Crop_recommendation_system

August 31, 2025

#Crop Recommendation System (ML Project)

Author: Mukundhan S
Description: Preprocessing, analysis, and modeling of crop prediction dataset using Machine Learning techniques.

[ ]:
```
import pandas as pd

df = pd.read_csv('crop_data.csv')  # Replace with your dataset path
print("First 5 rows of dataset:")
print(df.head())
```

```
First 5 rows of dataset:
    N   P   K  temperature   humidity        ph    rainfall label
0  90  42  43    20.879744  82.002744  6.502985  202.935536  rice
1  85  58  41    21.770462  80.319644  7.038096  226.655537  rice
2  60  55  44    23.004459  82.320763  7.840207  263.964248  rice
3  74  35  40    26.491096  80.158363  6.980401  242.864034  rice
4  78  42  42    20.130175  81.604873  7.628473  262.717340  rice
```

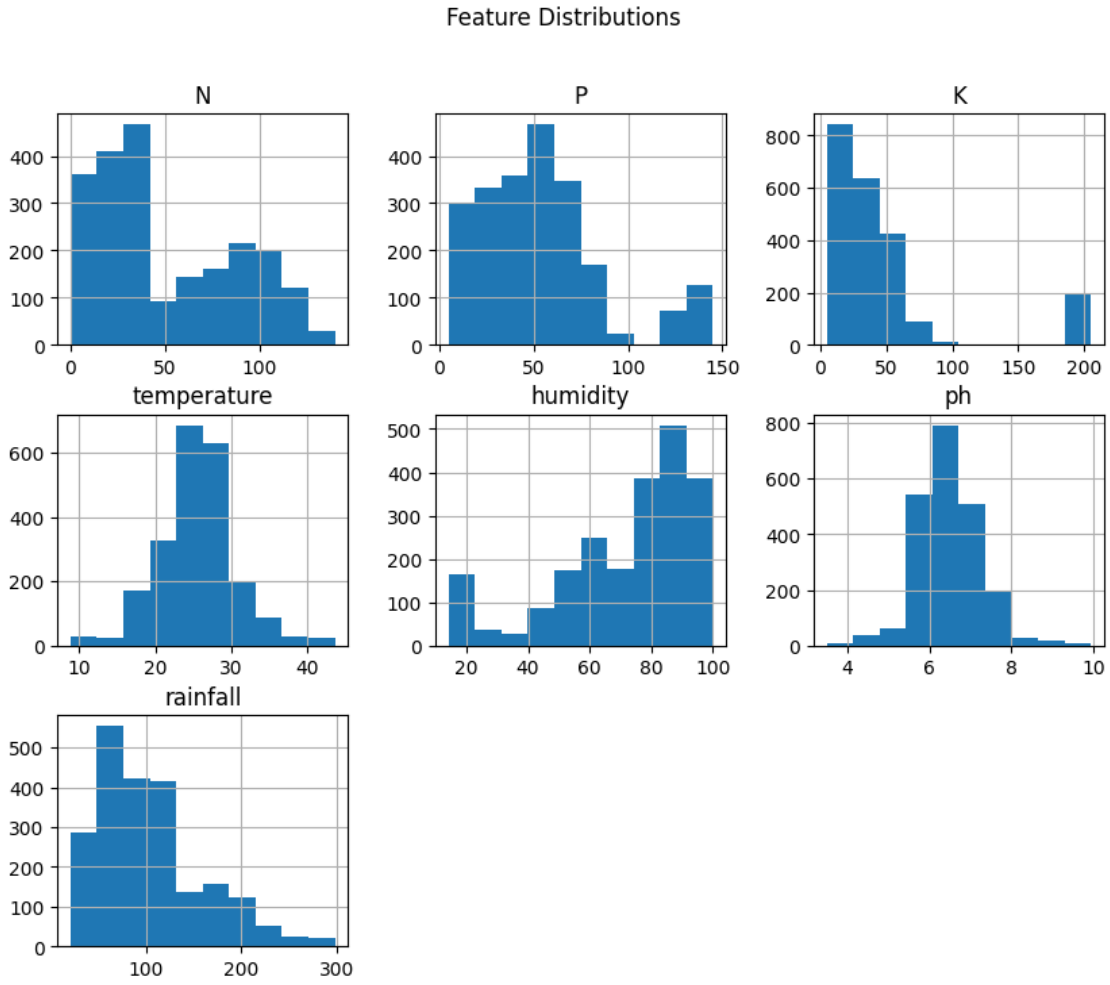
[ ]:
```
print("\nDataset Info:")
print(df.info())

print("\nMissing Values:")
print(df.isnull().sum())

print("\nStatistical Summary:")
print(df.describe())

print("\nCrop Counts:")
print(df['label'].value_counts())
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   N            2200 non-null   int64
```

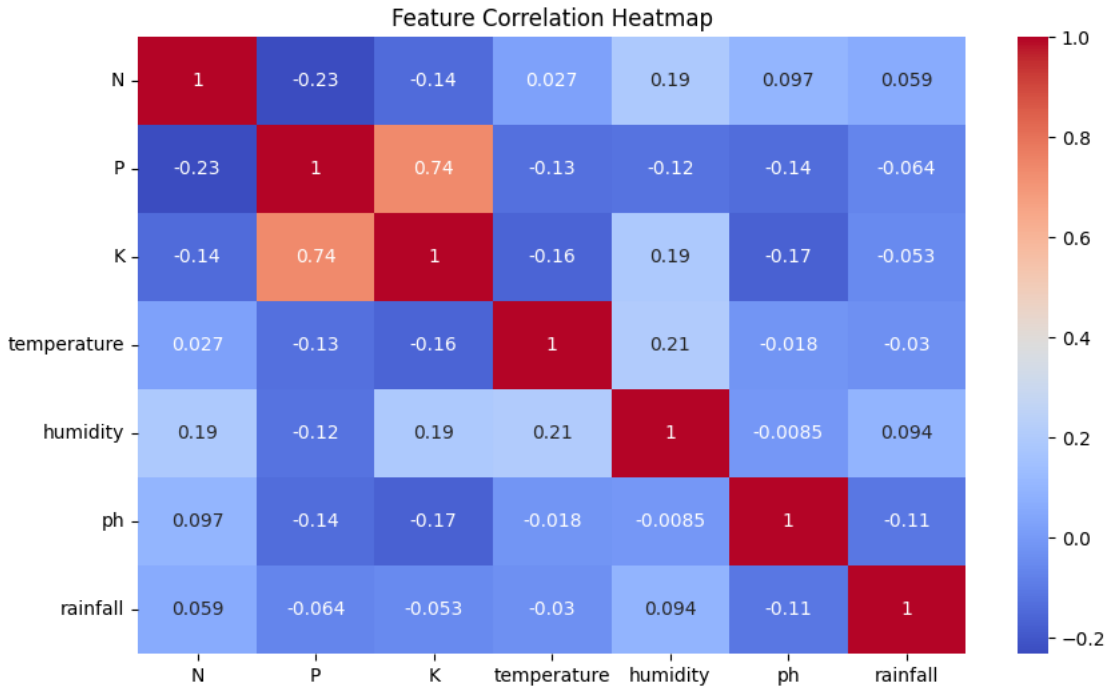
```
 1   P            2200 non-null   int64
 2   K            2200 non-null   int64
 3   temperature  2200 non-null   float64
 4   humidity     2200 non-null   float64
 5   ph           2200 non-null   float64
 6   rainfall     2200 non-null   float64
 7   label        2200 non-null   object
dtypes: float64(4), int64(3), object(1)
memory usage: 137.6+ KB
None

Missing Values:
N              0
P              0
K              0
temperature    0
humidity       0
ph             0
rainfall       0
label          0
dtype: int64

Statistical Summary:
                 N            P            K  temperature     humidity  \
count  2200.000000  2200.000000  2200.000000  2200.000000  2200.000000
mean     50.551818    53.362727    48.149091    25.616244    71.481779
std      36.917334    32.985883    50.647931     5.063749    22.263812
min       0.000000     5.000000     5.000000     8.825675    14.258040
25%      21.000000    28.000000    20.000000    22.769375    60.261953
50%      37.000000    51.000000    32.000000    25.598693    80.473146
75%      84.250000    68.000000    49.000000    28.561654    89.948771
max     140.000000   145.000000   205.000000    43.675493    99.981876

                ph     rainfall
count  2200.000000  2200.000000
mean      6.469480   103.463655
std       0.773938    54.958389
min       3.504752    20.211267
25%       5.971693    64.551686
50%       6.425045    94.867624
75%       6.923643   124.267508
max       9.935091   298.560117

Crop Counts:
label
rice           100
maize          100
chickpea       100
```

```
kidneybeans    100
pigeonpeas     100
mothbeans      100
mungbean       100
blackgram      100
lentil         100
pomegranate    100
banana         100
mango          100
grapes         100
watermelon     100
muskmelon      100
apple          100
orange         100
papaya         100
coconut        100
cotton         100
jute           100
coffee         100
Name: count, dtype: int64
```

[ ]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Histogram
df.hist(figsize=(10,8))
plt.suptitle("Feature Distributions")
plt.show()
```



[ ]:
```
# Correlation Heatmap
plt.figure(figsize=(10,6))
sns.heatmap(df.drop('label', axis=1).corr(), annot=True, cmap='coolwarm')
plt.title("Feature Correlation Heatmap")
plt.show()
```



[ ]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
df['label'] = le.fit_transform(df['label'])

# Optional: view mapping
mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("\nLabel encoding mapping:", mapping)
```

```
Label encoding mapping: {'apple': np.int64(0), 'banana': np.int64(1),
'blackgram': np.int64(2), 'chickpea': np.int64(3), 'coconut': np.int64(4),
'coffee': np.int64(5), 'cotton': np.int64(6), 'grapes': np.int64(7), 'jute':
np.int64(8), 'kidneybeans': np.int64(9), 'lentil': np.int64(10), 'maize':
np.int64(11), 'mango': np.int64(12), 'mothbeans': np.int64(13), 'mungbean':
np.int64(14), 'muskmelon': np.int64(15), 'orange': np.int64(16), 'papaya':
np.int64(17), 'pigeonpeas': np.int64(18), 'pomegranate': np.int64(19), 'rice':
np.int64(20), 'watermelon': np.int64(21)}
```

[ ]:
```
from sklearn.preprocessing import StandardScaler

X = df.drop('label', axis=1)
y = df['label']

scaler = StandardScaler()
```

```
X_scaled = scaler.fit_transform(X)

# Optional: save preprocessed data
preprocessed_df = pd.DataFrame(X_scaled, columns=X.columns)
preprocessed_df['label'] = y
preprocessed_df.to_csv('preprocessed_crop_data.csv', index=False)
print("\nPreprocessed dataset saved as 'preprocessed_crop_data.csv'")
```

```
Preprocessed dataset saved as 'preprocessed_crop_data.csv'
```

[ ]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

print("Training data shape:", X_train.shape)
print("Testing data shape:", X_test.shape)
print("Training labels shape:", y_train.shape)
print("Testing labels shape:", y_test.shape)
```

```
Training data shape: (1760, 7)
Testing data shape: (440, 7)
Training labels shape: (1760,)
Testing labels shape: (440,)
```

[ ]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
print("Model training complete.")
```

```
Model training complete.
```