

Intelligent Crop Yield Prediction for Precision Farming

Kashyap Gorasiya [2413103]

CS530 : MACHINE LEARNING

MINI PROJECT



Table of Contents

1. Problem Statement
2. Dataset Overview
3. Data Augmentation & Missing Values
4. Feature Correlation
5. Dimensionality reduction and visualization :
6. Machine Learning Algorithms
7. Train-Test Split
8. K-Nearest Neighbors (KNN)
9. KNN Results
10. Decision Tree
11. Decision Tree Results
12. Support Vector Machine (SVM)
13. SVM Results
14. Predict Labels – test50.csv
15. Predicted Output – test50_pred.csv
16. Model Comparison
17. Conclusion

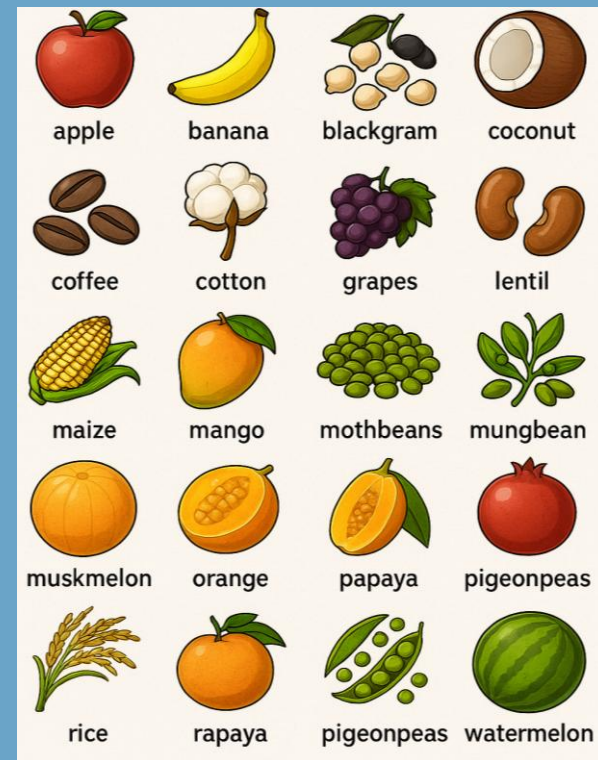
Problem Statement

- Precision farming leverages data-driven techniques to optimize agricultural productivity.
- Predicting crop yield is essential for informed decision-making by farmers.
- Traditional methods are often inaccurate, and do not scale well with data.
- **Objective:** Develop an AI-based crop yield prediction system using environmental parameters such as soil quality, weather conditions, and location.
- Use ML algorithms (KNN, Decision Trees, SVM) for both classification and regression tasks.



Dataset Overview

- **Dataset Used:** Crop Recommendation Dataset
- **Shape:** (2200 rows, 7 features + 1 label)
- **Features:**
 - Soil Quality : N (Nitrogen), P (Phosphorus), K (Potassium)
 - Weather : Temperature, Humidity
 - Location : pH, Rainfall
 - Crop Type : Label
- **Classes:** 22 distinct crop types:



['apple', 'banana', 'blackgram', 'chickpea', 'coconut', 'coffee', 'cotton', 'grapes', 'jute', 'kidneybeans', 'lentil', 'maize', 'mango', 'mothbeans', 'mungbean', 'muskmelon', 'orange', 'papaya', 'pigeonpeas', 'pomegranate', 'rice', 'watermelon']

Data Augmentation for Missing Values

- **Handling Missing Values:**
 - Ensured data completeness using statistical imputation.
 - Filled missing values with respective **Class-wise Feature Means**.
- **Descriptive Statistics (used for imputation):**

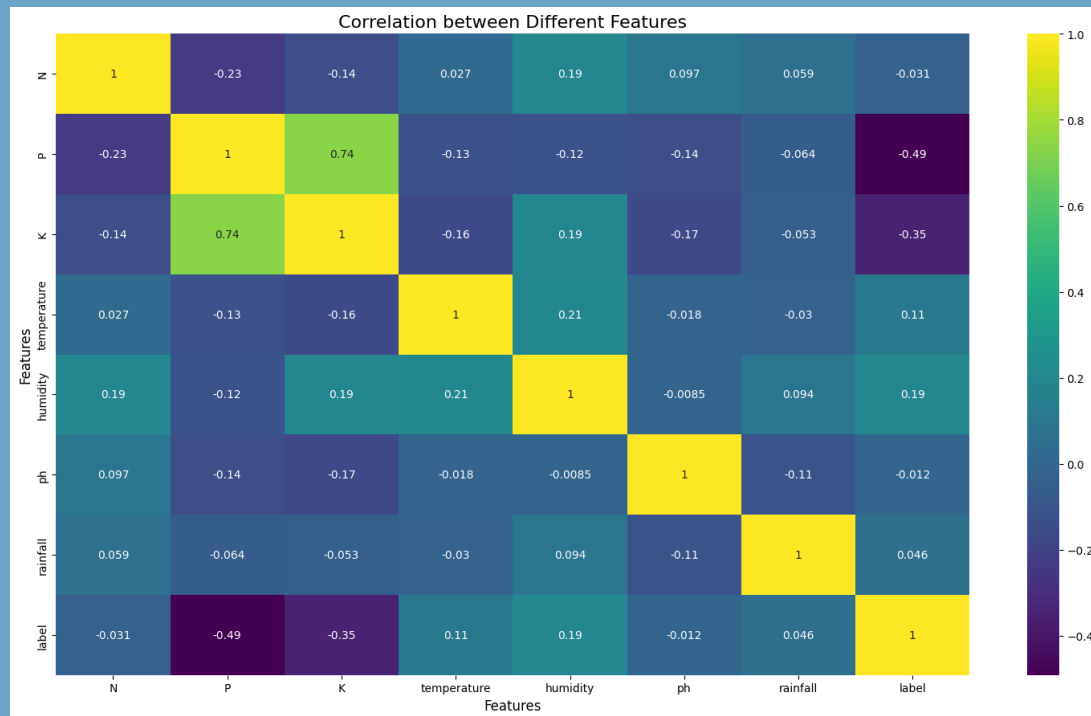
Crop	N	P	K	Temp (°C)	Humidity (%)	pH	Rainfall (mm)
Apple	20.80	134.22	199.89	22.63	92.33	5.93	112.65
Banana	100.23	82.01	50.05	27.38	80.36	5.98	104.63
Blackgram	40.02	67.47	19.24	29.97	65.12	7.13	67.88
Chickpea	40.09	67.79	79.92	18.87	16.86	7.34	80.06
Coconut	21.98	16.93	30.59	27.41	94.84	5.98	175.69
...
Crop	N	P	K	Temp (°C)	Humidity (%)	pH	Rainfall (mm)

$$\mu_{\text{class}}(f) = \frac{1}{n} \sum_{i=1}^n f_i$$

here n=100 (samples of classes),
f = features (N,P,K Temp , ..)

- This ensured cleaner, more reliable data for modeling.

Feature Correlation



Correlation measures how two variables move in relation to each other.

- +1 → Perfect positive correlation
- 0 → No correlation (independent)
- -1 → Perfect negative correlation

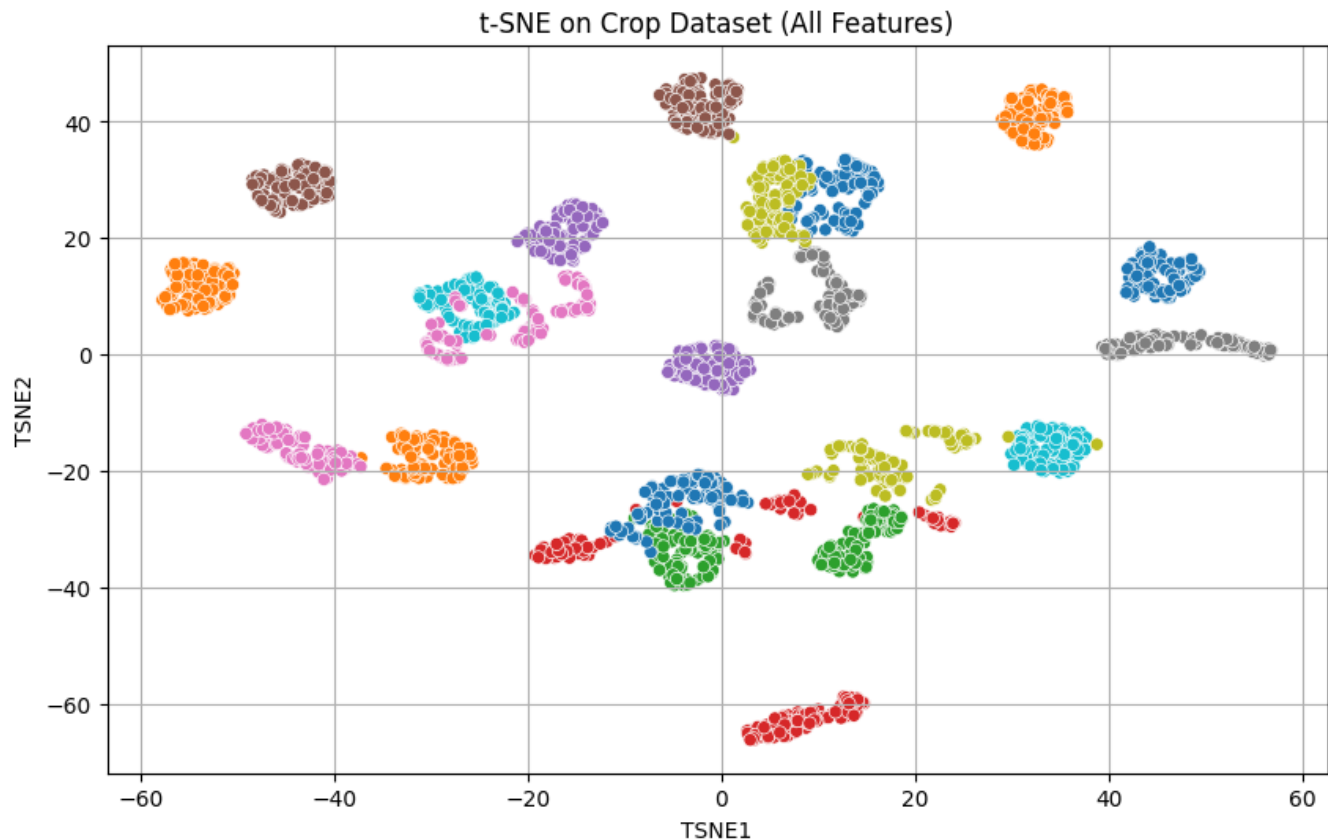
Pearson Correlation Coefficient

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \cdot \sum(y_i - \bar{y})^2}}$$

Insights:

- P & K: strong positive correlation (0.74)
- P & Label: moderate negative correlation (-0.49)
- Others: mostly weak or no correlation

Dimensionality reduction and visualization : t-SNE



Machine Learning Algorithms

Goal: Predict suitable crops or yield based on soil and weather conditions using machine learning.

Algorithm	Task Type	Application	Strengths	Limitations
KNN	Classification	Recommend crop based on similar conditions	Simple, intuitive, no training phase	Slow on large datasets, affected by noise
	Regression	Estimate yield based on nearest neighbors	Good for local patterns, non-parametric	Sensitive to feature scaling
Decision Tree	Classification	Decision Rule-based crop recommendation	Easy to interpret, handles non-linear separation	Can overfit without pruning
	Regression	Yield prediction with decision rules	Fast inference, handles missing values	Unstable—small data changes can alter the tree
Support Vector	Classification	Finds optimal boundary for crop classification	Works well in high-dimensional space, robust to outliers	Requires careful tuning, slower on large data
	Regression (SVR)	Margin-based model for precise yield prediction	High accuracy with proper kernel/parameter tuning	Computationally expensive, sensitive to scaling

Summary

- **KNN** is great for detecting **local patterns** – similar soil/weather → similar crops or yields
- **Decision Trees** help **understand feature impact** – which factors are most influential
- **Support Vector (SVM/SVR)**: Effective for **complex, non-linear relationships**-strong performance with tuning
- Used both for **classification (crop type)** and **regression (yield estimation)**

Train-Test Split

Dataset Breakdown (Total Rows: 2200)

Training: 70% (1540 rows)

Testing: 30% (660 rows)

```
x_train, x_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_state=2)
```

Dataset	Rows	Columns
x_train	1540	7
x_test	660	7
y_train	1540	1
y_test	660	1

K-Nearest Neighbors (KNN)

Overview:

- A supervised learning algorithm used for classification
- Classifies based on similarity to nearby points
- Uses distance functions to find the nearest neighbors:

- Euclidean Distance

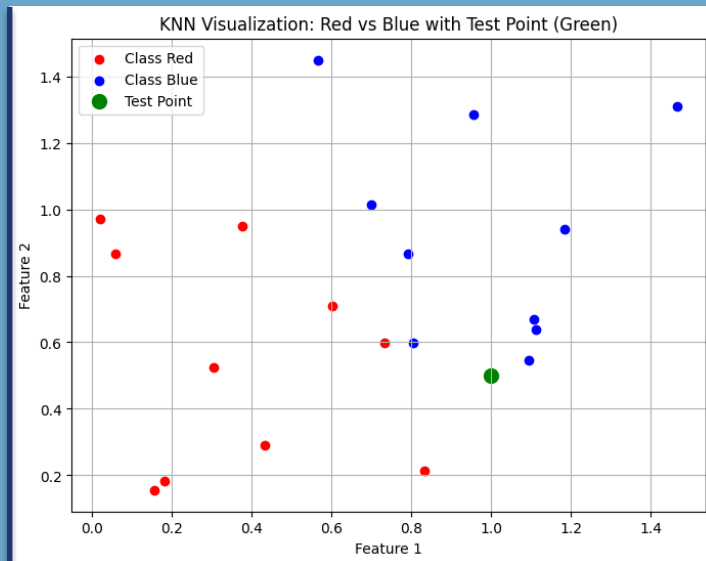
$$d = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

- Manhattan Distance

$$d = \sum_{i=1}^n |x_i - y_i|$$

Algorithm Steps:

1. Input a new data point (test sample)
2. Calculate distance to all points in the training set
3. Sort the distances in ascending order
4. Select top 'k' nearest neighbors
5. Classify the new point based on majority vote from neighbors



Hyperparameter: k

- Small k (e.g., 1-3):

Sensitive to noise and outliers

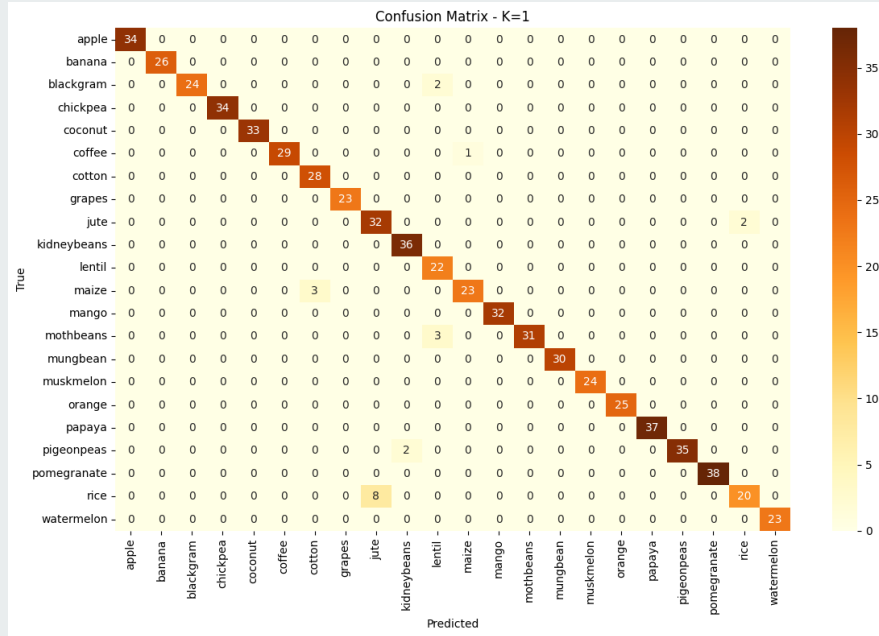
- Large k (e.g., 10+):

More stable, but may ignore local structure

- $K < \sqrt{n}$; n = total number of data points
- Use cross-validation to find the optimal k

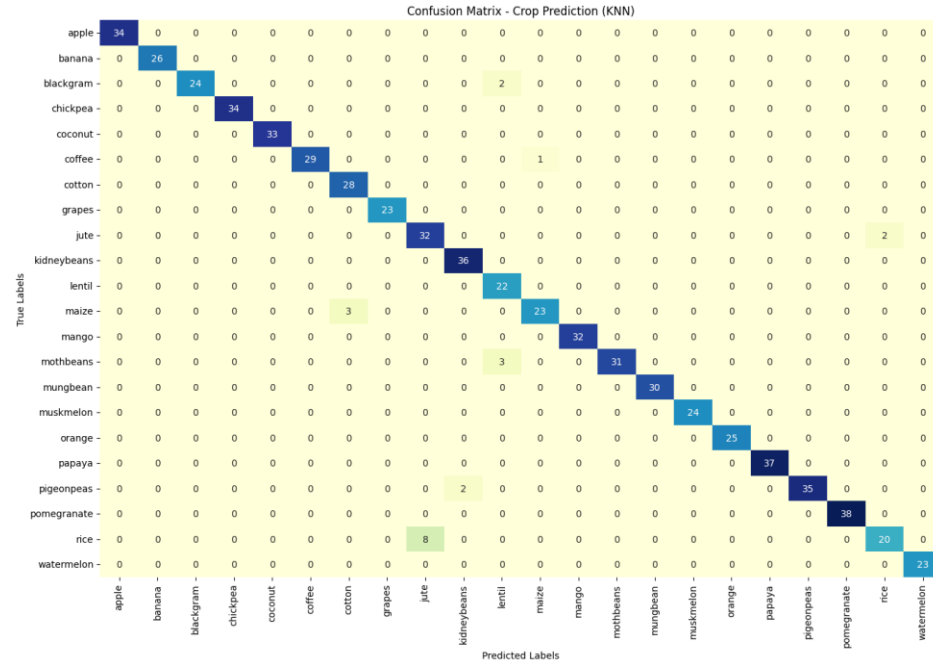
K-Nearest Neighbors (KNN)

Results (Confusion Matrix)



k = 1

Accuracy Score: 0.9681



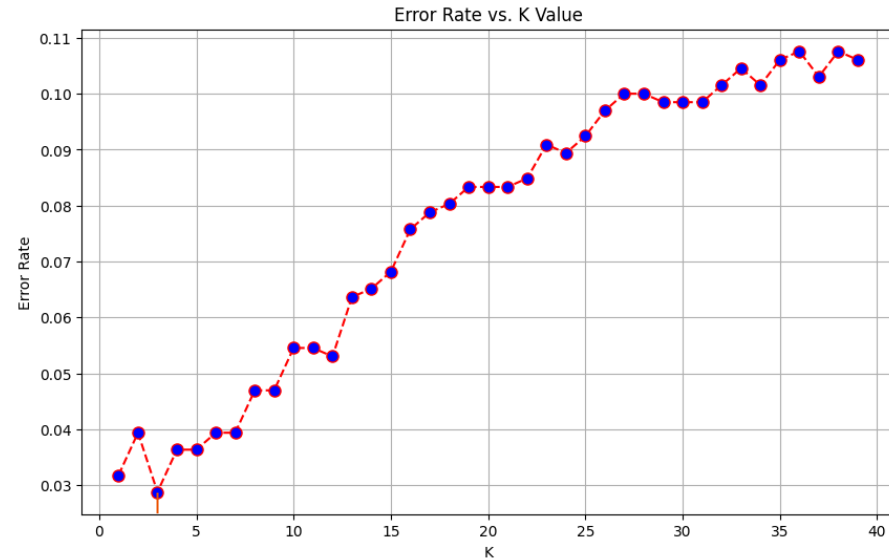
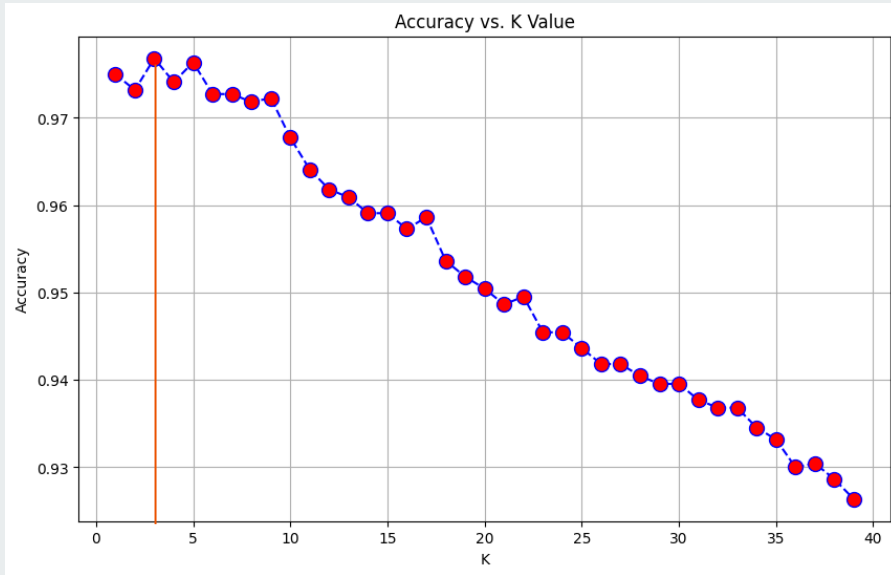
best_k = 3

Accuracy Score: 0.9712

K-Nearest Neighbors (KNN)

Results (Accuracy and Error Rate)

Cross-Validation



- The optimal value of $k = 3$ gave the **highest accuracy** and **lowest error rate** among all tested values from 1 to 39.

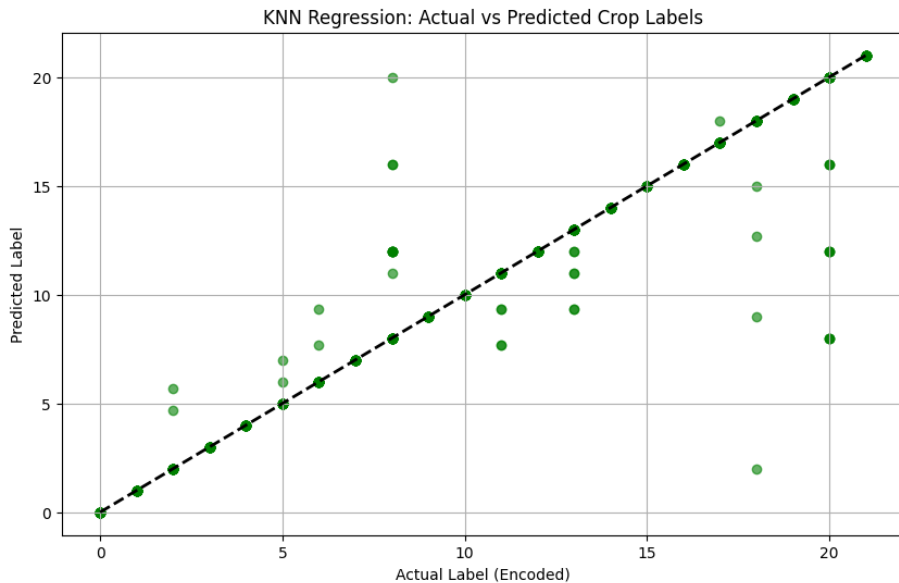
KNN Regression Results



Performance Metrics:

- R^2 Score: 0.9343
- Mean Squared Error (MSE): 2.6554

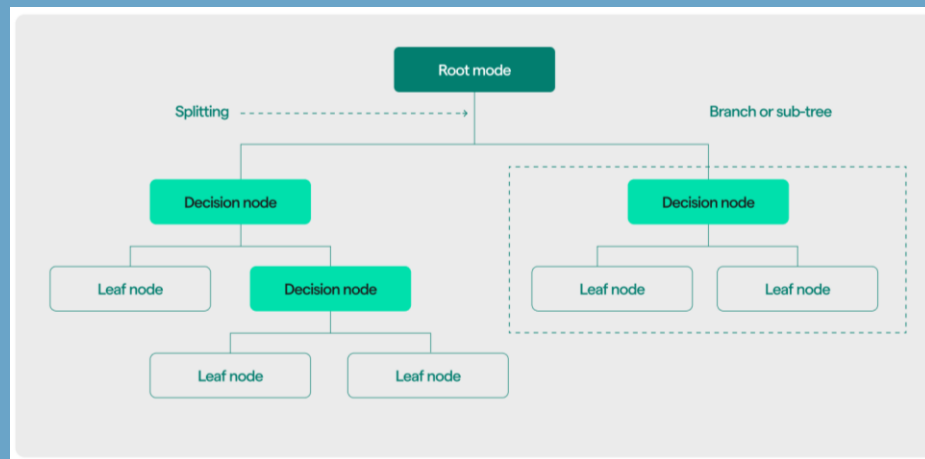
Actual vs Predicted Crop Labels



Decision Tree

A Decision Tree is a supervised learning algorithm used for **classification** and **regression**. It models decisions with a tree-like structure where:

- **Internal nodes:** Test on features.
- **Branches:** Represent outcomes.
- **Leaf nodes:** Final predictions.



Decision rule assigns a class label based on splitting criteria.
The tree grows by choosing the **best feature** to split on.

Decision Tree

How to Select a Split?

Purity of a Split (Entropy & Gini Index)

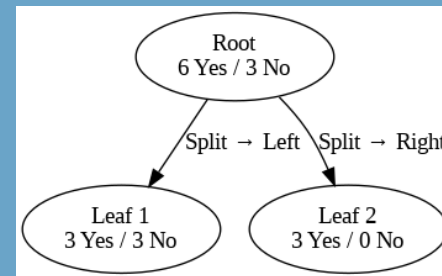
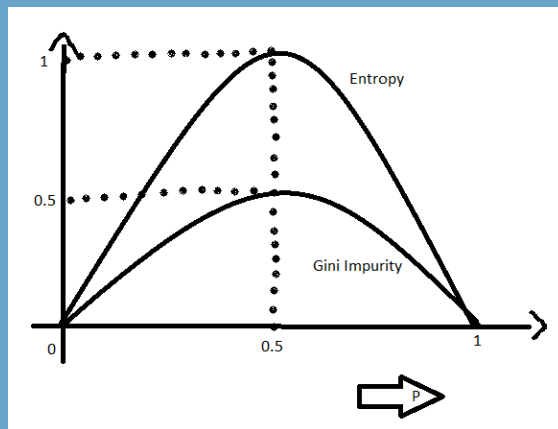
- Entropy (Disorder):

$$Entropy = - \sum_{i=1}^n p_i \log_2 p_i$$

- Range: 0 (pure) to 1 (impure).
- Gini Index (Impurity):

$$GiniIndex = 1 - \sum_{i=1}^n p_i^2$$

- Range: 0 (pure) to 1 (impure).



When to Split?

- Best Split: Select the feature with the **Highest Information Gain**.

$$InformationGain = Entropy(Parent) - \sum_{k=1}^m \frac{|S_k|}{|S|} \times Entropy(S_k)$$

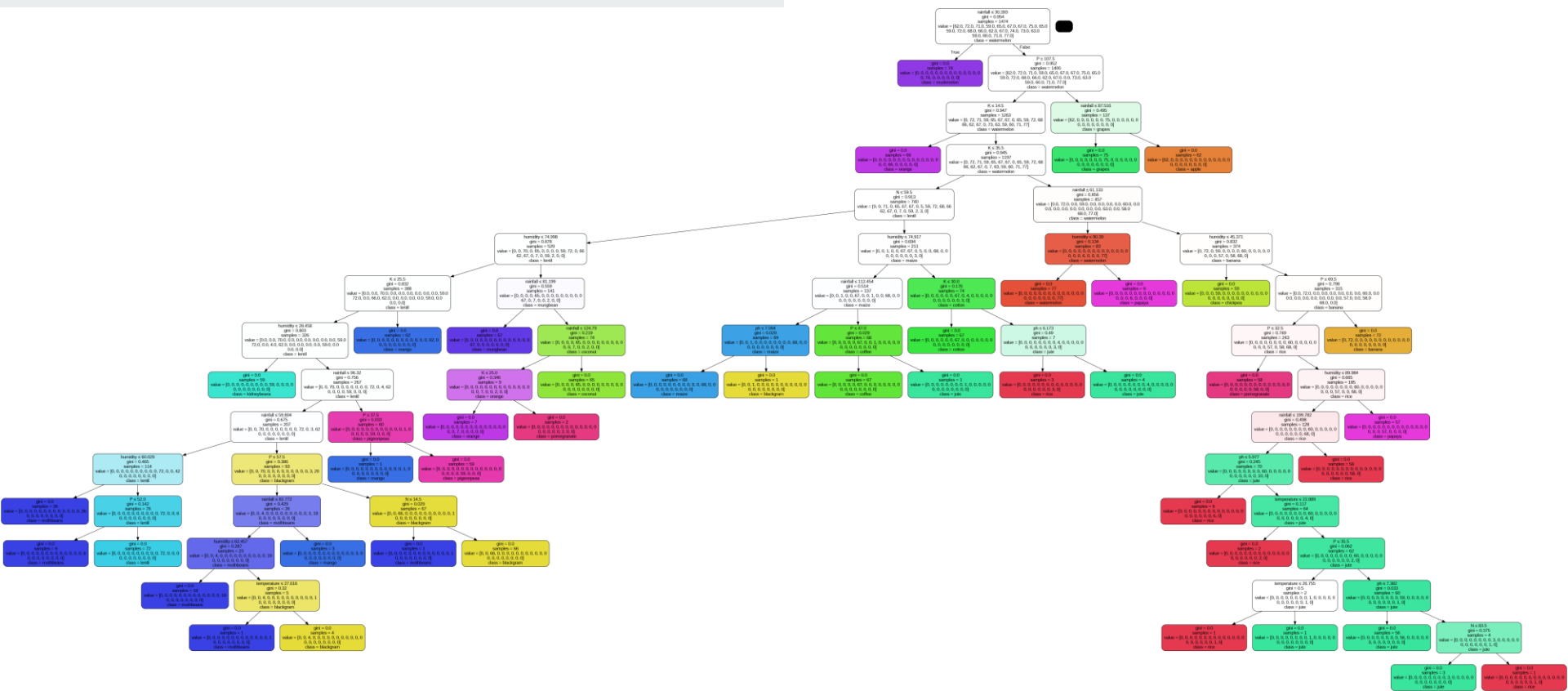
- Entropy(Parent): Entropy of the parent node (original dataset).
- |S_k|: Size of subset k after the split.
- |S|: Total size of the parent dataset.
- S_k: Subsets formed after split based on a feature.
- m: Total number of subsets after the split.

Decision Tree : Overfitting & Solutions

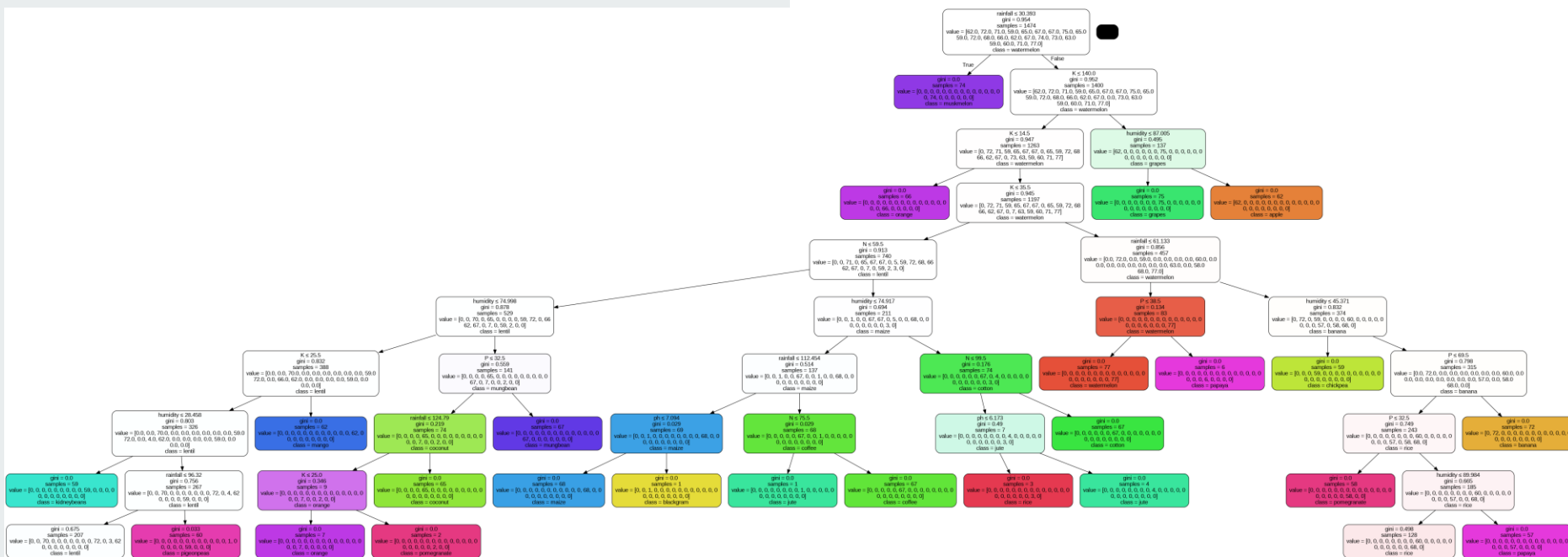
- **Overfitting:** The tree becomes too complex, leading to poor generalization.
- **Pruning:** Removes unnecessary parts of the tree.
 - **Post-pruning:** Prune after the tree is built.
 - **Pre-pruning:** Stop tree growth before it's fully built.
- **Hyperparameters for Overfitting:**
 - **Max Depth:** Limits tree depth.
 - **Min Samples Split:** Minimum samples required to split a node.
 - **Min Samples Leaf:** Minimum samples in a leaf node.

Decision Tree

Results (Complete Decision tree)



Postpruning max_depth=9

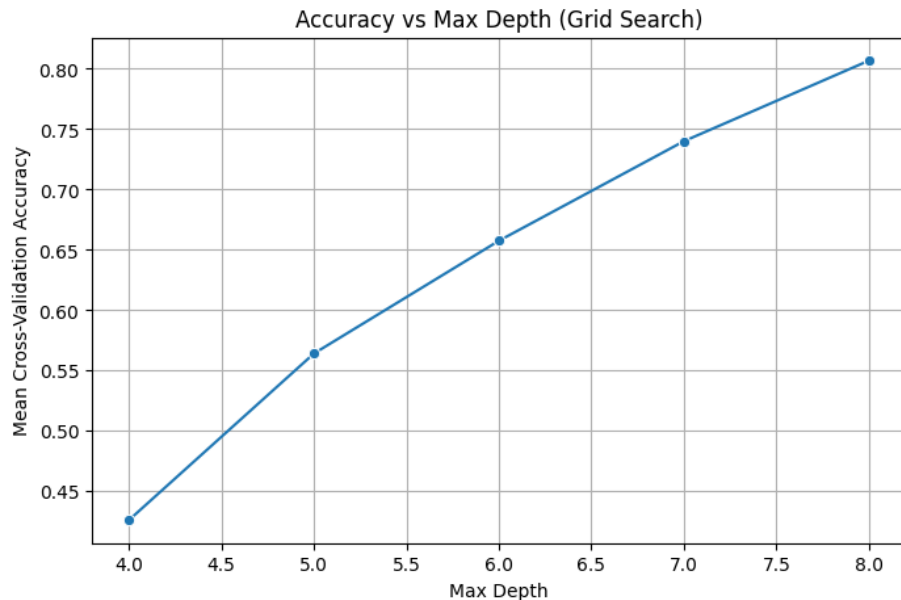


Decision Tree

Results (Decision tree :Cross-validation)

Prepruning max_depth=8

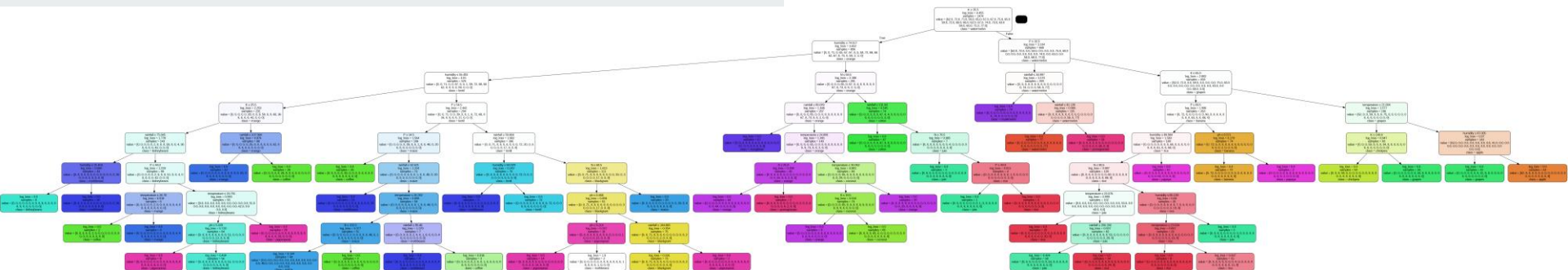
```
{ 'criterion': 'log_loss',  
  'max_depth': 8,  
  'max_features': 'sqrt',  
  'splitter': 'best' }
```



Decision Tree

Results (Decision tree :Cross-validation)

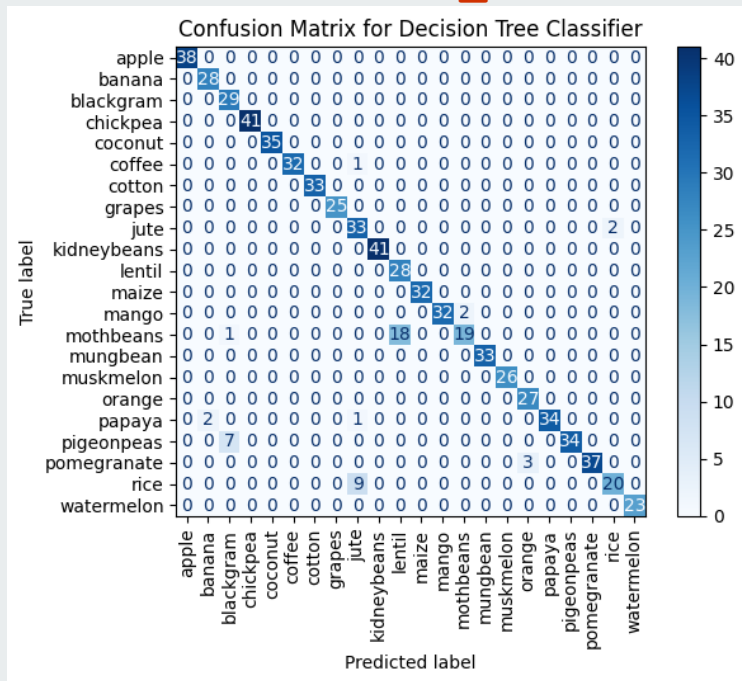
Prepruning max_depth=8



Decision Tree

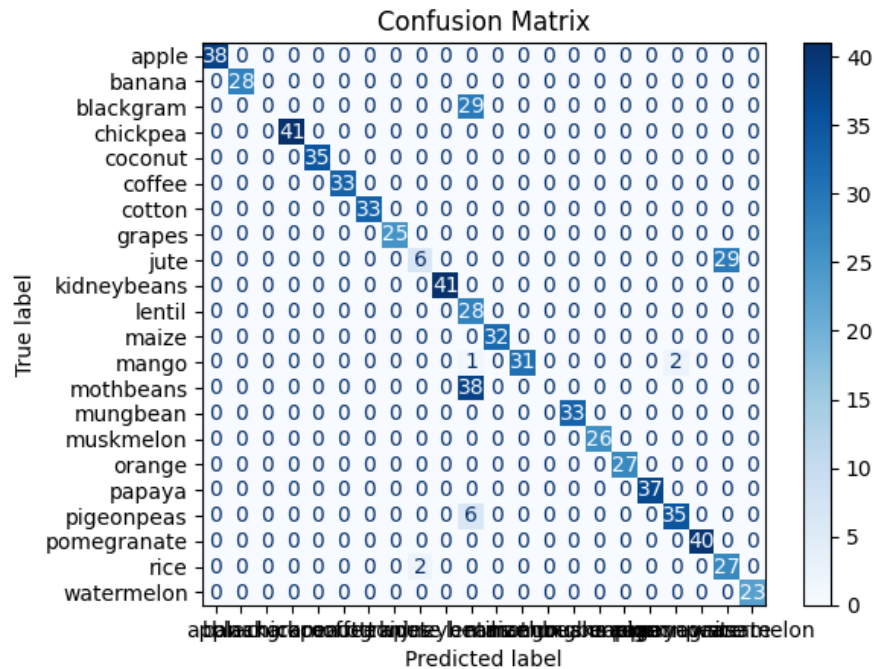
Results (Decision tree)

Prepruning max_depth=8



Accuracy Score: 0.9366

Postpruning max_depth=9



Accuracy Score: 0.85

Decision Tree Regression

- A supervised algorithm for **predicting continuous values** (e.g., crop yield).
- Splits the data based on input features to create regions with low target variance.
- At each **leaf node**, prediction = **mean** of target values in that node.

How It Works:

1. Start from the root (whole dataset).
2. At each node, find the **best split** to **reduce variance** in the target variable.
3. Repeat recursively to grow the tree.

$$Var(S) = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2$$

y_i = target value of the i -th sample
 \bar{y} = mean of target values in the node
 n = number of samples in the node

$$VarianceReduction = Var(Parent) - \sum_{i=1}^m w_i \cdot Var(Child_i)$$

$w_i = n_i / n$ --> Weight of each child node
 n_i = number of samples in child node i
 n = total number of samples in parent node
 $Var(Child_i)$ = Variance of child node i

Each split is to **select the feature and threshold** that gives the **maximum Variance Reduction**.

$$BestSplit = \arg \max_{\text{all possible splits}} (VarianceReduction)$$

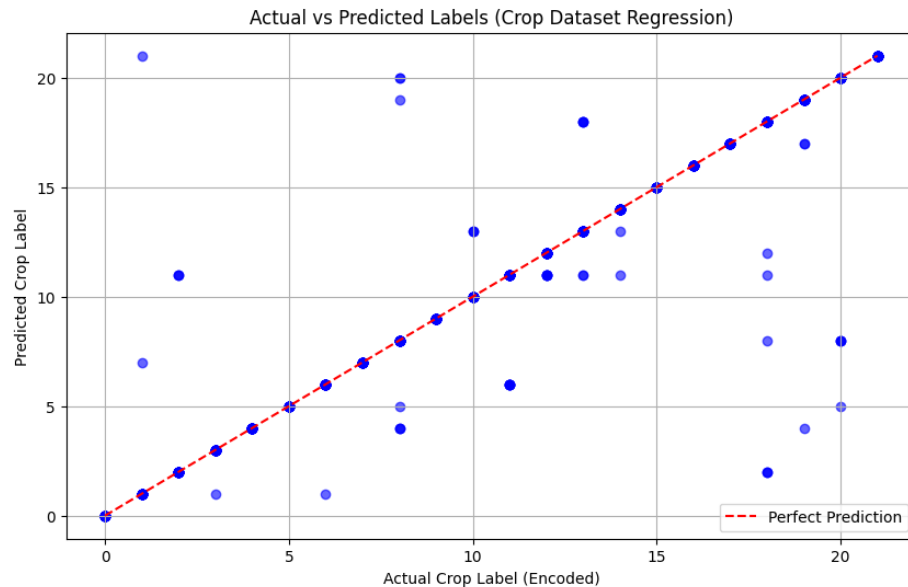
Decision Tree Regression Results

Best Parameter

```
{'criterion': 'absolute_error',  
'max_depth': 12,  
'max_features': 'log2',  
'splitter': 'best'}
```

Performance Metrics:

• R^2 Score: 0.8786

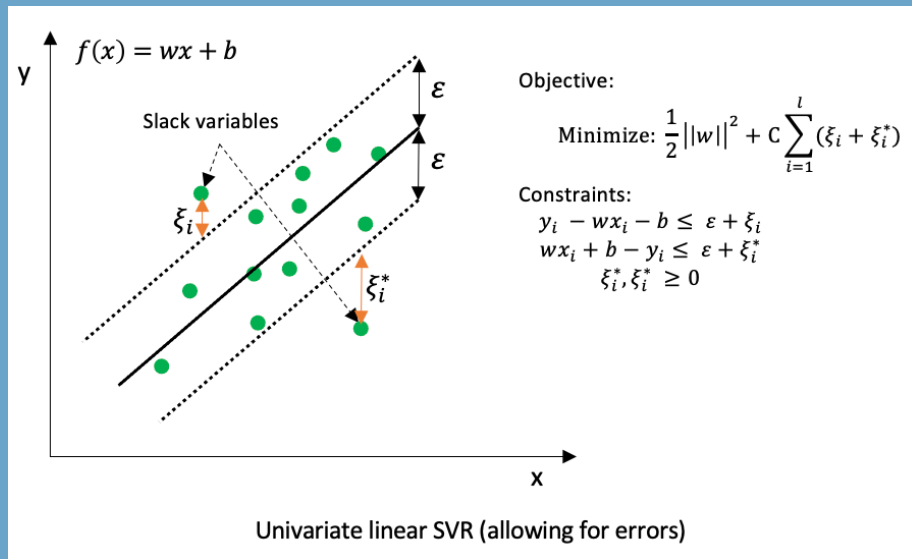


Support Vector Regression

- SVR is used for **regression**, predicting a **continuous output**.
- Instead of separating classes, SVR fits a **function within a margin ϵ** from the true values.

Margin in SVR:

- **Objective:** Keep model flat \rightarrow Minimize: $\frac{1}{2} \|w\|^2$
- Similar to classification:
maximize margin = minimize weights



ϵ : tolerance margin

ξ_i, ξ_i^* : **slack variables** (errors beyond ϵ)

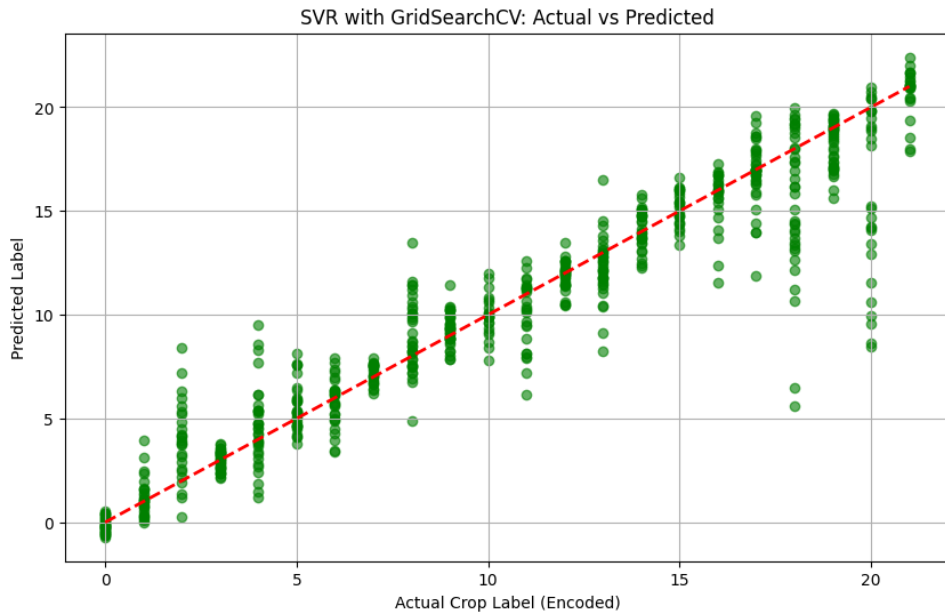
C: regularization parameter

Support Vector Regression Results



Performance Metrics:

- R^2 Score: 0 . 8988
- Mean Squared Error (MSE): 4 . 0918



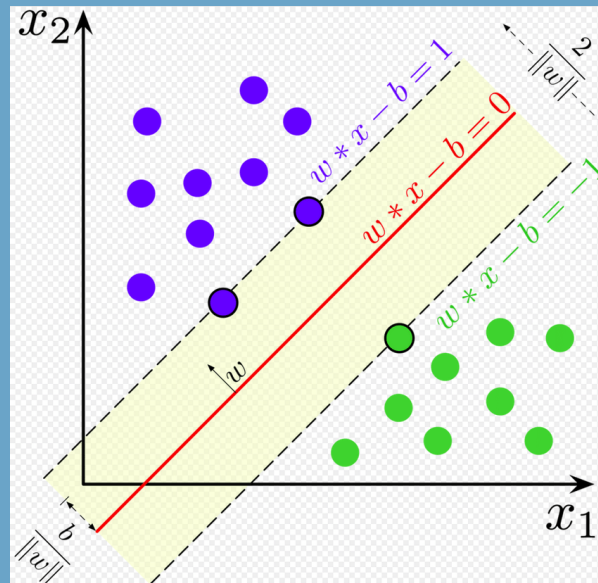
Support Vector Classification

- A supervised classification algorithm that finds the optimal hyperplane to separate data classes with the maximum margin.
- Supports linear and non-linear classification (via kernel trick).

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$

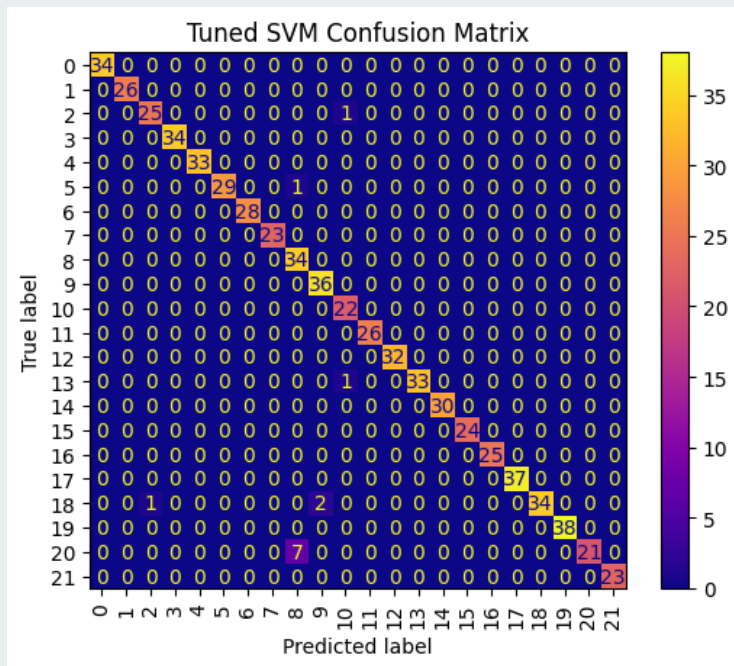
- $y_i \in \{-1, +1\}$ is the true label
- x_i : Feature vector
- w : Weight vector
- b : Bias

$$\text{Goal : Maximize margin} (\Rightarrow) \text{Minimize} \left(\frac{1}{2} \|\mathbf{w}\|^2 \right)$$

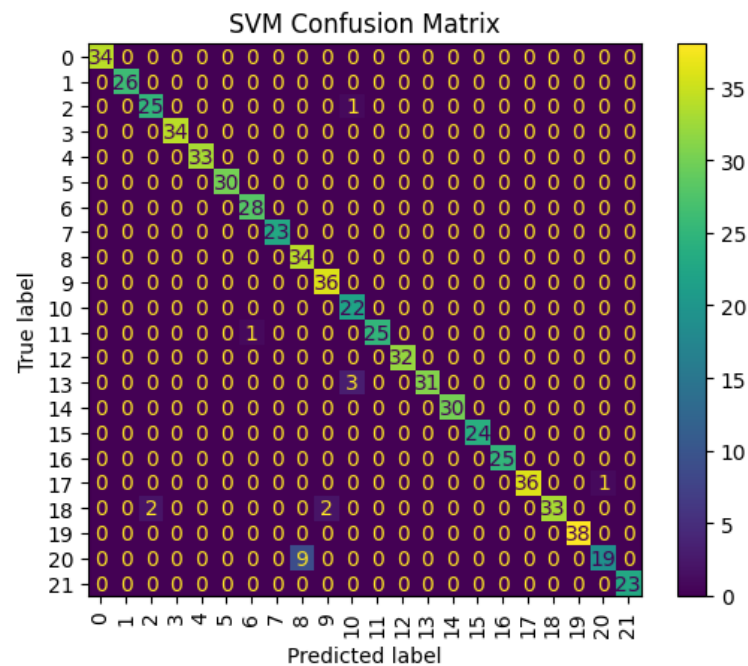


Support Vector Classification Results

Best Parameters: {'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}



Accuracy Score: 0.9803



Accuracy Score: 0.9712

Predict Labels : test50.csv

N	P	K	temperature	humidity	ph	rainfall
135.225	21.90391	108.2182	22.255591	66.38508	4.631889	264.6474
124.2138	19.28092	25.02405	15.42698	95.8879	9.829598	102.9343
47.09306	109.2855	142.1804	28.2724	78.98857	5.77959	298.3498
107.2684	7.640329	120.6689	41.153255	88.39695	7.53867	44.77721
40.1617	129.3661		9.1643906	30.07124	9.200731	208.9162
96.23455	107.1161	171.9446	26.782171	50.61415	8.793912	55.49008
121.6445	87.57683	29.01266	31.436731	82.19354	4.062357	293.3981
86.75497	94.87949	106.7264	24.805208	34.16507	6.663124	54.50357
41.00509	17.60328	22.96365	9.8803498	77.4472		80.64829
46.4787	7.46267	143.6062	16.989264	82.87719	6.606696	218.6624
104.0513	17.62106		40.998747	64.5186	3.742532	139.8785
38.02581			30.386221	16.83398	5.030952	261.3976
85.68839	126.3525		27.872203	88.57107	4.721368	41.21712
133.7115	73.98994	113.396		15.32035	6.734751	105.4392
10.1192	100.9658	151.9367	36.839521	32.3834	5.005432	85.21655
106.1618	95.02864	41.79687		28.13905		174.9878
126.4804	124.7721	58.91236		35.84702	3.532603	257.5849

Predict Labels : test50_pred.csv

N	P	K	temperature	humidity	ph	rainfall	knn_pred	knn_conf	tree_pred	tree_conf	svm_pred	svm_conf
135.225	21.90391	108.2182	22.25559	66.38508	4.631889	264.6474	rice	1	pomegranate	1	rice	0.305958
124.2138	19.28092	25.02405	15.42698	95.8879	9.829598	102.9343	cotton	1	pomegranate	1	mothbeans	0.145109
47.09306	109.2855	142.1804	28.2724	78.98857	5.77959	298.3498	rice	0.666667	grapes	1	papaya	0.179357
107.2684	7.640329	120.6689	41.15326	88.39695	7.53867	44.77721	papaya	1	muskmelon	1	papaya	0.234835
40.1617	129.3661	48.14909	9.164391	30.07124	9.200731	208.9162	chickpea	1	chickpea	1	mothbeans	0.135047
96.23455	107.1161	171.9446	26.78217	50.61415	8.793912	55.49008	chickpea	1	grapes	1	mothbeans	0.154357
121.6445	87.57683	29.01266	31.43673	82.19354	4.062357	293.3981	rice	1	rice	1	rice	0.185558
86.75497	94.87949	106.7264	24.80521	34.16507	6.663124	54.50357	chickpea	1	banana	0.7	chickpea	0.237091
41.00509	17.60328	22.96365	9.88035	77.4472	6.46948	80.64829	orange	1	mungbean	1	orange	0.467035
46.4787	7.46267	143.6062	16.98926	82.87719	6.606696	218.6624	rice	0.666667	grapes	1	papaya	0.228102
104.0513	17.62106	48.14909	40.99875	64.5186	3.742532	139.8785	mango	1	muskmelon	1	pigeonpeas	0.15246
38.02581	53.36273	48.14909	30.38622	16.83398	5.030952	261.3976	pigeonpeas	1	grapes	1	pigeonpeas	0.477947
85.68839	126.3525	48.14909	27.8722	88.57107	4.721368	41.21712	banana	1	banana	1	banana	0.222943
133.7115	73.98994	113.396	25.61624	15.32035	6.734751	105.4392	chickpea	0.666667	banana	0.7	chickpea	0.148744
10.1192	100.9658	151.9367	36.83952	32.3834	5.005432	85.21655	grapes	1	grapes	1	grapes	0.228147
106.1618	95.02864	41.79687	25.61624	28.13905	6.46948	174.9878	coffee	1	jute	1	pigeonpeas	0.201805
126.4804	124.7721	58.91236	25.61624	35.84702	3.532603	257.5849	rice	0.666667	rice	1	pigeonpeas	0.139842

Number of rows where all three predictions match: 9

Number of rows where any two predictions match: 33

Model Comparison – Classification

Classification Task: Crop Recommendation


Dataset: Crop Recommendation Dataset (Multi-class)

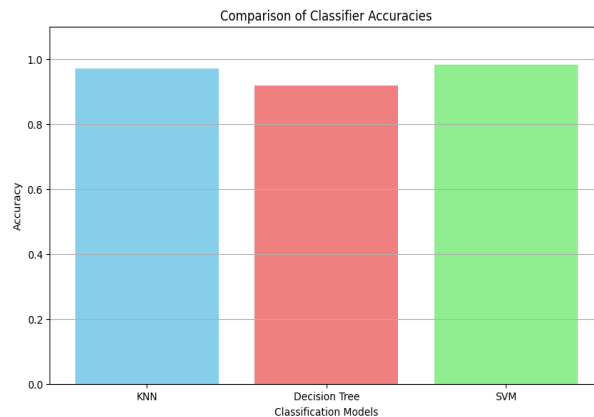
Models Evaluated:

- KNN Classifier (k=3)
- Decision Tree (criterion='log_loss', max_depth=8, max_features='sqrt')
- SVM Classifier (C=10, kernel='rbf', gamma='scale')



Accuracy Results:

Model	Accuracy
SVM	98.03% 
KNN	97.12%
Decision Tree	93.66%



- **SVM** showed the **best performance** in terms of classification accuracy and generalization on the test data.

Model Comparison – Regression

Regression Task: Crop Prediction

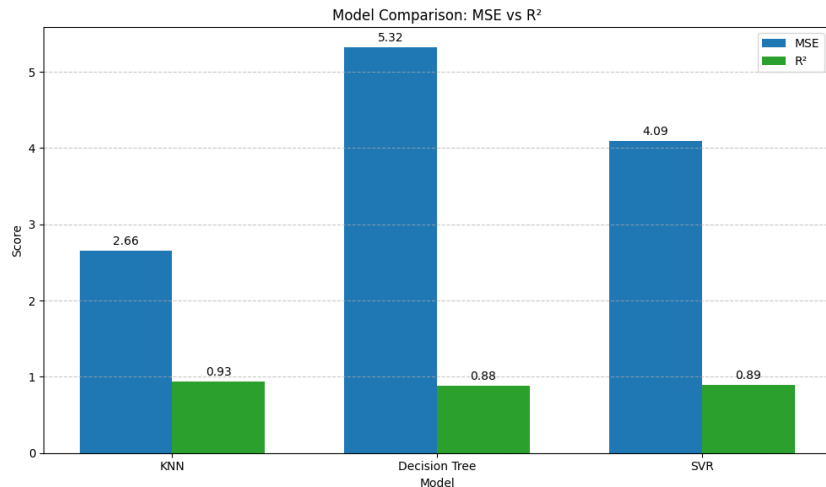
Models Evaluated:

- KNN Regressor (k=3)
- Decision Tree Regressor (criterion='absolute_error', max_depth=12, max_features='log2')
- SVR (C=10, epsilon=0.1, kernel='rbf')



Evaluation Metrics:

Model	MSE	R ² Score
KNN	2.6554 ✓	0.9343 ✓
Decision Tree	5.3182	0.8786
SVR	4.0918	0.8941



- **KNN Regressor** achieved the **lowest MSE** and **highest R²**, making it best suited for this regression task.

Conclusion



- **SVM** is the best model for **classification** (high accuracy and generalization).
- **KNN** is the most effective for **regression** (lowest error, best fit).

Questions?



References

- <https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset>
- <https://scikit-learn.org/stable/index.html>
- <https://matplotlib.org/stable/gallery/index.html>
- <https://seaborn.pydata.org/tutorial.html>
- [Krish Naik](#)