

**A  
Minor Project-2  
Report  
On**

**“Crop Recommendation using ML”**

**Submitted in partial fulfillment of  
The requirements for the 6<sup>th</sup> Semester Sessional  
Examination of**

**BACHELOR OF TECHNOLOGY  
*IN***

**COMPUTER SCIENCE & ENGINEERING**

**By**

**Priya Khetan (20UG010184)  
Piyush Kumar (20UG01071)  
Anurag Thakur (20UG010169)**

**Under the able Supervision of  
Dr. Rashmita Panigrahi**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



**GIET UNIVERSITY, GUNUPUR  
2022 - 23**



# GIET UNIVERSITY, GUNUPUR

Dist. - Rayagada, Odisha-765022, Contact:- +91 7735745535,  
06857-250170,172, Visit us:- [www.giet.edu](http://www.giet.edu)

## Department of Computer Science & Engineering

### **CERTIFICATE**

*This is to certify that the project work entitled “**Crop Recommendation using ML** is done by **Piyush Kumar (20UG010171)**, **Anurag Thakur (20UG010169)**, **Priya Khetan (20IG010184)** in partial fulfillment of the requirements for the 6<sup>th</sup> Semester Sessional Examination of Bachelor of Technology in **Computer Science and Engineering** during the academic year 2022-23. This work is submitted to the department as a part of evaluation of 6<sup>th</sup> Semester Minor Project-2.*

***Project Supervisor***

***Class Teacher***

***Project Coordinator, 3rd Year***

***HoD, CSE***

## **ACKNOWLEDGEMENT**

We would like to thank my class teacher **Mr. G V S Narayana** for giving methis golden wonderful opportunity to do this project.

Secondly, we would like to thanks our Project Coordinator **Mr. Bhavani SankarPanda** for guiding, supporting and encouraging us throughout the project.

A special gratitude to our **HOD Dr. (Mrs.) Bandita Sahoo**, for providing us with all the facilities needed to complete my project.

A special gratitude to our **Deputy Dean Dr. Sanjay Kuanar** for making our path clear and supporting us in every moment of developing this project.

Lastly but not least we consider it a privilege and honor to express our sincere gratitude to our supervisor **Dr. Rashmita Panigrahi** for her valuable guidance throughout the tenure of this project work.

We are also thankful to seniors and friends who helped us directly or indirectly in solving problems and making the project more efficient and working.

PRIYA KHETAN – 20UG010184  
PIYUSH KUMAR – 20UG010171  
ANURAG THAKUR – 20IG010169

## **ABSTRACT**

Crop recommendation in agriculture science is the primary concern for every country, as the food demand is increasing at a fast rate due to an increase in population. Moreover, the increased use of technology today has increased the efficiency and accuracy of detecting diseases in crops. Most of the times it is observed that farmers tend to sow the crop according to its market value and possible financial profits rather than taking factors like soil conditions, sustainability etc. in to the account. The detection process marks the beginning of a series of activities to fight the diseases and reduce their spread. Some diseases are also transmitted between plants, making it hard to fight them. Machine learning algorithms allow choosing the most profitable crop list or predicting the crop yield for a user-selected crop. Our focus is to clarify the details about the diseases and how to detect diseases in crops automatically use of machine learning algorithm.

# Table of Contents

<b>1. Preface</b>	<b>1-2</b>
i. Introduction	1
ii. Existing System	1
iii. Drawbacks of Existing System	1
iv. Proposed System	2
v. Plan of Implementation	2
vi. Problem Statement	2
vii. Objective of the Project	2
<b>2. Theoretical Background</b>	<b>3-8</b>
i. Overview on Machine Learning	3
ii. Supervised and Unsupervised Learning	3
iii. Machine Learning Tools	4
iv. SciKit-learn	5
v. Google Colab	6
vi. Flask	6
vii. Dataset	6
viii. Data Preprocessing	7
ix. ML Algorithms	7
<b>3. System Requirements Specification</b>	<b>9-10</b>
i. Functional Requirement	9
ii. Non-Functional Requirement	9
<b>4. System Analysis</b>	<b>11-12</b>
i. Feasibility Study	11
ii. Analysis	11
<b>5. System Design</b>	<b>13-15</b>
i. System Development Methodology	13
ii. Model Phases	13
iii. System Architecture	14
iv. Sequence diagram	15
<b>6. Implementation</b>	<b>16-22</b>
<b>7. Conclusion</b>	<b>23</b>
<b>8. Bibliograph</b>	<b>24</b>

# 1. PREFACE

## 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.

## 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 Neighbors, and Regularized Greedy Forest is demonstrated 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.

## Drawbacks of Existing System

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.

## 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.

## Plan of Implementation

The steps involved in this system implementation are: -

**1. 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 dataset downloaded from Kaggle.

Datasets include: -

**Crops Name** along with their respective:

**Soil nutrient contents** (Nitrogen, Phosphorus, Potassium)

**Rainfall, Temperature, Humidity, Ph**

**2. Data Preprocessing:** In this step we transform raw data into a useful, understandable format. Real-world or raw data usually has inconsistent formatting, human errors, and can also be incomplete. Data preprocessing resolves such issues and makes datasets completer and more efficient to perform data analysis.

**3. 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.

**4. Deploying Web App:** After model is created and ready for prediction, we have used Flask to integrate our python code with Web to have good UI for users.

## 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 parameters.

## Objective of the Project

To build a robust model to give correct and accurate prediction of crop sustainability for the particular soil type and climatic conditions.

Provide recommendation of the best suitable crops in the area so that the farmer does not incur any losses

## 2. THEORETICAL BACKGROUND

### 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 artifacts. 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 confident 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 beforehand.

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.

### Supervised and Unsupervised Learning

Machine learning techniques can be broadly categorized 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 generalized 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, unlabeled 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 training 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 unlabeled 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 unlabeled 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 unlabeled data for training typically a small amount of labeled data and a large amount of unlabeled 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.

## **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 categorized as follows:

### **Pre-built application-based solutions.**

Programming languages which have specialized 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 programming 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 packages like RODBC, Gmodels, Class and Tm which are used in the field of machine 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 learning. 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 learning 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.

## 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 machine 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 machine 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 available 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.

## Google Colab

Google Colab was developed by Google to provide free access to GPU's and TPU's to anyone who needs them to build a machine learning or deep learning model. Google Colab can be defined as an improved version of Jupyter 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.

## Flask

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by Armin Ronacher who leads an international group of python enthusiasts (POCCO). Flask is based on Werkzeug WSGI (Web Server Gateway Interface) toolkit and Jinja2 template engine

## Dataset

For the system, we are using dataset downloaded from kaggle.

Datasets include: -

**Crops Name** along with their respective:

**Soil nutrient contents** (Nitrogen, Phosphorus, Potassium)

**Rainfall, Temperature, Humidity, Ph**

A brief description of the datasets:

**N** - ratio of Nitrogen content in soil

**P** - ratio of Phosphorous content in soil

**K** - ratio of Potassium content in soil

**temperature** - temperature in degree Celsius

**humidity** - relative humidity in %

**ph** - ph value of the soil

**rainfall** - rainfall in mm

## Data Preprocessing

After analysing and visualizing 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.

Missing data can be dealt with in two ways. The first method is to simply remove the entire row which contains the missing value. While this is an easy to execute method, it is better to use only on large datasets. Using this method on small datasets can reduce the dataset size too much, especially if there are a lot of missing values. This can severely affect the accuracy of the result. Since ours is a relatively small dataset, we will not be using this method.

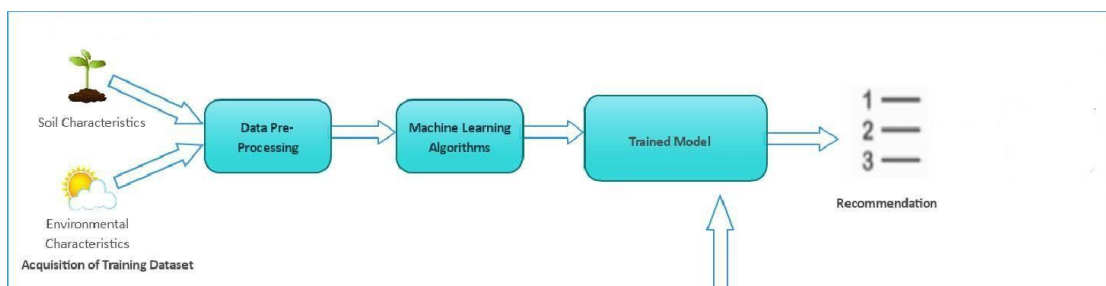


Figure 3.1: Machine Learning process

## Machine Learning Algorithms

Machine Learning algorithms used in the recommendation system are:

### Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis method used to find a linear combination of features that characterizes or separates classes. The resulting combination is used for dimensionality reduction before classification.

The goal of LDA (supervised) is to find the feature subspace that optimizes class separability.

### Logistic Regression (LR)

Logistic regression, despite its name, is a classification algorithm rather than regression algorithm. Based on a given set of independent variables, it is used to estimate discrete value (0 or 1, yes/no, true/false). It is also called logit or MaxEnt Classifier.

Basically, it measures the relationship between the categorical dependent variable and one or more independent variables by estimating the probability of occurrence of an event using its logistics function.

### **Gaussian Naive Bayes (NB)**

A Gaussian Naive Bayes algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e, normal distribution.

### **SVC**

A class of Support vector machines (SVMs) whose implementation is based on libsvm. This class handles the multiclass support according to one-vs-one scheme.

SVMs are powerful yet flexible supervised machine learning methods used for classification, regression, and, outliers' detection. SVMs are very efficient in high dimensional spaces and generally are used in classification problems. SVMs are popular and memory efficient because they use a subset of training points in the decision function.

### **K Neighbors Classifier**

The K in the name of this classifier represents the k nearest neighbors, where k is an integer value specified by the user. Hence as the name suggests, this classifier implements learning based on the k nearest neighbors.

### **Decision Tree Classifier**

Decision Tree is the most powerful non-parametric supervised learning method. They can be used for the classification and regression tasks. The main goal of DTs is to create a model predicting target variable value by learning simple decision rules deduced from the data features. Decision trees have two main entities; one is root node, where the data splits, and other is decision nodes or leaves, where we got final output.

### 3. 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 attributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources.

#### 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:

All the data must be in the same format as a structured data.

The data collected will be vectorized and sent across to the classifier

#### 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 behaviors. 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.

#### 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. 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.

## **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 point 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.

## **4. System Configuration**

### Hardware System Configuration:

Processor: 2 gigahertz (GHz) or faster processor or SoC.

RAM: 6 gigabytes (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. Seaborn
4. Matplotlib
5. Sklearn
6. Flask

## 4. SYSTEM ANALYSIS

### 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.

- **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 consume minimal cost and is economically feasible.

- **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

- **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.

### Analysis

- **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.



- **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.

- **Economic 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.

## 5. SYSTEM DESIGN

### System Development Methodology

System Development methodology is 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).

### 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.

1. **Requirement Analysis:** This phase is concerned about collection of requirements of the system. This process involves generating document and requirement review.
2. **System Design:** Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on: - algorithm, data structure, software architecture etc.
3. **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
4. **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 software documentation.
5. **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.
6. **Maintenance:** The maintenance phase is the longest phase in which the software is updated to fulfill 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.

### 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.

### System Architecture

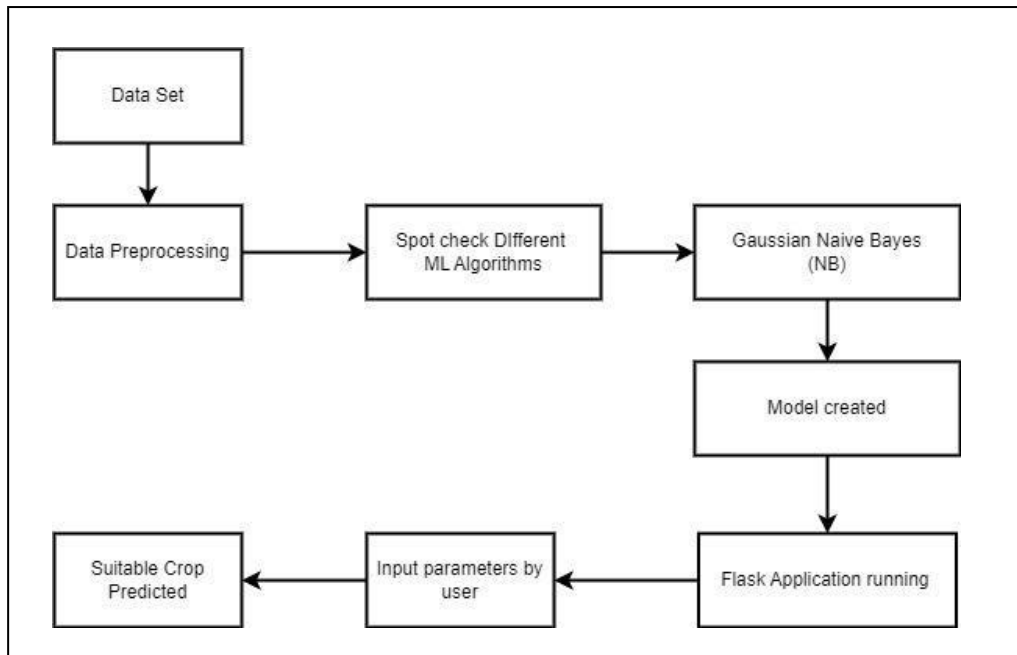


Fig. System Architecture

A system architecture is a conceptual model using which we can define the structure and behavior 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. Figure 6.2 depicts the system architecture and is explained in the following section.

## 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 realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios. Sequence diagrams are an easy and intuitive way of describing the behavior 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.

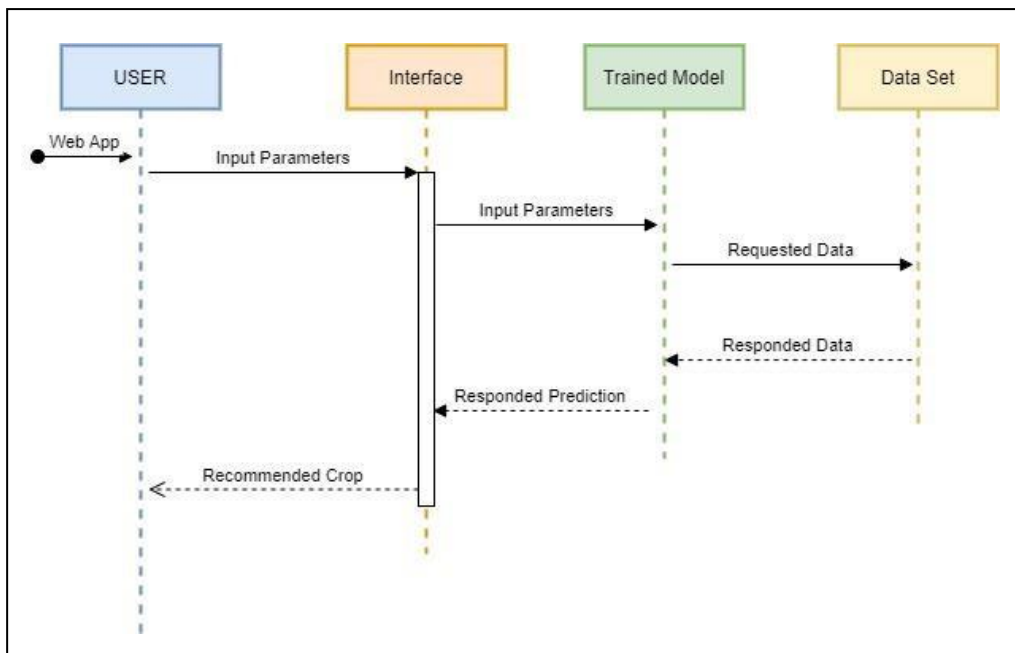


Fig. Sequence Diagram

## 5. IMPLEMENTATION

### Read the Data Set

```
#Acquisition of Training Dataset
df = pd.read_csv('/content/Crop_recommendation.csv')
df
```

	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 x 8 columns

### Data Analysis

```
# Analyze Data
def explore_data(df):
    print("Number of Instances and Attributes:", df.shape)
    print('\n')
    print('Dataset columns:', df.columns)
    print('\n')
    print('Data types of each columns: ', df.info())
    explore_data(df)
```

Number of Instances and Attributes: (2200, 8)

Dataset columns: Index(['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall', 'label'], dtype='object')

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
#   Column          Non-Null Count  Dtype
---  -
0   N                2200 non-null   int64
1   P                2200 non-null   int64
2   K                2200 non-null   int64
3   temperature      2200 non-null   float64
4   humidity         2200 non-null   float64
5   ph               2200 non-null   float64
6   rainfall         2200 non-null   float64
7   label            2200 non-null   object
dtypes: float64(4), int64(3), object(1)
memory usage: 137.6+ KB
Data types of each columns: None
```

## Data preprocessing

```
# Checking for duplicates
def checking_removing_duplicates(df):
    count_dups = df.duplicated().sum()
    print("Number of Duplicates: ", count_dups)
    if count_dups >= 1:
        df.drop_duplicates(inplace=True)
        print('Duplicate values removed!')
    else:
        print('No Duplicate values')

checking_removing_duplicates(df)
```

```
Number of Duplicates: 0
No Duplicate values
```

```
df.isna().sum()
```

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

## Checking and removing outliers

To check outliers of each columns, use boxplot  
These 6 columns have outliers:

1. P
2. K
3. temperature
4. humidity
5. ph
6. rainfall

```
# All columns contain outliers except for rice and label you can check outliers by using boxplot
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
df_out = df[~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
```

## Train model

```
# Spot-Check Algorithms
def GetModel():
    Models = []
    Models.append(('LR' , LogisticRegression()))
    Models.append(('LDA' , LinearDiscriminantAnalysis()))
    Models.append(('KNN' , KNeighborsClassifier()))
    Models.append(('CART' , DecisionTreeClassifier()))
    Models.append(('NB' , GaussianNB()))
    Models.append(('SVM' , SVC(probability=True)))
    return Models

# Test options and evaluation metric
def fit_model(X_train, y_train,models):
    num_folds = 10
    scoring = 'accuracy'

    results = []
    names = []
    for name, model in models:
        kfold = KFold(n_splits=num_folds, shuffle=True, random_state=0)
        cv_results = cross_val_score(model, X_train, y_train, cv=kfold, scoring=scoring)
        results.append(cv_results)
        names.append(name)
        msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
        print(msg)

    return names, results

# Split training and validation set
def read_in_and_split_data(data, target):
    X = data.drop(target, axis=1)
    y = data[target]
    X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=0)
    return X_train, X_test, y_train, y_test

target = 'label'
X_train, X_test, y_train, y_test = read_in_and_split_data(df, target)

models = GetModel()
names,results = fit_model(X_train, y_train,models)
```

```
LR: 0.955114 (0.016749)
LDA: 0.965909 (0.013203)
KNN: 0.979545 (0.005207)
CART: 0.984091 (0.011307)
NB: 0.993750 (0.004720)
SVM: 0.976705 (0.008214)
```

## Experiment Different preprocessing techniques

```
def NormalizedModel(nameOfScaler):
    if nameOfScaler == 'standard':
        scaler = StandardScaler()
    elif nameOfScaler == 'minmax':
        scaler = MinMaxScaler()
    elif nameOfScaler == 'normalizer':
        scaler = Normalizer()
    elif nameOfScaler == 'binarizer':
        scaler = Binarizer()

    pipelines = []
    pipelines.append((nameOfScaler+'LR' , Pipeline([('Scaler', scaler),('LR' , LogisticRegression())])))
    pipelines.append((nameOfScaler+'LDA' , Pipeline([('Scaler', scaler),('LDA' , LinearDiscriminantAnalysis())])
    )
    pipelines.append((nameOfScaler+'KNN' , Pipeline([('Scaler', scaler),('KNN' , KNeighborsClassifier())])))
    pipelines.append((nameOfScaler+'CART' , Pipeline([('Scaler', scaler),('CART' , DecisionTreeClassifier())])))
    pipelines.append((nameOfScaler+'NB' , Pipeline([('Scaler', scaler),('NB' , GaussianNB())])))
    pipelines.append((nameOfScaler+'SVM' , Pipeline([('Scaler', scaler),('SVM' , SVC())])))

    return pipelines

#Normalization by minmax
ScaledModel = NormalizedModel('minmax')

name,results = fit_model(X_train, y_train, ScaledModel
```

```
minmaxLR: 0.932955 (0.025763)
minmaxLDA: 0.965909 (0.013203)
minmaxKNN: 0.978409 (0.008351)
minmaxCART: 0.984091 (0.011017)
minmaxNB: 0.993750 (0.004720)
minmaxSVM: 0.982955 (0.006723)
```

```
#Normalization by standard
ScaledModel = NormalizedModel('standard')
name,results = fit_model(X_train, y_train, ScaledModel)
```

```
standardLR: 0.969886 (0.014827)
standardLDA: 0.965909 (0.013203)
standardKNN: 0.968750 (0.011149)
standardCART: 0.982386 (0.011768)
standardNB: 0.993750 (0.004720)
standardSVM: 0.982386 (0.008966)
```



```
#Normalization by normalizer
ScaledModel = NormalizedModel('normalizer')
name,results = fit_model(X_train, y_train, ScaledModel)
```

```
normalizerLR: 0.853409 (0.024871)
normalizerLDA: 0.934659 (0.012768)
normalizerKNN: 0.944318 (0.015622)
normalizerCART: 0.934659 (0.010554)
normalizerNB: 0.956250 (0.011090)
normalizerSVM: 0.951705 (0.012768)
```

## Performance Evaluation

After apply the data set for Different preprocessing techniques, it seems **Gaussian Naïve Bayes Algorithm** is best fit for the given dataset.

Hence, using it to generate the prediction model.

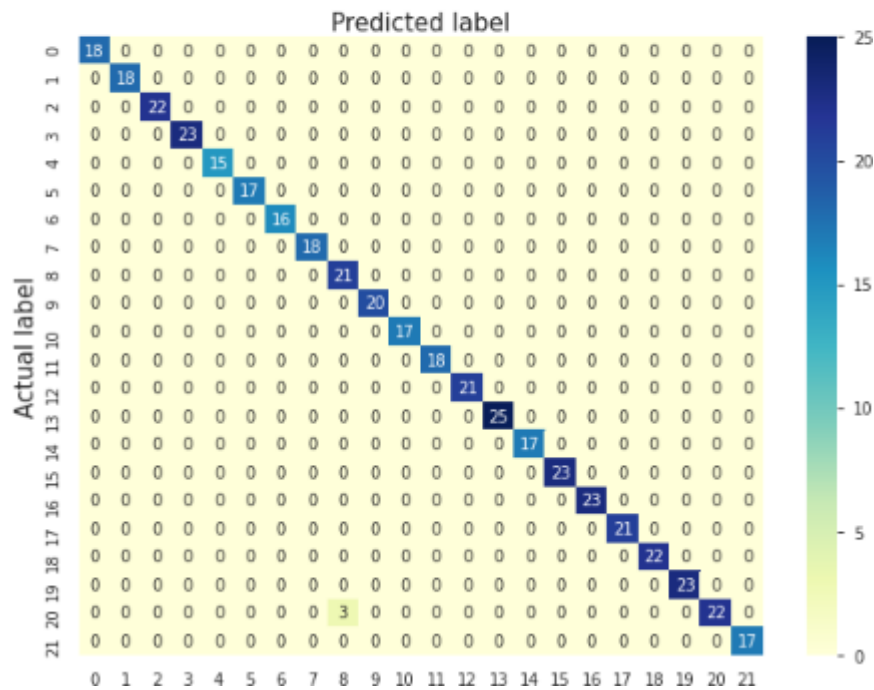
```
# Performance Measure
def classification_metrics(model, conf_matrix):
    print(f"Training Accuracy Score: {model.score(X_train, y_train) * 100:.1f}%")
    print(f"Validation Accuracy Score: {model.score(X_test, y_test) * 100:.1f}%")
    fig,ax = plt.subplots(figsize=(8,6))
    sns.heatmap(pd.DataFrame(conf_matrix), annot = True, cmap = 'YlGnBu',fmt = 'g')
    ax.xaxis.set_label_position('top')
    plt.tight_layout()
    plt.title('Confusion Matrix', fontsize=20, y=1.1)
    plt.ylabel('Actual label', fontsize=15)
    plt.xlabel('Predicted label', fontsize=15)
    plt.show()
    print(classification_report(y_test, y_pred))
```

```
# Save trained model
def save_model(model,filename):
    pickle.dump(model, open(filename, 'wb'))
```

```
#creating model with GaussianNB
pipeline = make_pipeline(MinMaxScaler(), GaussianNB())
model = pipeline.fit(X_train, y_train)
y_pred = model.predict(X_test)
conf_matrix = confusion_matrix(y_test,y_pred)
classification_metrics(pipeline, conf_matrix)
save_model(model, 'model.pkl')
```

Training Accuracy Score: 99.5%  
 Validation Accuracy Score: 99.3%

Confusion Matrix



	precision	recall	f1-score	support
apple	1.00	1.00	1.00	18
banana	1.00	1.00	1.00	18
blackgram	1.00	1.00	1.00	22
chickpea	1.00	1.00	1.00	23
coconut	1.00	1.00	1.00	15
coffee	1.00	1.00	1.00	17
cotton	1.00	1.00	1.00	16
grapes	1.00	1.00	1.00	18
jute	0.88	1.00	0.93	21
kidneybeans	1.00	1.00	1.00	20
lentil	1.00	1.00	1.00	17
maize	1.00	1.00	1.00	18
mango	1.00	1.00	1.00	21
mothbeans	1.00	1.00	1.00	25
mungbean	1.00	1.00	1.00	17
muskmelon	1.00	1.00	1.00	23
orange	1.00	1.00	1.00	23
papaya	1.00	1.00	1.00	21
pigeonpeas	1.00	1.00	1.00	22
pomegranate	1.00	1.00	1.00	23
rice	1.00	0.88	0.94	25
watermelon	1.00	1.00	1.00	17
accuracy			0.99	448
macro avg	0.99	0.99	0.99	448
weighted avg	0.99	0.99	0.99	448

## Predict unseen data

```
N = 90
P = 42
K = 43
temperature = 20.82312
humidity = 82.00284
ph = 6.50232
rainfall = 202.93536

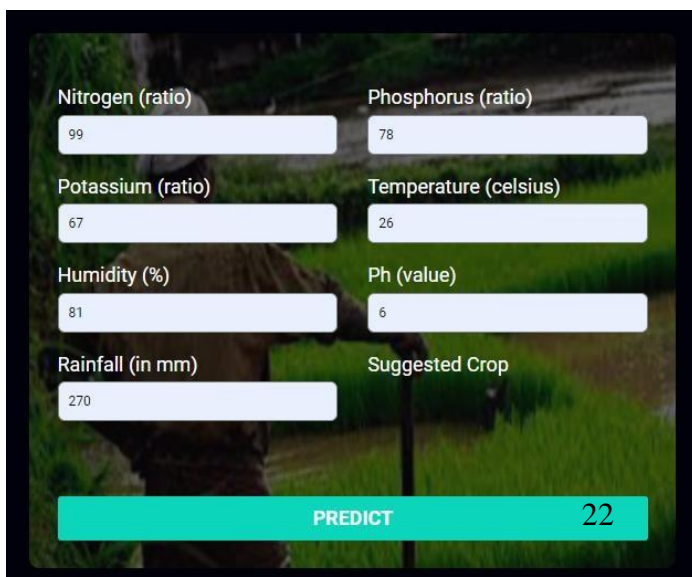
sample = [N, P, K, temperature, humidity, ph, rainfall]
single_sample = np.array(sample).reshape(1,-1)
pred = model.predict(single_sample)
pred.item().title()
```

**"Rice"**

## Integrating with Flask to view result as web app

```
from flask import Flask,request,jsonify,url_for,render_template
import numpy as np
import pandas as pd
import pickle

app=Flask(__name__)
pic_model=pickle.load(open('model.pkl','rb'))
@app.route('/')
def home():
    return render_template("home.html")
@app.route('/',methods=['POST'])
def predict():
    data=[float(i) for i in request.form.values()]
    data=np.array(data).reshape(1,-1)
    output=pic_model.predict(data)
    return render_template("home.html",predicted_value='{ }'.format(output[0]))
app.run()
```



Nitrogen (ratio)	Phosphorus (ratio)
<input type="text" value="99"/>	<input type="text" value="78"/>
Potassium (ratio)	Temperature (celsius)
<input type="text" value="67"/>	<input type="text" value="26"/>
Humidity (%)	Ph (value)
<input type="text" value="81"/>	<input type="text" value="6"/>
Rainfall (in mm)	Suggested Crop
<input type="text" value="270"/>	
<input type="button" value="PREDICT"/>	
22	

# Web App Snap

## 7. 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 from incurring losses. The system can be extended to the mobile app and can be accessed by millions of farmers across the country in just a single click.

We have achieved an accuracy of 99.5 percent from the Gaussian Naive Bayes. Further development is to integrate the crop recommendation system with another subsystems, Profit predictor and yield predictor that would also provide the farmer an estimate of profit and production if he plants the recommended crop.

## **WORKS DONE IN THE RELATED AREA**

These are few research papers that we have went through for reference and found which is better for implementations of project.

<b>Paper/ Source</b>	<b>Topic</b>	<b>Problem Faced</b>	<b>Stage-of Project</b>
Dataset	Data pre-processing	Data Conversion , managing missing values	<b><u>Initial</u></b>
Data mining Techniques  By- <u>Valmik B Nikam</u>	Removing null values, noisy data	Data in huge amounts regularly will be unreliable or inaccurate	<b><u>Initial</u></b>
Logistic Regression  By- <u>Gaurav Chauhan</u>	Detecting error and accuracy	Improved Accuracy	<b><u>Middle</u></b>
Deploy Machine Learning Models to our project	Logistics Regression, Linear Regression, Decision Tree	Error in Prediction of some algorithm like SVM, NB classifier	<b><u>Middle</u></b>
Development of Front- end and done with its connections	HTML, CSS , Flask	Some problem faced in installing Flask and its path selection	<b><u>FINAL</u></b>

## 8. BIBLIOGRAPHY

- [1] AgroConsultant: Intelligent Crop Recommendation System Using Machine Learning Algorithms. Zeel Doshi, Subhash Nadkarni, Rashi Agrawal, Prof. Neepa Shah.
- [2] Tom M. Mitchell, Machine Learning, India Edition 2013, McGraw Hill Education.
- [3] <https://towardsdatascience.com/types-of-machine-learning-algorithms-you-should-know-953a08248861>
- [4] <https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset>