PLANT MORPHOLOGY METRICS

AN ARTIFICIAL INTELLIGENCE MINOR PROJECT REPORT SUBMITTED TO

THE NATIONAL INSTITUTE OF ENGINEERING, MYSURU
(An Autonomous Institute under Visvesvaraya Technological University, Belagavi)

in partial fulfillment for the award of degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted By

Roshni V S Gowda 4NI23CI087
Sanmitha H 4NI23CI096
Souparnika R 4NI23CI107

Under the guidance of

Balaji Vijaykumar
Assistant Professor
Department of CS&E
NIE, Mysuru

Department of Computer Science & Engineering
THE NATIONAL INSTITUTE OF ENGINEERING
(Autonomous Institution)
Mysuru - 570 008
2024-25

THE NATIONAL INSTITUTE OF ENGINEERING

(An Autonomous Institute under Visvesvaraya Technological University, Belagavi)

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the Artificial Intelligence with course code BAI402 project work entitled "**Plant Morphology Metrics**" is a Bonafide work carried out by **Roshni V S Gowda 4NI23CI087, Sanmitha H 4NI23CI096, Souparnika R 4NI23CI107** in partial fulfillment for the award of degree of **Bachelor of Engineering in Computer Science and Engineering**, of Visvesvaraya Technological University, Belagavi, during the year **2024-25**. It is certified that all corrections / suggestions indicated during internal assessment have been incorporated and the corrected copy has been deposited in the department library. This project report has been approved in partial fulfillment for the award of the said degree as per academic regulations of The National Institute of Engineering (Autonomous Institution).

Balaji Vijaykumar **Assistant Professor** Dept. of CSE NIE, Mysuru	**Soujanya KV** **Assistant Professor** Dept. of CS&E NIE, Mysuru	**Dr. Anitha R** **Professor and Head** Dept. of CS&E

ABSTRACT

Typically, plant morphology—the study of the physical form and structure of plants—plays a crucial role in understanding plant growth, health, and productivity. Morphological traits such as canopy height, leaf area, root depth, and biomass distribution provide essential information that helps researchers and agronomists evaluate how plants respond to different environmental conditions. Accurate measurement and analysis of these features are fundamental for optimizing crop yield, improving breeding programs, and managing plant health effectively.

This study presents a comprehensive analysis of plant morphology using a detailed dataset comprising attributes such as Canopy Height, Plant Height, Vegetarian Wet %, Vegetarian Dry %, Root Wet %, Root Dry %, Leaf Area, Leaf Count, Root Depth, and Root Length. These metrics collectively offer insights into both above-ground and below-ground plant structures, enabling an in-depth understanding of plant growth patterns and health. The inclusion of moisture-related parameters further allows evaluation of plant hydration and biomass distribution, essential for assessing adaptation to environmental conditions.

To analyse the dataset, four machine learning algorithms were employed: **Logistic Regression, Linear Regression and K-Nearest Neighbours (KNN).** These algorithms were used to classify plant health indicators, predict growth metrics, and uncover underlying patterns within the morphological data. Data visualization techniques, such as boxplots, were utilized to detect outliers and examine variability across features, aiding in the interpretation of plant development stages and variability.

The entire data processing and analysis workflow was implemented using Python's scientific computing ecosystem, including **Pandas** for data manipulation, **NumPy** for numerical operations, and **Matplotlib** for visualization. The machine learning models were developed with **scikit-learn**, providing efficient tools for training, evaluation, and prediction. This integrated approach facilitates informed decisions in plant breeding, crop management, and environmental monitoring, laying the groundwork for future studies linking plant morphology to productivity and stress resilience.

ACKNOWLEDGMENT

I sincerely owe my gratitude to all the persons who helped and guided me in completing this mini project.

I am thankful to **Dr. Rohini Nagapadma**, Principal, NIE College, Mysuru, for all the support she has rendered.

I thank **Dr. Anitha R**, Professor and Head, Department of Computer Science and Engineering, for her constant support and encouragement throughout the tenure for the mini project work.

I would like to sincerely thank our guide **Mr.Balaji Vijaykumar**, Assistant Professor, Department of Computer Science and Engineering, for providing relevant information, valuable guidance and encouragement to complete this minor project.

Roshni V S Gowda 4NI23CI087
Sanmitha H 4NI23CI096
Souparnika R 4NI23CI107

TABLE OF CONTENT

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Problem Statement

Plant morphology, which involves the measurement and analysis of various physical characteristics of plants, plays a critical role in understanding plant health, development, and productivity. With the advancement of data collection methods in greenhouse environments, large-scale morphological data is now available—capturing traits such as canopy height, plant height, leaf area, root structure, and water content.

However, extracting meaningful insights and predictive patterns from this multivariate data remains a challenge. Traditional methods are often inadequate for identifying subtle correlations and trends that could inform decisions in agriculture, breeding, and crop management.

1.2 Objective

The objective of this project is to build a robust, data-driven system capable of analysing plant morphological characteristics to detect patterns, anomalies, and predict outcomes related to plant performance. This includes:

- Preprocessing and visualizing morphological features such as canopy height, plant height, leaf area, and root depth.
- Applying and comparing multiple machine learning models—Logistic Regression, Linear Regression, K-Nearest Neighbours, and Naive Bayes—for predictive analysis.
- Evaluating the effectiveness of each model using appropriate performance metrics.
- Utilizing Python libraries such as Pandas, NumPy, Matplotlib, and Scikit-learn for the entire data analysis and modelling pipeline.

1.3 Scope of the Project

This project focuses on analysing and predicting plant morphological traits using a fixed dataset collected in a greenhouse setting. It applies supervised machine learning algorithms to identify patterns and predict outcomes based on features like plant height, canopy height, and root length. The scope is limited to the use of Python-based tools for data analysis, visualization, and model evaluation.

Chapter 2

SYSTEM REQUIREMENT SPECIFICATION

2.1 Hardware Requirements:

- Processor: Intel i3 or equivalent
- RAM: Minimum 4 GB (8 GB recommended)
- Hard Disk: At least 500 MB free space
- Display: 1024 × 768 resolution or higher
- Input Devices: Standard keyboard and mouse

2.2 Software Requirements:

- Operating System: Windows 10 / Linux / macOS
- Programming Language: Python 3.8 or above

2.2.1 Libraries/Packages:

- Pandas – for data handling and preprocessing
- NumPy – for numerical operations
- Matplotlib – for data visualization
- Scikit-learn – for machine learning models

- IDE: Jupyter Notebook / VS Code / PyCharm
- Optional: Plotly (for interactive visualizations)

Chapter 3

SYSTEM DESIGN

3.1 Use Case Diagram

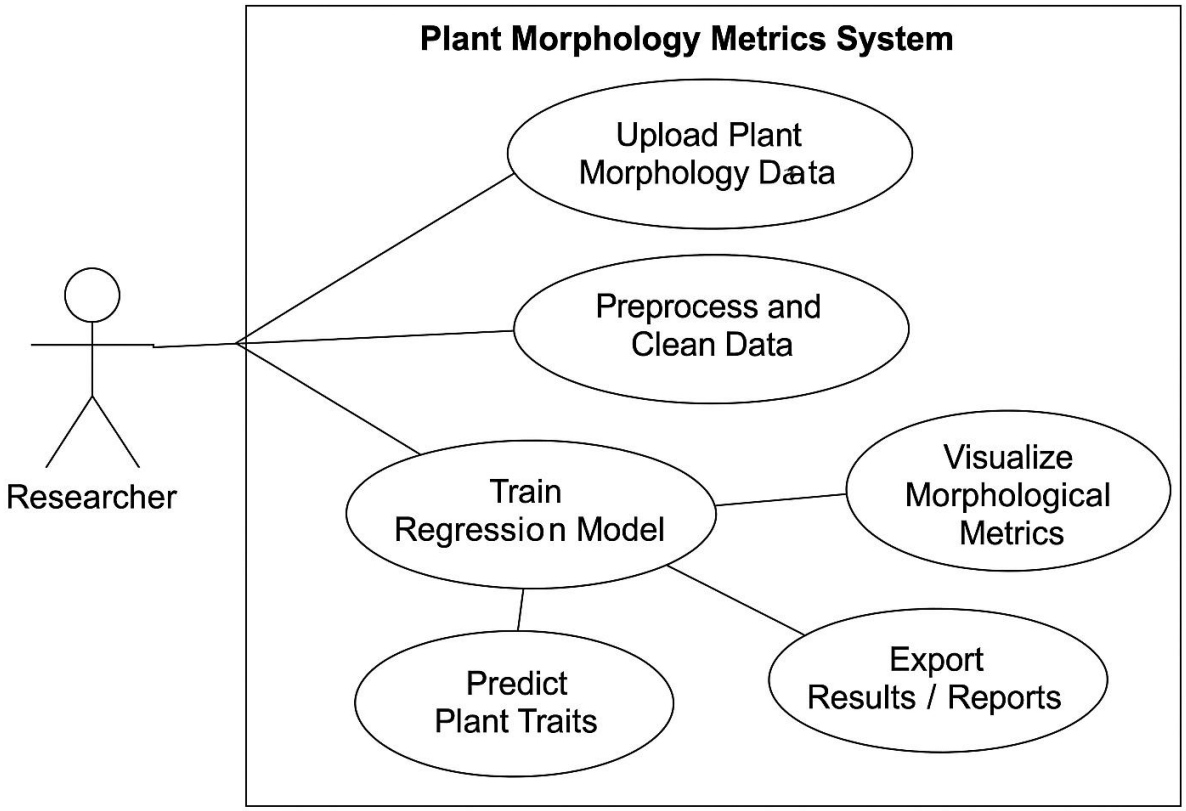


Fig 3.1 Use Case Diagram

The use case diagram for the Plant Morphology Metrics System presents a clear and structured view of how a Researcher interacts with the system to carry out various analytical tasks. At the center, the system includes six core functionalities represented by ovals: Upload Plant Morphology Data, Preprocess and Clean Data, Train Regression Model, Predict Plant Traits, Visualize Morphological Metrics, and Export Results / Reports. Each use case is directly connected to the researcher, showing that they initiate and control each process. The diagram uses a clean, black-and-white UML format and is laid out in a landscape orientation, making it easy to follow and professional in appearance. This visualization helps clarify the system's scope and user interaction in a concise and organized manner.

Chapter 4

METHODOLOGY

4.1 Linear Regression

Linear Regression is a supervised learning algorithm in artificial intelligence used to model the relationship between a dependent variable and one or more independent variables by fitting a straight line to the observed data. It is primarily used for predicting continuous numerical outcomes.

y = mx + c

y = Dependent variable (output)

x = Independent variable (input)

m = Slope of the line (coefficient)

c = Y-intercept (value of y when x=0)

$$y_i = mx_i + c + \varepsilon_i$$

$$\text{Cost Function: } J(m,c) = \sum_{i=1}^{n}(y_i - (mx_i + c))^2$$

Derivative with respect to m:

$$\frac{\partial J}{\partial m} = -2\sum(y_i - mx_i - c)x_i$$

Derivative with respect to c:

$$\frac{\partial J}{\partial c} = -2\sum(y_i - mx_i - c)$$

$$\sum(y_i - mx_i - c)x_i = 0$$

$$\sum(y_i - mx_i - c) = 0$$

Slope:

$$m = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

Intercept:

$$c = \frac{\sum y_i - m\sum x_i}{n}$$

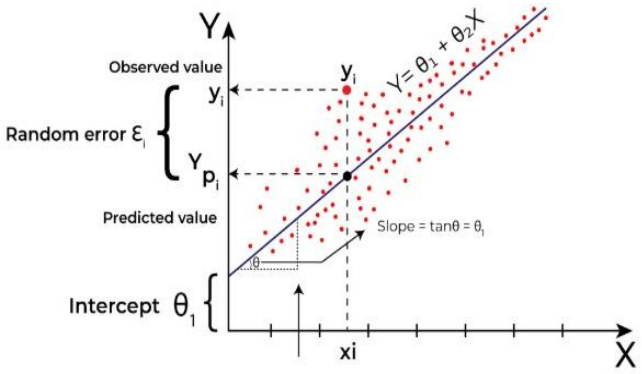


Fig 4.1 Linear Regression Graph

Linear regression models the relationship between a dependent variable and one or more independent variables by fitting a line **y = mx + c**. For example, predicting **plant height** from **canopy height** involves plotting data points and finding the best-fit line. Using **Pandas**, data is cleaned and visualized with **Matplotlib**. The **scikit-learn** Linear Regression model is then trained to minimize mean squared error. Once trained, it predicts outcomes for new inputs. This simple yet effective method is commonly applied in fields like agriculture, economics, and healthcare.

4.2 Logistics Regression

Logistic regression is widely used in artificial intelligence and data science for predicting outcomes based on input features. For example, it can be used in medical diagnosis to classify whether a plant is healthy or not based on features like leaf area, canopy height, or root depth. The output of logistic regression lies between 0 and 1, and a threshold (commonly 0.5) is applied to decide the final class.

It is popular because it's easy to interpret, fast to train, and often performs well for linearly separable classification problems.

Logistic Regression is a supervised machine learning algorithm used for binary or multi-class classification tasks. Unlike linear regression, which predicts continuous values, logistic regression predicts probabilities and classifies data into categories (e.g., 0 or 1, yes or no, healthy or unhealthy).

$P(y=1|x) = \frac{1}{1+e^{-(mx+c)}}$

P(y=1|x) is the probability that the output is 1 (positive class)

m = weight or slope

c = bias or intercept

e = Euler's number (≈ 2.718)

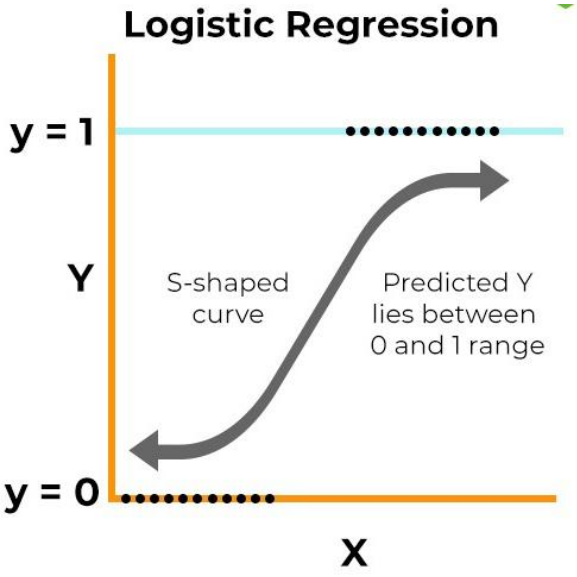


Fig 4.2 Logistic Regression Graph

$$P = \text{Probability of success} = P(y = 1 \mid x)$$

$$1 - P = \text{Probability of failure} = P(y = 0 \mid x)$$

$$\text{Odds} = \frac{P}{1 - P}$$

$$\frac{P}{1 - P} = e^{mx+c}$$

$$P = e^{mx+c} - Pe^{mx+c}$$

$$\boxed{P = \frac{1}{1 + e^{-(mx+c)}}}$$

Logistic regression is a statistical method used for binary classification problems. It models the probability that a given input belongs to a particular class by transforming a linear combination of input features using the sigmoid function, which maps any real-valued number into the range between 0 and 1. This probability represents the likelihood of the input belonging to the positive class. The model then applies a threshold (commonly 0.5) to decide the final class label. Logistic regression estimates the model parameters by maximizing the likelihood of the observed data, making it well-suited for problems where the outcome is categorical, such as determining plant health status based on morphological feature

4.3 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a simple, non-parametric, instance-based learning algorithm used for classification and regression. It works by finding the 'k' training samples closest to a given input point based on a distance metric (usually Euclidean), and then predicting the output based on the majority vote (for classification) or average (for regression) of those neighbors. Since KNN makes no explicit assumptions about the data distribution and doesn't build a model during training, it's called a lazy learner. The accuracy of KNN heavily depends on the choice of 'k', the distance metric, and the scaling of data.

4.3.1 Euclidean Distance

Euclidean distance is defined as the straight-line distance between two points in a plane or space. You can think of it like the shortest path you would walk if you were to go directly from one point to another.

$$\text{distance}(x, X_i) = \sqrt{\sum_{j=1}^{d}(x_j - X_{i_j})^2]}$$

4.3.2 Manhattan Distance

This is the total distance you would travel if you could only move along horizontal and vertical lines like a grid or city streets. It's so called "taxicab distance" because a taxi can only drive along the grid-like streets of a city.

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

4.3.3 Minkowski Distance

Minkowski distance is like a family of distances, which includes both Euclidean and Manhattan distances as special cases.

$$d(x, y) = (\sum_{i=1}^{n} (x_i - y_i)^p)^{\frac{1}{p}}$$

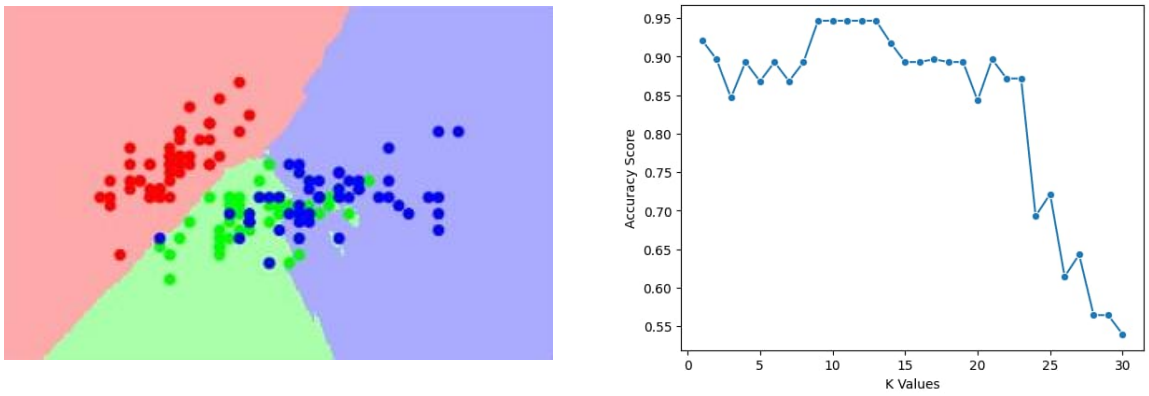


Fig 4.3 Sample KNN Graphs

Chapter 5

IMPLEMENTATION

5.1 Linear Regression

How it works: Linear Regression models the relationship between continuous input features and a continuous target by fitting a line that minimizes prediction errors. It assumes a linear relationship between predictors and the output.

In our project: We used Linear Regression to explore how plant features like canopy height and root length predict continuous growth traits. It helped us understand the dependencies between different morphological measurements.

```
from sklearn.linear_model import LinearRegression

X = df[['canopy height', 'plant height', 'leaf area', 'root length']]
y = df['target']  # continuous target

model = LinearRegression()
model.fit(X, y)
predictions = model.predict(X)
print("Coefficients:", model.coef_)
```

5.2 Logistic Regression

How it works: Logistic Regression estimates the probability that a sample belongs to a specific class by applying a logistic (sigmoid) function to a linear combination of features. It is used for classification tasks.

In our project: We applied Logistic Regression to classify plant samples into categories based on features like leaf count and root depth. This enabled predicting plant health or type from morphology

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

X = df[['canopy height', 'plant height', 'leaf area', 'root length']]
y = df['target']  # categorical target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)
print("Accuracy:", model.score(X_test, y_test))
```

5.3 K-Nearest Neighbors (KNN)

How it works: KNN classifies a sample based on the majority class of its 'k' nearest neighbors in feature space, using distance metrics like Euclidean distance. It's simple and non-parametric. **In our project:** We used KNN to assign plant categories by comparing new samples to the most similar plants based on measurements such as root wet percentage and leaf area.

```
from sklearn.neighbors import KNeighborsClassifier

X = df[['canopy height', 'plant height', 'leaf area', 'root length']]
y = df['target']

model = KNeighborsClassifier(n_neighbors=5)
model.fit(X, y)
predicted = model.predict(X)
print("Sample prediction:", predicted[:5])
```

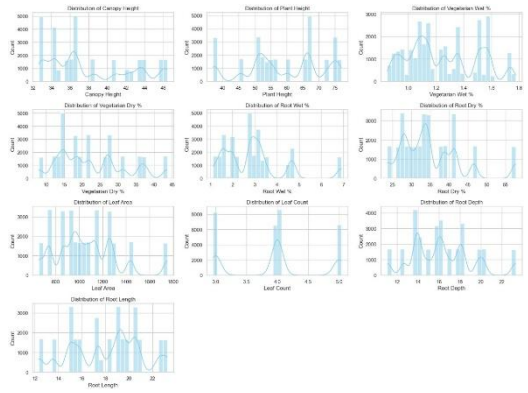
NOTE:

K-Nearest Neighbors (KNN) is a lazy learning algorithm, meaning it doesn't build a model during training but rather stores the dataset and makes decisions at prediction time. This can lead to high memory usage and slow predictions, especially with large datasets. A lesser-known fact is that K=1, while sensitive to noise, often performs surprisingly well in low-noise environments. Moreover, the choice of distance metric (Euclidean, Manhattan, Minkowski, or even cosine similarity) can drastically impact performance, yet many default to Euclidean without testing alternatives. Additionally, KNN offers no model interpretability—it acts as a black box, offering no insights into feature importance or decision logic.

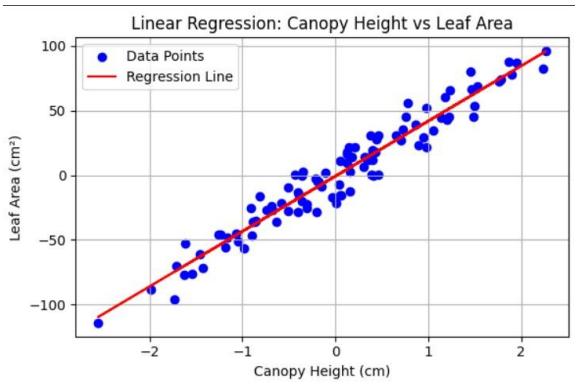
Logistic Regression and Linear Regression are both generalized linear models, but many don't realize that logistic regression is essentially linear regression passed through a sigmoid function to map outputs to probabilities. Logistic Regression can also be extended using regularization (L1 or L2) to perform feature selection—this is how Lasso (L1) can shrink coefficients to exactly zero. Linear regression, on the other hand, assumes homoscedasticity (equal variance in errors), which is often violated in real-world data. Also, both models are surprisingly sensitive to multicollinearity, which can distort coefficient estimates—something often overlooked in practice.

Chapter 6

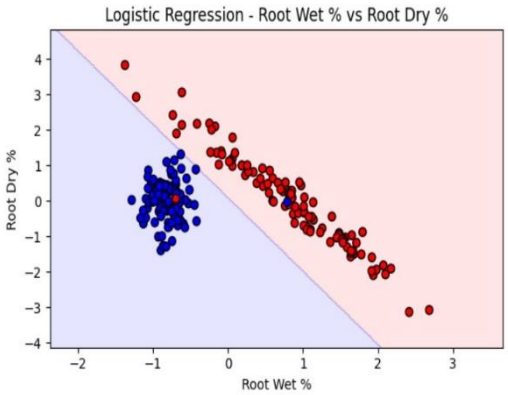
RESULT



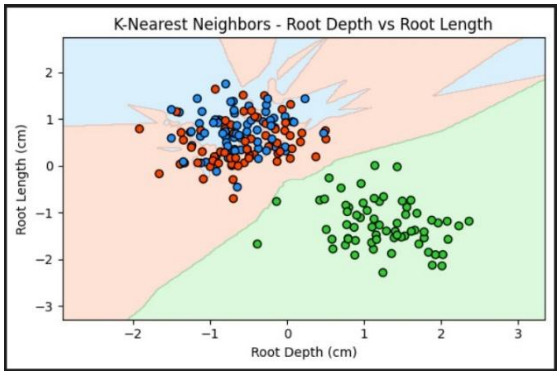
Screenshot 6.1 Plot Histograms



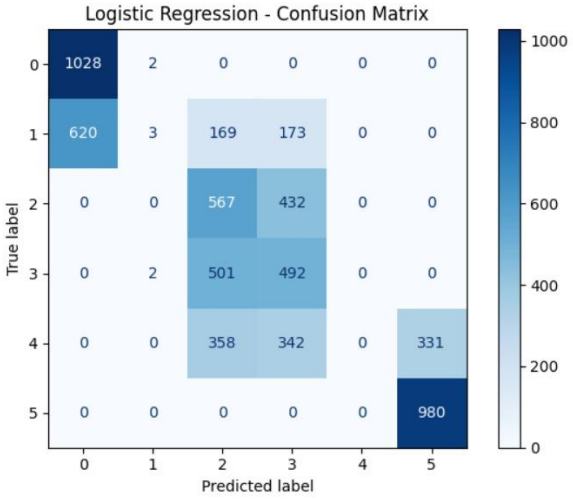
Screenshot 6.2 Linear Regression



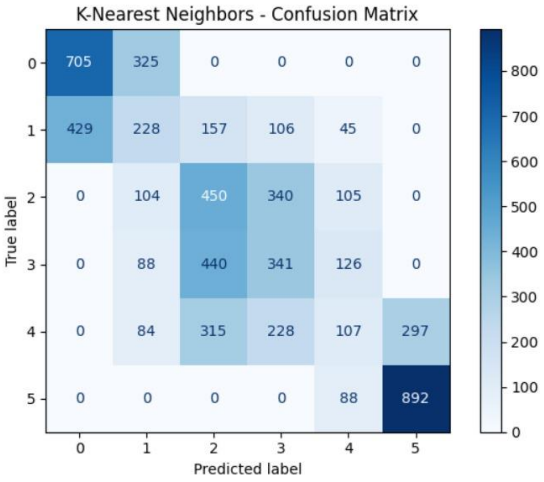
Screenshot 6.3 Logistic Regression



Screenshot 6.4 K-NN



Screenshot 6.5 LR Confusion Matrix



Screenshot 6.6 K-NN Confusion Matrix

Chapter 7

CONCLUSION

In our plant morphology metrics project, we applied four machine learning algorithms—Linear Regression, Logistic Regression, K-Nearest Neighbors, and Gaussian Naive Bayes—to analyze and classify plant samples based on various morphological features. Linear Regression helped uncover relationships between continuous traits, while Logistic Regression, KNN, and Naive Bayes provided effective classification of plants into different categories. These algorithms complemented each other by offering insights into both prediction and classification tasks, improving our understanding of plant growth and health patterns.

The project relied on essential Python libraries such as **NumPy** and **Pandas** for efficient data handling and preprocessing, **Matplotlib** for visualizing key metrics and model performance, and **scikit-learn** for implementing machine learning algorithms and evaluation metrics. Together, these tools and models built a comprehensive framework that supports accurate, data-driven decision-making in agricultural research, helping optimize plant monitoring and management strategies.

Chapter 8

REFERENCES

1. Kaggle. (n.d.). *Greenhouse Plant Morphology Dataset*. Retrieved from https://www.kaggle.com/
2. GitHub. (n.d.). *Machine Learning Code Repositories*. Retrieved from https://github.com/
3. OpenAI. (2023). *ChatGPT (GPT-4) [Large language model]*. Retrieved from https://chat.openai.com/
4. Anthropic. (2023). *Claude AI [Large language model]*. Retrieved from https://www.anthropic.com/
5. Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach* (4th ed.). Pearson.
6. Towards Data Science. (n.d.). *A resource for AI and machine learning*. Retrieved from https://towardsdatascience.com/
7. NumPy Developers. (2023). NumPy: Fundamental Package for Scientific Computing in Python. Retrieved from https://numpy.org/
8. Scikit-learn Developers. (2023). Scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org/
9. Google AI. (n.d.). Google AI Blog and Research. Retrieved from https://ai.googleblog.com/