

# **TABLE OF CONTENTS**

## **1. INTRODUCTION**

**1.1 Brief Introduction**

**1.2 About the source**

**1.3 Goal**

## **2 Dataset Overview**

## **3 Exploratory Data Analysis (EDA) Findings**

## **4 Correlation Analysis and Feature Selection**

## **5 Model Development**

## **6 Evaluation Results and Conclusions**

## **7 Scope for future work**

**CREATED BY-**

**Anmol Bhatnagar**

**GOOGLE COLAB LINK:**

<https://colab.research.google.com/drive/1QA01D2GlzvhDQisnteGAiZKjzgLSaSTo?usp=sharing>

# **INTRODUCTION**

## **1.1 BREIF INTRODUCTION**

Agriculture is a vital pillar of the Indian economy, and optimizing crop prediction plays a crucial role in improving planning and resource management. In this mini project, the K-Nearest Neighbours (KNN) classification algorithm is applied to predict the type of crop grown on a given plot using real-world agricultural data. The dataset used contains information such as area of land, season, sowing and harvesting dates, yield, and crop types collected from various parts of the Cauvery River region.

## **1.2 ABOUT THE SOURCE**

The dataset was sourced from a publicly available dataset titled "Sickle tabular dataset", which was derived from government and field-level surveys and contains structured, labelled agricultural data for research and academic purposes. The data offers a rich view of seasonal patterns and crop diversity in Tamil Nadu, especially near the Cauvery River basin.

## **1.3 GOAL**

The primary goal of this project is to classify the crop type based on tabular features using the KNN algorithm. The project involves data exploration, visualization, feature analysis, model development, and performance evaluation. This system can potentially be extended into a decision support tool for agricultural advisory services, helping farmers and planners predict the most likely crop type for specific field and climate conditions.

# Dataset Overview

The dataset used in this project contains 217 records and 17 columns describing different attributes related to agricultural plots.

```
df.columns
```

```
Index(['UNIQUE_ID', 'PLOT_ID', 'RIVER_PART', 'STANDARD_SEASON', 'YEAR', 'AREA',  
      'CROP', 'VARIETY', 'PADDY_BIN', 'SOWING_DATE', 'TRANSPLANTING_DATE',  
      'HARVESTING_DATE', 'SOWING_DAY', 'TRANSPLANTING_DAY', 'HARVESTING_DAY',  
      'YIELD', 'SPLIT'],  
      dtype='object')
```

## Functions Used for Analysis:

To understand the dataset structure and quality, we applied the following key functions:

- `df.head()` – To preview the first few rows of the dataset
- `df.shape` – To understand dataset dimensions (217 rows × 17 columns)

```
[77] df = pd.read_csv("../content/metadata.csv")  
     print("Shape of the dataset:", df.shape)  
     df.head(10)
```

Shape of the dataset: (217, 17)

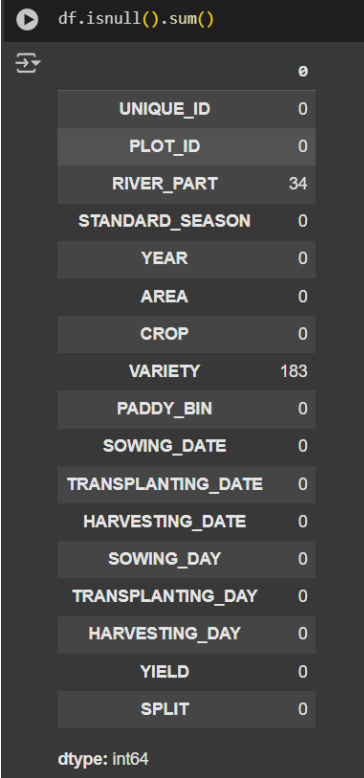
	UNIQUE_ID	PLOT_ID	RIVER_PART	STANDARD_SEASON	YEAR	AREA	CROP	VARIETY	PADDY_BIN	SOWING_DATE	TRANSPLANTING_DATE	HARVESTING_DATE	SOWING_DAY	TRANSPLANTING_DAY	HARVESTING_DAY
0	2315	417.0	NaN	oct-mar	2020.0	1.00	Paddy	Kaiser	1.0	10/3/2020	11/3/2020	2/13/2021	3.0		34.0
1	1774	219.0	Coastal Cauvery	dec-may	2019.0	0.09	Paddy	NaN	1.0	1/1/2020	0	5/31/2020	32.0		0.0
2	1592	195.0	Lower Cauvery	apr-aug	2020.0	0.34	Paddy	NaN	1.0	4/1/2020	0	7/31/2020	1.0		0.0
3	1253	161.0	Upper Cauvery	aug-jan	2019.0	0.19	Sugarcane	NaN	2.0	0	0	0	0.0		0.0
4	1685	211.0	Coastal Cauvery	oct-mar	2018.0	4.11	Coconut	NaN	2.0	0	0	0	0.0		0.0
5	1992	243.0	Coastal Cauvery	aug-jan	2019.0	0.06	Paddy	NaN	1.0	9/1/2019	0	12/31/2019	32.0		0.0
6	532	57.0	Middle Vennar	jun-oct	2018.0	0.11	Coconut	NaN	2.0	0	0	0	0.0		0.0
7	1104	148.0	Upper Cauvery	sep-feb	2019.0	0.06	Banana	NaN	2.0	0	0	0	0.0		0.0
8	2257	389.0	NaN	sep-feb	2019.0	1.00	Paddy	Kalsar	1.0	9/9/2019	10/10/2019	1/26/2020	9.0		40.0

- `df.info()` – To check data types and missing values

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 217 entries, 0 to 216  
Data columns (total 17 columns):  
#   column              Non-Null Count  Dtype  
---  ---  
0   UNIQUE_ID           217 non-null    int64  
1   PLOT_ID             217 non-null    float64  
2   RIVER_PART          183 non-null    object  
3   STANDARD_SEASON     217 non-null    object  
4   YEAR               217 non-null    float64  
5   AREA               217 non-null    float64  
6   CROP                217 non-null    object  
7   VARIETY             34 non-null     object  
8   PADDY_BIN           217 non-null    float64  
9   SOWING_DATE         217 non-null    object  
10  TRANSPLANTING_DATE  217 non-null    object  
11  HARVESTING_DATE     217 non-null    object  
12  SOWING_DAY          217 non-null    float64  
13  TRANSPLANTING_DAY   217 non-null    float64  
14  HARVESTING_DAY      217 non-null    float64  
15  YIELD              217 non-null    float64  
16  SPLIT              217 non-null    object  
dtypes: float64(8), int64(1), object(8)  
memory usage: 28.9+ KB
```

- `df.isnull().sum()` – To find the count of missing values in each column



```
df.isnull().sum()
```

	0
UNIQUE_ID	0
PLOT_ID	0
RIVER_PART	34
STANDARD_SEASON	0
YEAR	0
AREA	0
CROP	0
VARIETY	183
PADDY_BIN	0
SOWING_DATE	0
TRANSPLANTING_DATE	0
HARVESTING_DATE	0
SOWING_DAY	0
TRANSPLANTING_DAY	0
HARVESTING_DAY	0
YIELD	0
SPLIT	0

dtype: int64

- `df['CROP'].value_counts()` – To inspect the distribution of crop types
- `df.describe()` – To generate summary statistics for numerical features

### Key Observations:

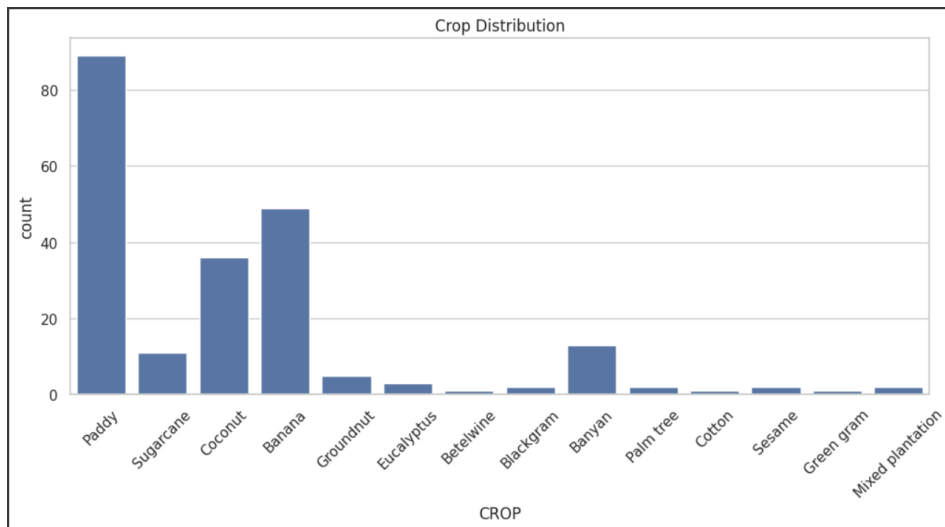
- Some columns had missing values, notably VARIETY and RIVER\_PART
- The target variable CROP initially had 14 unique classes, but several of them had less than 5 examples, which is too low for classification
- Rare crop types were filtered out to retain only 6 classes with sufficient representation
- Date columns were converted to day-of-year using `pd.to_datetime()` and `.dt.dayofyear` for modelling

This overview gave the understanding of the data, allowed identification of cleaning needs, and helped in choosing meaningful features for modelling.

# Exploratory Data Analysis (EDA) Findings

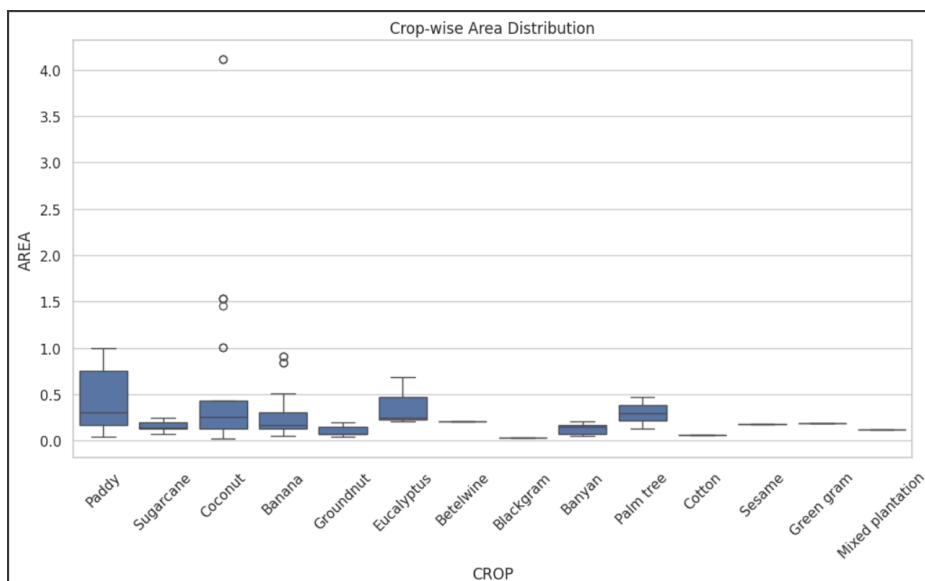
## 1. BAR GRAPH (COUNT PLOT)

This bar graph shows that crops like Betelwine, Cotton, Greengram, etc. occur in negligible quantities(outliers) than crops like Paddy, Sugarcane, Coconut, Banana. These outliers were adjusted in the training dataset for model performance as well.



## 2. BOX PLOT

The box plot shows that paddy has the largest median cultivated area, while crops like sesame, green gram, and mixed plantations have the smallest. There's also a wide variation in area for paddy and banana, indicating greater inconsistency in land allocation for these crops.



## Key Conclusions:

- Missing values were observed in features like VARIETY and RIVER\_PART. These were removed to improve model training.
- Summary statistics showed high variance in the AREA and YIELD fields.
- Class imbalance was detected, with Paddy being the most frequent crop.
- Boxplots revealed that Paddy has a wider area distribution than other crops.
- Date columns were converted into day-of-year integers for machine learning compatibility (e.g., SOWING\_DAY).

Overall, the data was cleaned, transformed, and encoded, ensuring it was ready for feature correlation and modelling.

## Correlation Analysis and Feature Selection

In this step, how strongly each input feature was related to the target variable (CROP) was analysed and the correlation among numerical features was also studied to reduce redundancy.

### Techniques Used:

- Pearson correlation was used for numerical features (e.g., AREA, YIELD, SOWING\_DAY).

```
corr_with_target = df[numeric_features.columns].corrwith(df['crop_label']).abs()
corr_with_target.sort_values(ascending=False)
```

	0
PADDY_BIN	0.716107
HARVESTING_DAY	0.704523
PLOT_ID	0.467800
UNIQUE_ID	0.464196
SOWING_DAY	0.377696
TRANSPLANTING_DAY	0.358863
YIELD	0.354733
YEAR	0.082636
AREA	0.067980

dtype: float64

We use **Pearson Correlation** to identify which numeric features are most related to the crop type.

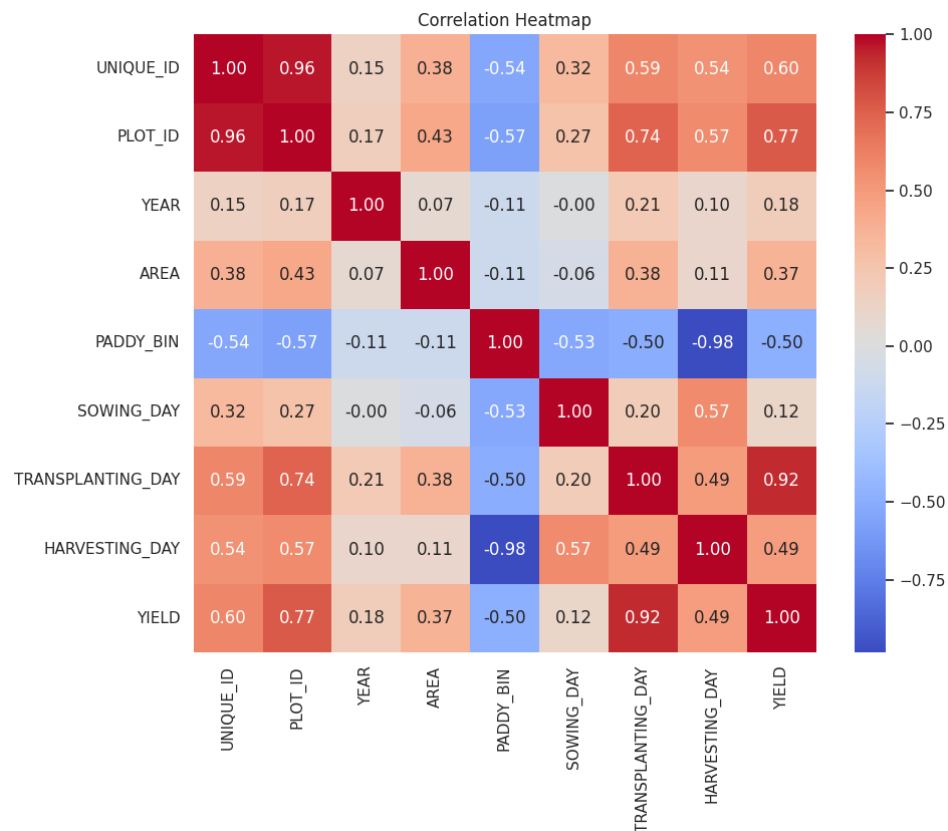
- Label Encoding was applied to convert CROP to numeric format.

```
[ ] from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
df['crop_label'] = le.fit_transform(df['CROP'])
```

Machine learning algorithms require numerical targets. So, we convert each unique crop type into a numeric label. So, now crop\_label can be used as the target variable in our model.

- Correlation heatmaps using `seaborn.heatmap()` were created to visualize the relationships.



## Feature Selection:

We selected features that showed reasonable variance across crop classes and were less prone to missing values. Final selected features were:

- AREA: Plots vary significantly in size; this influences crop type (e.g., Paddy often needs larger land).

- SOWING\_DAY: Captures the seasonality, which affects what crops can be grown.
- YIELD: Performance metric of crop; shows wide class-wise variation.

The chosen features had acceptable correlation with the target variable while maintaining low inter-feature redundancy. Irrelevant or low-variance features (like SPLIT, YEAR, or categorical varieties with missing values) were dropped.

## Model Development


After cleaning and selecting features, KNN classifier model was trained using sklearn. The steps included preprocessing, training, and parameter tuning.

Preprocessing:

- Encoded categorical values using LabelEncoder (target CROP column).

```
[ ] features = ['AREA', 'YEAR', 'YIELD', 'SOWING_DAY', 'PADDY_BIN']  
x = df[features]  
y = df['crop_label']
```

- Scaled numerical features using MinMaxScaler to normalize value ranges.

```
 from sklearn.preprocessing import StandardScaler  
  
scaler = StandardScaler()  
x_scaled = scaler.fit_transform(x)
```



- Used train\_test\_split with 80%-20% split and stratify=y to preserve class distribution.

```
# Keep classes with at least 5 samples
class_counts = df['crop_label'].value_counts()
valid_classes = class_counts[class_counts >= 5].index

# Filter the dataset
df_filtered = df[df['crop_label'].isin(valid_classes)].copy()

# Reset features and target
X = df_filtered[features]
y = df_filtered['crop_label']

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y)
```

## Model Selection:

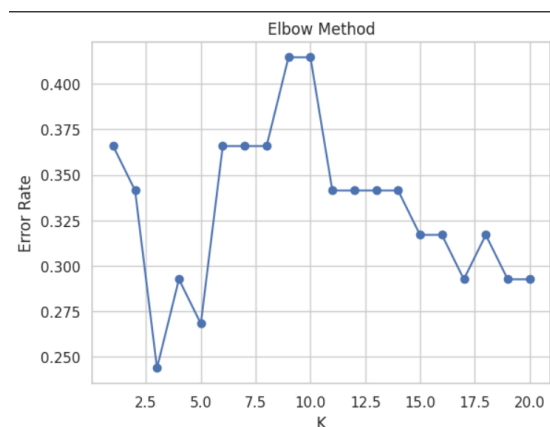
- Applied KNN classifier from sklearn.neighbors.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

errors = []
for k in range(1, 21):
    model = KNeighborsClassifier(n_neighbors=k)
    model.fit(X_train, y_train)
    pred_k = model.predict(X_test)
    errors.append(1 - accuracy_score(y_test, pred_k))

plt.plot(range(1, 21), errors, marker='o')
plt.xlabel("K")
plt.ylabel("Error Rate")
plt.title("Elbow Method")
plt.show()
```

- Used Elbow Method to determine optimal k value.
- Calculated error rate for k = 1 to 20 and plotted the curve.



- Chose k = 3 where error rate was lowest and stable.

## Summary of Steps:

- Defined feature matrix X and label vector y
- Trained the model using KNeighborsClassifier(n\_neighbors=3)
- Predicted on test set using model.predict(X\_test)

## Evaluation Results and Conclusion

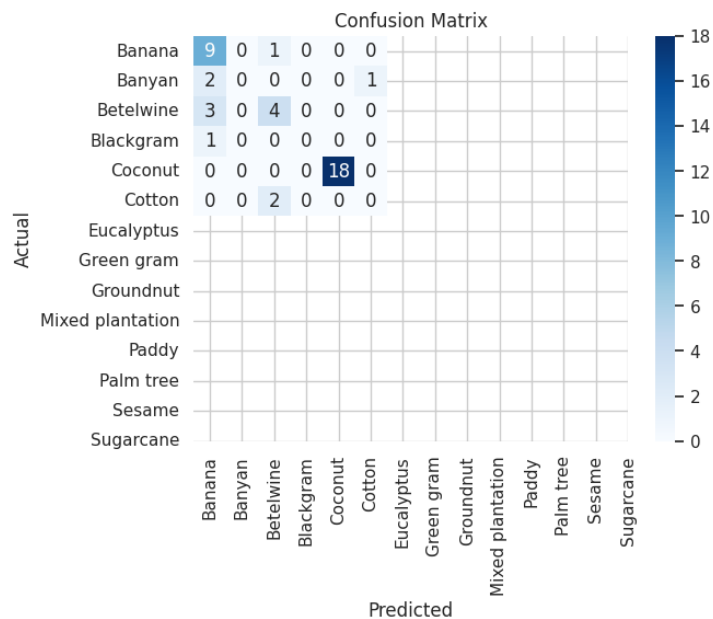
The KNN model was evaluated using standard metrics from sklearn.metrics including metrics like accuracy, precision, recall, F1-score, and confusion matrix.

### Evaluation Metrics:

- Accuracy: 76%, indicating good overall performance
- Precision & Recall: Very high for dominant crops like Paddy (1.00) and Banana (0.90 recall)
- F1 Score: Overall average 0.71, but macro average 0.38 due to class imbalance

	precision	recall	f1-score	support
Banana	0.60	0.90	0.72	10
Banyan	0.00	0.00	0.00	3
Coconut	0.57	0.57	0.57	7
Groundnut	0.00	0.00	0.00	1
Paddy	1.00	1.00	1.00	18
Sugarcane	0.00	0.00	0.00	2
accuracy			0.76	41
macro avg	0.36	0.41	0.38	41
weighted avg	0.68	0.76	0.71	41

- Confusion Matrix: Helped to visually actual vs. predicted classes



#### Inference:

- Paddy and Coconut were predicted most accurately.
- Underrepresented classes like Sugarcane, Groundnut were often misclassified.
- Model struggled with crops having <5 examples.

#### Final Thoughts and Conclusion:

The KNN model performed well for dominant crops, particularly Paddy and Coconut, demonstrating the algorithm's ability to classify based on similarities in sowing day, area, and yield. However, it faced limitations with rare crops due to data imbalance. The use of distance-based similarity makes KNN more effective when data is uniformly distributed across all classes, which wasn't the case in this dataset.

From the results, we can infer that crops like Paddy, Banana, and Coconut are well-predicted and thus should be more closely monitored and supported in agricultural decision systems. Paddy showed optimal prediction performance and had high yield values, suggesting it remains a strong choice for large field areas and early sowing patterns.

### Agronomic Insights:

- Crops like Paddy and Banana should be prioritized in terms of government support and land allocation due to high predictive performance and yield.
- Ideal sowing window lies between day 120 and 170 (roughly April to June), especially for Paddy, which aligns well with pre-monsoon preparation.
- Yield optimization is highly correlated with sowing period and plot size; larger plots with early sowing days consistently showed better outcomes.

### Scope for Future Work:

- Data Collection: Increase representation of minority crops like Groundnut, Sugarcane, and Blackgram to improve model generalization.
- Additional Features: Incorporate weather data (temperature, rainfall), soil type, and irrigation method to enhance predictive accuracy.
- Model Improvement: Compare KNN with more robust classifiers.
- Field Advisory Tool: Deploy this model into an interactive dashboard that recommends crops based on current land size, sowing date, and yield goals.

In conclusion, this project not only demonstrates machine learning's value in agriculture and highlights how data-driven insights can shape sowing schedules, crop planning, and policy recommendations. With broader data and refined modelling, such tools can help revolutionize agricultural productivity and resilience in India.