

Project Report On
CROP YIELD PREDICTION SYSTEM
B.E. (INFORMATION TECHNOLOGY)

SUBMITTED BY
SIDDHANT BHOI (D15B – 07)
RUCHITA DALVI (D15B – 12)
KIRTI EPPANAPELLI (D15B – 16)
SHUBHAM JHA (D15B – 24)

UNDER THE GUIDANCE OF
Mrs. BINCY IVIN.



Department Of Information Technology
Vivekanand Education Society's Institute Of Technology
2022 – 2023

INDEX

Sr. No.	Table Of Content	Page No
1.	Introduction	1
1.1	Description/Background	2
1.2	Purpose	3
1.3	Scope of the project	5
1.4	Problem Statement & Solution	6
1.5	Details of the Project	7
2.	Project Methodology	9
2.1	Data PreProcessing	10
2.2	Yield Production Using Deep Neutral Network (SGD)	10
2.3	Results	11
2.4	Analysis	11
2.5	Feature Selection	12
3.	System Architecture Design	13
3.1	System Architecture Design	14
3.2	Use Case	14
4.	UML Diagrams	17
4.1	Data Flow Diagram Representation	18
4.2	Activity Diagram Representation	21
5.	Implementation	24
5.1	Implementation	26
6.	Conclusion &References	28
6.1	Conclusion	2
6.2	References	29

FIGURES

Figure. No.	Title	Page No
3.2	Use Case	16
4.1	Data Flow Diagram	19
4.2	Activity Diagram	23

ROSPL LAB PROJECT

The Recent Open Source Project Lab (ROSPL) is a hands-on course designed to provide students with practical experience in open-source software development. Open-source software (OSS) allows anyone to view, modify, and enhance source code. ROSPL aims to bridge the gap between academic learning and real-world practices by engaging students with widely used code in a global context.

- **Goals of ROSPL**

Key objectives include:

1. **Understanding OSS Contributions:** Highlighting the importance of OSS in modern software development.
2. **Skill Enhancement:**
 - **Version Control Systems (VCS):** Mastering Git and platforms like GitHub for code management.
 - **Collaborative Workflows:** Learning how developers collaborate on large projects through pull requests and issue tracking.
 - **Project Management:** Navigating open-source conventions and understanding contribution reviews.
 - **Community Engagement:** Fostering connections with a broader developer community for feedback and learning.

Students explore open-source repositories, identify issues or enhancements, and submit contributions through pull requests (PRs). They also improve documentation to ensure accessibility for new contributors and users. ROSPL serves as a practical platform for applying knowledge, learning industry practices, and building a portfolio of contributions.

- **Activities in ROSPL**

In ROSPL, I participated in two main activities:

1. **Contributing to Open Source Repositories:**

- I discovered and contributed to various OSS projects on GitHub, ranging from simple utilities to complex systems. My goal was to resolve at least 20 issues and achieve acceptance of 10 pull requests.
- I identified relevant issues, cloned repositories, developed solutions, and submitted PRs. Engaging in the review process and addressing feedback helped me gain experience in a real-world software environment, where I:
 - Collaborated with other developers.
 - Followed coding best practices.
 - Managed version control and resolved conflicts.
 - Communicated effectively through commit messages and PR descriptions.

2. Enhancing Documentation:

I improved my own project repository by creating a comprehensive README file. This document serves as an introduction for users and contributors, outlining the project's purpose and installation steps.

My README included:

Project Overview: A brief description and goals.

Installation Guide: Step-by-step setup instructions.

By adding this README, I made my repository more accessible, helping new users understand how to get involved. Good documentation is essential in OSS, facilitating collaboration and reducing barriers for newcomers.

Through these activities, ROSPL provided me with valuable hands-on experience in coding and documentation within a professional context, laying a strong foundation for my development skills and community-driven approach.

• Why We Study ROSPL

The Recent Open Source Project Lab (ROSPL) provides students with essential hands-on experience in open-source software development. Through this lab, we learn to engage with real-world projects and understand the collaborative nature of open-source contributions, which are increasingly vital in the tech industry.

Activities in ROSPL

As part of ROSPL, I participated in two main activities:

1. Contributing to Open Source Repositories:

- The first phase involved identifying and contributing to various open-source projects on platforms like GitHub. These projects span a range of domains, including web development, machine learning, and automation. My goal was to resolve at least 20 issues and submit 10 accepted pull requests.
- I began by exploring repositories to find tasks that aligned with my skills, such as bug fixes and feature requests. Using GitHub's issue tracker, I filtered through active projects and cloned the relevant repositories. After developing solutions locally, I submitted pull requests, which were then reviewed by project maintainers. After addressing their feedback, I successfully merged my contributions.

2. This experience allowed me to:

- Collaborate with other developers on meaningful projects.
- Follow best practices for writing clean and efficient code.
- Manage version control effectively and resolve merging conflicts.
- Communicate clearly through commit messages and pull request descriptions.

3. By the end of this phase, I had engaged deeply with the open-source ecosystem, enhancing my coding skills and understanding of collaboration.

4. **Improving Documentation:**

- The second component focused on enhancing the documentation of my own project by creating a comprehensive README file. This file serves as an introductory guide for potential users and contributors, explaining the project's purpose, installation process, and contribution guidelines.
- My README included:
 - **Project Overview:** A brief description of the project and its objectives.
 - **Installation Guide:** Step-by-step instructions for setup, including prerequisites and commands.

5. By adding this README, I improved the accessibility of my repository, helping new users and contributors understand its structure and how to participate. Effective documentation is crucial in open-source projects, as it facilitates collaboration and reduces the learning curve for newcomers.

Through these activities, ROSPL provided valuable practical experience in both coding and documentation within a professional open-source context. This exposure to real-world coding practices and collaboration on platforms like GitHub laid a strong foundation for my software development skills and community-oriented approach.

Chapter 1

INTRODUCTION

INTRODUCTION

1.1 Description/Background:

Crop yield estimation is a very important problem faced by our farmers. Various techniques have been used traditionally to determine the yield. Modern day algorithms, techniques like machine learning and neural networks have also been used previously to solve this problem. Historic data based on different parameters has been used to train the models and predict yield. Previous papers have considered the data as a whole to estimate results. In this paper we try to sample the data over different time intervals and granularise our parameters so that accurate results can be obtained. We use artificial neural networks over our sampled data to estimate the yield. Our results show a great improvement in the accuracy of prediction.

Accurate yield prediction requires fundamental understanding of the functional relationship between yield and these interactive factors, and to reveal such relationship requires both comprehensive datasets and powerful algorithms. In the 2018 Syngenta Crop Challenge, Syngenta released several large datasets that recorded the genotype and yield performances of 2,267 maize hybrids planted in 2,247 locations between 2008 and 2016 and asked participants to predict the yield performance in 2017. With perfect weather data, the RMSE would be reduced to 11% of the average yield and 46% of the standard deviation. We also performed feature selection based on the trained DNN model, which successfully decreased the dimension of the input space without significant drop in the prediction accuracy. Our computational results suggested that this model significantly outperformed other popular methods such as Lasso, shallow neural networks (SNN), and regression tree (RT). The results also revealed that environmental factors had a greater effect on the crop yield than genotype.

1.2 Purpose

A Crop Yield Prediction System using the Stochastic Gradient Descent (SGD) algorithm serves as a sophisticated tool for agricultural management, leveraging machine learning techniques to forecast crop yields accurately. Here are some key purposes and functionalities of such a system:

- **Accurate Yield Prediction** : The primary purpose of the Crop Yield Prediction System is to forecast crop yields with a high degree of accuracy. By analyzing historical data, weather patterns, soil characteristics, and other relevant factors, the system employs the Stochastic Gradient Descent algorithm to generate predictive models that estimate future crop yields for different regions and crop types.
- **Data Integration and Management** : The Crop Yield Prediction System centralizes data related to agriculture, including historical crop yields, weather data, soil properties, crop types, and farming practices. This centralized repository facilitates efficient data management, ensuring that relevant information is readily accessible for analysis and modeling.
- **Algorithm Optimization** : The system utilizes the Stochastic Gradient Descent algorithm, which is well-suited for large-scale optimization problems commonly encountered in machine learning applications. Through iterative optimization techniques, SGD refines model parameters to minimize prediction errors and enhance the accuracy of yield forecasts.
- **Seasonal and Regional Variability** : Crop yield prediction models must account for seasonal variations in weather patterns, soil conditions, and other environmental factors. The Crop Yield Prediction System incorporates these variables into its predictive models, allowing farmers and agricultural stakeholders to anticipate fluctuations in crop yields across different regions and growing seasons.

→ **Decision Support for Farmers** : By providing accurate yield forecasts, the system serves as a valuable decision support tool for farmers and agricultural practitioners. Armed with reliable predictions, farmers can make informed decisions regarding crop selection, planting schedules, irrigation management, fertilizer application, and harvesting strategies, ultimately optimizing their agricultural operations for improved productivity and profitability.

→ **Research and Development:** The Crop Yield Prediction System also facilitates research and development efforts in agriculture by providing valuable insights into the factors influencing crop yields. Researchers can leverage the system's predictive models to study the effects of climate change, agronomic practices, crop genetics, and other variables on agricultural productivity, leading to innovations in crop breeding, cultivation techniques, and sustainable farming practices.

In essence, a Crop Yield Prediction System powered by the Stochastic Gradient Descent algorithm serves as a pivotal tool for optimizing agricultural management, enabling stakeholders to make data-driven decisions, mitigate risks, and enhance productivity in the dynamic field of crop production.

1.3 Scope Of Our Project

- i. **Agricultural Technology Sector** : The agricultural technology sector is a promising market for crop yield prediction systems. These systems can be marketed to agricultural technology companies, startups, and research institutions that are focused on developing innovative solutions to enhance agricultural productivity and sustainability. By providing accurate crop yield predictions, these systems can empower farmers and agricultural stakeholders to make data-driven decisions, optimize resource allocation, and maximize crop yields.
- ii. **Agribusiness and Farm Management Companies** : Agribusiness and farm management companies represent another lucrative market for crop yield prediction systems. These companies provide a range of services to farmers, including crop monitoring, precision agriculture, and farm management solutions. By integrating crop yield prediction systems into their offerings, these companies can enhance their value proposition and provide farmers with actionable insights to improve their farming practices, increase profitability, and mitigate risks associated with uncertain crop yields.
- iii. **Government Agencies and Agricultural Policy Makers:** Government agencies and agricultural policy makers can benefit from the insights provided by crop yield prediction systems to inform agricultural policies, programs, and initiatives. By accurately predicting crop yields, these systems can help governments optimize agricultural production, manage food security, and support rural development efforts. Additionally, agricultural policy makers can use the data generated by these systems to identify areas of intervention, allocate resources effectively, and address challenges such as climate change, water scarcity, and food insecurity.

1.4 Problem Statement & Solution

→ Problem Statement –

Farmers face uncertainties in crop yield due to various factors such as weather conditions, soil quality, pest infestations, and management practices. Predicting crop yields accurately is crucial for farmers to make informed decisions regarding crop selection, resource allocation, and marketing strategies. Traditional methods of crop yield prediction often lack accuracy and scalability, hindering farmers' ability to optimize agricultural productivity and profitability. Therefore, there is a need for an efficient and reliable crop yield prediction system that leverages advanced machine learning algorithms to provide accurate forecasts and support sustainable farming practices.

→ Solution –

We propose the development of a Crop Yield Prediction System using the Stochastic Gradient Descent (SGD) algorithm, a powerful optimization technique commonly used in machine learning for regression tasks. The system will utilize historical crop yield data, weather data, soil information, crop management practices, and other relevant factors as input features to train predictive models.

1) Data Collection & Preprocessing:

- ✓ Gather historical crop yield, weather, soil, and management data.
- ✓ Clean, normalize, and transform data for training.

2) Model Development:

- ✓ Utilize SGD algorithm for predictive modeling.
- ✓ Optimize model parameters for accuracy.

3) Evaluation & Validation:

- ✓ Assess model performance using metrics like MAE and RMSE.

- ✓ Validate across diverse agricultural conditions.

❖ Objectives:

The objective of the Crop Yield Prediction System using the Stochastic Gradient Descent (SGD) algorithm is to develop a robust and accurate predictive model that empowers farmers with actionable insights to optimize agricultural productivity and profitability.

So Crop Yield Prediction System objectives are as following:-

- Enhance Agricultural Decision Making.
- Improve Sustainability and Resource Efficiency.
- Mitigate Risks and Uncertainties.
- Facilitate Precision Agriculture.
- Empower Stakeholders with Data-Driven Insights.
- Promote Adoption of Advanced Technologies.

1.5 Details Of The Project

❑ **Modules of Crop Yield Prediction System:**

- ✓ **Data Collection Module:** Used for collecting data related to crops, including environmental factors, soil conditions, historical yield data, etc.
- ✓ **Data Preprocessing Module:** Used for cleaning, transforming, and preparing the collected data for analysis.
- ✓ **Feature Selection Module:** Used for selecting relevant features that impact crop yield prediction.
- ✓ **Stochastic Gradient Descent (SGD) Algorithm Module:** Implements the SGD algorithm for training the predictive model.
- ✓ **Model Evaluation Module:** Evaluates the performance of the predictive model using various metrics.
- ✓ **Prediction Module:** Utilizes the trained model to predict crop yields based on input data.

- ✓ **Visualization Module:** Provides visual representations of the data, model performance, and predicted yields.

❑ **Input Data and Validation:**

- ✓ Ensure all input data related to environmental factors, soil conditions, etc., are validated and do not contain invalid values.
- ✓ Implement checks to handle missing or incomplete data during preprocessing.
- ✓ Validate input fields to prevent blank or erroneous entries.
- ✓ Integrate data validation checks throughout the system to maintain data integrity.

❑ **Algorithm Implementation and Testing:**

- ✓ Implement the stochastic gradient descent algorithm for training the predictive model.
- ✓ Test the algorithm with various datasets to ensure robustness and accuracy.
- ✓ Record any errors encountered during testing and make necessary modifications to improve performance.

❑ **Validation for User Input:**

- ✓ Implement input validation mechanisms to ensure that users provide valid inputs for prediction.
- ✓ Provide feedback to users in case of invalid inputs and guide them to correct their inputs.

❑ **Coding Standards and Documentation:**

- ✓ Adhere to coding standards to maintain code readability, consistency, and scalability.
- ✓ Document the codebase thoroughly to facilitate understanding and future maintenance.

❑ **Functionality Testing:**

- ✓ Test the functionality of each module independently and in conjunction with others to ensure seamless integration.
- ✓ Verify that the system handles various types of calculations and predictions accurately.

❑ **Visualization and Reporting:**

- ✓ Provide visualizations of data trends, model training progress, and predicted crop yields for easy interpretation.

- ✓ Generate reports summarizing the performance of the crop yield prediction system and any recommendations for improvement.

Chapter 2

PROJECT METHODOLOGY

PROJECT METHODOLOGY

2.1 Data Preprocessing:

Approximately 37% of the genotype data had missing values. To address this issue, we used a two-step approach to preprocess the genotype data before they can be used by the neural network model. First, we used a 97% call rate to discard genetic markers whose non-missing values were below this call rate. Then we also discarded genetic markers whose lowest frequent allele's frequency were below 1%, since these markers were less heterozygous and therefore less informative. As a result, we reduced the number of genetic markers from 19,465 to 627. To impute the missing data in the remaining part of the genotype data, we tried multiple imputation techniques, including mean, median, and most frequent (Allison, 2001), and found that the median approach led to the most accurate predictions. The yield and environment datasets were complete and did not have missing data.

2.2 Yield Prediction Using Deep Neural Networks (SGD)

We trained two deep neural networks, one for yield and the other for check yield, and then used the difference of their outputs as the prediction for yield difference. This model structure was found to be more effective than using one single neural network for yield difference, because the genotype and environment effects are more directly related to the yield and check yield than their difference.

We used SGD with a mini-batch size of 64. The Adam optimizer was used with a learning rate of 0.03%, which was divided by 2 every 50,000 iterations (Kingma and Ba, 2014). Batch normalization was used before activation for all hidden layers except the first hidden layer.

2.3 Results:

The two deep neural networks were implemented in Python using the Tensorflow open-source software library (Abadi et al., 2016). The training process took approximately 1.4 h on a Tesla K20m GPU for each neural network. We also implemented three other popular prediction models for comparison: The least absolute shrinkage and selection operator (Lasso), shallow neural network (having a single hidden layer with 300 neurons), and regression tree (Breiman, 2017). To ensure fair comparisons, two sets of these three models were built to predict yield and check yield separately, and the differences of their outputs were used as the prediction for the yield difference. All of these models were implemented in Python in the most efficient manner that we were capable of and tested under the same software and hardware environments to ensure fair comparisons.

The following hyperparameters were used for the regression tree. The maximum depth of the tree was set to 10 to avoid overfitting. We set the minimum number of samples required to split an internal node of tree to be 2. All features were used to train the regression tree. We tried different values for the coefficient of L1 term (Ng, 2004) in the Lasso model, and found that values between 0.1 and 0.3 led to the most accurate predictions.

2.4 Analysis:

To compare the individual importance of genotype, soil and weather components in the yield prediction, we obtained the yield prediction results using following models:

DNN(S): This model uses the DNN model to predict the phenotype based on the soil data (without using the genotype and weather data), which is able to capture linear and nonlinear effects of soil conditions.

2.5 Feature Selection:

First, we fed all validation samples to the DNN model and computed the average activation of all neurons in the last hidden layer of the network. We set the gradient of activated neurons to be 1 and the other neurons to be 0. Then, the gradients of the activated neurons were backpropagated to the input space to find the associated input variables based on the magnitude of the gradient (the bigger, the more important). Illustrate the estimated effects of genetic markers, soil conditions, and weather components, respectively. The estimated effects indicate the relative importance of each feature compared to the other features. The effects were normalized within each group namely, genetic markers, soil conditions, and weather components to make the effects comparable.

Chapter 3

SYSTEM

ARCHITECTURE

DESIGN

SYSTEM ARCHITECTURE DESIGN

3.1 System Architecture Design:

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

- **Architectural Design –**

The architectural design of a system emphasizes the design of the systems architecture that describes the structure, behavior and more views of that system and analysis.

- **Logical Design –**

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included.

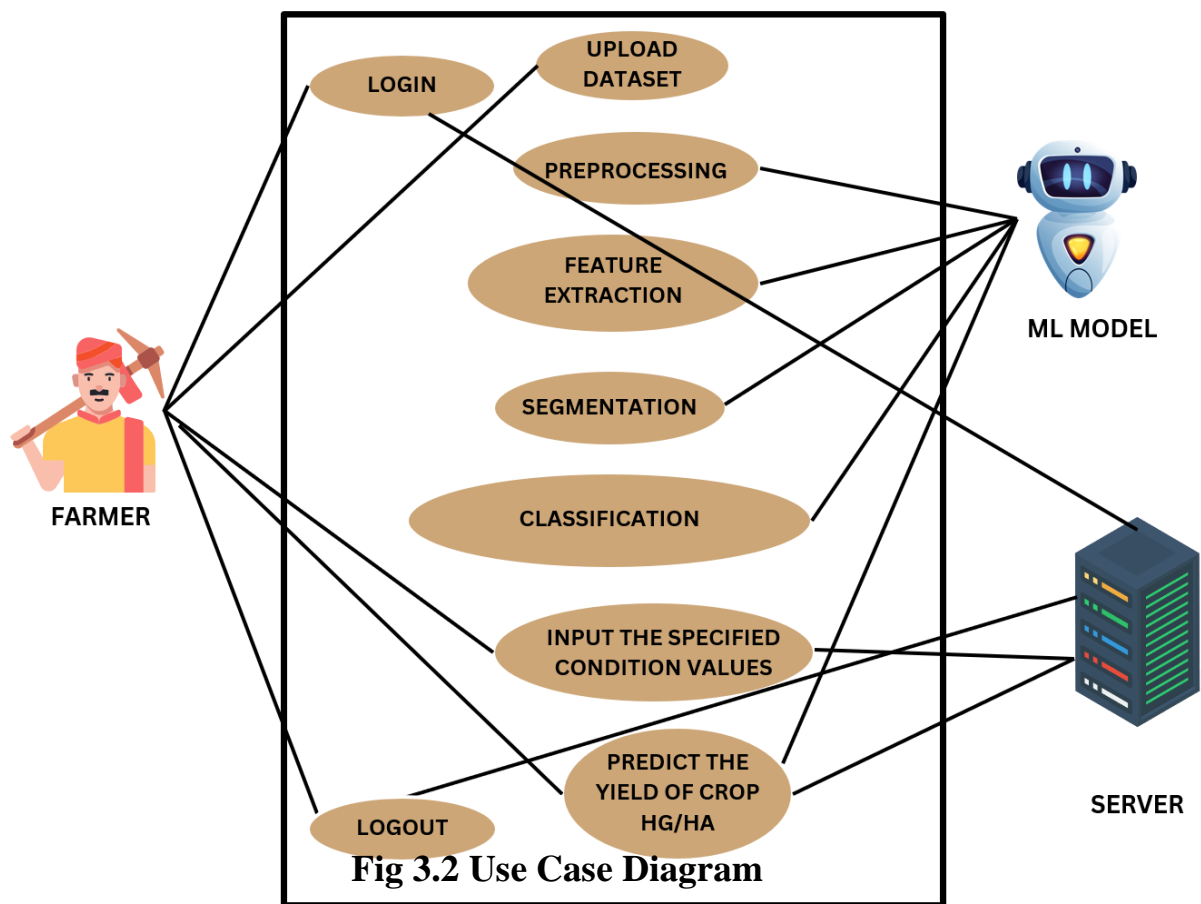
3.2 Use Case:

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

A use case diagram is usually simple. It does not show the detail of the use cases:

- It only summarizes **some of the relationships** between use cases, actors, and systems.

- It does **not show the order** in which steps are performed to achieve the goals of each use case.



Use Case (Description):

1. **Login:** The farmer logs in to the system.
2. **Preprocessing:** The system preprocesses the data uploaded by the farmer. This may involve cleaning the data and formatting it for analysis.
3. **Feature Extraction:** The system extracts features from the data. These features are characteristics of the data that can be used to make predictions.
4. **ML Model:** The system uses a machine learning model to predict the yield of the crop. The model has been trained on a large dataset of labeled data.
5. **Segmentation:** The system may also segment the data. This involves dividing the data into smaller parts.

- 6. Classification:** The system classifies the data. This involves assigning the data to a particular category.
- 7. Input the Specified Condition Values:** The farmer inputs specific condition values, such as country, which crop, average rainfall, average pesticides, average temperature.
- 8. Predict the Yield of Crop (HG/HA):** The system predicts the yield of the crop in quintals per hectare (HG/HA) based on the data uploaded by the farmer and the specified condition values.
- 9. Logout:** The farmer logs out of the system.

Chapter 4

UML DIAGRAMS

4.1 Data Flow Diagram Representation:

A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow there are no decision rules and no loops.

→ Purpose of DFD's:

- **Visualize Data Flow:** DFDs illustrate how data moves within a system.
- **System Understanding:** They aid stakeholders in understanding system architecture and interactions.
- **Process Identification:** DFDs help identify individual system processes.
- **Redundancy Detection:** They reveal redundant processes or data flows.
- **Communication Aid:** DFDs facilitate communication among stakeholders.

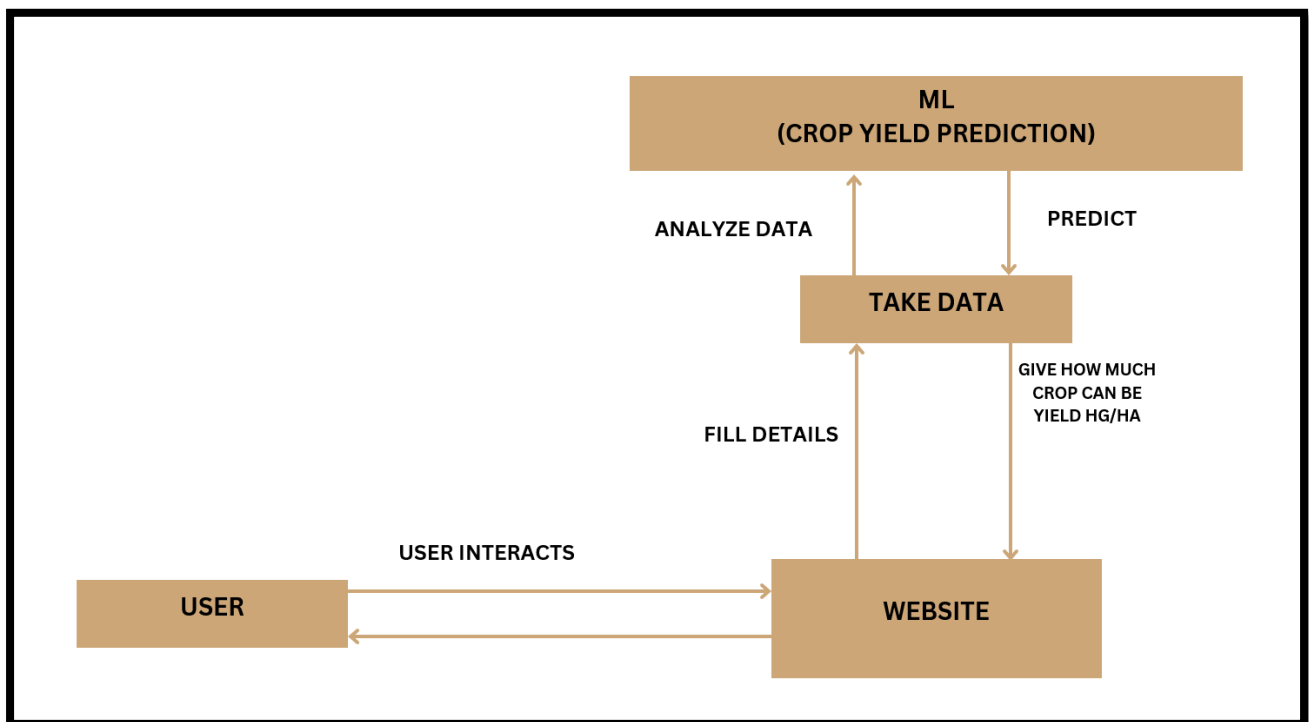


Fig 4.1 DFD Diagram

The above data flow diagram (DFD) is a high-level overview of a crop yield prediction system. It uses the following symbols:

- Rectangles represent processes that transform data.
- Rounded rectangles represent data stores.
- Arrows represent data flows.
- Squares represent external entities that interact with the system.

Here's a breakdown of the DFD:

- **External Entity:** The DFD shows a single external entity labeled “USER” which interacts with the system through a “WEBSITE”. This suggests that the system is accessed through a web interface.
- **Process:** The main process involved in the system is labeled “ML (CROP YIELD PREDICTION)”. This process takes data from the user and uses a machine learning model to predict crop yield.
- **Data Flows:** There are three data flows in the DFD:
 - “ANALYZE DATA” flows from “USER” to “ML (CROP YIELD PREDICTION)”. This data likely includes information about the crop, such as the type of crop, the size of the field, and the planting date.
 - “TAKE DATA” flows from “ML (CROP YIELD PREDICTION)” to an unspecified data store. This data likely includes the predicted crop yield.
 - “GIVE HOW MUCH CROP CAN BE YIELD HG/HA” flows from “ML (CROP YIELD PREDICTION)” to “USER”. This data represents the predicted crop yield in quintals per hectare (HG/HA).

→ **Limitations of a High-Level DFD**

This DFD provides a high-level overview of the system and doesn't detail the inner workings of the ML process. Here are some of the missing details:

- **Data preprocessing:** The DFD doesn't show any data preprocessing steps that might be performed on the user-provided data before it's fed into the machine learning model.
- **Feature extraction:** The DFD doesn't show any feature extraction steps that might be performed on the data. Feature extraction involves identifying the characteristics of the data that are most relevant to the prediction task.
- **Machine learning model:** The DFD doesn't specify the type of machine learning model that is used to predict crop yield. There are many different machine learning models that could be used for this task.

Overall, the DFD provides a basic understanding of the crop yield prediction system. It shows that the system takes data from the user, uses a machine learning model [Stochastic Gradient Descent] to predict crop yield, and returns the prediction to the user.

4.2 Activity Diagram Representation:

Activity diagrams show the flow of one activity to another within a system or process. Even complex systems can be visualized using activity diagrams. They are used within organizations to model customer journeys, to show the process of receiving an order through shipping to the customer, and to model sales processes.

→ Activity Diagram Notations:

- **Activity:** Represented by rounded rectangles, activities depict actions or tasks performed in the system.
- **Initial Node:** Depicted by a filled circle, the initial node signifies the starting point of the activity diagram.
- **Final Node:** Represented by a solid circle with a border, the final node indicates the end point of the activity diagram.
- **Action:** Illustrated by rectangles with rounded corners, actions represent specific steps or operations performed during the activity.
- **Decision Node:** Represented by a diamond shape, decision nodes depict points in the process where a decision is made, leading to different paths.
- **Fork Node:** Depicted by a line splitting into multiple branches, fork nodes show parallel paths where multiple activities can occur simultaneously.
- **Join Node:** Opposite of a fork node, join nodes merge multiple parallel paths back into a single flow.
- **Merge Node:** Similar to a join node, merge nodes combine alternate paths back into a single flow.
- **Control Flow:** Arrows connecting different elements indicate the sequence in which activities are performed.
- **Guard Condition:** Text within square brackets on edges leaving decision nodes represents conditions that determine the flow of control.
- **Object Node:** Represented by a rectangle with a vertical line on the left, object nodes signify data or objects used within activities.
- **Swimlanes:** Vertical or horizontal partitions within the diagram represent different actors, roles, or organizational units involved in the activities.

→ **Uses of Activity Diagram:**

- **Process Modeling:** Activity diagrams are used to model and visualize the sequence of activities and actions within a system or process.
- **Behavioral Analysis:** They aid in analyzing the behavior of a system by illustrating the flow of control and data among various activities.
- **Communication Tool:** Activity diagrams serve as a communication tool between stakeholders, facilitating discussions about system behavior and requirements.
- **Process Improvement:** They help identify bottlenecks, redundancies, and inefficiencies in processes, enabling organizations to optimize and improve their workflows.

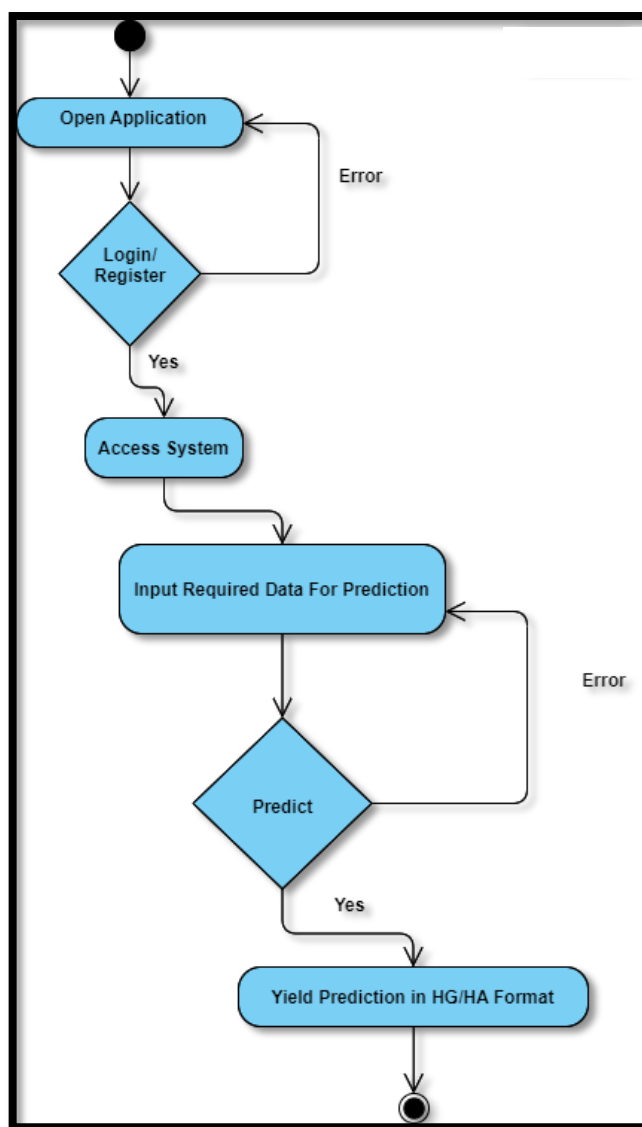


Fig 4.2 Activity Diagram

→ Activities –

- **Start (A):** This is the starting point of the process.
- **Login (B):** The farmer logs in to the system.
- **Data Input (C):** The farmer enters data about the crop, field size, planting date, and other relevant information.
- **Data Preprocessing (D):** The system cleans and formats the data entered by the farmer. This might involve handling missing values, outliers, and ensuring consistency in the data format.
- **Feature Extraction (E):** The system extracts relevant features from the preprocessed data. These features are characteristics of the data that can be used to predict crop yield. For example, features might include average temperature, rainfall, soil nutrients, and historical yield data.
- **Calculate Loss (G):** The system calculates the difference between the predicted yield (from the model) and the actual yield (if available). This difference is called the loss.
- **Update Weights (H):** The system updates the weights of the machine learning model using the stochastic gradient descent algorithm. This algorithm iteratively adjusts the weights to minimize the loss between predicted and actual yield.
- **Predict Yield (J) [Yes]:** If the stopping criteria are met (Yes), the system predicts the crop yield in quintals per hectare (HG/HA) using the final weights of the machine learning model.
- **Predict Yield (J) [No]:** If the stopping criteria are not met (No), the system continues to update the weights (H) and calculate loss (G). This loop continues until the stopping criteria are satisfied.
- **Display Results (K):** The predicted crop yield is presented to the farmer.
- **Logout (N):** The farmer logs out of the system.
- **End (O):** This is the ending point of the process.

Chapter 5

IMPLEMENTATION

5.1 Implementation:

→ Model Accuracy –


```
pf=PolynomialFeatures(degree=2)
X_new_train=pf.fit_transform(X_train)
X_new_test=pf.fit_transform(X_test)
model=SGDRegressor()
model.fit(X_new_train,y_train)
y_pred_test=model.predict(X_new_test)
y_pred_train=model.predict(X_new_train)
test_score=r2_score(y_test,y_pred_test)
train_score=r2_score(y_train,y_pred_train)
print(train_score)
print(test_score)
```

0.9486267940146509

0.943761127227828

→ Crop Prediction –

Crop Yield Prediction



Select Language

English

Country

India

Crop

Wheat

Average Rainfall (mm-per-year)

1400.00

-

+

Pesticides per Tonnes Use (tonnes of active ingredients)

200.00

-

+

Average Temperature (degree celcius)

50.00

-

+

1 hg = 100 grams

1 ha = 2 acres

Predict

The Production of Crop Yields: The
Production of Crop Yields:
[70307.55039205] hg/ha yield

→ Multilingual –

Crop Yield Prediction



Select Language

Hindi

देश

India

फसल

Wheat

औसत वार्षिक वर्षा (मिमी प्रतिवर्ष)

1400.00

प्रति टन उपयोग किए गए कीटनाशक (सक्रिय तत्वों के टन)

200.00

औसत तापमान (डिग्री सेल्सियस)

50.00

1 hg = 100 grams

1 ha = 2 acres

अग्रिम

**फसल उत्पादन: The Production of
Crop Yields: [70307.55039205]
hg/ha yield**

Chapter 6

CONCLUSION

&

REFERENCES

6.1 Conclusion:

The utilization of Stochastic Gradient Descent (SGD) algorithm for crop yield prediction presents a promising approach in agriculture. By leveraging machine learning techniques, particularly SGD, we can effectively analyze vast amounts of agricultural data to predict crop yields with considerable accuracy. This technology offers numerous benefits including improved resource allocation, optimized farming practices, and enhanced decision-making for farmers and stakeholders in the agricultural sector. However, it's crucial to continuously refine and validate predictive models, ensuring their reliability across diverse environmental conditions and crop varieties. Additionally, collaboration between data scientists, agronomists, and farmers is essential to tailor these predictive models to specific agricultural contexts and maximize their practical utility in real-world farming scenarios. Overall, the integration of SGD-based crop yield prediction systems has the potential to revolutionize agricultural practices, contributing to increased productivity, sustainability, and food security on a global scale.

ROSPL Part B : Adding a Github Readme file

📖 README



We are doing this for our ROSPL Project

Krushivikas

Farmers face uncertainties in crop yield due to various factors such as weather conditions, soil quality, pest infestations, and management practices. Predicting crop yields accurately is crucial for farmers to make informed decisions regarding crop selection, resource allocation, and marketing strategies. Traditional methods of crop yield prediction often lack accuracy and scalability, hindering farmers' ability to optimize agricultural productivity and profitability. Therefore, there is a need for an efficient and reliable crop yield prediction system that leverages advanced machine learning algorithms to provide accurate forecasts and support sustainable farming practices.

Table of Contents

1. [Features](#)
2. [Technologies](#)
3. [Output](#)
4. [How to Run This Project](#)

Features

- **Access to Crucial Information:** 'Krushivikas' provides vital info on govt. and loan schemes, aiding farming decisions.
- **Climate Resilience:** 'Krushivikas' guides crop choices and practices for climate adaptation in agriculture.
- **Financial Inclusion:** 'Krushivikas' simplifies loan applications, fostering financial support for farmers' practices.
- **Community Support:** 'Krushivikas' cultivates a supportive community for shared knowledge and resilience.

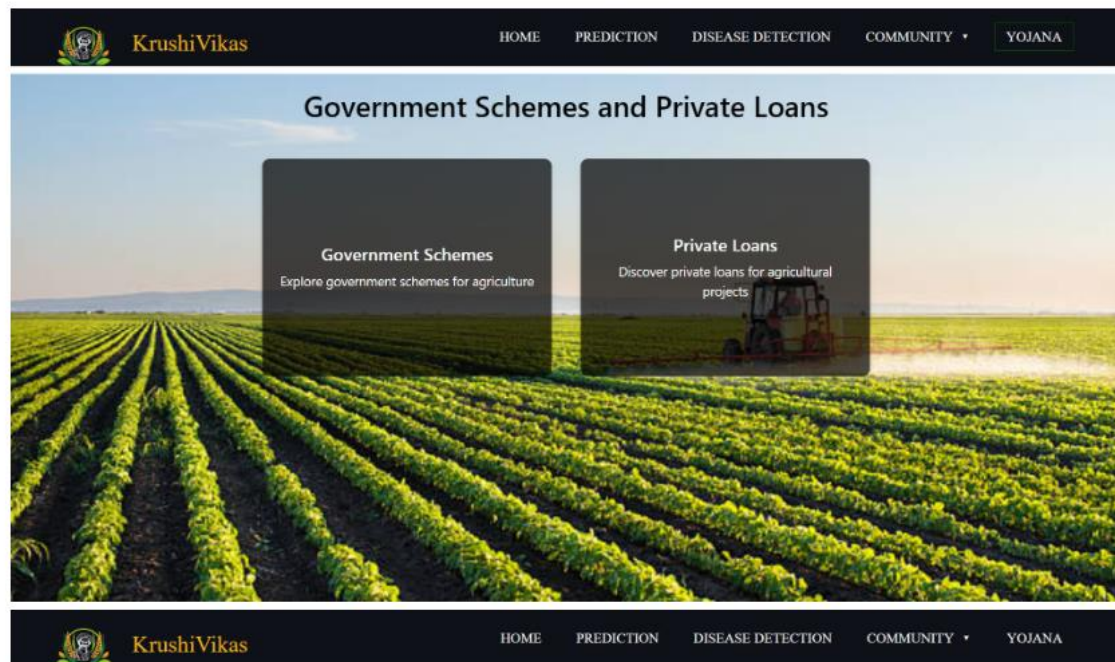
Technologies

- React
- Html
- CSS
- JavaScript
- Firebase
- Flask
- Bootstrap


Output



Welcome to **KrushiVikas**



Available Loans




Axis Bank

Description -

- 1. Interest Rate: 10.49% per annum.
- 2. Loan Amount: Upto 40 Lakh
- 3. Tenure: 1-7 years
- 4. Processing Fee: Upto 2%

[Apply](#)



FEDERAL BANK


YOUR PERFECT BANKING PARTNER

Federal Bank

Description -

- 1. Interest Rate: 11.49% per annum.
- 2. Loan Amount: Upto 25 Lakh
- 3. Tenure: Upto 5 years
- 4. Processing Fee: Upto 3%

[Apply](#)



HDFC BANK

HDFC Bank

Description -

- 1. Interest Rate: 10.50% per annum.
- 2. Loan Amount: Upto 40 Lakh
- 3. Tenure: Upto 6 years
- 4. Processing Fee: Upto Rs 4999

[Apply](#)



IDFC Bank

Description -

1. Interest Rate: 10.79% - 36% per annum.
2. Loan Amount: Upto 1 Crore
3. Tenure: 5 years
4. Processing Fee: Upto 4%

[Apply](#)



IndusInd Bank

Description -

1. Interest Rate: 10.49% - 26% per annum.
2. Loan Amount: Upto 40 Lakh
3. Tenure: 1-5 years
4. Processing Fee: Upto 4%

[Apply](#)



L&T Bank

Description -

1. Interest Rate: 12% - 24% per annum.
2. Loan Amount: Upto 07 Lakh
3. Tenure: Upto 4 years
4. Processing Fee: Upto 2%

[Apply](#)



KreditBee Personal Loan

Description -

1. Interest Rate: 16% - 29.95%per annum.
2. Loan Amount: Upto 4 Lakh
3. Tenure: Upto 2 years
4. Processing Fee: Rs 1250.

[Apply](#)



MoneyTap Personal Loan

Description -

1. Interest Rate: 13% per annum.
2. Loan Amount: Upto 5 Lakh
3. Tenure: Upto 3 years
4. Processing Fee: For loan amounts between Rs 3,000 and Rs 5,000: Rs 199 + GST. For loan amounts of Rs 25,000 and above: 2% of the total amount + GST

[Apply](#)



AdityaBirla Personal Loan


Description -

1. Interest Rate: Approx 13% per annum.
2. Loan Amount: Upto 50 Lakh
3. Tenure: Upto 7 years
4. Processing Fee: Upto 3%

[Apply](#)



© 2023 Copyright:KrushiVikas.com



KrushiVikas

[HOME](#)
[PREDICTION](#)
[DISEASE DETECTION](#)
[COMMUNITY](#)
[YOJANA](#)

Available Schemes



Pradhan Mantri Kisan Samman Nidhi (PM-KISAN)

PM-KISAN is a central sector scheme launched on 24th February 2019 to supplement financial needs of land holding farmers, subject to exclusions. Under the scheme, financial benefit of Rs. 6000/- per year is transferred in three equal four-monthly installments into the bank accounts of farmers' families across the country, through Direct Benefit Transfer (DBT) mode.

Eligibility criteria

1. Small and Marginal Farmers: The scheme is designed for small and marginal farmers who own cultivable land. Small farmers own up to 2 hectares of land, while marginal farmers own between 2 and 5 hectares.
2. Financial Support: Eligible farmers receive Rs. 6000 per year in three equal instalments of Rs. 2000 each, credited directly to their bank accounts.
3. Funding and Disbursement: The scheme is funded entirely by the Government of India, with an initial allocation of Rs. 75,000 crore per year. The money is disbursed through Direct Benefit Transfer (DBT) to the beneficiaries' bank accounts.

[Apply](#)



Pradhan Mantri Kisan Maandhan Yojana (PM-KMY)

Pradhan Mantri Kisan Maandhan Yojna (PMKMY) is a central sector scheme launched on 12th September 2019 to provide security to the most vulnerable farmer families. PM-KMY is contributory scheme, small and marginal farmers (SMFs), subject to exclusion criteria, can opt to become member of the scheme by paying monthly subscription to the Pension Fund.

Eligibility criteria:

1. PM-KMY is a pension scheme for small and marginal farmers aged 18 to 40.
2. It offers a monthly pension of Rs. 3000 upon reaching 60 years, with contributions ranging from Rs. 55 to Rs. 200 per month.
3. Spouses are also eligible for a pension, and the scheme is managed by LIC.

[Apply](#)




Pradhan Mantri Fasal Bima Yojana (PMFBY)

MFBY was launched in 2016 in order to provide a simple and affordable crop insurance product to ensure comprehensive risk cover for crops to farmers against all non-preventable natural risks from pre-sowing to post-harvest and to provide adequate claim amount.

Eligibility criteria:

1. To be eligible for crop insurance, farmers must be cultivators or sharecroppers on the insured land, possessing a valid land ownership certificate or a land tenancy agreement.
2. Farmers must apply for insurance coverage within the prescribed time frame, typically within two weeks of the start of the sowing season, and must not have received compensation for the same crop loss from any other source.
3. Additionally, farmers must have a valid Kisan Credit Card (KCC) or obtain one within six months of enrollment, along with a valid bank account and identity proof provided at the time of enrollment.


[Apply](#)



Tech Farming

I used to rely solely on traditional farming methods, but incorporating technology has revolutionized my farm. From automated irrigation systems to drones for crop monitoring, technology has not only increased my efficiency but also improved my crop yields significantly.


@Kirti Eppanapelli



Field Victory

Starting with a small piece of land and a dream, I faced numerous challenges in my early years as a farmer. But through perseverance and learning from failures, I gradually improved my farming techniques. Today, I proudly harvest healthy crops, thanks to my hard work and dedication.

@Kirti Eppanapelli




Community Support

During tough times, the support of my fellow farmers and the local community has been my greatest strength. Whether it's sharing resources or offering moral support, the sense of camaraderie among farmers is truly remarkable and has helped me overcome many challenges.

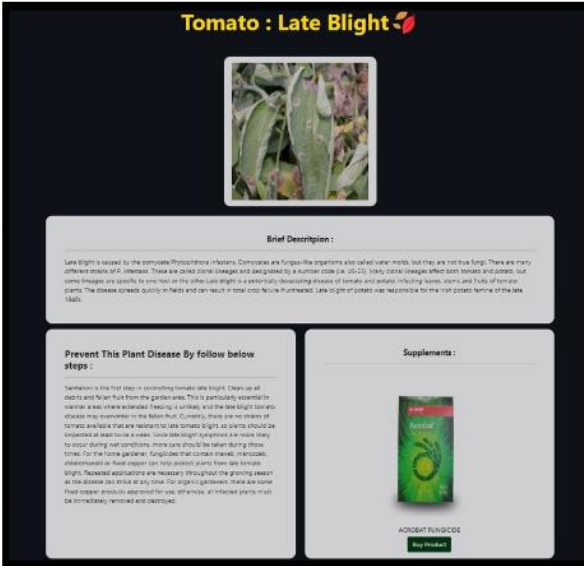
@Kirti Eppanapelli

Crop Yield Prediction



The Production of Crop Yields: The Production of Crop Yields: [23430.82607834] hg/ha yield

Tomato : Late Blight




Brief Description :

Late blight is caused by the oomycete *Phytophthora infestans*. Oomycetes are fungus-like organisms also called water molds, but they are not true fungi. There are many different strains of *P. infestans*. These are called clone lineages and designated by a number code (e.g. 00-10). Early clone lineages affect both tomatoes and potatoes, but some lineages are specific to one host or the other. Late blight is a potentially devastating disease of tomato and potato, inflicting leaves, stems and fruit of tomato plants. The disease spreads quickly in fields and can result in total crop failure if untreated. Late blight on potato was responsible for the Irish potato famine of the late 1840s.

Prevent This Plant Disease By follow below steps :

Sanitation is the first step in controlling tomato late blight. Clean up all debris and plant parts from the garden area. This is particularly essential to remove areas where late blight is common, and the late blight tomato disease may develop in the future. Currently, there are no clones of tomato available that are resistant to late blight, so plants should be inspected at least once a week. Once late blight symptoms are visible, they should be removed. If late blight is present, more care should be taken during those times. For the home gardener, fungicides that contain mancozeb, mancozeb, or chlorothalonil can help protect plants from late blight. Fungicide applications are necessary throughout the growing season. As the disease is not an easy one to control, for organic gardeners, there are some food grade products approved for use. Otherwise, all infected plants must be immediately removed and destroyed.

Supplements :



ACROBAT FUNGICIDE
Buy Product

How to Run This Project

1. Clone the Repository:

```
git clone https://github.com/Ruchita-20/Quassar_Hackathon_krushivikas.git
```



2. Navigate to the Project Directory:

```
cd Quassar_Hackathon_krushivikas
```



3. Install the NPM Packages:

```
npm install
```



4. Start the React App:

```
npm start
```



6.2 References:

- [1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). “TensorFlow: a system for large scale machine learning,” in OSDI’16 Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation, Vol. 16, (Savannah, GA), 265–283.

- [2] Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. 5, 157–166. doi: 10.1109/72.279181