# Indian Institute of Technology, Dharwad


CS209 : Artificial Intelligence
And
CS214 : Artificial Intelligence Laboratory
Project Report : **Team-3**

**Course Instructor:**
Dr. Dileep A.D.
**Mentor Name:**
Anupama Bidargaddi
**Submitted by:**
1. Saipushkar Nagaraj
2. Utkarsh Raj
3. Sunny Raj
4. Nakirekanti Viraj

**Abstract**

This project focuses on an unsupervised approach to optimizing key agronomic factors that influence soybean growth to achieve maximal yield, discussing the methodology, implementation and results. Rather than relying on labeled data, we apply clustering techniques to categorize soyabean growth patterns into distinct levels based on a variety of phenotypic and genotypic variables.
The insights obtained from the model can be utilized for efficient and effective soyabean cultivation.

# Contents

# Chapter 1
# Introduction

## 1.1 Overview

Artificial Intelligence (CS209) Course Project by Team-3, a hybrid approach combining supervised and unsupervised learning models to extract from an **Advanced Soybean Agriculture** dataset to optimize and detect optimal growth conditions.

**Link to Repository :** Repository

## 1.2 Introduction

Soybean [1] plays a crucial role in global agriculture, offering high-protein value and serving as a key crop in both food and industrial sectors. Enhancing soybean yield is of significant interest, but direct yield prediction is often hindered by the variability of environmental conditions and limited availability of labeled data. In such scenarios, unsupervised learning provides a powerful alternative to unravel hidden patterns within the data.
In this study, we employ various unsupervised learning techniques for clustering soyabean growth records into different levels based on a variety of environmental, genotypic and phenotypic variables.
These growth levels are further used to analyze the most influential input factors contributing to maximal yield. This approach emphazises on data-driven decision making without the need for explicit yield variables, making it broadly applicable in precision farming systems.

## 1.3 Plan of Action

The plan of action taken to achieve the above described objectives are:

- Employ Clustering Algorithms on the dataset to obtain growth levels dependent on the input parameters of phenotypes and genotypes.
  - Clustering algorithms such as KMeans and KMedoid are employed to cluster the data to obtain distinct growth levels.
  - Clustering is performed on different versions on the same data - input parameters *C, G, and S*, selective output parameters and the entirety of data.
  - Best performing clusters are evaluated by using various metrics like **Inertia Score**, **Silhoutte Score**, **Davies Bouldin Score**, **Calinski Harabasz Score**.
- Employ Classification Algorithms to predict the growth level of a given set of phenotypes and genotypes.
  - Classification Algorithms such as **XGBoost**, **SVM**, **RandomForest** are employed to build a classifier that can analyze given set of inputs and predict the yield for said inputs.
  - Best hyper-parameters are determined using **GridSearchCV**. The models so formed are evaluated using metrics like **Accuracy Score Precision Score**, **Recall Score**, **F1-Score** and **Cross-Validation Score**.
- Analyze the data from the above models and extract meaningful insights regarding the optimization of parameters to maximise the yield.
  - Analysis Methods like **SHAP score** are used to establish distinct growth levels and obtain optimized values of parameters for maximal yield

## 1.4 Dataset Description

This dataset consists of 55,450 rows and 13 columns, capturing essential agricultural parameters related to soybean plants. The dataset includes the following key attributes:

- **Plant Height** – Measures the growth of the soybean plant.
- **Number of Pods** – The count of soybean pods per plant.
- **Biological Weight** – The total biomass of the plant.
- **Chlorophyll Content** – Indicates the plant's photosynthetic efficiency.
- **Protein Percentage** – The percentage of protein in the soybean seeds.
- **Seed Yield** – The total soybean seed production.
- **Relative Water Content** – The water retention capacity of leaves.

A key feature of this dataset is the **Parameters** column, which encodes experimental conditions affecting soybean growth. The parameters include:

- **G (Genotype)**: Six different soybean genotypes.
- **C (Salicylic Acid)**: Three levels – 250 mg, 450 mg, and a control level.
- **S (Water Stress)**: Two levels –
  - Water stress at 5% of field capacity.
  - Water stress at 70% of field capacity.

# Chapter 2
# Methodology

## 2.1 Data Cleaning & Preprocessing

### 2.1.1 Handling Null values:

The dataset does not contain null values in any of the fields and thus no preprocessing with regards to null values was performed

### 2.1.2 Handling Duplicate Values:

Among the **55,450** rows, it was reported that **55,342** records were duplicates. A fresh modified dataset was thus made upon removing the duplicate records. (3.1.5a)

### 2.1.3 Eliminating redundant fields:

Redundant fields such as ***Parameters*** (Upon which feature engineering is performed) and ***Random*** were eliminated to reduce redundancy.

### 2.1.4 Standardization of fields:

All Attributes were standardised to eliminate any form of bias when employing clustering or classification algorithms on this data. (3.1.2a)

### 2.1.5 Handling Outliers:

As can be seen from 3.1.1b, outliers exist in the data, but due to the following reasons, no preprocessing techniques were applied to tackle them:

- Owing to the presence of very less data points (3.1.5a), removing these outliers was an expensive choice as the number of data points left behind would be far too less to perform analysis.
- Owing to the presence of very less data points (3.1.5a), imputing these outliers with standard tendencies or applying regression techniques would incurr loss of relationship of these variables with other parameters in the dataset.

## 2.2 Statistical Inferences

- A correlation heatmap was generated to visualize the strength of linear relationships between numeric variables. 3.1.1a
- Boxplots were utilized to detect the presence of outliers. 3.1.1b
- A pairplot was created to analyze pairwise relationships between all numerical variables. The diagonal elements show distributions, while the off-diagonal scatter plots reveal trends, clusters, or potential outliers.3.1.2a
- The distribution of each numerical feature was plotted to understand its underlying shape. Most features were found to follow (normal/skewed) distributions as the plot was made on the Normalized dataset, guiding further preprocessing steps.3.1.2b
- Data visualisation 3.1.3a 3.1.3b

## 2.3 Feature Engineering

### 2.3.1 Ordinal-Encoding of the Parameters Attribute:

The ***Parameters*** Attribute that had the parameters - **C**,**G** and **S** were extracted and encoded into seperate fields, namely **C**,**G** and **S**.

### 2.3.2 One-Hot-Encoding of the Parameters Attribute:

Once Ordinal Encoding on the ***Parameters*** attribute is performed, One-Hot-Encoding is performed to obtain seperate fields for each class a given *Parameter*

### 2.3.3 Mutual Information Regression

To extract meaningful insights certain parameters were chosen to be **optimized** in-order for a higher yield output. They are **SYUA - Seed Yield per Unit Area**,**Number of Seeds per Pod**,**Sugars (SU)**,**Protein Content (PCO)** Mutual Information Regression plots were made to determine the effect and impact of other parameters on these ***target*** variables. 3.1.4

## 2.4 Clustering

### 2.4.1 Algorithms used

Clustering was performed in different phases by employing **KMeans** [2] and **KMedoid** [3] Clustering techniques. Other sophisticated clustering methods such as **DBSCAN**, **Agglomerative Clustering** were avoided due to the sparsity of data leading to poor performance from these algorithms.

## 2.5 Implementation

### 2.5.1 Techniques employed for visualization:

#### TSNE (T-distributed Stochastic Neighbor Embedding)

t-SNE (t-Distributed Stochastic Neighbor Embedding) [4] is a technique to reduce the dimensions of data, mainly for visualization, while keeping similar points close together.

#### PCA (Principle Component Analysis)

A versatile feature engineering tool with many applications, one of which is to reduce the dimensions of data with minimal loss. [5]

### 2.5.2 Phase-1: KMeans Clustering was applied on all parameters

KMeans Clustering techniques on the data set that was reduced to lower dimensions using **PCA** and **TSNE**. The effective number of clusters was identified using **Elbow-method** upon plotting **Inertia** against the number of clusters.3.2.1a

### 2.5.3 Phase-2: KMeans Clustering on All Output Parameters

Kmeans and KMedoid Clustering Techniques were applied on all ***Ouput parameters*** in order to test the hypothesis *"Distinct Growth Levels can be established by clustering all output parameters"*. We have defined **output parameters** as those that are measured after the soyabean crop has been harvested. From the dataset, we observe that these parameters are namely,

*'Plant Height (PH)', 'Number of Pods (NP)', 'Biological Weight (BW)', 'Sugars (Su)', 'Relative Water Content in Leaves (RWCL)', 'ChlorophyllA663', 'Chlorophyllb649', 'Protein Percentage (PPE)', 'Weight of 300 Seeds (W3S)', 'Leaf Area Index (LAI)', 'Seed Yield per Unit Area (SYUA)', 'Number of Seeds per Pod (NSP)', 'Protein Content (PCO)'* .

The effective number of clusters was determined by using **Elbow Method** upon plotting inertia values against number of clusters. 3.2.1b

### 2.5.4 Phase-3: Applying Gaussian Mixture Model Clustering on PCA data

**Gaussian Mixture Model** is a probabilistic model that works under certain assumptions:

- Data is generated from a mixture of several Gaussian Normal Distributions, each with its own mean and variance.
- A data point belongs to each cluster with a certain probability.

The **Gaussian Mixture Model** was trained on PCA-2 data, and the best clustering model was chosen using metrics like **BIC** and **AIC**. (3.2.5a)(3.2.5b)

### 2.5.5 Phase-4: Applying K-Means and K-Medoids Clustering with $k = 2$

Given the small dataset size (only 108 unique samples), applying clustering with more than two clusters resulted in fewer samples per group, making it impractical for reliable training and analysis. Therefore, $k = 2$ was selected as a balanced trade-off to allow downstream supervised learning and explainability.

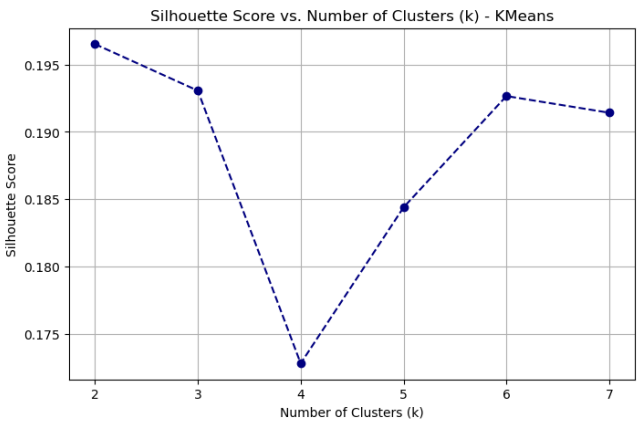
**Why $k = 2$?**

- The dataset contains a limited number of unique samples.
- Higher values of $k$ would result in very small cluster sizes, which are unsuitable for robust model training.
- Visualizations of production metrics like *Seed Yield per Unit Area (SYUA)* and *Protein Percentage (PPE)* showed clear separation when $k = 2$ was used.

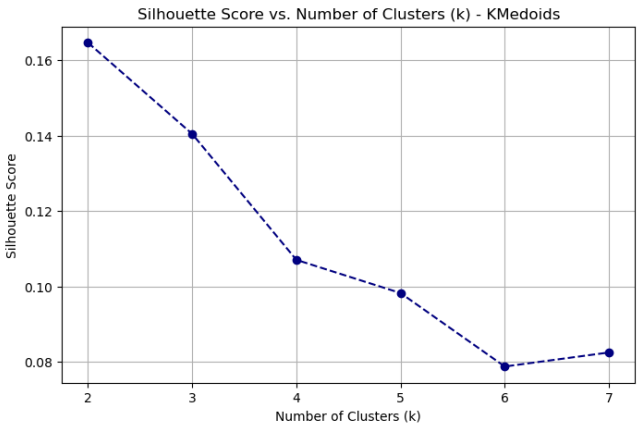
To support the choice of $k = 2$ for both clustering methods (K-Means and K-Medoids), we analyze:

- Silhouette scores for various $k$ values.
- Scatter plots of the clusters in reduced dimensions.

**K-Means ($k = 2$) and k-Mediods ($k = 2$)**




(a) Silhouette Score vs $k$ for K-Means Clustering




(b) Silhouette Score vs $k$ for K-Medoids Clustering

## 2.6 Evaluation

Various different metrics were used in order to evaluate the effectiveness of the clusters like

- Inertia Values using **Elbow Method ??**
- Silouhette Score: [6]
- Davies Bouldin Score: [7]
- Calinski Harabasz Score: [8]
- Bayesian Information Criterion (BIC): [9]
- Akaike Information Criterion (AIC): [10]

### 2.6.1 Phase 1 Results:

Visualizing the metrics corresponding to each of the clusters helps filter out clusters that are not efficient, this way the **KMeans-6 Cluster Model** on all parameters was found to perform the best. (3.2.2)

### 2.6.2 Phase 2 Results:

Visualizing the metrics corresponding to each of the clusters helps filter out clusters that are not efficient, this way the **KMeans-6 Cluster Model** on all output parameters was found to perform the best. (3.2.3)

### 2.6.3 Phase 3 Results:

In the case of the **Gaussian Mixture Model**, Bayesia Information Criterion (BIC) and Akaike Information Criterion (AIC) were employed to identify the most optimal clusters. Lower values of both AIC and BIC indicate a better model.

- In the case where only output parameters were considered **refer:3.2.5a**, optimal number of Clusters turned out to be **5** because of a low BIC and AIC score.
- In the case where all parameters were considered, **refer:3.2.5b**, optimal number of Clusters turned out to be **6** because of a low BIC and AIC score.

### 2.6.4 Why K-Means and K-Medoids with *k=2* Were Chosen

For clustering the soybean samples based on phenotype and genotype features, we selected $k = 2$ for both **K-Means** and **K-Medoids** due to the following reasons:

- **Interpretability:** Boxplots of selected output parameters across clusters showed clear and meaningful separation between the two classes. This provided a biologically interpretable split into "high production" and "low production" categories.
- **Dataset Size:** The dataset consisted of only 108 unique observations. Using more than 2 clusters would lead to fewer samples per cluster, making it impractical for downstream classification and analysis tasks.
- **Robustness:** K-Medoids, being less sensitive to outliers compared to K-Means, was especially useful when dealing with real-world agricultural data where minor noise or measurement error is common.

## 2.7 Classification Algorithms

The following classification algorithms were used to implement classification algorithms to classify the input data into distinct growth levels.

- **Random Forest Classifier**
- **SVM**
- **XGBoost Classifier**
- **Neural Networks**

Upon performing **SHAP** and **Statistical Analysis** of the different clusters that were made **ref:??**, we observed the **2-Cluster Model** that was obtained by applying KMeans and KMedoid Algorithms on select output parameters had the most effective results in terms of obtaining distinct growth levels from the clusters.

## 2.8 Model Training and Testing

### 2.8.1 Preparing Data for classification task

- Owing to the small number of data points, we have split the dataset into **Train** and **Test** sets, with an **80-20** split without an explicit validation set.
- **GridSearchCV** algorithm was used to find the most optimal hyper-parameters to be used in the classification algorithm. **GridSearchCV** performs implicit validation checks to determine optimal parameters hence eliminating the need to have an explicit validation set.

### 2.8.2 Training the Algorithm

- The classification algorithms mentioned above (2.7) were trained on 3 different sets of training data, namely
  - Training data consisting of only the input parameters, i.e, **C**,**G** and **S** parameters.
  - Training data consisting of only the Output Parameters (2.5.3)
  - Training data consisting of both of the above sets of variables.
  - The above steps were repeated for both KMeans(2) Clustered Data as well as KMedoid(2) Clustered Data.

### 2.8.3 Random Forest Algorithm

**Hyperparameters used to train the models and their Roles**

- **n_estimators**: Number of trees in the forest.
- **criterion**: Metric used to evaluate split quality; either `gini` or `entropy`.
- **max_depth**: Maximum depth of each tree to prevent overfitting.
- **min_samples_split**: Minimum number of samples required to split a node.
- **min_samples_leaf**: Minimum number of samples required at a leaf node.

### 2.8.4 Support Vector Machine (SVM)

Support Vector Machines (SVMs) are powerful supervised learning algorithms used for classification tasks. SVM works by finding the hyperplane that best separates data points from different classes with the maximum margin.

#### Hyperparameters Used

- **kernel**: Specifies the kernel type to be used:
  - `linear`: linear hyperplane
  - `poly`: polynomial kernel $K(x, x') = (x \cdot x' + c)^d$
  - `rbf`: Radial Basis Function (Gaussian) $K(x, x') = \exp(-\gamma \|x - x'\|^2)$
  - `sigmoid`: Sigmoid kernel function
- **C**: Regularization parameter that controls the trade-off between achieving a low error on the training data and minimizing the norm of the weights.
- **degree**: Degree of the polynomial kernel function (used only for `poly`).
- **gamma**: Defines the influence of a single training example in `rbf`, `poly`, and `sigmoid`.

#### Implementation Description

In this project, SVM was trained using different kernel functions. A 5-fold Stratified Cross-Validation approach was adopted to evaluate performance metrics. For the polynomial kernel, a hyperparameter tuning process was done using GridSearchCV with:

- degree = 2
- C = 0.1
- gamma = auto

### 2.8.5 XGBoost Classifier

XGBoost (Extreme Gradient Boosting) is an optimized implementation of the gradient boosting algorithm, which is widely used for classification and regression tasks due to its high performance and scalability.

#### Model Configurations

#### Model 1: KMeans (Input Parameters)

```
colsample_bytree=1, gamma=0, learning_rate=0.1, max_depth=3,
n_estimators=150, reg_alpha=0, reg_lambda=1, subsample=1
```

#### Model 2: KMeans (Output Parameters)

```
colsample_bytree=0.9, gamma=0, learning_rate=0.6, max_depth=4,
n_estimators=180, reg_alpha=0.05, reg_lambda=3, subsample=0.9
```

#### Model 3: KMeans (All Parameters)

```
learning_rate=0.2, max_depth=4, n_estimators=190,
eval_metric='error', booster='gbtree', min_child_weight=1,
colsample_bytree=0.8, subsample=0.9, gamma=0, scale_pos_weight=1
```

#### Model 4: KMedoid (Input Parameters)

```
learning_rate=0.05, max_depth=2, n_estimators=150
```

**Model 5: KMedoid (Output Parameters)**

`learning_rate=0.05, max_depth=2, n_estimators=150`

**Model 6: KMedoid (All Parameters - Fully Tuned)**

```
seed=100, n_estimators=100, learning_rate=0.075, gamma=0.0,
colsample_bytree=0.7, reg_alpha=0, reg_lambda=0.005,
max_depth=6, min_child_weight=1, subsample=0.6
```

Each stage involved cross-validation to select the best performing parameter.

### 2.8.6 Neural Networks

#### Introduction

A **Neural Network (NN)** is a supervised machine learning model inspired by the structure of the human brain. It consists of layers of interconnected *neurons* (nodes), where each neuron applies a transformation to the input data. Neural networks are particularly useful for modeling complex, non-linear relationships.

#### Activation Functions Used

In this experiment, the following activation functions were tested:

- `identity`: $\phi(x) = x$
- `logistic` (sigmoid): $\phi(x) = \frac{1}{1+e^{-x}}$
- `tanh`: $\phi(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- `relu`: $\phi(x) = \max(0, x)$

#### Hyperparameters Used

The `MLPClassifier` was configured with the following key hyperparameters:

- `hidden_layer_sizes=(100,)`: One hidden layer with 100 neurons.
- `activation`: One of `'identity'`, `'logistic'`, `'tanh'`, or `'relu'`.
- `max_iter=1000`: Maximum number of iterations for training.
- `random_state=42`: For reproducibility.

#### Cross-Validation and Evaluation

- **Stratified 5-Fold Cross-Validation** was used to evaluate performance while preserving class distribution.
- Metrics collected per fold: Accuracy, Precision, Recall, and F1 Score.

## 2.9 Testing Algorithms & Performance Metrics

### 2.9.1 Random Forest Algorithm (3.3.1)

| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| RF on CGS (KMeans) | 0.6818 | 0.5000 | 0.4286 | 0.4615 |
| RF on Output Params (KMeans) | 0.8636 | 0.8333 | 0.7143 | 0.7692 |
| RF on All Params (KMeans) | 0.8182 | 0.8000 | 0.5714 | 0.6667 |
| RF on CGS (KMedoids) | 0.7273 | 0.8571 | 0.7500 | 0.8000 |
| RF on Output Params (KMedoids) | 0.9091 | 1.0000 | 0.8750 | 0.9333 |
| RF on All Params (KMedoids) | **0.9545** | **0.9610** | **0.9545** | **0.9556** |

Table 2.9.1: Evaluation Metrics for Random Forest Classifier on KMeans and KMedoids

**Observation:** The best overall performance was observed for the Random Forest classifier trained on **all parameters clustered using KMedoids**, achieving a test accuracy of 95.45% and an F1-score of 0.9556.

### 2.9.2 Support Vector Machine (SVM) Algorithm (3.3.2)

| Dataset | Linear | Polynomial | RBF | Sigmoid |
|---|---|---|---|---|
| CGS (KMeans) | 0.6571 | 0.8511 | 0.8147 | 0.5935 |
| Output Params (KMeans) | 0.9255 | 0.9091 | 0.7870 | 0.6667 |
| All Params (KMeans) | **0.9264** | 0.9091 | 0.7870 | 0.6667 |
| CGS (KMedoids) | 0.5835 | 0.6684 | 0.7052 | 0.5541 |
| Output Params (KMedoids) | 0.9082 | 0.8636 | 0.6957 | 0.5377 |
| All Params (KMedoids) | 0.8714 | 0.8182 | 0.6957 | 0.5377 |

Table 2.9.2: Average Accuracy of SVM Classifiers Across Different Kernels and Clustering Strategies.

**Observation:** Based on the average accuracy scores, the best performing model is the **Support Vector Machine with a linear kernel**, applied to the **KMeans-clustered dataset using all parameters**, achieving an average accuracy of **0.9264**. This model outperformed all others across both clustering techniques and kernel types.

.

### 2.9.3 XGBoost Classifier (3.3.5)

| Dataset | Accuracy | Precision | Recall | F1 Score | CV Score |
|---|---|---|---|---|---|
| KMeans - CGS | 0.6364 | 0.4286 | 0.4286 | 0.4286 | 0.7121 |
| KMeans - Output Params | 0.9091 | 1.0000 | 0.7143 | 0.8333 | 0.9264 |
| KMeans - All Params | 0.9545 | 1.0000 | 0.8571 | 0.9231 | 0.9264 |
| KMedoids - CGS | 0.6818 | 0.8462 | 0.6875 | 0.7586 | 0.6481 |
| KMedoids - Output Params | 0.9545 | 1.0000 | 0.9375 | 0.9677 | 0.8519 |
| KMedoids - All Params | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 0.8792 |

Table 2.9.3: XGBoost Performance Metrics Across Datasets and Parameter Sets

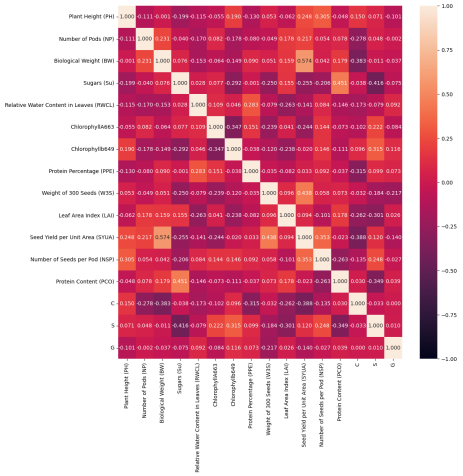
### 2.9.4 Neural Networks (3.3.3)

**Observation:**

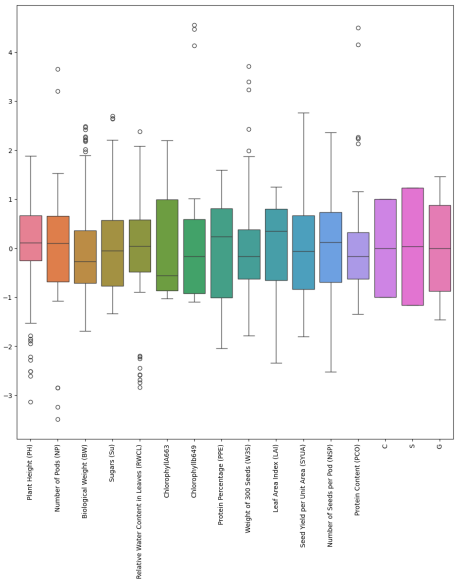
- **Best Performance:** The highest accuracy (0.8424) is achieved by the neural network using the `ReLU` activation function, trained on `input parameters` with `KMeans(2)` clustering.
- **Clustering Impact:**
  - `KMeans(2)` clustering generally results in better model performance compared to `KMedoids(2)`, especially for `ReLU` and `logistic` activations.
  - For example, `NN (logistic)` on all parameters achieves an accuracy of **0.8242** with KMeans, compared to **0.7216** with KMedoids.
- **Activation Function Comparison:**
  - `ReLU` and `logistic` activations consistently outperform `identity` and `tanh` activations.
  - The `identity` activation shows poor performance, particularly for output-only training with KMeans (**0.3333** accuracy).

# Chapter 3
# Results

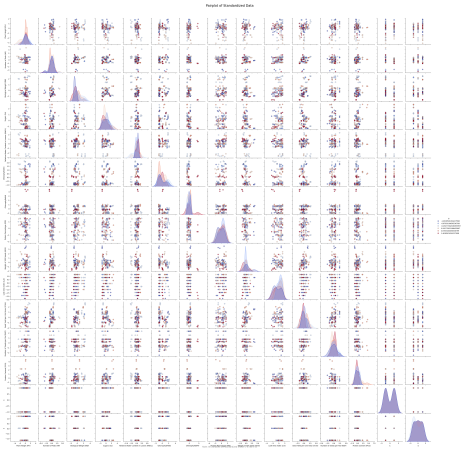
## 3.1 EDA results



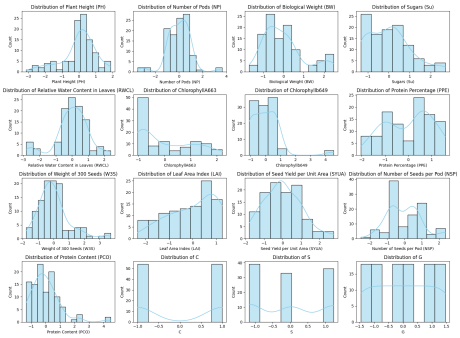
(a) Correlation HeatMap



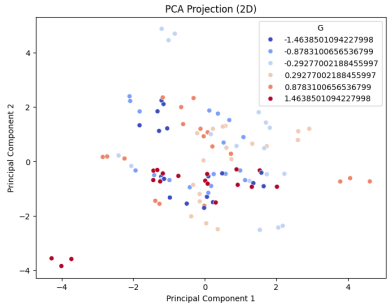
(b) Box Plot of Numerical Attributes



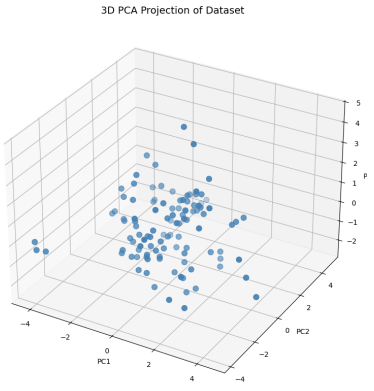
(a) Pair Plot after standardizing



(b) Distribution of all Numerical Features




(a) 2-D projection of the entire dataset




(b) 3D-Projection of the entire dataset

Figure 3.1.3: Projection of the dataset onto lower dimensions for easy visualization

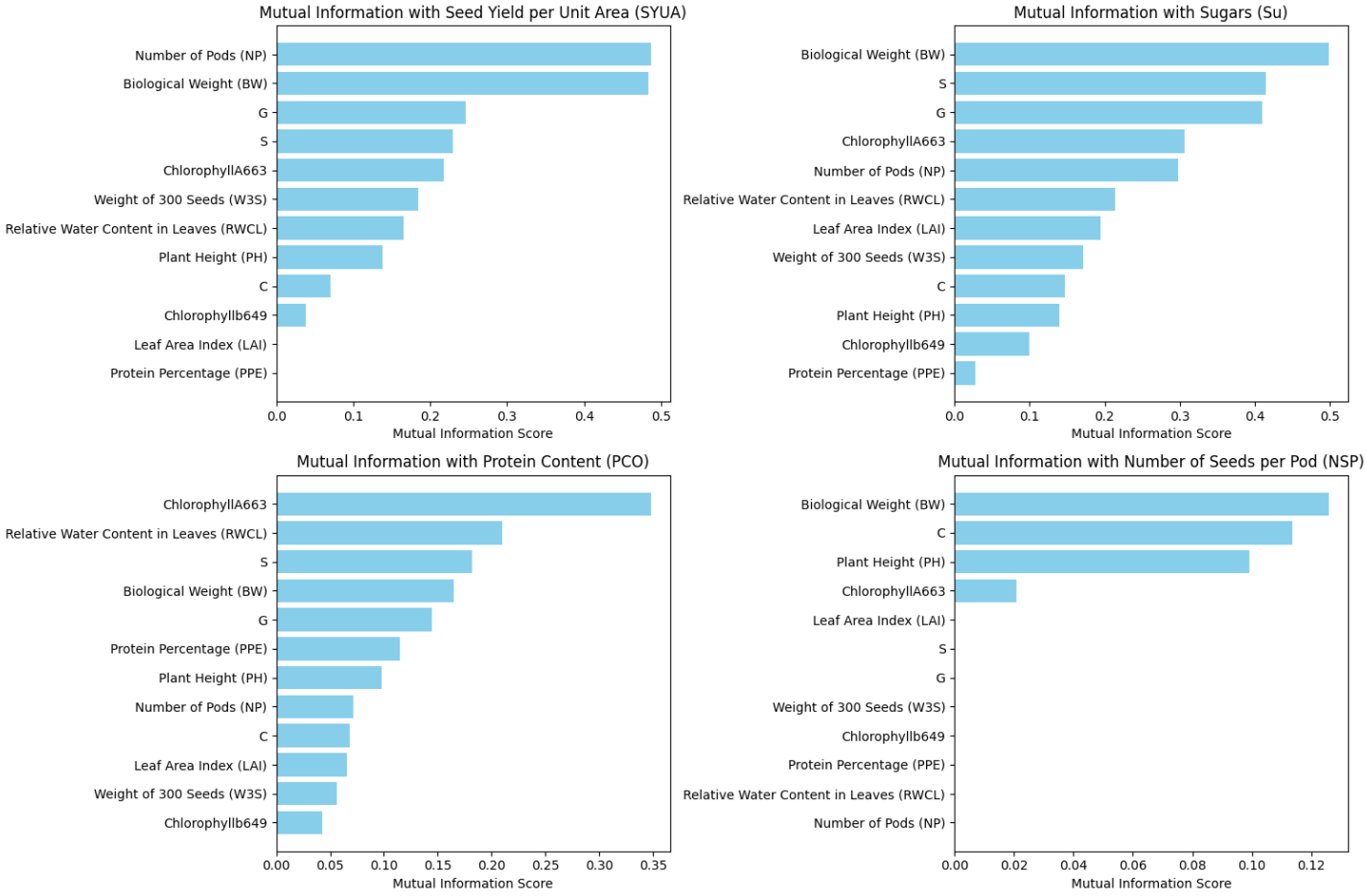



Figure 3.1.4: Mutual Information Regression Plot

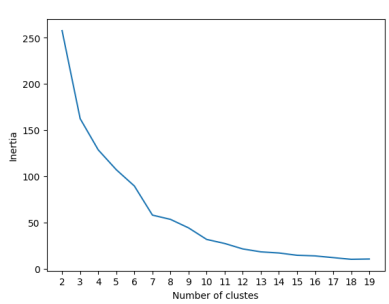
```
data_original.duplicated().sum()
```

```
np.int64(55342)
```

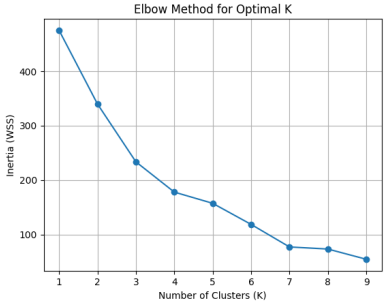
(a) Duplicated Data Values

Figure 3.1.5: Overview of Original Dataset and Its Analysis

## 3.2 Clustering Results

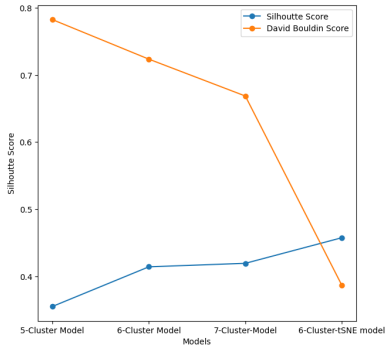



(a) Elbow Curve Method Analysis for optimal Clusters

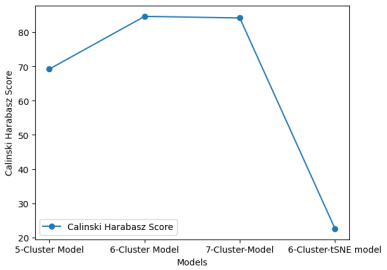



(b) Elbow Curve Method Analysis for optimal Clusters

Figure 3.2.1: PCA-2 elbow Curves

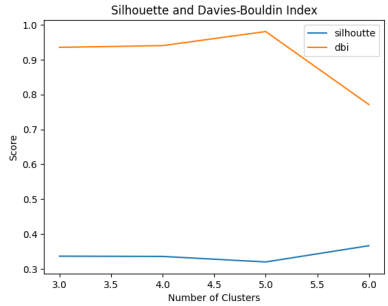



(a) Silouhtette & Bouldin Scores

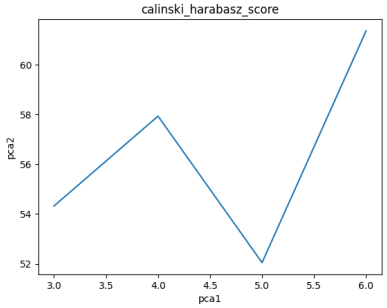



(b) Calinski's Score

Figure 3.2.2: Result evaluation of PCA-2 on all parameters

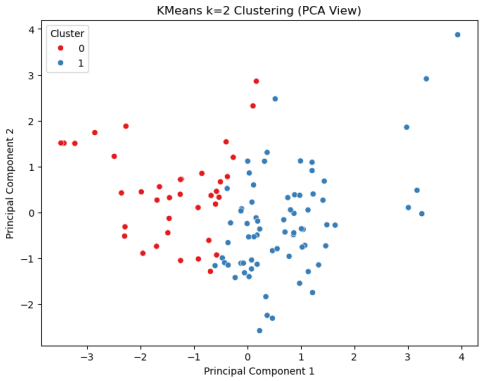



(a) Silouhtette & Bouldin Scores

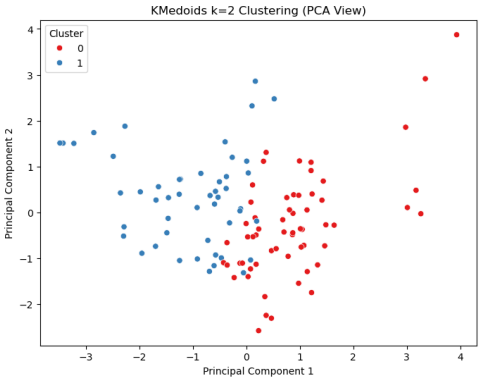



(b) Calinski's Score

Figure 3.2.3: Result evaluation of PCA-2 on output parameters

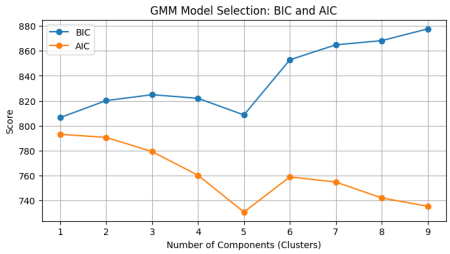



(a) Cluster Scatter Plot for K-Means with $k = 2$

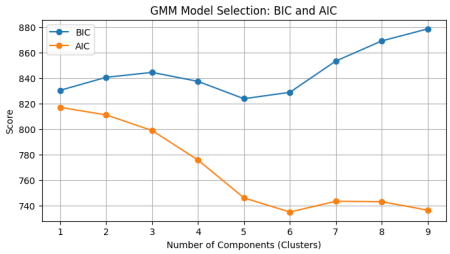



(b) Cluster Scatter Plot for K-Medoids with $k = 2$

Figure 3.2.4: Projection of the dataset onto lower dimensions for easy visualization




(a) Optimal Cluster using GMM on Output Parameters




(b) Optimal Cluster using GMM on All Parameters

Figure 3.2.5: Evaluation of GMM Clustering

## 3.3 Classification Results

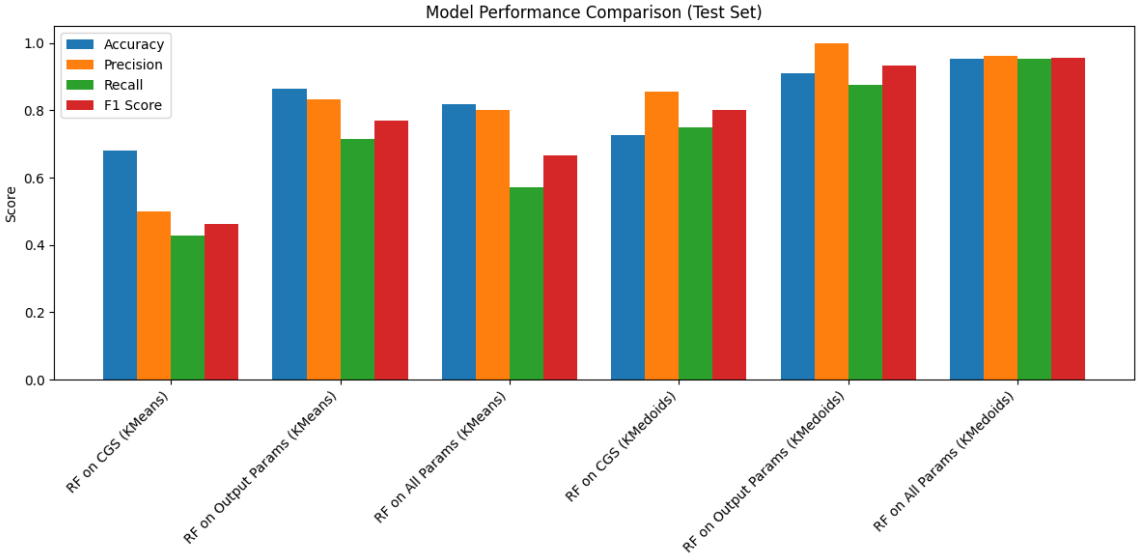



Figure 3.3.1: Evaluation Metrics for Random Forest Algorithm

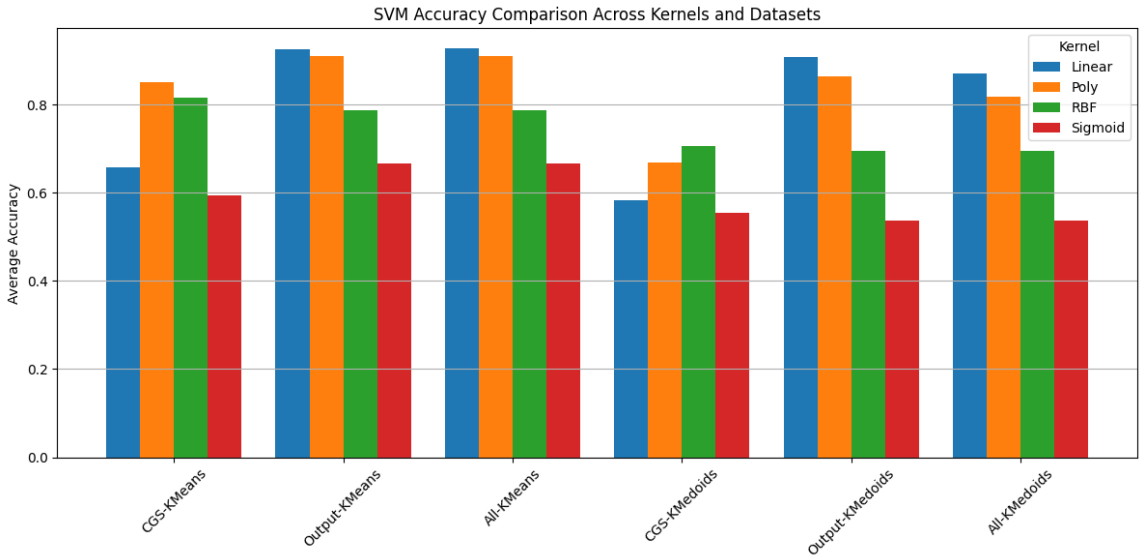



Figure 3.3.2: SVM accuracy plot across all models

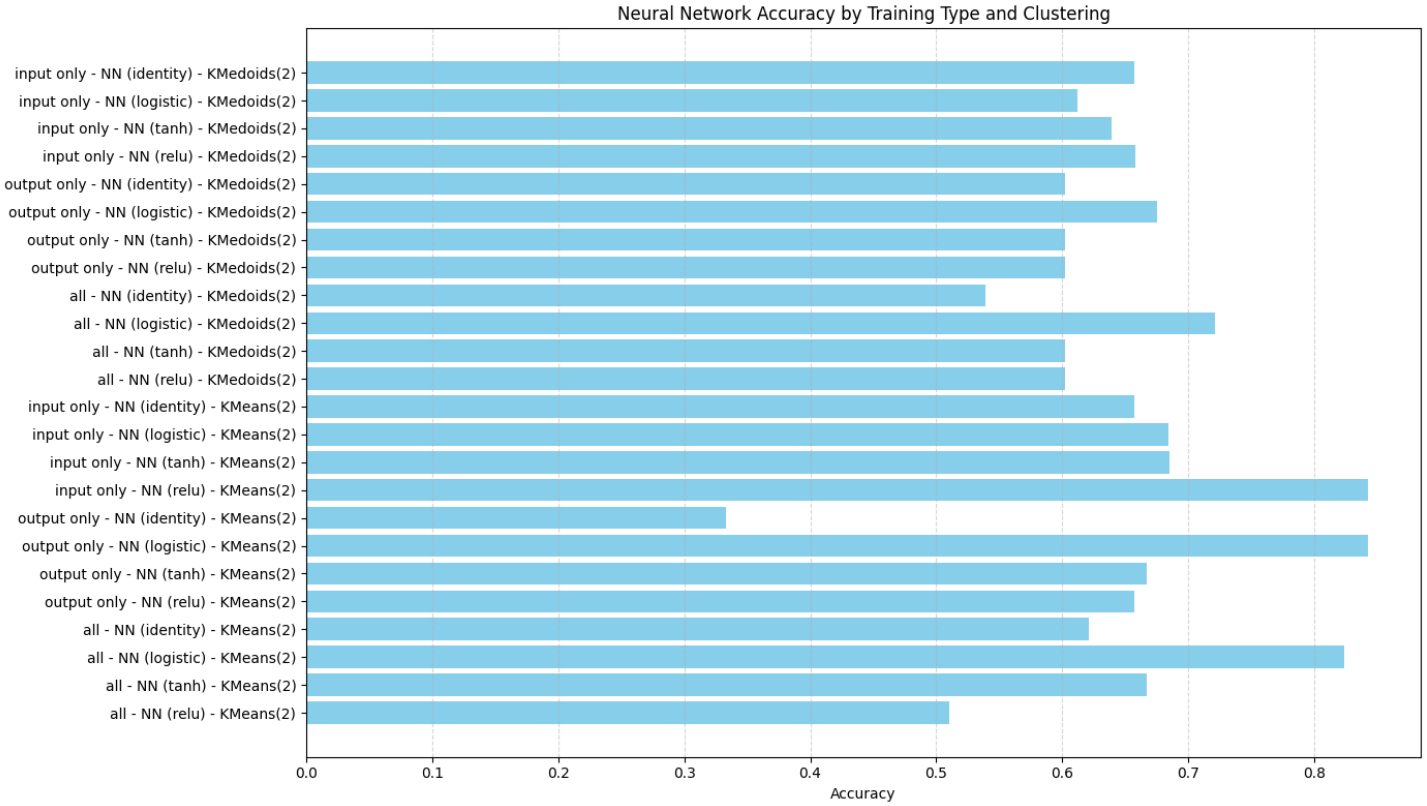



Figure 3.3.3: Neural Networks Evaluation across all models

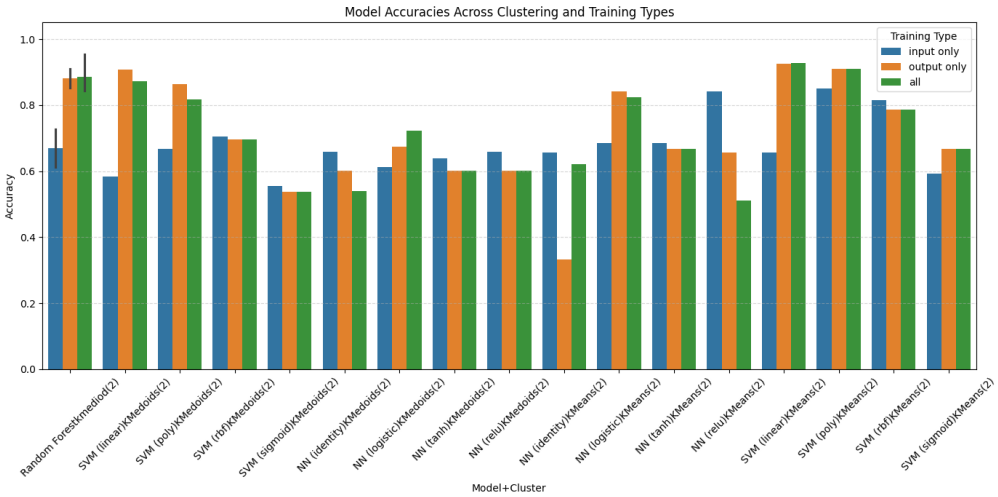



Figure 3.3.4: Classification Model Evaluation

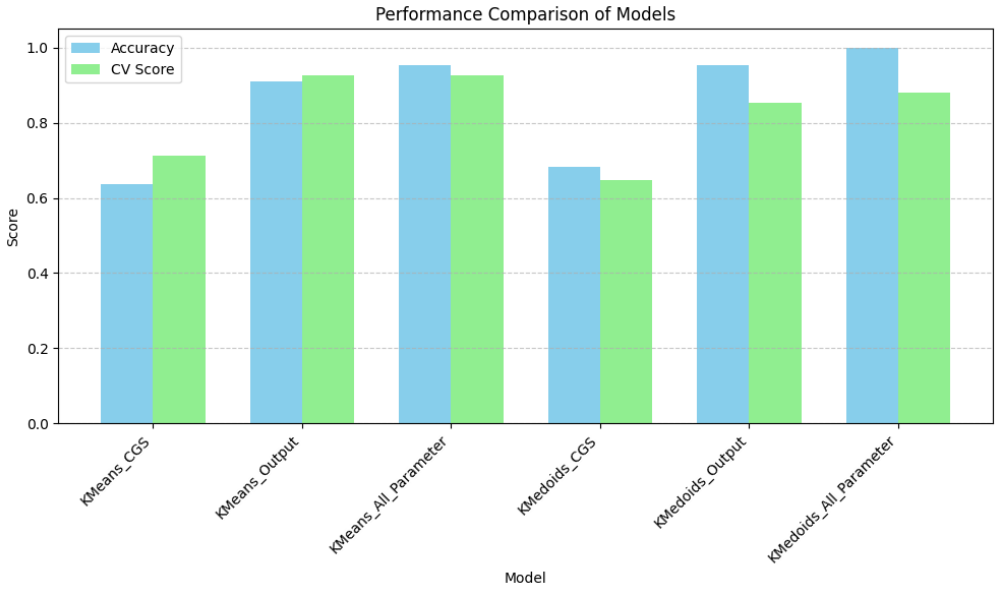



Figure 3.3.5: XGBoost - Model Evaluation

Based on the comprehensive evaluation of various models trained on different parameter subsets (input only, output only, all parameters) and clustered using **KMeans(2)** and **KMedoids(2)**, we draw the following conclusion:
**Best Model:** **XGBoost** on KMeans(2) with All parameters, having an accuracy of **0.9545**, and cross validation score of **0.9264**

This configuration demonstrates the highest balance between performance metrics and cross-validation stability, making it the most robust classifier for soybean classification.

## Observations

- Training on **All Parameters** yields the best results compared to training on input/output parameters alone.
- **XGBoost** consistently outperforms other models when hyperparameter tuning is applied.
- **KMeans(2)** clustering slightly edges out KMedoids(2) in top-performing combinations.
- **Neural Networks** show inconsistent performance and are not preferred.
- **SVM with Linear Kernel** also delivers competitive results, especially when used with all or output parameters.

### Recommendation

For optimal soybean classification performance, we recommend using:

| **XGBoost** trained on **All Parameters** with **KMeans(2)** clustering |
|---|

## 3.4 Analysis to Get Optimal Conditions for Maximizing Soybean Productivity

This section focuses on identifying conditions that maximize soybean productivity using two complementary approaches:

- **(1) Cluster-based analysis:** Models were trained after applying K-Means clustering ($k = 2$) to separate high vs low productivity groups.
- **(2) Regression-based analysis:** Without using clustering, we trained predictive models using a composite yield metric derived from biological indicators.

### 3.4.1 Composite Yield Metric Definition

To quantify soybean productivity while considering both yield and nutritional quality, we constructed a new metric called the **Composite Yield Metric**, defined as follows:

1. Compute Weighted Yield per Unit Area (WYUA):

$$WYUA = \left(\frac{\text{Weight of 300 Seeds (W3S)}}{300}\right) \times \text{Seed Yield per Unit Area (SYUA)}$$

2. Normalize WYUA and Protein Percentage (PPE) to [0, 1] using MinMaxScaler.
3. Compute the final composite score:

$$\text{Composite\_Yield\_Metric} = \left(\frac{WYUA_{\text{scaled}} + PPE_{\text{scaled}}}{2}\right) \times 100$$

This metric ranges between 0 and 100, providing a balanced view of both yield quantity and protein content.

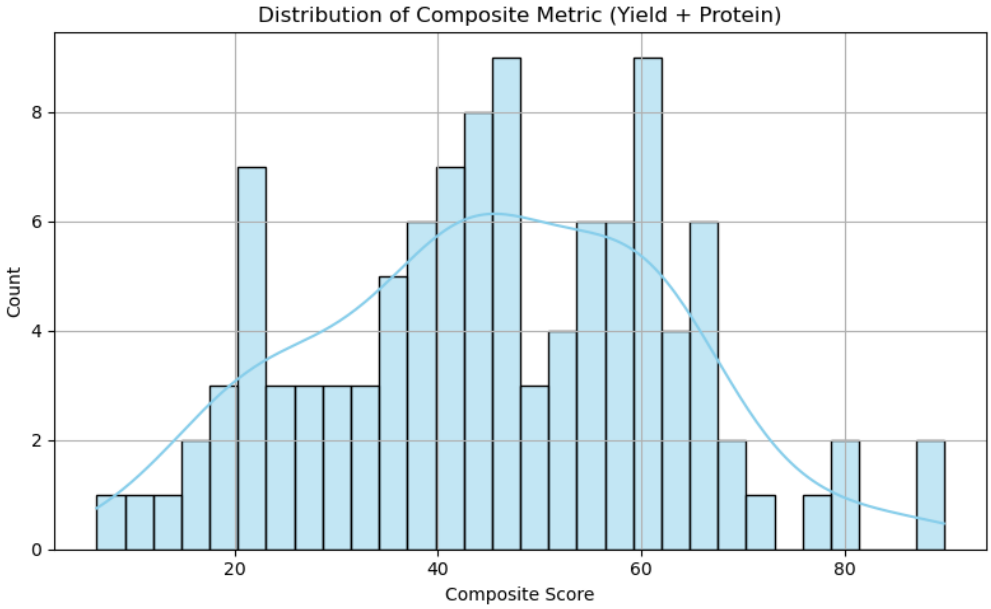
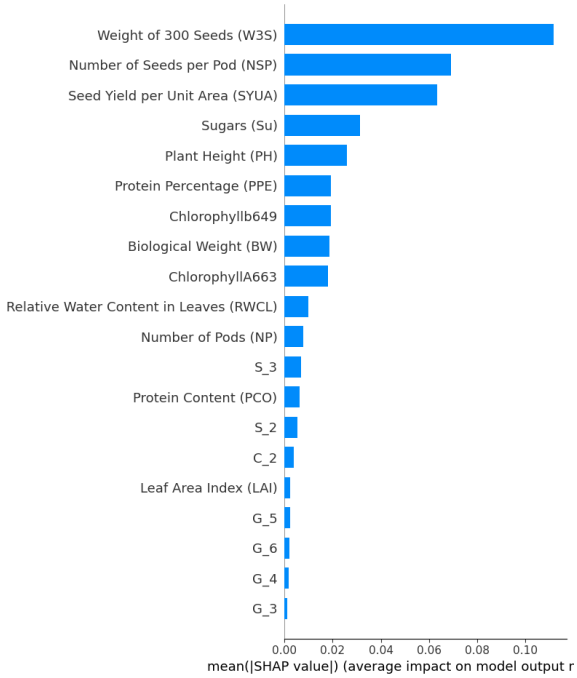



Figure 3.4.1: Distribution of the Composite Yield Metric

### 3.4.2 SHAP Feature Importance (Cluster-Based)

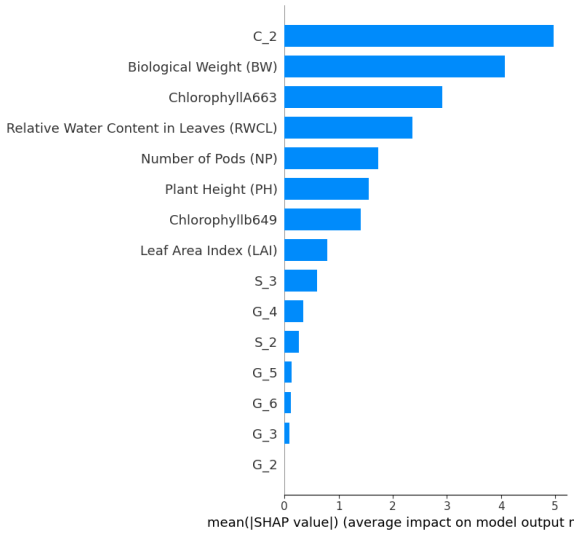



(a) SHAP Bar Plot – Cluster-based

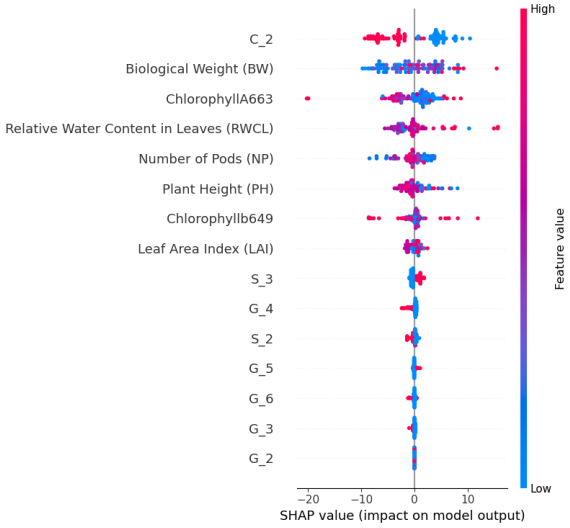

(b) SHAP Dot Plot – Cluster-based

Figure 3.4.2: SHAP Summary Plots (Cluster-based Model)

### 3.4.3 SHAP Feature Importance (Regression-Based)




(a) SHAP Bar Plot – Regression




(b) SHAP Dot Plot – Regression

Figure 3.4.3: SHAP Summary Plots (Regression-based Composite Yield)

### 3.4.4 Feature Importance and Ranges

#### Top Features (Regression Importance)

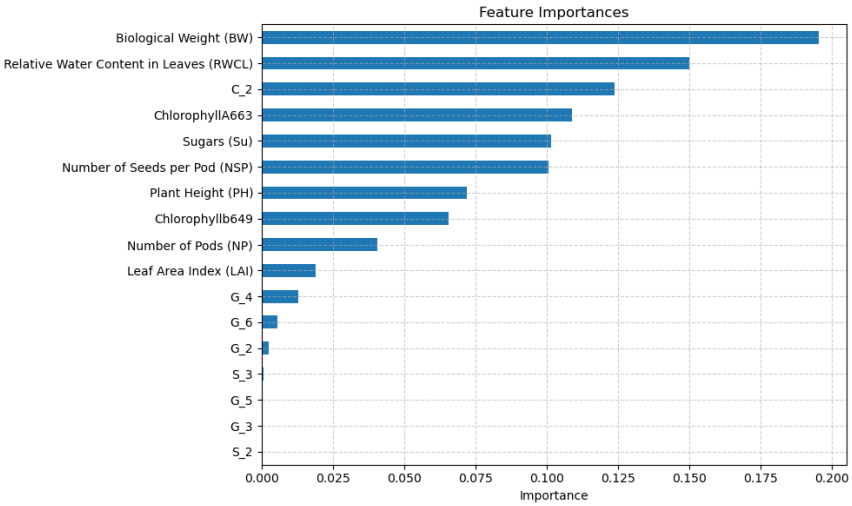
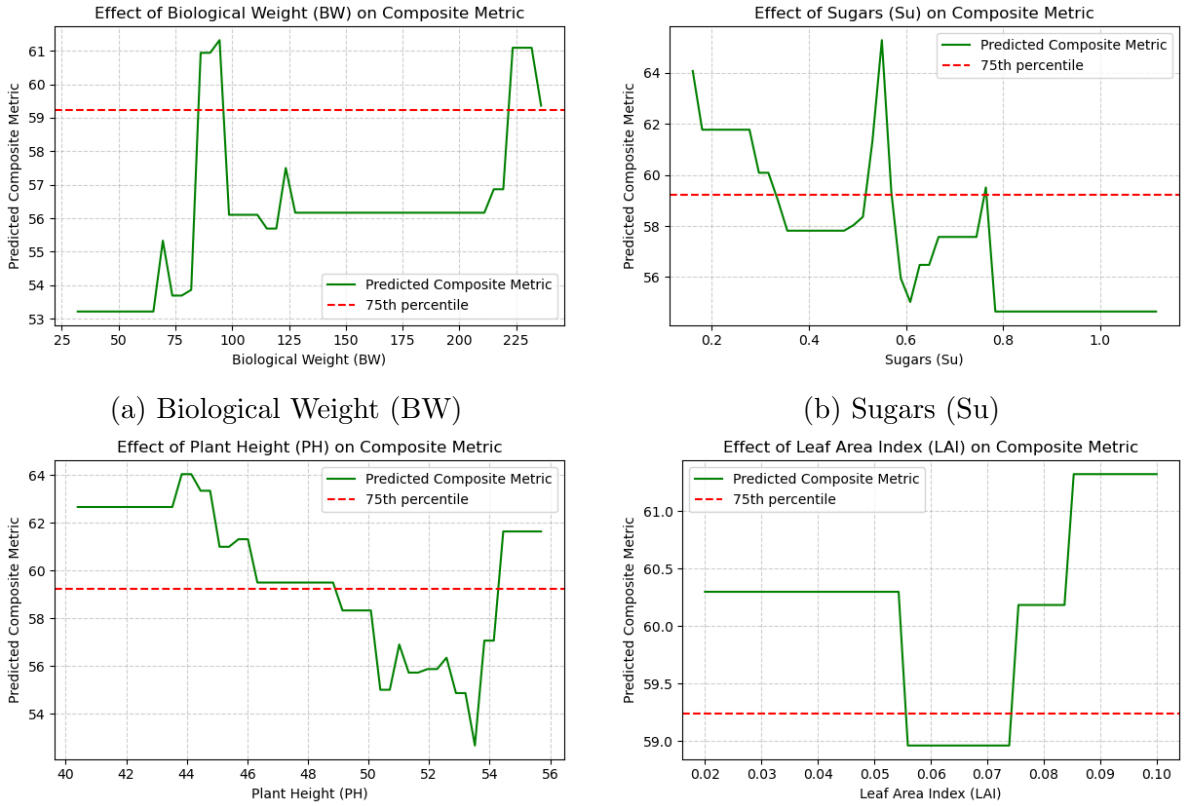



Figure 3.4.4: Feature Importance for Composite Yield Metric

#### Optimal Ranges Identified

Based on univariate analysis of key features, the following ranges are associated with higher productivity:

- **Biological Weight (BW):** 80–90
- **Sugars (Su):** 0.2–0.3 and 0.45–0.55
- **Plant Height (PH):** 40–49 and 54–56
- **Chlorophyllb649:** 1–3
- **Leaf Area Index (LAI):** 0.02–0.05 and 0.075–0.10




(a) Biological Weight (BW)

(b) Sugars (Su)

(c) Plant Height (PH)

(d) Leaf Area Index (LAI)

Figure 3.4.5: Effect of Key Features on Composite Yield Metric

### 3.4.5 Partial Dependence and SHAP Waterfall

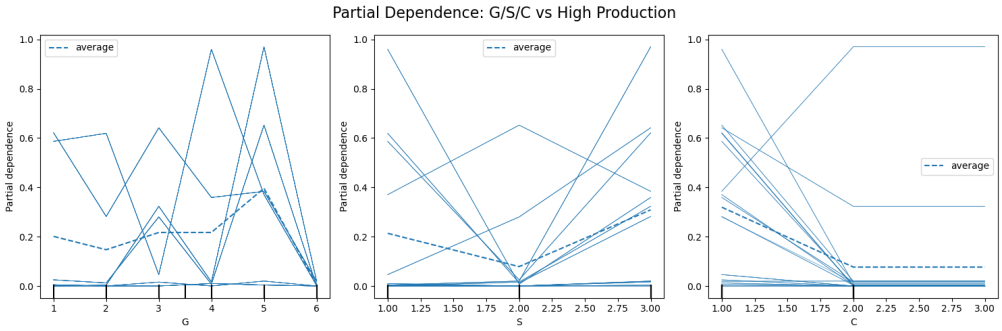



Figure 3.4.6: Partial Dependence Plot – G/S/C vs High Production Probability

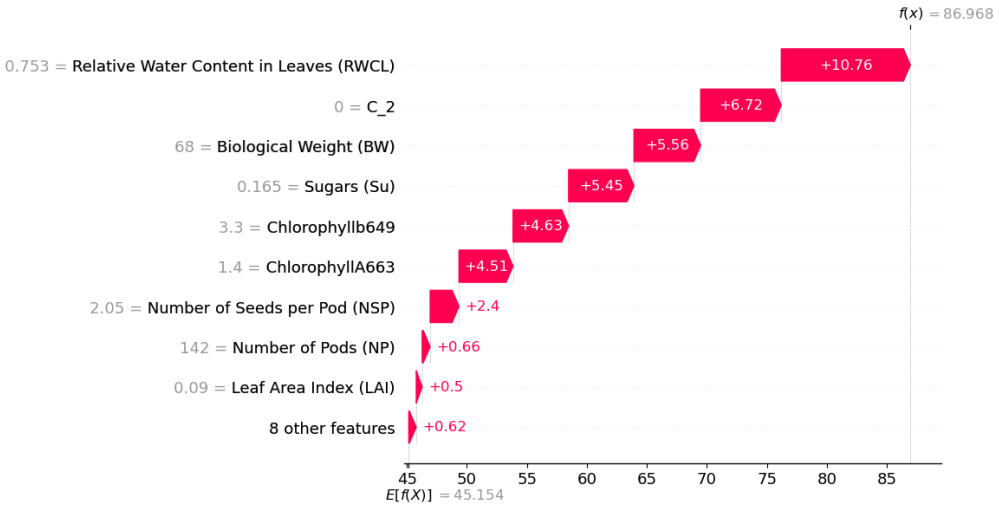



Figure 3.4.7: SHAP Waterfall Plot – Top Performing Sample

### 3.4.6 Feature Impact: Experimental Conditions

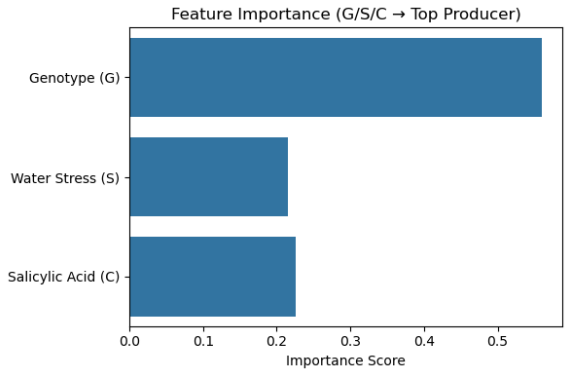
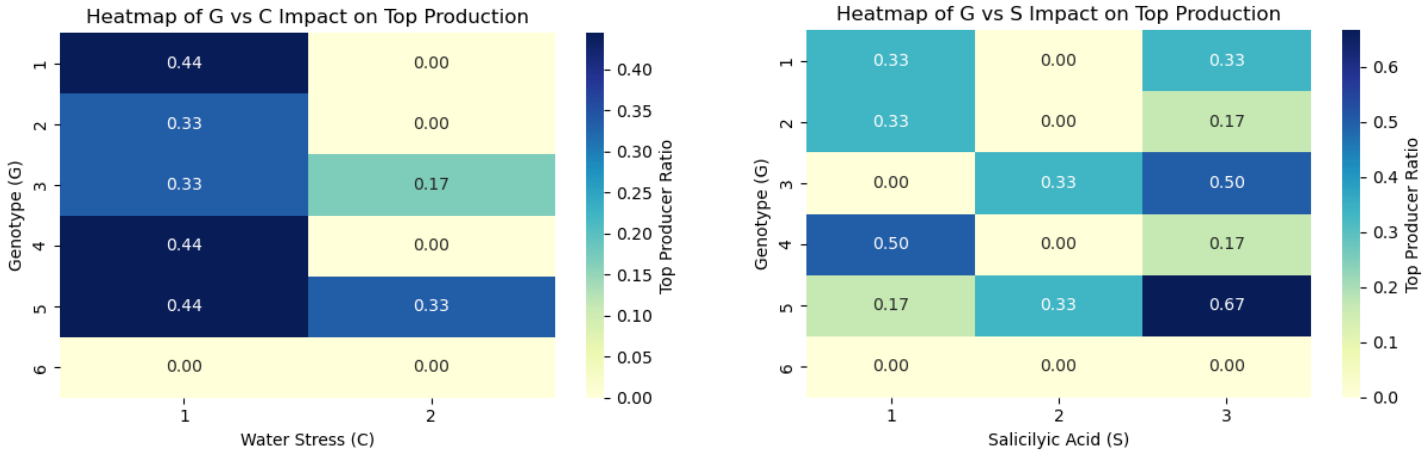



Figure 3.4.8: Feature Importance (G, S, C → High Production)

### 3.4.7 Heatmaps: Genotype and Treatment Combinations




(a) Genotype vs Salicylic Acid (C)

(b) Genotype vs Water Stress (S)

Figure 3.4.9: Heatmaps Showing Top Performing Combinations

### 3.4.8 Interpretable Decision Tree Rules for High Productivity

To derive human-interpretable insights, a simple decision tree regressor was trained to model the Composite Yield Metric. The resulting decision rules are summarized below, showing feature thresholds that lead to high or low productivity outcomes.

**Decision Tree Rules for High Yield + Protein Composite:**

- **If** Biological Weight (BW) $\leq 85.5$:
  - **If** C_2 $\leq 0.5$:
    - **If** Chlorophyllb649 $\leq 3.05$:
      - **If** BW $\leq 75.0$: **Predicted Composite Score = 59.85**
      - **Else**: **Score = 44.81**
    - **Else if** Chlorophyllb649 $\leq 3.2$: **Score = 68.20**
    - **Else**: **Score = 89.77**
  - **Else if** C_2 $\geq 0.5$:
    - **If** ChlorophyllA663 $\leq 1.55$:
      - **If** Chlorophyllb649 $\leq 3.15$: **Score = 30.80**
      - **Else**: **Score = 46.00**
    - **Else if** ChlorophyllA663 $\geq 1.55$:
      - **If** G_6 $\leq 0.5$: **Score = 20.39**
      - **Else**: **Score = 34.36**
- **Else if** BW $\geq 85.5$:
  - **If** Leaf Area Index (LAI) $\leq 0.08$: **Score = 66.82**
  - **Else**: **Score = 29.19**

Figure 3.4.10: Extracted Decision Tree Rules for High Composite Yield Metric

# Chapter 4
# Conclusion and Future Scope

## 4.1 Conclusion

Key findings include:

- The **XGBoost classifier**, trained on **all parameters** and clustered using **KMeans(2)**, emerged as the best-performing model with an accuracy of **95.45%**, precision and F1-score of **1.0** and **0.9231**, respectively, and a cross-validation score of **92.64%**.
- Training on **all parameters** consistently outperformed models trained on input-only or output-only features, highlighting the benefit of comprehensive data inclusion.
- Feature interpretation through **SHAP** and **Partial Dependence Plots (PDPs)** emphasized the importance of **Seed Yield per Unit Area**, **Protein Content**, **W3S**, and **Sugars** in driving productivity.
- Agronomic treatment factors such as **Genotype G1/G3**, **250 mg Salicylic Acid**, and **moderate water stress (70% FC)** were associated with optimal soybean performance.

Together, these insights form a robust data-driven framework to support decision-making in soybean cultivation and stress management.

## 4.2 Future Scope

The current work opens several avenues for further exploration:

1. **Temporal Analysis:** Integrate time-series data to model the progression of stress effects and crop growth, enabling dynamic prediction and adaptive treatment scheduling.
2. **Explainable AI:** Further deploy advanced interpretability techniques (e.g., counterfactual analysis, LIME [11]) to understand decision boundaries and model behavior in uncertain scenarios.
3. **Transfer Learning:** Explore transfer learning frameworks to adapt the trained models to other leguminous crops or new soybean genotypes without retraining from scratch [12].
4. **Precision Agriculture Integration:** Combine this classification framework with drone or satellite-based phenotyping data for real-time in-field decision support systems [13].

This work provides a blueprint for integrating machine learning with agronomic science to optimize crop productivity, especially under complex biotic and abiotic stress conditions. Continued research in this direction holds promise for enhancing food security and sustainable agriculture practices.

# Bibliography


[1] Wikipedia contributors, "Soybean — wikipedia," 2024, accessed: 2025-04-15. [Online]. Available: https://en.wikipedia.org/wiki/Soybean

[2] scikit-learn developers, "sklearn.cluster.kmeans — scikit-learn documentation," 2024, accessed: 2025-04-15. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[3] Wikipedia contributors, "K-medoids — wikipedia," 2024, accessed: 2025-04-15. [Online]. Available: https://en.wikipedia.org/wiki/K-medoids

[4] scikit-learn developers, "sklearn.manifold.tsne," 2024, accessed: 2025-04-11. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

[5] ——, "sklearn.decomposition.pca — scikit-learn documentation," 2024, accessed: 2025-04-15. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[6] H. Gültekin, "Silhouette score," 2020, accessed: 2025-04-11. [Online]. Available: https://medium.com/biased-algorithms/silhouette-score-d85235e7638b

[7] scikit-learn developers, "sklearn.metrics.davies_bouldin_score — scikit-learn documentation," 2024, accessed: 2025-04-15. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html

[8] ——, "sklearn.metrics.calinski_harabasz_score — scikit-learn documentation," 2024, accessed: 2025-04-15. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html

[9] W. Contributors, "Bayesian information criterion," https://en.wikipedia.org/wiki/Bayesian_information_criterion, 2025, accessed: 2025-04-15.

[10] ——, "Akaike information criterion," https://en.wikipedia.org/wiki/Akaike_information_criterion, 2025.

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, 2016.

[12] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[13] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018.