Day#2 Report

Date: [25/02/2025]

Prepared by: [group7]

Tasks Completed

1. Flow Chart Completion

Objective: To visualize the process and workflow of the application

Finalized the flow chart that outlines the key components and interactions within the system.

Ensured that all user inputs, system processes, and outputs are clearly represented.

Reviewed the flow chart with the team for feedback and made necessary adjustments based on their input.

2. System Design Finalization

Objective: To establish a robust architecture for the application.

Completed the system design documentation, including:

- Database schema
- API endpoints
- User interface wireframes

Collaborated with team members to ensure that the design aligns with project requirements and user needs.

Conducted a design review session to gather insights and finalize the design specifications.

3. Application Development Using Flask

Objective: To develop a web application using the Flask framework to predict draugth.

Set up the Flask environment and created the initial project structure.

Implemented core functionalities, including:

- Data input forms
- Integration with the database

Tested the application for bugs and ensured that all features are functioning as intended.

Documented the code and created a README file for future reference.

4. Machine Learning Training

Objective: To enhance the application with machine learning capabilities.

Engaged in training sessions focused on machine learning algorithms relevant to the project.

Explored various models and techniques that can be integrated into the application for predictive analytics.

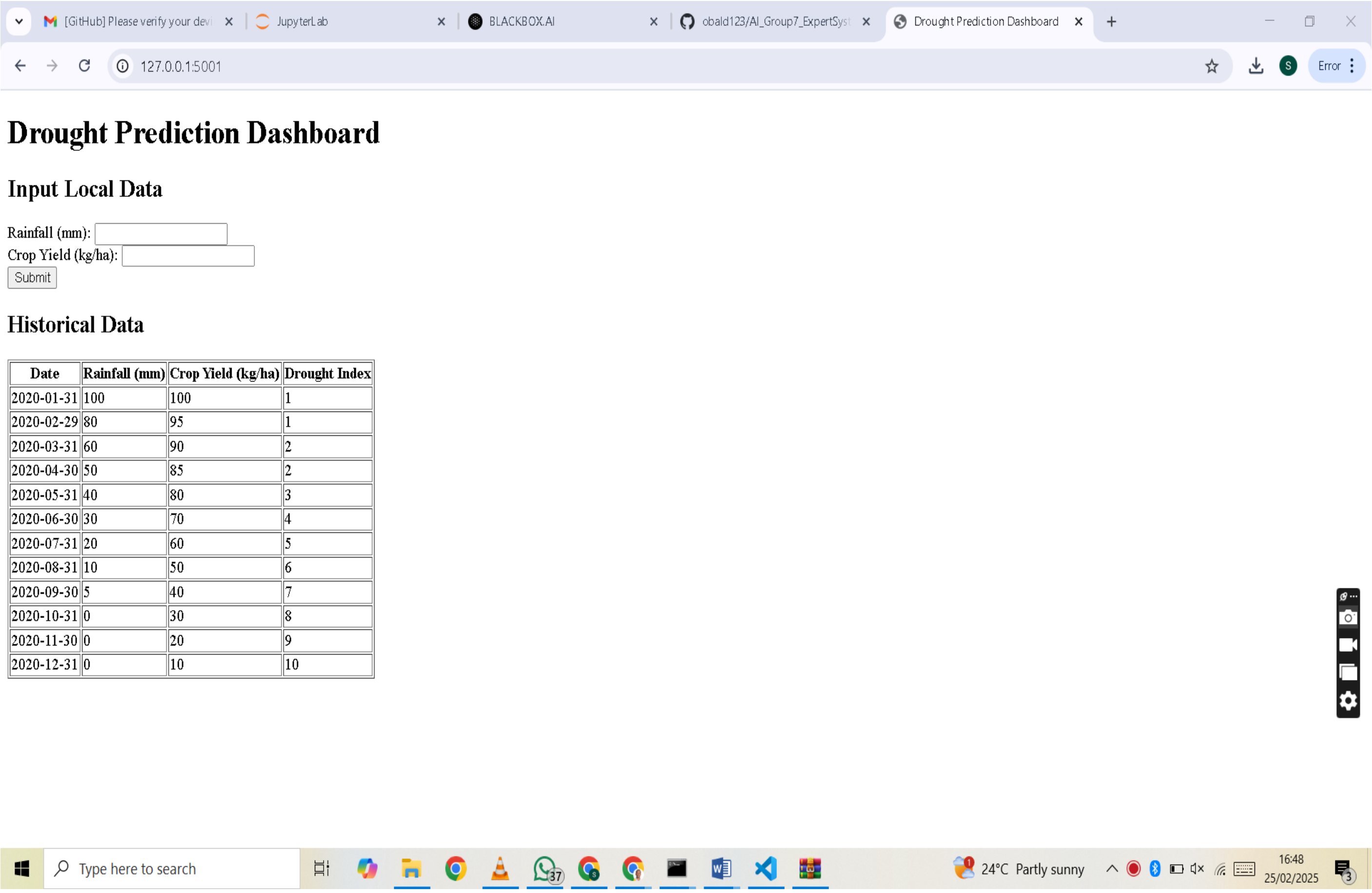
Started preliminary work on data preprocessing and feature selection for the machine learning model.

Collaborated with data scientists to understand best practices and gather insights on model evaluation.

Conclusion:

Today was productive, with significant progress made in completing the flow chart and specially developing the Flask application. The training in machine learning is also progressing well, setting a solid foundation for future enhancements.

System design



Drought Prediction Dashboard

Input Local Data

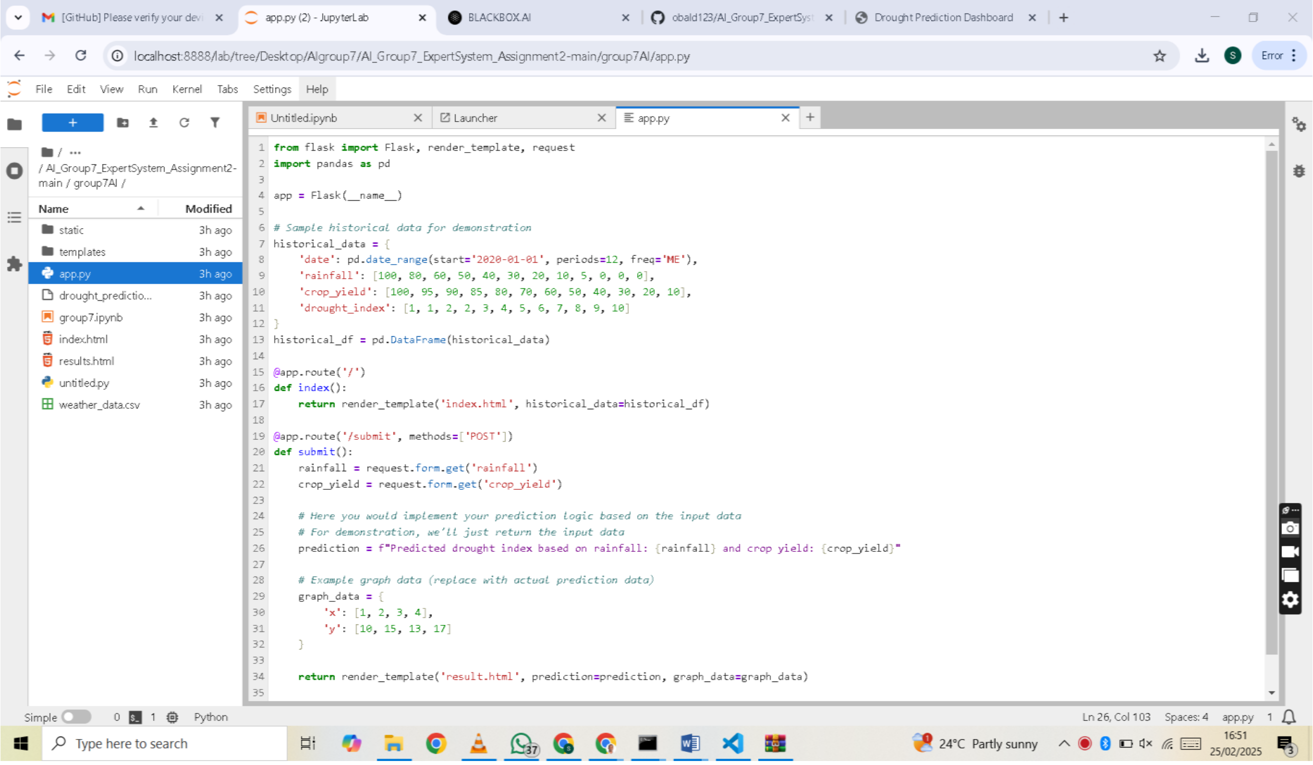
Rainfall (mm):

Crop Yield (kg/ha):

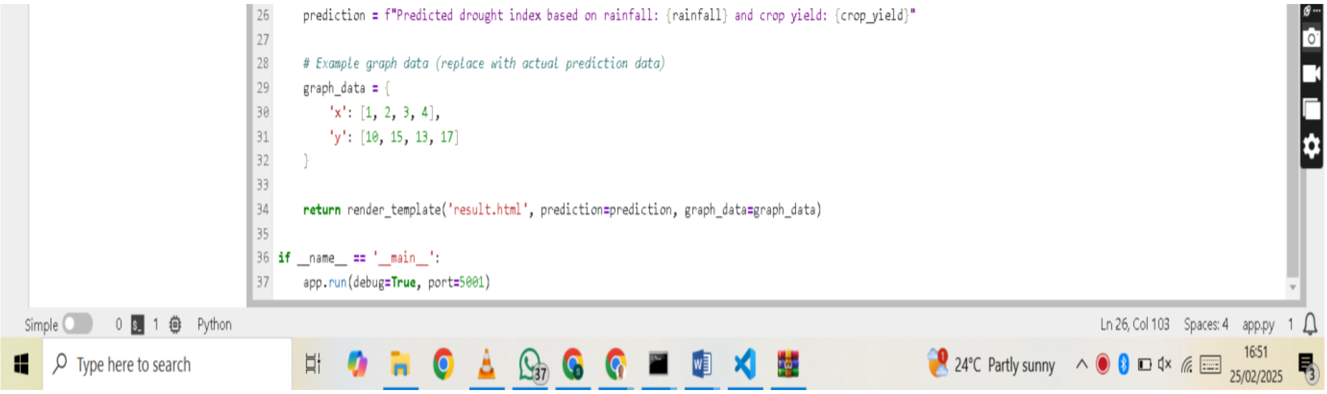
Historical Data

Date	Rainfall (mm)	Crop Yield (kg/ha)	Drought Index
2020-01-31	100	100	1
2020-02-29	80	95	1
2020-03-31	60	90	2
2020-04-30	50	85	2
2020-05-31	40	80	3
2020-06-30	30	70	4
2020-07-31	20	60	5
2020-08-31	10	50	6
2020-09-30	5	40	7
2020-10-31	0	30	8
2020-11-30	0	20	9
2020-12-31	0	10	10

Python code

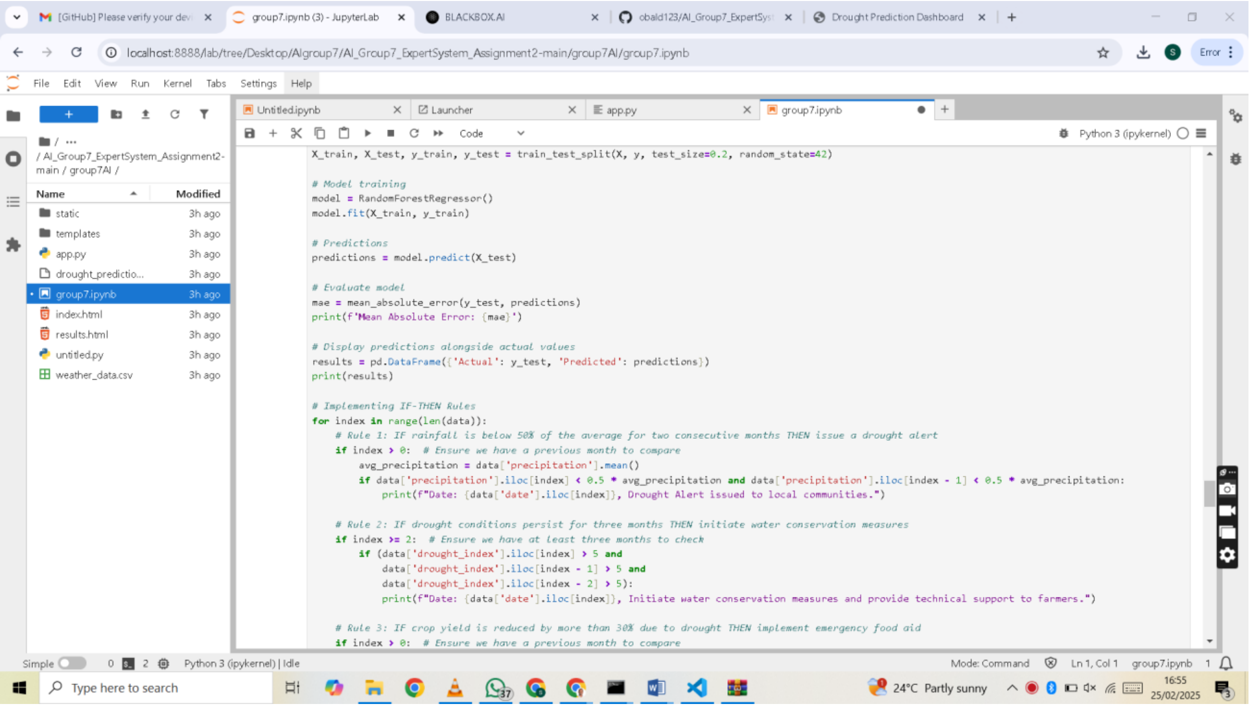

[GitHub] Please verify your devi
app.py (2) - JupyterLab
BLACKBOX.AI
obald123/AI_Group7_ExpertSys
Drought Prediction Dashboard
localhost:8888/lab/tree/Desktop/AIgroup7/AI_Group7_ExpertSystem_Assignment2-main/group7AI/app.py
Error
File Edit View Run Kernel Tabs Settings Help
/ AI_Group7_ExpertSystem_Assignment2-main / group7AI /
Name Modified
static 3h ago
templates 3h ago
app.py 3h ago
drought_predictio... 3h ago
group7.ipynb 3h ago
index.html 3h ago
results.html 3h ago
untitled.py 3h ago
weather_data.csv 3h ago
Untitled.ipynb
Launcher
app.py
Simple 0 1 Python
Ln 26, Col 103 Spaces: 4 app.py 1
Type here to search
24°C Partly sunny
16:51
25/02/2025

```python
from flask import Flask, render_template, request
import pandas as pd

app = Flask(__name__)

# Sample historical data for demonstration
historical_data = {
    'date': pd.date_range(start='2020-01-01', periods=12, freq='ME'),
    'rainfall': [100, 80, 60, 50, 40, 30, 20, 10, 5, 0, 0, 0],
    'crop_yield': [100, 95, 90, 85, 80, 70, 60, 50, 40, 30, 20, 10],
    'drought_index': [1, 1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10]
}
historical_df = pd.DataFrame(historical_data)

@app.route('/')
def index():
    return render_template('index.html', historical_data=historical_df)

@app.route('/submit', methods=['POST'])
def submit():
    rainfall = request.form.get('rainfall')
    crop_yield = request.form.get('crop_yield')

    # Here you would implement your prediction logic based on the input data
    # For demonstration, we'll just return the input data
    prediction = f"Predicted drought index based on rainfall: {rainfall} and crop yield: {crop_yield}"

    # Example graph data (replace with actual prediction data)
    graph_data = {
        'x': [1, 2, 3, 4],
        'y': [10, 15, 13, 17]
    }

    return render_template('result.html', prediction=prediction, graph_data=graph_data)

```

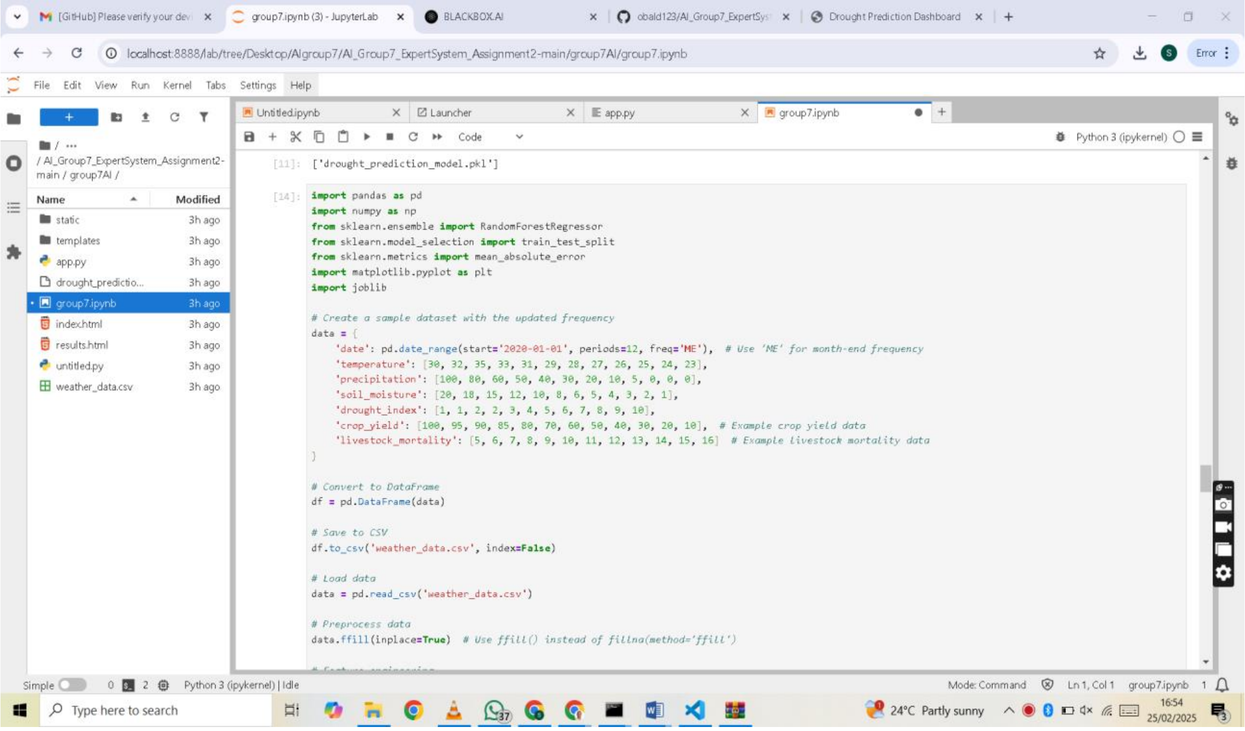


```python
    prediction = f"Predicted drought index based on rainfall: {rainfall} and crop yield: {crop_yield}"

    # Example graph data (replace with actual prediction data)
    graph_data = {
        'x': [1, 2, 3, 4],
        'y': [10, 15, 13, 17]
    }

    return render_template('result.html', prediction=prediction, graph_data=graph_data)

if __name__ == '__main__':
    app.run(debug=True, port=5001)
```

Ai code

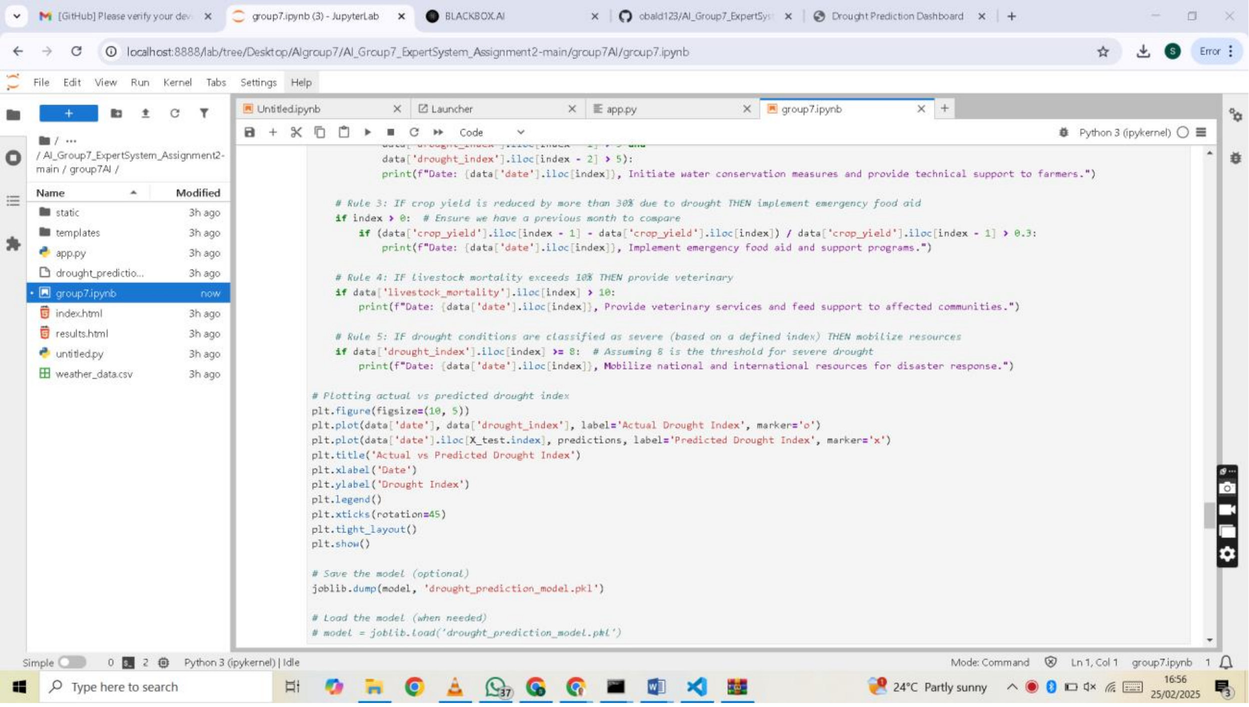

[GitHub] Please verify your dev
group7.ipynb (3) - JupyterLab
BLACKBOX.AI
obald123/AI_Group7_ExpertSys
Drought Prediction Dashboard
localhost:8888/lab/tree/Desktop/AIgroup7/AI_Group7_ExpertSystem_Assignment2-main/group7AI/group7.ipynb
Error
File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Launcher
app.py
group7.ipynb
Code
Python 3 (ipykernel)
/ AI_Group7_ExpertSystem_Assignment2-main / group7AI /
Name
Modified
static 3h ago
templates 3h ago
app.py 3h ago
drought_predictio... 3h ago
group7.ipynb 3h ago
index.html 3h ago
results.html 3h ago
untitled.py 3h ago
weather_data.csv 3h ago
Simple 0 2 Python 3 (ipykernel) | Idle
Mode: Command Ln 1, Col 1 group7.ipynb 1
Type here to search
24°C Partly sunny
16:55
25/02/2025

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
model = RandomForestRegressor()
model.fit(X_train, y_train)

# Predictions
predictions = model.predict(X_test)

# Evaluate model
mae = mean_absolute_error(y_test, predictions)
print(f'Mean Absolute Error: {mae}')

# Display predictions alongside actual values
results = pd.DataFrame({'Actual': y_test, 'Predicted': predictions})
print(results)

# Implementing IF-THEN Rules
for index in range(len(data)):
    # Rule 1: IF rainfall is below 50% of the average for two consecutive months THEN issue a drought alert
    if index > 0:  # Ensure we have a previous month to compare
        avg_precipitation = data['precipitation'].mean()
        if data['precipitation'].iloc[index] < 0.5 * avg_precipitation and data['precipitation'].iloc[index - 1] < 0.5 * avg_precipitation:
            print(f"Date: {data['date'].iloc[index]}, Drought Alert issued to local communities.")

    # Rule 2: IF drought conditions persist for three months THEN initiate water conservation measures
    if index >= 2:  # Ensure we have at least three months to check
        if (data['drought_index'].iloc[index] > 5 and
            data['drought_index'].iloc[index - 1] > 5 and
            data['drought_index'].iloc[index - 2] > 5):
            print(f"Date: {data['date'].iloc[index]}, Initiate water conservation measures and provide technical support to farmers.")

    # Rule 3: IF crop yield is reduced by more than 30% due to drought THEN implement emergency food aid
    if index > 0:  # Ensure we have a previous month to compare
```


[GitHub] Please verify your dev
group7.ipynb (3) - JupyterLab
BLACKBOX.AI
obald123/AI_Group7_ExpertSys
Drought Prediction Dashboard
localhost:8888/lab/tree/Desktop/AIgroup7/AI_Group7_ExpertSystem_Assignment2-main/group7AI/group7.ipynb
Error
File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Launcher
app.py
group7.ipynb
Code
Python 3 (ipykernel)
/ AI_Group7_ExpertSystem_Assignment2-main / group7AI /
Name
Modified
static 3h ago
templates 3h ago
app.py 3h ago
drought_predictio... 3h ago
group7.ipynb 3h ago
index.html 3h ago
results.html 3h ago
untitled.py 3h ago
weather_data.csv 3h ago
Simple 0 2 Python 3 (ipykernel) | Idle
Mode: Command Ln 1, Col 1 group7.ipynb 1
Type here to search
24°C Partly sunny
16:54
25/02/2025

```
[11]: ['drought_prediction_model.pkl']
```

```python
[14]: import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt
import joblib

# Create a sample dataset with the updated frequency
data = {
    'date': pd.date_range(start='2020-01-01', periods=12, freq='ME'),  # Use 'ME' for month-end frequency
    'temperature': [30, 32, 35, 33, 31, 29, 28, 27, 26, 25, 24, 23],
    'precipitation': [100, 80, 60, 50, 40, 30, 20, 10, 5, 0, 0, 0],
    'soil_moisture': [20, 18, 15, 12, 10, 8, 6, 5, 4, 3, 2, 1],
    'drought_index': [1, 1, 2, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'crop_yield': [100, 95, 90, 85, 80, 70, 60, 50, 40, 30, 20, 10],  # Example crop yield data
    'livestock_mortality': [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]  # Example livestock mortality data
}

# Convert to DataFrame
df = pd.DataFrame(data)

# Save to CSV
df.to_csv('weather_data.csv', index=False)

# Load data
data = pd.read_csv('weather_data.csv')

# Preprocess data
data.ffill(inplace=True)  # Use ffill() instead of fillna(method='ffill')

# Feature engineering
```


[GitHub] Please verify your dev
group7.ipynb (3) - JupyterLab
BLACKBOX.AI
obald123/AI_Group7_ExpertSys
Drought Prediction Dashboard
localhost:8888/lab/tree/Desktop/AIgroup7/AI_Group7_ExpertSystem_Assignment2-main/group7AI/group7.ipynb
Error
File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Launcher
app.py
group7.ipynb
Code
Python 3 (ipykernel)
/ AI_Group7_ExpertSystem_Assignment2-main / group7AI /
Name
Modified
static 3h ago
templates 3h ago
app.py 3h ago
drought_predictio... 3h ago
group7.ipynb now
index.html 3h ago
results.html 3h ago
untitled.py 3h ago
weather_data.csv 3h ago
Simple 0 2 Python 3 (ipykernel) | Idle
Mode: Command Ln 1, Col 1 group7.ipynb 1
Type here to search
24°C Partly sunny
16:56
25/02/2025

```
            data['drought_index'].iloc[index - 1] > 5 and
            data['drought_index'].iloc[index - 2] > 5):
            print(f"Date: {data['date'].iloc[index]}, Initiate water conservation measures and provide technical support to farmers.")

    # Rule 3: IF crop yield is reduced by more than 30% due to drought THEN implement emergency food aid
    if index > 0:  # Ensure we have a previous month to compare
        if (data['crop_yield'].iloc[index - 1] - data['crop_yield'].iloc[index]) / data['crop_yield'].iloc[index - 1] > 0.3:
            print(f"Date: {data['date'].iloc[index]}, Implement emergency food aid and support programs.")

    # Rule 4: IF livestock mortality exceeds 10% THEN provide veterinary
    if data['livestock_mortality'].iloc[index] > 10:
        print(f"Date: {data['date'].iloc[index]}, Provide veterinary services and feed support to affected communities.")

    # Rule 5: IF drought conditions are classified as severe (based on a defined index) THEN mobilize resources
    if data['drought_index'].iloc[index] >= 8:  # Assuming 8 is the threshold for severe drought
        print(f"Date: {data['date'].iloc[index]}, Mobilize national and international resources for disaster response.")

# Plotting actual vs predicted drought index
plt.figure(figsize=(10, 5))
plt.plot(data['date'], data['drought_index'], label='Actual Drought Index', marker='o')
plt.plot(data['date'].iloc[X_test.index], predictions, label='Predicted Drought Index', marker='x')
plt.title('Actual vs Predicted Drought Index')
plt.xlabel('Date')
plt.ylabel('Drought Index')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Save the model (optional)
joblib.dump(model, 'drought_prediction_model.pkl')

# Load the model (when needed)
# model = joblib.load('drought_prediction_model.pkl')
```