PROYECTO 2 - Autor: Smit Jonatan Villafranca Romero

March 10, 2022

1 PROYECTO 2

1.0.1 INDICACIONES GENERALES:

Link de la data : https://www.dropbox.com/s/tr8jf1dqlv4hsl8/TRAIN_FUGA.csv?dl=0

Librerias principales:

[1]:
```
import os             #Para direcionar la ruta de trabajo
import pandas as pd   #Para archivos csv, excel, spss, stata
import numpy as np    #Para trabajar con matrices/arrays
import matplotlib.pyplot as plt
import seaborn as sns
import math as math
import warnings
from IPython.display import Image
%matplotlib inline
```

1.1 CASO 1:

Teniendo en cuenta la base de datos TRAIN_FUGA.csv que corresponde a los datos de entrenamiento de una data que analiza la fuga de clientes de una entidad bancaria realizar las siguientes tareas:

1. Análisis exploratorio de las variables: medidas y visualización. (5 puntos)
2. Realizar un análisis exploratorio sobre presencia de outliers. (2 puntos)
3. Realizar una discretización de las variables : INGRESO_BRUTO_M1 y EDAD teniendo en cuenta al menos dos técnicas de discretización no supervisada y agregar las variables discretizadas a nuestro conjunto de datos original (2 puntos)
4. Aplicar dos técnicas de balanceo de datos a nuestra variable TARGET (objetivo) y agregarlas a nuestro conjunto de datos original. Use los parámetros vistos en clase. (3 puntos)

Nota: revisar el diccionario de variables del caso.

1.2 1. Análisis exploratorio de las variables: medidas y visualización.

1.2.1 1.1. EXPLORACIÓN

1.2.2 1.1.1. ANALIZANDO LOS DATOS

[2]:
```
os.chdir("E:\PYTHOM\MODULO 1\EXAMEN FINAL-MODULO1")#direccionando la ruta
archivo_csv="TRAIN_FUGA.csv"
df=pd.read_csv(archivo_csv,sep=",", encoding="ISO-8859-1")
df.head(50)
```

[2]:

	Unnamed: 0	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO
0	1	201411	0	46	F	PIURA
1	2	201411	0	54	M	LORETO
2	3	201411	0	81	M	NaN
3	4	201411	0	42	M	PIURA
4	5	201411	0	52	M	MOQUEGUA
5	6	201411	0	74	M	LA LIBERTAD
6	7	201411	0	66	M	LA LIBERTAD
7	8	201411	0	57	M	LIMA
8	9	201411	0	65	M	CALLAO
9	10	201411	0	63	M	ANCASH
10	11	201411	0	43	M	LIMA
11	12	201411	0	64	F	LIMA
12	13	201411	0	114	M	NaN
13	14	201411	0	48	F	LIMA
14	15	201411	0	48	F	MOQUEGUA
15	16	201411	0	114	F	NaN
16	17	201411	0	114	M	NaN
17	18	201411	0	42	F	LIMA
18	19	201411	0	93	M	LIMA
19	20	201411	0	114	F	NaN
20	21	201411	0	114	F	NaN
21	22	201411	0	39	M	LIMA
22	23	201411	0	45	M	LIMA
23	24	201411	0	50	M	LIMA
24	25	201411	0	114	F	NaN
25	26	201411	0	114	M	NaN
26	27	201411	0	114	F	NaN
27	28	201411	0	114	M	NaN
28	29	201411	0	114	M	NaN
29	30	201411	0	114	M	NaN
30	31	201411	0	114	M	NaN
31	32	201411	0	114	M	NaN
32	33	201411	0	114	F	NaN
33	34	201411	0	52	F	AREQUIPA
34	35	201411	0	57	M	LIMA
35	36	201411	0	114	F	NaN

36	37	201411	0	48	M	LIMA
37	38	201411	0	57	F	LIMA
38	39	201411	0	52	M	CALLAO
39	40	201411	0	61	F	ANCASH
40	41	201411	0	48	M	LIMA
41	42	201411	0	46	M	CALLAO
42	43	201411	0	45	F	LIMA
43	44	201411	0	54	F	LIMA
44	45	201411	0	44	M	LIMA
45	46	201411	0	42	M	LIMA
46	47	201411	0	44	F	LIMA
47	48	201411	0	53	M	NaN
48	49	201411	0	47	F	LIMA
49	50	201411	0	37	M	LIMA

	INGRESO_BRUTO_M1	FLG_CLIENTE	SEGMENTO	FLG_ADEL_SUELDO_M1	FREC_AGENTE	\
0	NaN	NO CLIENTE	2	0	0	
1	4718.0	CLIENTE	1BC	0	0	
2	NaN	CLIENTE	6	0	0	
3	936.0	CLIENTE	2	0	0	
4	5844.0	CLIENTE	1BC	0	0	
5	NaN	CLIENTE	6	0	0	
6	4232.0	CLIENTE	1BC	0	0	
7	1580.0	NO CLIENTE	2	0	0	
8	NaN	NO CLIENTE	2	0	0	
9	936.0	CLIENTE	2	0	0	
10	1421.0	CLIENTE	2	0	0	
11	NaN	NO CLIENTE	2	0	0	
12	NaN	CLIENTE	6	0	0	
13	809.0	CLIENTE	2	0	0	
14	739.0	CLIENTE	3	0	0	
15	NaN	CLIENTE	6	0	0	
16	NaN	CLIENTE	6	0	0	
17	739.0	CLIENTE	3	0	0	
18	NaN	NO CLIENTE	6	0	0	
19	NaN	CLIENTE	6	0	0	
20	NaN	CLIENTE	6	0	0	
21	749.0	NO CLIENTE	3	0	0	
22	936.0	CLIENTE	3	0	0	
23	936.0	CLIENTE	3	0	0	
24	NaN	CLIENTE	6	0	0	
25	NaN	CLIENTE	6	0	0	
26	NaN	CLIENTE	6	0	0	
27	NaN	CLIENTE	6	0	0	
28	NaN	CLIENTE	6	0	0	
29	NaN	CLIENTE	6	0	0	
30	NaN	CLIENTE	6	0	0	

31	NaN	CLIENTE	6	0	0	
32	NaN	CLIENTE	6	0	0	
33	739.0	CLIENTE	2	0	0	
34	858.0	CLIENTE	3	0	0	
35	NaN	CLIENTE	6	0	0	
36	NaN	NO CLIENTE	2	0	0	
37	3962.0	NO CLIENTE	1BC	0	0	
38	NaN	NO CLIENTE	3	0	0	
39	739.0	CLIENTE	5	0	0	
40	858.0	CLIENTE	3	0	0	
41	NaN	NO CLIENTE	6	0	0	
42	NaN	NO CLIENTE	2	0	0	
43	NaN	NO CLIENTE	6	0	0	
44	818.0	CLIENTE	3	0	0	
45	2777.0	CLIENTE	3	0	0	
46	NaN	NO CLIENTE	6	0	0	
47	NaN	CLIENTE	6	0	0	
48	1093.0	CLIENTE	3	0	0	
49	NaN	NO CLIENTE	6	0	0	

	FLG_VEH_SF	FLG_CONV_SF	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0.0	0.0	0	0	0	
1	0.0	1.0	0	0	0	
2	NaN	NaN	0	0	0	
3	NaN	NaN	0	0	0	
4	NaN	NaN	0	0	0	
5	NaN	NaN	0	0	0	
6	0.0	0.0	0	0	6	
7	NaN	NaN	5	6	6	
8	0.0	0.0	0	0	0	
9	NaN	NaN	0	0	0	
10	0.0	0.0	0	0	0	
11	NaN	NaN	0	0	0	
12	NaN	NaN	0	0	0	
13	NaN	NaN	0	0	0	
14	NaN	NaN	0	0	0	
15	NaN	NaN	0	0	0	
16	NaN	NaN	0	0	0	
17	NaN	NaN	0	0	0	
18	NaN	NaN	0	0	0	
19	NaN	NaN	0	0	0	
20	NaN	NaN	0	0	0	
21	NaN	NaN	0	0	0	
22	NaN	NaN	0	0	0	
23	NaN	NaN	0	0	0	
24	NaN	NaN	0	0	0	
25	NaN	NaN	0	0	0	

26	NaN	NaN	0	0	0
27	NaN	NaN	0	0	0
28	NaN	NaN	0	0	0
29	NaN	NaN	0	0	0
30	NaN	NaN	0	0	0
31	NaN	NaN	0	0	0
32	NaN	NaN	0	0	0
33	NaN	NaN	0	0	0
34	NaN	NaN	0	0	0
35	NaN	NaN	0	0	0
36	NaN	NaN	0	0	0
37	0.0	0.0	2	0	0
38	0.0	0.0	0	0	0
39	NaN	NaN	0	0	0
40	NaN	NaN	0	0	0
41	0.0	0.0	0	0	0
42	0.0	0.0	0	0	0
43	NaN	NaN	0	0	0
44	0.0	0.0	0	0	0
45	0.0	0.0	3	0	0
46	NaN	NaN	0	0	0
47	NaN	NaN	0	0	0
48	NaN	NaN	0	0	0
49	0.0	0.0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	REC_AGENTE_TD	CTD_RECLAMOS_M1
0	0.000000	224.0	NaN	0
1	0.000000	123.0	NaN	0
2	0.000000	264.0	NaN	0
3	0.000000	263.0	NaN	0
4	0.000000	263.0	NaN	0
5	0.000000	256.0	NaN	0
6	0.000000	85.0	NaN	0
7	2.166667	151.0	NaN	0
8	3.333333	778.0	NaN	0
9	0.000000	272.0	NaN	0
10	0.000000	11.0	NaN	0
11	0.000000	21.0	NaN	0
12	0.000000	281.0	NaN	0
13	0.000000	209.0	NaN	0
14	0.000000	208.0	NaN	0
15	0.000000	233.0	NaN	0
16	0.000000	216.0	NaN	0
17	0.000000	233.0	NaN	0
18	0.000000	281.0	NaN	0
19	0.000000	263.0	NaN	0
20	0.000000	221.0	NaN	0

21	0.000000	163.0	NaN	0
22	0.000000	215.0	NaN	0
23	0.000000	257.0	NaN	0
24	0.000000	206.0	NaN	0
25	0.000000	281.0	NaN	0
26	0.000000	280.0	NaN	0
27	0.000000	220.0	NaN	0
28	0.000000	275.0	NaN	0
29	0.000000	263.0	NaN	0
30	0.000000	280.0	NaN	0
31	0.000000	263.0	NaN	0
32	0.000000	203.0	NaN	0
33	0.000000	202.0	NaN	0
34	0.000000	264.0	NaN	0
35	0.000000	263.0	NaN	0
36	0.000000	265.0	NaN	0
37	0.000000	778.0	NaN	0
38	0.000000	85.0	NaN	0
39	0.000000	257.0	NaN	0
40	0.000000	191.0	NaN	0
41	0.000000	162.0	NaN	0
42	0.000000	185.0	NaN	0
43	0.000000	159.0	NaN	0
44	0.000000	280.0	NaN	0
45	0.333333	76.0	NaN	0
46	0.000000	778.0	NaN	0
47	0.000000	280.0	NaN	0
48	0.000000	164.0	NaN	0
49	0.000000	209.0	NaN	0

Elimando la columna Unnamed: 0 debido a que esa columna no pertenece al DataFrame

[3]:
```
data_resultante=df.drop(["Unnamed: 0"], axis=1)
data_resultante.head(50)
```

[3]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	INGRESO_BRUTO_M1	\
0	201411	0	46	F	PIURA	NaN	
1	201411	0	54	M	LORETO	4718.0	
2	201411	0	81	M	NaN	NaN	
3	201411	0	42	M	PIURA	936.0	
4	201411	0	52	M	MOQUEGUA	5844.0	
5	201411	0	74	M	LA LIBERTAD	NaN	
6	201411	0	66	M	LA LIBERTAD	4232.0	
7	201411	0	57	M	LIMA	1580.0	
8	201411	0	65	M	CALLAO	NaN	
9	201411	0	63	M	ANCASH	936.0	
10	201411	0	43	M	LIMA	1421.0	

11	201411	0	64	F	LIMA	NaN
12	201411	0	114	M	NaN	NaN
13	201411	0	48	F	LIMA	809.0
14	201411	0	48	F	MOQUEGUA	739.0
15	201411	0	114	F	NaN	NaN
16	201411	0	114	M	NaN	NaN
17	201411	0	42	F	LIMA	739.0
18	201411	0	93	M	LIMA	NaN
19	201411	0	114	F	NaN	NaN
20	201411	0	114	F	NaN	NaN
21	201411	0	39	M	LIMA	749.0
22	201411	0	45	M	LIMA	936.0
23	201411	0	50	M	LIMA	936.0
24	201411	0	114	F	NaN	NaN
25	201411	0	114	M	NaN	NaN
26	201411	0	114	F	NaN	NaN
27	201411	0	114	M	NaN	NaN
28	201411	0	114	M	NaN	NaN
29	201411	0	114	M	NaN	NaN
30	201411	0	114	M	NaN	NaN
31	201411	0	114	M	NaN	NaN
32	201411	0	114	F	NaN	NaN
33	201411	0	52	F	AREQUIPA	739.0
34	201411	0	57	M	LIMA	858.0
35	201411	0	114	F	NaN	NaN
36	201411	0	48	M	LIMA	NaN
37	201411	0	57	F	LIMA	3962.0
38	201411	0	52	M	CALLAO	NaN
39	201411	0	61	F	ANCASH	739.0
40	201411	0	48	M	LIMA	858.0
41	201411	0	46	M	CALLAO	NaN
42	201411	0	45	F	LIMA	NaN
43	201411	0	54	F	LIMA	NaN
44	201411	0	44	M	LIMA	818.0
45	201411	0	42	M	LIMA	2777.0
46	201411	0	44	F	LIMA	NaN
47	201411	0	53	M	NaN	NaN
48	201411	0	47	F	LIMA	1093.0
49	201411	0	37	M	LIMA	NaN

	FLG_CLIENTE	SEGMENTO	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FLG_VEH_SF	\
0	NO CLIENTE	2	0	0	0.0	
1	CLIENTE	1BC	0	0	0.0	
2	CLIENTE	6	0	0	NaN	
3	CLIENTE	2	0	0	NaN	
4	CLIENTE	1BC	0	0	NaN	
5	CLIENTE	6	0	0	NaN	

6		CLIENTE	1BC	0	0	0.0
7	NO	CLIENTE	2	0	0	NaN
8	NO	CLIENTE	2	0	0	0.0
9		CLIENTE	2	0	0	NaN
10		CLIENTE	2	0	0	0.0
11	NO	CLIENTE	2	0	0	NaN
12		CLIENTE	6	0	0	NaN
13		CLIENTE	2	0	0	NaN
14		CLIENTE	3	0	0	NaN
15		CLIENTE	6	0	0	NaN
16		CLIENTE	6	0	0	NaN
17		CLIENTE	3	0	0	NaN
18	NO	CLIENTE	6	0	0	NaN
19		CLIENTE	6	0	0	NaN
20		CLIENTE	6	0	0	NaN
21	NO	CLIENTE	3	0	0	NaN
22		CLIENTE	3	0	0	NaN
23		CLIENTE	3	0	0	NaN
24		CLIENTE	6	0	0	NaN
25		CLIENTE	6	0	0	NaN
26		CLIENTE	6	0	0	NaN
27		CLIENTE	6	0	0	NaN
28		CLIENTE	6	0	0	NaN
29		CLIENTE	6	0	0	NaN
30		CLIENTE	6	0	0	NaN
31		CLIENTE	6	0	0	NaN
32		CLIENTE	6	0	0	NaN
33		CLIENTE	2	0	0	NaN
34		CLIENTE	3	0	0	NaN
35		CLIENTE	6	0	0	NaN
36	NO	CLIENTE	2	0	0	NaN
37	NO	CLIENTE	1BC	0	0	0.0
38	NO	CLIENTE	3	0	0	0.0
39		CLIENTE	5	0	0	NaN
40		CLIENTE	3	0	0	NaN
41	NO	CLIENTE	6	0	0	0.0
42	NO	CLIENTE	2	0	0	0.0
43	NO	CLIENTE	6	0	0	NaN
44		CLIENTE	3	0	0	0.0
45		CLIENTE	3	0	0	0.0
46	NO	CLIENTE	6	0	0	NaN
47		CLIENTE	6	0	0	NaN
48		CLIENTE	3	0	0	NaN
49	NO	CLIENTE	6	0	0	0.0

	FLG_CONV_SF	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	PROM_CTD_TRX_6M	\
0	0.0	0	0	0	0.000000	

1	1.0	0	0	0	0.000000
2	NaN	0	0	0	0.000000
3	NaN	0	0	0	0.000000
4	NaN	0	0	0	0.000000
5	NaN	0	0	0	0.000000
6	0.0	0	0	6	0.000000
7	NaN	5	6	6	2.166667
8	0.0	0	0	0	3.333333
9	NaN	0	0	0	0.000000
10	0.0	0	0	0	0.000000
11	NaN	0	0	0	0.000000
12	NaN	0	0	0	0.000000
13	NaN	0	0	0	0.000000
14	NaN	0	0	0	0.000000
15	NaN	0	0	0	0.000000
16	NaN	0	0	0	0.000000
17	NaN	0	0	0	0.000000
18	NaN	0	0	0	0.000000
19	NaN	0	0	0	0.000000
20	NaN	0	0	0	0.000000
21	NaN	0	0	0	0.000000
22	NaN	0	0	0	0.000000
23	NaN	0	0	0	0.000000
24	NaN	0	0	0	0.000000
25	NaN	0	0	0	0.000000
26	NaN	0	0	0	0.000000
27	NaN	0	0	0	0.000000
28	NaN	0	0	0	0.000000
29	NaN	0	0	0	0.000000
30	NaN	0	0	0	0.000000
31	NaN	0	0	0	0.000000
32	NaN	0	0	0	0.000000
33	NaN	0	0	0	0.000000
34	NaN	0	0	0	0.000000
35	NaN	0	0	0	0.000000
36	NaN	0	0	0	0.000000
37	0.0	2	0	0	0.000000
38	0.0	0	0	0	0.000000
39	NaN	0	0	0	0.000000
40	NaN	0	0	0	0.000000
41	0.0	0	0	0	0.000000
42	0.0	0	0	0	0.000000
43	NaN	0	0	0	0.000000
44	0.0	0	0	0	0.000000
45	0.0	3	0	0	0.333333
46	NaN	0	0	0	0.000000
47	NaN	0	0	0	0.000000

48	NaN	0	0	0	0.000000
49	0.0	0	0	0	0.000000

	ANT_CLIENTE	REC_AGENTE_TD	CTD_RECLAMOS_M1
0	224.0	NaN	0
1	123.0	NaN	0
2	264.0	NaN	0
3	263.0	NaN	0
4	263.0	NaN	0
5	256.0	NaN	0
6	85.0	NaN	0
7	151.0	NaN	0
8	778.0	NaN	0
9	272.0	NaN	0
10	11.0	NaN	0
11	21.0	NaN	0
12	281.0	NaN	0
13	209.0	NaN	0
14	208.0	NaN	0
15	233.0	NaN	0
16	216.0	NaN	0
17	233.0	NaN	0
18	281.0	NaN	0
19	263.0	NaN	0
20	221.0	NaN	0
21	163.0	NaN	0
22	215.0	NaN	0
23	257.0	NaN	0
24	206.0	NaN	0
25	281.0	NaN	0
26	280.0	NaN	0
27	220.0	NaN	0
28	275.0	NaN	0
29	263.0	NaN	0
30	280.0	NaN	0
31	263.0	NaN	0
32	203.0	NaN	0
33	202.0	NaN	0
34	264.0	NaN	0
35	263.0	NaN	0
36	265.0	NaN	0
37	778.0	NaN	0
38	85.0	NaN	0
39	257.0	NaN	0
40	191.0	NaN	0
41	162.0	NaN	0
42	185.0	NaN	0

```
 43        159.0            NaN               0
 44        280.0            NaN               0
 45         76.0            NaN               0
 46        778.0            NaN               0
 47        280.0            NaN               0
 48        164.0            NaN               0
 49        209.0            NaN               0
```

Información del DataFrame

[4]:
```
data_resultante.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 19 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   CODMES              787495 non-null  int64
 1   TARGET_MODEL2       787495 non-null  int64
 2   EDAD                787495 non-null  int64
 3   SEXO                787495 non-null  object
 4   DEPARTAMENTO        760520 non-null  object
 5   INGRESO_BRUTO_M1    600241 non-null  float64
 6   FLG_CLIENTE         787495 non-null  object
 7   SEGMENTO            787495 non-null  object
 8   FLG_ADEL_SUELDO_M1  787495 non-null  int64
 9   FREC_AGENTE         787495 non-null  int64
 10  FLG_VEH_SF          518112 non-null  float64
 11  FLG_CONV_SF         518112 non-null  float64
 12  FREC_KIOSKO         787495 non-null  int64
 13  FREC_BPI_TD         787495 non-null  int64
 14  FREC_MON_TD         787495 non-null  int64
 15  PROM_CTD_TRX_6M     787495 non-null  float64
 16  ANT_CLIENTE         786572 non-null  float64
 17  REC_AGENTE_TD       92289 non-null   float64
 18  CTD_RECLAMOS_M1     787495 non-null  int64
dtypes: float64(6), int64(9), object(4)
memory usage: 114.2+ MB
```

En la informacion de nuestra data podemos observar que hay datos faltantes

Dimensión de nuestra data

[5]:
```
data_resultante.shape
```

[5]: (787495, 19)

Veamos en que columnas se encuentran valores nulos(NAN)

[6]:
```
data_resultante.isnull().any()
```

[6]:
```
CODMES                False
TARGET_MODEL2         False
EDAD                  False
SEXO                  False
DEPARTAMENTO           True
INGRESO_BRUTO_M1       True
FLG_CLIENTE           False
SEGMENTO              False
FLG_ADEL_SUELDO_M1    False
FREC_AGENTE           False
FLG_VEH_SF             True
FLG_CONV_SF            True
FREC_KIOSKO           False
FREC_BPI_TD           False
FREC_MON_TD           False
PROM_CTD_TRX_6M       False
ANT_CLIENTE            True
REC_AGENTE_TD          True
CTD_RECLAMOS_M1       False
dtype: bool
```

Ahora veamos cual es el porcentaje de esos valores nulos

[7]:
```
data_resultante.isnull().sum()/len(df)*100
```

[7]:
```
CODMES                 0.000000
TARGET_MODEL2          0.000000
EDAD                   0.000000
SEXO                   0.000000
DEPARTAMENTO           3.425419
INGRESO_BRUTO_M1      23.778437
FLG_CLIENTE            0.000000
SEGMENTO               0.000000
FLG_ADEL_SUELDO_M1     0.000000
FREC_AGENTE            0.000000
FLG_VEH_SF            34.207582
FLG_CONV_SF           34.207582
FREC_KIOSKO            0.000000
FREC_BPI_TD            0.000000
FREC_MON_TD            0.000000
PROM_CTD_TRX_6M        0.000000
ANT_CLIENTE            0.117207
REC_AGENTE_TD         88.280687
CTD_RECLAMOS_M1        0.000000
dtype: float64
```

Podemos apreciar que hay datos que sobre pasan mas del 30% de los faltos faltantes por lo tanto procedemos a eliminar esas columnas ya que son perjudiciales para nuestro análisis

Eliminando las columnas que poseen mas del 30% de los datos faltantes

[8]:
```
lista=["CODMES","TARGET_MODEL2","EDAD","SEXO","DEPARTAMENTO","INGRESO_BRUTO_M1","FLG_CLIENTE","S
      ␣
 ↪"FREC_AGENTE","FREC_KIOSKO","FREC_BPI_TD","FREC_MON_TD","PROM_CTD_TRX_6M","ANT_CLIENTE","CTD_
data_resultante_2=data_resultante[lista]
data_resultante_2.head(50)
```

[8]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	INGRESO_BRUTO_M1	\
0	201411	0	46	F	PIURA	NaN	
1	201411	0	54	M	LORETO	4718.0	
2	201411	0	81	M	NaN	NaN	
3	201411	0	42	M	PIURA	936.0	
4	201411	0	52	M	MOQUEGUA	5844.0	
5	201411	0	74	M	LA LIBERTAD	NaN	
6	201411	0	66	M	LA LIBERTAD	4232.0	
7	201411	0	57	M	LIMA	1580.0	
8	201411	0	65	M	CALLAO	NaN	
9	201411	0	63	M	ANCASH	936.0	
10	201411	0	43	M	LIMA	1421.0	
11	201411	0	64	F	LIMA	NaN	
12	201411	0	114	M	NaN	NaN	
13	201411	0	48	F	LIMA	809.0	
14	201411	0	48	F	MOQUEGUA	739.0	
15	201411	0	114	F	NaN	NaN	
16	201411	0	114	M	NaN	NaN	
17	201411	0	42	F	LIMA	739.0	
18	201411	0	93	M	LIMA	NaN	
19	201411	0	114	F	NaN	NaN	
20	201411	0	114	F	NaN	NaN	
21	201411	0	39	M	LIMA	749.0	
22	201411	0	45	M	LIMA	936.0	
23	201411	0	50	M	LIMA	936.0	
24	201411	0	114	F	NaN	NaN	
25	201411	0	114	M	NaN	NaN	
26	201411	0	114	F	NaN	NaN	
27	201411	0	114	M	NaN	NaN	
28	201411	0	114	M	NaN	NaN	
29	201411	0	114	M	NaN	NaN	
30	201411	0	114	M	NaN	NaN	
31	201411	0	114	M	NaN	NaN	
32	201411	0	114	F	NaN	NaN	
33	201411	0	52	F	AREQUIPA	739.0	
34	201411	0	57	M	LIMA	858.0	
35	201411	0	114	F	NaN	NaN	

36	201411	0	48	M	LIMA	NaN
37	201411	0	57	F	LIMA	3962.0
38	201411	0	52	M	CALLAO	NaN
39	201411	0	61	F	ANCASH	739.0
40	201411	0	48	M	LIMA	858.0
41	201411	0	46	M	CALLAO	NaN
42	201411	0	45	F	LIMA	NaN
43	201411	0	54	F	LIMA	NaN
44	201411	0	44	M	LIMA	818.0
45	201411	0	42	M	LIMA	2777.0
46	201411	0	44	F	LIMA	NaN
47	201411	0	53	M	NaN	NaN
48	201411	0	47	F	LIMA	1093.0
49	201411	0	37	M	LIMA	NaN

	FLG_CLIENTE	SEGMENTO	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	\
0	NO CLIENTE	2	0	0	0	
1	CLIENTE	1BC	0	0	0	
2	CLIENTE	6	0	0	0	
3	CLIENTE	2	0	0	0	
4	CLIENTE	1BC	0	0	0	
5	CLIENTE	6	0	0	0	
6	CLIENTE	1BC	0	0	0	
7	NO CLIENTE	2	0	0	5	
8	NO CLIENTE	2	0	0	0	
9	CLIENTE	2	0	0	0	
10	CLIENTE	2	0	0	0	
11	NO CLIENTE	2	0	0	0	
12	CLIENTE	6	0	0	0	
13	CLIENTE	2	0	0	0	
14	CLIENTE	3	0	0	0	
15	CLIENTE	6	0	0	0	
16	CLIENTE	6	0	0	0	
17	CLIENTE	3	0	0	0	
18	NO CLIENTE	6	0	0	0	
19	CLIENTE	6	0	0	0	
20	CLIENTE	6	0	0	0	
21	NO CLIENTE	3	0	0	0	
22	CLIENTE	3	0	0	0	
23	CLIENTE	3	0	0	0	
24	CLIENTE	6	0	0	0	
25	CLIENTE	6	0	0	0	
26	CLIENTE	6	0	0	0	
27	CLIENTE	6	0	0	0	
28	CLIENTE	6	0	0	0	
29	CLIENTE	6	0	0	0	
30	CLIENTE	6	0	0	0	

31	CLIENTE	6	0	0	0
32	CLIENTE	6	0	0	0
33	CLIENTE	2	0	0	0
34	CLIENTE	3	0	0	0
35	CLIENTE	6	0	0	0
36	NO CLIENTE	2	0	0	0
37	NO CLIENTE	1BC	0	0	2
38	NO CLIENTE	3	0	0	0
39	CLIENTE	5	0	0	0
40	CLIENTE	3	0	0	0
41	NO CLIENTE	6	0	0	0
42	NO CLIENTE	2	0	0	0
43	NO CLIENTE	6	0	0	0
44	CLIENTE	3	0	0	0
45	CLIENTE	3	0	0	3
46	NO CLIENTE	6	0	0	0
47	CLIENTE	6	0	0	0
48	CLIENTE	3	0	0	0
49	NO CLIENTE	6	0	0	0

	FREC_BPI_TD	FREC_MON_TD	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1
0	0	0	0.000000	224.0	0
1	0	0	0.000000	123.0	0
2	0	0	0.000000	264.0	0
3	0	0	0.000000	263.0	0
4	0	0	0.000000	263.0	0
5	0	0	0.000000	256.0	0
6	0	6	0.000000	85.0	0
7	6	6	2.166667	151.0	0
8	0	0	3.333333	778.0	0
9	0	0	0.000000	272.0	0
10	0	0	0.000000	11.0	0
11	0	0	0.000000	21.0	0
12	0	0	0.000000	281.0	0
13	0	0	0.000000	209.0	0
14	0	0	0.000000	208.0	0
15	0	0	0.000000	233.0	0
16	0	0	0.000000	216.0	0
17	0	0	0.000000	233.0	0
18	0	0	0.000000	281.0	0
19	0	0	0.000000	263.0	0
20	0	0	0.000000	221.0	0
21	0	0	0.000000	163.0	0
22	0	0	0.000000	215.0	0
23	0	0	0.000000	257.0	0
24	0	0	0.000000	206.0	0
25	0	0	0.000000	281.0	0

26	0	0	0.000000	280.0	0
27	0	0	0.000000	220.0	0
28	0	0	0.000000	275.0	0
29	0	0	0.000000	263.0	0
30	0	0	0.000000	280.0	0
31	0	0	0.000000	263.0	0
32	0	0	0.000000	203.0	0
33	0	0	0.000000	202.0	0
34	0	0	0.000000	264.0	0
35	0	0	0.000000	263.0	0
36	0	0	0.000000	265.0	0
37	0	0	0.000000	778.0	0
38	0	0	0.000000	85.0	0
39	0	0	0.000000	257.0	0
40	0	0	0.000000	191.0	0
41	0	0	0.000000	162.0	0
42	0	0	0.000000	185.0	0
43	0	0	0.000000	159.0	0
44	0	0	0.000000	280.0	0
45	0	0	0.333333	76.0	0
46	0	0	0.000000	778.0	0
47	0	0	0.000000	280.0	0
48	0	0	0.000000	164.0	0
49	0	0	0.000000	209.0	0

[9]:
```
data_resultante_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 16 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   CODMES              787495 non-null  int64
 1   TARGET_MODEL2       787495 non-null  int64
 2   EDAD                787495 non-null  int64
 3   SEXO                787495 non-null  object
 4   DEPARTAMENTO        760520 non-null  object
 5   INGRESO_BRUTO_M1    600241 non-null  float64
 6   FLG_CLIENTE         787495 non-null  object
 7   SEGMENTO            787495 non-null  object
 8   FLG_ADEL_SUELDO_M1  787495 non-null  int64
 9   FREC_AGENTE         787495 non-null  int64
 10  FREC_KIOSKO         787495 non-null  int64
 11  FREC_BPI_TD         787495 non-null  int64
 12  FREC_MON_TD         787495 non-null  int64
 13  PROM_CTD_TRX_6M     787495 non-null  float64
 14  ANT_CLIENTE         786572 non-null  float64
```

```
 15  CTD_RECLAMOS_M1     787495 non-null  int64
dtypes: float64(3), int64(9), object(4)
memory usage: 96.1+ MB
```

[10]:
```
data_resultante_2.isnull().sum()/len(data_resultante_2)*100
```

[10]:
```
CODMES                 0.000000
TARGET_MODEL2          0.000000
EDAD                   0.000000
SEXO                   0.000000
DEPARTAMENTO           3.425419
INGRESO_BRUTO_M1      23.778437
FLG_CLIENTE            0.000000
SEGMENTO               0.000000
FLG_ADEL_SUELDO_M1     0.000000
FREC_AGENTE            0.000000
FREC_KIOSKO            0.000000
FREC_BPI_TD            0.000000
FREC_MON_TD            0.000000
PROM_CTD_TRX_6M        0.000000
ANT_CLIENTE            0.117207
CTD_RECLAMOS_M1        0.000000
dtype: float64
```

Descripción de la data

[11]:
```
data_resultante_2.describe()
```

[11]:

	CODMES	TARGET_MODEL2	EDAD	INGRESO_BRUTO_M1	\
count	787495.000000	787495.000000	787495.000000	600241.000000	
mean	201471.503451	0.055551	39.250776	2565.256405	
std	43.304872	0.229052	15.752984	3313.887381	
min	201411.000000	0.000000	0.000000	681.000000	
25%	201411.000000	0.000000	28.000000	1011.000000	
50%	201502.000000	0.000000	36.000000	1533.000000	
75%	201503.000000	0.000000	46.000000	2781.000000	
max	201503.000000	1.000000	114.000000	214284.000000	

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	\
count	787495.000000	787495.000000	787495.000000	787495.000000	
mean	0.066060	0.434386	0.913956	0.426572	
std	0.248387	1.174521	1.681188	1.330529	
min	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	
75%	0.000000	0.000000	1.000000	0.000000	
max	1.000000	6.000000	6.000000	6.000000	

	FREC_MON_TD	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1
count	787495.000000	787495.000000	786572.000000	787495.000000
mean	0.648228	0.617803	105.254629	0.000192
std	1.480704	2.901885	158.734024	0.013846
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	24.000000	0.000000
50%	0.000000	0.000000	61.000000	0.000000
75%	0.000000	0.000000	114.000000	0.000000
max	6.000000	190.333333	782.000000	1.000000

1.2.3 1.1.2. IMPUTACIÓN SIMPLE

Las variables que haremos por imputación simple serán **DEPARTAMENTO(3.425419%)** y **ANT_CLIENTE(0.117207%)** debido a que tienes un bajo porcentaje de NaN

Llamamos a nuestra DataFrame

[12]:
```
data_resultante_2.head(50)
```

[12]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	INGRESO_BRUTO_M1	\
0	201411	0	46	F	PIURA	NaN	
1	201411	0	54	M	LORETO	4718.0	
2	201411	0	81	M	NaN	NaN	
3	201411	0	42	M	PIURA	936.0	
4	201411	0	52	M	MOQUEGUA	5844.0	
5	201411	0	74	M	LA LIBERTAD	NaN	
6	201411	0	66	M	LA LIBERTAD	4232.0	
7	201411	0	57	M	LIMA	1580.0	
8	201411	0	65	M	CALLAO	NaN	
9	201411	0	63	M	ANCASH	936.0	
10	201411	0	43	M	LIMA	1421.0	
11	201411	0	64	F	LIMA	NaN	
12	201411	0	114	M	NaN	NaN	
13	201411	0	48	F	LIMA	809.0	
14	201411	0	48	F	MOQUEGUA	739.0	
15	201411	0	114	F	NaN	NaN	
16	201411	0	114	M	NaN	NaN	
17	201411	0	42	F	LIMA	739.0	
18	201411	0	93	M	LIMA	NaN	
19	201411	0	114	F	NaN	NaN	
20	201411	0	114	F	NaN	NaN	
21	201411	0	39	M	LIMA	749.0	
22	201411	0	45	M	LIMA	936.0	
23	201411	0	50	M	LIMA	936.0	
24	201411	0	114	F	NaN	NaN	
25	201411	0	114	M	NaN	NaN	
26	201411	0	114	F	NaN	NaN	

27	201411	0	114	M	NaN	NaN
28	201411	0	114	M	NaN	NaN
29	201411	0	114	M	NaN	NaN
30	201411	0	114	M	NaN	NaN
31	201411	0	114	M	NaN	NaN
32	201411	0	114	F	NaN	NaN
33	201411	0	52	F	AREQUIPA	739.0
34	201411	0	57	M	LIMA	858.0
35	201411	0	114	F	NaN	NaN
36	201411	0	48	M	LIMA	NaN
37	201411	0	57	F	LIMA	3962.0
38	201411	0	52	M	CALLAO	NaN
39	201411	0	61	F	ANCASH	739.0
40	201411	0	48	M	LIMA	858.0
41	201411	0	46	M	CALLAO	NaN
42	201411	0	45	F	LIMA	NaN
43	201411	0	54	F	LIMA	NaN
44	201411	0	44	M	LIMA	818.0
45	201411	0	42	M	LIMA	2777.0
46	201411	0	44	F	LIMA	NaN
47	201411	0	53	M	NaN	NaN
48	201411	0	47	F	LIMA	1093.0
49	201411	0	37	M	LIMA	NaN

	FLG_CLIENTE	SEGMENTO	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	\
0	NO CLIENTE	2	0	0	0	
1	CLIENTE	1BC	0	0	0	
2	CLIENTE	6	0	0	0	
3	CLIENTE	2	0	0	0	
4	CLIENTE	1BC	0	0	0	
5	CLIENTE	6	0	0	0	
6	CLIENTE	1BC	0	0	0	
7	NO CLIENTE	2	0	0	5	
8	NO CLIENTE	2	0	0	0	
9	CLIENTE	2	0	0	0	
10	CLIENTE	2	0	0	0	
11	NO CLIENTE	2	0	0	0	
12	CLIENTE	6	0	0	0	
13	CLIENTE	2	0	0	0	
14	CLIENTE	3	0	0	0	
15	CLIENTE	6	0	0	0	
16	CLIENTE	6	0	0	0	
17	CLIENTE	3	0	0	0	
18	NO CLIENTE	6	0	0	0	
19	CLIENTE	6	0	0	0	
20	CLIENTE	6	0	0	0	
21	NO CLIENTE	3	0	0	0	

22	CLIENTE	3	0	0	0	
23	CLIENTE	3	0	0	0	
24	CLIENTE	6	0	0	0	
25	CLIENTE	6	0	0	0	
26	CLIENTE	6	0	0	0	
27	CLIENTE	6	0	0	0	
28	CLIENTE	6	0	0	0	
29	CLIENTE	6	0	0	0	
30	CLIENTE	6	0	0	0	
31	CLIENTE	6	0	0	0	
32	CLIENTE	6	0	0	0	
33	CLIENTE	2	0	0	0	
34	CLIENTE	3	0	0	0	
35	CLIENTE	6	0	0	0	
36	NO CLIENTE	2	0	0	0	
37	NO CLIENTE	1BC	0	0	2	
38	NO CLIENTE	3	0	0	0	
39	CLIENTE	5	0	0	0	
40	CLIENTE	3	0	0	0	
41	NO CLIENTE	6	0	0	0	
42	NO CLIENTE	2	0	0	0	
43	NO CLIENTE	6	0	0	0	
44	CLIENTE	3	0	0	0	
45	CLIENTE	3	0	0	3	
46	NO CLIENTE	6	0	0	0	
47	CLIENTE	6	0	0	0	
48	CLIENTE	3	0	0	0	
49	NO CLIENTE	6	0	0	0	

	FREC_BPI_TD	FREC_MON_TD	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1
0	0	0	0.000000	224.0	0
1	0	0	0.000000	123.0	0
2	0	0	0.000000	264.0	0
3	0	0	0.000000	263.0	0
4	0	0	0.000000	263.0	0
5	0	0	0.000000	256.0	0
6	0	6	0.000000	85.0	0
7	6	6	2.166667	151.0	0
8	0	0	3.333333	778.0	0
9	0	0	0.000000	272.0	0
10	0	0	0.000000	11.0	0
11	0	0	0.000000	21.0	0
12	0	0	0.000000	281.0	0
13	0	0	0.000000	209.0	0
14	0	0	0.000000	208.0	0
15	0	0	0.000000	233.0	0
16	0	0	0.000000	216.0	0

17	0	0	0.000000	233.0	0
18	0	0	0.000000	281.0	0
19	0	0	0.000000	263.0	0
20	0	0	0.000000	221.0	0
21	0	0	0.000000	163.0	0
22	0	0	0.000000	215.0	0
23	0	0	0.000000	257.0	0
24	0	0	0.000000	206.0	0
25	0	0	0.000000	281.0	0
26	0	0	0.000000	280.0	0
27	0	0	0.000000	220.0	0
28	0	0	0.000000	275.0	0
29	0	0	0.000000	263.0	0
30	0	0	0.000000	280.0	0
31	0	0	0.000000	263.0	0
32	0	0	0.000000	203.0	0
33	0	0	0.000000	202.0	0
34	0	0	0.000000	264.0	0
35	0	0	0.000000	263.0	0
36	0	0	0.000000	265.0	0
37	0	0	0.000000	778.0	0
38	0	0	0.000000	85.0	0
39	0	0	0.000000	257.0	0
40	0	0	0.000000	191.0	0
41	0	0	0.000000	162.0	0
42	0	0	0.000000	185.0	0
43	0	0	0.000000	159.0	0
44	0	0	0.000000	280.0	0
45	0	0	0.333333	76.0	0
46	0	0	0.000000	778.0	0
47	0	0	0.000000	280.0	0
48	0	0	0.000000	164.0	0
49	0	0	0.000000	209.0	0

Eliminamos la variable Ingreso Bruto Debido a la variable **INGRESO_BRUTO_M1** tiene un porcentaje de 23.778437% de NaN, entonces no se puede aplicar una **IMPUTACIÓN SIMPLE** sino una **IMPUTACIÓN SOFISTICADA**. Por ello eliminamos por el momento esa variable y lo guardamos en un objeto para luego concatenarlo con las otras variables que han sido completadas

[13]:
```
X=data_resultante_2.drop(["INGRESO_BRUTO_M1"],axis=1)
X.head()
```

[13]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	NaN	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	

```
4  201411              0    52    M     MOQUEGUA     CLIENTE      1BC

   FLG_ADEL_SUELDO_M1  FREC_AGENTE  FREC_KIOSKO  FREC_BPI_TD  FREC_MON_TD  \
0                   0            0            0            0            0
1                   0            0            0            0            0
2                   0            0            0            0            0
3                   0            0            0            0            0
4                   0            0            0            0            0

   PROM_CTD_TRX_6M  ANT_CLIENTE  CTD_RECLAMOS_M1
0              0.0        224.0                0
1              0.0        123.0                0
2              0.0        264.0                0
3              0.0        263.0                0
4              0.0        263.0                0
```

Guardando la variable INGRESO_BRUTO_M1

[14]:
```
y=data_resultante_2["INGRESO_BRUTO_M1"]
y.head(50)
```

[14]:
```
0        NaN
1     4718.0
2        NaN
3      936.0
4     5844.0
5        NaN
6     4232.0
7     1580.0
8        NaN
9      936.0
10    1421.0
11       NaN
12       NaN
13     809.0
14     739.0
15       NaN
16       NaN
17     739.0
18       NaN
19       NaN
20       NaN
21     749.0
22     936.0
23     936.0
24       NaN
25       NaN
```

```
26       NaN
27       NaN
28       NaN
29       NaN
30       NaN
31       NaN
32       NaN
33     739.0
34     858.0
35       NaN
36       NaN
37    3962.0
38       NaN
39     739.0
40     858.0
41       NaN
42       NaN
43       NaN
44     818.0
45    2777.0
46       NaN
47       NaN
48    1093.0
49       NaN
Name: INGRESO_BRUTO_M1, dtype: float64
```

Sacando las columnas de mi objeto X (data sin la variable INGRESO_BRUTO_M1)

[15]:
```
columnas=X.columns.to_list()
columnas
```

[15]:
```
['CODMES',
 'TARGET_MODEL2',
 'EDAD',
 'SEXO',
 'DEPARTAMENTO',
 'FLG_CLIENTE',
 'SEGMENTO',
 'FLG_ADEL_SUELDO_M1',
 'FREC_AGENTE',
 'FREC_KIOSKO',
 'FREC_BPI_TD',
 'FREC_MON_TD',
 'PROM_CTD_TRX_6M',
 'ANT_CLIENTE',
 'CTD_RECLAMOS_M1']
```

[16]:
```
X.dtypes
```

[16]:
```
CODMES                  int64
TARGET_MODEL2           int64
EDAD                    int64
SEXO                   object
DEPARTAMENTO           object
FLG_CLIENTE            object
SEGMENTO               object
FLG_ADEL_SUELDO_M1      int64
FREC_AGENTE             int64
FREC_KIOSKO             int64
FREC_BPI_TD             int64
FREC_MON_TD             int64
PROM_CTD_TRX_6M       float64
ANT_CLIENTE           float64
CTD_RECLAMOS_M1         int64
dtype: object
```

Agrupando según si tipo de variable

[17]:
```
X.columns.to_series().groupby(X.dtypes).size()
```

[17]:
```
int64      9
float64    2
object     4
dtype: int64
```

[18]:
```
tipos = X.columns.to_series().groupby(X.dtypes).groups
tipos
```

[18]:
```
{int64: ['CODMES', 'TARGET_MODEL2', 'EDAD', 'FLG_ADEL_SUELDO_M1', 'FREC_AGENTE',
'FREC_KIOSKO', 'FREC_BPI_TD', 'FREC_MON_TD', 'CTD_RECLAMOS_M1'], float64:
['PROM_CTD_TRX_6M', 'ANT_CLIENTE'], object: ['SEXO', 'DEPARTAMENTO',
'FLG_CLIENTE', 'SEGMENTO']}
```

Como se observa en la descripcion del DataFrame la columna del tipo entero no presenta ningún valor NaN, entonces solo completaremos las columnas del tipo categorico y flotante

Armando las listas de las columnas categóricas y numéricas (flotante)

[19]:
```
#Armando lista de columnas categóricas
#creamos la columna categórica
col_categorica = tipos[np.dtype('object')].to_list()
print("La cantidad de columnas con datos categóricos son: ",len(col_categorica))
print("lista:\n",col_categorica)

#Armando lista de columnas numéricas de tipo flotante
```

```
col_flotante = tipos[np.dtype('float64')].to_list()
print("La cantidad de columnas con datos flotantes son: ",len(col_flotante))
print("lista:\n",col_flotante)
```

```
La cantidad de columnas con datos categóricos son:  4
lista:
 ['SEXO', 'DEPARTAMENTO', 'FLG_CLIENTE', 'SEGMENTO']
La cantidad de columnas con datos flotantes son:  2
lista:
 ['PROM_CTD_TRX_6M', 'ANT_CLIENTE']
```

Completanto los valores faltantes

[20]:
```
#Completando los valores faltantes para variables categóricas.
for cat in col_categorica:
     #Vamos a guardar la moda en el objeto moda y se utilizara el metodo .mode()
    moda=X[cat].mode()[0]
    #Vamos rellenar a los elementos vacios por la moda para eso utilizaremos el
    #método .fillna(el valor que quieres almacenar)
    X[cat]=X[cat].fillna(moda)
```

[21]:
```
#Completando los valores faltantes para variables categóricas.
for flot in col_flotante:
     #Vamos a guardar la moda en el objeto moda y se utilizara el metodo .mode()
    mediana=X[flot].median()
    #Vamos rellenar a los elementos vacios por la moda para eso utilizaremos el
    #método .fillna(el valor que quieres almacenar)
    X[flot]=X[flot].fillna(mediana)
```

Verificamos si se completo

[22]:
```
X.isnull().sum()/len(X)*100
```

[22]:
```
CODMES                0.0
TARGET_MODEL2         0.0
EDAD                  0.0
SEXO                  0.0
DEPARTAMENTO          0.0
FLG_CLIENTE           0.0
SEGMENTO              0.0
FLG_ADEL_SUELDO_M1    0.0
FREC_AGENTE           0.0
FREC_KIOSKO           0.0
FREC_BPI_TD           0.0
FREC_MON_TD           0.0
PROM_CTD_TRX_6M       0.0
ANT_CLIENTE           0.0
CTD_RECLAMOS_M1       0.0
```

```
dtype: float64
```

[23]:
```
#Verificación de la data limpia de NAs
X.isnull().any().any()
```

[23]:
```
False
```

Mostramos la Data Frame resultante para las varibles que tuvieron un bajo porcentaje de NaN

[24]:
```
X.head(50)
```

[24]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	LIMA	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC	
5	201411	0	74	M	LA LIBERTAD	CLIENTE	6	
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC	
7	201411	0	57	M	LIMA	NO CLIENTE	2	
8	201411	0	65	M	CALLAO	NO CLIENTE	2	
9	201411	0	63	M	ANCASH	CLIENTE	2	
10	201411	0	43	M	LIMA	CLIENTE	2	
11	201411	0	64	F	LIMA	NO CLIENTE	2	
12	201411	0	114	M	LIMA	CLIENTE	6	
13	201411	0	48	F	LIMA	CLIENTE	2	
14	201411	0	48	F	MOQUEGUA	CLIENTE	3	
15	201411	0	114	F	LIMA	CLIENTE	6	
16	201411	0	114	M	LIMA	CLIENTE	6	
17	201411	0	42	F	LIMA	CLIENTE	3	
18	201411	0	93	M	LIMA	NO CLIENTE	6	
19	201411	0	114	F	LIMA	CLIENTE	6	
20	201411	0	114	F	LIMA	CLIENTE	6	
21	201411	0	39	M	LIMA	NO CLIENTE	3	
22	201411	0	45	M	LIMA	CLIENTE	3	
23	201411	0	50	M	LIMA	CLIENTE	3	
24	201411	0	114	F	LIMA	CLIENTE	6	
25	201411	0	114	M	LIMA	CLIENTE	6	
26	201411	0	114	F	LIMA	CLIENTE	6	
27	201411	0	114	M	LIMA	CLIENTE	6	
28	201411	0	114	M	LIMA	CLIENTE	6	
29	201411	0	114	M	LIMA	CLIENTE	6	
30	201411	0	114	M	LIMA	CLIENTE	6	
31	201411	0	114	M	LIMA	CLIENTE	6	
32	201411	0	114	F	LIMA	CLIENTE	6	
33	201411	0	52	F	AREQUIPA	CLIENTE	2	
34	201411	0	57	M	LIMA	CLIENTE	3	

35	201411	0	114	F	LIMA		CLIENTE	6
36	201411	0	48	M	LIMA	NO	CLIENTE	2
37	201411	0	57	F	LIMA	NO	CLIENTE	1BC
38	201411	0	52	M	CALLAO	NO	CLIENTE	3
39	201411	0	61	F	ANCASH		CLIENTE	5
40	201411	0	48	M	LIMA		CLIENTE	3
41	201411	0	46	M	CALLAO	NO	CLIENTE	6
42	201411	0	45	F	LIMA	NO	CLIENTE	2
43	201411	0	54	F	LIMA	NO	CLIENTE	6
44	201411	0	44	M	LIMA		CLIENTE	3
45	201411	0	42	M	LIMA		CLIENTE	3
46	201411	0	44	F	LIMA	NO	CLIENTE	6
47	201411	0	53	M	LIMA		CLIENTE	6
48	201411	0	47	F	LIMA		CLIENTE	3
49	201411	0	37	M	LIMA	NO	CLIENTE	6

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
5	0	0	0	0	0	
6	0	0	0	0	6	
7	0	0	5	6	6	
8	0	0	0	0	0	
9	0	0	0	0	0	
10	0	0	0	0	0	
11	0	0	0	0	0	
12	0	0	0	0	0	
13	0	0	0	0	0	
14	0	0	0	0	0	
15	0	0	0	0	0	
16	0	0	0	0	0	
17	0	0	0	0	0	
18	0	0	0	0	0	
19	0	0	0	0	0	
20	0	0	0	0	0	
21	0	0	0	0	0	
22	0	0	0	0	0	
23	0	0	0	0	0	
24	0	0	0	0	0	
25	0	0	0	0	0	
26	0	0	0	0	0	
27	0	0	0	0	0	
28	0	0	0	0	0	
29	0	0	0	0	0	

30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	2	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	3	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1
0	0.000000	224.0	0
1	0.000000	123.0	0
2	0.000000	264.0	0
3	0.000000	263.0	0
4	0.000000	263.0	0
5	0.000000	256.0	0
6	0.000000	85.0	0
7	2.166667	151.0	0
8	3.333333	778.0	0
9	0.000000	272.0	0
10	0.000000	11.0	0
11	0.000000	21.0	0
12	0.000000	281.0	0
13	0.000000	209.0	0
14	0.000000	208.0	0
15	0.000000	233.0	0
16	0.000000	216.0	0
17	0.000000	233.0	0
18	0.000000	281.0	0
19	0.000000	263.0	0
20	0.000000	221.0	0
21	0.000000	163.0	0
22	0.000000	215.0	0
23	0.000000	257.0	0
24	0.000000	206.0	0

25	0.000000	281.0	0
26	0.000000	280.0	0
27	0.000000	220.0	0
28	0.000000	275.0	0
29	0.000000	263.0	0
30	0.000000	280.0	0
31	0.000000	263.0	0
32	0.000000	203.0	0
33	0.000000	202.0	0
34	0.000000	264.0	0
35	0.000000	263.0	0
36	0.000000	265.0	0
37	0.000000	778.0	0
38	0.000000	85.0	0
39	0.000000	257.0	0
40	0.000000	191.0	0
41	0.000000	162.0	0
42	0.000000	185.0	0
43	0.000000	159.0	0
44	0.000000	280.0	0
45	0.333333	76.0	0
46	0.000000	778.0	0
47	0.000000	280.0	0
48	0.000000	164.0	0
49	0.000000	209.0	0

Concatenamos las Datas Frame (X e y)

[25]:
```
data_semifinal=pd.concat([X,y],axis=1)
data_semifinal.head(50)
```

[25]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	LIMA	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC	
5	201411	0	74	M	LA LIBERTAD	CLIENTE	6	
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC	
7	201411	0	57	M	LIMA	NO CLIENTE	2	
8	201411	0	65	M	CALLAO	NO CLIENTE	2	
9	201411	0	63	M	ANCASH	CLIENTE	2	
10	201411	0	43	M	LIMA	CLIENTE	2	
11	201411	0	64	F	LIMA	NO CLIENTE	2	
12	201411	0	114	M	LIMA	CLIENTE	6	
13	201411	0	48	F	LIMA	CLIENTE	2	
14	201411	0	48	F	MOQUEGUA	CLIENTE	3	

15	201411	0	114	F	LIMA		CLIENTE	6
16	201411	0	114	M	LIMA		CLIENTE	6
17	201411	0	42	F	LIMA		CLIENTE	3
18	201411	0	93	M	LIMA	NO	CLIENTE	6
19	201411	0	114	F	LIMA		CLIENTE	6
20	201411	0	114	F	LIMA		CLIENTE	6
21	201411	0	39	M	LIMA	NO	CLIENTE	3
22	201411	0	45	M	LIMA		CLIENTE	3
23	201411	0	50	M	LIMA		CLIENTE	3
24	201411	0	114	F	LIMA		CLIENTE	6
25	201411	0	114	M	LIMA		CLIENTE	6
26	201411	0	114	F	LIMA		CLIENTE	6
27	201411	0	114	M	LIMA		CLIENTE	6
28	201411	0	114	M	LIMA		CLIENTE	6
29	201411	0	114	M	LIMA		CLIENTE	6
30	201411	0	114	M	LIMA		CLIENTE	6
31	201411	0	114	M	LIMA		CLIENTE	6
32	201411	0	114	F	LIMA		CLIENTE	6
33	201411	0	52	F	AREQUIPA		CLIENTE	2
34	201411	0	57	M	LIMA		CLIENTE	3
35	201411	0	114	F	LIMA		CLIENTE	6
36	201411	0	48	M	LIMA	NO	CLIENTE	2
37	201411	0	57	F	LIMA	NO	CLIENTE	1BC
38	201411	0	52	M	CALLAO	NO	CLIENTE	3
39	201411	0	61	F	ANCASH		CLIENTE	5
40	201411	0	48	M	LIMA		CLIENTE	3
41	201411	0	46	M	CALLAO	NO	CLIENTE	6
42	201411	0	45	F	LIMA	NO	CLIENTE	2
43	201411	0	54	F	LIMA	NO	CLIENTE	6
44	201411	0	44	M	LIMA		CLIENTE	3
45	201411	0	42	M	LIMA		CLIENTE	3
46	201411	0	44	F	LIMA	NO	CLIENTE	6
47	201411	0	53	M	LIMA		CLIENTE	6
48	201411	0	47	F	LIMA		CLIENTE	3
49	201411	0	37	M	LIMA	NO	CLIENTE	6

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
5	0	0	0	0	0	
6	0	0	0	0	6	
7	0	0	5	6	6	
8	0	0	0	0	0	
9	0	0	0	0	0	

10	0	0	0	0	0
11	0	0	0	0	0
12	0	0	0	0	0
13	0	0	0	0	0
14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0
17	0	0	0	0	0
18	0	0	0	0	0
19	0	0	0	0	0
20	0	0	0	0	0
21	0	0	0	0	0
22	0	0	0	0	0
23	0	0	0	0	0
24	0	0	0	0	0
25	0	0	0	0	0
26	0	0	0	0	0
27	0	0	0	0	0
28	0	0	0	0	0
29	0	0	0	0	0
30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	2	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	3	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
0	0.000000	224.0	0	NaN
1	0.000000	123.0	0	4718.0
2	0.000000	264.0	0	NaN
3	0.000000	263.0	0	936.0
4	0.000000	263.0	0	5844.0

5	0.000000	256.0	0	NaN
6	0.000000	85.0	0	4232.0
7	2.166667	151.0	0	1580.0
8	3.333333	778.0	0	NaN
9	0.000000	272.0	0	936.0
10	0.000000	11.0	0	1421.0
11	0.000000	21.0	0	NaN
12	0.000000	281.0	0	NaN
13	0.000000	209.0	0	809.0
14	0.000000	208.0	0	739.0
15	0.000000	233.0	0	NaN
16	0.000000	216.0	0	NaN
17	0.000000	233.0	0	739.0
18	0.000000	281.0	0	NaN
19	0.000000	263.0	0	NaN
20	0.000000	221.0	0	NaN
21	0.000000	163.0	0	749.0
22	0.000000	215.0	0	936.0
23	0.000000	257.0	0	936.0
24	0.000000	206.0	0	NaN
25	0.000000	281.0	0	NaN
26	0.000000	280.0	0	NaN
27	0.000000	220.0	0	NaN
28	0.000000	275.0	0	NaN
29	0.000000	263.0	0	NaN
30	0.000000	280.0	0	NaN
31	0.000000	263.0	0	NaN
32	0.000000	203.0	0	NaN
33	0.000000	202.0	0	739.0
34	0.000000	264.0	0	858.0
35	0.000000	263.0	0	NaN
36	0.000000	265.0	0	NaN
37	0.000000	778.0	0	3962.0
38	0.000000	85.0	0	NaN
39	0.000000	257.0	0	739.0
40	0.000000	191.0	0	858.0
41	0.000000	162.0	0	NaN
42	0.000000	185.0	0	NaN
43	0.000000	159.0	0	NaN
44	0.000000	280.0	0	818.0
45	0.333333	76.0	0	2777.0
46	0.000000	778.0	0	NaN
47	0.000000	280.0	0	NaN
48	0.000000	164.0	0	1093.0
49	0.000000	209.0	0	NaN

Veamos los porcentajes de los datos faltantes

[26]:
```
data_semifinal.isnull().sum()/len(data_semifinal)*100
```

[26]:
```
CODMES                 0.000000
TARGET_MODEL2          0.000000
EDAD                   0.000000
SEXO                   0.000000
DEPARTAMENTO           0.000000
FLG_CLIENTE            0.000000
SEGMENTO               0.000000
FLG_ADEL_SUELDO_M1     0.000000
FREC_AGENTE            0.000000
FREC_KIOSKO            0.000000
FREC_BPI_TD            0.000000
FREC_MON_TD            0.000000
PROM_CTD_TRX_6M        0.000000
ANT_CLIENTE            0.000000
CTD_RECLAMOS_M1        0.000000
INGRESO_BRUTO_M1      23.778437
dtype: float64
```

[27]:
```
data_semifinal.describe()
```

[27]:

	CODMES	TARGET_MODEL2	EDAD	FLG_ADEL_SUELDO_M1
count	787495.000000	787495.000000	787495.000000	787495.000000
mean	201471.503451	0.055551	39.250776	0.066060
std	43.304872	0.229052	15.752984	0.248387
min	201411.000000	0.000000	0.000000	0.000000
25%	201411.000000	0.000000	28.000000	0.000000
50%	201502.000000	0.000000	36.000000	0.000000
75%	201503.000000	0.000000	46.000000	0.000000
max	201503.000000	1.000000	114.000000	1.000000

	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD
count	787495.000000	787495.000000	787495.000000	787495.000000
mean	0.434386	0.913956	0.426572	0.648228
std	1.174521	1.681188	1.330529	1.480704
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000
75%	0.000000	1.000000	0.000000	0.000000
max	6.000000	6.000000	6.000000	6.000000

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
count	787495.000000	787495.000000	787495.000000	600241.000000
mean	0.617803	105.202759	0.000192	2565.256405
std	2.901885	158.648199	0.013846	3313.887381
min	0.000000	0.000000	0.000000	681.000000

25%	0.000000	24.000000	0.000000	1011.000000
50%	0.000000	61.000000	0.000000	1533.000000
75%	0.000000	114.000000	0.000000	2781.000000
max	190.333333	782.000000	1.000000	214284.000000

1.2.4 1.1.3. IMPUTACIÓN POR MÉTODO SOFISTICADO

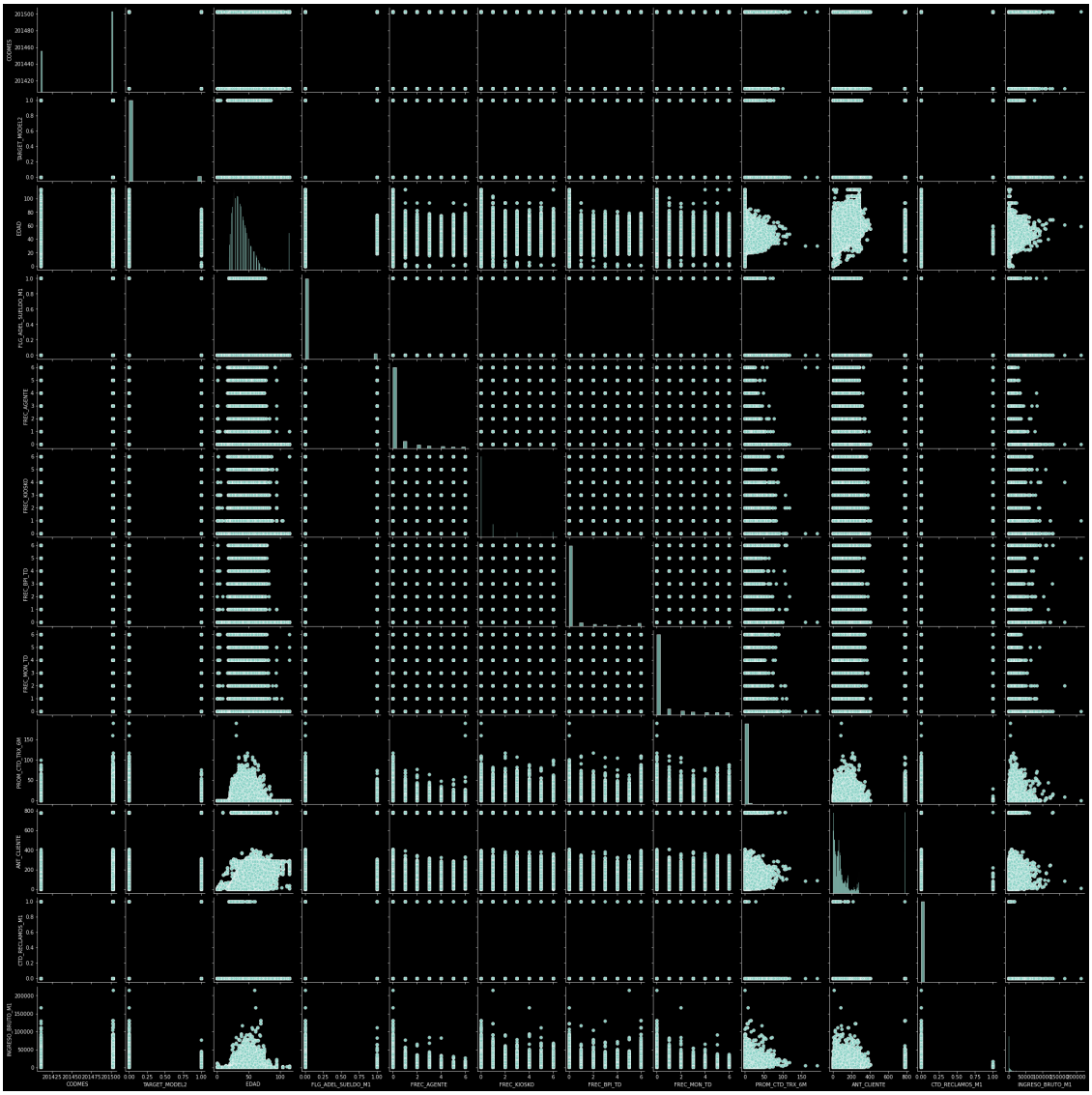
[28]:
```
from sklearn.linear_model import LinearRegression #Para imputación por regresión.
```

[29]:
```
data_semifinal.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 16 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   CODMES              787495 non-null  int64  
 1   TARGET_MODEL2       787495 non-null  int64  
 2   EDAD                787495 non-null  int64  
 3   SEXO                787495 non-null  object 
 4   DEPARTAMENTO        787495 non-null  object 
 5   FLG_CLIENTE         787495 non-null  object 
 6   SEGMENTO            787495 non-null  object 
 7   FLG_ADEL_SUELDO_M1  787495 non-null  int64  
 8   FREC_AGENTE         787495 non-null  int64  
 9   FREC_KIOSKO         787495 non-null  int64  
 10  FREC_BPI_TD         787495 non-null  int64  
 11  FREC_MON_TD         787495 non-null  int64  
 12  PROM_CTD_TRX_6M     787495 non-null  float64
 13  ANT_CLIENTE         787495 non-null  float64
 14  CTD_RECLAMOS_M1     787495 non-null  int64  
 15  INGRESO_BRUTO_M1    600241 non-null  float64
dtypes: float64(3), int64(9), object(4)
memory usage: 96.1+ MB
```

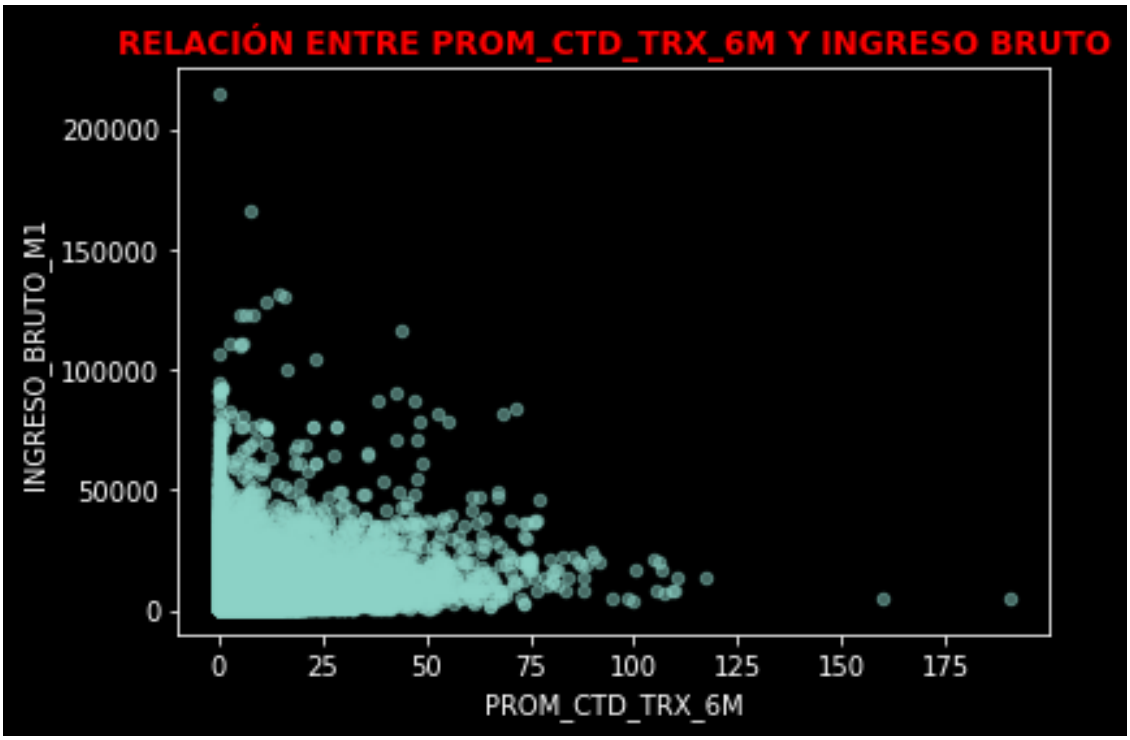
Vizualización de las relaciones entre variables cuantitativas

[30]:
```
with plt.style.context('dark_background'):
    sns.pairplot(data_semifinal)
```



Hallando la correlación de las variables cuantitativas más relacionadas

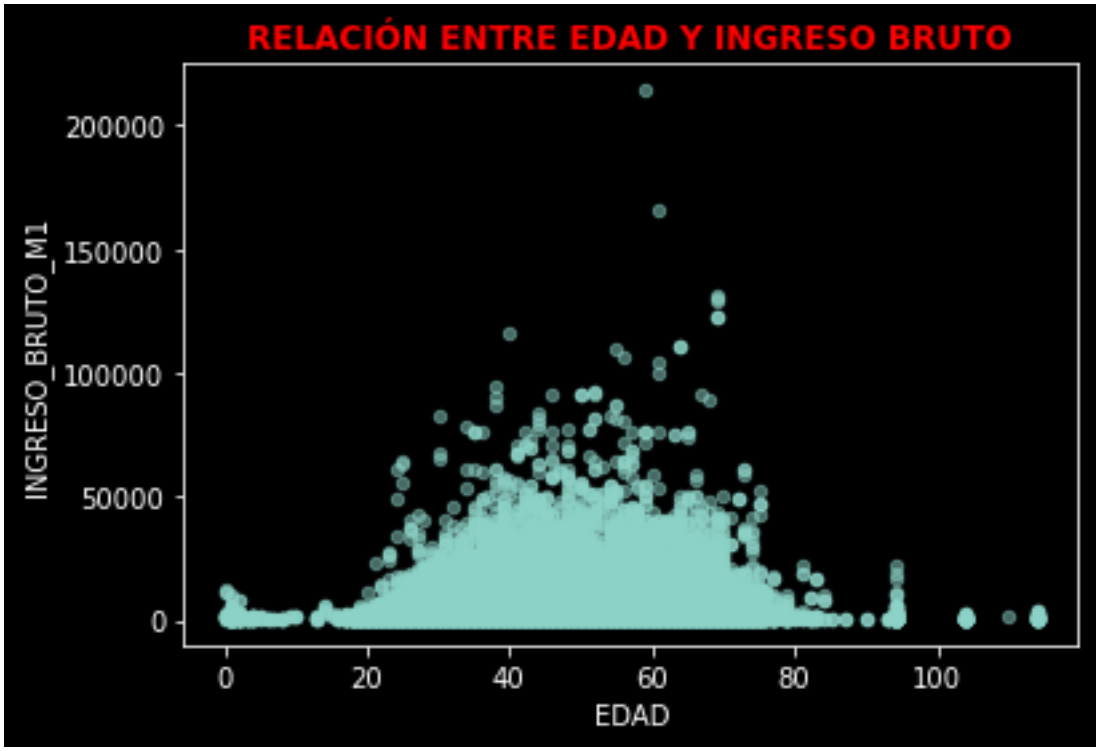
[31]:
```
with plt.style.context('dark_background'):
    data_semifinal.plot.scatter(x="PROM_CTD_TRX_6M",
                                y="INGRESO_BRUTO_M1",
                               alpha=0.5)
plt.title("RELACIÓN ENTRE PROM_CTD_TRX_6M Y INGRESO␣
 ↪BRUTO",weight='bold',color='red')
plt.show()
```



[32]:
```
P_C_T=data_semifinal["PROM_CTD_TRX_6M"]
ingreso=data_semifinal["INGRESO_BRUTO_M1"]
# PIB.cov(desempleo)
print("LA CORRELACION ES:",P_C_T.corr(ingreso)*100,"%")
print("LA COVARIANZA ES:",P_C_T.cov(ingreso))
```

```
LA CORRELACION ES: 29.629876433451624 %
LA COVARIANZA ES: 3218.81006427478
```

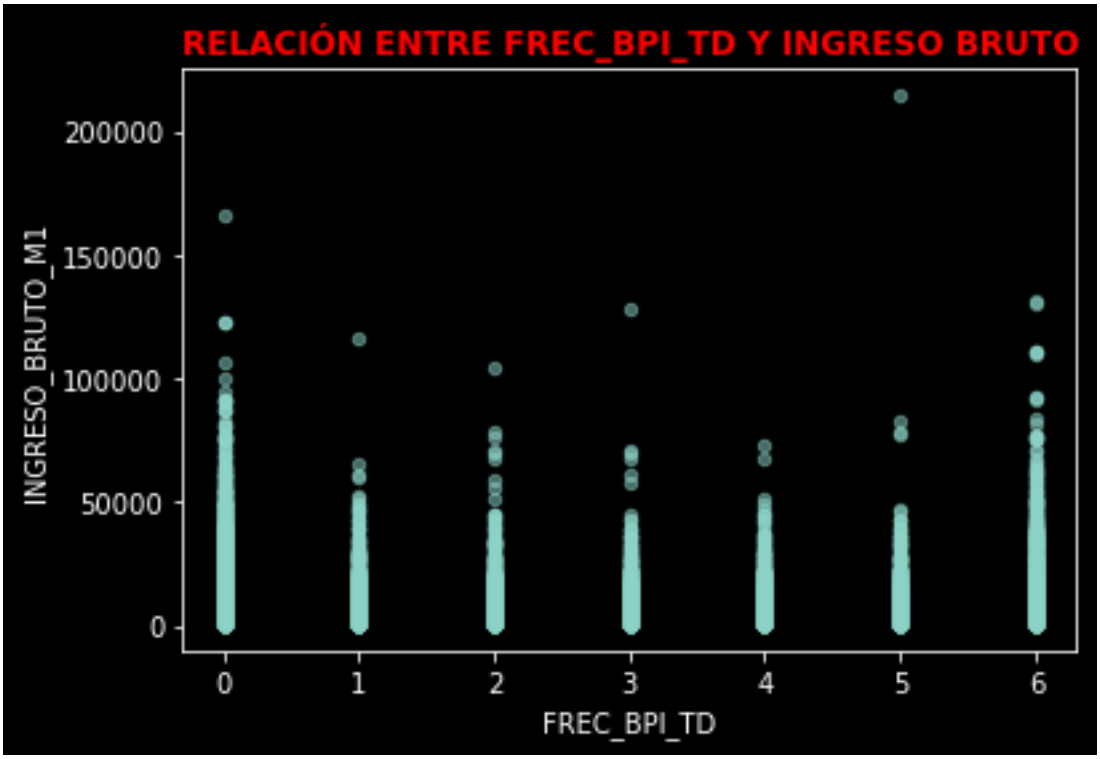
[33]:
```
with plt.style.context('dark_background'):
    data_semifinal.plot.scatter(x="EDAD",
                          y="INGRESO_BRUTO_M1",
                         alpha=0.5)
plt.title("RELACIÓN ENTRE EDAD Y INGRESO BRUTO",weight='bold',color='red')
plt.show()
```



[34]:
```
EDAD=data_semifinal["EDAD"]
ingreso=data_semifinal["INGRESO_BRUTO_M1"]
# PIB.cov(desempleo)
print("LA CORRELACION ES:",EDAD.corr(ingreso)*100,"%")
print("LA COVARIANZA ES:",EDAD.cov(ingreso))
```

```
LA CORRELACION ES: 18.577215007412857 %
LA COVARIANZA ES: 7463.2407863866765
```

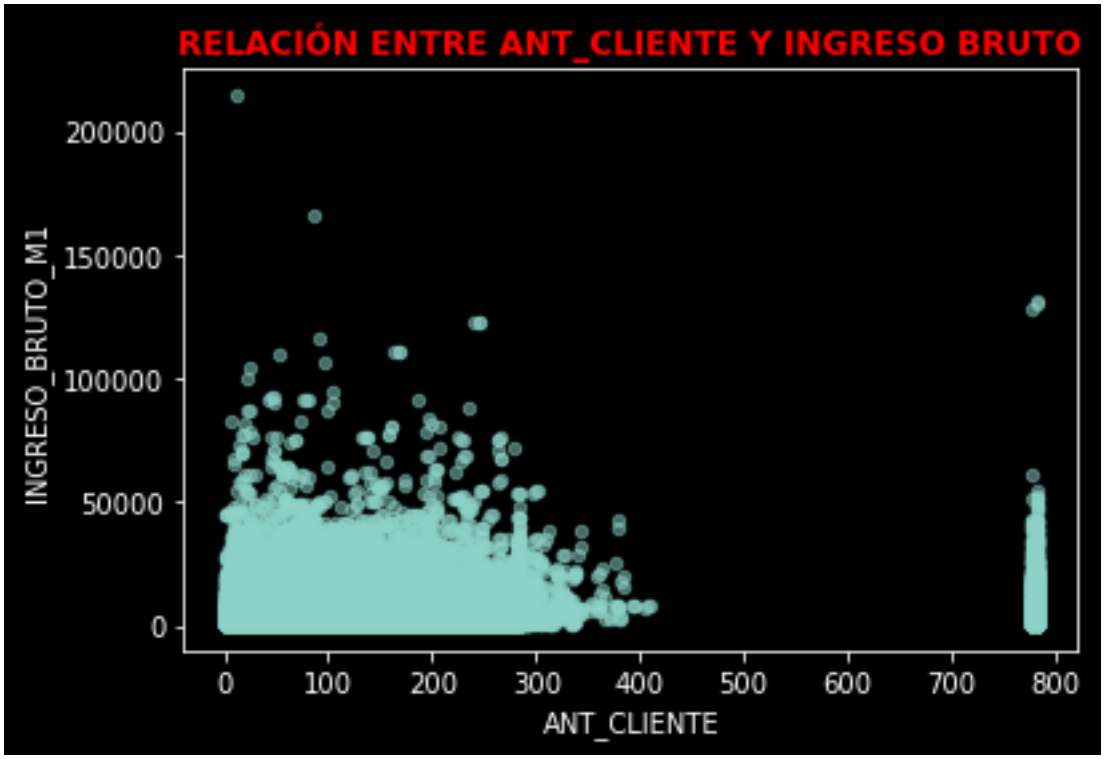
[35]:
```
with plt.style.context('dark_background'):
    data_semifinal.plot.scatter(x="FREC_BPI_TD",
                          y="INGRESO_BRUTO_M1",
                         alpha=0.5)
plt.title("RELACIÓN ENTRE FREC_BPI_TD Y INGRESO BRUTO",weight='bold',color='red')
plt.show()
```



[36]:
```
F_B_T=data_semifinal["FREC_BPI_TD"]
ingreso=data_semifinal["INGRESO_BRUTO_M1"]
# PIB.cov(desempleo)
print("LA CORRELACION ES:",F_B_T.corr(ingreso)*100,"%")
print("LA COVARIANZA ES:",F_B_T.cov(ingreso))
```

```
LA CORRELACION ES: 20.08935852242798 %
LA COVARIANZA ES: 951.6475553985903
```

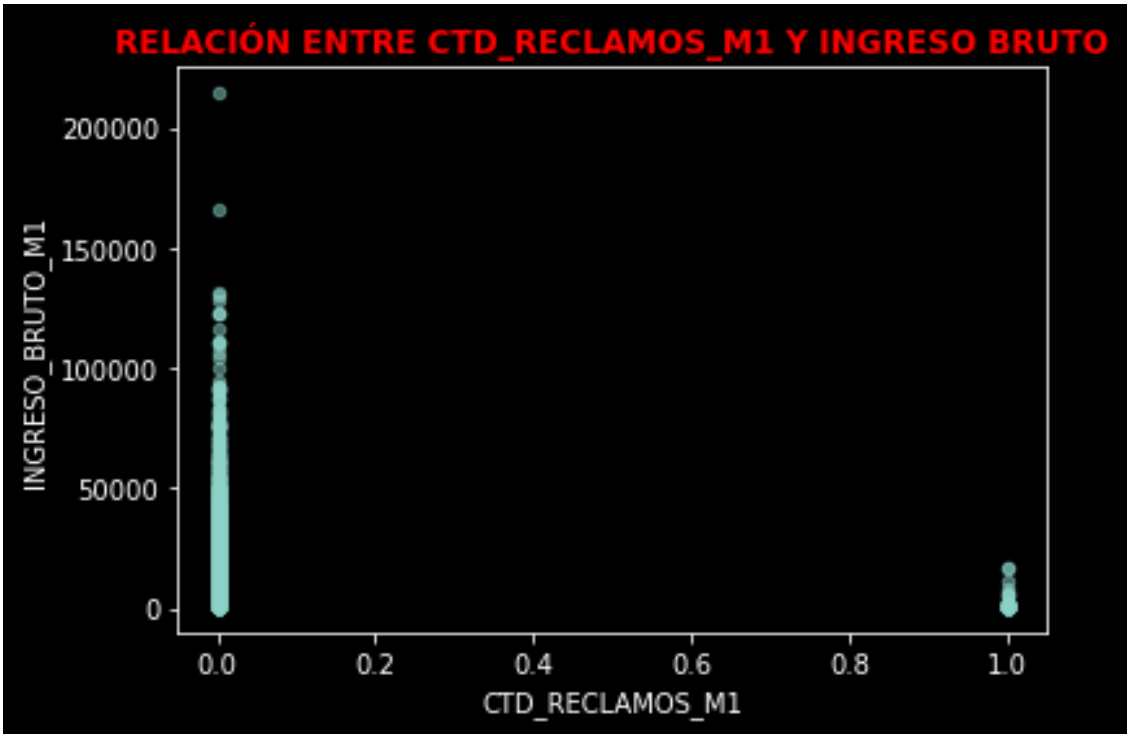
[37]:
```
with plt.style.context('dark_background'):
    data_semifinal.plot.scatter(x="ANT_CLIENTE",
                                y="INGRESO_BRUTO_M1",
                                  alpha=0.5)
plt.title("RELACIÓN ENTRE ANT_CLIENTE Y INGRESO BRUTO",weight='bold',color='red')
plt.show()
```



[38]:
```
CLIENTE=data_semifinal["ANT_CLIENTE"]
ingreso=data_semifinal["INGRESO_BRUTO_M1"]
# PIB.cov(desempleo)
print("LA CORRELACION ES:",CLIENTE.corr(ingreso)*100,"%")
print("LA COVARIANZA ES:",CLIENTE.cov(ingreso))
```

```
LA CORRELACION ES: 11.784921975960192 %
LA COVARIANZA ES: 52292.18469928967
```

[39]:
```
with plt.style.context('dark_background'):
    data_semifinal.plot.scatter(x="CTD_RECLAMOS_M1",
                                y="INGRESO_BRUTO_M1",
                               alpha=0.5)
plt.title("RELACIÓN ENTRE CTD_RECLAMOS_M1 Y INGRESO␣
 ↪BRUTO",weight='bold',color='red')
plt.show()
```



[40]:
```
RECLAMOS=data_semifinal["CTD_RECLAMOS_M1"]
ingreso=data_semifinal["INGRESO_BRUTO_M1"]
# PIB.cov(desempleo)
print("LA CORRELACION ES:",RECLAMOS.corr(ingreso)*100,"%")
print("LA COVARIANZA ES:",RECLAMOS.cov(ingreso))
```

```
LA CORRELACION ES: -0.2977346863640127 %
LA COVARIANZA ES: -0.12986249859350055
```

Llegamos a la conclusión que las variables mas relacionadas con la variable **INGRESO_BRUTO_M1** son:

1. PROM_CTD_TRX_6M(29.629876433451624 %)
2. FREC_BPI_TD(20.08935852242798 %)

1.2.5 REGRESICIÓN LINEAL

Mostrando los valores nulos

[41]:
```
nulos=pd.isna(data_semifinal.loc[:,"INGRESO_BRUTO_M1"])
nulos
```

[41]:
```
0     True
1    False
2     True
3    False
```

```
4         False
          ...
787490    False
787491    False
787492    False
787493    False
787494    False
Name: INGRESO_BRUTO_M1, Length: 787495, dtype: bool
```

Almacenamos en el objeto "data_nueva_nulos" todos los datos nulos y en el objeto "data_nueva_completos" los datos completos

[42]:
```
data_semifinal_nulos=data_semifinal.loc[nulos]
data_semifinal_completos=data_semifinal.loc[~nulos]
data_semifinal_completos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 600241 entries, 1 to 787494
Data columns (total 16 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   CODMES              600241 non-null  int64
 1   TARGET_MODEL2       600241 non-null  int64
 2   EDAD                600241 non-null  int64
 3   SEXO                600241 non-null  object
 4   DEPARTAMENTO        600241 non-null  object
 5   FLG_CLIENTE         600241 non-null  object
 6   SEGMENTO            600241 non-null  object
 7   FLG_ADEL_SUELDO_M1  600241 non-null  int64
 8   FREC_AGENTE         600241 non-null  int64
 9   FREC_KIOSKO         600241 non-null  int64
 10  FREC_BPI_TD         600241 non-null  int64
 11  FREC_MON_TD         600241 non-null  int64
 12  PROM_CTD_TRX_6M     600241 non-null  float64
 13  ANT_CLIENTE         600241 non-null  float64
 14  CTD_RECLAMOS_M1     600241 non-null  int64
 15  INGRESO_BRUTO_M1    600241 non-null  float64
dtypes: float64(3), int64(9), object(4)
memory usage: 77.9+ MB
```

[43]:
```
xtrain=data_semifinal_completos[["PROM_CTD_TRX_6M","FREC_BPI_TD"]]
ytrain=data_semifinal_completos[["INGRESO_BRUTO_M1"]]

xtest=data_semifinal_nulos[["PROM_CTD_TRX_6M","FREC_BPI_TD"]]
```

Crear un objeto de clase LinearRegression

[44]:
```
regre=LinearRegression()
type(regre)
```

[44]:
```
sklearn.linear_model._base.LinearRegression
```

Aprendo del subconjunto de completos

[45]:
```
regre.fit(xtrain,ytrain)
```

[45]:
```
LinearRegression()
```

[46]:
```
regre.score(xtrain,ytrain)
```

[46]:
```
0.1081069871931738
```

obteniendo los datos faltantes

[47]:
```
ypredicho=regre.predict(xtest)
ypredicho=np.round(ypredicho,1)
ypredicho
```

[47]:
```
array([[2184.9],
       [2184.9],
       [2184.9],
       ...,
       [2184.9],
       [2184.9],
       [2184.9]])
```

Mostramos los indeces donde los valores son nulos

[48]:
```
data_semifinal_nulos.index
```

[48]:
```
Int64Index([     0,      2,      5,      8,     11,     12,     15,     16,
                18,     19,
            ...
            787393, 787396, 787398, 787403, 787406, 787428, 787440, 787451,
            787478, 787480],
           dtype='int64', length=187254)
```

Incorporando los valores imputados al DF original

[49]:
```
data_semifinal.loc[data_semifinal_nulos.index,"INGRESO_BRUTO_M1"]=ypredicho
data_semifinal.head()
```

[49]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	

```
2  201411              0    81    M         LIMA     CLIENTE        6
3  201411              0    42    M        PIURA     CLIENTE        2
4  201411              0    52    M     MOQUEGUA     CLIENTE      1BC

   FLG_ADEL_SUELDO_M1  FREC_AGENTE  FREC_KIOSKO  FREC_BPI_TD  FREC_MON_TD  \
0                   0            0            0            0            0
1                   0            0            0            0            0
2                   0            0            0            0            0
3                   0            0            0            0            0
4                   0            0            0            0            0

   PROM_CTD_TRX_6M  ANT_CLIENTE  CTD_RECLAMOS_M1  INGRESO_BRUTO_M1
0              0.0        224.0                0            2184.9
1              0.0        123.0                0            4718.0
2              0.0        264.0                0            2184.9
3              0.0        263.0                0             936.0
4              0.0        263.0                0            5844.0
```

Porcentaje de NAs por Columnas

[50]:
```
data_semifinal.isnull().sum()*100/len(data_semifinal)
```

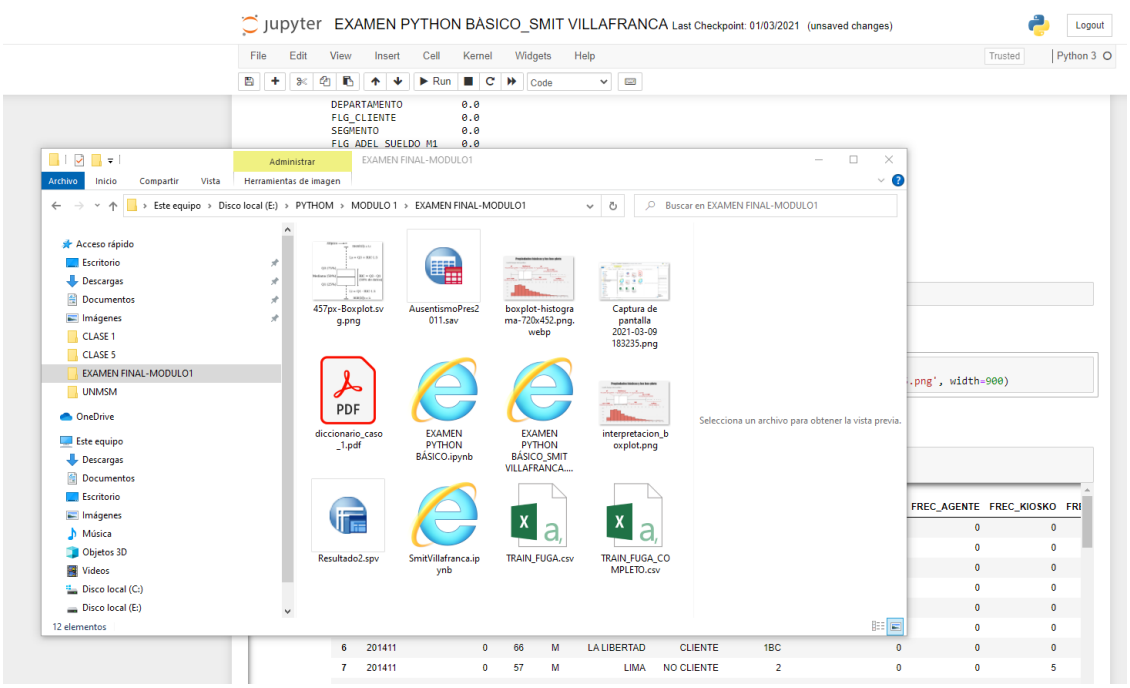
[50]:
```
CODMES                0.0
TARGET_MODEL2         0.0
EDAD                  0.0
SEXO                  0.0
DEPARTAMENTO          0.0
FLG_CLIENTE           0.0
SEGMENTO              0.0
FLG_ADEL_SUELDO_M1    0.0
FREC_AGENTE           0.0
FREC_KIOSKO           0.0
FREC_BPI_TD           0.0
FREC_MON_TD           0.0
PROM_CTD_TRX_6M       0.0
ANT_CLIENTE           0.0
CTD_RECLAMOS_M1       0.0
INGRESO_BRUTO_M1      0.0
dtype: float64
```

[51]:
```
data_final=data_semifinal
```

Guardando la data Imputada

[52]:
```
data_final.to_csv("TRAIN_FUGA_COMPLETO.csv", index=False)
Image(filename='E:\PYTHOM\MODULO 1\EXAMEN FINAL-MODULO1/data_completa.png',␣
 ↪width=900)
```

[52]:



1.2.6 1.2. Visualización

[53]:
```
dat=pd.read_csv("TRAIN_FUGA_COMPLETO.csv",sep=",", encoding="ISO-8859-1")
dat.head(50)
```

[53]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	LIMA	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC	
5	201411	0	74	M	LA LIBERTAD	CLIENTE	6	
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC	
7	201411	0	57	M	LIMA	NO CLIENTE	2	
8	201411	0	65	M	CALLAO	NO CLIENTE	2	
9	201411	0	63	M	ANCASH	CLIENTE	2	
10	201411	0	43	M	LIMA	CLIENTE	2	
11	201411	0	64	F	LIMA	NO CLIENTE	2	
12	201411	0	114	M	LIMA	CLIENTE	6	
13	201411	0	48	F	LIMA	CLIENTE	2	
14	201411	0	48	F	MOQUEGUA	CLIENTE	3	
15	201411	0	114	F	LIMA	CLIENTE	6	
16	201411	0	114	M	LIMA	CLIENTE	6	
17	201411	0	42	F	LIMA	CLIENTE	3	
18	201411	0	93	M	LIMA	NO CLIENTE	6	

19	201411	0	114	F	LIMA		CLIENTE	6
20	201411	0	114	F	LIMA		CLIENTE	6
21	201411	0	39	M	LIMA	NO	CLIENTE	3
22	201411	0	45	M	LIMA		CLIENTE	3
23	201411	0	50	M	LIMA		CLIENTE	3
24	201411	0	114	F	LIMA		CLIENTE	6
25	201411	0	114	M	LIMA		CLIENTE	6
26	201411	0	114	F	LIMA		CLIENTE	6
27	201411	0	114	M	LIMA		CLIENTE	6
28	201411	0	114	M	LIMA		CLIENTE	6
29	201411	0	114	M	LIMA		CLIENTE	6
30	201411	0	114	M	LIMA		CLIENTE	6
31	201411	0	114	M	LIMA		CLIENTE	6
32	201411	0	114	F	LIMA		CLIENTE	6
33	201411	0	52	F	AREQUIPA		CLIENTE	2
34	201411	0	57	M	LIMA		CLIENTE	3
35	201411	0	114	F	LIMA		CLIENTE	6
36	201411	0	48	M	LIMA	NO	CLIENTE	2
37	201411	0	57	F	LIMA	NO	CLIENTE	1BC
38	201411	0	52	M	CALLAO	NO	CLIENTE	3
39	201411	0	61	F	ANCASH		CLIENTE	5
40	201411	0	48	M	LIMA		CLIENTE	3
41	201411	0	46	M	CALLAO	NO	CLIENTE	6
42	201411	0	45	F	LIMA	NO	CLIENTE	2
43	201411	0	54	F	LIMA	NO	CLIENTE	6
44	201411	0	44	M	LIMA		CLIENTE	3
45	201411	0	42	M	LIMA		CLIENTE	3
46	201411	0	44	F	LIMA	NO	CLIENTE	6
47	201411	0	53	M	LIMA		CLIENTE	6
48	201411	0	47	F	LIMA		CLIENTE	3
49	201411	0	37	M	LIMA	NO	CLIENTE	6

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
5	0	0	0	0	0	
6	0	0	0	0	6	
7	0	0	5	6	6	
8	0	0	0	0	0	
9	0	0	0	0	0	
10	0	0	0	0	0	
11	0	0	0	0	0	
12	0	0	0	0	0	
13	0	0	0	0	0	

14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0
17	0	0	0	0	0
18	0	0	0	0	0
19	0	0	0	0	0
20	0	0	0	0	0
21	0	0	0	0	0
22	0	0	0	0	0
23	0	0	0	0	0
24	0	0	0	0	0
25	0	0	0	0	0
26	0	0	0	0	0
27	0	0	0	0	0
28	0	0	0	0	0
29	0	0	0	0	0
30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	2	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	3	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
0	0.000000	224.0	0	2184.9
1	0.000000	123.0	0	4718.0
2	0.000000	264.0	0	2184.9
3	0.000000	263.0	0	936.0
4	0.000000	263.0	0	5844.0
5	0.000000	256.0	0	2184.9
6	0.000000	85.0	0	4232.0
7	2.166667	151.0	0	1580.0
8	3.333333	778.0	0	3081.5

9	0.000000	272.0	0	936.0
10	0.000000	11.0	0	1421.0
11	0.000000	21.0	0	2184.9
12	0.000000	281.0	0	2184.9
13	0.000000	209.0	0	809.0
14	0.000000	208.0	0	739.0
15	0.000000	233.0	0	2184.9
16	0.000000	216.0	0	2184.9
17	0.000000	233.0	0	739.0
18	0.000000	281.0	0	2184.9
19	0.000000	263.0	0	2184.9
20	0.000000	221.0	0	2184.9
21	0.000000	163.0	0	749.0
22	0.000000	215.0	0	936.0
23	0.000000	257.0	0	936.0
24	0.000000	206.0	0	2184.9
25	0.000000	281.0	0	2184.9
26	0.000000	280.0	0	2184.9
27	0.000000	220.0	0	2184.9
28	0.000000	275.0	0	2184.9
29	0.000000	263.0	0	2184.9
30	0.000000	280.0	0	2184.9
31	0.000000	263.0	0	2184.9
32	0.000000	203.0	0	2184.9
33	0.000000	202.0	0	739.0
34	0.000000	264.0	0	858.0
35	0.000000	263.0	0	2184.9
36	0.000000	265.0	0	2184.9
37	0.000000	778.0	0	3962.0
38	0.000000	85.0	0	2184.9
39	0.000000	257.0	0	739.0
40	0.000000	191.0	0	858.0
41	0.000000	162.0	0	2184.9
42	0.000000	185.0	0	2184.9
43	0.000000	159.0	0	2184.9
44	0.000000	280.0	0	818.0
45	0.333333	76.0	0	2777.0
46	0.000000	778.0	0	2184.9
47	0.000000	280.0	0	2184.9
48	0.000000	164.0	0	1093.0
49	0.000000	209.0	0	2184.9

[54]:
```
dat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 16 columns):
```

```
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   CODMES               787495 non-null  int64
 1   TARGET_MODEL2        787495 non-null  int64
 2   EDAD                 787495 non-null  int64
 3   SEXO                 787495 non-null  object
 4   DEPARTAMENTO         787495 non-null  object
 5   FLG_CLIENTE          787495 non-null  object
 6   SEGMENTO             787495 non-null  object
 7   FLG_ADEL_SUELDO_M1   787495 non-null  int64
 8   FREC_AGENTE          787495 non-null  int64
 9   FREC_KIOSKO          787495 non-null  int64
 10  FREC_BPI_TD          787495 non-null  int64
 11  FREC_MON_TD          787495 non-null  int64
 12  PROM_CTD_TRX_6M      787495 non-null  float64
 13  ANT_CLIENTE          787495 non-null  float64
 14  CTD_RECLAMOS_M1      787495 non-null  int64
 15  INGRESO_BRUTO_M1     787495 non-null  float64
dtypes: float64(3), int64(9), object(4)
memory usage: 96.1+ MB
```

1.2.7 1.2.1. GRAFICAS DE SECTORES

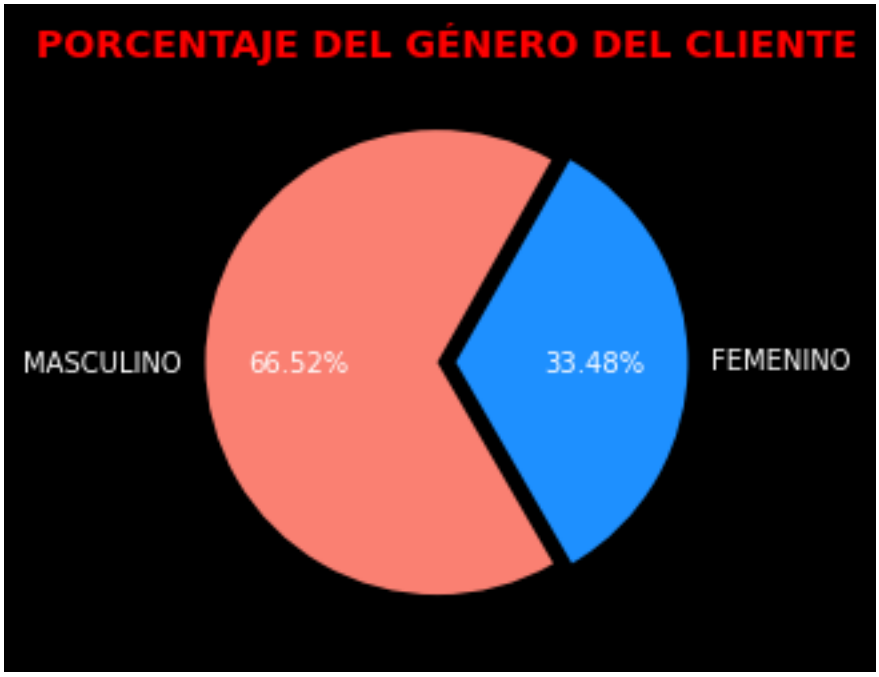
Graficamos

[55]:
```
desfase_1 = (0.04 ,0.04)
colors  = ("dodgerblue","salmon", "palevioletred",
           "steelblue", "seagreen", "plum",
           "blue", "indigo", "beige", "yellow")
```

[56]:
```
SEXO_frec=dat.groupby('SEXO').SEXO.count()
SEXO_frec
```

[56]:
```
SEXO
F    263667
M    523828
Name: SEXO, dtype: int64
```

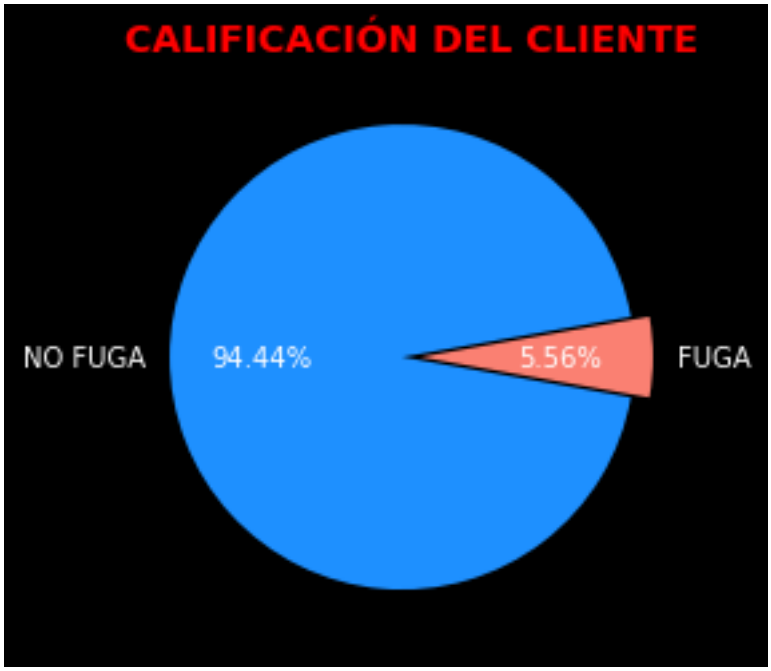
[57]:
```
nombres_genero = ["FEMENINO","MASCULINO"]
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(5,4))
    plt.
 ↪pie(SEXO_frec,labels=nombres_genero,colors=colors,startangle=-60,explode=desfase_1,autopct='%
 ↪2f%%')
plt.title("PORCENTAJE DEL GÉNERO DEL CLIENTE",weight='bold', size=14,␣
 ↪loc='center',color="red")
plt.show()
```



[58]:
```
TARGET_MODEL2_freq =dat.groupby('TARGET_MODEL2').TARGET_MODEL2.count()
TARGET_MODEL2_freq
```

[58]:
```
TARGET_MODEL2
0    743749
1     43746
Name: TARGET_MODEL2, dtype: int64
```

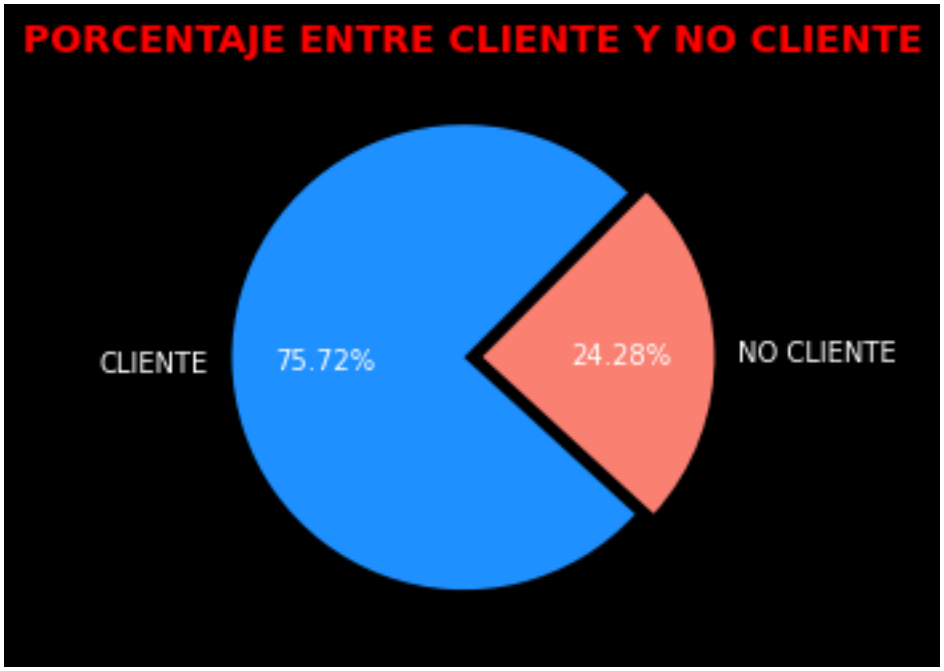
[59]:
```
nombre_cliente = ["NO FUGA","FUGA"]
TARGET_MODEL2_freq =dat.groupby('TARGET_MODEL2').TARGET_MODEL2.count()
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(5,4))
    plt.pie(TARGET_MODEL2_freq,labels=nombre_cliente␣
 ↪,colors=colors,startangle=10,explode=desfase_1,autopct='%1.2f%%')
plt.title("CALIFICACIÓN DEL CLIENTE",weight='bold', size=14,␣
 ↪loc='center',color="red")
plt.show()
```



[60]:
```
FLG_CLIENTE_freq=dat.groupby('FLG_CLIENTE').FLG_CLIENTE.count()
FLG_CLIENTE_freq
```

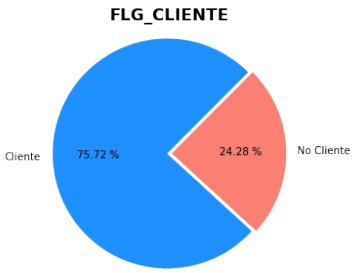
[60]:
```
FLG_CLIENTE
CLIENTE       596273
NO CLIENTE    191222
Name: FLG_CLIENTE, dtype: int64
```

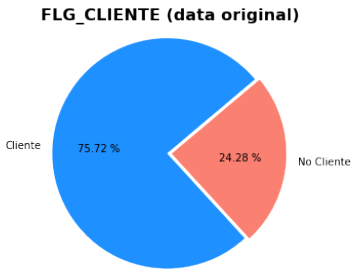
[61]:
```
nombre_cliente_1= ["CLIENTE","NO CLIENTE"]
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(5,4))
    plt.
 ↪pie(FLG_CLIENTE_freq,labels=nombre_cliente_1,colors=colors,startangle=45,explode=desfase_1,au
 ↪2f%%')
plt.title("PORCENTAJE ENTRE CLIENTE Y NO CLIENTE",weight='bold', size=14,␣
 ↪loc='center',color="red")
plt.show()
```



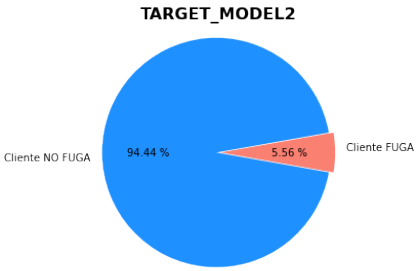
[62]:
```
desfase = (0.1 ,0.1)
```

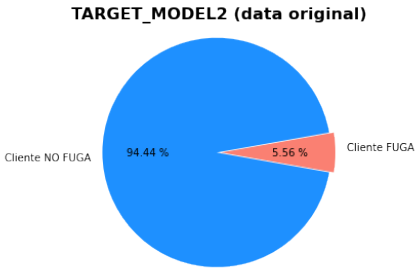
[63]:
```
fig= plt.figure(figsize=(30,10))
plt.subplot2grid((2,3),(0,0))
data_inicial = dat.groupby('FLG_CLIENTE').FLG_CLIENTE.count()
nombres = ["Cliente","No Cliente"]
plt.pie(data_inicial, labels=nombres, autopct="%0.2f %%", colors=colors,
        explode=desfase,radius=5,pctdistance=0.6,rotatelabels=0,startangle=45)
plt.title("FLG_CLIENTE",fontsize=16, weight="bold")
plt.axis("equal")
#-------------------------------------------------------------------------
plt.subplot2grid((2,3),(0,1))
data_final = data_resultante.groupby('FLG_CLIENTE').FLG_CLIENTE.count()
nombres = ["Cliente","No Cliente"]
plt.pie(data_final , labels=nombres, autopct="%0.2f %%", colors=colors,
        explode=desfase,radius=5,pctdistance=0.6,rotatelabels=0,startangle=40)
plt.title("FLG_CLIENTE (data original)",fontsize=16, weight="bold")
plt.axis("equal")
plt.show()
```





[64]:
```
fig= plt.figure(figsize=(30,10))
plt.subplot2grid((2,3),(0,0))
data_inicial=dat.groupby('TARGET_MODEL2').TARGET_MODEL2.count()
nombres = ["Cliente NO FUGA","Cliente FUGA"]
plt.pie(data_inicial, labels=nombres, autopct="%0.2f %%", colors=colors,
        explode=desfase,radius=5,pctdistance=0.6,rotatelabels=50,startangle=10)
plt.title("TARGET_MODEL2",fontsize=16, weight="bold")
plt.axis("equal")
#-------------------------------------------------------------------------
plt.subplot2grid((2,3),(0,1))
target_graf =dat.groupby('TARGET_MODEL2').TARGET_MODEL2.count()
nombres = ["Cliente NO FUGA","Cliente FUGA"]
plt.pie(target_graf, labels=nombres, autopct="%0.2f %%", colors=colors,
        explode=desfase,radius=5,pctdistance=0.6,rotatelabels=50,startangle=10)
plt.title("TARGET_MODEL2 (data original)",fontsize=16, weight="bold")
plt.axis("equal")
plt.show()
```

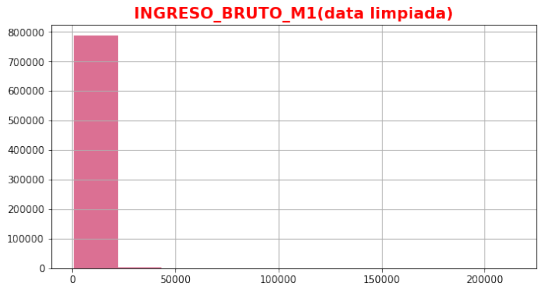


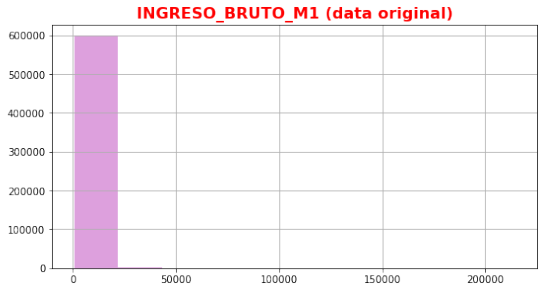


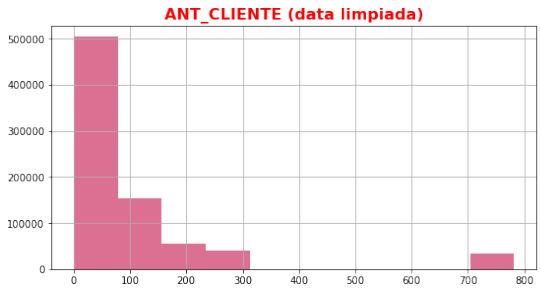
1.2.8 1.2.2. GRÁFICA DE HISTOGRAMA

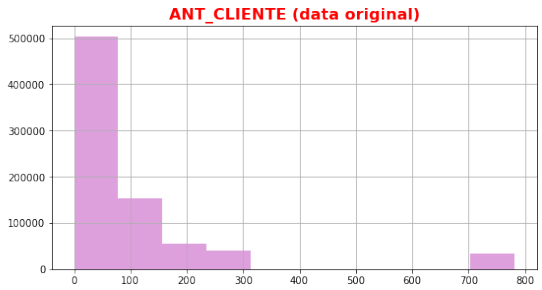
[65]:
```
fig= plt.figure(figsize=(30,10))
plt.subplot2grid((2,3),(0,0))
dat['INGRESO_BRUTO_M1'].hist(color=colors[2]).set_title("INGRESO_BRUTO_M1(data␣
 ↪limpiada)",
```

```
                                                        fontsize=16,␣
 ↪weight="bold",color="red")
#-------------------------------------------------------------------------------
plt.subplot2grid((2,3),(0,1))
data_resultante['INGRESO_BRUTO_M1'].hist(color=colors[5]).
 ↪set_title("INGRESO_BRUTO_M1 (data original)",
                                                              fontsize=16,␣
 ↪weight="bold",color="red")
#-------------------------------------------------------------------------------
fig= plt.figure(figsize=(30,10))
plt.subplot2grid((2,3),(1,0))
dat['ANT_CLIENTE'].hist(color=colors[2]).set_title("ANT_CLIENTE (data limpiada)",
                                                        fontsize=16,␣
 ↪weight="bold",color="red")
#-------------------------------------------------------------------------------
plt.subplot2grid((2,3),(1,1))
data_resultante['ANT_CLIENTE'].hist(color=colors[5]).set_title("ANT_CLIENTE␣
 ↪(data original)",
                                                              fontsize=16,␣
 ↪weight="bold",color="red")
plt.show()
```









1.2.9 1.2.3. GRAFICAS DE BARRAS

VEAMOS LA DISTRIBUSCION DE LOS DEPARTAMENTOS DE LOS CLIENTES

[66]:
```
departamento_freq=pd.value_counts(dat.DEPARTAMENTO)
departamento_freq
```

[66]:
```
LIMA             498174
CALLAO            43021
LA LIBERTAD       33483
PIURA             28401
ICA               23374
LAMBAYEQUE        21906
AREQUIPA          21101
ANCASH            17426
CUSCO             14115
JUNIN             13040
PUNO              11041
CAJAMARCA         10779
LORETO            10676
MOQUEGUA           5548
SAN MARTIN         5410
TACNA              4888
APURIMAC           4790
PASCO              4259
AYACUCHO           4038
UCAYALI            3411
HUANUCO            2809
MADRE DE DIOS      2452
TUMBES             2108
HUANCAVELICA        705
AMAZONAS            540
Name: DEPARTAMENTO, dtype: int64
```

[67]:
```
departamento_freq/sum(departamento_freq)*100
```

[67]:
```
LIMA             63.260592
CALLAO            5.463019
LA LIBERTAD       4.251837
PIURA             3.606499
ICA               2.968146
LAMBAYEQUE        2.781732
AREQUIPA          2.679509
ANCASH            2.212839
CUSCO             1.792392
JUNIN             1.655884
PUNO              1.402041
CAJAMARCA         1.368771
```

```
LORETO            1.355691
MOQUEGUA          0.704512
SAN MARTIN        0.686988
TACNA             0.620702
APURIMAC          0.608258
PASCO             0.540829
AYACUCHO          0.512765
UCAYALI           0.433146
HUANUCO           0.356701
MADRE DE DIOS     0.311367
TUMBES            0.267684
HUANCAVELICA      0.089524
AMAZONAS          0.068572
Name: DEPARTAMENTO, dtype: float64
```

[68]:
```
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(15,7))
    plot = departamento_freq.plot(kind='barh',rot=0,color=colors)
plot
plt.title('FRECUENCIAS DE LOS DEPARTAMENTOS DE LOS␣
 ↪CLIENTES',weight='bold',color='red')
plt.text(498174,0,"63.26%",weight="bold",color="yellow")
plt.text(43021,1,"5.46%",weight="bold",color="yellow")
plt.text(33483,2,"4.25%",weight="bold",color="yellow")
plt.text(28401,3,"3.61%",weight="bold",color="yellow")
plt.text(23374,4,"2.97%",weight="bold",color="yellow")
plt.text(21906,5,"2.78%",weight="bold",color="yellow")
plt.text(21101,6,"2.68%",weight="bold",color="yellow")
plt.text(17426,7,"2.21%",weight="bold",color="yellow")
plt.text(14115,8,"1.79%",weight="bold",color="yellow")
plt.text(13040,9,"1.66%",weight="bold",color="yellow")
plt.text(11041,10,"1.40%",weight="bold",color="yellow")
plt.text(10779,11,"1.37%",weight="bold",color="yellow")
plt.text(10676,12,"1.36%",weight="bold",color="yellow")
plt.text(5548,13,"0.70%",weight="bold",color="yellow")
plt.text(5410,14,"0.69%",weight="bold",color="yellow")
plt.text(4888,15,"0.62%",weight="bold",color="yellow")
plt.text(4790,16,"0.61%",weight="bold",color="yellow")
plt.text(4259,17,"0.54%",weight="bold",color="yellow")
plt.text(4038,18,"0.51%",weight="bold",color="yellow")
plt.text(3411,19,"0.43%",weight="bold",color="yellow")
plt.text(2809,20,"0.36%",weight="bold",color="yellow")
plt.text(2452,21,"0.31%",weight="bold",color="yellow")
plt.text(2108,22,"0.27%",weight="bold",color="yellow")
plt.text(705,23,"0.09%",weight="bold",color="yellow")
plt.text(540,24,"0.07%",weight="bold",color="yellow")
plt.show()
```

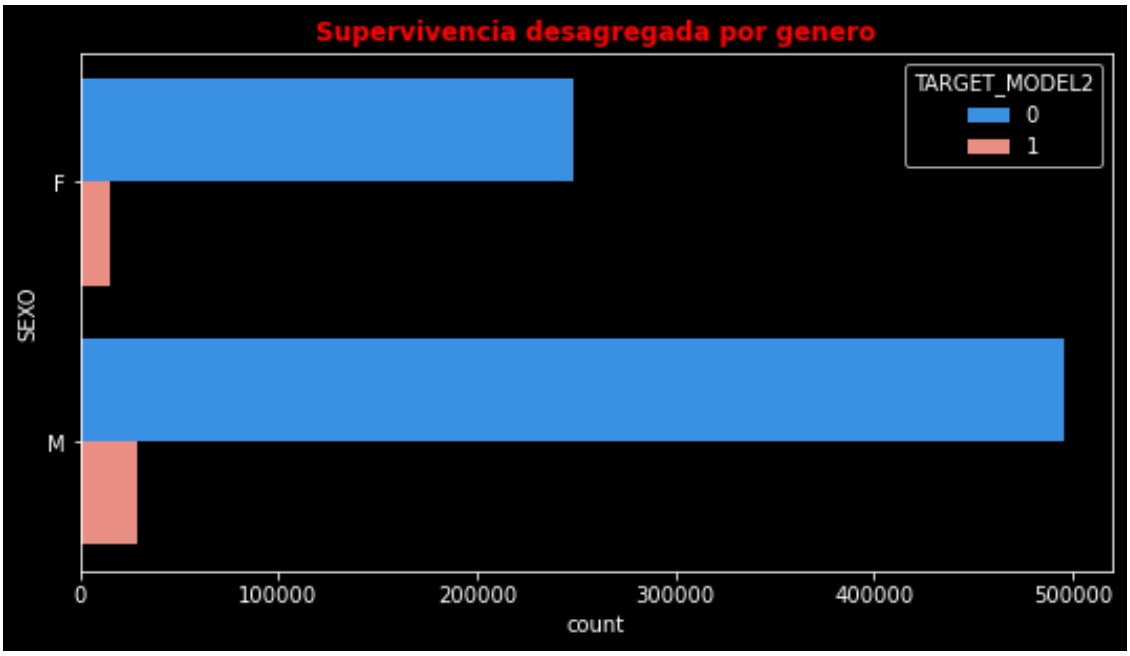
VEAMOS LA DISTRIBUCIÓN DE LA EDAD DE LOS CLIENTES PERO AGRUPADOS

VAMOS AGRUPAR DE LA SIGUIENTE MANERA:

1. MENORES DE EDAD: (0, 18]
2. MAYORES DE EDAD: (18, 65]
3. ANCIANOS: (65, 114]

SUPERVIVENCIA DESAGREGADA POR GÉNERO

[69]:
```
with plt.style.context('dark_background'):
    fig= plt.figure(figsize=(30,10))
    plt.subplot2grid((2,3),(0,0))
    sns.countplot(y='SEXO', hue='TARGET_MODEL2', data=dat,palette=colors)
plt.title('Supervivencia desagregada por genero',weight='bold',color='red')
plt.show()
```



VEAMOS LA DISTRIBUCIÓN DE LA EDAD DE LOS CLIENTES PERO AGRUPADOS

VAMOS AGRUPAR DE LA SIGUIENTE MANERA:

1. MENORES DE EDAD: (0, 18]
2. MAYORES DE EDAD: (18, 65]
3. ANCIANOS: (65, 114]

[70]:
```
dat["EDAD_CAT"]=pd.cut(dat.EDAD,bins=[dat.EDAD.min(),18,65,dat.EDAD.max()])
dat.head()
```

[70]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	LIMA	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC	

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1	EDAD_CAT
0	0.0	224.0	0	2184.9	(18, 65]

```
1              0.0        123.0                0           4718.0   (18, 65]
2              0.0        264.0                0           2184.9  (65, 114]
3              0.0        263.0                0            936.0   (18, 65]
4              0.0        263.0                0           5844.0   (18, 65]
```

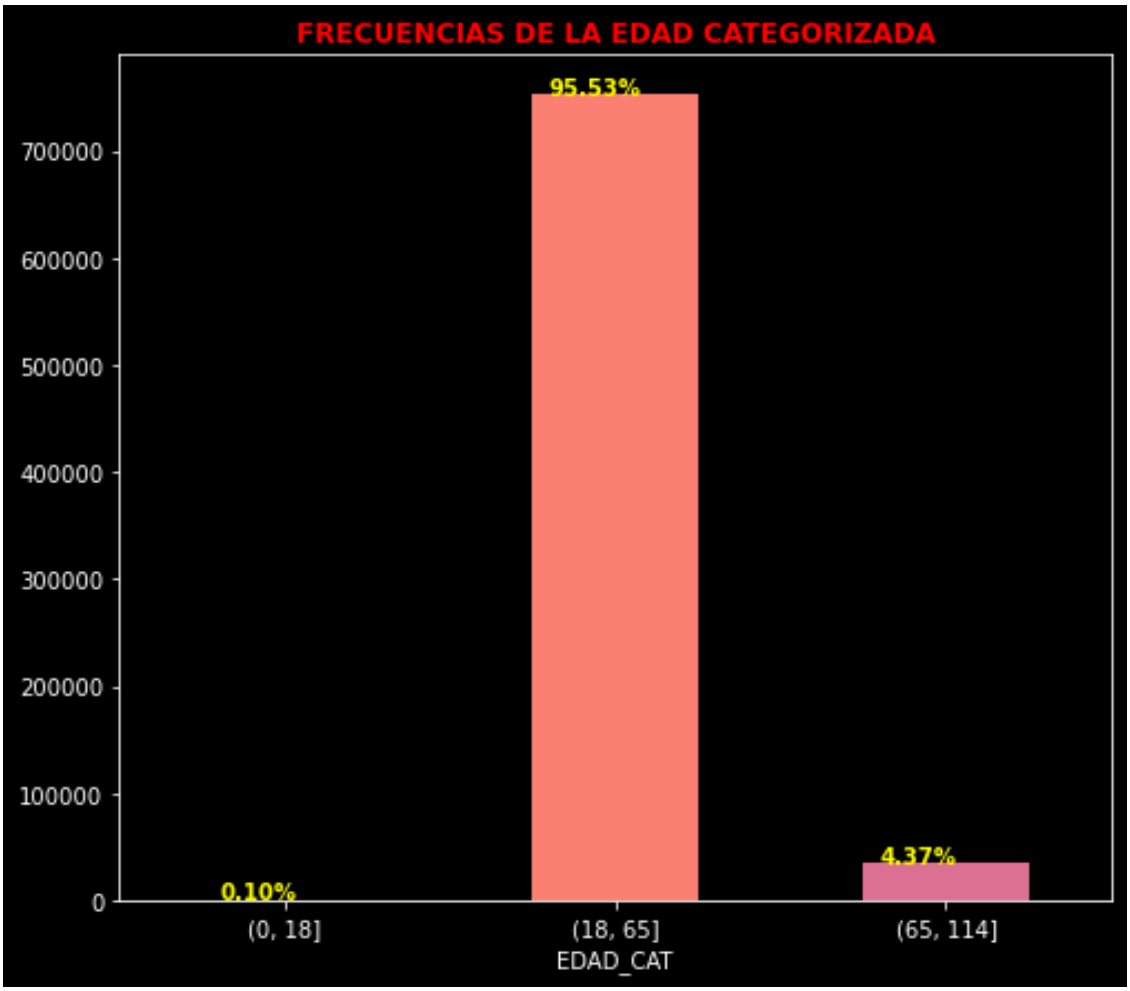
[71]:
```
edad=dat.groupby("EDAD_CAT").size()
edad
```

[71]:
```
EDAD_CAT
(0, 18]          798
(18, 65]      752244
(65, 114]      34412
dtype: int64
```

[72]:
```
e=round((edad/sum(edad))*100,2)
print("El % de datos es: " )
print("-------------", e)
```

```
El % de datos es:
------------- EDAD_CAT
(0, 18]        0.10
(18, 65]      95.53
(65, 114]      4.37
dtype: float64
```

[73]:
```
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(8,7))
    plot =edad.plot(kind='bar',
                          rot=0,color=colors)
plot
plt.title('FRECUENCIAS DE LA EDAD CATEGORIZADA',weight='bold',color='red')
plt.text(-0.2,798,"0.10%",weight="bold",color="yellow")
plt.text(0.8,752244,"95.53%",weight="bold",color="yellow")
plt.text(1.8,34412,"4.37%",weight="bold",color="yellow")
plt.show()
```



VEAMOS LA DISTRIBUCIÓN DEL INGRESO BRUTO DE LOS CLIENTES PERO AGRUPADOS

LO AGRUPAMOS DE LA SIGUIENTE MANERA:

1. CLASE POBRE: (681.0, 1200.0]
2. CLASE MEDIA BAJA: (1200.0, 5000.0]
3. CLASE MEDIA ALTA: (5000.0, 10000.0]
4. CLASE ALTA: (10000.0, 214284.0]

[74]:
```
dat_1=pd.read_csv("TRAIN_FUGA_COMPLETO.csv",sep=",", encoding="ISO-8859-1")
dat_1['INGRESO_BRUTO_M1_CAT']=pd.cut(dat_1.INGRESO_BRUTO_M1,bins=[dat_1.
 ↪INGRESO_BRUTO_M1.min(),1200,5000,10000,dat_1.INGRESO_BRUTO_M1.max()])
dat_1.head()
```

[74]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	

```
2  201411              0    81    M         LIMA     CLIENTE        6
3  201411              0    42    M        PIURA     CLIENTE        2
4  201411              0    52    M     MOQUEGUA     CLIENTE      1BC

   FLG_ADEL_SUELDO_M1  FREC_AGENTE  FREC_KIOSKO  FREC_BPI_TD  FREC_MON_TD  \
0                   0            0            0            0            0
1                   0            0            0            0            0
2                   0            0            0            0            0
3                   0            0            0            0            0
4                   0            0            0            0            0

   PROM_CTD_TRX_6M  ANT_CLIENTE  CTD_RECLAMOS_M1  INGRESO_BRUTO_M1  \
0              0.0        224.0                0            2184.9
1              0.0        123.0                0            4718.0
2              0.0        264.0                0            2184.9
3              0.0        263.0                0             936.0
4              0.0        263.0                0            5844.0

  INGRESO_BRUTO_M1_CAT
0     (1200.0, 5000.0]
1     (1200.0, 5000.0]
2     (1200.0, 5000.0]
3      (681.0, 1200.0]
4    (5000.0, 10000.0]
```

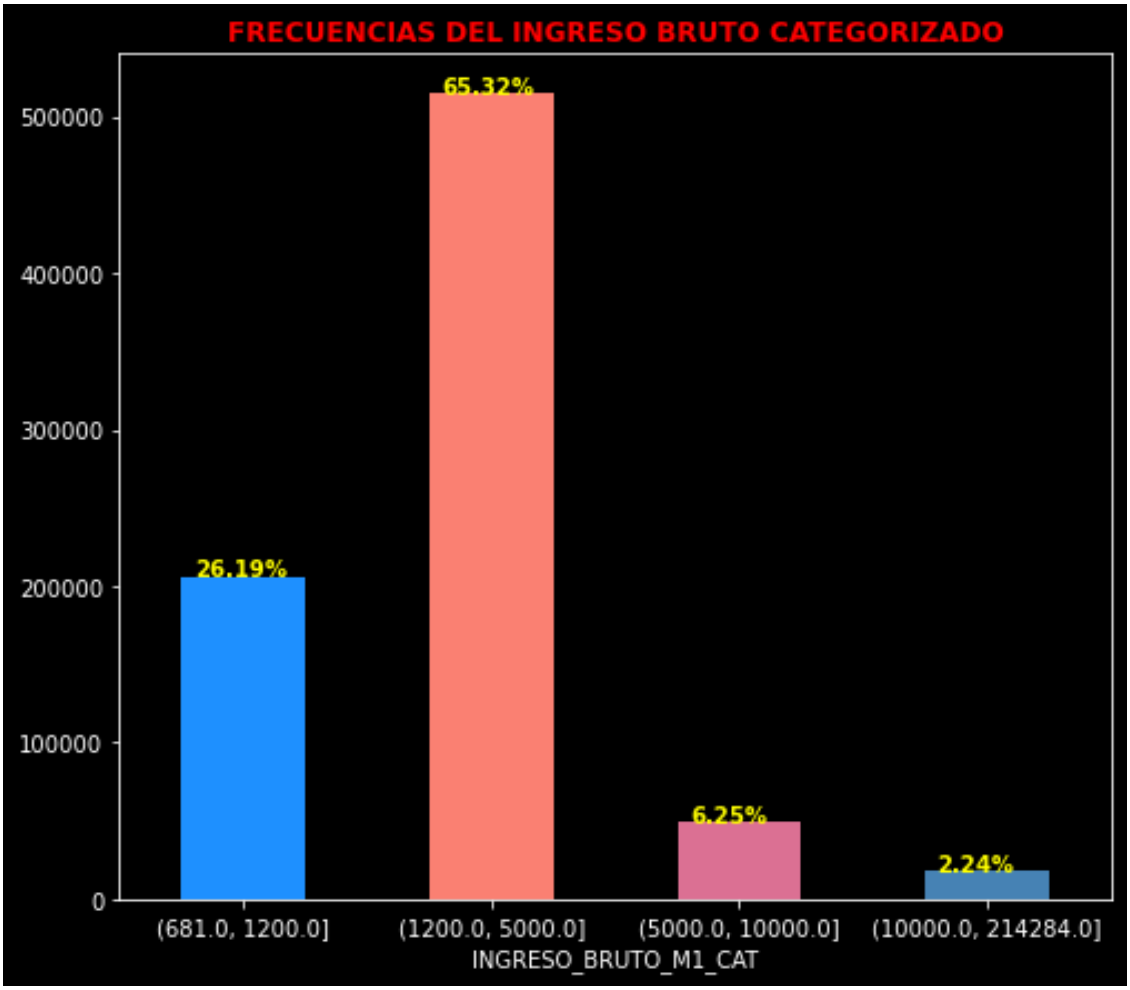
[75]:
```
ingreso=dat_1.groupby('INGRESO_BRUTO_M1_CAT').size()
ingreso
```

[75]:
```
INGRESO_BRUTO_M1_CAT
(681.0, 1200.0]        206224
(1200.0, 5000.0]       514384
(5000.0, 10000.0]       49208
(10000.0, 214284.0]     17671
dtype: int64
```

[76]:
```
p=round((ingreso/sum(ingreso))*100,2)
print("El % de datos es: " )
print("--------------------", p)
```

```
El % de datos es:
-------------------- INGRESO_BRUTO_M1_CAT
(681.0, 1200.0]        26.19
(1200.0, 5000.0]       65.32
(5000.0, 10000.0]       6.25
(10000.0, 214284.0]     2.24
dtype: float64
```

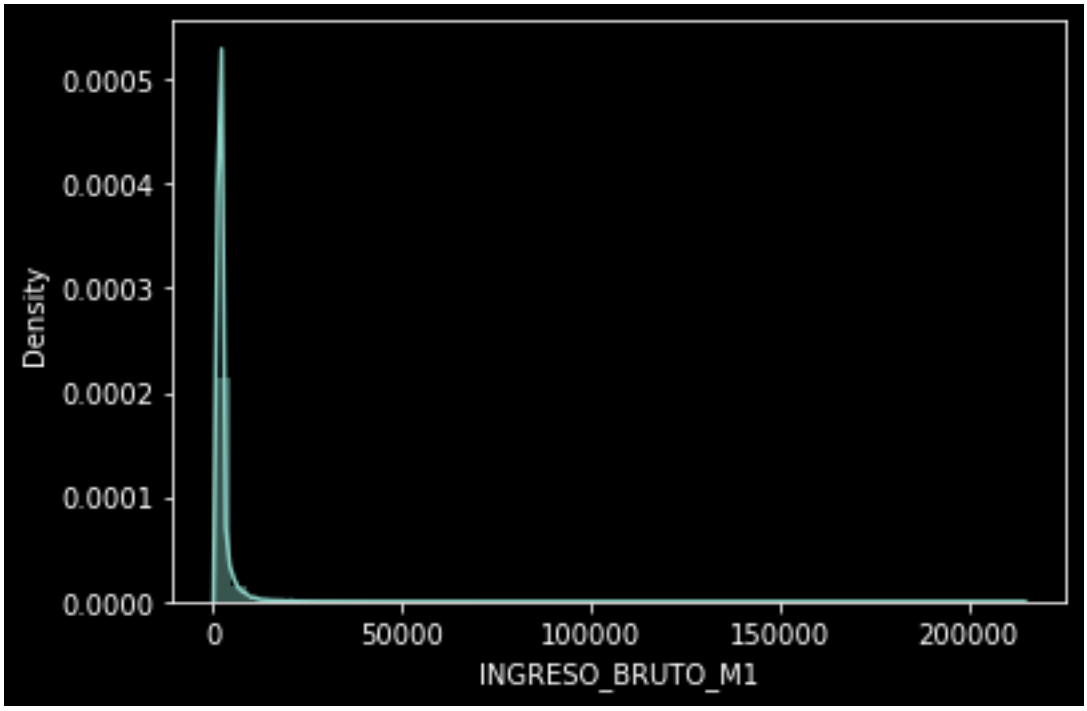
[77]:
```
with plt.style.context('dark_background'):
    fig = plt.figure(figsize=(8,7))
    plot =ingreso.plot(kind='bar',
                       rot=0,color=colors)
plot
plt.title('FRECUENCIAS DEL INGRESO BRUTO CATEGORIZADO',weight='bold',color='red')
plt.text(-0.2,206224,"26.19%",weight="bold",color="yellow")
plt.text(0.8,514384,"65.32%",weight="bold",color="yellow")
plt.text(1.8,49208,"6.25%",weight="bold",color="yellow")
plt.text(2.8,17671,"2.24%",weight="bold",color="yellow")
plt.show()
```



VEAMOS LA DESINDAD DEL INGRESO BRUTO

[78]:
```
warnings.filterwarnings("ignore")
dat_2=pd.read_csv("TRAIN_FUGA_COMPLETO.csv",sep=",", encoding="ISO-8859-1")
with plt.style.context('dark_background'):
```

```
    ax1=sns.distplot(dat_2['INGRESO_BRUTO_M1'])
plt.show()
```



DEBIDO A LA CANTIDAD DE LOS DATOS DE NUESTRO DATAFRAME NO SE PUEDE APRECIAR BIEN LA SIMETRÍA O ASIMETRÍA DE LA VARIABLE INGRESO BRUTO , PARA ELLO VAMOS A TOMAR UNA MUESTRA DE LOS DATOS OBTENIDOS PARA PODER REALIZAR UN MEJOR ANÁLISIS.

[79]:
```
dat_2=pd.read_csv("TRAIN_FUGA_COMPLETO.csv",sep=",", encoding="ISO-8859-1")
medio=(dat_2['INGRESO_BRUTO_M1']>2500)&(dat_2['INGRESO_BRUTO_M1']<18000)
medio
```

[79]:
```
0         False
1          True
2         False
3         False
4          True
          ...
787490    False
787491     True
787492     True
787493    False
787494     True
Name: INGRESO_BRUTO_M1, Length: 787495, dtype: bool
```

[80]:
```
nuevo=dat_2[medio]
nuevo.head(10)
```

[80]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO
1	201411	0	54	M	LORETO	CLIENTE	1BC
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC
8	201411	0	65	M	CALLAO	NO CLIENTE	2
37	201411	0	57	F	LIMA	NO CLIENTE	1BC
45	201411	0	42	M	LIMA	CLIENTE	3
83	201411	0	52	F	LIMA	NO CLIENTE	1BC
84	201411	0	34	M	LIMA	CLIENTE	1BC
106	201411	0	40	M	ANCASH	CLIENTE	4
114	201411	0	29	M	LIMA	CLIENTE	1A

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD
1	0	0	0	0	0
4	0	0	0	0	0
6	0	0	0	0	6
8	0	0	0	0	0
37	0	0	2	0	0
45	0	0	3	0	0
83	0	1	6	0	0
84	0	0	0	0	0
106	0	0	0	6	2
114	0	0	2	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
1	0.000000	123.0	0	4718.0
4	0.000000	263.0	0	5844.0
6	0.000000	85.0	0	4232.0
8	3.333333	778.0	0	3081.5
37	0.000000	778.0	0	3962.0
45	0.333333	76.0	0	2777.0
83	0.000000	778.0	0	3983.0
84	0.000000	129.0	0	4179.0
106	0.000000	93.0	0	11193.0
114	0.000000	778.0	0	3578.0

[81]:
```
nuevo=dat_2[medio]
nuevo.shape
```

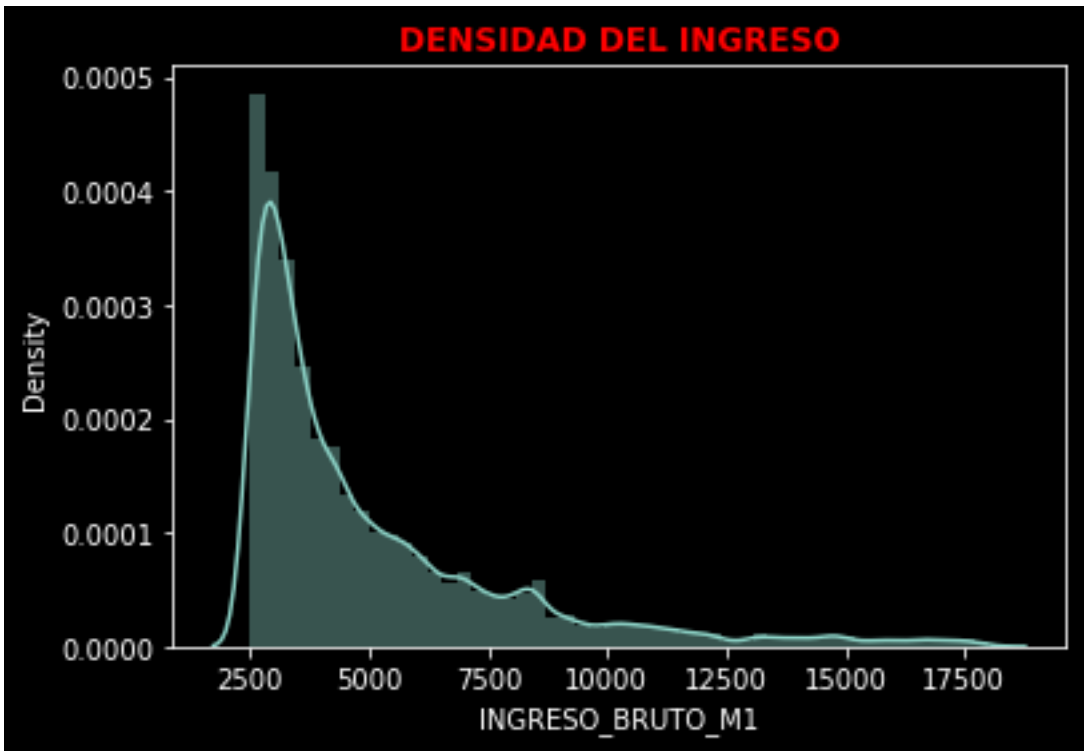
[81]:
```
(180973, 16)
```

[82]:
```
mediana=nuevo["INGRESO_BRUTO_M1"].median()
media=nuevo["INGRESO_BRUTO_M1"].mean()
moda=nuevo["INGRESO_BRUTO_M1"].mode()
```

```
print("La mediana:",mediana)
print("La media:",media)
print("La moda:",moda)
minimo=nuevo["INGRESO_BRUTO_M1"].min()
print("El valor mínimo:",minimo)
maximo=nuevo["INGRESO_BRUTO_M1"].max()
print("El valor máximo",maximo)
```

```
La mediana: 3903.0
La media: 5046.991752913896
La moda: 0    2522.7
dtype: float64
El valor mínimo: 2501.0
El valor máximo 17997.0
```

[83]:
```
warnings.filterwarnings("ignore")
with plt.style.context('dark_background'):
    ax2=sns.distplot(nuevo['INGRESO_BRUTO_M1'])
plt.title("DENSIDAD DEL INGRESO",weight='bold',color='red')
plt.show()
```

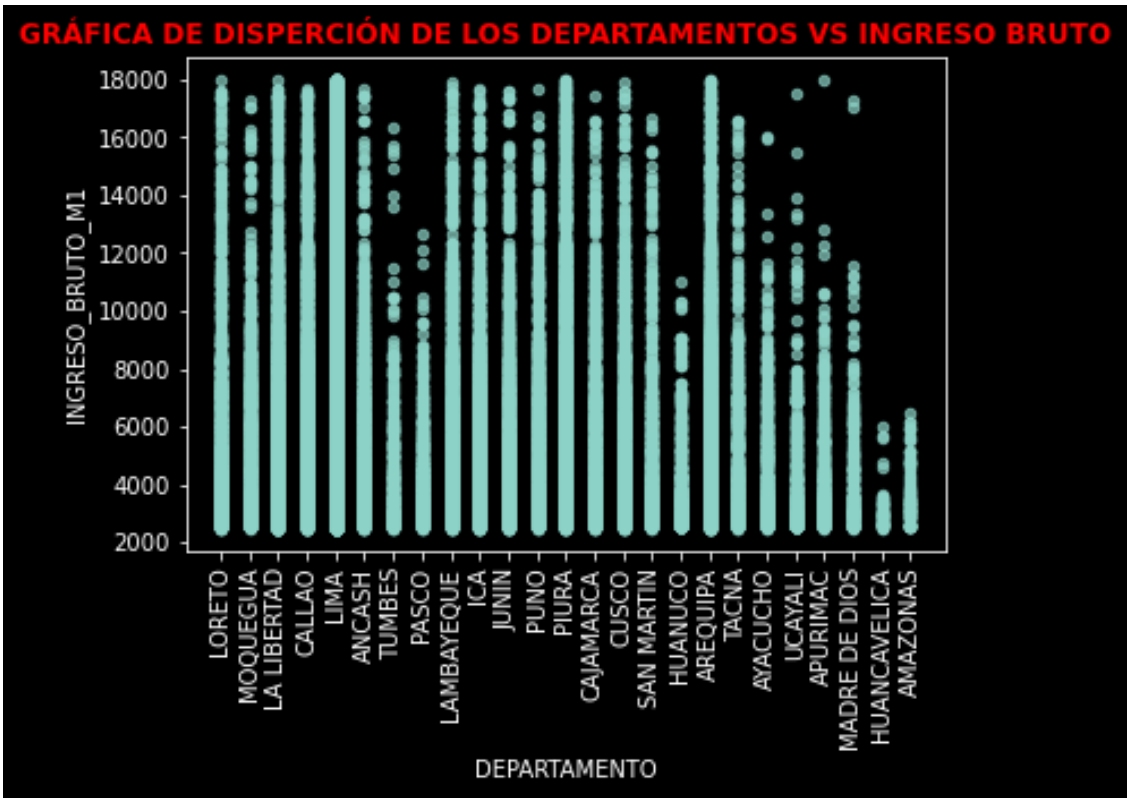


PODEMOS VER QUE TIENE UNA ASIMETRíA POSITIVA DE LA VARIABLE INGRESO BRUTO

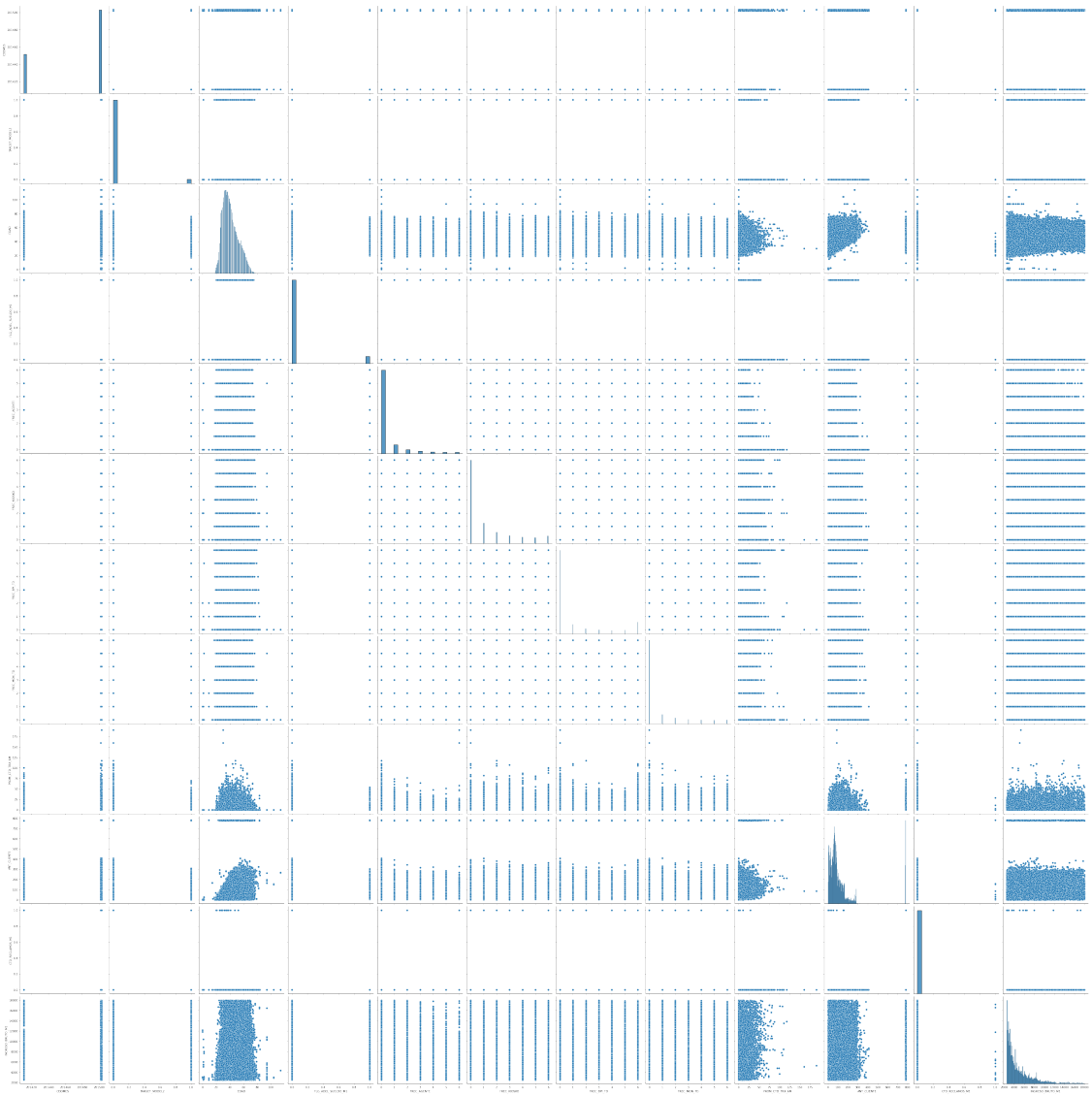
1.2.10 1.2.4. GRÁFICA DE DISPERCIÓN

VEAMOS SI EXISTE UNA CORRELACIÓN ENTRE LAS VARIABLES

[84]:
```
with plt.style.context('dark_background'):
    nuevo.plot.scatter(x="DEPARTAMENTO",
                       y="INGRESO_BRUTO_M1",
                       alpha=0.7,rot=90) #sombreado de los puntos, menor valor es más␣
 ↪claro.
plt.title("GRÁFICA DE DISPERCIÓN DE LOS DEPARTAMENTOS VS INGRESO␣
 ↪BRUTO",weight='bold',color='red')
plt.show()
```



[85]:
```
columnas=nuevo.columns
sns.pairplot(nuevo[columnas], #data y sus columnas seleccionadas
             height = 4.5) #tamaño de la gráfica
plt.show()
```

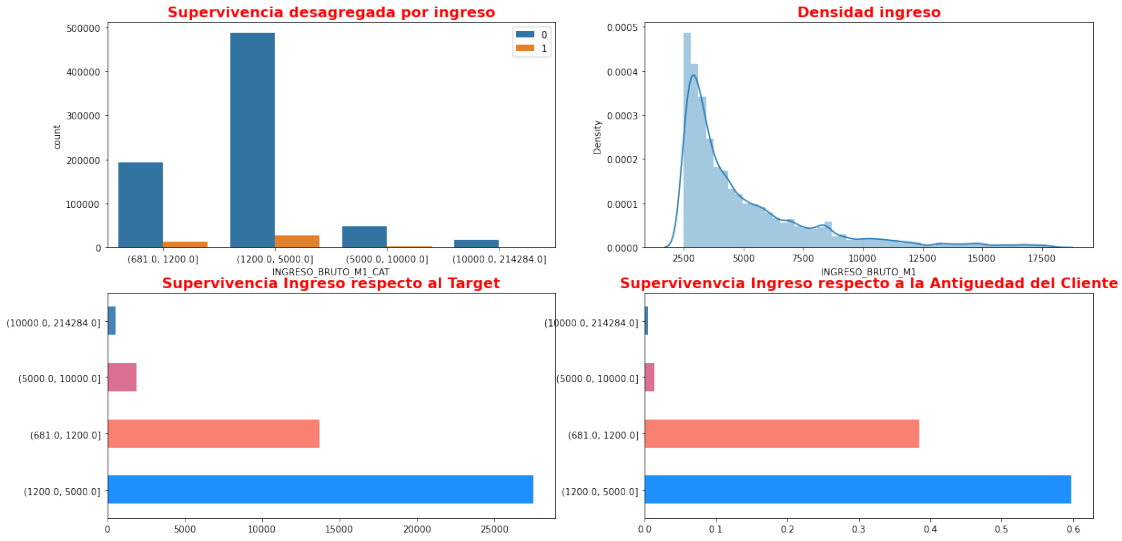


PODEMOS OBSERVAR QUE NO EXISTE UNA CORRELACION ENTRE LOS DATOS.

1.2.11 GRÁFICOS INFORMATIVOS

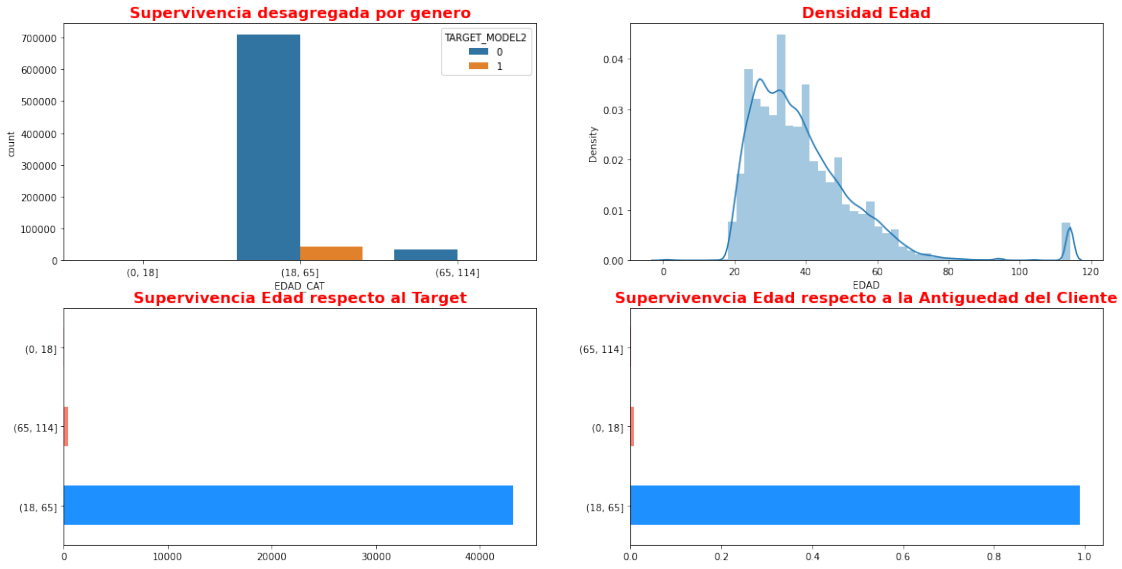
[86]:
```
fig= plt.figure(figsize=(30,10))
#with plt.style.context('dark_background'):
plt.subplot2grid((2,3),(0,0))
sns.countplot(x='INGRESO_BRUTO_M1_CAT', hue='TARGET_MODEL2', data=dat_1)
plt.title('Supervivencia desagregada por ingreso',fontsize=16,␣
 ↪weight="bold",color='red')
plt.legend(loc="upper right")

plt.subplot2grid((2,3),(0,1))
```

```
ax=sns.distplot(nuevo['INGRESO_BRUTO_M1'])
plt.title('Densidad ingreso',fontsize=16, weight="bold",color='red')

plt.subplot2grid((2,3),(1,0))
dat_1.INGRESO_BRUTO_M1_CAT[dat_2.TARGET_MODEL2 == 1].value_counts().
 ↪plot(kind="barh", color = colors)
plt.title("Supervivencia Ingreso respecto al Target",fontsize=16,␣
 ↪weight="bold",color='red')

plt.subplot2grid((2,3),(1,1))
dat_1.INGRESO_BRUTO_M1_CAT[dat_2.ANT_CLIENTE == 1].value_counts(normalize=True).
 ↪plot(kind="barh",color = colors)
plt.title("Supervivenvcia Ingreso respecto a la Antiguedad del␣
 ↪Cliente",fontsize=16, weight="bold",color='red')

plt.show()
```



[87]:
```
fig= plt.figure(figsize=(30,10))
plt.subplot2grid((2,3),(0,0))
sns.countplot(x='EDAD_CAT', hue='TARGET_MODEL2', data=dat)
plt.title('Supervivencia desagregada por genero',fontsize=16,␣
 ↪weight="bold",color='red')

plt.subplot2grid((2,3),(0,1))
ax=sns.distplot(dat_2['EDAD'])
plt.title('Densidad Edad',fontsize=16, weight="bold",color='red')

plt.subplot2grid((2,3),(1,0))
```

```
dat.EDAD_CAT[dat_2.TARGET_MODEL2 == 1].value_counts().plot(kind="barh", color =␣
 ↪colors)
plt.title("Supervivencia Edad respecto al Target",fontsize=16,␣
 ↪weight="bold",color='red')

plt.subplot2grid((2,3),(1,1))
dat.EDAD_CAT[dat_2.ANT_CLIENTE == 1].value_counts(normalize=True).
 ↪plot(kind="barh",color = colors)
plt.title("Supervivenvcia Edad respecto a la Antiguedad del␣
 ↪Cliente",fontsize=16, weight="bold",color='red')

plt.show()
```

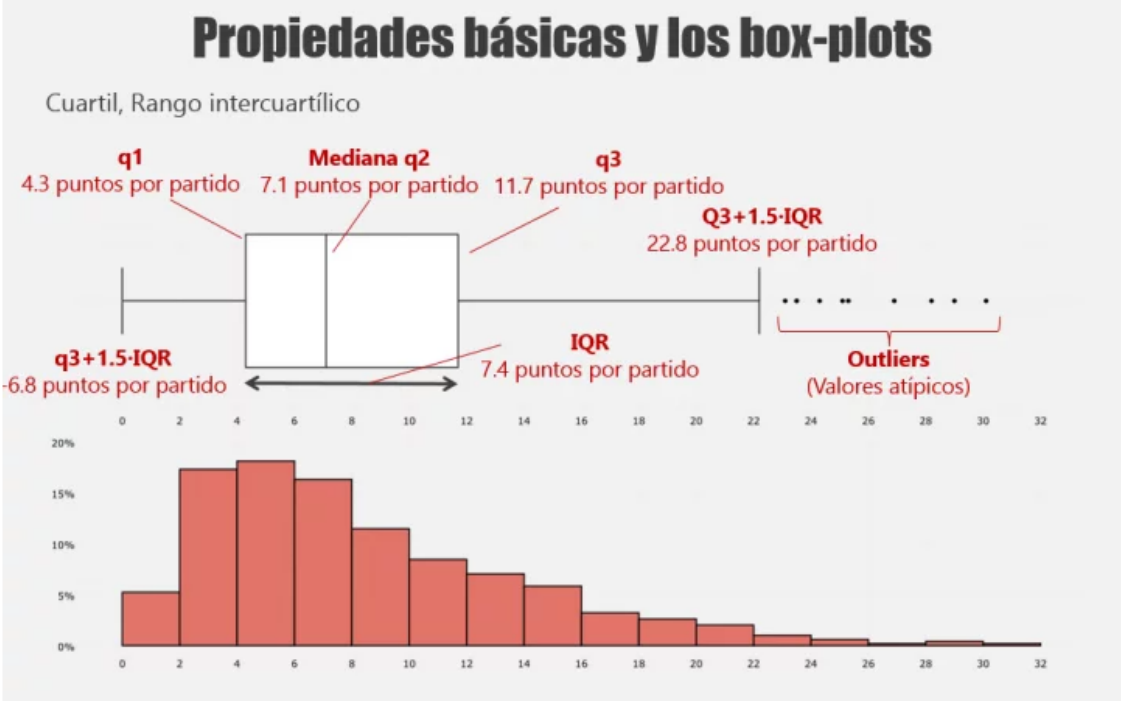


[ ]:

1.3 2. Realizar un análisis exploratorio sobre presencia de outliers.

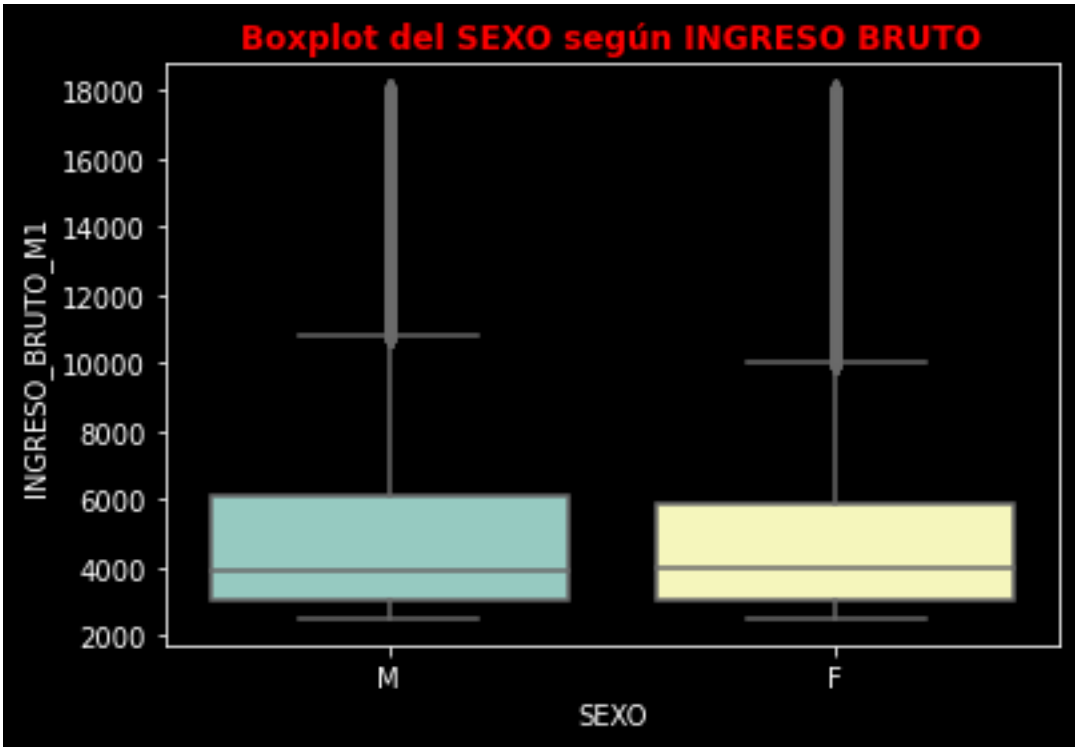
[88]:
```
from IPython.display import Image
%matplotlib inline
Image(filename='E:\PYTHOM\MODULO 1\EXAMEN FINAL-MODULO1/interpretacion_boxplot.
 ↪png', width=600)
```

[88]:



Graficando antes de eliminar los Outliers BOXPLOT DEL SEXO SEGÚN EL INGRESO BRUTO

[89]:
```
with plt.style.context('dark_background'):
    sns.boxplot(x=nuevo['SEXO'], #Será la variable categorizadora o separadora.
            y=nuevo['INGRESO_BRUTO_M1']) #La variable cuantitativa de rpta.
plt.title("Boxplot del SEXO según INGRESO BRUTO",weight='bold',color='red')
plt.show()
```

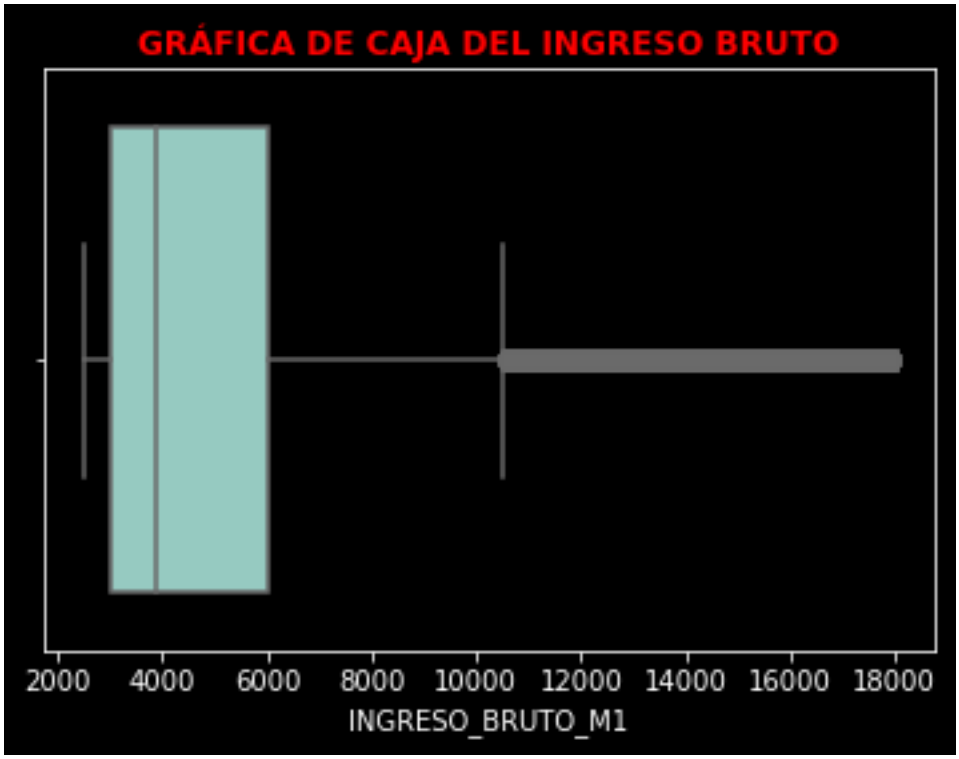


[90]:
```
mediana=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.median()
Q1=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.quantile(0.25)
Q3=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.quantile(0.75)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("INFORMACIÓN DEL GÉNERO MASCULINO SEGÚN EL INGRESO BRUTO\n")
print("Primer cuartil:",Q1)
print("Tercer cuartil:",Q3)
print("La mediana:",mediana)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
print("-------------------------------------------------------------")
mediana=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.median()
Q1=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.quantile(0.25)
Q3=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.quantile(0.75)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("INFORMACIÓN DEL GÉNERO FEMENINO SEGÚN EL INGRESO BRUTO\n")
print("Primer cuartil:",Q1)
print("Tercer cuartil:",Q3)
print("La mediana:",mediana)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL GÉNERO MASCULINO SEGÚN EL INGRESO BRUTO

Primer cuartil: 3021.0
Tercer cuartil: 6131.0
La mediana: 3886.0
Bigote superior de mi boxplot es:  10796.0
Bigote inferior de mi boxplot es:  1466.0
------------------------------------------------------------
INFORMACIÓN DEL GÉNERO FEMENINO SEGÚN EL INGRESO BRUTO

Primer cuartil: 3046.0
Tercer cuartil: 5839.0
La mediana: 3949.0
Bigote superior de mi boxplot es:  10028.5
Bigote inferior de mi boxplot es:  1649.5
```

BOXPLOT DEL INGRESO BRUTO

[91]:
```
with plt.style.context('dark_background'):
    sns.boxplot(nuevo.INGRESO_BRUTO_M1)
plt.title("GRÁFICA DE CAJA DEL INGRESO BRUTO",weight='bold',color='red')
plt.show()
```

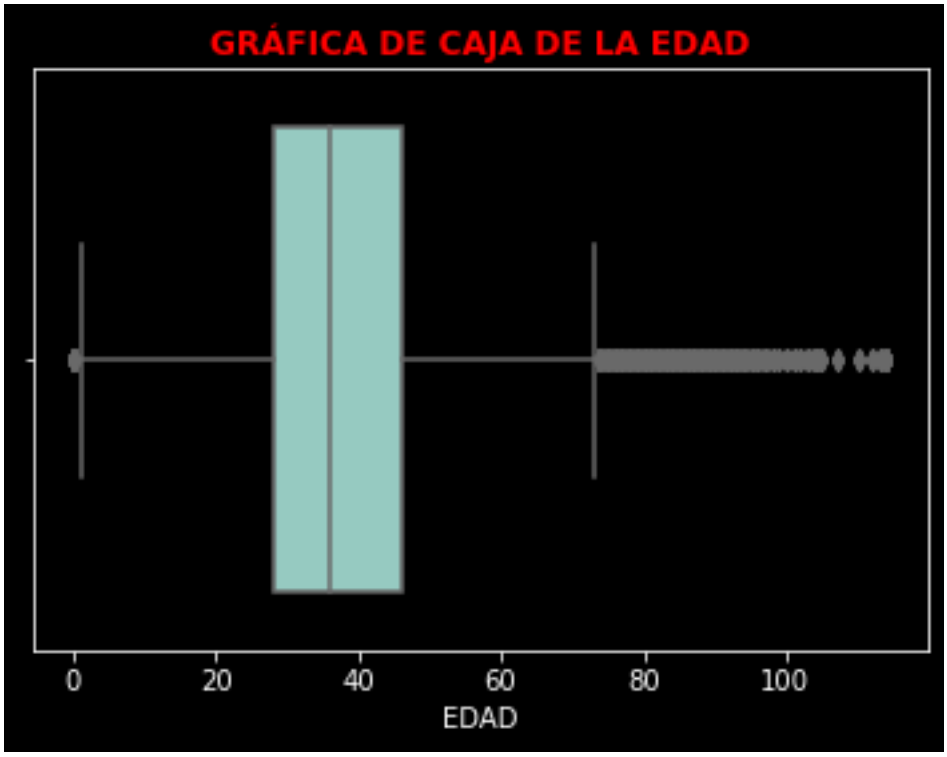


[92]:
```
print("INFORMACIÓN DEL BOXPLOT DEL INGRESO BRUTO\n")
Q1=nuevo.INGRESO_BRUTO_M1.quantile(0.25)
print("Primer cuartil:",Q1)
Q3=nuevo.INGRESO_BRUTO_M1.quantile(0.75)
print("Tercer cuartil:",Q3)
Rango_inter_cuart=Q3-Q1
print("El rango intercuartil:",Rango_inter_cuart)
mediana=nuevo.INGRESO_BRUTO_M1.median()
print("La mediana:",mediana)
minimo=nuevo.INGRESO_BRUTO_M1.min()
print("El valor mínimo:",minimo)
maximo=nuevo.INGRESO_BRUTO_M1.max()
print("El valor máximo",maximo)
N=Q3+1.5*(Q3-Q1)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL BOXPLOT DEL INGRESO BRUTO

Primer cuartil: 3028.0
Tercer cuartil: 6022.0
El rango intercuartil: 2994.0
La mediana: 3903.0
El valor mínimo: 2501.0
El valor máximo 17997.0
Bigote superior de mi boxplot es:  10513.0
Bigote inferior de mi boxplot es:  1531.0
```

BOXPLOT DE LA EDAD

[93]:
```
with plt.style.context('dark_background'):
    sns.boxplot(dat_2.EDAD)
plt.title("GRÁFICA DE CAJA DE LA EDAD",weight='bold',color='red')
plt.show()
```



[94]:
```
print("INFORMACIÓN DEL BOXPLOT DE LA EDAD\n")
Q1=nuevo.EDAD.quantile(0.25)
print("Primer cuartil:",Q1)
Q3=nuevo.EDAD.quantile(0.75)
print("Tercer cuartil:",Q3)
Rango_inter_cuart=Q3-Q1
print("El rango intercuartil:",Rango_inter_cuart)
mediana=nuevo.EDAD.median()
print("La mediana:",mediana)
minimo=nuevo.EDAD.min()
print("El valor mínimo:",minimo)
maximo=nuevo.EDAD.max()
print("El valor máximo",maximo)
N=Q3+1.5*(Q3-Q1)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL BOXPLOT DE LA EDAD

Primer cuartil: 32.0
Tercer cuartil: 48.0
El rango intercuartil: 16.0
```

```
La mediana: 39.0
El valor mínimo: 0
El valor máximo 114
Bigote superior de mi boxplot es:  72.0
Bigote inferior de mi boxplot es:  24.0
```

Eliminando los Outliers

[95]:
```
dat_3=pd.read_csv("TRAIN_FUGA_COMPLETO.csv",sep=",", encoding="ISO-8859-1")
dat_3.head(50)
```

[95]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO
0	201411	0	46	F	PIURA	NO CLIENTE	2
1	201411	0	54	M	LORETO	CLIENTE	1BC
2	201411	0	81	M	LIMA	CLIENTE	6
3	201411	0	42	M	PIURA	CLIENTE	2
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC
5	201411	0	74	M	LA LIBERTAD	CLIENTE	6
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC
7	201411	0	57	M	LIMA	NO CLIENTE	2
8	201411	0	65	M	CALLAO	NO CLIENTE	2
9	201411	0	63	M	ANCASH	CLIENTE	2
10	201411	0	43	M	LIMA	CLIENTE	2
11	201411	0	64	F	LIMA	NO CLIENTE	2
12	201411	0	114	M	LIMA	CLIENTE	6
13	201411	0	48	F	LIMA	CLIENTE	2
14	201411	0	48	F	MOQUEGUA	CLIENTE	3
15	201411	0	114	F	LIMA	CLIENTE	6
16	201411	0	114	M	LIMA	CLIENTE	6
17	201411	0	42	F	LIMA	CLIENTE	3
18	201411	0	93	M	LIMA	NO CLIENTE	6
19	201411	0	114	F	LIMA	CLIENTE	6
20	201411	0	114	F	LIMA	CLIENTE	6
21	201411	0	39	M	LIMA	NO CLIENTE	3
22	201411	0	45	M	LIMA	CLIENTE	3
23	201411	0	50	M	LIMA	CLIENTE	3
24	201411	0	114	F	LIMA	CLIENTE	6
25	201411	0	114	M	LIMA	CLIENTE	6
26	201411	0	114	F	LIMA	CLIENTE	6
27	201411	0	114	M	LIMA	CLIENTE	6
28	201411	0	114	M	LIMA	CLIENTE	6
29	201411	0	114	M	LIMA	CLIENTE	6
30	201411	0	114	M	LIMA	CLIENTE	6
31	201411	0	114	M	LIMA	CLIENTE	6
32	201411	0	114	F	LIMA	CLIENTE	6
33	201411	0	52	F	AREQUIPA	CLIENTE	2
34	201411	0	57	M	LIMA	CLIENTE	3
35	201411	0	114	F	LIMA	CLIENTE	6

36	201411	0	48	M	LIMA	NO CLIENTE	2	
37	201411	0	57	F	LIMA	NO CLIENTE	1BC	
38	201411	0	52	M	CALLAO	NO CLIENTE	3	
39	201411	0	61	F	ANCASH	CLIENTE	5	
40	201411	0	48	M	LIMA	CLIENTE	3	
41	201411	0	46	M	CALLAO	NO CLIENTE	6	
42	201411	0	45	F	LIMA	NO CLIENTE	2	
43	201411	0	54	F	LIMA	NO CLIENTE	6	
44	201411	0	44	M	LIMA	CLIENTE	3	
45	201411	0	42	M	LIMA	CLIENTE	3	
46	201411	0	44	F	LIMA	NO CLIENTE	6	
47	201411	0	53	M	LIMA	CLIENTE	6	
48	201411	0	47	F	LIMA	CLIENTE	3	
49	201411	0	37	M	LIMA	NO CLIENTE	6	

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
5	0	0	0	0	0	
6	0	0	0	0	6	
7	0	0	5	6	6	
8	0	0	0	0	0	
9	0	0	0	0	0	
10	0	0	0	0	0	
11	0	0	0	0	0	
12	0	0	0	0	0	
13	0	0	0	0	0	
14	0	0	0	0	0	
15	0	0	0	0	0	
16	0	0	0	0	0	
17	0	0	0	0	0	
18	0	0	0	0	0	
19	0	0	0	0	0	
20	0	0	0	0	0	
21	0	0	0	0	0	
22	0	0	0	0	0	
23	0	0	0	0	0	
24	0	0	0	0	0	
25	0	0	0	0	0	
26	0	0	0	0	0	
27	0	0	0	0	0	
28	0	0	0	0	0	
29	0	0	0	0	0	
30	0	0	0	0	0	

31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	2	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	3	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
0	0.000000	224.0	0	2184.9
1	0.000000	123.0	0	4718.0
2	0.000000	264.0	0	2184.9
3	0.000000	263.0	0	936.0
4	0.000000	263.0	0	5844.0
5	0.000000	256.0	0	2184.9
6	0.000000	85.0	0	4232.0
7	2.166667	151.0	0	1580.0
8	3.333333	778.0	0	3081.5
9	0.000000	272.0	0	936.0
10	0.000000	11.0	0	1421.0
11	0.000000	21.0	0	2184.9
12	0.000000	281.0	0	2184.9
13	0.000000	209.0	0	809.0
14	0.000000	208.0	0	739.0
15	0.000000	233.0	0	2184.9
16	0.000000	216.0	0	2184.9
17	0.000000	233.0	0	739.0
18	0.000000	281.0	0	2184.9
19	0.000000	263.0	0	2184.9
20	0.000000	221.0	0	2184.9
21	0.000000	163.0	0	749.0
22	0.000000	215.0	0	936.0
23	0.000000	257.0	0	936.0
24	0.000000	206.0	0	2184.9
25	0.000000	281.0	0	2184.9

26	0.000000	280.0	0	2184.9
27	0.000000	220.0	0	2184.9
28	0.000000	275.0	0	2184.9
29	0.000000	263.0	0	2184.9
30	0.000000	280.0	0	2184.9
31	0.000000	263.0	0	2184.9
32	0.000000	203.0	0	2184.9
33	0.000000	202.0	0	739.0
34	0.000000	264.0	0	858.0
35	0.000000	263.0	0	2184.9
36	0.000000	265.0	0	2184.9
37	0.000000	778.0	0	3962.0
38	0.000000	85.0	0	2184.9
39	0.000000	257.0	0	739.0
40	0.000000	191.0	0	858.0
41	0.000000	162.0	0	2184.9
42	0.000000	185.0	0	2184.9
43	0.000000	159.0	0	2184.9
44	0.000000	280.0	0	818.0
45	0.333333	76.0	0	2777.0
46	0.000000	778.0	0	2184.9
47	0.000000	280.0	0	2184.9
48	0.000000	164.0	0	1093.0
49	0.000000	209.0	0	2184.9

IQR score

[96]:
```
Q1 = dat_3.quantile(0.25)
Q3 = dat_3.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
CODMES                  92.0
TARGET_MODEL2            0.0
EDAD                    18.0
FLG_ADEL_SUELDO_M1       0.0
FREC_AGENTE              0.0
FREC_KIOSKO              1.0
FREC_BPI_TD              0.0
FREC_MON_TD              0.0
PROM_CTD_TRX_6M          0.0
ANT_CLIENTE             90.0
CTD_RECLAMOS_M1          0.0
INGRESO_BRUTO_M1      1085.0
dtype: float64
```

[97]:
```
print((dat_3 < (Q1 - 1.5 * IQR)) | (dat_3 > (Q3 + 1.5 * IQR)))
```

```
        ANT_CLIENTE  CODMES  CTD_RECLAMOS_M1  DEPARTAMENTO   EDAD  \
0             False   False            False         False  False
1             False   False            False         False  False
2              True   False            False         False   True
3              True   False            False         False  False
4              True   False            False         False  False
...             ...     ...              ...           ...    ...
787490        False   False            False         False  False
787491        False   False            False         False  False
787492        False   False            False         False  False
787493        False   False            False         False  False
787494        False   False            False         False  False

        FLG_ADEL_SUELDO_M1  FLG_CLIENTE  FREC_AGENTE  FREC_BPI_TD  \
0                    False        False        False        False
1                    False        False        False        False
2                    False        False        False        False
3                    False        False        False        False
4                    False        False        False        False
...                    ...          ...          ...          ...
787490               False        False        False        False
787491               False        False        False        False
787492               False        False        False        False
787493               False        False        False        False
787494               False        False        False        False

        FREC_KIOSKO  FREC_MON_TD  INGRESO_BRUTO_M1  PROM_CTD_TRX_6M  SEGMENTO  \
0             False        False             False            False     False
1             False        False              True            False     False
2             False        False             False            False     False
3             False        False             False            False     False
4             False        False              True            False     False
...             ...          ...               ...              ...       ...
787490        False        False             False            False     False
787491        False        False              True             True     False
787492        False        False              True             True     False
787493         True         True             False            False     False
787494        False        False              True             True     False

         SEXO  TARGET_MODEL2
0       False          False
1       False          False
2       False          False
3       False          False
4       False          False
...       ...            ...
787490  False          False
787491  False          False
```

```
787492  False           True
787493  False          False
787494  False          False

[787495 rows x 16 columns]
```

[98]:
```
dat_3_out= dat_3[~((dat_3 < (Q1 - 1.5 * IQR)) |(dat_3 > (Q3 + 1.5 * IQR))).
 ↪any(axis=1)]
dat_3_out.head(50)
```

[98]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
10	201411	0	43	M	LIMA	CLIENTE	2	
11	201411	0	64	F	LIMA	NO CLIENTE	2	
13	201411	0	48	F	LIMA	CLIENTE	2	
14	201411	0	48	F	MOQUEGUA	CLIENTE	3	
17	201411	0	42	F	LIMA	CLIENTE	3	
21	201411	0	39	M	LIMA	NO CLIENTE	3	
22	201411	0	45	M	LIMA	CLIENTE	3	
33	201411	0	52	F	AREQUIPA	CLIENTE	2	
38	201411	0	52	M	CALLAO	NO CLIENTE	3	
40	201411	0	48	M	LIMA	CLIENTE	3	
41	201411	0	46	M	CALLAO	NO CLIENTE	6	
42	201411	0	45	F	LIMA	NO CLIENTE	2	
43	201411	0	54	F	LIMA	NO CLIENTE	6	
48	201411	0	47	F	LIMA	CLIENTE	3	
49	201411	0	37	M	LIMA	NO CLIENTE	6	
51	201411	0	39	F	LIMA	NO CLIENTE	5	
52	201411	0	37	F	LIMA	NO CLIENTE	5	
53	201411	0	50	M	CALLAO	CLIENTE	3	
54	201411	0	58	M	LIMA	CLIENTE	3	
55	201411	0	40	F	LIMA	NO CLIENTE	5	
57	201411	0	51	M	LIMA	CLIENTE	6	
59	201411	0	37	M	LORETO	NO CLIENTE	6	
61	201411	0	34	F	LIMA	CLIENTE	3	
62	201411	0	69	M	LIMA	CLIENTE	2	
63	201411	0	54	F	LIMA	NO CLIENTE	6	
65	201411	0	64	M	LIMA	CLIENTE	3	
66	201411	0	49	M	LIMA	CLIENTE	3	
67	201411	0	59	M	SAN MARTIN	NO CLIENTE	6	
68	201411	0	43	F	LIMA	CLIENTE	3	
69	201411	0	50	M	LIMA	CLIENTE	2	
70	201411	0	47	M	JUNIN	NO CLIENTE	4	
71	201411	0	61	M	LIMA	CLIENTE	3	
72	201411	0	49	F	LIMA	CLIENTE	3	
73	201411	0	41	M	LIMA	CLIENTE	3	
74	201411	0	32	M	LIMA	CLIENTE	3	

76	201411	0	43	F	CALLAO		CLIENTE	5	
77	201411	0	41	M	LORETO		CLIENTE	3	
78	201411	0	46	M	LIMA		CLIENTE	3	
79	201411	0	45	F	LIMA	NO	CLIENTE	3	
80	201411	0	36	M	LIMA	NO	CLIENTE	2	
82	201411	0	41	M	ICA		CLIENTE	3	
85	201411	0	31	M	LIMA	NO	CLIENTE	6	
86	201411	0	33	F	LIMA		CLIENTE	5	
87	201411	0	32	F	LIMA	NO	CLIENTE	2	
88	201411	0	34	M	PIURA	NO	CLIENTE	2	
89	201411	0	54	M	ICA	NO	CLIENTE	6	
92	201411	0	40	M	LIMA	NO	CLIENTE	2	
93	201411	0	38	M	AYACUCHO	NO	CLIENTE	6	
99	201411	0	30	M	LIMA		CLIENTE	3	

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
10	0	0	0	0	0	
11	0	0	0	0	0	
13	0	0	0	0	0	
14	0	0	0	0	0	
17	0	0	0	0	0	
21	0	0	0	0	0	
22	0	0	0	0	0	
33	0	0	0	0	0	
38	0	0	0	0	0	
40	0	0	0	0	0	
41	0	0	0	0	0	
42	0	0	0	0	0	
43	0	0	0	0	0	
48	0	0	0	0	0	
49	0	0	0	0	0	
51	0	0	0	0	0	
52	0	0	0	0	0	
53	0	0	0	0	0	
54	0	0	0	0	0	
55	0	0	0	0	0	
57	0	0	0	0	0	
59	0	0	0	0	0	
61	0	0	0	0	0	
62	0	0	1	0	0	
63	0	0	0	0	0	
65	0	0	0	0	0	
66	0	0	0	0	0	
67	0	0	0	0	0	
68	0	0	0	0	0	
69	0	0	0	0	0	

70	0	0	0	0	0
71	0	0	0	0	0
72	0	0	0	0	0
73	0	0	0	0	0
74	0	0	0	0	0
76	0	0	0	0	0
77	0	0	0	0	0
78	0	0	0	0	0
79	0	0	0	0	0
80	0	0	0	0	0
82	0	0	0	0	0
85	0	0	0	0	0
86	0	0	0	0	0
87	0	0	0	0	0
88	0	0	0	0	0
89	0	0	0	0	0
92	0	0	0	0	0
93	0	0	0	0	0
99	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
0	0.0	224.0	0	2184.9
10	0.0	11.0	0	1421.0
11	0.0	21.0	0	2184.9
13	0.0	209.0	0	809.0
14	0.0	208.0	0	739.0
17	0.0	233.0	0	739.0
21	0.0	163.0	0	749.0
22	0.0	215.0	0	936.0
33	0.0	202.0	0	739.0
38	0.0	85.0	0	2184.9
40	0.0	191.0	0	858.0
41	0.0	162.0	0	2184.9
42	0.0	185.0	0	2184.9
43	0.0	159.0	0	2184.9
48	0.0	164.0	0	1093.0
49	0.0	209.0	0	2184.9
51	0.0	197.0	0	2184.9
52	0.0	164.0	0	2184.9
53	0.0	179.0	0	1235.0
54	0.0	126.0	0	858.0
55	0.0	161.0	0	2184.9
57	0.0	167.0	0	2184.9
59	0.0	161.0	0	2184.9
61	0.0	215.0	0	739.0
62	0.0	164.0	0	858.0
63	0.0	203.0	0	2184.9

65	0.0	156.0	0	1235.0
66	0.0	245.0	0	858.0
67	0.0	152.0	0	2184.9
68	0.0	142.0	0	1093.0
69	0.0	154.0	0	1186.0
70	0.0	162.0	0	2184.9
71	0.0	161.0	0	936.0
72	0.0	159.0	0	809.0
73	0.0	158.0	0	1235.0
74	0.0	143.0	0	1235.0
76	0.0	140.0	0	1156.0
77	0.0	147.0	0	936.0
78	0.0	140.0	0	1235.0
79	0.0	126.0	0	2184.9
80	0.0	139.0	0	2184.9
82	0.0	141.0	0	1235.0
85	0.0	131.0	0	2184.9
86	0.0	128.0	0	1078.0
87	0.0	70.0	0	2184.9
88	0.0	125.0	0	1500.0
89	0.0	132.0	0	2184.9
92	0.0	65.0	0	2184.9
93	0.0	113.0	0	2184.9
99	0.0	101.0	0	936.0

[99]:
```
len(dat_3_out)
```

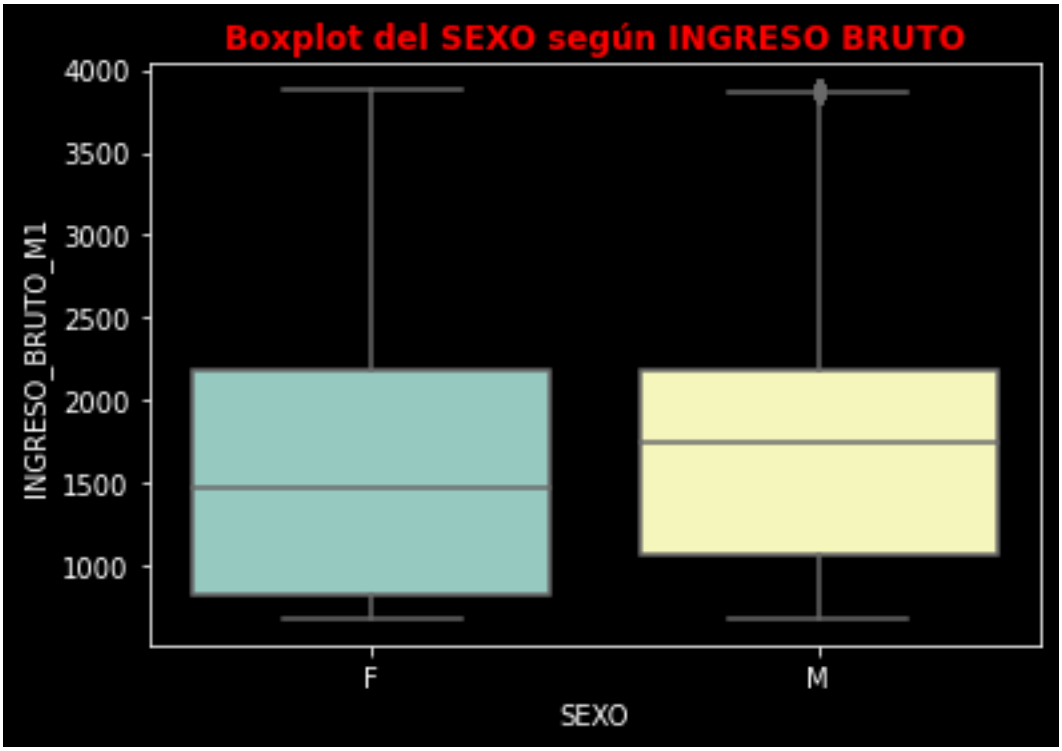
[99]: 321236

[100]:
```
len(dat_3)
```

[100]: 787495

Graficando despues de eliminar los Outliers BOXPLOT DEL SEXO SEGÚN EL INGRESO BRUTO

[101]:
```
with plt.style.context('dark_background'):
    sns.boxplot(x=dat_3_out['SEXO'], #Será la variable categorizadora o␣
 ↪separadora.
            y=dat_3_out['INGRESO_BRUTO_M1']) #La variable cuantitativa de rpta.
plt.title("Boxplot del SEXO según INGRESO BRUTO",weight='bold',color='red')
plt.show()
```

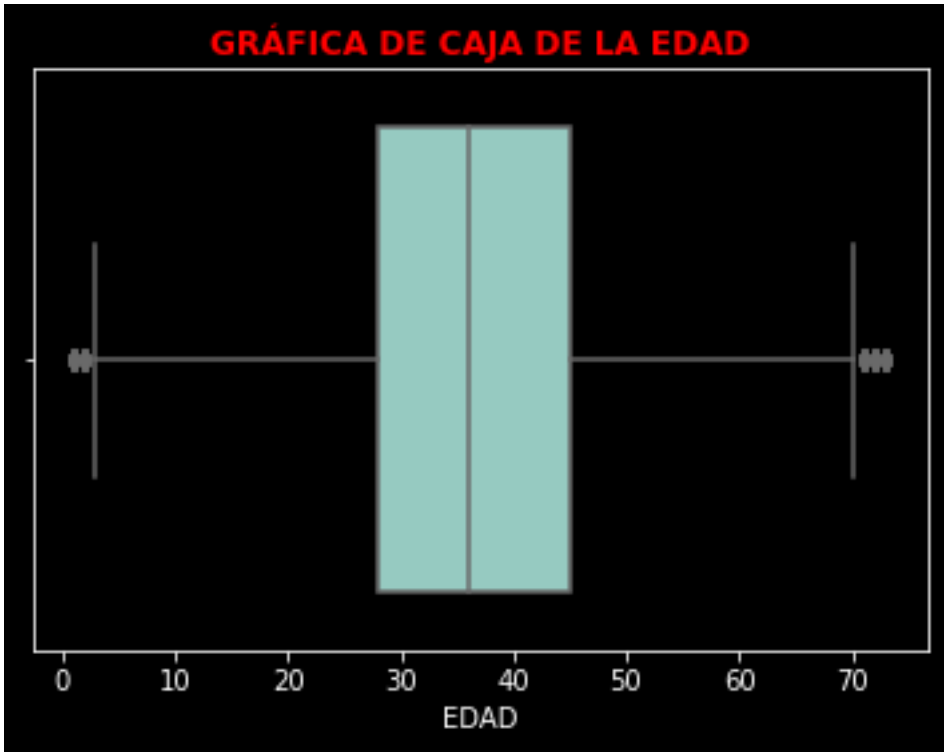


[102]:
```
print("INFORMACIÓN DEL GÉNERO MASCULINO SEGÚN EL INGRESO BRUTO\n")
mediana=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.median()
Q1=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.quantile(0.25)
Q3=nuevo[nuevo.SEXO == 'M'].INGRESO_BRUTO_M1.quantile(0.75)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Primer cuartil:",Q1)
print("Tercer cuartil:",Q3)
print("La mediana:",mediana)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)

print("INFORMACIÓN DEL GÉNERO FEMENINO SEGÚN EL INGRESO BRUTO\n")
mediana=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.median()
Q1=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.quantile(0.25)
Q3=nuevo[nuevo.SEXO == 'F'].INGRESO_BRUTO_M1.quantile(0.75)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Primer cuartil:",Q1)
print("Tercer cuartil:",Q3)
print("La mediana:",mediana)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL GÉNERO MASCULINO SEGÚN EL INGRESO BRUTO

Primer cuartil: 3021.0
Tercer cuartil: 6131.0
La mediana: 3886.0
Bigote superior de mi boxplot es:  10796.0
Bigote inferior de mi boxplot es:  1466.0
INFORMACIÓN DEL GÉNERO FEMENINO SEGÚN EL INGRESO BRUTO

Primer cuartil: 3046.0
Tercer cuartil: 5839.0
La mediana: 3949.0
Bigote superior de mi boxplot es:  10028.5
Bigote inferior de mi boxplot es:  1649.5
```

BOXPLOT DE LA EDAD

[103]:
```
with plt.style.context('dark_background'):
    sns.boxplot(dat_3_out.EDAD)
plt.title("GRÁFICA DE CAJA DE LA EDAD",weight='bold',color='red')
plt.show()
```



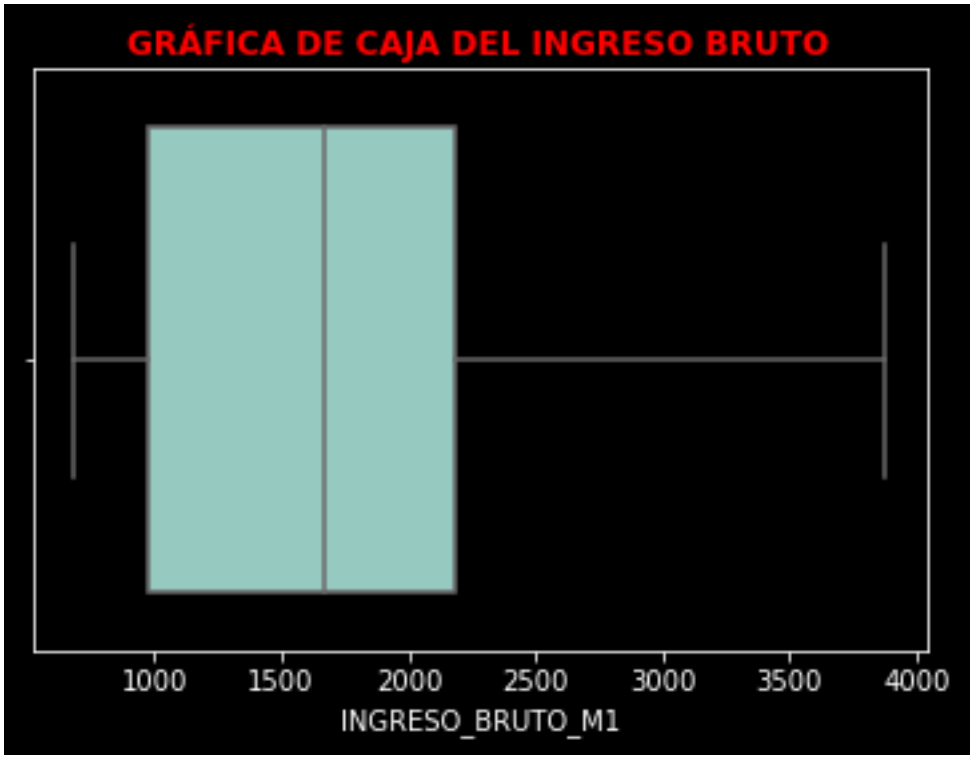
[104]:
```
print("INFORMACIÓN DEL BOXPLOT DE LA EDAD\n")
Q1=dat_3_out.EDAD.quantile(0.25)
```

```
print("Primer cuartil:",Q1)
Q3=dat_3_out.EDAD.quantile(0.75)
print("Tercer cuartil:",Q3)
Rango_inter_cuart=Q3-Q1
print("El rango intercuartil:",Rango_inter_cuart)
mediana=dat_3_out.EDAD.median()
print("La mediana:",mediana)
minimo=dat_3_out.EDAD.min()
print("El valor mínimo:",minimo)
maximo=dat_3_out.EDAD.max()
print("El valor máximo",maximo)
N=Q3+1.5*(Q3-Q1)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL BOXPLOT DE LA EDAD

Primer cuartil: 28.0
Tercer cuartil: 45.0
El rango intercuartil: 17.0
La mediana: 36.0
El valor mínimo: 1
El valor máximo 73
Bigote superior de mi boxplot es:  70.5
Bigote inferior de mi boxplot es:  19.5
```

BOXPLOT DEL INGRESO BRUTO

[105]:
```
with plt.style.context('dark_background'):
    sns.boxplot(dat_3_out.INGRESO_BRUTO_M1)
plt.title("GRÁFICA DE CAJA DEL INGRESO BRUTO",weight='bold',color='red')
plt.show()
```



[106]:
```
print("INFORMACIÓN DEL BOXPLOT DEL INGRESO BRUTO\n")
Q1=dat_3_out.INGRESO_BRUTO_M1.quantile(0.25)
print("Primer cuartil:",Q1)
Q3=dat_3_out.INGRESO_BRUTO_M1.quantile(0.75)
print("Tercer cuartil:",Q3)
Rango_inter_cuart=Q3-Q1
print("El rango intercuartil:",Rango_inter_cuart)
mediana=dat_3_out.INGRESO_BRUTO_M1.median()
print("La mediana:",mediana)
minimo=dat_3_out.INGRESO_BRUTO_M1.min()
print("El valor mínimo:",minimo)
maximo=dat_3_out.INGRESO_BRUTO_M1.max()
print("El valor máximo",maximo)
N=Q3+1.5*(Q3-Q1)
BS=Q3+1.5*(Q3-Q1)
BI=Q3-1.5*(Q3-Q1)
print("Bigote superior de mi boxplot es: ", BS)
print("Bigote inferior de mi boxplot es: ", BI)
```

```
INFORMACIÓN DEL BOXPLOT DEL INGRESO BRUTO

Primer cuartil: 972.0
Tercer cuartil: 2184.9
El rango intercuartil: 1212.9
```

```
La mediana: 1671.0
El valor mínimo: 681.0
El valor máximo 3876.0
Bigote superior de mi boxplot es:  4004.25
Bigote inferior de mi boxplot es:  365.54999999999995
```

Podemos vizualizar que acurrido algunos ajustes con las gráficas de cajas y con la información de las cajas y esto ocurrio ya que se eliminio los valores atípicos de nuestro DataFrame

1.4 3. Realizar una discretización de las variables : INGRESO_BRUTO_M1 y EDAD teniendo en cuenta al menos dos técnicas de discretización no supervisada y agregar las variables discretizadas a nuestro conjunto de datos original

[107]:
```
from sklearn.preprocessing import KBinsDiscretizer
```

[108]:
```
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 16 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   CODMES              787495 non-null  int64  
 1   TARGET_MODEL2       787495 non-null  int64  
 2   EDAD                787495 non-null  int64  
 3   SEXO                787495 non-null  object 
 4   DEPARTAMENTO        787495 non-null  object 
 5   FLG_CLIENTE         787495 non-null  object 
 6   SEGMENTO            787495 non-null  object 
 7   FLG_ADEL_SUELDO_M1  787495 non-null  int64  
 8   FREC_AGENTE         787495 non-null  int64  
 9   FREC_KIOSKO         787495 non-null  int64  
 10  FREC_BPI_TD         787495 non-null  int64  
 11  FREC_MON_TD         787495 non-null  int64  
 12  PROM_CTD_TRX_6M     787495 non-null  float64
 13  ANT_CLIENTE         787495 non-null  float64
 14  CTD_RECLAMOS_M1     787495 non-null  int64  
 15  INGRESO_BRUTO_M1    787495 non-null  float64
dtypes: float64(3), int64(9), object(4)
memory usage: 96.1+ MB
```

1.4.1 3.1 Descretización por intervalos de igual amplitud

PARA LA VARIABLA EDAD

[109]:
```
n=len(dat_2)
k=1+math.log2(n)
k=round(k,0)
```

```
k
```

[109]:
```
21.0
```

[110]:
```
amplitud=KBinsDiscretizer(n_bins=21,encode="ordinal",strategy="uniform")
nueva_dat_2=amplitud.fit_transform(dat_2[['EDAD']])
dat_2["EDAD_amplitud"]=nueva_dat_2
dat_2["EDAD_amplitud"]=dat_2["EDAD_amplitud"].astype(np.int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 17 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   CODMES              787495 non-null  int64
 1   TARGET_MODEL2       787495 non-null  int64
 2   EDAD                787495 non-null  int64
 3   SEXO                787495 non-null  object
 4   DEPARTAMENTO        787495 non-null  object
 5   FLG_CLIENTE         787495 non-null  object
 6   SEGMENTO            787495 non-null  object
 7   FLG_ADEL_SUELDO_M1  787495 non-null  int64
 8   FREC_AGENTE         787495 non-null  int64
 9   FREC_KIOSKO         787495 non-null  int64
 10  FREC_BPI_TD         787495 non-null  int64
 11  FREC_MON_TD         787495 non-null  int64
 12  PROM_CTD_TRX_6M     787495 non-null  float64
 13  ANT_CLIENTE         787495 non-null  float64
 14  CTD_RECLAMOS_M1     787495 non-null  int64
 15  INGRESO_BRUTO_M1    787495 non-null  float64
 16  EDAD_amplitud       787495 non-null  int64
dtypes: float64(3), int64(10), object(4)
memory usage: 102.1+ MB
```

[111]:
```
graf_edad_amplitud=dat_2.groupby(dat_2.EDAD_amplitud).size()
graf_edad_amplitud
```

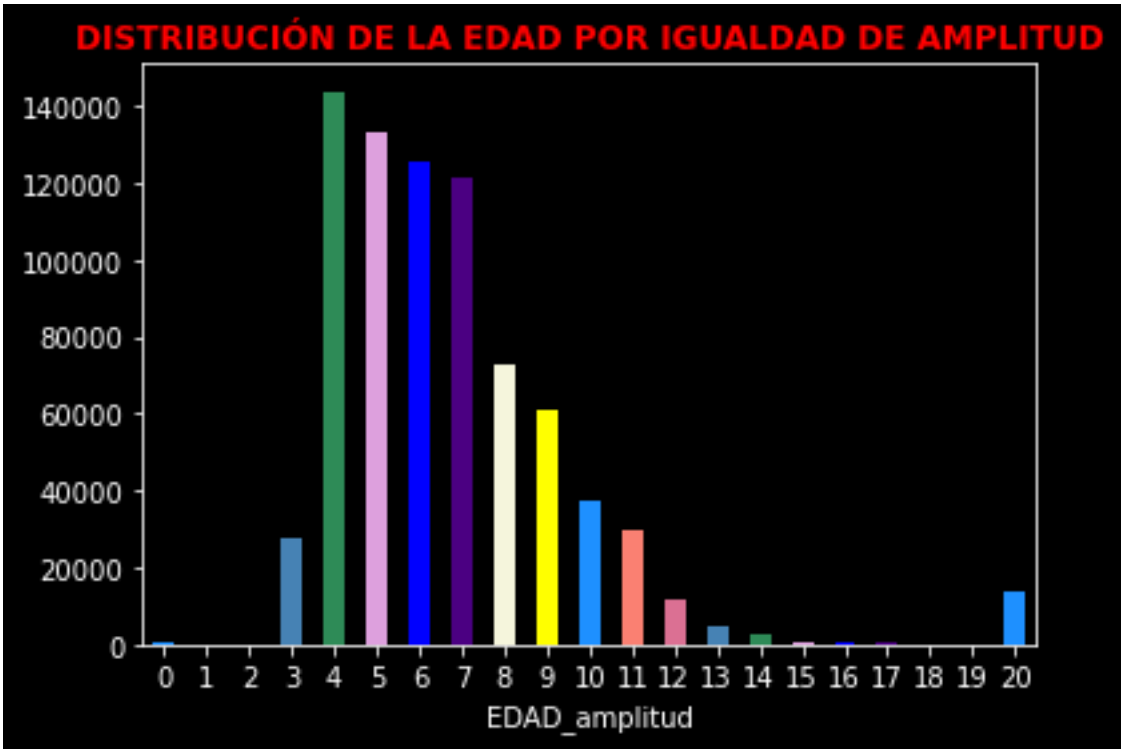
[111]:
```
EDAD_amplitud
0        349
1         24
2         84
3      27361
4     143750
5     133439
6     125875
7     121279
```

```
8      72547
9      61150
10     37344
11     29881
12     11479
13      4566
14      2566
15       920
16       386
17       749
18        37
19       198
20     13511
dtype: int64
```

[112]:
```
graf_edad_amplitud/sum(graf_edad_amplitud)
```

[112]:
```
EDAD_amplitud
0     0.000443
1     0.000030
2     0.000107
3     0.034744
4     0.182541
5     0.169447
6     0.159842
7     0.154006
8     0.092124
9     0.077651
10    0.047421
11    0.037944
12    0.014577
13    0.005798
14    0.003258
15    0.001168
16    0.000490
17    0.000951
18    0.000047
19    0.000251
20    0.017157
dtype: float64
```

[113]:
```
with plt.style.context('dark_background'):
    graf_edad_amplitud.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DE LA EDAD POR IGUALDAD DE␣
 ↪AMPLITUD",weight='bold',color='red')
plt.show()
```



PARA LA VARIABLE INGRESO BRUTO

[114]:
```
amplitud_ingreso=KBinsDiscretizer(n_bins=21,encode="ordinal",strategy="uniform")
nueva_dat_2_ingreso=amplitud_ingreso.fit_transform(dat_2[['INGRESO_BRUTO_M1']])
dat_2["INGRESO_BRUTO_M1_amplitud"]=nueva_dat_2_ingreso
dat_2["INGRESO_BRUTO_M1_amplitud"]=dat_2["INGRESO_BRUTO_M1_amplitud"].astype(np.
 ↪int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 18 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   CODMES                    787495 non-null  int64
 1   TARGET_MODEL2             787495 non-null  int64
 2   EDAD                      787495 non-null  int64
 3   SEXO                      787495 non-null  object
 4   DEPARTAMENTO              787495 non-null  object
 5   FLG_CLIENTE               787495 non-null  object
 6   SEGMENTO                  787495 non-null  object
 7   FLG_ADEL_SUELDO_M1        787495 non-null  int64
 8   FREC_AGENTE               787495 non-null  int64
 9   FREC_KIOSKO               787495 non-null  int64
```

```
 10  FREC_BPI_TD                787495 non-null  int64
 11  FREC_MON_TD                787495 non-null  int64
 12  PROM_CTD_TRX_6M            787495 non-null  float64
 13  ANT_CLIENTE                787495 non-null  float64
 14  CTD_RECLAMOS_M1            787495 non-null  int64
 15  INGRESO_BRUTO_M1           787495 non-null  float64
 16  EDAD_amplitud              787495 non-null  int64
 17  INGRESO_BRUTO_M1_amplitud  787495 non-null  int64
dtypes: float64(3), int64(11), object(4)
memory usage: 108.1+ MB
```

[115]:
```
graf_ingreso_amplitud=dat_2.groupby(dat_2.INGRESO_BRUTO_M1_amplitud).size()
graf_ingreso_amplitud
```

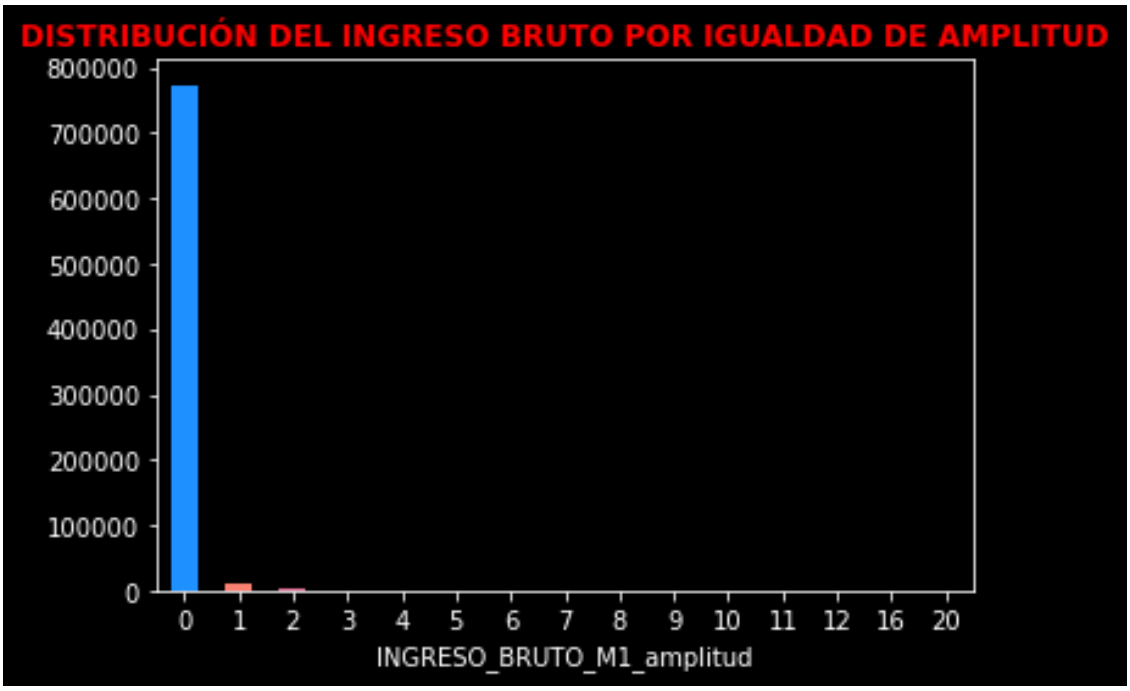
[115]:
```
INGRESO_BRUTO_M1_amplitud
0     772901
1      12055
2       1621
3        563
4        187
5         71
6         35
7         28
8         15
9          4
10         6
11         4
12         3
16         1
20         1
dtype: int64
```

[116]:
```
graf_ingreso_amplitud/sum(graf_ingreso_amplitud)
```

[116]:
```
INGRESO_BRUTO_M1_amplitud
0     0.981468
1     0.015308
2     0.002058
3     0.000715
4     0.000237
5     0.000090
6     0.000044
7     0.000036
8     0.000019
9     0.000005
10    0.000008
11    0.000005
```

```
12    0.000004
16    0.000001
20    0.000001
dtype: float64
```

[117]:
```
with plt.style.context('dark_background'):
    graf_ingreso_amplitud.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DEL INGRESO BRUTO POR IGUALDAD DE␣
 ↪AMPLITUD",weight='bold',color='red')
plt.show()
```



1.4.2 3.2 Discretización por Cuantiles

PARA LA VARIABLA EDAD

[118]:
```
quartil=KBinsDiscretizer(n_bins=4, encode="ordinal",strategy="quantile")
nuevo_dat_2_cuantil=quartil.fit_transform(dat_2[['EDAD']])
dat_2["EDAD_quartil"]=nuevo_dat_2_cuantil
dat_2["EDAD_quartil"]=dat_2["EDAD_quartil"].astype(np.int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 19 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
```

```
 0   CODMES                       787495 non-null  int64
 1   TARGET_MODEL2                787495 non-null  int64
 2   EDAD                         787495 non-null  int64
 3   SEXO                         787495 non-null  object
 4   DEPARTAMENTO                 787495 non-null  object
 5   FLG_CLIENTE                  787495 non-null  object
 6   SEGMENTO                     787495 non-null  object
 7   FLG_ADEL_SUELDO_M1           787495 non-null  int64
 8   FREC_AGENTE                  787495 non-null  int64
 9   FREC_KIOSKO                  787495 non-null  int64
 10  FREC_BPI_TD                  787495 non-null  int64
 11  FREC_MON_TD                  787495 non-null  int64
 12  PROM_CTD_TRX_6M              787495 non-null  float64
 13  ANT_CLIENTE                  787495 non-null  float64
 14  CTD_RECLAMOS_M1              787495 non-null  int64
 15  INGRESO_BRUTO_M1             787495 non-null  float64
 16  EDAD_amplitud                787495 non-null  int64
 17  INGRESO_BRUTO_M1_amplitud    787495 non-null  int64
 18  EDAD_quartil                 787495 non-null  int64
dtypes: float64(3), int64(12), object(4)
memory usage: 114.2+ MB
```

[119]:
```
graf_edad_quartil=dat_2.groupby(dat_2.EDAD_quartil).size()
graf_edad_quartil
```

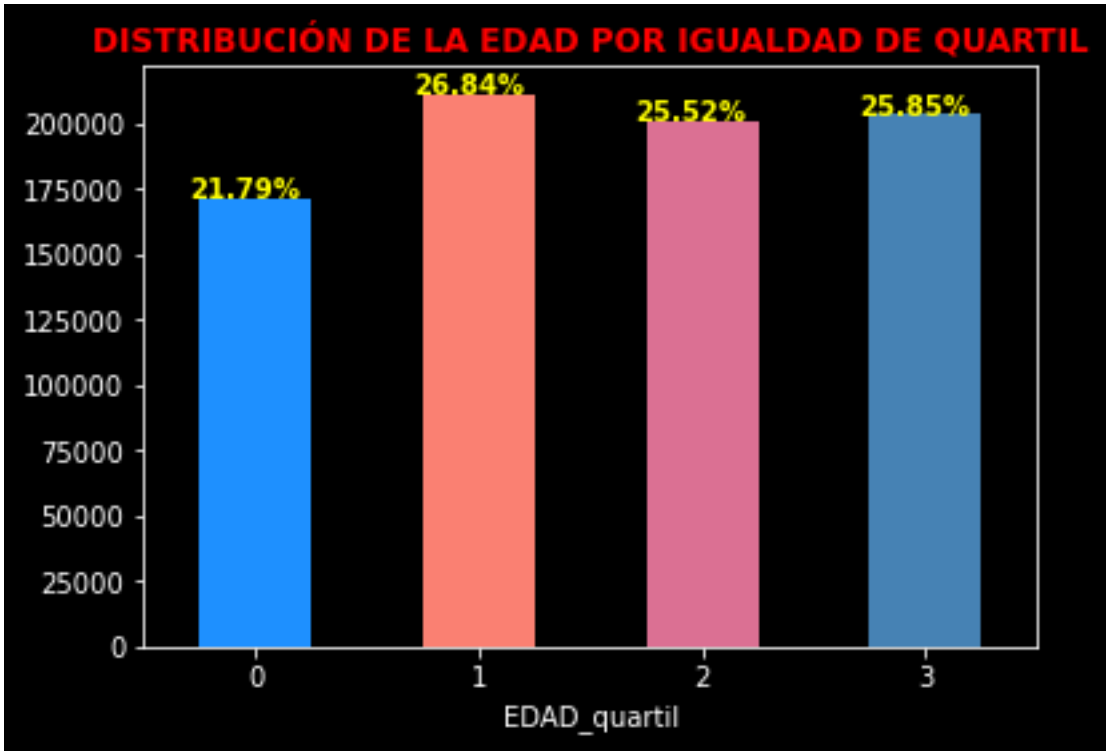
[119]:
```
EDAD_quartil
0    171568
1    211373
2    200973
3    203581
dtype: int64
```

[120]:
```
graf_edad_quartil/sum(graf_edad_quartil)*100
```

[120]:
```
EDAD_quartil
0    21.786551
1    26.841186
2    25.520543
3    25.851720
dtype: float64
```

[121]:
```
with plt.style.context('dark_background'):
    graf_edad_quartil.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DE LA EDAD POR IGUALDAD DE␣
 ↪QUARTIL",weight='bold',color='red')
plt.text(-0.3,171568,"21.79%",weight="bold",color="yellow")
plt.text(0.7,211373,"26.84%",weight="bold",color="yellow")
```

```
plt.text(1.7,200973,"25.52%",weight="bold",color="yellow")
plt.text(2.7,203581,"25.85%",weight="bold",color="yellow")
plt.show()
```



PARA LA VARIABLE INGRESO BRUTO

[122]:
```
quartil_ingreso=KBinsDiscretizer(n_bins=4,encode="ordinal",strategy="quantile")
nueva_dat_2_quartil_ingreso=quartil_ingreso.
 ↪fit_transform(dat_2[['INGRESO_BRUTO_M1']])
dat_2["INGRESO_BRUTO_M1_quartil"]=nueva_dat_2_quartil_ingreso
dat_2["INGRESO_BRUTO_M1_quartil"]=dat_2["INGRESO_BRUTO_M1_quartil"].astype(np.
 ↪int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 20 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   CODMES                    787495 non-null  int64
 1   TARGET_MODEL2             787495 non-null  int64
 2   EDAD                      787495 non-null  int64
 3   SEXO                      787495 non-null  object
 4   DEPARTAMENTO              787495 non-null  object
```

```
 5   FLG_CLIENTE                787495 non-null  object
 6   SEGMENTO                   787495 non-null  object
 7   FLG_ADEL_SUELDO_M1         787495 non-null  int64
 8   FREC_AGENTE                787495 non-null  int64
 9   FREC_KIOSKO                787495 non-null  int64
 10  FREC_BPI_TD                787495 non-null  int64
 11  FREC_MON_TD                787495 non-null  int64
 12  PROM_CTD_TRX_6M            787495 non-null  float64
 13  ANT_CLIENTE                787495 non-null  float64
 14  CTD_RECLAMOS_M1            787495 non-null  int64
 15  INGRESO_BRUTO_M1           787495 non-null  float64
 16  EDAD_amplitud              787495 non-null  int64
 17  INGRESO_BRUTO_M1_amplitud  787495 non-null  int64
 18  EDAD_quartil               787495 non-null  int64
 19  INGRESO_BRUTO_M1_quartil   787495 non-null  int64
dtypes: float64(3), int64(13), object(4)
memory usage: 120.2+ MB
```

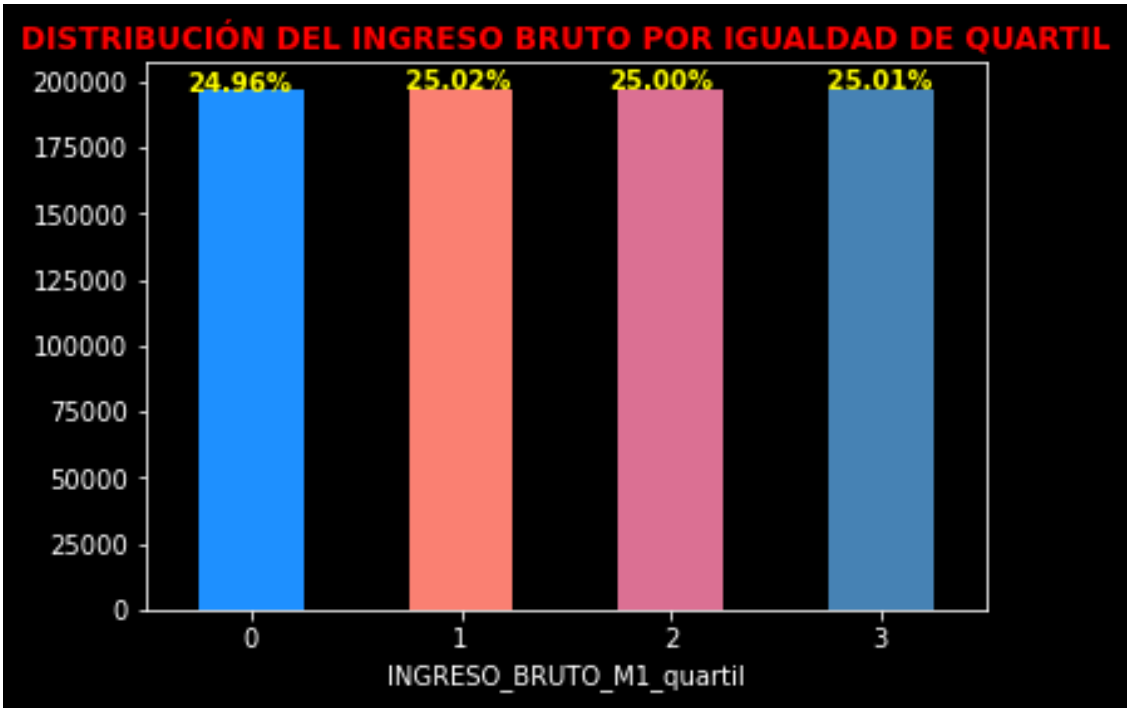
[123]:
```
graf_ingreso_quartil=dat_2.groupby(dat_2.INGRESO_BRUTO_M1_quartil).size()
graf_ingreso_quartil
```

[123]:
```
INGRESO_BRUTO_M1_quartil
0    196597
1    197066
2    196855
3    196977
dtype: int64
```

[124]:
```
graf_ingreso_quartil/sum(graf_ingreso_quartil)*100
```

[124]:
```
INGRESO_BRUTO_M1_quartil
0    24.964857
1    25.024413
2    24.997619
3    25.013111
dtype: float64
```

[125]:
```
graf_ingreso_quartil=dat_2.groupby(dat_2.INGRESO_BRUTO_M1_quartil).size()
with plt.style.context('dark_background'):
    graf_ingreso_quartil.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DEL INGRESO BRUTO POR IGUALDAD DE␣
 ↪QUARTIL",weight='bold',color='red')
plt.text(-0.3,196597,"24.96%",weight="bold",color="yellow")
plt.text(0.7,197066," 25.02%",weight="bold",color="yellow")
plt.text(1.7,196855,"25.00%",weight="bold",color="yellow")
plt.text(2.7,196977," 25.01%",weight="bold",color="yellow")
plt.show()
```



1.4.3 3.3. Discretización por KMeans

PARA LA VARIABLA EDAD

[126]:
```
kmeans=KBinsDiscretizer(n_bins=6,encode="ordinal",strategy="kmeans")
nuevo_dat_2_kmeans=kmeans.fit_transform(dat_2[['EDAD']])
dat_2["EDAD_kmeans"]=nuevo_dat_2_kmeans
dat_2["EDAD_quartil"]=dat_2["EDAD_kmeans"].astype(np.int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 21 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   CODMES                    787495 non-null  int64
 1   TARGET_MODEL2             787495 non-null  int64
 2   EDAD                      787495 non-null  int64
 3   SEXO                      787495 non-null  object
 4   DEPARTAMENTO              787495 non-null  object
 5   FLG_CLIENTE               787495 non-null  object
 6   SEGMENTO                  787495 non-null  object
 7   FLG_ADEL_SUELDO_M1        787495 non-null  int64
 8   FREC_AGENTE               787495 non-null  int64
 9   FREC_KIOSKO               787495 non-null  int64
```

```
 10  FREC_BPI_TD                787495 non-null  int64
 11  FREC_MON_TD                787495 non-null  int64
 12  PROM_CTD_TRX_6M            787495 non-null  float64
 13  ANT_CLIENTE                787495 non-null  float64
 14  CTD_RECLAMOS_M1            787495 non-null  int64
 15  INGRESO_BRUTO_M1           787495 non-null  float64
 16  EDAD_amplitud              787495 non-null  int64
 17  INGRESO_BRUTO_M1_amplitud  787495 non-null  int64
 18  EDAD_quartil               787495 non-null  int64
 19  INGRESO_BRUTO_M1_quartil   787495 non-null  int64
 20  EDAD_kmeans                787495 non-null  float64
dtypes: float64(4), int64(13), object(4)
memory usage: 126.2+ MB
```

Hallando la frecuencia de la EDAD y su porcentaje

[127]:
```
graf_edad_kmeans=dat_2.groupby(dat_2.EDAD_kmeans).size()
graf_edad_kmeans
```

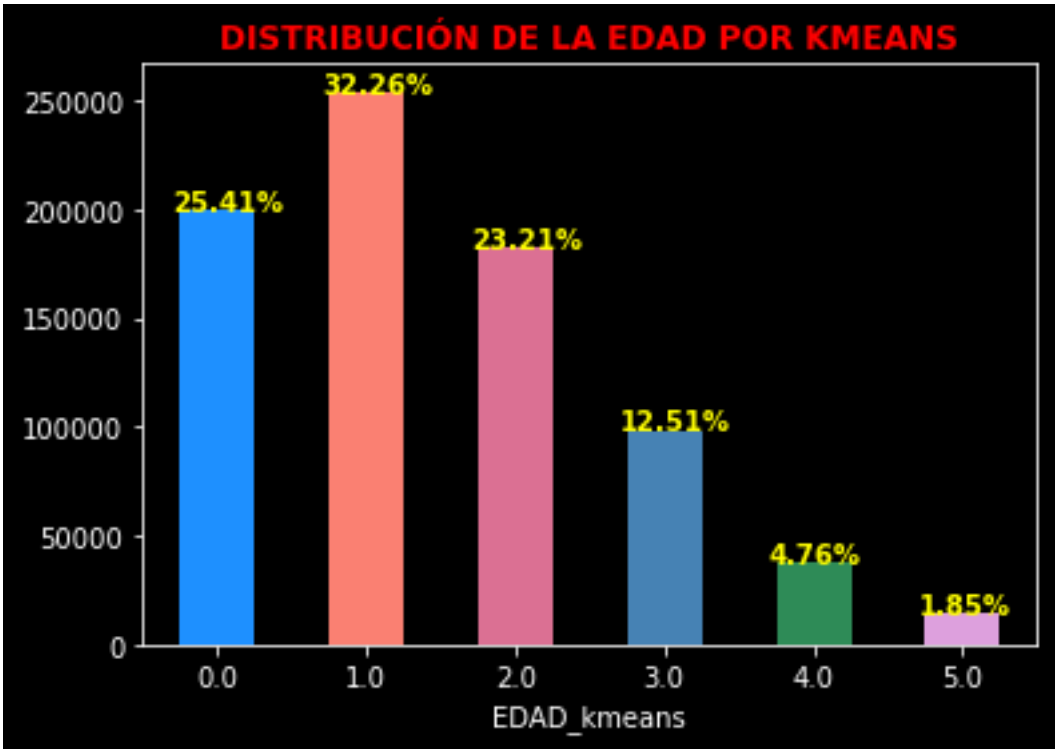
[127]:
```
EDAD_kmeans
0.0    200067
1.0    254058
2.0    182762
3.0     98509
4.0     37502
5.0     14597
dtype: int64
```

[128]:
```
graf_edad_kmeans/sum(graf_edad_kmeans)*100
```

[128]:
```
EDAD_kmeans
0.0    25.405495
1.0    32.261538
2.0    23.208020
3.0    12.509159
4.0     4.762189
5.0     1.853599
dtype: float64
```

[129]:
```
with plt.style.context('dark_background'):
    graf_edad_kmeans.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DE LA EDAD POR KMEANS",weight='bold',color='red')
plt.text(-0.3, 200067,"25.41%",weight="bold",color="yellow")
plt.text(0.7, 254058,"32.26%",weight="bold",color="yellow")
plt.text(1.7,182762,"23.21%",weight="bold",color="yellow")
plt.text(2.7,98509,"12.51%",weight="bold",color="yellow")
plt.text(3.7,37502,"4.76%",weight="bold",color="yellow")
plt.text(4.7,14597,"1.85%",weight="bold",color="yellow")
```

```
plt.show()
```



PARA LA VARIABLE INGRESO BRUTO

[130]:
```
kmeans_ingreso=KBinsDiscretizer(n_bins=6,encode="ordinal",strategy="kmeans")
nuevo_dat_2_kmeans_ingreso=kmeans_ingreso.
 ↪fit_transform(dat_2[['INGRESO_BRUTO_M1']])
dat_2["INGRESO_BRUTO_M1_kmeans"]=nuevo_dat_2_kmeans_ingreso
dat_2["INGRESO_BRUTO_M1_quartil"]=dat_2["INGRESO_BRUTO_M1_kmeans"].astype(np.
 ↪int64)
dat_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 22 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   CODMES                    787495 non-null  int64
 1   TARGET_MODEL2             787495 non-null  int64
 2   EDAD                      787495 non-null  int64
 3   SEXO                      787495 non-null  object
 4   DEPARTAMENTO              787495 non-null  object
 5   FLG_CLIENTE               787495 non-null  object
 6   SEGMENTO                  787495 non-null  object
```

```
 7   FLG_ADEL_SUELDO_M1          787495 non-null  int64
 8   FREC_AGENTE                 787495 non-null  int64
 9   FREC_KIOSKO                 787495 non-null  int64
 10  FREC_BPI_TD                 787495 non-null  int64
 11  FREC_MON_TD                 787495 non-null  int64
 12  PROM_CTD_TRX_6M             787495 non-null  float64
 13  ANT_CLIENTE                 787495 non-null  float64
 14  CTD_RECLAMOS_M1             787495 non-null  int64
 15  INGRESO_BRUTO_M1            787495 non-null  float64
 16  EDAD_amplitud               787495 non-null  int64
 17  INGRESO_BRUTO_M1_amplitud   787495 non-null  int64
 18  EDAD_quartil                787495 non-null  int64
 19  INGRESO_BRUTO_M1_quartil    787495 non-null  int64
 20  EDAD_kmeans                 787495 non-null  float64
 21  INGRESO_BRUTO_M1_kmeans     787495 non-null  float64
dtypes: float64(5), int64(13), object(4)
memory usage: 132.2+ MB
```

Hallando la frecuencia del INGRESO BRUTO y su porcentaje

[131]:
```
graf_ingreso_kmeans=dat_2.groupby(dat_2.INGRESO_BRUTO_M1_kmeans).size()
graf_ingreso_kmeans
```

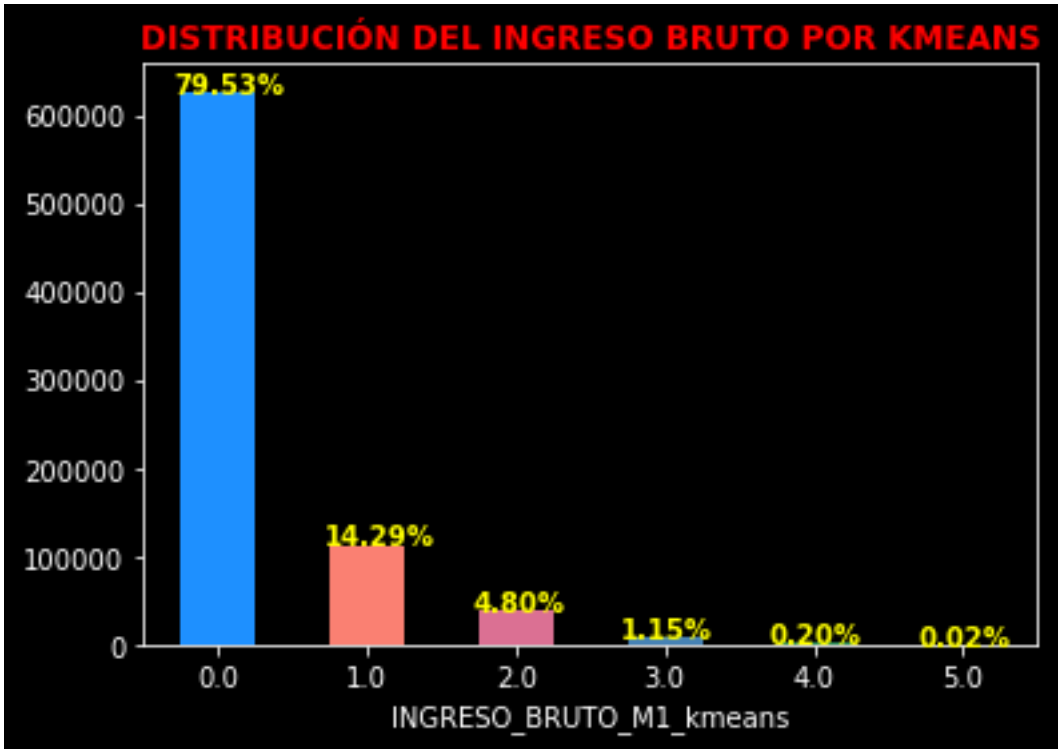
[131]:
```
INGRESO_BRUTO_M1_kmeans
0.0    626331
1.0    112544
2.0     37818
3.0      9092
4.0      1555
5.0       155
dtype: int64
```

[132]:
```
graf_ingreso_kmeans/sum(graf_ingreso_kmeans)*100
```

[132]:
```
INGRESO_BRUTO_M1_kmeans
0.0    79.534600
1.0    14.291392
2.0     4.802316
3.0     1.154547
4.0     0.197462
5.0     0.019683
dtype: float64
```

[133]:
```
with plt.style.context('dark_background'):
    graf_ingreso_kmeans.plot(kind="bar", rot=0,color=colors)
plt.title("DISTRIBUCIÓN DEL INGRESO BRUTO POR KMEANS",weight='bold',color='red')
plt.text(-0.3, 626331,"79.53%",weight="bold",color="yellow")
plt.text(0.7,112544,"14.29%",weight="bold",color="yellow")
```

```
plt.text(1.7,37818,"4.80%",weight="bold",color="yellow")
plt.text(2.7, 9092,"1.15%",weight="bold",color="yellow")
plt.text(3.7,1555,"0.20%",weight="bold",color="yellow")
plt.text(4.7,155 ,"0.02%",weight="bold",color="yellow")
plt.show()
```



Mostramos los resultados de la data dicretizadas en nuestra data original

[134]:
```
dat_2.head()
```

[134]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO	\
0	201411	0	46	F	PIURA	NO CLIENTE	2	
1	201411	0	54	M	LORETO	CLIENTE	1BC	
2	201411	0	81	M	LIMA	CLIENTE	6	
3	201411	0	42	M	PIURA	CLIENTE	2	
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC	

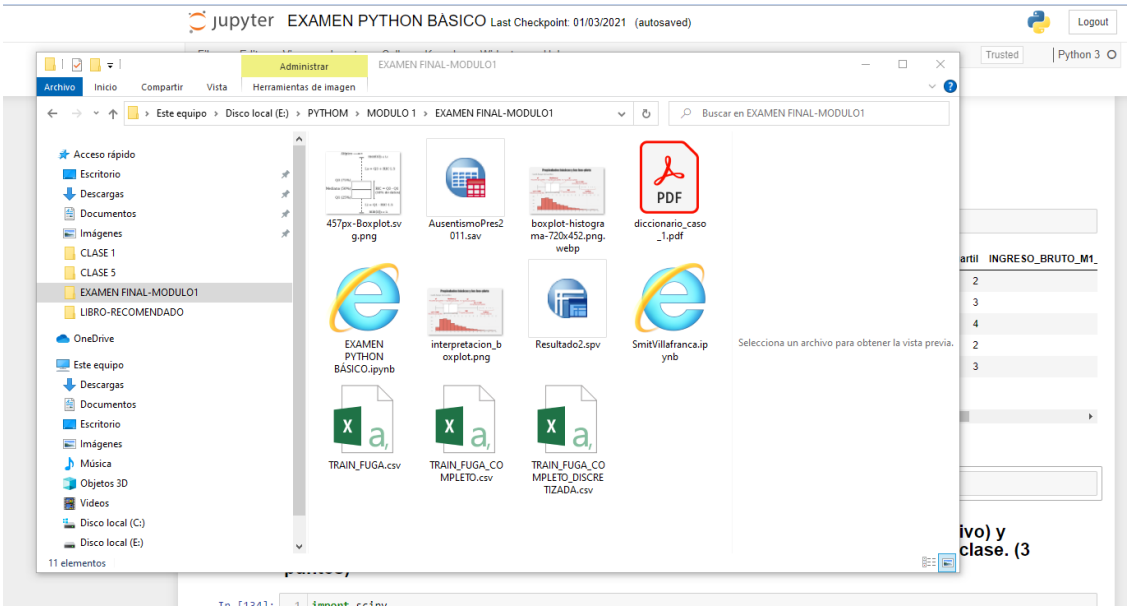
	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	...	PROM_CTD_TRX_6M	\
0	0	0	0	...	0.0	
1	0	0	0	...	0.0	
2	0	0	0	...	0.0	
3	0	0	0	...	0.0	
4	0	0	0	...	0.0	

```
   ANT_CLIENTE  CTD_RECLAMOS_M1  INGRESO_BRUTO_M1  EDAD_amplitud  \
0        224.0                0            2184.9              8
1        123.0                0            4718.0              9
2        264.0                0            2184.9             14
3        263.0                0             936.0              7
4        263.0                0            5844.0              9

   INGRESO_BRUTO_M1_amplitud  EDAD_quartil  INGRESO_BRUTO_M1_quartil  \
0                          0             2                         0
1                          0             3                         1
2                          0             4                         0
3                          0             2                         0
4                          0             3                         1

   EDAD_kmeans  INGRESO_BRUTO_M1_kmeans
0          2.0                      0.0
1          3.0                      1.0
2          4.0                      0.0
3          2.0                      0.0
4          3.0                      1.0

[5 rows x 22 columns]
```

Guardanos

[135]:
```
dat_2.to_csv("TRAIN_FUGA_COMPLETO_DISCRETIZADA.csv", index=False)
Image(filename='E:\PYTHOM\MODULO 1\EXAMEN FINAL-MODULO1/Captura de pantalla␣
 ↪2021-03-09 183235.png', width=900)
```

[135]:



1.5 4. Aplicar dos técnicas de balanceo de datos a nuestra variable TARGET (objetivo) y agregarlas a nuestro conjunto de datos original. Use los parámetros vistos en clase. (3 puntos)

[136]:
```
import scipy
import sklearn
import imblearn
```

[137]:
```
from imblearn.under_sampling import NearMiss
from imblearn.over_sampling import RandomOverSampler
from imblearn.combine import SMOTETomek
from imblearn.ensemble import BalancedBaggingClassifier
from sklearn.model_selection import train_test_split
import random
```

[138]:
```
archivo_csv="TRAIN_FUGA_COMPLETO.csv"
datos=pd.read_csv(archivo_csv,sep=",", encoding="ISO-8859-1")
datos.head(50)
```

[138]:

	CODMES	TARGET_MODEL2	EDAD	SEXO	DEPARTAMENTO	FLG_CLIENTE	SEGMENTO
0	201411	0	46	F	PIURA	NO CLIENTE	2
1	201411	0	54	M	LORETO	CLIENTE	1BC
2	201411	0	81	M	LIMA	CLIENTE	6
3	201411	0	42	M	PIURA	CLIENTE	2
4	201411	0	52	M	MOQUEGUA	CLIENTE	1BC
5	201411	0	74	M	LA LIBERTAD	CLIENTE	6
6	201411	0	66	M	LA LIBERTAD	CLIENTE	1BC
7	201411	0	57	M	LIMA	NO CLIENTE	2
8	201411	0	65	M	CALLAO	NO CLIENTE	2
9	201411	0	63	M	ANCASH	CLIENTE	2
10	201411	0	43	M	LIMA	CLIENTE	2
11	201411	0	64	F	LIMA	NO CLIENTE	2
12	201411	0	114	M	LIMA	CLIENTE	6
13	201411	0	48	F	LIMA	CLIENTE	2
14	201411	0	48	F	MOQUEGUA	CLIENTE	3
15	201411	0	114	F	LIMA	CLIENTE	6
16	201411	0	114	M	LIMA	CLIENTE	6
17	201411	0	42	F	LIMA	CLIENTE	3
18	201411	0	93	M	LIMA	NO CLIENTE	6
19	201411	0	114	F	LIMA	CLIENTE	6
20	201411	0	114	F	LIMA	CLIENTE	6
21	201411	0	39	M	LIMA	NO CLIENTE	3
22	201411	0	45	M	LIMA	CLIENTE	3
23	201411	0	50	M	LIMA	CLIENTE	3
24	201411	0	114	F	LIMA	CLIENTE	6
25	201411	0	114	M	LIMA	CLIENTE	6
26	201411	0	114	F	LIMA	CLIENTE	6

27	201411	0	114	M	LIMA		CLIENTE	6
28	201411	0	114	M	LIMA		CLIENTE	6
29	201411	0	114	M	LIMA		CLIENTE	6
30	201411	0	114	M	LIMA		CLIENTE	6
31	201411	0	114	M	LIMA		CLIENTE	6
32	201411	0	114	F	LIMA		CLIENTE	6
33	201411	0	52	F	AREQUIPA		CLIENTE	2
34	201411	0	57	M	LIMA		CLIENTE	3
35	201411	0	114	F	LIMA		CLIENTE	6
36	201411	0	48	M	LIMA	NO	CLIENTE	2
37	201411	0	57	F	LIMA	NO	CLIENTE	1BC
38	201411	0	52	M	CALLAO	NO	CLIENTE	3
39	201411	0	61	F	ANCASH		CLIENTE	5
40	201411	0	48	M	LIMA		CLIENTE	3
41	201411	0	46	M	CALLAO	NO	CLIENTE	6
42	201411	0	45	F	LIMA	NO	CLIENTE	2
43	201411	0	54	F	LIMA	NO	CLIENTE	6
44	201411	0	44	M	LIMA		CLIENTE	3
45	201411	0	42	M	LIMA		CLIENTE	3
46	201411	0	44	F	LIMA	NO	CLIENTE	6
47	201411	0	53	M	LIMA		CLIENTE	6
48	201411	0	47	F	LIMA		CLIENTE	3
49	201411	0	37	M	LIMA	NO	CLIENTE	6

	FLG_ADEL_SUELDO_M1	FREC_AGENTE	FREC_KIOSKO	FREC_BPI_TD	FREC_MON_TD	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
5	0	0	0	0	0	
6	0	0	0	0	6	
7	0	0	5	6	6	
8	0	0	0	0	0	
9	0	0	0	0	0	
10	0	0	0	0	0	
11	0	0	0	0	0	
12	0	0	0	0	0	
13	0	0	0	0	0	
14	0	0	0	0	0	
15	0	0	0	0	0	
16	0	0	0	0	0	
17	0	0	0	0	0	
18	0	0	0	0	0	
19	0	0	0	0	0	
20	0	0	0	0	0	
21	0	0	0	0	0	

22	0	0	0	0	0
23	0	0	0	0	0
24	0	0	0	0	0
25	0	0	0	0	0
26	0	0	0	0	0
27	0	0	0	0	0
28	0	0	0	0	0
29	0	0	0	0	0
30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	2	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	3	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0

	PROM_CTD_TRX_6M	ANT_CLIENTE	CTD_RECLAMOS_M1	INGRESO_BRUTO_M1
0	0.000000	224.0	0	2184.9
1	0.000000	123.0	0	4718.0
2	0.000000	264.0	0	2184.9
3	0.000000	263.0	0	936.0
4	0.000000	263.0	0	5844.0
5	0.000000	256.0	0	2184.9
6	0.000000	85.0	0	4232.0
7	2.166667	151.0	0	1580.0
8	3.333333	778.0	0	3081.5
9	0.000000	272.0	0	936.0
10	0.000000	11.0	0	1421.0
11	0.000000	21.0	0	2184.9
12	0.000000	281.0	0	2184.9
13	0.000000	209.0	0	809.0
14	0.000000	208.0	0	739.0
15	0.000000	233.0	0	2184.9
16	0.000000	216.0	0	2184.9

17	0.000000	233.0	0	739.0
18	0.000000	281.0	0	2184.9
19	0.000000	263.0	0	2184.9
20	0.000000	221.0	0	2184.9
21	0.000000	163.0	0	749.0
22	0.000000	215.0	0	936.0
23	0.000000	257.0	0	936.0
24	0.000000	206.0	0	2184.9
25	0.000000	281.0	0	2184.9
26	0.000000	280.0	0	2184.9
27	0.000000	220.0	0	2184.9
28	0.000000	275.0	0	2184.9
29	0.000000	263.0	0	2184.9
30	0.000000	280.0	0	2184.9
31	0.000000	263.0	0	2184.9
32	0.000000	203.0	0	2184.9
33	0.000000	202.0	0	739.0
34	0.000000	264.0	0	858.0
35	0.000000	263.0	0	2184.9
36	0.000000	265.0	0	2184.9
37	0.000000	778.0	0	3962.0
38	0.000000	85.0	0	2184.9
39	0.000000	257.0	0	739.0
40	0.000000	191.0	0	858.0
41	0.000000	162.0	0	2184.9
42	0.000000	185.0	0	2184.9
43	0.000000	159.0	0	2184.9
44	0.000000	280.0	0	818.0
45	0.333333	76.0	0	2777.0
46	0.000000	778.0	0	2184.9
47	0.000000	280.0	0	2184.9
48	0.000000	164.0	0	1093.0
49	0.000000	209.0	0	2184.9

Sacando las columnas analizar

[139]:
```
dat_analisis = datos[['EDAD','SEXO','INGRESO_BRUTO_M1','FREC_BPI_TD','FLG_ADEL_SUELDO_M1',
'PROM_CTD_TRX_6M','FREC_AGENTE','FREC_KIOSKO','ANT_CLIENTE','CTD_RECLAMOS_M1']]
dat_analisis.head()
```

[139]:

	EDAD	SEXO	INGRESO_BRUTO_M1	FREC_BPI_TD	FLG_ADEL_SUELDO_M1	\
0	46	F	2184.9	0	0	
1	54	M	4718.0	0	0	
2	81	M	2184.9	0	0	
3	42	M	936.0	0	0	

```
4    52    M            5844.0            0                   0

   PROM_CTD_TRX_6M  FREC_AGENTE  FREC_KIOSKO  ANT_CLIENTE  CTD_RECLAMOS_M1
0              0.0            0            0        224.0                0
1              0.0            0            0        123.0                0
2              0.0            0            0        264.0                0
3              0.0            0            0        263.0                0
4              0.0            0            0        263.0                0
```

[140]:
```
dat_analisis.shape
```

[140]:
```
(787495, 10)
```

Vizualicemos los tipos de variable

[141]:
```
dat_analisis.dtypes
```

[141]:
```
EDAD                  int64
SEXO                 object
INGRESO_BRUTO_M1    float64
FREC_BPI_TD           int64
FLG_ADEL_SUELDO_M1    int64
PROM_CTD_TRX_6M     float64
FREC_AGENTE           int64
FREC_KIOSKO           int64
ANT_CLIENTE         float64
CTD_RECLAMOS_M1       int64
dtype: object
```

Transformemos la variable SEXO que es de tipo object a tipo float

[142]:
```
dat_analisis["SEXO"]=dat_analisis["SEXO"].replace("F","0")
dat_analisis["SEXO"]=dat_analisis["SEXO"].replace("M","1")
dat_analisis["SEXO"]=dat_analisis["SEXO"].astype(float)
```

Mostramos los resultados

[143]:
```
dat_analisis.dtypes
```

[143]:
```
EDAD                  int64
SEXO                float64
INGRESO_BRUTO_M1    float64
FREC_BPI_TD           int64
FLG_ADEL_SUELDO_M1    int64
PROM_CTD_TRX_6M     float64
FREC_AGENTE           int64
FREC_KIOSKO           int64
ANT_CLIENTE         float64
```

```
CTD_RECLAMOS_M1          int64
dtype: object
```

Agrupando columnas por tipo de datos

[144]:
```
dat_analisis.columns.to_series().groupby(dat_analisis.dtypes).groups
```

[144]:
```
{int64: ['EDAD', 'FREC_BPI_TD', 'FLG_ADEL_SUELDO_M1', 'FREC_AGENTE',
'FREC_KIOSKO', 'CTD_RECLAMOS_M1'], float64: ['SEXO', 'INGRESO_BRUTO_M1',
'PROM_CTD_TRX_6M', 'ANT_CLIENTE']}
```

[145]:
```
tipos = dat_analisis.columns.to_series().groupby(dat_analisis.dtypes).groups
```

Armando lista de columnas categóricas

[146]:
```
entero = tipos[np.dtype("int64")]
print("Los números de tipo entero son:\n",entero )

flotante = tipos[np.dtype("float")]
print("Los números de tipo flotante son:\n",flotante )
```

```
Los números de tipo entero son:
 Index(['EDAD', 'FREC_BPI_TD', 'FLG_ADEL_SUELDO_M1', 'FREC_AGENTE',
       'FREC_KIOSKO', 'CTD_RECLAMOS_M1'],
      dtype='object')
Los números de tipo flotante son:
 Index(['SEXO', 'INGRESO_BRUTO_M1', 'PROM_CTD_TRX_6M', 'ANT_CLIENTE'],
dtype='object')
```

Transformamos los enteros a flotantes

[147]:
```
for c in entero:
    dat_analisis[c]=dat_analisis[c].astype(float)
```

[148]:
```
dat_analisis.dtypes
```

[148]:
```
EDAD                 float64
SEXO                 float64
INGRESO_BRUTO_M1     float64
FREC_BPI_TD          float64
FLG_ADEL_SUELDO_M1   float64
PROM_CTD_TRX_6M      float64
FREC_AGENTE          float64
FREC_KIOSKO          float64
ANT_CLIENTE          float64
CTD_RECLAMOS_M1      float64
dtype: object
```

Concatenamos dat_analisis y datos['TARGET_MODEL2']

[149]:
```
dat_analisis_2 = pd.concat([dat_analisis, datos['TARGET_MODEL2']], axis=1)
dat_analisis_2.head()
```

[149]:

	EDAD	SEXO	INGRESO_BRUTO_M1	FREC_BPI_TD	FLG_ADEL_SUELDO_M1	\
0	46.0	0.0	2184.9	0.0	0.0	
1	54.0	1.0	4718.0	0.0	0.0	
2	81.0	1.0	2184.9	0.0	0.0	
3	42.0	1.0	936.0	0.0	0.0	
4	52.0	1.0	5844.0	0.0	0.0	

	PROM_CTD_TRX_6M	FREC_AGENTE	FREC_KIOSKO	ANT_CLIENTE	CTD_RECLAMOS_M1	\
0	0.0	0.0	0.0	224.0	0.0	
1	0.0	0.0	0.0	123.0	0.0	
2	0.0	0.0	0.0	264.0	0.0	
3	0.0	0.0	0.0	263.0	0.0	
4	0.0	0.0	0.0	263.0	0.0	

	TARGET_MODEL2
0	0
1	0
2	0
3	0
4	0

Visualizamos los tipo

[150]:
```
dat_analisis_2.dtypes
```

[150]:
```
EDAD                 float64
SEXO                 float64
INGRESO_BRUTO_M1     float64
FREC_BPI_TD          float64
FLG_ADEL_SUELDO_M1   float64
PROM_CTD_TRX_6M      float64
FREC_AGENTE          float64
FREC_KIOSKO          float64
ANT_CLIENTE          float64
CTD_RECLAMOS_M1      float64
TARGET_MODEL2          int64
dtype: object
```

Hallemos las frecuencias de la variable TARGET_MODEL2

[151]:
```
frec_datos=pd.value_counts(dat_analisis_2.TARGET_MODEL2,sort=True)
frec_datos
```

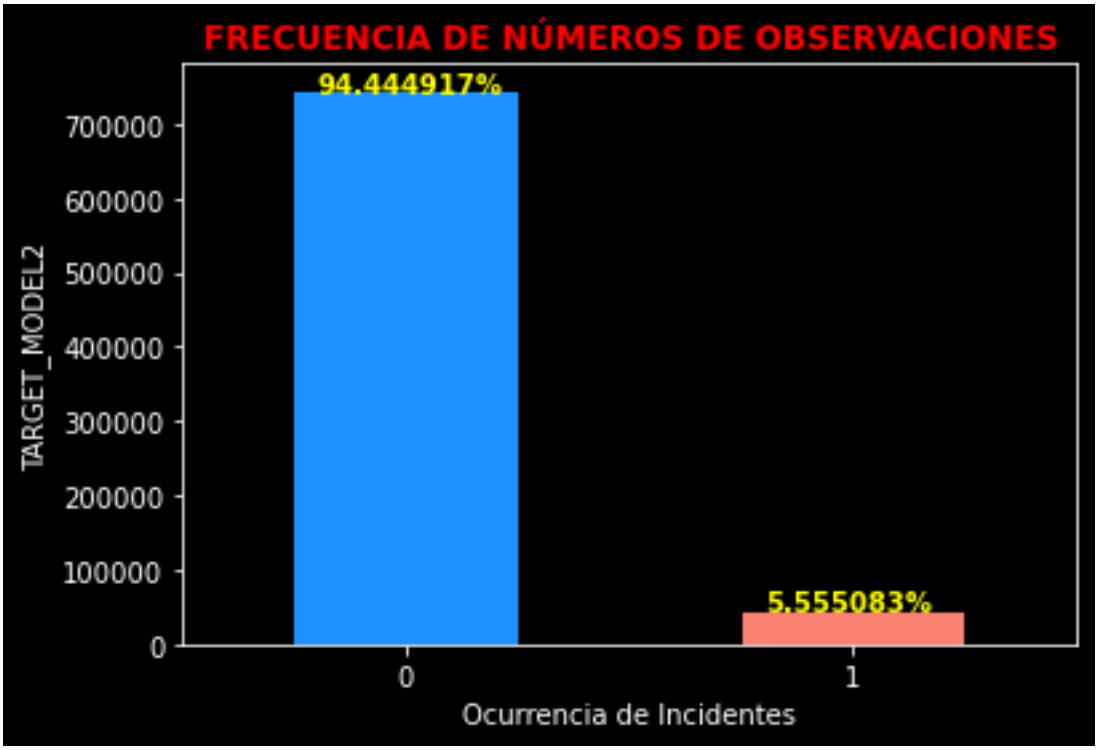
[151]:
```
0    743749
1     43746
```

```
Name: TARGET_MODEL2, dtype: int64
```

[152]:
```
frec_datos/sum(frec_datos)*100
```

[152]:
```
0    94.444917
1     5.555083
Name: TARGET_MODEL2, dtype: float64
```

[153]:
```
with plt.style.context('dark_background'):
    pd.value_counts(dat_analisis_2.TARGET_MODEL2).
 ↪plot(kind='bar',rot=0,color=colors)
plt.title("FRECUENCIA DE NÚMEROS DE OBSERVACIONES",weight='bold',color='red')
plt.xlabel("Ocurrencia de Incidentes")
plt.ylabel("TARGET_MODEL2")
plt.text(-0.2,743749,"94.444917%",weight="bold",color="yellow")
plt.text(0.8,43746,"5.555083%",weight="bold",color="yellow")
plt.show()
```



[154]:
```
dat_analisis_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 787495 entries, 0 to 787494
Data columns (total 11 columns):
 #   Column              Non-Null Count   Dtype
```

```
---  ------                --------------   -----
 0   EDAD                  787495 non-null  float64
 1   SEXO                  787495 non-null  float64
 2   INGRESO_BRUTO_M1      787495 non-null  float64
 3   FREC_BPI_TD           787495 non-null  float64
 4   FLG_ADEL_SUELDO_M1    787495 non-null  float64
 5   PROM_CTD_TRX_6M       787495 non-null  float64
 6   FREC_AGENTE           787495 non-null  float64
 7   FREC_KIOSKO           787495 non-null  float64
 8   ANT_CLIENTE           787495 non-null  float64
 9   CTD_RECLAMOS_M1       787495 non-null  float64
 10  TARGET_MODEL2         787495 non-null  int64
dtypes: float64(10), int64(1)
memory usage: 66.1 MB
```

Particionamiento de datos

[155]:
```
X, y = dat_analisis_2.iloc[:, 0:10].values, dat_analisis_2.iloc[:,10].values

X_train, X_test, y_train, y_test =\
    train_test_split(X, #valores de los predictores
                     y, #los valores del target
                     test_size=0.4, #proporción para datos de testeo
                     random_state=2021, #semilla
                     stratify=y) #la variable de estratificación
```

Datos de entrenamiento

[156]:
```
x_t= pd.DataFrame(X_train,␣
 ↪columns=['EDAD','SEXO','INGRESO_BRUTO_M1','FREC_BPI_TD','FLG_ADEL_SUELDO_M1',
          ␣
 ↪'PROM_CTD_TRX_6M','FREC_AGENTE','FREC_KIOSKO','ANT_CLIENTE','CTD_RECLAMOS_M1'])
y_t= pd.DataFrame(y_train, columns=['TARGET_MODEL2'])

dat_analisis_2_entrenados = pd.concat([x_t, y_t], axis=1)
dat_analisis_2_entrenados.head()
```

[156]:

	EDAD	SEXO	INGRESO_BRUTO_M1	FREC_BPI_TD	FLG_ADEL_SUELDO_M1	\
0	30.0	0.0	1191.0	0.0	0.0	
1	48.0	1.0	1880.0	0.0	0.0	
2	61.0	1.0	1728.0	0.0	0.0	
3	41.0	1.0	2184.9	0.0	0.0	
4	29.0	1.0	1004.0	0.0	0.0	

	PROM_CTD_TRX_6M	FREC_AGENTE	FREC_KIOSKO	ANT_CLIENTE	CTD_RECLAMOS_M1	\
0	0.000000	0.0	5.0	71.0	0.0	
1	0.166667	0.0	0.0	782.0	0.0	
2	0.000000	0.0	6.0	81.0	0.0	

```
3         0.000000          0.0          3.0        162.0              0.0
4         0.000000          0.0          0.0         96.0              0.0

   TARGET_MODEL2
0              0
1              0
2              0
3              0
4              0
```

[157]:
```
dat_analisis_2_entrenados.shape
```

[157]:
```
(472497, 11)
```

Hallemos las frecuencias de la variable TARGET_MODEL2 ya entrenada

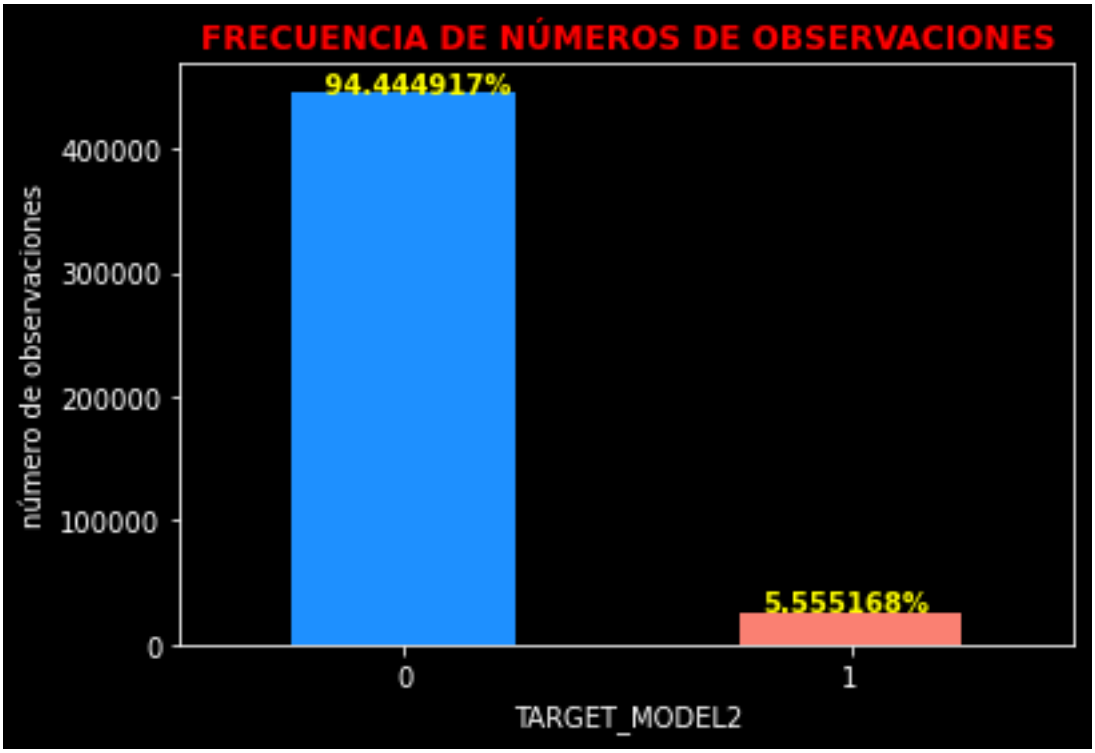
[158]:
```
count_classes=pd.value_counts(dat_analisis_2_entrenados.TARGET_MODEL2,sort=True)
count_classes
```

[158]:
```
0    446249
1     26248
Name: TARGET_MODEL2, dtype: int64
```

[159]:
```
count_classes/sum(count_classes)*100
```

[159]:
```
0    94.444832
1     5.555168
Name: TARGET_MODEL2, dtype: float64
```

[160]:
```
with plt.style.context('dark_background'):
    count_classes.plot(kind = 'bar',rot=0,color=colors)
plt.title("FRECUENCIA DE NÚMEROS DE OBSERVACIONES",weight='bold',color='red')
plt.xlabel("TARGET_MODEL2")
plt.ylabel("número de observaciones")
plt.text(-0.2,446249," 94.444917%",weight="bold",color="yellow")
plt.text(0.8,26248,"5.555168%",weight="bold",color="yellow")
plt.show()
```



[161]:
```
dat_analisis_2_entrenados.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 472497 entries, 0 to 472496
Data columns (total 11 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   EDAD                472497 non-null  float64
 1   SEXO                472497 non-null  float64
 2   INGRESO_BRUTO_M1    472497 non-null  float64
 3   FREC_BPI_TD         472497 non-null  float64
 4   FLG_ADEL_SUELDO_M1  472497 non-null  float64
 5   PROM_CTD_TRX_6M     472497 non-null  float64
 6   FREC_AGENTE         472497 non-null  float64
 7   FREC_KIOSKO         472497 non-null  float64
 8   ANT_CLIENTE         472497 non-null  float64
 9   CTD_RECLAMOS_M1     472497 non-null  float64
 10  TARGET_MODEL2       472497 non-null  int64  
dtypes: float64(10), int64(1)
memory usage: 39.7 MB
```

1.5.1 4.1. OVERSAMPLING

[162]:
```
pip install tensorflow
```

```
Note: you may need to restart the kernel to use updated packages.Collecting
tensorflow

  Using cached tensorflow-2.4.1-cp38-cp38-win_amd64.whl (370.7 MB)
Requirement already satisfied: keras-preprocessing~=1.1.2 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.1.2)
Requirement already satisfied: google-pasta~=0.2 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: flatbuffers~=1.12.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.12)
Requirement already satisfied: h5py~=2.10.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (2.10.0)

ERROR: Could not install packages due to an EnvironmentError: [WinError 5]
Acceso denegado: 'c:\\programdata\\anaconda3\\lib\\site-
packages\\wrapt-1.11.2.dist-info\\INSTALLER'
Consider using the `--user` option or check the permissions.


Requirement already satisfied: termcolor~=1.1.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.1.0)
Requirement already satisfied: numpy~=1.19.2 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.19.2)
Requirement already satisfied: gast==0.3.3 in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (0.3.3)
Requirement already satisfied: tensorboard~=2.4 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (2.4.1)
Requirement already satisfied: typing-extensions~=3.7.4 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.7.4.3)
Requirement already satisfied: grpcio~=1.32.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.32.0)
Requirement already satisfied: tensorflow-estimator<2.5.0,>=2.4.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (2.4.0)
Requirement already satisfied: protobuf>=3.9.2 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.15.5)
Processing c:\users\user\appdata\local\pip\cache\wheels\5f\fd\9e\b6cf5890494cb8e
f0b5eaff72e5d55a70fb56316007d6dfe73\wrapt-1.12.1-py3-none-any.whl
Requirement already satisfied: six~=1.15.0 in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (1.15.0)
Requirement already satisfied: opt-einsum~=3.3.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.3.0)
Requirement already satisfied: absl-py~=0.10 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (0.11.0)
Requirement already satisfied: wheel~=0.35 in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (0.35.1)
```

```
Requirement already satisfied: astunparse~=1.6.3 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.6.3)
Requirement already satisfied: requests<3,>=2.21.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(2.24.0)
Requirement already satisfied: setuptools>=41.0.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(50.3.1.post20201107)
Requirement already satisfied: werkzeug>=0.11.15 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(1.0.1)
Requirement already satisfied: google-auth<2,>=1.6.3 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(1.27.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(1.8.0)
Requirement already satisfied: markdown>=2.6.8 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(3.3.4)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
c:\programdata\anaconda3\lib\site-packages (from tensorboard~=2.4->tensorflow)
(0.4.3)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (1.25.11)
Requirement already satisfied: certifi>=2017.4.17 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (2020.6.20)
Requirement already satisfied: idna<3,>=2.5 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorboard~=2.4->tensorflow) (3.0.4)
Requirement already satisfied: rsa<5,>=3.1.4; python_version >= "3.6" in
c:\programdata\anaconda3\lib\site-packages (from google-
auth<2,>=1.6.3->tensorboard~=2.4->tensorflow) (4.7.2)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
c:\programdata\anaconda3\lib\site-packages (from google-
auth<2,>=1.6.3->tensorboard~=2.4->tensorflow) (4.2.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
c:\programdata\anaconda3\lib\site-packages (from google-
auth<2,>=1.6.3->tensorboard~=2.4->tensorflow) (0.2.8)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
c:\programdata\anaconda3\lib\site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard~=2.4->tensorflow) (1.3.0)
Requirement already satisfied: pyasn1>=0.1.3 in
```

```
c:\programdata\anaconda3\lib\site-packages (from rsa<5,>=3.1.4; python_version
>= "3.6"->google-auth<2,>=1.6.3->tensorboard~=2.4->tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in
c:\programdata\anaconda3\lib\site-packages (from requests-
oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard~=2.4->tensorflow)
(3.1.0)
Installing collected packages: wrapt, tensorflow
  Attempting uninstall: wrapt
    Found existing installation: wrapt 1.11.2
    Uninstalling wrapt-1.11.2:
```

[163]:
```
!pip install --user imbalanced-learn
```

```
Requirement already satisfied: imbalanced-learn in
c:\programdata\anaconda3\lib\site-packages (0.7.0)
Requirement already satisfied: scipy>=0.19.1 in
c:\programdata\anaconda3\lib\site-packages (from imbalanced-learn) (1.5.2)
Requirement already satisfied: scikit-learn>=0.23 in
c:\programdata\anaconda3\lib\site-packages (from imbalanced-learn) (0.23.2)
Requirement already satisfied: joblib>=0.11 in
c:\programdata\anaconda3\lib\site-packages (from imbalanced-learn) (0.17.0)
Requirement already satisfied: numpy>=1.13.3 in
c:\programdata\anaconda3\lib\site-packages (from imbalanced-learn) (1.19.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\programdata\anaconda3\lib\site-packages (from scikit-learn>=0.23->imbalanced-
learn) (2.1.0)
```

[164]:
```
pip install --user git+https://github.com/scikit-learn-contrib/imbalanced-learn.
↪git
```

```
Collecting git+https://github.com/scikit-learn-contrib/imbalanced-learn.gitNote:
you may need to restart the kernel to use updated packages.
  Cloning https://github.com/scikit-learn-contrib/imbalanced-learn.git to
c:\users\user\appdata\local\temp\pip-req-build-1bs40wrf


  ERROR: Error [WinError 2] El sistema no puede encontrar el archivo
especificado while executing command git clone -q https://github.com/scikit-
learn-contrib/imbalanced-learn.git 'C:\Users\user\AppData\Local\Temp\pip-req-
build-1bs40wrf'
ERROR: Cannot find command 'git' - do you have 'git' installed and in your PATH?
```

[165]:
```
over=RandomOverSampler(sampling_strategy=0.7,random_state=2021)
```

[166]:
```
Xtrain_over, ytrain_over =over.fit_resample(x_t, y_t)
```

[167]:
```
dat_analisis_2_entrenados_over=pd.concat([Xtrain_over,ytrain_over],axis=1)
dat_analisis_2_entrenados_over
```

[167]:

	EDAD	SEXO	INGRESO_BRUTO_M1	FREC_BPI_TD	FLG_ADEL_SUELDO_M1	\
0	30.0	0.0	1191.0	0.0	0.0	
1	48.0	1.0	1880.0	0.0	0.0	
2	61.0	1.0	1728.0	0.0	0.0	
3	41.0	1.0	2184.9	0.0	0.0	
4	29.0	1.0	1004.0	0.0	0.0	
...	...	...	...	...	...	
758618	24.0	1.0	15405.0	1.0	0.0	
758619	35.0	0.0	2184.9	0.0	0.0	
758620	20.0	1.0	1535.0	0.0	0.0	
758621	43.0	1.0	35430.0	6.0	0.0	
758622	23.0	1.0	1689.0	1.0	1.0	

	PROM_CTD_TRX_6M	FREC_AGENTE	FREC_KIOSKO	ANT_CLIENTE	\
0	0.000000	0.0	5.0	71.0	
1	0.166667	0.0	0.0	782.0	
2	0.000000	0.0	6.0	81.0	
3	0.000000	0.0	3.0	162.0	
4	0.000000	0.0	0.0	96.0	
...	...	...	...	...	
758618	0.000000	0.0	0.0	59.0	
758619	0.000000	2.0	0.0	4.0	
758620	0.000000	0.0	6.0	6.0	
758621	37.666667	0.0	0.0	174.0	
758622	0.000000	0.0	0.0	34.0	

	CTD_RECLAMOS_M1	TARGET_MODEL2
0	0.0	0
1	0.0	0
2	0.0	0
3	0.0	0
4	0.0	0
...	...	...
758618	0.0	1
758619	0.0	1
758620	0.0	1
758621	0.0	1
758622	0.0	1

[758623 rows x 11 columns]

[168]:
```
count_classes_over = pd.
 ↪value_counts(dat_analisis_2_entrenados_over['TARGET_MODEL2'], sort = True)
count_classes_over
```

[168]:
```
0    446249
1    312374
```

```
Name: TARGET_MODEL2, dtype: int64
```

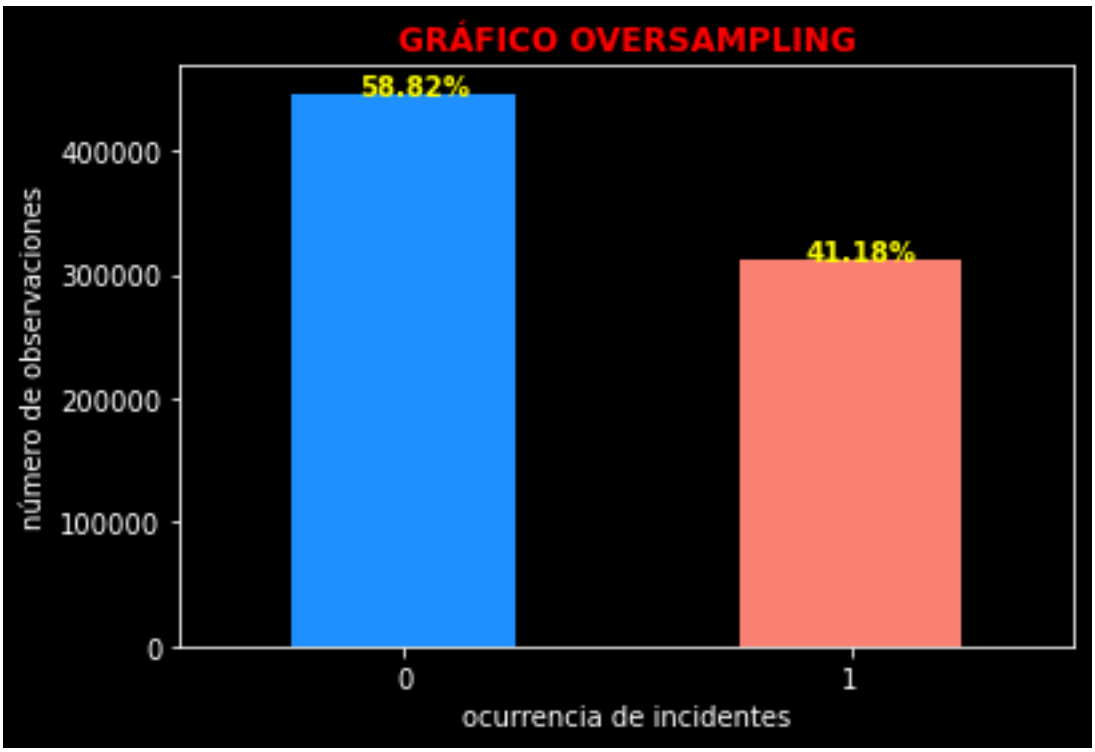
[169]:
```
count_classes_over/sum(count_classes_over)*100
```

[169]:
```
0    58.823553
1    41.176447
Name: TARGET_MODEL2, dtype: float64
```

[170]:
```
prop=round(count_classes_over[1]*100/count_classes_over[0],1)
prop
```

[170]:
```
70.0
```

[171]:
```
with plt.style.context('dark_background'):
    count_classes_over.plot(kind = 'bar',
                    rot=0,color=colors)
plt.xticks(range(2))
plt.title("GRÁFICO OVERSAMPLING",weight="bold",color="red")
plt.xlabel("ocurrencia de incidentes")
plt.ylabel("número de observaciones")
plt.text(-0.1,446249,"58.82%",weight="bold",color="yellow")
plt.text(0.9,312374,"41.18%",weight="bold",color="yellow")
plt.show()
```



1.5.2 4.2. SMOTETomek

[172]:
```
Smot=SMOTETomek(sampling_strategy=0.7,random_state=2021)
```

[173]:
```
Xtrain_Smot, ytrain_Smot =Smot.fit_resample(x_t, y_t)
```

[174]:
```
dat_analisis_2_entrenamiento_Smot=pd.concat([Xtrain_Smot,ytrain_Smot],axis=1)
dat_analisis_2_entrenamiento_Smot
```

[174]:

	EDAD	SEXO	INGRESO_BRUTO_M1	FREC_BPI_TD	FLG_ADEL_SUELDO_M1	\
0	30.000000	0.0	1191.000000	0.000000	0.000000	
1	48.000000	1.0	1880.000000	0.000000	0.000000	
2	61.000000	1.0	1728.000000	0.000000	0.000000	
3	41.000000	1.0	2184.900000	0.000000	0.000000	
4	29.000000	1.0	1004.000000	0.000000	0.000000	
...	...	...	...	...	...	
747184	21.890811	0.0	1175.187749	0.000000	0.000000	
747185	37.613014	1.0	6901.803082	0.277397	0.000000	
747186	30.109053	1.0	969.630316	0.000000	0.000000	
747187	55.000000	0.0	1266.002125	0.000000	0.002125	
747188	23.434933	1.0	945.000000	0.000000	0.000000	

	PROM_CTD_TRX_6M	FREC_AGENTE	FREC_KIOSKO	ANT_CLIENTE	\
0	0.000000	0.000000	5.000000	71.000000	
1	0.166667	0.000000	0.000000	782.000000	
2	0.000000	0.000000	6.000000	81.000000	
3	0.000000	0.000000	3.000000	162.000000	
4	0.000000	0.000000	0.000000	96.000000	
...	...	...	...	...	
747184	0.000000	3.515314	0.296937	28.000000	
747185	0.000000	0.000000	0.000000	31.058220	
747186	0.000000	0.630316	0.630316	4.260631	
747187	0.000000	0.008498	0.006374	40.036117	
747188	0.000000	0.000000	1.695202	37.869865	

	CTD_RECLAMOS_M1	TARGET_MODEL2
0	0.0	0
1	0.0	0
2	0.0	0
3	0.0	0
4	0.0	0
...	...	...
747184	0.0	1
747185	0.0	1
747186	0.0	1
747187	0.0	1
747188	0.0	1

```
[747189 rows x 11 columns]
```

[175]:
```
count_classes_Smot = pd.
 ↪value_counts(dat_analisis_2_entrenamiento_Smot['TARGET_MODEL2'], sort = True)
count_classes_Smot
```

[175]:
```
0    440532
1    306657
Name: TARGET_MODEL2, dtype: int64
```

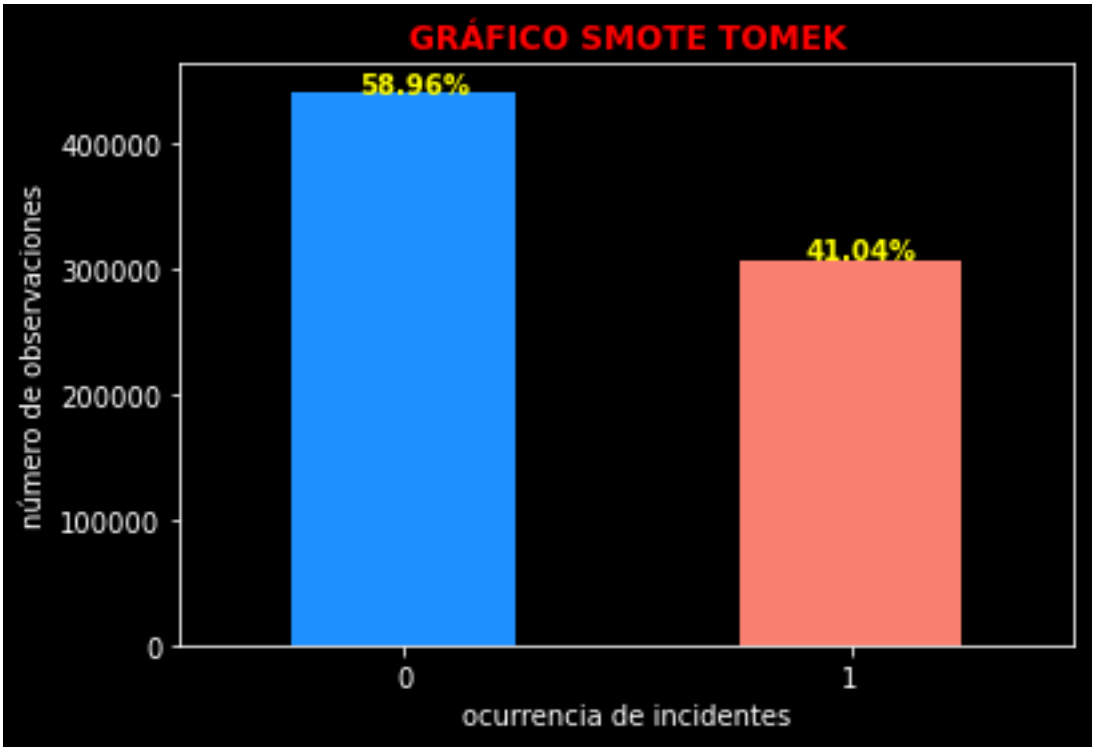
[176]:
```
count_classes_Smot/sum(count_classes_Smot)*100
```

[176]:
```
0    58.958577
1    41.041423
Name: TARGET_MODEL2, dtype: float64
```

[177]:
```
prop=round(count_classes_Smot[1]*100/count_classes_Smot[0],1)
prop
```

[177]:
```
69.6
```

[178]:
```
with plt.style.context('dark_background'):
    count_classes_Smot.plot(kind = 'bar',#bar: gráfico de barras
                    rot=0,color =colors)#0 = no rotación de las etiquetas del eje␣
 ↪x
plt.xticks(range(2))
plt.title("GRÁFICO SMOTE TOMEK",weight="bold",color="red")
plt.xlabel("ocurrencia de incidentes")
plt.ylabel("número de observaciones")
plt.text(-0.1,440532,"58.96%",weight="bold",color="yellow")
plt.text(0.9,306657,"41.04%",weight="bold",color="yellow")
plt.show()
```



2 CASO 2:

La presente aplicación captura datos socioeconómicos a nivel distrital para la realización de un ejemplo de reducción de dimensiones haciendo uso del análisis de componentes principales y factorial. Las variables a reducir son: porcentaje de hogares sin medios de comunicación (porc_hogares_sin_medios), porcentaje de alfabetismo (alfabetismo), porcentaje de hogares con 2 o más necesidades básicas incubiertas (porc_2_nbi), índice de desarrollo humano (IDH) y el coeficiente de desigualdad de GINI (GINI). 1. Realizar un análisis de componentes principales para reducción de la dimensionalidad (4 puntos) 2. Realizar un análisis factorial para reducción de la dimensionalidad (4 puntos)

Las librerias para Los Componentes Principales son:

[179]:
```
import scipy.stats as stats#Para calculo de probabilidades estadisticos
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA#Para descomposición de varianza en el PCA
from sklearn.preprocessing import MinMaxScaler#Para la normalización de dato
```

Importamos nuestro base de datos

[180]:
```
os.chdir("E:\PYTHOM\MODULO 1\EXAMEN FINAL-MODULO1")#direccionando la ruta
archivo_spss="AusentismoPres2011.sav"
```

```
df=pd.read_spss(path=archivo_spss)
df.head()
```

[180]:

	ubigeo	departamento	dom_Geo	provincia	distrito	total_electoral	total_ausentismo	porc_Ausentismo	ord_Ausentismo	dic_Ausentismo	porc_hogares_sin_medios	IDH	alfabetismo	partidoGanador	porc_2_NBI	GINI
0	010102	Amazonas	Norte	Chachapoyas	Asunción	234.0	59.0	25.213675	Ausentismo Grave	Ausentismo Fuerte	100.00	0.581463	86.893200	PERU POSIBLE	51.11	0.30
1	010103	Amazonas	Norte	Chachapoyas	Balsas	848.0	208.0	24.528301	Ausentismo Grave	Ausentismo Fuerte	94.87	0.562141	86.527290	FUERZA 2011	20.23	0.31
2	010104	Amazonas	Norte	Chachapoyas	Cheto	478.0	51.0	10.669457	Ausentismo Bajo	Ausentismo Leve	99.40	0.599150	92.838196	GANA PERU	16.87	0.28
3	010105	Amazonas	Norte	Chachapoyas	Chiliquín	638.0	197.0	30.877743	Ausentismo Grave	Ausentismo Fuerte	99.60	0.545484	86.541740	FUERZA 2011	31.73	0.29
4	010106	Amazonas	Norte	Chachapoyas	Chuquibamba	1161.0	333.0	28.682170	Ausentismo Grave	Ausentismo Fuerte	99.62	0.584659	92.598430	FUERZA 2011	17.55	0.31

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1833 entries, 0 to 1832
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ubigeo                  1833 non-null   object
 1   departamento            1833 non-null   object
 2   dom_Geo                 1833 non-null   object
 3   provincia               1833 non-null   object
 4   distrito                1833 non-null   object
 5   total_electoral         1833 non-null   float64
 6   total_ausentismo        1833 non-null   float64
 7   porc_Ausentismo         1833 non-null   float64
 8   ord_Ausentismo          1833 non-null   category
```

```
 9   dic_Ausentismo           1833 non-null   category
 10  porc_hogares_sin_medios  1832 non-null   float64
 11  IDH                      1833 non-null   float64
 12  alfabetismo              1833 non-null   float64
 13  partidoGanador           1833 non-null   object
 14  porc_2_NBI               1832 non-null   float64
 15  GINI                     1832 non-null   float64
dtypes: category(2), float64(8), object(6)
memory usage: 204.5+ KB
```

hacemos una pequeña limpieza de datos

[182]:
```
data=df.dropna(subset=['GINI'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1832 entries, 0 to 1832
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   ubigeo                   1832 non-null   object
 1   departamento             1832 non-null   object
 2   dom_Geo                  1832 non-null   object
 3   provincia                1832 non-null   object
 4   distrito                 1832 non-null   object
 5   total_electoral          1832 non-null   float64
 6   total_ausentismo         1832 non-null   float64
 7   porc_Ausentismo          1832 non-null   float64
 8   ord_Ausentismo           1832 non-null   category
 9   dic_Ausentismo           1832 non-null   category
 10  porc_hogares_sin_medios  1832 non-null   float64
 11  IDH                      1832 non-null   float64
 12  alfabetismo              1832 non-null   float64
 13  partidoGanador           1832 non-null   object
 14  porc_2_NBI               1832 non-null   float64
 15  GINI                     1832 non-null   float64
dtypes: category(2), float64(8), object(6)
memory usage: 218.5+ KB
```

2.0.1 PARTICIONANDO LOS DATOS

[183]:
```
x=data.iloc[:, [10,11,12,14,15]].values
y=data.iloc[:, 9].values

#Dividimos un conjunto de prueba y de testeo en 70%,30%
xtrain, xtest, ytrain, ytest =train_test_split(x, #valores de los predictores
                                               y, #los valores del target
```

```
                                                    test_size=0.3, #proporción para␣
↪datos de testeo
                                                    random_state=2021, #semilla
                                                    stratify=y) #la variable de␣
↪estratificación
```

2.0.2 ESCALAMIENTO DE VARIABLES

[184]:
```
#Estandarización: Instancia StandardScaler
sc=StandardScaler()
```

[185]:
```
xtrain_std=sc.fit_transform(xtrain)
xtrain_std
```

[185]:
```
array([[ 0.16450726,  0.1979088 ,  0.51463464, -1.24779777,  0.33242611],
       [-1.28193821,  1.06962655,  0.89228746, -0.29579961,  1.36569139],
       [-0.82989869,  0.76262526,  1.24082579,  0.39104722,  1.88232404],
       ...,
       [ 0.78828941, -0.76063607, -0.45094965,  1.31982591, -0.9591555 ],
       [ 0.7814035 , -1.7295712 , -2.30881169,  0.38205904,  1.62400771],
       [ 0.76155589, -0.73728434, -0.33625731, -0.0621069 , -1.73410447]])
```

[186]:
```
#Con lo aprendido de Xtrain debemos realizar la transformacion para eñ xtest
xtest_std=sc.transform(xtest)
xtest_std
```

[186]:
```
array([[-2.07665287,  1.468092  ,  1.3493421 , -0.70176578,  0.59074243],
       [ 0.28156768, -1.20312545, -2.09942701,  0.13338603,  0.33242611],
       [-0.81491171,  0.59655889,  0.59981742, -1.060544  ,  0.33242611],
       ...,
       [-0.46130015, -0.76530266, -0.72522879, -0.96467007,  1.62400771],
       [-0.407023  ,  0.23793806,  0.59027679, -1.01110901,  0.33242611],
       [ 0.77978329,  0.58338742,  0.62255295, -0.05162069,  0.07410978]])
```

[187]:
```
columnas=["porc_hogares_sin_medios","IDH ","alfabetismo","porc_2_NB","GINI"]
```

[188]:
```
df_std=pd.DataFrame(xtrain_std,
                    columns=columnas)
df_std.head()
```

[188]:

	porc_hogares_sin_medios	IDH	alfabetismo	porc_2_NB	GINI
0	0.164507	0.197909	0.514635	-1.247798	0.332426
1	-1.281938	1.069627	0.892287	-0.295800	1.365691
2	-0.829899	0.762625	1.240826	0.391047	1.882324
3	0.778568	-0.298223	0.138127	0.366330	1.624008
4	0.661103	-1.328473	-1.740207	0.078708	-1.992421

[189]:
```
df_std.shape
```

[189]:
```
(1282, 5)
```

2.1 1. ANÁLISIS DE COMPONENTES PRINCIPALES

2.1.1 CONSTRUCCION DE LA MATRIZ DE CORRELACIÓN

[190]:
```
df_corr=df_std.corr(method="pearson")
df_corr
```

[190]:

	porc_hogares_sin_medios	IDH	alfabetismo	porc_2_NB	GINI
porc_hogares_sin_medios	1.000000	-0.791122	-0.558868	0.443077	-0.183663
IDH	-0.791122	1.000000	0.844755	-0.403202	0.114857
alfabetismo	-0.558868	0.844755	1.000000	-0.229304	0.066781
porc_2_NB	0.443077	-0.403202	-0.229304	1.000000	-0.082237
GINI	-0.183663	0.114857	0.066781	-0.082237	1.000000

2.1.2 Prueba de esferacidad de Bartlet

[191]:
```
n = df_std.shape[0] #número de observaciones
p = df_std.shape[1] #número de columnas
chi2 = -(n-1-(2*p+5)/6)*math.log(np.linalg.det(df_corr))
chi2
```

[191]:
```
3381.6716275968943
```

[192]:
```
ddl = p*(p-1)/2
ddl
```

[192]:
```
10.0
```

[193]:
```
p= stats.chi2.pdf(chi2,ddl)
p
```

[193]:
```
0.0
```

COMO EN LA PRUEBA DE LA ESFERICIDAD DE BARTLET NOS SALE “0.0”, POR LO TANTO, SI ES OPTIMO APLICAR EL MÉTODO DE ANÁLISIS DE COMPONENTES PRINCIPALES

2.1.3 ANÁLISIS DE COMPONENTES PRINCIPALES

[194]:
```
pca=PCA()#Inicializamos en PCA
xtrain_pca=pca.fit_transform(xtrain_std)
VarianzaExplicada=pca.explained_variance_ratio_
VarianzaExplicada
```

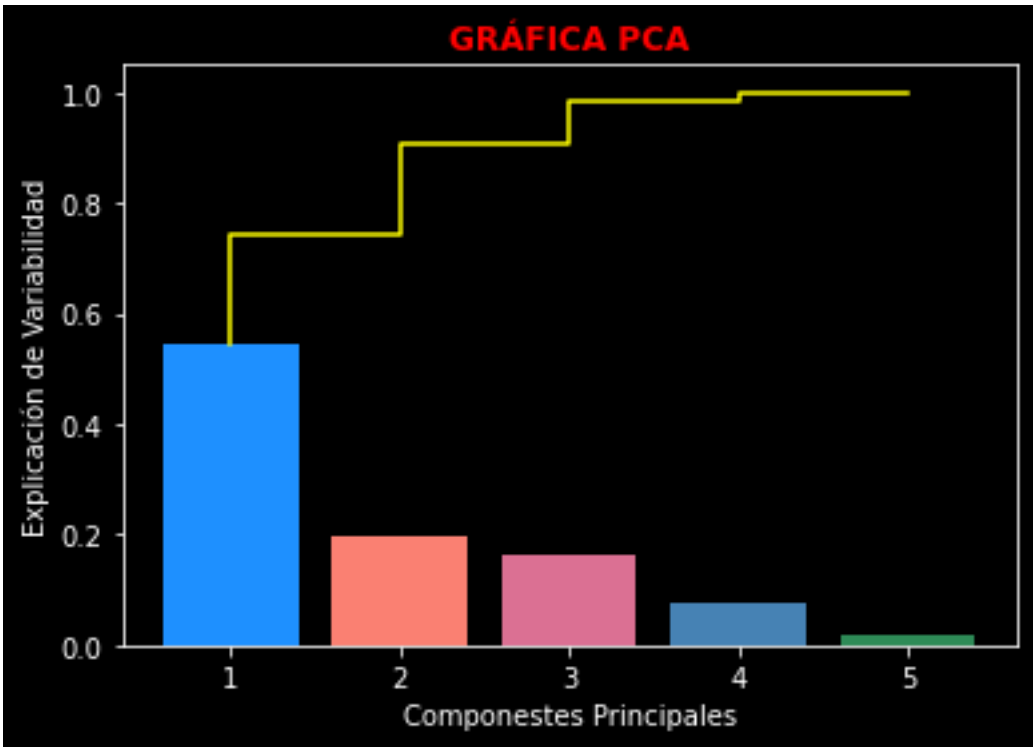
[194]:
```
array([0.54564387, 0.19724054, 0.16457896, 0.07555977, 0.01697686])
```

[195]:
```
VarianzaAcomulada=np.cumsum(pca.explained_variance_ratio_)
VarianzaAcomulada
```

[195]:
```
array([0.54564387, 0.7428844 , 0.90746337, 0.98302314, 1.        ])
```

2.1.4 GRÁFICA PCA

[196]:
```
with plt.style.context('dark_background'):
    plt.bar(range(1, 6), VarianzaExplicada,color=colors)
    plt.step(range(1, 6), VarianzaAcomulada, color="yellow")
plt.ylabel('Explicación de Variabilidad')
plt.xlabel('Componestes Principales')
plt.title("GRÁFICA PCA",weight='bold',color='red')
plt.show()
```



ANALIZANDO LA GRÁFICA PCA PODEMOS OBSERVAR QUE LA MAYOR CANTIDAD DE VARIABLES SE ENCUENTRA EN LA PRIMERA COMPONENTE PERO PARA ESTAR SEGUROS USEMOS EL CRITERIO DE KAISER Y DEBEMOS OBSERVAR QUE VARIABLES DE LA MATRIZ DE CORRELACION DE VARIANZA SEA MAYOR A UNO. AQUELLOS DATOS QUE CUMPLAN CON ESA CONDICION SERAN MIS **COMPONENTES PRINCIPALES**.

Criterio de Kaiser

[197]:
```
cov_mat = np.cov(xtrain_std.T)
eigen_vals, eigen_vecs = np.linalg.eig(cov_mat)
print('\nEigenvalues \n%s' % eigen_vals)
```

```
Eigenvalues
[2.73034911 0.08495056 0.3780938  0.82353719 0.98697255]
```

[198]:
```
print("La cantidad de COMPONENTES PRINCIPALES ES: ",(eigen_vals>0.98).sum() )
```

```
La cantidad de COMPONENTES PRINCIPALES ES:  2
```

SEGÚN EL CRITERIO DE KAISER LA CANTIDAD DE COMPONESTES PRINCIPALES SON 3 YA QUE HABIAMOS PREDICHO PERO AHORA SI LO PODEMOS ASEGURAR

[199]:
```
print("El porcentaje de los datos tomados por LOS COMPONENTES PRINCIPALES ES:")
print((eigen_vals[0]/sum(eigen_vals))*100 + (eigen_vals[1]/sum(eigen_vals))*100␣
 ↪+ (eigen_vals[2]/sum(eigen_vals))*100,"%")
```

```
El porcentaje de los datos tomados por LOS COMPONENTES PRINCIPALES ES:
63.81805037816027 %
```

2.1.5 GENERANDO EL NÚMERO DE COMPONENTES PRINCIPALES

[200]:
```
pca = PCA(n_components=2) #n_components es el número de componentes que nos␣
 ↪indicó Kaiser
#fit_transform:ajuste el modelo con X y la reducción de dimensionalidad en X.
x_std = pca.fit_transform(xtrain_std)
```

[201]:
```
df_x =pd.DataFrame(x_std)
df_x.columns = ['PC1','PC2']
df_x.head()
```

[201]:

	PC1	PC2
0	0.757592	0.293202
1	2.022982	1.073916
2	1.614128	1.454490
3	-0.428084	1.506877
4	-2.278097	-1.419195

[202]:
```
df_y = pd.DataFrame(ytrain)
df_y.columns = ['departamento']
df_y.head()
```

[202]:

	departamento
0	Ausentismo Fuerte
1	Ausentismo Fuerte
2	Ausentismo Leve
3	Ausentismo Fuerte
4	Ausentismo Leve

Nuevo conjunto de datos

[203]:
```
df_rd = pd.concat([df_x, df_y], axis=1)
df_rd.head(10)
```

[203]:

	PC1	PC2	departamento
0	0.757592	0.293202	Ausentismo Fuerte
1	2.022982	1.073916	Ausentismo Fuerte
2	1.614128	1.454490	Ausentismo Leve
3	-0.428084	1.506877	Ausentismo Fuerte
4	-2.278097	-1.419195	Ausentismo Leve
5	0.600716	-1.452937	Ausentismo Fuerte
6	-1.649830	0.391663	Ausentismo Fuerte
7	-1.013876	1.850695	Ausentismo Fuerte
8	0.341575	-0.274406	Ausentismo Leve
9	2.471436	0.731297	Ausentismo Leve

[204]:
```
df_rd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1282 entries, 0 to 1281
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   PC1           1282 non-null   float64
 1   PC2           1282 non-null   float64
 2   departamento  1282 non-null   category
dtypes: category(1), float64(2)
memory usage: 21.5 KB
```

2.2 2. ANÁLISIS FACTORIAL

Las librerias para el Analisis Factorial son:

[205]:
```
pip install factor_analyzer
```

```
Requirement already satisfied: factor_analyzer in
c:\programdata\anaconda3\lib\site-packages (0.3.2)
Requirement already satisfied: scikit-learn in
c:\programdata\anaconda3\lib\site-packages (from factor_analyzer) (0.23.2)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-
packages (from factor_analyzer) (1.1.3)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-
packages (from factor_analyzer) (1.5.2)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-
packages (from factor_analyzer) (1.19.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\programdata\anaconda3\lib\site-packages (from scikit-learn->factor_analyzer)
(2.1.0)
Requirement already satisfied: joblib>=0.11 in
c:\programdata\anaconda3\lib\site-packages (from scikit-learn->factor_analyzer)
(0.17.0)
Requirement already satisfied: python-dateutil>=2.7.3 in
c:\programdata\anaconda3\lib\site-packages (from pandas->factor_analyzer)
(2.8.1)
Requirement already satisfied: pytz>=2017.2 in
c:\programdata\anaconda3\lib\site-packages (from pandas->factor_analyzer)
(2020.1)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-
packages (from python-dateutil>=2.7.3->pandas->factor_analyzer) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

[206]:
```
import sys
from factor_analyzer import FactorAnalyzer #Ojo es necesario descargar el␣
 ↪paquete y colocarlo en una de las direcciones del path
from factor_analyzer.factor_analyzer import calculate_bartlett_sphericity
from factor_analyzer.factor_analyzer import calculate_kmo
from sklearn.decomposition import FactorAnalysis
```

PRUEBA DE ESFERICIDAD DE BARTLET

[207]:
```
chi_square_value,p_value=calculate_bartlett_sphericity(df_std)
chi_square_value, p_value
```

[207]: (3386.6599228168543, 0.0)

KMO (Kaiser-Meyer-Olkin)

[208]:
```
#KMO (Kaiser-Meyer-Olkin) Test
kmo_all,kmo_model=calculate_kmo(df_std)
kmo_model
```

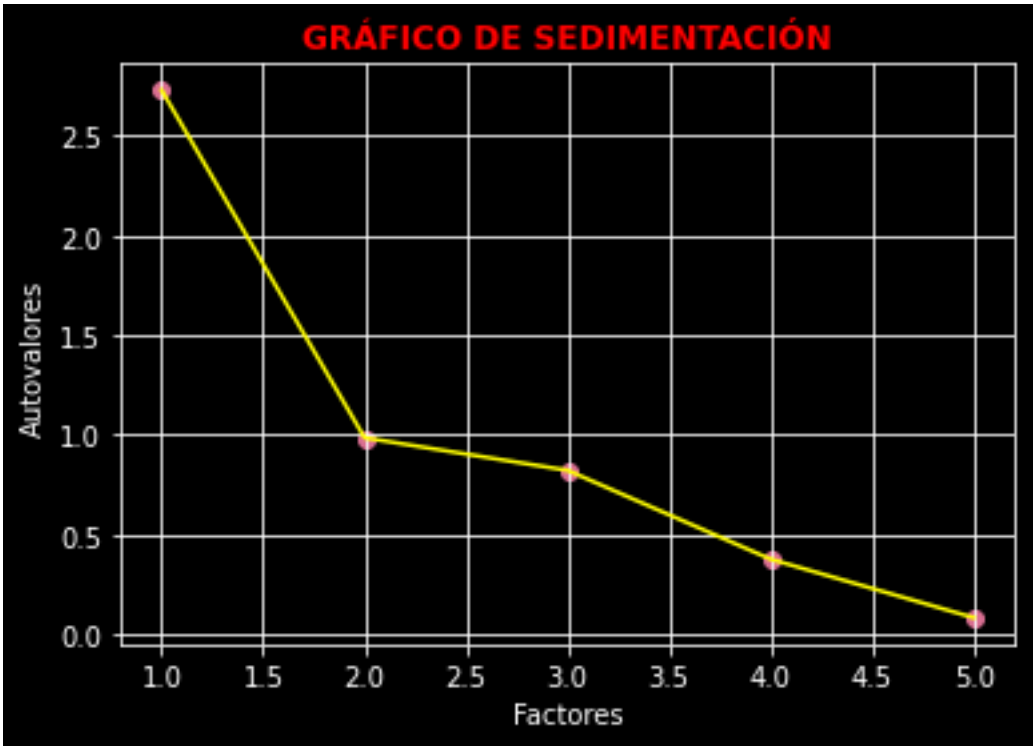
[208]: 0.6241282611609793

UNA VEZ ANALIZADOS LOS DATOS POR LA ESFERICIDAD DE BARTLET Y EL KMO PODEMOS CONCLUIR QUE SI PODEMOS UTILIZAR EL MÉTODO DE ANÁLISIS FACTORIAL

[209]:
```
factorial= FactorAnalyzer()#Creamos la instancia FactorAnalyzer()
factorial.fit(df_std)
# Check Eigenvalues
ev, v = factorial.get_eigenvalues()
ev.round(2)
```

[209]:
```
array([2.73, 0.99, 0.82, 0.38, 0.08])
```

[210]:
```
with plt.style.context('dark_background'):
    plt.scatter(range(1,df_std.shape[1]+1),ev,color=colors[2])
    plt.plot(range(1,df_std.shape[1]+1),ev,color="yellow")
plt.title('GRÁFICO DE SEDIMENTACIÓN',weight='bold',color='red')
plt.xlabel('Factores')
plt.ylabel('Autovalores')
plt.grid()
plt.show()
```



[211]:
```
factores=factorial.get_factor_variance()
factores
```

[211]:
```
(array([1.69314883, 0.847383  , 0.14461304]),
 array([0.33862977, 0.1694766 , 0.02892261]),
 array([0.33862977, 0.50810637, 0.53702897]))
```

[212]:
```
factorial_factores = FactorAnalyzer(n_factors=2,rotation='varimax')
factorial_factores.fit(df_std)
```

[212]:
```
FactorAnalyzer(n_factors=2, rotation='varimax', rotation_kwargs={})
```

[213]:
```
factorial_factores.loadings_
```

[213]:
```
array([[-0.43727583,  0.8053728 ],
       [ 0.77446682, -0.56555805],
       [ 0.9720115 , -0.16294065],
       [-0.16539021,  0.46541404],
       [ 0.02904004, -0.19168108]])
```

[214]:
```
#Comunalidades
factorial_factores.get_communalities()
```

[214]:
```
array([0.8398355 , 0.91965476, 0.97135601, 0.24396415, 0.03758496])
```

[215]:
```
#Especifidades
factorial_factores.get_uniquenesses()
```

[215]:
```
array([0.1601645 , 0.08034524, 0.02864399, 0.75603585, 0.96241504])
```

[216]:
```
xtrain_factorial=factorial_factores.fit_transform(xtrain_std)
```

[217]:
```
df_fact =pd.DataFrame(xtrain_factorial)
df_fact.columns = ['PC1','PC2']
df_fact.head(10)
```

[217]:

	PC1	PC2
0	0.546524	0.291849
1	0.733526	-0.949263
2	1.162536	-0.121876
3	0.241352	0.813412
4	-1.715462	0.019680
5	0.473249	0.092079
6	-1.321875	-0.064650
7	-0.474362	0.326619
8	0.243236	-0.124218
9	0.489995	-1.464198

[218]:
```
df_fact.shape
```

[218]:
```
(1282, 2)
```

[219]:
```
df_y = pd.DataFrame(ytrain)
df_y.columns = ['departamento']
df_y.head(10)
```

[219]:

	departamento
0	Ausentismo Fuerte
1	Ausentismo Fuerte
2	Ausentismo Leve
3	Ausentismo Fuerte
4	Ausentismo Leve
5	Ausentismo Fuerte
6	Ausentismo Fuerte
7	Ausentismo Fuerte
8	Ausentismo Leve
9	Ausentismo Leve

[220]:
```
df_rd_fact = pd.concat([df_fact, df_y], axis=1)
df_rd_fact.head(10)
```

[220]:

	PC1	PC2	departamento
0	0.546524	0.291849	Ausentismo Fuerte
1	0.733526	-0.949263	Ausentismo Fuerte
2	1.162536	-0.121876	Ausentismo Leve
3	0.241352	0.813412	Ausentismo Fuerte
4	-1.715462	0.019680	Ausentismo Leve
5	0.473249	0.092079	Ausentismo Fuerte
6	-1.321875	-0.064650	Ausentismo Fuerte
7	-0.474362	0.326619	Ausentismo Fuerte
8	0.243236	-0.124218	Ausentismo Leve
9	0.489995	-1.464198	Ausentismo Leve

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: