Project Report

Project Title: Predict CO2 Emissions in Rwanda

Prepared by Yeswanth Koti, Rohith Surya Podugu, Jaswanth Reddy Botta, Sagar Addala Ram, Sai Nikhitha Madireddy.

Table of Contents

1. Project Description:

The objective of this challenge is to create machine learning models that use open-source emissions data (from Sentinel-5P satellite observations) to predict carbon emissions.
Approximately 497 unique locations were selected from multiple areas in Rwanda, with a distribution around farmlands, cities, and power plants. The data for this competition is split by time; the years 2019 - 2021 are included in the training data and testing data.

2. Abstract:

The increasing threat of climate change necessitates accurate monitoring of carbon emissions, a crucial step in mitigating its impact. In Africa, where ground-based monitoring is limited, this Kaggle Playground Series challenges participants to develop machine learning models using open-source CO2 emissions data from Sentinel-5P satellite observations. The competition aims to predict future carbon emissions across Africa, enabling governments and stakeholders to estimate emission levels even in areas where on-the-ground monitoring is challenging.
Acknowledging the importance of this initiative, the competition credits Carbon Monitor for providing the GRACED dataset and recognizes the contribution of Darius Moruri from Zindi in preparing the dataset and starter notebooks. The collaboration between Kaggle and Zindi reflects their commitment to fostering community-driven impact in Africa and advancing the capabilities of data scientists.

3. Responsibility of each Team member:

S. No	Name	Responsibility
1.	Rohith Surya Podugu (Project Lead)	Data Processing, Data Modelling
2.	Koti Yeswanth	Data Modelling, Documentation
3.	Jaswanth Reddy Botta	Feature Engineering
4.	Sagar Addala Ram	Data Analysis, Documentation
5.	Sai Nikitha Madireddy	Output Metrics

4. Information about the Data:

Seven main features were extracted weekly from Sentinel-5P from January 2019 to November 2022. Each feature (Sulphur Dioxide, Carbon Monoxide, etc) contain sub features such as column_number_density, which is the vertical column density at ground level, calculated using the DOAS technique. You can read more about each feature in the below links, including how they are measured and variable definitions. You are given the values of these features in the test set and your goal to predict CO2 emissions using time information as well as these features.

- Sulphur Dioxide - COPERNICUS/S5P/NRTI/L3_SO2
- Carbon Monoxide - COPERNICUS/S5P/NRTI/L3_CO
- Nitrogen Dioxide - COPERNICUS/S5P/NRTI/L3_NO2
- Formaldehyde - COPERNICUS/S5P/NRTI/L3_HCHO
- UV Aerosol Index - COPERNICUS/S5P/NRTI/L3_AER_AI
- Ozone - COPERNICUS/S5P/NRTI/L3_O3
- Cloud - COPERNICUS/S5P/OFFL/L3_CLOUD

5. Files used for training and testing:

- **train.csv** – Data set (The data is divided into 65% training and 35% testing)

6. Tools and Technologies:

Programming language **:** Python
Framework **:** jupyter notebook
Libraries **:** sckitlearn, NumPy, matplotlib
ML Algorithms used **:** Radius neighbors Regressor, Decision Tree Regressor

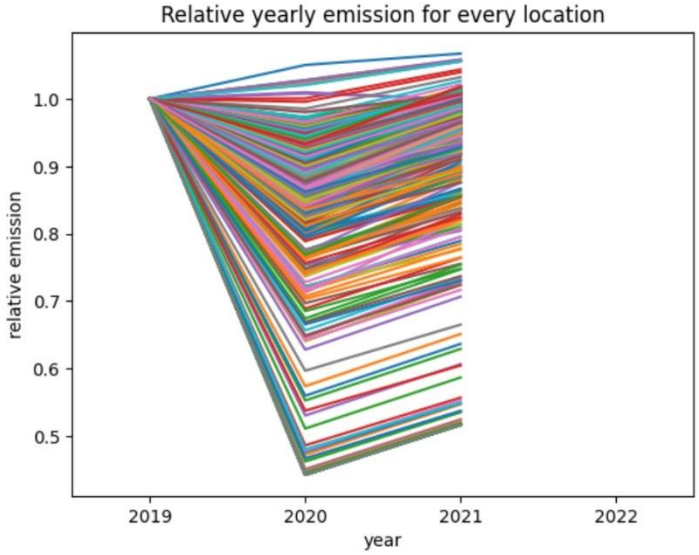
7. Approach:

If we divide the training data by location, we can see that each location has 159 emission prediction data points scatter along the year.

```
print(df.groupby(['latitude', 'longitude']).size().sort_values())
```

```
latitude  longitude
-3.299    30.301       159
-1.444    30.856       159
-1.450    29.350       159
-1.482    30.618       159
-1.486    29.614       159
                      ...
-2.293    29.507       159
-2.300    29.200       159
-2.301    29.899       159
-2.257    30.243       159
-0.510    29.290       159
Length: 497, dtype: int64
```

Covid-19 is an outlier for the training data since the emission are lower than the normal emissions in 2019 and 2021, this posses a risk for the prediction, so covid-19 months can be ignored to have better algorithm RMSE.



And if we, the features with gases in the atmosphere have a lot of missing values and noise and we can omit it to have better prediction.
So, the only features we use for our algorithm are longitude, latitude, week, year and emission data.

We have used simple algorithm for our prediction, we tried working with Decision Tree Regressor and Radius Neighbour Regressor for our predictions.

8. ML Algorithms:

Parameter	Radius Neighbors Regressor	Decision Tree Regressor
Algorithm Description	Utilizes a radius-based approach to find neighbors within a specified radius and calculates the average target variable of the neighbors as the prediction.	Constructs a tree structure based on feature splits to make predictions. Suitable for capturing complex relationships in data.
Applicability:	Effective in capturing localized patterns in the data, suitable for spatially clustered emissions data.	Versatile for various data patterns, including temporal and non-linear relationships.
Advantages:	1.Handles non-linear relationships well. 2. Robust to outliers.	1. Intuitive interpretation. 2.Versatile for capturing complex relationships.
Dis-Advantages:	1.Sensitive to the choice of radius. 2.Computationally expensive for large datasets.	1.Prone to overfitting. 2.Sensitivity to variations in training data.
Performance Metrics	**RMSE value:** 148.4398	**RMSE value:** Standard Decision Tree Regressor: 18.9514 No covid Decision Tree Regressor:18.5736

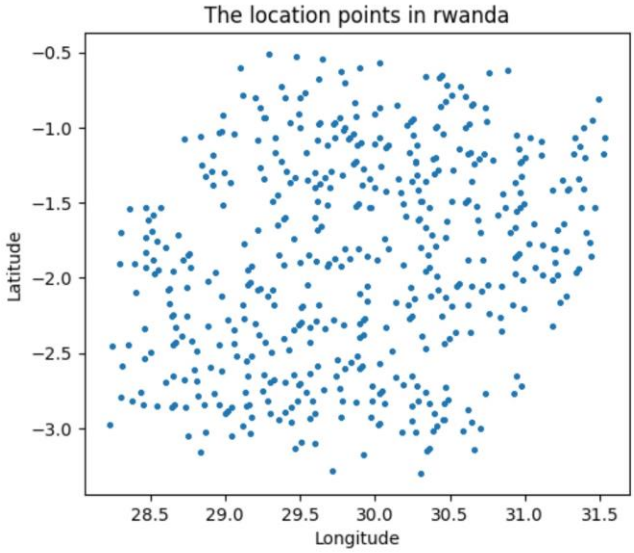
Radius Neighbors Regressor has a high RMSE since it depends on the nearest neighbor's data if the current location isn't present in the data. Since neighbors have fluctuating data, the Radius neighbors regressor algorithm is not preferred.

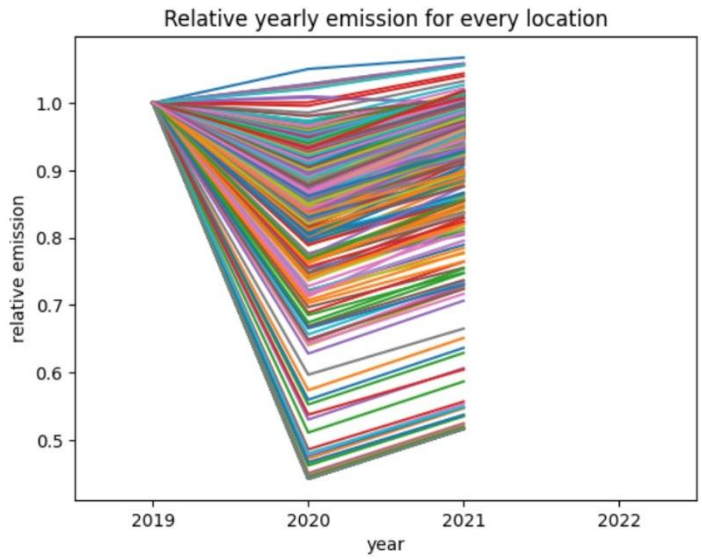
We use all the data for the Standard Decision Tree Regressor, including covid related; since covid months are an outlier, the RMSE for the Standard decision Tree regressor is greater than the No Covid Decision Tree Regressor, but no significant change in RMSE is observed.

The only data attributes used are longitude, latitude, week, year, and emission. The rest of the features are not used for predicting the emission since they are noisy and have a lot of missing values, and they don't provide any significance in our prediction.

9. Results of our Work:

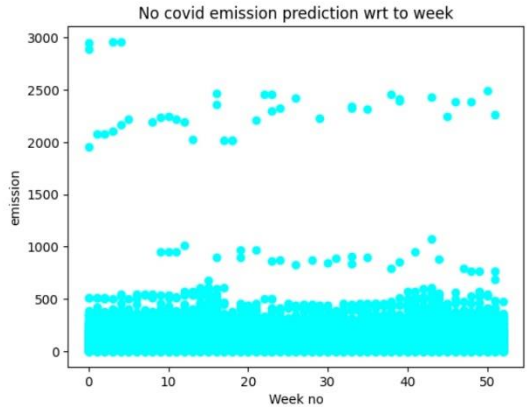
Data Analysis for training data:





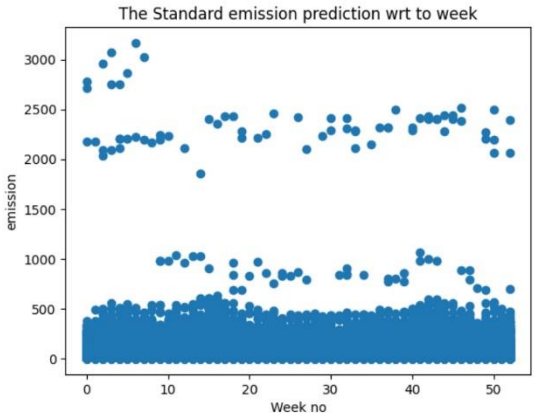
Visualization for prediction results:

```
plt.title("No covid emission prediction wrt to week")
plt.scatter(X_test_nocovid.week_no, y_pred_nocovid, color='cyan')
plt.xlabel('Week no')
plt.ylabel('emission')
plt.show()
```



[252]:

```
plt.title("The Standard emission prediction wrt to week")
plt.scatter(X_test_standard.week_no, y_pred_standard)
plt.xlabel('Week no')
plt.ylabel('emission')
plt.show()
```



RMSE values for Radius Neighbor regressor, decision tree regressor algorithms:

```
X = df[feature_columns]
y = df['emission']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.35, random_state=3)

model = DecisionTreeRegressor(random_state=2)
model.fit(X_train[['longitude', 'latitude', 'week_no', 'year']], y_train)
y_pred_standard = model.predict(X_test[['longitude', 'latitude', 'week_no', 'year']])
X_test_standard = X_test

# submission_standard = pd.Series(y_pred, name='emission', index=X_test.index)
standard_rmse = mean_squared_error(y_pred_standard, y_test, squared=False)
print("The RMSE value for Standard Decision Tree Regressor is", standard_rmse)
```

```
The RMSE value for Standard Decision Tree Regressor is 18.951456252347004
```

[13]:

```
# Drop the covid weeks as outliers
train_nocovid = df[(df.year == 2019) |
                       (df.year == 2020) & (df.week_no <= 8) |
                       (df.year == 2021) & (df.week_no > 8)]

X = train_nocovid[feature_columns]
y = train_nocovid['emission']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.35, random_state=3)

model = DecisionTreeRegressor(random_state=2)
model.fit(X_train[['longitude', 'latitude', 'week_no', 'year']], y_train)
y_pred_nocovid = model.predict(X_test[['longitude', 'latitude', 'week_no', 'year']])
X_test_nocovid = X_test

# submission_nocovid = pd.Series(y_pred, name='emission', index=X_test.index)

nocovid_rmse = mean_squared_error(y_pred_nocovid, y_test, squared=False)
print("The RMSE value for No Covid Decision Tree Regressor is", nocovid_rmse)
```

```
The RMSE value for No Covid Decision Tree Regressor is 18.573609732881746
```

:

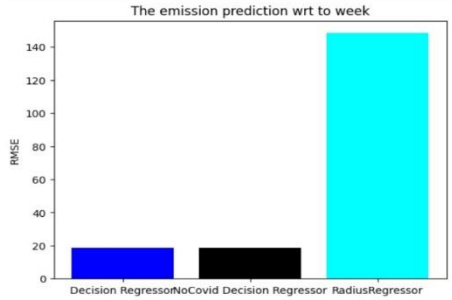
```
X = df[feature_columns]
y = df['emission']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.35, random_state=3)

model = RadiusNeighborsRegressor(radius=2)
model.fit(X_train[['longitude', 'latitude', 'week_no', 'year']], y_train)
y_pred = model.predict(X_test[['longitude', 'latitude', 'week_no', 'year']])

radius_rmse = mean_squared_error(y_pred, y_test, squared=False)
print("The RMSE value for RadiusNeighboursRegressor is", radius_rmse)
```

```
The RMSE value for RadiusNeighboursRegressor is 148.43982957762086
```

Comparison of RMSE Between the Radius regressor, decision regressor, and no covid decision regressor:

[208]:

```
plt.title("The emission prediction wrt to week")
plt.bar(['Decision Regressor', 'NoCovid Decision Regressor', 'RadiusRegressor'], [standard_rmse, nocovid_rmse, radius_rmse],color=['blue', 'black', 'cyan'])
plt.ylabel('RMSE')
# plt.gca().set_aspect('equal')
plt.show()
```



10.Conclusion:

In conclusion, Radius Neighbors Regressor has a high RMSE since it depends on the nearest neighbor's data if the current location isn't present in the data. Since neighbors have fluctuating data, the Radius neighbors regressor algorithm is not preferred.

We use all the data for the Standard Decision Tree Regressor, including covid related; since covid months are an outlier, the RMSE for the Standard decision Tree regressor is greater than the No Covid Decision Tree Regressor, but no significant change in RMSE is observed.

The only data attributes used are longitude, latitude, week, year, and emission. The rest of the features are not used for predicting the emission since they are noisy and have a lot of missing values, and they don't provide any significance in our prediction.