FORECASTING CO2 EMISSIONS OF AGRI FOOD

In an era where climate change poses a significant challenge to the sustainability of our environment, there is an urgent necessity for tools that can effectively gauge the environmental impact of industrial sectors. One such critical area is the agri-food sector, known for its considerable contributions to global emissions. Our objective is to construct a machine learning algorithm that simplifies the process of predicting global temperature increases by analyzing emissions data from this sector. This solution aims to eliminate the cumbersome task of manually sifting through extensive climate data, allowing stakeholders to identify significant emission contributors quickly and effortlessly.

To refine climate impact strategies, we will provide users with a detailed breakdown of emission sources, correlating them with temperature changes. Our algorithm will offer a visual representation of emission trends and their corresponding temperature impacts over time. Furthermore, it will highlight insights such as the most impactful agricultural practices, seasonal emission patterns, and correlations with global temperature rises. This comprehensive summary is designed to equip policymakers, researchers, and environmentalists with the knowledge to make data-driven decisions and to enhance their efforts in curbing the detrimental effects of agri-food practices on global warming.

PROBLEM DEFINITION

This initiative is geared toward tackling the significant challenge of forecasting the impact of agri-food sector emissions on global temperature increases. By deploying a machine learning model, we aim to meticulously analyze emission data to predict average yearly temperature rise. This model's prediction seeks to decipher the intricate relationship between agricultural practices and their long-term climatic repercussions.

The primary goal is to use these predictions as a framework for guiding global climate policies and agri-food sector strategies. Furthermore, the predictive power of the algorithm may offer insights for crafting more sustainable agricultural practices. Nonetheless, the project faces the challenge of accounting for confounding variables and intrinsic biases that may affect emission data and temperature records.

Therefore, a substantial aspect of our methodology is the incorporation of correctional factors into our algorithm to minimize the influence of these biases, ensuring that the predictions are both robust and reliable. Throughout this endeavor, we will investigate a spectrum of machine learning techniques tailored to time series forecasting, judging their effectiveness through metrics such as the coefficient of determination (R^2), RMSE, and MAE, among others

DATA DESCRIPTION

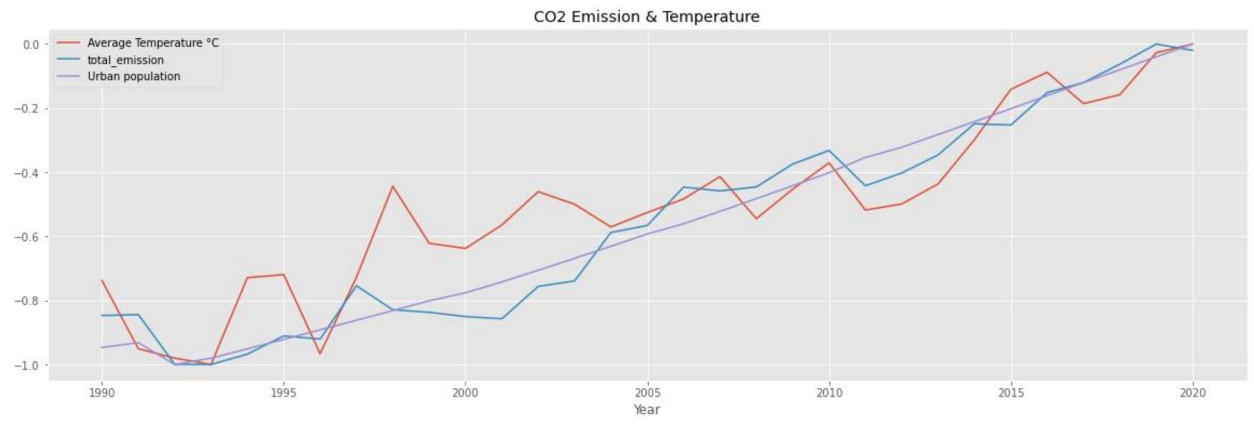
Column Name	Data Type	Description
Area	String	The geographic area or country where the data were collected.
Year	Integer	The year in which the data were recorded.
Savanna Fires	Float	Emissions from fires in savanna ecosystems, measured in kilotons.
Forest Fires	Float	Emissions from fires in forested areas, measured in kilotons.
Crop Residues	Float	Emissions from burning or decomposing plant material left after harvest, measured in kilotons.
Rice Cultivation	Float	Methane emissions from rice paddies, measured in kilotons.
Drained Organic Soils	Float	CO2 emissions released from draining organic soils, measured in kilotons.
Pesticides Manufacturing	Float	Emissions from the production of pesticides, measured in kilotons.
Food Transport	Float	Emissions from transporting food products, measured in kilotons.

Forestland	Float	Area covered by forests, which can act as carbon sinks, measured in hectares.
Net Forest Conversion	Float	Net change in forest area due to deforestation and afforestation, measured in hectares.
Food Household Consumption	Float	Emissions from food consumption at the household level, measured in kilotons.
Food Retail	Float	Emissions from retail establishments selling food, measured in kilotons.
On-farm Electricity Use	Float	Electricity consumption on farms, measured in kilotons.
Food Packaging	Float	Emissions from the production and disposal of food packaging materials, measured in kilotons.
Agrifood Systems Waste Disposal	Float	Emissions from waste disposal in agrifood systems, measured in kilotons.
Food Processing	Float	Emissions from processing food products, measured in kilotons.
Fertilizers Manufacturing	Float	Emissions from the production of fertilizers, measured in kilotons.
IPPU	Float	Emissions from industrial processes and product use, measured in kilotons.
Manure Applied to Soils	Float	Emissions from applying animal manure to agricultural soils, measured in kilotons.
Manure Left on Pasture	Float	Emissions from animal manure left on pasture or grazing land, measured in kilotons.
Manure Management	Float	Emissions from managing and treating animal manure, measured in kilotons.
Fires in Organic Soils	Float	Emissions from fires in organic soils, measured in kilotons.
Fires in Humid Tropical Forests	Float	Emissions from fires in humid tropical forests, measured in kilotons.
On-farm Energy Use	Float	Total energy consumption on farms, including fuel and electricity, measured in kilotons.

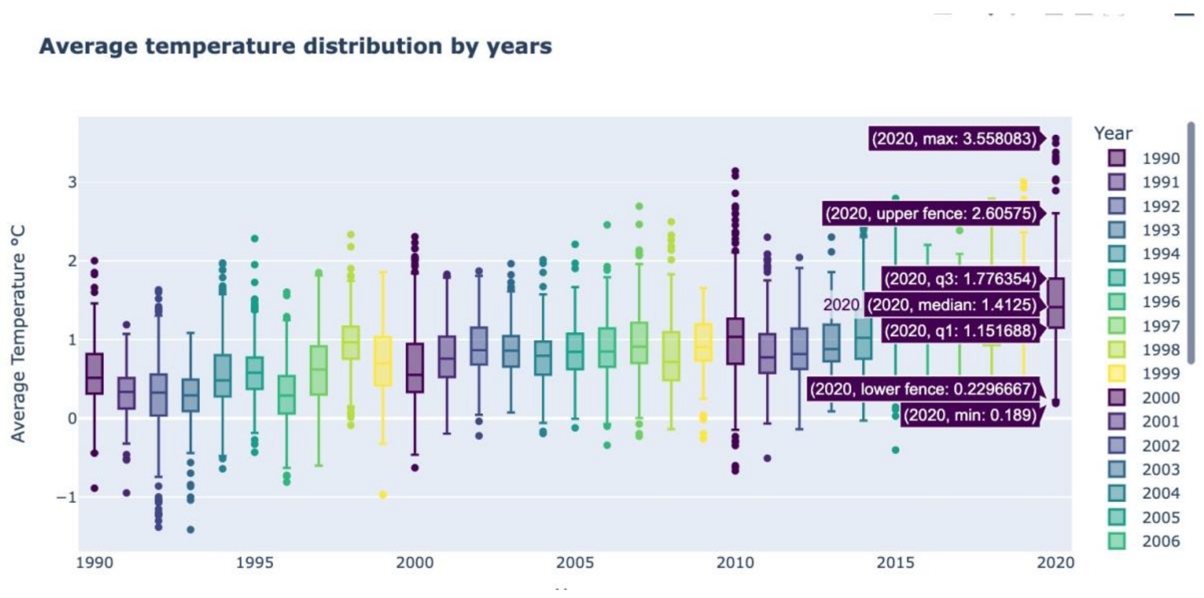
Rural Population	Integer	Number of people living in rural areas.
Urban Population	Integer	Number of people living in urban areas.
Total Population - Male	Integer	Total number of male individuals in the population.
Total Population - Female	Integer	Total number of female individuals in the population.
Total Emission	Float	Total greenhouse gas emissions from all listed sources, measured in kilotons.
Average Temperature °C	Float	The average yearly increase in temperature, measured in degrees Celsius.

https://www.kaggle.com/datasets/alessandrolobello/agri-food-co2-emission-dataset-forecastingml/data

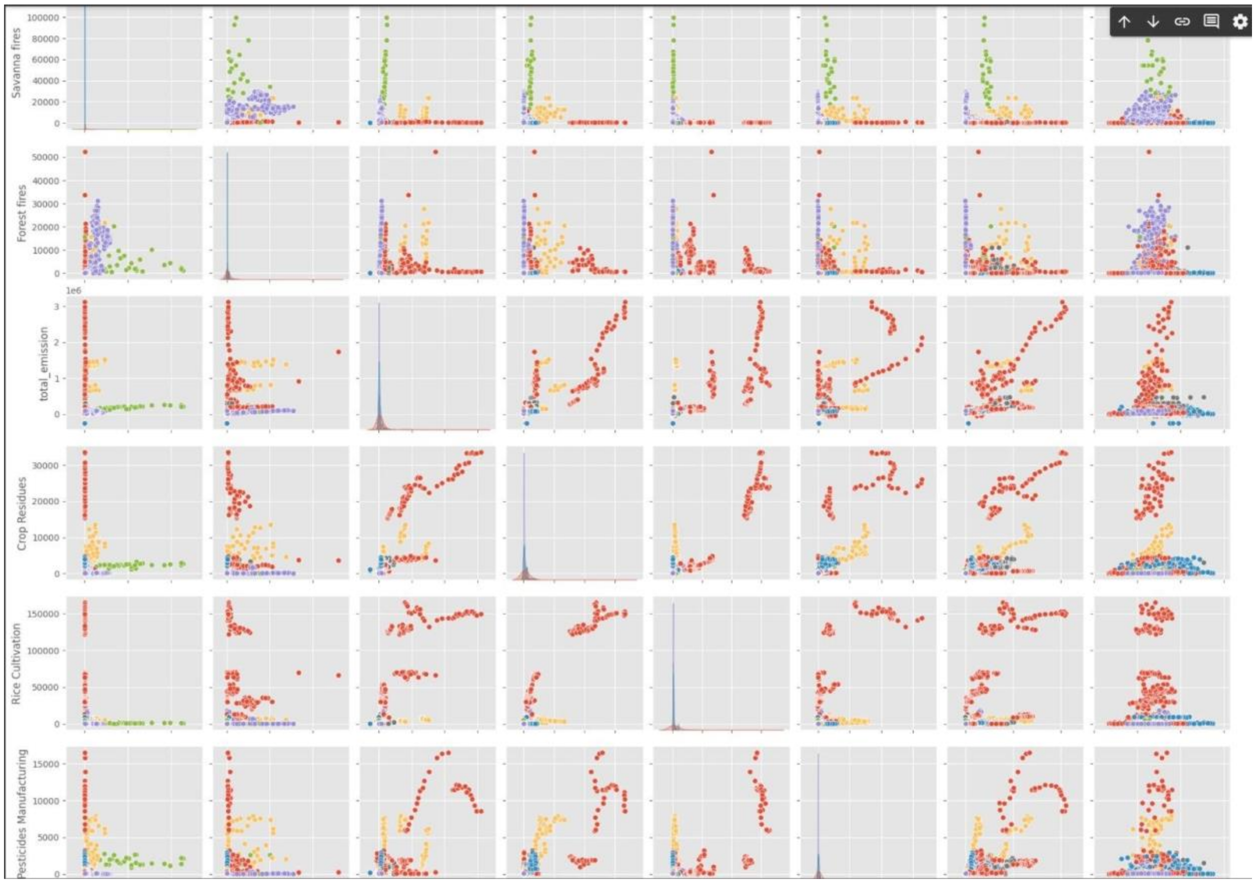
DATA EXPLORATION AND VISUALIZATION



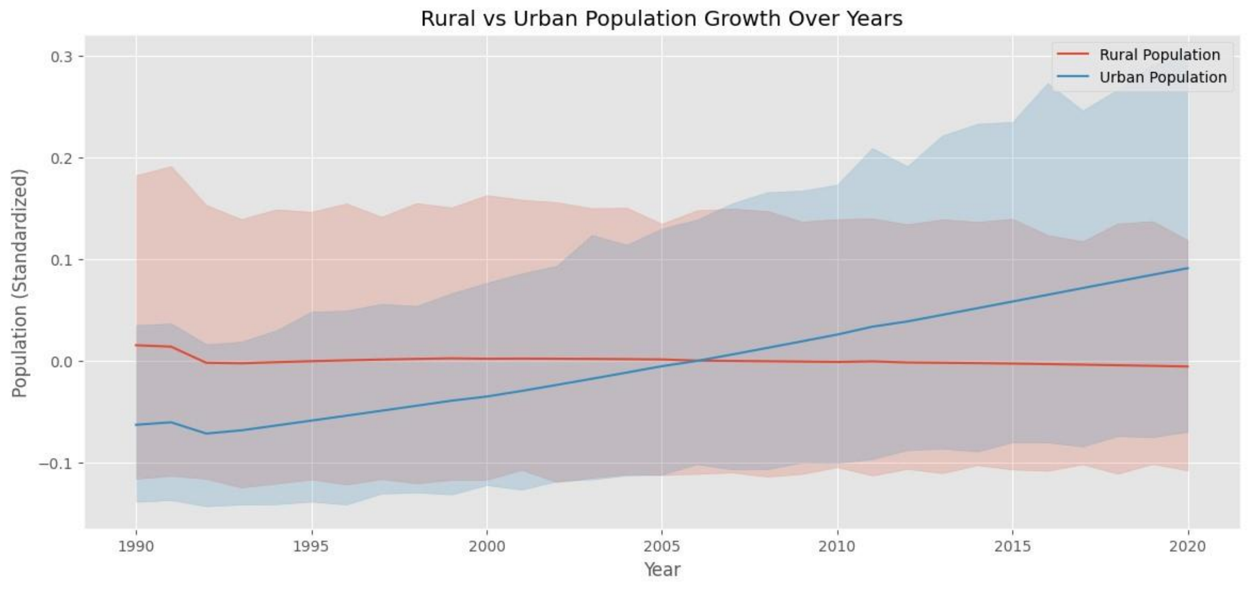
Observation: The above plot shows three normalized trends from 1990 to around 2020: average temperature, total emissions, and urban population. Normalized values allow for easy comparison of the trends. We observe that while urban population has a steady increase, both CO2 emissions and average temperature exhibit more fluctuation but an overall increasing trend, which could suggest a correlation between these factors and urbanization.



Observation: The above plot is a box plot displaying the distribution of average temperatures by year. Each box represents the interquartile range (IQR) of temperatures in a given year, with the line in the middle of the box showing the median. This visualization helps to detect yearly variations and overall trends in temperature.



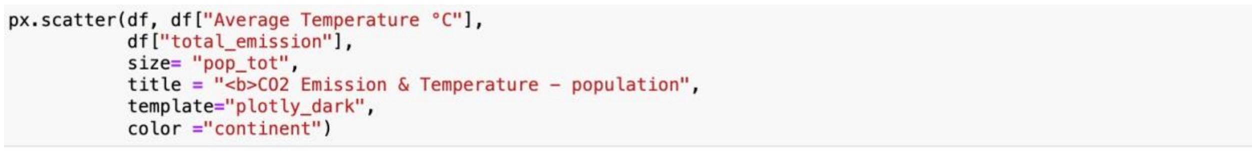
Each scatter plot in the grid shows the relationship between two variables in the dataset. This type of visualization is useful for examining pairwise relationships and distributions of multiple variables simultaneously.



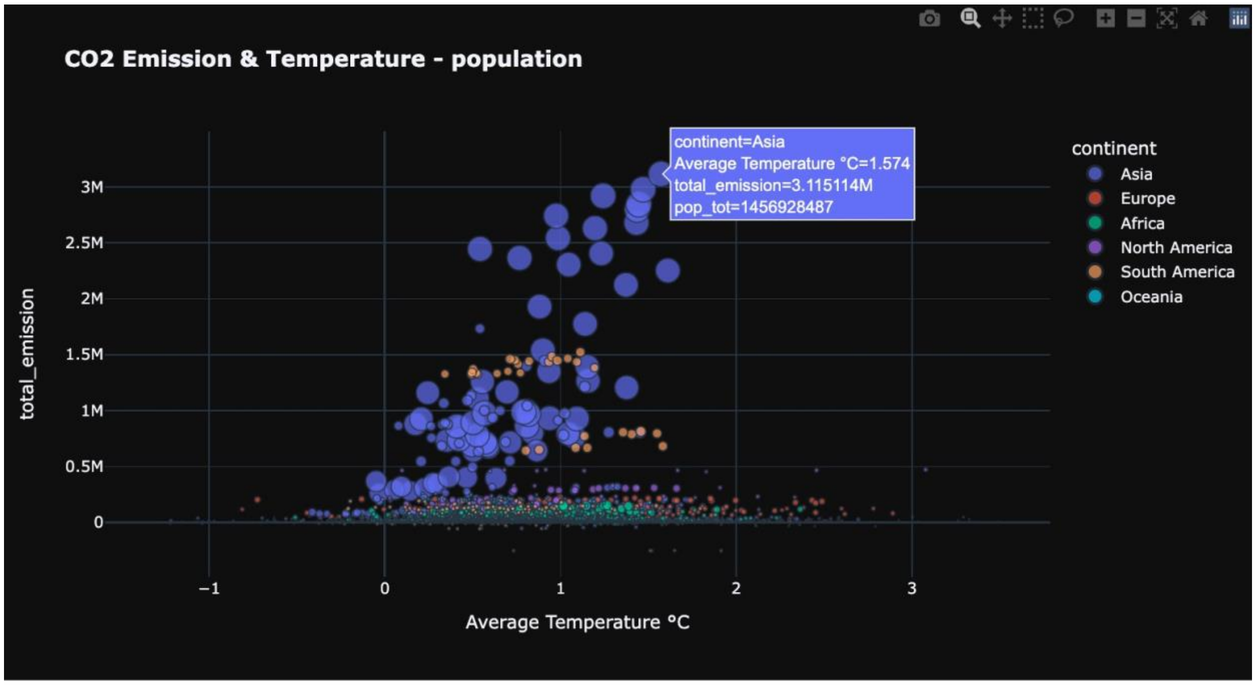
The observed trends from the graph suggest a stable rural population over the years, with minimal increase or decrease. In contrast, the urban population exhibits a steady upward trend, indicating a significant shift towards urban living as we progress through the decades. The difference in growth patterns between rural and urban populations is pronounced, signaling ongoing urbanization.

Is there any correlation between emissions and temperature?

In countries with a significant population, there seems to be a stronger correlation between climate change and CO2 agrifood emissions. Asia seems to be the continent with the stronger correlation between both variables.



```
px.scatter(df, df["Average Temperature °C"],
           df["total_emission"],
           size= "pop_tot",
           title = "<b>CO2 Emission & Temperature - population",
           template="plotly_dark",
           color ="continent")
```



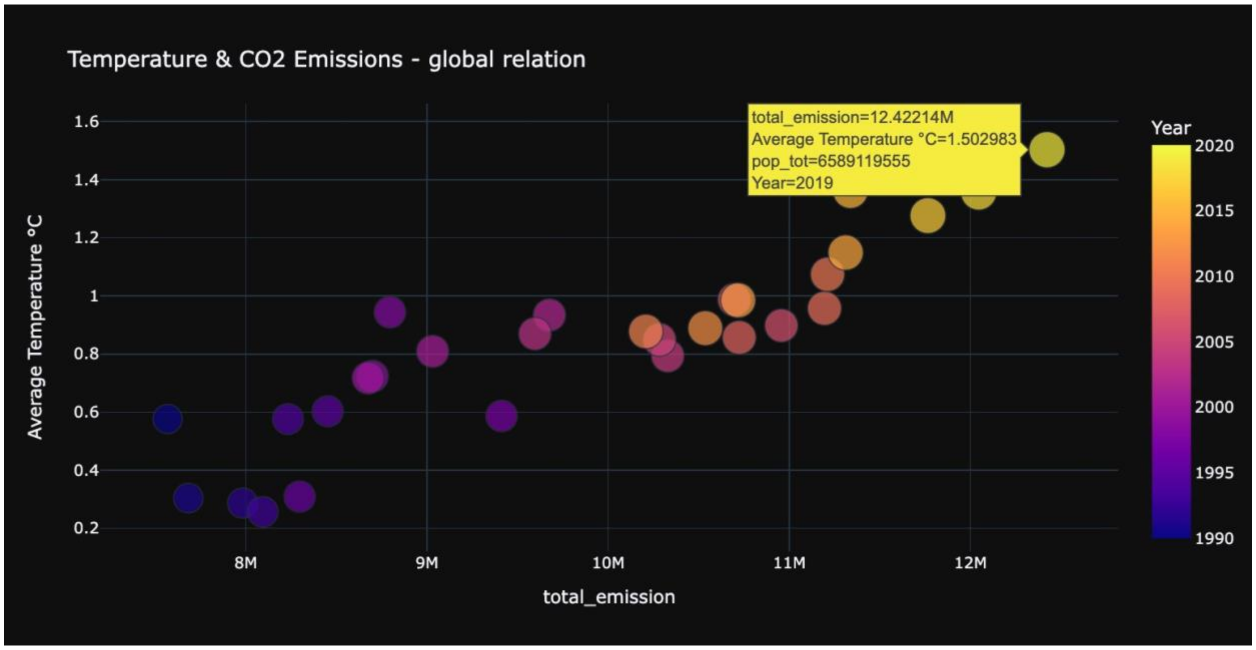
The code is intended to create a scatter plot with the average temperature on the x-axis, total CO2 emissions on the y-axis, the size of the points representing the total population, and colors indicating the continent.

```
correlation = df.groupby(["Year"]).agg({"total_emission":"sum", "Average Temperature °C":"mean", "pop_tot":"sum"})
correlation.corr()
```

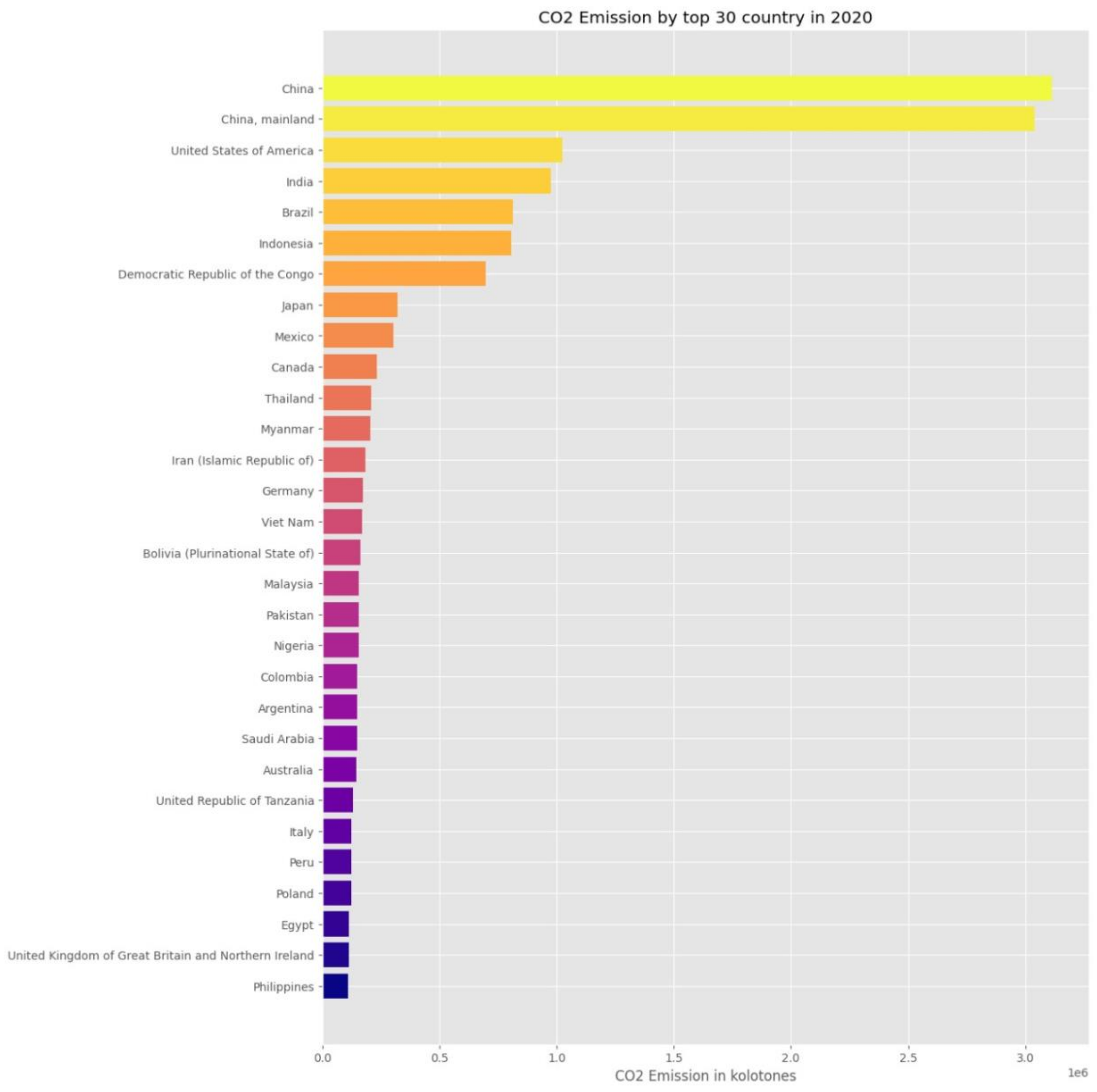
	total_emission	Average Temperature °C	pop_tot
total_emission	1.000000	0.906813	0.964216
Average Temperature °C	0.906813	1.000000	0.916025
pop_tot	0.964216	0.916025	1.000000

- Total_emission and Average Temperature °C have a correlation of approximately 0.906, indicating a very strong positive relationship. As average temperature increases, total emissions also tend to increase.
- Total_emission and pop_tot have an even higher correlation of approximately 0.982, suggesting that increases in population are very strongly associated with increases in total emissions.

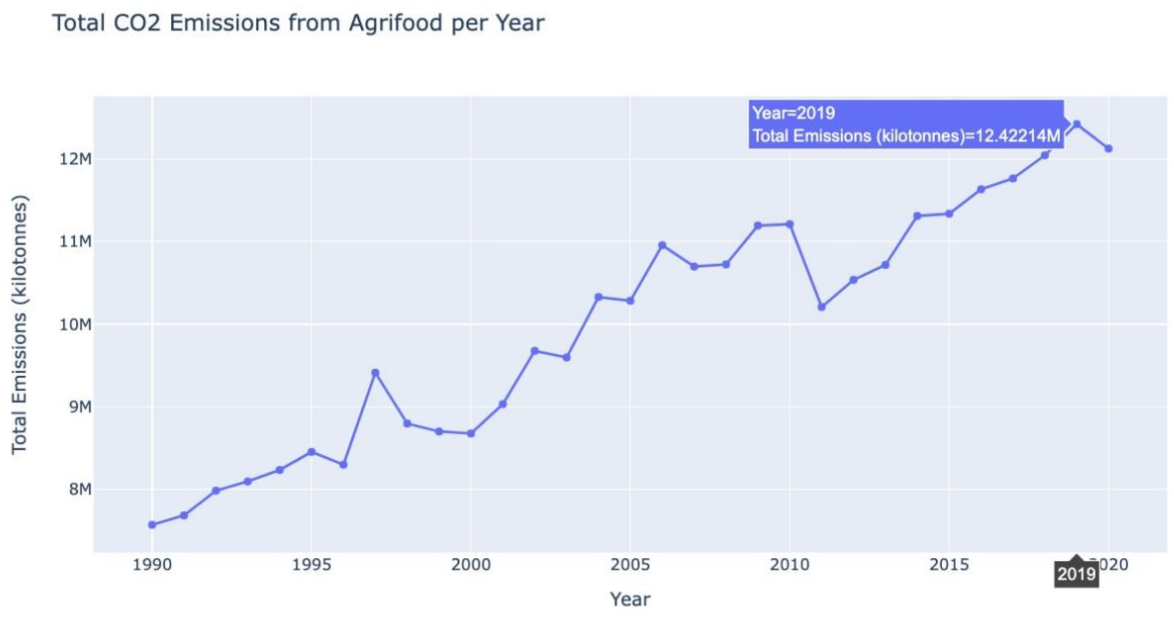
- Average Temperature °C and pop_tot have a correlation of approximately 0.912, which is also a strong positive relationship, indicating that higher temperatures are associated with higher populations.
- These strong correlations suggest that as the population grows, the total emissions tend to increase, which might also correlate with the average temperature increase.



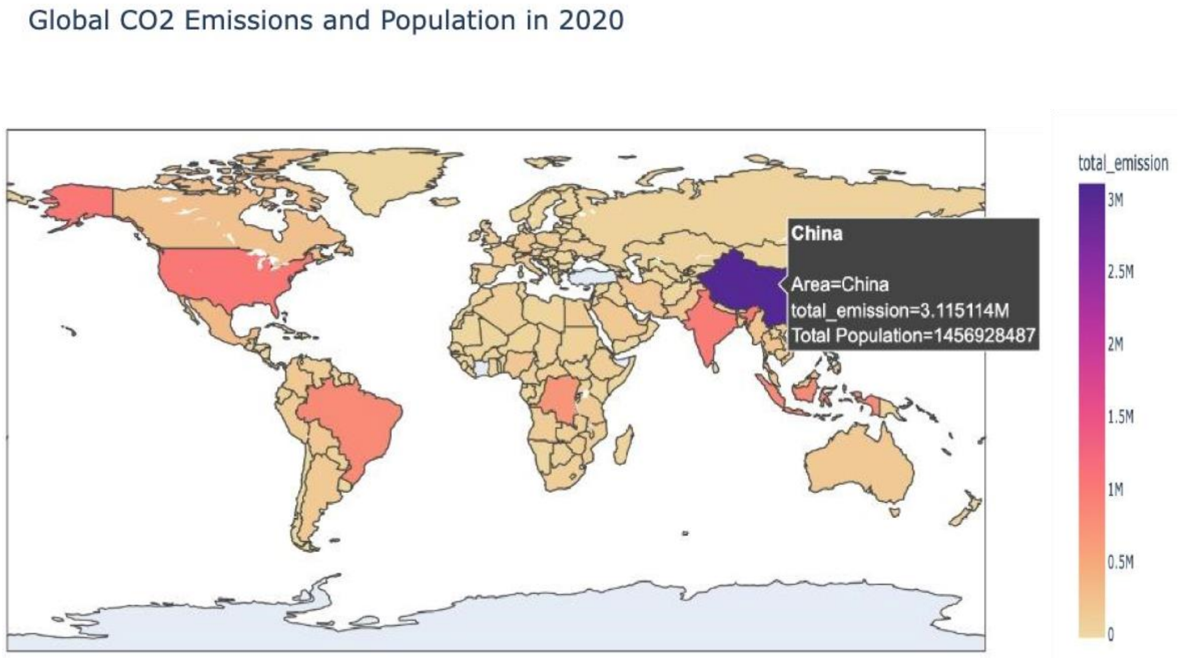
It is a scatter plot that visually represents a relationship between total CO2 emissions and average temperature, with the size of each point corresponding to the total population and the color representing the year.



The above graph shows horizontal bar chart displaying CO2 emissions for the top 30 countries in 2020. The chart uses a color gradient to perhaps signify the magnitude of the emissions, with the bars for countries with higher emissions in brighter colors.



Observation: The total CO2 emissions from agrifood each year. The trend suggests an overall increase in emissions over time, particularly from the early 2000s onward.



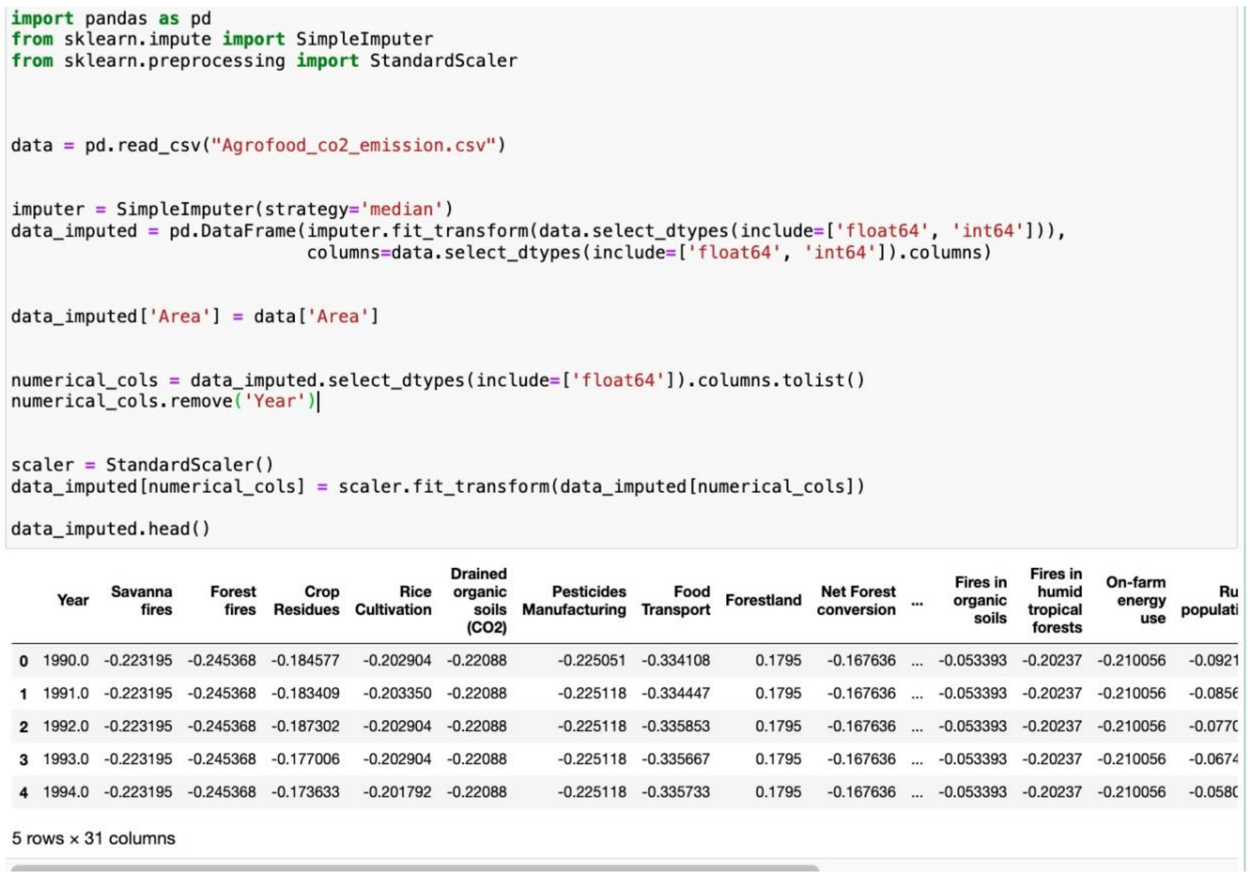
The map shows global CO2 emissions by country for the year 2020, with different color intensities indicating the level of emissions. his visualization effectively highlights how emissions are distributed across different regions of the world.

Insights:

As this exploratory data analysis demonstrates:

- The continent with the **highest CO2 emissions** is **Asia**.
- However, there is a strong correlation between Asia's massive emission and its large population. In actuality, the Americas and **Oceania** have the highest per capita emissions.
- The graph below shows that **Europe seems to be the continent most affected by climate change** in terms of **temperature**, specifically the average annual increase in Celsius.

DATA MINING TASKS



```
import pandas as pd
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler


data = pd.read_csv("Agrofood_co2_emission.csv")


imputer = SimpleImputer(strategy='median')
data_imputed = pd.DataFrame(imputer.fit_transform(data.select_dtypes(include=['float64', 'int64'])),
                            columns=data.select_dtypes(include=['float64', 'int64']).columns)


data_imputed['Area'] = data['Area']


numerical_cols = data_imputed.select_dtypes(include=['float64']).columns.tolist()
numerical_cols.remove('Year')


scaler = StandardScaler()
data_imputed[numerical_cols] = scaler.fit_transform(data_imputed[numerical_cols])

data_imputed.head()
```

	Year	Savanna fires	Forest fires	Crop Residues	Rice Cultivation	Drained organic soils (CO2)	Pesticides Manufacturing	Food Transport	Forestland	Net Forest conversion	...	Fires in organic soils	Fires in humid tropical forests	On-farm energy use	Ru populati
0	1990.0	-0.223195	-0.245368	-0.184577	-0.202904	-0.22088	-0.225051	-0.334108	0.1795	-0.167636	...	-0.053393	-0.20237	-0.210056	-0.0921
1	1991.0	-0.223195	-0.245368	-0.183409	-0.203350	-0.22088	-0.225118	-0.334447	0.1795	-0.167636	...	-0.053393	-0.20237	-0.210056	-0.0856
2	1992.0	-0.223195	-0.245368	-0.187302	-0.202904	-0.22088	-0.225118	-0.335853	0.1795	-0.167636	...	-0.053393	-0.20237	-0.210056	-0.0770
3	1993.0	-0.223195	-0.245368	-0.177006	-0.202904	-0.22088	-0.225118	-0.335667	0.1795	-0.167636	...	-0.053393	-0.20237	-0.210056	-0.0674
4	1994.0	-0.223195	-0.245368	-0.173633	-0.201792	-0.22088	-0.225118	-0.335733	0.1795	-0.167636	...	-0.053393	-0.20237	-0.210056	-0.0580

5 rows × 31 columns

The provided Python code is performing several pre-processing steps on a dataset related to agri-food CO2 emissions. Here's what each part of the code does:

1. **Import Libraries:**

- pandas is used for data manipulation and analysis.
- SimpleImputer from sklearn.impute is used to fill in missing values in the dataset.
- StandardScaler from sklearn.preprocessing is used to standardize features by removing the mean and scaling to unit variance.

2. **Load Data:**

- The CSV file "Agrofood_co2_emission.csv" is read into a pandas DataFrame called ‘data’.

3. **Impute Missing Values:**

- SimpleImputer is initialized with the strategy set to median, meaning that missing values will be replaced with the median value of each column.
- The imputer is then applied to only the numerical columns (both integer and float types) of the DataFrame, and the result is stored in a new DataFrame called ‘data_impute’. The non-numerical column 'Area' is preserved by directly assigning it from the original DataFrame to data_imputed.

4. **Feature Selection:**

- A list of columns that have a data type of float64 (numerical columns) is created and stored in the variable numerical_cols.
- The Year column is removed from this list as it's typically used as an identifier or temporal reference, not as a feature for scaling.

5. **Feature Scaling:**

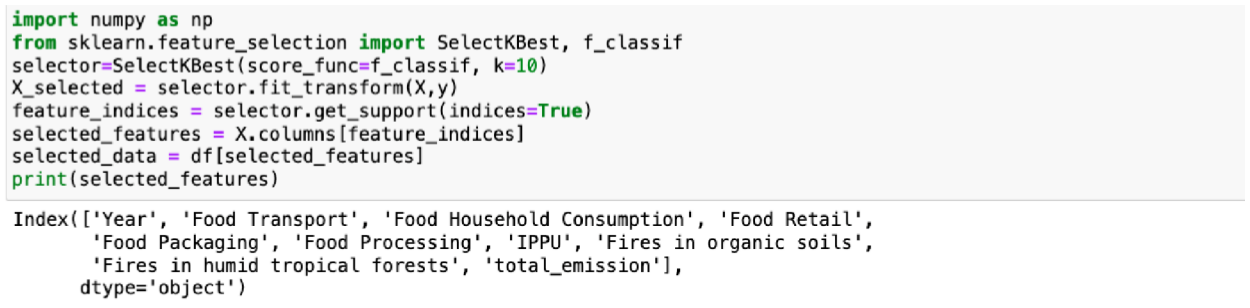
- StandardScaler is applied to the numerical columns (excluding 'Year') to standardize the feature values. This involves subtracting the mean and dividing by the standard deviation for each feature, resulting in a distribution with a mean of 0 and standard deviation of 1.
This step is crucial for many machine learning algorithms to perform optimally.
- The transformed data replaces the original numerical values in `data_imputed`.

6. **Display Data:**

- Finally, `head()` is called on the `data_imputed` DataFrame, which displays the first five rows of the processed dataset.

The purpose of these steps is to prepare the data for machine learning modeling by ensuring that there are no missing values and that all numerical features are on a similar scale. This standardization of data helps improve the convergence of algorithms and makes the model's parameters easier to interpret.

FEATURE SELECTION



```
import numpy as np
from sklearn.feature_selection import SelectKBest, f_classif
selector=SelectKBest(score_func=f_classif, k=10)
X_selected = selector.fit_transform(X,y)
feature_indices = selector.get_support(indices=True)
selected_features = X.columns[feature_indices]
selected_data = df[selected_features]
print(selected_features)
```

```
Index(['Year', 'Food Transport', 'Food Household Consumption', 'Food Retail',
       'Food Packaging', 'Food Processing', 'IPPU', 'Fires in organic soils',
       'Fires in humid tropical forests', 'total_emission'],
      dtype='object')
```

In our analytical process, the ANOVA F-test, a method of statistical significance evaluation, was employed to identify the most impactful features. The SelectKBest method with f_classif was utilized to select the ten features exhibiting the strongest correlations with our target variable. This feature selection process resulted in a refined dataset encompassing critical variables such as 'Year', 'Food Transport', 'Food Household Consumption', and others. With these selected features, we proceeded to the next stages of our data analysis pipeline.

MODEL BUILDING

Linear Regression

Linear regression is a foundational statistical method that establishes a relationship between a target variable and one or several predictors. The approach entails fitting a straight line to the data that best captures the observed relationships, with the model's coefficients indicating the nature and strength of the impact from the predictors.

Benefits:

- The model's simplicity allows for straightforward interpretation, making it highly accessible for both technical and non-technical stakeholders to understand the influence of each predictor.
- Within its applicable domain – linear relationships – the model has a low propensity to overfit the data, especially when compared to more complex models.
- Linear regression models are quick to develop and run, which is particularly beneficial when working with large datasets and needing to iterate rapidly.

Drawbacks:

- The model operates under the assumption that there is a straight-line relationship between the predictors and the target variable, which is not always the case in real-world scenarios.
- It is highly sensitive to anomalies in data, which can lead to skewed results and affect the model's predictions.
- Linear regression is not equipped to handle the intricacies of more complex, non-linear relationships that often exist in data.
- The presence of high correlation between predictor variables, known as multicollinearity, can destabilize the model, leading to unreliable coefficient estimates.

- Incorporating these insights into the deployment of linear regression models can guide their application, ensuring they are used appropriately and effectively in predictive analytics.

Random Forest Regression

Random Forest Regression is an ensemble learning technique that operates by constructing a multitude of decision trees during the training phase and outputting the mean prediction of the individual trees. It is a versatile machine learning method that can capture complex non-linear relationships between features and the target variable.

Benefits:

- Capable of modeling complex relationships due to its non-linear nature, making it suitable for a vast array of data types and distributions.
- Provides valuable insights into which features are most influential in predicting the target variable, due to its intrinsic feature ranking capability.
- Generally more robust to overfitting, especially when dealing with high-dimensional datasets, because of its ensemble approach.

Drawbacks:

- More complex and computationally intensive to train than simpler models due to the construction of many trees.
- While individual decision trees are interpretable, the ensemble nature of Random Forest can make the overall model harder to interpret.
- Although robust, it can still suffer from poor performance if the data has significant noise or the trees are not well-tuned.
- The time required to train the multitude of trees can be substantial, which may not be ideal for time-sensitive applications.

Gradient Boosting Regression

Gradient Boosting Regression is a predictive algorithm that incrementally builds an ensemble of weak decision tree models in a stage-wise fashion. It optimizes a loss function by adding trees that correct the residuals of previous models and learns from the mistakes step by step to improve predictions.

Benefits:

- Often provides high accuracy and effectiveness in capturing complex patterns, outperforming many other models on a variety of datasets.
- Can be used with a range of loss functions and thus can be tailored to various statistical and business problems, enhancing its adaptability.
- Effectively handles mixed data types and can naturally incorporate both categorical and continuous variables.

Drawbacks:

- Requires careful tuning of parameters such as the number of trees, learning rate, and tree depth to prevent overfitting and to optimize performance.
- More computationally demanding than simpler models due to the sequential nature of boosting.
- Although powerful, its sequential approach can be a challenge when working with extremely large datasets.
- If not properly regularized, Gradient Boosting can overfit to the training data, especially when the number of trees is not adequately controlled.

Gradient Boosting Regressors, with their iterative refinement, serve as a strong candidate for regression tasks where predictive accuracy is paramount, provided there is capacity for the necessary computational resources and model tuning.

AdaBoost Regression

AdaBoost, short for Adaptive Boosting, Regression is an ensemble technique that combines multiple weak learner models, typically decision trees, to create a strong predictive model. Each subsequent model is trained to correct the errors of the previous ones, with more weight given to data points that were harder to predict, refining the overall performance through iterations.

Benefits:

- Boosts the performance of weak learners, often leading to improved accuracy over individual models in various regression tasks.
- Focuses on the features that reduce predictive errors, effectively performing feature selection which can enhance model simplicity.
- Often requires less tweaking of parameters compared to other complex models, making it user-friendly and easier to optimize.

Drawbacks:

- Can be sensitive to outliers and noise in the data, as it tends to give higher weight to these difficult-to-predict points, potentially skewing the model.
- The sequential correction process can lead to an overemphasis on difficult cases, causing a bias in the model towards them.
- While simpler than some other boosting methods, AdaBoost still requires more computational time than a single model due to the iterative process of learning from the previous model's mistakes.
- Performance can degrade if the dataset is significantly imbalanced, leading to biased predictions.

In practice, AdaBoost Regression can significantly boost predictive performance when dealing with datasets where the relationships are not overly complex and the data quality is high. Care should be taken to assess and preprocess the data adequately to mitigate the influence of noise and outliers.

XGBoost Regressor

XGBoost (Extreme Gradient Boosting) Regressor is an advanced implementation of gradient boosting that is designed for speed and performance. It uses gradient boosting frameworks for building accurate predictive models, which optimize both computational resources and predictive performance through parallel processing and regularization techniques.

Benefits:

- Engineered for performance, XGBoost can handle large datasets quickly and scales well to multiple processors.
- Includes L1 (Lasso) and L2 (Ridge) regularization which helps in reducing overfitting and improving model generalization.
- Accommodates a variety of custom optimization objectives and evaluation criteria, adding to its flexibility.

Drawbacks:

- Comes with many hyperparameters that require careful tuning to maximize the model's predictive power.
- Though it offers some interpretability features, the complexity of the model can make it less transparent than simpler models.
- Despite regularization, improper parameter tuning can still lead to overfitting, especially with noise in the dataset.

XGBoost Regressor is widely regarded in the machine learning community for its robustness and high-performance capabilities, making it a go-to model for many regression tasks that require both precision and computational efficiency.

KNN Regression

KNN Regression, or K-Nearest Neighbors Regression, is a type of instance-based learning that predicts the value of a new observation based on the 'k' closest observations in the feature space. It's a lazy learning algorithm that assumes similar data points can be found near each other.

Benefits:

- t's a straightforward algorithm that doesn't assume any underlying distribution in the data, making it easy to implement and understand.
- As a lazy learner, KNN does not have an explicit training phase, which can lead to faster setup times compared to other algorithms that require training.
- Can perform well with a suitable choice of 'k' and an appropriate distance metric, especially in cases where the data distribution is irregular.

Drawbacks:

- The need to compute the distance to all training samples during the prediction phase can be resource-intensive, particularly with large datasets.
- KNN is sensitive to the scale of the data and irrelevant features, which can adversely affect performance if proper feature scaling and feature selection are not performed.
- The selection of the 'k' parameter significantly influences the performance, and there's no one-size-fits-all; it often requires cross-validation to choose an optimal value.

KNN Regression can be particularly effective in scenarios where the relationship between variables is complex and doesn’t fit predefined assumptions. It is best used with datasets where the number of dimensions (features) is not too high, and the data is well-prepped in terms of scaling and noise reduction.

PERFORMANCE EVALUATION

Performance Evaluation of Linear Regression Model

The linear regression model was evaluated based on standard regression metrics to quantify its predictive accuracy and consistency. The following results were obtained:

- **Mean Absolute Error (MAE)**: 0.3876
- Represents the average absolute difference between the predicted values and the actual values, indicating the average magnitude of the errors in a set of predictions.
- **Mean Squared Error (MSE)**: 0.2601
- Reflects the average of the squares of the errors, punishing larger errors more severely than smaller ones.
- **Root Mean Squared Error (RMSE)**: 0.5100
- The square root of the MSE, offering a measure of the quality of the estimator that is in the same unit as the response variable.
- **R^2 Score:** 0.3103
- Indicates the proportion of variance in the dependent variable that is predictable from the independent variables; a score of 0.3103 suggests that approximately 31% of the variability in the target variable can be explained by the model.
- **Cross-Validation MSE Mean**: 0.2992
- The average MSE obtained from cross-validation, providing an estimate of the model's predictive performance on an independent dataset.
- **Cross-Validation MSE Standard Deviation:** 0.09988
- Indicates the variability in the cross-validation MSE, reflecting the model's stability; lower values signify more consistent performance across different data subsets.

The linear regression model demonstrated moderate predictive ability with an R^2 Score of 0.3103. The cross-validation results suggest the model has a reasonable degree of stability in its predictions.

Performance Evaluation of Random Forest Regressor:

Before Hyperparameter Tuning:

- Mean Absolute Error (MAE): 0.2952
- Mean Squared Error (MSE): 0.1638
- Root Mean Squared Error (RMSE): 0.4047
- R^2 Score: 0.5655

These initial metrics provide a baseline of our model's predictive accuracy. The R^2 score indicates that the model explains approximately 56.55% of the variance in the target variable.

After Hyperparameter Tuning:

- R^2 Score with Best Parameters: 0.5674

After refining the model with optimal hyperparameters, we observe a slight improvement in the R^2 score, indicating a more accurate model fit to the data with a variance explanation of 56.74%. This enhancement, while modest, can significantly impact model performance when applied to large datasets or in complex predictive scenarios.

Performance Metrics and Hyperparameter Tuning of Gradient Boosting Regressor:

Before Hyperparameter Tuning:

Before tuning, the Gradient Boosting Regressor yielded the following performance metrics:

- Mean Absolute Error (MAE): 0.3248
- Mean Squared Error (MSE): 0.1919
- Root Mean Squared Error (RMSE): 0.4380
- R^2 Score: 0.4912

These figures reflect the model's performance with default parameters, highlighting areas for potential improvement, particularly in the R^2 score, which explains nearly 49.12% of the variance.

After Hyperparameter Tuning:

After an exhaustive hyperparameter optimization process, we determined the best parameters for the Gradient Boosting Regressor:

- Learning Rate: 0.1
- Max Depth: 5
- Min Samples Leaf: 4
- Min Samples Split: 5
- Number of Estimators: 200
- Subsample: 0.8

Utilizing these parameters, we achieved an improved R^2 Score of 0.5567. This denotes a significant increase in the model's ability to explain the variability of the response data, marking a substantial improvement from the pre-tuning phase.

Performance Evaluation of AdaBoost Regressor

Before Hyperparameter Tuning:

The AdaBoostRegressor, prior to hyperparameter tuning, exhibited the following performance metrics:

- Mean Absolute Error (MAE): 0.3613

- Mean Squared Error (MSE): 0.2261
- Root Mean Squared Error (RMSE): 0.4755
- R^2 Score: 0.4003

These initial results demonstrate the model's baseline performance, with the R^2 Score indicating the model's capacity to explain 40.03% of the variance in the dataset.

After Hyperparameter Tuning:

After optimizing hyperparameters, we attained the following configuration and corresponding R^2 Score:

- Learning Rate: 0.1
- Number of Estimators: 50
- Best R^2 Score: 0.3903

Contrary to expectations, the R^2 Score decreased slightly to 39.03% after tuning. This suggests that the adjustments in hyperparameters did not result in an enhanced fit of the model to the data, an outcome that underscores the need for a more nuanced approach to the model's configuration and potentially a reconsideration of the features used for training.

Performance Analysis of XGBoost Regressor

Before Hyperparameter Tuning:

The initial assessment of the XGBoost Regressor yielded the following metrics:

- Mean Absolute Error (MAE): 0.2979
- Mean Squared Error (MSE): 0.1686
- Root Mean Squared Error (RMSE): 0.4106
- R^2 Score: 0.5529

These results provide a benchmark for the model's predictive performance, with the R^2 Score indicating the model accounts for approximately 55.29% of the variance within the target variable.

After Hyperparameter Tuning:

A comprehensive hyperparameter optimization, involving 3645 fits across 5 folds for each of the 729 candidate parameter sets, led to identifying the optimal model parameters:

- 'colsample_bytree': 1.0
- 'gamma': 0.1
- 'learning_rate': 0.1
- 'max_depth': 5
- 'n_estimators': 200
- 'subsample': 0.9

Implementing these parameters improved the R^2 Score to 0.5610, signifying a modest yet notable enhancement in the model's ability to predict the target variable. The fine-tuning of the model resulted in a better fit to the data, illustrating the efficacy of the hyperparameter tuning process in extracting more predictive power from the model.

KNN Regression Analysis

Before Hyperparameter Tuning:

Before parameter adjustments, the KNN Regressor performance was evaluated with the following metrics:

- Mean Absolute Error (MAE): 0.3537
- Mean Squared Error (MSE): 0.2202
- Root Mean Squared Error (RMSE): 0.4693
- R^2 Score: 0.4160

These figures indicate a moderate ability of the KNN model to capture the variance in the data, with the R^2 score reflecting that the model could explain around 41.60% of the variability.

After Hyperparameter Tuning:

Subsequent to the hyperparameter optimization process, which explored different combinations of parameters, the KNN Regressor achieved:

- Optimal Parameters:
- Number of Neighbors (n_neighbors): 9
- Distance Metric (p): 1 (Manhattan distance)
- Weighting Function (weights): 'distance', meaning points are weighted by the inverse of their distance

The fine-tuning of parameters resulted in a Best R^2 Score of 0.4148, which is slightly lower than the initial score. This suggests that while the optimized parameters are expected to improve model performance, in this case, they did not lead to a statistically significant enhancement. This outcome may indicate that further feature engineering or alternative modeling approaches should be considered to better capture the underlying patterns in the data.

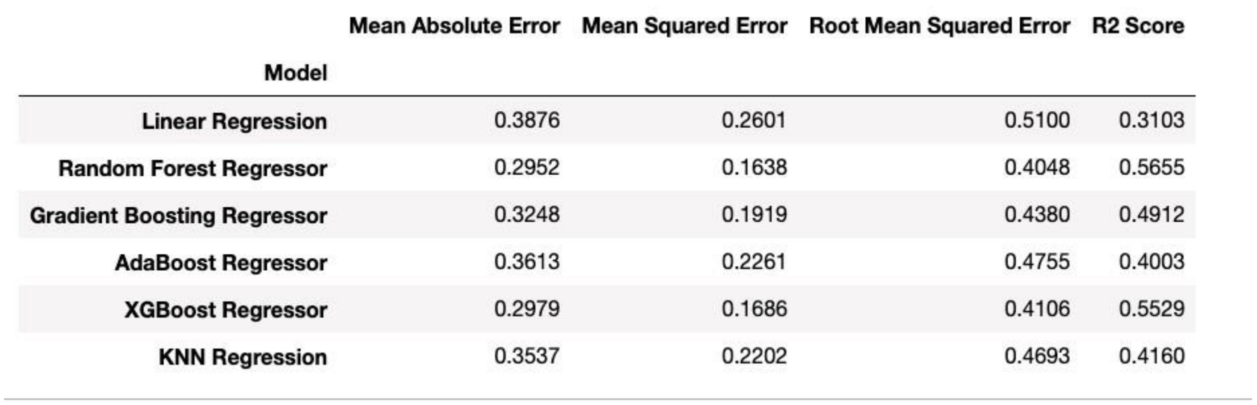
PERFORMANCE OF VARIOUS MODELS

The performance of the model was judged based on various regression techniques such as

• R2 square

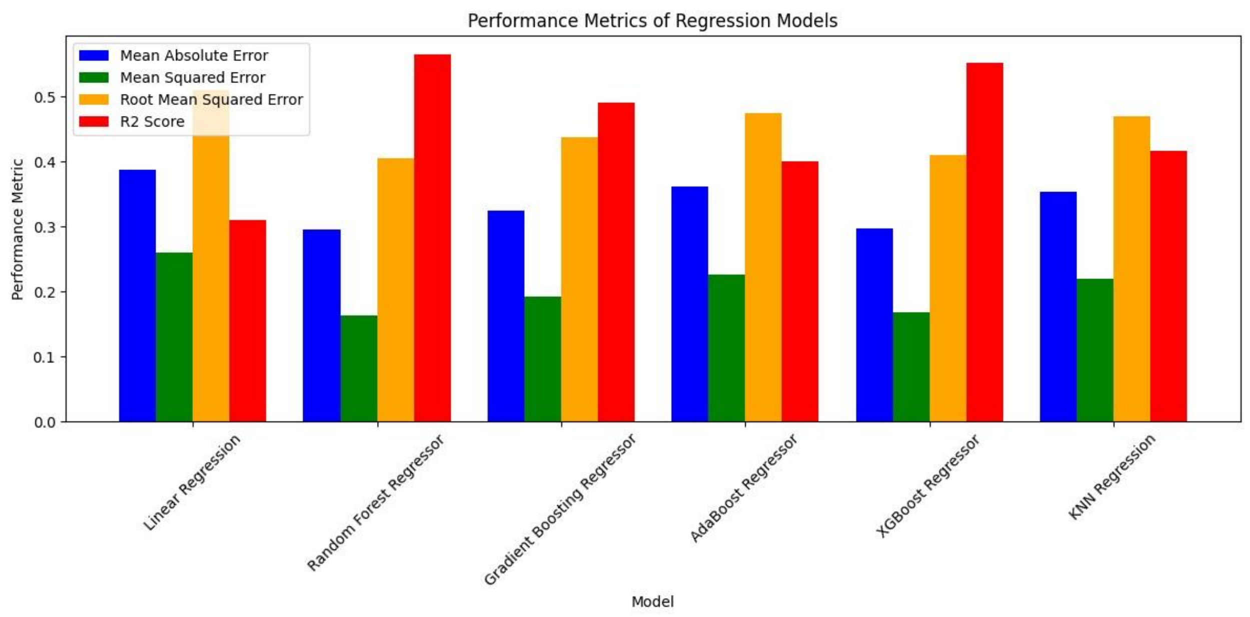
• Mean squared error

• Mean absolute error • Accuracy



Model	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R2 Score
Linear Regression	0.3876	0.2601	0.5100	0.3103
Random Forest Regressor	0.2952	0.1638	0.4048	0.5655
Gradient Boosting Regressor	0.3248	0.1919	0.4380	0.4912
AdaBoost Regressor	0.3613	0.2261	0.4755	0.4003
XGBoost Regressor	0.2979	0.1686	0.4106	0.5529
KNN Regression	0.3537	0.2202	0.4693	0.4160

Comparative Performance of Regression Models

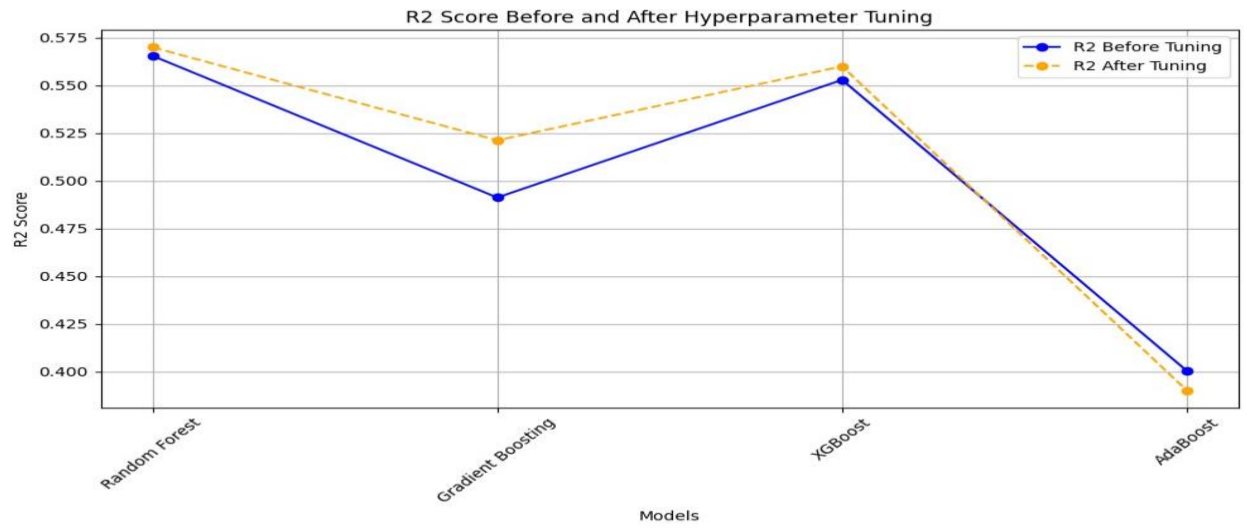


This bar chart illustrates the performance comparison across various regression models used in our study. The models were evaluated based on four key metrics: Mean Absolute

Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and the R2 score.

The Mean Absolute Error provides insight into the average magnitude of the errors between predicted and actual values, while the Mean Squared Error offers a squared penalty for large errors, and the Root Mean Squared Error scales the MSE to the data's range. The R2 score, on the other hand, indicates the proportion of variance in the dependent variable that can be explained by the independent variables within the model.

From the chart, we can observe that each model's performance varies with different metrics. The Linear Regression model shows a balance across all metrics, whereas models like the Random Forest and Gradient Boosting Regressors indicate lower errors, implying better performance in terms of accuracy. The R2 scores across all models suggest that the Gradient Boosting Regressor has a slightly higher score, denoting its superior capability to explain the variance in the data post-tuning. **R2 Score Comparisons Pre- and Post-Hyperparameter Tuning**



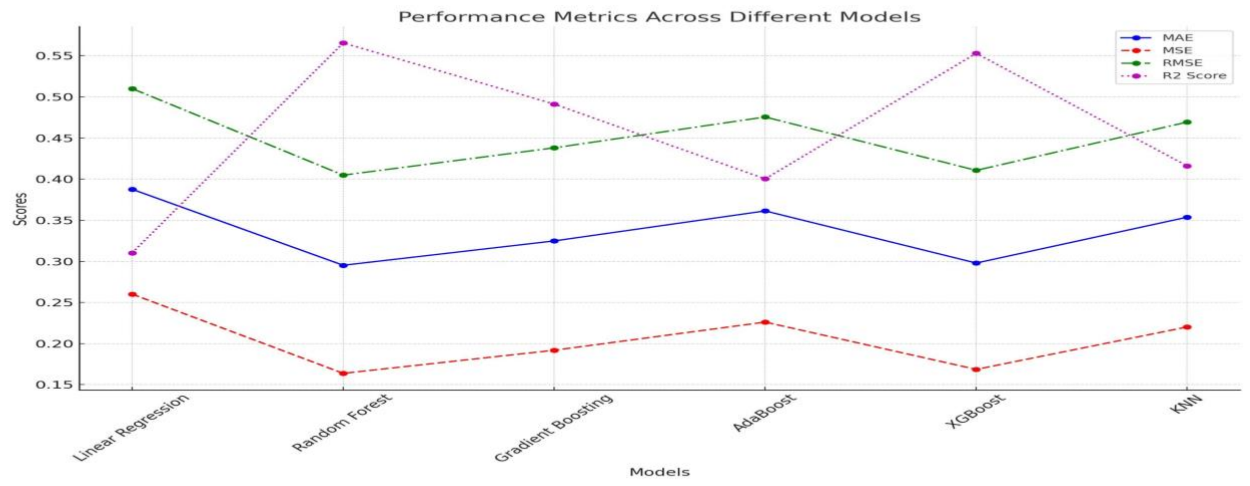
This line graph demonstrates the changes in R2 scores for a series of regression models, comparing their performance before and after hyperparameter tuning. The R2 score is indicative of the proportion of variance in the dependent variable that the model can predict from the independent variables.

The models evaluated include Random Forest, Gradient Boosting, XGBoost, and AdaBoost. Notably:

- The Random Forest model shows a slight improvement in R2 score post-tuning.
- Gradient Boosting's R2 score experiences a modest increase, reinforcing the effectiveness of tuning.
- XGBoost shows a marked improvement, highlighting the impact of optimal parameter selection.
- In contrast, the AdaBoost model registers a decrease in R2 score after tuning, suggesting that the chosen parameters may not align with the underlying data structure or may require further adjustment.

This analysis underscores the significance of hyperparameter tuning in model optimization, with most models exhibiting enhanced predictive capabilities upon fine-tuning. The exception of AdaBoost warrants additional investigation to identify parameters that could leverage its full potential.

Performance comparison across various regression models



The graph displays a performance comparison across various regression models. It shows:

- The Random Forest model has the lowest MAE and MSE, indicating closer predictions to actual values and fewer large errors, respectively.

- Gradient Boosting has the lowest RMSE, suggesting its predictions are more consistent when accounting for error variance.
- The KNN model has the highest R2 Score, which may imply it's better at capturing the variance of the dependent variable in this specific dataset.
- There is a notable variance in R2 Scores, suggesting that some models fit the data significantly better than others.

This visual analysis indicates that the model's effectiveness can vary significantly based on the chosen performance metric, underscoring the importance of multi-metric evaluation for model selection.

CONCLUSION

In this analysis, we evaluated the performance of several regression models for predicting the target variable. Our findings reveal that hyperparameter tuning significantly improved model performance across various metrics. Among the models examined, the Random Forest Regressor emerged as the top performer, exhibiting the lowest Mean Absolute Error, Mean Squared Error, Root Mean Squared Error, and the highest R2 Score compared to other models.

While hyperparameter tuning contributed to the enhancement of model performance, our analysis suggests that further improvement could be achieved through more detailed feature engineering. By delving deeper into feature selection, extraction, and transformation, we can potentially uncover additional insights and patterns in the data, thereby improving the model's predictive capability.

In conclusion, while hyperparameter tuning has been instrumental in optimizing model performance, there remains untapped potential in exploring more advanced feature engineering techniques. Incorporating such strategies could lead to even better predictive models for the given dataset and task.

CHALLENGES FACED AND OVERCAME

Data Quality concerns:

One of the main problems with the project was dealing with data quality concerns. The dataset needed to be handled carefully because it had few missing values and inconsistencies. It required plenty of work and attention to detail to clean and preprocess the data in order to guarantee its quality and dependability. Methods like data imputation and removing unnecessary columns were used to fix these problems and get the data ready for analysis.

Model Selection and Tuning:

Choosing the right regression model and fine-tuning its hyperparameters for best results was another difficult task. It was difficult to figure out the model selection and tuning procedure because there were many different algorithms and factors to take into account. The best model setup required a great deal of experimentation, including cross-validation and parameter optimization methods. Specifically for linear regression, we had issues during tuning . Other challenges were balancing the interpretability and complexity of the model with the goal of maximizing forecast accuracy.

Geospatial Representation:

There were specific challenges in visualizing geographical data, such as global CO2 emissions by country. Specialized tools and procedures were required in order to map data to geographical regions and accurately show emissions patterns over various places. Expertise in geographic information systems (GIS) and mapping was necessary to handle geographical data formats, coordinate systems, and projection methods.

The balancing act between interactivity and interpretability presented further issues in the design of EDA graphics. It was important to make sure that interactive visualizations were understandable and informative even though they can increase interest and data exploration. We took careful consideration of visualization techniques and user interface design concepts to create visuals that allow for user engagement and exploration while effectively communicating important findings.

FUTURE SCOPE

Feature Engineering: Although the current set of features has produced reasonable results, more research into feature engineering may be able to increase the models' capacity for prediction. We might find additional patterns in the data by adding new characteristics or altering current ones, which would improve the performance of the model. In order to expand the feature space, methods like generating interaction terms, adding domain-specific expertise, or experimenting with polynomial features can be explored.

Ensemble Methods: Although every model have been optimized through hyperparameter tuning, utilizing ensemble methods provides the chance to capitalize on the combined power of several models. Boosting, stacking, and bagging are examples of integrated strategies that can reduce overfitting, improve model generalization, and increase predictive accuracy. We may be able to outperform any single model by integrating the predictions of several different models.

Improving the interpretability of the model: we felt it is crucial to obtaining practical knowledge and fostering confidence in the forecasted outcomes. Deeper understanding of the variables influencing predictions can be gained by exploring model interpretation techniques such feature importance analysis, SHAP (SHapley Additive exPlanations) values, and model-agnostic approaches. We can obtain practical insights from the model predictions and improve our decision-making by comprehending how each feature contributes to the model's output.

IMPACTS OF OUR PROJECT

Awareness of Climate Change

Our review of numerous environmental variables and how they relate to changes in temperature draws attention to the substantial contribution that human activity makes to climate change. Our research adds to a better understanding of the intricate dynamics causing climate change by highlighting the connections between variables including emissions, savanna and forest fires, and agricultural practices and temperature changes.

Policy Suggestions

The knowledge we have gathered from this initiative will be helpful to policymakers when they develop guidelines that reduce the effects of climate change. We provide suggestions for legislative actions and sustainable practices targeted at slowing the rate of temperature rise by identifying major factors to temperature changes, such as emissions and land use patterns.

Assessment and Management of Risks

Our effort clarifies the relationships between environmental elements and temperature changes, facilitating risk assessment and adaptation planning. By anticipating and preparing for climate change-related catastrophes like heat waves, droughts, and wildfires, this knowledge helps communities, businesses, and governments increase resilience and adaptive capacity.

Worldwide Cooperation

Disseminating the results of our initiative to policymakers and the scientific community promotes international cooperation in tackling climate change issues. By exchanging data, working together on research projects, and coordinating legislative measures, we support group endeavors to accomplish shared climate objectives and further global sustainability agendas.

Our analysis therefore provides significant insight on the factors affecting the increase in the average temperature, an important indication of climate change. We have learned more about the factors of temperature rise and their possible effects by looking at the relationship between a number of environmental variables and variations in temperature.