**Crop Reccomendation System**

**Nathan Mungai-665286**

**Thomas Duncan-664324**

**Addressing SDG (Sustainable Development Goals) 2: Zero Hunger through Crop Recommendation System**

**Introduction to SDG 2: Zero Hunger**

The UN Sustainable Development Goal 2 (SDG 2) aims to end hunger, achieve food security and improved nutrition, and promote sustainable agriculture by 2030. This goal is crucial as it addresses the fundamental human right to food and seeks to eliminate the root causes of hunger and malnutrition globally. SDG 2 emphasizes the need for sustainable agricultural practices that increase productivity, improve the livelihoods of farmers, and ensure that food systems are resilient to climate change and other challenges.

**Our Specific Problem: Unpredictable Crop Yields**

One of the major challenges facing farmers, especially in developing regions, is the unpredictability of crop yields. Factors such as climate variability, soil health, and inadequate farming practices contribute to fluctuating crop yields, making it difficult for farmers to plan and manage their resources effectively. This unpredictability not only affects farmers’ incomes but also contributes to food insecurity in the broader community.

**Planned Machine Learning Application: Crop Recommendation System**

**Objective**

The primary objective of the Crop Recommendation System is to provide farmers with accurate recommendations on what crops to plant based on various input features such as weather conditions, soil characteristics, and historical crop data. By leveraging machine learning techniques, the system aims to help farmers make informed decisions, optimize their farming practices, and increase agricultural productivity and food security.

**Data** **Collection**

The system will utilize a variety of data sources to build an accurate and robust recommendation model:

- Weather Data: Information on temperature, precipitation, humidity, and other climatic factors.

- Soil Data: Soil properties such as nutrient levels, pH, and organic matter content.

- Historical Crop Yields: Historical records of crop yields for different regions and crop types.

**Machine Learning Model**

The recommendation system will employ a Random Forest Classifier to recommend suitable crops. The choice of Random Forest is due to its ability to handle complex interactions between input features and its robustness to overfitting.

1. Data Preprocessing:

- Clean and preprocess the data to handle missing values and normalize feature scales.

2. Model Training:

- Train the Random Forest Classifier using the training dataset.

3. Prediction and Evaluation:

- Use the trained model to recommend crops on the testing dataset.

**Implementation Steps**

1. Data Collection:

- Collect data from various sources including weather APIs, soil databases, and historical crop yield records.

2. Data Preprocessing:

- Handle missing values by imputing or removing them.

- Normalize the data to ensure consistency.

3. Model Training:

- Train the Random Forest Classifier on the preprocessed data.

- Evaluate the model’s performance using metrics such as accuracy, precision, and recall.

4. User Interface Development:

- Develop a web application to allow farmers to input their data.

- Provide forms for inputting location, crop type, soil properties, and recent weather conditions.

- Display recommended crops based on the input data.

**User Interface**

A user-friendly interface will be developed to allow farmers to input their data and receive recommendations. This interface can be in the form of a web application, providing:

- Data Input Forms: Fields for farmers to enter information about their location, soil properties, and recent weather conditions.

- Recommendation Results: Displaying the recommended crops and providing actionable insights for optimizing farming practices.

**Week 1 Conclusion**

The Crop Recommendation System aims to address the challenges of unpredictable crop yields by leveraging machine learning techniques to provide accurate and actionable insights to farmers. By helping farmers optimize their agricultural practices, the system contributes to achieving SDG 2: Zero Hunger, enhancing food security, and promoting sustainable agriculture.

**Design and Development of Our Crop Recommendation System**

**Introduction**

**Problem Definition**

The unpredictability of crop yields due to several factors such as climate variability, soil health, and inadequate farming practices poses a significant challenge to farmers. This unpredictability affects farmers’ incomes and contributes to food insecurity, thus hindering progress towards the UN SDG (Sustainable Development Goals) 2. Our Crop Recommendation System aims to provide recommendations on what crops to plant using machine learning techniques, enabling farmers to make informed decisions, optimize their practices, and increase agricultural productivity.

**Objectives**

- To recommend crops based on input features such as weather conditions, soil characteristics, and historical crop data.

- To provide actionable recommendations to farmers for optimizing farming practices.

- To contribute to achieving SDG 2: Zero Hunger by enhancing food security and promoting sustainable agriculture.

**Machine Learning Techniques**

**Selected Techniques**

- Classification: For predicting categorical outcomes (crop types). We have chosen to use the Random Forest Classifier for its robustness and ability to handle complex interactions between input features.

- Data Pre-processing: Standardization and handling of missing values to ensure data quality.

- Cross-Validation: To ensure model robustness and prevent overfitting.

**System Architecture**

1. Data Collection and Storage: Collecting data from various sources and storing it in a centralized database.

2. Data Pre-processing: Cleaning, normalizing, and preparing the data for model training.

3. Model Training: Training the classification model using the pre-processed data.

4. User Interface: Allowing farmers to input their data and receive predictions and recommendations.

5. Prediction and Recommendation Engine: Using the trained model to make predictions and generate recommendations.

**Detailed Architecture**

1. Data Collection and Storage:

- Weather Data: Collected from weather stations and APIs.

- Soil Data: Collected from soil databases.

- Historical Crop Yields: Collected from agricultural records and databases.

- Database: A relational database to store and manage the collected data.

2. Data Pre-processing:

- Cleaning: Removing or imputing missing values.

- Normalization: Standardizing feature scales to ensure consistency.

- Feature Engineering: Creating new features or modifying existing ones to improve model performance.

3. Model Training:

- Algorithm: Random Forest Classifier.

- Training: Splitting the data into training and testing sets. Using cross-validation to ensure robustness.

- Evaluation: Using metrics such as Accuracy, Precision, Recall, and F1 Score to evaluate model performance.

4. User Interface:

- Web Application: A user-friendly web application where farmers can input data and receive predictions.

- Input Forms: Fields for farmers to enter information about their location, crop type, soil properties, and recent weather conditions.

- Results Display: Showing the predicted crop type and recommendations.

5. Prediction and Recommendation Engine:

- Prediction: Using the trained model to predict suitable crops based on input data.

**Initial Development Progress and Planning**

1. Data Collection:

- Weather Data: Data is to be collected from weather APIs like OpenWeatherMap.

- Soil Data: Soil assessments are to be gathered from databases like ISRIC and soilgrids.

- Historical Crop Yields: Agricultural records are to be obtained from databases. FAOSTAT and World Bank Agricultural data are going to be used.

2. Data Pre-processing:

- Cleaning: Any missing values will be handled through imputation.

- Normalization: Features will be standardized to ensure consistency.

- Feature Engineering: New features will be created based on feedback from users.

3. Model Training:

- Algorithm: A Random Forest Classifier has been chosen to make predictions on crop yield.

- Training: The model will be trained on the collected data.

- Evaluation: The model’s performance will be evaluated using metrics such as Accuracy, Precision, Recall, and F1 Score.

4. User Interface Development:

- We plan to develop a basic web application with input forms for farmers to manually enter their data. This can be scaled at a later stage to use IoT sensors.

- Initial predictions and recommendations are to be displayed based on the input data.

**Week 2 Conclusion**

The design of the Crop Yield Prediction System has laid solid groundwork for a promising Machine Learning model to be built on. The readily available data from large corporations can be easily acquired to begin the creation of a dataset and the Random Forest Classifier model is sure to provide accurate predictions. Moving forward, we will code and train our model and design the user interface for farmer inputs as well as creating the feedback mechanism to ensure effectiveness.

**Progress Report on Crop Yield Prediction System**

**Introduction**

This progress report summarizes the development process, testing results, and challenges encountered while building a crop yield prediction system using machine learning techniques. The project's main goal is to predict the most suitable crop for given soil and weather conditions using a web application interface.

**Development Process**

1. Project Setup and Initial Planning

* Objective: To create a machine learning model that predicts the best crop to plant based on various environmental and soil parameters.
* Tools and Libraries: Python, Flask, pandas, scikit-learn, LabelEncoder, RandomForestClassifier, HTML/CSS/JS for the web interface.

1. Data Collection and Preparation

* Dataset: We are using dataset containing features such as Nitrogen, Phosphorus, Potassium, temperature, humidity, pH, and rainfall, along with the crop that strives under different conditions. The dataset is a combination of 2 separate datasets that we found on kaggle.
* Loading Data: Data loaded into a pandas DataFrame for easy manipulation and analysis.
* Feature Extraction: Extracted relevant features and separated the target variable.
* Label Encoding: Encoded the categorical target variable into numerical values using `LabelEncoder`.

1. Model Training

* Model Selection: We chose to use a `RandomForestClassifier` for its robustness and accuracy in handling classification tasks.
* Data Splitting: We split the dataset into training and testing sets to evaluate model performance.
* Training: Trained the RandomForestClassifier on the training set.
* Evaluation: Evaluated the model on the testing set to ensure it performs well on unseen data.

1. Web Application Development

* Flask Setup: We chose to set up a Flask web application to provide a user-friendly interface for predictions.
* HTML Form: Created an HTML form for users to interact with and input their data.
* Prediction Logic: Integrated the trained model to make predictions based on user input and return the predicted crop.

**Testing Results**

1. Model Performance

* Accuracy: The model achieved an accuracy of 98% on the testing set, indicating good predictive performance.
* Confusion Matrix: Analyzed the confusion matrix to understand the types of errors the model makes.
* Precision and Recall: Calculated precision and recall for each class to ensure the model's reliability across different crop types.

1. Web Application Testing

* Form Validation: Ensured the HTML form correctly collects and validates user input.
* Prediction Accuracy: Verified that the predictions made by the web application match the model's predictions in a local environment.
* User Experience: Tested the overall user experience, ensuring the web interface is intuitive and responsive.

**Challenges Encountered**

1. 1. Data Quality

* Missing Values: Some entries in the dataset had missing values, which required handling through imputation.
* Imbalanced Data: Certain crop classes were underrepresented, leading to potential bias in predictions.

1. Model Selection and Tuning

* Algorithm Choice: Initially experimented with different algorithms (Regression, Decision Trees) before settling on RandomForestClassifier for its superior performance.

1. Deployment Issues

* Environment Setup: Encountered issues with setting up the development environment, such as installing necessary libraries and configuring the Flask app.
* File Path Handling: Addressed path-related issues due to differences in operating systems by using raw strings and double backslashes.

1. User Interface

* Coming up with an eye-catching user interface while keeping it straightforward was a new but interesting challenge for us.

**Week 3 Conclusion and Next Steps**

The crop yield prediction system is functional, providing accurate predictions through a user-friendly web interface. The following steps are planned for further enhancement:

1. Model Improvement: Explore advanced models and techniques to improve prediction accuracy.
2. Big Data Integration: Incorporate larger datasets and use distributed computing frameworks like Apache Spark for scalability.
3. Prescriptive Analytics: Extend the system to provide actionable recommendations based on predictions.
4. User Feedback: Collect feedback from potential users (farmers) to refine the user interface and feature set.
5. IOT integration. Use of IOT devices to read soil and weather data for automatic inputs to the program.

**Final touch ups on our Crop Recommendation System**

We updated the user interface to include images for all the different crop outputs as well as showing the probability of success to help users understand how likely one crop is to be successful in comparison to another. We checked to ensure that the app runs fine and concluded our project