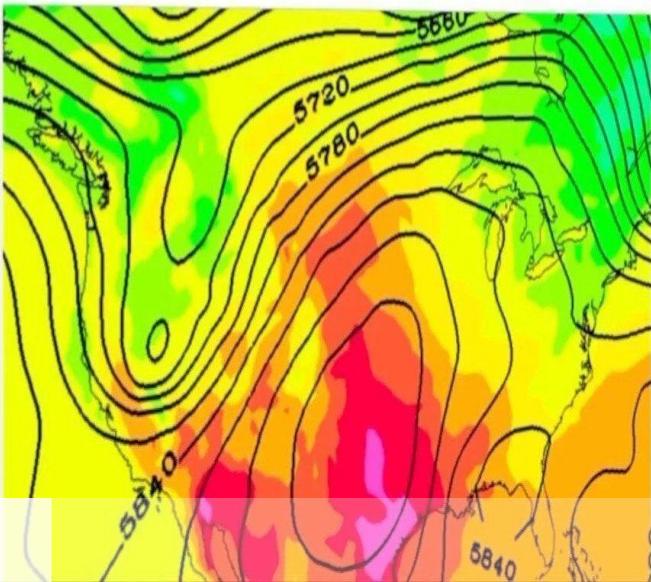


# Heat wave

Polythene orders  
in place; teachers,  
children made to  
remove hats before  
entering school at  
several locations

Super Powers



## Advancing Heat Resilience



Group ID - 2023-24-

# Meet Our Team



IT20142728  
Ranawaka.T.D



IT20145552  
Dissanayaka. D. M. S. M



IT20652500  
B.N.S. Gunadasa



IT20145552  
T.M.S.B. Thelv

# Introduction.

## Why Heatwave resilience is important?

- **Public health protection**

Even WHO has declared set of instructions to follow during heatwave conditions.

- **Provide Agricultural Resilience**

forecast soil moisture and recommend suitable crops which helps to farmers to mitigate the impact .

# Introduction.

## Why HeatGuard is important?

- **Early warning system**

Most people are not aware whether they are in a heatwave condition. This will help to reduce series health impacts

- **Public health protection**

Make people aware about heatwaves which brings significant risk to public health such as heat stroke, dehydration and heat exhaustion.

- **Provide Agricultural Resilience**

forecast soil moisture and recommend suitable crops which helps to farmers to mitigate the impact .

# Introduction.

## Though Out App,

- We do risk communication to the users and asses their functionalities and engagement with exposure to heat.
- The background research begins with examines the various impacts on different sectors such as public health , agricultural , environmental etc...

# Research Problems.

- How the data driven decision making can apply to predict heatwaves.
- How to forecast soil moisture and predict crops for mitigate the impacts for yields.
- How to apply data driven decision making to mitigate health risks.

# Background Research.

- **Heatwave Phenomenon:** Defines heatwaves, explores their effects on health, agriculture, infrastructure, and the environment.
- **Existing Research and Solutions:** Reviews previous studies on heatwave prediction, monitoring, and mitigation, identifying gaps and issues.
- **Mobile Applications for Heatwave Resilience:** Investigates the role of mobile apps in heatwave resilience, assessing features and user involvement strategies.

# Data Collection

Google Earth Engine

Search places and datasets...

Untitled File

```
* Imports (2 entries)
> var imagevisparam: ssm from 0 to 28
> var imagevisparam2: ssm from 0 to 28
1 // Define countries boundary
2 var Countries = ee.FeatureCollection('USDOES/LSIB_SIMPLE/2017');
3 var roi = Countries.filter(ee.Filter.eq('country_na', 'Sri Lanka'));
4 Map.addLayer(roi, {}, "Sri Lanka");
5 Map.centerObject(roi);
6
7 // List of years and months
8 var years = ee.List.sequence(2017, 2022); // Adjust as needed
9 var months = ee.List.sequence(1, 12);
10
11 // Load SMAP Data
12 var coll = ee.ImageCollection('NASA_USDA/HSL/SMAP10KM_soil_moisture');
13
14 // Filter data for the years 2017 to 2022
15 coll = coll.filterDate('2017-01-01', '2022-12-31');
16
17 // Check if the image collection contains valid images
18 if (coll.size().getInfo() === 0) {
19   print("No valid images found in the specified time period.");
20 } else {
```

Inspector   Console   Tasks

Search or cancel multiple tasks in the Task Manager

UNSUBMITTED TASKS

- Soil\_Moisture\_Sri\_Lanka\_2017\_to\_2022 **RUN**

SUBMITTED TASKS

- Soil\_Moisture\_Sri\_Lanka\_2017\_to\_2022 ✓ 2m
- Soil\_Moisture\_Sri\_Lanka\_2017 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka\_2017 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka\_2017 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka\_2017 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka3 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka3 ✓ <1m
- Soil\_Moisture\_Sri\_Lanka3 ✓ <1m

Layers   Map   Satellite

# Data Collection

kaggle

+ Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More

View Active Events

Search

ATHARVA INGLE · UPDATED 3 YEARS AGO

352

New Notebook

Download (65 kB)

Crop Recommendation Dataset

Maximize agricultural yield by recommending appropriate crops

Data Card    Code (135)    Discussion (9)    Suggestions (0)

About Dataset

Context

Precision agriculture is in trend nowadays. It helps the farmers to get informed decision about the farming strategy. Here, I present you a dataset which would allow the users to build a predictive model to recommend the most suitable crops to grow in a particular farm based on various parameters.

Context

This dataset was build by augmenting datasets of rainfall, climate and fertilizer data available for India.

Data fields

- N - ratio of Nitrogen content in soil

Usability 5.88

License Unknown

Expected update frequency Not specified

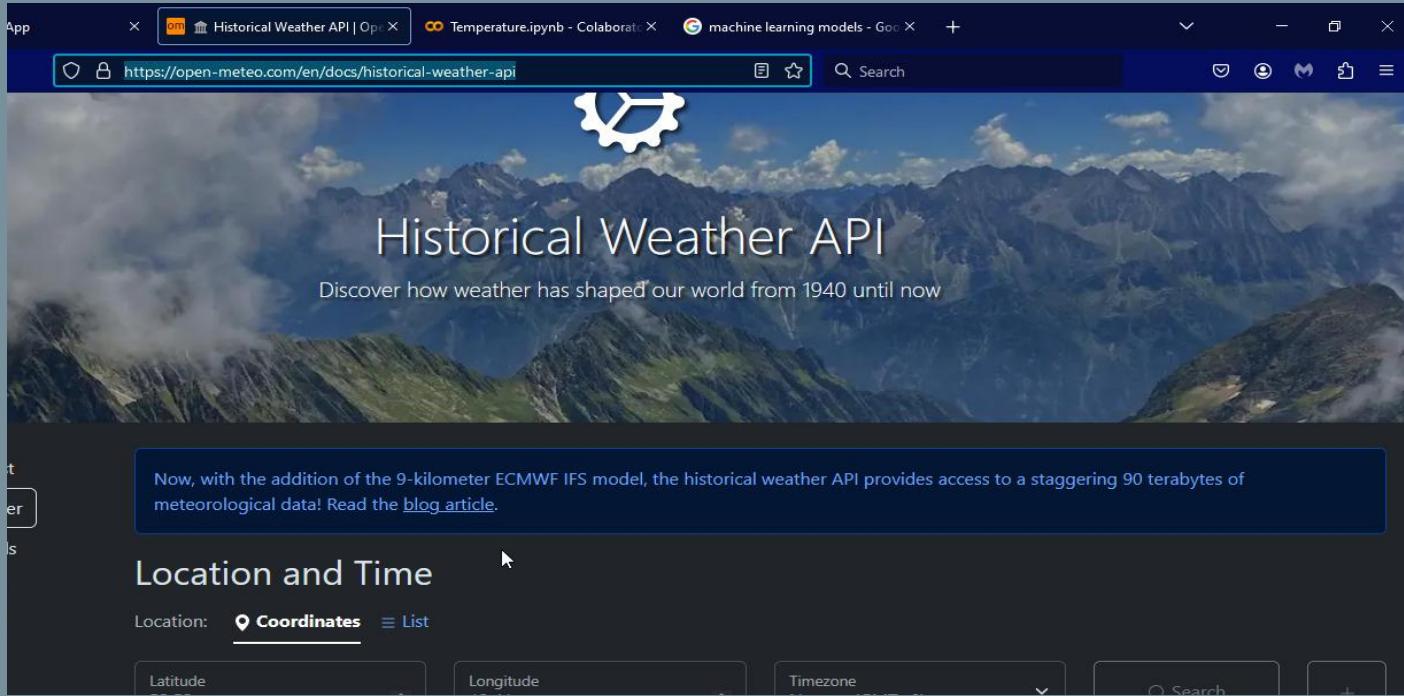
Tags

Tabular Agriculture

Recommender Systems



# Data Collection



The screenshot shows a web browser window with three tabs open. The active tab is titled "Historical Weather API | Open-Meteo". The URL in the address bar is <https://open-meteo.com/en/docs/historical-weather-api>. The page features a large background image of a mountain range under a cloudy sky. In the center, there is a white gear icon above the text "Historical Weather API" and a subtitle "Discover how weather has shaped our world from 1940 until now". A blue callout box contains the text: "Now, with the addition of the 9-kilometer ECMWF IFS model, the historical weather API provides access to a staggering 90 terabytes of meteorological data! Read the [blog article](#)." Below this, there is a section titled "Location and Time" with input fields for "Latitude", "Longitude", "Timezone", and a "Search" button.



**Ranawaka . T . D**  
**IT 20142728**

Specialization - Information Technology

SubTopic: Soil Moisture Forecasting and Crop R

# Introduction to Soil Moisture Forecasting and Crop Recommendation

## Definition and Importance:

- Soil moisture forecasting predicts soil water levels based on weather and soil data, helping farmers plan irrigation efficiently.
- Crop recommendation suggests the best crops to plant, considering soil moisture, climate, and soil type. This approach enhances productivity, conserves water, and boosts resilience to climate change, supporting sustainable and profitable farming practices.

- We took the soil moisture values for 28cm to 100cm in soil. This is the the range of root level in soil. Soil moisture is taken as volumetric measure
- According to the dataset we can identify the behaviour of soil moisture vary with temperature.



# Research Question

- How to forecast soil moisture accurately ?
- How can machine learning models improve their accuracy and dependability of predicting crops ?
- How to implement a system that cultivators can monitor soil moisture for their crops ?

# Research Gap

Features	HeatAlert+	Crisis24 Horizon Mibile	HeatGuard
Recommend Crops	✗	✗	✓
Soil moisture Forecasting	✗	✗	✓
Support for Sri Lanka	✗	✓	✓

# Novelty

- Integrating Soil moisture forecasting with crop prediction.
- There isn't a mobile app that supports for Sri Lankan geography acco

# Main Objective

- Create a system for monitoring soil moisture accurately and real time to effectively manage water requirements.

## Sub Objectives

- Use machine learning models to predict crops according to soil moisture levels.



# Tools and Algorithms

# Technologies

- **Algorithm :**RandomForestRegressor / RandomForestClassifier
- **Develop Mobile Application :** Flutter
- **Model Trainer:** google colab
- **Languages:** Dart
-

# Methodology

## Data Collection:

- Data collection forms the cornerstone of accurate soil moisture forecasting and effective crop recommendations

# Methodology

## Data Sources:

- **Meteorological Data:** Historical and real-time weather data, including rainfall, temperature, humidity, and solar radiation, obtained from national meteorological agencies and satellite imagery.
- **Soil Data:** Soil moisture levels, soil type, texture, and organic content sourced from field surveys, agricultural research institutions, and remote sensing technologies.
- **Crop Data:** Information on crop types, growth stages, water requirements, and yield data from agricultural databases and farmer records.
- **Geographical Data:** Topography, land use, and irrigation infrastructure data derived from remote sensing sources.

# Methodology

## Model Development:

Developing accurate soil moisture forecasting models and crop recommendation algorithms involves several key steps.

### Forecasting Models:

- Data Preprocessing
- Model Selection
- Training and Validation

# Methodology

## Validation Procedures:

Validation ensures the reliability and practical applicability of the soil moisture forecasts and crop yield recommendations

### Validation Methods:

- Cross-Validation
- Field Trials
- Stakeholder Feedback

# Methodology

## Steps in App Development:

- **Design Phase:** Create intuitive and user-friendly interfaces. Focus on ease of navigation and accessibility for farmers with varying levels of technological proficiency.
- **Development Phase:** Code the application using technologies. Integrate GIS mapping functionalities to visualize soil moisture data and crop recommendations.
- **Data Integration:** Connect the app to databases and APIs providing real-time weather data. Ensure seamless integration and accurate data representation.
- **Testing Phase:** Conduct rigorous testing to identify and resolve bugs.

# Completion of the project

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files and folders including `run.py`, `TempPredictionNotebook.ipynb`, `models` (containing `crop_recommendation`, `soil_moisture_prediction`, and `temp_prediction`), and `scaler.joblib`.
- Code Editor:** Displays the content of `TempPredictionNotebook.ipynb`. The code performs the following steps:
  - Loads a dataset from a CSV file.
  - Handles missing values and normalizes features by extracting datetime features (year, month, day, hour).
  - Normalizes features using a StandardScaler.
  - Splits the data into training and validation sets.
  - Trains a Random Forest Regressor model with 100 estimators.
- Terminal:** Shows a command-line session with the following output:

```
feature names, but StandardScaler was fitted with feature names
warnings.warn('Temperature prediction: 24.516999999999985
127.0.0.1 - [19/Mar/2024 02:02:46] "POST /getTempPred HTTP/1.1" 200 -
PS E:\PP2\Project updated> History restored
```
- Status Bar:** Shows "Spaces: 4 Cell 1 of 10 Go Live Spell".

# What's to be done

- Integrate components.
- Connecting IoT device for data input.



**Dissanayaka. D. M. S. M**  
**IT 20145552**

Specialization - Information Technology

SubTopic: Heat Vulnerability Index (HVI) Specifi



# Research Question

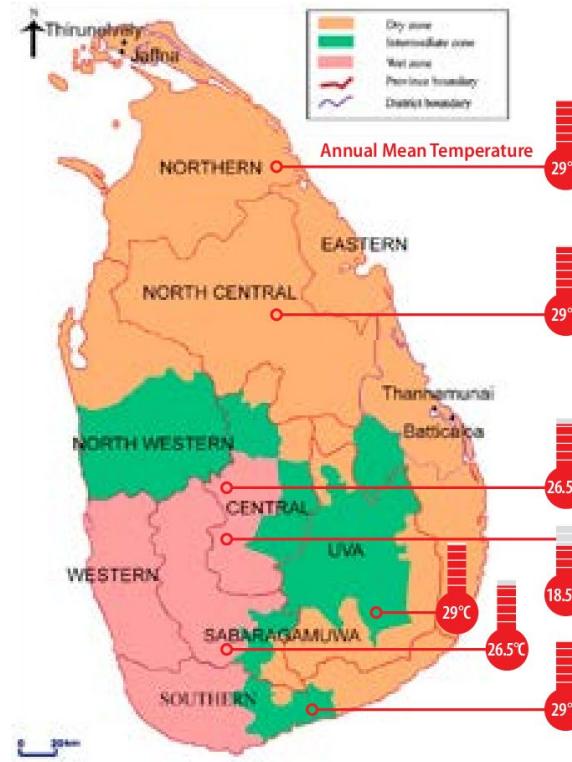
- What are the key elements contributing to vulnerability to heat-related risks in different **geographical areas**?
- How can machine learning algorithms be effectively utilized to process large datasets related to **heatwaves**?
- What are the most suitable data sources for developing a **heatwave** system?

# Research Gap

Inability to obtain information on heat wave resolution and the inability to obtain information simultaneously as a **single Location (District)**.



# Research Gap



# Research Gap

Features	HeatAlert+	Crisis24 Horizon
Prediction Crops (Recommend)	✓	✗
Suggest Irrigation levels	✓	✗
Heat Wave Prediction	✓	✗
Support for Sri Lanka	✓	✗

# Main Objective

- Develop machine learning algorithms capable of accurately analyzing heat waves datasets to identify localized vulnerability patterns to heat waves related risks in Sri Lanka



# Technologies Tools and Algorithms

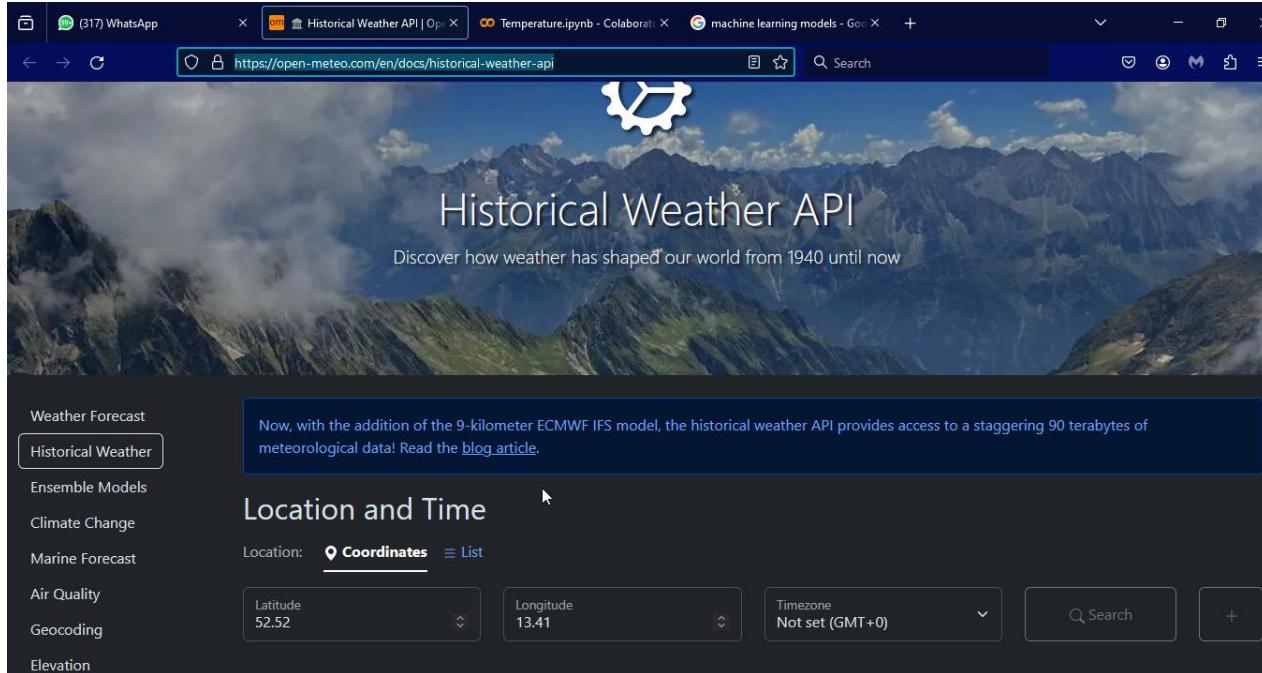
- **Algorithm :**RandomForestRegressor
- **Develop Mobile Application :** Flutter
- **Model Trainer:** google colab
- **Languages:** Dart



# Project Evidence

- Dataset given from that link:

<https://open-meteo.com/en/docs/historical-weather-api>



# Project Evidence

- Dataset:

The screenshot shows a Microsoft Excel spreadsheet titled "new\_data.csv - Excel". The window title bar also includes "IT - Manager of RALL". The ribbon menu is visible at the top, and the formula bar shows "E9" and "63". The main content is a table with the following data:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	longitude	latitude	time	temperature	relative_humidity_2m (%)	rain (mm)												
2	80.44992	7.4868193	2010-01-01T00:00	21.6	92	0												
3	80.44992	7.4868193	2010-01-01T01:00	21.7	91	0												
4	80.44992	7.4868193	2010-01-01T02:00	22.6	89	0												
5	80.44992	7.4868193	2010-01-01T03:00	24.4	82	0												
6	80.44992	7.4868193	2010-01-01T04:00	25.8	74	0												
7	80.44992	7.4868193	2010-01-01T05:00	26.7	68	0												
8	80.44992	7.4868193	2010-01-01T06:00	27.4	65	0												
9	80.44992	7.4868193	2010-01-01T07:00	27.9	63	0												
10	80.44992	7.4868193	2010-01-01T08:00	27.8	64	0												
11	80.44992	7.4868193	2010-01-01T09:00	28.1	63	0												
12	80.44992	7.4868193	2010-01-01T10:00	27.6	65	0												
13	80.44992	7.4868193	2010-01-01T11:00	27	68	0												
14	80.44992	7.4868193	2010-01-01T12:00	26.1	71	0												
15	80.44992	7.4868193	2010-01-01T13:00	25.4	73	0												
16	80.44992	7.4868193	2010-01-01T14:00	24.6	77	0												
17	80.44992	7.4868193	2010-01-01T15:00	24.2	80	0												
18	80.44992	7.4868193	2010-01-01T16:00	23.8	82	0												
19	80.44992	7.4868193	2010-01-01T17:00	23.6	83	0												
20	80.44992	7.4868193	2010-01-01T18:00	23.2	86	0												
21	80.44992	7.4868193	2010-01-01T19:00	23	85	0												
22	80.44992	7.4868193	2010-01-01T20:00	22.5	87	0												

# Methodology

## Data Collection:

- Data collection is the foundation of the HVI and the app, involving the aggregation of diverse datasets to capture the multifaceted nature of heat vulnerability.

# Methodology

## Data Sources:

- **Meteorological Data:** Temperature records, frequency, duration, and intensity of heatwaves obtained from the Sri Lanka Meteorological Department.
- **Socioeconomic Data:** Census data on income levels, employment, education, and housing conditions sourced from the Department of Census and Statistics, Sri Lanka.
- **Demographic Data:** Information on population density, age distribution, and health status collected from national health surveys and census reports.
- **Environmental Data:** Land use, vegetation cover, and urban heat island effect data derived from remote sensing (satellite imagery) and geographic information systems (GIS).

# Completion of the Project

# CODE EVIDENCE

The screenshot shows a Jupyter Notebook interface with the following details:

- File Explorer:** Shows the project structure with files like run.py, TempPredictionNotebook.ipynb, and TempPredictionNotebook.ipynb.
- Code Editor:** Displays a Python script for "TempPredictionNotebook.ipynb". The code performs the following steps:
  - Loads a dataset from a CSV file.
  - Handles missing values, normalizes features, and extracts datetime features (year, month, day, hour).
  - Selects features (longitude, latitude, year, month, day, hour) and the target variable (temperature).
  - Normalizes features using StandardScaler.
  - Splits the data into training and validation sets.
  - Trains a Random Forest Regressor model with 100 estimators.
- Terminal:** Shows the command used to run the notebook: `python run.py`.
- Status Bar:** Shows the path as E:\P2P\NProject\_updated, the current file as History restored, and other status indicators like battery level and signal strength.
- Bottom Right:** A notification from Microsoft about the Python extension.

The screenshot shows a Jupyter Notebook environment with the following details:

- File Structure:** The left sidebar displays the file tree under "NPROJECT\_UPDATED".
- Open Editors:** Two files are open: "run.py" and "TempPredictionNotebook.ipynb".
- Code in run.py:**

```
from flask import Flask, request, jsonify
from joblib import load
import os

# Flask app object
app = Flask(__name__)

# Load the models from the file
CropRecModel = joblib.load(os.path.join('models', 'crop_recommendation', 'CropRecModel.pkl'))
SoilMoistureModel = joblib.load(os.path.join('models', 'soil_moisture_prediction', 'SoilMoistureModel.pkl'))
TempPredModel = joblib.load(os.path.join('models', 'temp_prediction', 'TempPredModel.pkl'))

# Route to handle the Crop Recommendation
@app.route('/getCropRec', methods=['POST', 'GET'])
def crop_recommendation():
    data = request.get_json()
    temperature = data['temperature']
    humidity = data['humidity']
    ph = data['ph']
    rainfall = data['rainfall']

    # Get recommendations
    input_data = [[temperature, humidity, ph, rainfall]]
    num_of_recommendations = 3

    probabilities = CropRecModel.predict_proba(input_data)

    # Load the fitted transformation
    # warnings.warn("StandardScaler was fitted with feature names names", UserWarning)

    return jsonify({'probabilities': probabilities[0].tolist()})

if __name__ == '__main__':
    app.run(host='0.0.0.1', port=8000)
```
- PROBLEMS:** 21
- OUTPUT:** TERMINAL
- PORTS:** DEBUG CONSOLE
- FEATURES:** feature names, but StandardScaler was fitted with feature names  
warnings.warn(  
Temperature prediction: 24.516999999999985  
127.0.0.1 - [19/Mar/2024 02:02:46] "POST /getTempPred HTTP/1.1" 200 -  
PS E:\P2P\NProject\_updated>
- HISTORY:** History restored

# Completion of the Project

## MODEL RUNNING

The screenshot shows a Google Colab interface with a Jupyter notebook titled "Temperature.ipynb". The notebook contains Python code for data analysis, specifically reading a CSV file and performing regression modeling. The code includes imports for pandas, train-test split, Random Forest Regressor, mean\_squared\_error, StandardScaler, accuracy\_score, joblib, r2\_score, and matplotlib.pyplot. It also includes a cell for loading the dataset from "new\_data.csv". The interface includes a sidebar for files, a toolbar with various icons, and status bars at the bottom indicating disk usage and connection status.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
import joblib
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt

# Read Data Set
# Load your dataset (assuming it's in a CSV format)
data = pd.read_csv('new_data.csv')

# Data Description
```

# Completion of the Project

## MODEL RUNNING

The screenshot shows a Google Colab interface with a Jupyter notebook titled "Temperature.ipynb". The code cell [4] contains the command `data.describe()`. The resulting output is a table showing statistical summary data for columns: longitude, latitude, temperature, relative\_humidity\_2m (%), and rain (mm). The table includes rows for count, mean, std, min, 25%, 50%, 75%, and max.

	longitude	latitude	temperature	relative_humidity_2m (%)	rain (mm)
count	177398.000000	177398.000000	177398.000000	177398.000000	177398.000000
mean	81.072362	7.366794	26.55520	79.671935	0.198046
std	0.618114	0.169435	2.96076	13.822294	0.832390
min	80.449920	6.151143	16.60000	15.000000	0.000000
25%	80.449920	7.275923	24.30000	72.000000	0.000000
50%	81.127815	7.275923	26.10000	83.000000	0.000000
75%	81.693474	7.486819	28.30000	90.000000	0.000000
max	81.693474	7.486819	39.90000	100.000000	35.300000

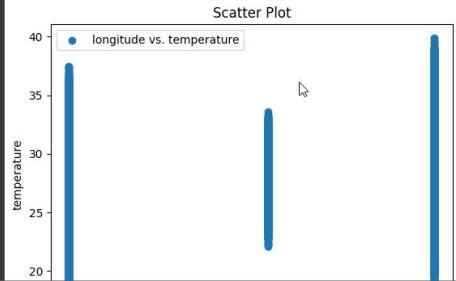
Below the code cell, there is a message: `Start coding or generate with AI.`. At the bottom of the notebook, there is a note: `Connected to Python 3 Google Compute Engine backend`.

# Completion of the Project

## MODEL RUNNING

Temperature.ipynb

```
45 plt.scatter(data['longitude'], data['temperature'], label='longitude vs. temperature')
plt.xlabel('longitude')
plt.ylabel('temperature')
plt.legend()
plt.title('Scatter Plot')
plt.show()
```



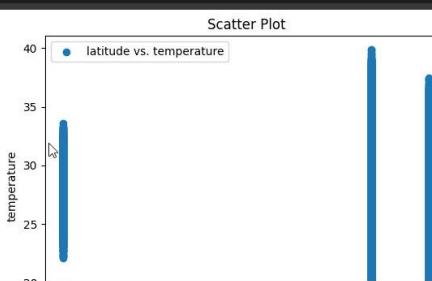
Scatter Plot

longitude	temperature
35	35
38	38

Connected to Python 3 Google Compute Engine backend

Temperature.ipynb

```
2s plt.scatter(data['latitude'], data['temperature'], label='latitude vs. temperature')
plt.xlabel('latitude')
plt.ylabel('temperature')
plt.legend()
plt.title('Scatter Plot')
plt.show()
```



Scatter Plot

latitude	temperature
32	32
38	38

Connected to Python 3 Google Compute Engine backend

# Completion of the Project

## UI EVIDENCE

The screenshot shows a development environment with the following details:

- IDE:** Visual Studio Code (VS Code) is used for editing the code.
- Project Structure:** The project is named "HEATGUARDV3". It contains several sub-directories like "Assets", "build", "app", "flutter\_assets", "kotlin", "native\_assets", "path\_provider\_android", "windows", "ios", "lib", "components", "core", "models", "routes", "screens", "widgets", "const.dart", and "main.dart".
- Code Editor:** The main.dart file is open in the editor. The code defines a StatelessWidget named MyApp that returns a MaterialApp. It sets the title to 'Heatguard', theme to ThemeData, and primarySwatch to Colors.blue. The routes are defined as follows:
  - "/": HomeScreen()
  - /heatwaves: HeatwavesScreen()
  - /crops: CropsScreen()
  - /health: HealthScreen()
  - /soilmoisture: SoilMoistureScreen()
- Emulator:** An Android emulator (MyAvid5554) is running, showing the app's splash screen. The screen displays the app's logo and four circular icons representing different features: Heatwaves, Crops, Soil Moisture, and Health. Below the icons, weather information is shown: Max: 24°C, Min: 23°C, Wind: 3m/s, and Humidity: 68%.
- Bottom Bar:** The VS Code status bar at the bottom shows the following information: Line 1, Col 1, Spaces: 2, UTF-8, CR/LF, Dart, Go Live, MyAvid (Android-x64 emulator), Prettier, and a Java Warning.

# Completion of the Project

## UI EVIDENCE

The screenshot shows a development environment for a Flutter application named "HEATGUARD3".

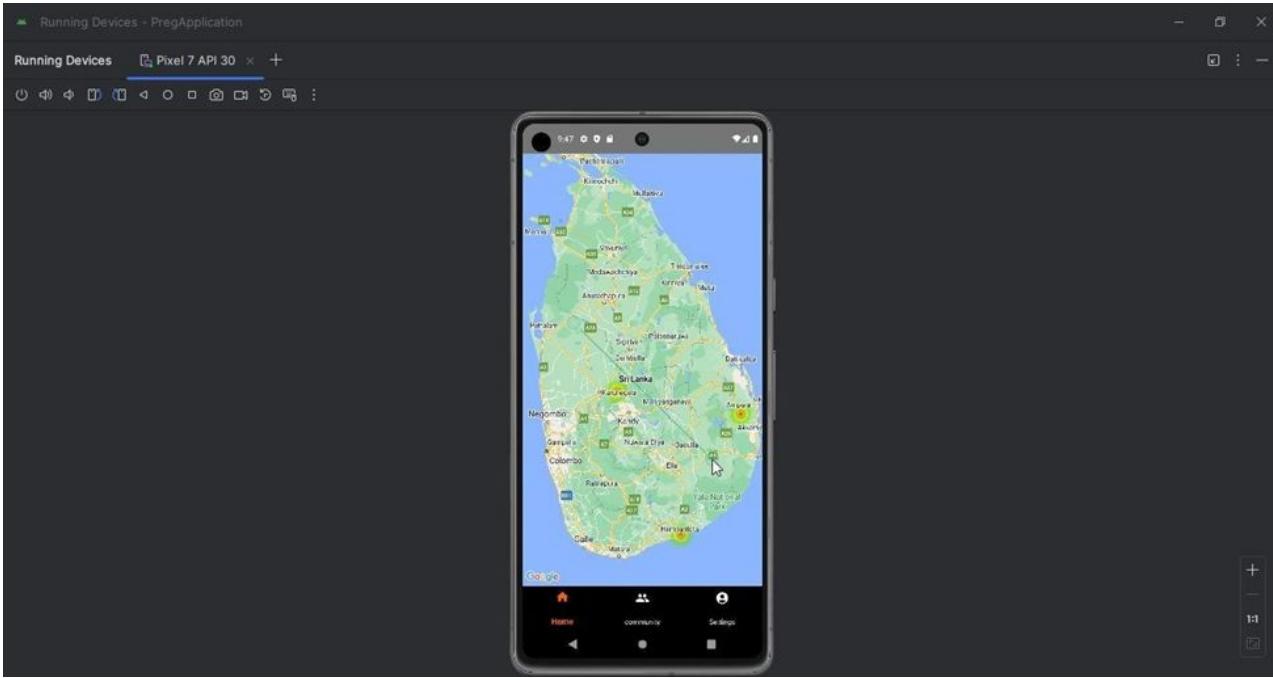
**VS Code Explorer:** Shows the project structure with files like .dart\_tool, android, Assets, build, lib, and test.

**VS Code Editor:** Displays the main.dart file:lib/main.dart
5 import 'package:heatguardv3/screens/crops\_screen.dart';
6 import 'package:heatguardv3/screens/soil\_moisture.dart';
7 import 'package:heatguardv3/screens/health\_screen.dart';
8
9
10 Run/Debug/Profile
11 void main() {
12 runApp(MyApp());
13 }
14
15 class MyApp extends StatelessWidget {
16 @override
17 Widget build(BuildContext context) {
18 return MaterialApp(
19 debugShowCheckedModeBanner: false,
20 title: 'HeatGuard',
21 theme: ThemeData(
22 primarySwatch: Colors.blue,
23 ), // ThemeData
24 home: SplashScreen(), // Custom splash screen
25 routes: [
26 AppRoutes.home: (context) => HomeScreen(),
27 AppRoutes.heatwaves: (context) => HeatwavesScreen(),
28 AppRoutes.crops: (context) => CropsScreen(),
29 AppRoutes.health: (context) => HealthScreen(),
30 AppRoutes.soilmoisture: (context) => SoilMoistureScreen(),
31 ],
32 );
33 }
34 AppBar appBar() {
35 return AppBar(
36 backgroundColor: const Color.fromRGBO(255, 200, 15, 1),
37 );
38 }
39 }
40
41

**Android Emulator:** Shows the "HeatGuard" app running on an "MyArd5554" device. The screen displays a weather forecast for Colombo, Sri Lanka, with a "PREDICT" button at the bottom.

# Completion of the Project

## UI EVIDENCE



# What's to be done

- Finished develop the application.
- Ensure smooth communication between frontend and backend.
- Test the integrated system for functionality and performance.

# Risk Mitigation

- Prioritize vital features to manage resource constraints effectively.
- Gather user feedback early on to streamline user interface design and reduce complexity.
- Implement robust security actions to protect user data from potential breaches.
- Maintain open communication with users
- Monitor and adjust project plans to mitigate taking off risks effectively.

*Thank You*