Report on Crop Recommendation Model

1. Problem Statement

Farmers often face challenges in selecting the right crop based on soil conditions (N, P, K) and climate factors (temperature, humidity, rainfall). The objective of this project is to build a model that predicts the most suitable crop based on these factors, helping farmers optimize crop selection for better yields.

2. Data Collection and Description

The dataset used in this project contains several features related to soil conditions and climate factors, and the target variable is the crop type (label). The features are:

- **N**: Nitrogen content in the soil.
- **P**: Phosphorus content in the soil.
- **K**: Potassium content in the soil.
- **Temperature**: Average temperature in the region (in °C).
- **Humidity**: Average humidity percentage in the region.
- **pH**: Soil pH level.
- **Rainfall**: Amount of rainfall in the region (in mm).
- **Label**: The target variable representing the crop type (e.g., rice, maize, cotton).

3. Approach

The approach applied in this project consists of the following steps:

1. **Data Preprocessing**:
    - The dataset was first explored and preprocessed to handle any missing or inconsistent values (although, in this case, no specific missing data was mentioned).
    - The features `N`, `P`, `K`, `temperature`, `humidity`, `ph`, and `rainfall` were extracted as input features, and the `label` column was used as the target variable.
    - The `LabelEncoder` from scikit-learn was used to convert the target labels (crop types) into numerical values, which is required by machine learning models.
2. **Train-Test Split**:
    - The data was split into training and test sets using the `train_test_split` function from scikit-learn, with 80% of the data used for training and 20% for testing.
3. **Model Selection**:
    - Multiple classification models were considered for the task, including:
        - **Logistic Regression**
        - **Decision Tree Classifier**

        - **Random Forest Classifier**
        - **SVM (Support Vector Machine)**
        - **Naive Bayes**
        - **K-Nearest Neighbors**
        - **Gradient Boosting Classifier**
    - These models were selected because they are widely used for classification tasks and provide a good balance between simplicity, interpretability, and performance.
4. **Pipeline Creation**:
    - A **Pipeline** was used to combine preprocessing (scaling) with the classification model. The pipeline simplifies the model training and evaluation process and ensures that preprocessing steps are correctly applied during both training and prediction phases.
    - The pipeline includes:
        - **StandardScaler**: This is used to standardize the features (input variables). Standardization is necessary for models like SVM and KNN, which rely on the scale of the features.
        - **Classifier**: A machine learning model such as Logistic Regression, Decision Tree, Random Forest, etc.
5. **Model Training**:
    - Each model was trained using the training data, and predictions were made on the test set. The evaluation metrics used were based on classification metrics such as precision, recall, F1-score, and accuracy, which were printed using the `classification_report` from scikit-learn.
6. **Final Model Selection**:
    - After training and evaluating all models, the best-performing model (based on classification metrics) was selected for final deployment. In this case, the **Gradient Boosting Classifier** was selected due to its superior performance in terms of accuracy and other classification metrics.
7. **Model Deployment**:
    - The final trained model, along with the preprocessing pipeline, was saved using **pickle** for later use in a Flask web application.
    - The Flask app was created to accept user inputs (soil and climate data) via a web form, predict the most suitable crop, and display the result to the user.

4. Pipeline and Model Explanation

The **pipeline** plays a crucial role in simplifying the model training and deployment process. The steps in the pipeline are:

- **StandardScaler**: This step standardizes the features to ensure they have a mean of 0 and a standard deviation of 1. This is particularly useful for models like SVM and KNN that are sensitive to the scale of the input features.
- **Classifier**: The classifier can be any model that performs classification. In this case, we used a **Gradient Boosting Classifier**, which is an ensemble method that builds multiple

decision trees and combines them for improved prediction accuracy. Gradient Boosting is particularly effective for structured/tabular data like the one used in this project.

The pipeline ensures that all preprocessing and training steps are applied consistently, reducing the risk of errors during deployment.

5. Evaluation of the Model

The selected model, **Gradient Boosting Classifier**, was evaluated based on its performance on the test data. Here are some key points:

- **Accuracy**: The model performed well in terms of predicting the correct crop based on input features.
- **Classification Metrics**: The classification report provides metrics such as precision, recall, and F1-score, which help in understanding how well the model performs for each class (crop type).
- **Overfitting**: Given that the model was trained on a relatively small dataset, it's important to verify that the model generalizes well to new, unseen data. The test set accuracy serves as a good indicator of this.

6. Conclusion

The approach implemented in this project successfully predicts the most suitable crop for a given set of