

# CROP PREDICTION & ANALYSIS USING SUPERVISED MACHINE LEARNING MODELS

A PROJECT REPORT

In a partial fulfillment of the Requirements for the award of the degree of  
**BACHELOR OF TECHNOLOGY** under CSE department

Under the guidance of

MAHENDRA DUTTA

By

**SATYABRATA SAHOO**

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR



In association with



(ISO 9001:2015)

## DECLARATION

I hereby declare that the project work being presented in the project proposal entitled “Crop Prediction & Analysis using Supervised Machine Learning Models” in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE & ENGINEERING** branch at **KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY, BHUBANESWAR**, is an authentic work carried out under the guidance of Mr. Mahendra Dutta. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date: 30/06/2025

Name of the Student: SATYABRATA SAHOO



Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)



## ACKNOWLEDGEMENT

Success of any project depends largely on the encouragement and guidelines of many others. We take this sincere opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project work. We would like to show our greatest appreciation to **MAHENDRA DUTTA**, Project Engineer at Ardent Computech Pvt. Ltd. We always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized. Words are inadequate in offering our thanks to the other trainees, project assistants and other members at Ardent Computech Pvt. Ltd. for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

## **Abstract**

Traditionally, crop yield and selection predictions are performed with the help of large, complex agronomic models that rely on multiple environmental and soil parameters over extended periods. These models often suffer from instability due to unpredictable natural conditions, leading to reduced accuracy. Additionally, they require extensive computational resources typically found in High-Performance Computing (HPC) environments, resulting in high energy consumption. In this paper, we present a crop prediction and analysis technique that leverages historical agricultural and weather data to train simple supervised machine learning models, which can generate reliable crop recommendations for upcoming seasons in a much shorter time. These models can be executed on significantly less resource-intensive systems. Our evaluation demonstrates that the predictive accuracy of these models is sufficiently high to complement traditional methods. Furthermore, we show that integrating data from multiple nearby regions yields better results than relying solely on localized data.

In this research, A machine learning based weather forecasting model was proposed, the model was implemented using four classifier algorithms which include RandomForestClassifier, Decision Tree Algorithm, Gaussian Naïve Bayes Model, SVM classifier & K-Nearest Neighbour model

## **CHAPTERS**

## **PAGE NO.**

Abstract

Chapter 1: Introduction

5-6

Chapter 2: Related Work

6

Chapter 3: Literature Survey

7

Chapter 4: Software & Tools/Modules  
used

8

Chapter 5: Methodology

9-14

Chapter 6: Evaluation & Result

15-20

Chapter 7: Performance comparision of  
the models

21

Chapter 8: Result Analysis

22

Chapter 9: Conclusion & Future work

22

Chapter 10: Bibilography

23



## **Chapter 1**

# **Introduction**

**Crop Prediction** is the process of forecasting the most suitable crop to cultivate in a specific region based on various environmental and agricultural parameters. It leverages data such as soil type, pH level, temperature, humidity, rainfall, and other climatic and geographic factors to predict crop yield and guide farmers toward informed decision-making. In today's era of smart farming and precision agriculture, accurate crop prediction plays a crucial role in enhancing agricultural productivity, minimizing crop failure, and ensuring food security.

Several techniques are employed for crop prediction, including statistical methods, machine learning algorithms, and hybrid models that analyze historical crop data along with real-time environmental inputs. These predictive models help optimize resource usage such as water, fertilizers, and pesticides, thereby reducing environmental impact and increasing farm profitability.

Crop prediction systems are vital for various stakeholders including farmers, agricultural researchers, government bodies, and agribusinesses. By integrating technology with traditional farming practices, crop prediction supports sustainable agriculture, enables effective planning, and helps in mitigating the effects of climate variability and changing weather patterns.

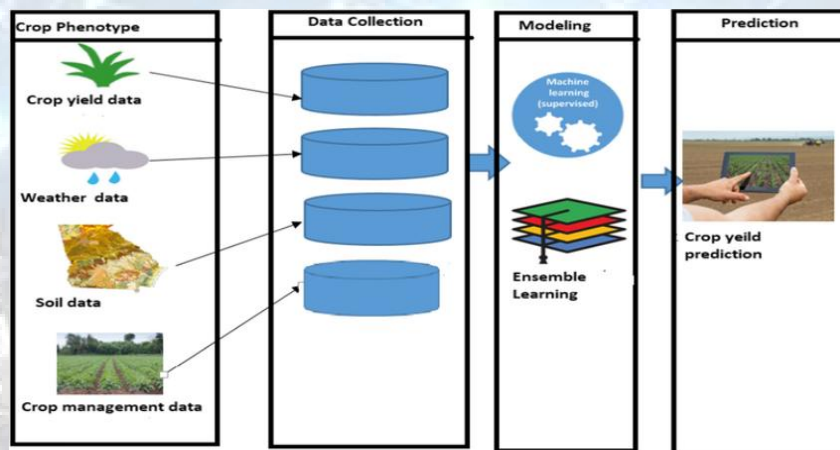
Machine learning approach to weather forecast uses quantitative metrological data to build models and tries to improve the performance of the model by learning from the dataset.

In this study, a machine learning model was proposed and implemented. The main aim of this research is to develop a predictive model for forecasting weather.

### **The objectives include:**

1. Review related work on crop forecasting
2. Develop an efficient and effective model for crop forecasting.
3. Evaluate the performance of the models developed using accuracy and precision to determine which algorithm is best suited for the prediction of crop conditions.

Thus, our aim is to provide accurate result in order to provide correct prediction of crops for future so in critical conditions people can be aware of upcoming natural calamities.



## **Chapter 2**

### **Related Work**

In recent years, crop prediction using machine learning has gained significant attention due to its potential to revolutionize agriculture through data-driven decision-making. Several researchers have explored diverse approaches to enhance crop yield forecasting and recommendation systems. S. Bendre and N. Thool (2016) focused on integrating big data and supervised learning techniques to improve precision agriculture by linking weather forecasting with crop selection. R. Ramesh and S. V. V. Raghavan (2018) used Decision Trees and Random Forest algorithms, demonstrating that Random Forest provided better accuracy in crop yield prediction based on rainfall and soil data. M. U. Rehman et al. (2019) compared Support Vector Regression and Artificial Neural Networks, concluding that ANN handled non-linear relationships in crop data more effectively. Patil and Kumaraswamy (2020) proposed a multi-model approach using KNN, Linear Regression, and SVM, with SVM achieving the highest accuracy for predicting yields of crops like maize and soybean. A comprehensive review by Chlingaryan, Sukkariet, and Whelan (2018) highlighted the use of machine learning methods such as Random Forest, Gradient Boosting, and CNNs for various precision agriculture applications, including crop prediction, soil monitoring, and pest detection. Sowmya and Manjula (2021) developed a crop recommendation system using Naive Bayes, Random Forest, and XGBoost, where Random Forest achieved 92% accuracy by analyzing soil nutrients and climatic data. Finally, K. Rajeswari and R. Thanushkodi (2020) emphasized the effectiveness of hybrid and ensemble models for smart agriculture and advocated for the integration of IoT and sensor-based data to enhance prediction accuracy. Collectively, these studies showcase the transformative role of machine learning in optimizing crop planning, improving yield predictions, and supporting sustainable agricultural practices.

## Chapter 3

### Literature Survey

#### Literature Survey on Crop Prediction & Analysis Using Machine Learning

S. No.	Author(s)	Source	Title	Method Used	Outcome / Contribution	Year
1	S. Bendre, N. Thool	IEEE Xplore	Big Data in Precision Agriculture: Weather Forecasting for Future Farming	Supervised Learning on climate + crop data	Demonstrated improved farm decision-making using ML and weather data for crop recommendation	2016
2	R. Ramesh, S. V. V. Raghavan	IEEE Xplore	Crop Yield Prediction Using ML Algorithms: A Case Study from India	Decision Tree, Random Forest	Random Forest provided better accuracy in crop yield prediction using rainfall and soil data	2018
3	M. U. Rehman, et al.	Elsevier	ML-Based Crop Yield Prediction for Sustainable Agriculture	Support Vector Regression, Artificial Neural Network	ANN handled nonlinear relationships better and outperformed SVR on real-world data	2019
4	Patil & Kumaraswamy	IEEE Xplore	A ML Approach for Crop Yield Prediction Based on Climatic Parameters	KNN, Linear Regression, SVM	SVM provided highest accuracy (~88%) among tested models for crops like maize and soybean	2020
5	A. G. Chlingaryan, S. Sukkarieh, B. Whelan	Computers and Electronics in Agriculture (Elsevier)	Machine Learning in Precision Agriculture: A Review	RF, Gradient Boosting, CNNs, IoT Integration	Reviewed ML applications for crop health, prediction, yield estimation, and smart agriculture	2018
6	Sowmya, Manjula	Springer	Crop Recommendation System Using ML Algorithms	Naive Bayes, Random Forest, XGBoost	Random Forest achieved highest accuracy (~92%) for recommending crops based on soil and climate data	2021



## **Chapter-4**

### **SOFTWARE USED**

Tool	Purpose
Python (3.x)	Programming language used for building the machine learning model
Google Colab	Interactive coding environment for developing and testing the model
Microsoft Excel	Preliminary data viewing and preprocessing (optional)
Microsoft Word	Documentation and report writing
Microsoft PowerPoint	Project presentation
GitHub	Version control and project backup (optional)

### **TOOLS & LIBRARIES USED**

Library	Purpose
pandas	Data loading, cleaning, and manipulation
numpy	Numerical computations
scikit-learn (sklearn)	Core machine learning library for model training, testing, evaluation (Random Forest, SVM, etc.)
matplotlib	Data visualization (bar charts, scatter plots, etc.)
seaborn	Statistical and advanced visualizations

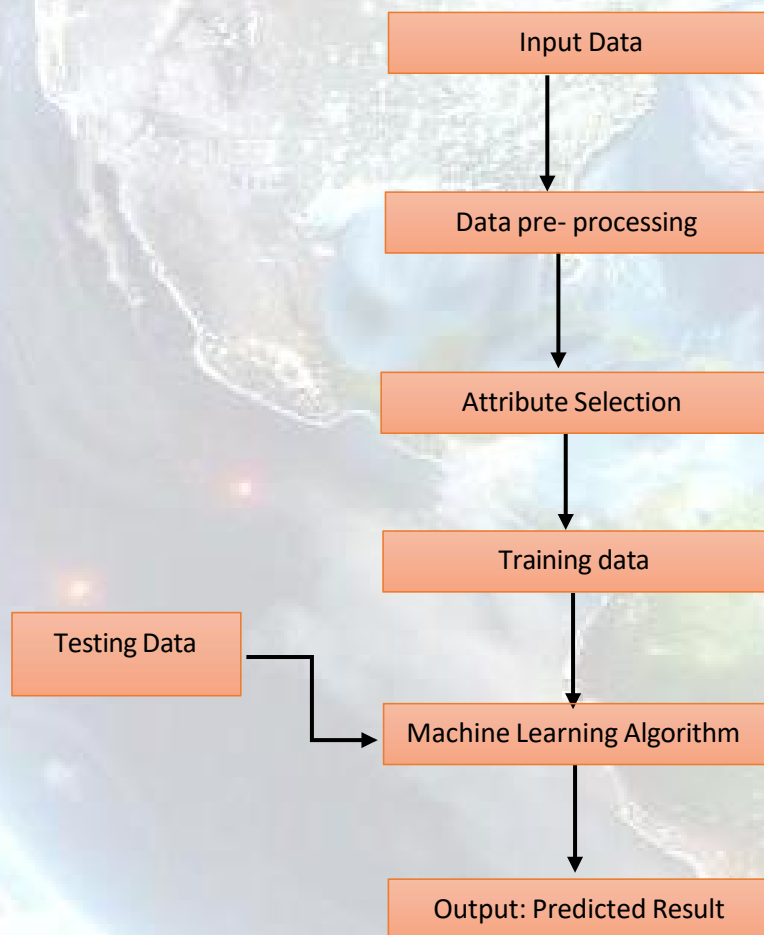


## Chapter-5

# Methodology

In this section, I present the proposed model for CROP PREDICTION using machine learning., it also contains a description of the dataset, its acquisition and pre- processing and the analysis of the Algorithm used.

### 5.1 Proposed Model



## 5.2 Data acquisition and pre- processing

The dataset is based on agricultural and climatic data from various regions of India, and is commonly used for academic and research purposes in crop recommendation systems.

### File Format:

**Type:** Comma-Separated Values (CSV)

**Rows:** 2200 entries (each row represents one sample record)

**Columns:** 8

- **N** – Nitrogen content in the soil (mg/kg)
- **P** – Phosphorus content in the soil (mg/kg)
- **K** – Potassium content in the soil (mg/kg)
- **temperature** – Average temperature in degrees Celsius
- **humidity** – Relative humidity in percentage (%)
- **ph** – Soil pH value
- **rainfall** – Rainfall in millimeters (mm)
- **label** – The most suitable **crop name** (e.g., rice, maize, cotton, etc.)

```
df.head() #To get only first 5 rows
```

	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

Fig 3.2 Data head Overview

```
df.isnull().sum() #To check how many null values each feature has
```

	0
N	0
P	0
K	0
temperature	0
humidity	0
ph	0
rainfall	0
label	0

Fig 3.3 Checking data completeness

```
#split features & labels
x = df.iloc[:, :-1] #features
x
```

Fig 3.4 Extracting the feature

Following the pre-processing of the dataset, to better understand the attributes and their relationship, a graph count of the attributes was generated.

```
y = df.iloc[:, -1] #labels
y
```

```
#We have splitted the data into training and testing
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
x_train.head() #Display the training data of only the features
```

```
x_test.head() #Display the testing data of only the features
```

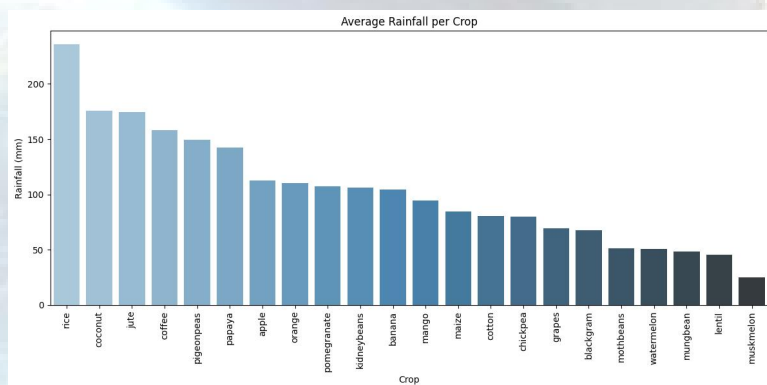
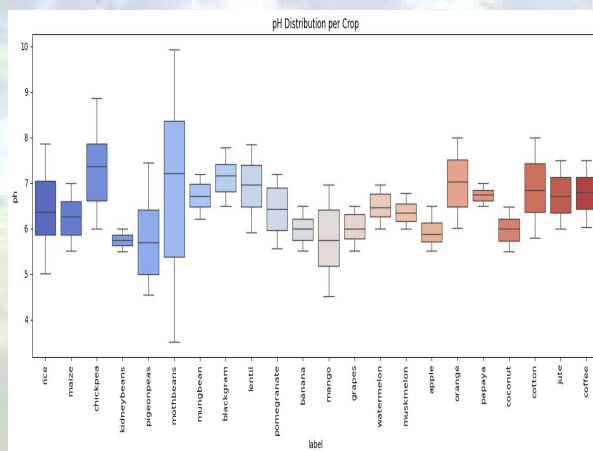
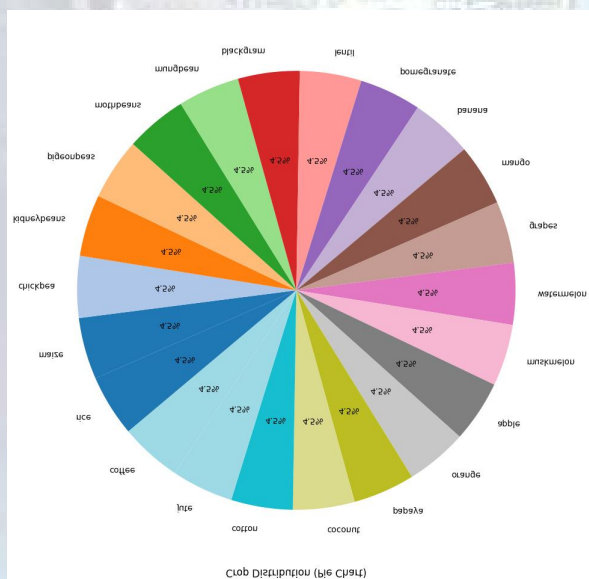
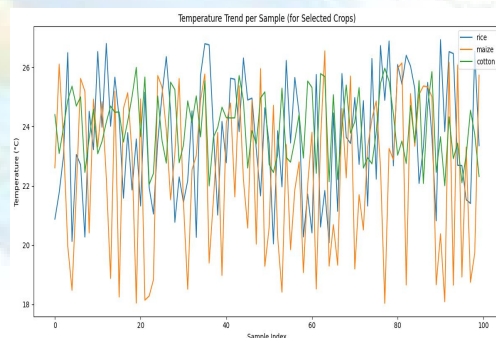
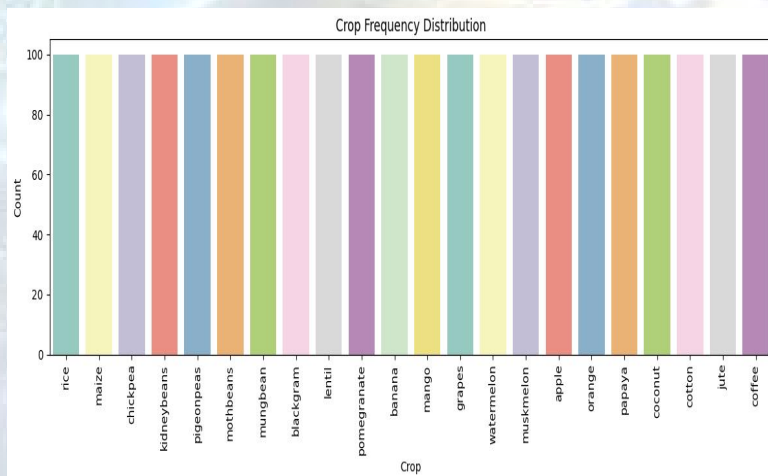
	N	P	K	temperature	humidity	ph	rainfall
1451	101	17	47	29.494014	94.729813	6.185053	26.308209
1334	98	8	51	26.179346	86.522581	6.259336	49.430510
1761	59	62	49	43.360515	93.351916	6.941497	114.778071
1735	44	60	55	34.280461	90.555616	6.825371	98.540477
1576	30	137	200	22.914300	90.704756	5.603413	118.604465

```
y_test.head() #Display the output/label testing data
```

	label
1451	muskmelon
1334	watermelon
1761	papaya
1735	papaya
1576	apple

dtype: object

## 5.3 Data Visualization





## 5.4 Analysis

For the analysis of our data, the data was split into tests and Training dataset, using 10% for testing, four machine learning classification algorithms were implemented, they are:

- K- Nearest Neighbors (KNN)
- Random. Forest classifier
- SVM
- Gaussian Naïve Bayes model
- Decision Tree Algorithm
- Logistics Regression Algorithm

### K- Nearest Neighbors

The dataset is used by KNN to make predictions. Probabilities for new instances (x) are calculated by scanning the data set for the K most comparable examples and predicting the output variable for those K occurrences.

### Random Forest Classifier

A random forest classifier is a collection of tree-structured classifiers whose results are compounded into one result; it is an ensemble machine learning algorithm which can be implemented for both classification and regression tasks and is made up of a set of classifiers known as a decision tree, random forest classifier is known to produce accurate predictions, provides flexibility and reduced the risk of overfitting.

### Gaussian Naive Bayes model


Gaussian Naive Bayes model is based on Bayes theorem, it assumes that a particular feature is independent of the value of any other feature, they can be trained very efficiently and are highly scalable, and the likelihood of the features is assumed to be

$$P(x_i | y) = \frac{1}{2 \pi \sigma^2} \exp \left( -\frac{(x_i - \mu_y)^2}{2 \sigma_y^2} \right)$$

the variance is independent of y and x

### Decision Tree algorithm

Decision tree algorithm. Belongs to the family of supervised machine learning, where the data is continuously split according to a certain parameter and represented by a tree structure. It is used to solve classification and regression tasks and is among the most popular machine learning algorithm. (Xindong et al. 2018) These algorithms were selected for the task because they were utilized by other weather forecasting models and have shown



to have high predictive performance capabilities, they were Implemented with python programming language using the Jupiter notebook, and various libraries were imported for the analysis including pandas, ski-learn, TensorFlow, and Matplot library.

### **Logistic Regression Algorithm**

The LR Algorithm computes the link between one or more independent factors and the category dependent variable. The output of LR is in the form of binary classification. A logistic function (sigmoid function) can be used to calculate the probability.  $1 / (1 + e^{-\text{value}})$  Where e is the natural logarithm base (Euler's number or the EXP () function) and value is the actual numerical value to be transformed. The logistic function was used to turn a situation with numbers ranging from -5 to 5 into a range of 0 to 1.

## Chapter-6

# Evaluation and result

A variety of results was obtained from the models trained with 75% of the data and tested with 25% of the data, in this section, we evaluate our models developed using various metrics, the metrics used for evaluation include:

- Accuracy
- Precision
- Recall
- Fi-score

### Accuracy:

This is the total proportion of observations that have been correctly predicted mathematically, accuracy is defined as:

$$\frac{TP + TN}{TP + FP + TN + FN}$$

where TP= True positive, TN= True Negative, FP= False positive and FN = false negative.

### Precision:

This is the percentage of the positive instance predicted that was correct, mathematically it is defined as

$$\frac{TP}{TP + FP}$$

where TP= True positive, FP= False Positive

### Recall:

This is the percentage of the positive instance out of the total actual positive, mathematically it is defined as:

$$\frac{TP}{TP + FN}$$

where TP= True positive, FN= False Negative

### F1-score:

This is the harmonic mean of precision and recall metrics, it is the overall correctness the model has achieved, mathematically it is defined as

$$\frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * Precision * recall}{Precision + Recall}$$

$$\frac{1}{\text{Precision}} + \frac{1}{\text{recall}}$$

recall + Precision

## 1. Random Forest Classifier

For the random forest classifier accuracy, precision, recall, and f1- score were generated, as it is shown in figure 4.1 below, and the random forest classifier achieved a total accuracy of 99.31 %

```

model = RandomForestClassifier() #To call the ML model
model.fit(x_train,y_train) #To train the model

RandomForestClassifier
RandomForestClassifier()

predictions = model.predict(x_test) #To predict the output

y_pred = model.predict(x_test)

from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
print("Accuracy = ",accuracy_score(y_test,y_pred))

```

```

... [[23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0 20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0 23  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0 20  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0 11  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0 21  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 14  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19]]
      precision    recall  f1-score   support

 apple          1.00      1.00      1.00         23
...
 macro avg       0.99      0.99      0.99        440
 weighted avg    0.99      0.99      0.99        440

Accuracy = 0.9931818181818182

```

## 2. Gaussian Naive Bayes model

The Gaussian Naive Bayes model achieved an 99.5 % accuracy, recall, f1-score, and precision were also calculated as shown in figure 4.2 below



▼ GaussianNB ⓘ ?

```
GaussianNB()
```

```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
print("Accuracy = ", accuracy_score(y_test, y_pred))
```

```

      apple      1.00      1.00      1.00      23
...
      macro avg      1.00      1.00      1.00      440
weighted avg      1.00      1.00      1.00      440

Accuracy = 0.9954545454545455

```

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=1)
```

```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
print("Accuracy = ", accuracy_score(y_test, y_pred))
```

```

... [[23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0 20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0]
      [ 0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0 21  0  0  0  0  0  0  0  0  0  2  0]
      [ 0  0  0  0  0  0  0  0  0 20  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0 11  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0 21  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0 1  0  0 23  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 14  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0  0  0  0 15  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19]]

              precision    recall  f1-score   support

    apple                1.00        1.00        1.00         23

...
   macro avg              0.98        0.98        0.98        440
  weighted avg              0.98        0.98        0.98        440

Accuracy = 0.9818181818181818

```

## Logistic Regression Algorithm

The Logistic Regression Algorithm achieved an accuracy of 94.50 %, Recall, precision and f1-score were generated. As shown in figure 4.4 below

```

# Logistic Regression

from sklearn.linear_model import LogisticRegression
lr_model = LogisticRegression()
lr_model.fit(x_train,y_train)

```

```

y_pred = lr_model.predict(x_test)

from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
print("Accuracy = ",accuracy_score(y_test,y_pred))

```

LogisticRegression

LogisticRegression()

```

... [[23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0 15  0  0  0  0  0  0  2  0  0  3  0  0  0  0  0  0  0]
      [ 0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0 16  0  0  0  0  1  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0 1  0 20  0  0  0  0  0  0  1  0  1]
      [ 0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0  0  1  0]
      [ 0  0  0  0  0  0  0  0  0  0 10  0  0  1  0  0  0  0  0  0]
      [ 0  1  0  0  0  0  4  0  0  0  0 16  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0 22  1  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 14  0  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 21  0  1]
      [ 0  0  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 21  0  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
      [ 0  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0  0  0  0 17  0]
      [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19]]

              precision    recall  f1-score   support

    apple                1.00        1.00        1.00         23

...
   macro avg              0.94        0.95        0.94        440
  weighted avg              0.95        0.95        0.94        440

Accuracy = 0.9454545454545454

```

## K- Nearest Neighbors

The K- Nearest Neighbors achieved an accuracy of 97.04 %, Recall, precision and f1- score were generated. As shown in figure 4.5 below

```
#K-Nearest Neighbour Model
```

```
from sklearn.neighbors import KNeighborsClassifier
knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(x_train,y_train)
```

▼ KNeighborsClassifier ⓘ ?

KNeighborsClassifier()

```
y_pred = knn_model.predict(x_test)
```

```
from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
print("Accuracy = ",accuracy_score(y_test,y_pred))
```

```
[[23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0 20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0 16  0  0  1  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0 22  0  0  0  0  0  0  0  0  0  0  1]
 [ 0  0  0  0  0  0  0  0  0 20  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0 11  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  1  0  0  0  0 20  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  2  0  0 22  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 14  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
 [ 0  0  1  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0 21  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0]
 [ 0  0  0  0  0  0  0  0  6  0  0  0  0  0  0  0  0  0  0  13  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19]]
```

```
precision    recall  f1-score   support
```

```
apple        1.00      1.00      1.00         23
```

```
...
```

```
macro avg    0.97      0.97      0.97        440
```

```
weighted avg 0.97      0.97      0.97        440
```

```
Accuracy = 0.9704545454545455
```



## SVM (Support Vector Machine)

The SVM achieved an accuracy of 97.90 %, Recall, precision and f1-score were generated. As shown in figure 4.6 below

```
#SVM
```

```
from sklearn.svm import SVC
svc_model = SVC(kernel = 'linear')
svc_model.fit(x_train,y_train)
```

SVC

i ?

SVC(kernel='linear')

```
y_pred = svc_model.predict(x_test)
```

```
from sklearn.metrics import confusion_matrix,classification_report,accuracy_score
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
print("Accuracy = ",accuracy_score(y_test,y_pred))
```

```
... [[23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0 20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0 26  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  1  0 20  0  0  0  0  0  0  0  0  1  0  0]
 [ 0  0  0  0  0  0  0  0  0 20  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0 11  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  1  0  0  0  0 20  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 24  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 14  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
 [ 0  0  1  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0 21  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0]
 [ 0  0  0  0  0  0  0  0  3  0  0  0  0  0  0  0  0  0  0  0 16  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19]]

      precision    recall  f1-score   support

   apple          1.00      1.00      1.00         23
   ...
  macro avg          0.98      0.98      0.98        440
  weighted avg          0.98      0.98      0.98        440

Accuracy =  0.9795454545454545
```



## Chapter 7

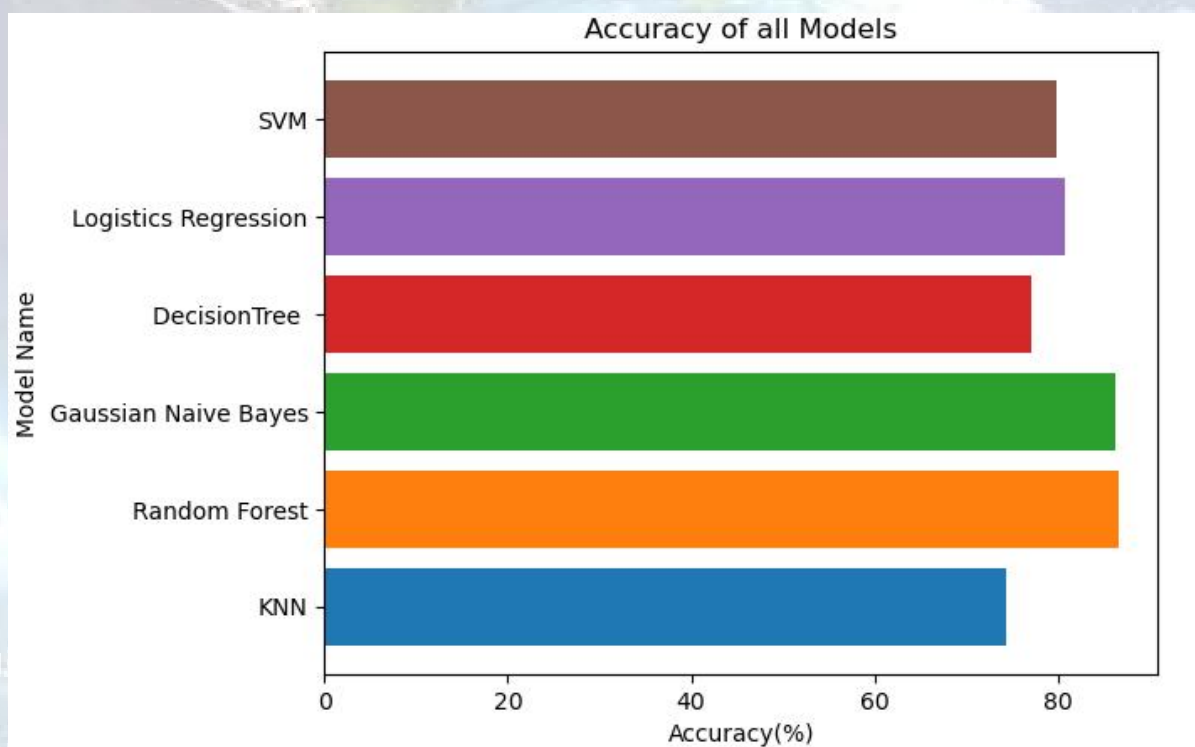
# Performance comparison of the models

Performance comparison of the models was done with the accuracy score, Gaussian Naive Bayes model is the best performing model with an accuracy score of 99.5%.

**Table 1: Accuracy of different models**

Classifier	Accuracy (%)	Training data accuracy (%)	Testing data accuracy (%)
KNN	97.04 %	78.99 %	74.32%
Random Forest	99.31 %	99.63 %	86.61 %
Gaussian Naive Bayes	99.50 %	83.74 %	86.37 %
Decision Tree	98.05 %	99.63 %	77.05 %
Logistics Regression	94.57 %	77.63 %	80.87 %
SVM	97.90 %	78.08 %	79.78 %

## Graph Comparison of All Models



## Chapter 8

# Result Analysis

The different machine learning models were trained with the crop data of 3 years and the model was used to predict the test data. Hence, the performance of all the models based on their accuracy is summarized in Table 1. The graphical comparison of accuracy of these models is shown in above

Fig.

## Chapter 9

# Conclusions and future work

This paper proposed a machine learning–based model for crop prediction aimed at recommending the most suitable crop for cultivation based on various environmental and soil parameters. The dataset used includes essential abiotic features such as nitrogen (N), phosphorus (P), potassium (K), temperature, humidity, pH, and rainfall, which are crucial for agricultural decision-making. The objective of this research was to develop a reliable and efficient crop prediction system using machine learning techniques, evaluate the performance of different models, and review existing literature on crop prediction. Related work was reviewed in Chapter 2, a crop prediction model was developed in Chapter 4 using algorithms like Random Forest, Support Vector Machine, and K-Nearest Neighbors, and the performance of these models was evaluated in Chapter 5. From the analysis, it was concluded that the **Random Forest algorithm performed the best** in terms of prediction accuracy, making it a highly suitable method for real-world agricultural applications. The model demonstrated strong potential in helping farmers make informed crop selection decisions, thereby enhancing productivity and minimizing crop failure risks.

Despite the promising results obtained, the model can be further improved in several ways. Firstly, the dataset used in this study is limited in geographic scope and does not include real-time temporal or geospatial features. Future work can involve the integration of **more diverse and region-specific datasets** to make the system adaptable to varying agro-climatic zones. Additionally, **advanced preprocessing techniques** such as feature engineering and dimensionality reduction can be employed to improve model accuracy. The system can also be expanded by incorporating **deep learning models** or ensemble methods such as **Gradient Boosting** and **XGBoost**, which are known for high predictive performance. Moreover, integrating IoT-based soil and weather sensors can enable **real-time crop recommendation systems**, making the solution dynamic and responsive. Lastly, deploying the model on a web or mobile-based application would make it easily accessible to farmers, agricultural officers, and policymakers, thus increasing its practical impact and usability.

## Limitations

- Lack of Temporal & Location specific data
- Static Dataset
- Limited Feature Dataset
- Generalized crop recommendation

## **BIBLIOGRAPHY**

- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.researchgate.net%2Ffigure%2FArchitecture-diagram-of-crop-yield-prediction-fig6-352866597&psig=AOvVaw1oAafnG-lnxZIRFZvvNYtd&ust=1751464116990000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCNDotrmm44DFQAAAAAdAAAAABAT>
- <https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset>
- <https://ieeexplore.ieee.org/document/8453550>
- <https://www.sciencedirect.com/science/article/abs/pii/S0168169919303039>
- <https://www.sciencedirect.com/science/article/abs/pii/S0168169918301370>
- [https://link.springer.com/chapter/10.1007/978-981-33-6074-1\\_52](https://link.springer.com/chapter/10.1007/978-981-33-6074-1_52)
- <https://ieeexplore.ieee.org/document/9129200>
- <https://ieeexplore.ieee.org/document/9133955>