Machine Learning

“Predicción de la calidad del mineral de hierro”

Proyecto Final

Alumno: Céspedes Jaén Abdiel

Experto: Dillan Aguirre Sedeño

Grupo: machine - learning 2103

Fecha de entrega: Jueves 5 de agosto de 2021

Predicción de la calidad del mineral de hierro

Índice

I. Resumen

En el presente documento se pretenden abarcar los aspectos básicos que se tomaron en cuenta a lo largo de la preparación del proyecto del módulo de Machine Learning. El presente documento cuenta con una breve introducción que destaca la motivación detrás de la realización de los procesos, un conjunto de objetivos, un breve marco teórico introductorio, capturas de pantalla del desarrollo del proyecto, así como una serie de conclusiones y enlaces a otros documentos complementarios que también forman parte de la presentación del proyecto final.

II. Introducción

A lo largo de los módulos estudiados en el curso de BEDU, se han puesto en práctica los conocimientos adquiridos a través de las clases a través de diferentes conjuntos de datos, de tal manera que, haciendo una comparativa entre el repositorio al que este proyecto pertenece contra el repositorio del proyecto del curso de Data Analysis, uno puede percatarse del cambio de enfoque y de conjunto de datos que se tuvo. Así pues, debido al interés personal del estudiante de buscar tópicos más interesantes para el desarrollo del proyecto a lo largo del módulo de especialización fue que se decidió explorar otro conjunto de bases de datos.

Asimismo, ya sea en el repositorio o en las bitácoras de proyecto, se puede comprobar que hubo otro cambio en la concepción del proyecto. El primer tipo de proyecto propuesto por el alumno fue con respecto al desarrollo de algoritmos de aprendizaje supervisado que clasificaron si una persona era propensa a tener una cardiopatía o no. Este primer enfoque fue sustituido por el presentado a continuación ya que, debido a la limitada cantidad de datos de la BBDD se cayó en un overfitting de los algoritmos, dando como resultado precisiones extremadamente ideales y ajenas a un enfoque realista. Por esta razón es que se decidió recurrir a una base de datos más numerosa que permitiera poner en práctica los conocimientos adquiridos en BEDU.

2.1. Planteamiento del problema

El presente proyecto corresponde al análisis de una BBDD perteneciente a un proceso de minería, más específicamente, el de un proceso llamado flotación. Más adelante se abordará este tema en particular, por lo pronto el problema puede igualmente ser encontrado en la página de Kaggle a la cual fue subidos y consiste en lo siguiente:

> “El objetivo principal es usar estos datos para predecir cuánta impureza hay en el concentrado de mineral” (EduardoMagalhaesOliveira, 2017).

La justificación para resolver el problema en particular radica en que, dicho igualmente en palabras del autor: “[...], si podemos predecir cuánto sílice (impureza) hay en el concentrado de mineral, podemos ayudar a los ingenieros, proporcionándoles la información para tomar acción. Por ello serán capaces de tomar acciones preventivas [...].” (EduardoMagalhaesOliveira, 2017).

Asimismo, el uso de machine learning está más que justificado por la imperiosa necesidad de generar modelos predictivos que sean capaces de proveer la información adecuada que permita a los ingenieros tomar decisiones en función de una estimación a futuro y evitar la generación de impureza en los concentrados de mineral de hierro. De acuerdo con Desconocido (2019), quien a su vez cita a Francisco Santibáñez, “general” de DataQu, explica que: “con este tipo de modelos es posible entender comportamiento de datos, detectar patrones y, en función a eso, realizar sugerencias a las compañías, las cuales pueden ser utilizadas en su toma de decisiones”.

2.2. Objetivos

A continuación, se enlistan un conjunto de objetivos orientados a la resolución del problema planteado y que radican en el uso de los conocimientos adquiridos a lo largo del módulo de Machine Learning.

2.2.1. General

- Poner en prácticas los conocimientos adquiridos a lo largo del módulo de Machine Learning.

2.2.2. Específicos

- Identificar el tipo de aprendizaje, así como el modelo, necesario para dar solución a lo solicitado.
- Contrastar diferentes tipos de algoritmos de Machine Learning para poder justificar la selección de un mejor modelo.
- Determinar cuál es el mejor algoritmo para predecir la cantidad de impureza hay en un concentrado de mineral.

III. Marco teórico

3.1. Los materiales

3.1.1. El mineral de hierro

De acuerdo con el Dr. Gaucher (2014), el mineral de hierro es “una concentración natural de metal que se puede explotar económicamente para extraer el hierro”. Este a su vez forma el acero.

De acuerdo con Katz (2011) el hierro se encuentra en numerosos minerales, pero solo pocos de estos son los que se usan de manera comercial para obtener este material. El hierro puede encontrarse en diversos minerales y “mineraloides” como, por ejemplo: “magnetita (Fe3 O4), hematita (Fe2 O3), limonita (Fe2 O3 nH2 O), siderita (FeCO3), pirita (FeS2), etc.” (Katz, 2011). He ahí la necesidad de aplicar métodos para remover las impurezas, como el que nos concierne en el análisis de este proyecto y en el que la impureza que nos concierne es el sílice.

3.1.2. Hidrofílicos e hidrofóbicos

Hidrofílico: Propiedad que presentan aquellos materiales que se pueden mezclar con el agua, según Desconocido (2014). Algunos ejemplos son sulfatos, carbonatos y fosfatos.

Hidrofóbico: Propiedad que presentan aquellos materiales que no se pueden mezclar con el agua, según Desconocido (2014). Algunos ejemplos son el talco, el carbón y el grafito.

Igualmente, de acuerdo a este mismo autor, “el que una molécula sea hidrofílica o sea hidrofóbica podríamos decir que dependerá básicamente de su carga o polaridad. El agua es una molécula polar, tiene carga positiva y negativa (osea es un dipolo) y sólo se juntará con otras sustancias que tengan también carga, un ejemplo visual de este comportamiento serían los imanes. En cambio las moléculas hidrofóbicas son apolares, osea que no tienen carga, y por eso no se mezclan con moléculas polares como el agua.” (Desconocido, 2014).

De acuerdo con NORTE MINERO TV (2019), los factores que afectan la flotación son:

- El porcentaje de sólidos en la pulpa.
- Granulometría.
- Reactivos.
- Agitación.
- Aire.

3.2. Proceso de flotación

De acuerdo con NORTE MINERO TV (2019), “la flotación es un proceso que separa a los minerales valiosos de material estéril”, esto depende de si los materiales que entran en contactos con ciertos contactos químicos se hunden (hidrofílicos) o no (hidrofóbicos). Se requiere hacer uso de los siguientes reactivos de acuerdo con NORTE MINERO TV (2019):

1. **Espumantes:** Forman una cama de burbujas de aire para mantener sustancias de valor en la superficie de la celda.
2. **Colectores:** Modifican ángulos de contacto de minerales con el agua y evitan que se mojen para que las burbujas de aire las arrastren a la superficie.
3. **Activadores:** Vuelven flotables a los sulfuros valiosos.
4. **Depresores:** Mantienen en la fase acuosa a ciertos minerales que son flotables en la etapa inicial.
5. **Modificadores:** Permiten el cambio del medio acuoso por medio de dispersantes y reguladores de pH.

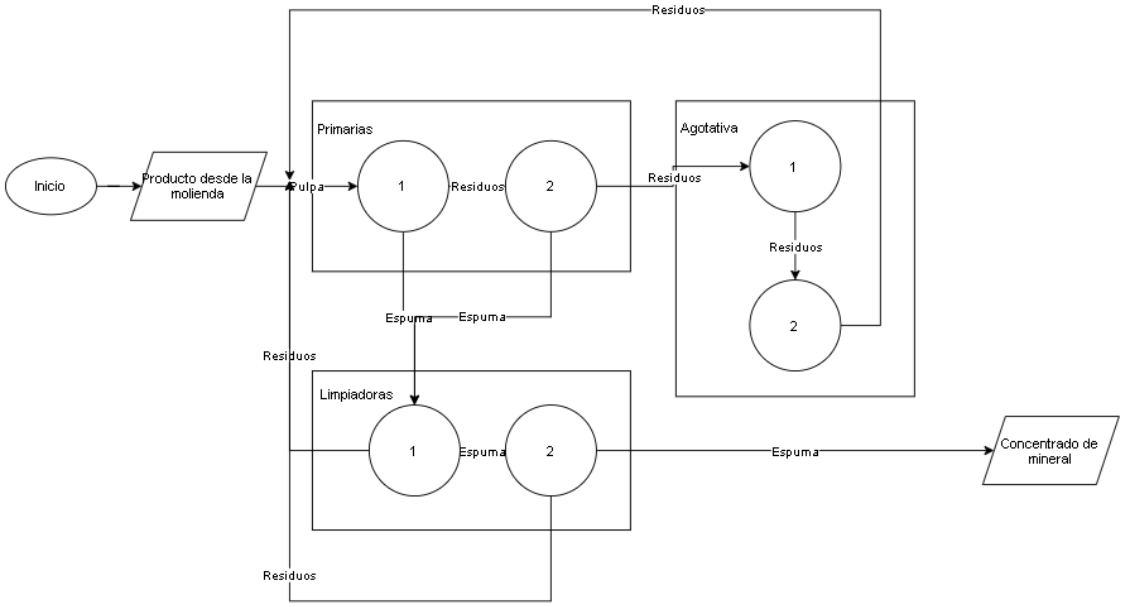
El aire es el que permite la formación de burbujas, lo cual a su vez permite que las burbujas puedan arrastrar ciertos elementos a la superficie.

3.2.1. Etapas del proceso

De acuerdo con NORTE MINERO TV (2019), el proceso se compone por varias celdas que se clasifican por sus funciones:

1. **Primarias:** Comienza el proceso en donde se genera espuma.
2. **Limpiadoras:** Se almacena la espuma de las celdas primarias y permite eliminar las partículas indeseables que flotan con ciertos sulfuros valiosos.
3. **Agotativas:** Se encargan de tener bajas “las colas” del primer grupo para mantener ciertos valores para que no se pierdan.

La pulpa de mineral debe pasar por todo el proceso para evitar desperdicios. A continuación, se presenta un diagrama de flujo de ejemplo del proceso:



3.3. Estructura de la BBDD

La base de datos está compuesta por 737,453 registros en donde cada una de estas observaciones tiene 24 características. La primera de ellas corresponde a la fecha en que el registro fue capturado, los demás ya pertenecen propiamente a valores tipo *float* como se menciona a continuación. Gracias a EduardoMagalhaesOliveira (2017) sabemos que la estructura de la base de datos es la siguiente:

La segunda y tercera columnas son las que corresponden al porcentaje de alimentación de hierro y de sílice, respectivamente, medidas de calidad justo antes de que la pulpa de mineral de hierro sea enviada a la planta de flotación.

El autor describe las columnas 4 a 8 como las que representan a las variables más importantes que impactan en la calidad del mineral al final del proceso. Estas columnas son:

- Flujo de almidón.
- Flujo de amina.
- Flujo de pulpa de mineral.
- Ph de pulpa de mineral.
- Densidad de la pulpa de mineral.

El resto de las columnas representan datos capturados a lo largo del proceso (en el tiempo que representa la columna 1), estas columnas representan los niveles de flujo de aire dentro de las columnas de flotación. Las últimas dos columnas representan las medidas de la calidad de la pulpa de mineral. El objetivo señalado

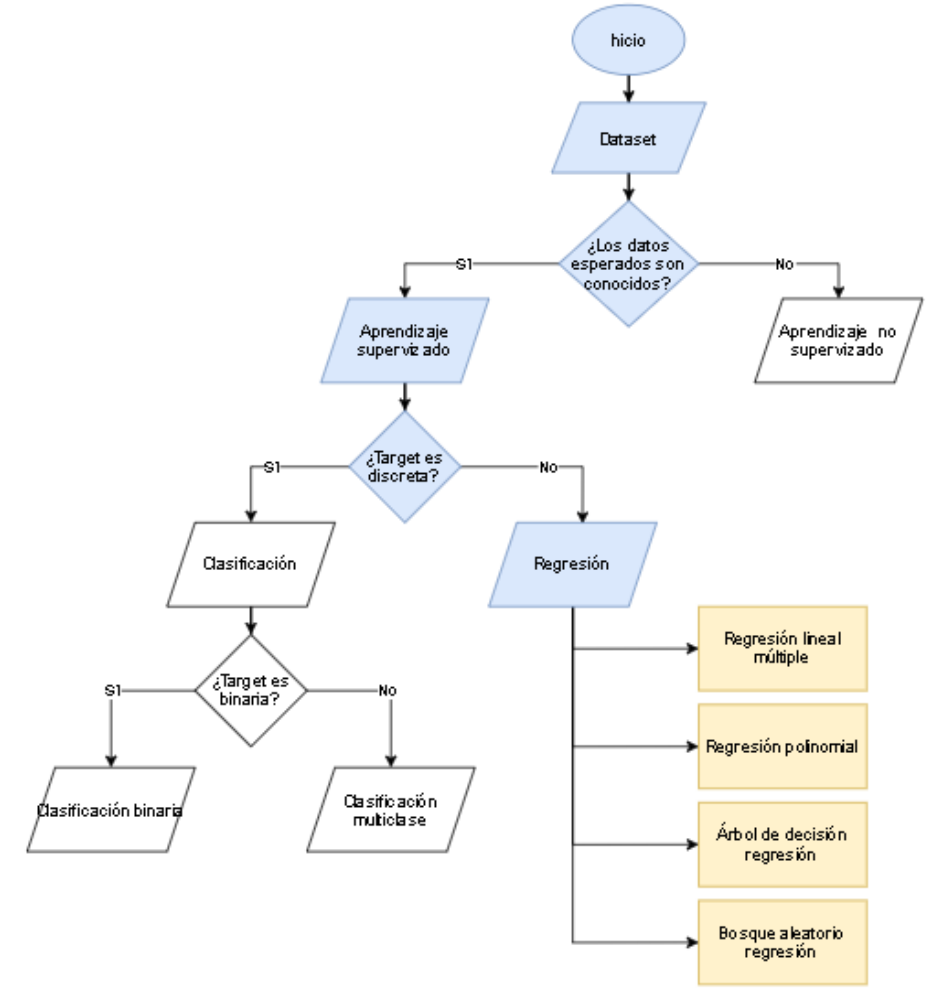
por el autor es el de poder predecir la última columna, la cual corresponde al porcentaje de sílice (la impureza) en el concentrado de mineral.

3.4. Selección de algoritmos

En el siguiente diagrama de flujo se puede observar el proceso por medio del cual se eligió el tipo de aprendizaje y los algoritmos correspondientes a ser evaluados a lo largo de este proyecto. Al contar con una variable objetivo (*target*) sabemos que el aprendizaje de debe ser de tipo **supervisado** ya que conocemos los resultados que obtendremos, por otra parte, el tipo de dato de la variable objetivo fue lo que ayudó a definir el siguiente apartado.

Al ser la variable objetivo de tipo continua, no se puede aplicar un enfoque de clasificación, sino más bien de **regresión** para poder predecir los valores solicitados en función de las características observadas y registradas en la base de datos. Así pues, los algoritmos empleados en el proyecto son:

- Regresión lineal múltiple.
- Regresión polinomial.
- Árbol de decisión regresión.
- Bosque aleatorio regresión.


hicio
Dataset
¿Los datos esperados son conocidos?
S1
No
Aprendizaje supervizado
Aprendizaje no supervizado
¿Target es discreta?
S1
No
Clasificación
Regresión
¿Target es binaria?
S1
No
Clasificación binaria
Clasificación multiclase
Regresión lineal múltiple
Regresión polinomial
Árbol de decisión regresión
Bosque aleatorio regresión

IV. Desarrollo

4.1. Análisis de datos (EDA)

4.1.1. Librerías

Librerías ¶

```
# Análisis de Datos
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Machine Learning
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PowerTransformer
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import roc_curve, auc
from sklearn.tree import DecisionTreeRegressor
import warnings
warnings.filterwarnings('ignore')
```

4.1.2. Limpieza de BBDD y tipos de datos

Limpieza de BBDD y tipos de datos ¶

```
df = pd.read_csv('MiningProcess.csv')
df.head()
```

	date	porcentaje_iron_feed	porcentaje_silica_feed	starch_flow	amina_flow	ore_pulp_flow	ore_pulp_pH	ore_pulp_density	flotation_column_01_air_flo
0	42804.04167	55,2	16,98	3019,53	557434	395713	100664	1,74	2492
1	42804.04167	55,2	16,98	3024,41	563965	397383	100672	1,74	2497
2	42804.04167	55,2	16,98	3043,46	568054	399668	10068	1,74	2497
3	42804.04167	55,2	16,98	3047,36	568665	397939	100689	1,74	2499
4	42804.04167	55,2	16,98	3033,69	558167	400254	100697	1,74	2502

5 rows × 23 columns

```
df.shape
```

```
(737453, 23)
```

```
df.isna().sum()
```

```
date                          0
porcentaje_iron_feed          0
porcentaje_silica_feed        0
starch_flow                   0
amina_flow                    0
ore_pulp_flow                 0
ore_pulp_pH                   0
ore_pulp_density              0
flotation_column_01_air_flow  0
flotation_column_02_air_flow  0
flotation_column_03_air_flow  0
flotation_column_04_air_flow  0
flotation_column_05_air_flow  0
flotation_column_06_air_flow  0
flotation_column_01_level     0
flotation_column_02_level     0
flotation_column_03_level     0
flotation_column_04_level     0
flotation_column_05_level     0
flotation_column_06_level     0
flotation_column_07_level     0
porcentaje_iron_concentrate   0
porcentaje_silica_concentrate 0
dtype: int64
```

```
df.dtypes
```

```
date                          float64
porcentaje_iron_feed           object
porcentaje_silica_feed         object
starch_flow                    object
amina_flow                     object
ore_pulp_flow                  object
ore_pulp_pH                    object
ore_pulp_density               object
flotation_column_01_air_flow   object
flotation_column_02_air_flow   object
flotation_column_03_air_flow   object
flotation_column_04_air_flow   object
flotation_column_05_air_flow   object
flotation_column_06_air_flow   object
flotation_column_01_level      object
flotation_column_02_level      object
flotation_column_03_level      object
flotation_column_04_level      object
flotation_column_05_level      object
flotation_column_06_level      object
flotation_column_07_level      object
porcentaje_iron_concentrate    object
porcentaje_silica_concentrate  object
dtype: object
```

```
df['porcentaje_iron_feed'] = df['porcentaje_iron_feed'].str.replace(",",".")
df['porcentaje_iron_feed']
```

```
0          55.2
1          55.2
2          55.2
3          55.2
4          55.2
          ...
737448    49.75
737449    49.75
737450    49.75
737451    49.75
737452    49.75
Name: porcentaje_iron_feed, Length: 737453, dtype: object
```

```
diccionario_de_conversion = {
    'date': 'datetime64[ns]',
    'porcentaje_iron_feed': 'float',
    'porcentaje_silica_feed': 'float',
    'starch_flow': 'float',
    'amina_flow': 'float',
    'ore_pulp_flow': 'float',
    'ore_pulp_pH': 'float',
    'ore_pulp_density': 'float',
    'flotation_column_01_air_flow': 'float',
    'flotation_column_02_air_flow': 'float',
    'flotation_column_03_air_flow': 'float',
    'flotation_column_04_air_flow': 'float',
    'flotation_column_05_air_flow': 'float',
    'flotation_column_06_air_flow': 'float',
    'flotation_column_01_level': 'float',
    'flotation_column_02_level': 'float',
    'flotation_column_03_level': 'float',
    'flotation_column_04_level': 'float',
    'flotation_column_05_level': 'float',
    'flotation_column_06_level': 'float',
    'flotation_column_07_level': 'float',
    'porcentaje_iron_concentrate': 'float',
    'porcentaje_silica_concentrate': 'float'
}
```

```
df_nuevo.dtypes
```

```
date                          datetime64[ns]
porcentaje_iron_feed                 float64
porcentaje_silica_feed               float64
starch_flow                          float64
amina_flow                           float64
ore_pulp_flow                        float64
ore_pulp_pH                          float64
ore_pulp_density                     float64
flotation_column_01_air_flow         float64
flotation_column_02_air_flow         float64
flotation_column_03_air_flow         float64
flotation_column_04_air_flow         float64
flotation_column_05_air_flow         float64
flotation_column_06_air_flow         float64
flotation_column_01_level            float64
flotation_column_02_level            float64
flotation_column_03_level            float64
flotation_column_04_level            float64
flotation_column_05_level            float64
flotation_column_06_level            float64
flotation_column_07_level            float64
porcentaje_iron_concentrate          float64
porcentaje_silica_concentrate        float64
dtype: object
```

4.1.3. Análisis exploratorio

4.1.3.1. Medidas de tendencia central

Medidas de tendencia central

```
# Porcentaje de Iron Feed
print(f"La media es: {round(df_nuevo['porcentaje_iron_feed'].mean(), 2)}")
print(f"La mediana es: {df_nuevo['porcentaje_iron_feed'].median()}")
print(f"La moda es: {df_nuevo['porcentaje_iron_feed'].mode()}")
```

```
La media es: 56.29
La mediana es: 56.08
La moda es: 0    64.03
dtype: float64
```

```
# Porcentaje de Silica Feed
print(f"La media es: {round(df_nuevo['porcentaje_silica_feed'].mean(), 2)}")
print(f"La mediana es: {df_nuevo['porcentaje_silica_feed'].median()}")
print(f"La moda es: {df_nuevo['porcentaje_silica_feed'].mode()}")
```

```
La media es: 14.65
La mediana es: 13.85
La moda es: 0    6.26
dtype: float64
```

```
# Starch Flow
print(f"La media es: {round(df_nuevo['starch_flow'].mean(), 2)}")
print(f"La mediana es: {df_nuevo['starch_flow'].median()}")
print(f"La moda es: {df_nuevo['starch_flow'].mode()}")
```

```
La media es: 1339411786919.71
La mediana es: 3518.82
La moda es: 0    2562.5
```

4.1.3.2. Medidas de dispersión

Medidas de dispersión

```
# Porcentaje de Iron Feed
maximo_iron_feed = df_nuevo['porcentaje_iron_feed'].max()
minimo_iron_feed = df_nuevo['porcentaje_iron_feed'].min()

print(f"El valor máximo es: {maximo_iron_feed}")
print(f"El valor mínimo es: {minimo_iron_feed}")
print(f"El rango es: {maximo_iron_feed - minimo_iron_feed}")
print(f"La desviación estándar es: {round(df_nuevo['porcentaje_iron_feed'].std(), 2)}")
```

```
El valor máximo es: 65.78
El valor mínimo es: 42.74
El rango es: 23.04
La desviación estándar es: 5.16
```

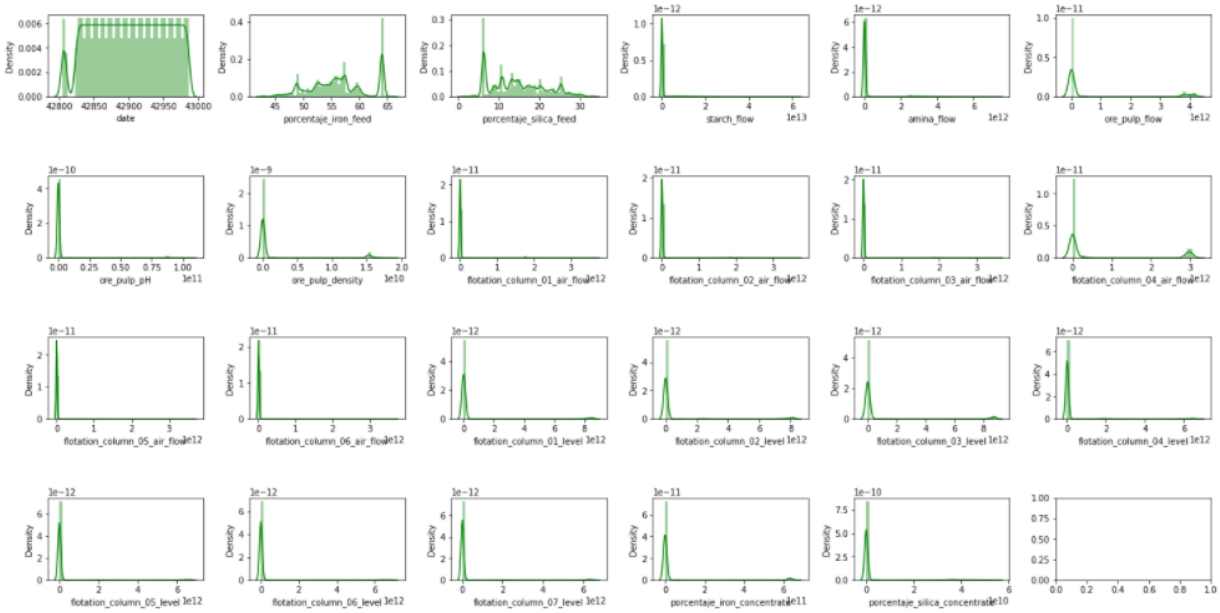
```
# Porcentaje Silica Feed
maximo_silica_feed = df_nuevo['porcentaje_silica_feed'].max()
minimo_silica_feed = df_nuevo['porcentaje_silica_feed'].min()

print(f"El valor máximo es: {maximo_silica_feed}")
print(f"El valor mínimo es: {minimo_silica_feed}")
print(f"El rango es: {maximo_silica_feed - minimo_silica_feed}")
print(f"La desviación estándar es: {round(df_nuevo['porcentaje_silica_feed'].std(), 2)}")
```

```
El valor máximo es: 33.4
El valor mínimo es: 1.31
El rango es: 32.089999999999996
La desviación estándar es: 6.81
```

4.1.3.3. Visualizaciones

Visualizaciones

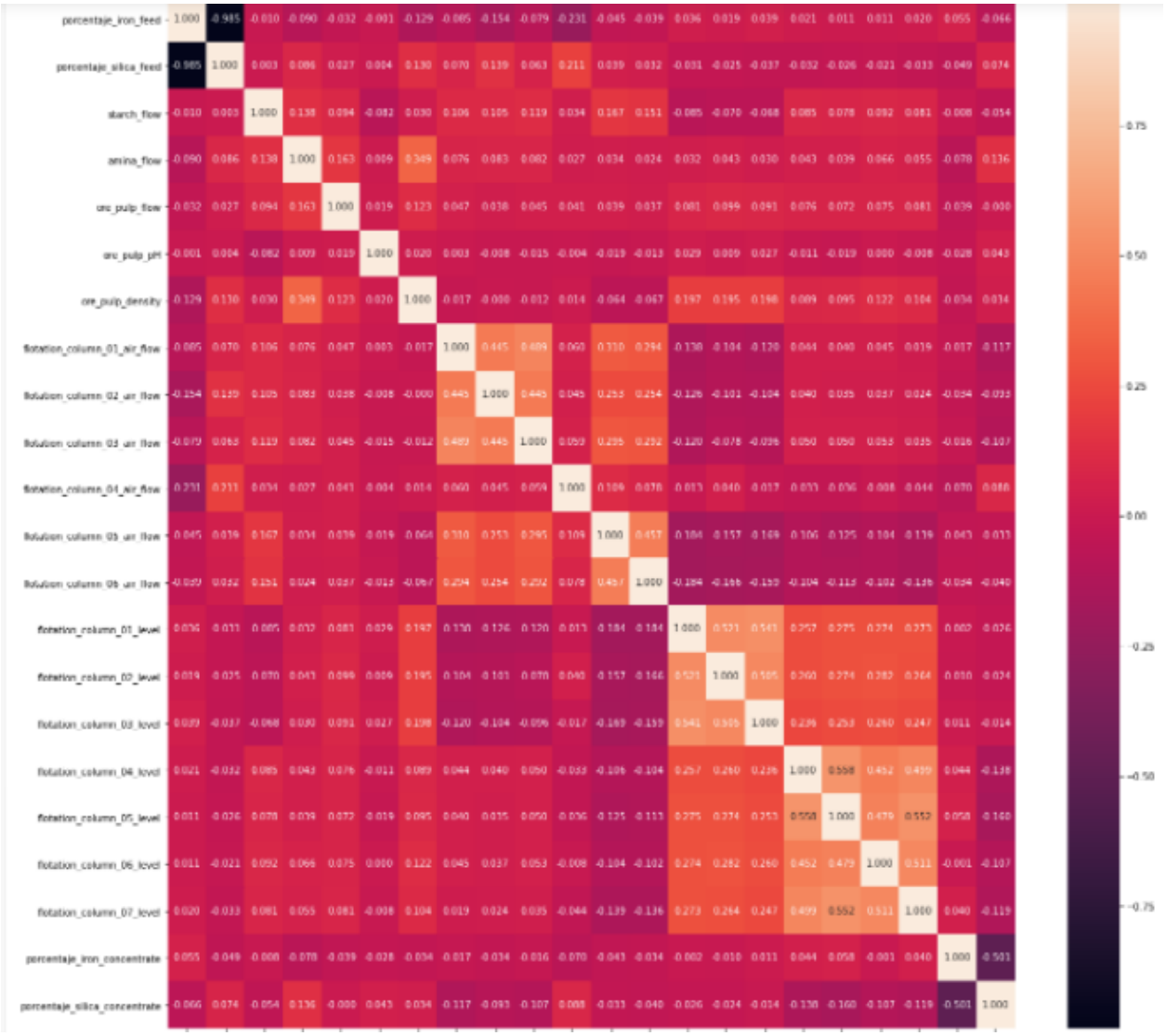
```
fig, ax = plt.subplots(ncols=6, nrows=4, figsize=(20,10))
index = 0
ax = ax.flatten()

for col, value in df_nuevo.items():
    sns.distplot(value, color='g', ax=ax[index])
    index += 1
plt.tight_layout(pad=0.5, w_pad=0.7, h_pad=5.0)
plt.show()
```



4.1.3.4. Correlaciones

Correlaciones

```
corr_mining = df_nuevo.corr(method = 'spearman')
fig, ax = plt.subplots(figsize=(20, 20))
sns.heatmap(corr_mining, annot = True, fmt=".3f").set_title("Mining Process")
plt.show()
```

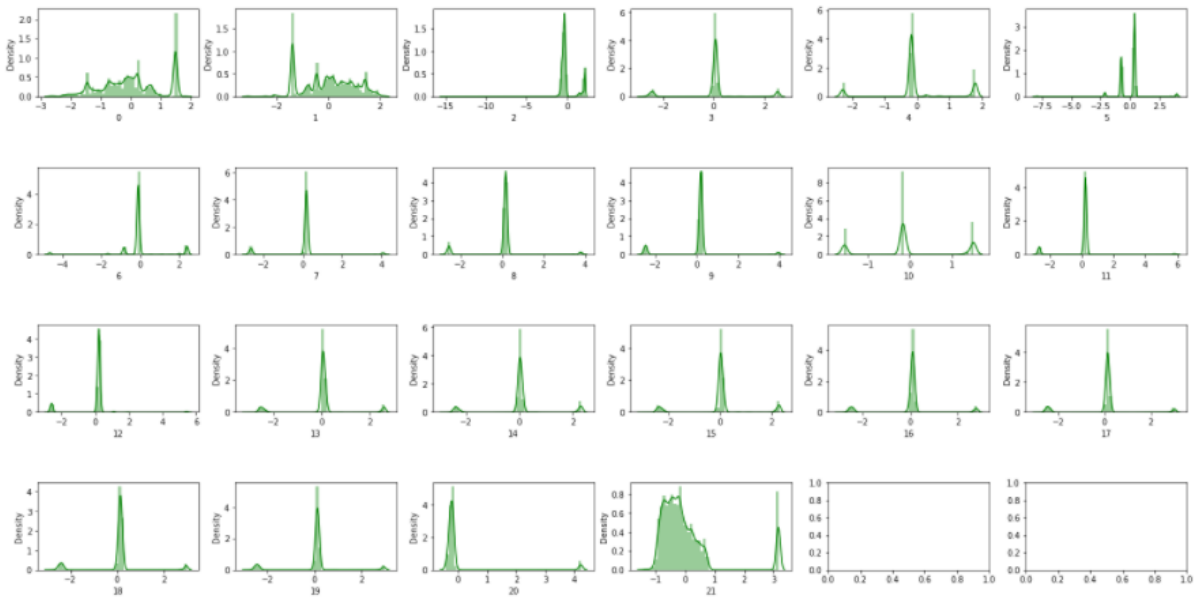

percentaje_iron_feed
percentaje_silica_feed
starch_flow
amina_flow
ore_pulp_flow
ore_pulp_ph
ore_pulp_density
flotation_column_01_air_flow
flotation_column_02_air_flow
flotation_column_03_air_flow
flotation_column_04_air_flow
flotation_column_05_air_flow
flotation_column_06_air_flow
flotation_column_01_level
flotation_column_02_level
flotation_column_03_level
flotation_column_04_level
flotation_column_05_level
flotation_column_06_level
flotation_column_07_level
percentaje_iron_concentrate
percentaje_silica_concentrate
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75

4.2. Machine learning

4.2.1. Estandarización de variables

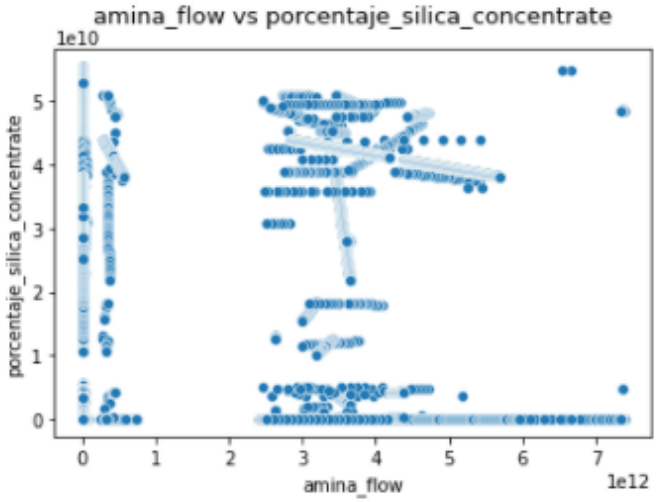
```
ss = PowerTransformer()
X_scaled = ss.fit_transform(df_2)

df_scaled = pd.DataFrame(X_scaled)
print(df_scaled.describe().T.round(2))
```

	count	mean	std	min	25%	50%	75%	max
0	737453.0	0.0	1.0	-2.66	-0.70	-0.04	0.67	1.83
1	737453.0	-0.0	1.0	-3.11	-0.80	0.04	0.80	2.15
2	737453.0	0.0	1.0	-15.43	-0.51	-0.35	-0.20	2.20
3	737453.0	0.0	1.0	-2.80	0.02	0.07	0.10	2.59
4	737453.0	0.0	1.0	-2.32	-0.18	-0.17	-0.17	1.80
5	737453.0	-0.0	1.0	-8.08	-0.75	0.38	0.40	4.04
6	737453.0	0.0	1.0	-4.70	-0.11	-0.10	-0.10	2.44
7	737453.0	-0.0	1.0	-2.87	0.14	0.20	0.20	4.19
8	737453.0	0.0	1.0	-2.77	0.14	0.20	0.21	3.88
9	737453.0	-0.0	1.0	-2.65	0.16	0.22	0.22	4.05
10	737453.0	0.0	1.0	-1.58	-0.17	-0.16	-0.16	1.52
11	737453.0	-0.0	1.0	-2.87	0.16	0.23	0.24	5.95
12	737453.0	-0.0	1.0	-2.78	0.16	0.23	0.23	5.52
13	737453.0	-0.0	1.0	-3.08	0.02	0.07	0.12	2.60
14	737453.0	0.0	1.0	-2.77	-0.01	0.03	0.08	2.32
15	737453.0	-0.0	1.0	-3.00	-0.03	0.02	0.08	2.29
16	737453.0	-0.0	1.0	-2.91	0.05	0.10	0.15	2.79
17	737453.0	0.0	1.0	-2.88	0.09	0.13	0.18	3.04
18	737453.0	-0.0	1.0	-2.88	0.09	0.13	0.19	2.98
19	737453.0	-0.0	1.0	-2.86	0.08	0.12	0.17	2.99
20	737453.0	-0.0	1.0	-0.42	-0.27	-0.22	-0.19	4.22
21	737453.0	0.0	1.0	-1.40	-0.59	-0.25	0.20	3.15

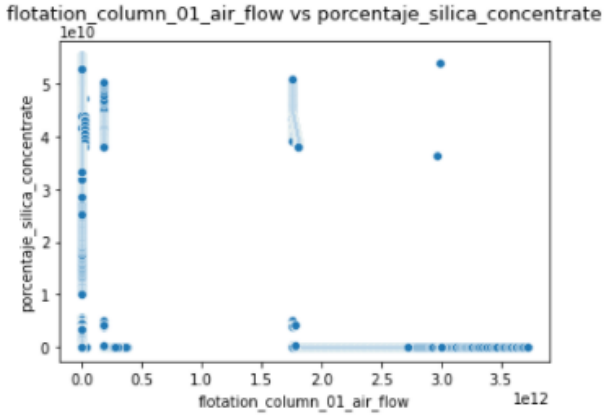


4.2.2. Scatterplots de los datos más correlacionados

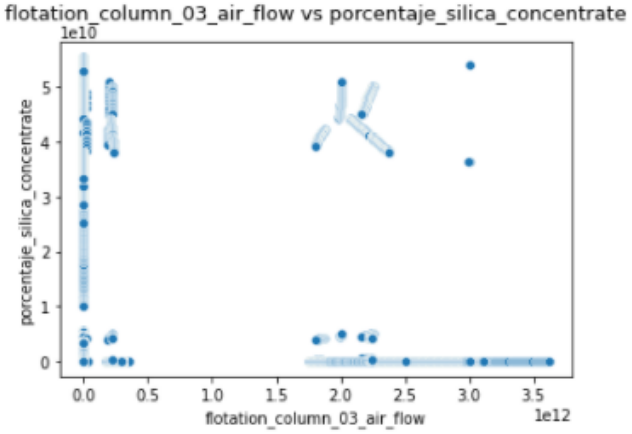
```
ax = sns.scatterplot(df_2['amina_flow'], df_2['porcentaje_silica_concentrate'])
ax.set_title('amina_flow vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



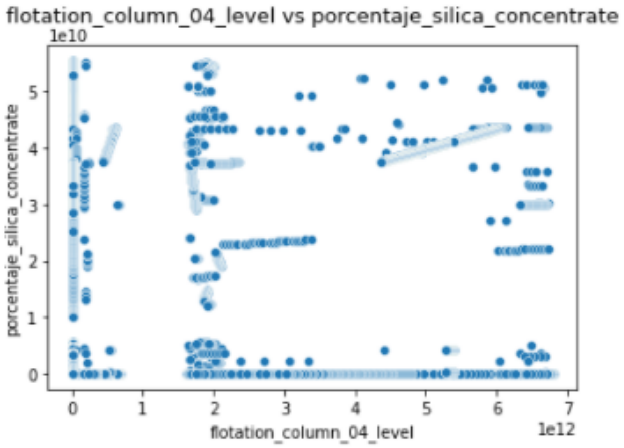

```
ax = sns.scatterplot(df_2['flotation_column_01_air_flow'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_01_air_flow vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



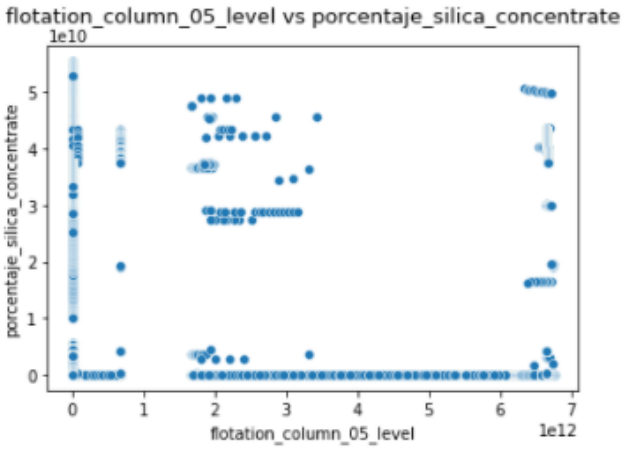

```
ax = sns.scatterplot(df_2['flotation_column_03_air_flow'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_03_air_flow vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



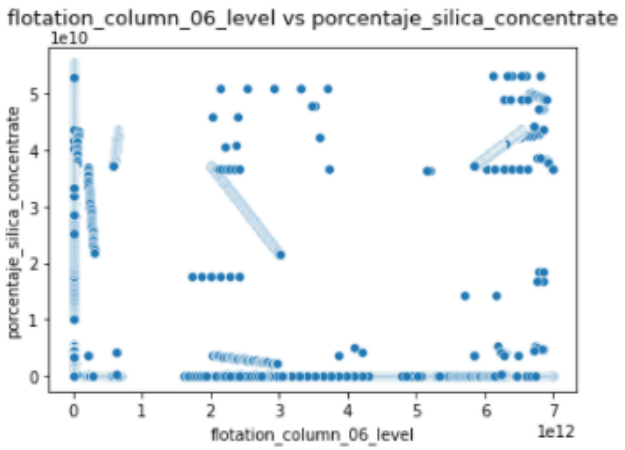
```
ax = sns.scatterplot(df_2['flotation_column_04_level'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_04_level vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



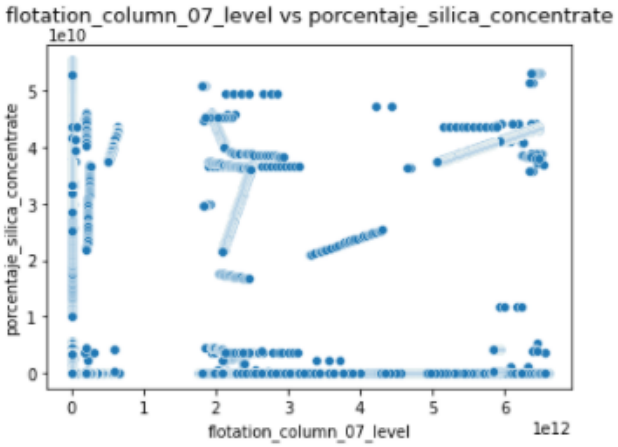

```
ax = sns.scatterplot(df_2['flotation_column_05_level'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_05_level vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



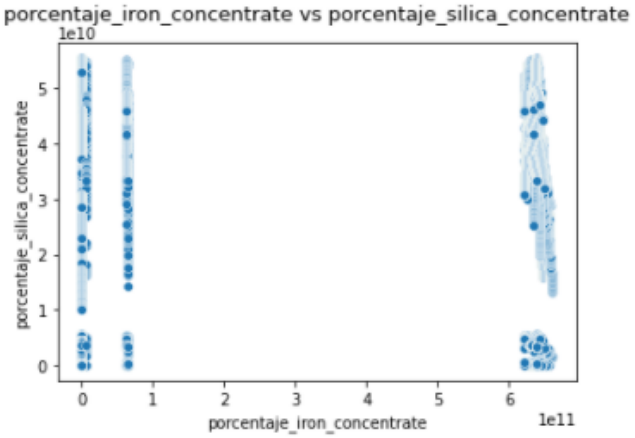

```
ax = sns.scatterplot(df_2['flotation_column_06_level'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_06_level vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



```
ax = sns.scatterplot(df_2['flotation_column_07_level'], df_2['porcentaje_silica_concentrate'])
ax.set_title('flotation_column_07_level vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```




```
ax = sns.scatterplot(df_2['porcentaje_iron_concentrate'], df_2['porcentaje_silica_concentrate'])
ax.set_title('porcentaje_iron_concentrate vs porcentaje_silica_concentrate', fontsize=13, pad=15);
plt.show()
```



4.2.3. Separación y evaluación

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, df_2['porcentaje_silica_concentrate'], test_size = 0.4, random_sta

print("X_train shape: "+str(X_train.shape[0]), "y_train shape: "+str(y_train.shape[0]), sep = "\n")
print("X_test shape:"+str(X_test.shape[0]), "y_test shape:"+str(y_test.shape[0]), sep = "\n")
```

```
X_train shape: 442471
y_train shape: 442471
X_test shape:294982
y_test shape:294982
```

4.2.4. Aprendizajes supervisados

4.2.4.1. Regresión

4.2.4.1.1. Lineal múltiple

Regresión

Lineal (con todos los datos)

```
# Algoritmo
lin_model = LinearRegression(normalize = True)
lin_model.fit(X_train, y_train)
```

```
LinearRegression(normalize=True)
```

```
# Evaluación del modelo
y_train_predict = lin_model.predict(X_train)
MSE = mean_squared_error(y_train,y_train_predict)
print("Entrenamiento: MSE ="+str(MSE))

y_test_predict = lin_model.predict(X_test)
MSE = (mean_squared_error(y_test, y_test_predict))
print("Pruebas: MSE ="+str(MSE))
```

```
Entrenamiento: MSE =2.695229066363164e+19
Pruebas: MSE =2.683815683916006e+19
```

	valor_real	prediccion	diferencia
0	3.34	3.697264e+09	-3.697264e+09
1	1.00	-4.230241e+09	4.230241e+09
2	2.08	4.308655e+08	-4.308655e+08
3	1.68	2.932210e+08	-2.932210e+08
4	2.30	1.192761e+09	-1.192761e+09
5	1.01	-4.312483e+09	4.312483e+09
6	1.10	-3.364828e+09	3.364828e+09
7	4.26	4.944361e+09	-4.944361e+09
8	4.12	5.334801e+09	-5.334801e+09
9	0.92	-4.936594e+09	4.936594e+09

	valor_real	prediccion	diferencia
294972	1.060000e+00	-3.714992e+09	3.714992e+09
294973	1.530000e+00	-1.679312e+09	1.679312e+09
294974	1.450000e+00	-2.271594e+09	2.271594e+09
294975	1.740000e+00	-6.146434e+07	6.146434e+07
294976	2.030000e+00	1.417540e+09	-1.417540e+09
294977	1.490000e+00	-1.826274e+09	1.826274e+09
294978	4.670000e+00	6.218221e+09	-6.218221e+09
294979	3.877795e+10	3.587803e+10	2.899921e+09
294980	1.310000e+00	-2.573018e+09	2.573018e+09
294981	1.850000e+00	2.537148e+08	-2.537148e+08

4.2.4.1.2. Polinomial

Polinomial (con todos los datos)

```
# Algoritmo
poly_model = LinearRegression()
poly = PolynomialFeatures(degree=2)

Xpolytrain = poly.fit_transform(X_train)
Xpolytest = poly.fit_transform(X_test)

poly_model.fit(Xpolytrain, y_train)
y_train_predict = poly_model.predict(Xpolytrain)
```

```
# Evaluación del modelo
MSE = mean_squared_error(y_train,y_train_predict)
print("Entrenamiento: MSE ="+str(MSE))

y_test_predict = poly_model.predict(Xpolytest)
MSE = (mean_squared_error(y_test, y_test_predict))
print("Pruebas: MSE ="+str(MSE))
```

```
Entrenamiento: MSE =1.359648231879702e+19
Pruebas: MSE =1.3545576697806381e+19
```

	valor_real	prediccion	diferencia
0	3.34	2.087171e+07	-2.087170e+07
1	1.00	7.026453e+08	-7.026453e+08
2	2.08	-1.244742e+08	1.244742e+08
3	1.68	-6.140421e+08	6.140421e+08
4	2.30	-2.945550e+08	2.945550e+08
5	1.01	1.284578e+09	-1.284578e+09
6	1.10	1.000231e+09	-1.000231e+09
7	4.26	1.091727e+09	-1.091727e+09
8	4.12	1.045812e+09	-1.045812e+09
9	0.92	1.335229e+09	-1.335229e+09

	valor_real	prediccion	diferencia
294972	1.060000e+00	1.305665e+09	-1.305665e+09
294973	1.530000e+00	-2.454203e+08	2.454203e+08
294974	1.450000e+00	-2.033205e+08	2.033205e+08
294975	1.740000e+00	-1.139930e+09	1.139930e+09
294976	2.030000e+00	-9.082002e+08	9.082002e+08
294977	1.490000e+00	4.736565e+08	-4.736565e+08
294978	4.670000e+00	2.746703e+09	-2.746703e+09
294979	3.877795e+10	3.508870e+10	3.689242e+09
294980	1.310000e+00	2.795050e+08	-2.795050e+08
294981	1.850000e+00	-2.036224e+08	2.036224e+08

4.2.4.2. Árbol de decisión regresión

```
# Algoritmo
adr = DecisionTreeRegressor(max_depth = 5)
adr.fit(X_train, y_train)
y_predict_adr = adr.predict(X_test)
print(y_predict_adr)
```

```
[2.78000327e+03 2.78000327e+03 2.78000327e+03 ... 3.80901842e+10
 2.78000327e+03 2.78000327e+03]
```

```
# Evaluación del modelo
print('Precisión del modelo de entrenamiento: ', adr.score(X_train, y_train))
print('Precisión del modelo de prueba: ', adr.score(X_test, y_test))
```

```
Precisión del modelo de entrenamiento:  0.9997852290438421
Precisión del modelo de prueba:  0.9997801911363925
```

	valor_real	prediccion	diferencia
0	3.34	2780.00327	-2776.66327
1	1.00	2780.00327	-2779.00327
2	2.08	2780.00327	-2777.92327
3	1.68	2780.00327	-2778.32327
4	2.30	2780.00327	-2777.70327
5	1.01	2780.00327	-2778.99327
6	1.10	2780.00327	-2778.90327
7	4.26	2780.00327	-2775.74327
8	4.12	2780.00327	-2775.88327
9	0.92	2780.00327	-2779.08327

	valor_real	prediccion	diferencia
294972	1.060000e+00	2.780003e+03	-2.778943e+03
294973	1.530000e+00	2.780003e+03	-2.778473e+03
294974	1.450000e+00	2.780003e+03	-2.778553e+03
294975	1.740000e+00	2.780003e+03	-2.778263e+03
294976	2.030000e+00	2.780003e+03	-2.777973e+03
294977	1.490000e+00	2.780003e+03	-2.778513e+03
294978	4.670000e+00	2.780003e+03	-2.775333e+03
294979	3.877795e+10	3.809018e+10	6.877631e+08
294980	1.310000e+00	2.780003e+03	-2.778693e+03
294981	1.850000e+00	2.780003e+03	-2.778153e+03

4.2.4.3. Bosque aleatorio regresión

```
# Algoritmo
bar = RandomForestRegressor(n_estimators = 300, max_depth = 8)
bar.fit(X_train, y_train)
y_predict = bar.predict(X_test)
print(y_predict)
```

```
[2.19910565e+00 2.18540781e+00 2.18540781e+00 ... 3.87444376e+10
 2.18540781e+00 2.18540781e+00]
```

```
# Evaluación del modelo
print('Precisión del modelo de entrenamiento: ', bar.score(X_train, y_train))
print('Precisión del modelo de prueba: ', bar.score(X_test, y_test))
```

```
Precisión del modelo de entrenamiento:  0.9999950599159596
Precisión del modelo de prueba:  0.9999949046646498
```

	valor_real	prediccion	diferencia
0	3.34	2.199106	1.140894
1	1.00	2.185408	-1.185408
2	2.08	2.185408	-0.105408
3	1.68	2.185408	-0.505408
4	2.30	2.185408	0.114592
5	1.01	2.185408	-1.175408
6	1.10	2.185408	-1.085408
7	4.26	2.199106	2.060894
8	4.12	2.199106	1.920894
9	0.92	2.185408	-1.265408

	valor_real	prediccion	diferencia
294972	1.060000e+00	2.185408e+00	-1.125408e+00
294973	1.530000e+00	2.185408e+00	-6.554078e-01
294974	1.450000e+00	2.185408e+00	-7.354078e-01
294975	1.740000e+00	2.185408e+00	-4.454078e-01
294976	2.030000e+00	2.185408e+00	-1.554078e-01
294977	1.490000e+00	2.185408e+00	-6.954078e-01
294978	4.670000e+00	2.199106e+00	2.470894e+00
294979	3.877795e+10	3.874444e+10	3.350965e+07
294980	1.310000e+00	2.185408e+00	-8.754078e-01
294981	1.850000e+00	2.185408e+00	-3.354078e-01

V. Conclusión

El algoritmo seleccionado para tomar parte en la predicción de la calidad del mineral de hierro es el árbol de decisión regresión, pues a diferencia del bosque, este fue menos costoso en términos de ejecución (tiempo).

VI. Anexo

En este apartado se presentan enlaces a documentos que complementan el proyecto realizado hasta el momento.

6.1. Liga a repositorio

- https://github.com/AbdielCJ/checkpoint2.git

6.2. Liga a presentación

- https://docs.google.com/presentation/d/1EjN9pm7tlosLCo847ilkv6hCdgr7ysuBzNnHM8Yr_1M/edit?usp=sharing

VII. Referencias bibliográficas

- Desconocido. (2019, marzo 11). *Análisis de datos para optimizar procesos en la minería*. minería chilena. Retrieved agosto 05, 2021, from https://www.mch.cl/2019/03/11/analisis-datos-optimizar-procesos-la-mineria/
- EduardoMagalhaesOliveira. (2017). *Quality Prediction in a Mining Process*. Kaggle. Retrieved junio 30, 2021, from https://www.kaggle.com/edumagalhaes/quality-prediction-in-a-mining-process
- González, L. (2018, junio 15). *DIFERENCIA ENTRE APRENDIZAJE SUPERVISADO Y NO SUPERVISADO*. YouTube MX. Retrieved agosto 05, 2021, from https://www.youtube.com/watch?v=7saDlsTCG5o&t=83s
- Katz, M. (2011). *Materiales y materias primas*. http://www.inet.edu.ar/wp-content/uploads/2012/11/minerales-de-hierro.pdf
- NORTE MINERO TV. (2019, junio 25). *8. La etapa de flotación y su equipo*. YouTube MX. Retrieved agosto 05, 2021, from https://www.youtube.com/watch?v=-nyxBINNC04
- PEDECIBA. (2014, octubre 29). *¿Qué es el mineral de hierro?* YouTube MX. Retrieved agosto 05, 2021, from https://www.youtube.com/watch?v=SP-g5HeJ5so
- Desconocido. (2014, enero 12). *Hidrofílicos, hidrofóbicos y diferencia de densidades: mezclemos agua y aceite*. Scienceaholic. Retrieved agosto 05, 2021, from http://adictoaciencia.blogspot.com/2014/01/hidrofilicos-hidrofobicos-y-diferencia.html