

Predicción de Concentraciones de CO2 en Rwanda: Un Enfoque de Machine Learning para la Gestión del Cambio Climático

1. Introducción

Este informe presenta el proyecto Kaggle: concursos para desarrollar un modelo de aprendizaje automático basado en el conjunto de datos. El objetivo principal de este proyecto es crear un modelo que pueda predecir futuras emisiones de dióxido de carbono (CO2) utilizando datos recopilados por las observaciones del satélite SENTINEL-5P. Se ha compilado un conjunto de datos que cubre alrededor de 497 ubicaciones diferentes en diferentes regiones de Ruanda, incluidas áreas urbanas, áreas agrícolas y centrales eléctricas. Este conjunto de datos contiene variables importantes como "latitud", "longitud", "año" y "número de semanas" registradas cada semana desde enero de 2019 hasta noviembre de 2022. La variable objetivo de nuestro estudio son las "emisiones", que representan las emisiones de dióxido de carbono. de estos lugares. A continuación se muestra un enlace al conjunto de datos específico con el que trabajamos.

2. Objetivo

El objetivo de este proyecto es desarrollar un modelo de pronóstico de emisiones de CO2 para Ruanda con una precisión objetivo del 75% o mejor. Se basa en tener muchas variables importantes como "latitud", "longitud", "año", "SulfurDioxide_SO2_slant_column_number_density", "SulfurDioxide_SO2_column_number_density". Estas variables nos permiten identificar patrones de emisiones de CO2 en diferentes regiones y en diferentes momentos.

Además de ser importante para la previsión, este modelo, una vez implementado con éxito, puede proporcionar información valiosa a las agencias ambientales. Las personas, las agencias gubernamentales y las empresas pueden utilizar esta información para tomar decisiones informadas que mejoren la calidad de vida en Ruanda. Estas decisiones pueden incluir la promoción de medios de transporte respetuosos con el medio ambiente en las ciudades, la promoción de alternativas como la bicicleta y la reducción del consumo de energía tanto a nivel doméstico como industrial.

Cabe señalar que este proyecto también ofrece oportunidades de ganar dinero. Algunas de las oportunidades incluyen la capacidad de ofrecer suscripciones, asociarse con agencias gubernamentales o empresas interesadas en reducir las emisiones de carbono y brindar servicios de consultoría en sostenibilidad y neutralidad de carbono utilizando el modelo desarrollado. Estas habilidades adicionales pueden contribuir a la sostenibilidad y viabilidad a largo plazo de este proyecto.

3. Exploración y análisis del DataSet

Descripción general del conjunto de datos

Inicialmente, realizamos una revisión rápida del conjunto de datos para comprender su estructura y los valores contenidos en el mismo. Simultáneamente, identificamos las columnas y verificamos la integridad de la carga de datos. En una primera observación, notamos la presencia de valores faltantes en la fila con la identificación "ID_-0.510_29.290_2019_03".

```
[25] df.head()
```

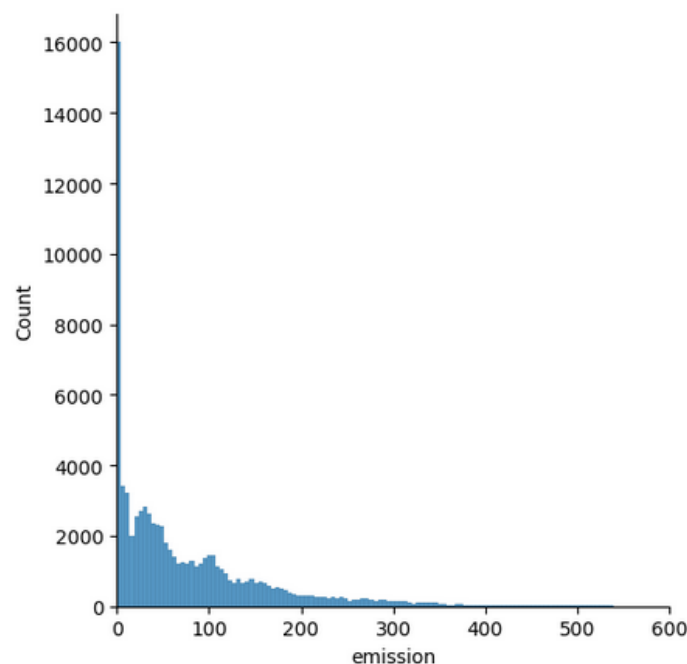
ID_LAT Lon_YEAR_WEEK	Latitude	longitude	year	week_no	SulphurDioxide_SO2_column_number_density	SulphurDioxide_SO2_column_number_density_anf	SulphurDioxide_SO2_slant_column_number_density	SulphurDioxide_cloud_fraction
ID_-0.510_29.290_2019_00	-0.51	29.29	2019	0	-0.000108	0.603019	-0.000065	0.255668
ID_-0.510_29.290_2019_01	-0.51	29.29	2019	1	0.000021	0.728214	0.000014	0.130988
ID_-0.510_29.290_2019_02	-0.51	29.29	2019	2	0.000514	0.748199	0.000385	0.110018
ID_-0.510_29.290_2019_03	-0.51	29.29	2019	3	NaN	NaN	NaN	NaN
ID_-0.510_29.290_2019_04	-0.51	29.29	2019	4	-0.000079	0.676296	-0.000048	0.121164

5 rows x 9 columns

Por otro lado, se aplican un par de funciones para entender un poco más el DataSet para describir estadísticas y poder entender un poco más el comportamiento del DataSet.

Análisis de la variable objetivo “emission”

En primera instancia se analiza la distribución de la variable objetivo “emission” para identificar posibles sesgos en ella y así poder corregir con alguna transformación si es el caso



en la gráfica se logra observar que la distribución de la variable objetivo es asimetría ya que tiene la cola derecha muy larga por lo que se procede a calcular la asimetría “slewness” de la variable objetivo

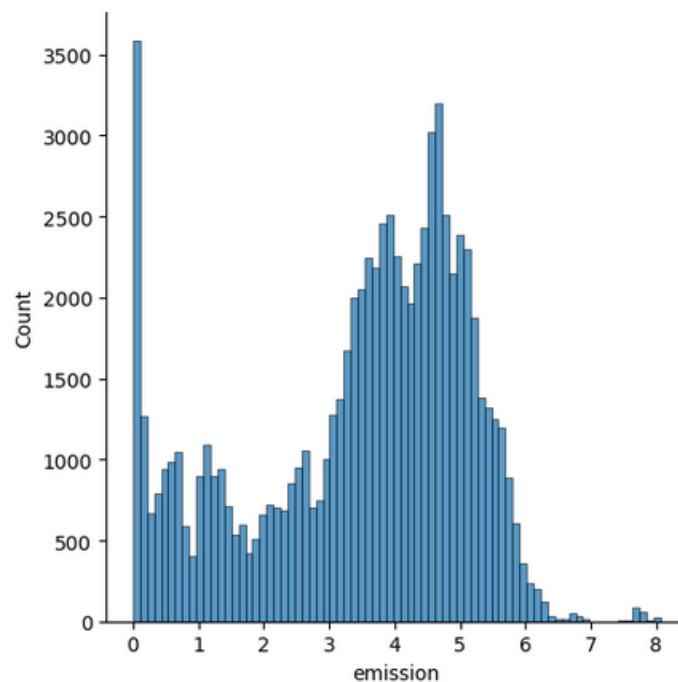
```
[27] print('La asimetira (Slewness) de la variable objetivo es:', df['emission'].skew())
```

La asimetira (Slewness) de la variable objetivo es: 10.173825825101622

- Asimetría: 10.173825825101622

la asimetría de la variable objetivo en nuestro caso de estudio es 10.173825825101622, es un valor bastante alto y positivo lo que significa que la distribución de la variable objetivo está sesgada hacia la derecha “sesgo positivo”, esto indica que, en promedio los valores de la variable objetivo son mayores que la mediana, lo que indica que en la mayoría de las observaciones tienen valores más bajos pero un pequeño porcentaje de las observaciones tienen valores extremadamente altos que influyen en la distribución de la variable objetivo.

Ya que la asimetría de la variable objetivo es un valor bastante alto es necesario aplicar una transformación para corregir dicho fenómeno en este caso aplicamos la transformación logarítmica (Logaritmo Natural), se volvió a graficar la distribución de la variable objetivo y se evidenciaron los siguientes resultados.



se observa una distribución más clara aunque aun así siendo transformada logarítmicamente se denota la cola a la derecha de la distribución original, también se observa con en la distribución original que el dato más recurrente es el 0, a demás para esta distribución transformada también le calculamos su asimetría “slewnees” y nos da el siguiente resultado

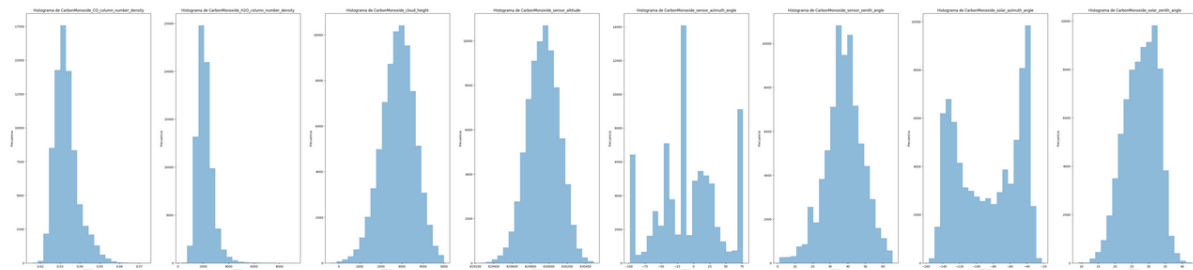
```
[45] print('La asimetria (Slewness) de la variable objetivo despues de una tranformacion logaritmica es:', df['emission'].skew())  
La asimetira (Slewness) de la variable objetivo despues de una tranformacion logaritmica es: -0.6091471436943189
```

- *Asimetría:* -0.6091471436943189

Después de aplicar la transformada logarítmica a la variable objetivo su asimetría se redujo significativamente, lo que indica que la distribución se volvió más simétrica o incluso en este caso sesgada hacia la izquierda, esta asimetría negativa indica que los valores tienden a agruparse hacia la izquierda de la distribución y que los valores extremadamente altos se atenuaron.

Exploración de variables

Se realiza la identificación del tipo de datos de cada columna para tomar decisiones acerca de cómo visualizar cada una de las columnas, al realizar la identificación nos damos cuenta que las variables que componen el DataSet son de tipo “float64” y “int64” es decir todas las variables son numéricas por lo que es conveniente visualizarlas con histogramas.



Cabe aclarar que por falta de conocimiento en el tema no fue posible identificar las variables más relevantes del DataSet por lo que se optó por graficar cada variable para conocer su distribución y dar claridad del comportamiento de cada una de ellas.

Análisis de datos faltantes

Al examinar nuestros datos, observamos que a algunas variables les faltan valores, lo cual es un aspecto importante de nuestro análisis. Profundizando en este problema, encontramos que algunas de estas variables muestran un alarmante porcentaje de datos faltantes, un 99,44% para ser exactos. Este hallazgo plantea serias preocupaciones sobre la integridad y confiabilidad de nuestros resultados.

Una cantidad significativa de datos faltantes puede afectar negativamente a nuestro análisis estadístico. Esto se debe a que los datos faltantes pueden afectar nuestras conclusiones, afectar la representatividad de la muestra y, en última instancia, reducir la precisión de los modelos o las conclusiones extraídas de estos datos incompletos.

	Total	Percent
UvAerosolLayerHeight_aerosol_height	78584	99.444466
UvAerosolLayerHeight_sensor_azimuth_angle	78584	99.444466
UvAerosolLayerHeight_aerosol_pressure	78584	99.444466
UvAerosolLayerHeight_aerosol_optical_depth	78584	99.444466
UvAerosolLayerHeight_sensor_zenith_angle	78584	99.444466
...
latitude	0	0.000000
longitude	0	0.000000
week_no	0	0.000000
year	0	0.000000
emission	0	0.000000

75 rows x 2 columns

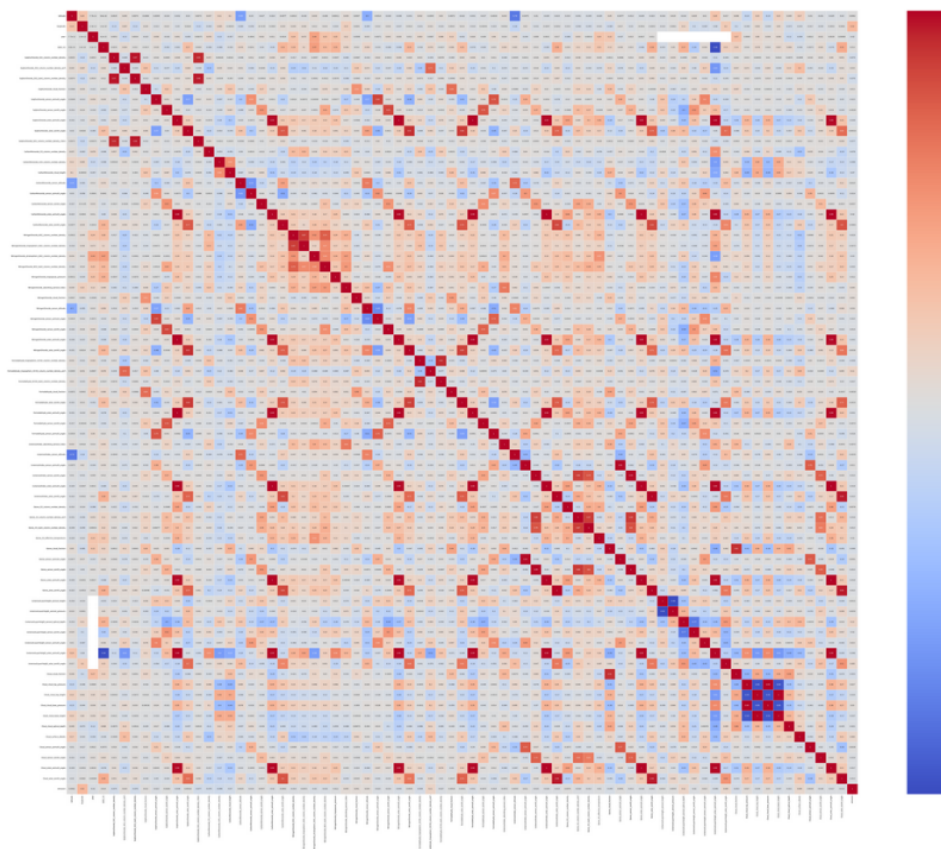
Analisis de correlacion

Al realizar el análisis de correlación nos permite identificar el tipo de correlación que existe entre las variables del DataSet ya sea una correlación negativa perfecta (Variables inversamente proporcionales) o una correlación positiva perfecta (Variables directamente proporcional) o si indica falta de correlación.

- **Matriz de correlación:**

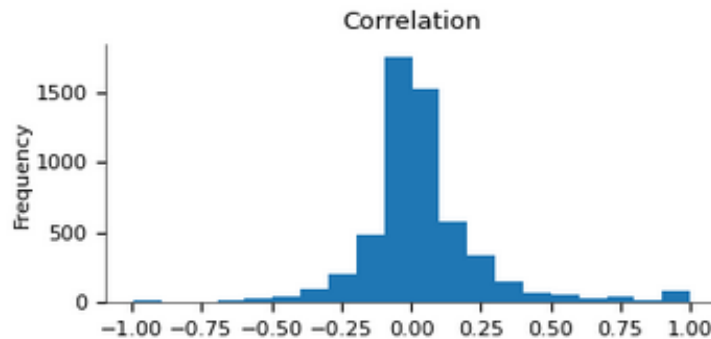
Al realizar la matriz de correlación logramos identificar varias cosas a simple vista, dejando de lado la diagonal principal ya que representa la correlación de la variable con ella misma, por otro lado en primera instancia podemos observar algunas relaciones positivas perfectas o fuertes lo que nos quiere decir que esas variables tienen un comportamiento muy similar, por otro lado también se observan algunas correlaciones negativas perfectas o débiles, además se logran observar algunos patrones curiosos en la matriz que serían un espacio de cuadrícula en la parte inferior derecha.

Algunas correlaciones para destacar podrían la variable “cloud_cloud_top_height” y “cloud_cloud_top_pressure” la cual tienen una correlación de “- 0.95%” es decir que son inversamente proporcionales y se podría interpretar como la dispersión de gases, ya que la presión y el tamaño de las nubes influyen en la dispersión y mezcla de gases en la atmósfera.



Por otro lado podemos ver que la distribución de la matriz de correlación es una distribución simétrica esto podría significar que hay presencia de independencia entre variables ya que la

falta de correlación entre las variable es lineal lo que denota que las variables son independientes entre sí, esto tiene aspectos buenos y malos como punto positivo es que facilita los análisis estadísticos y negativos es que en la vida real rara vez se tienen datos con estas características es fundamental indagar de la naturaleza de los datos.



- **Correlación con respecto a la variable objetivo “emission”**

Al analizar la correlación de las variables del DataSet con respecto a la variable objetivo “emission”, a simple vista podemos observar que las correlaciones se trata de “correlación débil” y en la mayoría de los casos se trata de una “correlación muy débil”, esto indica que la variables no están fuertemente relacionadas con la variable objetivo por lo tanto, se pueden considerar variables independientes.

Cabe destacar que la correlación no debe ser el único factor que determine la importancia de una variable, es posible que una variable con una correlación muy baja aporte información valiosa en combinación con otras variables para predecir la variable objetivo es decir que en conjunto se vuelven altamente significativas.

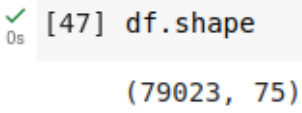
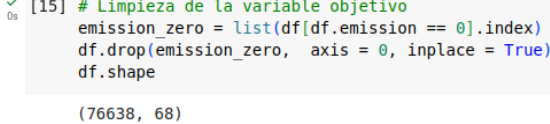
	emission
emission	1.000000
longitude	0.318397
UvAerosolLayerHeight_aerosol_optical_depth	0.118135
Formaldehyde_tropospheric_HCHO_column_number_density_amf	0.113583
UvAerosolLayerHeight_solar_zenith_angle	0.084772
...	...
SulphurDioxide_SO2_column_number_density	-0.074261
SulphurDioxide_SO2_slant_column_number_density	-0.075850
SulphurDioxide_SO2_column_number_density_15km	-0.076736
CarbonMonoxide_CO_column_number_density	-0.077760
CarbonMonoxide_H2O_column_number_density	-0.097528

75 rows x 1 columns

4. Limpieza y reparación del DataSet

Limpieza de la variable objetivo “emission”

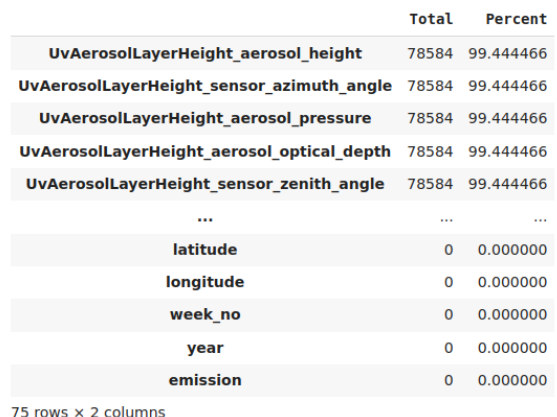
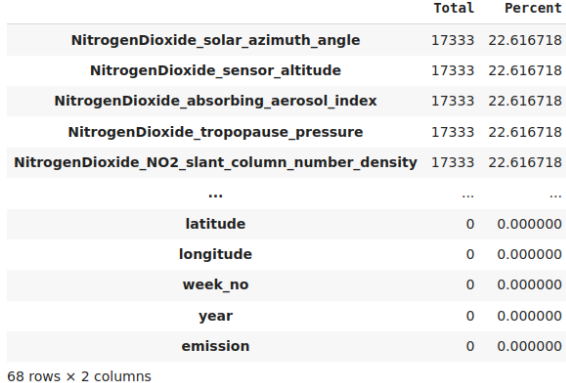
En la exploración de los datos, al analizar la variable objetivo se encontró que presentaba algunos datos faltantes, por lo tanto se remueven las filas que contienen dichos valores 0, ya que estos valores no indican ninguna medición y no tiene sentido utilizarlos en el entrenamiento del modelo.

Antes	Después
	

Cómo se logra observar en las imágenes después de realizar la limpieza de la variable objetivo se eliminaron se eliminaron 2.385 filas que contienen valores 0 en la variable objetivo.

Eliminación de columnas con datos faltantes

En la exploración de los datos, al analizar la cantidad y porcentaje de datos faltantes por variable en el DataSet se identificó algunas variables con un porcentaje de datos faltantes del 99.44%, se toma la decisión de eliminar las variables que contengan una cantidad considerable de datos faltantes. Como criterio se decide que las variables que tengan un porcentaje igual o mayor al 50% se eliminará del DataSet

Antes	Después
	

Como se observa en las imágenes Antes de realizar la eliminación de las variables se contaba con 75 variables, después de hacer la eliminación de las variables con el 50% o más de datos

faltantes el DataSet queda con 68 variables, por lo que concluimos que se eliminaron 7 variables que presentaban altos porcentajes de datos faltantes.

Rellenado de datos faltantes

Anteriormente se eliminaron las variables con un porcentaje igual o mayor al 50% de datos faltantes pero esto no es suficiente para tener un DataSet óptimo para hacer el entrenamiento de un modelo confiable ya que aún hay muchos datos faltantes en las variables que aún hacen parte del DataSet por lo que se opta por rellenar estos datos faltantes con la “*mediana*” de cada variable.

La decisión de realizar el relleno de los datos faltantes con la “*mediana*” fue por su robustez frente a datos atípicos ya que la “*mediana*” es una medida de tendencia central que es menos sensible a los valores atípicos o extremos que la media. Esto significa que, si tienes valores extremos del DataSet, la mediana puede proporcionar una mejor estimación del valor central sin verse afectada por esos valores extremos.

Antes	Después
<pre> 0s [17] df.isnull().sum() latitude 0 longitude 0 year 0 week_no 0 SulphurDioxide_S02_column_number_density 14609 ... Cloud_sensor_azimuth_angle 484 Cloud_sensor_zenith_angle 484 Cloud_solar_azimuth_angle 484 Cloud_solar_zenith_angle 484 emission 0 Length: 75, dtype: int64 </pre>	<pre> 0s [18] df.isnull().sum() latitude 0 longitude 0 year 0 week_no 0 SulphurDioxide_S02_column_number_density 0 ... Cloud_sensor_azimuth_angle 0 Cloud_sensor_zenith_angle 0 Cloud_solar_azimuth_angle 0 Cloud_solar_zenith_angle 0 emission 0 Length: 68, dtype: int64 </pre>

Como se observa en las imágenes antes de realizar el relleno de datos faltantes el DataSet contaba con Variables de 14.689 datos faltantes, después de realizar el relleno podemos observar que los datos de cada una de las variables están completos.

En conclusión realizando estos procedimientos de limpieza del DataSet contamos con mayor calidad de datos, simplificación del análisis, menos sesgo en los modelos entre otros aspectos positivos que se ganan realizando una limpieza a el DataSet

5. Modelos predictivos e iteraciones

A continuación se presenta un análisis de la utilización de modelos de machine learning. Se considerarán tanto modelos supervisados como lo son “Linear Regression”, “Decision Tree Regressor” y “Random Forest Regressor”, así como modelos de machine learning no supervisados como lo son “PCA”, “NMF”

a. Preparación del DataSet:

Lo primero que se debe hacer para la implementación de diferentes modelos de “Machine Learning” es preparar los datos para esto.

- Separación de la variable objetivo de las demás variables dependientes del DataSet. Como podemos observar con esta instrucción aislamos la variable objetivo donde “X” son todas la variables dependientes y “y” es la variable objetivo cada una con 76638 observaciones.

```
✓ [32] # Separacion de la variable objetivo "emmission"
0s X = df.values[:, :-1]
    y = df['emission'].values
    print (X.shape, y.shape)

(76638, 67) (76638,)
```

- División del DataSet en “train” y “test” en un porcentaje de 70% y 30% respectivamente, esto con el fin de evaluar la capacidad de predicción precisa en datos que no ha visto o con los que no se ha entrenado el modelo.

```
✓ [33] # Division de datos 70% train - 30% test
0s Xtr, Xts, ytr, yts = train_test_split(X,y, test_size=0.3)
    print (Xtr.shape, ytr.shape, Xts.shape, yts.shape)

(53646, 67) (53646,) (22992, 67) (22992,)
```

1. **Xtr**: Partición del DataSet “X” para entrenamiento
2. **Xts**: Partición del DataSet “X” para test
3. **ytr**: Partición del DataSet “y” para entrenamiento
4. **yts**: Partición del DataSet “y” para test

b. Modelos Supervisados

❖ *Linear Regression*

Dado que este modelo proporciona una interpretación simple entre las variables independientes y la variable dependiente o variable objetivo por otro lado su eficiencia computacional es eficiente en comparación con modelos más complejos por lo que se decidió implementar este modelo en primera instancia.

➤ *Resultados*

Para obtener los resultados del modelo “Linear Regression” se optó por utilizar la función “cross_val_score” la cual realiza una validación cruzada. Se utiliza ShuffleSplit como estrategia de particionamiento. Con n_splits=10, se realiza la validación cruzada en 10 iteraciones y se utiliza la métrica “rel_mrae” anteriormente definida.

```
[ ] z = cross_val_score(lr, X, y, cv = ShuffleSplit(n_splits=10, test_size=0.3), scoring=rel_mrae)
print (z)
print ("test score  %.3f (±%.4f)"%(np.mean(z), np.std(z)))

[4.08485395 3.58497864 4.10900419 3.56410509 4.63992899 3.73575649
 3.48419682 3.54704408 3.49455548 3.81177142]
test score  3.806 (±0.3528)
```

Las puntuaciones obtenidas en cada una de las 10 iteraciones, la media de dichas puntuaciones y la desviación estándar para resumir el rendimiento del modelo. Como se puede observar obtuvimos una media de 3.806 y una desviación estándar de 0.35 o 35 %. Una desviación estándar del 35% podría interpretarse como una variabilidad moderada, lo que significa que el modelo tiene cierta capacidad predictiva, pero aún hay margen para mejorar, lo que explica el sesgo que se encuentra entre las predicciones del modelo y los valores esperados.

➤ Predicción

Al generar la predicción del modelo se obtuvieron valores cercanos pero un poco sesgados como se verá a continuación, lo que se hizo fue imprimir los 10 primeros valores de predicción y los 10 primeros valores esperados para hacer una comparación.

Predicciones del modelo	Valores esperados
<pre># Predicciones con el modelo "Linear Regression" prediccionLR = lr.predict(Xts) prediccionLR[:10] array([3.79146901, 3.30486976, 3.05414073, 4.39258027, 3.87025628, 3.66628555, 3.69028855, 4.10776482, 2.92678345, 4.21546626])</pre>	<pre># Valores esperados yts[:10] array([4.66753776, 4.02835765, 4.58069018, 5.51975703, 4.32899248, 3.77574746, 1.990483 , 5.33702934, 3.44071523, 5.92200321])</pre>

Como podemos observar los valores predcidos por el modelo están un poco sesgados a los valores esperados.

➤ Error

Para analizar el error del modelo se utilizó el mediano absoluto (medAE), ya que es menos sensible a los valores extremos que el error cuadrático medio (MSE), lo que es favorable ya que es una métrica de evaluación robusta cuando tenemos datos atípicos como es el caso

```
[62] # Error del modelo
# Error mediano absoluto (medAE)
medaeLR = median_absolute_error(yts, lr.predict(Xts))
print (medaeLR)

1.0241322788022815
```

Como podemos observar tenemos un (medAE) de 1.024, lo que significa la diferencia entre los valores predcidos por el modelo y el valor esperado es de 1.024 lo que es una valor muy alto para los objetivos del proyecto.

❖ Decision Tree Regressor

➤ Resultados

Al igual que en el modelo anterior para obtener los resultados del modelo “Decision Tree Regressor” se optó por utilizar la función “cross_val_score” la cual realiza una validación cruzada. Se utiliza ShuffleSplit como estrategia de particionamiento. Con n_splits=10, se realiza la validación cruzada en 10 iteraciones y se utiliza la métrica “rel_mrae” anteriormente definida.

```
[63] z = cross_val_score(drt, X, y, cv = ShuffleSplit(n_splits=10, test_size=0.3), scoring=rel_mrae)
      print (z)
      print ("test score  %.3f (±%.4f)"%(np.mean(z), np.std(z)))

[0.04069547 0.04276888 0.04014766 0.04193854 0.04052056 0.03986934
 0.04170158 0.04299      0.04307703 0.040196 ]
test score  0.041 (±0.0012)
```

Al igual que en el modelo anterior tenemos las puntuaciones obtenidas en cada una de las 10 iteraciones, la media de dichas puntuaciones y la desviación estándar para resumir el rendimiento del modelo. Como se puede observar obtuvimos una media de 0.041 y una desviación estándar de 0.0012 o 0.12%. Una desviación estándar del 0.12% podría interpretarse como una variabilidad baja, lo que significa que el modelo tiene una alta capacidad predictiva, y hay muy poco margen para mejorar, pero puede que un modelo un poco más complejo lo mejore y nos dé mejores resultados y más precisión de predicción.

➤ Predicción

Al igual que en el modelo anterior para analizar la predicción del modelo se obtuvieron valores cercanos pero un poco sesgados pero esta vez mucho más cerca a los valores esperados como se verá a continuación, lo que se hizo fue imprimir los 10 primeros valores de predicción y los 10 primeros valores esperados para hacer una comparación.

Predicciones del modelo	Valores esperados
<pre>#Predicciones del modelo "Decision Tree Regressor" prediccionDRT = drt.predict(Xts) prediccionDRT[:10] array([4.53835826, 3.85280948, 4.52229655, 5.51418636, 5.08059267, 3.77275105, 2.0084643 , 5.19786501, 3.4858425 , 5.07664417])</pre>	<pre># Valores esperados yts[:10] array([4.66753776, 4.02835765, 4.58069018, 5.51975703, 4.32899248, 3.77574746, 1.990483 , 5.33702934, 3.44071523, 5.92200321])</pre>

Como podemos observar los valores predichos por el modelo están mucho más cercanos a los valores esperados que en el modelo anterior, el sesgo es mínimo en este modelo.

➤ Error

Al igual que en el modelo anterior para analizar el error del modelo se utilizó el mediano absoluto (medAE), ya que es menos sensible a los valores extremos que el error cuadrático medio (MSE), lo que es favorable ya que es una métrica de evaluación robusta cuando tenemos datos atípicos como es el caso

```
# Error del modelo
# Error mediano absoluto (medAE)
medaeDRT = median_absolute_error(yts, drt.predict(Xts))
print (medaeDRT)

0.04631949246912037
```

Como podemos observar a diferencia del anterior modelo en este tenemos un (medAE) de 0.046, lo que significa la diferencia entre los valores predecidos por el modelo y el valor esperado es de 0.046 lo que es un valor muy pequeño y ya nos deja ver la alta precisión del modelo al predecir nuevos valores.

❖ *RandomForest Regressor*

➤ Resultados

Al igual que en los dos primeros modelos para obtener los resultados del modelo “*Decision Tree Regressor*” se optó por utilizar la función “*cross_val_score*” la cual realiza una validación cruzada, Se utiliza *ShuffleSplit* como estrategia de particionamiento. Con *n_splits=10*, se realiza la validación cruzada en 10 iteraciones y se utiliza la métrica “*rel_mrae*” anteriormente definida.

```
[58] z = cross_val_score(rfr, X, y, cv = ShuffleSplit(n_splits=10, test_size=0.3), scoring=rel_mrae)
print (z)
print ("test score  %.3f (±%.4f)"%(np.mean(z), np.std(z)))

[0.03585695 0.03639306 0.03750236 0.03704602 0.03566726 0.03834894
 0.03709611 0.03528309 0.03939083 0.03830539]
test score  0.037 (±0.0013)
```

Al igual que en el modelo anterior tenemos las puntuaciones obtenidas en cada una de las 10 iteraciones, la media de dichas puntuaciones y la desviación estándar para resumir el rendimiento del modelo. Como se puede observar obtuvimos una media de 0.037 y una desviación estándar de 0.0013 o 0.13%. Una desviación estándar del 0.13% podría interpretarse como una variabilidad muy baja, lo que significa que el modelo tiene una muy alta capacidad predictiva.

➤ Predicción

Al igual que en el modelo anterior para analizar la predicción del modelo se obtuvieron valores cercanos pero un poco sesgados pero esta vez mucho más cerca a los valores esperados como se verá a continuación, lo que se hizo fue imprimir los 10 primeros valores de predicción y los 10 primeros valores esperados para hacer una comparación.

Predicciones de modelo	Valores esperados
<pre>[54] # Prediccion de los 10 primeros datos con le modelo "RamdonForestRegressor" prediccionRFR = rfr.predict(Xts) prediccionRFR[:10] array([4.63702504, 3.98938517, 4.52418517, 5.51500598, 4.38858597, 3.81079987, 2.06047472, 5.31041091, 3.33602244, 5.22835085])</pre>	<pre>[55] # Valores esperados yts[:10] array([4.66753776, 4.02835765, 4.58069018, 5.51975703, 4.32899248, 3.77574746, 1.990483 , 5.33702934, 3.44071523, 5.92200321])</pre>

Como podemos observar los valores predecidos por el modelo son muy cercanos a los valores esperados, el sesgo es casi imperceptible en este modelo.

➤ Error

Al igual que en el modelo anterior para analizar el error del modelo se utilizó el mediano absoluto (medAE), ya que es menos sensible a los valores extremos que el error cuadrático medio (MSE), lo que es favorable ya que es una métrica de evaluación robusta cuando tenemos datos atípicos como es el caso

```
# Error del modelo
# Error mediano absoluto (medAE)
medaeRFR = median_absolute_error(yts, rfr.predict(Xts))
print (medaeRFR)

0.04802187932883106
```

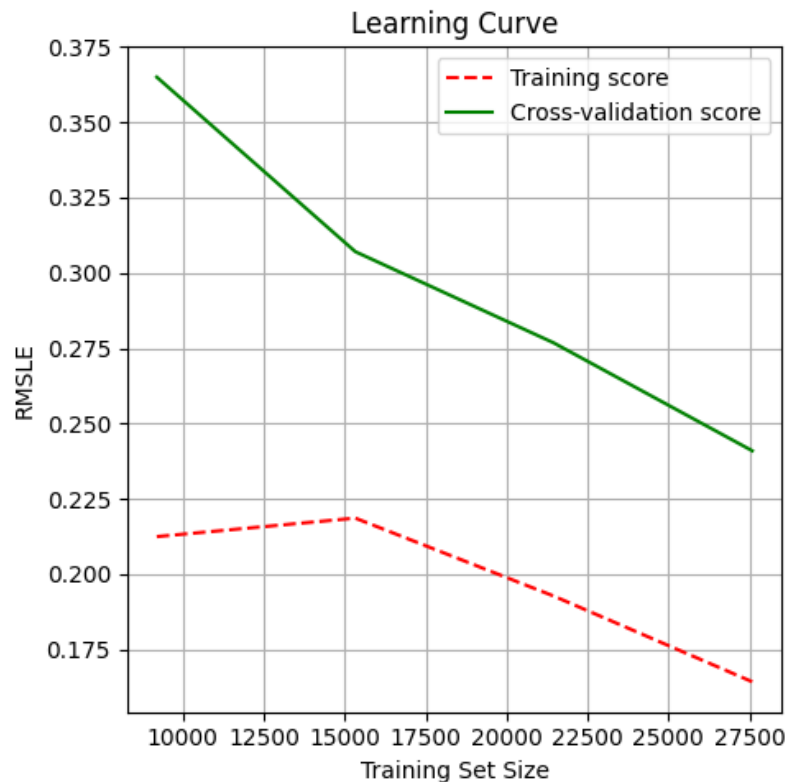
Como podemos observar a diferencia del anterior modelo en este tenemos un (medAE) de 0.048, lo que significa la diferencia entre los valores predecidos por el modelo y el valor esperado es de 0.048 lo que es un valor muy pequeño y ya nos deja ver la alta precisión del modelo al predecir nuevos valores.

6. Curva de aprendizaje

Estas curvas son útiles para entender cómo se está desempeñando un modelo a medida que se ajusta a más ejemplos y para identificar posibles problemas o mejoras en el proceso de aprendizaje.

- Decision Tree Regressor

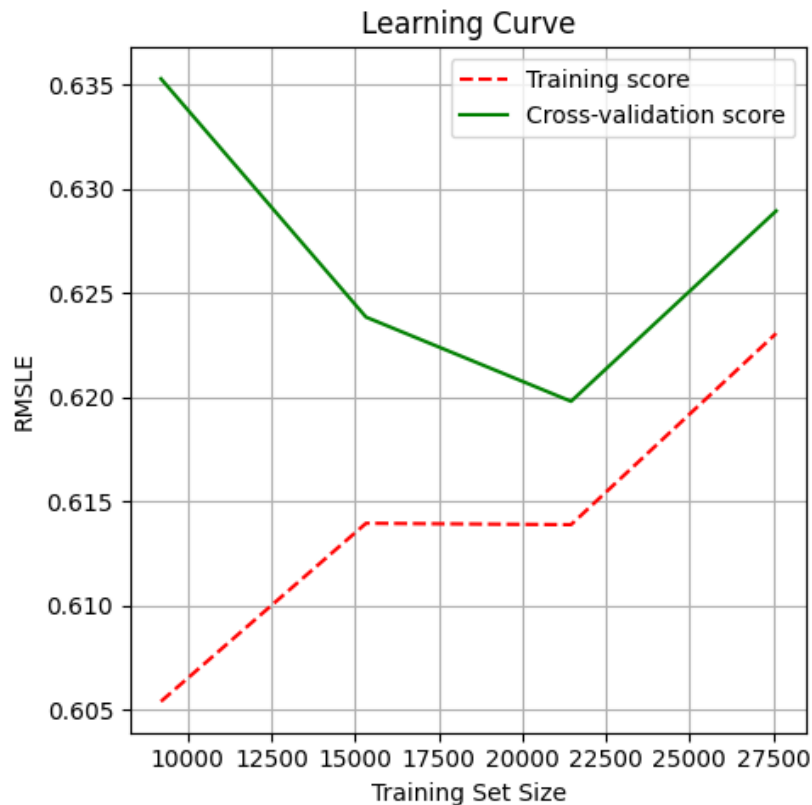
Para realizar la curva de aprendizaje de este modelo se utilizó el “Cross-validation score” y el “training score” para realizar la comparación entre sus comportamientos.



Como podemos observar al en un inicio las métricas tienen un comportamiento inversamente proporcional lo que quiere decir me mientras “cross-validation score” disminuye, el “training score” crece hasta un punto cercano a los 15000 datos de “test” donde su comportamiento cambia a un comportamiento directamente proporcional donde los dos parámetros disminuye lo que se puede interpretar como que hay cierta complejidad en el proceso de entrenamiento del modelo y podría presentar un comportamiento de sobreajuste.

- RandomForest Regressor

Al igual que en el gráfico anterior para realizar la curva de aprendizaje de este modelo se utilizó el “Cross-validation score” y el “training score” para realizar la comparación entre sus comportamientos.



Notamos que las métricas del gráfico tienen un comportamiento inicial inversamente proporcional luego un comportamiento constante y por último en comportamiento directamente proporcional, lo que podría interpretarse como que el modelo inicia con un aprendizaje estable y termina con posiblemente presenta “overfitting”.

7. Retos y consideraciones de despliegue

El principal reto al que me enfrente en este proyecto fue con respecto a la interpretación y relación entre las variables de la base de datos ya que la base de datos del proyecto es muy técnica y los nombres de sus variables exigen un conocimiento previo del campo de estudio relacionado con la base de datos, en forma de ejemplo algunos de los nombres son “NitrogenDioxide_tropopause_pressure”, “Ozone_O3_column_number_density_amf”, “UvAerosolLayerHeight_aerosol_optical_depth”, “SulphurDioxide_cloud_fraction”, esta complejidad dificulta la tarea interpretar relaciones entre variables y comprender su impacto en la variable objetivo “emission”, sin embargo mediante investigación logre superar este reto y adquirir una comprensión sólida de la base de datos, y de esta manera desarrollar un modelo de predicción efectivo.

Por otro lado se deben tener en cuenta alguna consideración para el despliegue del modelo a continuación enumerare algunas fundamentales

- **integración con una aplicación de software:** la integración con una app ya sea Web o Móvil es importante para que el público pueda hacer un uso eficiente del modelo mediante una aplicación que puedan utilizar diariamente.
- **Escalabilidad:** Para que la aplicación sea escalable es importante que el modelo maneje grandes volúmenes de datos y que estos datos puedan ser en tiempo real proporcionada por sensores o estaciones de monitores.
- **Ética y responsabilidad:** Es importante que la carga de datos sea segura de manera que en la base de datos no se carguen datos falsos y generen una predicción fuera de la realidad y de esta manera no sea un modelo eficiente o por otro lado que el modelo sea manipulado para satisfacer las intereses de terceras personas.
- **Documentación completa:** Proporcionar documentación detallada que describa el modelo, su entrenamiento, sus limitaciones y cómo interpretar sus resultados.
- **Actualización del modelo:** Establecer un proceso para actualizar periódicamente el modelo con nuevos datos y reentrenarlo para mantener su precisión a lo largo del tiempo.
- **Seguridad y Privacidad:** Implementar medidas de seguridad robustas para proteger tanto los datos de entrada como las predicciones del modelo. Asegurarse de cumplir con las regulaciones de privacidad y protección de datos.

8. Conclusiones

Dadas todas las situaciones que se presentaron durante el proyecto y lo aprendido en su realización puedo sacar algunas conclusiones que son bastante importantes en cualquier proyecto de estas características.

- La fase de limpieza de datos es esencial para garantizar la calidad de los resultados. La correcta manipulación y tratamiento de datos faltantes, outliers y otros problemas contribuyen significativamente a la fiabilidad del modelo.
- Se probaron varios modelos de machine learning, incluyendo regresión lineal, árbol de decisión y Random Forest Regressor. Esto demuestra un enfoque metodológico y una consideración de diversas técnicas para abordar el problema.
- El Random Forest Regressor se identificó como el modelo más eficiente entre los probados. Esto sugiere que la complejidad del problema de predicción de CO₂ puede ser mejor abordada mediante un modelo más avanzado y robusto como el Random Forest.
- Es importante evaluar bien el modelo generaliza a nuevos datos. Se deben haber utilizado métricas de evaluación, como error cuadrático medio (MSE) o coeficiente de determinación (R^2), para medir la capacidad del modelo para realizar predicciones precisas en datos no vistos.
- Se puede analizar la importancia de las características para comprender qué variables tienen el mayor impacto en las predicciones del modelo. Esto podría proporcionar información valiosa sobre los factores clave que influyen en los niveles de CO₂.