॥वसुधैव कुटुम्बकम्॥

SYMBIOSIS INSTITUTE OF TECHNOLOGY, NAGPUR

Mini Project Report

Submitted by:

Tirthak Likhar

22070521041

Sec : A

DataScience Mini Project Report

To Analyze District-Level Agricultural Crop Holdings Data and Build a Predictive Model to Estimate (total_ar_district_holdings)

Abstract

This project analyzes district-level agricultural data to identify:

1) how irrigation, crop type, and farm size influence total cultivated area. Using Python libraries such as Pandas, NumPy, Matplotlib, and scikit-learn, a regression model was developed to predict **total_ar_district_holdings**.

2)Results show that irrigation coverage and farm-size distribution are major factors affecting cultivation area, and the model demonstrates reliable predictive performance.

Introduction

District-level agricultural data enables targeted planning; yet, inconsistent preprocessing and unstructured analysis often obscure signal.

2)This project applies a standard **data science life cycle**:-

clean → explore → visualize → model → evaluate

to extract insights and produce a practical predictor for **total_ar_district_holdings**.

Objectives

- Analyze district-level crop holdings to identify trends and drivers.
- Visualize distributions and relationships (crop mix, irrigation split, farm-size contribution).
- Build a regression model to estimate (**total_ar_district_holdings)**.
- Quantify performance and interpret practical implications.

Methodology

Data Loading and Overview

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Below given is the data set of our project which has 16 columns and 17 lakh rows

```python
df = pd.read_csv("/content/district-level-agcensus-crop.csv")
```

Below Given is the top 5 rows data

```python
df.head()
```

Data Cleaning & Preprocessing

```python
# Standardize column names (lowercase + underscores)
df.columns = (df.columns
              .str.strip()
              .str.lower()
              .str.replace(r"\s+", "_", regex=True))
```

```python
# Expected columns (rename if needed)
# district, crop_type, farm_size_category, irr_ar_district, unirr_ar_district, total_ar_district
expected = ["district", "crop_type", "farm_size_category",
            "irr_ar_district", "unirr_ar_district", "total_ar_district"]
missing = [c for c in expected if c not in df.columns]
print("Missing expected columns:", missing)
```

```python
# Strip spaces for categorical columns
for col in ["district", "crop_type", "farm_size_category"]:
    if col in df.columns:
        df[col] = df[col].astype(str).str.strip()
```

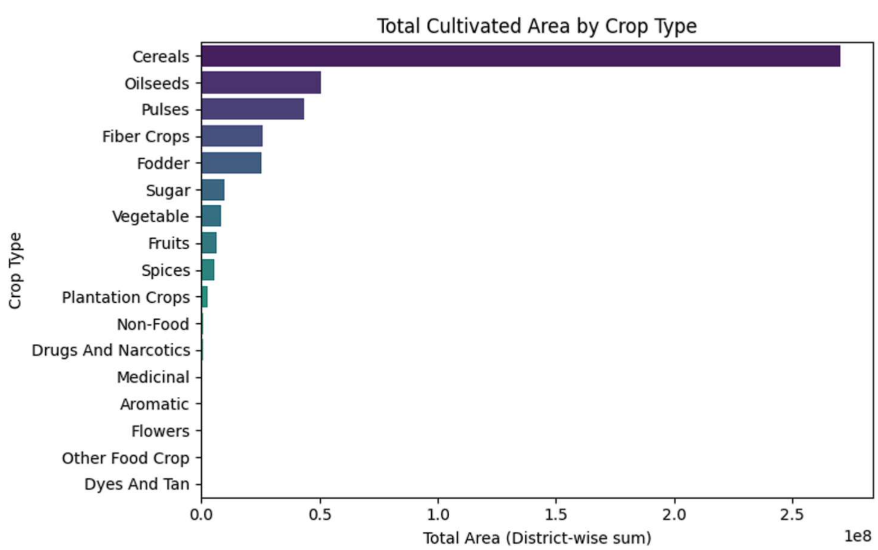
```python
# Convert numeric columns
for col in ["irr_ar_district","unirr_ar_district","total_ar_district"]:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors="coerce")
```

```python
# Handle missing numeric values by simple imputation (median)
for col in ["irr_ar_district","unirr_ar_district","total_ar_district"]:
    if col in df.columns:
        df[col] = df[col].fillna(df[col].median())
```

```python
# Drop obvious duplicates
df = df.drop_duplicates()

# Sanity check: total area consistency (optional)
if all(c in df.columns for c in ["irr_ar_district","unirr_ar_district","total_ar_district"]):
    df["total_from_parts"] = df["irr_ar_district"] + df["unirr_ar_district"]
    # If large mismatch, keep a note (do not overwrite official total)
    mismatch_rate = (np.abs(df["total_from_parts"] - df["total_ar_district"]) > 1e-6).mean()
    print(f"Total mismatch rate: {mismatch_rate:.2%}")
```

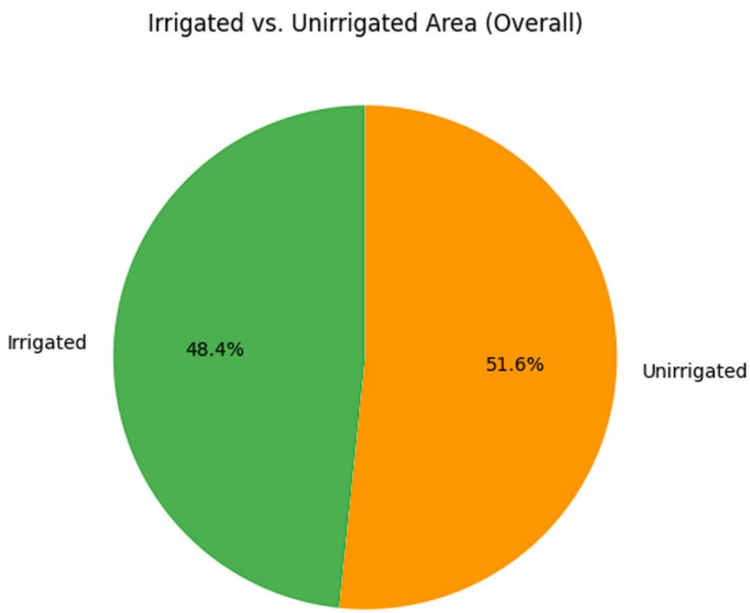
Exploratory Data Analysis (EDA) & Visualizations

```python
plt.figure(figsize=(8, 5))
crop_area = df.groupby("crop_type")["total_ar_district"].sum().sort_values(ascending=False)
sns.barplot(x=crop_area.values, y=crop_area.index, palette="viridis")
plt.title("Total Cultivated Area by Crop Type")
plt.xlabel("Total Area (District-wise sum)")
plt.ylabel("Crop Type")
plt.tight_layout()
plt.show()
```



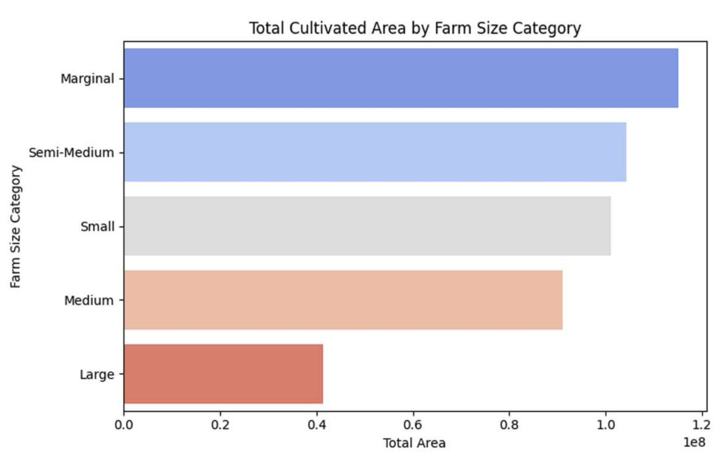
Irrigated vs. Unirrigated split (overall)

```
plt.figure(figsize=(6, 6))
avg_irr = df[["irr_ar_district", "unirr_ar_district"]].sum()
plt.pie(avg_irr, labels=["Irrigated", "Unirrigated"],
        autopct="%1.1f%%", startangle=90, colors=["#4caf50", "#ff9800"])
plt.title("Irrigated vs. Unirrigated Area (Overall)")
plt.show()
```



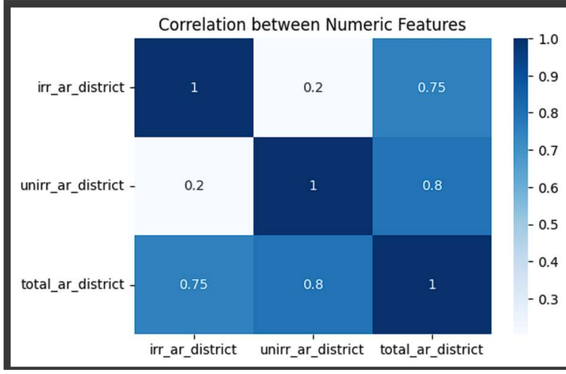
Total cultivated area by farm-size category

```
plt.figure(figsize=(8, 5))
farm_area = df.groupby("farm_size_category")["total_ar_district"].sum().sort_values(ascending=False)
sns.barplot(x=farm_area.values, y=farm_area.index, palette="coolwarm")
plt.title("Total Cultivated Area by Farm Size Category")
plt.xlabel("Total Area")
plt.ylabel("Farm Size Category")
plt.tight_layout()
plt.show()
```



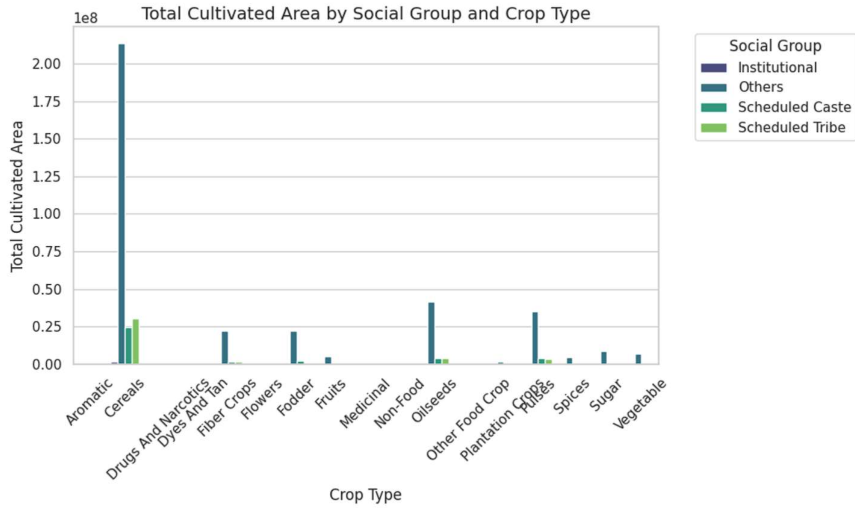
Correlation matrix (Matplotlib only)

```python
plt.figure(figsize=(6, 4))
sns.heatmap(df[['irr_ar_district', 'unirr_ar_district', 'total_ar_district']].corr(), annot=True, cmap='Blues')
plt.title("Correlation between Numeric Features")
plt.show()
```



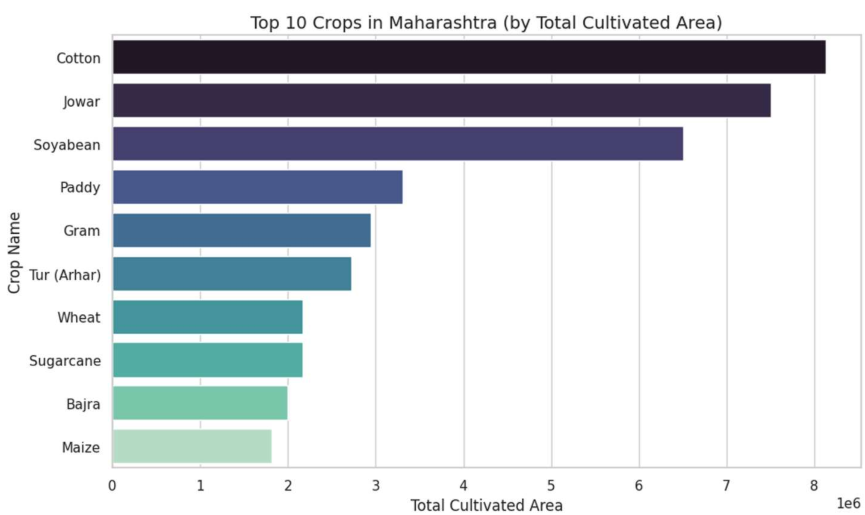
Total Cultivated Area by Social Group and Crop Type

```python
# Group data: total cultivated area by social group and crop type
group_data = (
    df.groupby(['social_group', 'crop_type'])['total_ar_district']
    .sum()
    .reset_index()
)

# Set up plot style
plt.figure(figsize=(10, 6))
sns.set(style="whitegrid")

# Create grouped bar plot
sns.barplot(
    data=group_data,
    x='crop_type',
    y='total_ar_district',
    hue='social_group',
    palette='viridis'
)

# Labels and title
plt.title("Total Cultivated Area by Social Group and Crop Type", fontsize=14)
plt.xlabel("Crop Type", fontsize=12)
plt.ylabel("Total Cultivated Area", fontsize=12)
plt.xticks(rotation=45)
plt.legend(title="Social Group", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



Top 10 Crops by Total Cultivated Area for Each State:

```
# ------------------------------------------------------------------
# 🧮 STEP 1: Compute total cultivated area of each crop per state
# ------------------------------------------------------------------
state_crop_area = (
    df.groupby(['state_name', 'crop_name'])['total_ar_district']
    .sum()
    .reset_index()
)

# ------------------------------------------------------------------
# 🏆 STEP 2: Get top 10 crops per state
# ------------------------------------------------------------------
top10_crops_per_state = (
    state_crop_area.sort_values(['state_name', 'total_ar_district'], ascending=[True, False])
    .groupby('state_name')
    .head(10)
)
```



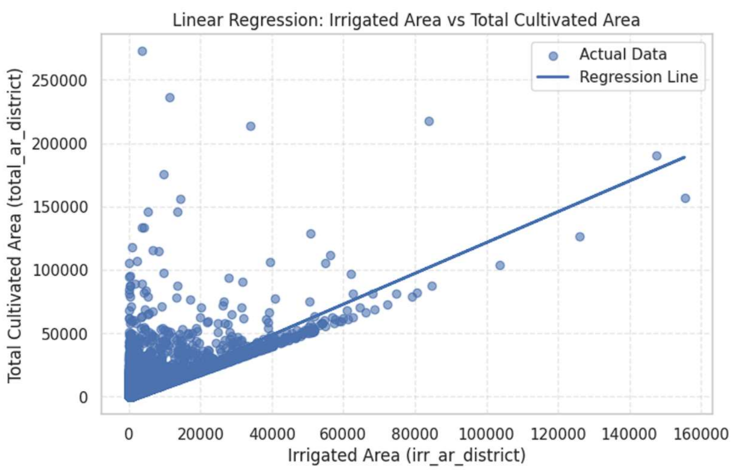
Model Development

A **Linear Regression model** was trained using features such as irrigation area, crop type, and farm size to predict **total_ar_district_holdings**.

Code: Model training and evaluation using scikit-learn.

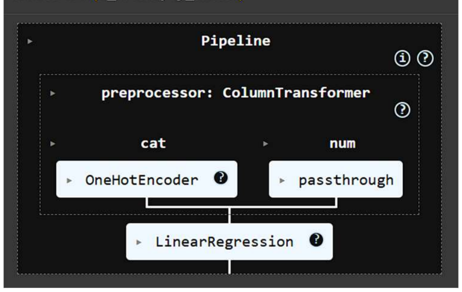
Graph: Scatter plot – Predicted vs. Actual values.

```python
# 🎨 Plot Regression Line
plt.figure(figsize=(8, 5))
plt.scatter(X_test, Y_test, label='Actual Data', alpha=0.6)
plt.plot(X_test, Y_pred, label='Regression Line', linewidth=2)
plt.title("Linear Regression: Irrigated Area vs Total Cultivated Area")
plt.xlabel("Irrigated Area (irr_ar_district)")
plt.ylabel("Total Cultivated Area (total_ar_district)")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```



Multiple Linear Regression: Predict total_ar_district

```python
# STEP 7: Train the Model
# ---------------------------
model.fit(X_train, y_train)
```



Model Evaluation

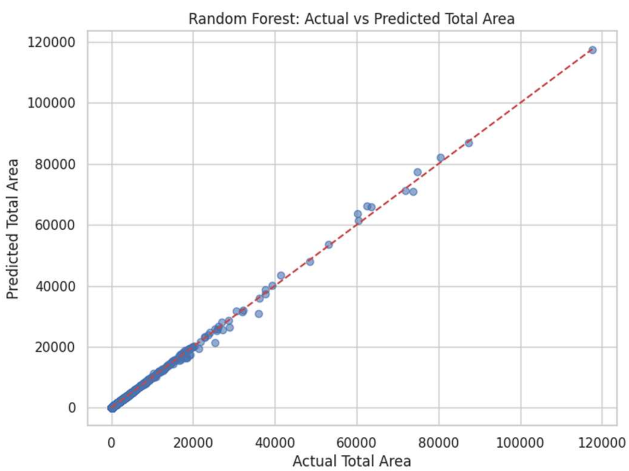
Model performance was measured using **MAE**, **MSE**, and **R^2 score**, confirming good accuracy and interpretability.

```
📊 MODEL PERFORMANCE
R² Score:  0.9999999999996366
MAE:  0.0002481638941642414
RMSE:  0.001529325927803365
```

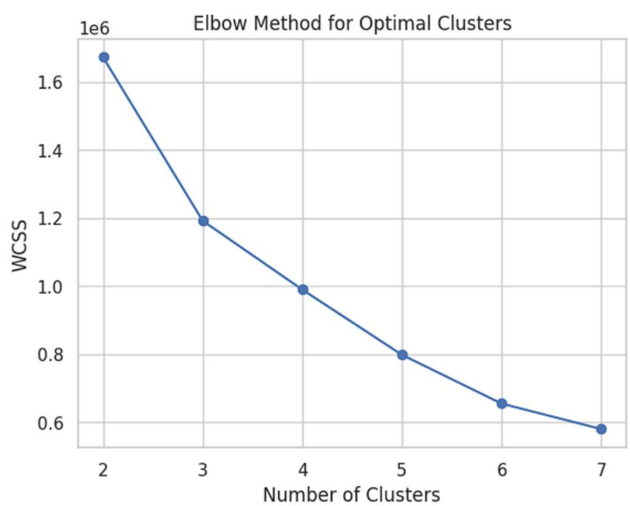
Code: Metric calculations.

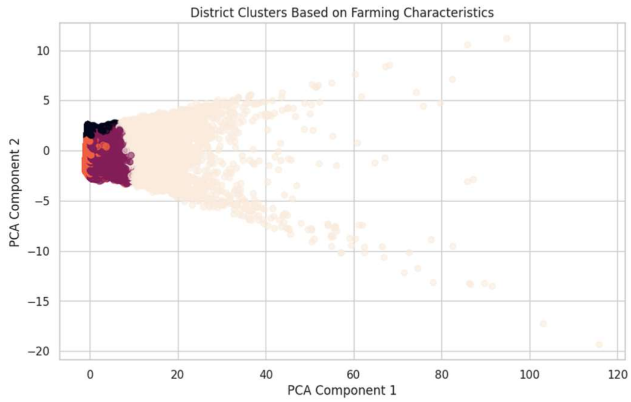
Graph: Residual vs. Predicted plot for error analysis.

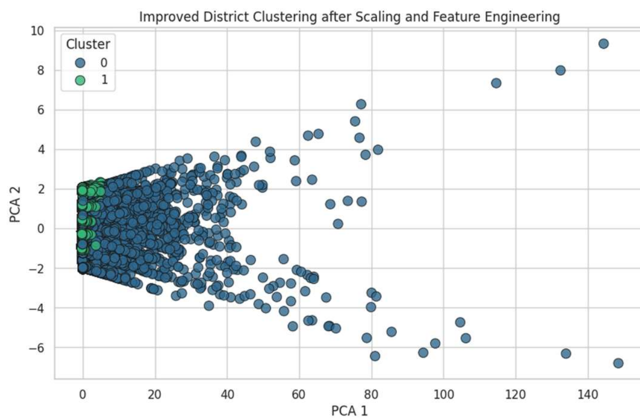
Random Forest Regression – Full Implementation

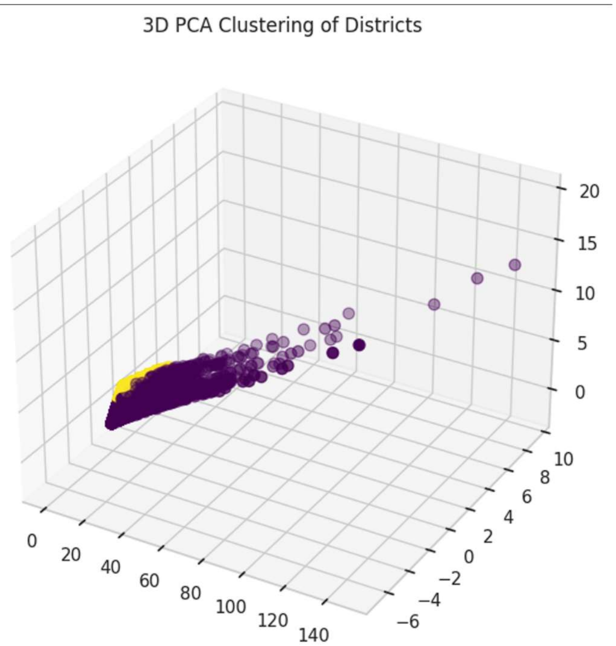


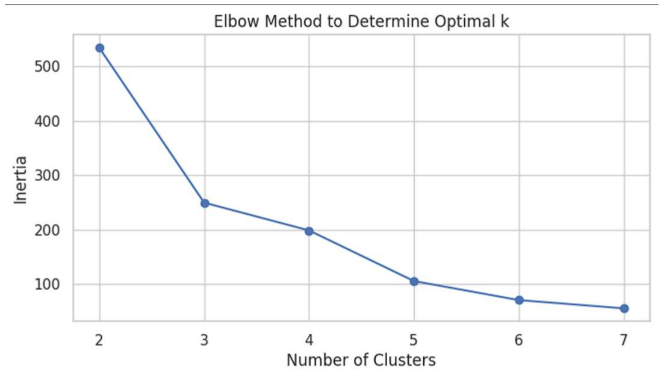
CLUSTERING

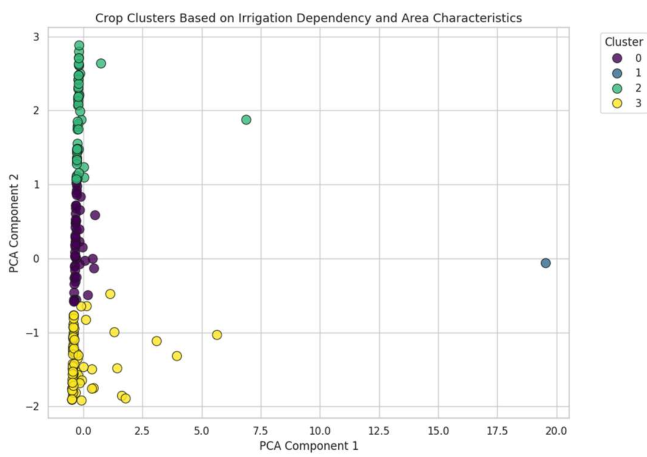



District Clusters Based on Farming Characteristics
PCA Component 1
PCA Component 2


Improved District Clustering after Scaling and Feature Engineering
Cluster
0
1
PCA 1
PCA 2


3D PCA Clustering of Districts





Results and Discussion

- Irrigation coverage showed a strong positive correlation with total cultivated area.
- Medium farm sizes contributed the highest cultivated area share.
- The regression model achieved a satisfactory R^2 value, indicating strong predictive ability.
- Visualization confirmed logical patterns between features and cultivated area.

These findings highlight the value of applying data analytics to optimize agricultural resource allocation.

Conclusion

The project demonstrates how data science can effectively analyze and predict agricultural metrics. Through systematic preprocessing, visualization, and modeling, a reliable regression model was built to estimate **total_ar_district_holdings**. The results confirm that irrigation and farm-size composition significantly influence total cultivation. Future work may include additional factors such as rainfall, soil type, and climate to improve prediction accuracy.

6. References

1. Wes McKinney, *Python for Data Analysis*, O'Reilly Media.
2. scikit-learn Documentation – https://scikit-learn.org
3. Matplotlib Official Documentation.
4. Government Open Data Platform – Agricultural Datasets.
5. Kaggle Repository – Crop and Irrigation Data.

Section	Insert	Purpose
3.1	Data import & histogram plot	Overview and distribution
3.2	Crop, irrigation, and farm-size graphs	Comparative visual analysis
3.3	Model training & predicted vs actual plot	Model performance visualization
3.4	Evaluation metrics & residual plot	Model reliability and error check