Apéndice a

Análisis de datos exploratorio

23 de agosto de 2024

Santiago Mora Cruz

Gabriel Reynoso Escamilla

Paulina Martinez Lopez

Guillermo Villegas Morales

In [ ]:
```python
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```python
db = pd.read_parquet('24o_medicom.parquet', engine='pyarrow')
db['ventas'] = db['ventas'].astype('float')
db.head()
```

Out[ ]:

	fecha	id_material	id_cliente	ventas
0	2013-05-06	768	7939	384.0
1	2011-09-20	768	7939	384.0
2	2014-01-08	768	7939	384.0
3	2011-04-19	768	7939	384.0
4	2013-03-21	768	7805	384.0

In [ ]:
```python
# db.to_csv('df.csv', index = False)
```

In [ ]:
```python
db.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 830517 entries, 0 to 830516
Data columns (total 4 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   fecha        830517 non-null  object 
 1   id_material  830517 non-null  int64  
 2   id_cliente   830517 non-null  int64  
 3   ventas       830517 non-null  float64
dtypes: float64(1), int64(2), object(1)
memory usage: 25.3+ MB
```

La base de datos se observa completa, sin valores nulos

In [ ]:
```python
db['ventas'].describe()
```

Out[ ]:

	ventas
count	8.305170e+05
mean	2.852943e+03
std	1.359341e+04
min	-1.149758e+06
25%	4.231500e+02
50%	9.525000e+02
75%	2.316800e+03
max	2.022000e+06

dtype: float64

Podemos observar que contamos con valores de ventas negativos que representan pérdidas. Además, vemos que el mínimo y el máximo distan demasiado de los valores donde caen los cuartiles.

In [ ]:
```python
print('clientes: ', len(db['id_cliente'].unique()))
print('materiales: ', len(db['id_material'].unique()))
```

```
clientes:  454
materiales:  1471
```

In [ ]:

In [ ]:
```python
# Clientes que más compran
clientes_mas_compran = db.groupby('id_cliente')['ventas'].sum().sort_values(ascending=False)
print("Clientes que más compran:")
print(clientes_mas_compran.head())

# Clientes que menos compran
clientes_menos_compran = db.groupby('id_cliente')['ventas'].sum().sort_values(ascending=True)
print("\nClientes que menos compran:")
print(clientes_menos_compran.head())
```

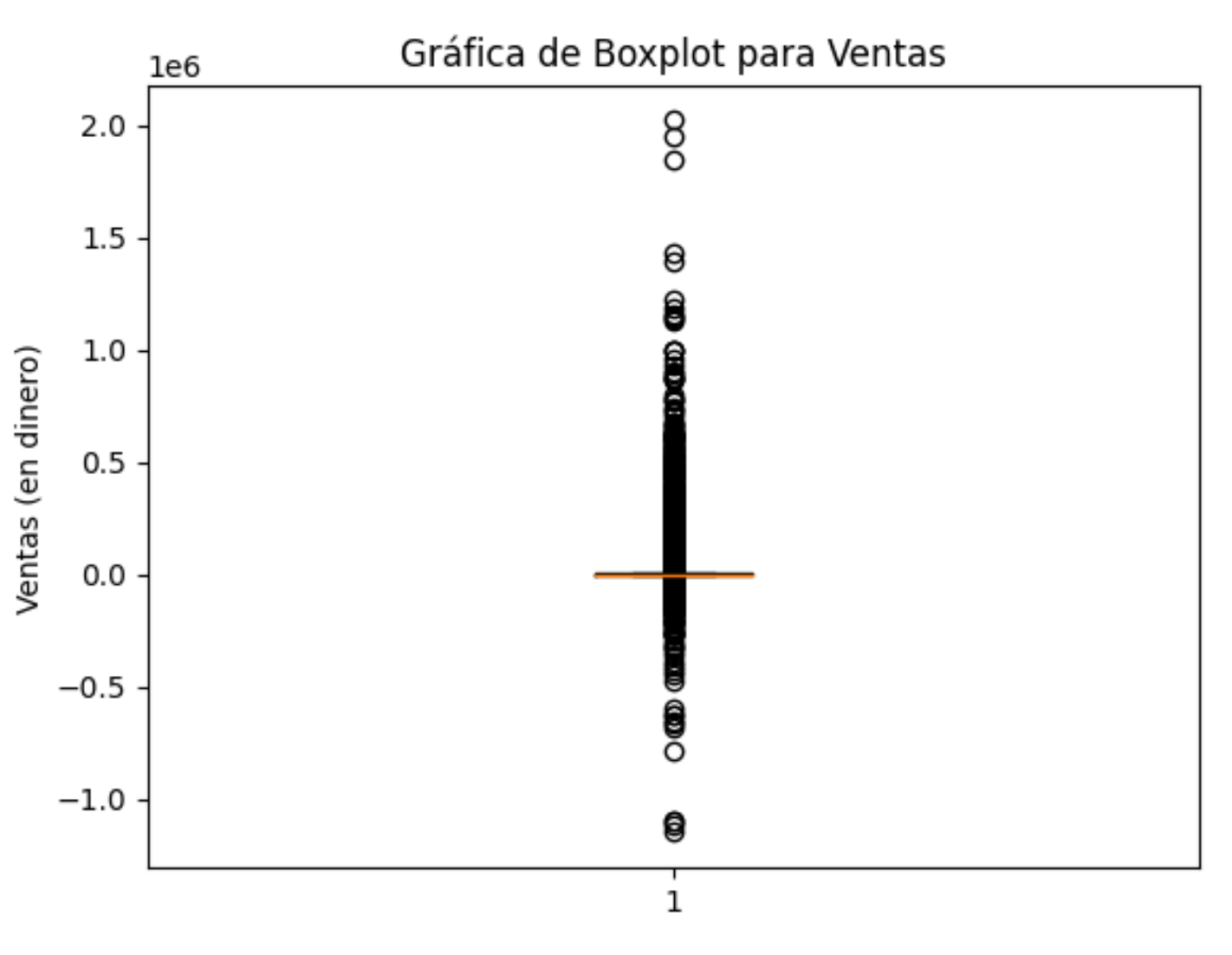
```
Clientes que más compran:
id_cliente
8342    3.419803e+08
8635    1.095702e+08
8318    7.813064e+07
7713    7.060284e+07
7806    5.166043e+07
Name: ventas, dtype: float64

Clientes que menos compran:
id_cliente
8182   -48987.90
308    -38823.60
8233   -10610.00
8202    -6869.85
7698    -1033.80
Name: ventas, dtype: float64
```

In [ ]:
```python
# Materiales que más se venden
materiales_mas_venden = db.groupby('id_material')['ventas'].sum().sort_values(ascending=False)
print("\nMateriales que más se venden:")
print(materiales_mas_venden.head())

# Materiales que menos se venden
materiales_menos_venden = db.groupby('id_material')['ventas'].sum().sort_values(ascending=True)
print("\nMateriales que menos se venden:")
print(materiales_menos_venden.head())
```

```
Materiales que más se venden:
id_material
591    1.343663e+08
893    1.236979e+08
601    1.207977e+08
590    1.065122e+08
772    9.313992e+07
Name: ventas, dtype: float64

Materiales que menos se venden:
id_material
2643   -2485.12
1585    -722.50
6929    -310.60
3488    -115.60
1586     -62.50
Name: ventas, dtype: float64
```

In [ ]:
```python
plt.boxplot(db['ventas']);
plt.title("Gráfica de Boxplot para Ventas")
plt.ylabel('Ventas (en dinero)')
```

Out[ ]:
```
Text(0, 0.5, 'Ventas (en dinero)')
```



Observamos que existen muchos outliers

Convertimos `fecha` a variable tipo `datetime`

In [ ]:
```python
db['fecha'] = pd.to_datetime(db['fecha'])
```

In [ ]:
```python
years = list(range(2011, 2025, 1))
count4year = []
for item in years:
    count4year.append(db[db['fecha'].dt.year == item]['fecha'].count())
```

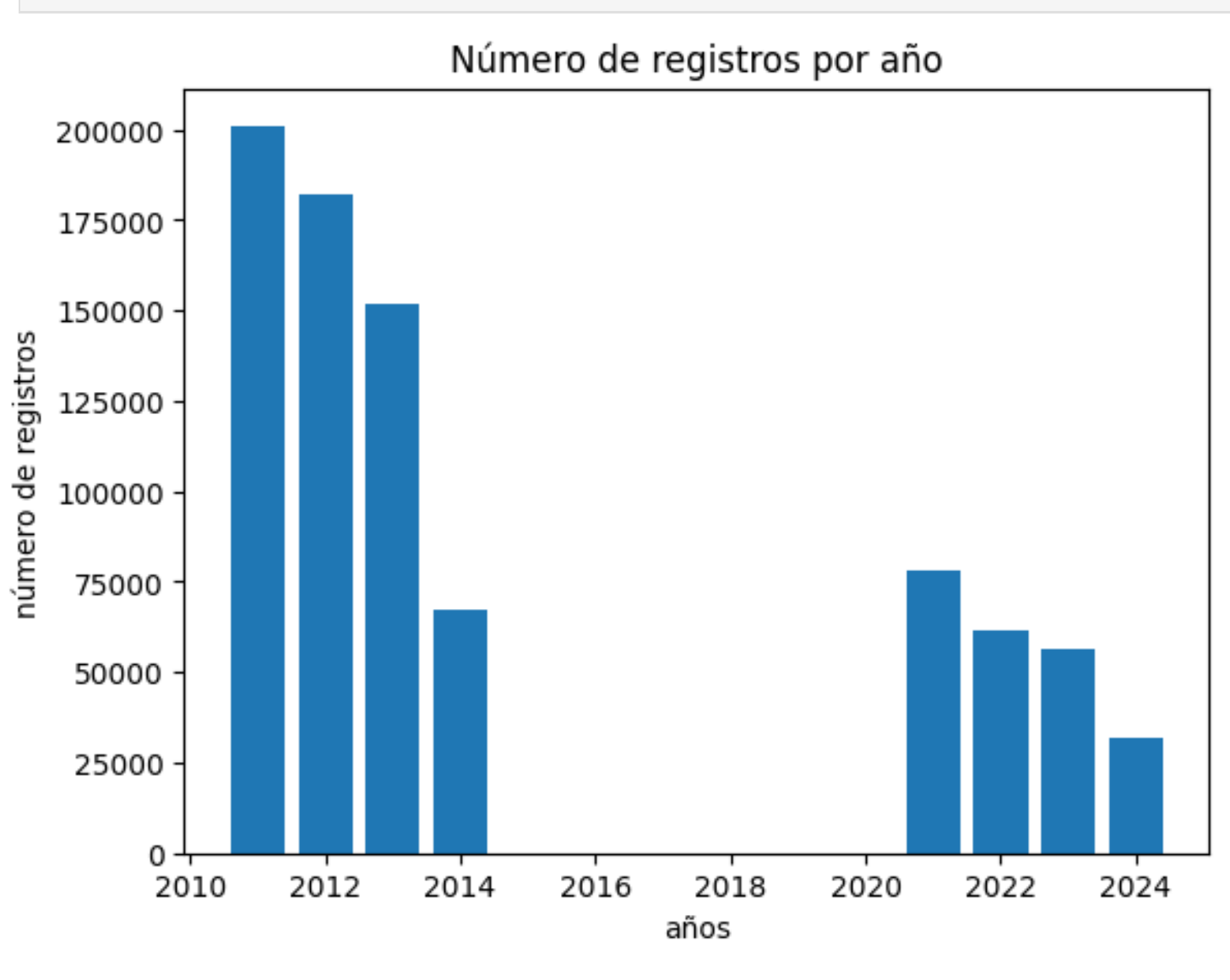
In [ ]:
```python
import numpy as np
np.sum(count4year)
```

Out[ ]:
```
830517
```

In [ ]:
```python
count4year
```

Out[ ]:
```
[201111, 182082, 152053, 67476, 0, 0, 0, 0, 0, 0, 78186, 61282, 56684, 31643]
```

In [ ]:
```python
plt.figure()
plt.bar(years, count4year)
plt.title('Número de registros por año')
plt.xlabel('años')
plt.ylabel('número de registros')
plt.savefig('Barplot_registros.jpeg')
```



Aquí podemos observar que no existen regristros de los años 2015 a 2020, además que no es un error de formato ya que la suma de todos los registros que obtuvimos de esta manera es igual al tamaño de la base de datos original

In [ ]:
```python
db_p1 = db[db['fecha'].dt.year <= 2014]
db_p2 = db[db['fecha'].dt.year >= 2021]
```

In [ ]:
```python
len(db_p1['id_cliente'].unique())
```

Out[ ]:
```
417
```

In [ ]:
```python
len(db_p2['id_cliente'].unique())
```

Out[ ]:
```
158
```

Con este análisis anterior, se encontró que de 2014 para atrás, se tenían 417 clientes, mientras que de 2021 en adelante, solo están 158. Por esto mismo, despreciaremos los registros que se situen de 2011-2014 ya que la mayoría de los clientes no continuan después de 2021. Además esto hará un mejor modelo ya que la brecha en los tiempos y la inconsistencia en los clientes sesgaría el proceso

In [ ]:
```python
# Clientes que más compran
clientes_mas_compran = db_p2.groupby('id_cliente')['ventas'].sum().sort_values(ascending=False)
print("Clientes que más compran:")
print(clientes_mas_compran.head())

# Clientes que menos compran
clientes_menos_compran = db_p2.groupby('id_cliente')['ventas'].sum().sort_values(ascending=True)
print("\nClientes que menos compran:")
print(clientes_menos_compran.head())
```

```
Clientes que más compran:
id_cliente
8342    3.022364e+08
8635    8.345564e+07
7713    6.060637e+07
9066    4.142447e+07
7806    3.916261e+07
Name: ventas, dtype: float64

Clientes que menos compran:
id_cliente
9213     3502.77
8107     4600.20
3452     8184.24
3451    21436.34
8363    23151.80
Name: ventas, dtype: float64
```

In [ ]:
```python
# Materiales que más se venden
materiales_mas_venden = db_p2.groupby('id_material')['ventas'].sum().sort_values(ascending=False)
print("\nMateriales que más se venden:")
print(materiales_mas_venden.head())

# Materiales que menos se venden
materiales_menos_venden = db_p2.groupby('id_material')['ventas'].sum().sort_values(ascending=True)
print("\nMateriales que menos se venden:")
print(materiales_menos_venden.head())
```

```
Materiales que más se venden:
id_material
591    97903712.02
601    84710470.41
893    84654422.81
590    78186308.77
772    77445106.93
Name: ventas, dtype: float64

Materiales que menos se venden:
id_material
6929   -310.6
3488   -115.6
7207      0.0
6056      0.0
2934      0.0
Name: ventas, dtype: float64
```

Obtenemos nuevamente los clientes que más y menos han comprado en cuanto a ventas así como los materiales que más y menos se han vendido, pero solo para la segunda mitad de los datos (2021-2024). De este punto en adelante solo trabajaremos con los datos mencionados

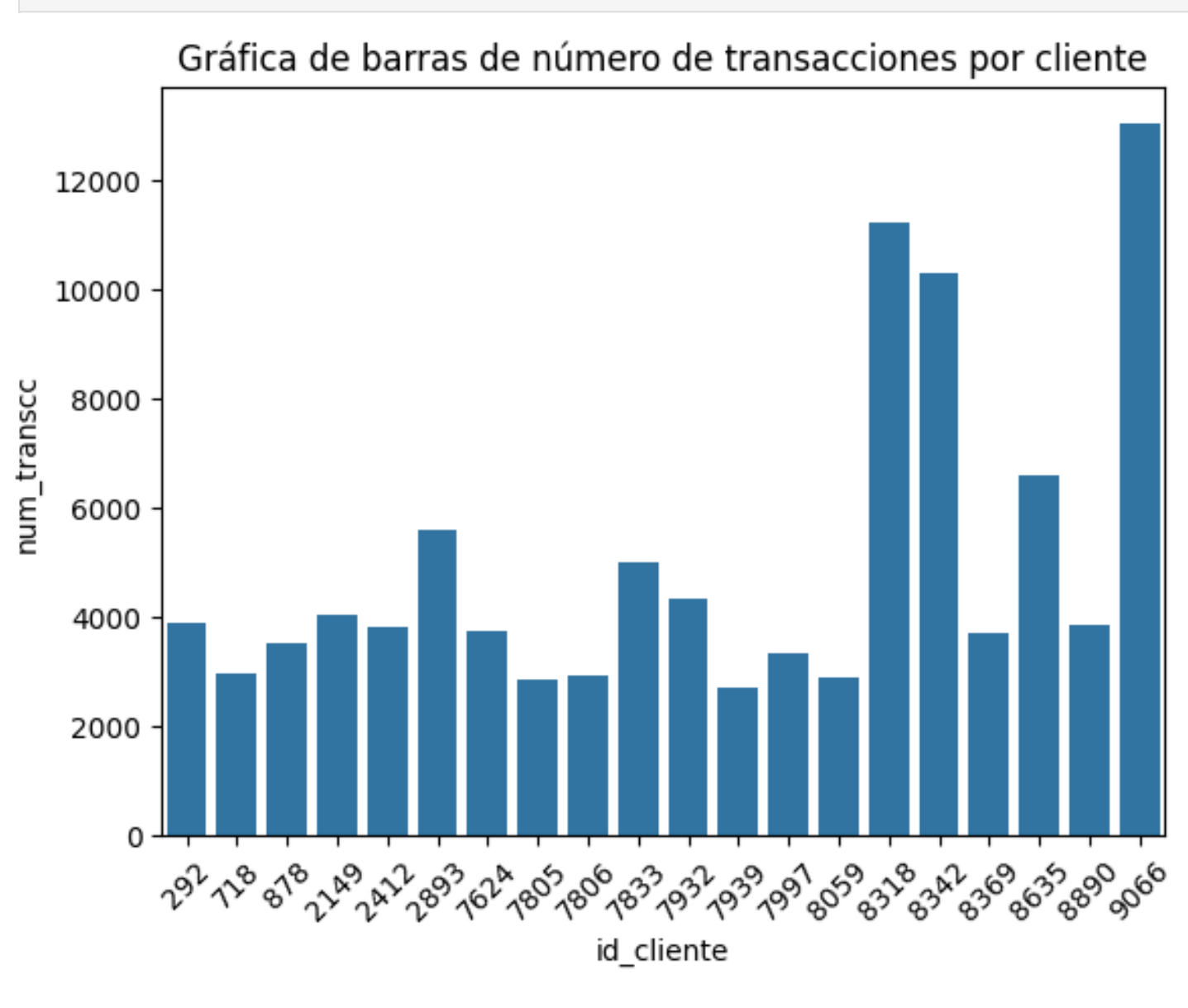
Buscamos los clientes que cuentan con más regsitros y los graficamos en un barplot

In [ ]:
```python
clientes_mas_transacciones = db_p2.groupby('id_cliente').size().sort_values(ascending=False).head(20)
dfct = pd.DataFrame({'id_cliente':clientes_mas_transacciones.index, 'num_transcc':clientes_mas_transacciones.values})
dfct.head()
```

Out[ ]:

	id_cliente	num_transcc
0	9066	13051
1	8318	11240
2	8342	10284
3	8635	6601
4	2893	5576

In [ ]:
```python
import seaborn as sns
sns.barplot(data = dfct, x = 'id_cliente', y = 'num_transcc')
plt.xticks(rotation = 45)
plt.title('Gráfica de barras de número de transacciones por cliente')
plt.show();
```



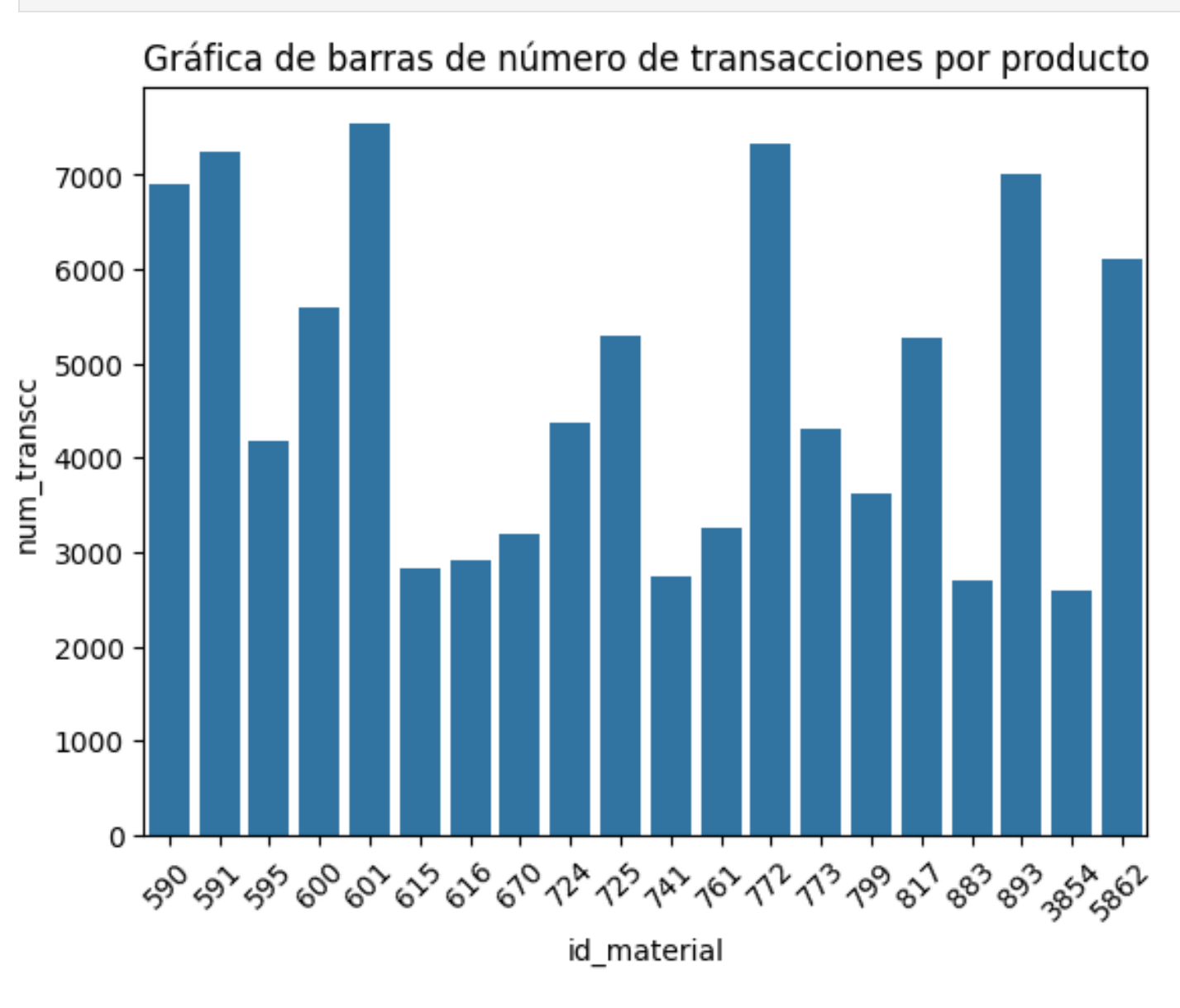
Hacemos el mismo proceso pero ahora por producto

In [ ]:
```python
materiales_mas_transacciones = db_p2.groupby('id_material').size().sort_values(ascending=False).head(20)
dfmt = pd.DataFrame({'id_material':materiales_mas_transacciones.index, 'num_transcc':materiales_mas_transacciones.values})
dfmt.head()
```

Out[ ]:

	id_material	num_transcc
0	601	7551
1	772	7341
2	591	7256
3	893	7007
4	590	6902

In [ ]:
```python
sns.barplot(data = dfmt, x = 'id_material', y = 'num_transcc')
plt.xticks(rotation = 45)
plt.title('Gráfica de barras de número de transacciones por producto')
plt.show();
```



Buscamos a qué se deben los valores negativos y qué clientes o productos los generan

In [ ]:
```python
db_minus = db_p2[db_p2['ventas'] <=0].reset_index()
db_minus
```

Out[ ]:

	index	fecha	id_material	id_cliente	ventas
0	10	2023-01-24	768	9066	-484.92
1	103	2024-05-13	768	9066	-183.38
2	108	2023-01-20	768	9066	-161.64
3	195	2023-06-22	768	9066	-519.00
4	196	2023-06-02	768	9066	-519.00
...	...	...	...	...	...
6546	830188	2023-07-31	767	7805	-1170.00
6547	830273	2023-07-06	767	8635	-2925.00
6548	830278	2023-07-14	767	7805	-1462.50
6549	830279	2023-07-17	767	7805	-1462.50
6550	830481	2021-01-07	3071	7842	-5265.46

6551 rows × 5 columns

existen 6,551 registros con ventas negativas

In [ ]:
```python
db_minus.shape[0]/db_p2.shape[0]
```

Out[ ]:
```
0.028758313395816415
```

Aquí podemos ver que los valores negativos solo corresponden a poco menos de 3% de la información, por lo que rechazarla es opción. Primero buscaremos los materiales y clientes que generan estos valores

In [ ]:
```python
cliente_minus = db_minus['id_cliente'].value_counts()
```

In [ ]:
```python
minus_total = []
for i in cliente_minus.index:
  minus_total.append(db_p2[db_p2['id_cliente'] == i].shape[0])
db_clientes_minus = pd.DataFrame({'cliente': cliente_minus.index,
                                  'count_minus': cliente_minus,
                                  'count_total':minus_total})
db_clientes_minus['difference'] = round(db_clientes_minus['count_minus']/db_clientes_minus['count_total'],3)
db_clientes_minus
```

Out[ ]:

	cliente	count_minus	count_total	difference
id_cliente				
8342	8342	681	10284	0.066
8635	8635	540	6601	0.082
9066	9066	287	13051	0.022
7806	7806	251	2927	0.086
7648	7648	210	2199	0.095
...	...	...	...	...
8682	8682	1	600	0.002
1103	1103	1	50	0.020
8209	8209	1	378	0.003
8099	8099	1	1125	0.001
8598	8598	1	43	0.023

136 rows × 4 columns

In [ ]:
```python
sns.histplot(db_clientes_minus['difference'])
plt.title('Gráfica de número de clientes con porcentaje de valores negativos')
```

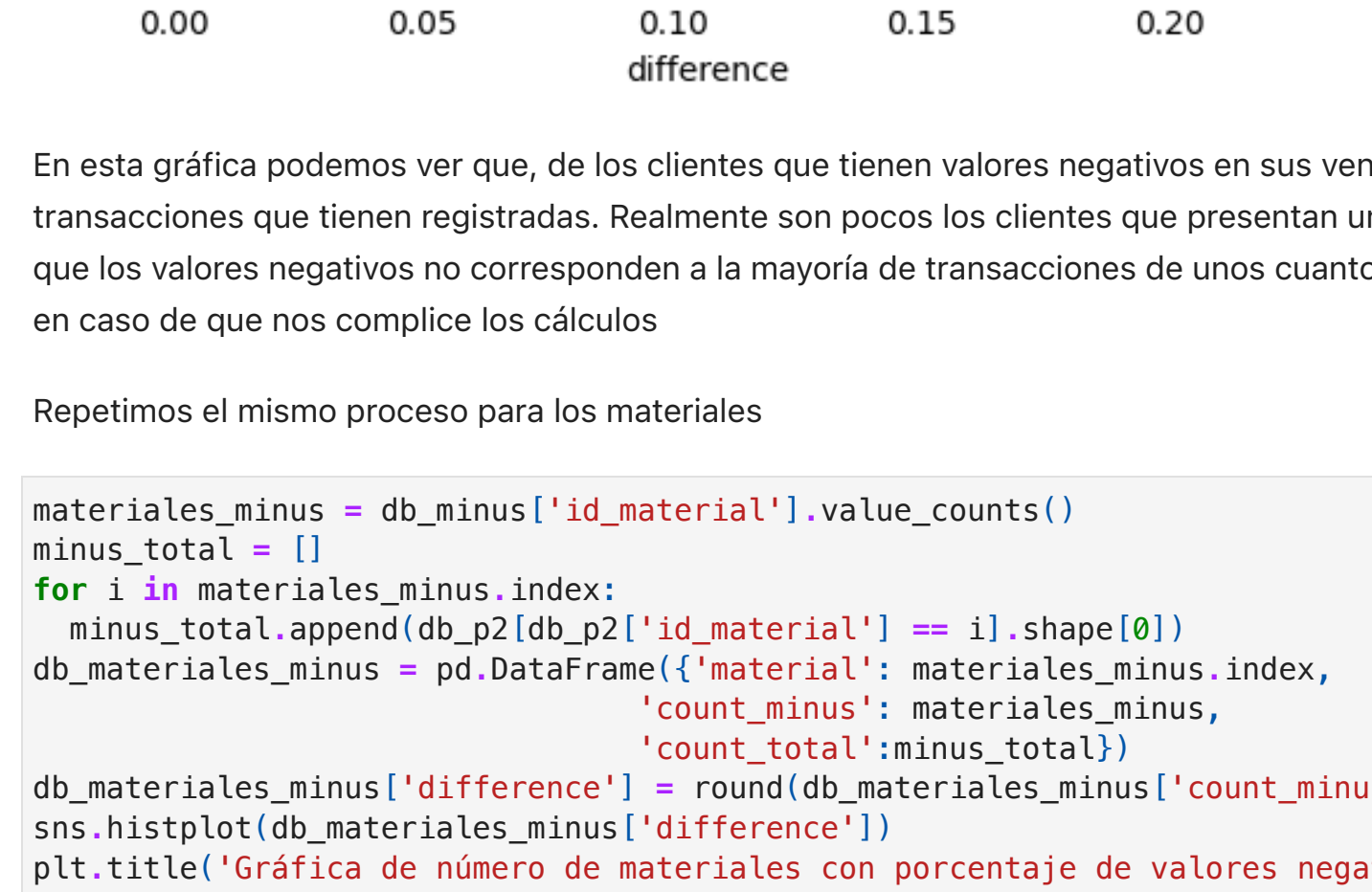
Out[ ]:
```
Text(0.5, 1.0, 'Gráfica de número de clientes con porcentaje de valores negativos')
```

En esta gráfica podemos ver que, de los clientes que tienen valores negativos en sus ventas, la mayoría tiene una proporción muy chica de valores negativos con respecto al total de transacciones que tienen registradas. Realmente son pocos los clientes que presentan una proporción mayor a una décima parte y no superan el 25% de los registros. Ahora sabemos que los valores negativos no corresponden a la mayoría de transacciones de unos cuantos clientes, y que tampoco son muchas. Por lo tanto podemos descpreciar esta información en caso de que nos complice los cálculos

Repetimos el mismo proceso para los materiales

In [ ]:
```python
materiales_minus = db_minus['id_material'].value_counts()
minus_total = []
for i in materiales_minus.index:
  minus_total.append(db_p2[db_p2['id_material'] == i].shape[0])
db_materiales_minus = pd.DataFrame({'material': materiales_minus.index,
                                    'count_minus': materiales_minus,
                                    'count_total':minus_total})
db_materiales_minus['difference'] = round(db_materiales_minus['count_minus']/db_materiales_minus['count_total'],3)
sns.histplot(db_materiales_minus['difference'])
plt.title('Gráfica de número de materiales con porcentaje de valores negativos')
```

Out[ ]:
```
Text(0.5, 1.0, 'Gráfica de número de materiales con porcentaje de valores negativos')
```



Observamos un patrón muy similar en la disitrbución de las proporciones de los valores negativos cuando los contamos por mateiral, aunque son más los materiales que suelen presentar pérdidas. Expecionalmente aparecen valores que siempre a representado pérdidas

In [ ]:
```python
db_materiales_minus[db_materiales_minus['difference'] == 1]
```

Out[ ]:

	material	count_minus	count_total	difference
id_material				
6929	6929	1	1	1.0
3488	3488	1	1	1.0

Como vemos que son materiales que se vendieron una única vez y presentaron pérdidas, podemos despreciarlos.