

**Inteligencia Artificial**

**Informe**

**Entrega Final**



**Therry Jones Bent O'Neill**

**Julian Mateo Mena Urrego**

**Miguel Angel Rivera Florez**

**Universidad de Antioquia**

**Facultad de Ingeniería**

**Departamento de Ingeniería Eléctrica**

**Medellín**

**2023-2**

## Introducción

En este proyecto, se abordó la tarea de predecir las emisiones de CO2 en Rwanda utilizando un conjunto de datos que incluye información sobre diversas características ambientales de las emisiones que se dan que fueron medidas por el satélite Sentinel-5P. Al utilizar la métrica de Root Mean Squared Error durante la implementación del modelo de predicción fue muy fundamental para evaluar su rendimiento. Este indicador proporcionó una medida cuantitativa de la discrepancia entre las predicciones del modelo y los valores reales. RMSE fue esencial para evaluar la eficacia del modelo de predicción, ya que también proporciona una interpretación intuitiva de la magnitud de los errores, cuanto menor sea el valor de RMSE, más precisa será la capacidad del modelo para prever los valores reales. Esto fue muy relevante en cuestión de la regresión.

## Descripción del progreso alcanzado

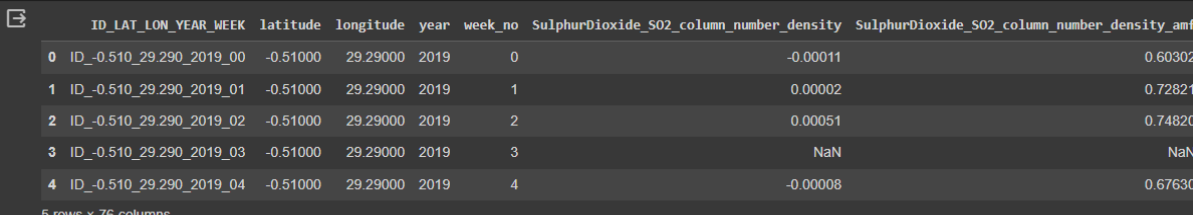
### Preprocesamiento del dataset.

El dataset “Predict CO2 Emissions in Rwanda” está compuesto de los siguientes archivos:

- sample\_submission.csv
- test.csv
- train.csv

Antes de realizar modelos predictivos es necesario el preprocesamiento de los datos. Preprocesar los datos se trata de cualquier tipo de procesamiento que se realiza con los datos brutos para transformarlos en datos que tengan formatos que sean más fáciles de utilizar.

Para el primer paso en el preprocesamiento se tomaron los datos presentes en el archivo “train.csv” que está compuesto de 76 columnas y 5 filas. La primera columna es el ID con un prefijo de latitud, longitud, año y semana\_no, Cada fila del train contiene cuatro columnas de índice (latitud, longitud, año y semana\_no), y las otras son 70 características y la emisión que es el objetivo.



	ID_LAT_LON_YEAR_WEEK	latitude	longitude	year	week_no	SulphurDioxide_S02_column_number_density	SulphurDioxide_S02_column_number_density_amf
0	ID_-0.510_29.290_2019_00	-0.51000	29.29000	2019	0	-0.00011	0.60302
1	ID_-0.510_29.290_2019_01	-0.51000	29.29000	2019	1	0.00002	0.72821
2	ID_-0.510_29.290_2019_02	-0.51000	29.29000	2019	2	0.00051	0.74820
3	ID_-0.510_29.290_2019_03	-0.51000	29.29000	2019	3	NaN	NaN
4	ID_-0.510_29.290_2019_04	-0.51000	29.29000	2019	4	-0.00008	0.67630

5 rows x 76 columns

**Figura 1. Composición del archivo “train.csv”**

Las 70 características vienen en 8 grupos distintos tamaños las cuales son: Dióxido de azufre, Monóxido de carbono, Dióxido de nitrógeno Formaldehído, Índice de aerosol UV, Ozono, Nube.

El grupo de características contienen diferentes sub características, pero en cada grupo aparecen cuatro sub características:

- 'sensor\_azimuth\_angle',
- 'sensor\_zenith\_angle',
- ángulo azimut solar,
- ángulo cenital solar",

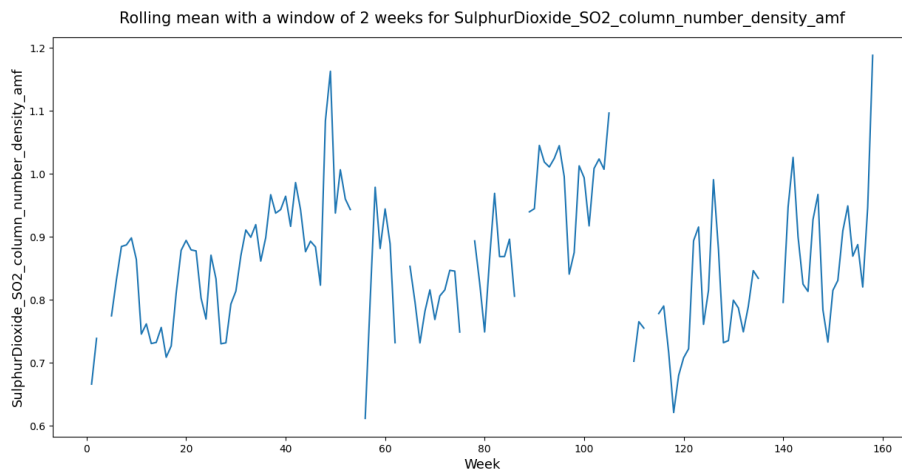
Las subcaracterísticas son necesarias para interpretar las mediciones. Como los datos fueron medidos por un satélite que no está directamente por encima del lugar medido, el ángulo del satélite afecta a la medición. Además, el ángulo del sol y las nubes influyen en las mediciones.

Cabe resaltar que el satélite mide la concentración de varios gases en la atmósfera, pero no mide el CO2. Nuestro objetivo consiste en predecir la emisión de CO2 para cada lugar y punto en el tiempo basándonos en la concentración de los demás gases. No sabemos cómo se miden las emisiones de CO2. El satélite Sentinel-5P no las mide.

Para analizar los datos del train, filtramos una ubicación y graficamos la media correspondiente a las emisiones.

```
1 # 1. Crear una ubicación única desde latitud y longitud
2 train['location'] = train['latitude'].round(2).astype(str) + '_' + train['longitude'].round(2).astype(str)
3
4 # 2. Filtrar el conjunto de datos para la ubicación deseada
5 example_loc = train[train['location'] == '-0.51_29.29']
6
7 # 3. Calcular la media móvil con una ventana de 2 semanas
8 rolling_mean = example_loc['SulphurDioxide_SO2_column_number_density_amf'].rolling(window=2).mean()
9
10 # 4. Visualizar los resultados
11 plt.figure(figsize=(15, 7))
12 rolling_mean.plot()
13 plt.title('Rolling mean with a window of 2 weeks for SulphurDioxide_SO2_column_number_density_amf', y=1.02, fontsize=15)
14 plt.xlabel('Week', y=1.05, fontsize=13)
15 plt.ylabel('SulphurDioxide_SO2_column_number_density_amf', x=1.05, fontsize=13)
16 plt.show()
```

**Figura 2. filtro de ubicación.**



**Figura 3. Gráfica de la media.**

observamos que hay datos faltantes por qué la gráfica no presenta continuidad y hay espacios entre partes de la gráfica.

### Completación de datos

Excepto las columnas de índice (latitude, longitude, year and week\_no) y emisión, en todas las columnas faltan valores, se puede asumir que faltan valores si durante toda una semana el satélite nunca pudo realizar una medición fiable de un lugar.

Como faltan datos en el train y en el test, no podemos simplemente eliminar las filas con datos que faltan, sino que necesitamos atribuir los valores que faltan por lo cual usamos las mediciones de lugares cercanos para atribuir los valores que faltan.

```
4 with pd.option_context("display.min_rows", 14):
5     # Mostrar la suma de valores faltantes ordenados
6     display(train.isna().sum().sort_values())
7
8 # Dejar un espacio en blanco entre las visualizaciones
9 print()
10
11 # Para el conjunto de prueba (test)
12 with pd.option_context("display.min_rows", 14):
13     # Mostrar la suma de valores faltantes ordenados
14     display(test.isna().sum().sort_values())
```

**Figura 4. Completación de datos faltantes.**

A el resultado de la completación de los datos que será una serie de pandas con las combinaciones únicas de latitud y longitud como índice y el valor promedio de las emisiones como datos.

```

1 #El resultado de esta operación será una serie de pandas con las combinaciones únicas de latitud
2 # y longitud como índice y el valor promedio de las emisiones como datos
3 train.groupby(['latitude', 'longitude']).emission.mean().sort_values()

```

**Figura 5. Media de datos completados.**

## Modelo

El modelo se basa en la ubicación y el patrón anual, que no necesariamente se utilizan mediciones satelitales ni realizar extrapolaciones. La predicción sería el promedio de las emisiones de los últimos años para la misma ubicación y semana, no se va a requerir imputación de valores. Además de que las emisiones tuvieron una caída del 23% en el segundo trimestre de 2020 debido posiblemente a la influencia de el COVID-19, pero se recuperaron con un aumento del 25% en el segundo trimestre de 2021, lo que muestra la influencia significativa de eventos excepcionales en la tendencia y la dificultad de predecir basándose en promedios anteriores.

Inicialmente se hizo un análisis del comportamiento del sulfuro de dióxido so2 para dimensionar la tendencia temporal.

```

3 train_roll_mean = train.groupby('location')[train.columns[5:]].rolling(window=2).mean().reset_index()
4 train_roll_mean = train_roll_mean.rename(columns={col: col + '_roll_mean' for col in train_roll_mean.columns[2:]})
5
6 # Feature engineering para el conjunto de prueba (test)
7 test['location'] = test[['latitude', 'longitude']].round(2).astype(str).agg('.'.join, axis=1)
8 test_roll_mean = test.groupby('location')[test.columns[5:]].rolling(window=2).mean().reset_index()
9 test_roll_mean = test_roll_mean.rename(columns={col: col + '_roll_mean' for col in test_roll_mean.columns[2:]})
10 test_roll_mean.head()
11

```

**Figura 6. Tendencia temporal al sulfuro de dióxido de so2.**

Posterior a eso se fusionaron las características con el conjunto de entrenamiento (train) y del conjunto de prueba (test) siguiendo el orden de ubicación, año y número de semana.

```

1 # Fusión de características ingenierizadas con el conjunto de entrenamiento (train)
2
3 # Paso 1: Ordenar el conjunto de entrenamiento por 'location', 'year' y 'week_no'.
4 train = train.sort_values(by=['location', 'year', 'week_no'], ignore_index=True)
5
6 # Paso 2: Fusionar el conjunto de entrenamiento ordenado con las características ingenierizadas (train_roll_mean).
7 train_eng = train.merge(train_roll_mean, how='left', left_index=True, right_index=True)
8
9 # Fusión de características ingenierizadas con el conjunto de prueba (test)
10
11 # Paso 1: Ordenar el conjunto de prueba por 'location', 'year' y 'week_no'.
12 test = test.sort_values(by=['location', 'year', 'week_no'], ignore_index=True)
13
14 # Paso 2: Fusionar el conjunto de prueba ordenado con las características ingenierizadas (test_roll_mean).
15 test_eng = test.merge(test_roll_mean, how='left', left_index=True, right_index=True)
16
17 # Vista previa del conjunto de prueba con características ingenierizadas
18 test_eng.head()
19

```

```

1 # Feature engineering train
2 train_roll_mean = train.sort_values(by = ['location', 'year', 'week_no']).groupby(['location'])[train.columns[5:].tolist()].rolling(
3 train_roll_mean.drop(['level_1', 'emission', 'location'], axis = 1, inplace = True)
4 train_roll_mean.columns = [col + '_roll_mean' for col in train_roll_mean.columns]
5
6 # Feature engineering test
7 test.latitude, test.longitude = round(test.latitude, 2), round(test.longitude, 2)
8 test['location'] = [str(x) + '_' + str(y) for x, y in zip(test.latitude, test.longitude)]
9 test_roll_mean = test.sort_values(by = ['location', 'year', 'week_no']).groupby(['location'])[test.columns[5:].tolist()].rolling(w
10 test_roll_mean.drop(['level_1', 'location'], axis = 1, inplace = True)
11 test_roll_mean.columns = [col + '_roll_mean' for col in test_roll_mean.columns]
12 test_roll_mean.head()

1 #Train
2 train_eng = train.sort_values(by = ['location', 'year', 'week_no'], ignore_index = True).merge(train_roll_mean, how = 'left',
3 left_index=True, right_index=True)
4 # Test
5 test_eng = test.sort_values(by = ['location', 'year', 'week_no'], ignore_index = True).merge(test_roll_mean, how = 'left',
6 left_index=True, right_index=True)
7 # Preview engineered test set
8 test_eng.head()

```

**Figura 7. Fusión de características.**

Se procesa y se hace la selección de cada variable independiente del conjunto de entrenamiento (train) y del conjunto de prueba (test).

```

1
2
3 X = train_eng.drop(['ID_LAT_LON_YEAR_WEEK', 'location', 'emission'], axis = 1).fillna(0)
4 y = train_eng.emission
5
6 # Splitting the data into training and testing sets
7 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = SEED)
8

```

**Figura 8. Selección de cada variable independiente.**

Se usa un modelo de regresión utilizando el algoritmo RandomForestRegressor de scikit-learn. A continuación, se entrena el modelo utilizando el conjunto de entrenamiento (X\_train, y\_train) y se realizan predicciones en el conjunto de prueba (X\_test). Finalmente, se mide la precisión del modelo utilizando la métrica de Error Cuadrático Medio (RMSE), que se imprime en la consola.

El rendimiento del modelo se evaluó utilizando la métrica RMSE (Root Mean Squared Error). Además, se creó un modelo de referencia simple basado en la ubicación y el patrón anual.

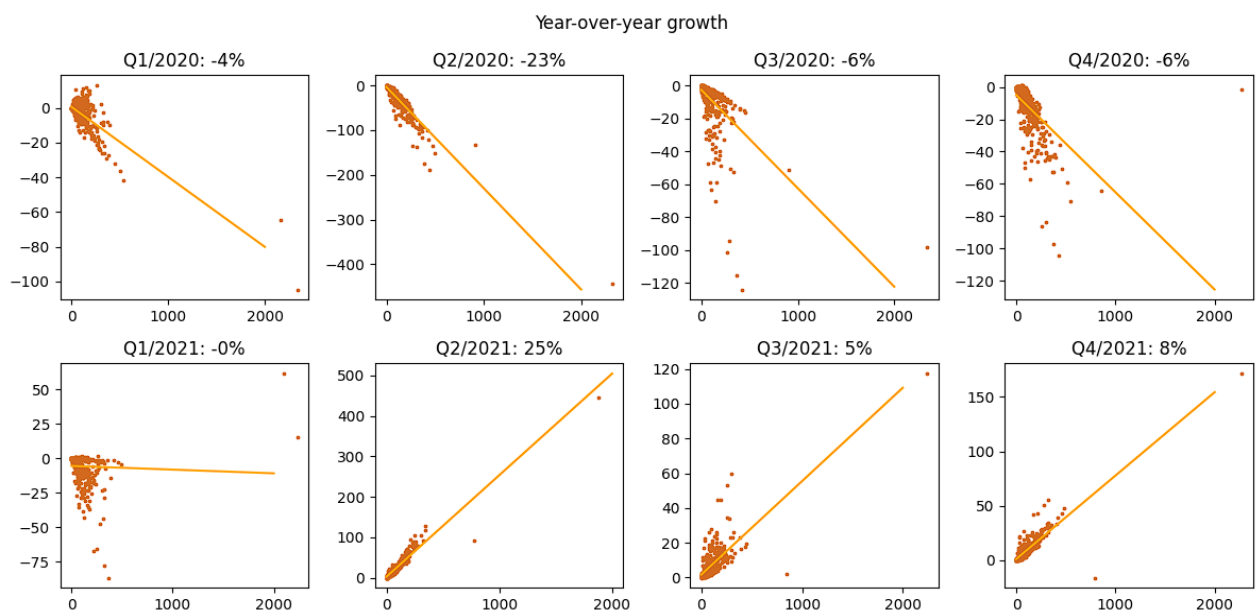
```

1
2 # Instantiating the model
3 clf = RandomForestRegressor(random_state = SEED, n_jobs=-1)
4 clf.fit(X_train, y_train)
5
6 # Making predictions
7 y_pred = clf.predict(X_test)
8
9 # Measuring the accuracy of the model
10 print(f'RMSE Score: {mean_squared_error(y_test, y_pred, squared=False)}') #
11

```

**Figura 9. Modelo de regresión y su rendimiento.**

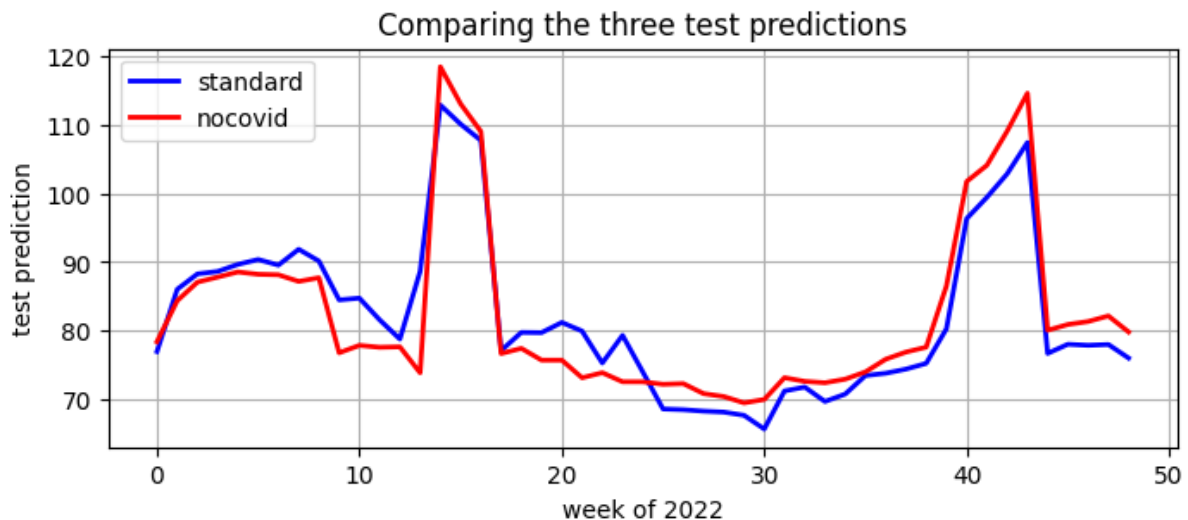
## Graficos y analisis



**Figura 10. Gráfico de crecimiento.**

El gráfico de crecimiento año tras año revela patrones significativos en las emisiones promedio a lo largo de diferentes ubicaciones y trimestres. La distribución de puntos muestra una tendencia general, donde la mayoría forma una línea en el gráfico. Cada punto representa una ubicación específica, y las líneas de regresión lineal ajustadas indican la relación entre las emisiones actuales y el cambio en las emisiones después de un año. Las pendientes de estas líneas, expresadas en porcentaje, proporcionan insights sobre las tasas de crecimiento año tras año. Se observa variabilidad significativa entre ubicaciones y trimestres, con algunas áreas experimentando un crecimiento rápido, otras mostrando un crecimiento más moderado, e incluso casos de decrecimiento. La comparación entre trimestres permite identificar patrones

estacionales. Este análisis proporciona una comprensión detallada de las dinámicas de emisiones en función de la ubicación y el tiempo, lo que puede ser crucial para la toma de decisiones y la formulación de estrategias ambientales.



**Figura 11. Gráfico comparativo.**

Se realizaron análisis comparativos, destacando la influencia de eventos excepcionales, como la pandemia de COVID-19, en las emisiones. Se ajustó un modelo excluyendo datos relacionados con la pandemia y se compararon las predicciones con un modelo estándar.

## Modelos supervisados

El código se centra principalmente en modelos de regresión (supervisados) y análisis exploratorio de datos.

El modelo de regresión mostró un rendimiento aceptable, con una influencia significativa de eventos excepcionales en las predicciones. Se compararon las curvas de aprendizaje y métricas de modelos supervisados, resaltando la importancia de considerar eventos excepcionales en la predicción de emisiones.

RMSE como métrica para evaluar el rendimiento del modelo de regresión



```
# Instantiating the model
clf = RandomForestRegressor(random_state = SEED, n_jobs=-1)
clf.fit(X_train, y_train)

# Making predictions
y_pred = clf.predict(X_test)

# Measuring the accuracy of the model
print(f'RMSE Score: {mean_squared_error(y_test, y_pred, squared=False)}') #
```

Gráfico 12. Métricas

### **Retos y Consideraciones de Despliegue**

La implementación efectiva de los modelos en entornos de producción implica consideraciones prácticas y desafíos. Se exploraron posibles obstáculos y estrategias para abordarlos, incluyendo la necesidad de ajustar hiper parámetros y la importancia de realizar una validación robusta.

### **Conclusiones**

- El modelo de regresión muestra un rendimiento aceptable, pero la inclusión de eventos excepcionales puede afectar significativamente las predicciones.
- El análisis comparativo sugiere que la pandemia de COVID-19 tuvo un impacto notable en las emisiones, destacando la importancia de considerar eventos excepcionales al desarrollar modelos de predicción.
- Se recomienda explorar más a fondo la inclusión de características adicionales y ajustar los hiper parámetros del modelo para mejorar la precisión de las predicciones.