Entrega final - Introducción a la Inteligencia Artificial: Predecir las emisiones de CO2 en Ruanda

Anderson Estiven Villa Sierra <estiven.villa@udea.edu.co>
Sebastián Ochoa González <sebastian.ochoa1@udea.edu.co>
Simón Alfredo Narváez Restrepo <simon.narvaez@udea.edu.co>

1. Descripción del problema predictivo a resolver

El problema que se pretende resolver es la predicción de la emisión de CO2 o emisiones de carbono, y se pretende abordar creando un modelo de aprendizaje a partir de datos de fuentes abiertas sobre emisiones de CO2, el cual busca determinar si se producirán emisiones de CO2 en Ruanda o no, esta clasificación se realiza usando información relacionada con las ubicaciones en múltiples zonas de Ruanda, y su distribución con respecto a tierras de cultivo y centrales eléctricas.

2. Descripción del dataset seleccionado

Para el desarrollo del ejercicio previamente mencionado, se hará uso del siguiente dataset Predict CO2 Emissions tomando de Kaggle, de la sesión de Competitions y está compuesto por 497 ubicaciones únicas de múltiples zonas de Ruanda y como estas ubicaciones se encuentran distribuidas con respecto a zonas de cultivos, ciudades y plantas eléctricas. Estos datos hacen referencia al rango años de 2019 a 2021, y lo que se busca es predecir las emisiones de CO2 para el siguiente año, o sea, noviembre. Este dataset contiene un total de 79024 filas y un total de 76 columnas, que nos cuenten distintas características con respecto a nuestro conjunto de datos, como lo son Densidad de la columna de dióxido de azufre, índice de aerosoles absorbentes para el NO2 y demás características,la que nos encargaremos de predecir será la última de las 76 columnas que se llama “emisión”, que indica si se generó emisión de CO2 o no, en la ubicación. El data set cuenta con los siguientes datos faltantes

```
[17]  k = d.isna().sum()
      k[k!=0]
```

```
SulphurDioxide_SO2_column_number_density          14609
SulphurDioxide_SO2_column_number_density_amf      14609
SulphurDioxide_SO2_slant_column_number_density    14609
SulphurDioxide_cloud_fraction                     14609
SulphurDioxide_sensor_azimuth_angle               14609
                                                  ...
Cloud_surface_albedo                                484
Cloud_sensor_azimuth_angle                          484
Cloud_sensor_zenith_angle                           484
Cloud_solar_azimuth_angle                           484
Cloud_solar_zenith_angle                            484
Length: 70, dtype: int64
```

```
d.isnull().sum()
```

```
ID_LAT_LON_YEAR_WEEK            0
latitude                        0
longitude                       0
year                            0
week_no                         0
                               ...
Cloud_sensor_azimuth_angle    484
Cloud_sensor_zenith_angle     484
Cloud_solar_azimuth_angle     484
Cloud_solar_zenith_angle      484
emission                        0
Length: 76, dtype: int64
```

3. Iteraciones de desarrollo

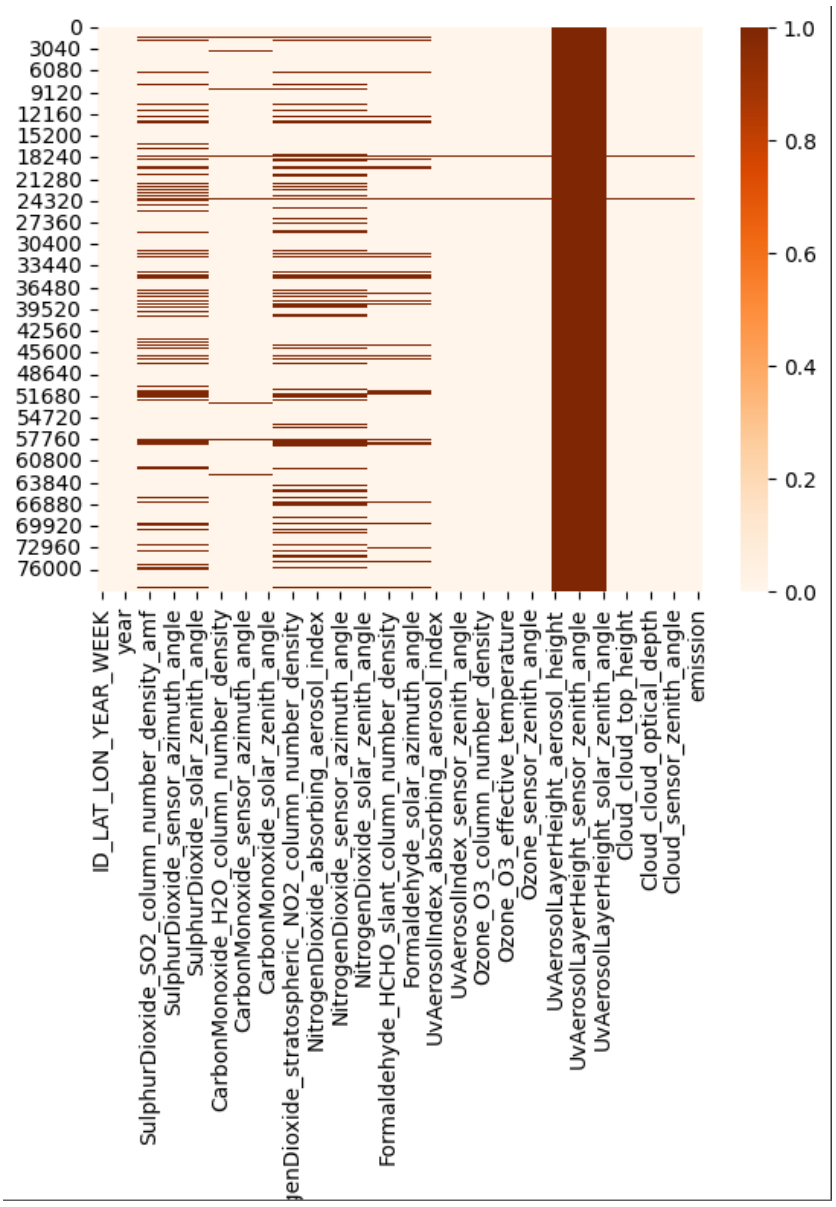
a. Preprocesado de datos

En el proceso de exploración del dataset, se encuentran con varios aspectos relacionados con los datos, primero que hay grandes cantidades de datos faltantes, por ejemplo hay columnas como SulphurDioxide_SO2_column_number_density que tienen 14.609 valores faltantes, lo cual consideramos que hay cantidades significativas faltantes y se procede a realizar un análisis más a profundidad respecto a estos datos para poder aplicar alguna estrategia frente a estos datos faltantes. Se procede a analizar cuál es el porcentaje de valores faltantes por columna en el dataset, y nos encontramos que hay columnas que cuyos datos representan un 99.444466% de datos faltantes, lo que representa una gran cantidad y se plantean frente a esto dos estrategías, la primera es una imputación de valores faltantes, donde sustituiremos los valores faltantes por la media de los datos, y la segunda es que se eliminarán las filas con valores nulo. Para esto hacemos un análisis sobre todas las columnas para mirar cuáles representan gran porcentaje de

datos faltantes, donde nos encontramos con las siguientes columnas que solo cuentan con 439 datos.

```
57  UvAerosolLayerHeight_aerosol_height                439 non-null    float64
58  UvAerosolLayerHeight_aerosol_pressure              439 non-null    float64
59  UvAerosolLayerHeight_aerosol_optical_depth         439 non-null    float64
60  UvAerosolLayerHeight_sensor_zenith_angle           439 non-null    float64
61  UvAerosolLayerHeight_sensor_azimuth_angle          439 non-null    float64
62  UvAerosolLayerHeight_solar_azimuth_angle           439 non-null    float64
63  UvAerosolLayerHeight_solar_zenith_angle            439 non-null    float64
```

haciendo un diagrama de calor, confirmamos lo ya descrito, que las características relacionadas con UvAerosolLayerHeight, representan una gran parte de datos vacíos.



Así que el punto de partida en común para las dos estrategías de datos faltantes es la eliminación de dichas columnas, y las candidatas para dicha eliminación son todas las relacionadas con las categorías de UvAerosolLayerHeight_ que están son descritas en el dataset desde la posición 57 hasta la 63, ambos extremos incluidos.

Ya con la eliminación de estas columnas obtenemos un nuevo dataset que será con el cual aplicaremos las dos estrategías anteriormente mencionadas con respecto a los otros datos faltantes, que no representaban tanto porcentaje dentro del dataset, pero igual se tienen que tratar. El primer dataset que obtendremos es en el cual usaremos una estrategia de llenado de datos faltantes con la media de los datos, y tendremos el primer dataframe imputado, que no cuenta con datos faltantes, ya que fueron reemplazados con la media de los datos.El segundo dataset a obtener es aquel donde las filas que poseían datos faltantes fueron eliminadas, así con dos dataset resultantes de dicho proceso de limpieza y de tratamiento de datos faltantes se tienen datos para empezar a construir y trabajar con los modelos seleccionados.

b. Modelos Supervisados

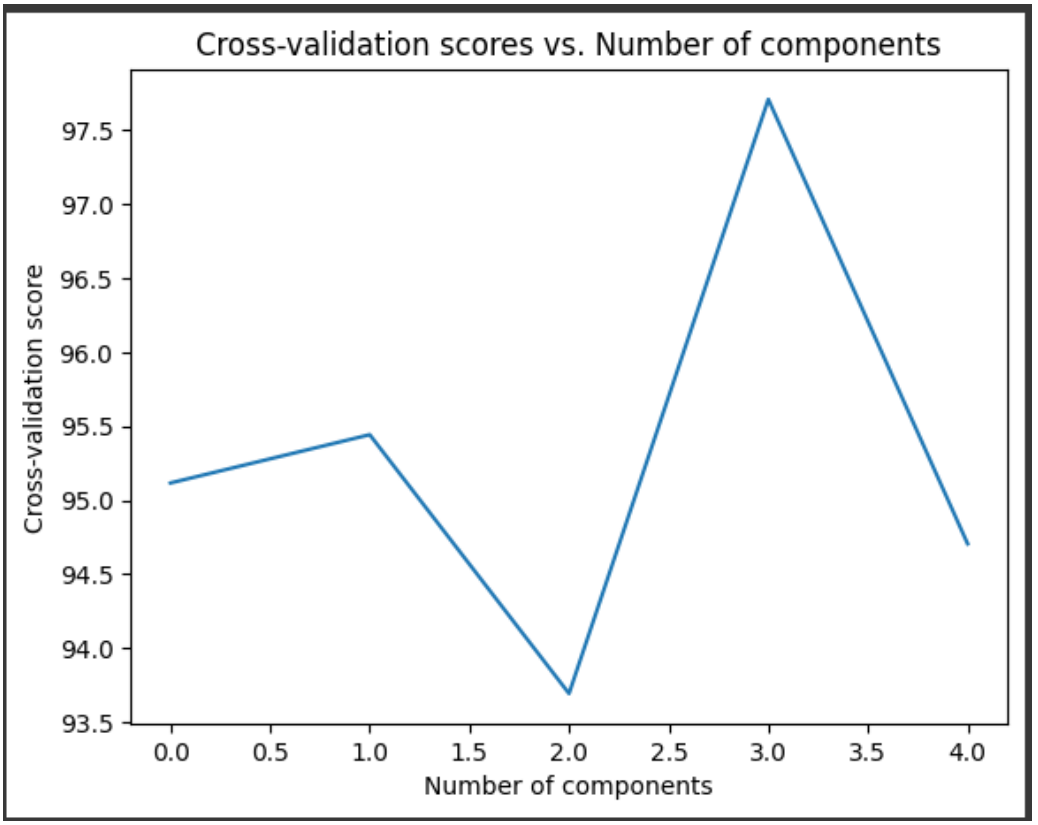
Para este modelo de predicción, se tuvieron en cuenta 4 modelos de regresión distintos, siendo estos : Random Forest, LightGBM, XGBC y CatBoost ya que nos pueden ayudar a describir de buena manera el presente problema de predicción, haciendo uso de estos algoritmos de regresión podemos describir o intentar describir de mejor manera dicho problema y su solución.

Para estos modelos se hicieron uso de los dos dataframes resultantes de la limpieza de datos, esto con el fin y el interés de conocer cómo distintas estrategías de tratamiento de datos faltantes afectan en las predicciones resultantes y cómo se ven reflejadas en las métricas propuestas para analizar cada modelo. En este caso, por la naturaleza del problema estaremos enfocados en las siguientes métricas RMSE score, R^2, y MAE. Con dichas métricas buscaremos encontrar los mejores resultados, y conocer qué modelo será capaz de predecir los datos de la mejor manera posible.

c. Modelos no supervisados

Para los modelos no supervisados usamos un análisis de componentes principales o PCA, sobre el dataset que eliminamos las columnas con un 99.4% de datos faltantes, y con los datos faltantes llenados con la media, para esto primero se hizo un escalado de datos, usando el método MinMaxScaler, con el fin de normalizar los datos en un rango de [0,1]. Con los datos normalizados se busca encontrar el número de componentes adecuados para aplicar el PCA, para esto nos apoyamos del cross val score, que nos arroja que la cantidad óptima de componentes a usar es 3 y con esto se aplica el ajuste y transformación al conjunto de datos escalados y se definen las variables X, y Y, con las características resultantes y la variable

objetivo respectivamente, para proceder a construir y entrenar los datos con estos resultados obtenidos de la transformación. Donde se obtienen los siguientes resultados.



	Modelo	RMSE	R^2	MAE
0	Random Forest PCA	152.387531	-0.154176	79.817529
1	LightGBM PCA	142.472609	-0.008872	73.473872
2	XGBC PCA	147.869058	-0.086745	75.049414
3	CatBoost PCA	147.869058	-0.086745	75.049414

Donde se evidencia que los resultados obtenidos no son los esperados, ni los mejores, esto puede ser debido a múltiples factores, como la cantidad de componentes selectos, los datos, la manera de tratar los datos faltantes y demás. Pero se tuvo la oportunidad de analizar cómo este tipo de modelos se comportan frente a estos escenarios así no hayan obtenido resultados muy adecuados.

d. Resultados, métricas y curvas de aprendizaje

Las métricas con las cuales mediremos el rendimiento y la capacidad de predecir de un modelo son las anteriores mencionadas RMSE, R^2 y MAE, donde con el RMSE

cuanto menor sea el valor mejor será el modelo, este mide la raíz cuadrada de la media de los errores cuadráticos entre las predicciones del modelo y los valores reales, el R^2 cuanto más cerca de 1, nos indica que será mejor el modelo, donde un R^2 de 1 indica una predicción perfecta, mientras que un R^2 de 0 indica que el modelo no explica nada de la variabilidad. Y por último la MAE, que también entre menor sea el valor, mejor será el modelo, donde se mide mide el promedio de las diferencias absolutas entre las predicciones del modelo y los valores reales

Random Forest

```
RandomForestRegressor(n_jobs=-1, random_state=45)
```

```
[37]  y_pred = clf.predict(X_test)
```

```
[77]  RMSE_random_f = mean_squared_error(y_test, y_pred, squared=False)
      r2_random_f = r2_score(y_test, y_pred)
      MAE_random_f = mean_absolute_error(y_test,y_pred)
      print(f'RMSE Score: {RMSE_random_f}')
      print("R^2 : ",r2_random_f )
      print("MAE :", MAE_random_f)
```

```
RMSE Score: 22.535370367758908
R^2 :  0.9747592356050864
MAE : 8.773575390916621
```

LightGBM

```
LGBMRegressor(random_state=45)
```

```
YLGBM_pred = lgbm_model.predict(XLGBM_test)
```

```
RMSE_lgbm = mean_squared_error(YLGBM_test, YLGBM_pred, squared=False)
r2_lgbm = r2_score(YLGBM_test, YLGBM_pred)
MAE_lgbm = mean_absolute_error(YLGBM_test,YLGBM_pred)

print(f'RMSE Score: {RMSE_lgbm}')
print("R^2 : ", r2_lgbm)
print("MAE :", MAE_lgbm)
```

```
RMSE Score: 29.936194991650158
R^2 :  0.9554583500190286
MAE : 19.632414345534347
```

XGBC

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

```
y_rc_pred = xgb.predict(X_RC_test)
```

```
RMSE_xgbc = mean_squared_error(Y_RC_test, y_rc_pred, squared=False)
r2_xgbc = r2_score(Y_RC_test, y_rc_pred)
MAE_xgbc = mean_absolute_error(Y_RC_test,y_rc_pred)

print(f'RMSE Score: {RMSE_xgbc}')
print("R^2 : ", r2_xgbc)
print("MAE :", MAE_xgbc)
```

```
RMSE Score: 25.709075973684286
R^2 :  0.9657398312435168
MAE : 13.879364612260398
```

CatBoost

```
979:    learn: 23.4140586       total: 35.1s    remaining: 717ms
980:    learn: 23.4029814       total: 35.2s    remaining: 681ms
981:    learn: 23.3790792       total: 35.2s    remaining: 646ms
982:    learn: 23.3744583       total: 35.3s    remaining: 610ms
983:    learn: 23.3680792       total: 35.3s    remaining: 575ms
984:    learn: 23.3655194       total: 35.4s    remaining: 539ms
985:    learn: 23.3620247       total: 35.4s    remaining: 503ms
986:    learn: 23.3557084       total: 35.5s    remaining: 468ms
987:    learn: 23.3388465       total: 35.5s    remaining: 432ms
988:    learn: 23.3340366       total: 35.6s    remaining: 396ms
989:    learn: 23.3308243       total: 35.7s    remaining: 360ms
990:    learn: 23.3105626       total: 35.7s    remaining: 324ms
991:    learn: 23.3047073       total: 35.7s    remaining: 288ms
992:    learn: 23.2993115       total: 35.8s    remaining: 252ms
993:    learn: 23.2938209       total: 35.8s    remaining: 216ms
994:    learn: 23.2818822       total: 35.9s    remaining: 180ms
995:    learn: 23.2672005       total: 35.9s    remaining: 144ms
996:    learn: 23.2594728       total: 36s      remaining: 108ms
997:    learn: 23.2518536       total: 36s      remaining: 72.1ms
998:    learn: 23.2396777       total: 36s      remaining: 36.1ms
999:    learn: 23.2322964       total: 36.1s    remaining: 0us
<catboost.core.CatBoostRegressor at 0x7da0fd431c60>
```

```
YCATBOOST_pred = catboost_model.predict(XCATBOOST_test)
```

```
RMSE_cat_boost = mean_squared_error(YCATBOOST_test, YCATBOOST_pred, squared=False)
r2_cat_boost = r2_score(YCATBOOST_test, YCATBOOST_pred)
MAE_cat_boost = mean_absolute_error(YCATBOOST_test,YCATBOOST_pred)

print(f'RMSE Score: {RMSE_cat_boost}')
print("R^2 : ", r2_cat_boost)
print("MAE :", MAE_cat_boost)
```

```
RMSE Score: 27.09344455326553
R^2 :  0.9650920410511953
MAE : 16.610861247799356
```

Donde en nuestro ejercicio obtuvimos los siguientes resultados consolidados de modelos supervisados.

	Modelo	RMSE	R^2	MAE
0	Random Forest	22.535370	0.974759	8.773575
1	LightGBM	29.936195	0.955458	19.632414
2	XGBC	25.709076	0.965740	13.879365
3	CatBoost	27.093445	0.965092	16.610861

Se puede observar que en general el modelo Random Forest parece describir de mejor manera el problema, con un mejor rendimiento en general con respecto a los otros modelos, donde podemos notar con su R^2 que es capaz de explicar una gran proporción de la variabilidad en la variable objetivo, que en este caso era “emission”, con un RMSE de 22.54 que nos indica la raíz cuadrada de la media de los errores cuadráticos y un MAE de 8.77 que nos habla que en promedio las predicciones están desviadas en 8.77 unidades de la verdad.

En general los otros tres modelos también tuvieron un buen rendimiento y buenos resultados, hubieron algunos modelos que tuvieron mejores resultados en ciertas métricas que otro, pero igual se obtuvieron buenos resultados en general, no están muy alejados entre sí. En este caso decidimos darle más importancia al RMSE, ya que consideramos que es una métrica muy apropiada para la naturaleza del problema de predicción, en este caso el Random Forest es el modelo que más se adapta a las métricas propuestas y esperadas.

4. Retos y consideraciones de despliegue

De los mayores retos que se tuvieron presente durante la elaboración del proyecto, fueron escoger correctamente las métricas posiblemente más apropiadas para el problema de predicción elegido, ya que esto era lo que nos mediría si nuestros modelos eran acertados o no, entonces esa decisión significó un reto importante de entender qué requería nuestro proyecto. También el conocer qué modelos eran mejor para nuestro problema y entender de qué forma abordarlos significó un reto, que conlleva a una tarea de investigar y entender un poco más a profundidad.

5. Conclusiones

Como conclusión podemos rescatar la importancia de la etapa del preprocesamiento de datos, ya que esta puede darle un rumbo completamente diferente a nuestro proyecto, debido a que en este punto se toman decisiones muy importantes como el qué hacer con los datos faltantes, qué hacer con los datos que me generan ruido, y demás incógnitas que deben ser resueltas antes de empezar a trabajar con un modelo. Ya que dependiendo de estas decisiones nuestros datos tendrán ciertos comportamientos que se verán reflejados en los resultados de los modelos, por esto se considera como una etapa clave en este tipo de proyecto.

También como distintos modelos pueden tener mayor o menor grado de acierto respecto a las métricas seleccionadas para calificar los resultados de nuestros modelos, por eso es importante también tener en cuenta qué métrica será la que determinará la eficiencia y rendimiento de nuestro modelo, ya que con esto podemos tener mayor criterio con respecto a qué modelo nos genera mayor acierto sobre otro.

Adicionalmente, para este problema se puede evidenciar cómo aplicar técnicas de escalamiento y trabajar con componentes principales, puede significar mucho cambio con respecto a los datos obtenidos, ya que por ejemplo para este problema, de 76 características se pasaron a 3, lo cuál generó un cambio basta apreciable en el procedimiento y se obtuvieron datos completamente distintos a los generados inicialmente con todas las características, entonces también es sumamente importante conocer y tener en cuenta cómo está compuesto nuestro conjunto de datos, para saber que tan impactante pueden ser aplicar estos métodos a los datos con los que trabajaremos con nuestros modelos.

Con respecto a los algoritmos supervisados, se puede notar nuevamente que no existe mucha diferencia entre los resultados obtenidos, todos son muy coincidentes y presentan comportamientos muy similares, pero como la métrica RMSE, era nuestra métrica selecta para determinar nuestro modelo ideal, por eso por más similares y cercanos que los resultados hayan sido, se nota como el Random Forest

generó mejores resultados respecto a dicha métrica, por lo cual se considera como el modelo más apropiado para el caso expuesto.

A diferencia de los modelos no supervisados, que en su totalidad nos generaron datos demasiado alejados a lo obtenido en los modelos supervisados, esto puede ser debido a la cantidad de componentes usados para el método PCA, o por la naturaleza de los datos, pero fueron resultados para nada positivos, puesto que unos fueron muy elevados como el RMSE y el R^2 en los 4 modelos, nos dio negativo, lo cual es muy mal indicio con respecto a la predicción que realiza.

Finalmente cabe destacar que el procedimiento se llevó a cabo de manera satisfactoria, los modelos supervisados parecen reaccionar de manera correcta a los datos, y predecir los resultados de manera acertada. Adicionalmente también se cuentan con buenas métricas de predicción, que nos indican que los modelos selectos tienen buena capacidad predictiva.