Machine Learning Report

Crop and Fertilizer Recommendation System

Nainil Rajendra Maladkar
College of Engineering
Northeastern University
NUID: 002780019
maladkar.n@northeastern.edu

Simran Nagpurkar
College of Engineering
Northeastern University
NUID: 002922747
nagpurkar.s@northeastern.edu

ABSTRACT

This project aims to optimize agricultural productivity by developing a machine learning-based crop and fertilizer recommendation system. It evaluates various algorithms—Naive Bayes, Random Forest and Neural Networks against metrics like precision, recall, F1-score, and accuracy. The initial phase involved exploratory data analysis and feature scaling to standardize the dataset. Naive Bayes and Random Forest exhibited exceptional performance, its suitability for datasets with independent features. Advanced models were also assessed to capture complex patterns.

Confusion matrices were used to fine-tune predictions, guiding improvements for misclassified instances. The project underscores the importance of selecting an appropriate model based on the dataset's nuances and cross-validation to ensure model reliability. Finally, a Flask application was created as an interface, allowing for seamless interaction with the model's recommendations.

1. PROBLEM DEFINITION:

The Crop and Fertilizer Recommendation System is a Python Machine Learning project aimed at recommending optimal crops to farmers based on various soil and environmental factors. The goal is to leverage data-driven insights to suggest the most suitable crops using different fertilizers, thereby enhancing agricultural productivity and sustainability.
In this project, we will develop a model that can analyze soil characteristics (like Nitrogen, Phosphorus, Potassium levels), environmental conditions (temperature, humidity), and rainfall patterns and type of fertilizer to recommend the most suitable crops for cultivation.
The goal is to predict the type of crop to be recommended, and the type of fertilizer which falls into distinct categories or classes. For instance, the output might include classes such as Wheat, Rice, Maize, etc. along with details for fertilizer required based on the soil type.This aligns with the definition of a classification problem where the target variable is categorical.

2. DATASET:

The dataset for this project is sourced from a comprehensive agricultural study and includes key parameters influencing crop growth.
This data will be used to train and validate our crop recommendation model.
The training and testing data set is obtained from Kaggle Dataset Crops Recommendation dataset:
Case Study on Kaggle Competition : Crop Recommendation Dataset | Kaggle
Fertilizers Recommendation dataset: Github:Yash Thorbole

DATA FIELDS

N - ratio of Nitrogen content in soil
P - ratio of Phosphorous content in soil
K - ratio of Potassium content in soil
temperature - temperature in degree Celsius
humidity - relative humidity in %
ph - ph value of the soil
rainfall - rainfall in mm
fertilizer - There are 7 unique types of fertilizers in the dataset.

3. EXPLORATORY DATA ANALYSIS:

3.1 Descriptive Statistics

- **Continuous Variables**

N, P, K, temperature, humidity, ph, rainfall are all continuous variables.
Nitrogen (N): Ranges from 0 to 140 with a mean of around 50.55.
Phosphorus (P): Ranges from 5 to 145 with a mean of approximately 53.36.
Potassium (K): Has a wide range from 5 to 205, average near 48.15.
Temperature: Varies from 8.83°C to 43.68°C, average around 25.62°C.
Humidity: Ranges widely from 14.26% to nearly 100%, with an average of 71.48%.
pH: Varies from 3.50 to 9.94, with a mean value close to 6.47, which is slightly acidic.
Rainfall: Ranges from 20.21 mm to 298.56 mm, with an average of 103.46 mm.

Fertilizer:
Urea: Contains 37% Nitrogen, 0% Potassium, and 0% Phosphorous.
DAP (Diammonium phosphate): It contains 12% Nitrogen, 0% Potassium, and 36% Phosphorous.
Fourteen-Thirty Five-Fourteen: It contains 7% Nitrogen, 9% Potassium, and 30% Phosphorous.
Twenty Eight-Twenty Eight: It contains 22% Nitrogen, 0% Potassium, and 20% Phosphorous.
Seventeen-Seventeen-Seventeen: Contains 17% Nitrogen, 17% Potassium, and 17% Phosphorous.
Ten-Twenty Six-Twenty Six: Comprises 10% Nitrogen, 26% Potassium, and 26% Phosphorous.

- **Categorical Variables for crop Recommendation**

Label (Crop Type): There are 22 unique types of crops in the dataset.

	N	P	K	temperature	humidity	ph	rainfall
count	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000
mean	50.551818	53.362727	48.149091	25.616244	71.481779	6.469480	103.463655
std	36.917334	32.985883	50.647931	5.063749	22.263812	0.773938	54.958389
min	0.000000	5.000000	5.000000	8.825675	14.258040	3.504752	20.211267
25%	21.000000	28.000000	20.000000	22.769375	60.261953	5.971693	64.551686
50%	37.000000	51.000000	32.000000	25.598693	80.473146	6.425045	94.867624
75%	84.250000	68.000000	49.000000	28.561654	89.948771	6.923643	124.267508
max	140.000000	145.000000	205.000000	43.675493	99.981876	9.935091	298.560117

- **Categorical Variables for fertilizer Recommendation**

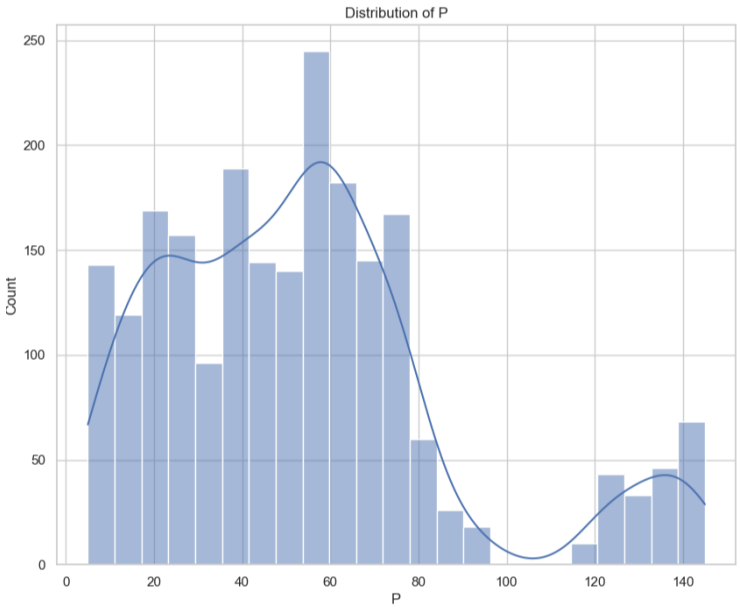
Fertilizer Name (fertilizer Type): There are 7 unique types of fertilizer in the dataset.

	Nitrogen	Potassium	Phosphorous	Fertilizer Name
0	37	0	0	Urea
1	12	0	36	DAP
2	7	9	30	Fourteen-Thirty Five-Fourteen
3	22	0	20	Twenty Eight-Twenty Eight
4	35	0	0	Urea

3.2(a) Data Distributions for *Crop* Data

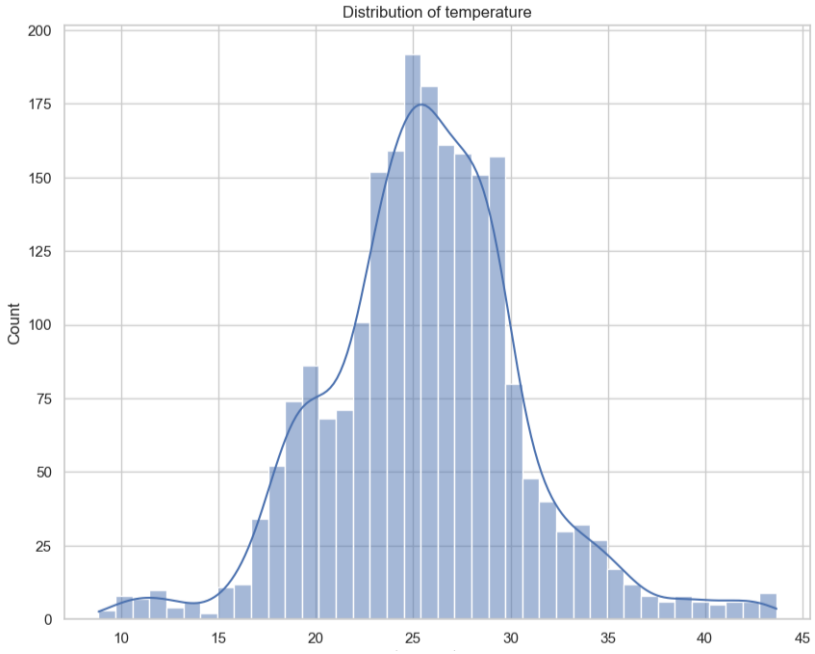
Continuous Variables

The histograms show the distributions of each continuous variable:

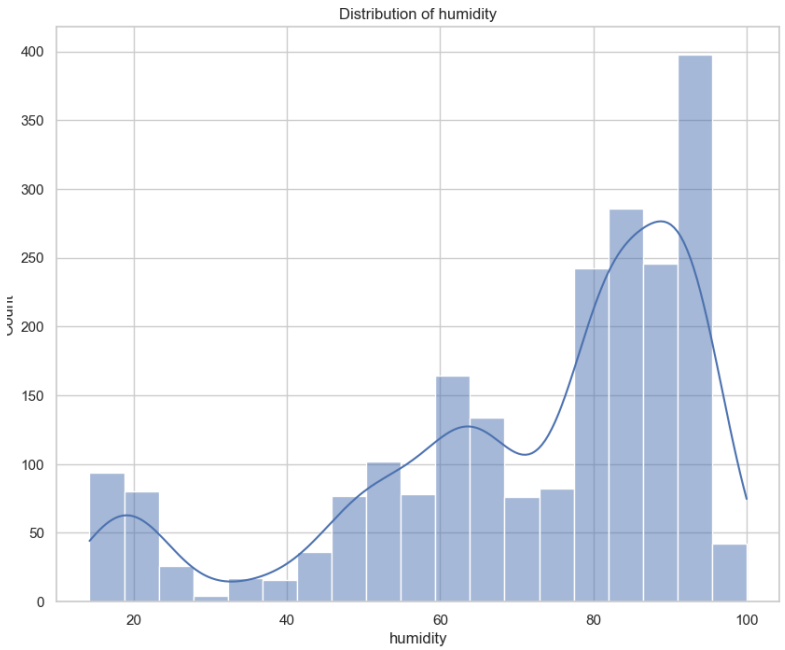


N, P, K: Some display a bimodal nature (having two peaks), suggesting different groups in the data.

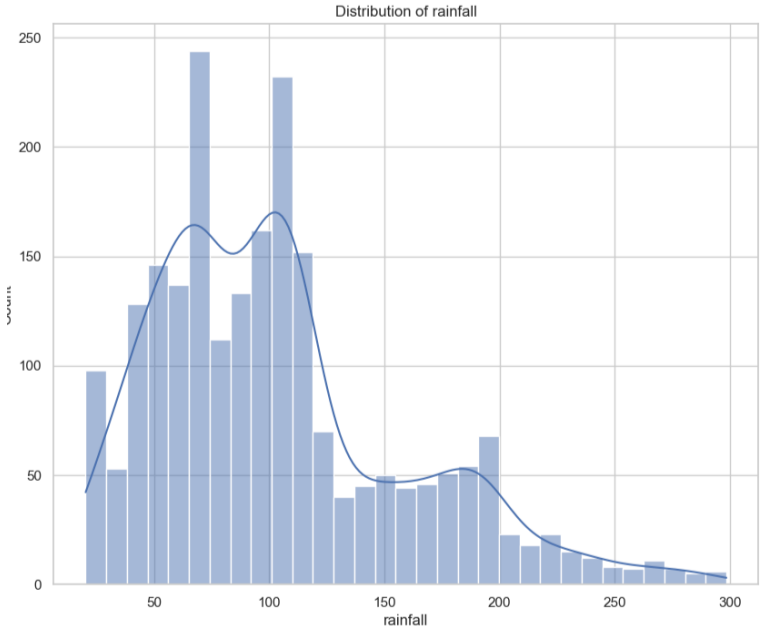
Temperature: Appears to be normally distributed.



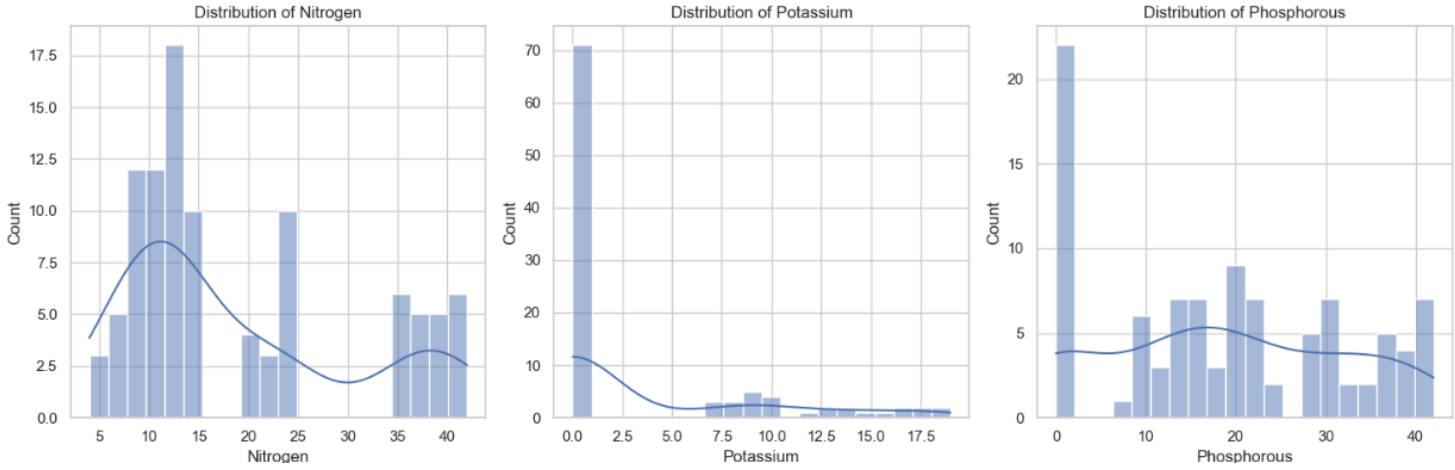
Humidity: Shows left-skewed distribution, with a high frequency of values towards the higher end.



Rainfall: Displays right-skewed distribution, indicating higher rainfall amounts are less common.



3.2(b) Data Distributions for *Fertilizer* Data



OBSERVATIONS:

1. Nitrogen:

The histogram exhibits a multimodal distribution with several peaks, suggesting multiple common values of Nitrogen in the dataset.
The line shows these modes as peaks in the probability density, indicating clusters of data points.

2. Potassium:

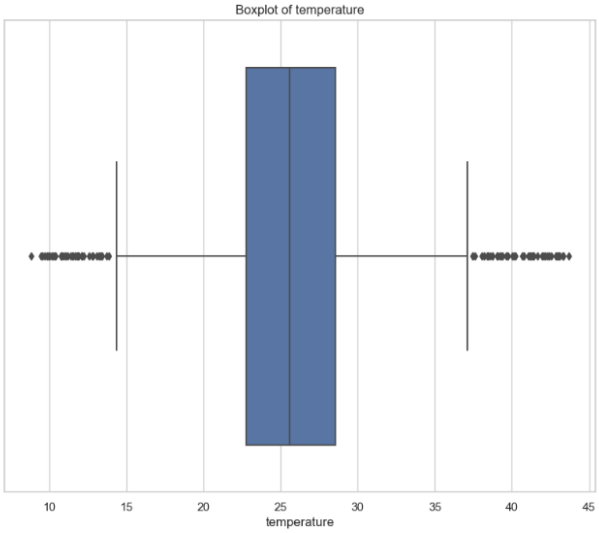
The distribution of Potassium is highly skewed towards the lower end, with a sharp peak at the lowest bin. This skewness is evident in the curve, which has a steep drop-off as the values increase.

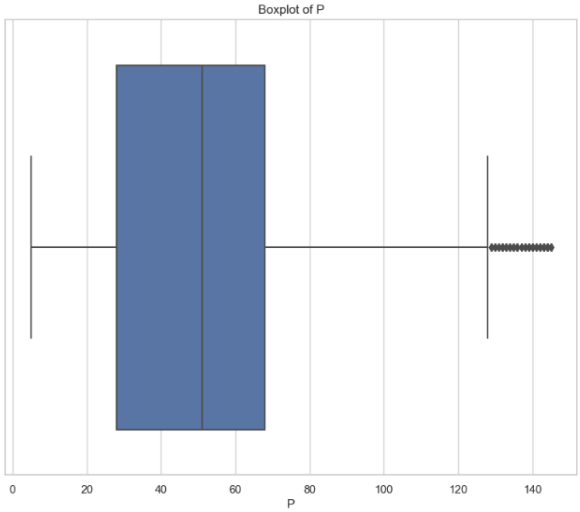
3. Phosphorous:

Phosphorous levels are more evenly spread across the range, with a slight concentration at the lower end. The curve for Phosphorous is flatter than that of Potassium, suggesting less skewness

3.3(a) Outlier Detection for *Crop* Recommendation

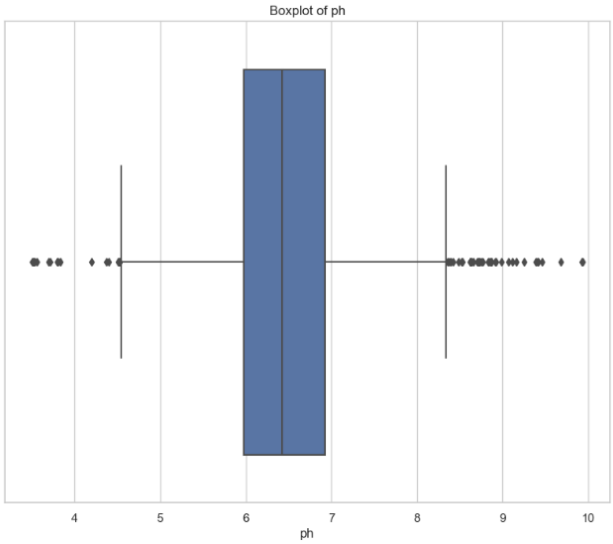
The boxplots provide insights into potential outliers in the continuous variables:

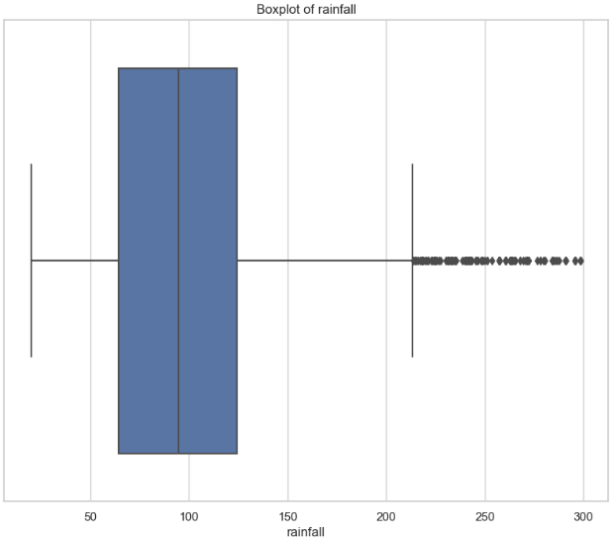




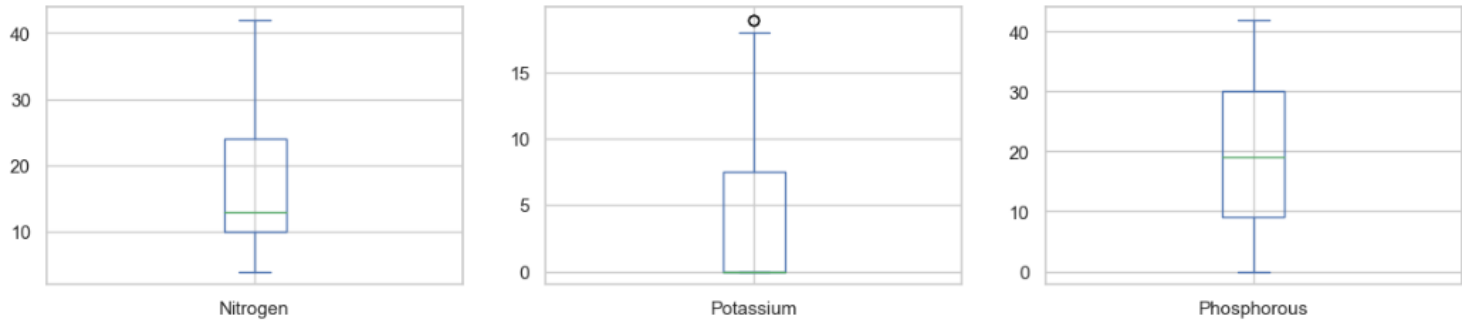
Soil Nutrients (N, P, K) : These features show some outliers, particularly on the higher end. This could be due to specific crops requiring significantly different nutrient levels.

Temperature: Few outliers are observed, particularly on the lower end.





3.3(b) Outlier Detection for *Fertilizer* Recommendation



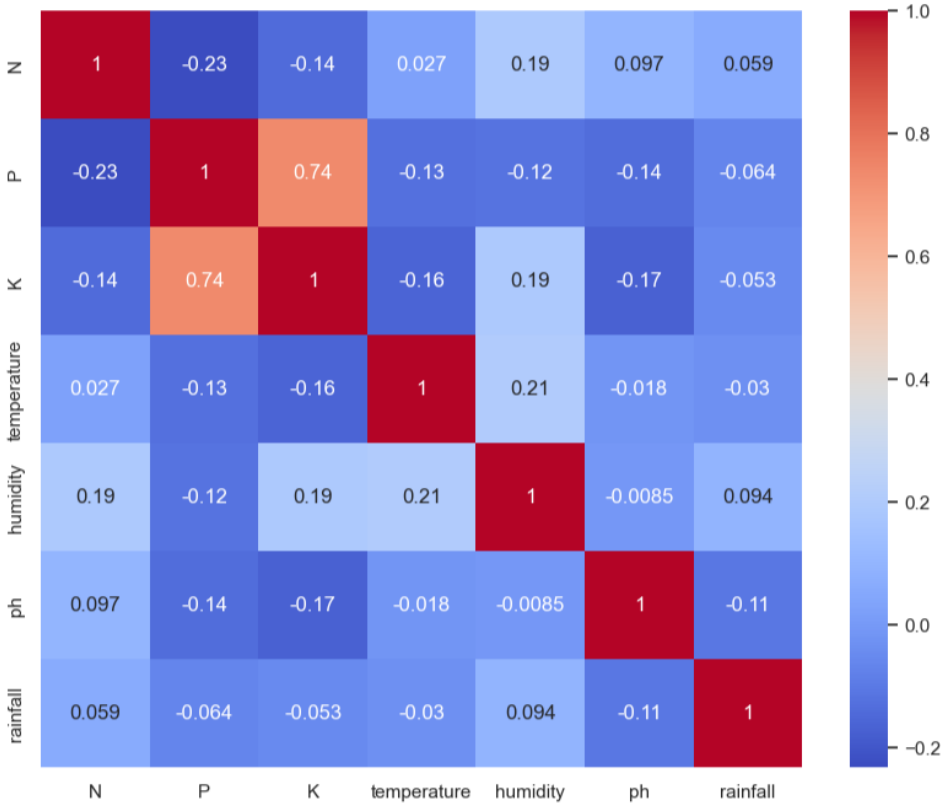
Nitrogen Box Plot:
The interquartile range (IQR), represented by the box, encapsulates the middle 50% of the Nitrogen data. The median is the central line in the box, dividing the IQR into two equal parts.
Whiskers extend from the hinges of the box to the highest & lowest values within 1.5 * IQR.

Potassium Box Plot:
The box plot for Potassium shows a similar IQR and median.
An outlier is noticeable, marked by a circle above the upper whisker, indicating an unusual value that stands apart from the rest of the data.

Phosphorous Box Plot:
The IQR and median for Phosphorous are displayed in a similar fashion to the other nutrients.
The distribution of Phosphorous levels appears relatively symmetric around the median.

3.4(a) Correlation Analysis for *Crop* prediction

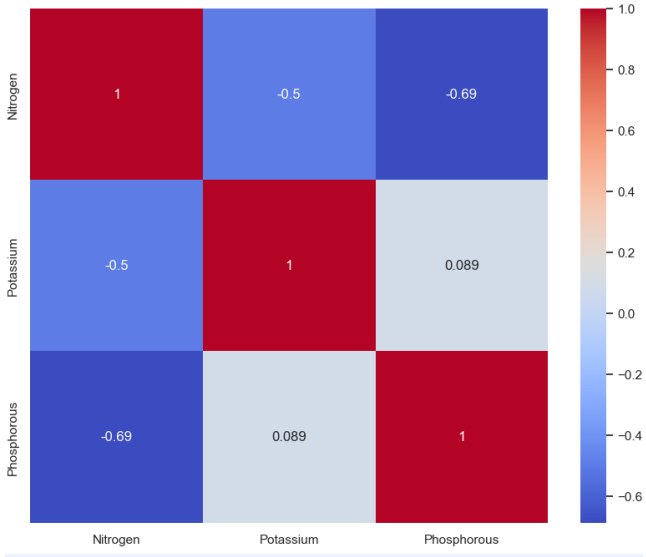
The correlation values range from -1 to 1, where
1 indicates a perfect positive correlation,
-1 indicates a perfect negative correlation, and
0 indicates no correlation between the columns.



The heatmap displays the correlation coefficients between the continuous variables:

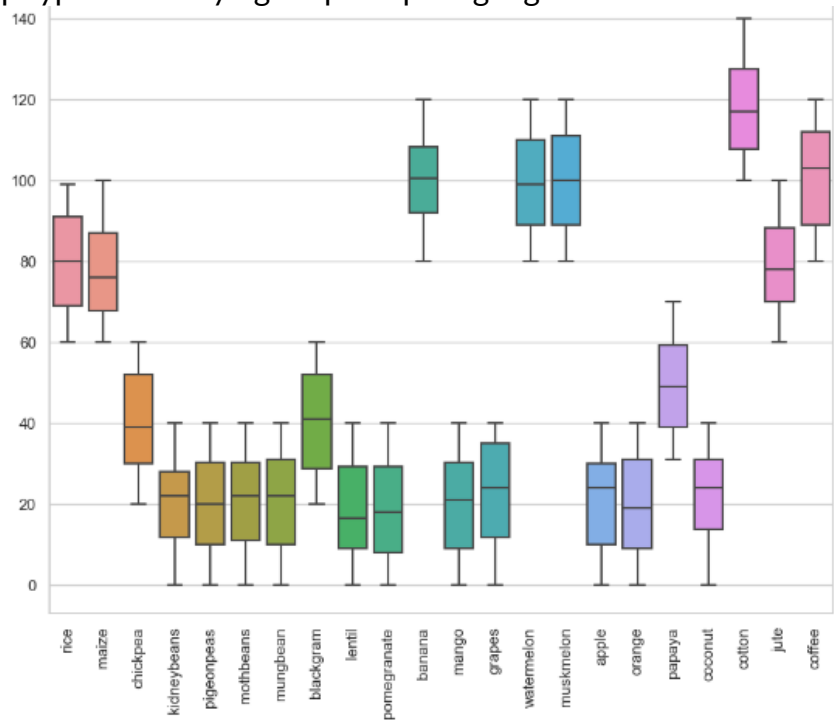
Strong Correlations: There aren't any extremely strong correlations (> 0.8 or < -0.8) observed, which is generally a positive sign for building machine learning models
Moderate Correlations: Some moderate correlations are noted. For example, temperature and humidity show a moderate negative correlation (higher temperatures and lower humidity levels).
Weak Correlations: Most variables show weak correlations with each other, indicating that each provides unique information for the crop recommendation.

3.4(b) Correlation Analysis for *Fertilizer* prediction



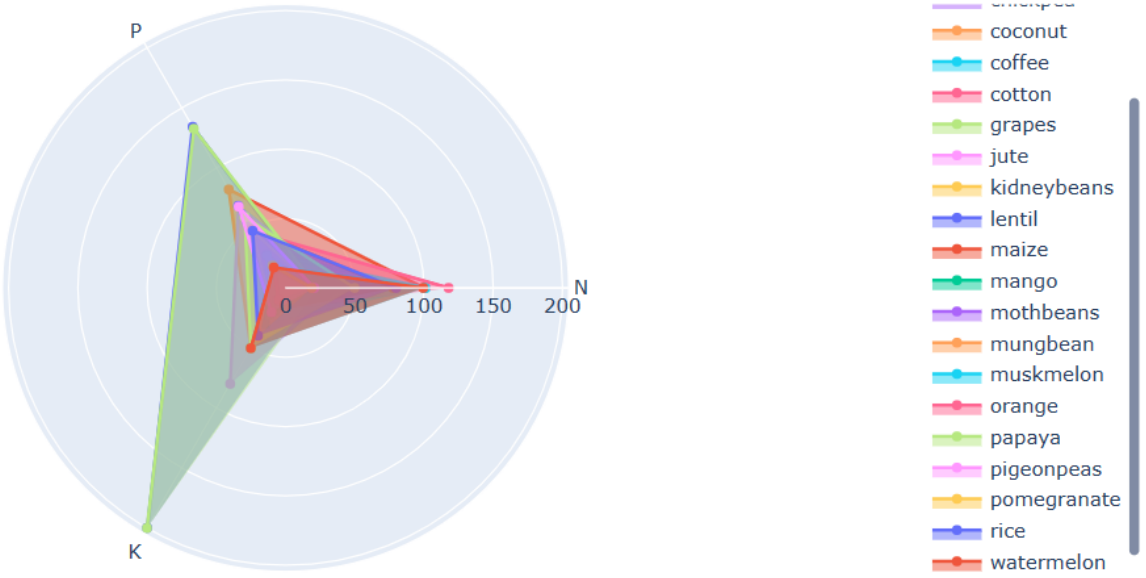
In this case, we can see that there is a negative correlation between nitrogen and potassium, and between nitrogen and phosphorous. This means that as the amount of nitrogen increases, the amount of potassium and phosphorous tends to decrease. A positive correlation between potassium & phosphorous, as potassium increases, the amount of phosphorous tends to increase.

Bivariate Analysis with Data Visualization for crop Prediction
Bivariate analysis is crucial in understanding the relationship between two different variables. Box plots are used to understand the distribution of a particular variable, such as Nitrogen content, across different crop types. Identifying crops requiring higher or lower levels of certain nutrients.



RADAR CHART FOR NUTRIENT LEVELS BY CROP

1. Each crop label is represented by a filled radar area, showing the average values of nutrients.
2. For comparing multiple quantitative variables (nutrient levels) across different categories.
3. The radial axis represents nutrients ('N', 'P', 'K'), and the angular axis represents each nutrient.
4. Filled areas enclosed by the lines for each crop label indicate the average nutrient levels,



4. FEATURE ENGINEERING:

4.1 (a) Categorical variables for *Crop* Recommendation

Converting Categorical varibales to a integer format

```
crop_dict = {
    'rice': 1,
    'maize': 2,
    'jute': 3,
    'cotton': 4,
    'coconut': 5,
    'papaya': 6,
    'orange': 7,
    'apple': 8,
    'muskmelon': 9,
    'watermelon': 10,
    'grapes': 11,
    'mango': 12,
    'banana': 13,
    'pomegranate': 14,
    'lentil': 15,
    'blackgram': 16,
    'mungbean': 17,
    'mothbeans': 18,
    'pigeonpeas': 19,
    'kidneybeans': 20,
    'chickpea': 21,
    'coffee': 22
}
crop['crop_num']=crop['label'].map(crop_dict)
```

Scale the features using MinMaxScaler

MinMaxScaler is a feature scaling technique that normalizes each feature to a specified range, typically [0, 1]. It does this by subtracting the minimum value of the feature and then dividing by the range (the maximum value minus the minimum value).

Why Use MinMaxScaler?

1. Normalizing Measurement Scales: In crop recommendation datasets, features like temperature, humidity, and soil pH can have different scales and units. MinMaxScaler ensures that these features with varying ranges don't disproportionately influence model.

2. Improving Model Performance: Many machine learning algorithms perform better when data is on a similar scale. MinMaxScaler can help in faster convergence and improved performance, especially for algorithms like neural networks and k-nearest neighbors.

3. Maintaining Proportions: Unlike some scalers, MinMaxScaler preserves the shape of the original distribution, scaling data points uniformly without reducing importance of outliers.

```
: X_train
```

```
: array([[0.12142857, 0.07857143, 0.045     , ..., 0.9089898 , 0.48532225,
          0.29685161],
         [0.26428571, 0.52857143, 0.07      , ..., 0.64257946, 0.56594073,
          0.17630752],
         [0.05      , 0.48571429, 0.1       , ..., 0.57005802, 0.58835229,
          0.08931844],
         ...,
         [0.07857143, 0.22142857, 0.13      , ..., 0.43760347, 0.46198144,
          0.28719815],
         [0.07857143, 0.85      , 0.995     , ..., 0.76763665, 0.44420505,
          0.18346657],
         [0.22857143, 0.52142857, 0.085     , ..., 0.56099735, 0.54465022,
          0.11879596]])
```

Importance of Standardization for Your Crop Recommendation Project
In the context of a crop recommendation project, standardizing features like temperature, rainfall, and pH levels is crucial due to their varying units and scales. Standardization ensures that all these features contribute equally to the model's predictions, preventing any single feature from dominating due to its variance or unit.

```
: X_train
```

```
: array([[-9.03426596e-01, -1.12616170e+00, -6.68506601e-01, ...,
           9.36586183e-01,  1.93473784e-01,  5.14970176e-03],
         [-3.67051340e-01,  7.70358846e-01, -5.70589522e-01, ...,
          -1.00470485e-01,  8.63917548e-01, -6.05290566e-01],
         [-1.17161422e+00,  5.89737842e-01, -4.53089028e-01, ...,
          -3.82774991e-01,  1.05029771e+00, -1.04580687e+00],
         ...,
         [-1.06433917e+00, -5.24091685e-01, -3.35588533e-01, ...,
          -8.98381379e-01, -6.34357580e-04, -4.37358211e-02],
         [-1.06433917e+00,  2.12501638e+00,  3.05234239e+00, ...,
           3.86340190e-01, -1.48467347e-01, -5.69036842e-01],
         [-5.01145154e-01,  7.40255346e-01, -5.11839275e-01, ...,
          -4.18045489e-01,  6.86860180e-01, -8.96531475e-01]])
```

5. MODEL SELECTION & DEPLOYMENT:

5.1 (a) Model Implementation for *Crop* Recommendation

Performance of multiple classifiers on a given dataset based on their accuracy scores

```
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import ExtraTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracy_score

# create instances of all models
models = {
    'Logistic Regression': LogisticRegression(),
    'Naive Bayes': GaussianNB(),
    'Support Vector Machine': SVC(),
    'K-Nearest Neighbors': KNeighborsClassifier(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Bagging': BaggingClassifier(),
    'AdaBoost': AdaBoostClassifier(),
    'Gradient Boosting': GradientBoostingClassifier(),
    'Extra Trees': ExtraTreeClassifier(),
}


for name, md in models.items():
    md.fit(X_train,Y_train)
    ypred = md.predict(X_test)

    print(f"{name}  with accuracy : {accuracy_score(Y_test,ypred)}")
```

```
Logistic Regression  with accuracy : 0.9636363636363636
Naive Bayes  with accuracy : 0.9954545454545455
Support Vector Machine  with accuracy : 0.9681818181818181
K-Nearest Neighbors  with accuracy : 0.9590909090909091
Decision Tree  with accuracy : 0.9863636363636363
Random Forest  with accuracy : 0.9931818181818182
Bagging  with accuracy : 0.9818181818181818
AdaBoost  with accuracy : 0.1409090909090909
Gradient Boosting  with accuracy : 0.9818181818181818
Extra Trees  with accuracy : 0.875
```

---

1. Naive Bayes (Accuracy: 99.55%)
High Accuracy: With an accuracy of 99.55%, Naive Bayes shows excellent performance in classifying crop types.
Probability-Based: As a probabilistic classifier, Naive Bayes is effective in making predictions based on the likelihood of various outcomes, which is valuable in crop recommendation where multiple factors influence the result.

2. Decision Trees (Accuracy: 98.64%)
Interpretability: Decision Trees provide a clear visualization of the decision-making process, making it easier to understand how different features contribute to the final recommendation.
Handling Non-Linear Relationships: They are capable of capturing complex, non-linear relationships between features, which is common in agricultural datasets.

3. Random Forest (Accuracy: 99.32%)
Robustness: Random Forest, an ensemble of Decision Trees, is more robust and less prone to overfitting compared to a single Decision Tree.
Handling Large Datasets: It excels in handling large datasets with many features, making it ideal for comprehensive agricultural data.

5.1.1 NAÏVE BAYES

Handling Continuous Data: GaussianNB is particularly effective when dealing with continuous data. It assumes that the continuous values associated with each feature are distributed according to a Gaussian distribution (normal distribution). This is relevant in agricultural datasets where many features such as temperature, rainfall, and pH levels are continuous and can be assumed to follow a Gaussian distribution.

Fast Model Training and Prediction: Given the potentially large size of agricultural data, the speed of training and prediction is crucial. Naive Bayes provides a faster solution compared to more complex models, making it ideal for rapid analysis and real-time decision-making in crop recommendations.

Good Performance with Small Datasets: Even with a smaller amount of data, Naive Bayes can perform quite well, making it a good choice for projects where the amount of data may be limited.

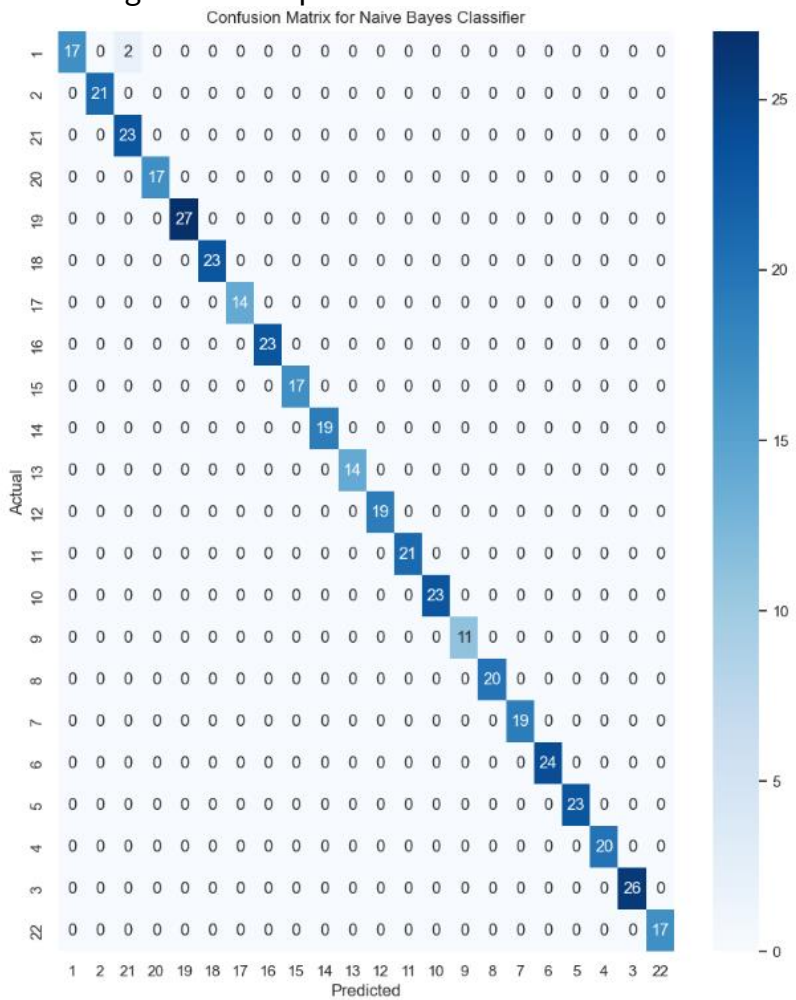
```
Precision: 0.9958181818181817
Recall: 0.9954545454545455
F1-score: 0.9954229797979798
Accuracy: 0.9954545454545455
```

ANALYSIS

High Values on the Diagonal: Indicate good performance for specific crop types.
High Values Off-Diagonal: For example, if a high number appears in the row for crop A and the column for crop B, it means crop A is often misclassified as crop B.

MODEL IMPROVEMENT

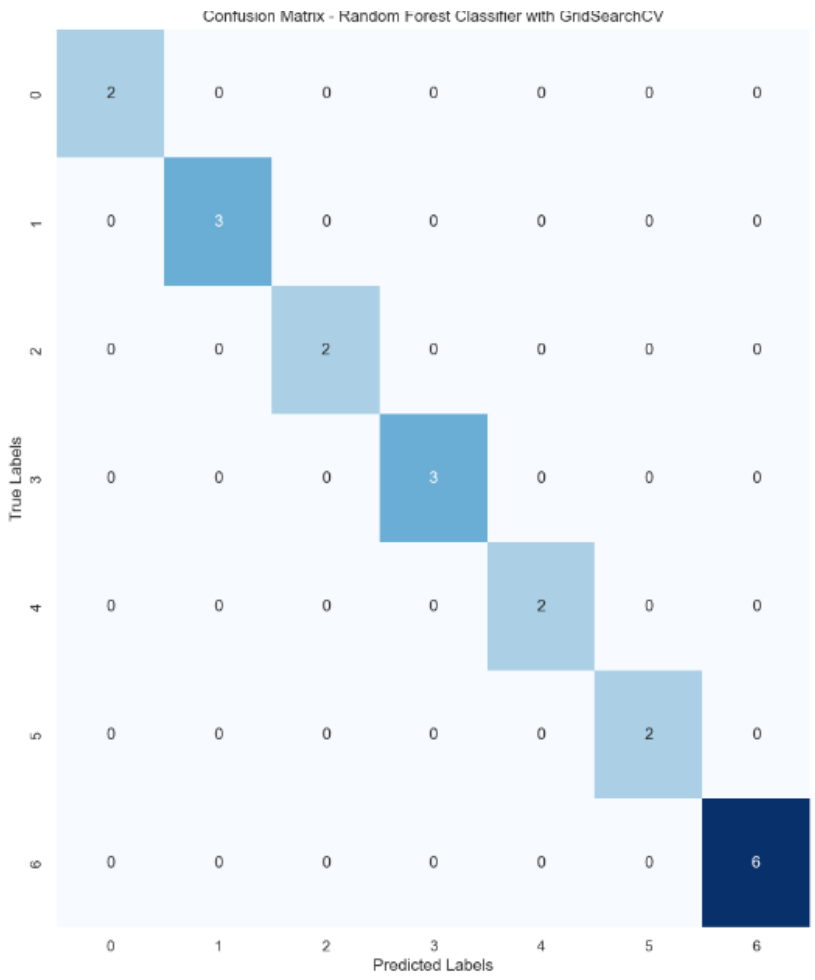
Certain crops are consistently misclassified, Misclassifications can guide you in refining the features used for training or in tweaking the model parameters.



5.1(b) Model Implementation for *Fertilizer* Recommendation

5.1.2 RANDOM FOREST CLASSIFIER

Robustness and Versatility: Random Forest is a robust and versatile ensemble learning method, suitable for both classification and regression tasks.
Handling of Complex Interactions: Random Forest can capture complex interactions between features, which is often the case in agricultural datasets where factors such as nutrient levels and soil conditions may interact in complex ways to influence fertilizer requirements.
Reduction of Overfitting: By utilizing multiple decision trees, Random Forest reduces the risk of overfitting, which can be a common problem with single decision trees.
Feature Importance: An inherent benefit of Random Forest is its ability to rank the importance of different features in prediction. In the context of fertilizer recommendation, the model can identify which soil nutrients or conditions are most predictive of the need for a particular type of fertilizer.
ANALYSIS
Predominance of Zero Off-Diagonal Values: Many off-diagonal cells have zero values, which suggests that there are no misclassifications between many pairs of classes.
Concentration of Errors: The errors are not spread out but concentrated between specific classes, which can indicate similar feature patterns for these classes causing confusion for the model.
MODEL IMPROVEMENT
To further improve the model, we should examine the feature importance given by the Random Forest and consider collecting more data for the misclassified classes or reevaluating the features that lead to confusion.
Another approach could be to look into more complex model architectures or feature engineering techniques that can capture the nuances between the classes that are being confused.



6. ADVANCED MODEL SELECTION

6.1(a) Model Implementation for *Crop* Recommendation

6.1.1 NEURAL NETWORKS

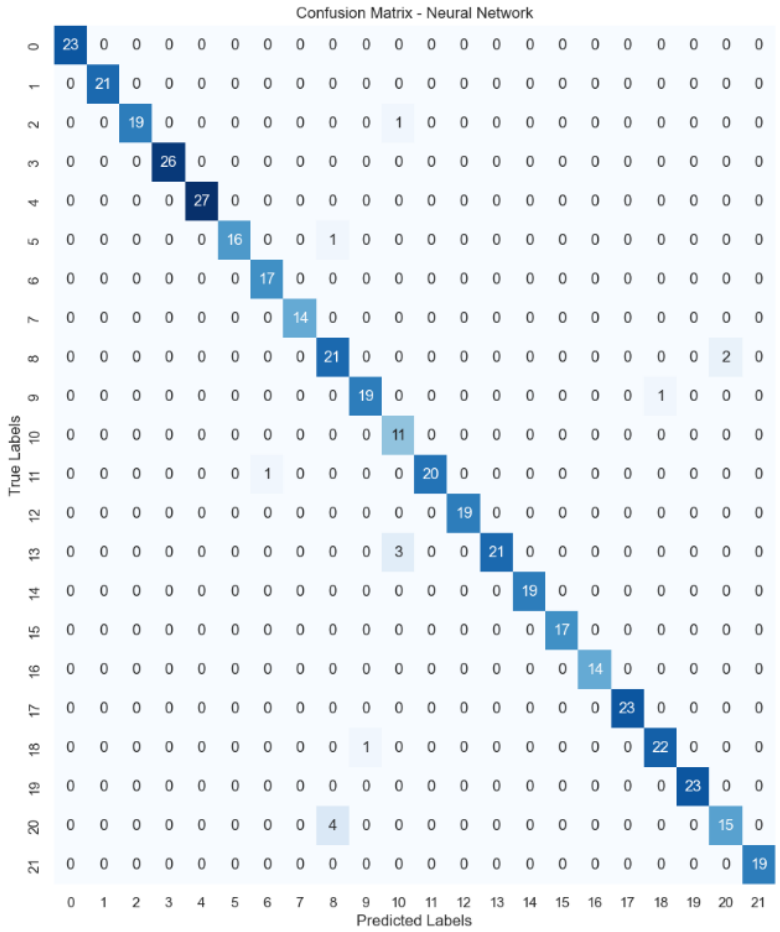
Neural Networks are particularly suited for crop and fertilizer recommendation because they can model complex, non-linear relationships in the data.
They excel in handling large datasets with many features, which is often the case in agricultural datasets where various environmental and soil factors interact in complex ways.

```
Accuracy: 0.9681818181818181
Precision: 0.9715087526852233
Recall: 0.9681818181818181
F1-score: 0.9687032498174405
```

Accuracy (0.9681): Indicates that the model correctly classified about 96.81% of the crop types. High accuracy is crucial for ensuring reliable crop recommendations.

ANALYSIS
High Values on the Diagonal: Such values illustrate strong predictive performance for specific crop types by the Neural Network model.
High Values Off-Diagonal: Notably, there are some off-diagonal values, such as a count of 4 for the predicted label 21 when the actual label was 27, indicating a case of misclassification.

MODEL IMPROVEMENT
The presence of misclassifications, although relatively low, indicates potential areas for improvement. It may be necessary to delve deeper into the features correlated with those specific crops to understand why misclassifications occurred. Enhancing the feature set, performing further feature engineering, or adjusting the model's hyperparameters could reduce these errors.



6.1(b) Model Implementation for *Fertilizer* Recommendation

6.1.2 NEURAL NETWORKS

Capability to Model Non-linear Relationships: Neural Networks, particularly MLPClassifier with 'relu' activation function and 'adam' solver, excel in modeling non-linear and complex relationships that are often present in agricultural datasets.

F1 Score: An F1 score of 0.966 is particularly impressive as it represents the harmonic mean of precision and recall. This high F1 score implies a balanced model that maintains both high precision and recall, indicating fewer misclassifications and missed cases, which is crucial for making accurate fertilizer recommendations.

ANALYSIS

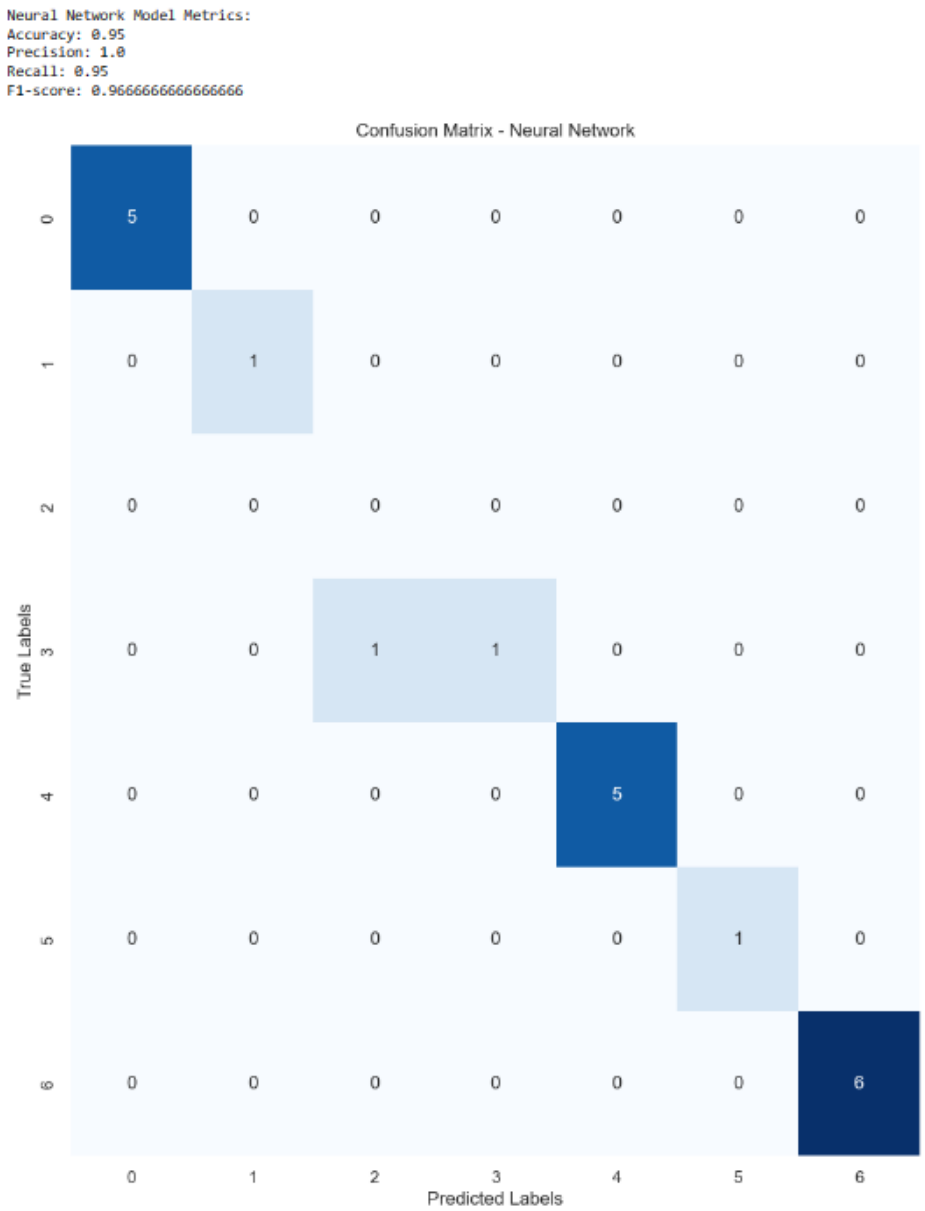
Predominant Diagonal Values: The high values on the diagonal for certain classes suggest that the model is particularly effective at correctly classifying these classes.

Sparse Off-Diagonal Values: The presence of few off-diagonal values indicates that there are relatively few misclassifications overall, which is a positive indicator of model performance.

MODEL IMPROVEMENT

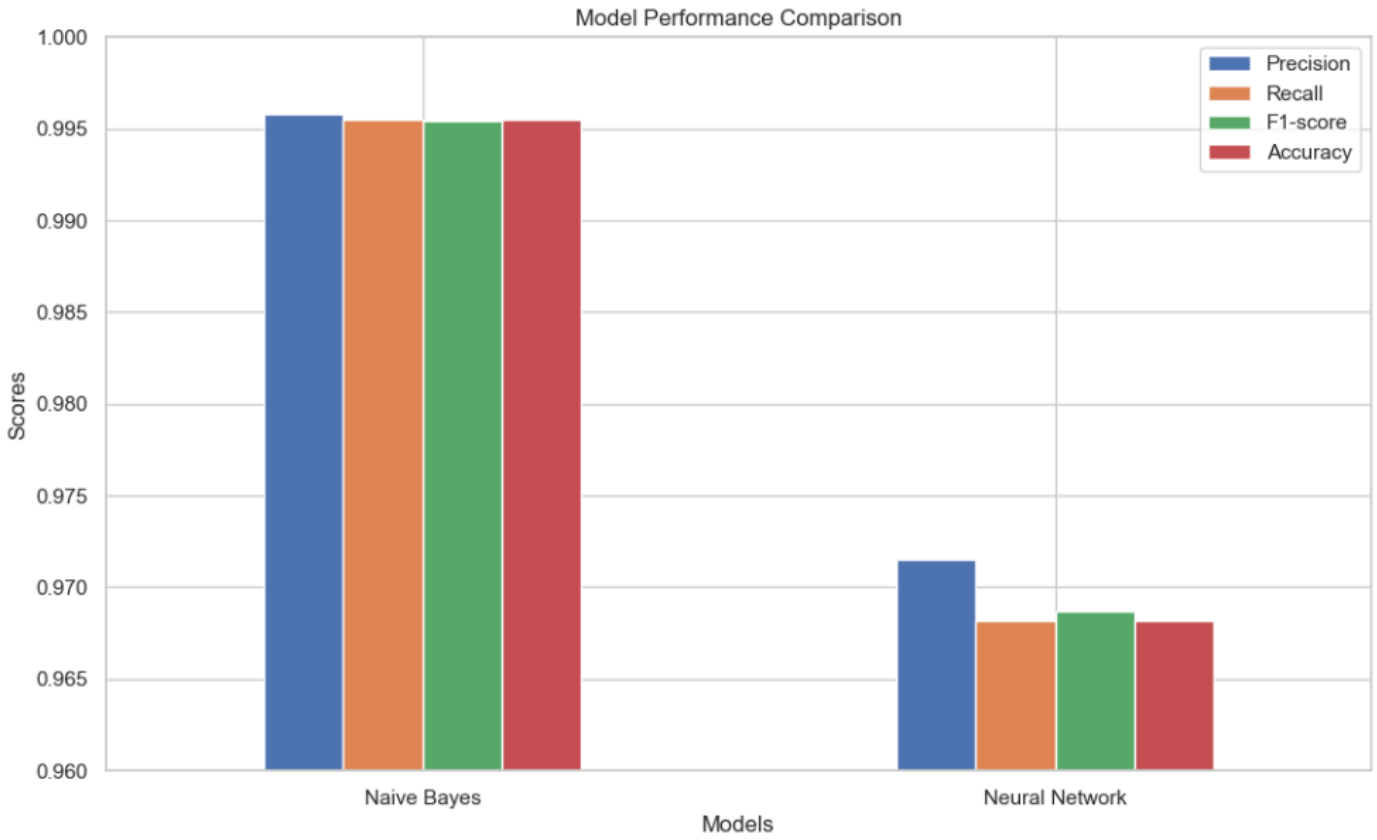
The misclassifications that do occur provide opportunities for model improvement

Enhancements might include feature engineering to better distinguish between these classes, collecting more representative data for the underperforming classes, or adjusting the model



7. RESULT:

7.1 (a) Results for Error calculations *Crop* Recommendation



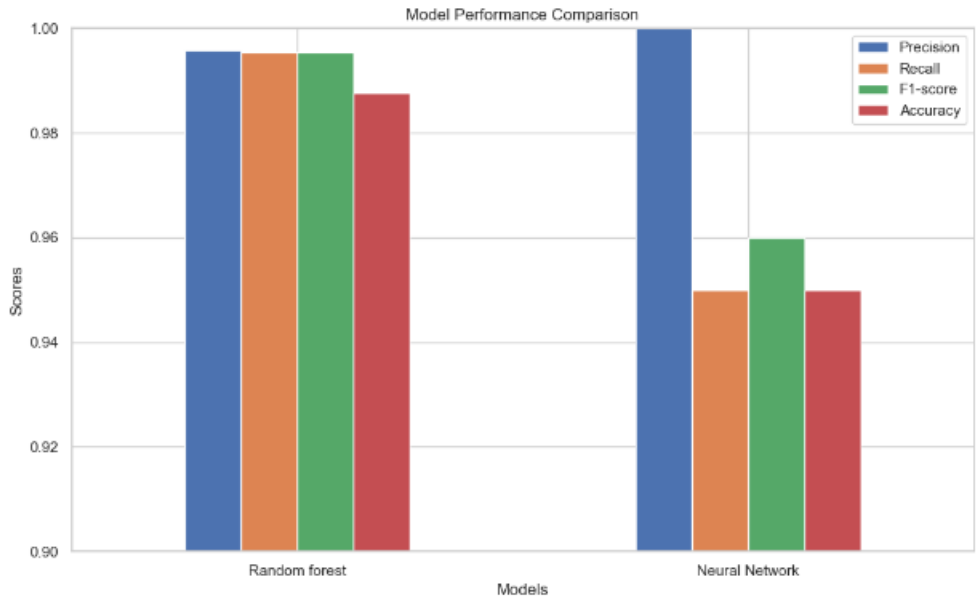
Explanation and Interpretation of Model Performance

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positives. High precision indicates a low false positive rate. Naive Bayes has the highest precision, suggesting it's best at correctly identifying crops without labeling many incorrectly.

Recall: Recall (Sensitivity) is the ratio of correctly predicted positive observations to all observations in the actual class. Naive Bayes again scores highest, indicating it is most capable of finding all relevant cases (all suitable crop types).

F1-score: The F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. The highest F1-score of Naive Bayes implies a balance between precision and recall.
Accuracy: This is the ratio of correctly predicted observation to the total observations. Naive Bayes has the highest accuracy, indicating it correctly identifies crop types most often.

7.1 (b) Results for Error calculations *Fertilizer* Recommendation



PRECISION:
Random Forest: Precision of 0.9958 suggests a very low rate of false positives. This model is excellent at correctly identifying the right fertilizer types without many errors.

Neural Network: A perfect precision score of 1.0 indicates no false positives in its predictions, which is exceptional.

RECALL:
Random Forest: The recall of 0.9955 indicates that this model is almost perfect in identifying all relevant instances of the correct fertilizer types.

Neural Network: A recall of 0.95, while still high, suggests it misses a few relevant cases compared to Random Forest.

F1-SCORE:
Random Forest: The F1 score of 0.9954 is very high, showing a strong balance between precision and recall.

Neural Network: An F1 score of 0.96, slightly lower than Random Forest, indicates a marginally less balanced performance in precision and recall.

ACCURACY:
Random Forest: An accuracy of 0.9876 means it correctly identifies the right fertilizer types in most cases.

Neural Network: The accuracy of 0.95 is high, but lower than Random Forest, suggesting a slightly reduced overall prediction capability.

7.2 (a) Results for Computational *Crop* Recommendation

```
Naive Bayes training time: 0.0348 seconds
Naive Bayes memory usage: 0.0092 MB; Peak: 0.1701 MB
Naive Bayes Log Loss: 0.0165


Neural Network training time: 7.2994 seconds
Neural Network memory usage: 0.2391 MB; Peak: 0.8956 MB
Neural Network Log Loss: 0.0854
```

Comparative Analysis of Naive Bayes and Neural Network Models
In our crop recommendation project, we conducted a comprehensive comparison between two machine learning models: Naive Bayes and Neural Network. This comparison extends beyond standard metrics like accuracy, precision, recall, and F1 score, to provide a deeper understanding of the practical implications of each model.

TRAINING TIME COMPARISON
The time taken to train a model is crucial, especially in large datasets or when frequent retraining is necessary.
Observations: Naive Bayes, typically, requires significantly less training time compared to Neural Networks, making it efficient for scenarios demanding quick model updates or limited computational resources.

MEMORY USAGE EVALUATION
We evaluated the memory consumption during the training phase of each model, a critical aspect in resource-constrained environments.
Observations: Neural Networks often consume more memory due to their complex architecture, while Naive Bayes, with its simpler structure, has a lower memory footprint, suitable for deployment with restricted memory.

LOG LOSS ANALYSIS
Log loss measures the confidence of the predictions, penalizing false classifications more heavily if the model is very confident in its incorrect predictions.
Observations: A lower log loss is indicative of a model with reliable and confident predictions. This aspect is vital in applications like crop recommendations, where uncertain predictions can lead to significant consequences.

MODEL COMPLEXITY AND SIZE
We also considered the complexity and size of each model, as a simpler model with fewer parameters is easier to deploy, especially in environments with limited computational resources.
Observations: Naive Bayes, being fundamentally simpler, has fewer parameters and thus lower complexity. In contrast, Neural Networks, due to their deep architectures, are more complex and resource-demanding.

7.2 (b) Results for Computational *Fertilizer* Recommendation

```
Random Forest Classifier:
Random Forest Classifier Training Time: 0.8821 seconds
Random Forest Classifier Memory Usage: 0.2875 MB; Peak: 0.3337 MB
Random Forest Classifier Log Loss: 0.03315874044298863


Neural Network:
Neural Network Training Time: 0.2411 seconds
Neural Network Memory Usage: 0.1895 MB; Peak: 0.4129 MB
Neural Network Log Loss: 0.16810064239464428
```

RANDOM FOREST CLASSIFIER:

Training Time: The model took approximately 0.891 seconds to train.

Memory Usage: The training process consumed 0.2875 MB of memory, with a peak memory usage of 3.0337 MB.

Performance Metrics:
Precision, Recall, and F1-Score: These metrics are perfect (1.00) across all classes, which include various categories like 'Fourteen-Thirty Five-Fourteen' and 'Twenty-Eight-Twenty-Eight'. This suggests that the model has predicted every class with 100% accuracy without any false positives or false negatives.

Support: This column indicates the number of true occurrences of each class in the dataset. The classes have varying support, with some having only 1 instance and others having more, up to 5.

Macro Average: Averages the performance metrics for each class, and these are also perfect (1.00), indicating uniform excellence across all classes despite the imbalance in their representation.

Weighted Average: Takes into account the support for each class, and again, the metrics are perfect (1.00).

NEURAL NETWORK:

Training Time: The Neural Network model took significantly longer to train, with 2.8411 seconds.

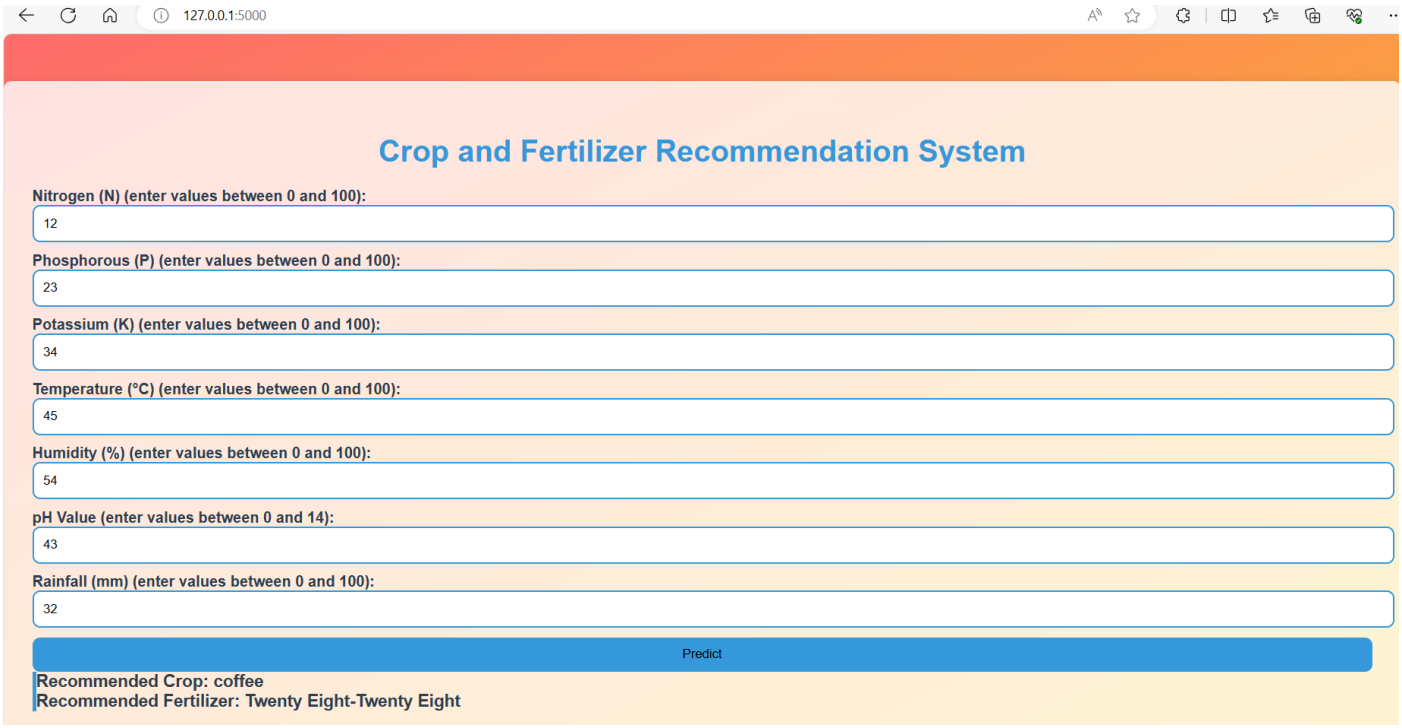
Memory Usage: It required more memory, with 0.1895 MB used and a peak of 4.4129 MB.

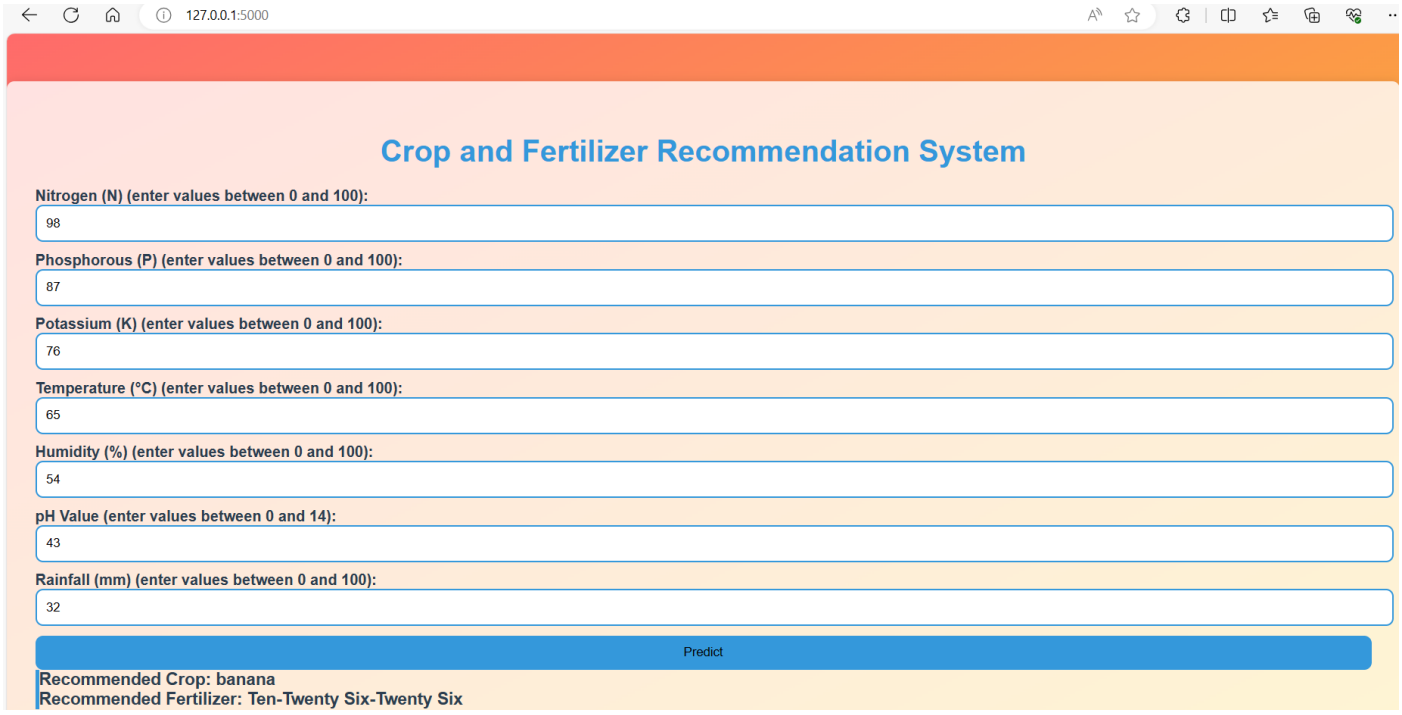
Performance Metrics:
Precision, Recall, and F1-Score: Like the Random Forest Classifier, the Neural Network also achieved perfect scores across all classes.

Support: The distribution of true occurrences is the same as for the Random Forest.

Macro and Weighted Averages: Both are perfect at 1.00.

8. FLASK APPLICATION:





9. STRENGTHS, WEAKNESS & IMPROVEMENTS:

9.1 (a) Analysis for *Crop* Recommendation

Model Performance Analysis for Crop and fertilizer Recommendation

The performance of various machine learning models on the Crop and fertilizer Recommendation project has been visualized in the bar chart, showing precision, recall, F1-score, and accuracy. Let's dive into a detailed analysis of each model's strengths, weaknesses, and potential improvements, and conclude with why Naive Bayes is outperforming the others.

Naive Bayes:

Strengths: Naive Bayes is simple, fast, and performs exceptionally well when the assumption of feature independence holds. It's particularly effective in high-dimensional spaces, which might be the case with our crop dataset.

Weaknesses: The assumption of feature independence rarely holds true in real-world data, which can limit its performance in some scenarios. Naive Bayes also struggles with zero-frequency problems where it assigns zero probability to unseen features/labels combinations.

Improvements: Applying smoothing techniques like Laplace estimation can help with zero-frequency problems. Feature engineering to reduce dependency among variables can also improve performance.

Neural Network:

Strengths: Neural Networks are highly flexible and can model complex non-linear relationships, making them suitable for the diverse and complex data typically found in crop and fertilizer recommendation datasets.

Weaknesses: They require a large amount of data to train and are not as interpretable as simpler models. They can also overfit if not properly regularized.

Improvements: Using dropout, regularization techniques, and proper validation strategies can help prevent overfitting. Neural architecture search can optimize the network structure.

Why Might Naive Bayes Perform Better?

The high performance of Naive Bayes suggests that the dataset likely has features that are relatively independent, a condition where Naive Bayes thrives. Its simplicity also helps to avoid overfitting, a problem that more complex models can sometimes face. Additionally, if the data has many categorical features or features following a probability distribution that Naive Bayes assumes, it can outperform other models.

9.1 (b) Analysis for *Fertilizer* Recommendation

Random Forest:

Strengths: Excellent for handling varied data types and complex relationships. Robust against overfitting due to ensemble nature.

Weaknesses: Can be computationally intensive. Interpretability is less straightforward than simpler models.

Improvements: Feature selection and hyperparameter tuning can enhance performance. Simplifying the model could improve interpretability and reduce computational load.

Neural Network

Strengths: Highly adaptable to complex, non-linear relationships. Exceptional in large datasets and diverse feature sets
.
Weaknesses: Prone to overfitting. Requires substantial data for training. Less interpretable.

Improvements : Regularization techniques and proper validation can reduce overfitting. More data and improved architecture could enhance performance.

Analysis of Model Performance

The superior performance of Random Forest in this context can be attributed to several factors:

Data Characteristics: The dataset might have features and relationships well-captured by the decision trees in Random Forest.

Overfitting Avoidance: Random Forest naturally avoids overfitting better than Neural Networks, especially if the dataset isn't massive.

Complexity Balance: Random Forest strikes a balance between handling complex relationships and not becoming too complex itself, unlike Neural Networks which can become overly complex.

The Neural Network's slightly lower scores might be due to overfitting, the need for more data, or complexity that isn't necessary for this specific dataset.

10. CONCLUSION:

Comprehensive Conclusion on Model Selection for Crop and Fertilizer Recommendation

Model Suitability: Model choice should be tailored to the dataset's unique traits and the project's objectives. Random Forest offers a balanced approach with high accuracy and low risk of overfitting, suitable for varied scenarios.

Random Forest Advantages: It handles complex data relationships effectively, with less computational demand, making it ideal for agricultural datasets where performance and usability are key.

Neural Networks Considerations: While Neural Networks are adept at managing complex datasets, they require more data and meticulous hyperparameter tuning to fully leverage their capabilities.

Naive Bayes Appropriateness: For datasets with independent features and possibly small or imbalanced data, Naive Bayes is efficient and simple, though caution is advised for more complex datasets where feature independence is not assured.

Cross-Validation Importance: Ensuring that the model performs well on unseen data through cross-validation is crucial, affecting the choice between simpler models like Naive Bayes and more complex ones like Neural Networks.

Final Decision Factors: The decision on the optimal model, be it Naive Bayes or Neural Networks, hinges on a balance between predictive performance, model transparency, and the dataset's complexity.

11. ACKNOWLEDGEMENT & REFERENCES:

Majority of work has been divided among team members
Nainil worked on the data preprocessing and EDA along with feature engineering.
Additionally, Model implementation including Naïve Bayes and NN was carried out by Nainil
Nainil built a flask framework to use model for training of web model.

Simran worked on some feature engineering, data visualization and carried out model implementation using Random Forest and Neural Networks
Simran made html interface with input from user for interactive application for crop and fertilizer recommendation

References

1. Recommender System lecture python notebook by Prof. Junwei Huang
2. [Crop Recommendation using ML] (https://ieeexplore.ieee.org/document/9734173)
3. Kaggle : (https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset)
4. Fertilizer Kaggle: (Plant Disease Classification - ResNet- 99.2% | Kaggle)
5. Microssoft Bing AI chat