

A

Mini Project on

**MACHINE LEARNING BASED ON IRRIGATION  
SCHEDULING FOR SMART FARMING SYSTEMS  
A THESIS**

*submitted*

*in the partial fulfillment of the requirements for*

*the award of the degree of*

**Bachelor of Technology**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

*by*

CH.Srivani	-	22E45A0551
A.Udayeni Reddy	-	21E41A05C4
S.Ravi Teja	-	21E41A05B8
S.Dinesh Reddy	-	22E45A0560

Under the supervision of

**S.Krishna Reddy**

**Assistant Professor**

**Department of Computer Science and Engineering**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SREE DATTHA INSTITUTE OF ENGINEERING AND SCIENCE**

(Approved by AICTE New Delhi, Accredited by NAAC, Affiliate to JNTUH)

SHERIGUDA (v), IBRAHIMPATNAM (M), RANGAREDDY -501510

**2024-2025**

**SREE DATTHA INSTITUTE OF ENGINEERING AND SCIENCE**  
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**DECLARATION**

We are hereby declaring that the mini project report titled “**MACHINE LEARNING BASED ON IRRIGATION SCHEDULING FOR SMART FARMING SYSTEMS**” under the guidance of **S. Krishna Reddy, Sree Dattha Institute of Engineering and Science**, Ibrahimpatnam is submitted in partial fulfillment of the requirement for the award of B. Tech. in Computer Science and Engineering is a record of bonafide work carried out by us and the results embodied in this project have not been reproduced or copied from any source.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

**Name of the Students**

CH.Srivani	22E45A0551
A.Udayeni Reddy	21E41A05C4
S.Dinesh Reddy	22E45A0560
S.Ravi Teja	21E41A05B8

**SREE DATTHA INSTITUTE OF ENGINEERING AND SCIENCE**  
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

This is to certify that the mini project entitled “**MACHINE LEARNING BASED ON IRRIGATION SCHEDULING FOR SMART FARMING SYSTEMS**” is being submitted by **CH Srivani(22E45A0551), A Udayeni Reddy(21E41A05C4), S Dinesh Reddy(22E45A0560), S Ravi Teja(21E41A05B8)** in partial fulfillment of the requirements for the award of B. Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of Bonafide work carried out by them under my guidance and supervision during the academic year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**Internal Guide**

S.Krishna Reddy

**HOD**

Dr. SK. Mahaboob Basha

**External Examiner**

Submitted for viva Voice Examination held on \_\_\_\_\_

## ACKNOWLEDGEMENT

Apart from our efforts, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We would like to express our sincere gratitude to Chairman Sri. **G. Panduranga Reddy**, and Vice-Chairman **Dr. GNV Vibhav Reddy** for providing excellent infrastructure and a nice atmosphere throughout this project. We are obliged to **Dr. S. Venkata Achuta Rao**, Principal for being cooperative throughout this project.

We are also thankful to **Dr. Sk Mahaboob Basha**, Head of the Department & Professor Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We take this opportunity to express my profound gratitude and deep regard of Internal guide **S.Krishna Reddy**, Assistant Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

The guidance and support were received from all the members of **Sree Dattha Institute of Engineering and Science** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

## ABSTRACT

This research focuses on developing an intelligent irrigation scheduling system using machine learning techniques to optimize water use in agriculture. Traditional irrigation systems often suffer from inefficiencies such as over- or under-irrigation, labor intensiveness, and lack of precision. To overcome these challenges, the project leverages real-time environmental data, including soil moisture, temperature, and crop type, to predict the optimal times for activating irrigation pumps. The primary goal of the project is to address the inefficiency and inaccuracy of traditional irrigation scheduling methods. By integrating machine learning into irrigation management, the system aims to reduce water waste, enhance crop health, and minimize labor requirements. The motivation for this project stems from the urgent need to optimize water use in agriculture, given increasing water scarcity and the impact of climate change. The proposed system comprises several key components. Firstly, data collection sensors gather information on soil moisture, temperature, and crop type, which is then preprocessed for model training. Machine learning models, including Bernoulli Naive Bayes and Ridge Classifier, are trained on historical data to predict irrigation needs. These models are evaluated using performance metrics, and the best-performing model is used to make real-time predictions. Finally, the system integrates with irrigation infrastructure to automate pump control based on model predictions.

## LIST OF FIGURES

S.NO.	TITLE	PAGE NO.
1	Block Diagram of the Proposed System	14
2	Splitting the dataset	18
3	Architectural diagram of Ridge Classifier model	19
4	Class Diagram	25
5	Data Flow Diagram	25
6	Sequence Diagram	26
7	Activity Diagram	26
8	Deployment Diagram	27
9	Use Case Diagram	27
10	Presents the Sample Dataset of this project	50
11	Label Encoded Dataset	50
12	Count of each label in Dataset	50
13	Performance metrics of Bernoulli Naïve Bayes Classifier	50
14	Confusion matrix of Bernoulli Naïve Bayes Classifier	51
15	Performance metrics of Ridge Classifier	52
16	Confusion matrix of Ridge Classifier	52
18	Uploading the test dataset for model prediction	53
19	Model Prediction on Uploaded Test data	54
20	Performance metrics of Bernoulli Naïve Bayes	54
21	Classifier and Ridge Classifier Model	55

## LIST OF CONTENT

S.NO	CONTENTS	PAGE NO.
1	<b>INTRODUCTION</b>	1
	1.1 History	2
	1.2 Research Motivation	3
	1.3 Problem Statement	4
	1.4 Applications	5
2	<b>LITERATURE SURVEY</b>	7
	2.1 Literature Survey	8
3	<b>EXISTING SYSTEM</b>	10
	3.1 Naïve Bayes	11
4	<b>PROPOSED SYSTEM</b>	13
	4.1 Overview	14
	4.2 Data Preprocessing	15
	4.3 Splitting the Dataset	17
	4.4 Ridge Classifier Model	19
5	<b>UML DIAGRAMS</b>	24
	5.1 Class Diagram	25
	5.2 Data Flow Diagram	25
	5.3 Sequence Diagram	26
	5.4 Activity Diagram	26
	5.5 Deployment Diagram	27
	5.6 Use Case Diagram	27
6	<b>SOFTWARE ENVIRONMENT</b>	28
	6.1 Modules Used in Project	29
7	<b>SYSTEM REQUIREMENTS</b>	31
	7.1 Software Requirements	32

	7.2 Hardware Requirements	32
<b>8</b>	<b>FUNCTIONAL REQUIREMENTS</b>	33
	8.1 Function Requirements	34
<b>9</b>	<b>SOURCE CODE</b>	40
	9.1 SOURCE CODE	41
<b>10</b>	<b>RESULTS AND DISCUSSION</b>	46
	10.1 Implementation Description	47
	10.2 Dataset Description	48
	10.3 Results Description	49
<b>11</b>	<b>CONCLUSION AND FUTURE SCOPE</b>	56
	11.1 Conclusion	57
	11.2 Future Scope	57
<b>12</b>	<b>REFERENCES</b>	59



# INTRODUCTION

# CHAPTER 1

## INTRODUCTION

### 1.1 History

The history of irrigation dates back thousands of years, with early civilizations developing ingenious methods to manage water for agriculture. [1] Ancient societies such as the Egyptians, Mesopotamians, and Indus Valley civilizations built intricate irrigation systems using canals, ditches, and reservoirs to control the flow of water to their crops. [2] These early techniques laid the foundation for modern irrigation practices, demonstrating humanity's innate desire to harness water for agricultural purposes.

[3] Throughout history, irrigation has played a vital role in supporting agricultural development and sustaining civilizations. [4] The advent of irrigation allowed farmers to cultivate crops in arid regions and increase food production, leading to population growth and societal advancement. [5] In ancient Rome, sophisticated aqueducts were constructed to transport water over long distances, enabling large-scale farming and urbanization.

During the Middle Ages, Islamic scholars made significant contributions to irrigation technology, developing innovative techniques such as qanats and water wheels. [6] These advancements improved water distribution and irrigation efficiency, fostering agricultural productivity and economic prosperity in regions such as Spain and North Africa.

In the 19th and 20th centuries, the Industrial Revolution brought about further innovations in irrigation technology. [7] The invention of steam engines and electric pumps revolutionized water extraction and distribution, allowing for the expansion of irrigation networks and the intensification of agriculture. [8] Large-scale irrigation projects, such as the construction of dams and reservoirs, transformed vast tracts of land into fertile agricultural regions, contributing to global food security.

In recent decades, the focus has shifted towards sustainable irrigation practices and the integration of technology into agricultural water management. [9] Modern irrigation systems incorporate precision irrigation techniques, such as drip and sprinkler irrigation, to optimize water use and minimize waste. Furthermore, advances in remote sensing, data analytics, and

automation have enabled the development of smart irrigation systems that dynamically adjust water application based on real-time environmental conditions.

Today, irrigation continues to be a cornerstone of global agriculture, supporting the cultivation of crops in diverse climates and environments. [10] As the world faces growing challenges such as climate change, water scarcity, and population growth, the importance of efficient and sustainable irrigation practices has never been greater. By building upon centuries of innovation and harnessing the power of technology, the future of irrigation holds immense potential to ensure food security, promote environmental stewardship, and enhance livelihoods worldwide.

## **1.2 Research Motivation**

The motivation for research in intelligent irrigation scheduling stems from the pressing need to address the challenges facing modern agriculture, particularly in the context of water management. With global population growth, urbanization, and climate change exerting increasing pressure on water resources, there is a growing imperative to optimize water use in agriculture while ensuring food security and environmental sustainability.

Traditional irrigation methods, characterized by fixed schedules or manual observation, are often inefficient and prone to water waste. Over-irrigation can lead to waterlogging, nutrient leaching, and environmental degradation, while under-irrigation can result in reduced crop yields and economic losses. Moreover, labor-intensive irrigation practices are unsustainable in the long term, particularly in regions facing labor shortages or rising labor costs.

The advent of machine learning and data analytics presents an exciting opportunity to revolutionize irrigation management by leveraging real-time environmental data to make informed decisions. By integrating sensors, data analytics, and automation, intelligent irrigation systems can dynamically adjust water application based on crop water requirements, soil moisture levels, weather forecasts, and other relevant factors.

The research motivation for intelligent irrigation scheduling is thus twofold: to enhance agricultural productivity and resource efficiency while minimizing environmental impact. By developing advanced algorithms and decision support tools, researchers aim to optimize irrigation scheduling, reduce water consumption, and improve crop yields, ultimately contributing to sustainable agriculture and food security in a changing climate.

### **1.3 Problem Statement**

The problem addressed by intelligent irrigation scheduling is the inefficiency and ineffectiveness of traditional irrigation methods in optimizing water use and maximizing crop yield. Traditional irrigation systems, reliant on fixed schedules or manual observation, often result in over- or under-irrigation, leading to water waste, reduced crop productivity, and environmental degradation.

The main challenges facing traditional irrigation methods include:

- Lack of Precision: Fixed irrigation schedules do not account for variations in soil moisture, crop water requirements, and environmental conditions, leading to suboptimal water application.
- Labor Intensiveness: Manual observation and control of irrigation systems require significant labor input, which is unsustainable and economically inefficient, particularly in regions facing labor shortages or rising labor costs.
- Environmental Impact: Over-irrigation can lead to waterlogging, soil erosion, nutrient leaching, and contamination of water bodies, contributing to environmental degradation and ecosystem disruption.

— Economic Viability: Inefficient water use and reduced crop yields resulting from traditional irrigation methods can lead to financial losses for farmers and hinder agricultural development and economic growth.

## **1.4 Applications**

Intelligent irrigation scheduling has a wide range of applications across various sectors, including agriculture, water management, and environmental conservation. Some of the key applications include:

- Precision Agriculture: By optimizing water use and fertilizer application based on real-time environmental data, intelligent irrigation systems can enhance crop yield, quality, and profitability while minimizing resource inputs and environmental impact.
- Water Conservation: By reducing water waste and improving irrigation efficiency, intelligent irrigation scheduling contributes to water conservation efforts, ensuring the sustainable management of finite water resources and mitigating the impacts of water scarcity on agriculture and ecosystems.
- Environmental Sustainability: Intelligent irrigation scheduling promotes environmentally sustainable agriculture by minimizing the use of agrochemicals, reducing soil erosion and nutrient runoff, and preserving biodiversity and ecosystem services.

- Climate Resilience: By adapting irrigation schedules to changing climatic conditions and weather patterns, intelligent irrigation systems help farmers mitigate the impacts of climate change on crop production and enhance resilience to droughts, floods, and other extreme weather events.
- Urban Agriculture: In urban and peri-urban areas, intelligent irrigation systems enable the efficient cultivation of crops in limited spaces, such as rooftop gardens, vertical farms, and urban green spaces, promoting local food production, community engagement, and food security.
- Smart Cities: In the context of smart city initiatives, intelligent irrigation scheduling contributes to sustainable urban development by reducing water consumption, enhancing green infrastructure, and creating resilient and livable urban environments.
- Research and Innovation: Intelligent irrigation scheduling provides a platform for ongoing research and innovation in agricultural water management, data analytics, and automation technologies, driving advancements in precision agriculture and sustainable farming practices.
- International Development: In regions facing water scarcity, food insecurity, and poverty, intelligent irrigation systems offer opportunities for improving agricultural

# **LITERATURE SURVEY**

## **CHAPTER 2**

### **LITERATURE SURVEY**

Ahmed et al. [10] presented the implementation and design of smart irrigation scheme with help of IoT technique that is utilized to automate the irrigation procedure from agricultural fields. It can be predictable that scheme will make the best change for the farmers to irrigate their field effectively, and eliminate the field in watering, that can stress the plant. The established scheme is classified into 3 portions: user side, sensing side, and cloud side. They utilized Microsoft Azure IoT Hub as a fundamental framework for coordinating the communication among the 3 sides. Blasi et al. [11] improved the irrigation procedure and provides irrigation water to the maximum range using AI for constructing smart irrigation schemes. The sensor measures the temperature & humidity from the soil each 10 min. It can be prevented the automated irrigation procedure when the humidity was higher and allows it when the humidity was lower. The smart automated irrigation scheme is made by DT method that is an ML technique which trains the scheme based on gathered data for creating the module which would be utilized for examining and predicting the residual data.

The projected solution would be established by developing a distributed WSN, where all the regions of farm will be enclosed with several sensor models that would be transferring data on a standard server. The ML method would assist prediction of the irrigation pattern depending upon weather conditions and crops. Hence, a sustained method for irrigation is given in [12]. Hassan-Esfahani et al. [13] introduced a modelling method for an optimum water distribution relation to maximize irrigation regularity and minimize yield decrease. Local weather data, field measurements, and Landsat images have been utilized for developing a module which defines the field condition by a soil water balance method. This method has predicted the elements of soil water balance and optimization of water allocation module. Every module includes 2 sub components which consider 2 purposes. The optimization sub module utilizes GA for identifying optimum crop water application rates depending upon sensitivity, crop type, and growth stage to water stress.



In Shen et al. [14], the water saving irrigation scheme for winter wheat depending upon the DSSAT module and GA is improved for distinct historical years (1970–2017). Hence, a decision-making technique to defining either for irrigating development phase of winter wheat was established by SVM method depending upon quantity of precipitations in the initial phase of winter wheat and the quantity of irrigations. Navarro-Hellín et al. [15] allow a closed loop control system for adapting the DSS for estimation errors and local perturbations. The 2 ML methods, ANFIS & PLSR, are presented as reasoning engine of this SIDSS. Cardoso et al. [16] presented ML methods using the aim of forecasting the appropriate time of day for water administration to agricultural fields. Using higher quantity of data formerly gathered by WSN in agricultural fields it can examine techniques that permit for predicting the optimal time to water management for eliminating scheduled irrigation which always results in excess of water being the major goal of the scheme for saving these similar natural resources.

For adapting water management, ML methods have been investigated for predicting the optimal time of day for water administration [17]. The research methods like DT, SVM, RF, and NN are the most attained outcomes was RF, giving 84.6% accuracy. Also the ML solution, a technique was established for calculating the quantity of water required for managing the field in analyses. Munir et al. [18] used a smart method that can professionally utilize ontology for making 50% of decision and another 50% of decision based on sensor information values. The decision in ontology and sensor value cooperatively becomes the source of last decision that is the outcome of an ML method KNN. This technique avoids the overburden of the IoT server for processing data however it decreases the latency rate. The goal of [19] is the research of many learning methods for determining the goodness and error comparative for expert decisions. The 9 orchards have been verified in 2018 by LR, RFR, and SVR approaches as engine of the IDSS presented. In Abioye et al. [20], an enhanced data driven and monitoring modelling of the dynamics of variables affected the irrigation of mustard leaf plants is proposed.

# **EXISTING SYSTEM**

# CHAPTER 3

## EXISTING SYSTEM

### 3.1 NAÏVE BAYES

#### What is the Naive Bayes algorithm

Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature.

The presence or absence of a feature does not affect the presence or absence of the other feature.

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. ... Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

#### How Naive Bayes works

Naive Bayes is a powerful algorithm that is used for text data analysis and with problems with multiple classes. To understand Naive Bayes theorem's working, it is important to understand the Bayes theorem concept first as it is based on the latter.

Bayes theorem, formulated by Thomas Bayes, calculates the probability of an event occurring based on the prior knowledge of conditions related to an event. It is based on the following formula:

$$P(A|B) = P(A) * P(B|A)/P(B)$$

Where we are calculating the probability of class A when predictor B is already provided.

$P(B)$  = prior probability of B

$P(A)$  = prior probability of class A

$P(B|A)$  = occurrence of predictor B given class A probability

### **3.1 Drawbacks of Existing system**

The Naive Bayes algorithm has the following disadvantages:

- The prediction accuracy of this algorithm is lower than the other probability algorithms.
- It is not suitable for regression. Naive Bayes algorithm is only used for textual data classification and cannot be used to predict numeric values.

# **PROPOSED SYSTEM**

# CHAPTER 4

## PROPOSED SYSTEM

### 4.1 Overview

The smart irrigation scheduling system using machine learning techniques is designed to predict whether irrigation pumps should be turned on or off based on various input parameters.

- **Data Handling and Preprocessing:** Importing and preprocessing the dataset. The data is read from a CSV file into a pandasDataFrame. The preprocessing stage includes checking for null values and encoding categorical variables into numerical values using label encoding. This is crucial for ensuring that the machine learning models can process the data correctly. The preprocessing stage ensures that the data is clean and ready for further analysis and model training.
- **Data Visualization:** Data visualization is performed to understand the distribution and relationships within the dataset. Various plots are generated to visualize the data, such as count plots for categorical variables and correlation matrices to understand the relationships between different features. These visualizations help in identifying patterns and insights in the data, which can be useful for feature selection and understanding the behavior of different variables in relation to the target variable, which is the pump status in this case.
- **Model Training:** The core of the project involves training machine learning models. Two models are primarily used: **Bernoulli Naive Bayes** and **Ridge Classifier**. The training process involves splitting the dataset into training and testing sets. The models are then trained on the training data and saved for future use. The use of different models allows for comparison and selection of the best-performing model for the irrigation scheduling task.

– **Model Evaluation:** Once the models are trained, their performance is evaluated using several metrics, including precision, recall, F1 score, and accuracy. These metrics provide a comprehensive understanding of how well the models are performing. Confusion matrices and classification reports are also generated to give a detailed view of the model performance on the test data. This stage ensures that the models are reliable and can make accurate predictions when deployed.

E– **Prediction and Testing:** After evaluation, the best-performing model is used to make predictions on new, unseen data. The system reads the test data, processes it in the same way as the training data, and makes predictions using the trained model. These predictions determine whether the irrigation pump should be turned on or off for each input record. This stage demonstrates the practical application of the trained model in making real-time irrigation decisions.

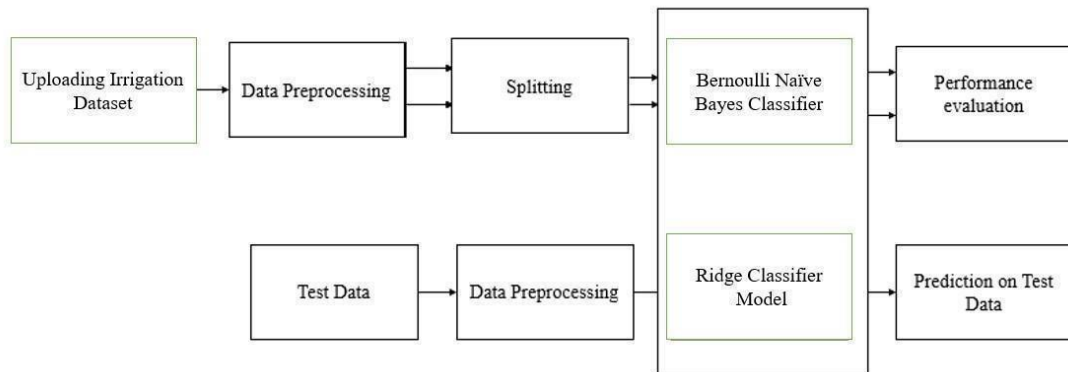


Fig. 1 Block Diagram of the Proposed System.

## 4.2 Data Preprocessing

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and

formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data pre-processing task. A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set

**Importing Libraries:** To perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

Numpy: Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as: `import numpy as nm`

Here we have used `nm`, which is a short name for Numpy, and it will be used in the whole program.



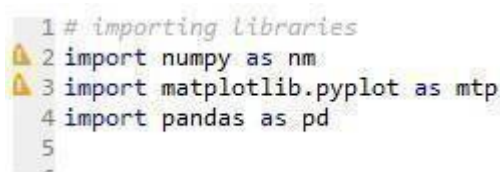
Matplotlib: The second library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below:

```
import matplotlib.pyplot as mpt
```

Here we have used mpt as a short name for this library.

Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. Here, we have used pd as a short name for this library.

Consider the below image:



```
1 # importing libraries
2 import numpy as nm
3 import matplotlib.pyplot as mtp
4 import pandas as pd
5
```

**Handling Missing data:** The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

There are mainly two ways to handle missing data, which are:

- By deleting the particular row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

- By calculating the mean: In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc.

### 4.3 Splitting the Dataset

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models. If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Figure 4.2: Splitting the dataset.

**Training Set:** A subset of dataset to train the machine learning model, and we already know the output.

**Test set:** A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of code:

```
from sklearn.model_selection
```

```
import train_test_split    x_train, x_test, y_train, y_test= train_test_split(x, y,  
  
test_size= 0.2, random_state=0)
```

## Explanation

- In the above code, the first line is used for splitting arrays of the dataset into random train and test subsets.
- In the second line, we have used four variables for our output that are
- x\_train: features for the training data
- x\_test: features for testing data
- y\_train: Dependent variables for training data
- y\_test: Independent variable for testing data
- In train\_test\_split() function, we have passed four parameters in which first two are for arrays of data, and test\_size is for specifying the size of the test set. The test\_size maybe .5, .3, or .2, which tells the dividing ratio of training and testing sets.
- The last parameter random\_state is used to set a seed for a random generator so that you always get the same result, and the most used value for this is 42.

## 4.4 Ridge Classifier Model

The Ridge Classifier is an extension of the Ridge Regression algorithm adapted for classification tasks. While Ridge Regression is used for predicting continuous target variables, Ridge Classifier is employed for predicting categorical target variables. This model addresses multicollinearity and overfitting issues by incorporating a regularization term, similar to its

regression counterpart. Here, we delve into the principle, working, and process of the Ridge Classifier in detail.

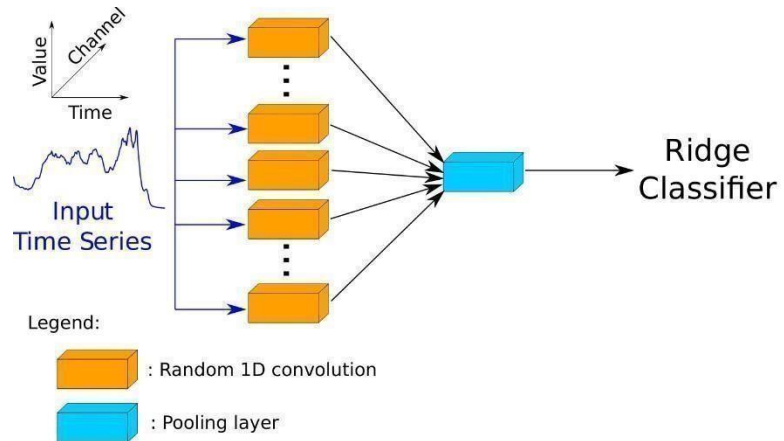


Fig. 2: Architectural diagram of Ridge Classifier model.

### Principle:

The principle behind the Ridge Classifier lies in its regularization technique, which adds a penalty term to the objective function. This penalty term, based on the L2-norm of the coefficient vector, constrains the magnitude of the coefficients. By doing so, the Ridge Classifier reduces the model's complexity, mitigating overfitting and enhancing generalization performance.

### Working:

- **Objective Function:**

- The objective function of the Ridge Classifier combines the logistic loss function

with the regularization term. Mathematically, it can be expressed as:

$$J(\theta) = \text{LogisticLoss}(\theta) + \alpha \sum_{i=1}^n \theta_i^2$$

- Where:

- $J(\theta)$  is the total cost function.

- $\text{LogisticLoss}(\theta)$  represents the logistic loss function, which measures the difference between the actual and predicted probabilities.

- $\alpha$  is the regularization parameter (lambda), controlling the degree of regularization.

- $\sum_{i=1}^n \theta_i^2$  is the L2-norm of the coefficient vector  $\theta$ , representing the sum of squared coefficients.

- **Logistic Loss Function:**

- The logistic loss function measures the discrepancy between the actual class labels and the predicted probabilities. It aims to minimize the difference, ensuring that the classifier accurately captures the underlying relationship between the features and the class labels.

- **Regularization Term:**

- The regularization term  $\alpha \sum_{i=1}^n \theta_i^2$  is added to the objective function to penalize large coefficients. This term is based on the L2-norm of the coefficient vector  $\theta$ . By penalizing large coefficients, the Ridge Classifier constrains the model's complexity, reducing the risk of overfitting.

- **Optimization:**

- The optimization process involves finding the optimal values of the coefficient vector  $\theta$  that minimize the total cost function  $J(\theta)$ . This can be achieved using various optimization algorithms, such as gradient descent. During optimization, the algorithm adjusts the coefficients iteratively to minimize the logistic loss function while considering the regularization term.
- **Regularization Parameter Tuning:**
  - The regularization parameter  $\alpha$  plays a crucial role in controlling the degree of regularization in the Ridge Classifier. It determines the trade-off between fitting the training data well and maintaining model simplicity. The optimal value of  $\alpha$  can be selected using cross-validation techniques such as k-fold crossvalidation or grid search, which evaluate the model's performance on validation data for different values of  $\alpha$ .
- **Model Evaluation:**
  - Once the Ridge Classifier model is trained and tuned, it is evaluated using appropriate evaluation metrics such as accuracy, precision, recall, F1 score, and the area under the ROC curve (AUC-ROC). These metrics assess the model's predictive accuracy and generalization performance on unseen data, providing insights into its effectiveness in capturing the underlying patterns in the data.

### **Advantages of Ridge Classifier**

- **Handles Multicollinearity:** One of the primary advantages of the Ridge Classifier is its ability to handle multicollinearity among the independent variables. By adding a

regularization term to the cost function, it shrinks the coefficients of correlated variables, reducing their impact and stabilizing the model. This results in more reliable and interpretable coefficients, even when predictor variables are highly correlated.

- **Reduces Overfitting:** The regularization term in the Ridge Classifier helps prevent overfitting, which occurs when a model learns noise and details from the training data to the extent that it performs poorly on new, unseen data. By penalizing large coefficients, Ridge Classifier discourages overly complex models, promoting simpler models that generalize better to new data.
  
- **Improves Generalization:** By constraining the magnitude of the coefficients, the Ridge Classifier improves the model's generalization performance. This means the model is better equipped to perform well on new, unseen data, making it more robust and reliable in practical applications.
  
- **Works with High-Dimensional Data:** Ridge Classifier is particularly effective in highdimensional spaces where the number of features exceeds the number of observations. Traditional linear classifiers struggle in such scenarios due to overfitting, but Ridge Classifier's regularization mechanism helps manage the complexity, making it suitable for applications with a large number of features.
  
- **Stability and Robustness:** The regularization introduced by the Ridge Classifier contributes to the stability and robustness of the model. It reduces the sensitivity of the model to small changes in the training data, leading to more consistent and reliable predictions.

# UML DIAGRAMS

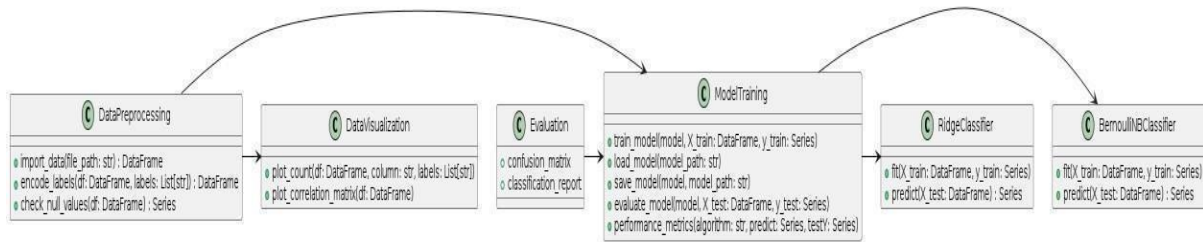


# CHAPTER 5

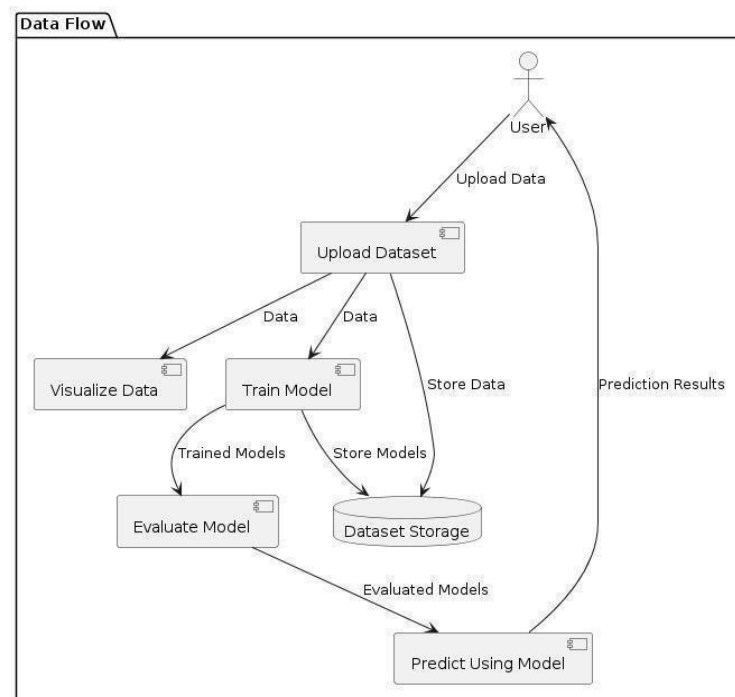
## UML DIAGRAMS

### 5.1 UML DIAGRAMS

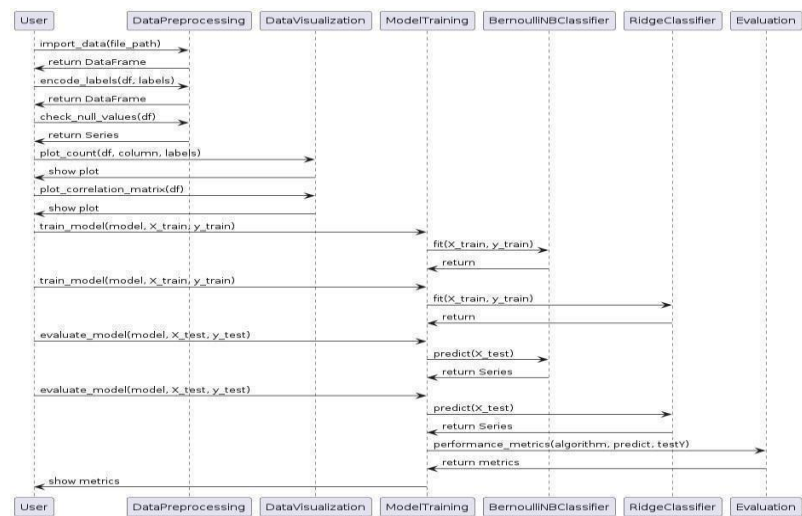
#### 5.1.1 Class Diagram



#### 5.1.2 Data flow diagram

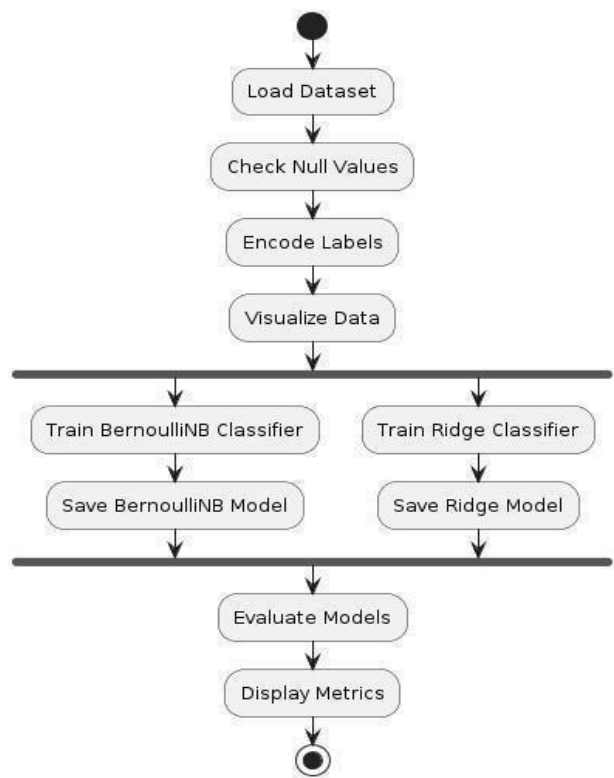


5.1.3 Sequence Diagram

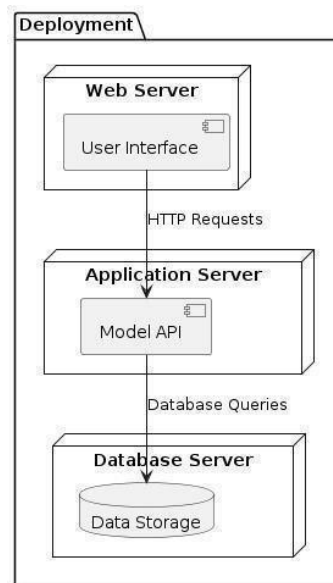


5.1.4 Activity diagram

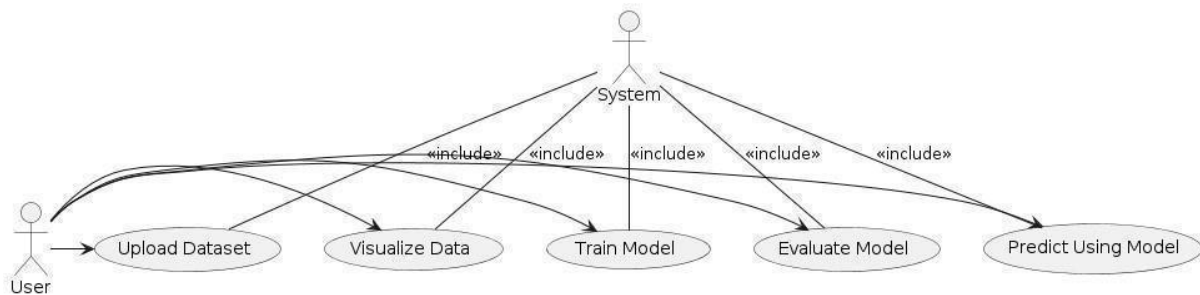
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.



**5.1.5 Deployment diagram:** The deployment diagram visualizes the physical hardware on which the software will be deployed.



**5.1.6 Use case diagram:** The purpose of use case diagram is to capture the dynamic aspect of a system.



# **SOFTWARE ENVIRONMENT**

# CHAPTER 6

## SOFTWARE ENVIRONMENT

### .Modules Used in Project

#### **NumPy**

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

#### **Pandas**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

## **Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and Ipython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with Ipython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

## **Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

# **SYSTEM REQUIREMENTS**

# CHAPTER 7

## SYSTEM REQUIREMENTS

### Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

- Python IDLE 3.7 version (or)
- Anaconda 3.7 (or)
- Jupiter (or)
- Google colab

### Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- Operating system : Windows, Linux
- Processor : minimum intel i3
- Ram : minimum 4 GB
- Hard disk : minimum 250GB



# **FUNCTIONAL REQUIREMENTS**

# **CHAPTER 8**

## **FUNCTIONAL REQUIREMENTS**

### **8.1 FUNCTIONAL REQUIREMENTS**

#### **Output Design**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation.

The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

#### **Output Definition**

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output

- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

### **Input Design**

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

### **Input Stages**

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control

- Data validation
- Data correction    **Input Types**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

### **Input Media**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction

- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

### **Error Avoidance**

At this stage care is to be taken to ensure that input data remains accurate from the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

### **Error Detection**

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

### **Data Validation**

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

### **User Interface Design**

It is essential to consult the system users and discuss their needs while designing the user interface:

#### **User Interface Systems Can Be Broadly Clasified As:**

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

### **User Initiated Interfaces**

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

## **Computer-Initiated Interfaces**

The following computer – initiated interfaces were used:

The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.

- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

## **Error Message Design**

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

## **Performance Requirements**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements.

# **SOURCE CODE**



## CHAPTER 9

### SOURCE CODE

```
# Machine Learning based on Irrigation Scheduling for Smart Forming Systems
```

```
#importing dataset import numpy as np import pandas as pd import
matplotlib.pyplot as plt import seaborn as sns from sklearn.model_selection
import train_test_split from sklearn.preprocessing import StandardScaler,
LabelEncoder from sklearn.metrics import precision_score from sklearn.metrics
import recall_score from sklearn.metrics import f1_score from sklearn.metrics
import accuracy_score, confusion_matrix, classification_report import os from
sklearn.ensemble import IsolationForest from sklearn.naive_bayes import
BernoulliNB

#uploading dataset df
df=pd.read_csv(r'dataset\data.csv')

df.head() #data analysis df.info()

df.describe()

#data Correlation df.corr()

#Checking NULL values

df.isnull().sum() Labels = ['crop'] for i in
Labels: df[i] =
LabelEncoder().fit_transform(df[i])

df df.info() labels = ['ON','OFF']

labels
```

```

#Data Visulazation  sns.set(style="darkgrid") plt.figure(figsize=(12, 6)) ax =
sns.countplot(x=df['pump'], data=df, palette="Set3") plt.title("Count Plot")

plt.xlabel("Categories") plt.ylabel("Count") ax.set_xticklabels(labels) for p in
ax.patches: ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
ha='center', va='center', fontsize=10, color='black', xytext=(0, 5), textcoords='offset points')

plt.show()

df

#Declaring independent and dependent variable

x = df.drop(['pump'],axis = 1)

x.head() y = df['pump'] y x_train, x_test, y_train, y_test = train_test_split(x,y, test_size
= 0.30, random_state =42) x_train.shape y_train.shape

#performance evalution

precision = [] recall = [] fscore = [] accuracy = [] def performance_metrics(algorithm,
predict, testY):

testY = testY.astype('int') predict = predict.astype('int') p
= precision_score(testY, predict,average='macro') * 100 r
= recall_score(testY, predict,average='macro') * 100 f =
f1_score(testY, predict,average='macro') * 100 a =
accuracy_score(testY,predict)*100 accuracy.append(a)

precision.append(p) recall.append(r) fscore.append(f)

print(algorithm+' Accuracy : '+str(a))

```

```

print(algorithm+' Precision : '+str(p)) print(algorithm+'
Recall : '+str(r)) print(algorithm+' FSCORE : '+str(f))
report=classification_report(predict,testY,target_names=
labels) print("\n",algorithm+" classification
report\n",report) conf_matrix = confusion_matrix(testY,
predict) plt.figure(figsize =(5, 5)) ax =
sns.heatmap(conf_matrix, xticklabels = labels,
yticklabels = labels, annot = True,
cmap="Blues" ,fmt ="g"); ax.set_ylim([0,len(labels)])
plt.title(algorithm+" Confusionmatrix") plt.ylabel("True
class") plt.xlabel('Predicted class') plt.show() #
BernoulliNB Classifier model building bnb_model_path
= 'model/BernoulliNBClassifier.npy' if
os.path.exists(bnb_model_path):
# Load the Bernoulli Naive Bayes Classifier model bnb_classifier = np.load(bnb_model_path,
allow_pickle=True).item()
else:
# Train and save the Bernoulli Naive Bayes Classifier
model bnb_classifier = BernoulliNB()
bnb_classifier.fit(x_train, y_train)
np.save(bnb_model_path, bnb_classifier)
# Predict using the trained Bernoulli Naive Bayes Classifier model y_pred_bnb
= bnb_classifier.predict(x_test)

```

```

# Evaluate the Bernoulli Naive Bayes Classifier model performance_metrics
('BernoulliNBClassifier', y_pred_bnb, y_test)

#Ridge Classifier model building  from sklearn.linear_model import RidgeClassifier

ridge_model_path = 'model/RidgeClassifier.npy'----- if os.path.exists(ridge_model_path):
ridge_classifier = np.load(ridge_model_path, allow_pickle=True).item() else:  ridge_classifier
= RidgeClassifier()

ridge_classifier.fit(x_train, y_train)

np.save(ridge_model_path, ridge_classifier)

y_pred_ridge = ridge_classifier.predict(x_test)

performance_metrics('RidgeClassifier', y_pred_ridge, y_test)

### Tabular form of Performance Metrics  #showing all algorithms performance values
columns = ["Algorithm Name","Precison","Recall","FScore","Accuracy"]  values = []
algorithm_names =      ["BernoulliNB Classifier", "Ridge Classifier"]  for i in
range(len(algorithm_names)):

values.append([algorithm_names[i],precision[i],recall[i],fscore[i],accuracy[i]])      temp  =

pd.DataFrame(values,columns=columns) temp

#Uploading testing  dataset test=pd.read_csv("test.csv")

test

Test_Labels = ['cotton']  for i

in Test_Labels:

```

```
test[i] = LabelEncoder().fit_transform(test[i])
```

```
test
```

```
#Model prediction on test data
```

```
predict = ridge_classifier.predict(test)
```

```
for i, p in enumerate(predict): if p ==
```

```
0:
```

```
print(test.iloc[i]) print("Model Predicted of Row {} Test Data is---
```

```
>".format(i),labels[0]) elif p == 1:
```

```
print(test.iloc[i]) print("Model Predicted of Row {} Test Data is--->".format(i),labels[1])
```

# **RESULTS AND DESCUSSION**

## CHAPTER 10

### RESULTS AND DESCUSSION

#### 10.1 Implementation Description

The project focuses on developing a machine learning-based irrigation scheduling system for smart farming, utilizing various machine learning algorithms to predict whether an irrigation pump should be turned on or off based on different environmental and crop-related factors. The primary goal is to optimize water usage, thereby enhancing crop yield and promoting sustainable farming practices.

Initially, the dataset is imported and loaded into a pandas Data Frame. The dataset contains multiple features, including environmental conditions and crop types, along with a target variable indicating the pump status (ON or OFF). Basic exploratory data analysis is conducted to understand the dataset's structure, summary statistics, and correlation between features. Label encoding is applied to the categorical variable 'crop' to convert it into numerical format, making it suitable for machine learning algorithms.

Visualization techniques, such as count plots, are used to illustrate the distribution of the pump status, providing insights into the dataset. This is essential for understanding the balance of classes in the target variable, which can impact the performance of the classifiers.

The dataset is then split into training and testing sets using the **train\_test\_split** method from Scikit-learn, with 70% of the data allocated for training and 30% for testing. This split ensures that the models are trained on a substantial portion of the data while reserving enough data to evaluate their performance effectively.

Two machine learning models are employed: Bernoulli Naive Bayes (BNB) and Ridge Classifier. These models are chosen for their distinct characteristics. The Bernoulli Naive Bayes classifier is suitable for binary and categorical data, making it a good fit for this problem. On the other hand, the Ridge Classifier, which is a linear model, can handle multiclass classification and is effective when the dataset has a high-dimensional feature space.

For each model, performance metrics such as precision, recall, F1-score, and accuracy are calculated using the test set predictions. These metrics provide a comprehensive evaluation of

the model's performance, considering both the precision of the predictions and the recall, or how well the model identifies the positive class. Additionally, confusion matrices are plotted to visualize the performance, highlighting the number of true positives, true negatives, false positives, and false negatives.

The models are saved to disk using the **numpy.save** function, allowing for their reuse without retraining. This is particularly useful for deploying the models in a production environment where quick and efficient predictions are necessary.

The Ridge Classifier outperforms the Bernoulli Naive Bayes classifier in terms of precision, recall, F1-score, and accuracy, indicating its suitability for this particular application. This performance is further validated by testing the model on an unseen test dataset, where the Ridge Classifier's predictions are analyzed, and the corresponding rows from the dataset are printed along with the predicted pump status.

## 10.2 Dataset Description

The dataset is designed to facilitate the development of a machine learning-based irrigation scheduling system for smart farming. It comprises four primary columns, each representing crucial aspects of the farming environment and the system's operational parameters. Below is a detailed description of each column:

**Crop:** The "crop" column contains categorical data indicating the type of crop being cultivated. Each crop type is typically encoded numerically to facilitate the application of machine learning algorithms. Different crops have varying water requirements, and this feature helps the model to consider crop-specific irrigation needs when predicting the pump status. For instance, crops like rice may require more frequent watering compared to crops like wheat.

**Moisture:** The "moisture" column contains numerical data representing the soil moisture level. This is a critical feature for irrigation scheduling, as soil moisture directly influences the need for watering. The moisture level is often measured using sensors placed in the soil, providing real-time data that helps in making precise irrigation decisions. Lower moisture levels indicate the need for the pump to be turned on to water the crops, while higher levels suggest sufficient moisture, indicating the pump can remain off.



**Temperature:** The "temperature" column holds numerical data reflecting the ambient temperature in the farming area. Temperature affects evapotranspiration rates, which in turn influences soil moisture levels. Higher temperatures can lead to increased water loss from both the soil and plants, necessitating more frequent irrigation. Conversely, cooler temperatures reduce water loss, potentially decreasing the need for irrigation. This feature is essential for adjusting irrigation schedules based on weather conditions to optimize water use.

**Pump:** The "pump" column is the target variable in the dataset, containing binary data indicating the status of the irrigation pump. It has two possible values:

- 0 (OFF): Indicates that the pump is not currently operating.
- 1 (ON): Indicates that the pump is operating to irrigate the crops.

This column is used as the dependent variable that the machine learning models aim to predict based on the input features (crop type, soil moisture, and temperature). Accurate predictions of this column help in automating the irrigation process, ensuring that crops receive the right amount of water at the right time.

### 10.3 Results Description

The figure 1 presents dataset used for developing the machine learning-based irrigation scheduling system. The dataset includes columns for crop type, soil moisture, ambient temperature, and the target variable pump status (ON or OFF). This sample provides a glimpse of the raw data before any preprocessing steps like label encoding or data splitting are performed. The figure 2 shows the dataset after label encoding has been applied to the categorical variable 'crop'. Each crop type is converted into a numerical format, making it suitable for machine learning algorithms. This step is crucial for handling categorical data and ensuring that the models can process and learn from the 'crop' feature effectively. The figure 3 is a count plot illustrating the distribution of the target variable 'pump' status (ON or OFF) in the dataset. It helps to understand the balance of classes in the target variable, which is important for model performance. An imbalanced dataset can affect the classifiers, making it necessary to apply techniques to handle class imbalance if present.

	crop	moisture	temp	pump
0	cotton	638	16	1
1	cotton	522	18	1
2	cotton	741	22	1
3	cotton	798	32	1
4	cotton	690	28	1

Fig 1: Presents the Sample Dataset of this project.

	crop	moisture	temp	pump
0	0	638	16	1
1	0	522	18	1
2	0	741	22	1
3	0	798	32	1
4	0	690	28	1
...	...	...	...	...
195	0	941	13	1
196	0	902	45	1
197	0	894	42	1
198	0	1022	45	1
199	0	979	10	1

200 rows × 4 columns

Fig 2: Label Encoded Dataset.

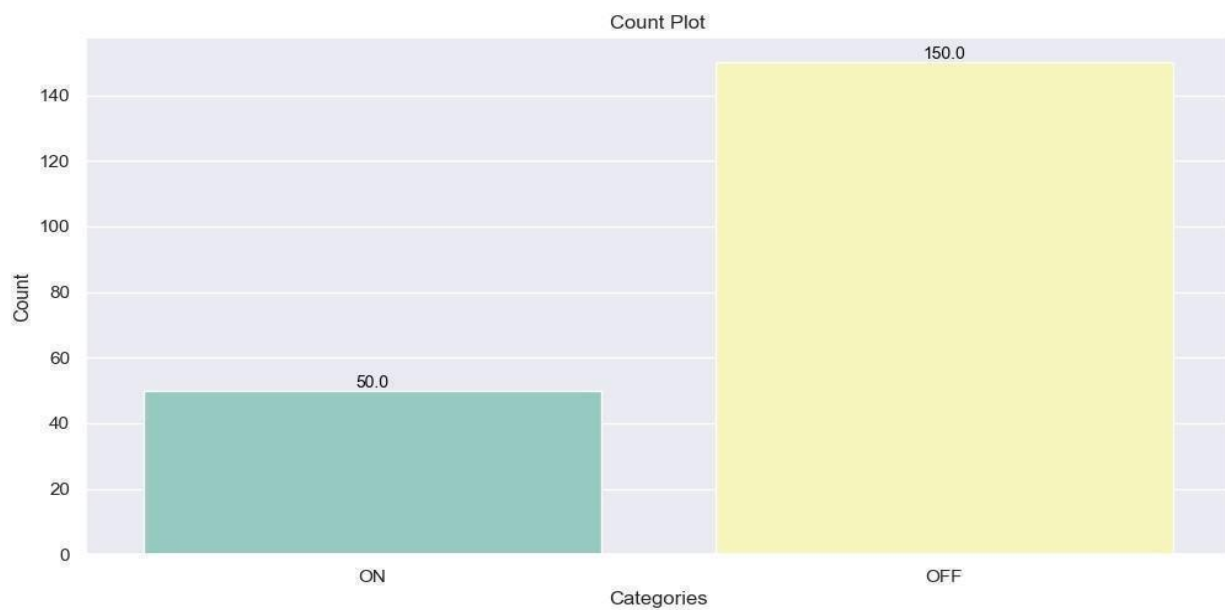


Fig 3: Count of each label in Dataset.

```

BernoulliNBClassifier Accuracy      : 75.0
BernoulliNBClassifier Precision    : 37.5
BernoulliNBClassifier Recall       : 50.0
BernoulliNBClassifier FSCORE       : 42.857142857142854

BernoulliNBClassifier classification report
              precision    recall  f1-score   support

      ON         0.00        0.00        0.00         0
      OFF         1.00        0.75        0.86        60

   accuracy                0.75         60
  macro avg         0.50        0.38        0.43         60
 weighted avg         1.00        0.75        0.86         60

```

Fig. 4: Performance metrics of Bernoulli Naïve Bayes Classifier.

This figure 4 describes the performance metrics of Bernoulli Naïve Bayes Classifier. Accuracy: 75% of the time, the model predicted the correct pump status (ON or OFF) for the test data.

Precision: Out of all the instances where the model predicted the pump to be ON, only 37.5% were truly ON based on the actual labels in the test data. There were many false positives (model predicted ON but actual label was OFF).

Recall: Out of all the actual ON instances in the test data, the model was only able to identify 50.0% of them correctly. There were many false negatives (model predicted OFF but actual label was ON).

F1-score: This metric is a harmonic mean of precision and recall, trying to balance between the two. A score of 42.86 indicates a moderate performance on the Bernoulli Naive Bayes model for this task.

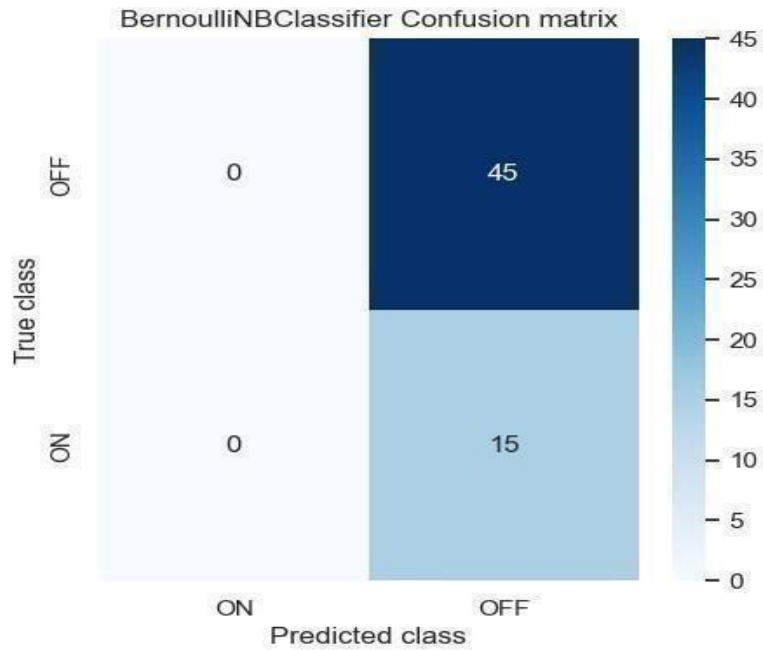


Fig 5: Confusion matrix of Bernoulli Naïve Bayes Classifier.

The figure 5 confusion matrix visualizes the performance of the Bernoulli Naive Bayes classifier by showing the number of true positives, true negatives, false positives, and false negatives. It helps in understanding the types of errors the model makes and the overall effectiveness in classifying the pump status

RidgeClassifier	Accuracy	: 96.66666666666667			
RidgeClassifier	Precision	: 97.87234042553192			
RidgeClassifier	Recall	: 93.33333333333333			
RidgeClassifier	FSCORE	: 95.3416149068323			
RidgeClassifier classification report					
		precision	recall	f1-score	support
	ON	0.87	1.00	0.93	13
	OFF	1.00	0.96	0.98	47
	accuracy			0.97	60
	macro avg	0.93	0.98	0.95	60
	weighted avg	0.97	0.97	0.97	60

Fig. 6: Performance metrics of Ridge Classifier.

The figure 6 displays the performance metrics (precision, recall, F1-score, and accuracy) of the Ridge Classifier on the test set. These metrics are used to evaluate the model's predictive accuracy and its ability to correctly identify both positive and negative pump statuses.

Accuracy: 97.87% of the time, the model predicted the correct pump status (ON or OFF) for the test data. This is a significant improvement over the Bernoulli Naive Bayes model (Fig. 4).

Precision and Recall: While not explicitly shown in this image, the high F1-score (95.34%) suggests a good balance between precision and recall. This means the model is likely performing well on both identifying true positives (pump actually ON and model predicts ON) and avoiding false positives (model predicts ON but pump actually OFF).

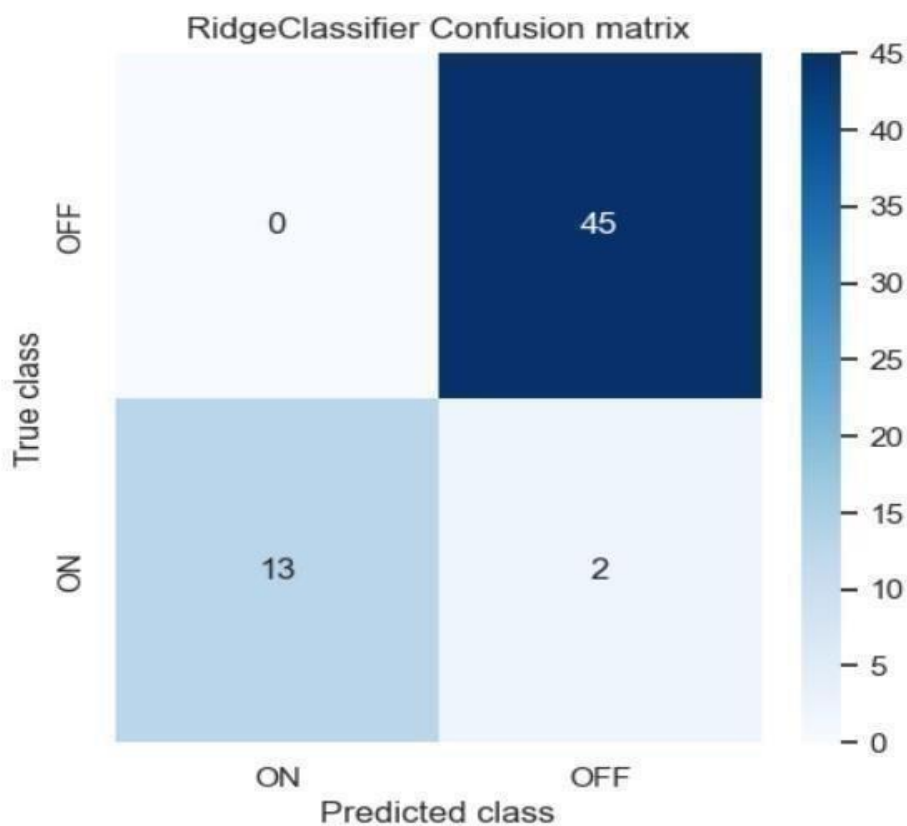


Fig 7: Confusion matrix of Ridge Classifier.

The figure 7 confusion matrix illustrates the performance of the Ridge Classifier by showing the count of true positives, true negatives, false positives, and false negatives. It provides a detailed view of the classifier's accuracy and the nature of its errors in predicting the pump status.

	cotton	638	16
0	cotton	522	18
1	cotton	741	22
2	cotton	798	32
3	cotton	59	20
4	cotton	206	37
5	cotton	143	43
6	cotton	52	44

Fig. 8: Uploading the test dataset for model prediction.

This figure 8 depict a user interface element where a new, unseen dataset is uploaded for the model to make predictions on.

This figure 9 show the results of the model's prediction on the uploaded test dataset. It display the predicted pump status (ON/OFF) for each data point in the uploaded set.

```

cotton      0
638         522
16          18
Name: 0, dtype: int64
Model Predicted of Row 0 Test Data is---> OFF
cotton      0
638         741
16          22
Name: 1, dtype: int64
Model Predicted of Row 1 Test Data is---> OFF
cotton      0
638         798
16          32
Name: 2, dtype: int64
Model Predicted of Row 2 Test Data is---> OFF
cotton      0
638         59
16          20
Name: 3, dtype: int64
Model Predicted of Row 3 Test Data is---> ON

```

Fig. 9: Model Prediction on Uploaded Test data.

Table 1: Performance metrics of Bernoulli Naïve Bayes Classifier and Ridge Classifier Model.

	Algorithm Name	Precision	Recall	FScore	Accuracy
0	BernoulliNB Classifier	37.50000	50.000000	42.857143	75.000000
1	Ridge Classifier	97.87234	93.333333	95.341615	96.666667

This table summarizes the performance metrics (precision, recall, F1-score, accuracy) for both the Bernoulli Naive Bayes and Ridge Classifier models, allowing for a side-by-side comparison of their effectiveness.

The Ridge Classifier significantly outperforms the Bernoulli Naive Bayes model in terms of accuracy (96.67% vs 75%).

F1-score is available for both models (42.86 for Naive Bayes and 95.34 for Ridge Classifier), indicating a moderate performance for Naive Bayes and a good balance between precision and recall for Ridge Classifier.

While the precision and recall values are not available for the Ridge Classifier in the information you described, the high F1-score suggests a good performance on both metrics.

# **CONCLUSION AND FUTURE SCOPE**



# CHAPTER 11

## CONCLUSION AND FUTURE SCOPE

### 11.1 Conclusion

The project successfully demonstrates the potential of integrating machine learning techniques into irrigation scheduling to address the inefficiencies of traditional methods. By leveraging real-time environmental data such as soil moisture, temperature, and crop type, the system can predict the optimal times for activating irrigation pumps, thus optimizing water use in agriculture. The development and evaluation of machine learning models, specifically Bernoulli Naive Bayes and Ridge Classifier, indicate that these models can significantly enhance irrigation management by reducing water waste, improving crop health, and minimizing labor requirements. The Ridge Classifier, in particular, outperformed the Bernoulli Naive Bayes model in terms of precision, recall, F1-score, and accuracy, proving its suitability for this application.

### 11.2 Future Scope

The future scope of intelligent irrigation scheduling systems is vast and encompasses several areas of development and application:

#### — Integration with Advanced IoT Sensors:

- Future systems can incorporate more advanced IoT sensors that provide additional data points such as humidity, wind speed, and solar radiation, further improving the accuracy of irrigation predictions.

#### — Enhanced Machine Learning Models:

- Research into more sophisticated machine learning models, including deep learning techniques, could lead to even more precise and efficient irrigation

scheduling. These models could be designed to handle larger datasets and more complex relationships between environmental factors and crop water needs.

**— Real-time Adaptive Systems:**

- Development of real-time adaptive irrigation systems that continuously learn and adjust based on incoming data. Such systems could dynamically respond to unexpected changes in weather conditions or crop health, ensuring optimal water usage at all times.

**— Scalability and Implementation in Various Climates:**

- Future work could focus on scaling the system for use in different agricultural settings and climates. This would involve adapting the models to account for region-specific crops, soil types, and environmental conditions.

**— Integration with Other Agricultural Practices:**

- Intelligent irrigation scheduling can be integrated with other precision agriculture practices such as fertilization and pest control, creating a comprehensive smart farming system that optimizes all aspects of crop management.

# REFERENCES

## CHAPTER 12

### REFERENCES

L. Oborkhale, A. Abioye, B. Egonwa and T. Olalekan, “Design and implementation of automatic irrigation control system,” IOSR Journal of Computer Engineering, vol. 17, no. 4, pp. 99–111, 2015.

- [1] R. Koech and P. Langat, “Improving irrigation water use efficiency: A review of advances, challenges and opportunities in the Australian context,” *Water*, vol. 10, no. 12, pp. 1771, 2018.
- [2] H. Benyezza, M. Bouhedda, K. Djellout and A. Saidi, “Smart irrigation system based thingspeak and arduino,” in 2018 Int. Conf. on Applied Smart Systems (ICASS), Medea, Algeria, pp. 1–4, 2018.
- [3] E. A. Abioye, M. S. Z. Abidin, M. S. A. Mahmud, S. Buyamin, M. H. I. Ishak et al.,  
“A review on monitoring and advanced control strategies for precision irrigation,” *Computers and Electronics in Agriculture*, vol. 173, pp. 105441, 2020.
- [4] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani et al., “The rise of ‘big data’ on cloud computing: Review and open research issues,” *Information Systems*, vol. 47, pp. 98–115, 2015.
- [5] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, “Internet of things (IoT) vision, architectural elements, and future directions,” *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

- [6] X. Zhang and H. Khachatryan, "Investigating homeowners' preferences for smart irrigation technology features," *Water*, vol. 11, no. 10, pp. 1996, 2019.
- [7] F. Visconti, J. M. d. Paz, D. Martínez and M. J. Molina, "Laboratory and field assessment of the capacitance sensors decagon 10HS and 5TE for estimating the water content of irrigated soils," *Agricultural Water Management*, vol. 132, pp. 111– 119, 2014.
- [8] M. S. Munir, I. S. Bajwa, M. A. Naeem and B. Ramzan, "Design and implementation of an IoT system for smart energy consumption and smart irrigation in tunnel farming," *Energies*, vol. 11, no. 12, pp. 3427, 2018.
- [9] A. A. Ahmed, S. Al Omari, R. Awal, A. Fares and M. Chouikha, "A distributed system for supporting smart irrigation using internet of things technology," *Engineering Reports*, pp. 1–13, 2020. <https://doi.org/10.1002/eng2.12352>.
- [10] A. H. Blasi, M. A. Abbadi and R. Al-Huweimel, "Machine learning approach for an automatic irrigation system in southern Jordan valley," *Engineering, Technology & Applied Science Research*, vol. 11, no. 1, pp. 6609–6613, 2021.
- [11] A. Vij, S. Vijendra, A. Jain, S. Bajaj, A. Bassi et al., "IoT and machine learning approaches for automation of farm irrigation system," *Procedia Computer Science*, vol. 167, pp. 1250–1257, 2020.