MADE Project Report:

Automated Pipeline Empowers Crop Recommendations, Production, and Climate Analysis

Author	Rahul Nitin Ramraje
Matriculation number	23081510
Git hub	https://github.com/Rahul2899/made-SS24-Rahul
E-Mail	rahul.ramraje02@fau.de
Course of study	MSc Data Science
Date of submission	06/06/2024

1. Introduction:

1.1 Question:

2. How can integrating and analyzing soil composition and climate data improve crop recommendations to enhance agricultural productivity, resource management, sustainability, and climate resilience?

Source 1: Crop Recommendation Dataset

- **Source:** Kaggle
- **Description**: Contains soil composition (Nitrogen, Phosphorus, Potassium) and environmental variables (Temperature, Humidity, pH Value, Rainfall) to aid in precise crop recommendations.
- **File Format:** CSV
- **License:** Standard open-data License Data
- **URL:** https://www.kaggle.com/datasets/varshitanalluri/crop-recommendation-dataset

Source 2: Crop Production and Climate Change Dataset

- **Source:** Kaggle
- **Description:** Provides data on crop yields, harvested areas, and production quantities for wheat, maize, rice, and soybeans from 2010-2016.
- **File Format:** CSV
- **License:** Creative Commons Attribution-ShareAlike
- **URL:** https://www.kaggle.com/datasets/thedevastator/the-relationship-between-crop-production-and-cli

License Compliance:

- Attribution of sources where required.
- Ensuring no data is shared outside the analysis team without proper anonymization and attribution.

3. Data Pipeline

3.1 Technologies Used:

- **Data Loading**: Pandas for data loading.
- **Storage**: Intermediate storage using Pandas Data Frames.
- **Processing**: Pandas and NumPy for data transformation and cleaning.
- **Automation**: Google Colab for orchestrating the pipeline..

3.2 Transformation and Cleaning Steps:

- **Data Loading**: Load CSV files into Pandas DataFrames.
- **Data Cleaning**:

1. **Deduplication Operation**: Remove duplicate entries based on relevant fields.
2. **Handling Missing Values**: Fill or drop missing values based on analysis needs.
3. **Normalization**: Standardize date formats and numerical values.

- **Data Enrichment**: Calculate additional metrics such as average crop yield over a period and climate anomalies.
- **Storage**: Store the cleaned and enriched data in Pandas DataFrames and export to CSV for reporting.

4. Error Handling and Adaptability:

- **Error Logging**: Use Python's logging module to capture and log errors during pipeline execution.
- **Data Validation**: To ensure data integrity, implement validation checks at each stage.
- **Scalability**: Design a pipeline to handle increasing data volumes and potential new data sources by modularizing components.

5. Result and Limitations

5.1 Outputs

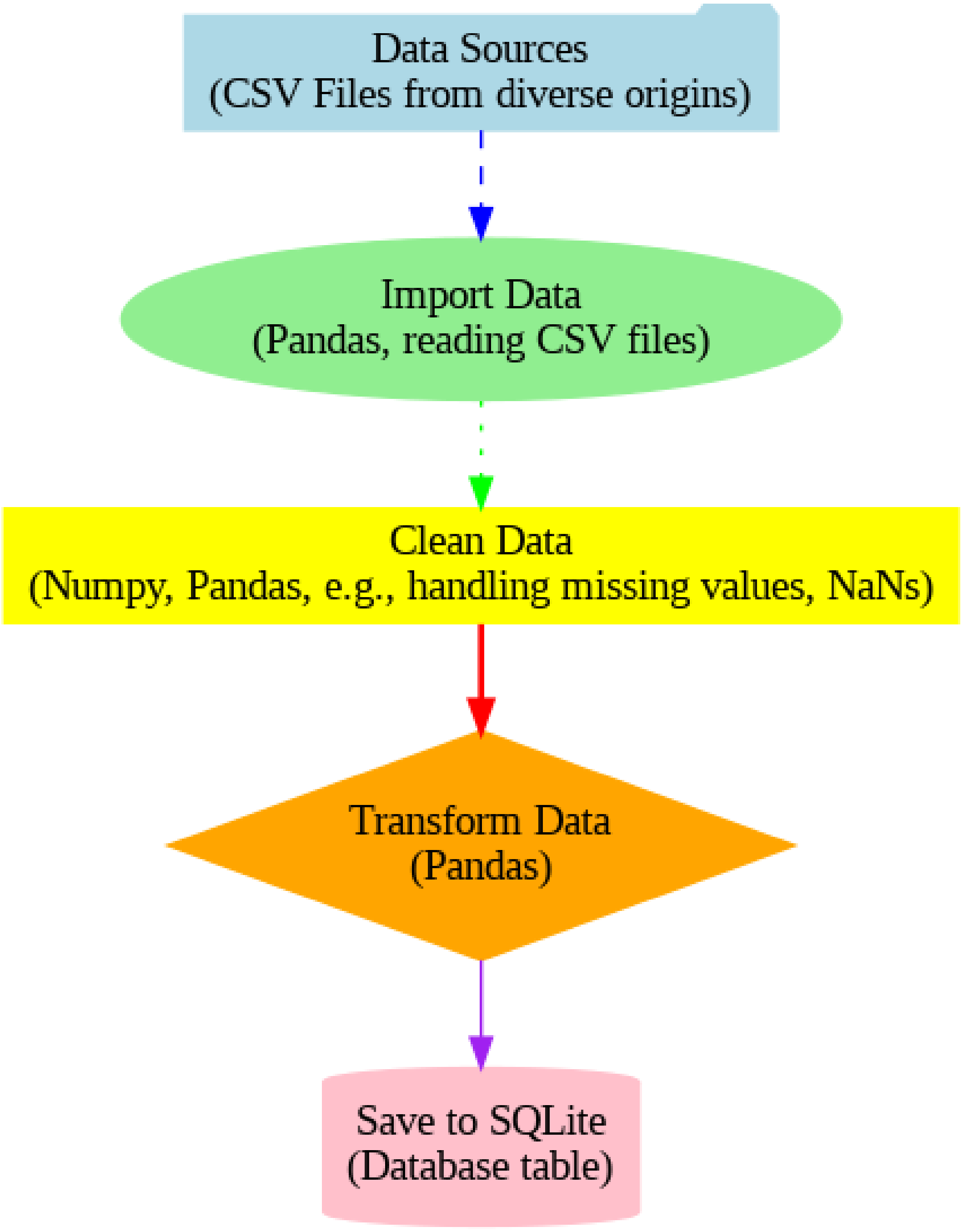
- **Structure**: Combined and cleaned datasets with fields for soil composition, environmental variables, crop yields, and additional calculated metrics.
- **Quality**: High-quality data with reduced noise, standardized formats, and enriched information for better analysis.

5.2 Critical Issues

- **Data Issues**: Potential biases in soil and climate data, and inconsistencies in crop yield reporting.
- **Future Improvements**:
  - Enhancement of anomaly detection in environmental data.
  - Integration of additional agricultural datasets for a more comprehensive analysis.
  - Continuous monitoring and adaptation of the pipeline for evolving data landscapes.

6. Figures

6.1 Figure 1: Data Pipeline Structure


Data Sources
(CSV Files from diverse origins)
Import Data
(Pandas, reading CSV files)
Clean Data
(Numpy, Pandas, e.g., handling missing values, NaNs)
Transform Data
(Pandas)
Save to SQLite
(Database table)