In [22]:
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import roc_curve, auc
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
```

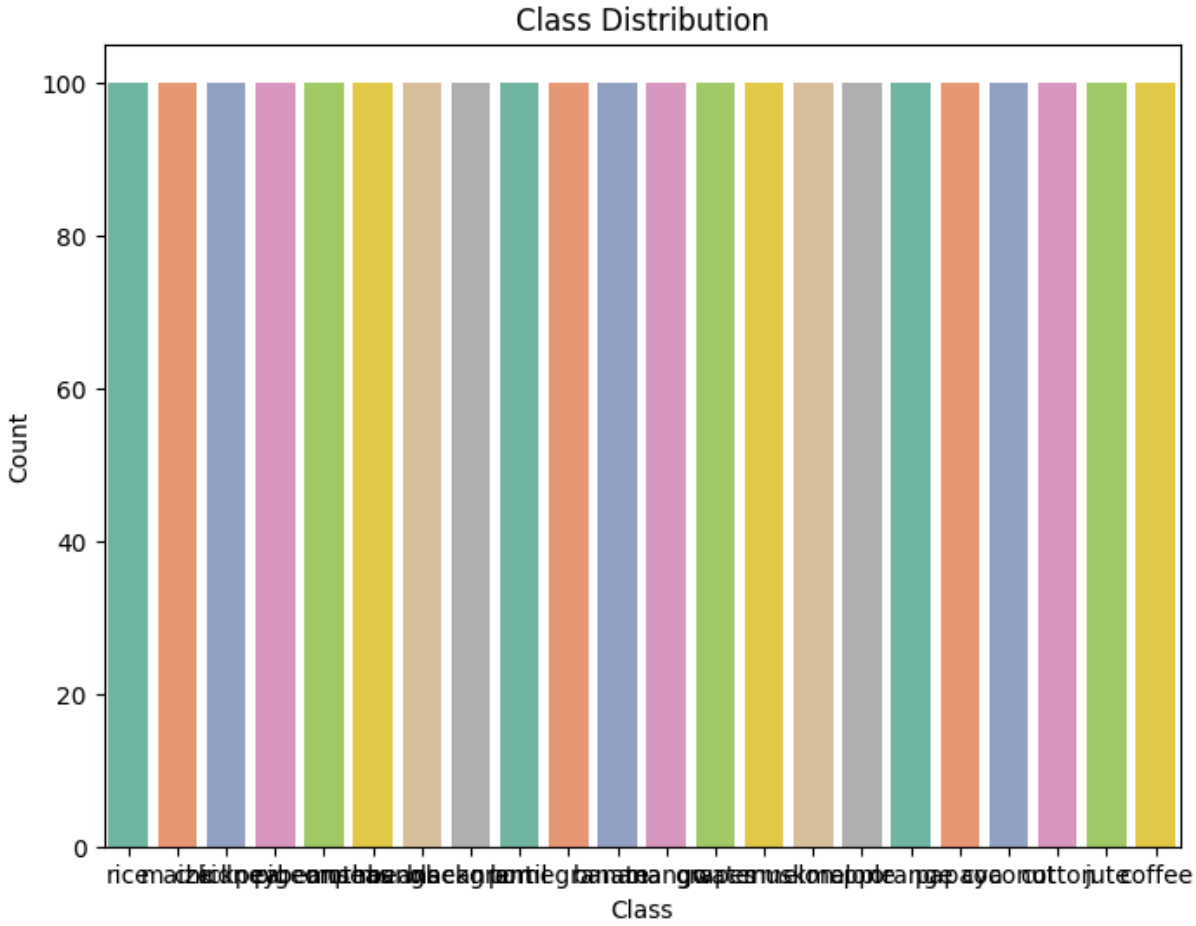
In [23]:
```
def plot_class_distribution(data, target_column):
    plt.figure(figsize=(8, 6))
    sns.countplot(x=target_column, data=data, palette='Set2')
    plt.title('Class Distribution')
    plt.xlabel('Class')
    plt.ylabel('Count')
    plt.show()

def plot_correlation_heatmap(data):
    numeric_data = data.select_dtypes(include=['float64', 'int64'])
    plt.figure(figsize=(12, 8))
    sns.heatmap(numeric_data.corr(), annot=True, fmt=".2f", cmap='coolwarm',
    plt.title('Feature Correlation Heatmap')
    plt.show()

def plot_pairplot(data, target_column):
    sns.pairplot(data, hue=target_column, palette='Set1', diag_kind='kde')
    plt.title('Pairplot of Features')
    plt.show()

def plot_feature_distributions(data, numeric_columns):
    data[numeric_columns].hist(bins=20, figsize=(15, 10), color='skyblue', e
    plt.suptitle(f'Feature Distributions of {numeric_columns}', fontsize=16)
    plt.show()

def plot_boxplots(data, numeric_columns, target_column):
    for col in numeric_columns:
        plt.figure(figsize=(8, 6))
        sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
        plt.title(f'Boxplot of {col} by {target_column}')
        plt.show()

def plot_feature_importance(data, features, target):
    model = RandomForestClassifier(random_state=42)
    model.fit(data[features], data[target])
    importances = pd.Series(model.feature_importances_, index=features)
    importances.sort_values(ascending=True).plot(kind='barh', figsize=(10, 6
    plt.title('Feature Importance')
    plt.xlabel('Importance Score')
    plt.ylabel('Features')
    plt.show()

def plot_pca_2d(data, features, target):
    scaler = StandardScaler()
    scaled_data = scaler.fit_transform(data[features])
```

```
    pca = PCA(n_components=2)
    pca_result = pca.fit_transform(scaled_data)
    pca_df = pd.DataFrame(pca_result, columns=['PCA1', 'PCA2'])
    pca_df[target] = data[target].values

    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='PCA1', y='PCA2', hue=target, data=pca_df, palette='Se
    plt.title('PCA Visualization (2D)')
    plt.xlabel('PCA1')
    plt.ylabel('PCA2')
    plt.legend(loc='best')
    plt.show()

def plot_confusion_matrix(y_true, y_pred, classes):
    cm = confusion_matrix(y_true, y_pred, labels=classes)
    disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=classe
    disp.plot(cmap='Blues', xticks_rotation='vertical')
    plt.title('Confusion Matrix')
    plt.show()

def plot_roc_curve(y_true, y_prob):
    fpr, tpr, _ = roc_curve(y_true, y_prob)
    roc_auc = auc(fpr, tpr)
    plt.figure(figsize=(8, 6))
    plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {
    plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.legend(loc="lower right")
    plt.show()
```

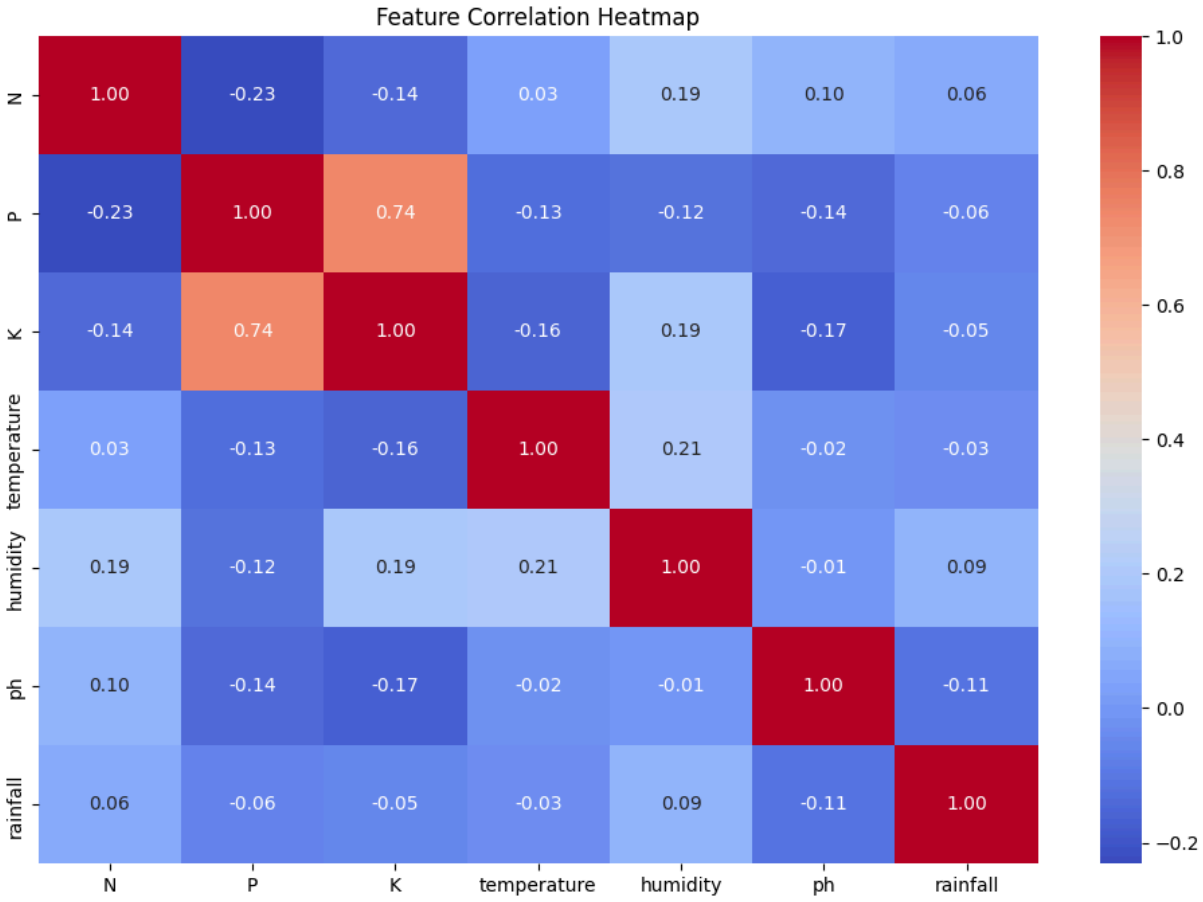
In [24]:
```
data_file_name = 'Crop_recommendation.csv'
data = pd.read_csv(data_file_name)
numeric_columns = ["N", "P", "K", "temperature", "humidity", "ph", "rainfall
target_column = "label"
```

In [25]:
```
plot_class_distribution(data, "label")
```

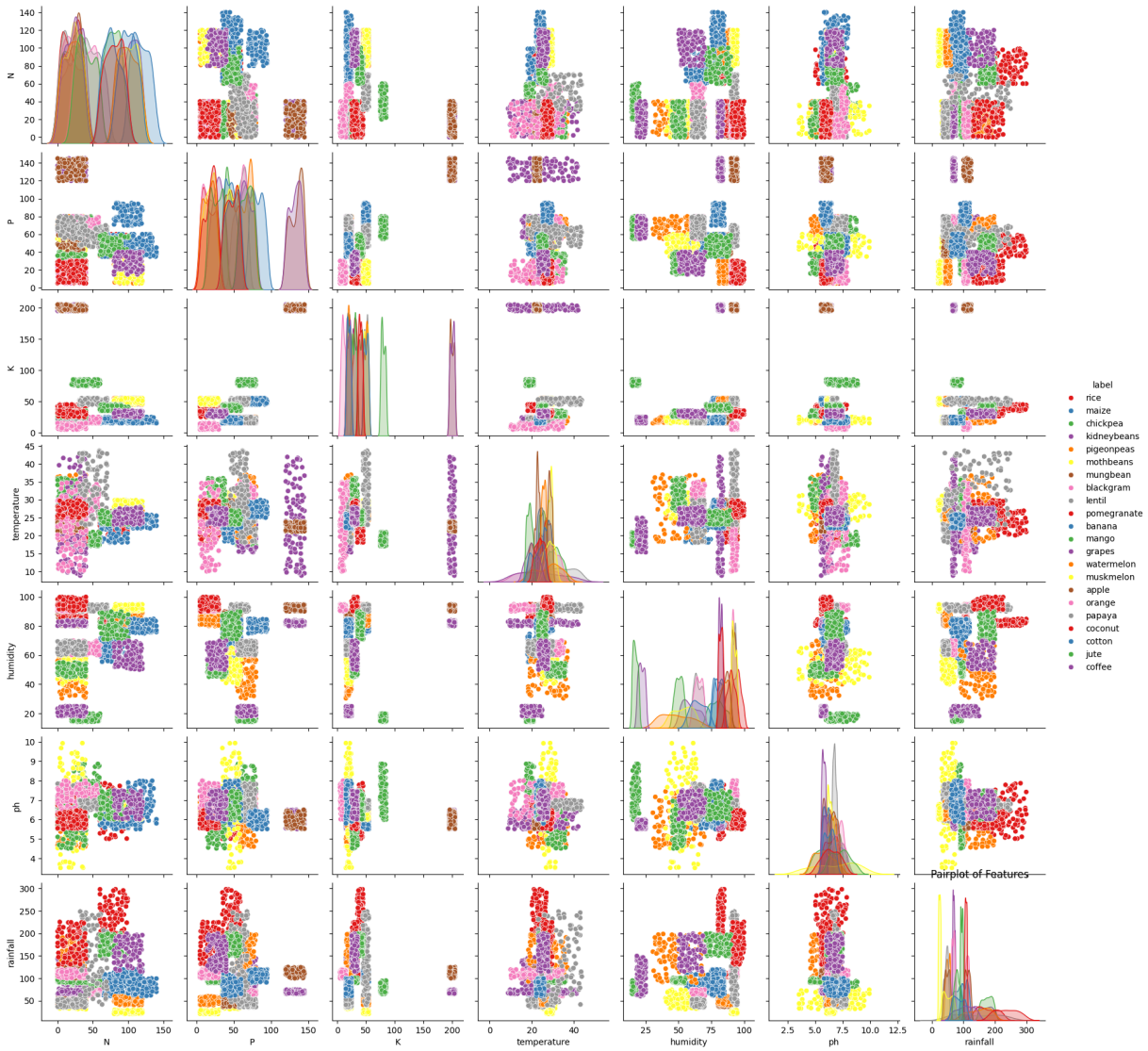
```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.countplot(x=target_column, data=data, palette='Set2')
```



In [26]:
```
plot_correlation_heatmap(data)
```

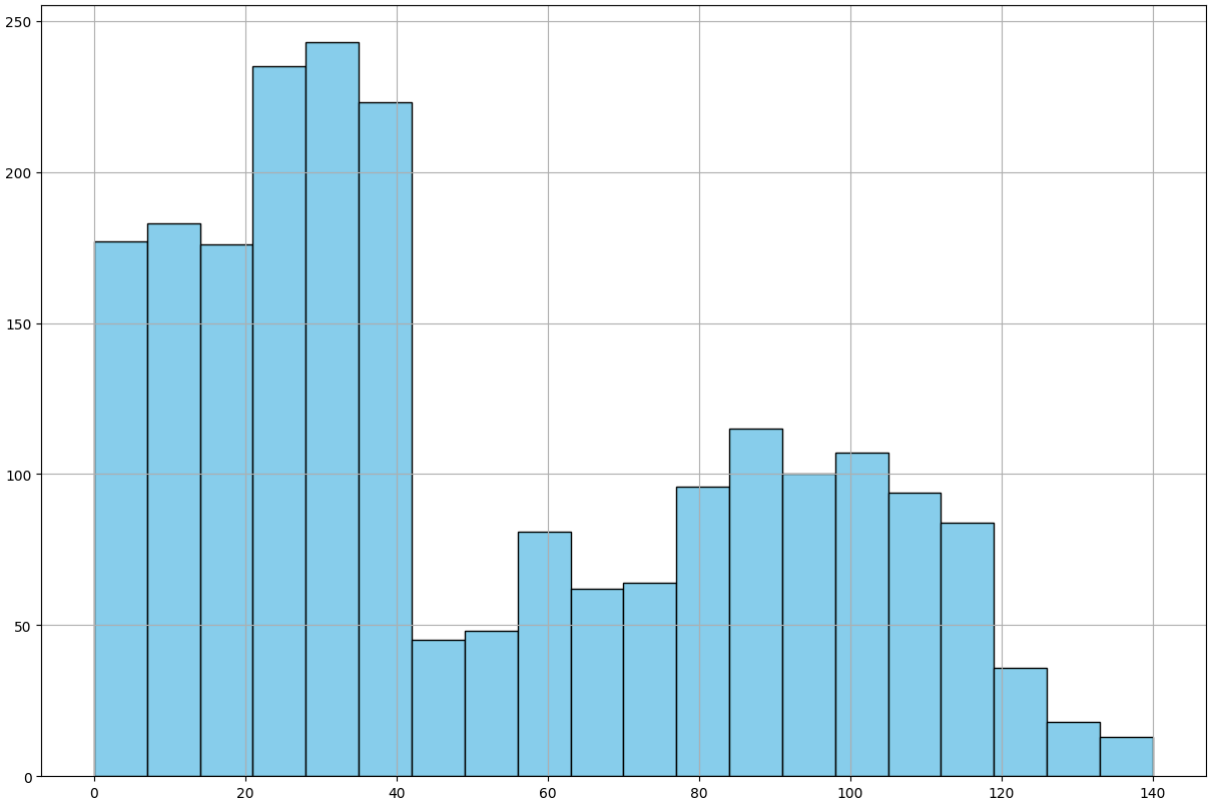


In [27]:
```
plot_pairplot(data, "label")
```

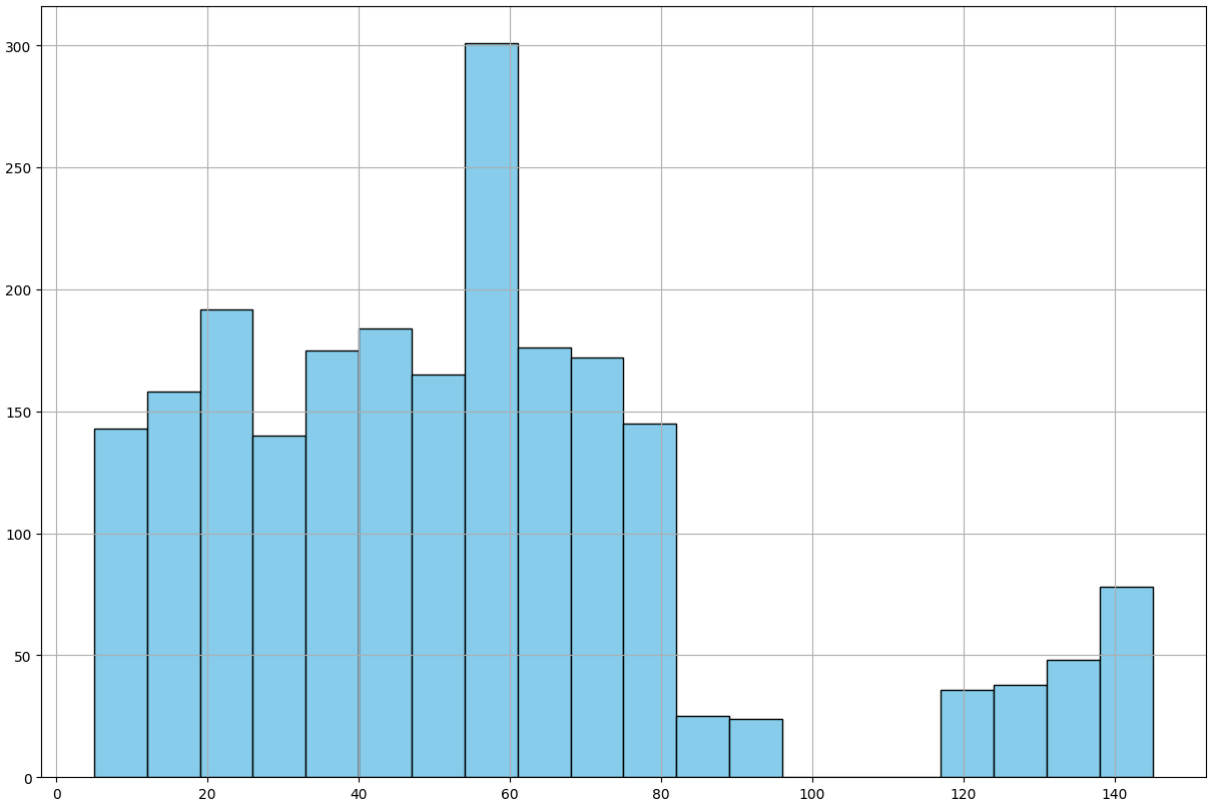


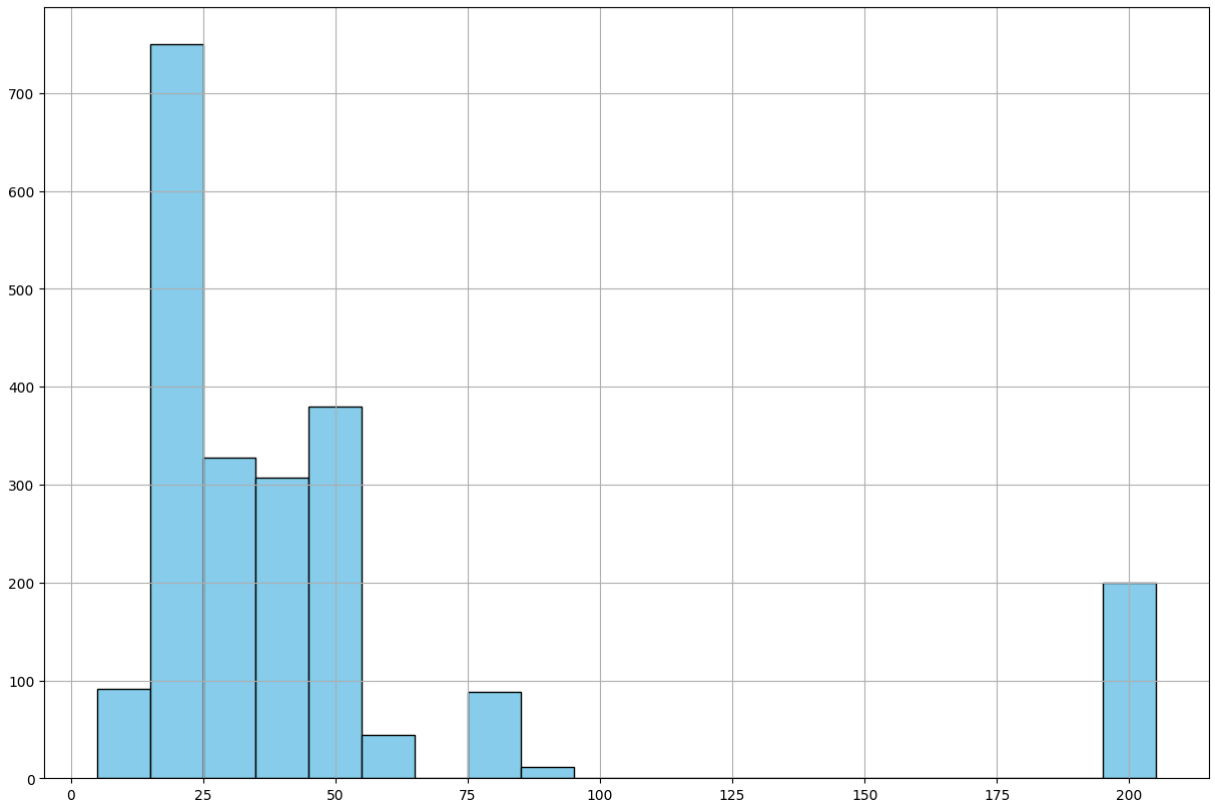
In [28]:
```
# N,P,K
plot_feature_distributions(data, "N")
plot_feature_distributions(data, "P")
plot_feature_distributions(data, "K")
```

Feature Distributions of N



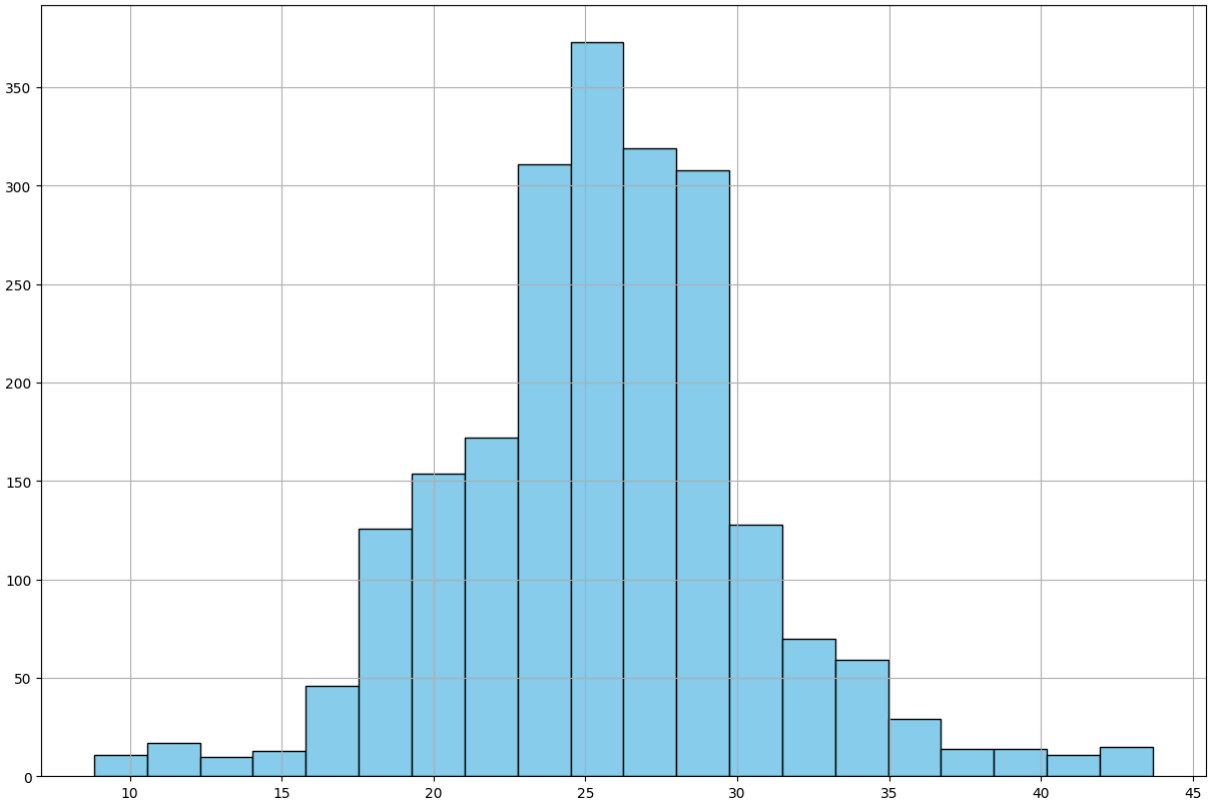
Feature Distributions of P



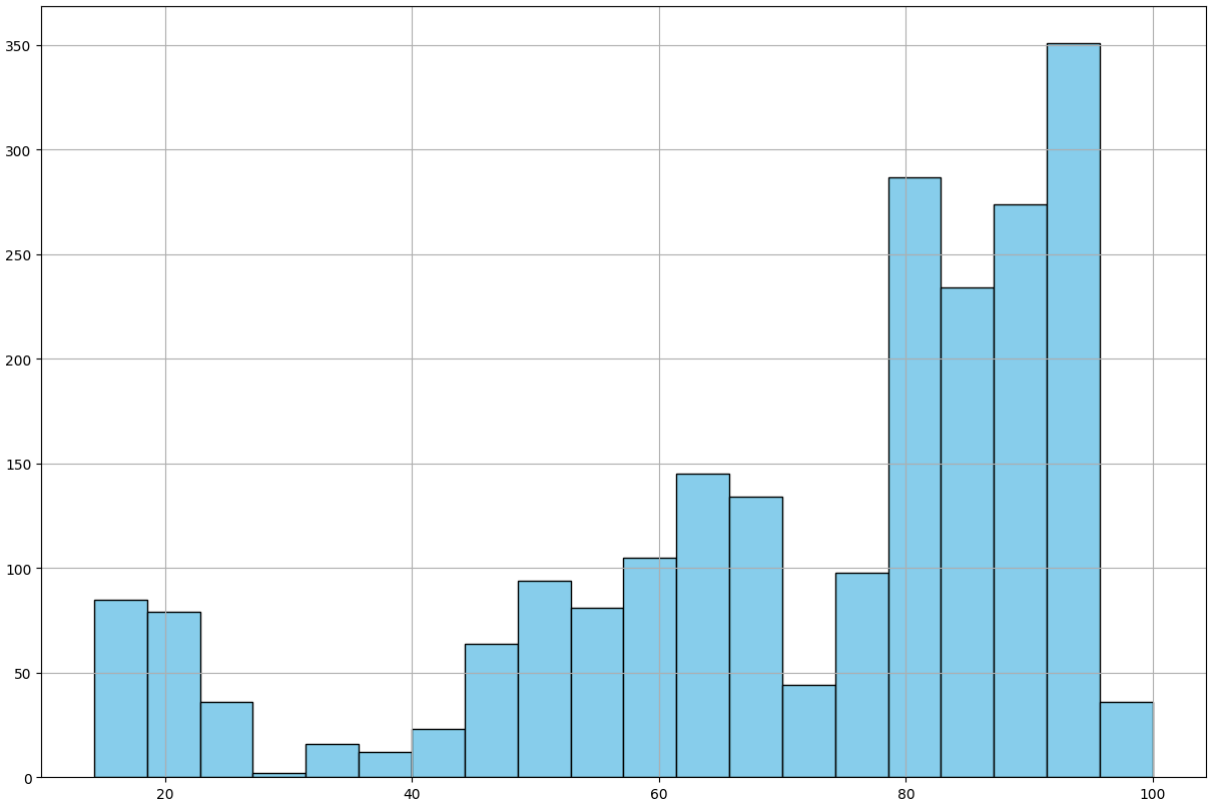


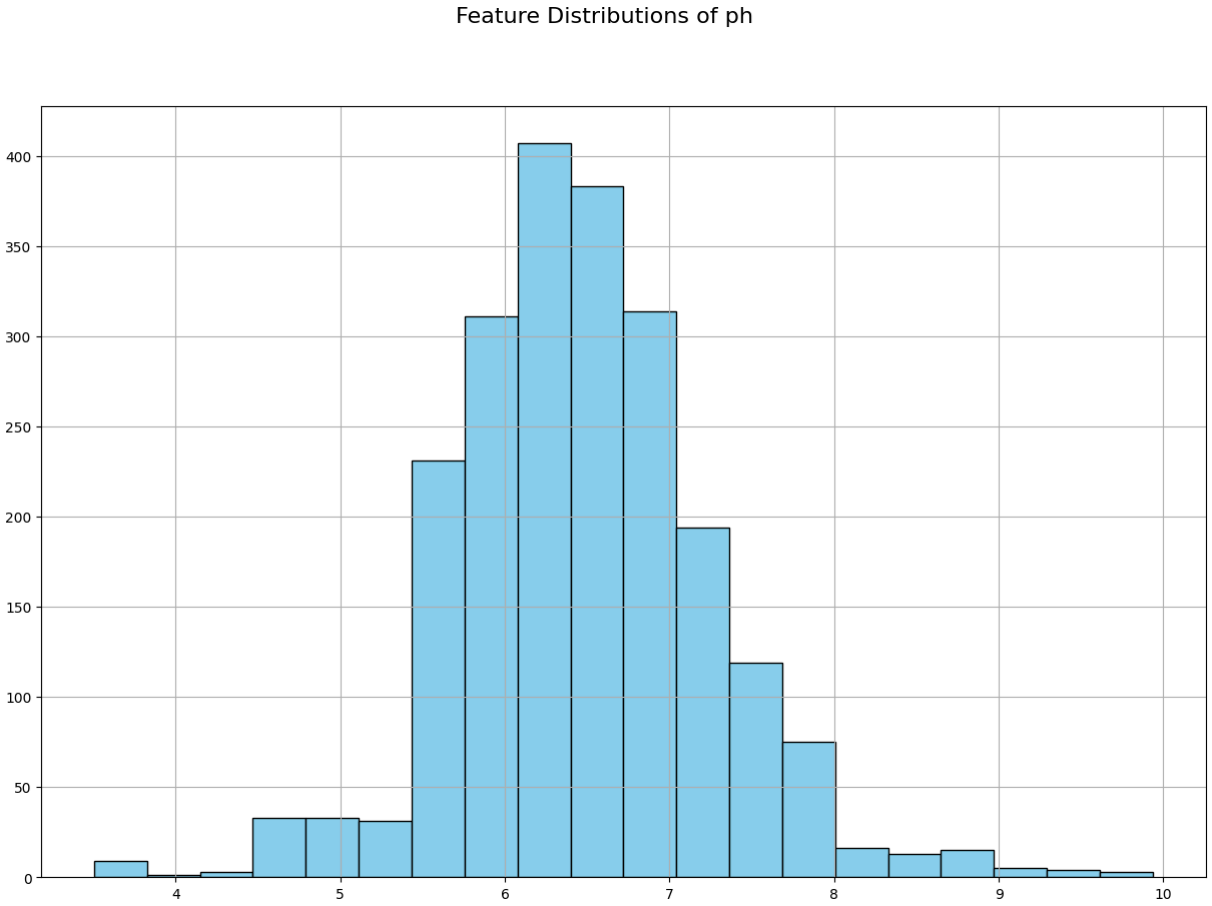
In [29]:
```
# temperature,humidity,ph,rainfall
plot_feature_distributions(data, "temperature")
plot_feature_distributions(data, "humidity")
plot_feature_distributions(data, "ph")
plot_feature_distributions(data, "rainfall")
```

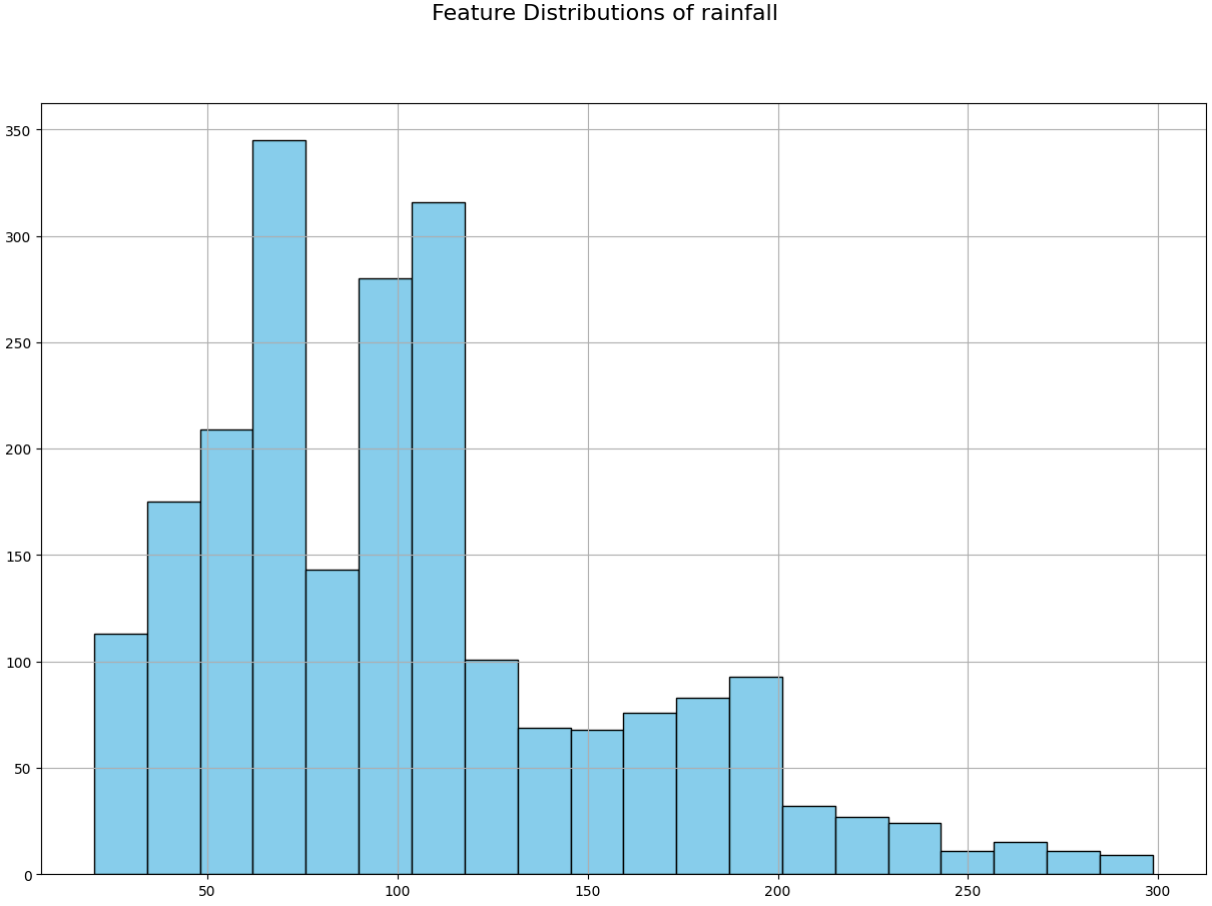
Feature Distributions of temperature



Feature Distributions of humidity

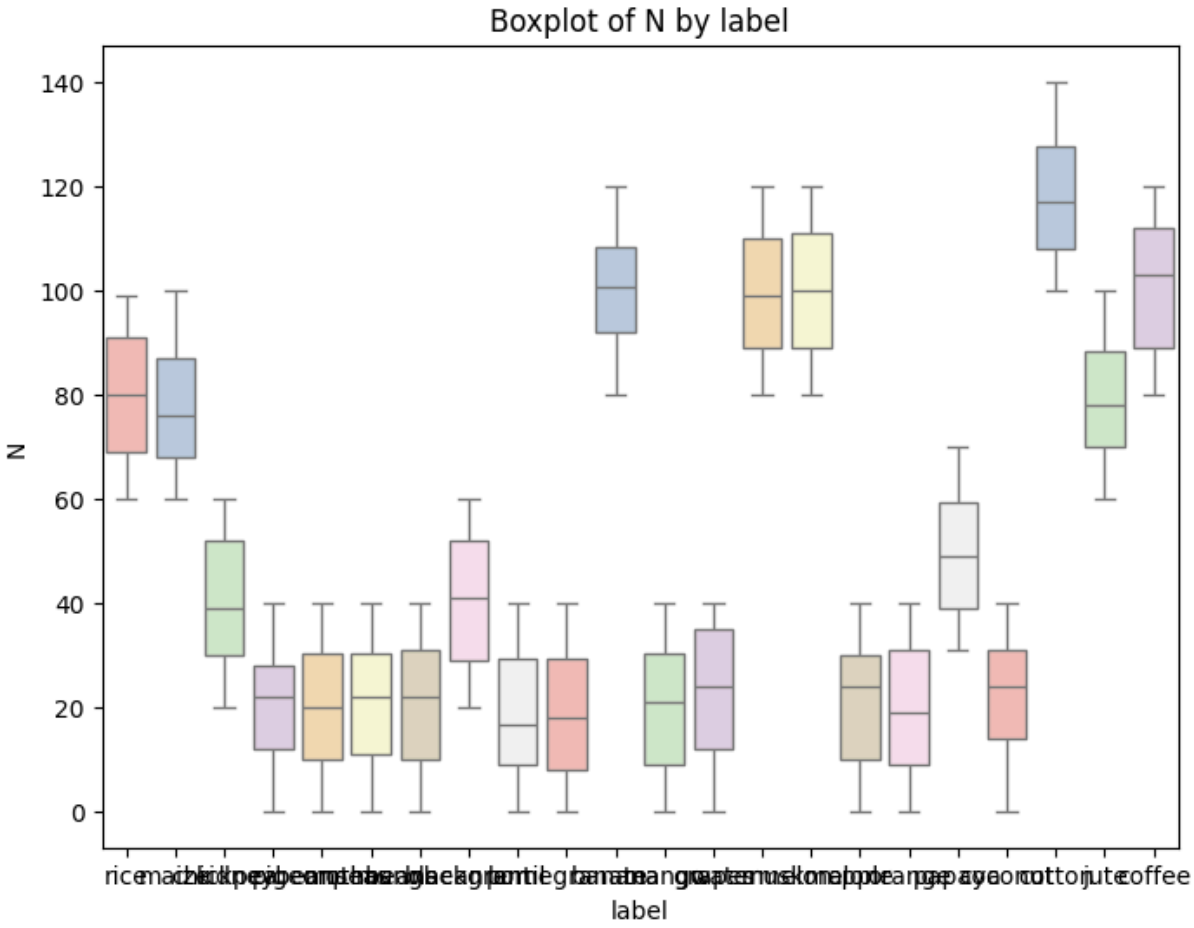






In [30]:
```
plot_boxplots(data, numeric_columns, target_column)
```

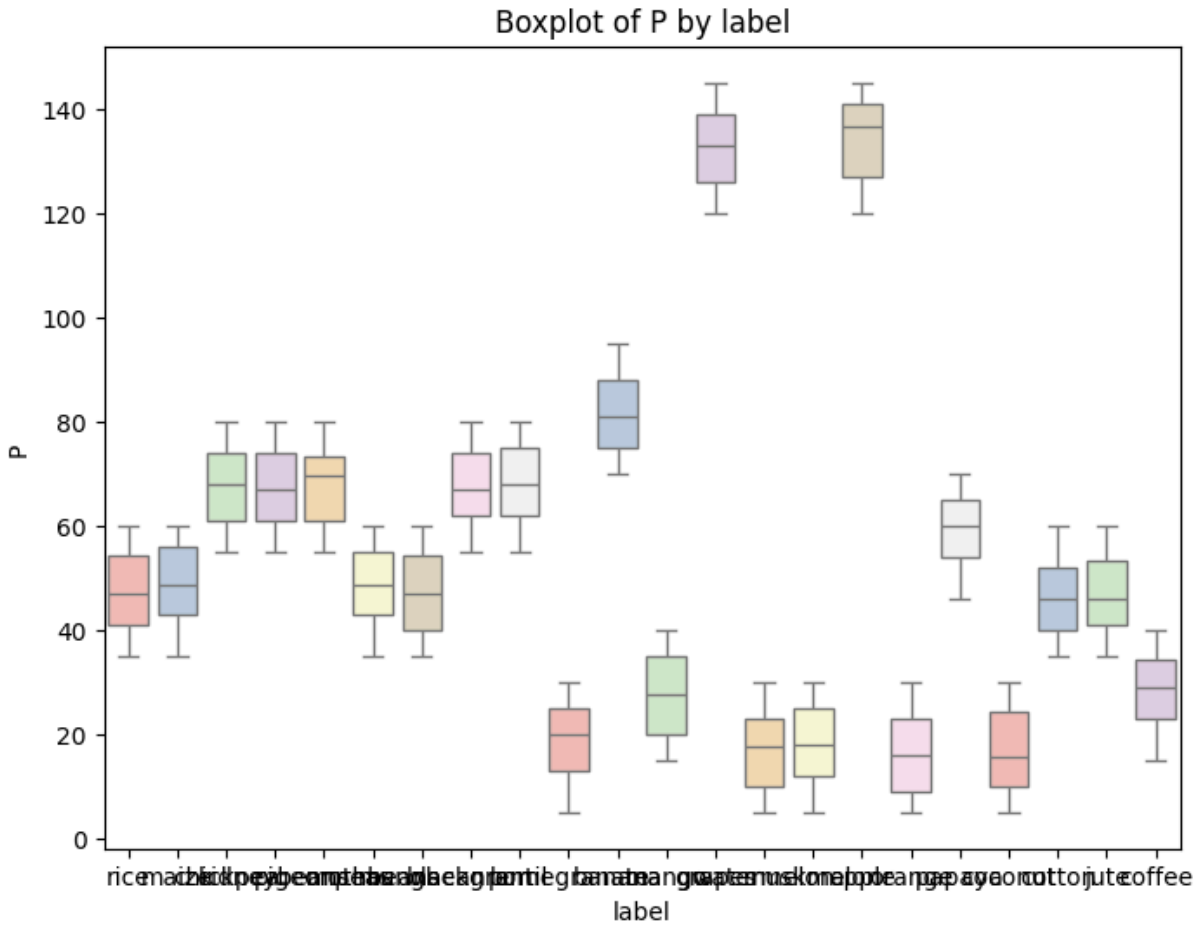
```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



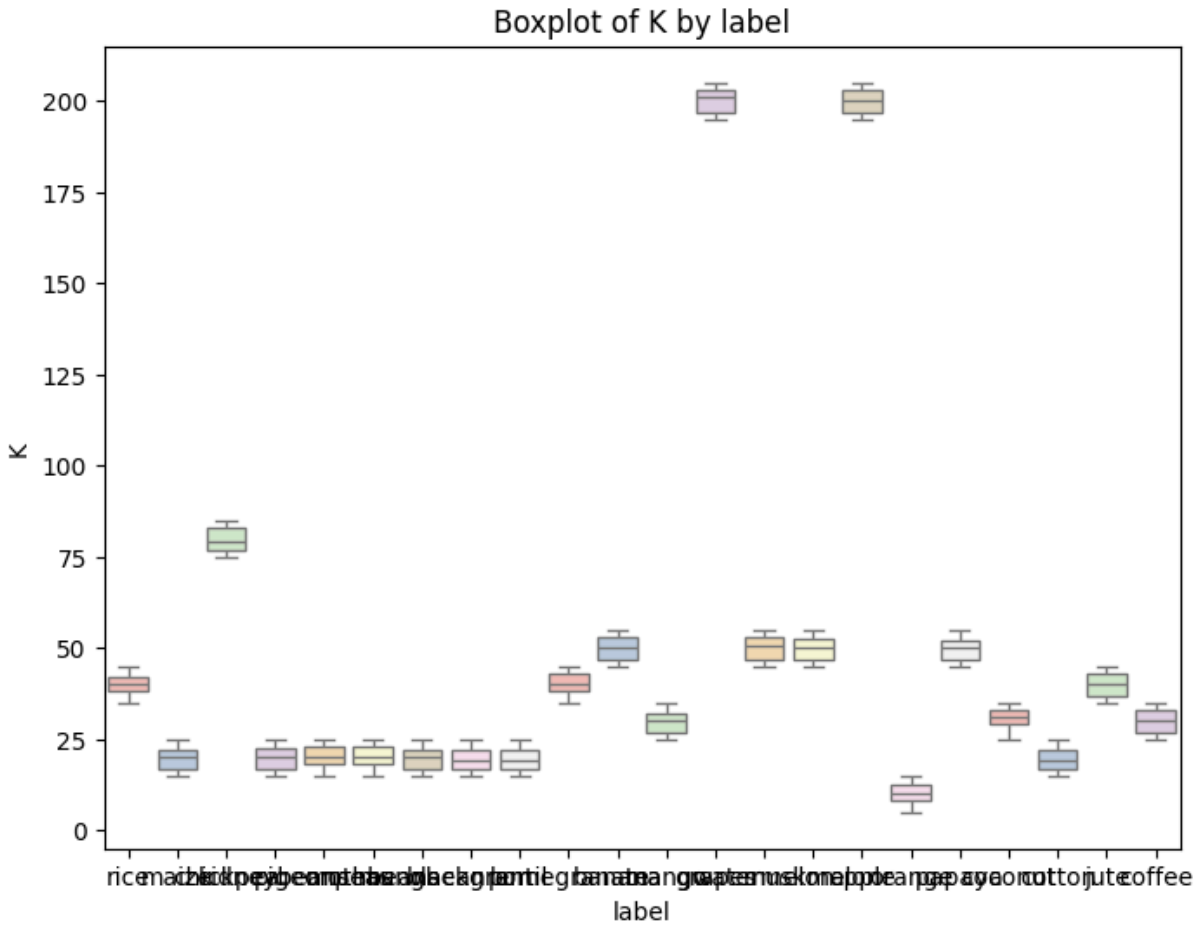
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

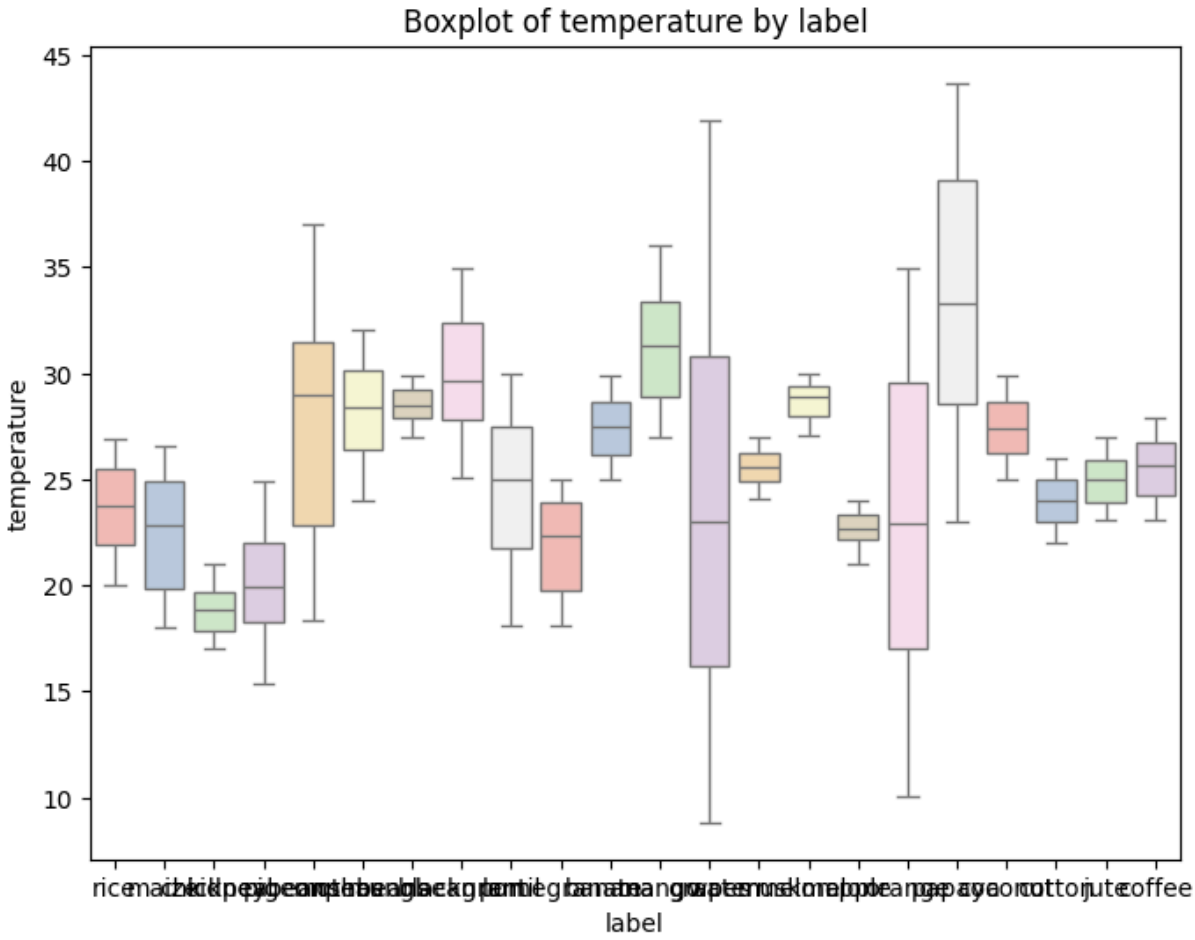
  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')



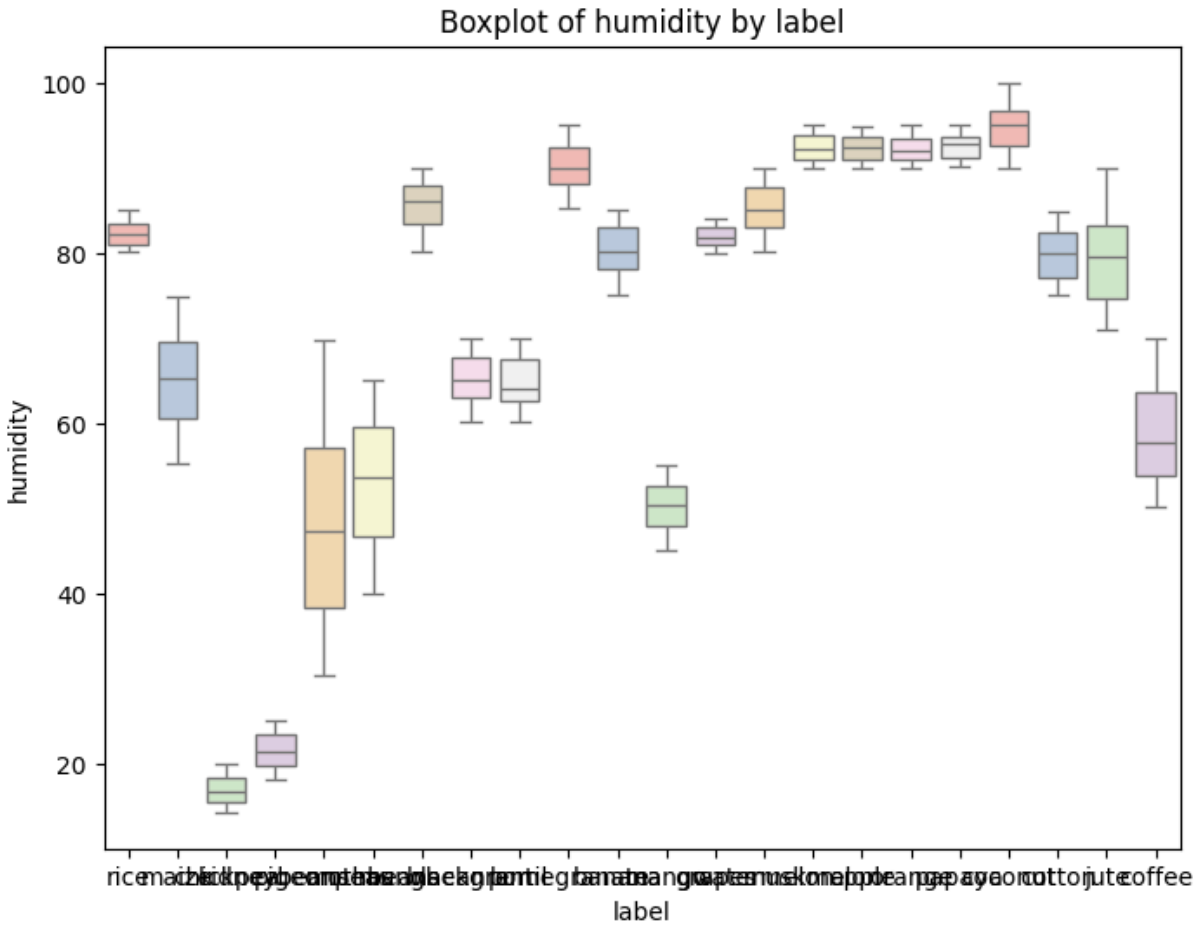

```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



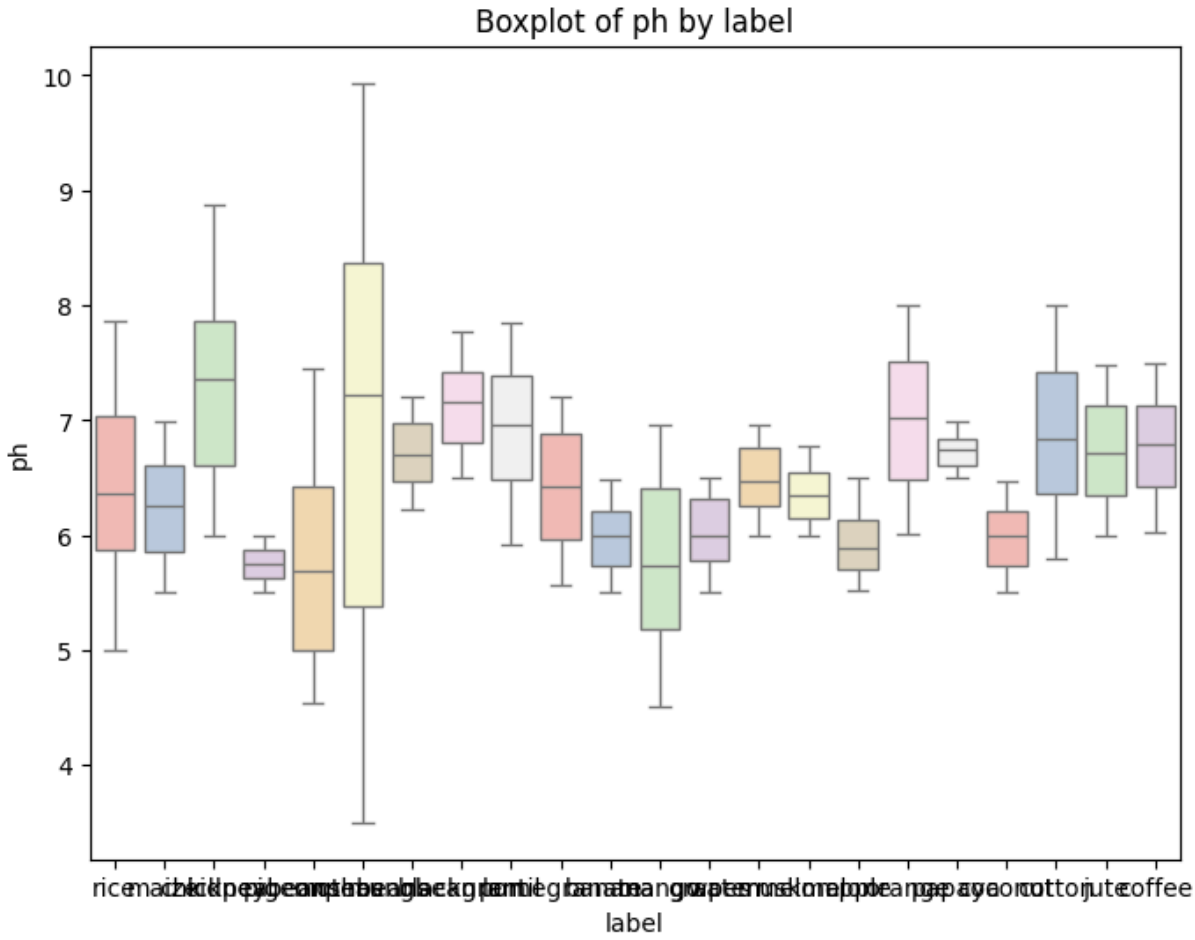

```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



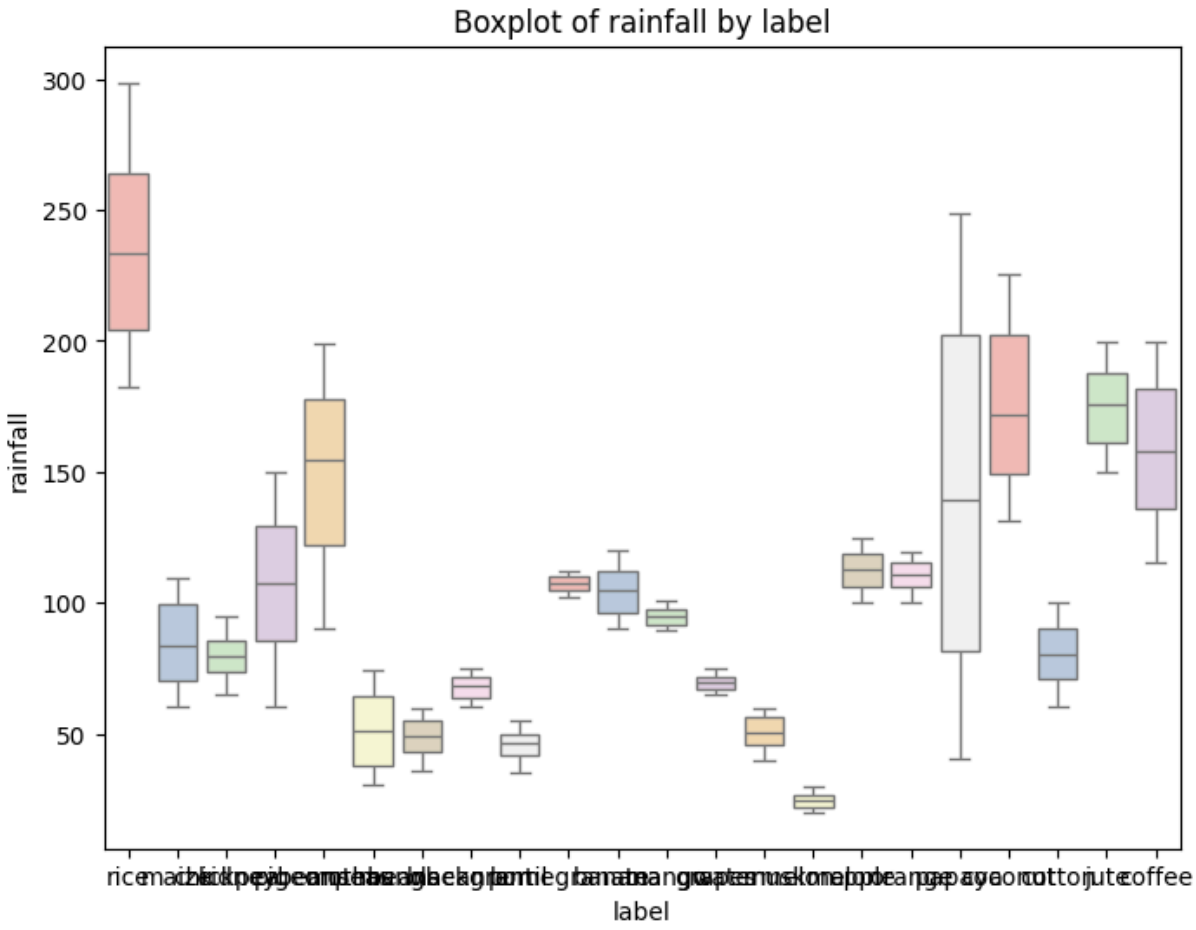

```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



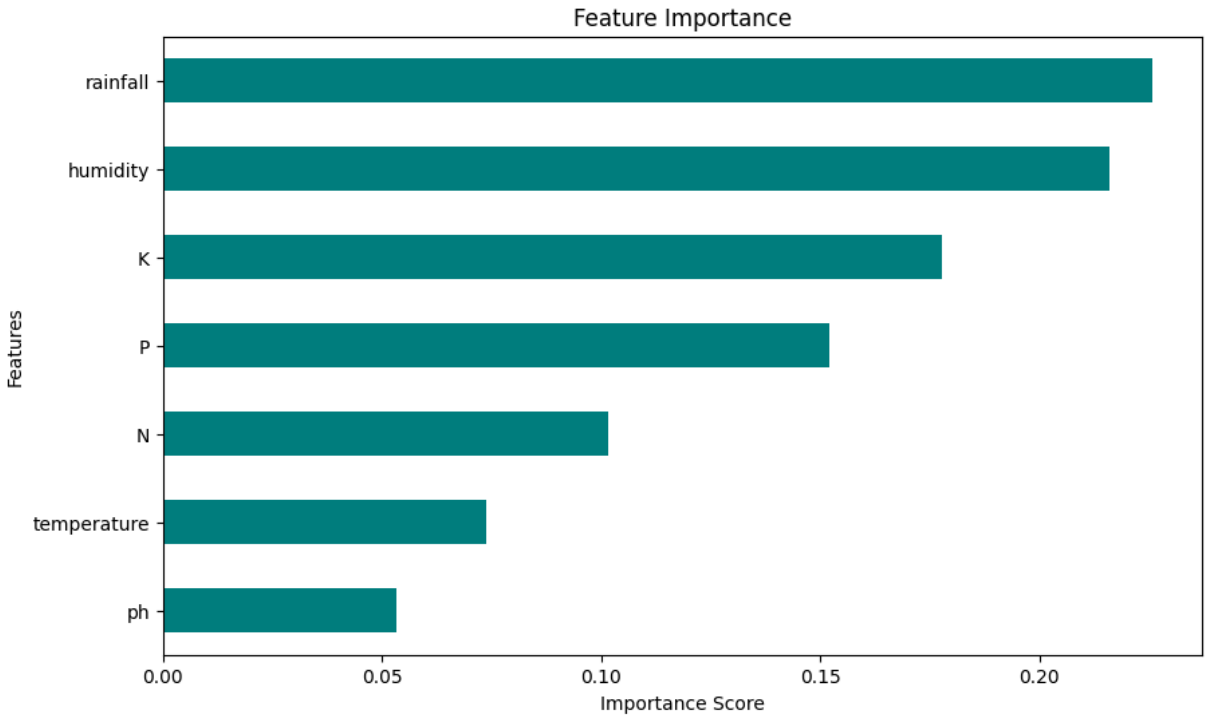

```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



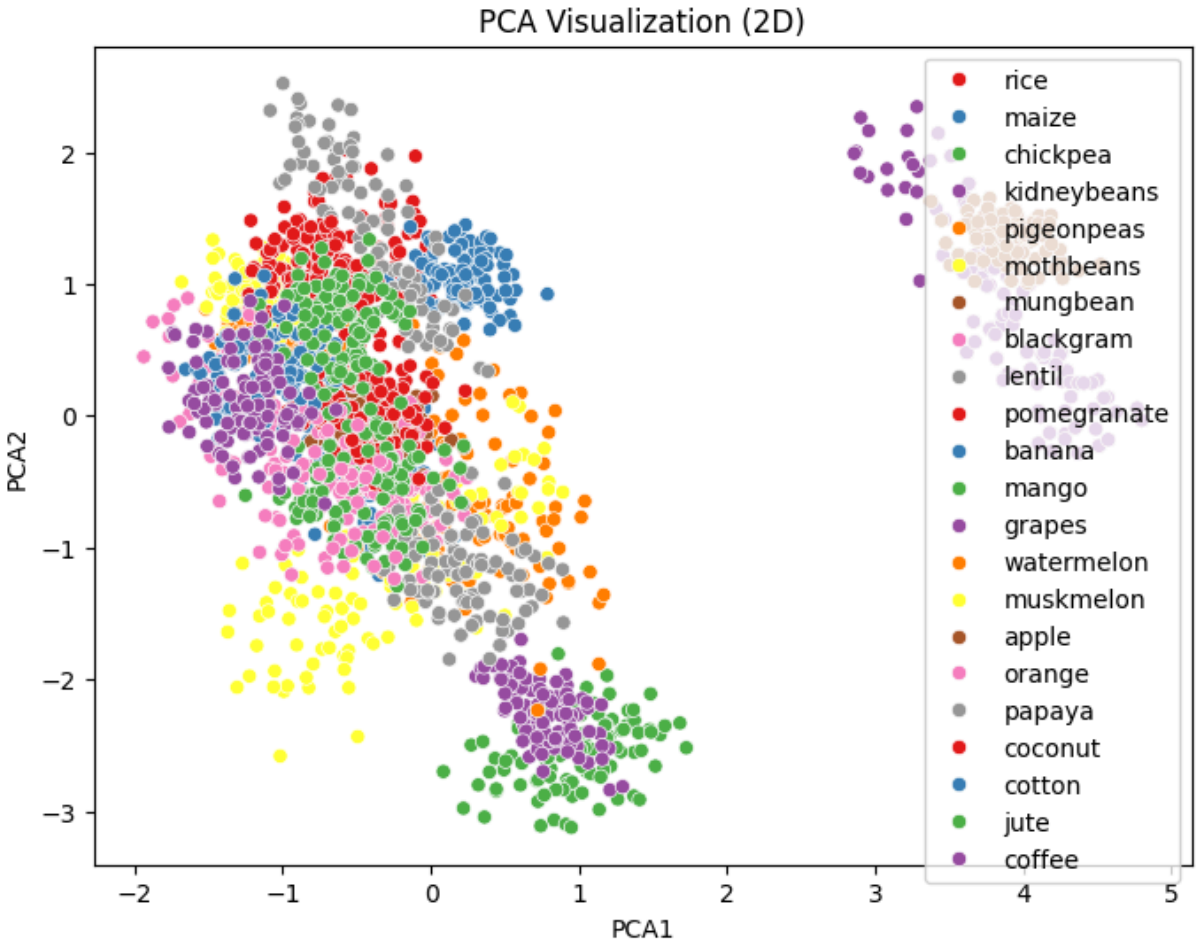

```
/var/folders/bq/zqxz2qdd1hndrg7ygfqnk2rc0000gn/T/ipykernel_10891/1459129845.
py:29: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x=target_column, y=col, data=data, palette='Pastel1')
```



In [32]:
```
plot_feature_importance(data, numeric_columns, target_column)
```



In [33]:
```
plot_pca_2d(data, numeric_columns, target_column)
```


PCA Visualization (2D)
PCA1
PCA2
rice
maize
chickpea
kidneybeans
pigeonpeas
mothbeans
mungbean
blackgram
lentil
pomegranate
banana
mango
grapes
watermelon
muskmelon
apple
orange
papaya
coconut
cotton
jute
coffee