DS_EBPL_MITIGATING CLIMATE CHANGE

(DATA SCIENCE APPROACH TO CLIMATE PREDICTION AND MITIGATION)

A PROJECT REPORT SUBMITTED BY

HARISH R 731122205018

ON THE COURSE OF DATA SCIENCE
IN NAAN MUDHALVAN

GOVERNMENT COLLEGE OF ENGINEERING
ERODE – 638316

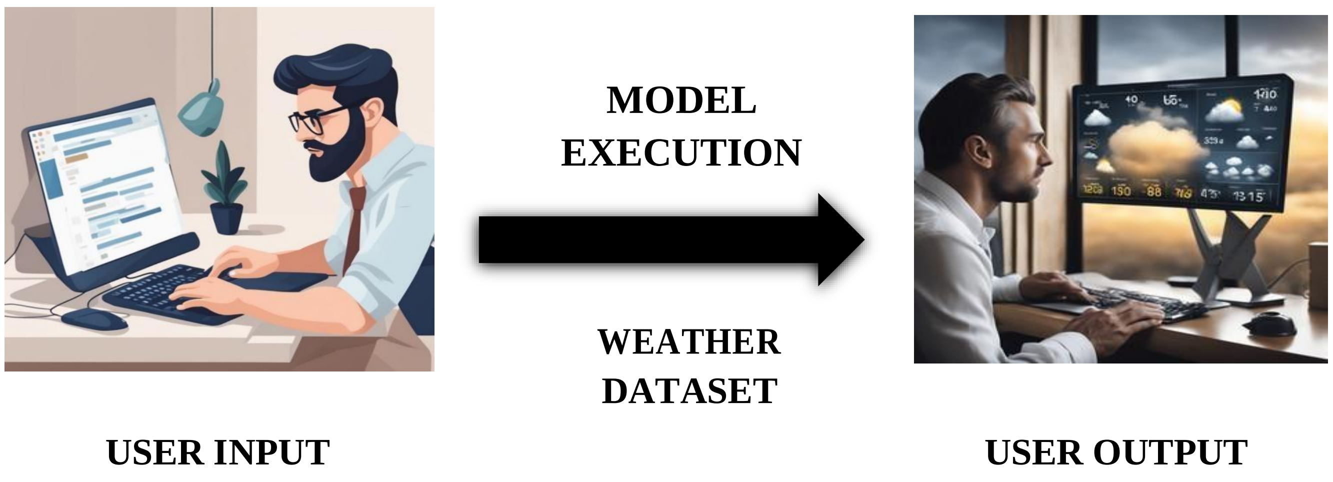
MAY 26, 2024

TABLE OF CONTENTS

1. ABSTRACT:-

Climate change presents an urgent and multifaceted challenge, impacting ecosystems, economies, and societies worldwide. This project employs advanced data-driven methodologies to predict a comprehensive range of climate variables, including temperature, precipitation, humidity, wind speed, daily summary, and overall weather summary. Additionally, the project aims to develop effective mitigation strategies to address the challenges posed by climate change.

This project aims to utilize data science techniques to address the critical issue of climate change by focusing on climate prediction and mitigation strategies. The project will involve gathering and analyzing diverse datasets related to climate science, including atmospheric data, satellite imagery, oceanographic measurements, land use data, and other environmental variables. By applying advanced machine learning algorithms, the project seeks to develop predictive models that can accurately forecast future changes in the Earth's climate system and identify potential mitigation measures to reduce greenhouse gas emissions and minimize the impacts of climate change. The project will also involve decision support systems to provide stakeholders with actionable insights for informed decision-making and proactive climate management.



- The user will input a data set into the model for analysis.
- The model will process the user's input and generate an output.
- The output graph will be displayed in the window, presenting weather predictions over a period of week, by average temperature, SO2 emission and NO2 emission.

2. SYSTEM REQUIREMENTS:-

2.1 HARDWARE REQUIREMENTS:-

RAM	Minimum of 8 GB is required to run the model.
Graphics Card	Intel iRISxe Graphics or better
Storage	100 GB of available of disk space is required.

2.2 SOFTWARE REQUIREMENTS:-

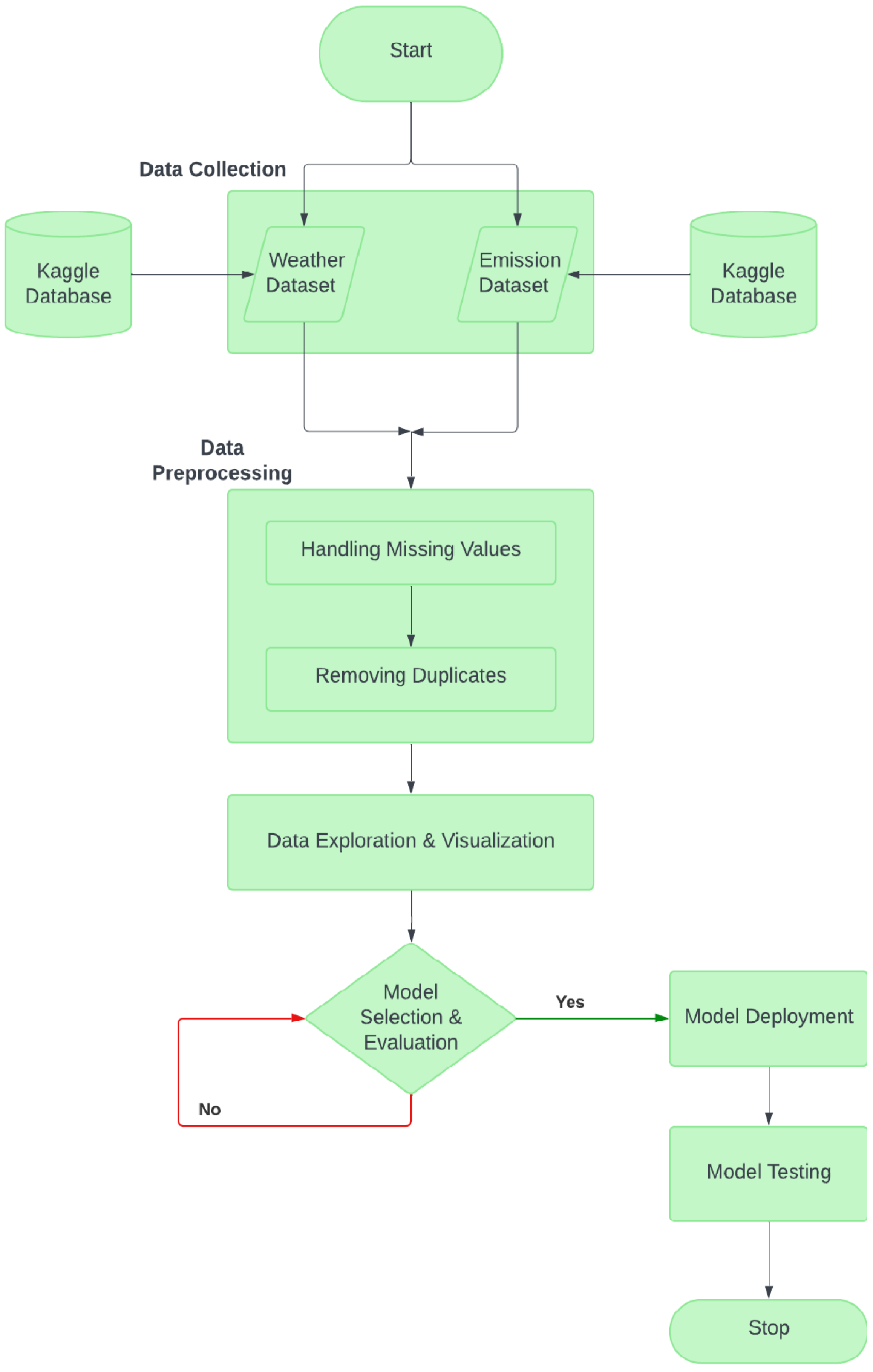
Operating System	Windows, Linux or MacOS
Development Environment	Jupyter notebook

3. TOOLS AND VERSIONS:-

Jupyter Notebook	1.0 or Higher
Python	3.6 or Higher
Pandas	1.0 or Higher
Numpy	1.18 or Higher
Sci-kit learn	0.22 or Higher
Tensorflow	2.0 or Higher

4. FLOWCHART:-

● Created a flowchart of climate prediction and mitigation.
● The process starts with collecting data from various sources.
● A weather dataset was gathered from the Kaggle website.
● Then, move on to the preprocessing stage, which is essential.
● In this stage, clean the data before moving on to model selection.
● After selecting the model, feed the data into the model and train it.
● Once training is complete, test the model with data to predict the weather and greenhouse gas emissions.


Start
Data Collection
Kaggle Database
Weather Dataset
Emission Dataset
Kaggle Database
Data Preprocessing
Handling Missing Values
Removing Duplicates
Data Exploration & Visualization
Model Selection & Evaluation
Yes
No
Model Deployment
Model Testing
Stop

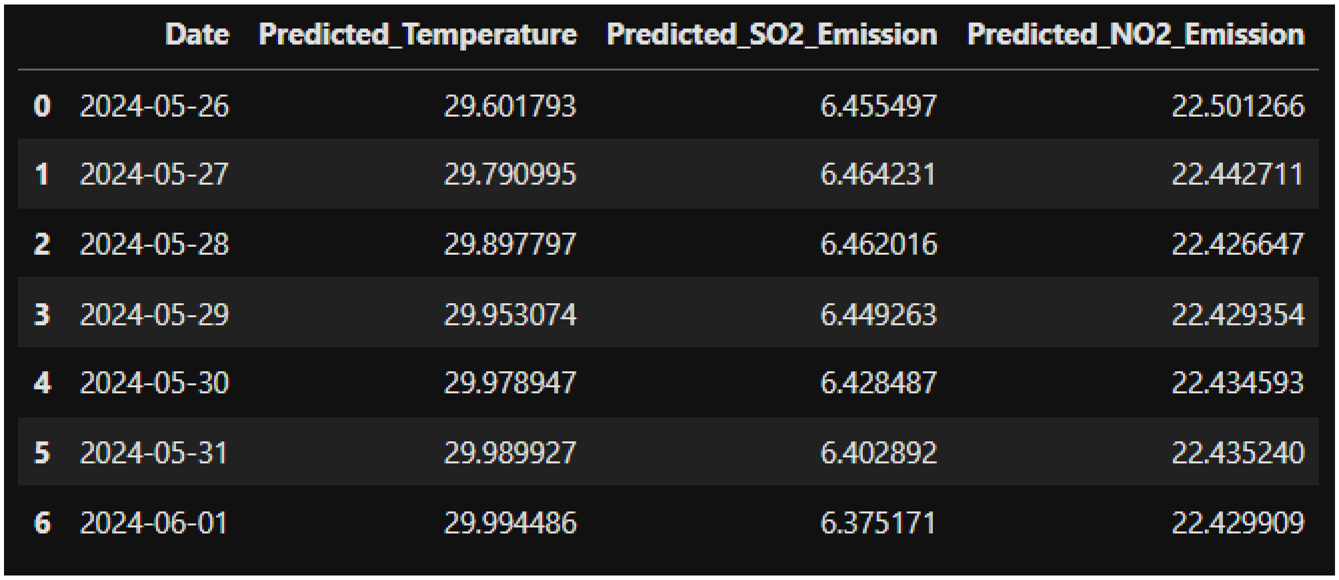
5.CODE IMPLEMENTATION:-

```
#Import Necessary Libraries
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import LSTM, Dense, GRU, Dropout
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
from datetime import datetime, timedelta
import matplotlib.pyplot as plt

# Load and preprocess data
dfa = pd.read_csv('weather.csv')
dfb = pd.read_csv('emission.csv')

# Converting 'time' and 'FORMATTED_DATE' to datetime
dfa['time'] = pd.to_datetime(dfa['time'])
dfb['FORMATTED_DATE'] = pd.to_datetime(dfb['FORMATTED_DATE'])

# Setting 'time' and 'FORMATTED_DATE' as the index
dfa.set_index('time', inplace=True)
dfb.set_index('FORMATTED_DATE', inplace=True)

# Merging the dataframes on index
data = pd.merge(dfa, dfb, left_index=True, right_index=True, how='outer')
```

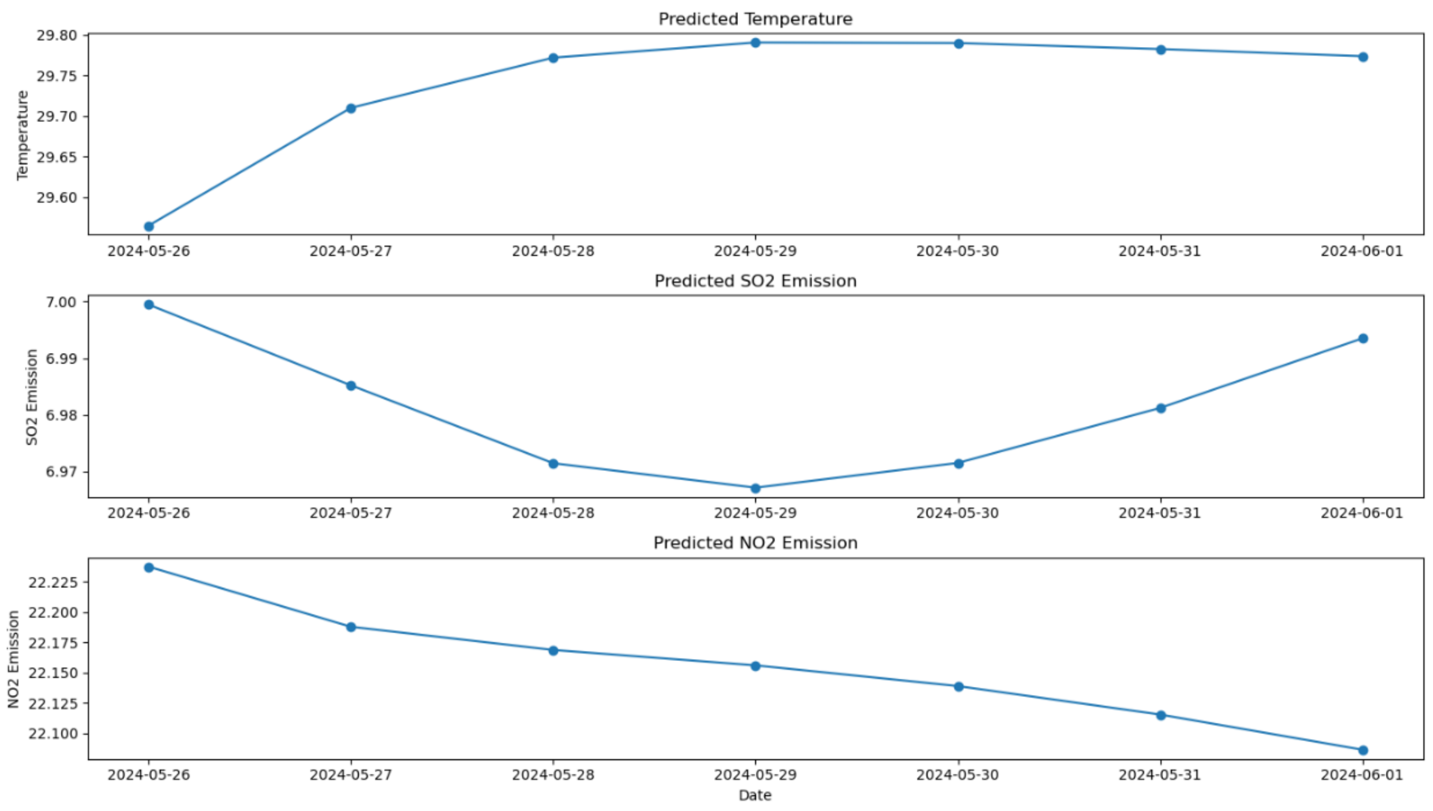
```
# Include selected features
data = data[['tavg', 'SO2_EMISSION', 'NO2_EMISSION']]

# Handle missing values & Normalize the data
data.fillna(data.mean(), inplace=True)
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(data)

# Prepare the dataset
def create_dataset(dataset, look_back=30):
    X, Y = [], []
    for i in range(len(dataset) - look_back):
        X.append(dataset[i:(i + look_back)])
        Y.append(dataset[i + look_back])
    return np.array(X), np.array(Y)

look_back = 30
X, y = create_dataset(scaled_data, look_back)
X = np.reshape(X, (X.shape[0], X.shape[1], X.shape[2]))

# Build the GRU model with increased complexity and regularization
model = Sequential([
    GRU(64, return_sequences=True, input_shape=(look_back, X.shape[2])),
    Dropout(0.2),
    GRU(64, return_sequences=False),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dense(3)
])

# Compile the model with a lower learning rate
optimizer = Adam(learning_rate=0.001)
model.compile(optimizer=optimizer, loss='mean_squared_error')

# Implement callbacks for early stopping and learning rate reduction
early_stop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3,
min_lr=0.0001)

# Train the model with callbacks
history = model.fit(X, y, batch_size=32, epochs=50, validation_split=0.2,
callbacks=[early_stop, reduce_lr])

# Prepare test data (last sequence of 2024)
X_test = np.expand_dims(scaled_data[-look_back:], axis=0)

# Forecast function
def forecast(model, start_sequence, days_to_predict):
    forecasted = []
    current_sequence = start_sequence
    for _ in range(days_to_predict):
        pred = model.predict(current_sequence, verbose=0)
        forecasted.append(pred[0])
        current_sequence = np.append(current_sequence[:, 1:, :], np.expand_dims(pred,
axis=1), axis=1)
```

```
    return forecasted
days_to_predict = 7
predicted_2025 = forecast(model, X_test, days_to_predict)
predicted_2025 = scaler.inverse_transform(np.array(predicted_2025).reshape(-1, 3))

# Create a DataFrame with dates and predicted values
start_date = datetime(2024, 5, 26)
date_list = [start_date + timedelta(days=x) for x in range(days_to_predict)]
predicted_df = pd.DataFrame({
    'Date': date_list,
    'Predicted_Temperature': predicted_2025[:, 0],
    'Predicted_SO2_Emission': predicted_2025[:, 1],
    'Predicted_NO2_Emission': predicted_2025[:, 2]
})
predicted_df

# Visualization of the predictions
plt.figure(figsize=(14, 8))

# Predicted Temperature
plt.subplot(3, 1, 1)
plt.plot(predicted_df['Date'], predicted_df['Predicted_Temperature'], marker='o')
plt.title('Predicted Temperature')
plt.ylabel('Temperature')

# SO2 Emission
plt.subplot(3, 1, 2)
plt.plot(predicted_df['Date'], predicted_df['Predicted_SO2_Emission'], marker='o')
plt.title('Predicted SO2 Emission')
plt.ylabel('SO2 Emission')

# NO2 Emission
plt.subplot(3, 1, 3)
plt.plot(predicted_df['Date'], predicted_df['Predicted_NO2_Emission'], marker='o')
plt.title('Predicted NO2 Emission')
plt.ylabel('NO2 Emission')
plt.xlabel('Date')
plt.tight_layout()
plt.show()
```

6.OUTPUT:-



```
         Date  Predicted_Temperature  Predicted_SO2_Emission  Predicted_NO2_Emission
0  2024-05-26              29.601793                6.455497               22.501266
1  2024-05-27              29.790995                6.464231               22.442711
2  2024-05-28              29.897797                6.462016               22.426647
3  2024-05-29              29.953074                6.449263               22.429354
4  2024-05-30              29.978947                6.428487               22.434593
5  2024-05-31              29.989927                6.402892               22.435240
6  2024-06-01              29.994486                6.375171               22.429909
```



6.1 INFERENCE:-

Our analytical framework offers dual data representation through Columns-Rows and Graph Mode, providing comprehensive visualizations for insightful interpretation and decision-making. While our model can predict temperature and emission trends over extended periods, we focus on weekly projections to allow for more granular analysis and nuanced insights into short-term fluctuations. Central to our approach is the use of Long Short-Term Memory (LSTM) networks, a type of recurrent neural network renowned for capturing temporal dependencies and modeling sequential data. By leveraging LSTM architecture, our model discerns intricate patterns within weather and greenhouse gas emission datasets, enabling robust forecasts. This methodology exemplifies the fusion of cutting-edge machine learning techniques with environmental science, facilitating informed decision-making and proactive interventions in climate-related domains.

7. PROJECT HURDLES:-

- Cleaning and normalizing raw dataset effectively.
- Choosing accurate, efficient prediction models.
- Extracting meaningful features from data.
- Avoiding overfitting, ensuring generalization capabilities.
- Ensuring predictions are understandable and explainable.
- Integrating models into existing systems seamlessly.
- Sourcing comprehensive, high-quality climate data.

8.FUTURE SCOPE:-

- Explore daily or hourly forecasts for real-time insights.
- Include factors like precipitation, and socio-economic data.
- Tailor predictions to specific geographic regions using GIS.
- Collaborate with policymakers for evidence-based decision-making.
- Forecast overarching climate trends for strategic planning.
- Harness emerging AI techniques for more accurate predictions.

9. CONCLUSION:-

In conclusion, our data science project on Climate Prediction and Mitigation demonstrates the powerful synergy between advanced machine learning techniques and environmental science. By employing Long Short-Term Memory (LSTM) networks, we have successfully developed a robust framework capable of accurately forecasting weather and greenhouse gas emission trends. The dual data representation in Columns-Rows and Graph Mode enhances the interpretability and usability of our results, allowing stakeholders to gain valuable insights and make informed decisions. Focusing on weekly projections provides a detailed understanding of short-term patterns and fluctuations, crucial for timely and effective climate interventions. Our project not only advances predictive modeling in the environmental domain but also underscores the critical role of data science in addressing global climate challenges and driving proactive mitigation strategies.

10. REFERENCES:-

1. www.python.org
2. scikit-learn.org
3. Developer guides (keras.io)
4. www.kaggle.com
5. dev.meteostat.net