INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

ROBERT GORDON UNIVERSITY ABERDEEN

CM2606- Data Engineering

Course Work

Course Work Documentation by

Janaka Dilshan Sendanayake

2237952 (RGU)

20221496 (IIT)

Supervised by

Module Co-ordinator / Lecturer – Mr. Mohamed Ayoob

Submitted in partial fulfilment of the requirements for the BSc (Hons) Artificial Intelligence and Data Science degree at the Robert Gordon University.

April 2024

Table of Contents

1 Understanding and Predicting Daily Formaldehyde Emissions in Sri Lankan Cities: A Spatial-Temporal Analysis and Machine Learning Approach

1.1 Introduction

This research investigates daily formaldehyde (HCHO) emissions in several cities in Sri Lanka, aiming to understand their patterns and influences. By conducting spatial-temporal analysis, we explore the impact of various factors such as weather conditions (temperature, precipitation), geographical features (elevation, proximity to the sea), demographic indicators, anthropogenic activities, and atmospheric components correlated with human actions (CO, NO2, O3), along with COVID-19 data.

Through the application of machine learning and statistical models including LSTM, FB Prophet, and SARIMA, we seek to predict HCHO levels in these cities. Additionally, a Power BI dashboard has been developed to facilitate easy data exploration for future reference by researchers and stakeholders.

1.2 Importance and impact of this project

This research highlights the significance of its findings regarding formaldehyde (HCHO) emissions and their implications for air quality management, public health, and environmental sustainability. Elevated HCHO levels pose health risks, including respiratory issues and cancer, particularly in urban areas. Addressing HCHO emissions is crucial for safeguarding community health and well-being. Moreover, reducing HCHO emissions not only improves air quality but also yields broader benefits such as enhanced quality of life, increased productivity, and reduced healthcare costs. Furthermore, mitigating HCHO emissions aligns with global efforts to combat climate change. Effective communication of these findings is vital for mobilizing support for evidence-based policies aimed at improving air quality and protecting public health.

1.3 Ethical Considerations

This research project was conducted using publicly available data sources, and proper credit has been attributed to all data providers. Utilizing publicly available data helps uphold ethical standards by ensuring transparency and accountability in the research process. By

acknowledging the sources of data, we honor the contributions of data providers and uphold the principles of academic integrity.

2 Data Collection and Preparation

2.1 City Wise HCHO Readings

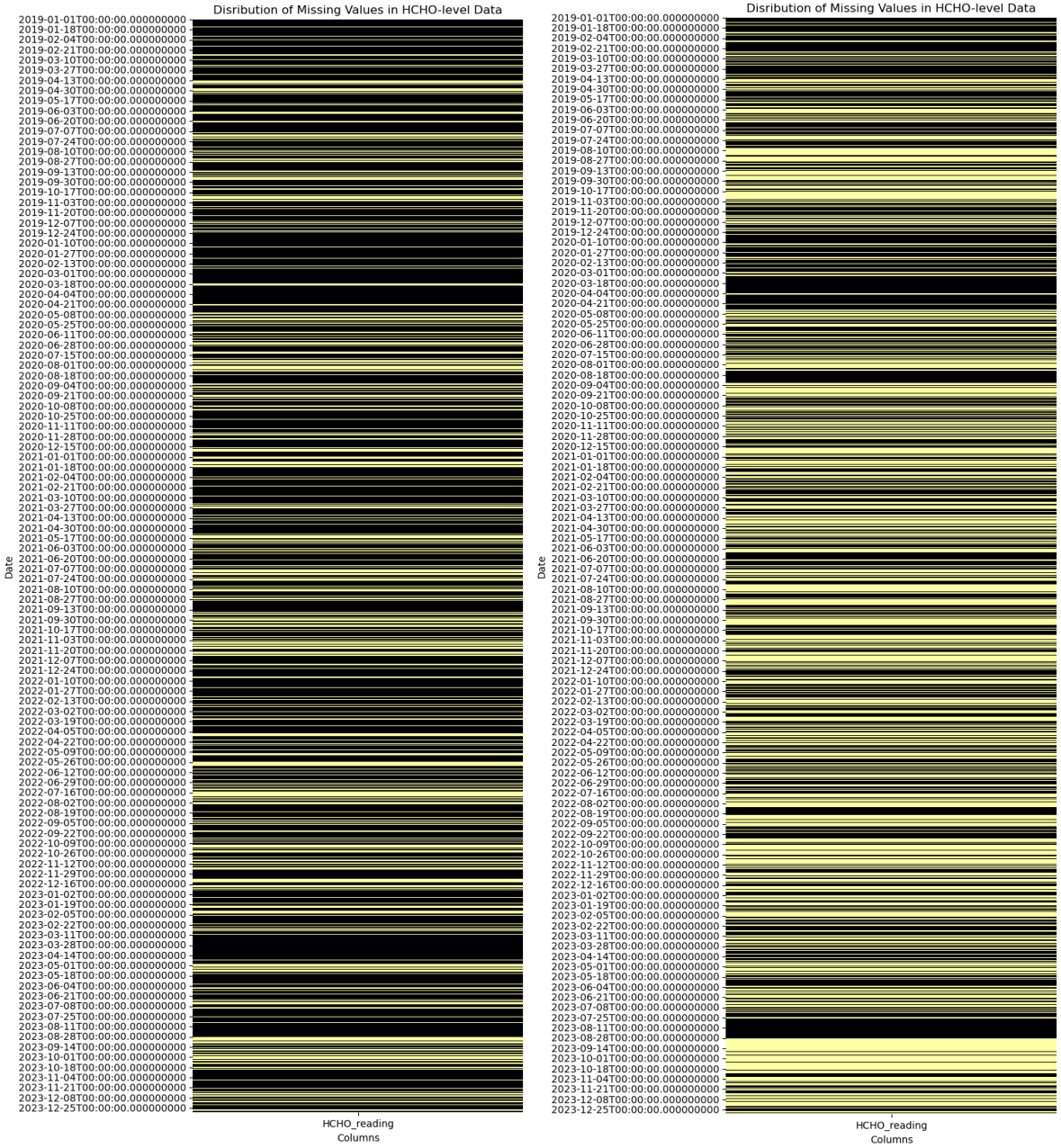
The Formaldehyde dataset from the TROPOMI satellite instrument offers insights into atmospheric formaldehyde concentrations. Analyses were conducted for several cities including Bibile, Colombo Proper, Deniyaya, Jaffna Proper, Kandy Proper, Kurunegala Proper, and Nuwara Eliya Proper, spanning from January 1, 2019, to December 31, 2023. This information aids in monitoring pollution levels, understanding atmospheric chemistry, and assessing air quality dynamics on regional scales, contributing to efforts to mitigate air pollution's impacts on human health and the environment.

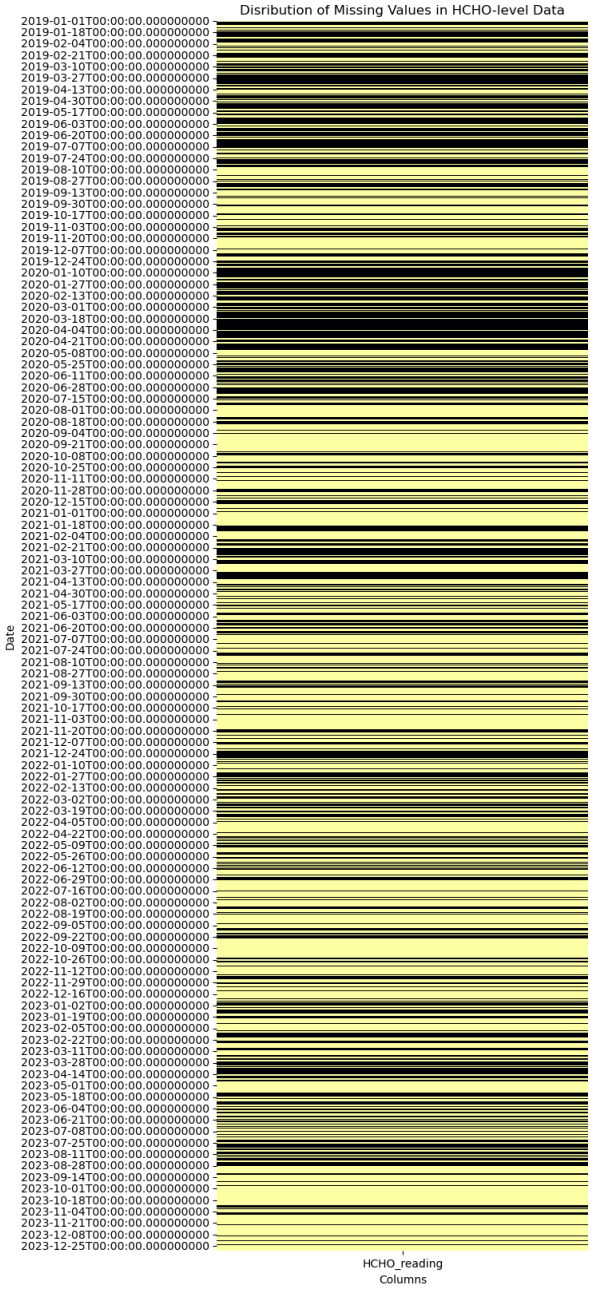
The dataset face limitations in providing continuous coverage across all regions. These limitations stem from several factors:

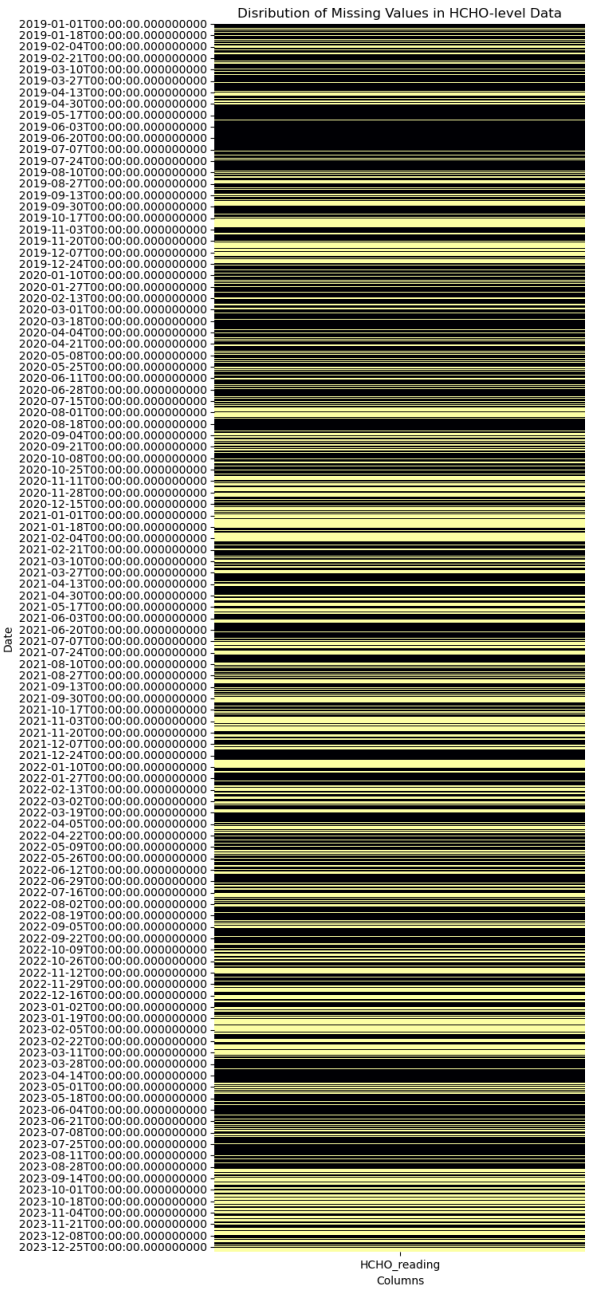
1. **Orbit Constraints**: Satellites follow predefined orbital paths around the Earth, which means they may not pass over certain areas at all times. This particularly affects regions at higher latitudes or far from the equator, which might have less frequent satellite coverage.
2. **Cloud Cover**: Clouds can obscure the view of the Earth's surface from space, hindering satellite instruments from effectively measuring atmospheric composition. In areas prone to frequent cloud cover, such as tropical regions or coastal areas, data availability may be limited.
3. **Sunlight Dependency**: Many satellite instruments rely on sunlight to make measurements. Consequently, regions experiencing extended periods of darkness, such as polar regions during winter months, may have reduced or no data availability due to the absence of sufficient sunlight for measurements.

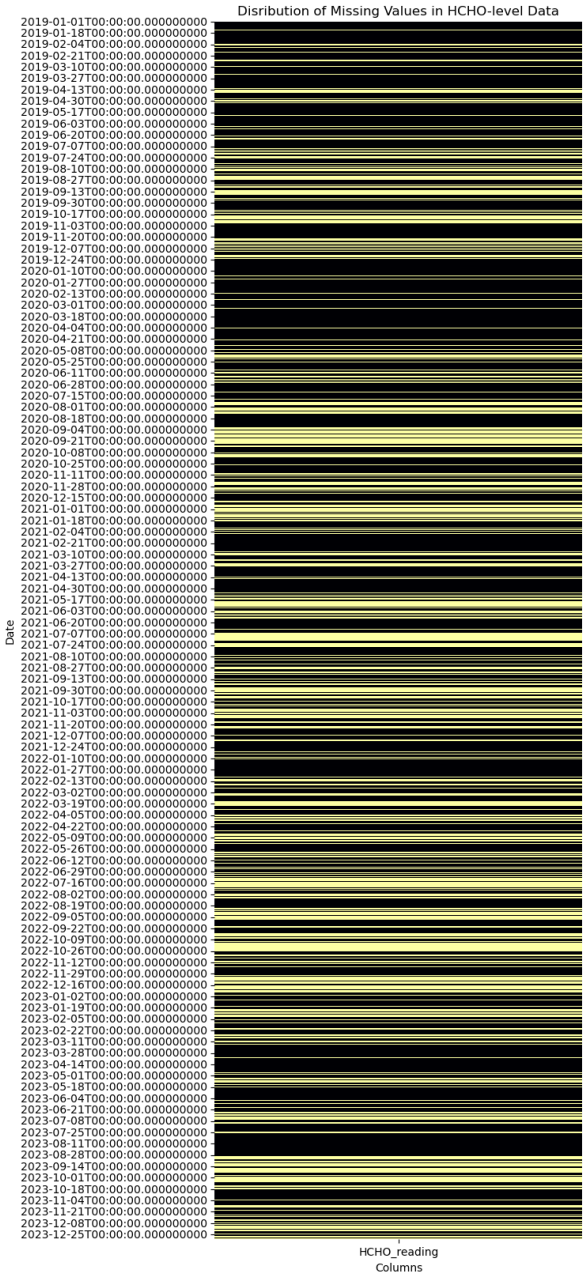
These factors collectively contribute to periods of data unavailability for certain regions or times, necessitating careful consideration when interpreting satellite-derived atmospheric datasets.

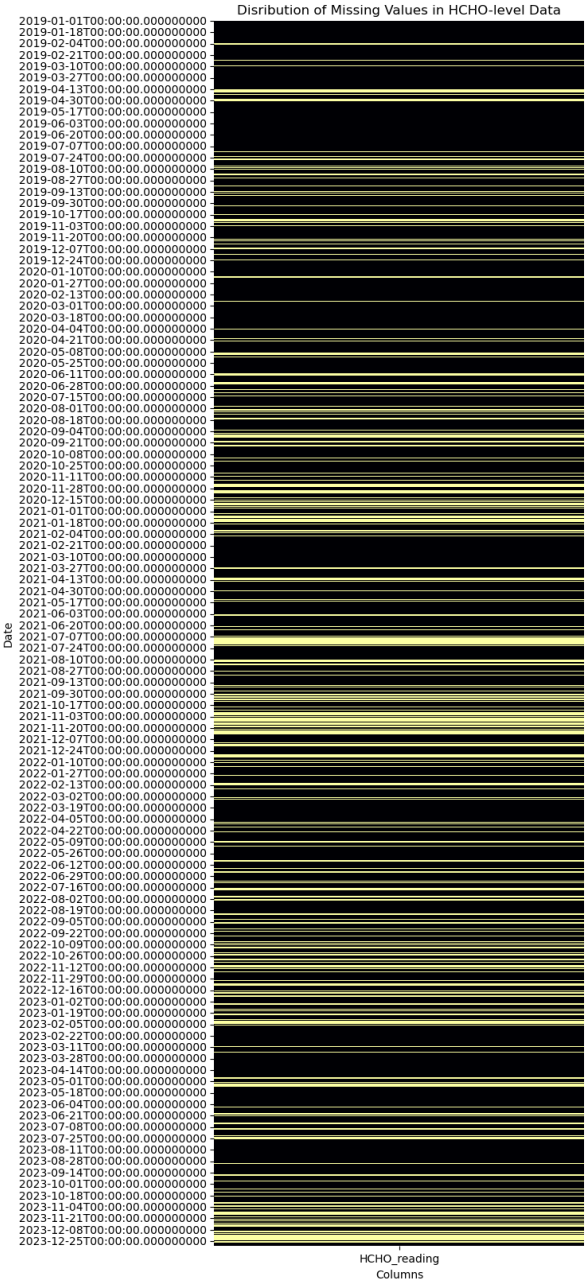
Distribution of null values across the dataset for Colombo, Matara, Nuwara Eliya, Bibile, Kurunegala, Jaffna and Kandy respectively.

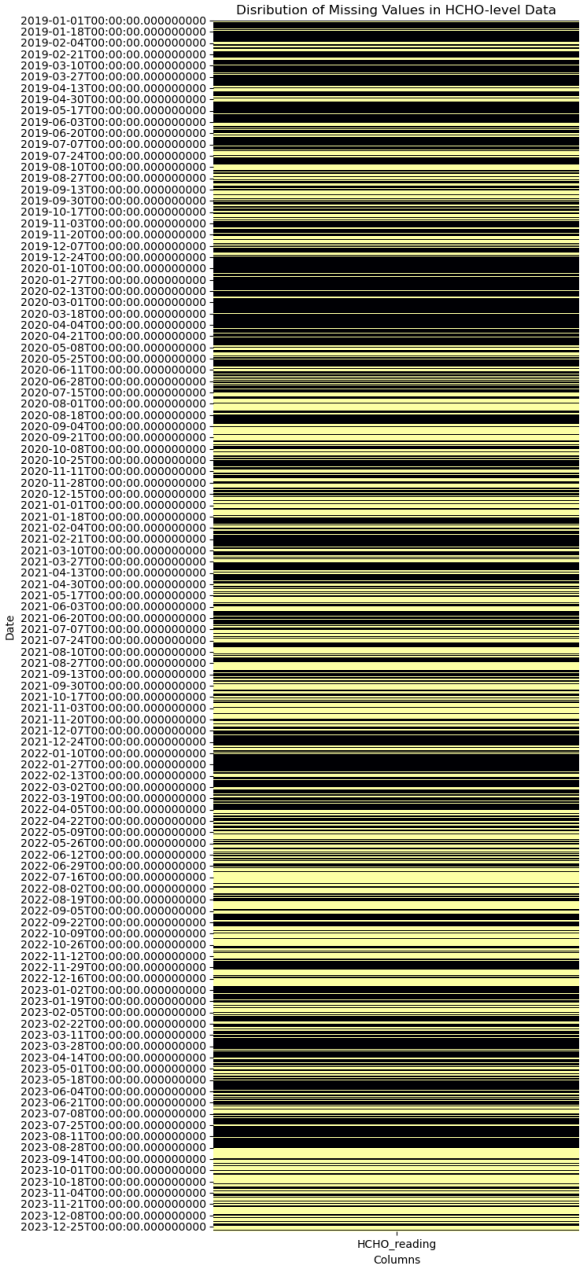



Disribution of Missing Values in HCHO-level Data
Date
HCHO_reading
Columns


Disribution of Missing Values in HCHO-level Data
Date
HCHO_reading
Columns


Disribution of Missing Values in HCHO-level Data
Date
HCHO_reading
Columns


Disribution of Missing Values in HCHO-level Data
Date
HCHO_reading
Columns



Null values in the dataset were addressed using various methods including the rolling window method, backward fill (bfill), forward fill (ffill), and a novel approach where null values were substituted with the mean of the same date from other years. These techniques were applied to each city individually. Subsequently, the effectiveness of each method in capturing seasonality was evaluated using autocorrelation plots. The method that best captured seasonality for each city, as determined by the autocorrelation analysis, was chosen as the final null imputation method.

2.2 City Wise weather data

The dataset comprises climate data gathered from two distinct sources, each contributing information for different sets of cities:

Climate Data Online (CDO) | National Climatic Data Center (NCDC) (noaa.gov):

- For the cities of Colombo, Kurunegala, and Nuwara Eliya, climate data on temperature and precipitation were obtained from the Climate Data Online (CDO) service provided by the National Climatic Data Center (NCDC), accessible through the website of the National Oceanic and Atmospheric Administration (NOAA).
- The National Climatic Data Center (NCDC) is a division of NOAA, responsible for archiving, monitoring, and providing access to climatic data collected from various sources, including weather stations, satellites, and other observation systems.
- NCDC is widely recognized for its authority in climate data management and quality assurance, ensuring the reliability and accuracy of the datasets it maintains.
- Data obtained from CDO is considered highly reliable and is widely used for domain related work.

NASA Power | NASA Langley Research Center (power.larc.nasa.gov):

- Climate data for the cities of Bibile (Monaragala), Deniyaya (Matara), Jaffna Proper, and Kandy Proper were sourced from NASA Power, provided by the NASA Langley Research Center.
- NASA Power provides access to various climate data products derived from satellite observations and numerical weather models, offering a global perspective on climate conditions.
- The Langley Research Center is one of NASA's major research centers, specializing in atmospheric and climate science research, satellite missions, and data analysis.
- Nonetheless, NASA Power data is widely utilized in climate research, remote sensing applications, and environmental monitoring due to its global coverage and accessibility.

The merged dataset includes standardized climate observations, such as temperature averages (TAVG), daily maximum (TMAX) and minimum (TMIN) temperatures, and precipitation

(PRCP), enabling comprehensive analysis and exploration of climate trends and their effect on HCHO emissions across different regions in Sri Lanka.

After preparing weather data other geographical data of each cities were also added. The main data source was google searches and for sea proximity an online map ruler tool was used.

```
data = {
    'NAME': ['Kurunegala Proper', 'Colombo Proper', 'Jaffna Proper', 'Kandy Proper', 'Deniyaya, Matara', 'Bibile, Mona
    'Sea Proximity (approximate-km)': [63, 0, 0, 89, 44, 52, 99],
    'Sea Proximity Categorised': ['Inland', 'Coastal', 'Coastal', 'Hill Country', 'Inland', 'Inland', 'Hill Country'],
    'Climate Zone': ['Intermediate Zone', 'Wet Zone', 'Dry Zone', 'Wet Zone', 'Wet Zone', 'Dry Zone', 'Intermediate Zo
}

sea_proximity = pd.DataFrame(data)

sea_proximity
```

	NAME	Sea Proximity (approximate-km)	Sea Proximity Categorised	Climate Zone
0	Kurunegala Proper	63	Inland	Intermediate Zone
1	Colombo Proper	0	Coastal	Wet Zone
2	Jaffna Proper	0	Coastal	Dry Zone
3	Kandy Proper	89	Hill Country	Wet Zone
4	Deniyaya, Matara	44	Inland	Wet Zone
5	Bibile, Monaragala	52	Inland	Dry Zone
6	Nuwara Eliya Proper	99	Hill Country	Intermediate Zone

After above steps each datasets were merged to get the final dataset.

```
merged_df = pd.merge(allcombined, sea_proximity, how='inner', left_on='NAME', right_on='NAME')
merged_df
```

	NAME	LATITUDE	LONGITUDE	ELEVATION	DATE	PRCP	TAVG	TMAX	TMIN	Sea Proximity (approximate-km)	Sea Proximity Categorised	Climate Zone
0	Bibile, Monaragala	6.8719	81.3489	151.0	2019-01-01	0.22	21.79	26.15	18.60	52	Inland	Dry Zone
1	Bibile, Monaragala	6.8719	81.3489	151.0	2019-01-02	0.03	21.09	25.80	17.52	52	Inland	Dry Zone
2	Bibile, Monaragala	6.8719	81.3489	151.0	2019-01-03	0.03	21.03	26.32	17.05	52	Inland	Dry Zone
3	Bibile, Monaragala	6.8719	81.3489	151.0	2019-01-04	0.02	20.88	26.46	16.77	52	Inland	Dry Zone
4	Bibile, Monaragala	6.8719	81.3489	151.0	2019-01-05	0.14	22.12	27.42	17.82	52	Inland	Dry Zone
...	...	...	...	...	...	...	...	...	...	...	...	...
12777	Nuwara Eliya Proper	6.9670	80.7670	1880.0	2023-12-27	0.35	61.50	64.00	59.00	99	Hill Country	Intermediate Zone
12778	Nuwara Eliya Proper	6.9670	80.7670	1880.0	2023-12-28	2.44	61.00	62.00	60.00	99	Hill Country	Intermediate Zone
12779	Nuwara Eliya Proper	6.9670	80.7670	1880.0	2023-12-29	0.64	60.00	61.00	59.00	99	Hill Country	Intermediate Zone
12780	Nuwara Eliya Proper	6.9670	80.7670	1880.0	2023-12-30	0.21	61.00	65.00	57.00	99	Hill Country	Intermediate Zone
12781	Nuwara Eliya Proper	6.9670	80.7670	1880.0	2023-12-31	0.28	60.00	63.00	57.00	99	Hill Country	Intermediate Zone

12782 rows × 12 columns

2.3 City wise data on effects of anthropogenic data

The original intent was to collect data specifically on anthropogenic activities and their potential impact on formaldehyde (HCHO) emissions in the selected cities, such data proved challenging to obtain directly.

In response to this limitation, an alternative approach was adopted. Atmospheric data on other relevant factors that could be influenced by anthropogenic activities, namely carbon monoxide (CO), nitrogen dioxide (NO2), and ozone (O3) levels, were collected using the "emissionsapi" , which provides access to atmospheric data collected by the European Space Agency's Sentinel-5P satellite. The Sentinel-5P satellite is equipped to monitor various atmospheric factors, including carbon monoxide (CO), nitrogen dioxide (NO2), and ozone (O3) levels. Through this API, users can query the data for specific locations and time periods, allowing for the retrieval of atmospheric data for cities worldwide, including those in Sri Lanka.

The dataset comprises atmospheric data collected from various cities in Sri Lanka, including Colombo, Kurunegala, Nuwara Eliya, Bibile (Monaragala district), Deniyaya (Matara district), Jaffna Proper, and Kandy Proper. The atmospheric factors analyzed include carbon monoxide (CO), nitrogen dioxide (NO2), and ozone (O3) levels.

For each city, the dataset includes the following information:

1. Date: The date of data collection.
2. Latitude: The geographic latitude of the city.
3. Longitude: The geographic longitude of the city.
4. City: The name of the city.
5. CO average: The average concentration of carbon monoxide measured in the atmosphere.
6. NO2 average: The average concentration of nitrogen dioxide measured in the atmosphere.
7. O3 average: The average concentration of ozone measured in the atmosphere.

The dataset has been preprocessed for each city, likely involving cleaning, filtering, and averaging the atmospheric data to ensure accuracy and consistency. After preprocessing, the datasets from different cities have been connected to create a unified dataset for analysis.

These atmospheric factors are commonly associated with industrial processes, vehicular emissions, and other human activities, thus serving as indirect indicators of anthropogenic influence on air quality.

Data Retrieval through API process

```
# API endpoint URL
url1 = "https://api.v2.emissions-api.org/api/v2/carbonmonoxide/statistics.json"
url2 = "https://api.v2.emissions-api.org/api/v2/nitrogendioxide/statistics.json"
url3= "https://api.v2.emissions-api.org/api/v2/ozone/statistics.json"



# Define query parameters
params = {
    "point": "79.8670,6.9000",
    "interval": "day",
    "begin": "2019-01-01",
    "end": "2023-12-31",
}

# Send GET request with parameters
response1 = requests.get(url1, params=params)
response2 = requests.get(url2, params=params)
response3 = requests.get(url3, params=params)

data1 = response1.json()
data2 = response2.json()
data3 = response3.json()

all_datasets=[]
for [i,j] in [[data1,'carbonmonoxide'],[data2,'nitrogendioxide'],[data3,'ozone']]:
    extracted_data=[]
    for entry in i:
        extracted_data.append({
            "DATE": entry["time"]["interval_start"],
            f"{j}_average": entry["value"]["average"],
        })
    df = pd.DataFrame(extracted_data)
    all_datasets.append(handle_dates(df))
```

```
def handle_dates(df):

    df['DATE'] = pd.to_datetime(df['DATE'].str[:10], format='%Y-%m-%d')
    df=df.sort_values(by=['DATE'])

    # Step 1: Check Time Range
    min_date = '2019-01-01'
    max_date = '2023-12-31'

    # Step 2: Generate Time Range
    complete_time_range = pd.date_range(start=min_date, end=max_date, freq='D')

    # Step 3: Identify Missing Values
    missing_values = complete_time_range[~complete_time_range.isin(df['DATE'])]

    if len(missing_values) == 0:
        pass
    else:
        # Step 4: Create DataFrame for missing dates
        missing_df = pd.DataFrame({'DATE': missing_values})

        # Step 5: Concatenate missing dates DataFrame with existing DataFrame
        df = pd.concat([df, missing_df], ignore_index=True)
        df['DATE']=pd.to_datetime(df['DATE'])

        # Sort the DataFrame by 'DATE' column
        df=df.sort_values(by=['DATE'])
        df = df.reset_index(drop=True)

        print('Date column was succefully handled')
        return df
```

Final Dataset

	DATE	carbonmonoxide_average	nitrogendioxide_average	ozone_average	region
0	2019-01-01	0.035565	0.000007	0.113006	Bibile, Monaragala
1	2019-01-02	0.032648	0.000011	0.113006	Bibile, Monaragala
2	2019-01-03	0.032855	0.000010	0.110589	Bibile, Monaragala
3	2019-01-04	0.033473	0.000009	0.113011	Bibile, Monaragala
4	2019-01-05	0.034100	0.000009	0.112202	Bibile, Monaragala
...	...	...	...	...	...
12777	2023-12-27	0.032671	0.000016	0.109249	Nuwara Eliya Proper
12778	2023-12-28	0.031646	0.000016	0.110141	Nuwara Eliya Proper
12779	2023-12-29	0.030793	0.000016	0.110725	Nuwara Eliya Proper
12780	2023-12-30	0.031598	0.000017	0.109886	Nuwara Eliya Proper
12781	2023-12-31	0.031346	0.000015	0.110000	Nuwara Eliya Proper

12782 rows × 5 columns

```
combined_df['region'].value_counts()
```

```
region
Bibile, Monaragala     1826
Colombo Proper         1826
Deniyaya, Matara       1826
Jaffna Proper          1826
Kandy Proper           1826
Kurunegala Proper      1826
Nuwara Eliya Proper    1826
Name: count, dtype: int64
```

2.4 Covid and demographic data

Unfortunately, despite efforts to gather city-specific COVID-19 data, such information was not available.The dataset was sourced using an API provided by NinjasAPI to obtain COVID-19 data. The dataset consists of aggregated COVID-19 data for the entire island rather than city-specific data.

The dataset includes the following columns:

1. Date: The date of data collection.
2. Daily new COVID-19 cases: The number of new COVID-19 cases reported on that particular day.
3. Total COVID-19 cases up to that day: The cumulative total of COVID-19 cases reported up to the specified date.

While city-wise COVID-19 data would have provided a more detailed analysis, the island-wide dataset still offers valuable insights into the overall COVID-19 situation in Sri Lanka.

In [2]:
```
import requests

country = 'Sri Lanka'
region='galle'
api_url = 'https://api.api-ninjas.com/v1/covid19?country={}'.format(country,region)
response = requests.get(api_url, headers={'X-Api-Key': 'ChFX2bim+4wKPTz29XhdDA==AnPaGTh554GAAJi8'})
if response.status_code == requests.codes.ok:
    print(response.text)
    print(type(response.text))
else:
    print("Error:", response.status_code, response.text)
```

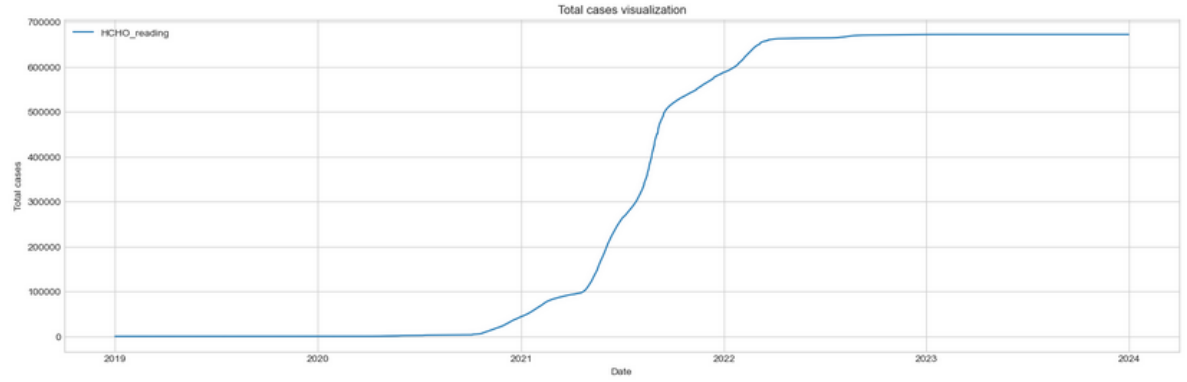

```
0}, "2023-01-20": {"total": 671976, "new": 6}, "2023-01-21": {"total": 671976, "new": 0}, "2023-01-22": {"total": 671976,
"new": 0}, "2023-01-23": {"total": 671981, "new": 5}, "2023-01-24": {"total": 671984, "new": 3}, "2023-01-25": {"total": 6
71984, "new": 0}, "2023-01-26": {"total": 671986, "new": 2}, "2023-01-27": {"total": 671986, "new": 0}, "2023-01-28": {"to
tal": 671987, "new": 1}, "2023-01-29": {"total": 671987, "new": 0}, "2023-01-30": {"total": 671988, "new": 1}, "2023-01-3
1": {"total": 671989, "new": 1}, "2023-02-01": {"total": 671991, "new": 2}, "2023-02-02": {"total": 671994, "new": 3}, "20
23-02-03": {"total": 671995, "new": 1}, "2023-02-04": {"total": 672000, "new": 5}, "2023-02-05": {"total": 672000, "new":
0}, "2023-02-06": {"total": 672002, "new": 2}, "2023-02-07": {"total": 672003, "new": 1}, "2023-02-08": {"total": 672004,
"new": 1}, "2023-02-09": {"total": 672005, "new": 1}, "2023-02-10": {"total": 672005, "new": 0}, "2023-02-11": {"total": 6
72007, "new": 2}, "2023-02-12": {"total": 672009, "new": 2}, "2023-02-13": {"total": 672012, "new": 3}, "2023-02-14": {"to
tal": 672015, "new": 3}, "2023-02-15": {"total": 672016, "new": 1}, "2023-02-16": {"total": 672017, "new": 1}, "2023-02-1
7": {"total": 672018, "new": 1}, "2023-02-18": {"total": 672018, "new": 0}, "2023-02-19": {"total": 672021, "new": 3}, "20
23-02-20": {"total": 672022, "new": 1}, "2023-02-21": {"total": 672023, "new": 1}, "2023-02-22": {"total": 672024, "new":
1}, "2023-02-23": {"total": 672025, "new": 1}, "2023-02-24": {"total": 672026, "new": 1}, "2023-02-25": {"total": 672026,
"new": 0}, "2023-02-26": {"total": 672027, "new": 1}, "2023-02-27": {"total": 672029, "new": 2}, "2023-02-28": {"total": 6
72030, "new": 1}, "2023-03-01": {"total": 672031, "new": 1}, "2023-03-02": {"total": 672031, "new": 0}, "2023-03-03": {"to
tal": 672032, "new": 1}, "2023-03-04": {"total": 672032, "new": 0}, "2023-03-05": {"total": 672034, "new": 2}, "2023-03-0
6": {"total": 672034, "new": 0}, "2023-03-07": {"total": 672036, "new": 2}, "2023-03-08": {"total": 672037, "new": 1}, "20
23-03-09": {"total": 672039, "new": 2}}}]
<class 'str'>
```


In [3]:
```
import json

# API response content
response_content = response.text
# Convert response content to JSON
data = json.loads(response_content)

# Create DataFrame directly
df = pd.DataFrame(data[0]['cases']).T.reset_index()
df = df.rename(columns={'index': 'date'})

# Add country and region columns
df['country'] = data[0]['country']
df['region'] = data[0]['region']

# Reorder columns
df = df[['country', 'region', 'date', 'total', 'new']]

# Print DataFrame
df.tail(10)
```

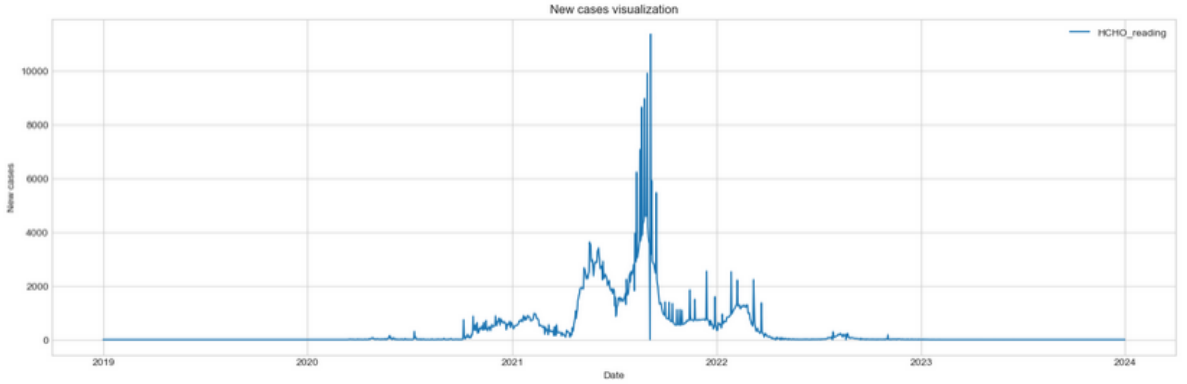
Out[3]:

	country	region	date	total	new
1133	Sri Lanka		2023-02-28	672030	1
1134	Sri Lanka		2023-03-01	672031	1
1135	Sri Lanka		2023-03-02	672031	0
1136	Sri Lanka		2023-03-03	672032	1
1137	Sri Lanka		2023-03-04	672032	0
1138	Sri Lanka		2023-03-05	672034	2
1139	Sri Lanka		2023-03-06	672034	0
1140	Sri Lanka		2023-03-07	672036	2
1141	Sri Lanka		2023-03-08	672037	1
1142	Sri Lanka		2023-03-09	672039	2

```
# Create the plot
plt.figure(figsize=(20, 6))
plt.plot( df['date'], df["total"], label="HCHO_reading")
plt.title("Total cases visualization")
plt.xlabel("Date")
plt.ylabel("Total cases")
plt.legend()
plt.show()
```




```
# Create the plot
plt.figure(figsize=(20, 6))
plt.plot( df['date'], df["new"], label="HCHO_reading")
plt.title("New cases visualization")
plt.xlabel("Date")
plt.ylabel("New cases")
plt.legend()
plt.show()
```



Population data was acquired from Registrar General's Department. Data tables were in pdf format, so they were extracted using MS Excel did the necessary filtering and preprocessing using pandas. Population density was acquired by dividing existing population data by districts areas (gained using google search).

	District	Area(km^2)	Year	Population	Population_density
0	Colombo Proper	698.7	2019	2448000	3503.649635
1	Colombo Proper	698.7	2020	2455000	3513.668241
2	Colombo Proper	698.7	2021	2480000	3549.448977
3	Colombo Proper	698.7	2022	2478000	3546.586518
4	Colombo Proper	698.7	2023	2460000	3520.824388
5	Kandy Proper	1939.5	2019	1476000	761.020882
6	Kandy Proper	1939.5	2020	1483000	764.630059
7	Kandy Proper	1939.5	2021	1501000	773.910802
8	Kandy Proper	1939.5	2022	1499000	772.879608
9	Kandy Proper	1939.5	2023	1482000	764.114462
10	Nuwara Eliya Proper	1741.2	2019	768000	441.075121
11	Nuwara Eliya Proper	1741.2	2020	773000	443.946703
12	Nuwara Eliya Proper	1741.2	2021	780000	447.966919
13	Nuwara Eliya Proper	1741.2	2022	783000	449.689869
14	Nuwara Eliya Proper	1741.2	2023	781000	448.541236
15	Deniyaya, Matara	1282.5	2019	863000	672.904483
16	Deniyaya, Matara	1282.5	2020	866000	675.243665
17	Deniyaya, Matara	1282.5	2021	873000	680.701754
18	Deniyaya, Matara	1282.5	2022	874000	681.481481
19	Deniyaya, Matara	1282.5	2023	869000	677.582846
20	Jaffna Proper	1025.3	2019	617000	601.775090
21	Jaffna Proper	1025.3	2020	621000	605.676387
22	Jaffna Proper	1025.3	2021	626000	610.553009
23	Jaffna Proper	1025.3	2022	629000	613.478982
24	Jaffna Proper	1025.3	2023	628000	612.503657
25	Kurunegala Proper	4815.8	2019	1719000	356.950039
26	Kurunegala Proper	4815.8	2020	1726000	358.403588
27	Kurunegala Proper	4815.8	2021	1743000	361.933635
28	Kurunegala Proper	4815.8	2022	1742000	361.725985
29	Kurunegala Proper	4815.8	2023	1727000	358.611238
30	Bibile, Monaragala	5638.7	2019	496000	87.963538
31	Bibile, Monaragala	5638.7	2020	501000	88.850267
32	Bibile, Monaragala	5638.7	2021	505000	89.559650
33	Bibile, Monaragala	5638.7	2022	509000	90.269034
34	Bibile, Monaragala	5638.7	2023	509000	90.269034

After preparing each dataset each of them were merged to get the final dataset.

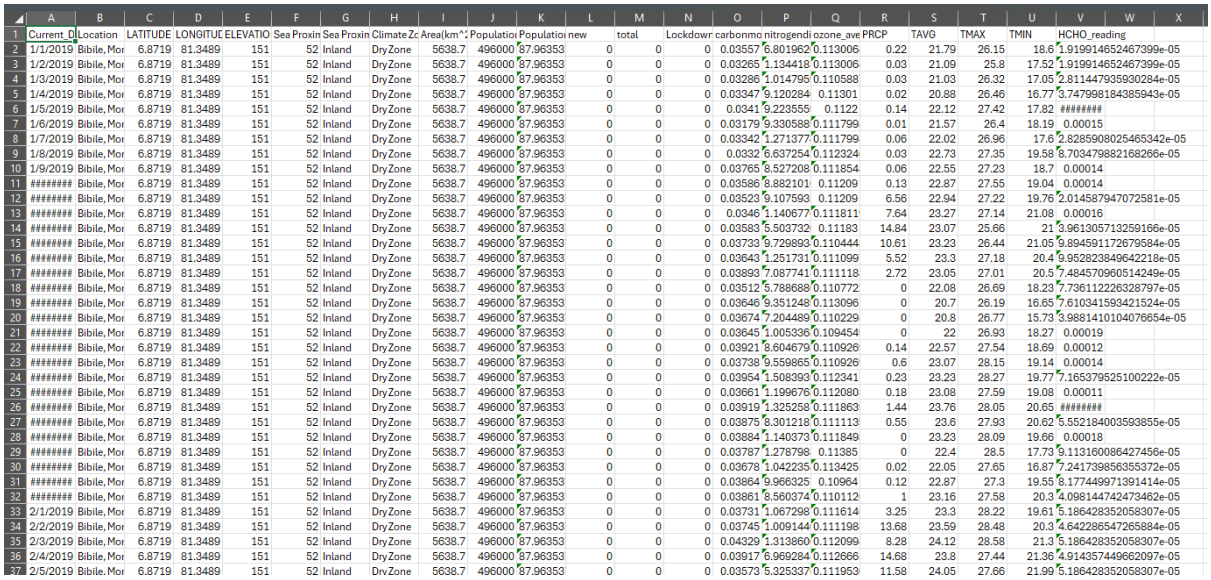
	District	Area(km^2)	Population	Population_density	date	total	new	Lockdown
0	Bibile, Monaragala	5638.7	496000	87.963538	2019-01-01	0.0	0.0	0
1	Bibile, Monaragala	5638.7	496000	87.963538	2019-01-02	0.0	0.0	0
2	Bibile, Monaragala	5638.7	496000	87.963538	2019-01-03	0.0	0.0	0
3	Bibile, Monaragala	5638.7	496000	87.963538	2019-01-04	0.0	0.0	0
4	Bibile, Monaragala	5638.7	496000	87.963538	2019-01-05	0.0	0.0	0
...	...	...	...	...	...	...	...	...
12777	Nuwara Eliya Proper	1741.2	781000	448.541236	2023-12-27	672039.0	0.0	0
12778	Nuwara Eliya Proper	1741.2	781000	448.541236	2023-12-28	672039.0	0.0	0
12779	Nuwara Eliya Proper	1741.2	781000	448.541236	2023-12-29	672039.0	0.0	0
12780	Nuwara Eliya Proper	1741.2	781000	448.541236	2023-12-30	672039.0	0.0	0
12781	Nuwara Eliya Proper	1741.2	781000	448.541236	2023-12-31	672039.0	0.0	0

12782 rows × 8 columns

2.5 Final Data set

After collecting data from multiple sources covering the effects on HCHO readings in Colombo, Kandy, Nuwara Eliya, Matara, Monaragala, Jaffna, and Kurunegala due to weather, anthropogenic activities, COVID-19, geographic, and demographic factors between January 1, 2019, and December 31, 2023, all datasets were merged.

```
In [38]: merged_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12782 entries, 0 to 12781
Data columns (total 22 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Current_Date                    12782 non-null  datetime64[ns]
 1   Location                        12782 non-null  object
 2   LATITUDE                        12782 non-null  float64
 3   LONGITUDE                       12782 non-null  float64
 4   ELEVATION                       12782 non-null  float64
 5   Sea Proximity (approximate-km)  12782 non-null  int64
 6   Sea Proximity Categorised       12782 non-null  object
 7   Climate Zone                    12782 non-null  object
 8   Area(km^2)                      12782 non-null  float64
 9   Population                      12782 non-null  int64
 10  Population_density              12782 non-null  float64
 11  new                             12782 non-null  float64
 12  total                           12782 non-null  float64
 13  Lockdown                        12782 non-null  int64
 14  carbonmonoxide_average          12782 non-null  float64
 15  nitrogendioxide_average         12782 non-null  float64
 16  ozone_average                   12782 non-null  float64
 17  PRCP                            12782 non-null  float64
 18  TAVG                            12782 non-null  float64
 19  TMAX                            12782 non-null  float64
 20  TMIN                            12782 non-null  float64
 21  HCHO_reading                    12782 non-null  float64
dtypes: datetime64[ns](1), float64(15), int64(3), object(3)
memory usage: 2.1+ MB

In [39]: merged_df.columns

Out[39]: Index(['Current_Date', 'Location', 'LATITUDE', 'LONGITUDE', 'ELEVATION',
       'Sea Proximity (approximate-km)', 'Sea Proximity Categorised',
       'Climate Zone', 'Area(km^2)', 'Population', 'Population_density', 'new',
       'total', 'Lockdown', 'carbonmonoxide_average',
       'nitrogendioxide_average', 'ozone_average', 'PRCP', 'TAVG', 'TMAX',
       'TMIN', 'HCHO_reading'],
      dtype='object')

In [40]: merged_df['Location'].value_counts()

Out[40]: Location
Bibile, Monaragala     1826
Colombo Proper         1826
Deniyaya, Matara       1826
Jaffna Proper          1826
Kandy Proper           1826
Kurunegala Proper      1826
Nuwara Eliya Proper    1826
Name: count, dtype: int64
```



Current_D	Location	LATITUDE	LONGITUD	ELEVATIO	Sea Proxin	Sea Proxin	Climate Zc	Area(km^2	Population	Populatio	new	total	Lockdowr	carbonmo	nitrogendi	ozone_ave	PRCP	TAVG	TMAX	TMIN	HCHO_reading
1/1/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03557	6.801962	0.113006	0.22	21.79	26.15	18.6	1.919914652467399e-05
1/2/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03265	1.134418	0.113006	0.03	21.09	25.8	17.52	1.919914652467399e-05
1/3/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03286	1.014795	0.110588	0.03	21.03	26.32	17.05	2.8114479359302884e-05
1/4/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03347	9.120284	0.11301	0.02	20.88	26.46	16.77	3.747998184385943e-05
1/5/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.0341	9.223555	0.1122	0.14	22.12	27.42	17.82	########
1/6/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03179	9.330588	0.111799	0.01	21.57	26.4	18.19	0.00015
1/7/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03342	1.271377	0.111799	0.06	22.02	26.96	17.6	2.8285908025465342e-05
1/8/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.0332	6.637254	0.112324	0.03	22.73	27.35	19.58	8.703479882168266e-05
1/9/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03765	8.527208	0.111854	0.06	22.55	27.23	18.7	0.00014
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03586	8.882101	0.11209	0.13	22.87	27.55	19.04	0.00014
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03523	9.107593	0.11209	6.56	22.94	27.22	19.76	2.0145879470725815e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.0346	1.140677	0.111811	7.64	23.27	27.14	21.08	0.00016
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03583	5.503732	0.11183	14.84	23.07	25.66	21	3.961305713259166e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03733	9.729893	0.110444	10.61	23.23	26.44	21.05	9.894591172679584e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03643	1.251731	0.111099	5.52	23.3	27.18	20.4	9.952823849642218e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03893	7.087741	0.111118	2.72	23.05	27.01	20.5	7.484570960514249e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03512	5.788688	0.110772	0	22.08	26.69	18.23	7.736112226328797e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03646	9.351248	0.113096	0	20.7	26.19	16.65	7.610341593421524e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03674	7.204489	0.110229	0	20.8	26.77	15.73	3.988141010476654e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03645	1.005336	0.109454	0	22	26.93	18.27	0.00019
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03921	8.604679	0.110926	0.14	22.57	27.54	18.69	0.00012
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03738	9.559865	0.110926	0.6	23.07	28.15	19.14	0.00014
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03954	1.508393	0.112341	0.23	23.23	28.27	19.77	7.165379525100222e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03661	1.199676	0.112080	0.18	23.08	27.59	19.08	0.00011
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03919	1.325258	0.111863	1.44	23.76	28.05	20.65	########
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03875	8.301218	0.111113	0.55	23.6	27.93	20.62	5.552184003593855e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03884	1.140373	0.111849	0	23.23	28.09	19.66	0.00018
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03787	1.278798	0.11385	0	22.4	28.5	17.73	9.113160086427456e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03678	1.042235	0.113425	0.02	22.05	27.65	16.87	7.241739856355372e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03864	9.966325	0.10964	0.12	22.87	27.3	19.55	8.177449971391414e-05
########	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03861	8.560374	0.110112	1	23.16	27.58	20.3	4.098144742473462e-05
2/1/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03731	1.067296	0.111614	3.25	23.3	28.22	19.61	5.186428352058307e-05
2/2/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03745	1.009144	0.111198	13.68	23.59	28.48	20.3	4.6422865472658884e-05
2/3/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.04329	1.313860	0.112099	8.28	24.12	28.58	21.3	5.186428352058307e-05
2/4/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03917	6.969284	0.112666	14.68	23.8	27.44	21.36	4.9143574496620976e-05
2/5/2019	Bibile, Mor	6.8719	81.3489	151	52	Inland	Dry Zone	5638.7	496000	87.96353	0	0	0	0.03573	5.325337	0.111953	11.58	24.05	27.66	21.99	5.186428352058307e-05

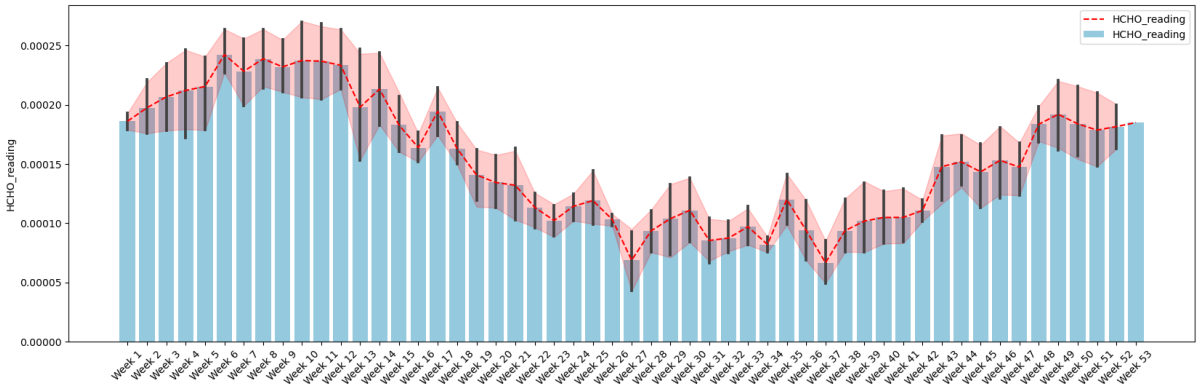
1. **Current_Date**: The date of the data entry or record.
2. **Location**: The geographical location where the data was collected.
3. **LATITUDE**: The latitude coordinate of the location.
4. **LONGITUDE**: The longitude coordinate of the location.
5. **ELEVATION**: The elevation (height above sea level) of the location.
6. **Sea Proximity (approximate-km)**: The distance of the location from the nearest sea or ocean, measured in kilometers.
7. **Sea Proximity Categorised**: Categorization of sea proximity, which may indicate whether the location is coastal, inland, etc.
8. **Climate Zone**: The climate zone or region where the location belongs, based on factors like temperature, precipitation, etc.
9. **Area(km^2)**: The area of the location in square kilometers.
10. **Population**: The population count of the area or location.
11. **Population_density**: The population density, often measured as the number of people per square kilometer.
12. **new**: The number of new COVID-19 cases reported on that particular day.
13. **total**: The cumulative total of COVID-19 cases reported up to the specified date.
14. **Lockdown**: A binary variable indicating whether the location was under lockdown or not.
15. **carbonmonoxide_average**: Average concentration of carbon monoxide in the air, typically measured in parts per million (ppm).

16. **nitrogendioxide_average**: Average concentration of nitrogen dioxide in the air, typically measured in parts per billion (ppb).
17. **ozone_average**: Average concentration of ozone in the air, typically measured in parts per billion (ppb).
18. **PRCP**: Precipitation amount, typically measured in millimeters (mm) or inches.
19. **TAVG**: Average temperature, typically measured in degrees Celsius or Fahrenheit.
20. **TMAX**: Maximum temperature recorded, typically measured in degrees Celsius or Fahrenheit.
21. **TMIN**: Minimum temperature recorded, typically measured in degrees Celsius or Fahrenheit.
22. **HCHO_reading**: Reading of formaldehyde concentration on that particular day., typically measured in parts per billion (ppb).

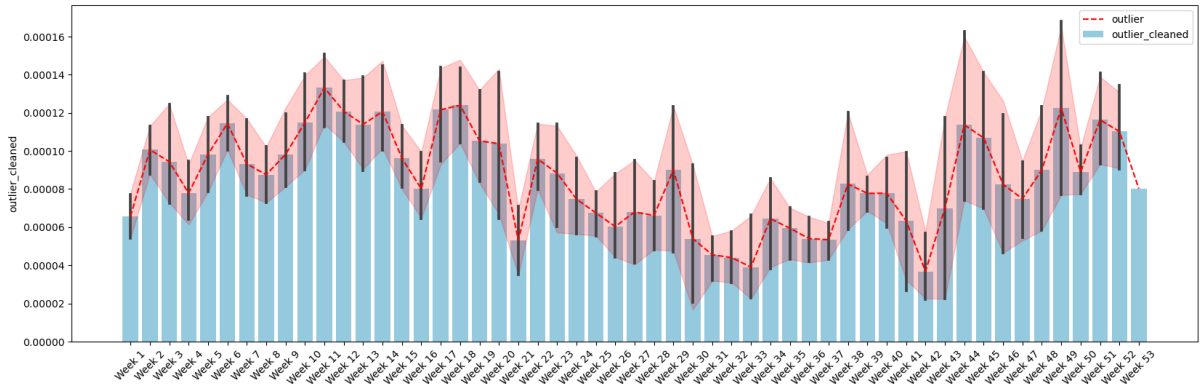
3 Spatial Temporal Analysis

3.1 Weekly Seasonality analysis of each city

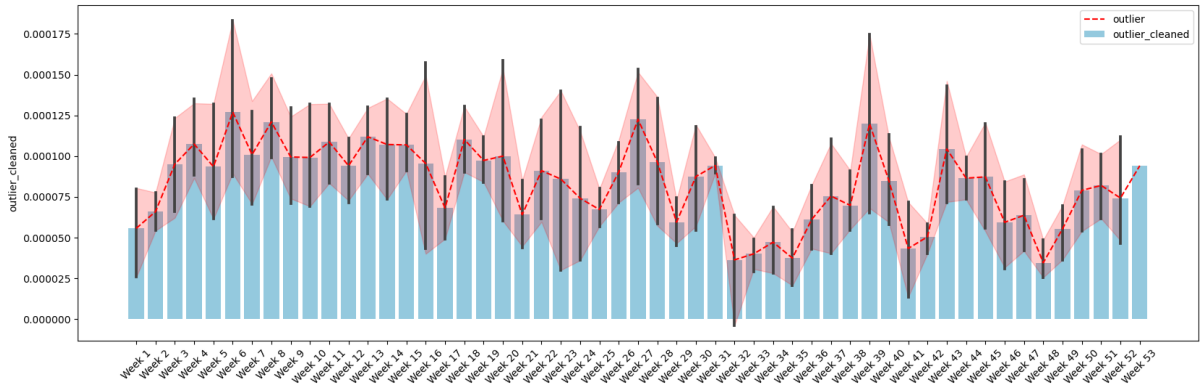
1. Colombo



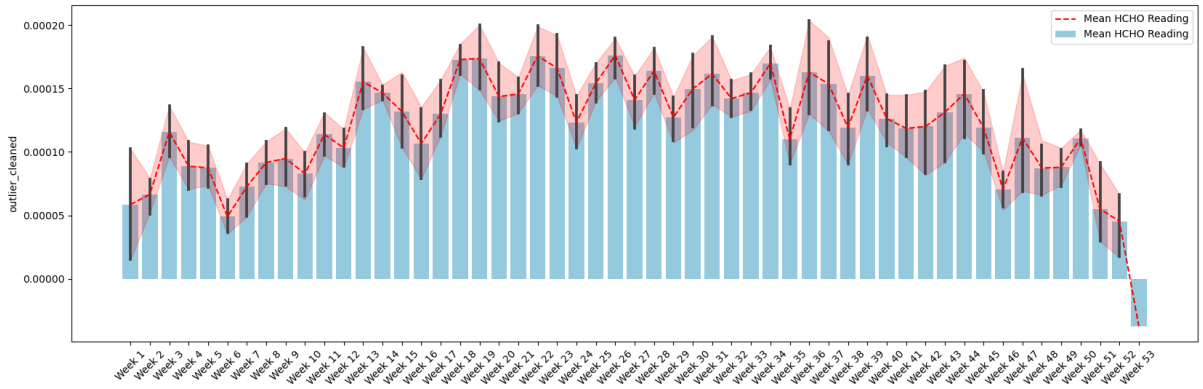
2. Matara Deniyaya



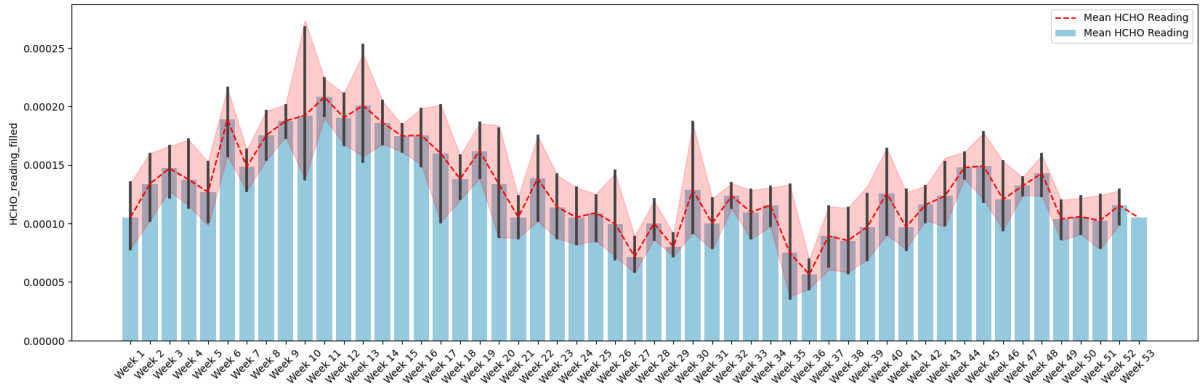
3. Nuwara Eliya



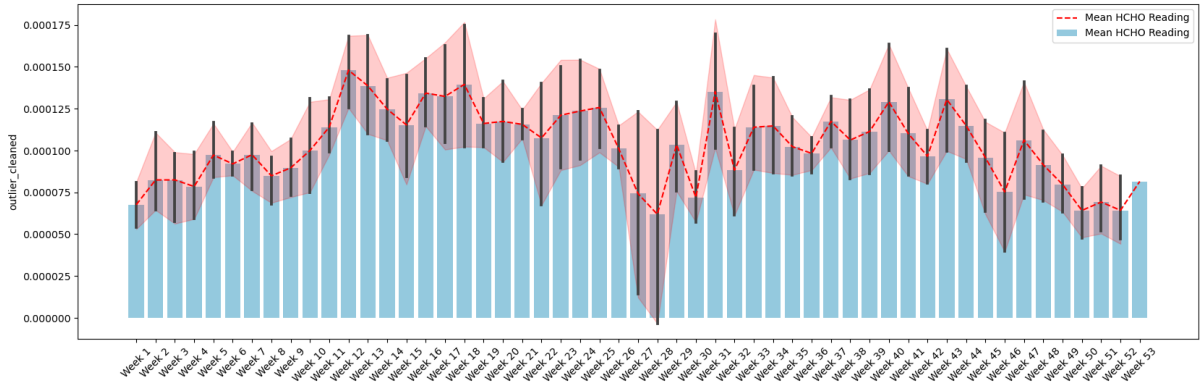
4. Bibile Monaragala



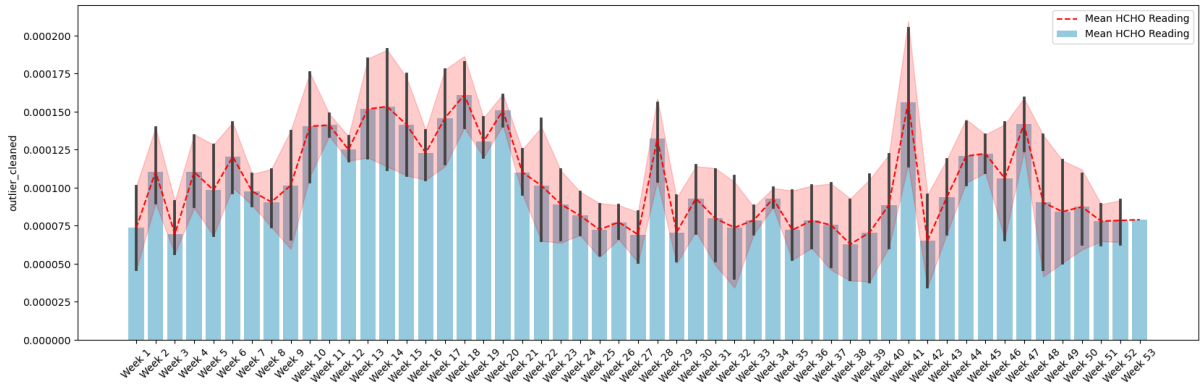
5. Kurunegala



6. Jaffna



7. Kandy



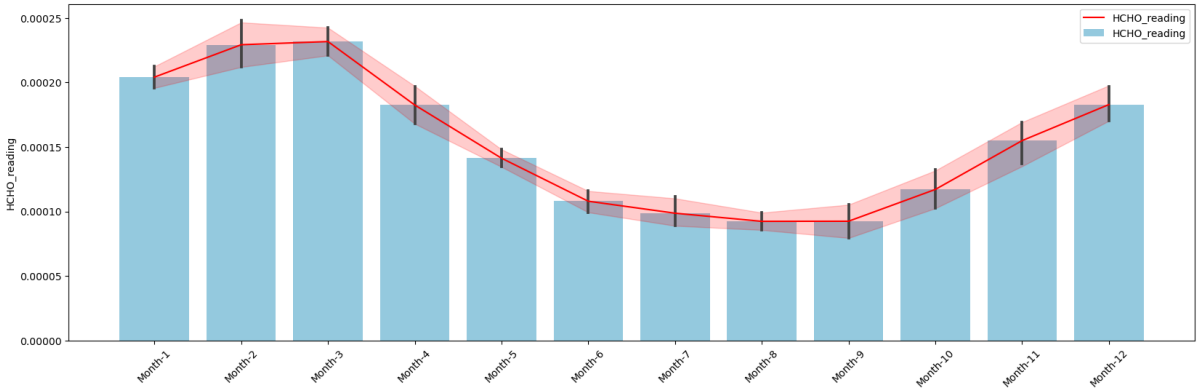
From the provided plot, we can determine the monthly mean formaldehyde (HCHO) readings over a certain period of time. The x-axis represents different months, while the y-axis represents the mean HCHO readings.

The blue bars show the mean HCHO readings for each month, providing a visual representation of the average level of formaldehyde observed during those periods. The red line overlaid on top of the bars represents the trend of mean HCHO readings over time, showing any overall patterns or fluctuations in the data.

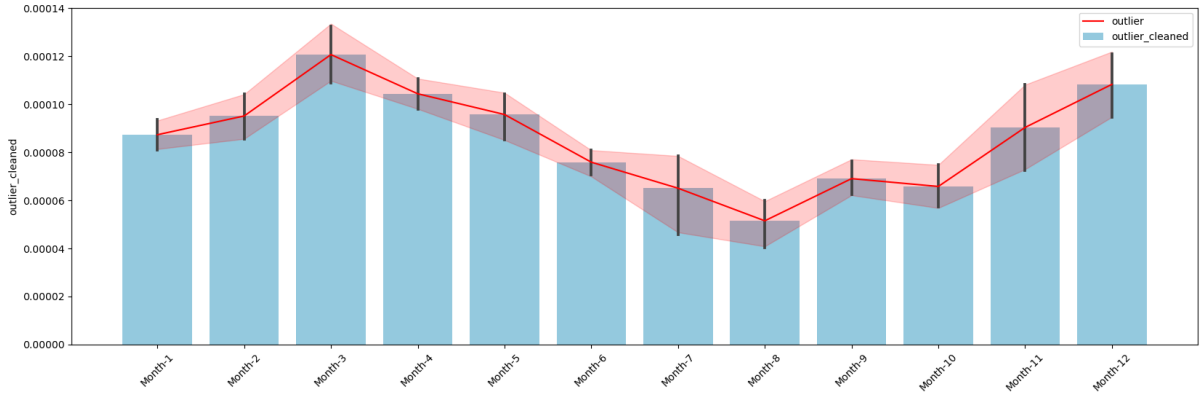
Overall, these plots allows us to visualize the average levels of formaldehyde and observe any trends or seasonal variations in its concentration over the specified time period.

3.2 Monthly Seasonality analysis of each city

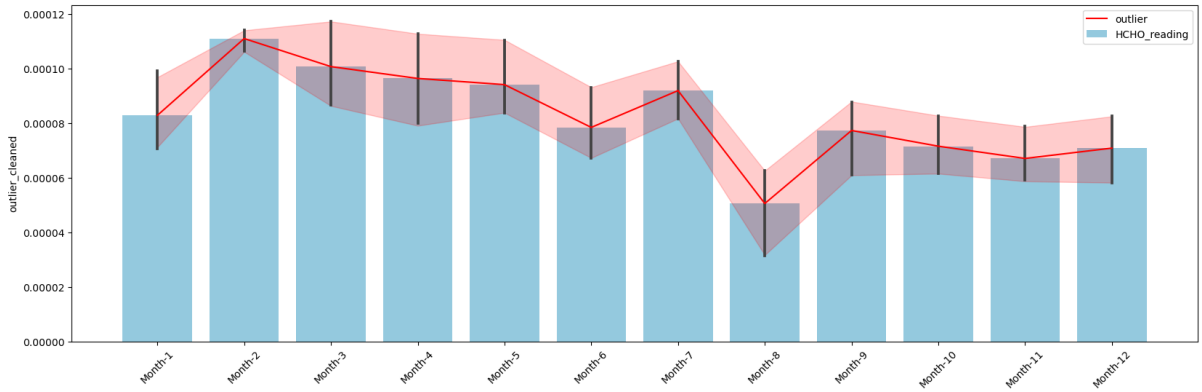
1. Colombo



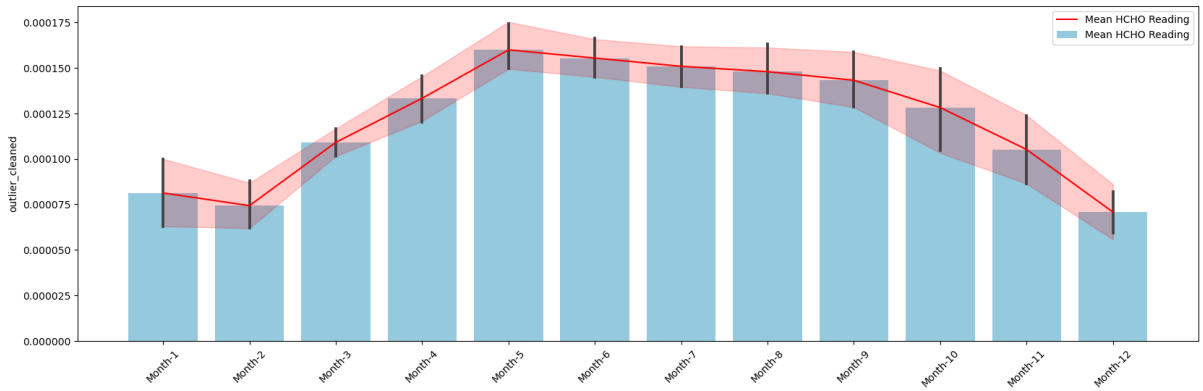
2. Matara Deniyaya



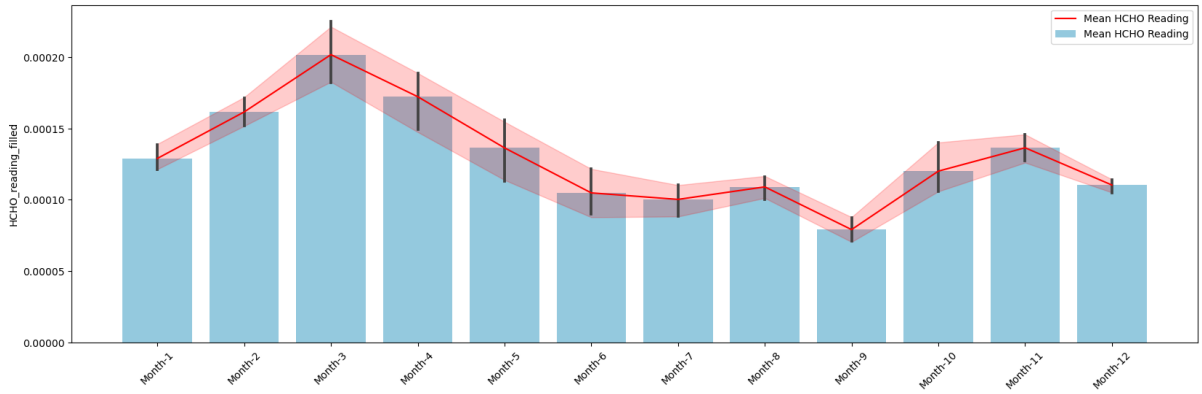
3. Nuwara Eliya



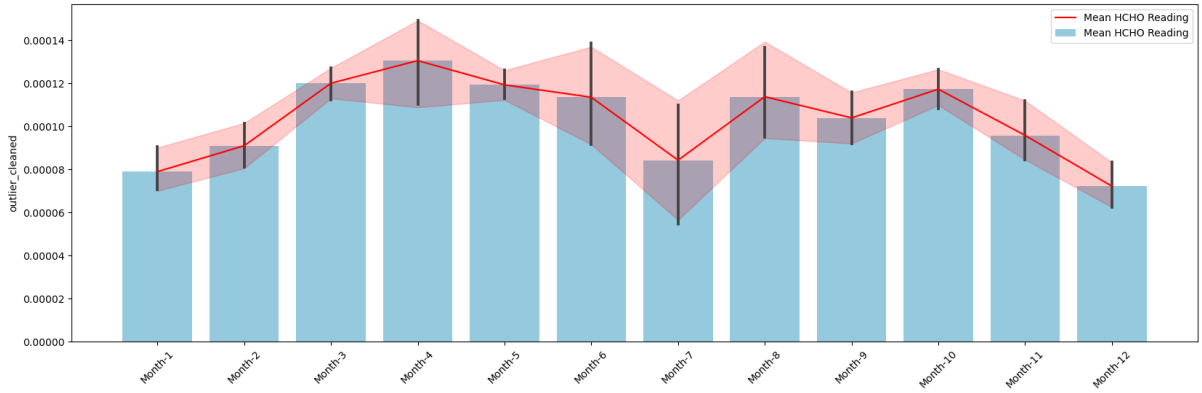
4. Bibile Monaragala



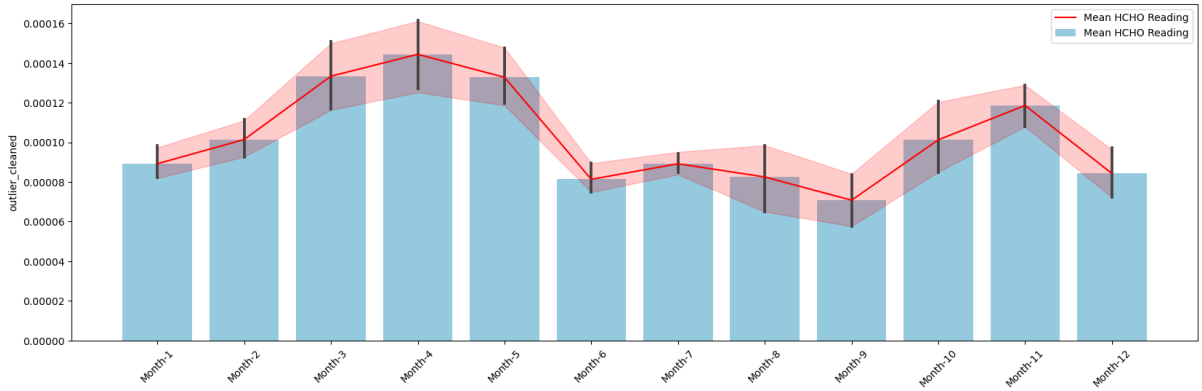
5. Kurunegala



6. Jaffna



7. Kandy



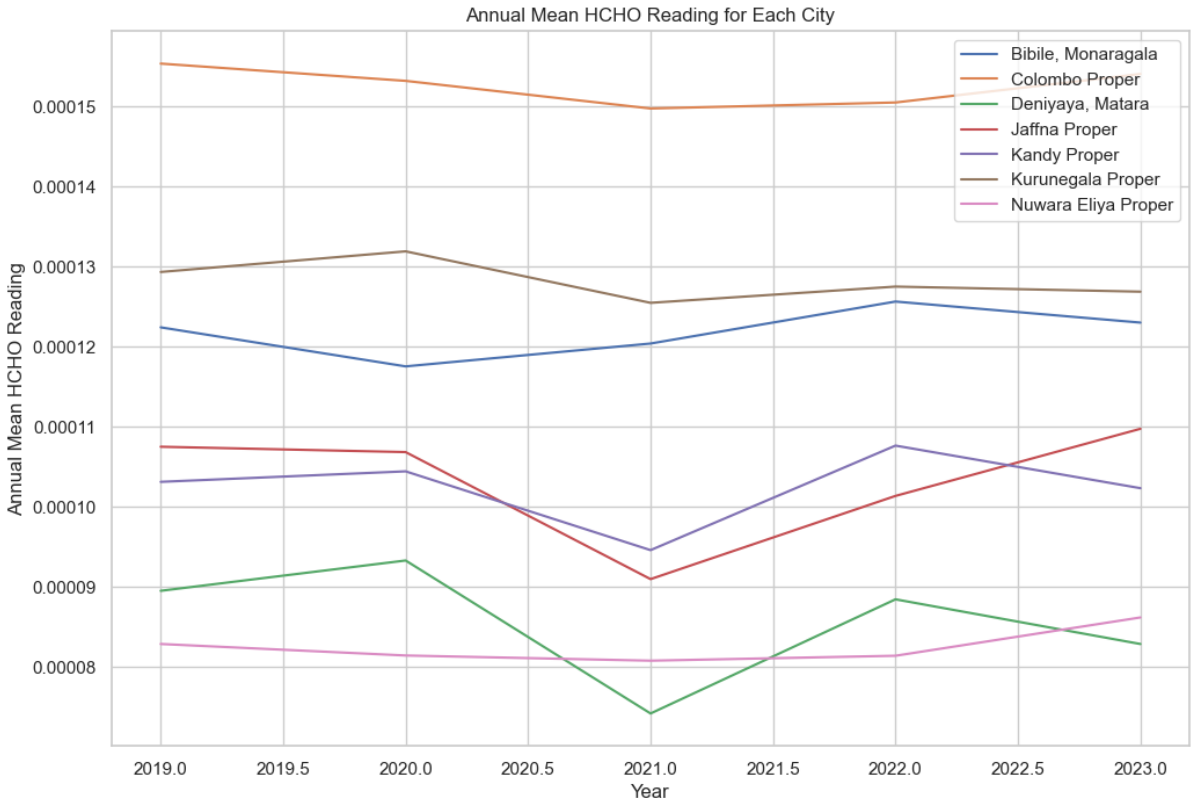
These plots shows the monthly mean formaldehyde (HCHO) readings over a certain period of time. The x-axis represents different months, while the y-axis represents the mean HCHO readings.

The blue bars represent the mean HCHO readings for each month, giving a visual indication of the average level of formaldehyde observed during those periods. The height of each bar indicates the magnitude of the mean reading for the corresponding month.

The red line overlaid on top of the bars represents the trend of mean HCHO readings over time. It connects the mean values for each month, allowing for the visualization of any overall patterns or changes in formaldehyde concentration throughout the specified time frame.

In summary, these plots provides insights into the average levels of formaldehyde and allows for the observation of trends or variations in its concentration over the given time period.

3.3 City wise annual mean formaldehyde (HCHO) readings comparison

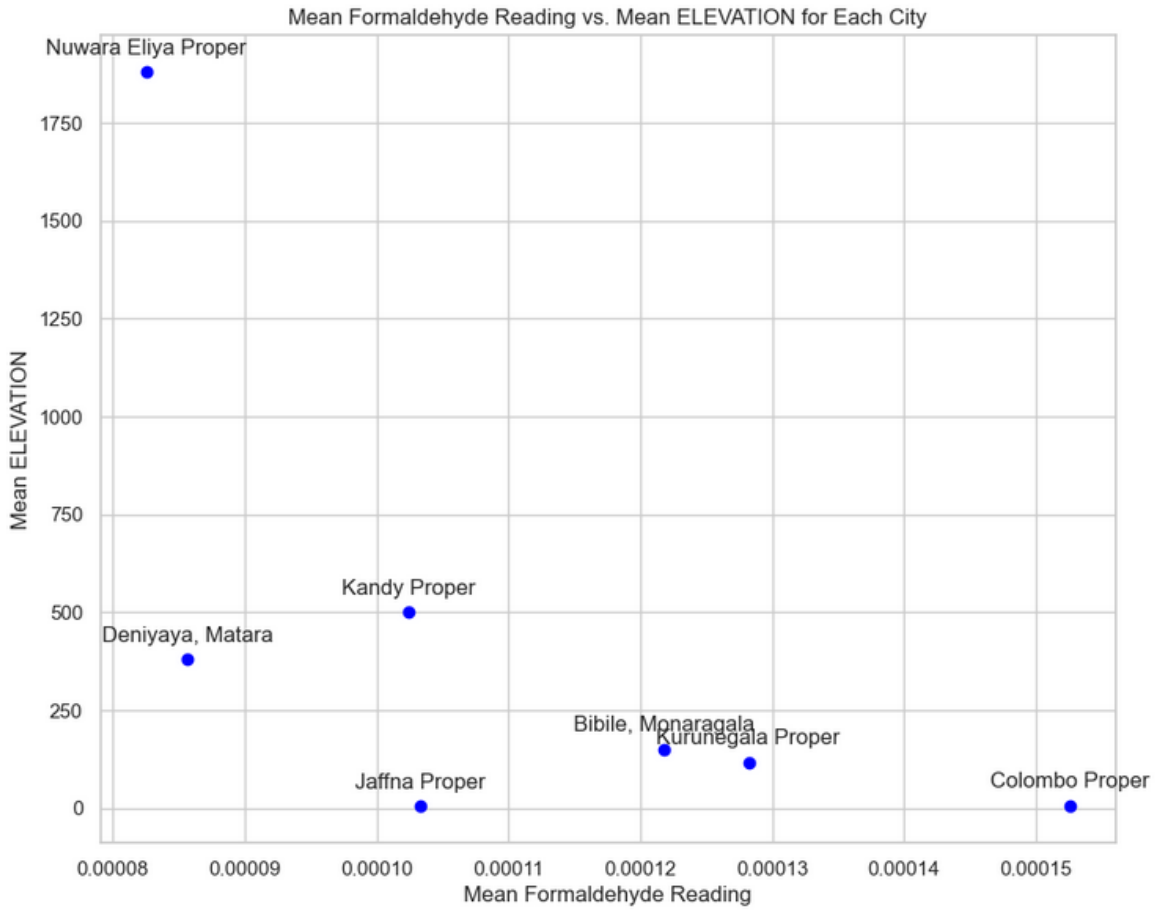


The plot displays annual mean formaldehyde (HCHO) readings for multiple cities over time. Each line represents a city, showing its HCHO levels across years. It allows comparison between cities, identification of trends, anomalies, and long-term patterns in air quality.

3.4 Relationship between the mean formaldehyde (HCHO) readings and various city attributes

1. Elevation

```
Mean HCHO_Reading vs Mean ELEVATION for each city
```

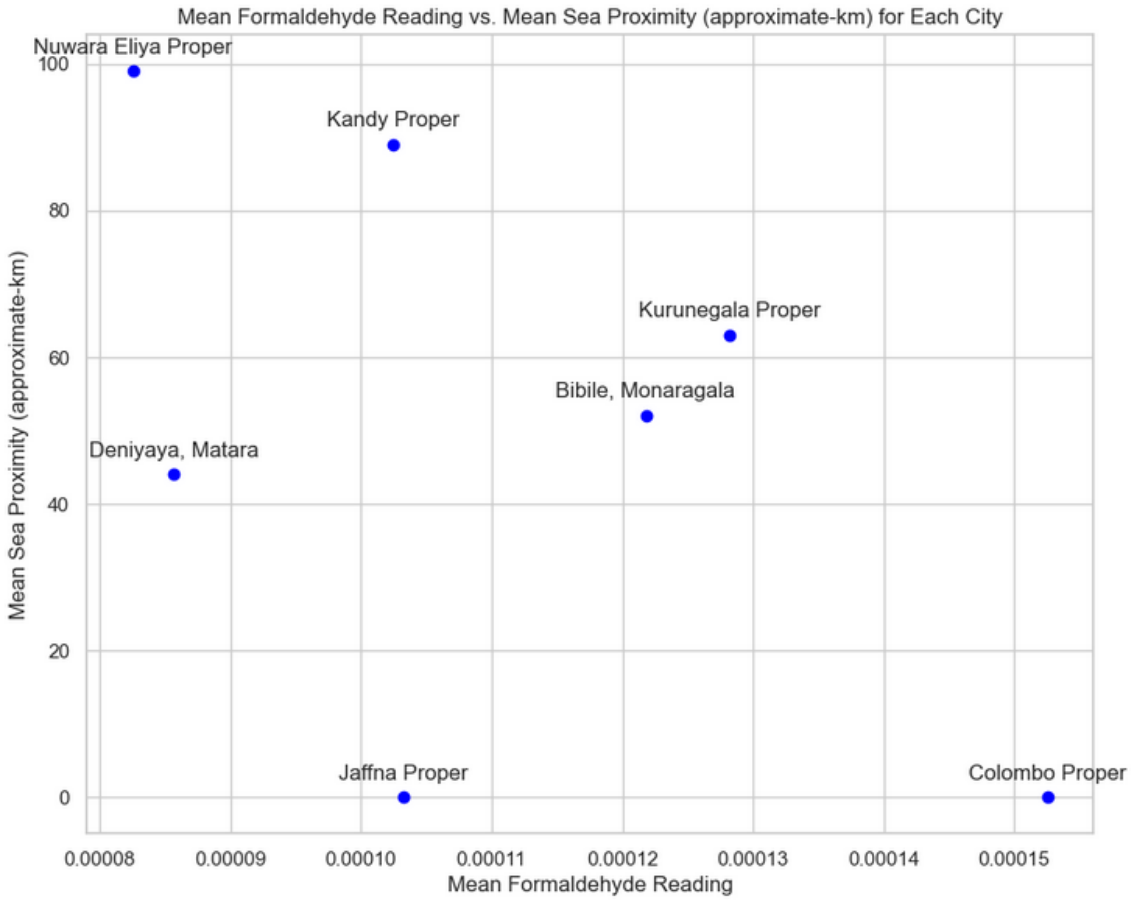



```
Correlation between mean HCHO readings and ELEVATION: -0.6378411172798543
```

- Correlation: The correlation coefficient of approximately -0.64 indicates a moderate to strong negative correlation between mean HCHO readings and elevation. As elevation increases, mean HCHO readings tend to decrease.
- Interpretation: Cities located at higher elevations, such as Nuwara Eliya Proper (1880 meters), exhibit lower mean HCHO readings compared to cities at lower elevations like Colombo Proper (7 meters) and Jaffna Proper (5 meters).

2. Sea Proximity

```
Mean HCHO_Reading vs Mean Sea Proximity (approximate-km) for each city
```

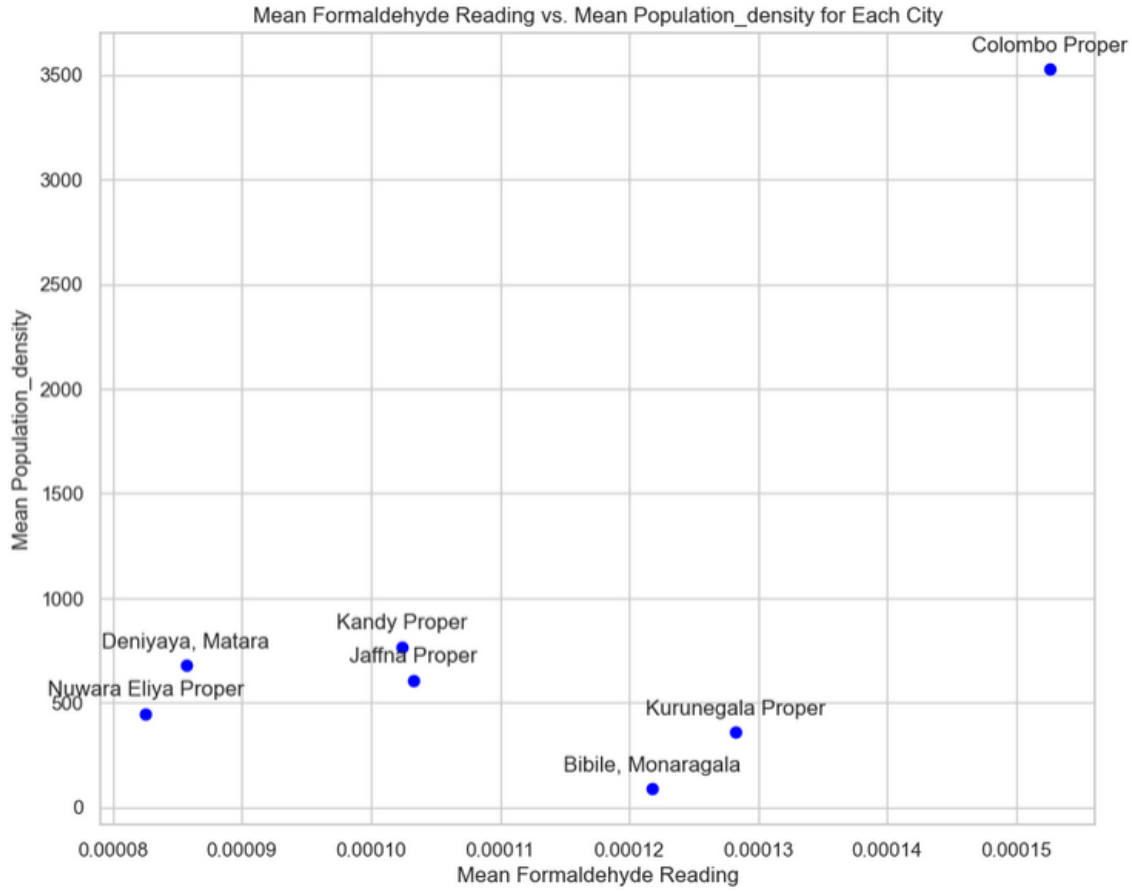



```
Correlation between mean HCHO readings and Sea Proximity (approximate-km): -0.5186167802375364
```

- Correlation: The correlation coefficient of approximately -0.52 suggests a moderate negative correlation between mean HCHO readings and sea proximity. As the distance from the sea increases, mean HCHO readings tend to rise.
- Interpretation: Cities with closer sea proximity, like Colombo Proper (0 km) and Jaffna Proper (0 km), tend to have higher mean HCHO readings compared to cities farther from the sea, such as Nuwara Eliya Proper (99 km).

3. Population Density

```
Mean HCHO_Reading vs Mean Population_density for each city
```




```
Correlation between mean HCHO readings and Population_density: 0.6468497541603153
```

- Correlation: The correlation coefficient of approximately 0.65 indicates a moderate to strong positive correlation between mean HCHO readings and population density. Higher population density is associated with higher mean HCHO readings.
- Interpretation: Cities with higher population densities, such as Colombo Proper (3526.83 people/km²) and Kandy Proper (767.31 people/km²), tend to have higher mean HCHO readings. Conversely, cities with lower population densities like Kurunegala Proper (359.52 people/km²) exhibit lower mean HCHO readings.

Summary-

In summary, the city attributes—elevation, sea proximity, and population density—reveal notable correlations with mean formaldehyde (HCHO) readings, offering insights into urban air quality dynamics:

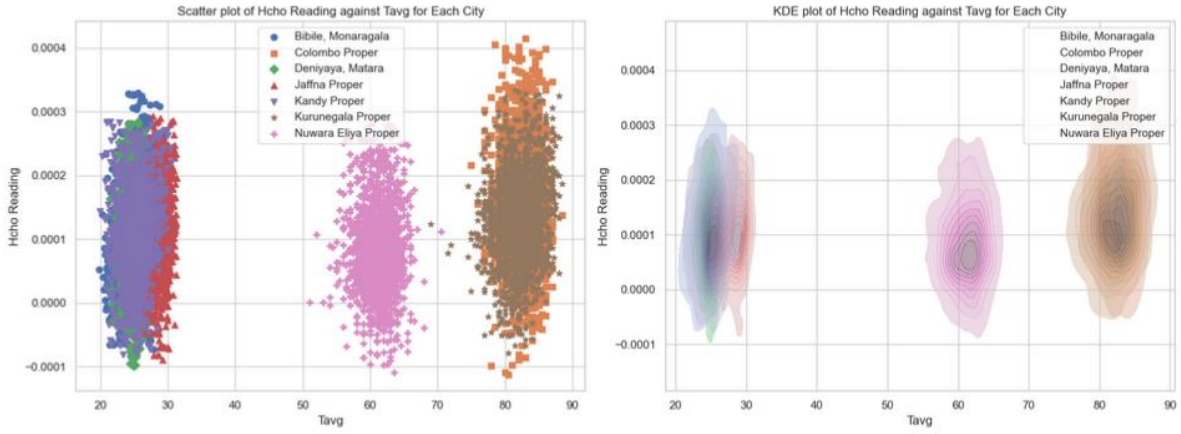
- Elevation: Higher elevation cities tend to exhibit lower HCHO levels, indicating a negative correlation.
- Sea Proximity: Cities closer to the sea generally have higher HCHO levels, reflecting a negative correlation.
- Population density: Higher population density correlates positively with increased HCHO levels, highlighting urban activity's impact on pollution.

Overall, these correlations underscore the complex interplay between urban characteristics and air quality. Understanding how city attributes influence pollutant levels is crucial for effective environmental management and public health initiatives aimed at mitigating air pollution and improving overall urban liveability.

3.5 Relationship between formaldehyde (HCHO) readings and various environmental factors

3.5.1 Temperature average

```
Connection between HCHO_reading and TAVG for Each City
```




```
Correlation between HCHO_reading and TAVG for Each City:
Bibile, Monaragala: 0.3372295431029538
Colombo Proper: -0.060205091249854285
Deniyaya, Matara: 0.0505045213725406
Jaffna Proper: 0.2380125283503767
Kandy Proper: 0.13182157953763363
Kurunegala Proper: 0.16228161966806248
Nuwara Eliya Proper: 0.009741125684240935

Correlations between HCHO_reading and TAVG with City Attributes:
ELEVATION: -0.3732139902986905
Sea Proximity (approximate-km): -0.013060008639682222
Population_density: -0.6702214654584585
```

Correlation between HCHO_reading and TAVG for Each City:

- The correlation values between formaldehyde (HCHO) readings and average temperature (TAVG) vary across different cities.

- The highest positive correlation is observed in Bibile, Monaragala (0.337), indicating a moderate positive relationship between formaldehyde readings and temperature.
- Colombo Proper shows a weak negative correlation (-0.060), suggesting a slight inverse relationship between formaldehyde readings and temperature.
- The rest of the cities show correlations ranging from very weak positive to weak positive.

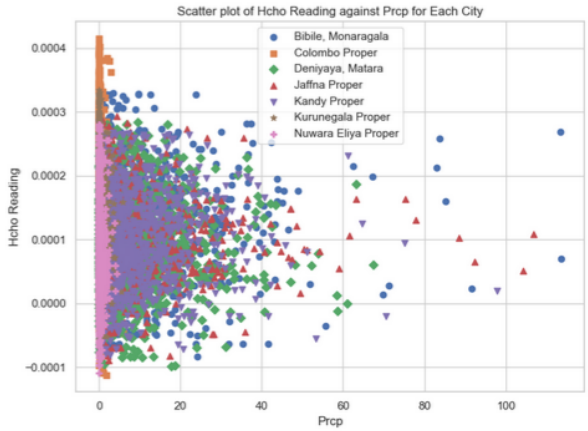
Correlations between “City Attributes” and “correlation between HCHO_Reading and TAVG”:

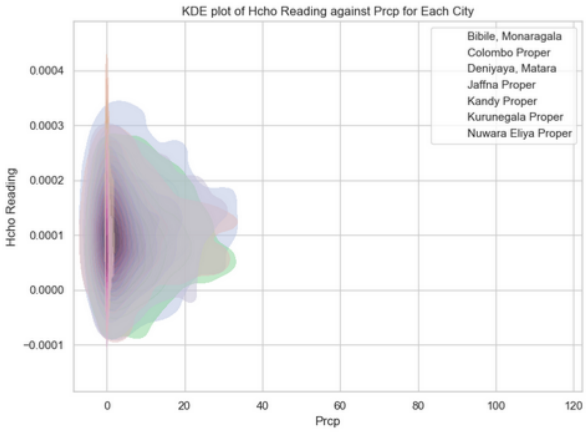
- Elevation: There is a moderate negative correlation (-0.373) between the correlation of HCHO_reading with TAVG and the elevation of cities. This indicates that as the elevation increases, the correlation between formaldehyde readings and temperature tends to decrease.
- Sea Proximity (approximate-km): There is a very weak negative correlation (-0.013) between the “correlation of HCHO_reading with TAVG” and the “proximity of cities to the sea”. This suggests a negligible relationship between sea proximity and the correlation between formaldehyde readings and temperature.
- Population Density: There is a strong negative correlation (-0.670) between the correlation of HCHO_reading with TAVG and the population density of cities. This indicates that as population density increases, the correlation between formaldehyde readings and temperature tends to decrease significantly.

Overall, these correlations provide insights into how the relationship between formaldehyde readings and temperature varies across different cities and how it relates to city attributes such as elevation, sea proximity, and population density.

3.5.2 Precipitation

```
Connection between HCHO_reading and PRCP for Each City
```






```
Correlation between HCHO_reading and PRCP for Each City:
Bibile, Monaragala: -0.01925527765296984
Colombo Proper: -0.08055654544979529
Deniyaya, Matara: -0.021408293705607623
Jaffna Proper: -0.03355965683658126
Kandy Proper: -0.030484655792663285
Kurunegala Proper: -0.0380229965162381
Nuwara Eliya Proper: -0.042819691127610096

Correlations between HCHO_reading and PRCP with City Attributes:
ELEVATION: 0.04394063450387981
Sea Proximity (approximate-km): 0.39484854263471564
Population_density: -0.9064183096686298
```

Correlation between HCHO_reading and PRCP for Each City:

- The correlation values between formaldehyde (HCHO) readings and precipitation (PRCP) across different cities are all negative, albeit very weak.
- This suggests a slight inverse relationship between formaldehyde readings and precipitation, indicating that higher levels of formaldehyde might be associated with lower precipitation levels, although the correlations are close to zero.

Correlations between “City Attributes” and “correlation between HCHO_Reading and PRCP”:

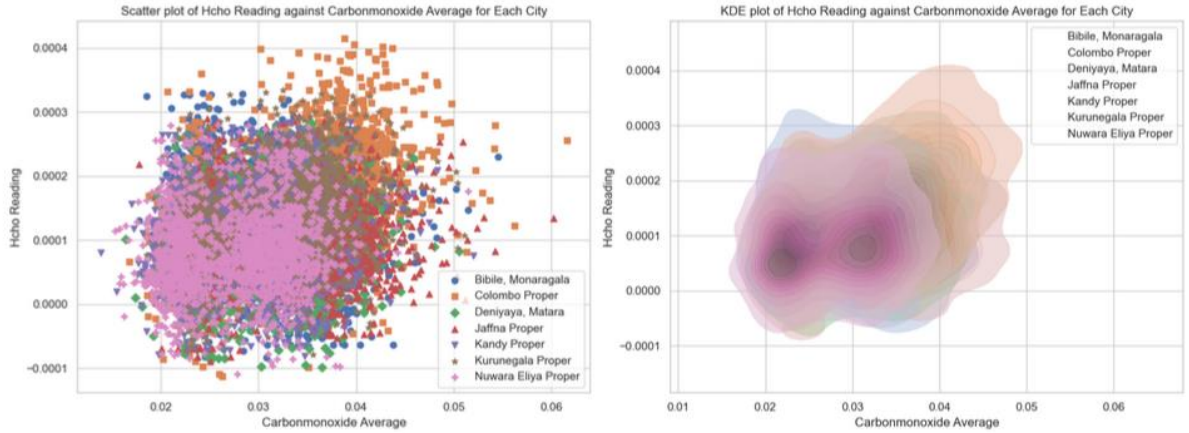
- Elevation: There is a very weak positive correlation (0.044) between the correlation of HCHO_reading with PRCP and the elevation of cities. This suggests a negligible relationship between elevation and the correlation between formaldehyde readings and precipitation.
- Sea Proximity (approximate-km): There is a moderate positive correlation (0.395) between the correlation of HCHO_reading with PRCP and the proximity of cities to the sea. This indicates that as cities are closer to the sea, the correlation between formaldehyde readings and precipitation tends to increase.

- Population Density: There is a strong negative correlation (-0.906) between the correlation of HCHO_reading with PRCP and the population density of cities. This indicates that as population density increases, the correlation between formaldehyde readings and precipitation tends to decrease significantly.

These correlations provide insights into how the relationship between formaldehyde readings and precipitation varies across different cities and how it relates to city attributes such as elevation, sea proximity, and population density.

3.5.3 Carbonmonoxide average

```
Connection between HCHO_reading and carbonmonoxide_average for Each City
```




```
Correlation between HCHO_reading and carbonmonoxide_average for Each City:
Bibile, Monaragala: -0.2547802078409578
Colombo Proper: 0.5494987539892361
Deniyaya, Matara: 0.23325059494271905
Jaffna Proper: 0.029594646984790016
Kandy Proper: 0.16200612071207518
Kurunegala Proper: 0.31585133415798883
Nuwara Eliya Proper: 0.13630160873144104

Correlations between HCHO_reading and carbonmonoxide_average with City Attributes:
ELEVATION: -0.08144587495832242
Sea Proximity (approximate-km): -0.22793695079855633
Population_density: 0.7490581496221295
```

Correlation between HCHO_reading and carbonmonoxide_average for Each City:

- The correlation values between formaldehyde (HCHO) readings and average carbon monoxide levels (carbonmonoxide_average) across different cities vary.
- Bibile, Monaragala has a moderate negative correlation (-0.255), indicating a somewhat inverse relationship between formaldehyde readings and carbon monoxide levels.
- Colombo Proper exhibits a strong positive correlation (0.549), suggesting a significant positive relationship between formaldehyde readings and carbon monoxide levels.

- The rest of the cities show correlations ranging from very weak positive to moderate positive.

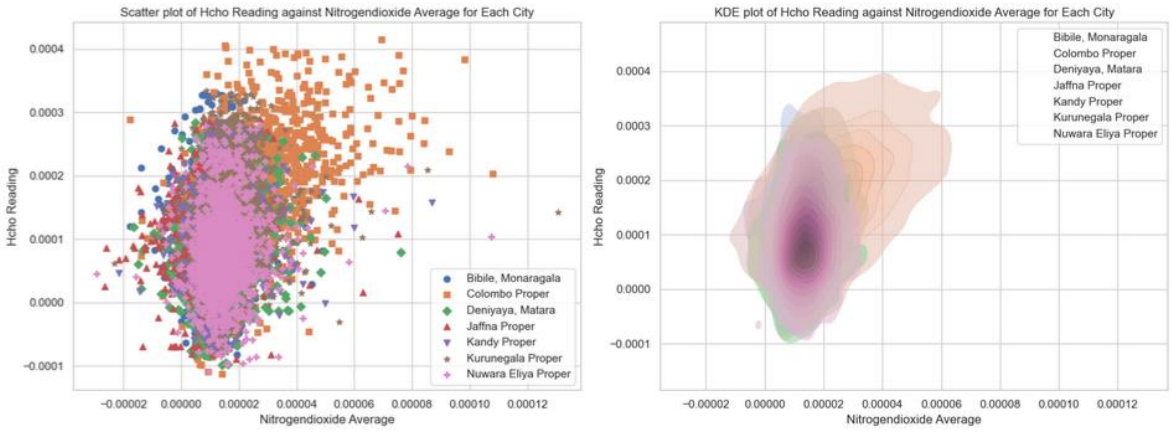
Correlations between “City Attributes” and “correlation between HCHO_Reading and carbonmonoxide_average”:

- Elevation: There is a very weak negative correlation (-0.081) between the correlation of HCHO_reading with carbonmonoxide_average and the elevation of cities. This indicates a negligible relationship between elevation and the correlation between formaldehyde readings and carbon monoxide levels.
- Sea Proximity (approximate-km): There is a moderate negative correlation (-0.228) between the correlation of HCHO_reading with carbonmonoxide_average and the proximity of cities to the sea. This suggests that as cities are closer to the sea, the correlation between formaldehyde readings and carbon monoxide levels tends to decrease.
- Population Density: There is a strong positive correlation (0.749) between the correlation of HCHO_reading with carbonmonoxide_average and the population density of cities. This indicates that as population density increases, the correlation between formaldehyde readings and carbon monoxide levels tends to increase significantly.

These correlations provide insights into how the relationship between formaldehyde readings and carbon monoxide levels varies across different cities and how it relates to city attributes such as elevation, sea proximity, and population density.

3.5.4 Nitrogendioxide

```
Connection between HCHO_reading and nitrogendioxide_average for Each City
```




```
Correlation between HCHO_reading and nitrogendioxide_average for Each City:
Bibile, Monaragala: 0.12001358034267195
Colombo Proper: 0.5283587151139871
Deniyaya, Matara: 0.15941546818250388
Jaffna Proper: 0.11442518308462683
Kandy Proper: 0.11726505604248977
Kurunegala Proper: 0.17338922592099018
Nuwara Eliya Proper: 0.08373296768646923

Correlations between HCHO_reading and nitrogendioxide_average with City Attributes:
ELEVATION: -0.3966928056009707
Sea Proximity (approximate-km): -0.5919833105532565
Population_density: 0.9630376667288855
```

Correlation between HCHO_reading and nitrogendioxide_average for Each City:

- The correlation values between formaldehyde (HCHO) readings and average nitrogen dioxide levels (nitrogendioxide_average) across different cities vary.
- Colombo Proper exhibits the highest positive correlation (0.528), indicating a moderate positive relationship between formaldehyde readings and nitrogen dioxide levels.
- The correlations for the other cities range from very weak positive to moderate positive, with values between approximately 0.083 and 0.173.

Correlations between “City Attributes” and “correlation between HCHO_Reading and nitrogendioxide_average”:

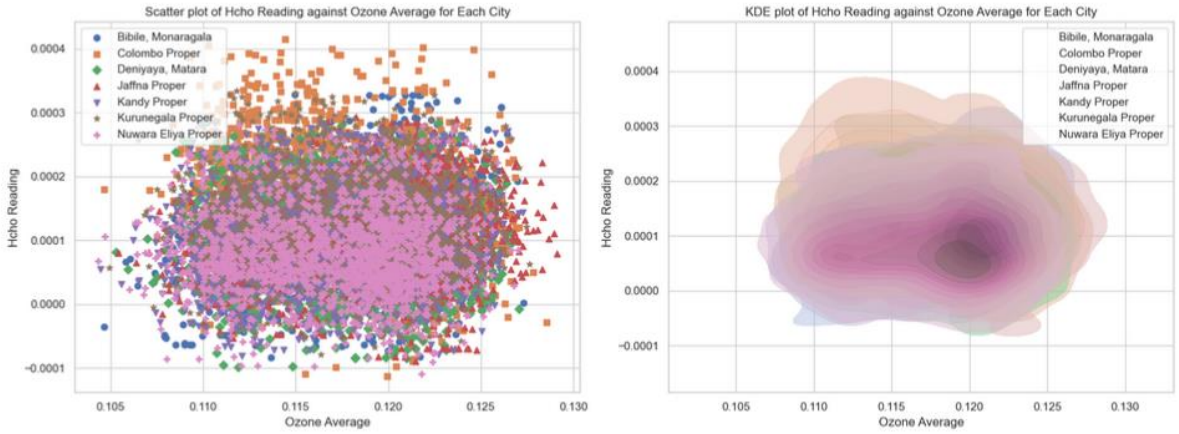
- Elevation: There is a moderate negative correlation (-0.397) between the correlation of HCHO_reading with nitrogendioxide_average and the elevation of cities. This indicates that as the elevation increases, the correlation between formaldehyde readings and nitrogen dioxide levels tends to decrease.
- Sea Proximity (approximate-km): There is a strong negative correlation (-0.592) between the correlation of HCHO_reading with nitrogendioxide_average and the proximity of cities to the sea. This suggests that as cities are closer to the sea, the correlation between formaldehyde readings and nitrogen dioxide levels tends to decrease significantly.

- Population Density: There is a very strong positive correlation (0.963) between the correlation of HCHO_reading with nitrogendioxide_average and the population density of cities. This indicates that as population density increases, the correlation between formaldehyde readings and nitrogen dioxide levels tends to increase significantly.

These correlations provide insights into how the relationship between formaldehyde readings and nitrogen dioxide levels varies across different cities and how it relates to city attributes such as elevation, sea proximity, and population density.

3.5.5 Ozone

```
Connection between HCHO_reading and ozone_average for Each City
```




```
Correlation between HCHO_reading and ozone_average for Each City:
Bibile, Monaragala: 0.38845649191309295
Colombo Proper: -0.3792764104279857
Deniyaya, Matara: -0.12227962643012684
Jaffna Proper: 0.18238976252011668
Kandy Proper: 0.022921071182112496
Kurunegala Proper: -0.09974460576900963
Nuwara Eliya Proper: -0.018272261032732346

Correlations between HCHO_reading and ozone_average with City Attributes:
ELEVATION: -0.013243142783596834
Sea Proximity (approximate-km): 0.17678640479638058
Population_density: -0.7499003233686119
```

Correlation between HCHO_reading and ozone_average for Each City:

- The correlation values between formaldehyde (HCHO) readings and average ozone levels (ozone_average) across different cities vary.
- Bibile, Monaragala exhibits the highest positive correlation (0.388), indicating a moderate positive relationship between formaldehyde readings and ozone levels.
- Colombo Proper shows a strong negative correlation (-0.379), suggesting a significant inverse relationship between formaldehyde readings and ozone levels.

- The correlations for the other cities range from very weak negative to weak positive, with values between approximately -0.122 and 0.183.

Correlations between “City Attributes” and “correlation between HCHO_Reading and ozone_average”:

- Elevation: There is a very weak negative correlation (-0.013) between the correlation of HCHO_reading with ozone_average and the elevation of cities. This suggests a negligible relationship between elevation and the correlation between formaldehyde readings and ozone levels.
- Sea Proximity (approximate-km): There is a weak positive correlation (0.177) between the correlation of HCHO_reading with ozone_average and the proximity of cities to the sea. This indicates that as cities are closer to the sea, the correlation between formaldehyde readings and ozone levels tends to increase slightly.
- Population Density: There is a strong negative correlation (-0.750) between the correlation of HCHO_reading with ozone_average and the population density of cities. This indicates that as population density increases, the correlation between formaldehyde readings and ozone levels tends to decrease significantly.

These correlations provide insights into how the relationship between formaldehyde readings and ozone levels varies across different cities and how it relates to city attributes such as elevation, sea proximity, and population density.

3.6 Covid analysis

```
# Calculate annual mean HCHO reading for each city without creating a new column for year
annual_mean_hcho = merged_df.groupby([merged_df['Current_Date'].dt.year, 'Location'])['HCHO_reading'].mean().unstack(l

# Calculate annual mean 'new COVID cases' for each city without creating a new column for year
annual_mean_covid = merged_df.groupby([merged_df['Current_Date'].dt.year, 'Location'])['new'].mean().unstack(level=1)

# Calculate correlation between annual mean HCHO readings and annual mean 'new COVID cases' for each city
correlation = annual_mean_hcho.corrwith(annual_mean_covid)

# Print correlation
print("Correlation between Annual Mean HCHO Readings and Annual Mean New COVID Cases:")
print(correlation)
```

```
Correlation between Annual Mean HCHO Readings and Annual Mean New COVID Cases:
Location
Bibile, Monaragala    -0.222091
Colombo Proper        -0.764273
Deniyaya, Matara      -0.825466
Jaffna Proper         -0.959711
Kandy Proper          -0.834530
Kurunegala Proper     -0.604850
Nuwara Eliya Proper   -0.542305
dtype: float64
```

The output shows the correlation coefficients between the annual mean formaldehyde (HCHO) readings and the annual mean new COVID-19 cases for each city. Here's the interpretation:

- **Negative Correlation**: All correlation coefficients are negative, indicating an inverse relationship between annual mean HCHO readings and annual mean new COVID-19 cases in each city.
- **Strength of Correlation**: The correlation coefficients range from approximately -0.96 to -0.22. The closer the coefficient is to -1, the stronger the negative correlation. For instance, Jaffna Proper has the strongest negative correlation (-0.96), indicating a very strong inverse relationship between HCHO levels and new COVID-19 cases. On the other hand, Bibile, Monaragala has the weakest negative correlation (-0.22), suggesting a comparatively weaker inverse relationship.
- **City-Specific Insights**: The correlation coefficients provide insights into how changes in air quality, as measured by HCHO levels, relate to the incidence of new COVID-19 cases in each city. Cities with stronger negative correlations may indicate a more pronounced impact of air quality on COVID-19 transmission, while weaker correlations suggest other factors may be influencing disease spread to a greater extent.

From the below plots, we can determine the following:

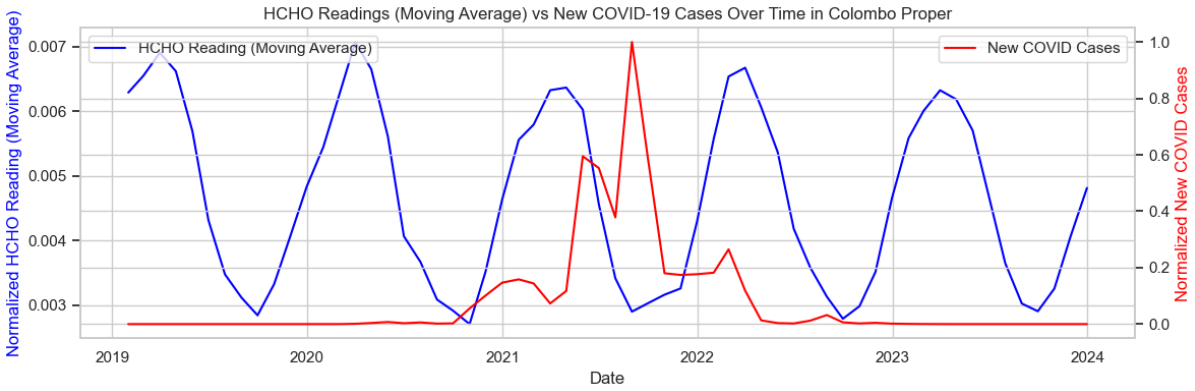
1. **Time Series Plot (Top)**:
    - This plot shows the 3-month moving average of formaldehyde (HCHO) readings (blue line) and the daily new COVID-19 cases (red line) over time for each location.
    - It allows us to observe trends and potential correlations between HCHO levels and COVID-19 cases. For example, we can identify periods of simultaneous increases or decreases in both HCHO levels and COVID-19 cases, which may suggest a relationship between air quality and disease transmission.

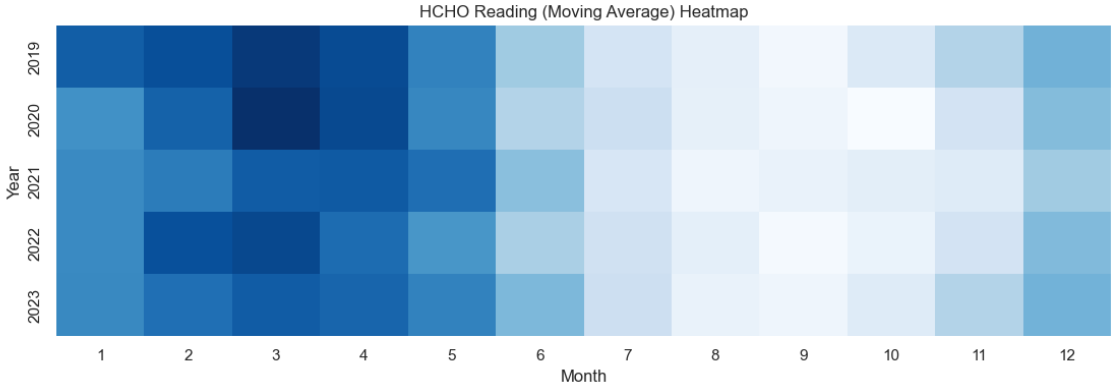
2. **Heatmap (Bottom)**:
    - The heatmap displays the monthly average HCHO readings (3-month moving average) for each year, allowing us to observe seasonal patterns and long-term trends in HCHO levels.
    - Each row represents a year, and each column represents a month. The color intensity represents the magnitude of HCHO levels, with darker shades indicating higher concentrations.
    - By examining changes in color intensity over time, we can identify any consistent patterns or anomalies in HCHO levels across different months and years.

Overall, these plots provide a comprehensive visualization of HCHO levels and COVID-19 cases over time for each location, facilitating the analysis of trends and potential correlations between air quality and disease spread.

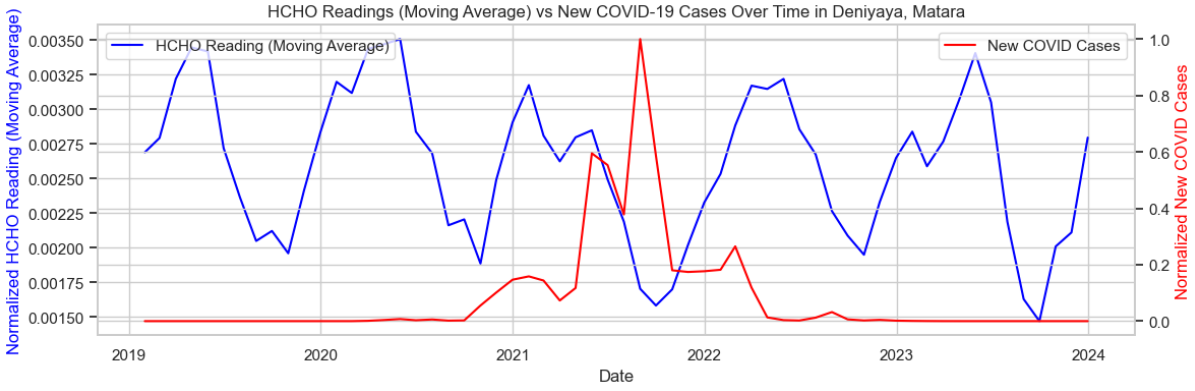
Bibile, Monaragala

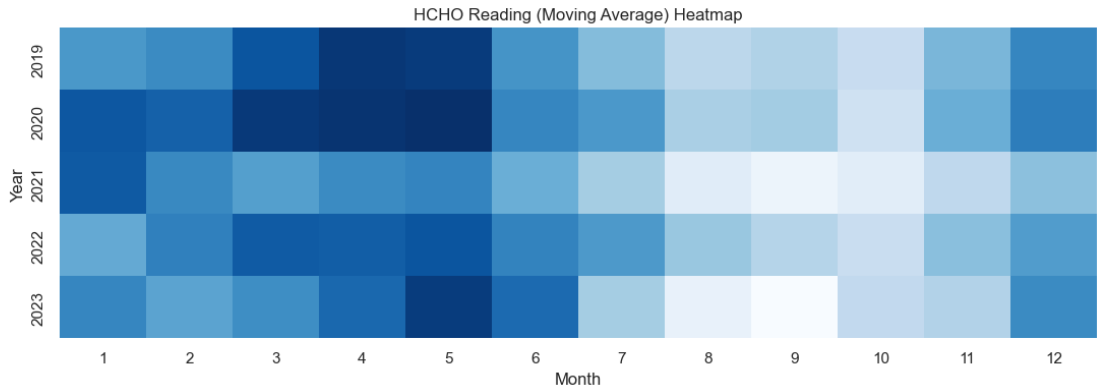
Colombo Proper



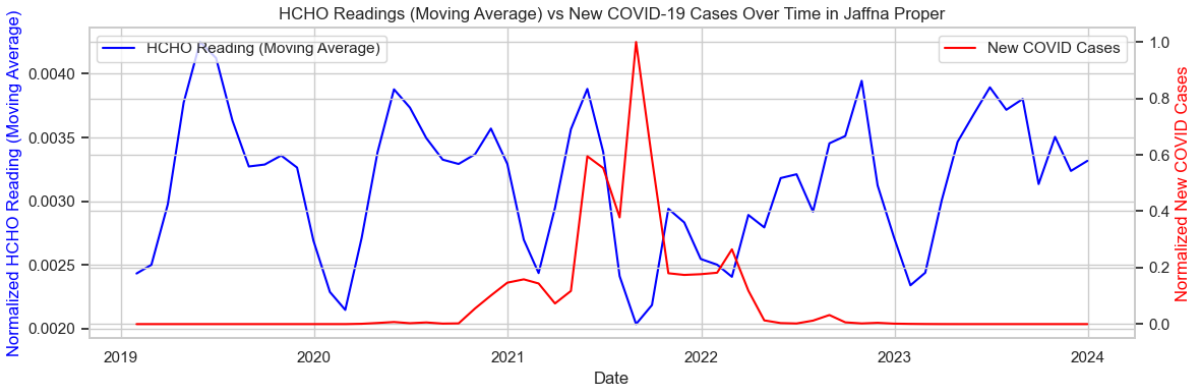


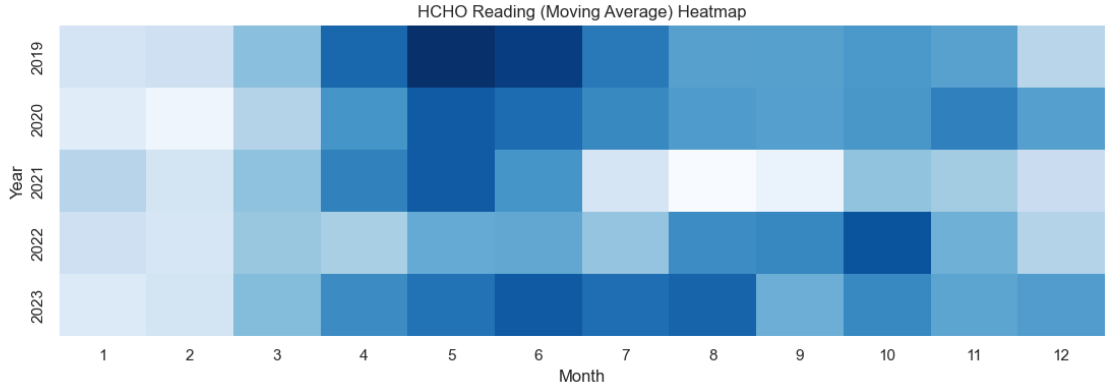
Deniyaya, Matara



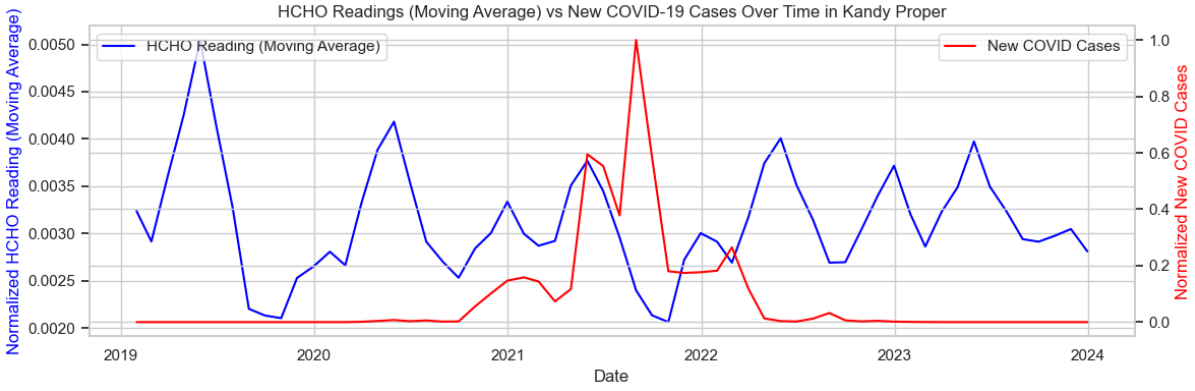


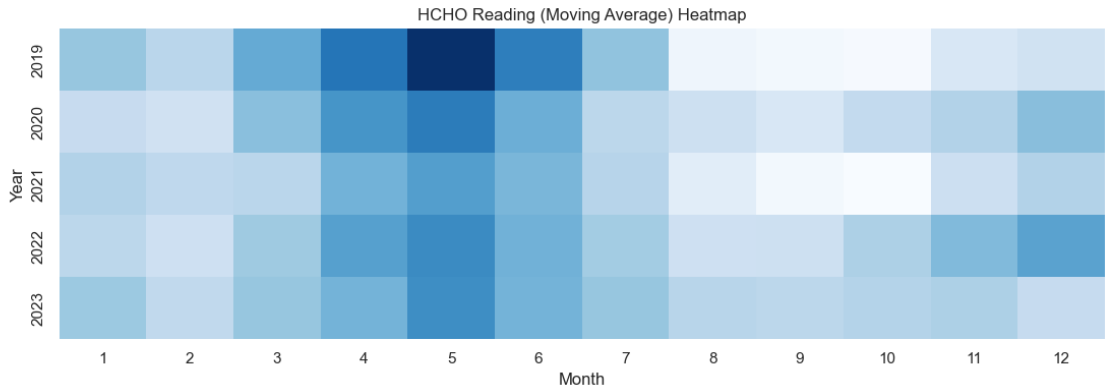
Jaffna Proper



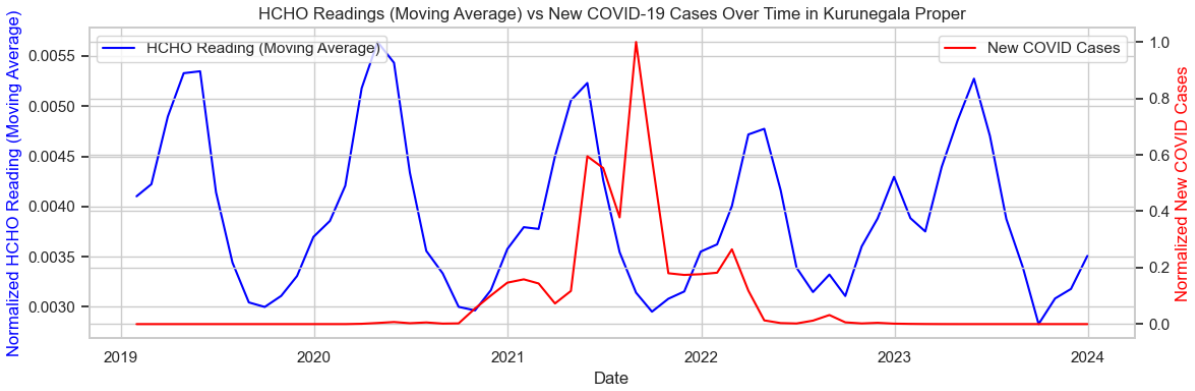


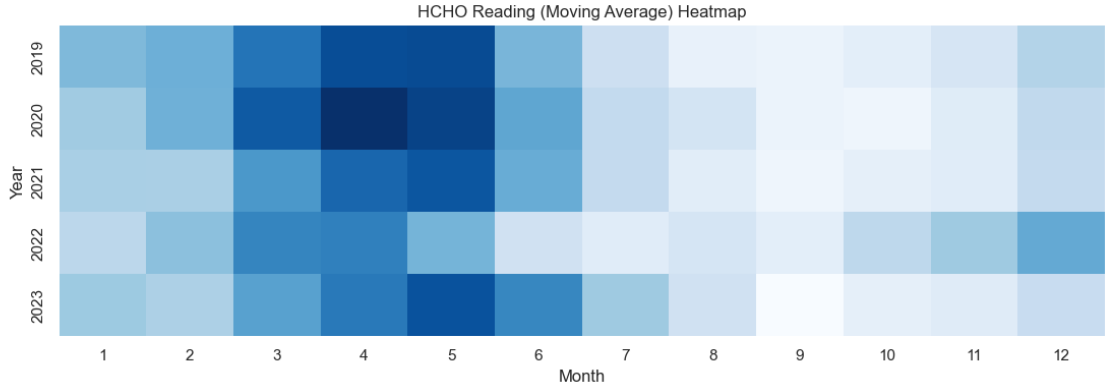
Kandy Proper



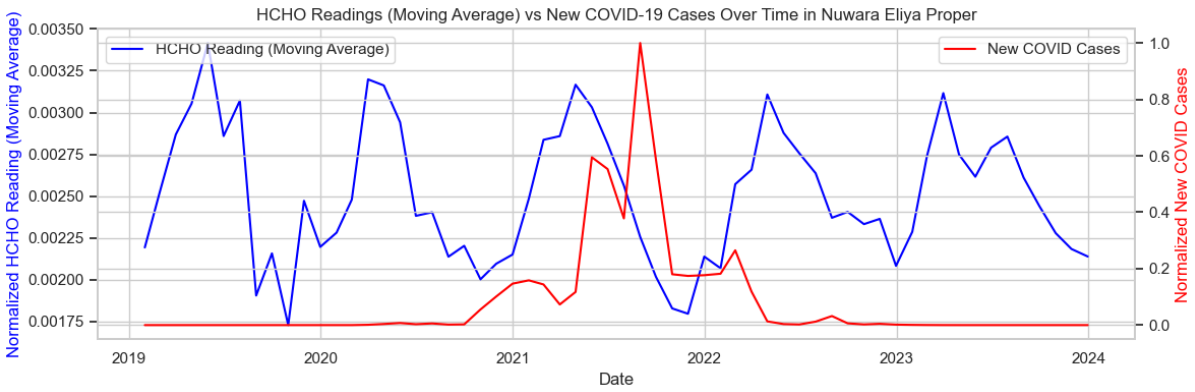


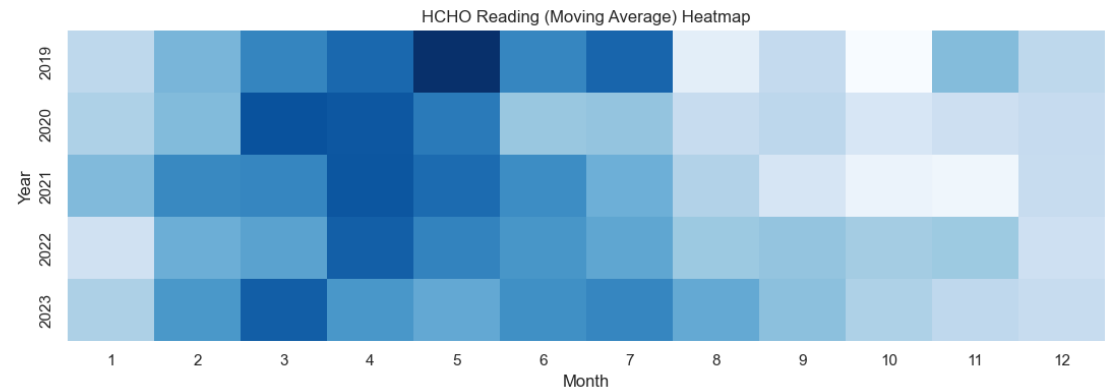
Kurunegala Proper





Nuwara Eliya Proper





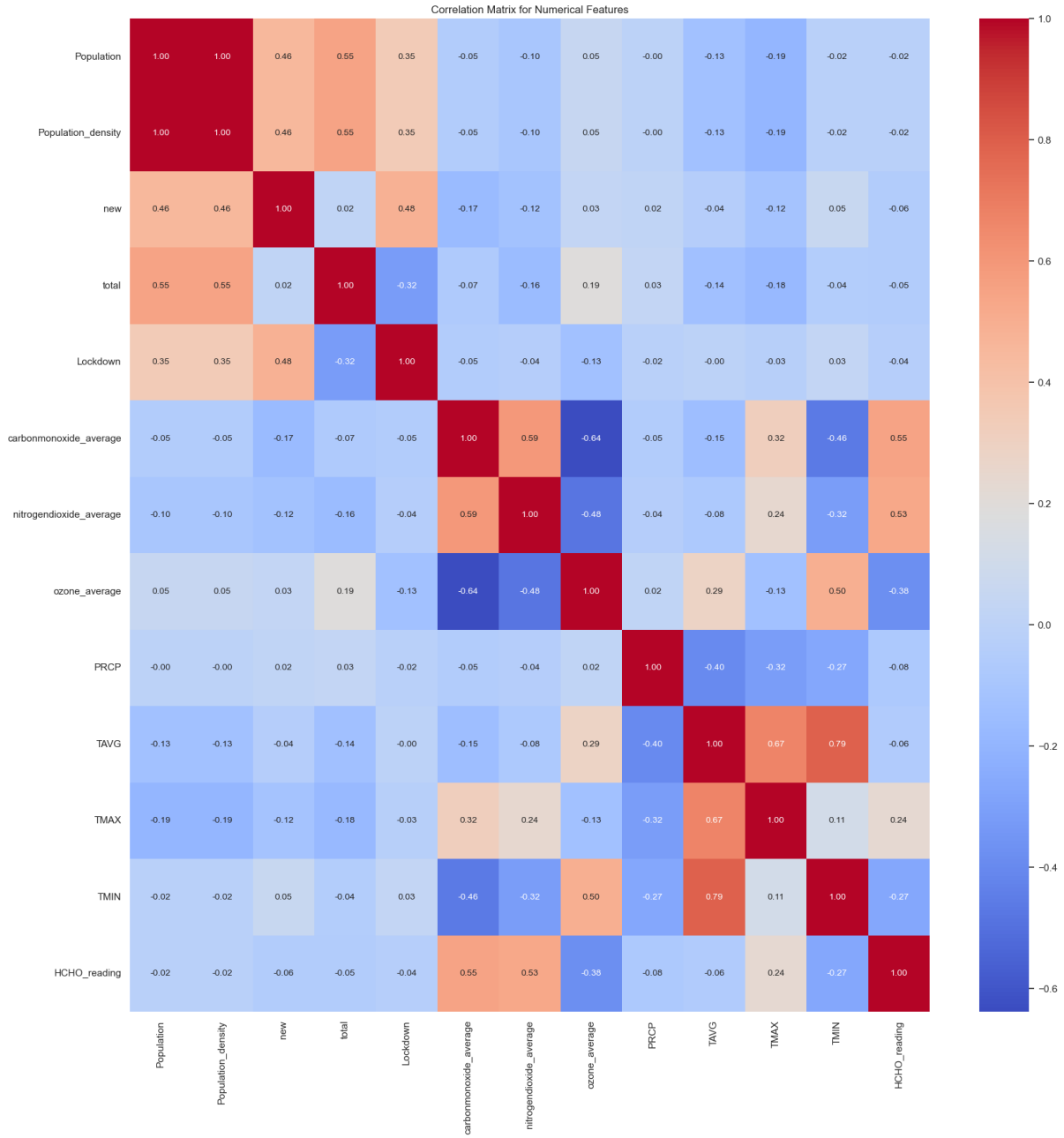
By observing the above plots we can see that there's a decrease in monthly sum of HCHO readings when the monthly new covid cases spike.

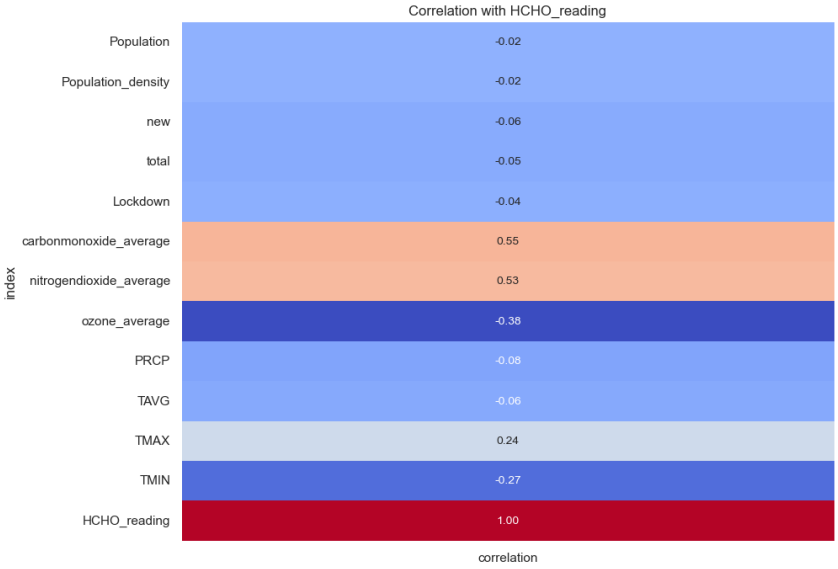
3.7 City wise Correlation analysis

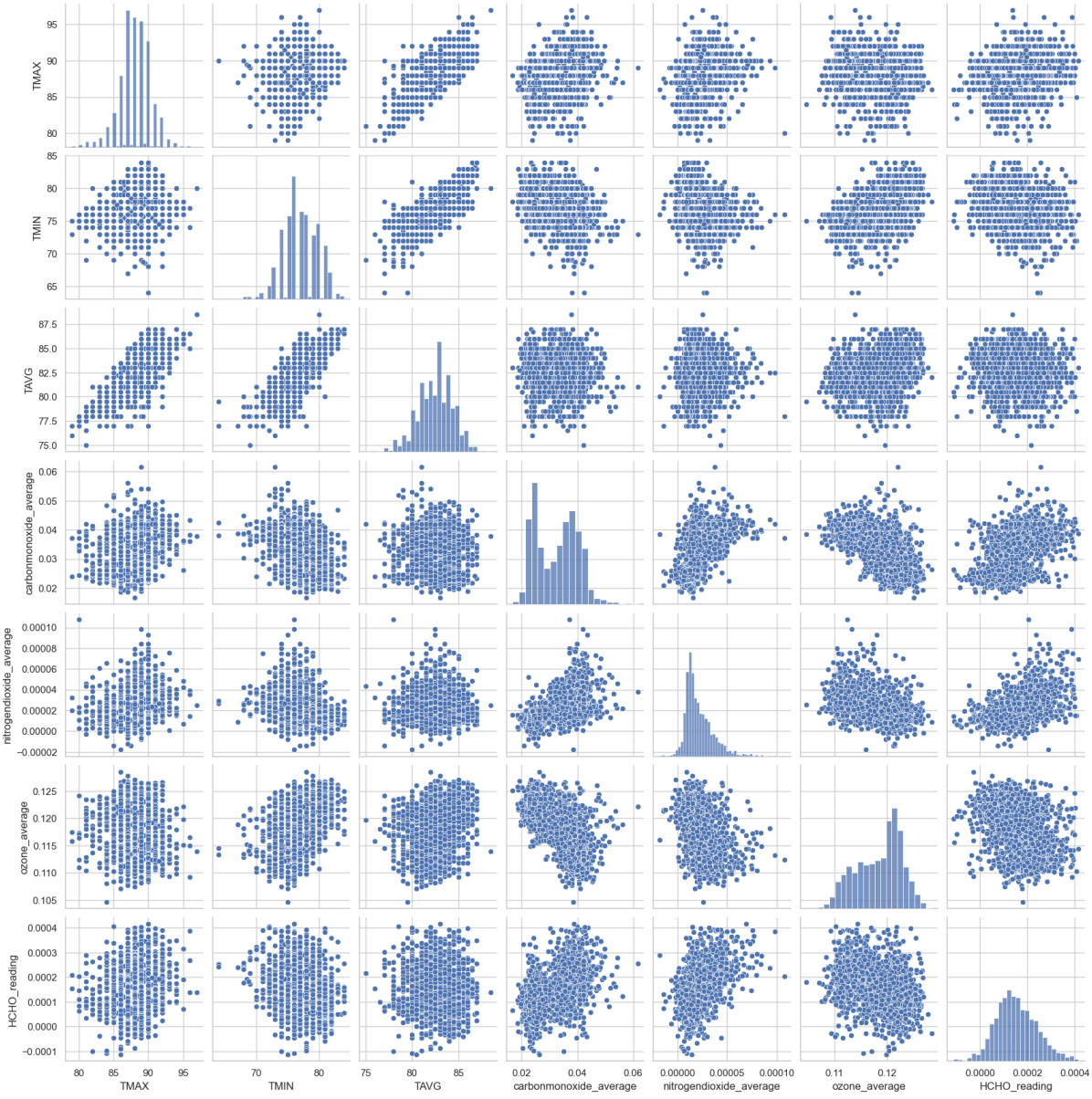
1. **Correlation Matrix Heatmap**:
    - This plot shows the correlation coefficients between all pairs of numerical features in the dataset.
    - Each cell represents the correlation coefficient between two features.
    - The color intensity and annotation within each cell represent the strength and direction of the correlation. Positive values indicate a positive correlation, negative values indicate a negative correlation, and values closer to zero indicate weaker or no correlation.
    - This plot helps identify potential multicollinearity issues between features and gives an overview of how each feature relates to others in the dataset.
2. **Correlation with HCHO_reading Heatmap**:
    - This plot specifically focuses on the correlation coefficients between numerical features and the target variable 'HCHO_reading'.
    - Similar to the correlation matrix heatmap, each cell represents the correlation coefficient between a feature and the target variable.
    - It helps identify which features have the strongest and weakest correlations with the target variable, aiding in feature selection for predictive modeling.
3. **Pairplot**:
    - The pairplot displays pairwise relationships between numerical columns in the dataset.
    - It shows scatterplots for each pair of features and histograms for each individual feature along the diagonal.
    - This plot helps visualize the distribution of each numerical feature, as well as the relationship between each pair of features. It can reveal linear or nonlinear patterns, outliers, and potential clusters in the data.
4. **Regression Plots**:
    - These plots depict the relationship between each numerical feature and the target variable 'HCHO_reading' using regression lines.
    - Each plot shows a scatterplot of the data points along with a regression line fitted to the data.
    - They help assess the strength and direction of the linear relationship between each feature and the target variable.
    - By observing the slope and direction of the regression line, we can determine whether the relationship is positive or negative and the magnitude of the effect of the feature on the target variable.

Overall, these plots provide valuable insights into the relationships and correlations within the dataset, helping to inform feature selection, identify patterns, and understand the potential predictive power of the features.

1. Colombo

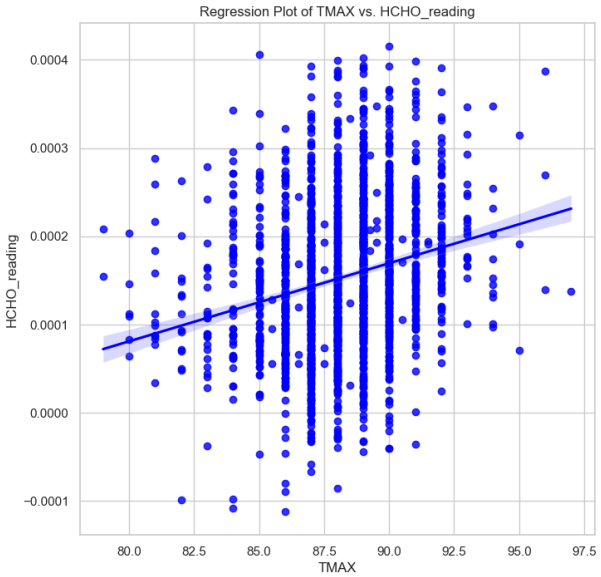

Correlation Matrix for Numerical Features
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

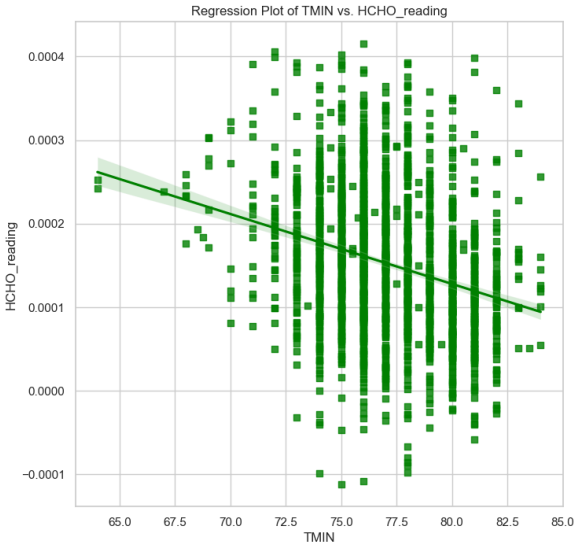

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
index
-0.02
-0.02
-0.06
-0.05
-0.04
0.55
0.53
-0.38
-0.08
-0.06
0.24
-0.27
1.00
correlation

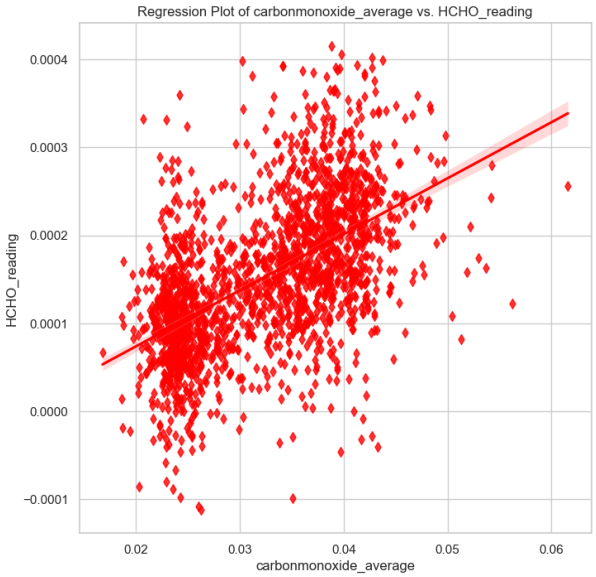

TMAX
TMIN
TAVG
carbonmonoxide_average
nitrogendioxide_average
ozone_average
HCHO_reading

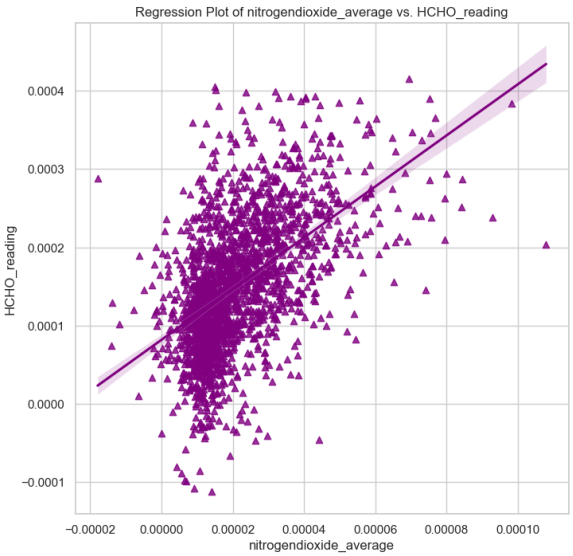
Regression plots

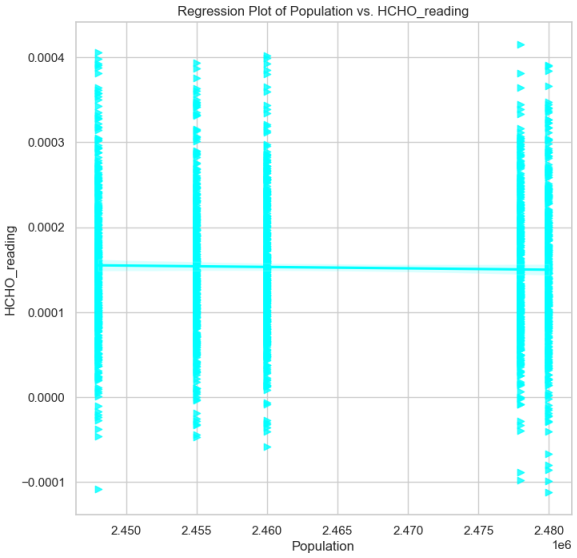
TMAX

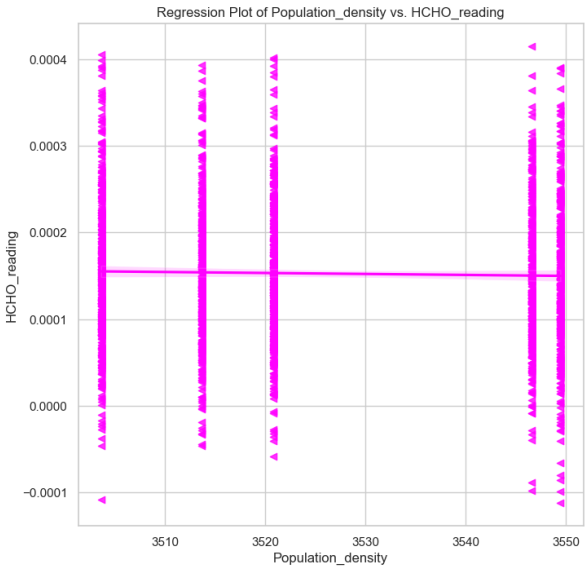


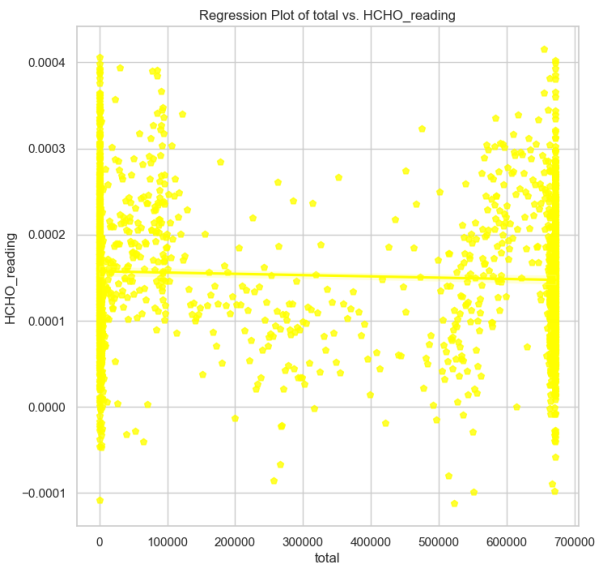


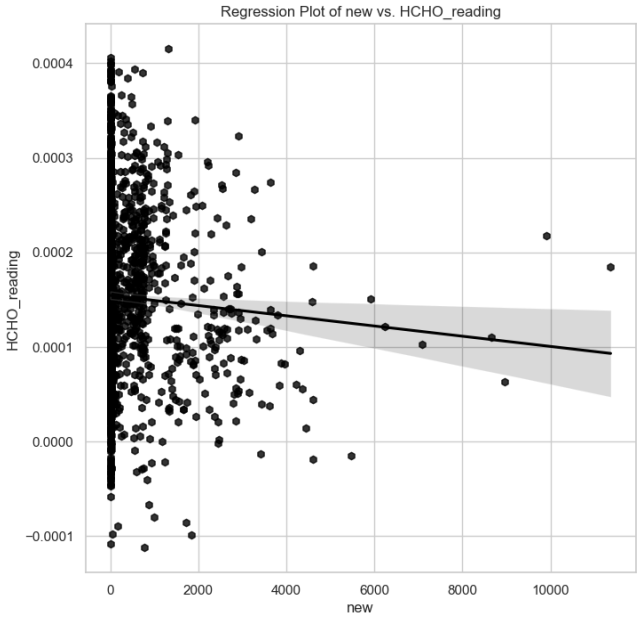






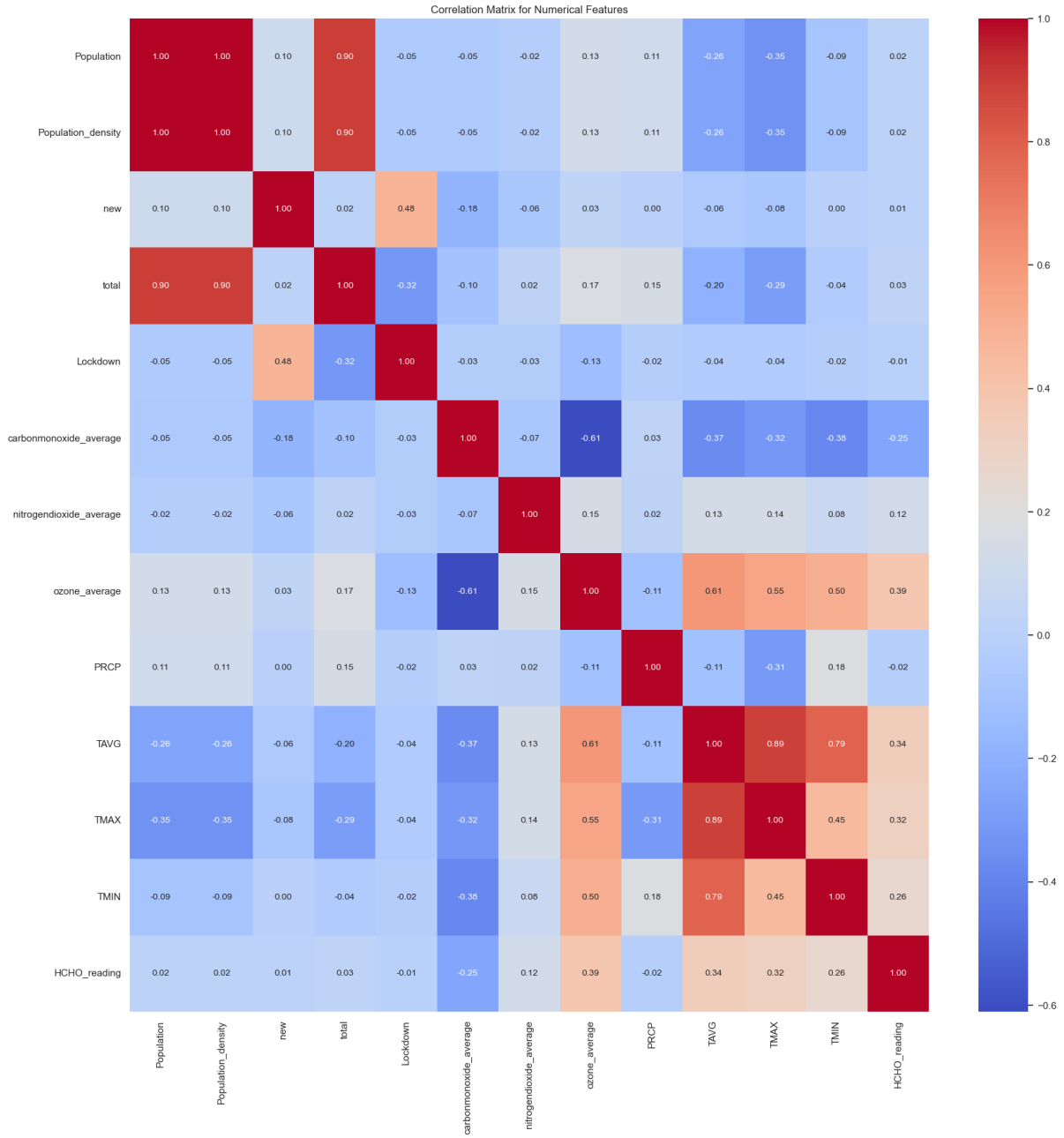


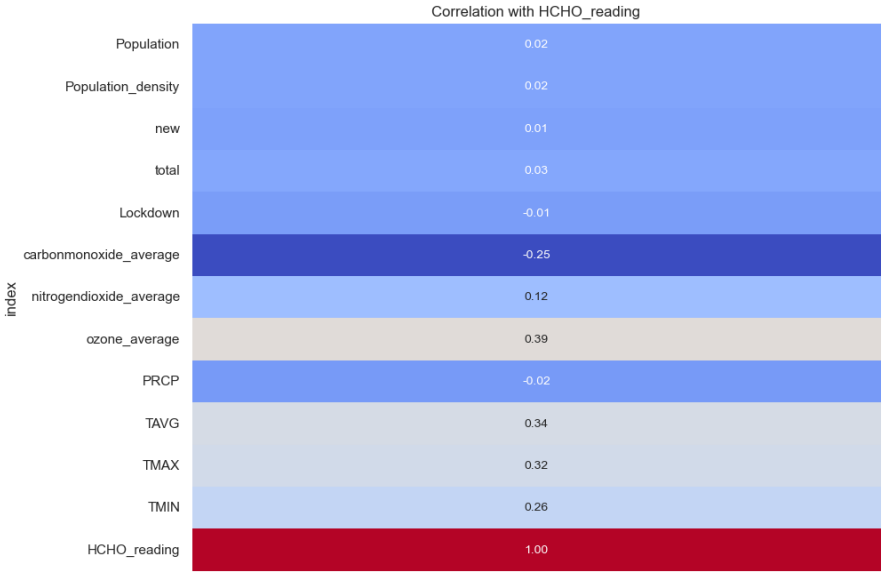


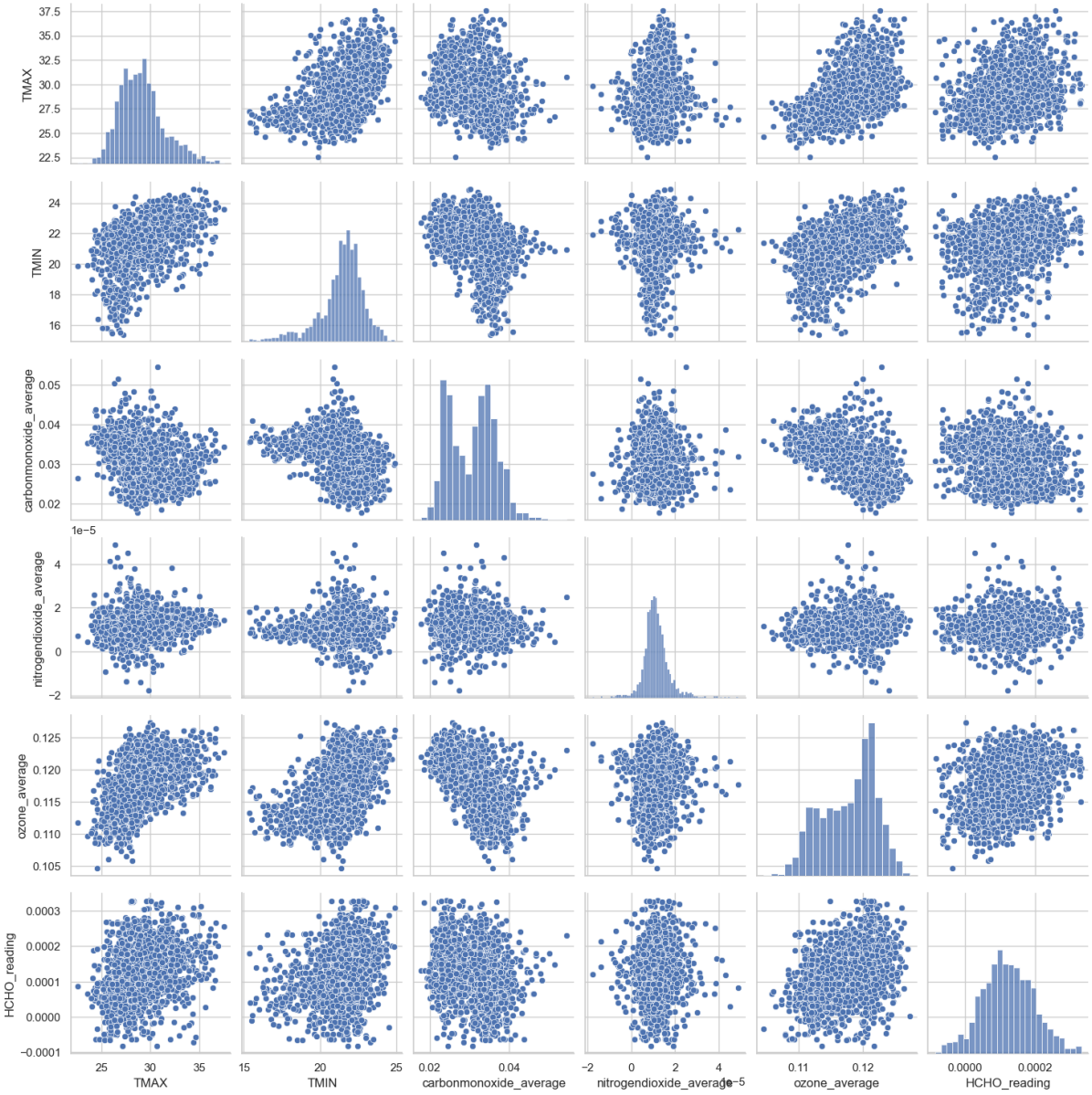


Index	Correlation	Ranking
carbonmonoxide_average	0.549499	1
nitrogendioxide_average	0.528359	2
TMAX	0.236589	3
ozone_average	-0.379276	4
TMIN	-0.272357	5
PRCP	-0.080557	6
TAVG	-0.060205	7
new	-0.055789	8
total	-0.053322	9
Lockdown	-0.035609	10
Population_density	-0.023881	11
Population	-0.023881	12

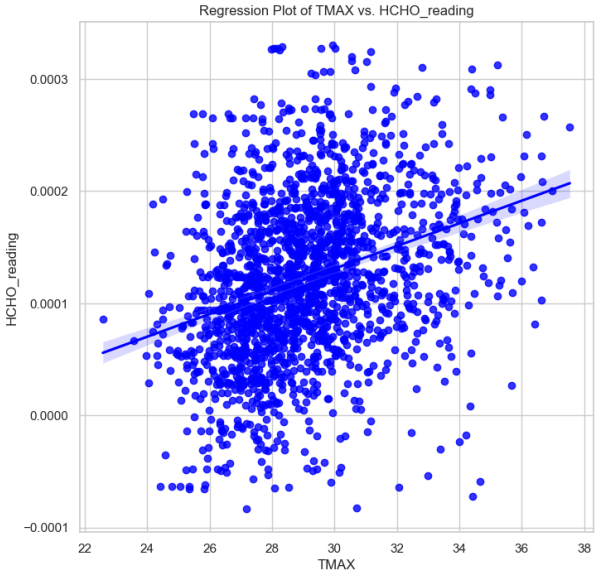
2. Bibile Monaragala

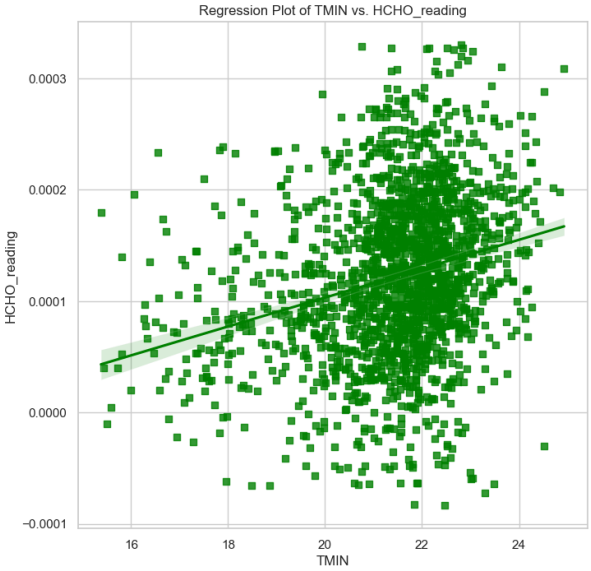


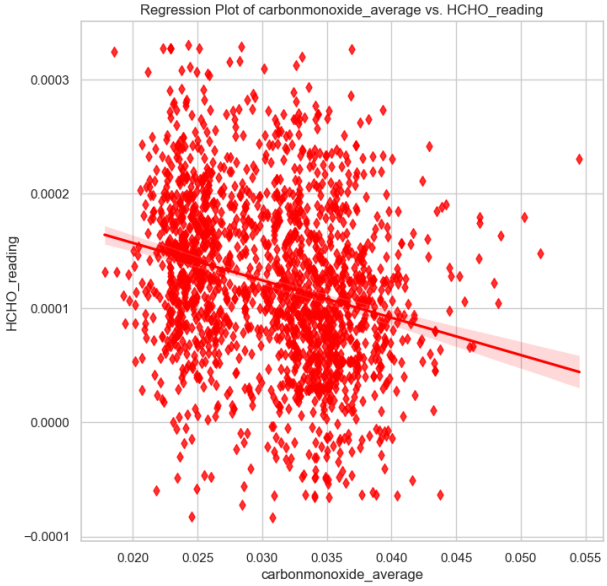

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
0.02
0.02
0.01
0.03
-0.01
-0.25
0.12
0.39
-0.02
0.34
0.32
0.26
1.00
index
correlation

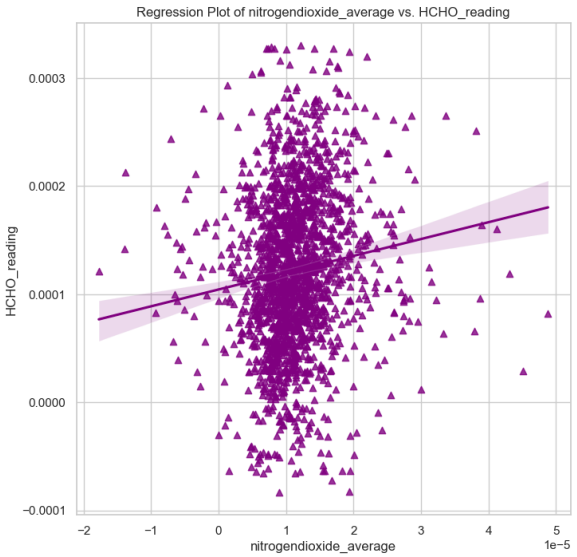

TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
HCHO_reading
1e−5
1e−5

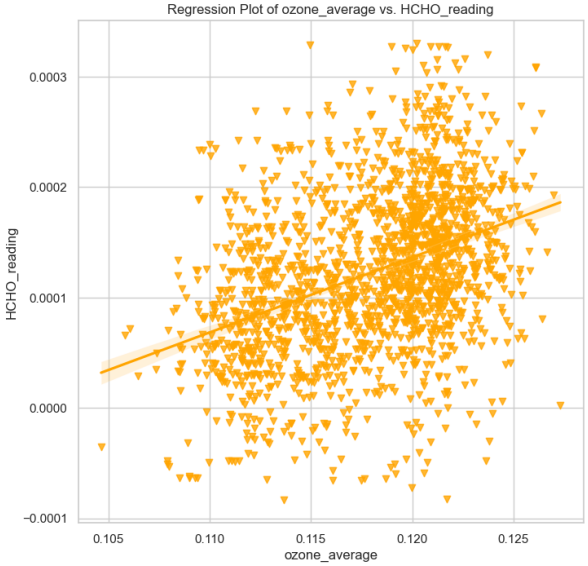
Regression Plots

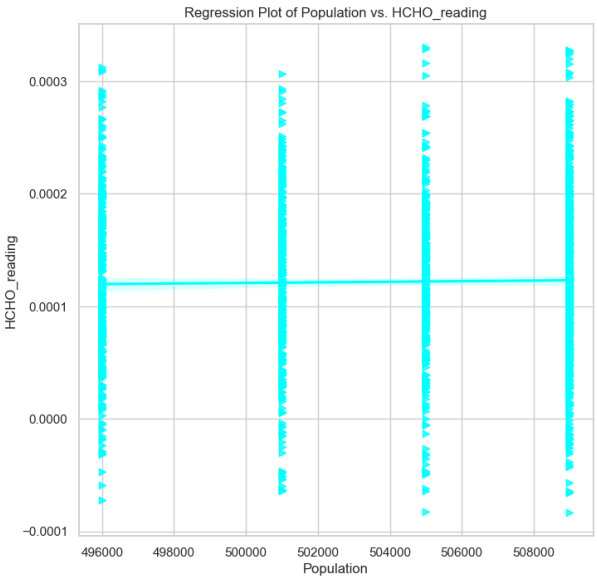


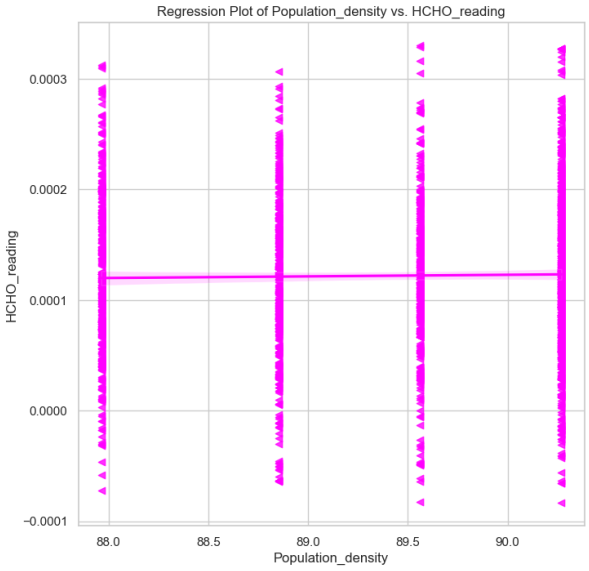


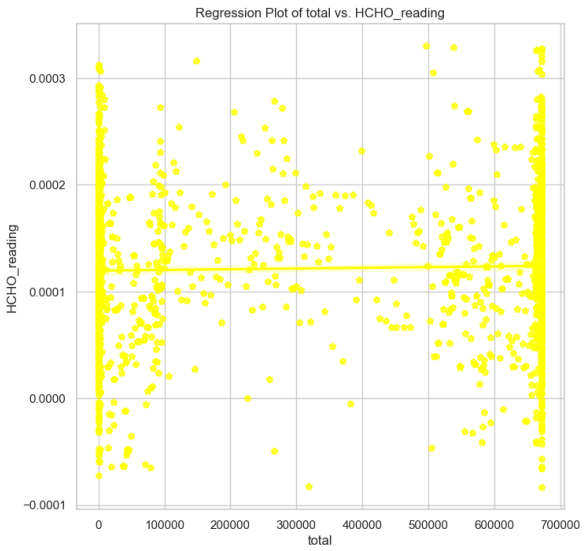


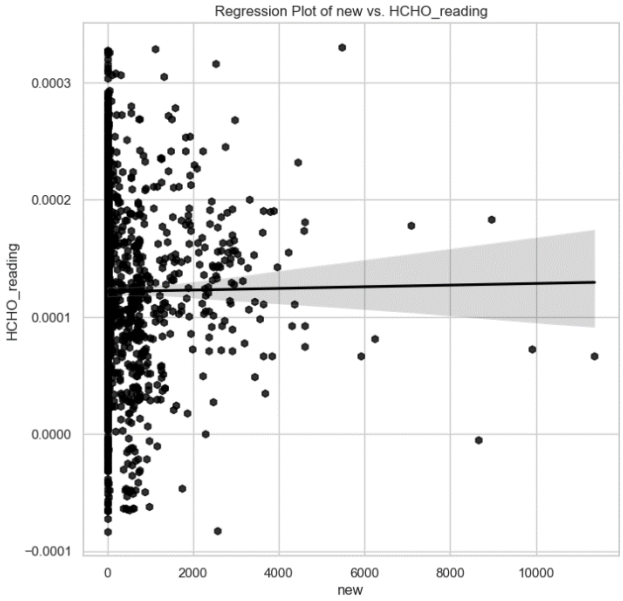






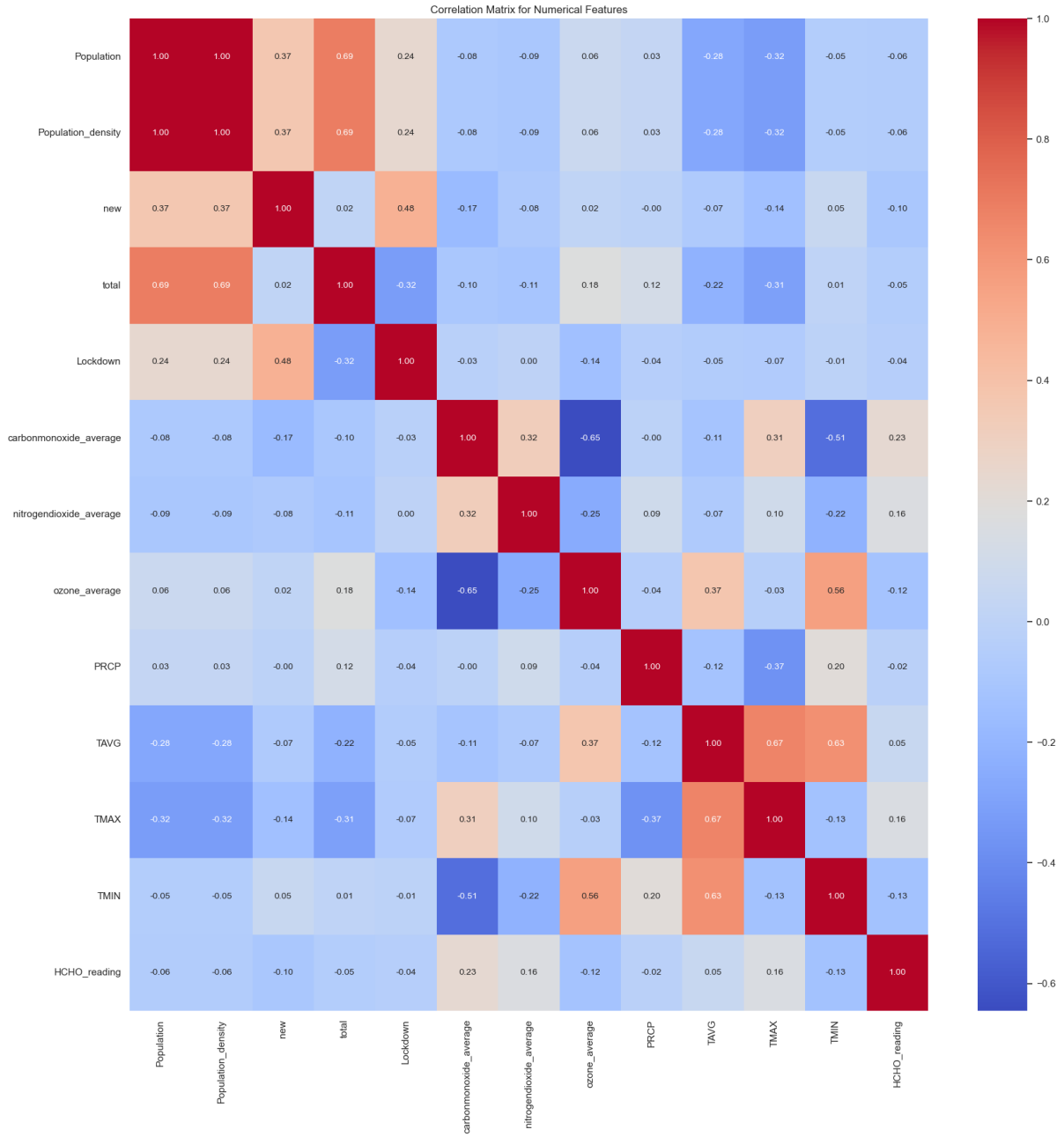


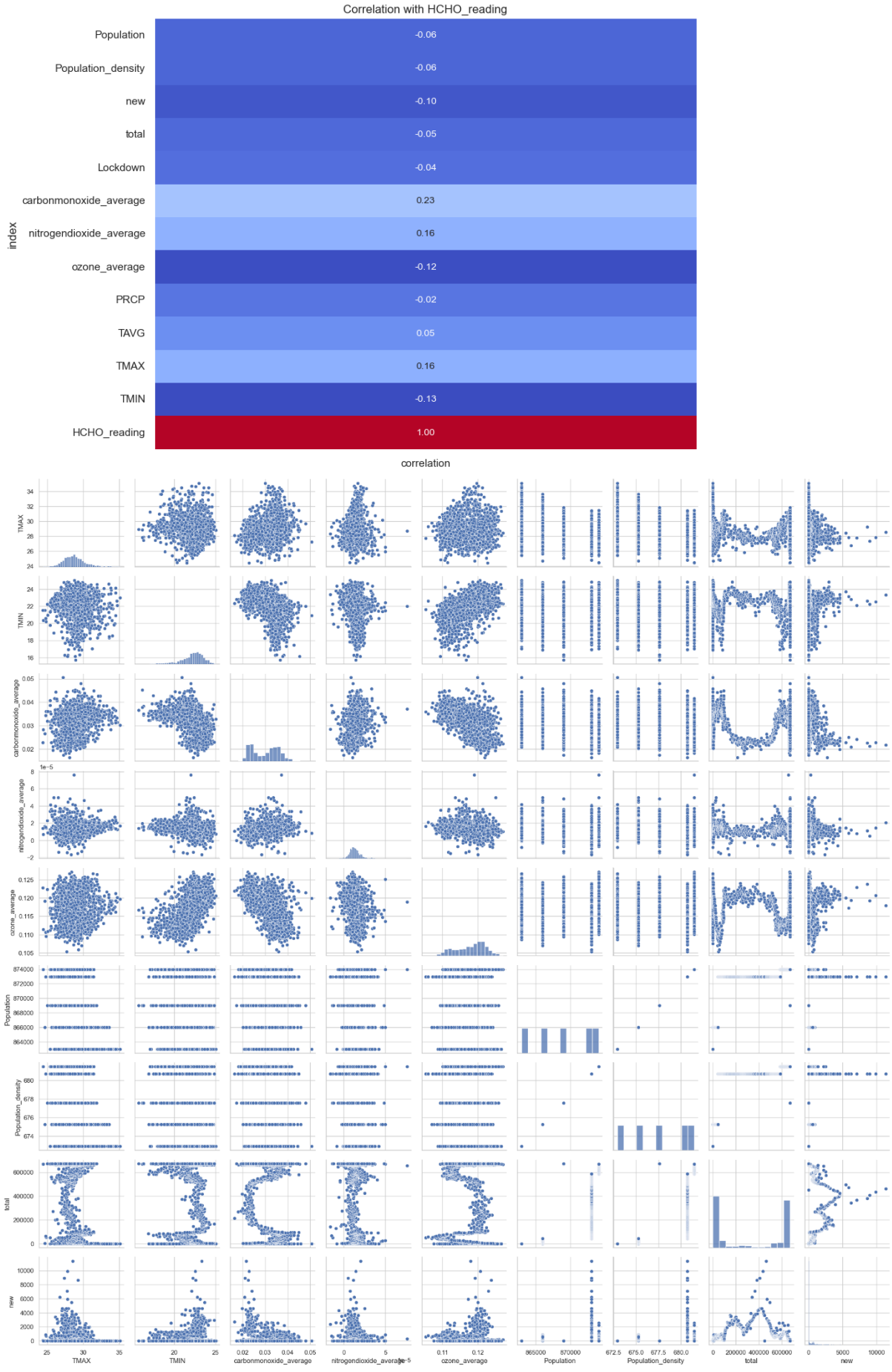




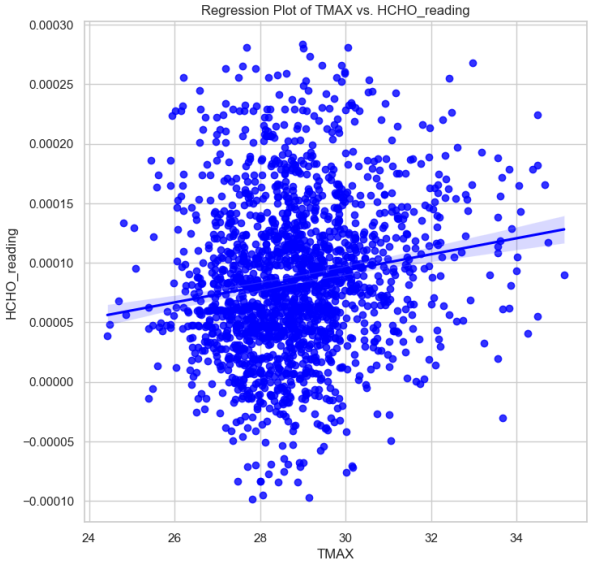
Index	Correlation	Ranking
ozone_average	0.388456	1
TAVG	0.337230	2
TMAX	0.317238	3
TMIN	0.259656	4
carbonmonoxide_average	-0.254780	5
nitrogendioxide_average	0.120014	6
total	0.027654	7
PRCP	-0.019255	8
Lockdown	-0.006427	9
Population_density	0.017186	10
Population	0.017186	11
new	0.008424	12

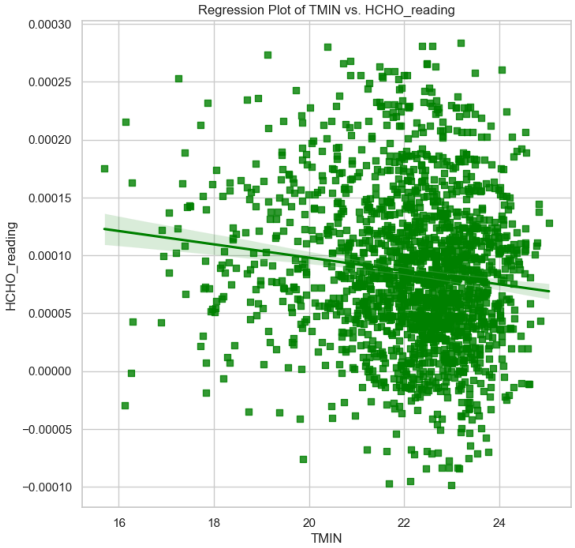
3. Deniyaya Matara

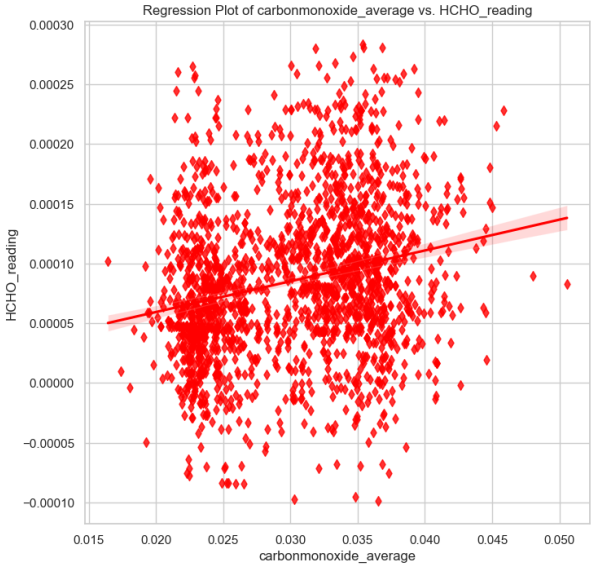


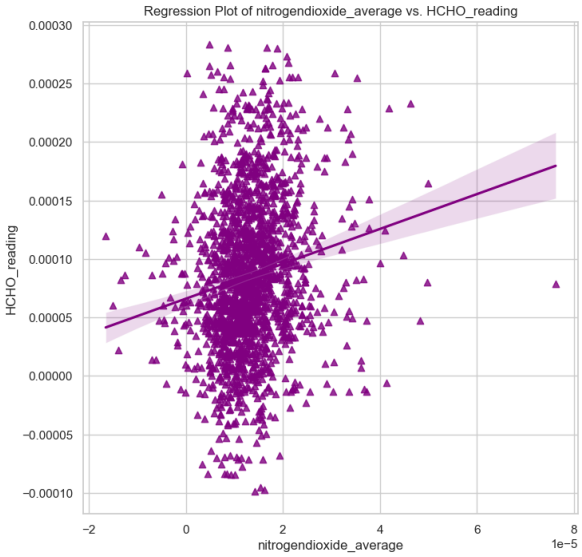

Correlation with HCHO_reading
Population -0.06
Population_density -0.06
new -0.10
total -0.05
Lockdown -0.04
carbonmonoxide_average 0.23
nitrogendioxide_average 0.16
ozone_average -0.12
PRCP -0.02
TAVG 0.05
TMAX 0.16
TMIN -0.13
HCHO_reading 1.00
index
correlation
TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
Population
Population_density
total
new

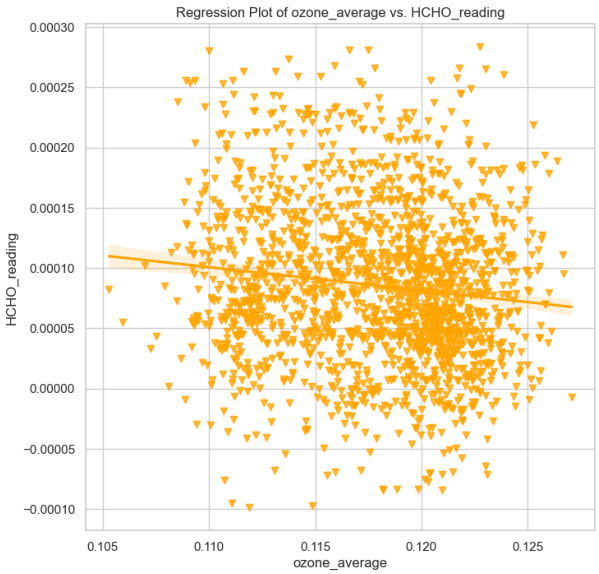
Regression Plots

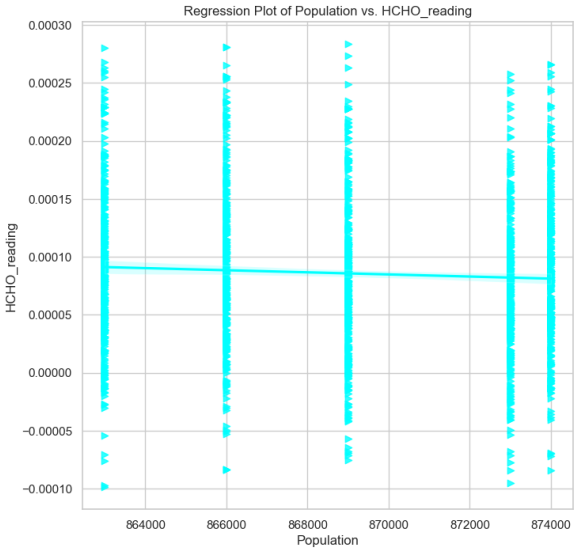


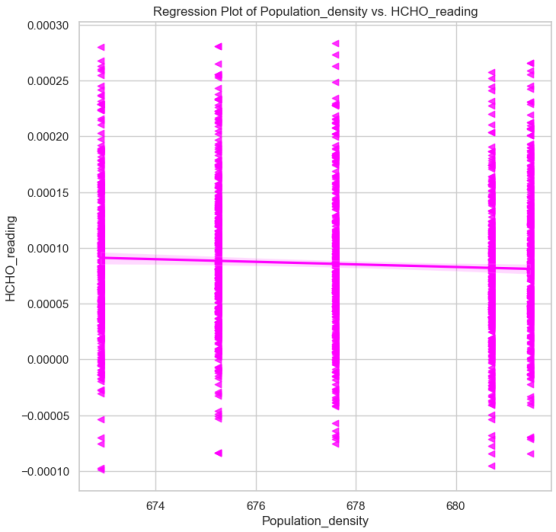


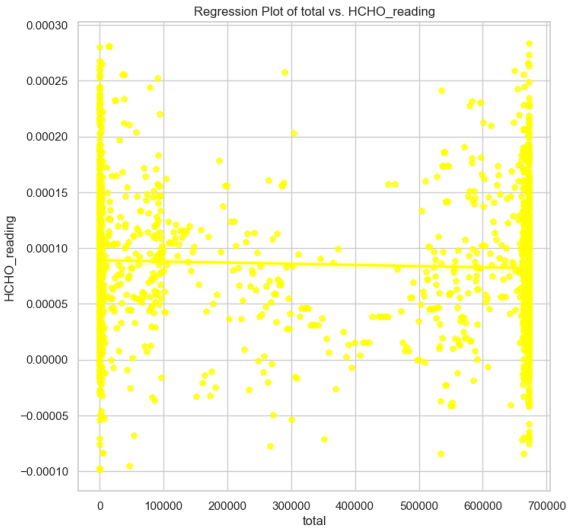


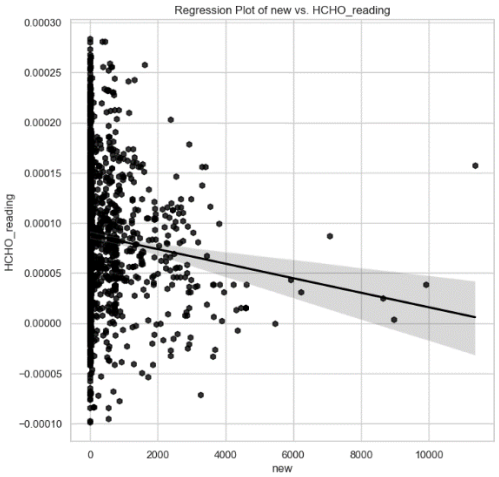






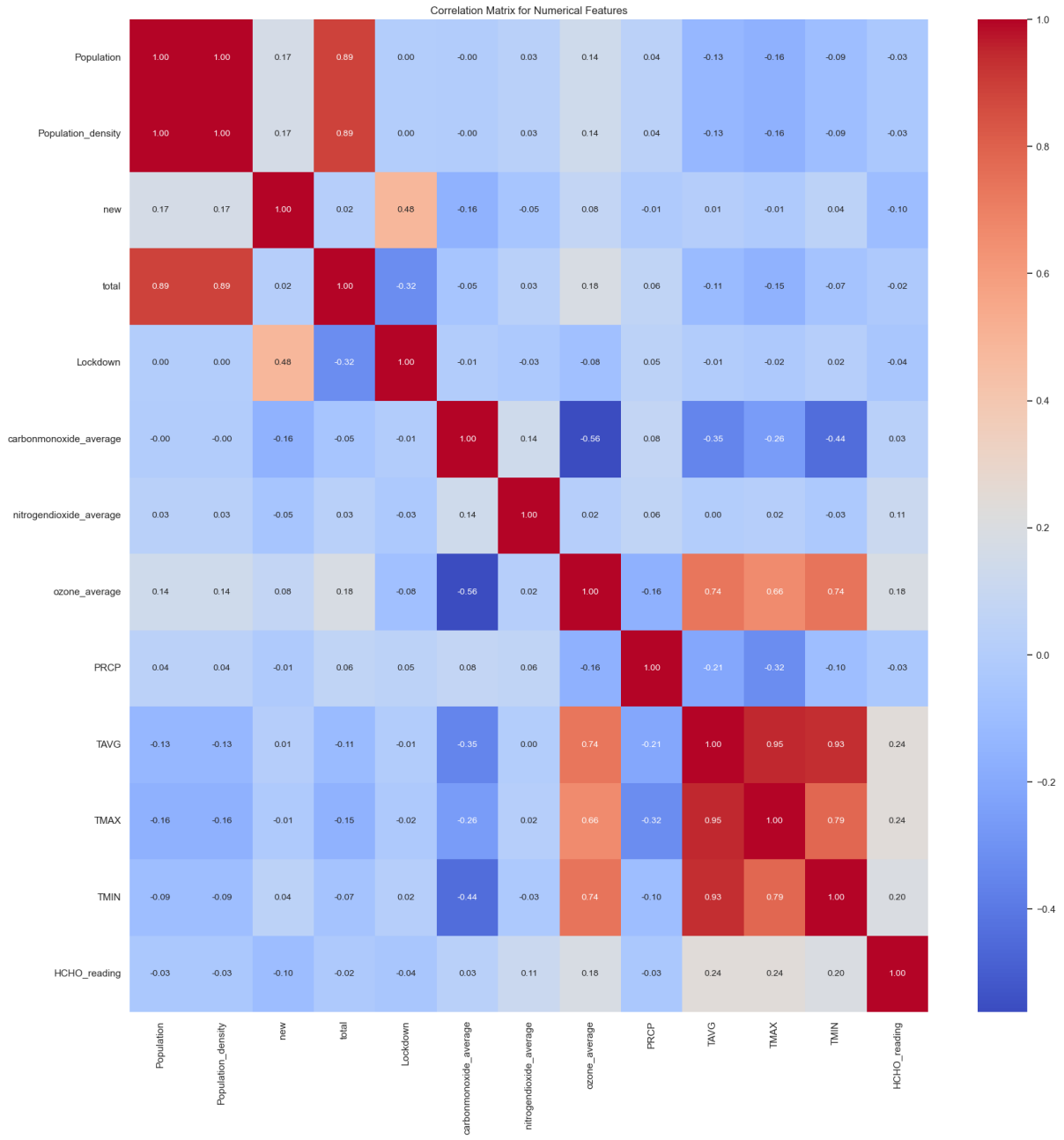


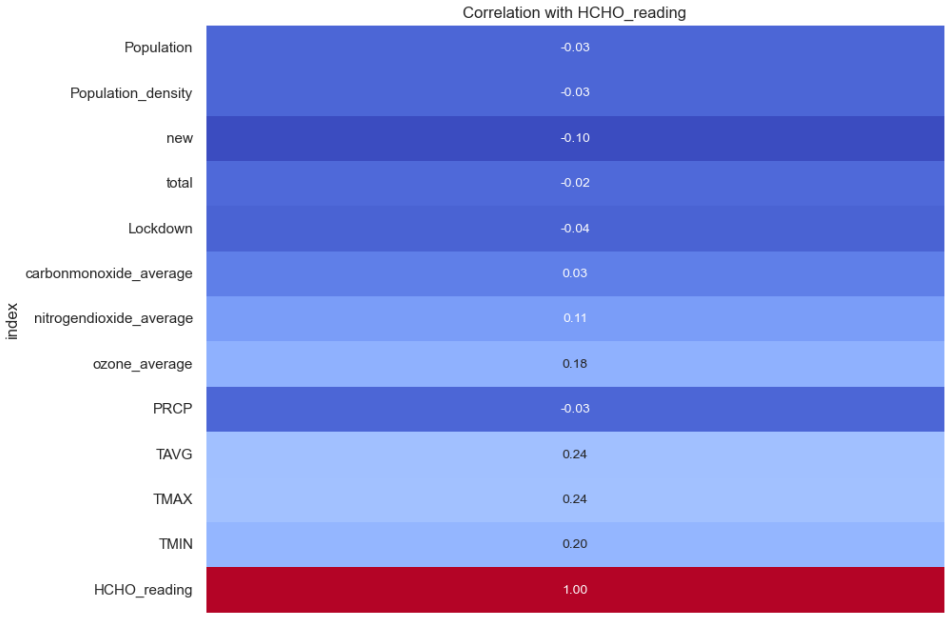


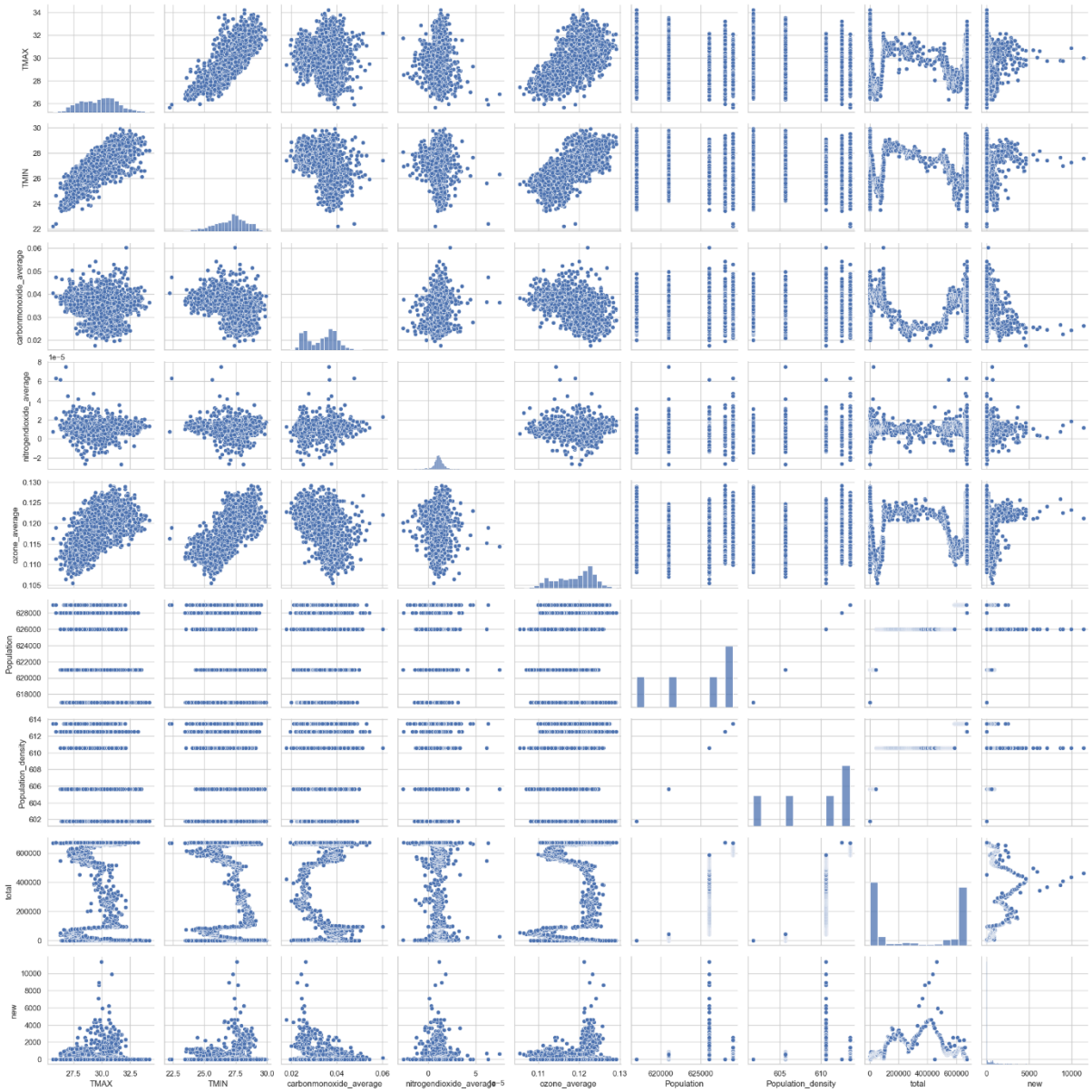


Index	Correlation	Ranking
carbonmonoxide_average	0.233251	1
nitrogendioxide_average	0.159415	2
TMAX	0.156737	3
TAVG	0.050505	4
Population	-0.057003	5
Population_density	-0.057003	6
total	-0.048787	7
Lockdown	-0.037158	8
ozone_average	-0.122280	9
TMIN	-0.127243	10
PRCP	-0.021408	11
new	-0.097509	12

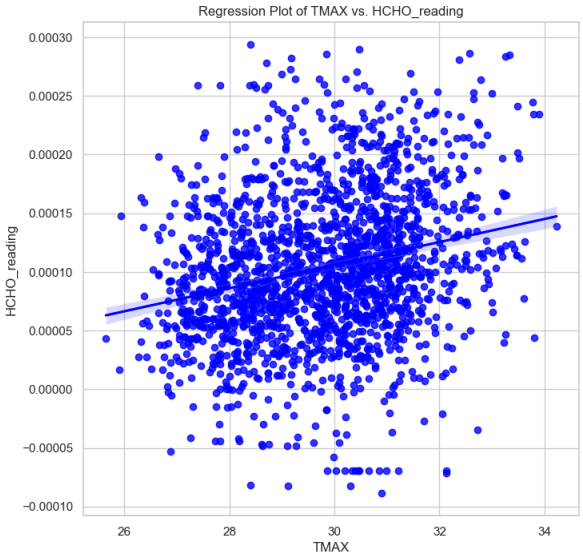
4. Jaffna

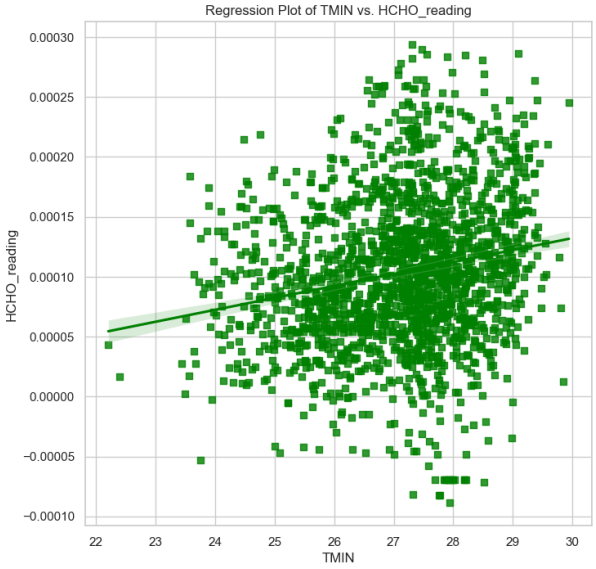

Correlation Matrix for Numerical Features
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

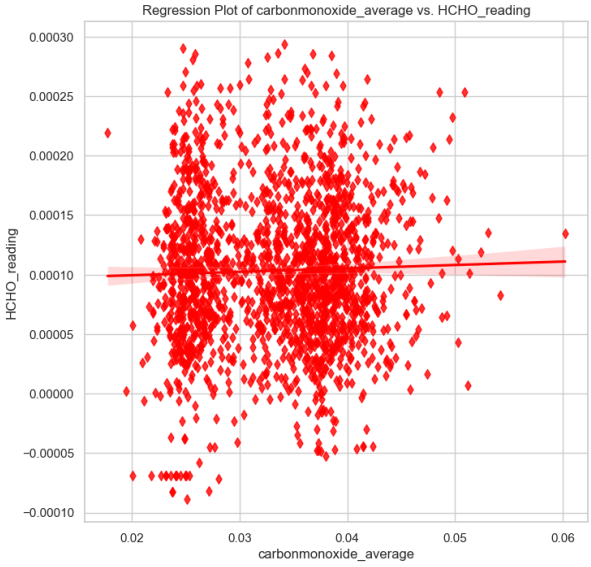

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
index
-0.03
-0.03
-0.10
-0.02
-0.04
0.03
0.11
0.18
-0.03
0.24
0.24
0.20
1.00
correlation

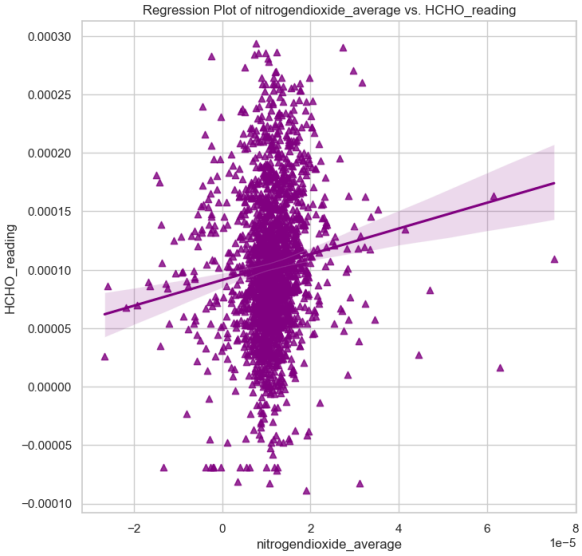

TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
Population
Population_density
total
new

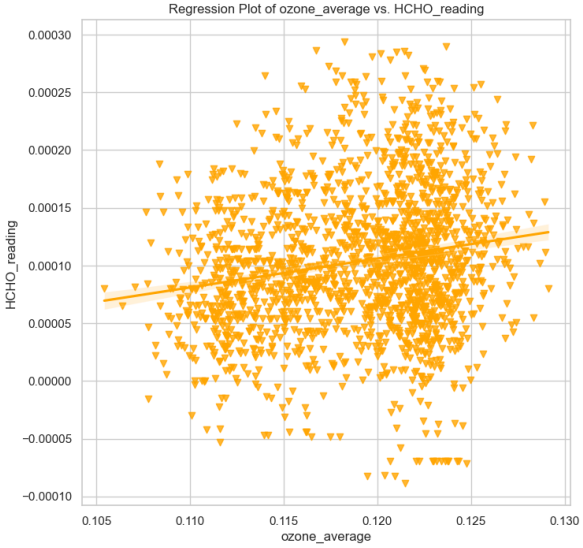
Regression Plots

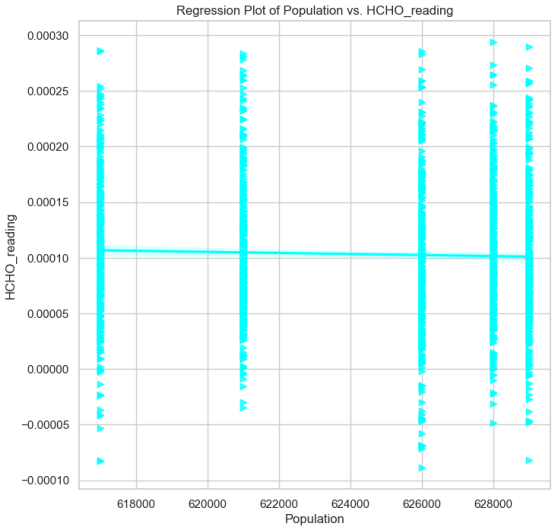


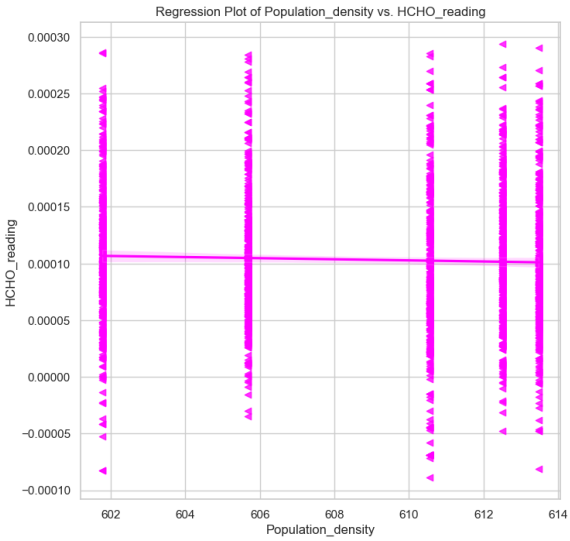


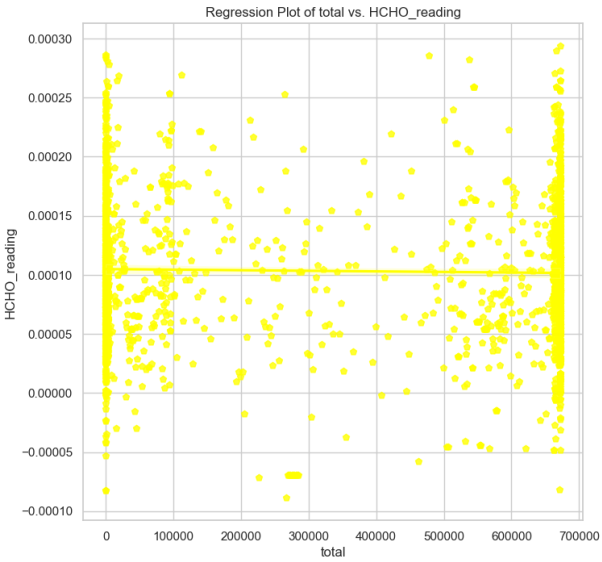


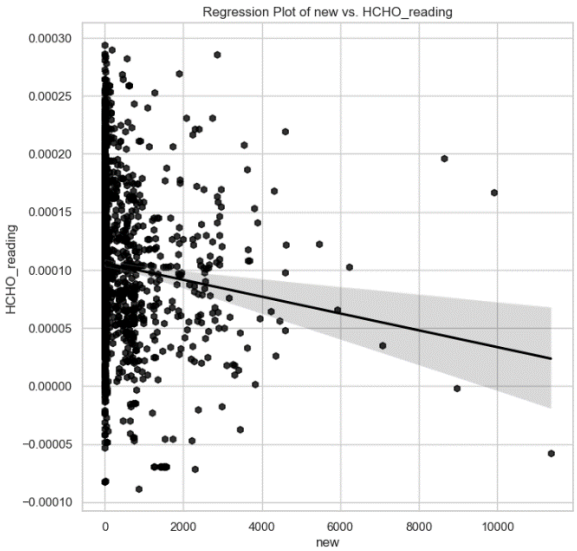






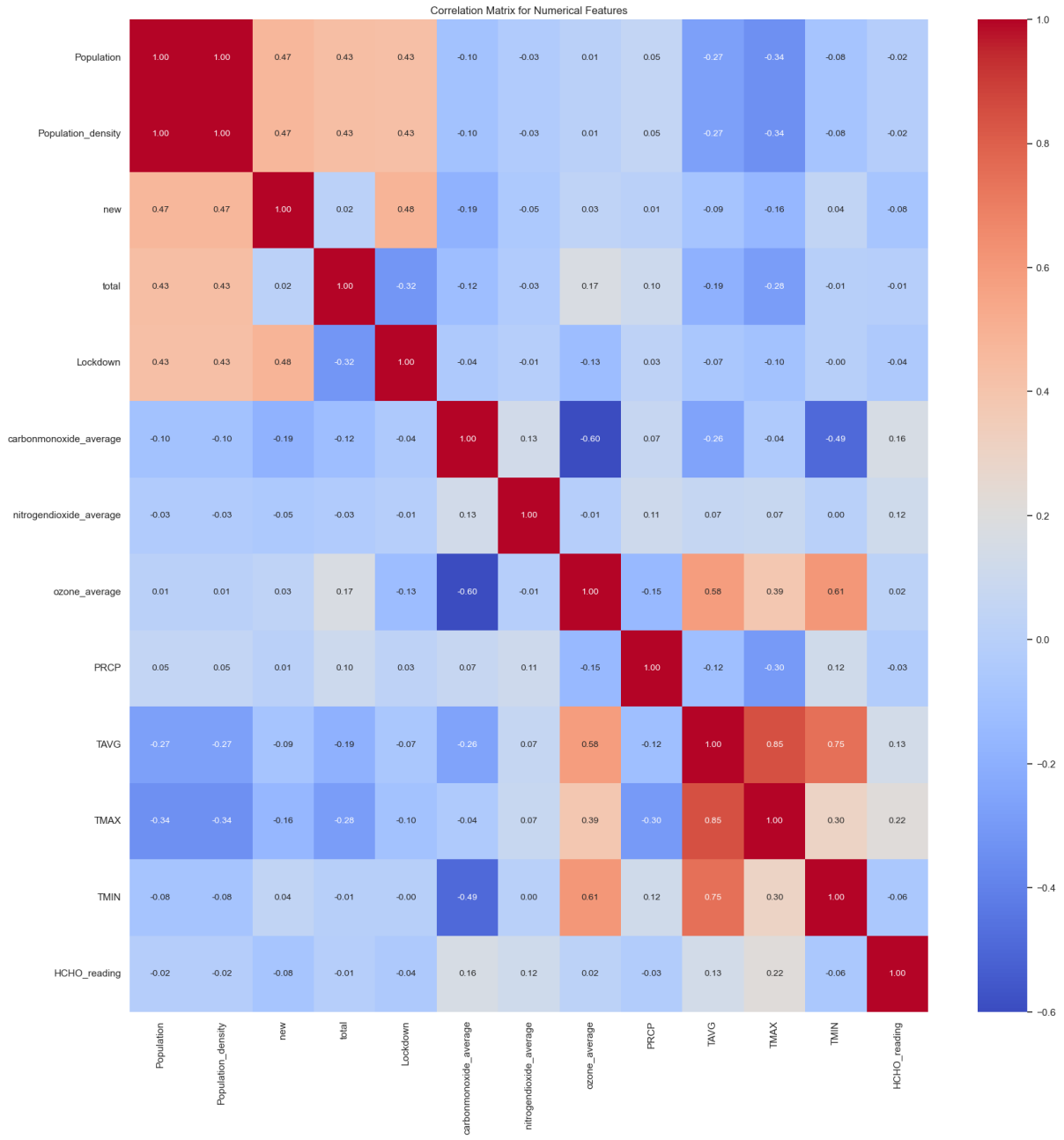


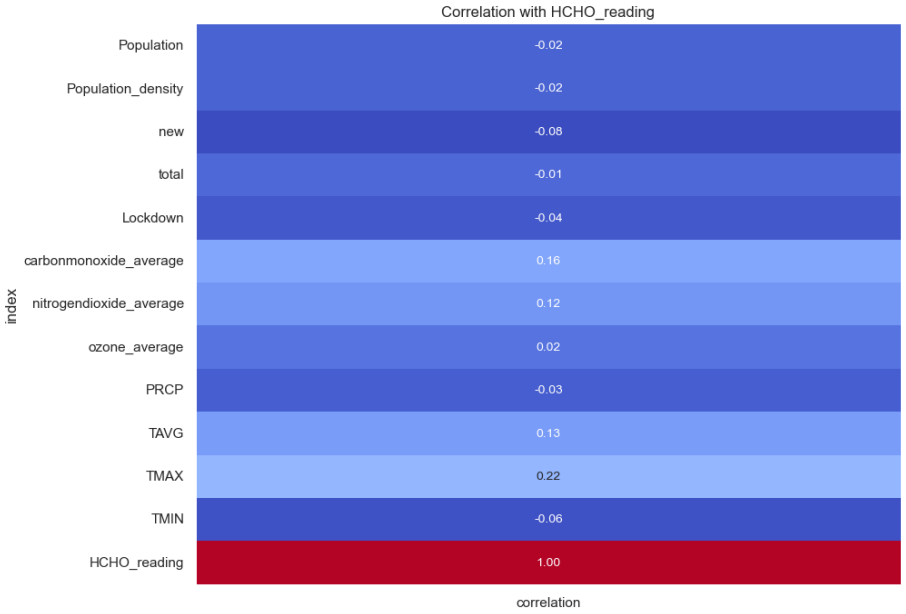


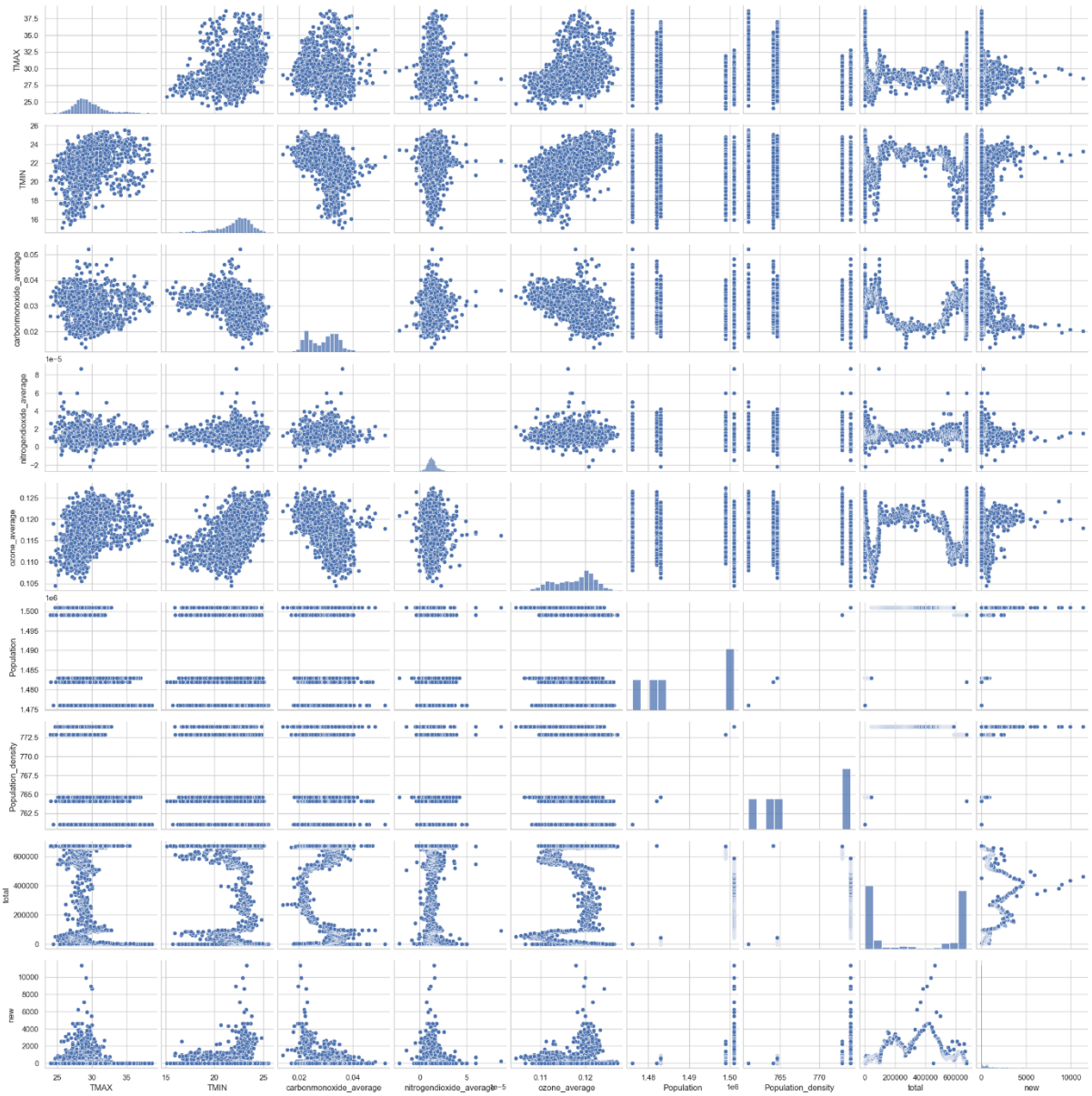


Index	Correlation	Ranking
TMAX	0.241938	1
TAVG	0.238013	2
TMIN	0.199191	3
ozone_average	0.182390	4
nitrogendioxide_average	0.114425	5
PRCP	-0.033560	6
Population	-0.033669	7
Population_density	-0.033669	8
Lockdown	-0.044732	9
total	-0.023844	10
carbonmonoxide_average	0.029595	11
new	-0.101108	12

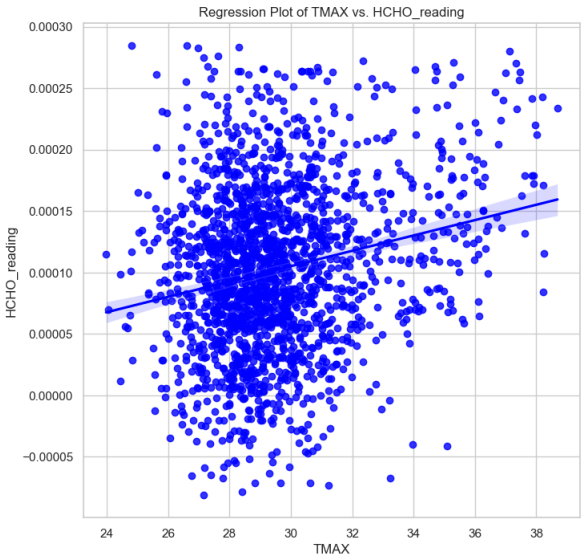
5. Kandy

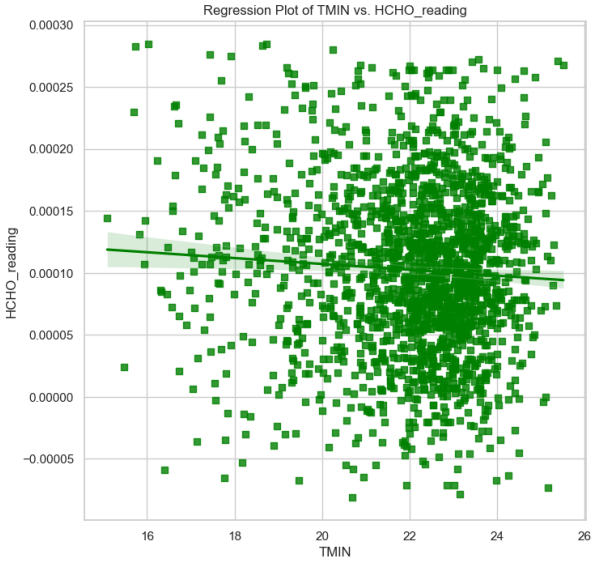

Correlation Matrix for Numerical Features
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

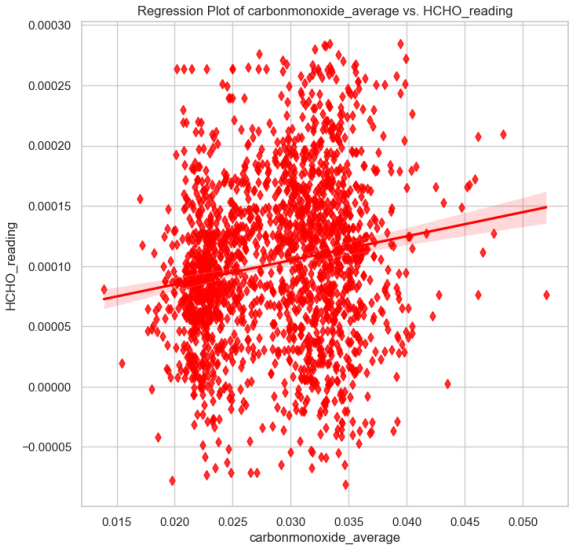

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
-0.02
-0.02
-0.08
-0.01
-0.04
0.16
0.12
0.02
-0.03
0.13
0.22
-0.06
1.00
index
correlation

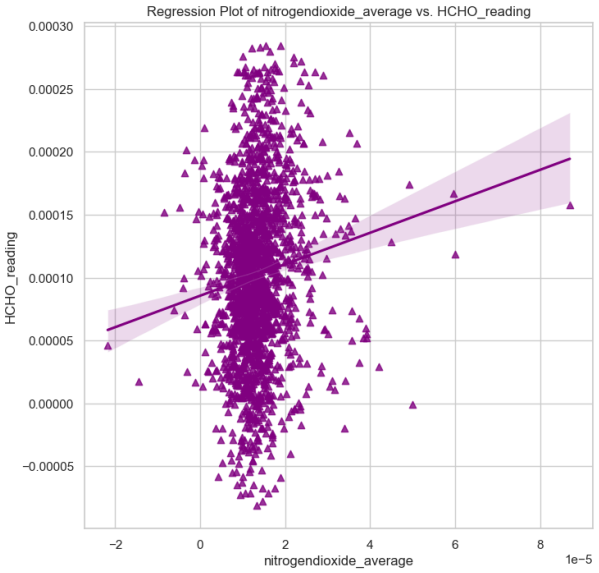

TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
Population
Population_density
total
new

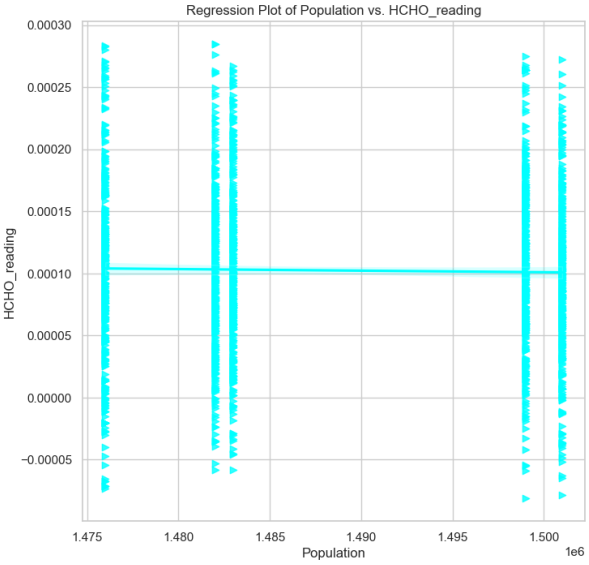
Regression Plots

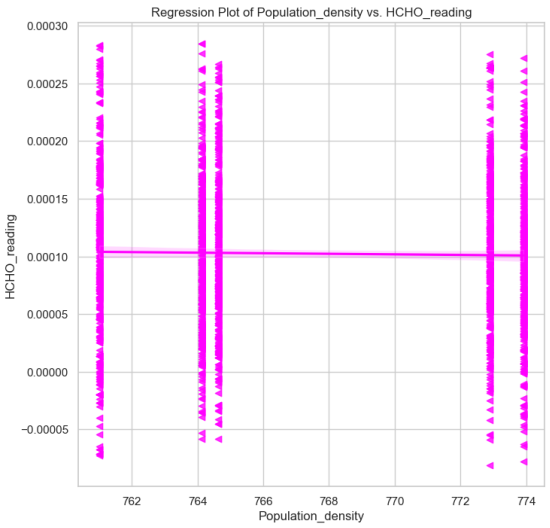


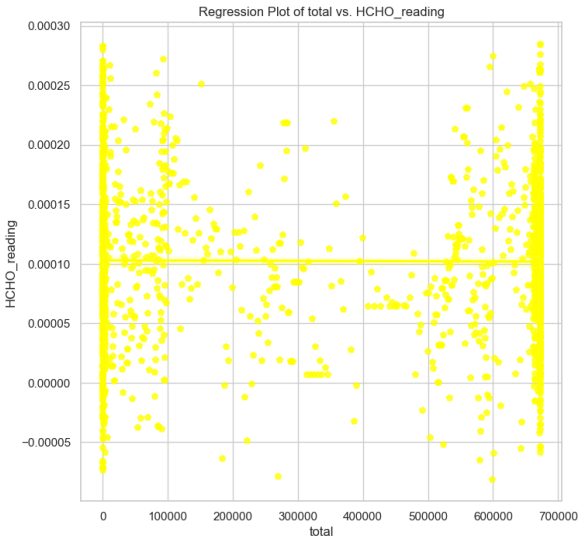


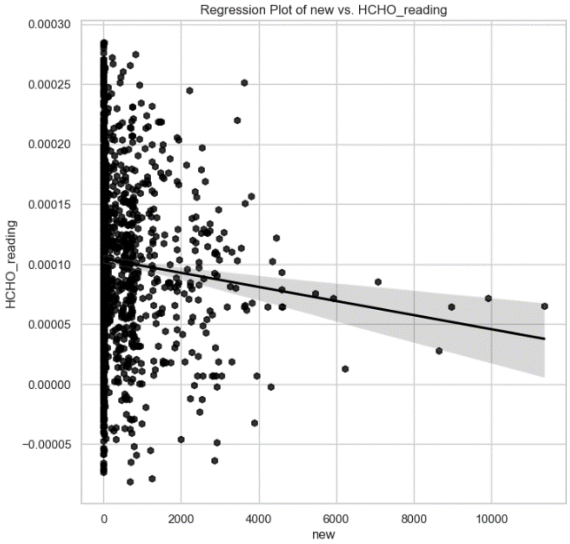






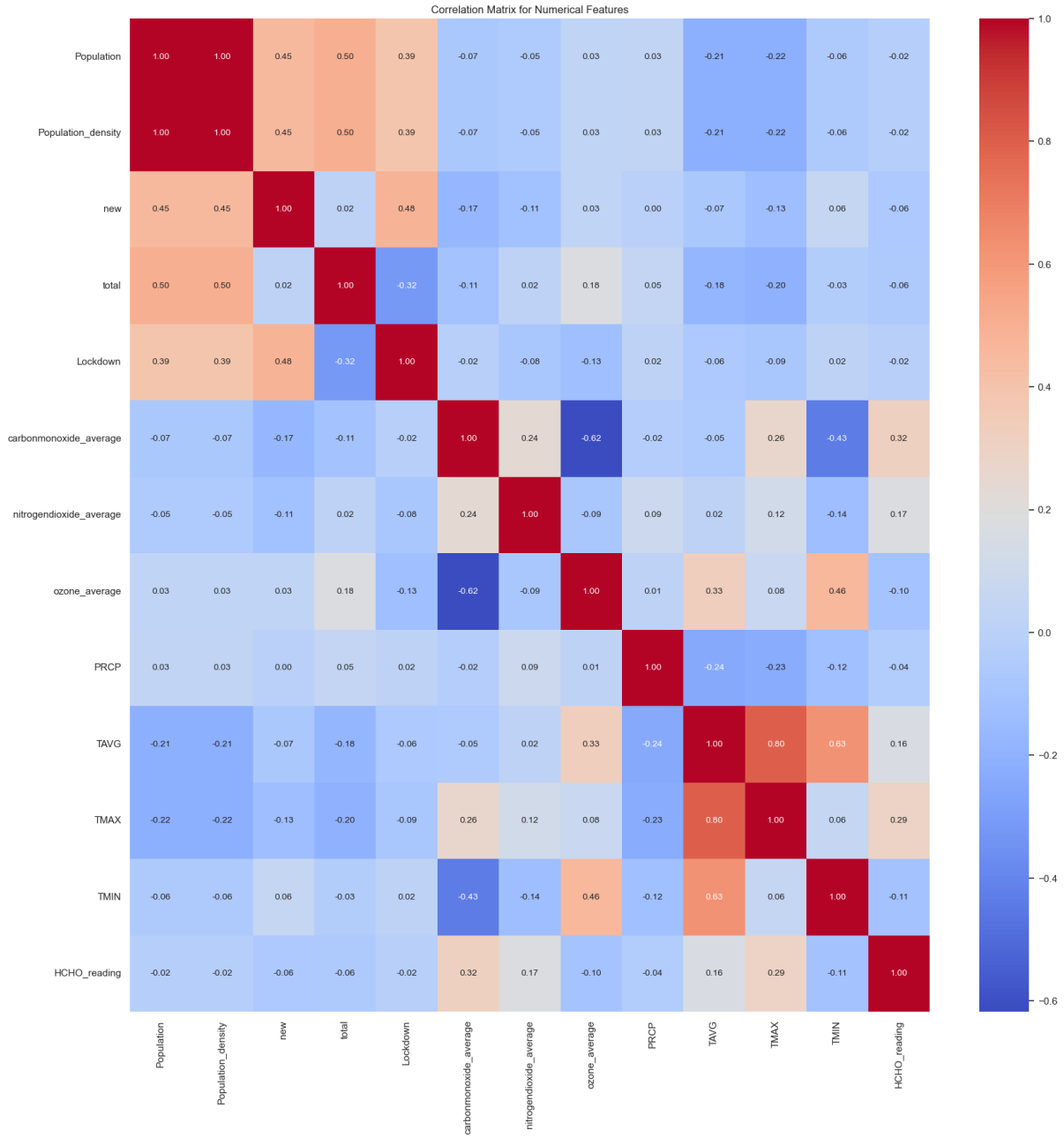


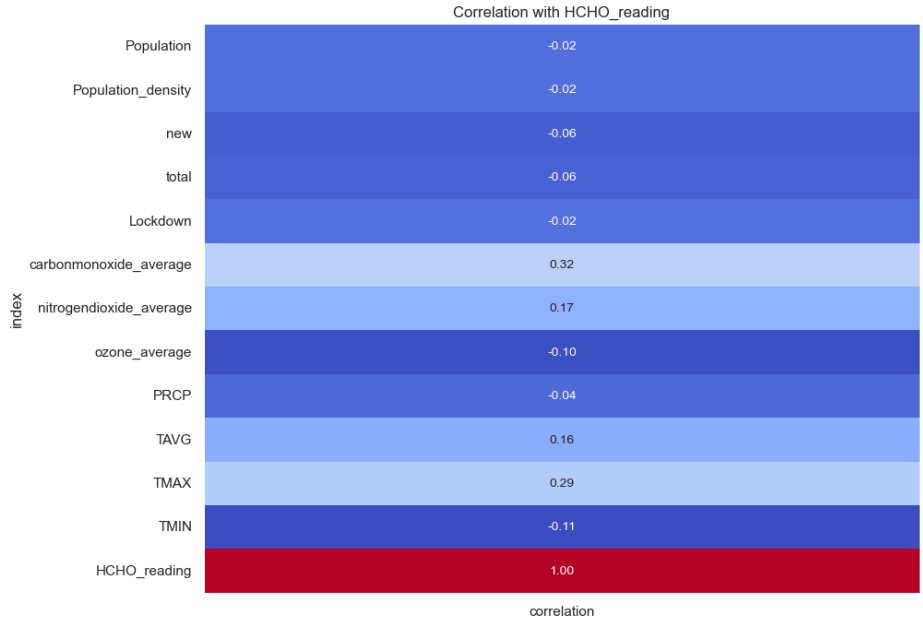


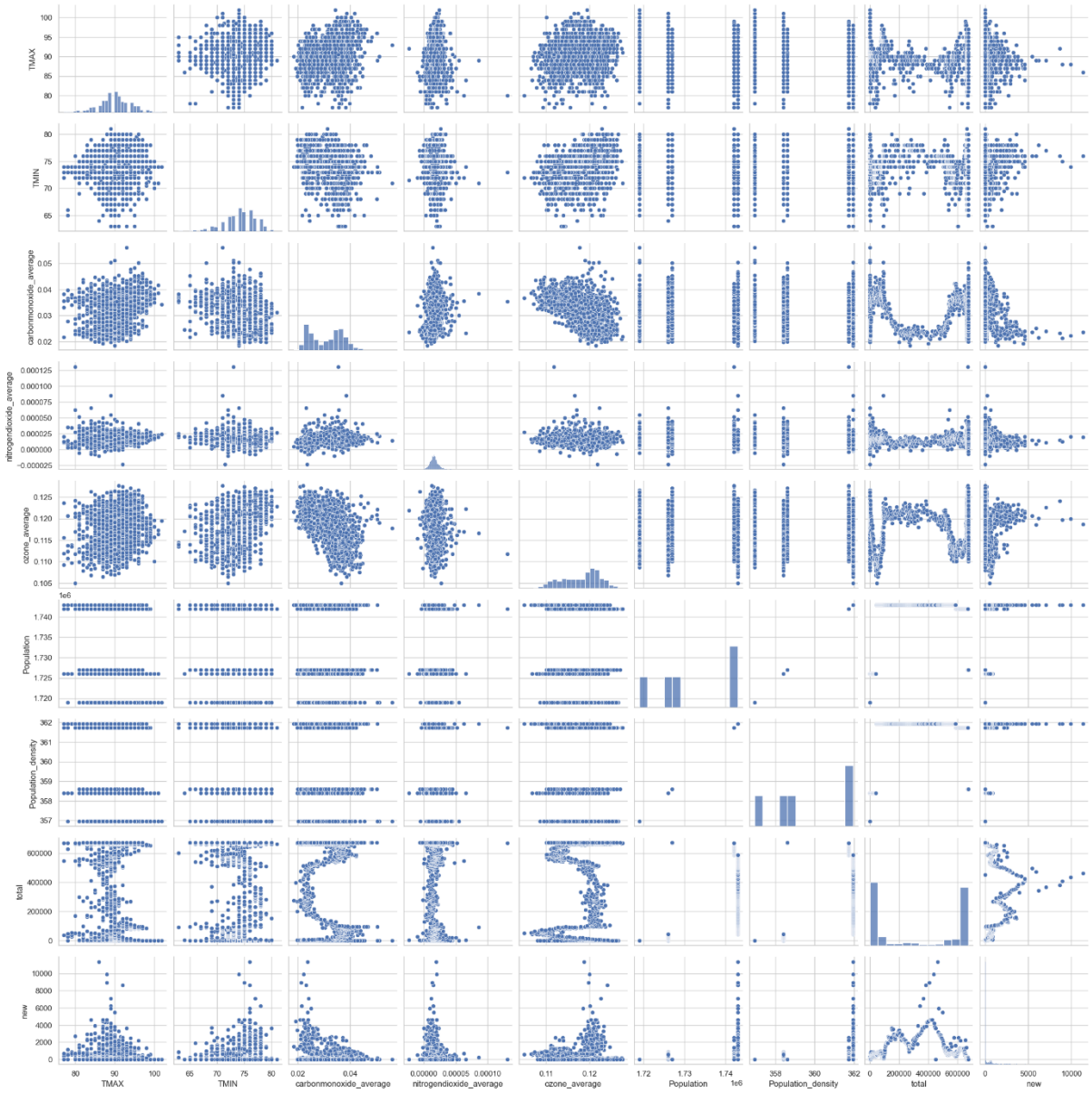


Index	Correlation	Ranking
TMAX	0.215312	1
carbonmonoxide_average	0.162006	2
TAVG	0.131822	3
nitrogendioxide_average	0.117265	4
ozone_average	0.022921	5
Population	-0.018224	6
Population_density	-0.018224	7
PRCP	-0.030485	8
TMIN	-0.060618	9
Lockdown	-0.044136	10
total	-0.007192	11
new	-0.076073	12

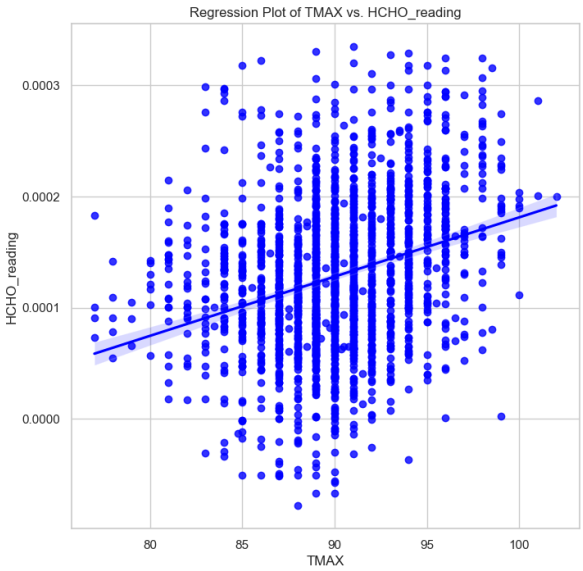
6. Kurunegala

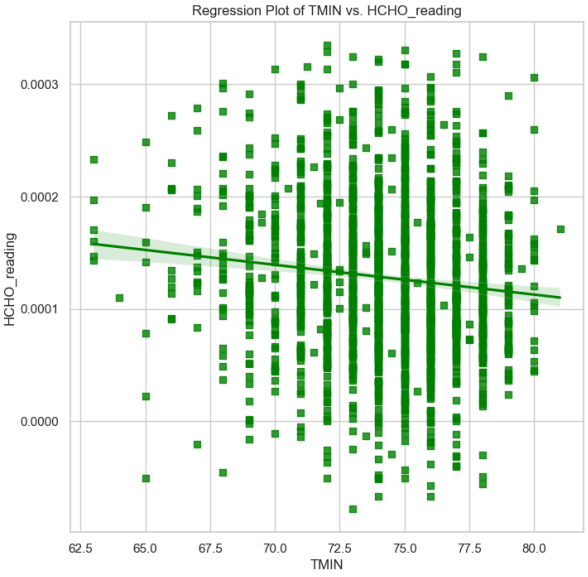


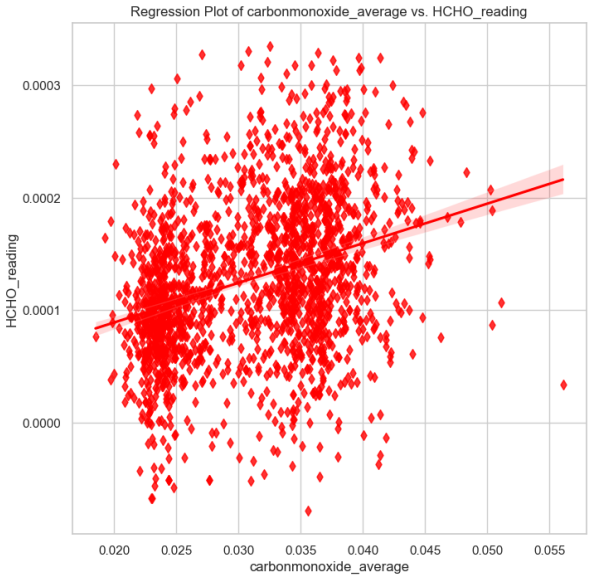

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
-0.02
-0.02
-0.06
-0.06
-0.02
0.32
0.17
-0.10
-0.04
0.16
0.29
-0.11
1.00
index
correlation

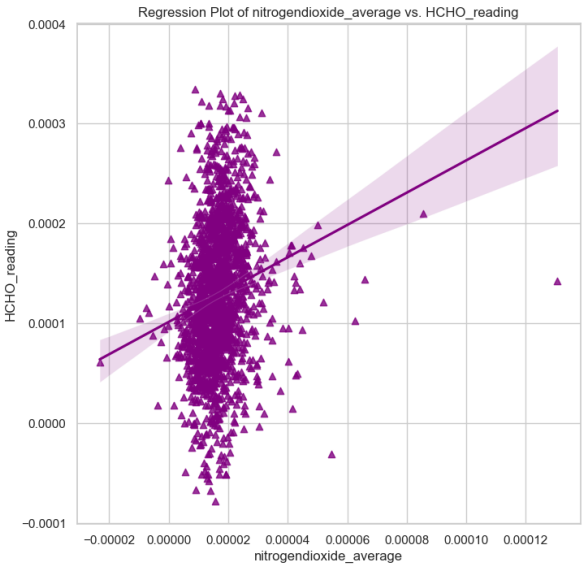

TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
Population
Population_density
total
new

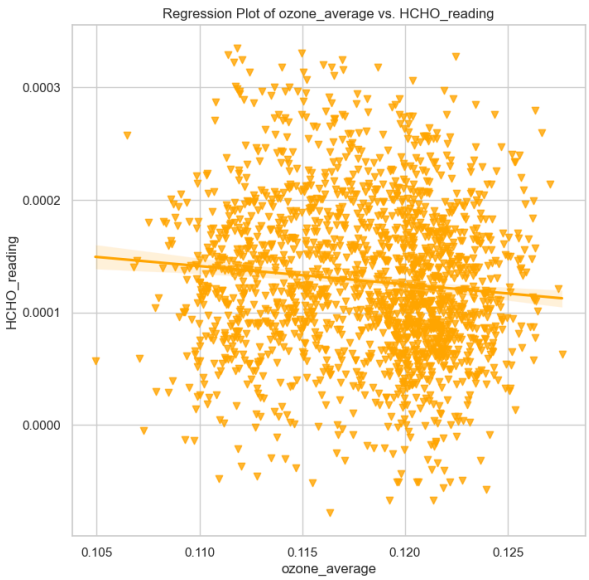
Regression Plots

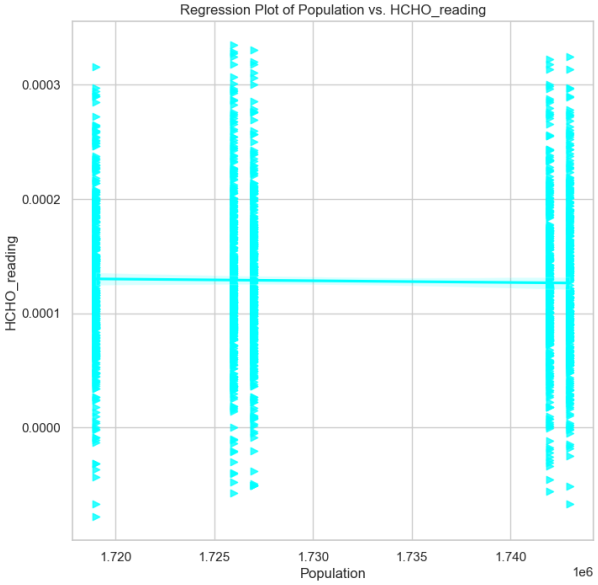


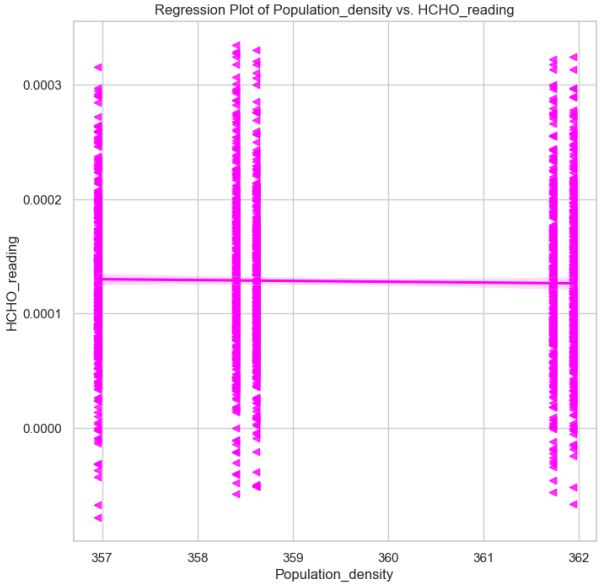


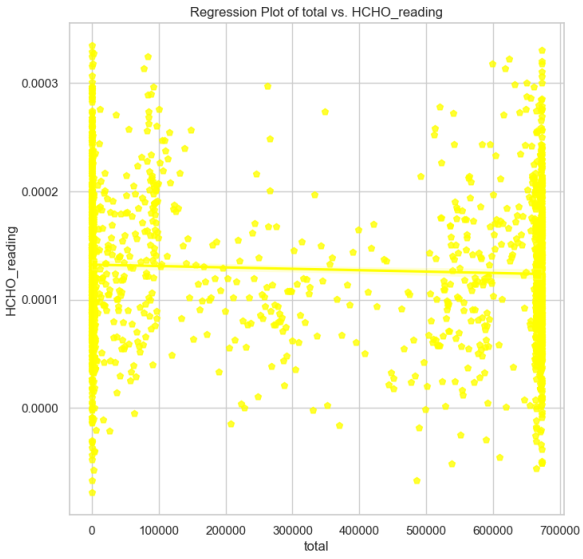


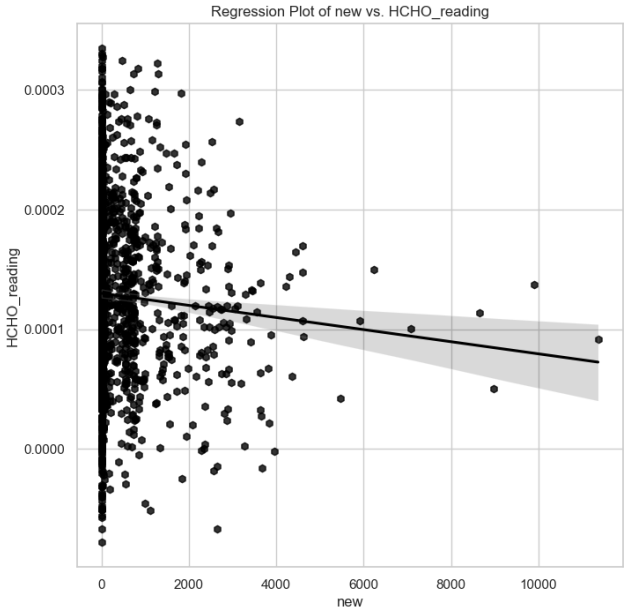






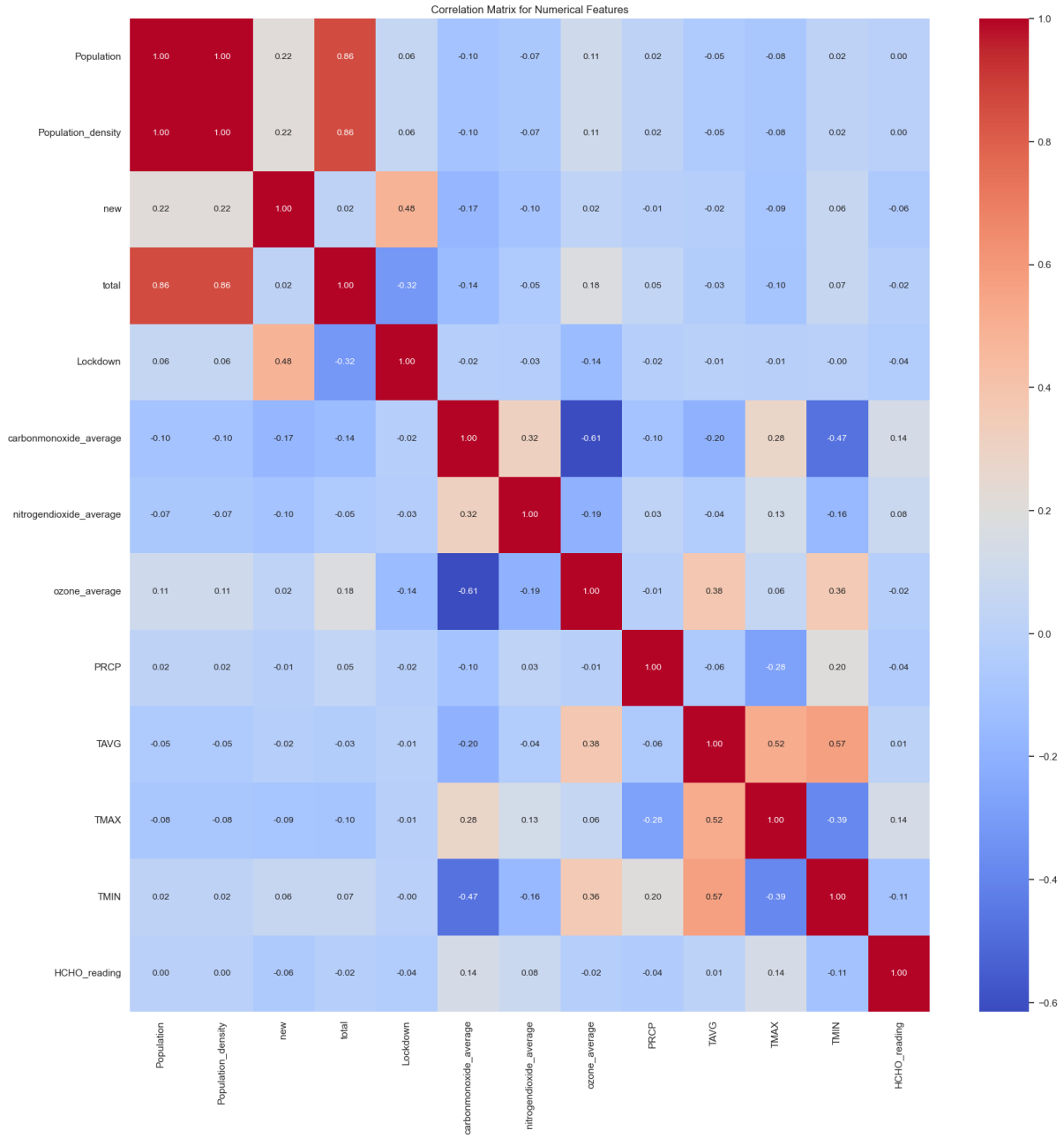


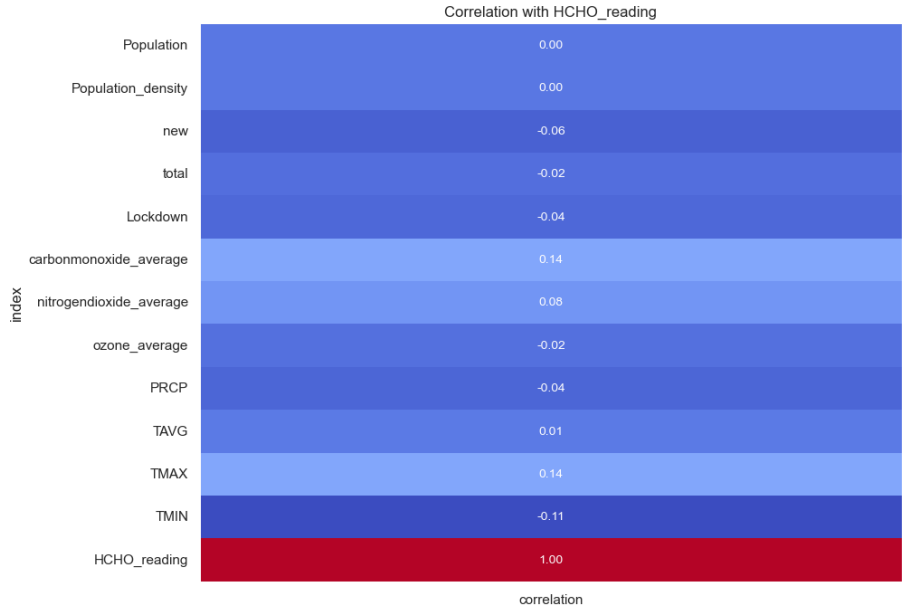


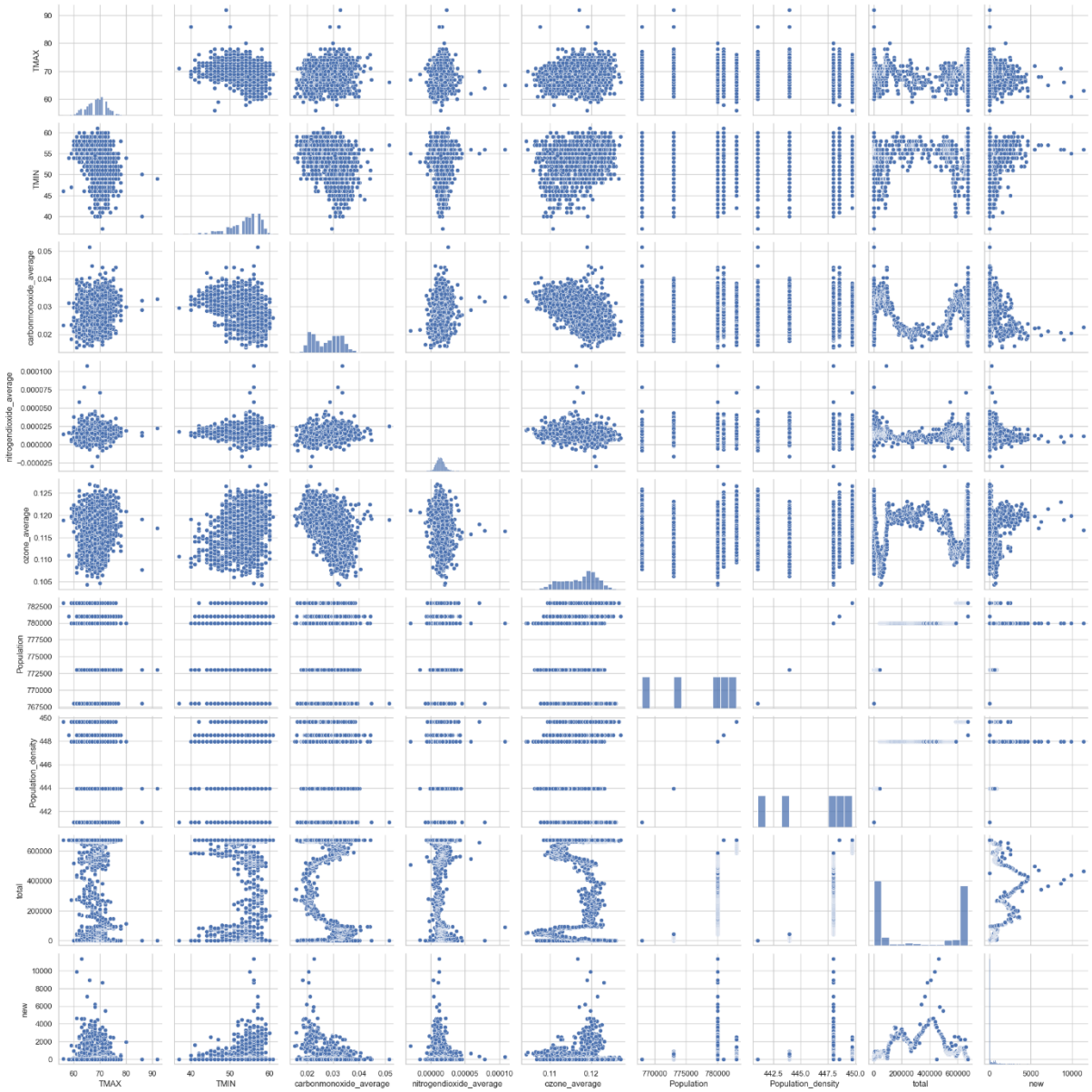


Index	Correlation	Ranking
carbonmonoxide_average	0.315851	1
TMAX	0.293148	2
TAVG	0.162282	3
nitrogendioxide_average	0.173389	4
ozone_average	-0.099745	5
TMIN	-0.109659	6
PRCP	-0.038023	7
Population	-0.020168	8
Population_density	-0.020168	9
Lockdown	-0.016016	10
total	-0.056526	11
new	-0.064005	12

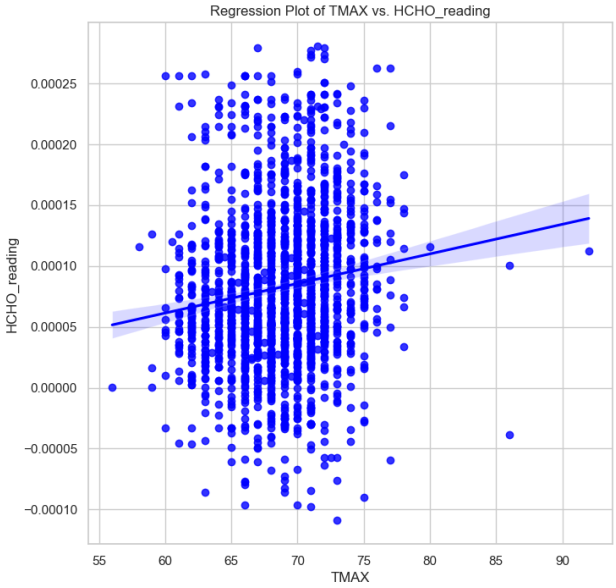
7. Nuwara Eliya

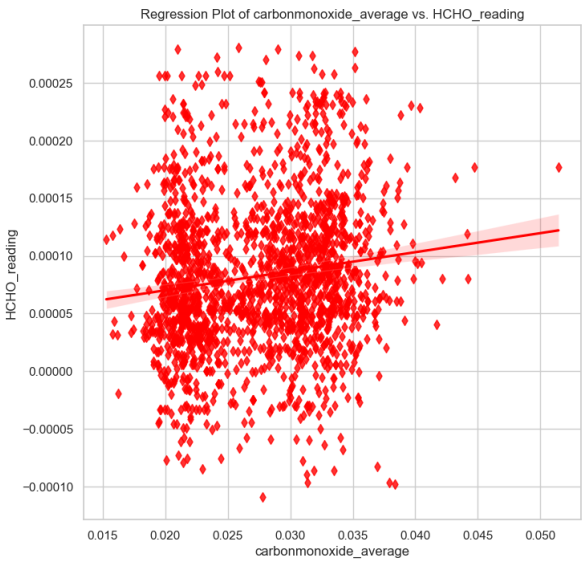


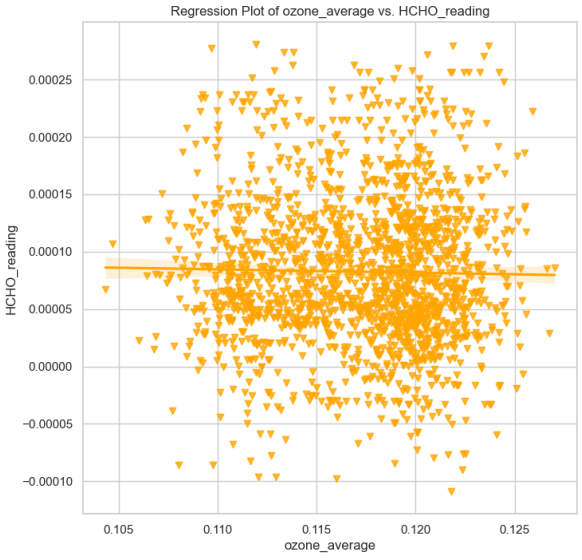

Correlation with HCHO_reading
Population
Population_density
new
total
Lockdown
carbonmonoxide_average
nitrogendioxide_average
ozone_average
PRCP
TAVG
TMAX
TMIN
HCHO_reading
index
0.00
0.00
-0.06
-0.02
-0.04
0.14
0.08
-0.02
-0.04
0.01
0.14
-0.11
1.00
correlation

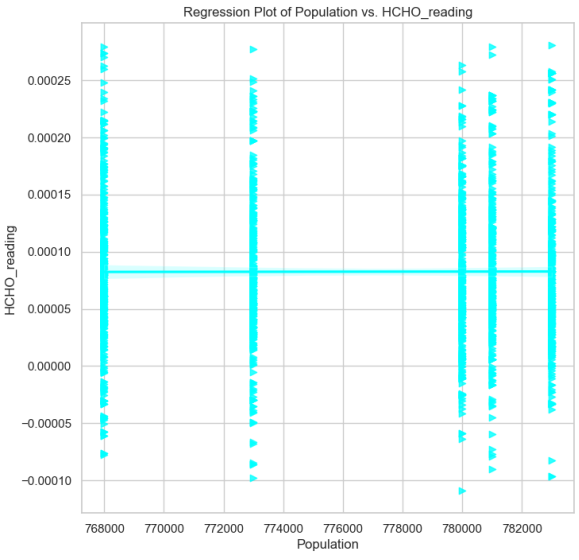

TMAX
TMIN
carbonmonoxide_average
nitrogendioxide_average
ozone_average
Population
Population_density
total
new

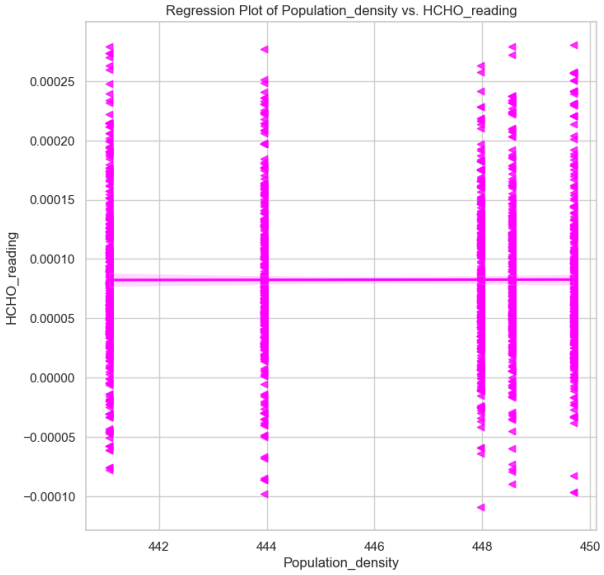
Regression plots

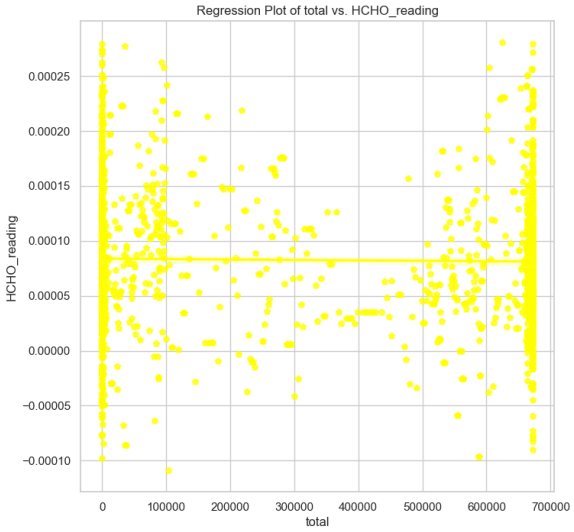


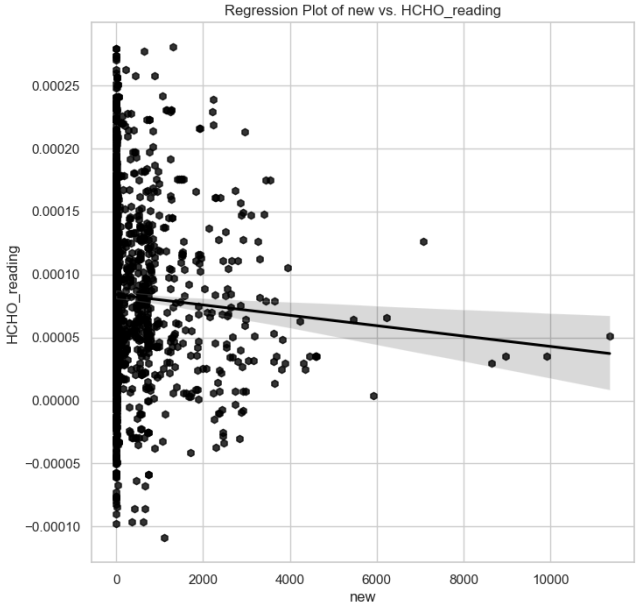






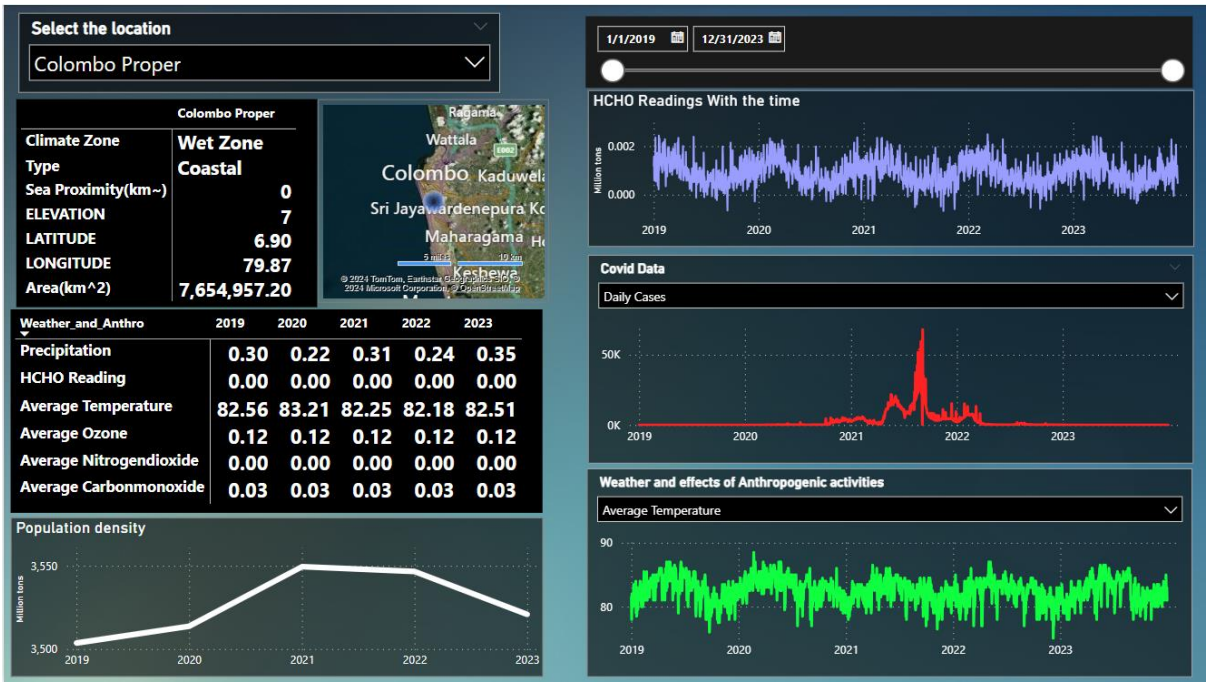






Index	Correlation	Ranking
carbonmonoxide_average	0.136302	1
TMAX	0.136188	2
nitrogendioxide_average	0.083733	3
TAVG	0.009741	4
Population	0.002117	5
Population_density	0.002117	6
ozone_average	-0.018272	7
total	-0.018862	8
Lockdown	-0.037034	9
PRCP	-0.042820	10
new	-0.055347	11
TMIN	-0.109526	12

4 Power BI dashboard

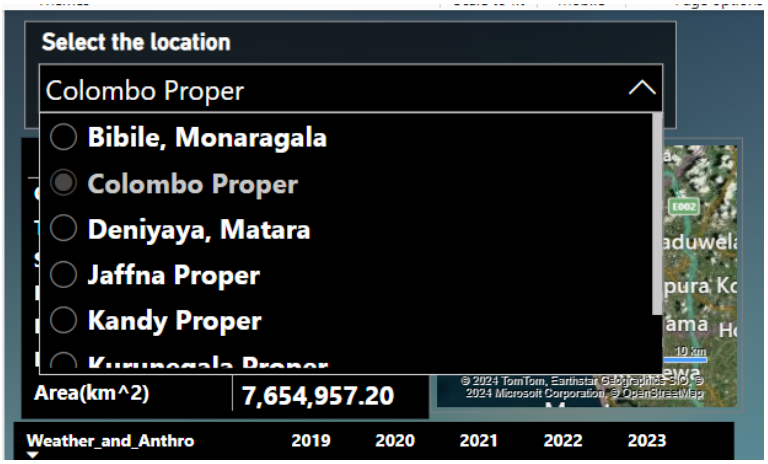


After preparing necessary and doing the spatial temporal analysis using pandas, a power BI dashboard was made for easy data exploration for the people referring to this research in future.

The initial data transformations needed to dashboard implementation was done using pandas and later the other transformations were done using power BI itself.

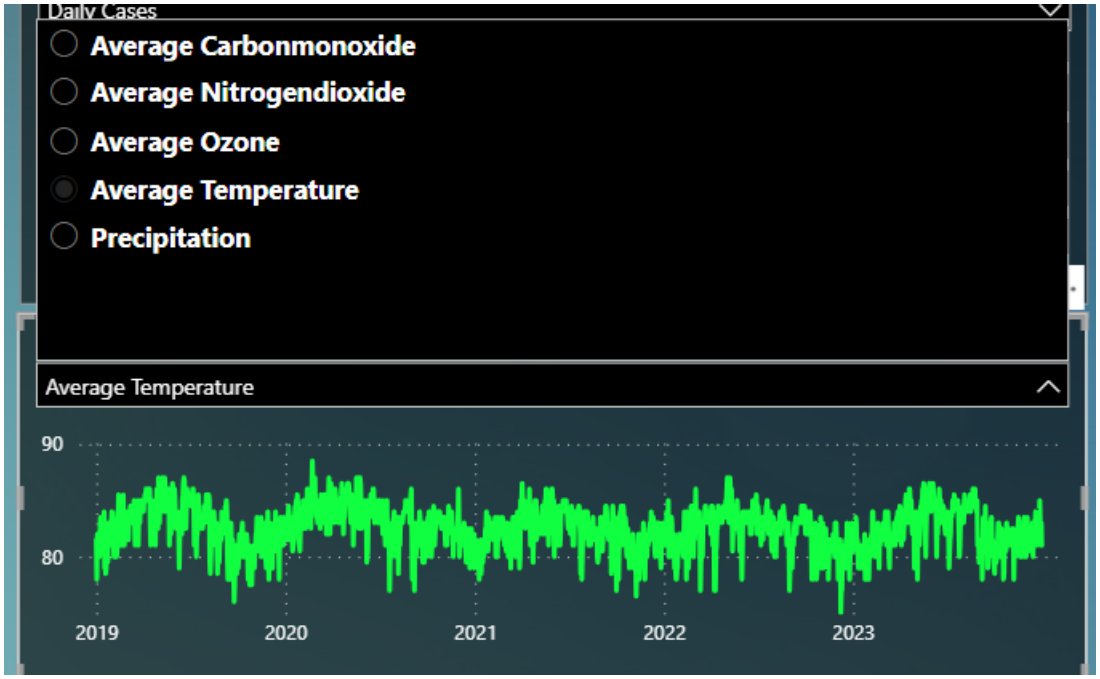
After carful handling of slicers time series plots were implemented to visualize HCHO_readings, weather and effects of anthropogenic activities, covid and demographic data. Also tables and matrixes were also implemented to visualize city specific attributes and averages of numerical data. Also a time slicer was implemented so the users can adjust the plots to necessary time periods for enhanced analysation of variations of data.

City slicer-

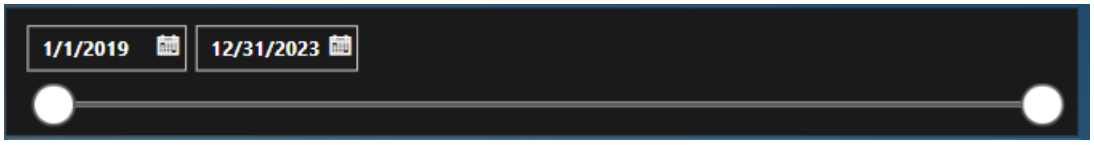


This is the top most slicer where users have to select the relevant city and it will filter out the data relevant to that city to be displayed in other plots and tables.

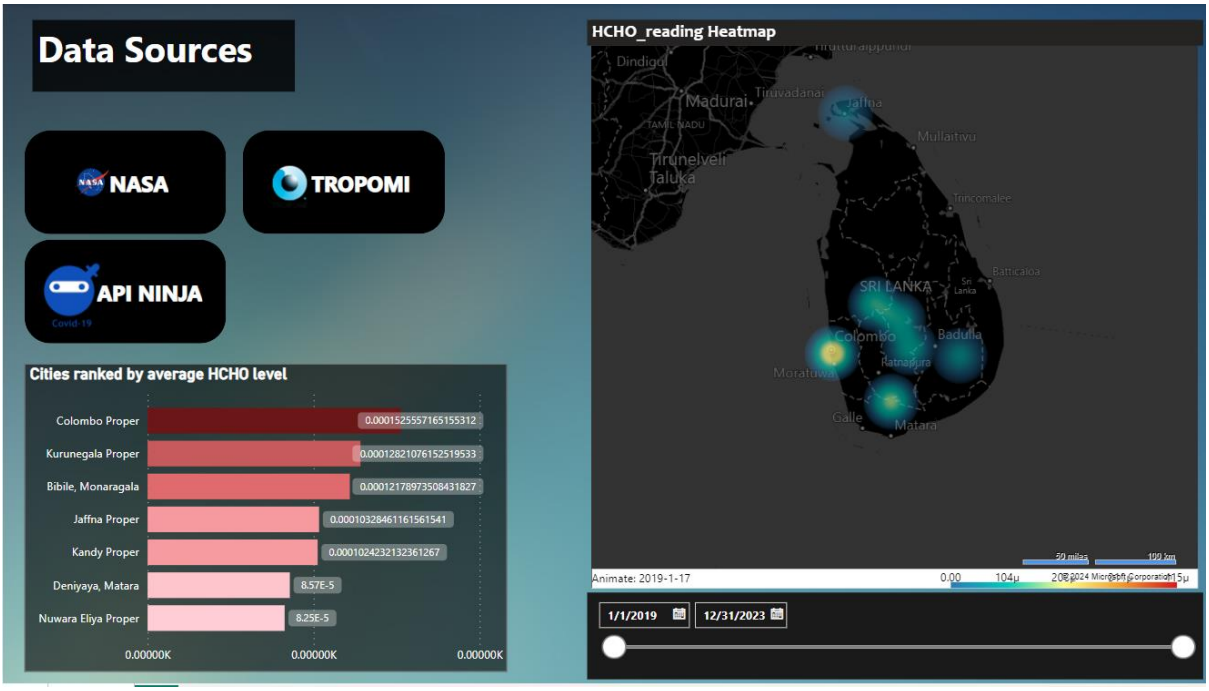
Time series plots slicers-



Time Slicers-



Also a interactive heatmap was also implemented to visualize the variations of HCHO readings across given cities through the time. A bar chart was implemented where the sum of HCHO emissions of each city is ranked. Also buttons linked to the data sources were implemented, so users can directly check credibility of the data used for the research.


Data Sources
NASA
TROPOMI
API NINJA
Cities ranked by average HCHO level
Colombo Proper
0.00015255571651553 12
Kurunegala Proper
0.00012821076152519533
Bibile, Monaragala
0.00012178973508431827
Jaffna Proper
0.00010328461161561541
Kandy Proper
0.00010242321323612 67
Deniyaya, Matara
8.57E-5
Nuwara Eliya Proper
8.25E-5
0.00000K
0.00000K
0.00000K
HCHO_reading Heatmap
Dindigul
Madurai
Tiruvadanai
Jaffna
TAMIL NADU
Mullaitivu
Tirunelveli Taluka
Trincomalee
Batticaloa
SRI LANKA
Sri Lanka
Colombo
Badulla
Ratnapura
Moratuwa
Galle
Matara
Animate: 2019-1-17
0.00
104µ
1/1/2019
12/31/2023

5 Machine learning models implementation

Models were implemented under 4 stages.

1. Single variate SARIMA models
2. Single variate facebook prophet models
3. Single variate LSTMs
4. Multivariate LSTMs

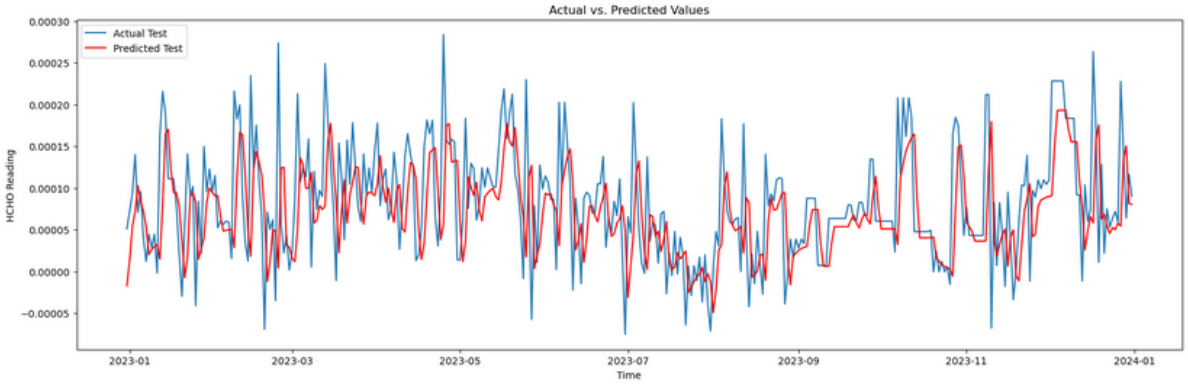
5.1 Univariate SARIMA models

SARIMA, or Seasonal Autoregressive Integrated Moving Average, extends the ARIMA model to capture seasonality. It comprises three components: AR (Autoregressive), I (Integrated), and MA (Moving Average). In the Matara implementation, SARIMA forecasts formaldehyde (HCHO) readings using historical data. After optimizing parameters with auto_arima, the model predicts test data. Evaluation metrics include MSE, R-squared Score, and MAE. Visualization aids in assessing accuracy by comparing actual and predicted values.

Like this for all the cities sperate models will be implemented.

```
# Load dataset
data = pd.read_csv(r"..\processed data\temp\matara_fomaldehyde.csv")
data.drop(columns=['Location', 'Next_Date'], inplace=True)
data['Current_Date'] = pd.to_datetime(data['Current_Date'])
data.set_index('Current_Date', inplace=True)

train_data, test_data = train_test_split(data, test_size=0.2, shuffle=False)

# Determine best parameters using auto_arima
auto_model = auto_arima(train_data['HCHO_reading'], seasonal=True, m=12,
                        stepwise=True, trace=True,
                        error_action='ignore', suppress_warnings=True)

# Extracting best parameters
best_order = auto_model.order
best_seasonal_order = auto_model.seasonal_order

print("Best SARIMA Order:", best_order)
print("Best Seasonal Order:", best_seasonal_order)

# Train SARIMA model with best parameters
model = SARIMAX(train_data['HCHO_reading'], order=best_order, seasonal_order=best_seasonal_order)
sarima_model = model.fit()

# Forecast
history = train_data['HCHO_reading'].values.tolist()
predictions = []
for t in range(len(test_data)):
    model = SARIMAX(history, order=best_order, seasonal_order=best_seasonal_order)
    model_fit = model.fit()
    output = model_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test_data['HCHO_reading'].iloc[t]
    history.append(obs)

# Evaluate the model
test_mse = mean_squared_error(test_data['HCHO_reading'], predictions)
test_r2 = r2_score(test_data['HCHO_reading'], predictions)
test_mae = mean_absolute_error(test_data['HCHO_reading'], predictions)

print(f'Test MSE: {test_mse}')
print(f'Test R2 Score: {test_r2}')
print(f'Test MAE: {test_mae}')

# Plotting
plt.figure(figsize=(20, 6))
plt.plot(test_data.index, test_data['HCHO_reading'], label='Actual Test')
plt.plot(test_data.index, predictions, color='red', label='Predicted Test')
plt.xlabel('Time')
plt.ylabel('HCHO Reading')
plt.title('Actual vs. Predicted Values')
plt.legend()
plt.show()
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[24] intercept   : AIC=inf, Time=5.73 sec
 ARIMA(0,0,0)(0,0,0)[24] intercept   : AIC=-23982.589, Time=0.16 sec
 ARIMA(1,0,0)(1,0,0)[24] intercept   : AIC=-23683.495, Time=3.64 sec
 ARIMA(0,0,1)(0,0,1)[24] intercept   : AIC=-24107.695, Time=2.84 sec
 ARIMA(0,0,0)(0,0,0)[24]             : AIC=-22512.992, Time=0.08 sec
 ARIMA(0,0,1)(0,0,0)[24] intercept   : AIC=-24109.223, Time=0.58 sec
 ARIMA(0,0,1)(1,0,0)[24] intercept   : AIC=-23623.724, Time=2.40 sec
 ARIMA(0,0,1)(1,0,1)[24] intercept   : AIC=inf, Time=8.10 sec
 ARIMA(1,0,1)(0,0,0)[24] intercept   : AIC=-24207.256, Time=0.68 sec
 ARIMA(1,0,1)(1,0,0)[24] intercept   : AIC=-23699.576, Time=5.35 sec
 ARIMA(1,0,1)(0,0,1)[24] intercept   : AIC=-24205.053, Time=3.37 sec
 ARIMA(1,0,1)(1,0,1)[24] intercept   : AIC=inf, Time=4.87 sec
 ARIMA(1,0,0)(0,0,0)[24] intercept   : AIC=-24186.034, Time=0.52 sec
 ARIMA(2,0,1)(0,0,0)[24] intercept   : AIC=-24218.489, Time=1.50 sec
 ARIMA(2,0,1)(1,0,0)[24] intercept   : AIC=-23708.607, Time=4.35 sec
 ARIMA(2,0,1)(0,0,1)[24] intercept   : AIC=-24216.423, Time=4.76 sec
 ARIMA(2,0,1)(1,0,1)[24] intercept   : AIC=inf, Time=5.20 sec
 ARIMA(2,0,0)(0,0,0)[24] intercept   : AIC=-24220.835, Time=1.01 sec
 ARIMA(2,0,0)(1,0,0)[24] intercept   : AIC=-23712.809, Time=4.07 sec
 ARIMA(2,0,0)(0,0,1)[24] intercept   : AIC=-24218.759, Time=4.18 sec
 ARIMA(2,0,0)(1,0,1)[24] intercept   : AIC=inf, Time=4.94 sec
 ARIMA(3,0,0)(0,0,0)[24] intercept   : AIC=-24219.986, Time=0.39 sec
 ARIMA(3,0,1)(0,0,0)[24] intercept   : AIC=-24215.734, Time=0.44 sec
 ARIMA(2,0,0)(0,0,0)[24]             : AIC=-23998.875, Time=0.39 sec

Best model:  ARIMA(2,0,0)(0,0,0)[24] intercept
Total fit time: 69.560 seconds
Best SARIMA Order: (2, 0, 0)
Best Seasonal Order: (0, 0, 0, 24)
Test MSE: 4.449809682376673e-09
Test R2 Score: 0.029179849390996826
Test MAE: 4.670216414293775e-05
```



5.2 Univariate facebook prophet models

Prophet is a time series forecasting model developed by Facebook, designed to handle seasonality, holiday effects, and trends of various time series data. It decomposes time series into trend, seasonal, and holiday components, enabling flexible modeling of complex patterns in the data.

For the demonstration the implementation done for colombo will be used. Same process will be followed for other cities

In the implementation for Colombo, the dataset is prepared by renaming columns and converting the 'ds' column to datetime format. The model is then initialized with specific hyperparameters, including the changepoint_prior_scale and seasonality_prior_scale. Additionally, a holiday component for the COVID-19 pandemic is incorporated. The model

is trained on the training data and then used to make predictions for the future period using the make_future_dataframe function. Performance metrics such as R-squared and Mean Squared Error (MSE) are calculated for evaluating the model's accuracy.

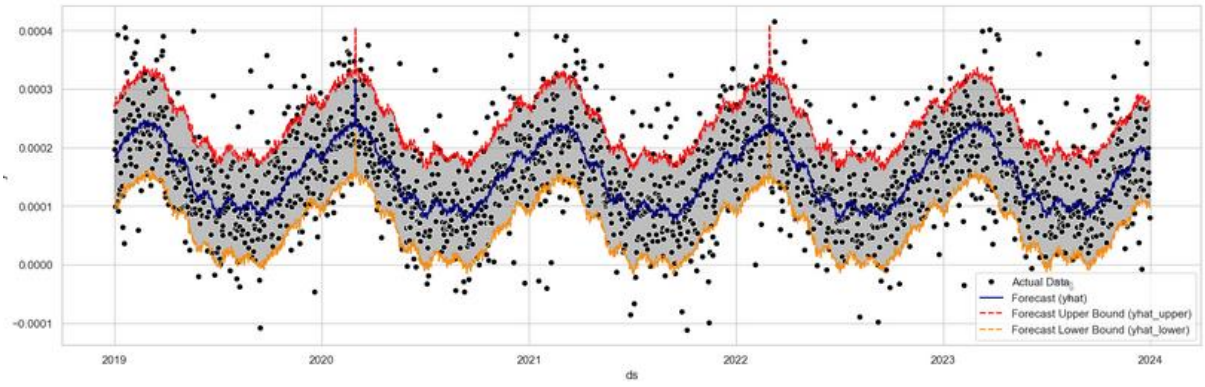
Finally, the actual data and forecasted values along with their confidence intervals are visualized using line plots. The components of the forecast, including trend, yearly seasonality, and holidays, are also displayed to provide further insights into the model's predictions.

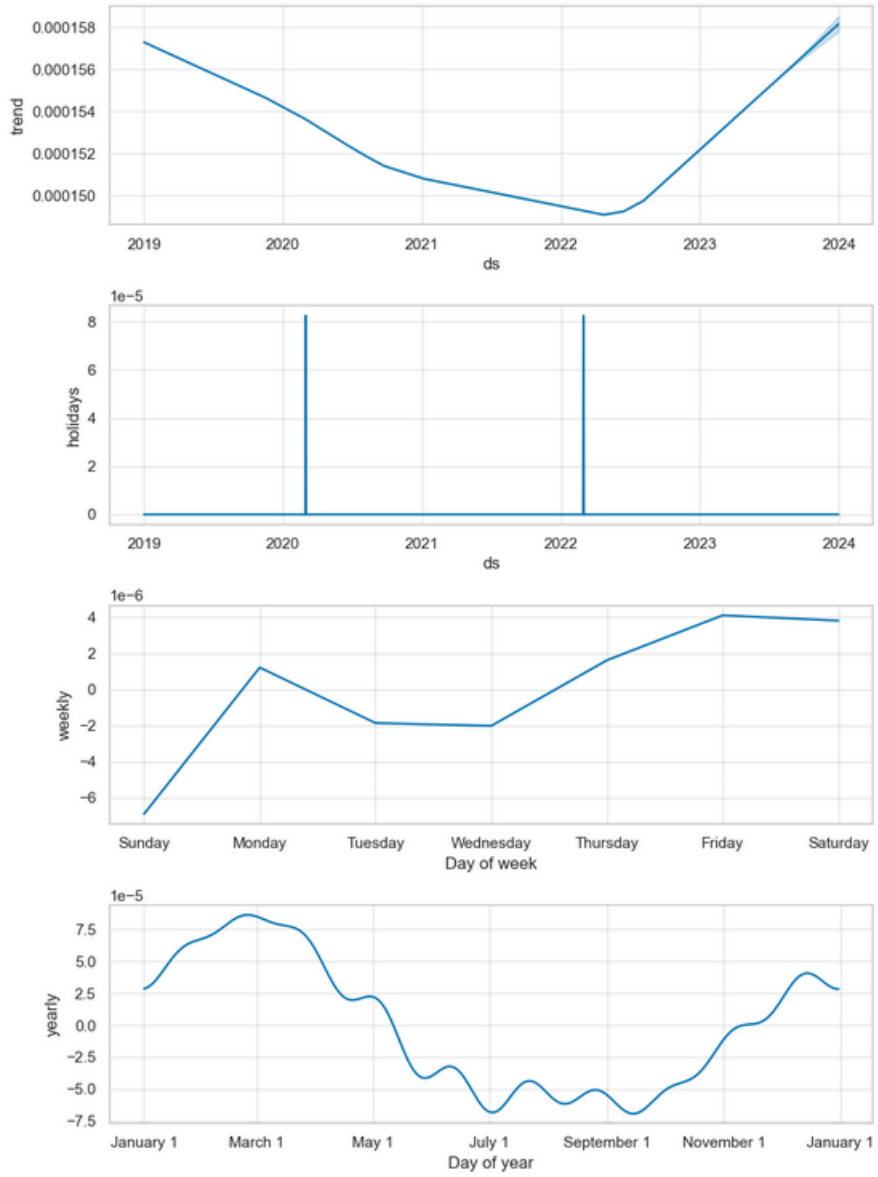
```
data = pd.read_csv(r"..\processed data\temp\colombo_fomaldehyde.csv")
data.drop(columns=['Location','Next_Date'],inplace=True)
data.rename(columns={'Current_Date': 'ds', 'HCHO_reading': 'y'}, inplace=True)
data['ds'] = pd.to_datetime(data['ds'])

# Test size can be adjusted based on your needs (e.g., 0.2 for 20% test data)
train, test = train_test_split(data, test_size=0.1, shuffle=False)
```

```
covid_holidays = pd.DataFrame({
    'holiday': 'covid_19',
    'ds': pd.to_datetime(['2020-03-01', '2022-03-01']),
})

# Create the model (data argument removed)
model = Prophet( changepoint_prior_scale=0.45,  # More flexible trend try-0.2
                 seasonality_prior_scale=5,
                 holidays=covid_holidays,
                yearly_seasonality=True
               )

# Fit the model on the training data
model.fit(train)

future = model.make_future_dataframe(periods=len(test))
forecast = model.predict(future)

r2=r2_score(test['y'], forecast[forecast['ds'].isin(test['ds'])]['yhat'])
mse=mean_squared_error(test['y'], forecast[forecast['ds'].isin(test['ds'])]['yhat'])
print(r2)
print(mse)
```

```
15:14:15 - cmdstanpy - INFO - Chain [1] start processing
15:14:15 - cmdstanpy - INFO - Chain [1] done processing
```

```
0.20722402948431462
4.569194718038498e-09
```



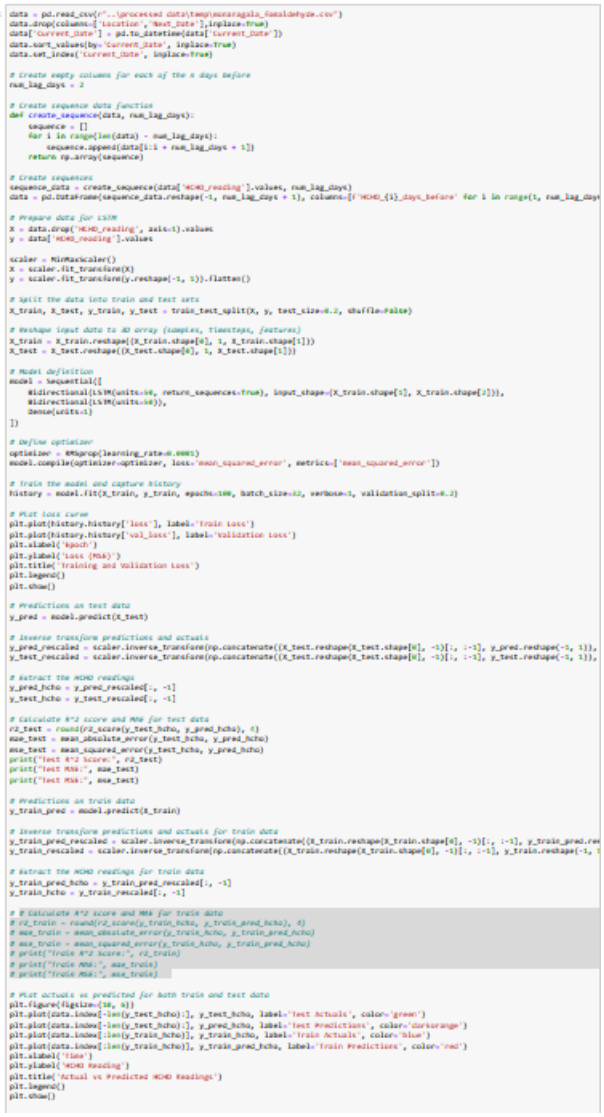


5.3 Univariate LSTMs

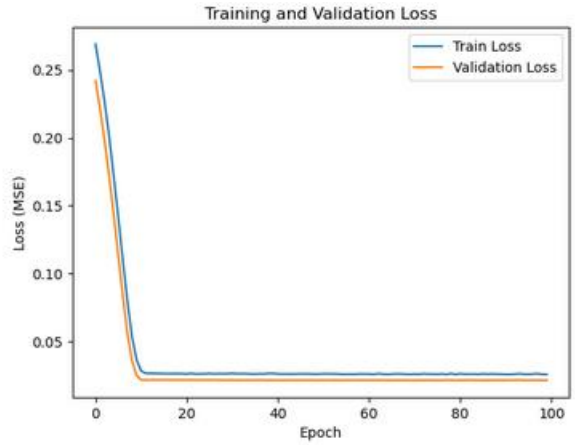
For the demonstration model implemented for Monaragala will be used.

The univariate LSTM (Long Short-Term Memory) model is a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data. In this implementation for Colombo, the dataset is prepared by creating lag features representing historical observations. The model architecture consists of multiple Bidirectional LSTM layers followed by a Dense layer for prediction. The model is trained using the RMSprop optimizer with a learning rate schedule. Performance is evaluated using Mean Squared Error (MSE) loss and visualized through loss curves. Predictions are made on both training and test data, and performance metrics such as R-squared score, Mean Absolute

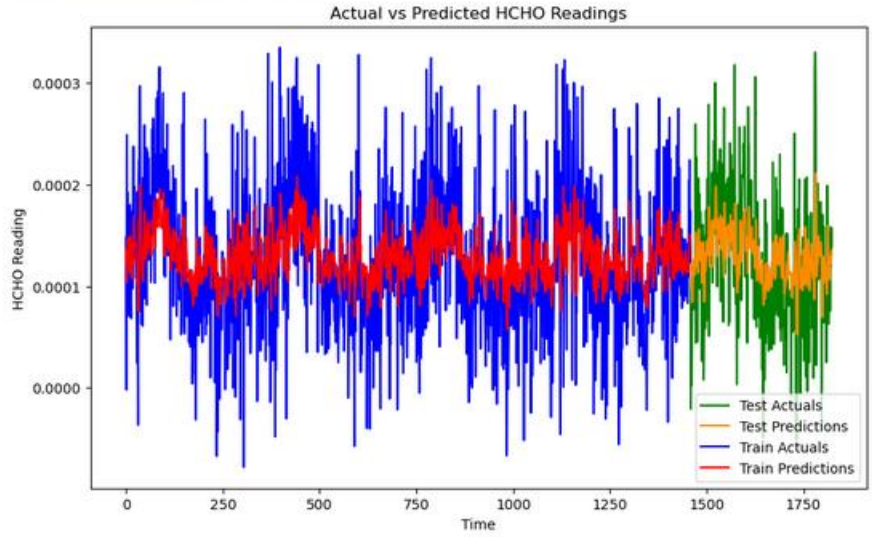
Error (MAE), and MSE are calculated. Finally, the actual and predicted HCHO readings are plotted over time for both training and test datasets to assess the model's accuracy.



```
data = pd.read_csv(r"..\processed data\temp\mosaragala_finaldefyde.csv")
data.drop(columns=['location', 'Next_Date'],inplace=True)
data['Current_Date'] = pd.to_datetime(data['Current_Date'])
data.sort_values(by='Current_Date', inplace=True)
data.set_index('Current_Date', inplace=True)

# Create empty columns for each of the n days before
num_lag_days = 2

# Create sequence data function
def create_sequence(data, num_lag_days):
    sequence = []
    for i in range(len(data) - num_lag_days):
        sequence.append(data[i:i + num_lag_days + 1])
    return np.array(sequence)

# Create sequences
sequence_data = create_sequence(data['HCHO_reading'].values, num_lag_days)
data = pd.DataFrame(sequence_data.reshape(-1, num_lag_days + 1), columns=[f'HCHO_{i}_days_before' for i in range(1, num_lag_days

# Prepare data for LSTM
X = data.drop('HCHO_reading', axis=1).values
y = data['HCHO_reading'].values

scaler = MinMaxScaler()
X = scaler.fit_transform(X)
y = scaler.fit_transform(y.reshape(-1, 1)).flatten()

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

# Reshape input data to 3D array (samples, timesteps, features)
X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

# Model definition
model = Sequential([
    Bidirectional(LSTM(units=50, return_sequences=True), input_shape=(X_train.shape[1], X_train.shape[2])),
    Bidirectional(LSTM(units=50)),
    Dense(units=1)
])

# Define optimizer
optimizer = RMSprop(learning_rate=0.0001)
model.compile(optimizer=optimizer, loss='mean_squared_error', metrics=['mean_squared_error'])

# Train the model and capture history
history = model.fit(X_train, y_train, epochs=100, batch_size=32, verbose=1, validation_split=0.2)

# Plot loss curve
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss (MSE)')
plt.title('Training and Validation Loss')
plt.legend()
plt.show()

# Predictions on test data
y_pred = model.predict(X_test)

# Inverse transform predictions and actuals
y_pred_rescaled = scaler.inverse_transform(np.concatenate((X_test.reshape(X_test.shape[0], -1)[:, :-1], y_pred.reshape(-1, 1)),
y_test_rescaled = scaler.inverse_transform(np.concatenate((X_test.reshape(X_test.shape[0], -1)[:, :-1], y_test.reshape(-1, 1)),

# Extract the HCHO readings
y_pred_hcho = y_pred_rescaled[:, -1]
y_test_hcho = y_test_rescaled[:, -1]

# Calculate R^2 score and MAE for test data
r2_test = round(r2_score(y_test_hcho, y_pred_hcho), 4)
mae_test = mean_absolute_error(y_test_hcho, y_pred_hcho)
mse_test = mean_squared_error(y_test_hcho, y_pred_hcho)
print("Test R^2 Score:", r2_test)
print("Test MAE:", mae_test)
print("Test MSE:", mse_test)

# Predictions on train data
y_train_pred = model.predict(X_train)

# Inverse transform predictions and actuals for train data
y_train_pred_rescaled = scaler.inverse_transform(np.concatenate((X_train.reshape(X_train.shape[0], -1)[:, :-1], y_train_pred.re
y_train_rescaled = scaler.inverse_transform(np.concatenate((X_train.reshape(X_train.shape[0], -1)[:, :-1], y_train.reshape(-1, 1

# Extract the HCHO readings for train data
y_train_pred_hcho = y_train_pred_rescaled[:, -1]
y_train_hcho = y_train_rescaled[:, -1]

# # Calculate R^2 score and MAE for train data
# r2_train = round(r2_score(y_train_hcho, y_train_pred_hcho), 4)
# mae_train = mean_absolute_error(y_train_hcho, y_train_pred_hcho)
# mse_train = mean_squared_error(y_train_hcho, y_train_pred_hcho)
# print("Train R^2 Score:", r2_train)
# print("Train MAE:", mae_train)
# print("Train MSE:", mse_train)

# Plot actuals vs predicted for both train and test data
plt.figure(figsize=(10, 6))
plt.plot(data.index[-len(y_test_hcho):], y_test_hcho, label='Test Actuals', color='green')
plt.plot(data.index[-len(y_test_hcho):], y_pred_hcho, label='Test Predictions', color='darkorange')
plt.plot(data.index[:len(y_train_hcho)], y_train_hcho, label='Train Actuals', color='blue')
plt.plot(data.index[:len(y_train_hcho)], y_train_pred_hcho, label='Train Predictions', color='red')
plt.xlabel('Time')
plt.ylabel('HCHO Reading')
plt.title('Actual vs Predicted HCHO Readings')
plt.legend()
plt.show()
```




```
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 68ms/step
Test R^2 Score: 0.173
Test MAE: 4.9563280781324514e-05
Test MSE: 4.0227793309871865e-09
46/46 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



5.4 Multivariate LSTM

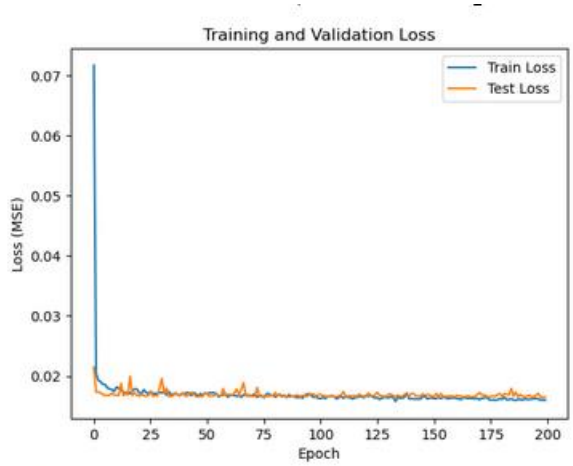
For the demonstration model implemented for Colombo will be used. Same process will be carried out for all the other cities.

In multivariate LSTM model demonstrated here, data from Colombo Proper is utilized for forecasting HCHO readings. The input features include various environmental factors and temporal attributes. After scaling the data, it's split into training and testing sets, reshaped to fit the model architecture, and then fed into a Sequential model consisting of LSTM layers with dropout regularization. The model is trained using the Adam optimizer with a custom learning rate schedule. The training and validation loss curves are plotted to monitor model performance. Predictions are made on both training and test data, and performance metrics such as R-squared score, Mean Absolute Error (MAE), and Mean Squared Error (MSE) are

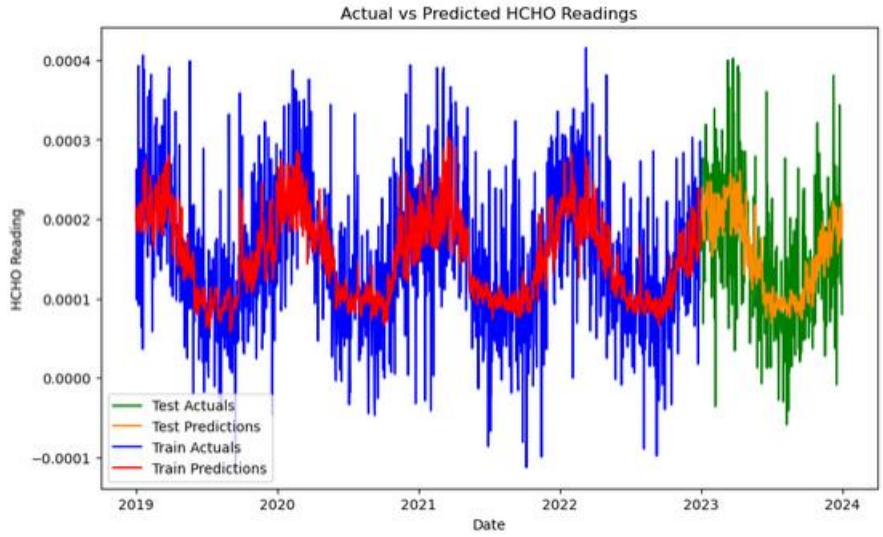
calculated for evaluation. Finally, the actual and predicted HCHO readings are visualized over time for both training and test datasets.

```
modeling_df = df[df['Location'] == 'Colombo Proper']


# Define input features (X) and target variable (y)
# Manual Feature selection
X_cols = ['PM10', 'carbonmonoxide_average', 'nitrogendioxide_average', 'ozone_average','Year','Month','Day','Population', 'total
y_col = ['HCHO_reading']

X = modeling_df[X_cols].values  # Features
y = modeling_df[y_col].values   # Target

scaler = MinMaxScaler()
X = scaler.fit_transform(X)
y = scaler.fit_transform(y)

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

# Reshape input data to 3D array (samples, timesteps, features)
X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

model = Sequential([
    LSTM(units=100, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.2),
    LSTM(units=100, return_sequences=True),
    Dropout(0.2),
    LSTM(units=100),
    Dropout(0.2),
    Dense(1)
])

# Define a learning rate schedule
learning_rate_schedule = schedules.ExponentialDecay(
    initial_learning_rate=0.001,  # Initial learning rate
    decay_steps=1000,             # Decay steps
    decay_rate=0.96,              # Decay rate
    staircase=True                # Whether to apply decay in a staircase manner
)

# Define the optimizer with the learning rate schedule
custom_optimizer = Adam(learning_rate=learning_rate_schedule)

# Compile the model with the custom optimizer
model.compile(optimizer=custom_optimizer, loss='mse')

# Train the model and capture history
history = model.fit(X_train, y_train, epochs=200, batch_size=16, verbose=1, validation_data=(X_test, y_test))

# Plot loss curve
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Test Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss (MSE)')
plt.title('Training and Validation Loss')
plt.legend()
plt.show()

# Predictions on test data
y_pred = model.predict(X_test)

# Predictions on training data
y_train_pred = model.predict(X_train)

# Rescale predictions and actuals
y_pred_rescaled = scaler.inverse_transform(np.concatenate((X_test.reshape(X_test.shape[0], -1)[:, :-1], y_pred), axis=1))
y_test_rescaled = scaler.inverse_transform(np.concatenate((X_test.reshape(X_test.shape[0], -1)[:, :-1], y_test.reshape(-1, 1)),
y_train_pred_rescaled = scaler.inverse_transform(np.concatenate((X_train.reshape(X_train.shape[0], -1)[:, :-1], y_train_pred),
y_train_rescaled = scaler.inverse_transform(np.concatenate((X_train.reshape(X_train.shape[0], -1)[:, :-1], y_train.reshape(-1,

# Extract the HCHO readings
y_pred_hcho = y_pred_rescaled[:, -1]
y_test_hcho = y_test_rescaled[:, -1]
y_train_pred_hcho = y_train_pred_rescaled[:, -1]
y_train_hcho = y_train_rescaled[:, -1]

# Calculate R^2 score and MAE for test data
r2_test = round(r2_score(y_test_hcho, y_pred_hcho), 4)
mae_test = mean_absolute_error(y_test_hcho, y_pred_hcho)
mse_test = mean_squared_error(y_test_hcho, y_pred_hcho)
print("Test R^2 Score:", r2_test)
print("Test MAE:", mae_test)
print("Test MSE:", mse_test)

# Calculate R^2 score and MAE for training data
r2_train = round(r2_score(y_train_hcho, y_train_pred_hcho), 4)
mae_train = mean_absolute_error(y_train_hcho, y_train_pred_hcho)
mse_train = mean_squared_error(y_train_hcho, y_train_pred_hcho)
print("Train R^2 Score:", r2_train)
print("Train MAE:", mae_train)
print("Train MSE:", mse_train)

# Plot actuals vs predicted for both training and test data
plt.figure(figsize=(10, 6))
plt.plot(modeling_df.index[-len(y_test_hcho):], y_test_hcho, label='Test Actuals', color='green')
plt.plot(modeling_df.index[-len(y_pred_hcho):], y_pred_hcho, label='Test Predictions', color='darkorange')
plt.plot(modeling_df.index[:len(y_train_hcho)], y_train_hcho, label='Train Actuals', color='blue')
plt.plot(modeling_df.index[:len(y_train_pred_hcho)], y_train_pred_hcho, label='Train Predictions', color='red')
plt.xlabel('Date')
plt.ylabel('HCHO Reading')
plt.title('Actual vs Predicted HCHO Readings')
plt.legend()
plt.show()
```

```
Epoch 1/200
C:\Users\jana\anaconda3\lib\site-packages\keras\src\layers\rnn\rnn.py:205: UserWarning: Do not pass an `input_shape`/`input_di
```




```
12/12 ━━━━━━━━━━━━━━━━━━━━ 1s 48ms/step
46/46 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
Test R^2 Score: 0.3475
Test MAE: 5.103588938396172e-05
Test MSE: 4.591818355928865e-09
Train R^2 Score: 0.415
Train MAE: 5.003299370275566e-05
Train MSE: 4.376502195699041 5e-09
```



6 References

Abulkhair, A., 2023. *Data Imputation Demystified: Time Series Data. Medium.*. [Online]
Available at: https://medium.com/@aaabulkhair/data-imputation-demystified-time-series-data-69bc9c798cb7
[Accessed 2024].

Agency, E. S., 2024. *Sentinel-5 Precursor (Sentinel-5P) Data Products.* [Online]
Available at: https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-5p/data-products
[Accessed 2024].

Anon., n.d. *Why Sentinel-5p Data for Aerosol Index have Negative Values?.* s.l.:https://www.researchgate.net/post/Why_Sentinel-5p_Data_for_Aerosol_Index_have_Negative_Values.

Center, N. L. R., 2024. *NASA POWER Data Access Viewer.* [Online]
Available at: https://power.larc.nasa.gov/data-access-viewer/

General., S. L. D. o. R., n.d. *Mid year population data.* [Online]
Available at: https://www.rgd.gov.lk/web/index.php?option=com_content&view=article&id=136&Itemid=294&lang=si#%E0%B6%AF%E0%B6%BB%E0%B7%8A%E0%B7%81%E0%B6%9A
[Accessed 2024].

LLC., G., 2024. *Copernicus Sentinel-5P Near Real-Time (NRT) data for carbon monoxide (CO).*. [Online]
Available at: https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S5P_NRTI_L3_CO
[Accessed 2024].

Ninjas., A., 2024. *COVID-19 API.*. [Online]
Available at: https://api-ninjas.com/api/covid19
[Accessed 2024].