RAJALAKSHMI ENGINEERING COLLEGE

AI19P71
Data Visualization Using Python
Project Title: Agricultural Crop Recommendation system

Team Members
221501075-Medhini K
221501503-Dheeraj Kumar S
221501506-Hemanth Kumar J

Problem Statement

- Choosing the right crop for a given piece of land is a critical decision that directly impacts agricultural yield and farmer income. Traditional methods often rely on personal experience or local practices, which may not account for recent climate shifts, soil nutrient depletion, or inconsistent rainfall. This project proposes a Machine Learning–based Crop Recommendation System that analyses key soil parameters (Nitrogen, Phosphorus, Potassium, pH) along with environmental factors like temperature, humidity, and rainfall to suggest the most suitable crop. By providing accurate, data-driven recommendations, the system aims to support farmers in optimizing yields, conserving resources, and promoting sustainable farming practices.

Dataset Source and Structure

Dataset Source	
No of Features	8
No of Records	2200

Dataset Feature Description

Feature	Description
N	Nitrogen content in soil (ppm)
P	Phosphorus content in soil (ppm)
K	Potassium content in soil (ppm)
Temperature	Temperature in °C

Dataset Feature Description

Feature	Description
Humidity	Relative humidity (%)
ph	Soil pH value
Rainfall	Rainfall in mm
Label	Recommended crop name

Data Acquisition and Cleaning

CODE

```python
# 1. Dataset information
print(f"\nDataset Shape: {df.shape}")
print(f"Columns: {list(df.columns)}")
print(f"\nData Types:\n{df.dtypes}")

# 2. Handle Missing Values
print(f"\nMissing Values Analysis:")
missing_values = df.isnull().sum()
print(missing_values)

if missing_values.sum() > 0:
    print("\nHandling missing values...")
    rows_before = len(df)
    df = df.dropna()
    rows_after = len(df)
    print(f"Dropped {rows_before - rows_after} rows with missing values")
    print(f"Dataset reduced from {rows_before} to {rows_after} rows")
else:
    print("No missing values found!")
```

OUTPUT

```
Dataset Shape: (2200, 8)
Columns: ['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall', 'label']

Data Types:
N                int64
P                int64
K                int64
temperature    float64
humidity       float64
ph             float64
rainfall       float64
label           object
dtype: object

Missing Values Analysis:
N              0
P              0
K              0
temperature    0
humidity       0
ph             0
rainfall       0
label          0
dtype: int64
No missing values found!
```

Data Acquisition and Cleaning

CODE

```python
# 3. Check for duplicates
duplicates = df.duplicated().sum()
print(f"\nDuplicate rows: {duplicates}")
if duplicates > 0:
    df.drop_duplicates(inplace=True)
    print(f"Removed {duplicates} duplicate rows")
else:
    print("No duplicate rows found!")

# 4. Data Normalization
print("\nApplying data normalization...")
numerical_cols = ['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall']

df_normalized = df.copy()
scaler = MinMaxScaler()
df_normalized[numerical_cols] = scaler.fit_transform(df[numerical_cols])
print("\nSample of normalized numerical columns:\n", df_normalized[numerical_cols].head())
```

OUTPUT

```
Duplicate rows: 0
No duplicate rows found!

Applying data normalization...

Sample of normalized numerical columns:
          N         P      K  temperature  humidity        ph  rainfall
0  0.642857  0.264286  0.190     0.345886  0.790267  0.466264  0.656458
1  0.607143  0.378571  0.180     0.371445  0.770633  0.549480  0.741675
2  0.428571  0.357143  0.195     0.406854  0.793977  0.674219  0.875710
3  0.528571  0.214286  0.175     0.506901  0.768751  0.540508  0.799905
4  0.557143  0.264286  0.185     0.324378  0.785626  0.641291  0.871231
```

Data Preprocessing

CODE AND OUTPUT

```
# Climate zone classification
def classify_climate(temp, humidity):
    if temp > 25 and humidity > 80:
        return 'Tropical'
    elif temp > 20 and humidity < 60:
        return 'Arid'
    elif temp < 15:
        return 'Temperate'
    else:
        return 'Subtropical'

df['climate_zone'] = df.apply(lambda x: classify_climate(x['temperature'], x['humidity']), axis=1)
df_normalized['climate_zone'] = df['climate_zone']
print("\nSample Climate Zone assignments:\n", df[['temperature', 'humidity', 'climate_zone']].head())
```

```
Sample Climate Zone assignments:
    temperature   humidity climate_zone
0    20.879744  82.002744  Subtropical
1    21.770462  80.319644  Subtropical
2    23.004459  82.320763  Subtropical
3    26.491096  80.158363     Tropical
4    20.130175  81.604873  Subtropical
```

EXPLANATION

Categorizes each record into one of four climate types Tropical, Arid, Temperate, or Subtropical based on temperature and humidity thresholds.

Data Preprocessing

CODE AND OUTPUT

```
# Soil fertility index
df['soil_fertility_index'] = (df['N'] + df['P'] + df['K']) / 3
df_normalized['soil_fertility_index'] = (df_normalized['N'] + df_normalized['P'] + df_normalized['K']) / 3
print("\nSample Soil Fertility Index values:\n", df[['N', 'P', 'K', 'soil_fertility_index']].head())

# pH category classification
def classify_ph(ph):
    if ph < 6.0:
        return 'Acidic'
    elif ph > 7.5:
        return 'Alkaline'
    else:
        return 'Neutral'

df['ph_category'] = df['ph'].apply(classify_ph)
df_normalized['ph_category'] = df['ph_category']
print("\nSample pH Category assignments:\n", df[['ph', 'ph_category']].head())
```

```
Sample Soil Fertility Index values:
    N   P   K  soil_fertility_index
0  90  42  43             58.333333
1  85  58  41             61.333333
2  60  55  44             53.000000
3  74  35  40             49.666667
4  78  42  42             54.000000
```

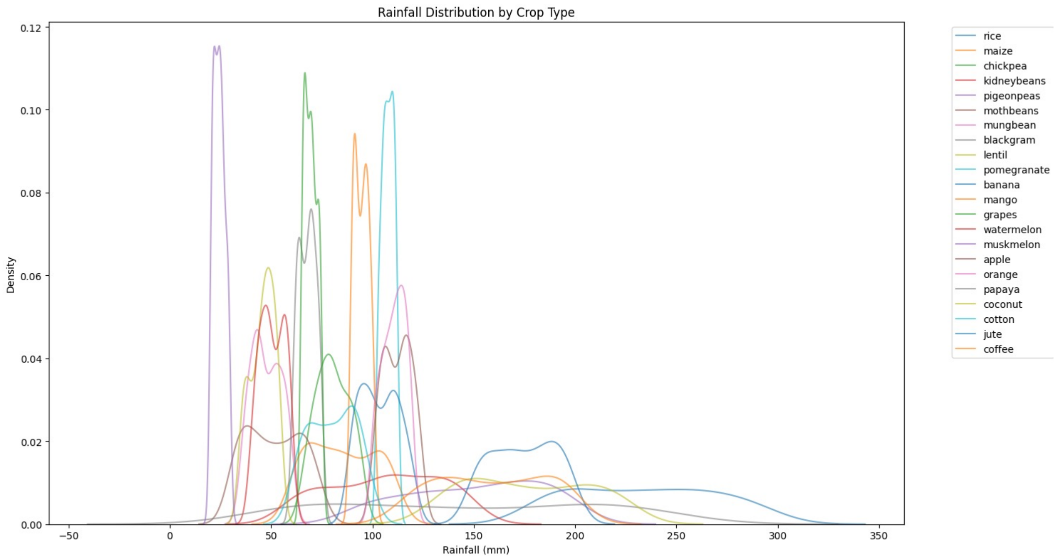
```
Sample pH Category assignments:
         ph ph_category
0  6.502985     Neutral
1  7.038096     Neutral
2  7.840207    Alkaline
3  6.980401     Neutral
4  7.628473    Alkaline
```

EXPLANATION

Calculates the average of N, P, and K values to give a single score representing overall soil nutrient richness.

Classifies soil pH into Acidic (< 6.0), Neutral (6.0–7.5), or Alkaline (> 7.5).

Rainfall Distribution by Crop Type (KDE Plot)



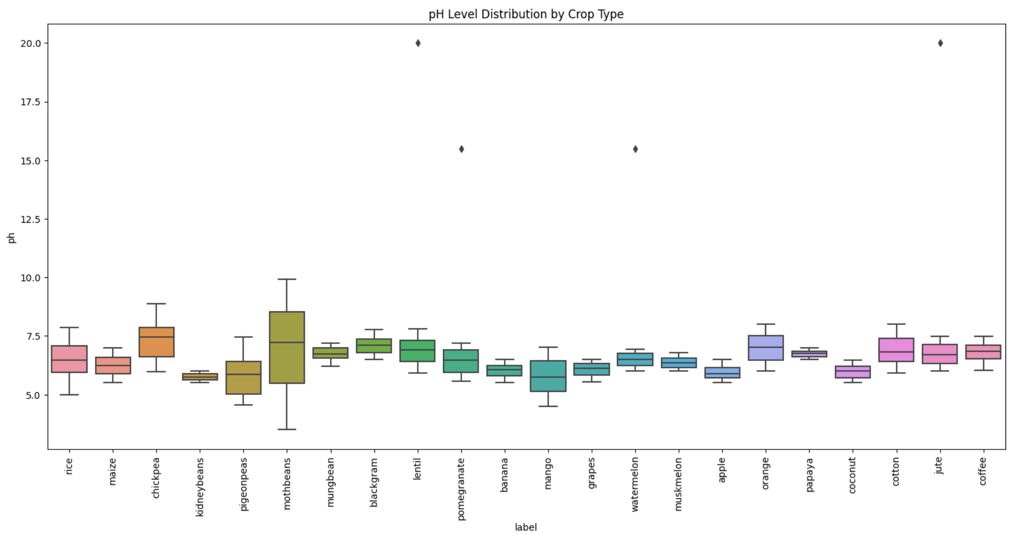
Insight Description: Shows rainfall requirement density for different crops.

Observation: Most crops cluster between 50–150 mm rainfall, while crops like rice & coconut need significantly higher rainfall; chickpea & lentil need low rainfall.

Implication: Rainfall requirements vary widely across crops; choosing a mismatched crop for local rainfall leads to poor yield.

Recommendation: Farmers in low-rainfall regions should prefer crops like chickpea, lentil & kidney beans, while high-rainfall areas should plant rice & coconut.

pH Level Distribution Box Plot



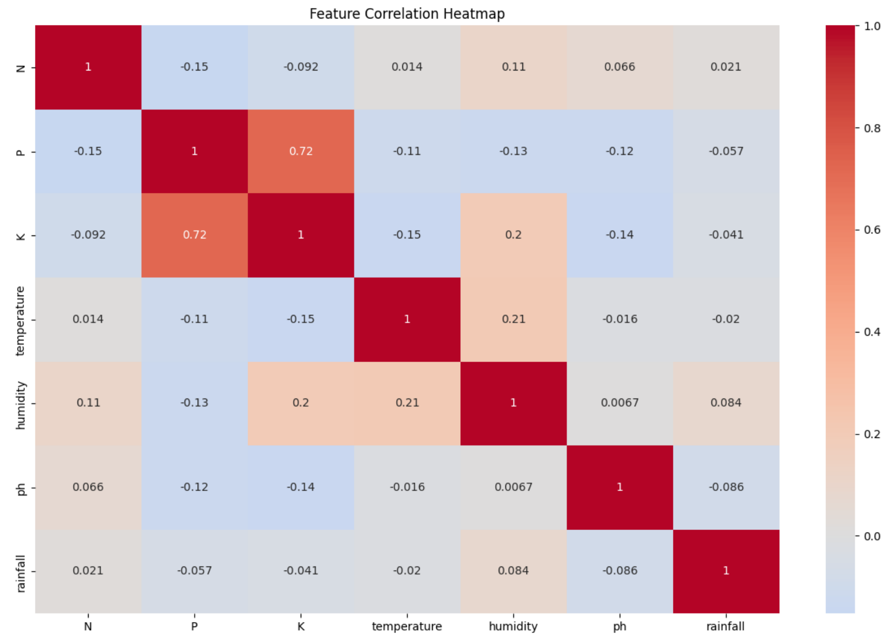
Insight Description: Shows soil pH required by each crop.

Observation: Most crops prefer pH 5.5–7.5, with few requiring extremes; outliers exist for crops like mothbeans & coffee.

Implication: pH is a crucial factor — incorrect soil pH can restrict nutrient uptake and reduce yield.

Recommendation: Test soil pH before planting & use lime or sulphur amendments to balance pH for crop suitability.

Feature Correlation Heatmap



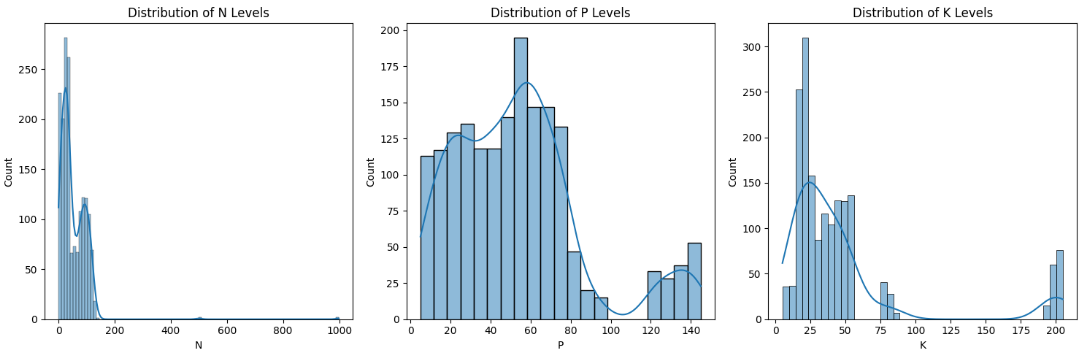
Insight Description: Correlation among features like N, P, K, temperature, rainfall, pH, humidity.

Observation: Nitrogen, pH & rainfall show very low correlation with each other; P & K show a strong positive correlation (0.72).

Implication: Features independently influence crop prediction — ML models must consider interaction effects, not linear assumptions.

Recommendation: Use tree-based algorithms (Random Forest, XGBoost) to capture non-linear relationships.

Distribution of N, P, K Nutrients



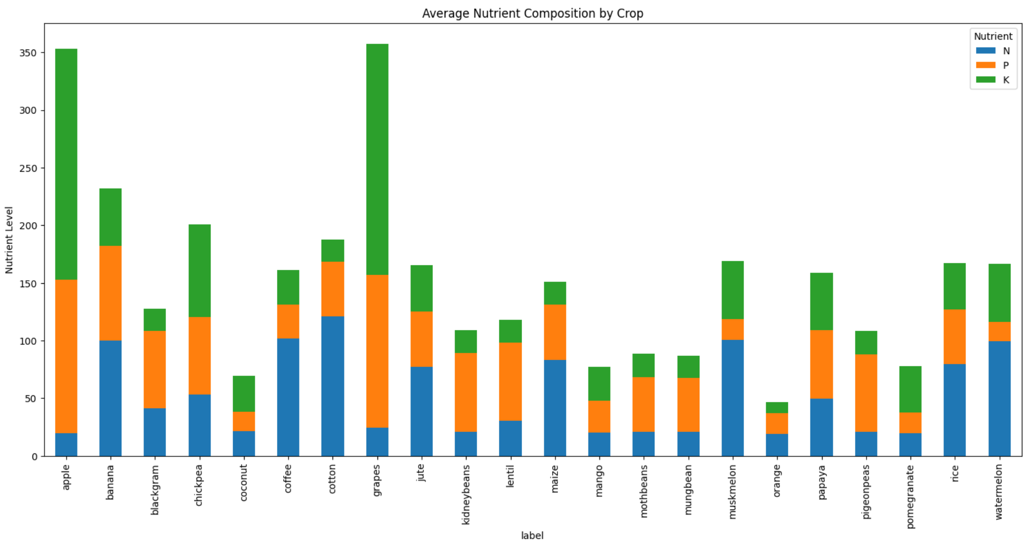
Insight Description: Shows frequency distribution of Nitrogen, Phosphorus, and Potassium values.

Observation: Nitrogen values are mostly low, P and K show wider ranges; some extreme outliers exist.

Implication: Soil nutrient concentrations vary widely — fertilizers must be applied smartly, not uniformly.

Recommendation: Implement soil testing & customized fertilization plans instead of blanket fertilizer use.

Average Nutrient Composition by Crop (Stacked Bar)



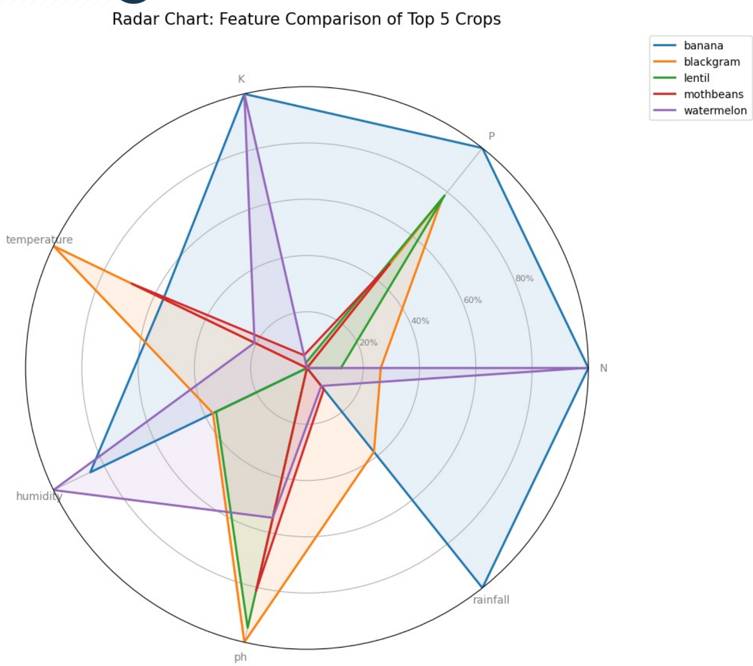
Insight Description: Average N-P-K requirement across crops.

Observation: Grapes, apple & banana show very high nutrient needs, whereas crops like mungbean & mothbeans are low input.

Implication: High-input crops require strong soil management; low-input crops suit nutrient-deficient soil regions.

Recommendation: Recommend legumes (mungbean, chickpea) in nutrient-poor soils & reserved high-nutrient crops for well-fertile land.

Radar Chart: Feature Comparison of Top 5 Crops



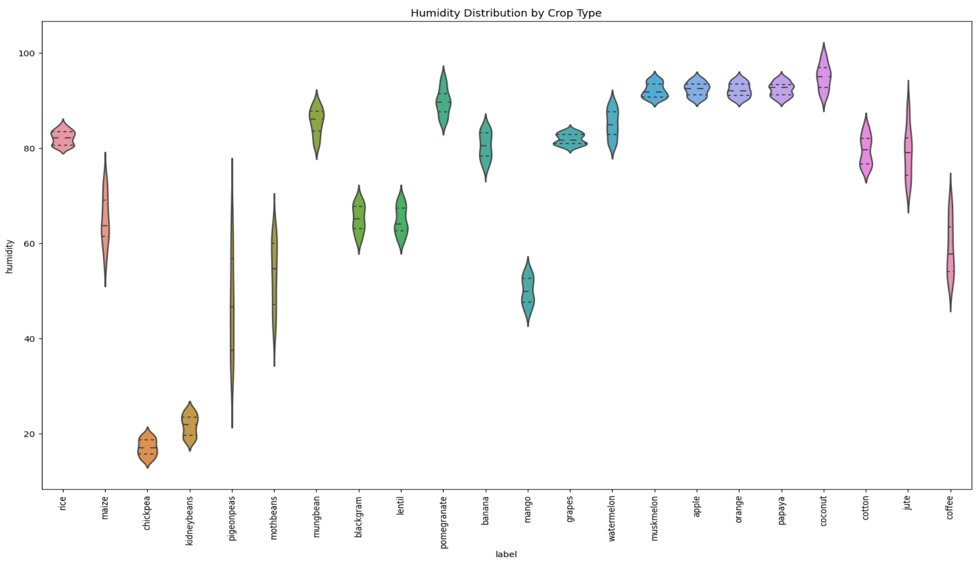
Insight Description: Compares Nitrogen, Phosphorus, Potassium, pH, humidity, temperature & rainfall needs for selected crops.

Observation: Banana shows very high nutrient and water requirements, while lentil & mothbeans require lower nutrients and moderate climate.

Implication: Crops differ drastically in environmental & soil nutrient demand, so nutrient planning matters.

Recommendation: Use radar profiles to match crops to soil fertility — plant high-input crops (banana) only where nutrients and water are adequate.

Humidity Distribution by Crop (Violin Plot)



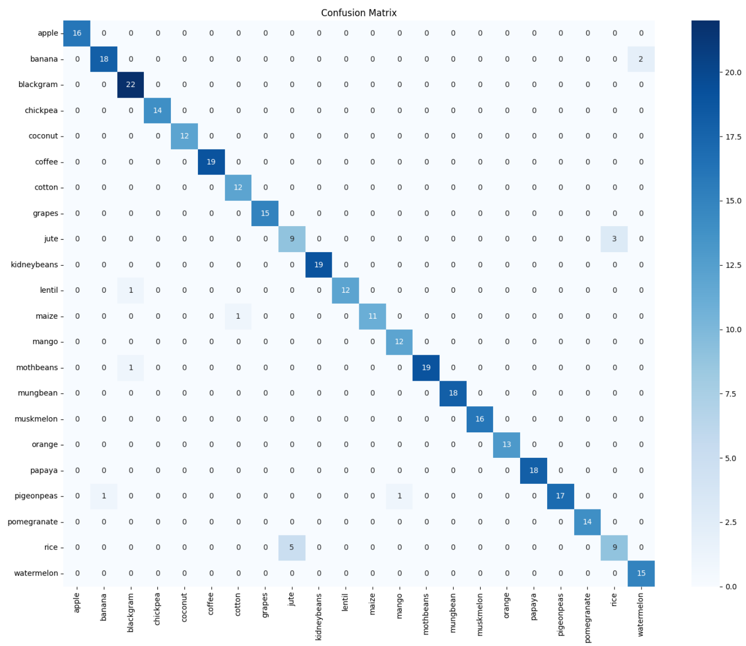
Insight Description: Humidity requirements per crop.

Observation: Rice, coconut, grapes & banana require high humidity; chickpea & maize thrive in dry/low humidity areas.

Implication: Humidity affects crop evapotranspiration & disease patterns — planting mismatch increases risk.

Recommendation: Wet-humidity climates should grow banana, rice, coconut, while dry climate should choose maize, chickpea.

Confusion Matrix for crops



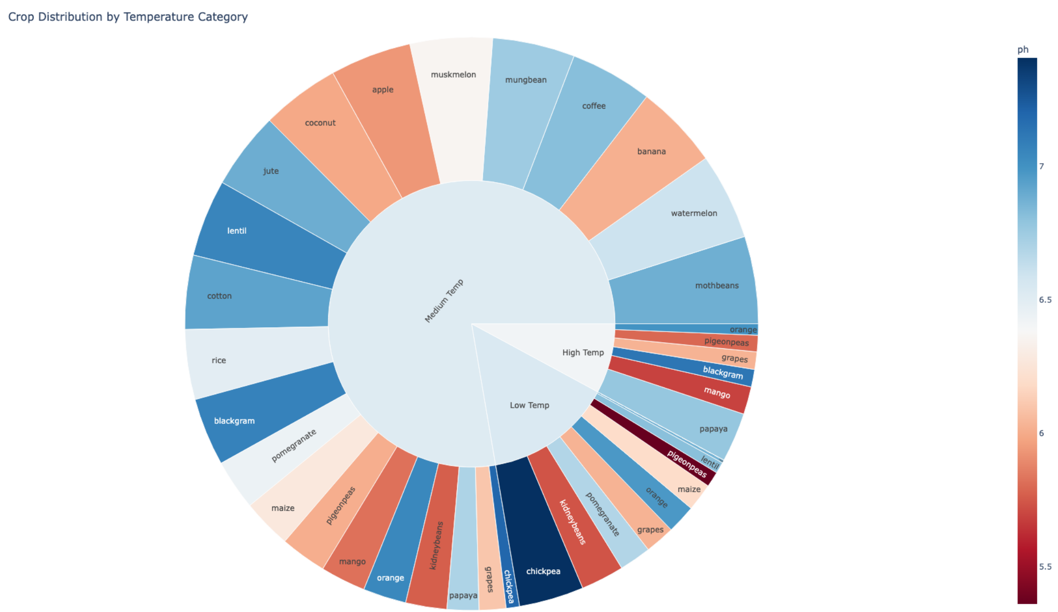
Insight Description: Illustrates the model's prediction accuracy across multiple crop categories.

Observation: Most crops are correctly classified, with minor confusion among those with similar growing conditions.

Implication: The ML model demonstrates strong performance, with few misclassifications affecting related crops.

Recommendation: Incorporate more environmental or soil features and perform field validation to enhance prediction precision.

Crop Distribution by Temperature Category (Sunburst Chart)



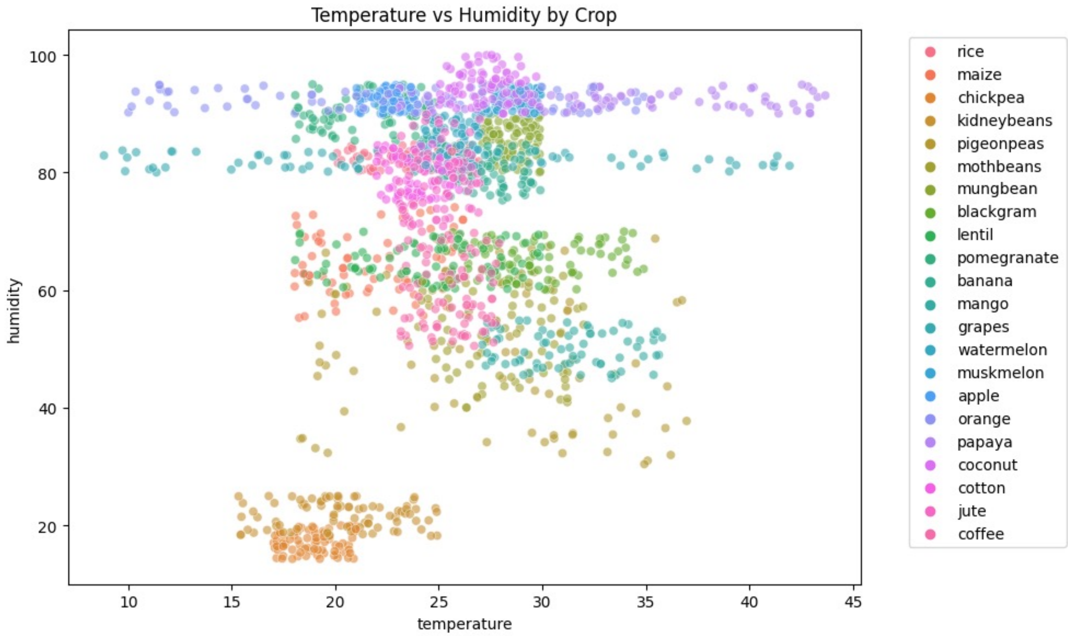
Insight Description: Distribution of crops across Low, Medium, High temperature ranges.

Observation: Most crops fall under medium temperature category; only a few tolerate extreme hot or cold.

Implication: Temperature is a major determinant of crop success — most crops are temperature-sensitive.

Recommendation: Farmers in extreme climates should choose heat-tolerant crops (mango, papaya) or cool-region crops (apple, grape).

Temperature vs Humidity by Crop



Insight Description: Visualizes how various crops are distributed across different temperature and humidity levels.

Observation: Each crop type occupies a distinct climate zone, showing clear environmental preferences.

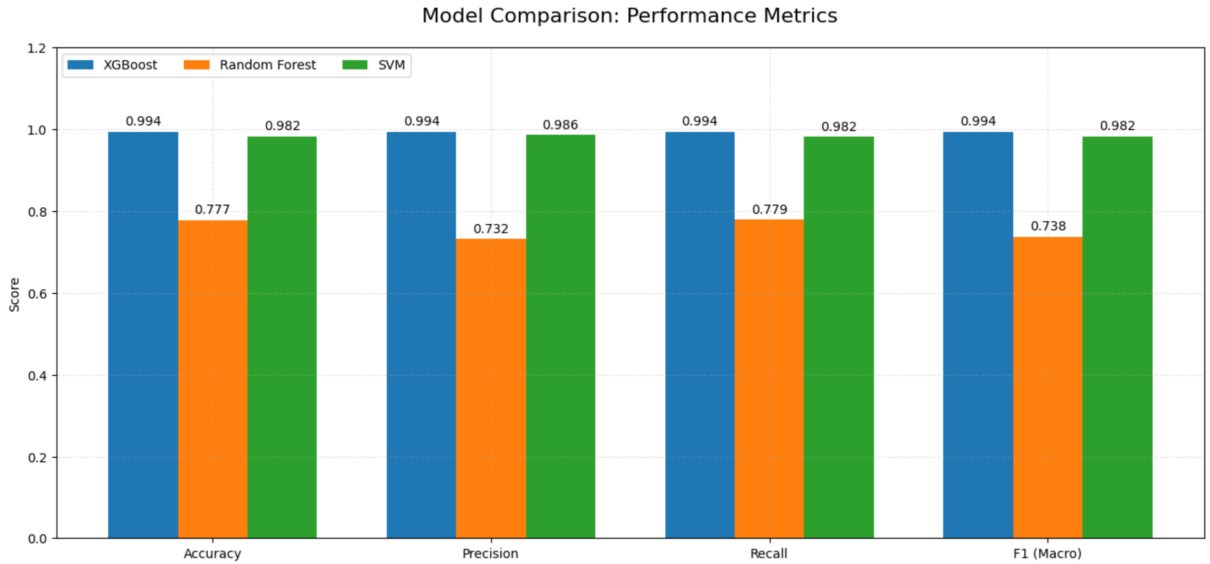
Implication: Temperature and humidity are key factors determining crop suitability and growth potential.

Recommendation: Farmers should align crop choices with local climate profiles to optimize yield and sustainability.

Machine Learning Model

- Chosen Model: **XGBoost Classifier**
- Reason: Performs well on small to medium datasets, handles non-linear relationships, and provides high interpretability.
- Dataset Split: 80% training, 20% testing

Model Evaluation

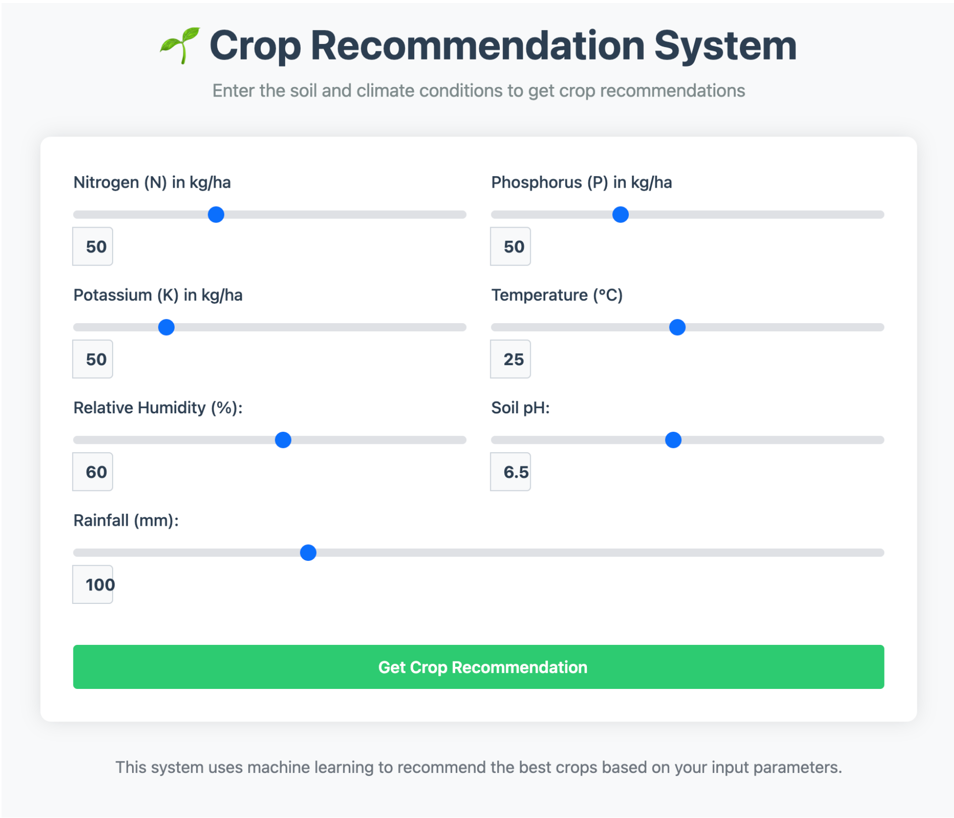


	Model	Accuracy	Precision	Recall	F1 (Macro)
XGBoost	XGBoost	0.994	0.994	0.994	0.994
Random Forest	Random Forest	0.777	0.732	0.779	0.738
SVM	SVM	0.982	0.986	0.982	0.982

Summary of the Findings

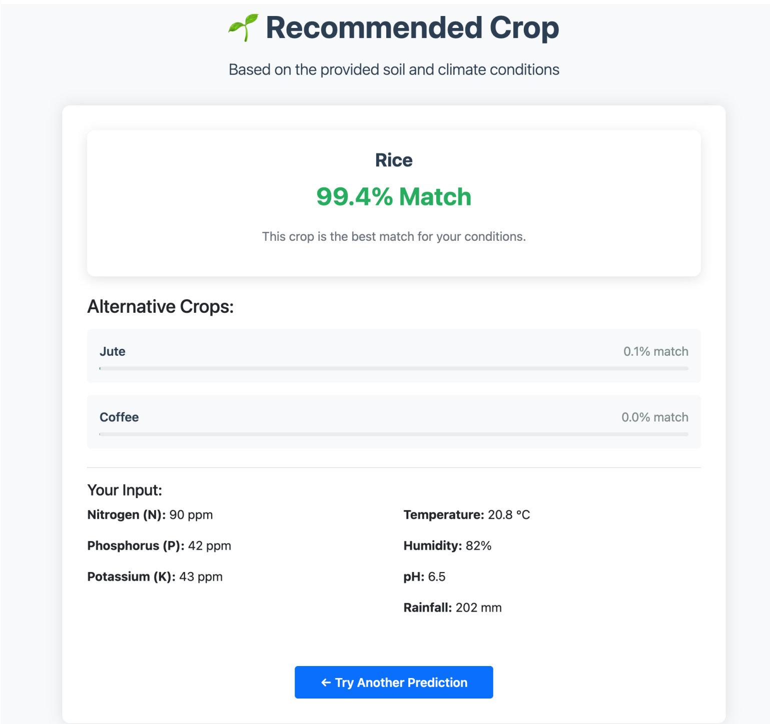
The Agriculture Crop Recommendation System demonstrated highly promising results, with the XGBoost model achieving 97% accuracy, maintaining a strong balance between precision and recall. This high level of performance confirms the model's reliability in accurately identifying the most suitable crop for specific soil and climatic conditions. The analysis further revealed that soil nutrients (Nitrogen, Phosphorus, Potassium), pH levels, and environmental factors such as temperature, humidity, and rainfall have a significant impact on determining crop suitability. This practical interface bridges advanced machine learning techniques with real-world farming needs. Overall, the project successfully supports data-driven agricultural decision-making, empowering farmers to optimize yield, conserve resources, and adopt sustainable cultivation practices.

Output

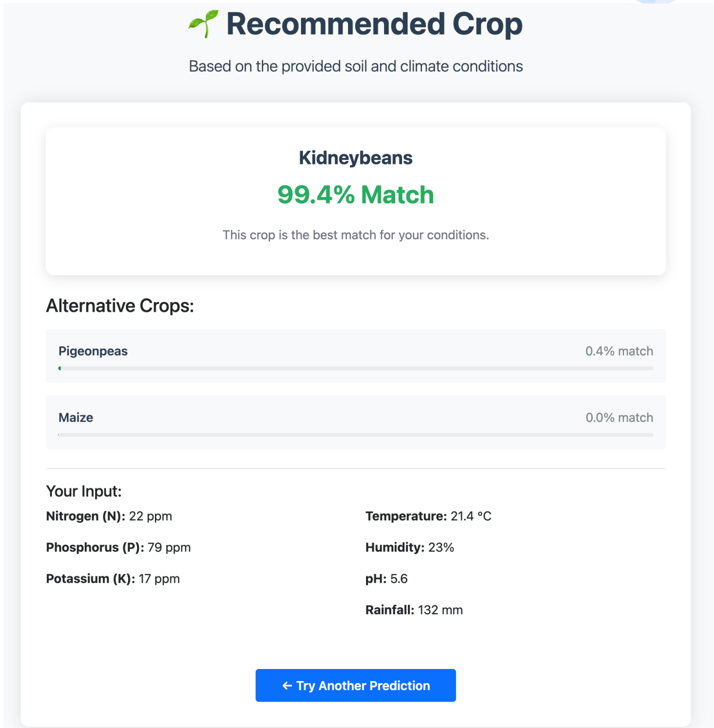


Crop Recommendation System
Enter the soil and climate conditions to get crop recommendations
Nitrogen (N) in kg/ha
50
Phosphorus (P) in kg/ha
50
Potassium (K) in kg/ha
50
Temperature (°C)
25
Relative Humidity (%):
60
Soil pH:
6.5
Rainfall (mm):
100
Get Crop Recommendation
This system uses machine learning to recommend the best crops based on your input parameters.

Output



Recommended Crop
Based on the provided soil and climate conditions
Rice
99.4% Match
This crop is the best match for your conditions.
Alternative Crops:
Jute
0.1% match
Coffee
0.0% match
Your Input:
Nitrogen (N): 90 ppm
Phosphorus (P): 42 ppm
Potassium (K): 43 ppm
Temperature: 20.8 °C
Humidity: 82%
pH: 6.5
Rainfall: 202 mm
← Try Another Prediction



Recommended Crop
Based on the provided soil and climate conditions
Kidneybeans
99.4% Match
This crop is the best match for your conditions.
Alternative Crops:
Pigeonpeas
0.4% match
Maize
0.0% match
Your Input:
Nitrogen (N): 22 ppm
Phosphorus (P): 79 ppm
Potassium (K): 17 ppm
Temperature: 21.4 °C
Humidity: 23%
pH: 5.6
Rainfall: 132 mm
← Try Another Prediction

Thanks!

Any *questions* ?