Nairobi Data Journey

Theme: create machine learning and deep learning models that use open-source CO2 emissions data (from satellite observations) to predict carbon emissions.

Outline

1. Environment Preparation and Loading Data
2. EDA
3. Feature Engineering
4. Model Building(LGBM)
5. Ensembling with XGB Boost Model
6. Key Figures

Setting up environment

1.Loaded the necessary libraries
(Pandas, numpy, matplotlib, seaborn,etc)

2. Loaded data(Kaggle notebook)

Train set
74005,76
Test set
28035,75

EDA

- **Checked for nulls**

Both has some nulls-filled them with the mean of the specific columns

- **Checked for duplicates**

No duplicates

- **Converted the data types**

Memory optimization-By converting data types to more memory-efficient ones, such as converting float64 to float32 or int64 to int32, the code reduces the memory footprint of the DataFrame.

Feature Engineering

1. **Computed new features** capturing the relationships between the features and the target variable for both the train and test Datasets.
2. New features for the train and test datasets were created
3. **Dropped features** that had a negative **correlation(35 columns)** with the target variable(angle, altitude, height, depth)

Identified the most important features that had the most significant impact on the model's output (to achieve parsimony)

Data Preprocessing: combined the train and test datasets, rounded the latitude and longitude coordinates to **reduce precision**, and creates a location column by combining latitude and longitude values.

Feature Engineering: feature engineering techniques to **capture underlying patterns and relationships** in the data. Trigonometric features are extracted for the month column using sine and cosine transformers. Periodic spline-based features are also created to represent cyclic patterns in the data-inspired by winning notebook in umojahack(2023)

Rolling Statistics: calculated rolling/moving statistics for selected columns, including the rolling mean, maximum, minimum, sum, standard deviation, and skewness. These rolling statistics capture trends and patterns over specific rolling windows, providing additional information about the data dynamics.

Location-based Statistics: calculated location-based statistics for selected columns, such as mean, standard deviation, minimum, maximum, and skewness. These statistics **provide insights into how the data varies across different locations**.

Encoding: The code uses label encoding to convert the 'location' column into numerical values. This enables the model to handle categorical data during the modeling phase.

Train-Test Split: The code separates the processed dataset back into train and test sets based on the 'ID_LAT_LON_YEAR_WEEK' column.
Resulting dataset shape
Train(74005,710)
Test set(28035,710)

Model Building

Model Training and Evaluation: process of training and evaluating a machine learning model using **LightGBM (LGBM**). LGBMRegressor is used to fit the model to the training data and evaluate its performance on the test data.

Model Parameters: The parameters, such as learning_rate, subsample, colsample_bytree, max_depth, and objective, are essential hyperparameters that control the behavior and performance of the model.(Improved after using GridSearchCV)

Cross-Validation: Employed StratifiedKFold, a cross-validation technique, to split the dataset into multiple folds and ensure balanced distribution across the folds. It iterates over each fold, training a separate model and evaluating its performance.

Early Stopping: Utilized early stopping during model training by setting the early_stopping_rounds parameter in the fit() method. This technique **helps prevent overfitting and improves efficiency** by stopping the training process if the model's performance on the validation set does not improve for a certain number of iterations.

Performance Evaluation: The code calculates the **root mean squared error (RMSE**) between the predicted and actual target values using mean_squared_error(). This metric is a common measure of regression model accuracy, and a lower RMSE indicates better predictive performance.

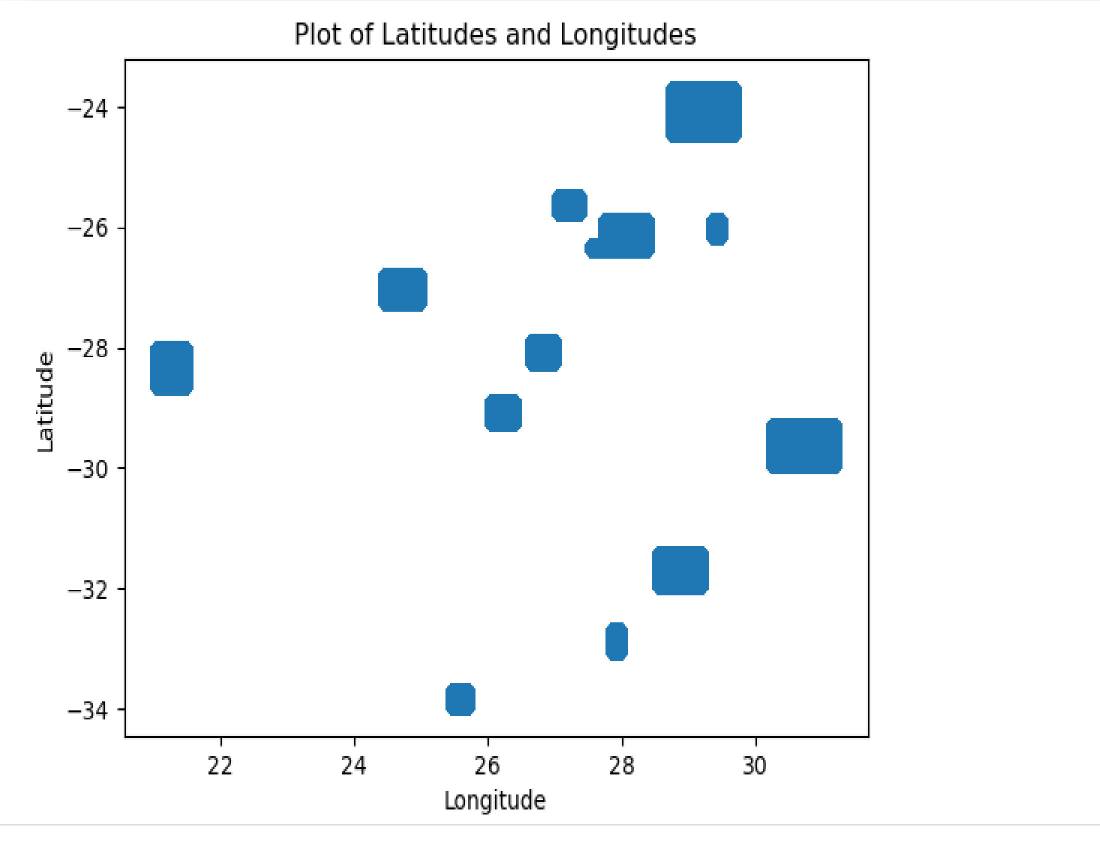
Out-of-Fold (OOF) Predictions: Stored the RMSE values for each fold in the oof_pred list and the predictions for the test set in the fold_pred list.

Performance Summary: The code calculates the mean of the RMSE values from each fold, providing an overall performance summary of the model across all folds.

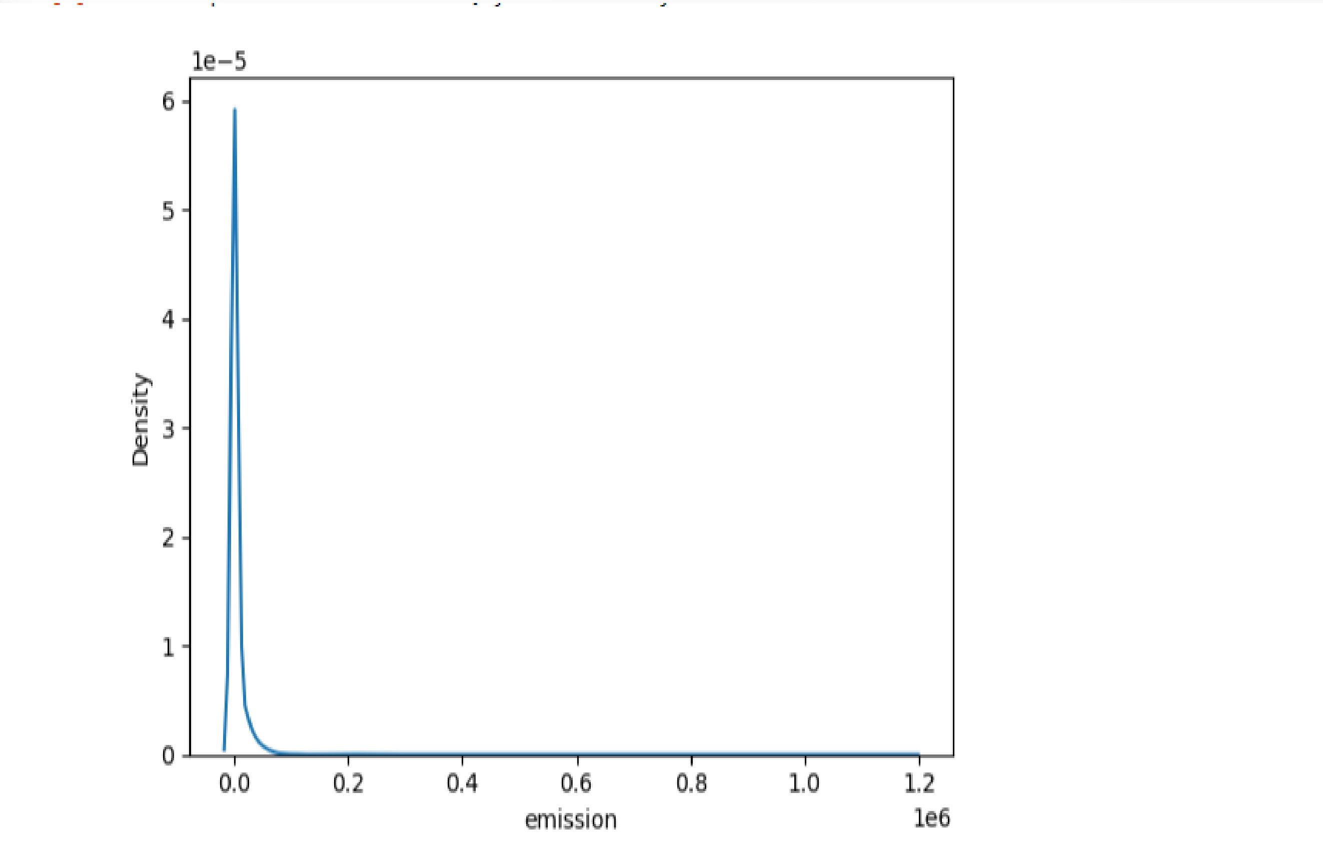
Xgb boost Model

- Defined the model
- Fit the model
- Made Predictions
- Did Simple averaging
- Evaluation metric-MSE
- Made submission

Figures



Plot of Latitudes and Longitudes



THANK YOU

The end!!