6))
sns.heatmap(data.corr(),annot=True,cmap='RdBu')
```

```
#checking for outliers in the data
for i in data.columns[:-2]:
    print('Variable Name :',i.upper())
    fig, axes = plt.subplots(1,2,figsize=(8,4))
    axes[0].set_title('Distribution')
    axes[1].set_title('Outliers Detection')
    data[i].hist(ax=axes[0])
    sns.boxplot(data[i],ax=axes[1])
    plt.show()
    print('\n')

# Removal of Outliers

data = data.loc[data["rainfall"]<200]
data = data.loc[data["temperature"]<35]
data = data.loc[data["P"]<105]
for i in data.columns[:-2]:
    print('Variable Name :',i.upper())
    fig, axes = plt.subplots(1,2,figsize=(8,4))
    axes[0].set_title('Distribution')
    axes[1].set_title('Outliers Detection')
    data[i].hist(ax=axes[0])
    sns.boxplot(data[i],ax=axes[1])
    plt.show()
    print('\n')

#which crops can grow at higher temperature .i.e., temperature > 30
x = pd.DataFrame(pd.crosstab(data.label[data.temperature > 30],'count',normalize=True)*100)
x.plot.pie(y = 'count',autopct='%1.1f%%',figsize=(8,8),legend=None,shadow=True, startangle=90)
plt.title('Probability of crops grow when temperature > 30')
plt.show()
```

```
#which crops can grow at higher ph value .i.e., (alkaline nature) ph > 7.5
x = pd.DataFrame(pd.crosstab(data.label[data.ph > 7.5],'count',normalize=True)*100)
x.plot.pie(y = 'count',autopct='%1.1f%%',figsize=(8,8),legend=None,shadow=True, startangle=90)
plt.title('Probability of crops grow when ph > 7.5 i.e.,alkaline nature')
plt.show()

#checking the corelation
plt.figure(figsize=(10,6))
sns.heatmap(data.corr(),annot=True,cmap='RdBu')

#Splitting the data into input and output
x = crop.iloc[:,:-2]
y = crop.Encoded_label
print('Input variables \n',x.head())
print('\nOutput Variable\n',y.head())

# Splitting into train and test data

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2,random_state=2)
print('Shape of Splitting :')
print('x_train = {}, x_test = {}, y_train = {}, y_test = {}'.format(x_train.shape,x_test.shape,y_train.shape,y_test.shape))

#importing necessary libraries

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, accuracy_score, plot_confusion_matrix
from sklearn.model_selection import GridSearchCV
```

```
#Initializing the model

d_tree = DecisionTreeClassifier()
pred_d_tree = d_tree.fit(x_train,y_train).predict(x_test)
print('Confusion Matrix : \n')
fig, ax = plt.subplots(figsize=(5,5))
plot_confusion_matrix(d_tree,x_test,y_test,ax=ax,cmap=plt.cm.Blues)
plt.show()

#classification report

print('REPORT : \n',classification_report(y_test,pred_d_tree))
acc_d_tree = accuracy_score(y_test,pred_d_tree)

x=metrics.accuracy_score(y_test,pred_d_tree)
y=metrics.precision_score(y_test,pred_d_tree,average='weighted')
acc.append(x)
pre.append(y)
model.append("Decision Tree")

#importing necessary libraries

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score, plot_confusion_matrix
from sklearn.model_selection import GridSearchCV

#Initializing the model

logis = LogisticRegression()
pred_logis = logis.fit(x_train,y_train).predict(x_test)
print('Confusion Matrix : \n')
fig, ax = plt.subplots(figsize=(15,15))
plot_confusion_matrix(logis,x_test,y_test,ax=ax,cmap=plt.cm.Blues)
plt.show()
```

```
#classification report

print('REPORT : \n',classification_report(y_test,pred_logis))
acc_logis = accuracy_score(y_test,pred_logis)

x=metrics.accuracy_score(y_test,pred_logis)
acc.append(x)
y=metrics.precision_score(y_test,pred_logis,average='weighted')
pre.append(y)
model.append("Logistic Regression")

#importing necessary libraries
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, plot_confusion_matrix
from sklearn.model_selection import GridSearchCV

#initializing the model and fitting for train data

rand = RandomForestClassifier()
pred_rand = rand.fit(x_train,y_train).predict(x_test)
acc_rand = accuracy_score(y_test,pred_rand)
acc_rand

x=metrics.accuracy_score(y_test,pred_rand)
acc.append(x)
y=metrics.precision_score(y_test,pred_rand,average='weighted')
pre.append(y)
model.append("Random Forest")
```

```
#selecting parameters using GridSearchCV

param = {'n_estimators':range(10,50,10),
         'criterion':['gini'],
         'max_depth':range(1,20,2),
         'max_features':[1,2,3],
         'min_samples_leaf':range(1,10,2),
         'min_samples_split':range(1,40,10)}
grid_rand = GridSearchCV(rand,param,cv=5,n_jobs=-1,verbose=1)
grid_rand.fit(x_train,y_train)

print(grid_rand.best_params_)
print(grid_rand.best_score_)


#predicting for test data and plotting confusion matrix
pred_rand = grid_rand.predict(x_test)
print('Confusion Matrix : \n')
fig, ax = plt.subplots(figsize=(15,15))
plot_confusion_matrix(grid_rand,x_test,y_test,ax=ax,cmap=plt.cm.Blues)
plt.show()

#classification report
print('REPORT : \n',classification_report(y_test,pred_rand))

plt.figure(figsize=[10,5],dpi=100)
plt.title("Accuracy Comparison")
plt.xlabel("Accuracy")
plt.ylabel("Algorithm")
sns.barplot(x=acc,y=model,palette="dark")
```

```
plt.figure(figsize=[5,3],dpi=100)
plt.title("Accuracy Comparison")
plt.xlabel("Algorithm")
plt.ylabel("Precision")
sns.barplot(x=model,y=pre,palette="dark")

#importing pickle file
import pickle
pickle_out = open('classifier.pkl','wb')
pickle.dump(grid_rand,pickle_out)
pickle_out.close()

a = [[180,65,140,30,80,7.5,100]]

pickle_in = open('classifier.pkl','rb')
model = pickle.load(pickle_in)
pre = model.predict_proba(a)
pre = pd.DataFrame(data = np.round(pre.T*100,2), index=classes.index,columns=['predicted_values'])
pre

high = pre.predicted_values.nlargest(6)
plt.figure(figsize=(15,10))
plt.rcParams['font.size']=15
plt.title('Crops Recommendations :',fontdict={'fontsize': 25, 'fontweight': 'medium'})
plt.pie(x=high,labels=high.index,autopct='%1.1f%%',explode=(0.1, 0, 0, 0, 0,0),shadow=True,startangle=90,
        colors=['green','red','cyan','brown','orange','yellow'])

plt.show()
```

ARCHIT GARG

B.Tech. - CSE

Ph: +91-7017417110

Email: archit1910158@akgec.ac.in

Ghaziabad, Uttar Pradesh, India - 201002

BRIEF OVERVIEW / CAREER OBJECTIVE / SUMMARY

Currently pursuing B.tech from Ajay Kumar Garg Engineering College. Looking forward to work with an organization where I could contribute towards the growth of the organization and achieve my personal goals too and also practice problem solving on leetcode with 5 badges and Geeksforgeeks.
Practice questions on Leetcode : https://leetcode.com/chasemaster0308/
Practice questions on GeeksforGeeks : https://auth.geeksforgeeks.org/user/chasemaster0308/profile

KEY EXPERTISE / SKILLS

React.js | Javascript | HTML5 | CSS | DBMS | SQL | CPP | Data Structures | Algorithms

EDUCATION

Ajay Kumar Garg Engineering College 2019 - 2023
B.Tech. - CSE | Percentage: **77.30** / **100.00**

Ram Eesh International School, Greater Noida 2019
12th | CBSE | Percentage: **83.80** / **100.00**

Ram Eesh International School, Greater Noida 2017
10th | CBSE | CGPA: **8.80** / **10.00**

PROJECTS

Policy bazaar clone May 9, 2022 - June 9, 2022
Team Size: 1

Key Skills: ReactJS

Policy bazaar clone- Created a clone web app of policy bazaar with React.js and other libraries.
Working link:-https://policybazaarclone.netlify.app/

Airbnb Clone May 9, 2022 - June 9, 2022

Key Skills: React.js

- Cloned complete Airbnb website with hotel booking and reactive maps also , Tech used was mainly Next.js and other libraries.
Working Link:- https://airbnbclone-drab.vercel.app/

SpaceX website- May 9, 2022 - June 9, 2022
Mentor: self | **Team Size:** 1

Key Skills: HTML5 | CSS | Animation

SpaceX website- Code frontend of a website and added animations too . Tech used was react.js
Working Link:-https://space-xspace.netlify.app/

ASSESSMENTS / CERTIFICATIONS

React.js July 31, 2062
Aggregate: 70.0 / 100.0
Subjects: React.js (70.0 / 100.0)

Key Skills: React.js

A 6-week online training on React. The training consisted of Introduction, Tic Tac Toe Game, Box Office App, Chat Application, Custom Backend, and Assignment & Summary modules.

PERSONAL DETAILS

Gender: Male
Marital Status: Unmarried
Current Address: 1507,Riddhi , Mahagunpuram, Avantika Extension, Ghaziabad, Uttar Pradesh, India - 201002
Email: archit1910158@akgec.ac.in

Date of Birth: Aug. 3, 2001
Permanent Address: 213,prempuri Dankaur, Gautam Budh Nagar, Greater Noida, Uttar Pradesh, India - 203201
Phone Number: +91-7017417110

Muzaffarnagar,India,251001
7668902327
aryanaggarwal144@gmail.com

PROFESSIONAL SUMMARY

Detail-orientated and adept at making critical decisions, managing deadlines and conducting team reviews. With expertise in analysis and quantitative problem-solving skills, dedicated to company growth and improvements.

CORE QUALIFICATIONS

- Programming Languages : Python, C, SQL, HTML, CSS\
- Libraries / Frameworks : Matplotlib, Pandas, Numpy
- Tools / Platforms : Tableau, Jupyter Lab, Sublime Text

Aryan Aggarwal

EXPERIENCE

June 2022 - August 2022
Data Science Intern Internshala | New Delhi, India

- Applied custom models and algorithms to data sets to evaluate and solve diverse company problems.
- Worked alongside team members and leaders to identify analytical requirements and collect information to meet customer and project demands.

September 2021 - October 2021
Intern Trainee Froyo Technologies | Noida, India

- Assessed data to identify trends and stay abreast with innovation, performing best practices.
- Assisted with data storage structures, data mining and data cleansing for archiving and updating.

June 2020 - July 2020
Intern Ajay Kumar Garg Engineering College | Ghaziabad, India

- Deep Analysis of various Data Structures & Algorithms.

EDUCATION

March 2017
Secondary School Certificate
Holy Angels' Convent School , Muzaffarnagar, UP, India

- CGPA: 10

March 2019
Higher Secondary Certificate | Science
S.D. Public School , Muzaffarnagar, UP, India

- Percentage: 90.8%

July 2023
Bachelors | Computer Science And Engineering
Ajay Kumar Garg Engineering College Ghaziabad, India

- Percentage: 81.51%

PROJECTS

- **ML Diabetes Classification**
  In this, model is able to predict whether patient is suffering from diabetes or not.
- **Crop Recommendation System**
  In this, model is able to predict which crop can be suitable in any particular area.
- **College Admission Form**
  Develop my College Admission Form containing various Fields.

CERTIFICATIONS

- Machine Learning With Python - Froyo Technologies
- Introduction to Data Science Specialization - Coursera
- Python for Data Science, AI & Development - Coursera
- Fundamentals of Visualization with Tableau - Coursera
- Python - HackerRank

Aryan Patel

aryanpatel.life@gmail.com | +91-8303020425

EDUCATION

AJAY KUMAR GARG ENGG. COLLEGE
B.TECH(CSE)
2019-2023
81 Percent till 6th Semester

THE LUCKNOW PUBLIC COLLEGIATE
Lucknow,India
10th: 9.8 CGPA
12th: 83 Percent

SKILLS

PROGRAMMING
Proficient With :
•C
Familiar With :
•C++
• Python
• HTML
• CSS

COURSEWORK

UNDERGRADUATE
Data Structures
Algorithms
Computer Networks
Database Management System

LINKS

Github:// **aryanpatel11**
LinkedIn:// **aryanpatel11**

OBJECTIVE

To become a successful expert in the field of Computer Science by channelizing my technical knowledge and skills to ensure personal and professional growth and to contribute to the prosperity of the organization.

EXPERIENCE

TRAINING | MACHINE LEARNING WITH PYTHON
Froyo Technologies

- The project assigned to me was based on classification. The aim of the project was to predict inflation using 4 given BALLOON DATASET.
- The database was given to me and the task was to perform exploratory data analysis on the data set and build a model using classification which ultimately accurately predicts the inflated variable based on input variables given.
- Github://**aryanpatel11**

TRAINING | DATA ANALYTICS WITH PYTHON
Froyo Technologies

- The aim of the project was to perform the crime analysis of Boston using the CRIME DATASET of 4 years based on various parameters.
- Github://**aryanpatel11**

TRAINING | PROBLEM SOLVING USING DATA STRUCTURES
Ajay Kumar Garg Engineering College

- Deep analysis of various data structure algorithm implementation.

ACHIEVEMENTS

•100 hrs certification in Employ-ability Skills(aptitude, soft skills, carrier skills life skills)
•Certification in Student Excellence Learning Program
•Member of college Musical Society.

CERTIFICATIONS

Machine Learning With Python | FROYO TECHNOLOGIES
Data Analytics, Machine Learning and Artificial Intelligence with Python | FROYO TECHNOLOGIES
The Fundamentals of Digital Marketing | GOOGLE
Python Basics | UDEMY

ARYAMAN BENARA

Male 21 yrs
www.linkedin.com/in/aryaman-benara

9927023388
aryamanbenara@gmail.com
https://github.com/AryamanBenara

EDUCATION

- **GRADUATION**
  Ajay Kumar Garg Engineering College, Ghaziabad
  B.Tech in Computer Science and Engineering
  2019-2023 | Pursuing | Overall Percentage: 81.08(Till 6th Sem)

- **INTERMEDIATE**
  Ess Ess Convent Senior Secondary School, Agra | 2019
  Overall Percentage: 86.8%

- **HIGH SCHOOL**
  St. Peter's College, Agra | 2017
  Overall Percentage: 93.4%

CAREER OBJECTIVE

My skills and abilities have grown from my childhood experience: writing, coding, reading, quizzing. Software development has always inspired and amazed me, to turn into a career is my dream. I possess good communication, negotiation to go with my leadership abilities. I am a software enthusiast with particular interest in Data science and Java programming.

ACADEMIC PROJECTS

- Project 1: Worked on a frontend project to develop a To-do list in 2020
  https://github.com/AryamanBenara/To-do-list
- Project 2: Worked on a Group Project On developing an E-invoice and Billing System in 2019
- Project 3: Worked on an Individual ML Regression project using Python in 2021
  https://github.com/AryamanBenara/ML--Project
- Project 4: Working on a group ML project to build a crop recommendation system

TRAININGS AND WORK EXPERIENCE

- Completed an online course on Python in August 2019
- Worked as an Intern at the company TCR innovation as a technical content writer in May-June 2021
- Worked as an Intern at Froyo technologies in the domain ML with Python in Sept-Oct 2021
- Worked as an Intern at Technophilia in the domain of Front-end Web Development in July-Aug 2022

ACHIEVEMENTS & EXTRACURRICULAR ACTIVITIES

- Completed 100+ problems on GeeksForGeeks
- Led a Quiz Team for the Inter School Quiz Competition in 2016
- As a Project lead for a ML regression project using Python in October 2021
- Seven Gold medals in the G.K. Quiz, Cultural Fest- Eternia, St. Peter's College, Agra
- Won Gold medal at the Inter House Cricket tournament in 2019

SKILLS

Technical:

- Java, C++
- HTML, CSS
- ML algorithms, Python
- Operating Systems, DBMS, Computer networks

Soft Skills:

- Innovative, Confident
- Resilient, Teamwork, Leadership