A Project Report on

CROP YIELD PREDICTION USING SUPERVISED MACHINE LEARNING TECHNIQUES

At

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES
CHITTOOR

Submitted in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

By

G.M. SANDEEP	**19751A0522**
C.GIRI	**19751A0517**
J. GOWTHAM	**19751A0536**
M.GUNADEEP	**19751A0561**

Under the esteemed guidance of
Mrs. M. Pushpanjali, MTech.,
Assistant Professor

Department of Computer Science and Engineering
Sreenivasa Institute of Technology And Management Studies
(Affiliated to JNTU Anantapur, Anantapur)
Murukambattu, Chittoor – 517 127

April 2023

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES

CHITTOOR-517127

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project work entitled **"Crop Yield Prediction Using Supervised Machine Learning Techniques"** is carried out **by G.M. Sandeep (19751A0522), C. Giri (19751A0517), J. Gowtham(19751A0536), M. Gunadeep(19751A0561)** under my supervision and guidance during the academic year 2022-2023 and the best of our knowledge is original work.

Mrs.M. Pushpanjali, MTech.,
Assistant Professor
INTERNAL GUIDE

Dr. M. Arthi, MTech, PhD
Associate Professor
Head of the Department

Submitted for Viva-voce examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES(Autonomous)

Institute Vision

To emerge as a Centre of Excellence for Learning and research in the domains of Artificial Engineering, Machine Learning, Data Science.

Institute Mission

- Provide congenial academic ambience with state art of resources for learning and research.
- Ignite the students to acquire self-reliance in the latest technologies.
- Un leash and encourage the in at potential and creativity of students.
- Inculcate confidence to face and experience new challenges.
- Faster enterprising spirit among students.
- Work collaboratively with technical Institutes /Universities / Industries of National and International repute.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION

To contribute for the society through excellence in Computer Science and Engineering with a deep passion for wisdom, culture and values.

MISSION

M1: Provide congenial academic ambience with necessary infrastructure and learning resources.

M2: Inculcate confidence to face and experience new challenges from industry and society.

M3: Ignite the students to acquire self-reliance in State-of-the-Art Technologies.

M4: Foster Enterprising spirit among students.

PROGRAMME EDUCATIONAL OBJECTIVES(PEOs):

Graduates of Computer Science and Engineering shall

PEO1: Excel in Computer Science and Engineering program through quality studies, enabling success in computing industry. **(Professional Competency)**

PEO2: Surpass in one's career by critical thinking towards successful services and growth of the organization, or as an entrepreneur in higher studies. **(Successful Career Goals)**

PEO3: Enhance knowledge by updating advanced technological concepts for facing the rapidly changing world and contribute to society through innovation and creativity. **(Continuing Education and Contribution to Society)**

PROGRAM SPECIFIC OUTCOMES(PSO's):

PSO1: Have Ability to understand, analyze and develop computer programs in the areas like algorithms, system software, web design, big data analytics, and networking.

PSO2: Deploy modern computer languages, environment, and platforms in creating innovative products and solutions.

PROGRAMME OUTCOMES (PO's)

Computer Science and Engineering Graduates will be able to:

PO1 - Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.

PO2-Problem analysis: Identify formulate research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3-Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.

PO4 - Conduct investigations of complex problems: Use researchbased knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 - Modern tool usage: Ability to design and develop hardware and software in emerging technology environments like cloud computing embedded products, real-time systems, Internet of Things, Big Data etc.

PO6- Engineering and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7-Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8- Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 - Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multi-disciplinary settings.

PO10-Communication: Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11-Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12-Life-long learning: Basic Knowledge in hardware/software methods and

tools for solving Real-life and R&D problems with an orientation to lifelong learning.

Course Outcomes for Project Work:

On completion of project work we will be able to,

CO1. Demonstrate in-depth knowledge on the project topic.

CO2. Identify, analyze and formulate complex problem chosen for project work to attain substantiated conclusions.

CO3. Design solutions to the chosen project problem.

CO4. Under take investigation of project problem to provide valid conclusions.

CO5. Use the appropriate techniques, resources and modern engineering tools necessary for project work.

CO6. Apply project results for sustainable development to the society.

CO7. Understand the impact of project results in the context of environmental sustainability.

CO8. Understand professional and ethical responsibilities while executing the project work.

CO9. Function effectively as individual and a member in the project team.

CO10. Develop communication skills, both oral and written for preparing and presenting project report.

CO11. Demonstrate knowledge and understanding of cost and time analysis required for carrying out the project.

CO12. Engage in lifelong learning to improve knowledge and competence in the chosen area of the project.

CO-MAPPING

COs \POs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	√											
CO2		√										
CO3			√									
CO4				√								
CO5					√							
CO6						√						
CO7							√					
CO8								√				
CO9									√			
CO10										√		
CO11											√	
CO12												√

Evaluation Rubrics for Project work

Rubric (CO)	Excellent (Wt=3)	Good (Wt= 2)	Fair (Wt= 1)
Selection of Topic (CO1)	Select a latest to pic through complete knowledge of facts and concepts.	Select a topic through partial knowledge of facts and concepts.	Select a to pic through improper knowledge of factsand concepts.
Analysis and Synthesis (CO2)	Thorough comprehension through analysis/ synthesis.	Reasonable comprehension through analysis/ synthesis.	Improper comprehensionthrough analysis/ synthesis.
Problem Solving (CO3)	Thorough comprehension about what is proposed in the literature papers.	Reasonable comprehension about what is proposed in the literature papers.	Improper comprehensionabout what is proposed in the literature.
Literature Survey (CO4)	Extensive literature Survey with standard references.	Consider able literature survey with standard references.	In complete literature surveywith substandard references.
Usage of Techniques& Tools (CO5)	Clearly identified and has complete knowledge of techniques & tools used in the project work.	Identified and has sufficient knowledge of techniques &tools used in the project work.	Identified and has inadequate knowledge of techniques & toolsused in project work.
Project work impact on Society (CO6)	Conclusion of project work has strong impact on society.	Conclusion of project work has considerable impact on society.	Conclusion of project work hasfeeble impact onsociety.
Project work impact on Environment (CO7)	Conclusion of project work has strong impact on Environment.	Conclusion of project work has considerable impact on environment.	Conclusion of project work hasfeeble impact onenvironment.

Ethical attitude (CO8)	Clearly understand sethical and social practices.	Moderate understanding of ethical and social practices.	Insufficient understanding of ethical and social practices.
Independent Learning (CO9)	Did literature survey and selected topic with a little guidance	Did literature survey and selected topic with considerable guidance	Selected a topic as suggested by the supervisor
Oral Presentation (CO10)	Presentation in logical sequence with key points, clear conclusion and excellent language	Presentation with key points, conclusion and good language	Presentation with insufficient key points and improper conclusion
Report Writing (CO10)	Status report with clear and logical sequence of chapters using excellent language	Status report with logical sequence of chapters using understandable language	Status report not properly organized
Time and Cost Analysis (CO11)	Comprehensive time and cost analysis	Moderate time and cost analysis	Reasonable time and cost analysis
Continuous learning (CO12)	Highly enthusiastic towards continuous learning	Interested in continuous learning	Inadequate interest in continuous learning

DECLARATION

I affirm that the project work titled **"Crop Yield Prediction Using Supervised Machine Learning Techniques"** being submitted in partial fulfilment for the award of **Bachelor of technology** is the original work carried out by me. It has not formed the part of any other project work submitted for award of any degree, either in this or any other university.

G.M. SANDEEP	**19751A0522**
C.GIRI	**19751A0517**
J. GOWTHAM	**19751A0536**
M.GUNADEEP	**19751A0561**

I certify that the declaration made above by the candidate is true.

Mrs.M. Pushpanjali, MTech.,
Assistant Professor

ABSTRACT

Agriculture provides most of the world's food and fabrics. Cotton, wool, and leather are all agricultural products. Agriculture also provides wood for construction and paper products. These products, as well as the agricultural methods used, may vary from one part of the world to another. In general, agriculture is the backbone of India and also plays an important role in the Indian economy. Now-a-days, food production and prediction are getting scarcity due to unnatural climatic changes, which will adversely affect the economy of farmers by getting a poor yield and also help the farmers to remain less familiar with forecasting the future crops. This research work helps the beginner farmer in such a way to guide them for reasonable crops by deploying machine learning, one of the advanced technologies in crop prediction. Supervised machine learning algorithm puts in the way to achieve it. The seed data of the crops are collected here, with the appropriate parameters like nitrogen, phosphorus, temperature, humidity, pH and rainfall content, which helps the crops to achieve successful growth. To prevent this problem, Agricultural sectors have to predict the crop from a given dataset using machine learning techniques. Also we are calculating the performance metrics.

ACKNOWLEDGEMENT

I am grateful to our honourable founder Late Sri **D.K. ADIKESAVULU NAIDU Garu,** chairperson Smt. **D.A. SATHYA PRABHA Garu,** for providing all the facilities in the college.

I would like to express the heartfelt gratitude to our Management, Chairperson **Sri. K. RANGANATHAM GARU** for providing excellent institutional support to complete this project.

I would like to express the profound gratitude to our beloved principal **Dr.N. VENKATACHALAPATHY, M.Tech., PhD., PGDPE(CIPET)., PGDI RPM., F.I.E.,** for giving us the opportunity to embark upon this course and make use of the facilities available in the college.

I pay my sincere note of felicitation to **Mrs.M. ARTHI, M.Tech**, Associate Professor & Head of the CSE Department for providing excellent departmental support as well as motivation which enhance the thrust to complete the project.

I thank my project guide **Mrs.M. PUSHPANJALI, M.Tech,** for her valuable guidance, support and offering necessary material and willingly giving us advice whenever needed.

Finally, I thank all the teaching faculty and non-teaching staff of the Department of Computer Science and Engineering for giving their advices and their help during the course of our project work as well as studies.

G.M. SANDEEP **19751A0522**
C.GIRI **19751A0517**
J. GOWTHAM **19751A0536**
M.GUNADEEP **19751A0561**

TABLE OF CONTENTS

Title **Page No.**

ABSTRACT **11**

ACKNOWLEDGEMENT **12**

TABLE OF CONTENTS **13**

LIST OF FIGURES **15**

ABBREVIATIONS……..…….**16**

NOTATIONS…….……**17**

CHAPTER 1 : INTRODUCTION ……...…...……**18**

1.1 Introduction to the Topic…..... 18

1.2 Motivation…….……..19

1.3 Objectives……..……..……19

1.4 Organization of thesis……..……20

CHAPTER 2 : LITERATURE REVIEW ……..…….**21**

CHAPTER 3 : SYSTEM ANALYSIS …….……..**24**

3.1 Existing Methods ……..…….24

3.2 Proposed Methods ……...……28

3.3 Feasibility Study……...……...29

CHAPTER 4 : SYSTEM SPECIFICATIONS……...…….**30**

4.1 Software Requirements……...……...……..30

4.2 Hardware Requirements……...…….....30

CHAPTER 5 : SOFTWARE DESCRIPTION……..……...……...**31**

CHAPTER 6 : PROJECT DESCRIPTION…….**32**

6.1 Problem Statement……32

6.2 Design Architecture……..32

6.3 Process of Dataflow……..34

6.4 UML Diagrams…….36

A Usecase Diagram…….36

B Class Diagram…….37

C Activity Diagram…….38

D Sequence Diagram……..…39

E Entity Relationship Diagram……..…..40
6.5 List of Modules……..…….41
6.6 Module Description……...……..41
CHAPTER 7: CONCLUSION AND FUTURE ENHANCEMENTS……..…….55
7.1 Conclusions…….…….55
7.2 Future Enhancements…….……..56
CHAPTER 8: APPENDIX……...…….57
8.1 Source Code…….57
8.2 Screenshots……...73
CHAPTER 9: REFERENCES ……75

LIST OF FIGURES

Figure no.	Title	Page No.
Figure 3.2.1	Process of the model	28
Figure 6.2.1	System Architecture	32
Figure 6.3.1	Process of a dataflow diagram	34
Figure 6.4.1	Use case diagram of the model	36
Figure 6.4.2	Class diagram of the model	37
Figure 6.4.3	Activity diagram of the model	38
Figure 6.4.4	Sequence diagram of the model	39
Figure 6.4.5	Entity relationship diagram of the model	40
Figure 6.6.1	Accuracy graph of naïve bayes algorithm	47
Figure 6.6.2	Confusion matrix of naïve bayes algorithm	48
Figure 6.6.3	Accuracy graph of random forest classifier	49
Figure 6.6.4	Confusion matrix of random forest classifier	49
Figure 6.6.5	Accuracy graph of decision tree classifier	51
Figure 6.6.6	Confusion matrix of decision tree classifier	51
Figure 6.6.7	Accuracy graph of logistic regression	53

ABBREVIATIONS

TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
TPR	True Positive Rate
FPR	False Positive Rate
SVM	Support Vector Machine
DT	Decision Tree
RF	Random Forest
LR	Logistic Regression

NOTATIONS

X	Input Variables
Y	Discrete Output Variables
n	No. of Observations
p	No. of Variables

CHAPTER 1: INTRODUCTION

1.1. INTRODUCTION TO THE TOPIC

Crop yield prediction is one of the challenging tasks in agriculture. It plays an essential role in decision-making at global, regional, and field levels. The prediction of crop yield is based on soil, meteorological, environmental, and crop parameters. Decision support models are broadly used to extract significant crop features for prediction. Precision agriculture focuses on monitoring (sensing technologies), management information systems, variable rate technologies, and responses to inter- and intravariability in cropping systems. The benefits of precision agriculture involve increasing crop yield and crop quality, while reducing the environmental impact.

Agriculture is an essential sector for providing food and fabrics to the world, with cotton, wool, and leather as some of the major agricultural products. However, the agricultural methods and products used can vary significantly from one region to another. In India, agriculture plays a vital role in the economy, with food production being a key aspect of it. Unfortunately, with unnatural climatic changes, predicting crop yield has become increasingly challenging. This unpredictability adversely affects the farmers by resulting in poor yields and a lack of knowledge in forecasting future crops. To address this issue, this research aims to guide beginner farmers by using advanced machine learning techniques for crop prediction. Specifically, supervised machine learning algorithms are employed to predict crop yield by analyzing the seed data with parameters such as nitrogen, phosphorus, temperature, humidity, pH, and rainfall content. By leveraging these algorithms, the agricultural sector can accurately predict crop yields and take necessary measures to prevent the adverse effects of unpredictable weather conditions. Additionally, the project also calculates performance metrics to evaluate the effectiveness of the algorithm. Overall, this project aims to provide a comprehensive solution to the challenges faced by farmers in predicting crop yield in India

The use of advanced technology, such as machine learning, has become increasingly popular in the agriculture sector to aid in the prediction of crop yields. By using supervised machine learning algorithms, this research work aims to achieve reasonable crop predictions. These algorithms are trained using historical data of crop yields and the corresponding environmental factors that affect crop growth. Once trained, the algorithms can be used to

predict crop yields for new data inputs, which can be used to make informed decisions for farming practices.

In this project, we focus on six critical environmental factors that affect crop growth, namely nitrogen, phosphorus, temperature, humidity, pH, and rainfall content. The dataset for this research is collected from various sources and contains information on various crop types, along with their corresponding environmental factors. We use this dataset to train the supervised machine learning algorithms, which can predict crop yields with high accuracy. Additionally, the performance of the algorithm is evaluated using various performance metrics such as accuracy, precision, recall, and F1 score. These metrics help to assess the effectiveness of the algorithm and identify areas for improvement. By using these metrics, we can make necessary changes to the algorithm to improve its accuracy in predicting crop yields.

Overall, this research work aims to provide a comprehensive solution to the challenges faced by farmers in predicting crop yields. By leveraging the power of machine learning, we can provide accurate and reliable crop yield predictions, which can help farmers make informed decisions about their farming practices. The findings of this research work will be valuable for researchers, farmers, and policymakers, contributing towards the development of the agriculture sector in India.

1.2 MOTIVATION

- Helps farmers make better decisions and improve crop yields.
- Provides more accurate and reliable predictions of crop yields.
- Reduces economic losses and improves the livelihoods of the farmers.
- Enables farmers to plan their farming activities more effectively and efficiently.
- Facilitates the adoption of sustainable farming practices.
- Advances the field of agriculture and machine learning research.

1.3 OBJECTIVES

- To develop a crop yield prediction model that uses supervised machine learning algorithms to accurately predict crop yield based on six critical environmental factors

- Evaluate the performance of the developed model by comparing it with traditional methods of crop yield prediction.
- Provide farmers with an easy-to-use tool that predicts crop yields and informs their farming decisions.
- Help farmers plan their farming activities more efficiently and reduce economic loss

1.4 ORGANIZATION OF THESIS

In **Chapter 1**, we have discussed the introduction of Crop Yield Prediction Using Supervised Machine Learning and how it works in general, the problem statement, Motivation and objectives of the proposed system.

In **Chapter 2**, we have discussed the literature survey and the introduction to each of the literature surveys.

In **Chapter 3**, we have discussed the existing methods for the problem. We also discussed the proposed system, UML Diagrams, its algorithm implementation and its explanation.

In **Chapter 4**, we have discussed the results obtained after executing the proposed system.

In **Chapter 5**, we have discussed the conclusions and future work which are needed to be extended and optimized.

CHAPTER 2: REVIEW OF LITERATURE

Mkhabela and Bullock [1] examined the potential use of Normalised Difference Vegetation Index (NDVI) data derived from the Moderate Resolution Imaging Spectro Radiometer (MODIS) sensor to forecast crop yields on the Canadian Prairies. Using growing season NDVI data from 2000 to 2006 and crop yield data for barley, canola, field peas and spring wheat, correlation and regression analyses were conducted to identify the best time for making reliable crop yield predictions. Results showed that MODIS-NDVI data can be used effectively to forecast crop yields in the region.

Becker-Reshef and Vermote [2] used remote sensing and crop statistics data to develop an empirical approach to forecast wheat yields. They combined a new BRDF-corrected, daily surface reflectance dataset developed from NASA's Moderate resolution Imaging Spectro-radiometer (MODIS) with official crop statistics to develop a regression-based model for forecasting winter wheat production in Kansas. This model was then applied to forecast winter wheat production in Ukraine. This study demonstrated the potential of using remote sensing data for crop yield forecasting, which can help ensure global food security.

Bolton and Friedl [3] utilized data from NASA's Moderate Resolution Imaging Spectro-radiometer (MODIS) and county-level data from the United States Department of Agriculture (USDA) to develop models predicting maize and soybean yields in the Central United States. They found that the MODIS Enhanced Vegetation Index 2 (EVI2) was more effective than the Normalized Difference Vegetation Index (NDVI) in predicting maize yields. In addition, including crop phenology information from MODIS improved model performance. The researchers also found that correlations between vegetation indices and yield were highest 65-75 days after greenup for maize and 80 days after greenup for soybeans. The study demonstrated the potential for MODIS data to be used in crop yield forecasting.

Sabareeswaran and Gunasundari [4] proposed a plant yield prediction model based on Firefly optimization algorithm for feature selection and modified fuzzy cognitive map algorithm for predicting the growth of plant yield. The model utilized plant images, soil characteristics, and weather factors as input features, and the predicted outcome was transmitted to farmers through smartphones to aid in improving harvesting. Experimental results showed that the

proposed technique was effective in predicting plant yield and could be compared with other prediction techniques.

Doraiswamy et al. [5] proposed a method to integrate satellite remotely sensed data with the crop model EPIC (Erosion Productivity Impact Calculator) for simulations at regional scales to monitor crop condition and production estimates at the state and county level. The study was conducted in the semi-arid region of North Dakota, and the primary objective was to evaluate the use of remotely sensed parameters as an input to a crop growth model to simulate spring wheat yields at the sub-county and county levels. A radiative transfer model, SAIL (Scattered by Arbitrary Inclined Leaves), provided the link between the satellite data and crop model. The model was calibrated using Landsat data over three southeast counties in North Dakota and was used to simulate crop yields for the state of North Dakota with inputs derived from NOAA AVHRR data. The study demonstrated the potential of integrating satellite remotely sensed data with crop models for real-time monitoring of crop conditions and yield estimates.

Miao et al. [6] developed a method for predicting crop yield using a crop growth model and weather data. The authors used the Decision Support System for Agrotechnology Transfer (DSSAT) to simulate maize and soybean yield in the midwestern United States. They found that using weather data to drive the crop model provided accurate yield predictions for both crops. The study demonstrated the potential for using weather data and crop models for yield prediction at regional scales.

Hernández-Suárez et al. [7] used remote sensing data to map sugarcane yield in Cuba. The authors used Landsat data and a vegetation index to estimate sugarcane yield. They found that the vegetation index was highly correlated with sugarcane yield, and the approach was useful for monitoring yield over large areas. The study demonstrated the potential for remote sensing data to provide timely and accurate information for crop management and decision-making.

Lobell et al. [8] investigated the relationship between remotely sensed vegetation indices and wheat yield in the Great Plains region of the United States. The authors used MODIS data to estimate wheat yield and found that vegetation indices were strongly correlated with yield. The study demonstrated the potential for using remote sensing data to monitor crop growth and yield at regional scales.

Peng et al. [9] developed a method for estimating rice yield using remote sensing data and machine learning techniques. The authors used MODIS data and a support vector machine (SVM) algorithm to predict rice yield in China. They found that the SVM algorithm provided accurate yield predictions, and the approach was useful for monitoring yield over large areas. The study demonstrated the potential for combining remote sensing data and machine learning techniques for yield prediction at regional scales.

Lu et al. [10] used remote sensing data to develop a framework for mapping and estimating rice yield at a regional scale in Cambodia. The study utilized Landsat, MODIS, and Sentinel-1 data, and the results demonstrated the potential for remote sensing to provide timely and accurate information for crop yield assessment.

Lopes et al. [11] proposed a method to estimate soybean yield using machine learning techniques and remote sensing data. The study was conducted in Brazil, and the authors found that integrating Landsat and MODIS data with machine learning algorithms improved the accuracy of yield estimates.

Chen et al. [12] developed a method for mapping maize yield at a regional scale using remote sensing data and crop modelling. The study was conducted in a county in the United States, and the results demonstrated the potential for integrating remote sensing data with crop models for real-time monitoring of crop conditions and yield estimates.

Mubiru et al. [13] investigated the potential for integrating remote sensing data with crop modelling to estimate maize yield in Uganda. The study utilized Landsat and MODIS data, and the authors found that the combination of remote sensing and crop modeling improved the accuracy of yield estimates compared to using either method alone.

Wang et al. [14] proposed a method for estimating wheat yield using remote sensing data and the light use efficiency model. The study was conducted in a county in China, and the results demonstrated the potential for remote sensing to provide timely and accurate information for crop yield assessment.

Saifullah et al. [15] investigated the potential for using remote sensing data to estimate cotton yield in Pakistan. The study utilized Landsat and MODIS data, and the authors found that integrating remote sensing data with machine learning algorithms improved the accuracy of yield estimates.

Ebrahimian et al. [16] proposed a method for estimating wheat yield using machine learning algorithms and remote sensing data. The study was conducted in Iran, and the authors found that integrating Landsat and MODIS data with machine learning algorithms improved the accuracy of yield estimates.

CHAPTER 3: SYSTEM ANALYSIS

3.1. EXISTING METHODS

Active microwave remote sensing has been shown to be an effective tool for providing crucial information on crop morphology and conditions, which can be used to support agronomic management at different scales. The use of synthetic aperture radar (SAR) backscatter data at different frequencies has been extensively studied over the past two decades, with the aim of filling knowledge gaps towards operational use in agriculture.

One of the challenges of using SAR backscatter data for agricultural applications is the complex nature of the backscatter signal, which is influenced by a range of environmental and crop-specific factors. The method described in the base paper aims to address this challenge by using variance-based global sensitivity analysis (GSA) to investigate the sensitivity of X-band backscattering to agronomic and morphological features of two different crops: maize and rice.

To conduct the study, the researchers combined empirical data on crop status and growth with high-resolution TerraSAR-X (TSX) data and microwave radiative transfer model (RTM) simulations. The approach allowed them to quantify the contributions of different scattering mechanisms for the two crops under varying observation setups, assess the sensitivity of X-band backscattering to morphostructural crop biophysical parameters (BPs) (and their interactions), and evaluate the effects of crop biomass on backscatter across growth stages.

The results of the study showed that the sensitivity of X-band backscattering to crop-specific features varied throughout the growing season, and that different scattering mechanisms contributed to the overall backscatter signal at different growth stages. For example, in maize, the effects of crop biomass on backscatter were found to be most pronounced during the vegetative growth stage, whereas in rice, the sensitivity of X-band backscattering to

morphostructural crop BPs was found to increase during the reproductive stage.

The researchers used multidimensional GSA outputs that accounted for model input correlations through Shapley effects to provide a comprehensive suite of information on the relative proportion of total backscatter variance explained by a range of parameters. The approach allowed them to quantitatively describe the different behavior in vertical and horizontal polarization (changing throughout plant growth) in paddy rice, and the mixed contribution of canopy density and leaf angle distribution (depending on the incident angle) in maize.

Overall, the study demonstrated the potential of using global sensitivity analysis as a tool for investigating the sensitivity of X-band backscattering to agronomic and morphological features of crops, and provided valuable insights into the behavior of the backscatter signal in different crops and growth stages. However, the approach described in the base paper may be limited by the need for specialized data sources and equipment, which may limit its accessibility to researchers and agronomists who do not have access to these resources.

Another potential limitation of the approach is the need for extended data collection and analysis periods. This may limit its suitability for real-time crop monitoring, which is an important aspect of agronomic management. However, the approach may be useful for retrospective analysis of crop performance and for guiding management decisions for future growing seasons.

One advantage of the method described in the base paper is that it provides a detailed and quantitative understanding of the relationship between SAR backscatter data and crop-specific features, which can help to inform decisions about agronomic management practices. For example, the approach could be used to identify which morphostructural crop BPs are most important for predicting crop yield, and to develop strategies for optimizing these features.

The approach could also be used in combination with other agricultural technologies, such as precision agriculture tools and unmanned aerial vehicles (UAVs), to provide a comprehensive view of crop performance and to guide management decisions at different scales. The use of SAR backscatter data could help to identify areas of the field.

The main outcomes of this study are as follows:

1. The researchers found that the development of a second-order model, incorporating interactions among canopy elements, is needed to correctly interpret X-band SAR data on maize, while a single-order model could be sufficient on rice. However, the developed model is unable to correctly simulate the SAR data in very dense rice canopies (>900 stems m−2), due to overestimation of vertical attenuation.

2. Phenology-informed model simulations based on crop biomass parameters measured in situ allowed the researchers to assess the main scattering mechanisms for rice and maize and to separate the relative contribution of various canopy elements under different polarizations and observation setups.

3. The researchers used multidimensional GSA, explicitly accounting for model input correlations through Shapley effects, to provide information about the relative proportion of total backscatter variance explained by different crop parameters, revealing the different behavior of rice in vertical and horizontal polarizations and their changes throughout the plant growth cycle and the mixed contribution of canopy density and leaf angle distribution for maize, strongly depending on the incident angle.

4. The study found that extending model simulations with empirical allometric estimation of rice and maize biomass (on area basis) allows highlighting the capabilities of X-band SAR in estimating standing crop biomass. The researchers suggested that when canopy attenuation is a relevant factor (0.15–1.10 kg m−2), HH polarization ($\theta = 20^\circ$–40°) can be used for rice biomass estimation without approaching lower detection limits (NESZ), with a sensitivity over 0.8 dB per 100 g m−2, confirmed by real case TSX data paddy fields in northern Italy. However, outside early growth phases, the competition of leaf scattering and canopy attention prevents the use of a similar relationship in maize, with sensitivity reduced to less than half for rice, that is, around 0.3 dB every 100 g m−2 (HH polarization).

5. Although the choice of not including ears into the RTM constitutes a theoretical

limitation of the findings in terms of sensitivity, the approach is nevertheless putting forward the knowledge of X-band crop backscatter drivers within a large part of the growing season and could easily be refined with future inclusion of future microwave modeling advancements.

6. Further advancements could focus on refining the RTM over rice tackling the underestimation of σ° certain canopy conditions, for example, by taking into account stem clusterization effects and reliable simulations of rice ears (and grains), extending the approach to other major crops (e.g., soybean, wheat), and running the sensitivity analysis at additional frequencies (C down to P band).

In conclusion, this study provides valuable insights into the sensitivity of X-band SAR to agronomic and morphological characteristics of rice and maize, highlighting the need for different models for different crops and the potential for using X-band SAR in estimating standing crop biomass. These findings can have implications for precision agriculture and crop management strategies. However, further research is needed to refine the RTM and extend the approach to other major crops and frequencies.

3.2. PROPOSED METHODS

The proposed system will predict the most suitable crop for particular land based on soil contents and weather parameters such as Temperature, Humidity, Soil PH, Nitrogen, Phosphorus, and Rainfall. Data collection is the most efficient method for collecting and measuring data from different resources. After collecting datasets from various resources, the dataset must be preprocessed before training to the model. The data pre-processing can be done in various stages, beginning with reading the collected dataset the process continues to data cleaning. In data cleaning the datasets contain some redundant attributes, those attributes are not considered for crop prediction. So, we have to drop unwanted attributes and datasets containing some missing values. we need to drop these missing values or fill them with unwanted nan values in order to get better accuracy. Statistical algorithms and machine learning techniques to identify future outcomes based on historical data. The goal is to go beyond knowing what has happened to provide the best assessment of what will happen in the future. In our system, we used a supervised machine learning algorithm decision-making as classification and regression. The classification algorithm will be most suitable for our system.

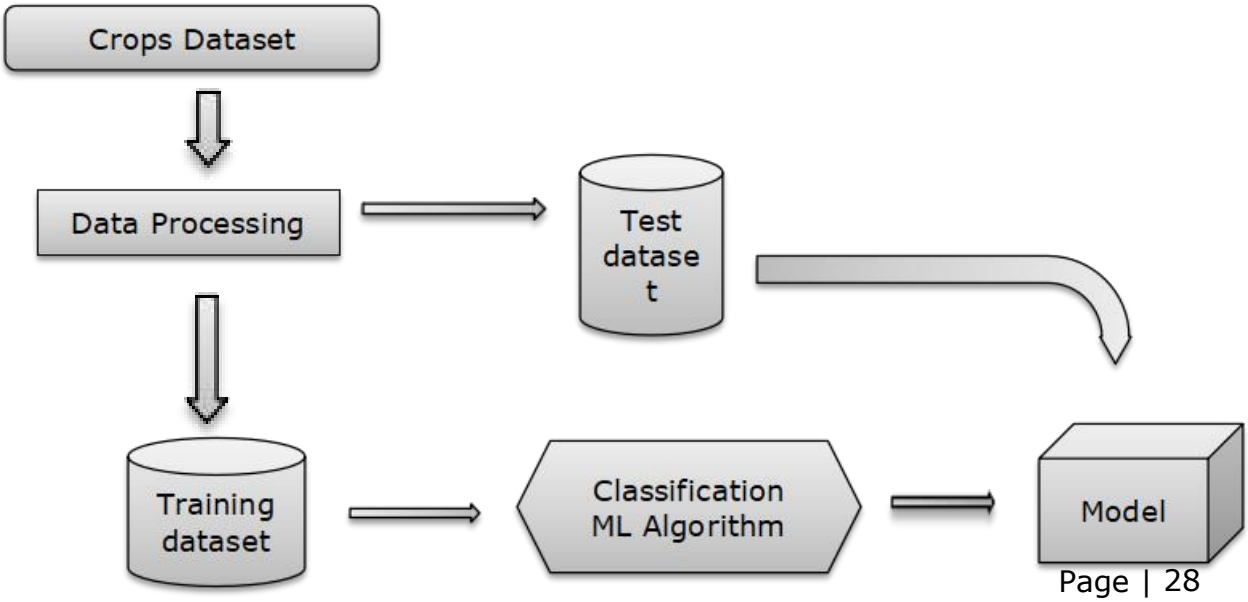


Fig 3.2.1 : Process of the model

3.3. Feasibility study:

The crop yield prediction using supervised machine learning algorithms project is a potentially valuable tool for farmers and agricultural researchers. Before proceeding with the project, a feasibility study was conducted to determine whether the project is practical and worth pursuing. The feasibility study considered several factors, including technical feasibility, economic feasibility, and operational feasibility.

3.3.1.Technical feasibility:

The project requires the use of machine learning algorithms, specifically Random Forest and Naive Bayes, to accurately predict crop yields. These algorithms have been well-studied and are widely used in many industries. There are several machine learning libraries available in various programming languages that make the implementation of these algorithms relatively straightforward. Therefore, the project is considered technically feasible.

3.3.2.Economic feasibility:

The cost of collecting and preprocessing the dataset, as well as the cost of implementing the machine learning algorithms, must be considered. The cost will depend on factors such as the size of the dataset and the computing resources required to run the algorithms. However, the potential benefits of the project, such as increased crop yields and reduced waste, could outweigh the costs. Therefore, the project is considered economically feasible.

3.3.3.Operational feasibility:

The project requires collaboration between agricultural researchers, farmers, and machine learning experts. The success of the project will depend on the willingness and ability of these stakeholders to work together and share data and knowledge. Therefore, the project is considered operationally feasible, but strong communication and collaboration will be essential for success.

CHAPTER 4: SYSTEM SPECIFICATIONS

4.1. Software Requirements:

Operating System : Windows 10 or later

Tool : Anaconda with Jupyter Notebook

4.2. Hardware requirements:

Processor : Intel i3

Hard disk : minimum 10 GB

RAM : minimum 4 GB

CHAPTER 5: SOFTWARE DESCRIPTION

General:

Requirements are the basic constrains that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environment requirements
    A. Hardware requirements
    B. software requirements

1. Functional requirements:

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like sk-learn, pandas, numpy, matplotlib and seaborn.

2. Non-Functional Requirements:

Process of functional steps,

1. Problem define
2. Preparing data
3. Evaluating algorithms
4. Improving results
5. Prediction the result

CHAPTER 6: PROJECT DESCRIPTION

6.1 PROBLEM STATEMENT

Farmers often struggle to predict crop yields accurately, resulting in inefficient farming practices and economic losses. This research aims to solve this problem by developing a crop yield prediction model that uses supervised machine learning algorithms and six critical environmental factors to predict crop yields accurately.

6.2 DESIGN ARCHITECTURE

The design architecture for crop yield prediction involves several key components including:

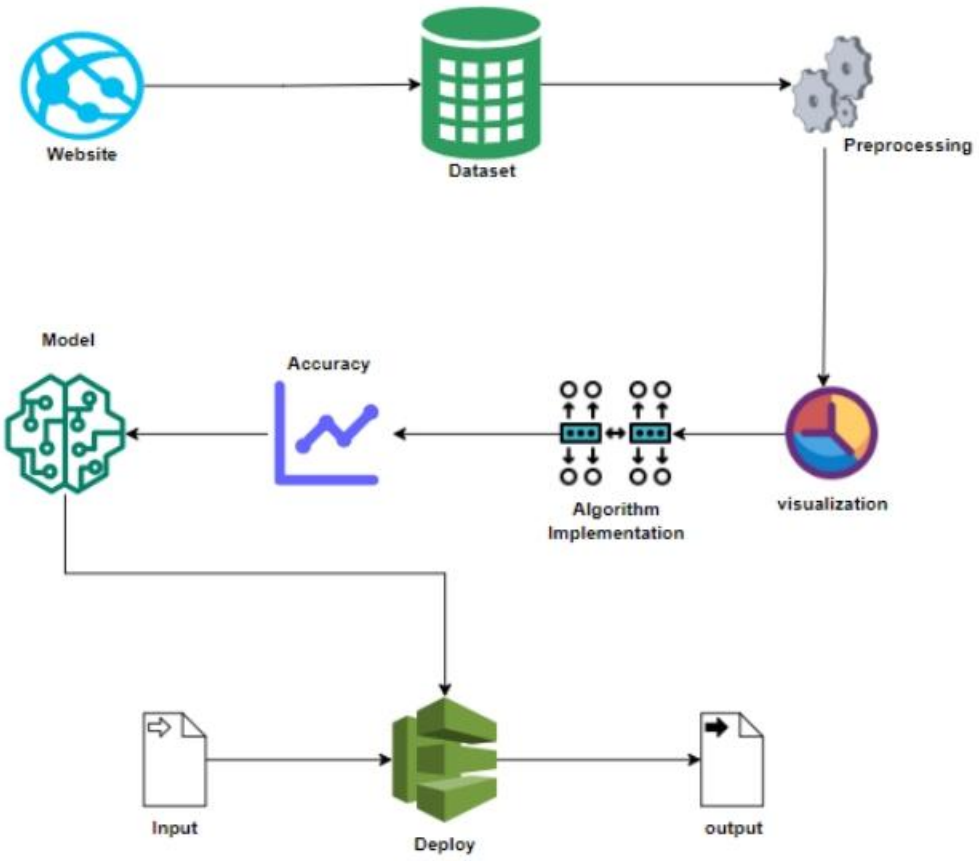


Fig 6.2.1: System Architecture

Website: The website is the user interface that enables users to input data and receive predictions. The website will be designed to be user-friendly and intuitive, with clear

instructions and feedback.

Dataset: The dataset is the collection of data that will be used to train the machine learning algorithm. The dataset will be sourced from a variety of sources, including weather data, soil data, and crop yield data.

Preprocessing: The preprocessing component involves transforming the raw data into a format that can be used by the machine learning algorithm. This may involve cleaning the data, handling missing values, and scaling features

Visualization: The visualization component involves creating visual representations of the data to help users understand the patterns and trends in the data. This may involve creating charts, graphs, and maps.

Algorithm Implementation: The algorithm implementation component involves selecting and implementing the machine learning algorithm that will be used to make predictions. This may involve selecting a regression or classification algorithm, choosing between linear or non-linear models, and selecting hyperparameters.

Accuracy: The accuracy component involves evaluating the performance of the machine learning algorithm. This may involve using metrics such as mean squared error or root mean squared error to assess the accuracy of the model.

Deploy: The final component involves deploying the model to the website, allowing users to input new data and receive predictions. This may involve creating a REST API that can be called by the website, or integrating the model directly into the website's backend.

By following this design architecture, we can create a powerful and accurate crop yield prediction system that can help farmers optimize their yields and improve their profitability.

6.3 PROCESS OF DATAFLOW:

The process of dataflow involves the following:

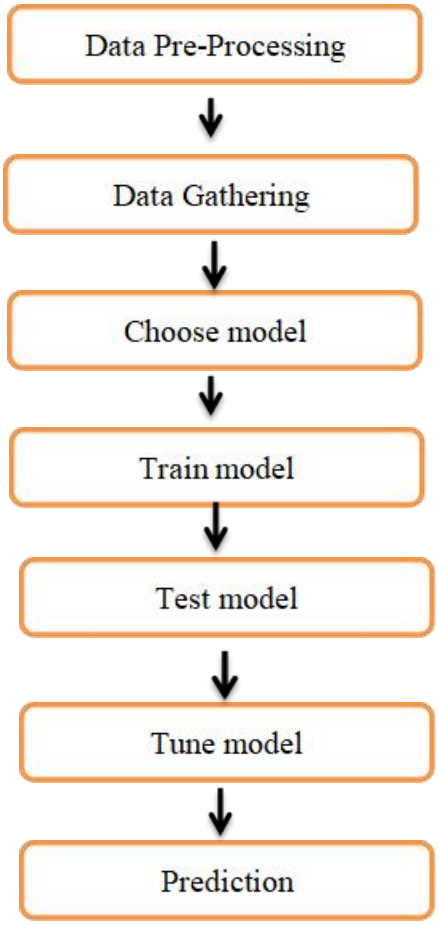


Fig 6.3.1 :Process of dataflow diagram

Data Gathering: The first step is to gather relevant data. This may involve collecting information on weather patterns, soil characteristics, crop types, fertilizer use, and other factors that may impact crop yields.

Data Preprocessing: Once you have collected the data, you will need to preprocess it to make it suitable for use in machine learning algorithms. This may involve cleaning the data, handling missing or invalid values, scaling features, and encoding categorical variables.

Choose Model: After preprocessing the data, you will need to select an appropriate machine learning algorithm for your task. This may involve choosing between supervised and unsupervised learning, selecting a regression or classification algorithm, and choosing between linear or non-linear models.

Train Model: With your data preprocessed and model selected, you will need to train the model on your dataset. This involves feeding the data into the model and adjusting its parameters to minimize the difference between predicted and actual crop yields.

Test Model: Once you have trained the model, you will need to test its performance on a separate dataset. This will help you evaluate the model's ability to generalize to new data and make accurate predictions.

Tune Model: Depending on the results of the testing phase, you may need to adjust the model's parameters or try different algorithms to improve its performance. This may involve tweaking the learning rate, adjusting regularization, or changing the model architecture.

Prediction: After you have fine-tuned your model, you can use it to make predictions on new data. This may involve integrating it into a web application, using it to generate reports, or using it to guide decision-making on the farm.

6.4. UML DIAGRAMS:

A) Use Case Diagram:

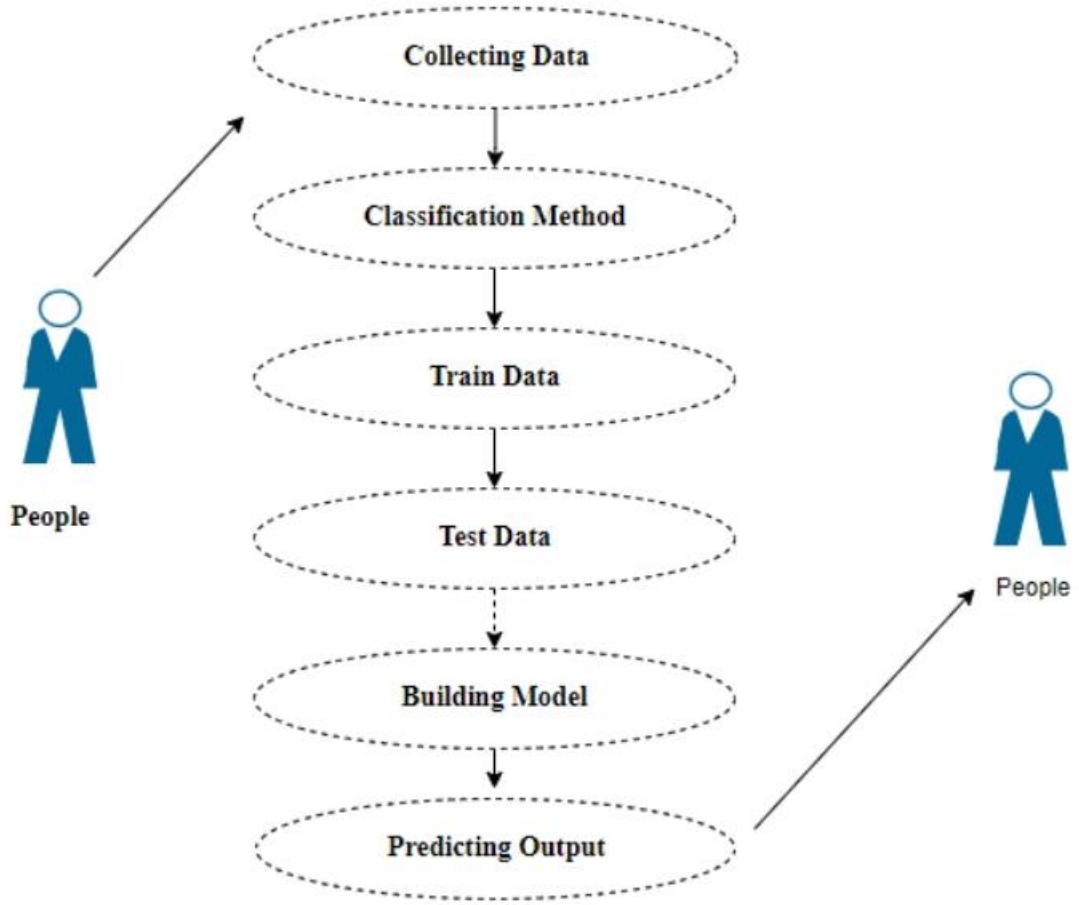


Fig 6.4.1: Use case diagram of the model

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analysed the functionalities are captured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner.

B) **Class Diagram**:

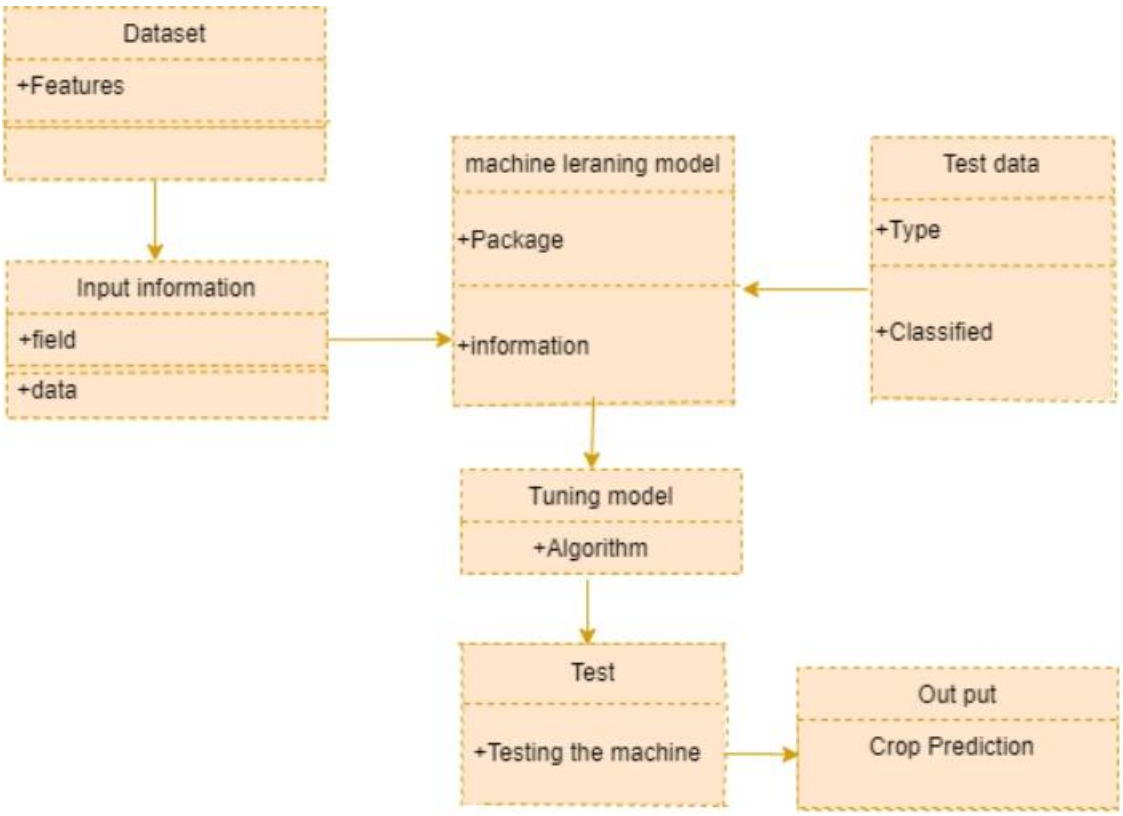


Fig 6.4.2: Class diagram of the model

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified and because, unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable to the developer/coder. Finally, before making the final version, the diagram should be drawn on plain paper and rework as many times as possible to make it correct.

C) **Activity Diagram**:

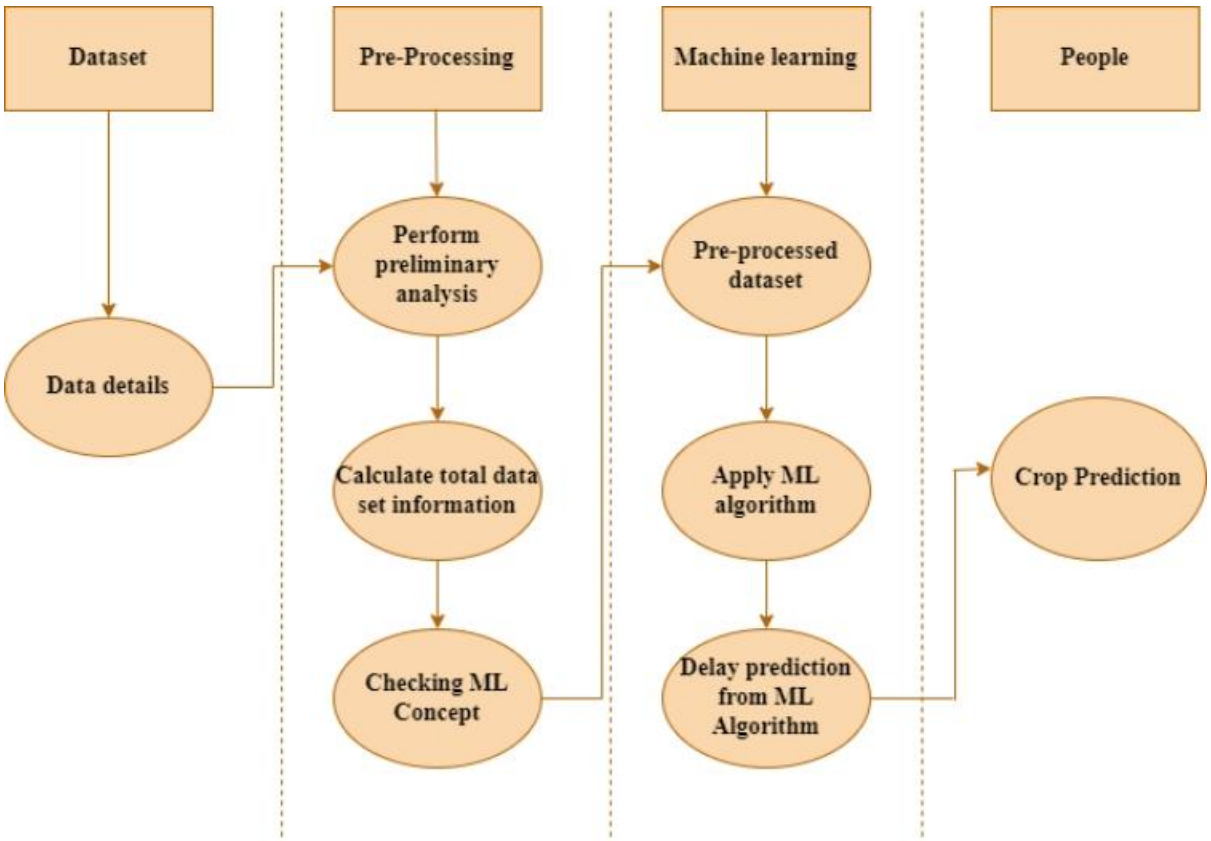


Fig 6.4.3: Activity diagram of the model

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

D) Sequence Diagram:

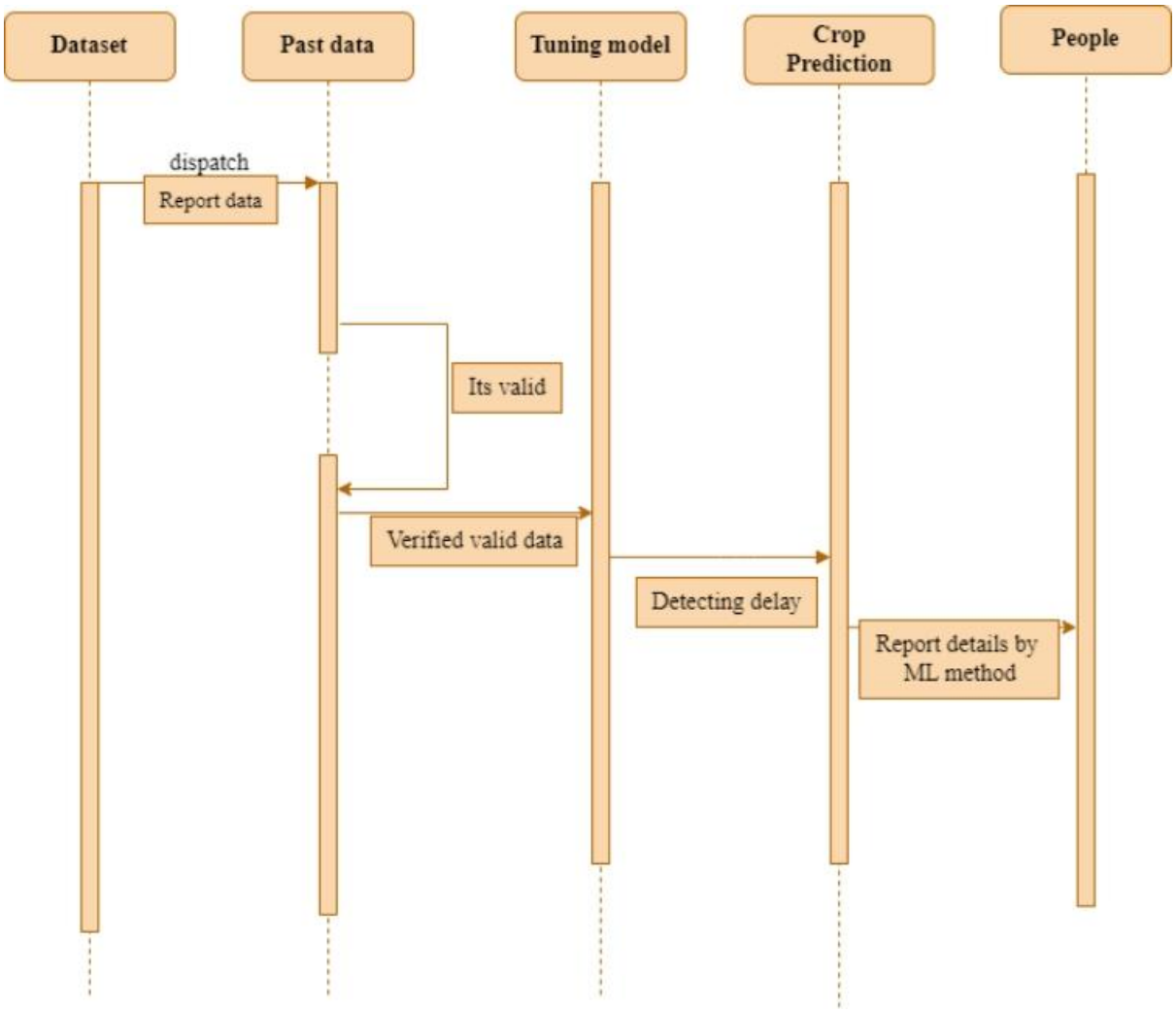


Fig 6.4.4: Sequence diagram of the model

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modelling, which focuses on identifying the behaviour within your system. Other dynamic modelling techniques include activity diagramming, communication diagramming, timing

diagramming, and interaction overview diagramming Sequence diagrams, along with class diagrams and physical data models are in my opinion the most important design-level models for modern business application development.

E) Entity Relationship Diagram (ERD):

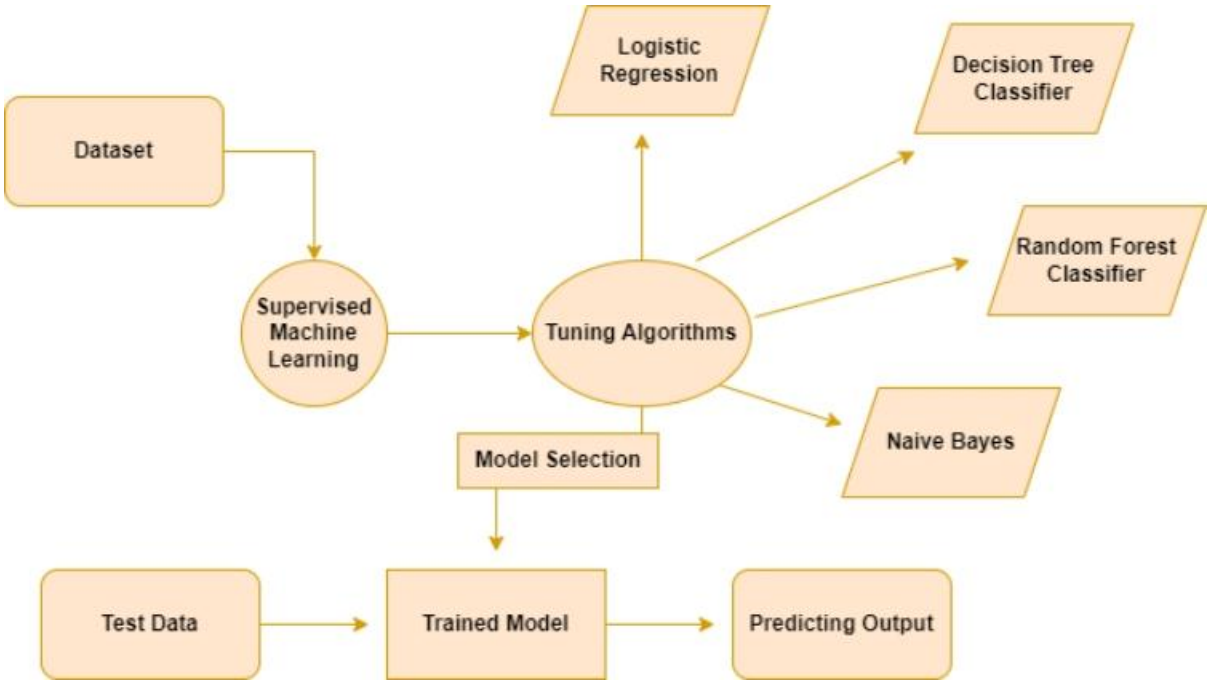


Fig 6.4.5: Entity relationship diagram of the model

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationship among people, objects, places, concepts or events within that system. An ERD is a data modelling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

6.5 LIST OF MODULES:

1 Data Pre-processing
2 Data Analysis of Visualization
3 Implementing Random Forest Algorithm
4 Implementing Naïve Bayes Algorithm
5 Implementing Decision Tree Algorithm
6 Implementing Logistic Regression Algorithm
7 Deployment Using Flask

6.6 MODULE DESCRIPTION:

1) DATA PRE-PROCESSING:

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters.

The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. It as machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model.

A number of different data cleaning tasks using Python's Pandas library and specifically, it focus on probably the biggest data cleaning task, missing values and it able to more quickly clean data. It wants to spend less time cleaning data, and more time exploring and modelling.

Some of these sources are just simple random mistakes. Other times, there can be a deeper reason why data is missing. It's important to understand these different types of missing data from a statistics point of view. The type of missing data will influence how to deal with filling in the missing values and to detect missing values, and do some basic imputation and detailed statistical approach for dealing with missing data. Before, joint into code, it's important to understand the sources of missing data. Here are some typical reasons why data is missing:

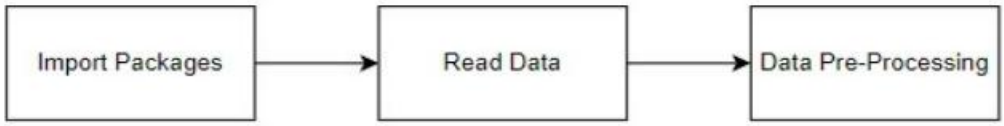
- User forgot to fill in a field.
- Data was lost while transferring manually from a legacy database.
- There was a programming error.
- Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.

Variable identification with Uni-variate, Bi-variate and Multi-variate analysis:

- import libraries for access and functional purpose and read the given dataset
- General Properties of Analyzing the given dataset
- Display the given dataset in the form of data frame
- show columns
- shape of the data frame
- To describe the data frame
- Checking data type and information about dataset
- Checking for duplicate data
- Checking Missing values of data frame
- Checking unique values of data frame
- Checking count values of data frame
- Rename and drop the given data frame
- To specify the type of values

- To create extra columns

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT
input: data
output: removing noisy data

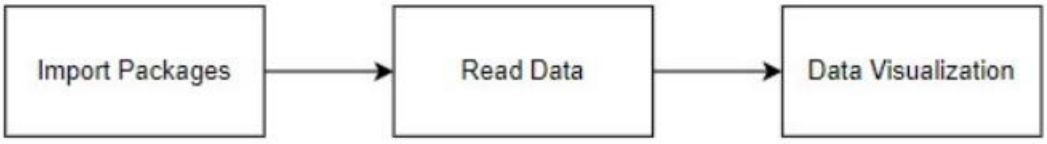
2) DATA VISUALIZATION:

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

- How to chart time series data with line plots and categorical quantities with bar charts.
- How to summarize data distributions with histograms and box plots.

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input: data

output: visualized data

ALGORITHM IMPLEMENTATION:

It is important to compare the performance of multiple different machine learning algorithms consistently and it will discover to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. When have a new dataset, it is a good idea to visualize the data using different techniques in order to look at the data from different perspectives. The same idea applies to model selection. You should use a number of different ways of looking at the estimated accuracy of your machine learning algorithms in order to choose the one or two to finalize. A way to do this is to use different visualization methods to show the average accuracy, variance and other properties of the distribution of model accuracies.

In the next section you will discover exactly how you can do that in Python with scikit-learn. The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data and it can achieve this by forcing each algorithm to be evaluated on a consistent test harness.

Performance Metrics to calculate:

False Positives (FP): A person who will pay predicted as defaulter. When actual class is no and predicted class is yes. E.g. if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN): A person who default predicted as payer. When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

True Positives (TP): A person who will not pay predicted as defaulter. These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN): A person who default predicted as payer. These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g., if actual class says this passenger did not survive and predicted class tells you the same thing.

True Positive Rate (TPR) = TP / (TP + FN)

False Positive Rate (FPR) = FP / (FP + TN)

Accuracy: The Proportion of the total number of predictions that is correct otherwise overall how often the model predicts correctly defaulters and non-defaulters.

Accuracy calculation:

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

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy

then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same.

Precision: The proportion of positive predictions that are actually correct.
Precision = TP / (TP + FP)
Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labelled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Recall: The proportion of positive observed values correctly predicted. (The proportion of actual defaulters that the model will correctly predict)
Recall = TP / (TP + FN)
Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

General Formula:
F- Measure = 2TP / (2TP + FP + FN)

F1-Score Formula:
F1 Score = 2*(Recall * Precision) / (Recall + Precision)

The below 4 different algorithms are compared:

- Naïve Bayes Classifier
- Random Forest
- Decision Tree Classifier

- Logistic Regression

3) NAÏVE BAYES CLASSIFIER:

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. The assumptions made by Naive Bayes are not generally correct in real-world situations. In-fact, the independence assumption is never correct but often works well in practice.

Real time Prediction: Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.

Multi class Prediction: This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.

Text classification/ Spam Filtering/ Sentiment Analysis: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)

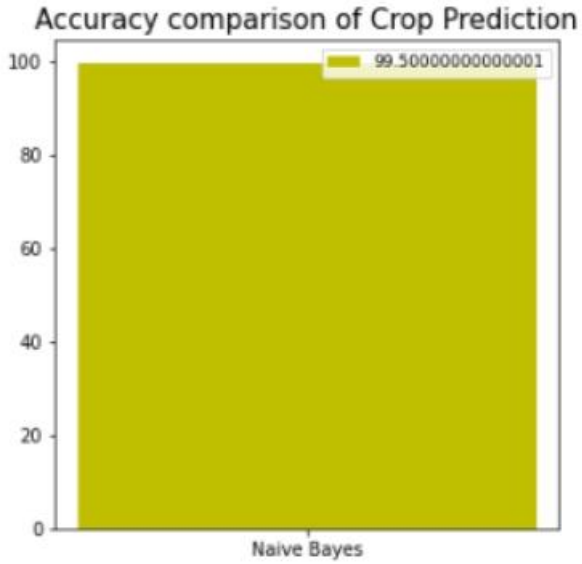
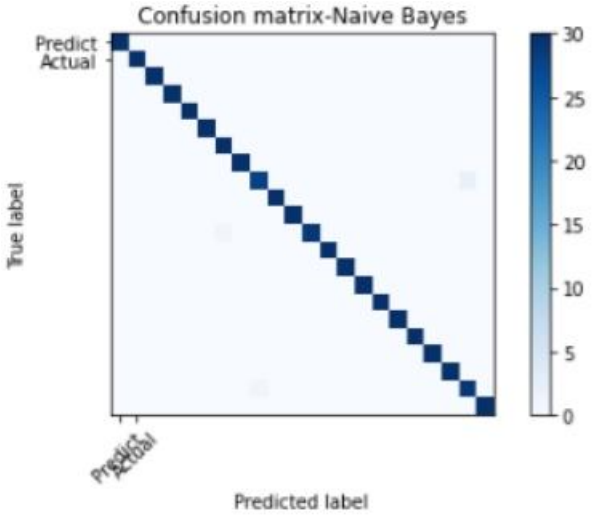
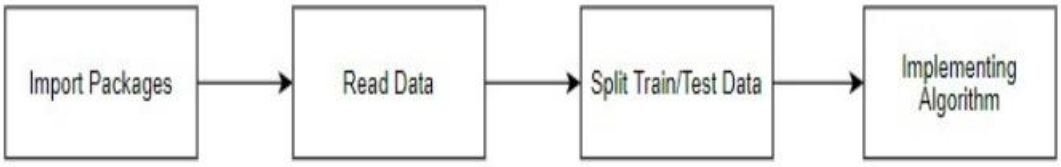


Fig 6.6.1: Accuracy graph of naïve bayes algorithm



6.6.2: Confusion matrix of naïve bayes algorithm

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input: data

output: getting accuracy

4) RANDOM FOREST CLASSIFIER:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

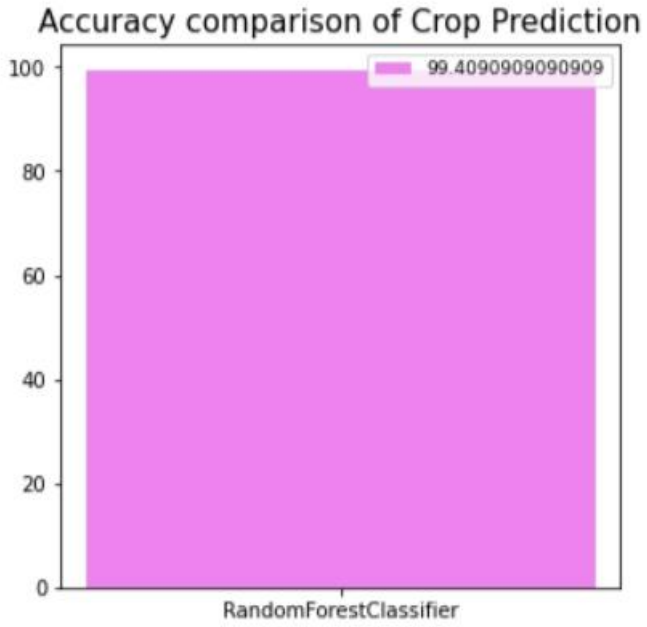


Fig 6.6.3: Accuracy graph of random forest classifier

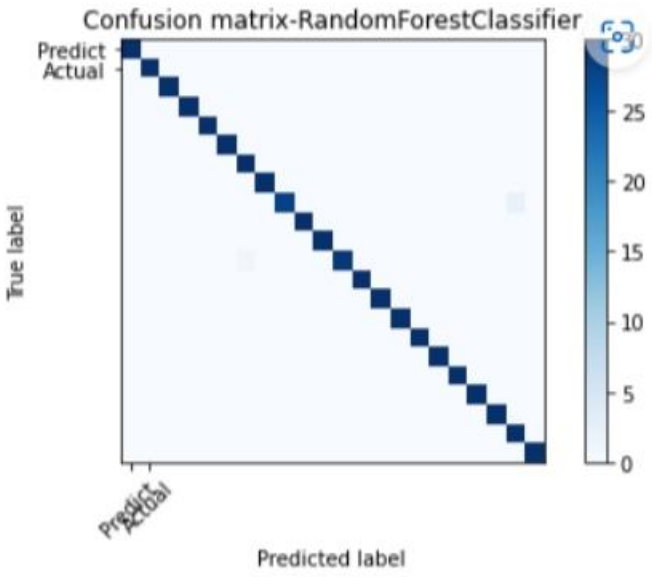
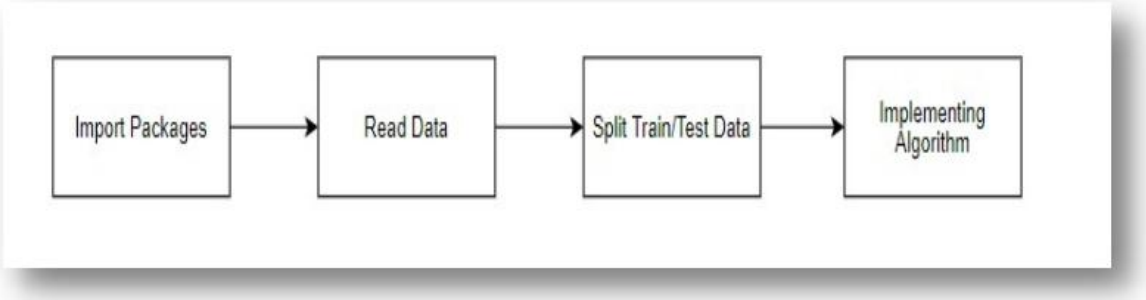


Fig 6.6.4: Confusion matrix of random forest classifier

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input: data

output: getting accuracy

5) DECISION TREE CLASSIFIER:

It is one of the most powerful and popular algorithm. Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.

Assumptions of Decision tree:

- At the beginning, we consider the whole training set as the root.
- Attributes are assumed to be categorical for information gain, attributes are assumed to be continuous.
- On the basis of attribute values records are distributed recursively.
- We use statistical methods for ordering attributes as root or internal node.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. Decision tree builds classification or regression models in the form of a tree structure. It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed.

This process is continued on the training set until meeting a termination condition. It is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers.

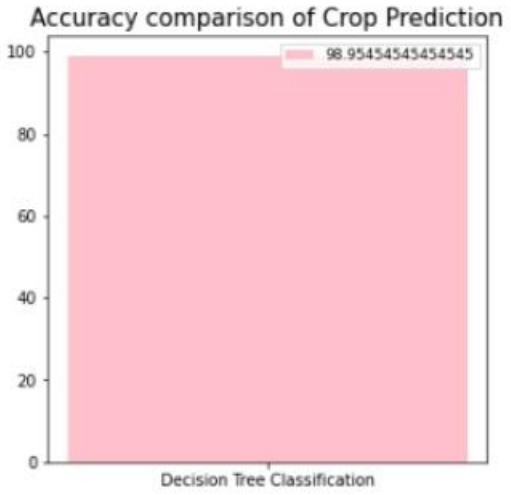


Fig 6.6.5: Accuracy graph of decision tree classifier

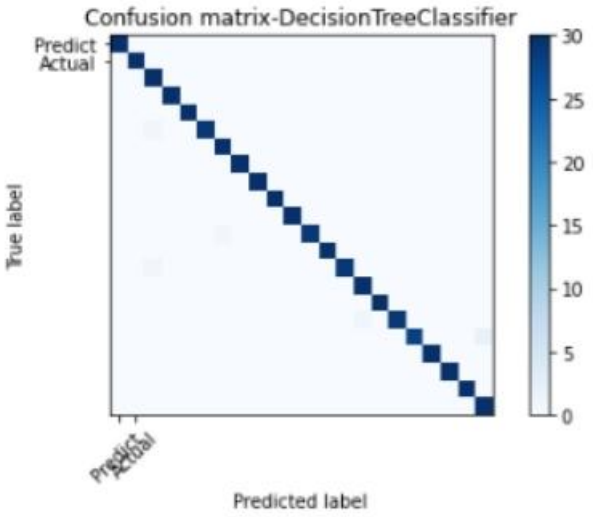
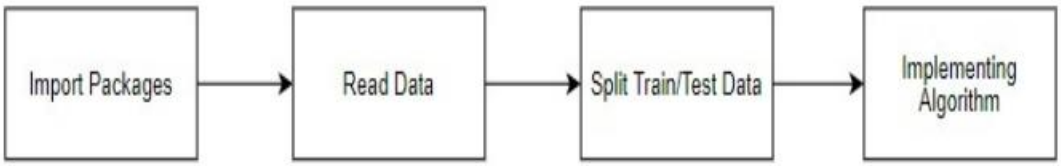


Fig 6.6.6: Confusion matrix of decision tree classifier

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input: data

output: getting accuracy

6) LOGISTIC REGRESSION:

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

In simple words, the dependent variable is binary in nature having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no).

Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.

Before diving into the implementation of logistic regression, we must be aware of the following assumptions about the same −

- In case of binary logistic regression, the target variables must be binary always and the desired outcome is represented by the factor level 1.
- There should not be any multi-collinearity in the model, which means the independent variables must be independent of each other.
- We must include meaningful variables in our model.
- We should choose a large sample size for logistic regression.

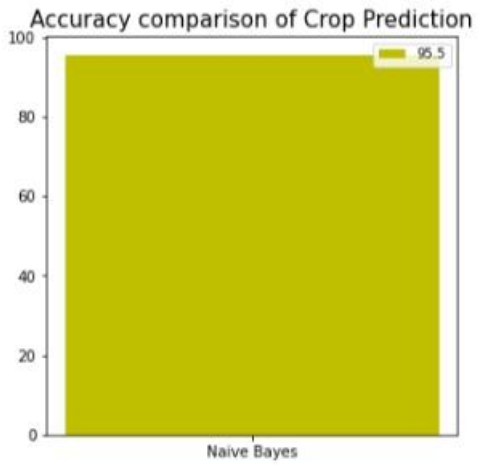
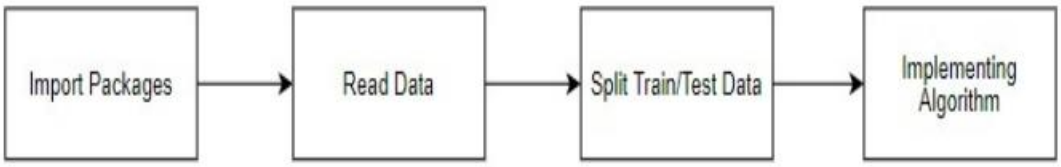


Fig 6.6.7: Accuracy graph of logistic regression

F

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

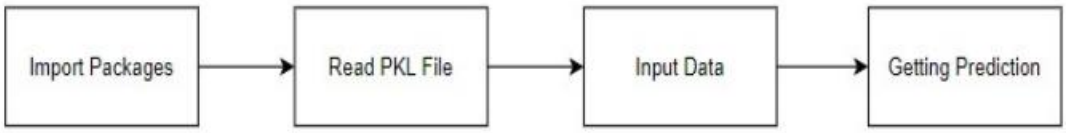
input: data

output: getting accuracy

7) Deploying the model predicting output:

In this module the trained machine learning model is converted into pickle data format file (.pkl file) which is then deployed for providing better user interface and predicting the output of Cirrhosis Severity.

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input: data values

output: predicting output

CHAPTER 7: CONCLUSION & FUTURE ENHANCEMENTS

7.1 Conclusion

We used supervised machine learning algorithms to predict crop yields based on various parameters such as nitrogen, phosphorus, temperature, humidity, pH, and rainfall content. Four different algorithms were tested: Naive Bayes, Random Forest Classifier, Logistic Regression, and Decision Tree Classifier.

The results of our experiments showed that Naive Bayes had the highest accuracy rate at 99.5%, while Random Forest Classifier had an accuracy rate of 99.4%. Logistic Regression and Decision Tree Classifier both had lower accuracy rates at 95.5% and 98.95% respectively.

These results indicate that machine learning algorithms can be effective in predicting crop yields based on various environmental parameters. Naive Bayes and Random Forest Classifier algorithms were found to be the most accurate in this study, which suggests that they could be useful for farmers in predicting future crop yields and planning accordingly.

The high accuracy rates achieved by these algorithms also suggest that they could be valuable tools for agricultural sectors in predicting crop yields and addressing the challenges posed by climate change and other environmental factors. By providing accurate predictions of crop yields, these algorithms could help farmers make more informed decisions about crop planning and management, which could ultimately improve their economic outcomes.

7.2 Future Enhancements

The utilization of machine learning algorithms can aid in crop prediction, which is crucial for the agricultural industry. Through the collection and analysis of data on various parameters such as nitrogen, phosphorus, temperature, humidity, pH, and rainfall content, machine learning can be used to predict the yield. The supervised machine learning algorithm

showed promising results in predicting the yield accurately. The application of this technology can greatly assist farmers in making informed decisions and achieving higher crop yields.

The research on crop yield prediction can be further improved by deploying the project on the cloud, which would increase accessibility and scalability.

Additionally, the implementation of this technology in the IoT system can help in the real-time monitoring of crop growth and assist farmers in taking timely actions to ensure successful crop growth.

Further exploration and implementation of different machine learning algorithms and techniques can also be conducted to improve the accuracy of crop yield prediction.

CHAPTER 8: APPENDIX

8.1 Source code

MODULE 1:Data validation and pre-processing technique

```
#import library packages
import numpy as n
import pandas as p

#Load given dataset
data = p.read_csv('crop.csv')

import warnings
warnings.filterwarnings('ignore')
Before drop the given dataset

data.head()

#shape
data.shape

data.columns
After drop the given dataset

data.head()

df=data.dropna()

df.shape

df.columns

df.describe()

df.info()
Checking duplicate value from dataframe

#Checking for duplicate data
df.duplicated()

#find sum of duplicate data
sum(df.duplicated())

#Checking sum of missing values
df.isnull().sum()
```

```
df.columns

df.nitrogen.unique()

df.phosphorus.unique()

df.potasium.unique()

df.potasium.sort_values().unique()

p.Categorical(df['label']).describe()

df['label'].value_counts()

df['potasium'].value_counts()

df.corr()

df.head()

df.columns

df['label'].unique()

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

df['label'].unique()

df.head(10)

df.isnull().sum()

df.tail(10)
```

Module 2: Exploration data analysis of visualization and training a model by given attributes

```
#import library packages
import pandas as p
import matplotlib.pyplot as plt
import seaborn as s
import numpy as n
```

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

data = p.read_csv("crop.csv")

df = data.dropna()

df.shape

df.columns

p.crosstab(df.rainfall,df.humidity)

df.isnull().sum()

df.columns

df['humidity'].hist(figsize=(7,6), color='violet', alpha=0.7)
plt.xlabel('humidity')
plt.ylabel('rainfall')
plt.title('humidity & rainfall')

df['nitrogen'].hist(figsize=(7,6), color='b', alpha=0.7)
plt.xlabel('nitrogen')
plt.ylabel('phosphorus')
plt.title('nitrogen & phosphorus')

#Propagation by variable
def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize = 10)
    ax.set_title(variable + ' \n', fontsize = 15)
    return n.round(dataframe_pie/df.shape[0]*100,2)
PropByVar(df, 'label')

#Propagation by variable
def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable + ' \n', fontsize = 15)
    return n.round(dataframe_pie/df.shape[0]*100,2)

PropByVar(df, 'nitrogen')

# Heatmap plot diagram
fig, ax = plt.subplots(figsize=(15,7))
s.heatmap(df.corr(), ax=ax, annot=True)

plt.boxplot(df['phosphorus'])
plt.show()
```

```
import seaborn as s
s.boxplot(df['nitrogen'], color='m')

from sklearn.preprocessing import LabelEncoder
var_mod =['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

df.columns

fig, ax = plt.subplots(figsize=(16,8))
ax.scatter(df['humidity'],df['rainfall'])
ax.set_xlabel('humidity')
ax.set_ylabel('rainfall')
ax.set_title('rainfall & humidity')
plt.show()

df.columns

plt.plot(df["label"], df["ph"], color='g')
plt.xlabel('label')
plt.ylabel('ph')
plt.title('ph value of crops')
plt.show()

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)
Splitting Train / Test

#preprocessing, split test and dataset, split response variable
X = df.drop(labels='label', axis=1)
#Response variable
y = df.loc[:,'label']

#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do
because there are so few fraudulent transactions.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1,
stratify=y)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

Module 3: Performance measurements of Gaussian Naive Bayes

```
#import library packages
import pandas as p
import matplotlib.pyplot as plt
import seaborn as s
import numpy as n

import warnings
warnings.filterwarnings('ignore')

#Load given dataset
data = p.read_csv('crop.csv')

df=data.dropna()

df.columns

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

#preprocessing, split test and dataset, split response variable
X = df.drop(labels='label', axis=1)
#Response variable
y = df.loc[:,'label']

'''We'll use a test size of 30%. We also stratify the split on the response variable,
which is very important to do because there are so few fraudulent transactions'''

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1,
stratify=y)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))

#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's
evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import cross_val_score

GNB= GaussianNB()
```

```
GNB.fit(X_train,y_train)

predictLR = GNB.predict(X_test)

print("")
print('Classification report of Logistic Regression Results:')
print("")
print(classification_report(y_test,predictLR))

print("")
cm1=confusion_matrix(y_test,predictLR)
print('Confusion Matrix result of Logistic Regression is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(GNB, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Logistic Regression is:",accuracy.mean() * 100)
LR=accuracy.mean() * 100

def graph():
    import matplotlib.pyplot as plt
    data=[LR]
    alg="Logistic Regression"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("c"))
    plt.title("Accuracy comparison of Crop Prediction",fontsize=15)
    plt.legend(b,data,fontsize=9)

graph()

TP = cm1[0][0]
FP = cm1[1][0]
FN = cm1[1][1]
TN = cm1[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
```

```
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm1, title='Confusion matrix-Logistic_Regression',
cmap=plt.cm.Blues):
    target_names=['Predict','Actual']
    plt.imshow(cm1, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = n.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm1=confusion_matrix(y_test, predictLR)
print('Confusion matrix-Logistic_Regression:')
print(cm1)
plot_confusion_matrix(cm1)
```

Exporting the module as a pkl file

```
# import joblib
# joblib.dump(logR,'lr.pkl')
```

Module 4 : Random Forest Classifier

```
#import library packages
import pandas as p
import matplotlib.pyplot as plt
import seaborn as s
```

```
import numpy as n

import warnings
warnings.filterwarnings('ignore')

data=p.read_csv('crop.csv')

df=data.dropna()

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

#preprocessing, split test and dataset, split response variable
X = df.drop(labels='label', axis=1)
#Response variable
y = df.loc[:,'label']

'''We'll use a test size of 30%. We also stratify the split on the response variable,
which is very important to do because there are so few fraudulent transactions'''

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1,
stratify=y)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))

#According to the cross-validated MCC scores, the random forest is the best-performing
model, so now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

RFC= RandomForestClassifier()

RFC.fit(X_train,y_train)

predictRF = RFC.predict(X_test)

print("")
print('Classification report of Random Forest Results:')
print("")
print(classification_report(y_test,predictRF))
```

```
print("")
cm1=confusion_matrix(y_test,predictRF)
print('Confusion Matrix result of Random Forest Classifier is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(RFC, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of RandomForestClassifier is:",accuracy.mean() * 100)
RF=accuracy.mean() * 100

def graph():
    import matplotlib.pyplot as plt
    data=[RF]
    alg="RandomForestClassifier"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("violet"))
    plt.title("Accuracy comparison of Crop Prediction",fontsize=15)
    plt.legend(b,data,fontsize=9)

graph()

TP = cm1[0][0]
FP = cm1[1][0]
FN = cm1[1][1]
TN = cm1[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
```

```
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm1, title='Confusion matrix-RandomForestClassifier',
cmap=plt.cm.Blues):
    target_names=['Predict','Actual']
    plt.imshow(cm1, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = n.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm1=confusion_matrix(y_test, predictRF)
print('Confusion matrix-RandomForestClassifier:')
print(cm1)
plot_confusion_matrix(cm1)
```

Exporting modules as pkl file

```
# import joblib
# joblib.dump(rfc,'rf.pkl')
```

Module 5 : Performance measurements of Logistic Regression

```
#import library packages
import pandas as p
import matplotlib.pyplot as plt
import seaborn as s
import numpy as n

import warnings
warnings.filterwarnings('ignore')

#Load given dataset
data = p.read_csv('crop.csv')

df = data.dropna()

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)
```

```
#preprocessing, split test and dataset, split response variable
X = df.drop(labels='label', axis=1)
#Response variable
y = df.loc[:,'label']

'''We'll use a test size of 30%. We also stratify the split on the response variable,
which is very important to do because there are so few fraudulent transactions'''

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1,
stratify=y)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
#According to the cross-validated MCC scores, the random forest is the best-performing
model, so now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score

LogR = LogisticRegression(random_state=42)
LogR.fit(X_train, y_train)


predictNB = LogR.predict(X_test)

print("")
print('Classification report of Naive Bayes Results:')
print("")
print(classification_report(y_test,predictNB))

print("")
cm1=confusion_matrix(y_test,predictNB)
print('Confusion Matrix result of Naive Bayes is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(LogR, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
```

```
print("Accuracy result of LogisticRegression is:",accuracy.mean() * 100)
VC=accuracy.mean() * 100

def graph():
    import matplotlib.pyplot as plt
    data=[VC]
    alg="Naive Bayes"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color=("y"))
    plt.title("Accuracy comparison of Crop Prediction",fontsize=15)
    plt.legend(b,data,fontsize=9)

graph()

TP = cm1[0][0]
FP = cm1[1][0]
FN = cm1[1][1]
TN = cm1[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm1, title='Confusion matrix-Naive Bayes', cmap=plt.cm.Blues):
    target_names=['Predict','Actual']
    plt.imshow(cm1, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = n.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm1=confusion_matrix(y_test, predictNB)
```

```
print('Confusion matrix-Naive Bayes:')
print(cm1)
plot_confusion_matrix(cm1)
```

Importing module as a pkl file

```
# import joblib
# joblib.dump(gnb,'nb.pkl')
```

Module 6 : Decision Tree Classifier

```
import pandas as p
import numpy as n
import matplotlib.pyplot as plt
import seaborn as s

import warnings
warnings.filterwarnings('ignore')

data=p.read_csv('crop.csv')

df=data.dropna()

df.columns

from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i]).astype(int)

#preprocessing, split test and dataset, split response variable
X = df.drop(labels='label', axis=1)
#Response variable
y = df.loc[:,'label']

'''We'll use a test size of 30%. We also stratify the split on the response variable,
which is very important to do because there are so few fraudulent transactions'''

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1,
stratify=y)
print("Number of training dataset: ", len(X_train))
print("Number of test dataset: ", len(X_test))
print("Total number of dataset: ", len(X_train)+len(X_test))
```

```
#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef, cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

dtree= DecisionTreeClassifier()

dtree.fit(X_train,y_train)

predictDT = dtree.predict(X_test)

print("")
print('Classification report of Decision Tree Results:')
print("")
print(classification_report(y_test,predictDT))

print("")
cm1=confusion_matrix(y_test,predictDT)
print('Confusion Matrix result of Decision Tree Classifier is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(dtree, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Decision Tree Classifier is:",accuracy.mean() * 100)
DT=accuracy.mean() * 100

def graph():
   import matplotlib.pyplot as plt
   data=[DT]
   alg="Decision Tree Classification"
   plt.figure(figsize=(5,5))
   b=plt.bar(alg,data,color=("pink"))
   plt.title("Accuracy comparison of Crop Prediction",fontsize=15)
      plt.legend(b,data,fontsize=9)

graph()
```

```
TP = cm1[0][0]
FP = cm1[1][0]
FN = cm1[1][1]
TN = cm1[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm1, title='Confusion matrix-DecisionTreeClassifier',
cmap=plt.cm.Blues):
    target_names=['Predict','Actual']
    plt.imshow(cm1, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = n.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm1=confusion_matrix(y_test, predictDT)
print('Confusion matrix-DecisionTreeClassifier:')
print(cm1)
plot_confusion_matrix(cm1)
```

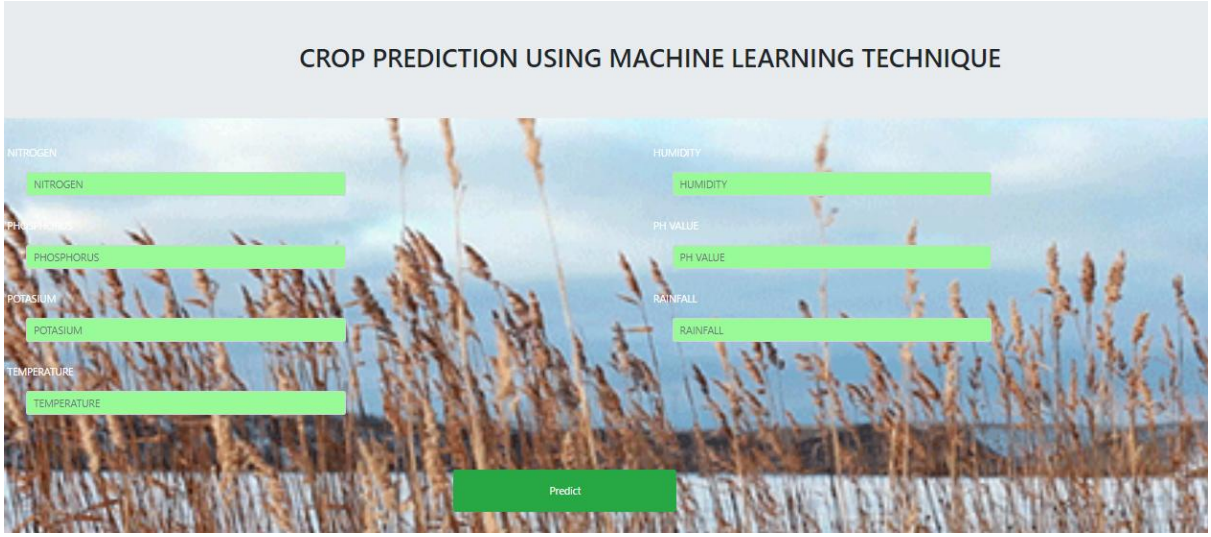
Flask Deploy

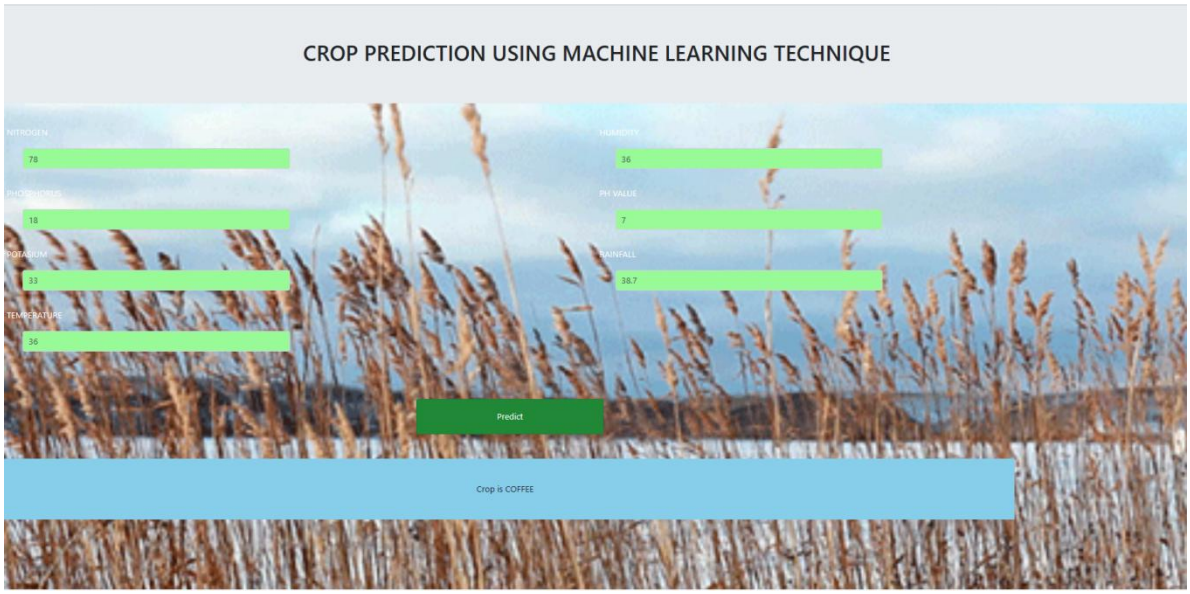
```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
import joblib
```

```
app = Flask(__name__)
model = joblib.load('rf.pkl')

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model.predict(final_features)

    output = prediction[0]
    if output==0:
        output='APPLE'
    elif output==1:
        output='BANANA'
    elif output==2:
        output='BLACK_GRAM'
    elif output==3:
        output='CHICKPEA'
    elif output==4:
        output='COCONUT'
    elif output==5:
        output='COFFEE'
    elif output==6:
        output='COTTON'
    elif output==7:
        output='GRAPES'
    elif output==8:
        output='JUTE'
    elif output==9:
        output='KIDNEY_BEANS'
    elif output==10:
        output='LENTIL'
    elif output==11:
        output='MAIZE'
    elif output==12:
        output='MANGO'
    elif output==13:
        output='MOTH_BEANS'
    elif output==14:
        output='MUNG_BEAN'
    elif output==15:
```

```
        output='MUSK_MELON'
    elif output==16:
        output='ORANGE'
    elif output==17:
        output='PAPAYA'
    elif output==18:
        output='PIGEON_PEAS'
    elif output==19:
        output='POMEGRANATE'
    elif output==20:
        output="RICE"
    else:
        output="WATER_MELON"


    return render_template('index.html', prediction_text='Crop is {}'.format(output))

if __name__ == "__main__":
    app.run(host="localhost", port=7000)
```

8.2 Screen shots:




CROP PREDICTION USING MACHINE LEARNING TECHNIQUE
NITROGEN
78
HUMIDITY
36
PHOSPHORUS
18
PH VALUE
7
POTASSIUM
33
RAINFALL
38.7
TEMPERATURE
36
Predict
Crop is COFFEE

CHAPTER 9: REFERENCES

Anwar et al. "Crop Yield Prediction Using Machine Learning Algorithms: A Review", Journal of King Saud University - Computer and Information Sciences. (2018).

Huang et al. "Supervised machine learning algorithms for predicting crop yield" (2019), Journal of Agricultural Engineering Research. (2019).

Adewole et al. "Crop Yield Prediction Using Machine Learning Techniques: A Comparative Study", Journal of Agricultural Informatics. (2017).

Kaur et al. "Agricultural Crop Yield Prediction using Machine Learning Algorithms: A Comparative Study", The International Journal of Advanced Trends in Computer Science and Engineering. (2018).

Park et al. "Crop yield prediction using machine learning techniques: An empirical study", Journal of Ambient Intelligence and Humanized Computing. (2020).

Al-Mansour et al. "Crop Yield Prediction using Machine Learning Algorithms for Better Agricultural Planning", International Journal of Applied Engineering Research. (2020).

Nadeem et al. "Crop Yield Prediction using Machine Learning Approaches: A Review", Journal of Advanced Research in Dynamical and Control Systems.(2021).

Ali et al. "Predicting Crop Yield Using Machine Learning Algorithms: A Review of Literature", Journal of Agricultural Engineering and Technology. (2022).

Zhang et al. "An Analysis of Supervised Machine Learning Algorithms for Crop Yield Prediction", International Journal of Agriculture and Biology. (2019).

Park et al. "Crop Yield Prediction using Machine Learning Techniques: A Case Study", Journal of Environmental Science and Technology.(2019).