

A la hora de desarrollar un modelo predictivo con machine learning es de vital importancia visualizar nuestros datos para conocer su comportamiento y distribución. Esta primera observación de datos posibilita aprender más sobre ellos siendo la forma más rápida y útil de conocer qué técnicas son las más adecuadas en pre y pos procesamiento.

FASE: Análisis de Datos. Visualización.

David Víctor Gómez Ramírez

*Técnico Superior en Desarrollo de Aplicaciones
Multiplataforma especialidad en BIGDATA*

PREÁMBULO	3
IMPORTAR LIBRERÍAS NECESARIAS	3
CARGAR EL CONJUNTO DE DATOS	3
VISUALIZACIÓN UNIVARIABLE	4
HISTOGRAMAS	4
DENSIDAD	5
BOXPLOTS	6
ESTUDIO VISUALIZACIÓN "POR_CP"	8
ESTUDIO VISUALIZACIÓN "DISTANCIA"	9
ESTUDIO VISUALIZACIÓN "RITMO"	10
ESTUDIO VISUALIZACIÓN "FREC_CARDIACA"	11
ESTUDIO VISUALIZACIÓN "CADENCIA"	12
ESTUDIO VISUALIZACIÓN "TSC"	13
ESTUDIO VISUALIZACIÓN "FP"	14
ESTUDIO VISUALIZACIÓN "LSS"	15
ESTUDIO VISUALIZACIÓN "OSC_VERTICAL"	16
ESTUDIO VISUALIZACIÓN "L_ZANCADA"	17
ESTUDIO VISUALIZACIÓN "RFP"	18
ESTUDIO VISUALIZACIÓN "RLSS"	19
ESTUDIO VISUALIZACIÓN "ROV"	20
ESTUDIO VISUALIZACIÓN "RE"	21
ESTUDIO VISUALIZACIÓN "AIRE"	22
ESTUDIO VISUALIZACIÓN "PENDIENTE"	23
ESTUDIO VISUALIZACIÓN "ALTITUD"	24
ESTUDIO VISUALIZACIÓN "DESNIVEL"	25
ESTUDIO VISUALIZACIÓN "RSS"	26
ESTUDIO VISUALIZACIÓN "DURACION"	27
CONCLUSIONES EXTRAÍDAS VISUALIZACIÓN	28
VISUALIZACIÓN MULTIVARIABLE	28
MATRIZ DE CORRELACIÓN	28
COEFICIENTES DE "PEARSON"	28
COEFICIENTES DE "SPEARMAN"	30
GRÁFICOS DE MATRIZ DE DISPERSIÓN	31
Función para Gráficos de Matriz de Dispersion	31
Creación de grupos para graficar Matrices de dispersión	31
Grupo 1 "POR_CP", "DISTANCIA", "DURACION"	32
Grupo 2 "RITMO", "FREC_CARDIACA", "DURACION"	32
Grupo 3 "CADENCIA", "TSC", "DURACION"	33
Grupo 4 "FP", "LSS", "DURACION"	33
Grupo 5 "OSC_VERTICAL", "L_ZANCADA", "DURACION"	34
Grupo 6 "RFP", "RLSS", "DURACION"	34

Grupo 7 "ROV", "RE", "DURACION"	35
Grupo 8 "AIRE", "PENDIENTE", "DURACION"	35
Grupo 9 "ALTITUD", "DESNIVEL", "DURACION".....	36
Grupo 10 "RSS", "DURACION"	36

PREÁMBULO

Lo primero que debemos realizar a la hora de trabajar con machine learning es visualizar nuestros datos para conocer su comportamiento y distribución. Esta primera observación de datos posibilita aprender más sobre ellos siendo la forma más rápida y útil de conocer qué técnicas son las más adecuadas en pre y pos procesamiento. En este sentido en esta tercera sección trabajaremos:

- Cómo crear gráficos para entender cada atributo de manera independiente.
- Cómo crear gráficos para entender las relaciones entre los diferentes atributos.

Los gráficos de las relaciones entre los atributos pueden darnos una idea de los atributos que pueden ser redundantes, los métodos de remuestreo que pueden ser necesarios y, en última instancia, la dificultad de un problema de predicción. Para ello, la fase de visualización puede dividirse en las siguientes partes:

- **Visualización univariable:** cuando queremos visualizar un atributo de manera independiente a los demás.
- **Visualización multivariable:** cuando queremos visualizar la interacción entre los diferentes atributos de nuestro conjunto de datos.

Importar librerías necesarias

```
# ****
# ***** LIBRERÍAS A IMPORTAR ****
# ****
import pandas           as pd
import matplotlib.pyplot as plt
import pandas           as pd
import numpy             as np
import seaborn          as sns
```

Cargar el conjunto de datos

Para esta práctica vamos a cargar el conjunto de datos de nuestro proyecto "SEGMENTOS_csv.csv" para hacer observaciones con las funciones que nos permitan hacer diferentes tipos de visualizaciones. Además y, conocedores que en "DESNIVEL" hay un registro vacío, procederemos a su resolución para un mejor tratamiento.

```
# ****
# ***** CARGAMOS NUESTRO DATAFRAME ****
# ****
# Cargamos los datos contenidos en "SEGMENTOS_csv.csv"
data = pd.read_csv('SEGMENTOS_csv.csv')
# Completamos los datos nulos con la media de cada uno
data['DESNIVEL'] = data['DESNIVEL'].fillna(data['DESNIVEL'].median())
```

	POR	CP	DISTANCIA	RETRO	FREC_CARDIACA	CADENCIA	TSC	FP	LSS	\
0	85.863	500	519	154	166	348	64	12.9		
1	84.816	1000	547	139	150	343	66	11.8		
2	84.293	1000	537	137	161	346	65	12.1		
3	84.816	1000	548	142	162	347	66	12.2		
4	84.816	1000	539	146	163	345	65	12.1		
5	84.816	1000	542	150	163	346	65	12.3		
6	84.816	1000	537	152	165	339	65	12.4		
7	85.863	600	524	155	165	341	64	12.1		
8	85.340	530	561	128	158	340	65	11.7		
9	84.816	5570	538	145	163	345	66	12.2		
10	84.816	2000	545	134	160	345	66	11.9		
11	84.816	2000	540	144	162	346	66	12.2		
12	84.816	2000	539	151	164	342	65	12.3		
13	84.816	3000	541	136	161	345	66	12.0		
14	84.816	3000	539	149	164	343	65	12.3		

	OSC_VERTICAL	L_ZANCADA	RFP	RLSS	ROV	RE	AIRE	PENDIENTE	\
0	5.06	0.697	39.024	146.341	7.228	0.965	1	-0.5	
1	5480.00	0.689	40.440	143.962	8.058	0.925	0	0.4	
2	5470.00	0.693	40.993	141.568	7.927	0.948	0	0.0	
3	5400.00	0.685	40.724	146.568	7.997	0.937	0	0.2	
4	5299.00	0.682	40.123	147.568	7.179	0.939	0	0.0	
5	5230.00	0.679	40.123	150.000	7.691	0.933	0	0.3	
6	5180.00	0.677	40.123	151.219	7.617	0.942	1	-0.2	
7	5070.00	0.688	39.024	147.560	7.347	0.946	1	-0.2	
8	5410.00	0.670	39.877	142.682	8.074	0.888	1	-0.8	
9	5340.00	0.683	40.740	148.788	7.852	0.939	0	0.1	
10	5480.00	0.693	40.740	145.121	7.942	0.935	0	0.2	
11	5360.00	0.685	40.740	148.788	7.882	0.937	0	0.1	
12	5210.00	0.678	40.123	150.000	7.661	0.938	1	0.0	
13	5460.00	0.689	40.740	146.341	7.913	0.936	0	0.2	
14	5240.00	0.677	40.123	150.000	7.594	0.937	0	0.0	

	ALTITUD	DESNIVEL	RSS	DURACION
0	19	2.0	5	259
1	16	7.0	10	547
2	18	4.0	9	537
3	18	6.0	9	540
4	19	5.0	9	539
5	18	7.0	9	542
6	18	2.0	9	537
7	19	3.0	6	317
8	16	5.0	5	300
9	18	27.0	51	3000
10	17	11.0	18	1082
11	19	11.0	18	1080
12	18	9.0	18	1079
13	17	17.0	28	1622
14	18	13.0	28	1619

Visualización Univariable

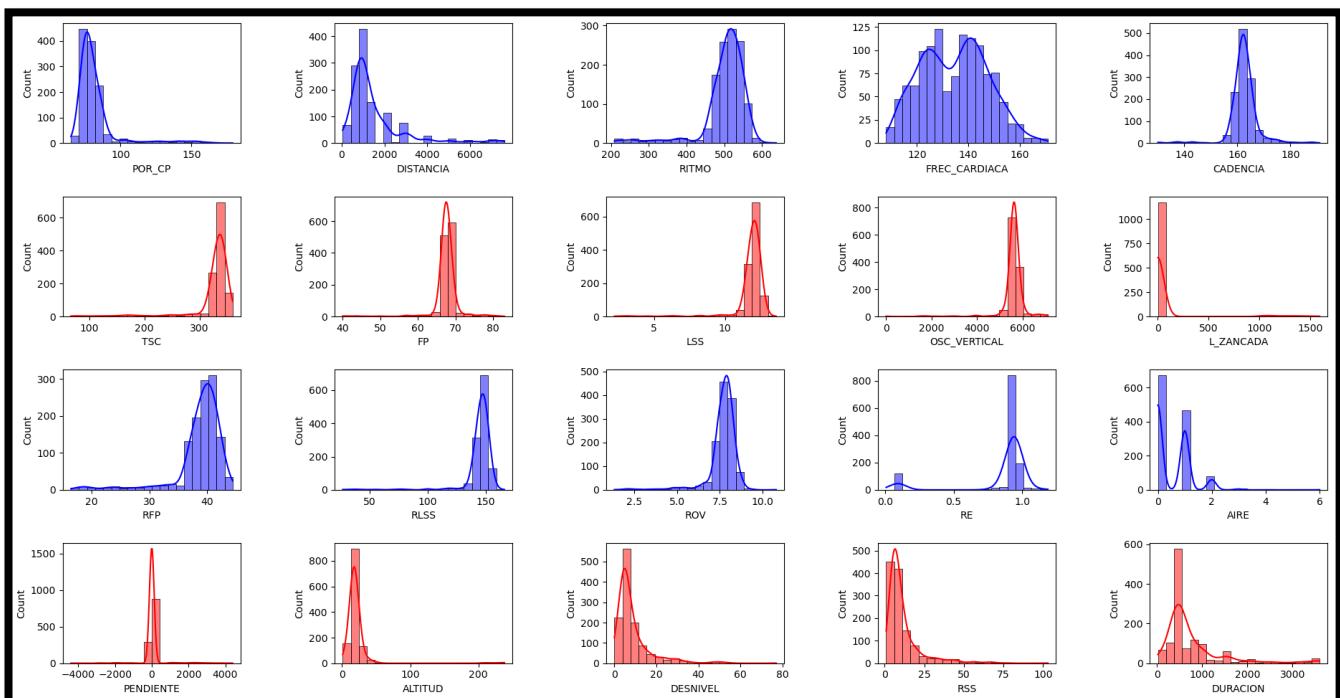
Como se ha comentado anteriormente, las gráficas univariable nos permiten visualizar los atributos individuales sin interacciones; las cuales, el objetivo principal de las mismas es aprender algo sobre la distribución, la tendencia y la propagación de cada atributo.

A continuación se describen las más relevantes.

Histogramas

A partir de la forma de los contenedores, puede tener una idea rápida de si un atributo es gaussiano, sesgado o incluso tiene una distribución exponencial. También puede ayudarlo a ver posibles valores atípicos, por lo que tanto Matplotlib como Seabornpueden ser potentes librerías de visualización de datos.

```
# ****
# ***** VIZUALIZACIONES *****
# ****
#   HISTOGRAMAS O DISTRIBUCIÓN CON DENSIDAD
f, axes = plt.subplots(4,5, figsize =(11.7,8.27))
sns.histplot(data["POR_CP"], ax= axes [0,0],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["DISTANCIA"], ax= axes [0,1],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["RITMO"], ax= axes [0,2],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["FREC_CARDIACA"], ax= axes [0,3],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["CADENCIA"], ax= axes [0,4],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["TSC"], ax= axes [1,0],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["FP"], ax= axes [1,1],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["LSS"], ax= axes [1,2],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["OSC_VERTICAL"], ax= axes [1,3],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["L_ZANCADA"], ax= axes [1,4],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["RFP"], ax= axes [2,0],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["RLSS"], ax= axes [2,1],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["ROV"], ax= axes [2,2],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["RE"], ax= axes [2,3],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["AIRE"], ax= axes [2,4],kde = True, bins = 20, color="Blue", fill=True)
sns.histplot(data["PENDIENTE"], ax= axes [3,0],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["ALTITUD"], ax= axes [3,1],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["DESNIVEL"], ax= axes [3,2],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["RSS"], ax= axes [3,3],kde = True, bins = 20, color="Red", fill=True)
sns.histplot(data["DURACION"], ax= axes [3,4],kde = True, bins = 20, color="Red", fill=True)
plt.tight_layout()
plt.show()
```



Siempre, a priori, y visualmente hablando, podemos observar que la mayoría de los atributos suelen tener una distribución casi gaussiana o normal (algunas simples, otras con doble campana FREC_CARDIACA y REC y otra triple campana AIRE) y aparentemente alguna exponencial L_ZANCADA. También podemos observar la existencia de sesgo en casi todas las distribuciones siendo este menor o casi inexistente en FREC_CARDIACA y CADENCIA. Esto es interesante porque muchas técnicas de aprendizaje automático suponen una distribución univariada gaussiana en las variables de entrada.

Densidad

Las gráficas se ven como un histograma abstracto con una curva suave dibujada a través de la parte superior de cada contenedor, al igual que su ojo intentó hacer con los histogramas. Podemos ver que la distribución de cada atributo es más clara que los histogramas

```
# GRAFICOS DE DENSIDAD
f, axes = plt.subplots(4,5, figsize =(11.7,8.27))
sns.kdeplot(data["POR_CP"], ax = axes [0,0], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["DISTANCIA"], ax = axes [0,1], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["RITMO"], ax = axes [0,2], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["FREC_CARDIACA"], ax = axes [0,3], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["CADENCIA"], ax = axes [0,4], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["TSC"], ax = axes [1,0], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["FP"], ax = axes [1,1], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["LSS"], ax = axes [1,2], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["OSC_VERTICAL"], ax = axes [1,3], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["L_ZANCADA"], ax = axes [1,4], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["RFP"], ax = axes [2,0], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["RLSS"], ax = axes [2,1], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["ROV"], ax = axes [2,2], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["RE"], ax = axes [2,3], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["AIRE"], ax = axes [2,4], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["PENDIENTE"], ax = axes [3,0], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["ALTITUD"], ax = axes [3,1], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.kdeplot(data["DESNIVEL"], ax = axes [3,2], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

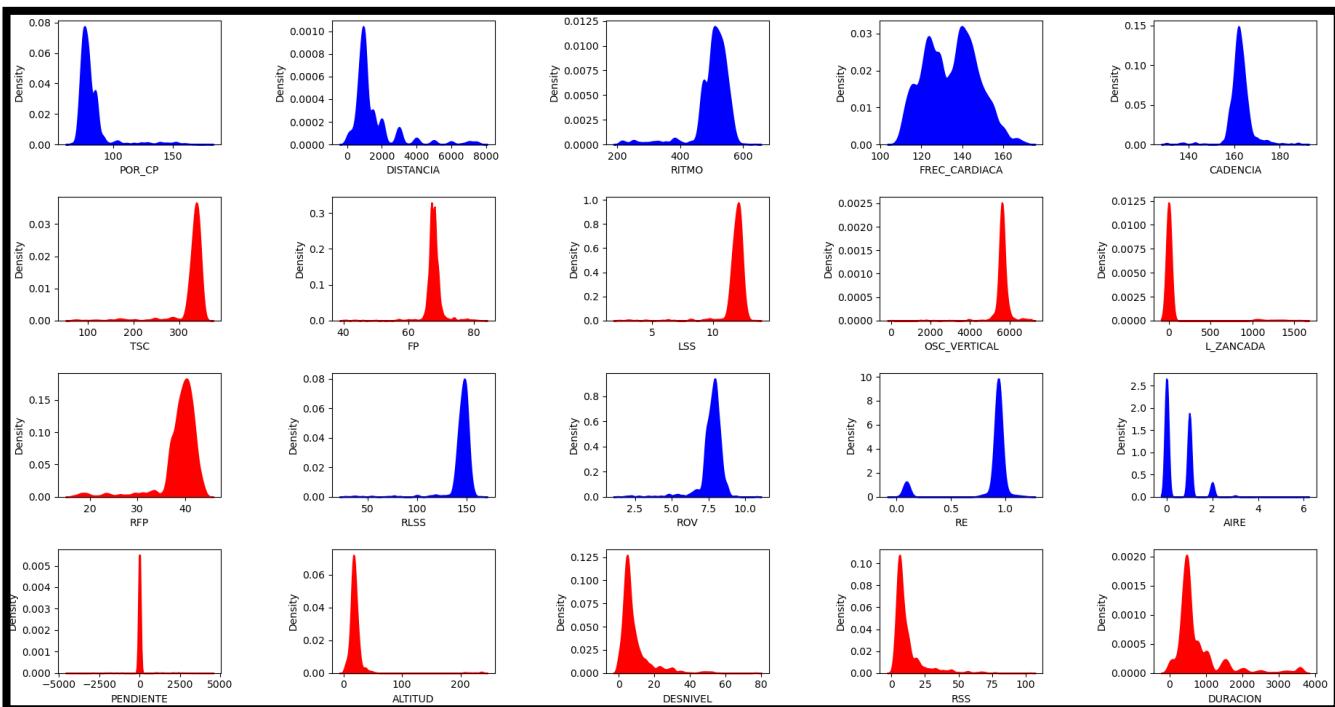
sns.kdeplot(data["RSS"], ax = axes [3,3], shade = True, color = "Red", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)
```

```

sns.kdeplot(data["DURACION"] ,
             ax = axes [3,4], shade = True, color = "Red", fill = True,
             bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

plt.tight_layout()
plt.show()

```



Boxplots

Podemos ver que la extensión de los atributos es bastante diferente. Algunos como la FREC_CARDIACA, L_ZANCADA y RFP parecen bastante sesgados hacia valores más pequeños.

```

# GRAFICOS BOXPLOTS
f, axes = plt.subplots(4,5, figsize =(11.7,8.27))
sns.boxplot(x = data["POR_CP"],           ax = axes [0,0], orient = "h", color = "lightblue", saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["DISTANCIA"],         ax = axes [0,1], orient = "h", color = "lightblue", saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["RITMO"],             ax = axes [0,2], orient = "h", color = "lightblue", saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["FREC_CARDIACA"],     ax = axes [0,3], orient = "h", color = "lightblue", saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["CADENCIA"],          ax = axes [0,4], orient = "h", color = "lightblue", saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["TSC"],               ax = axes [1,0], orient = "h", color = "lightgreen",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["FP"],                ax = axes [1,1], orient = "h", color = "lightgreen",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["LSS"],               ax = axes [1,2], orient = "h", color = "lightgreen",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["OSC_VERTICAL"],      ax = axes [1,3], orient = "h", color = "lightgreen",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["L_ZANCADA"],         ax = axes [1,4], orient = "h", color = "lightgreen",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["RFP"],               ax = axes [2,0], orient = "h", color = "lightblue",saturation= 1,
             width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

```

```

sns.boxplot(x = data["RLSS"],
            ax = axes [2,1], orient = "h", color = "lightblue", saturation = 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["ROV"],
            ax = axes [2,2], orient = "h", color = "lightblue", saturation = 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["RE"],
            ax = axes [2,3], orient = "h", color = "lightblue", saturation = 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["AIRE"],
            ax = axes [2,4], orient = "h", color = "lightblue", saturation = 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["PENDIENTE"],
            ax = axes [3,0], orient = "h", color = "lightgreen", saturation= 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["ALTITUD"],
            ax = axes [3,1], orient = "h", color = "lightgreen", saturation= 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

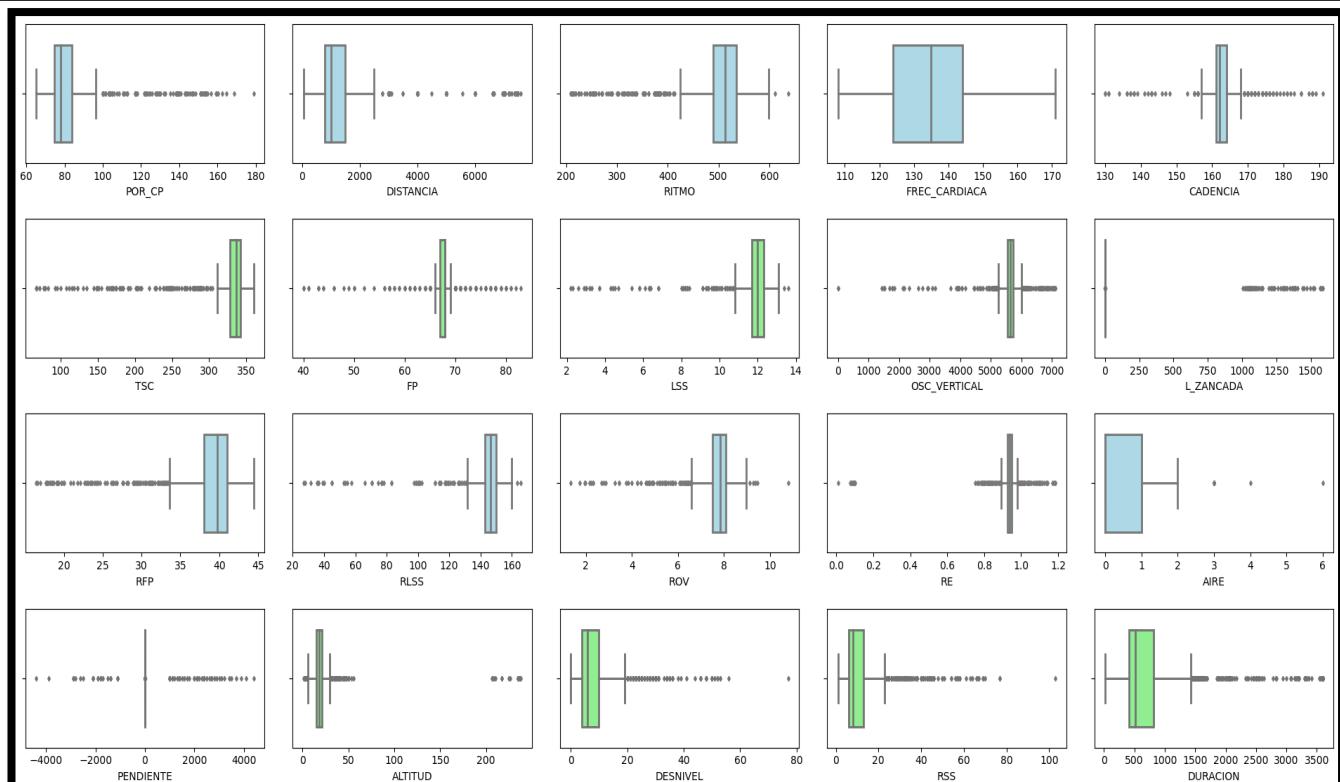
sns.boxplot(x = data["DESNIVEL"],
            ax = axes [3,2], orient = "h", color = "lightgreen", saturation= 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["RSS"],
            ax = axes [3,3], orient = "h", color = "lightgreen", saturation= 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

sns.boxplot(x = data["DURACION"],
            ax = axes [3,4], orient = "h", color = "lightgreen", saturation= 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)

plt.tight_layout()
plt.show()

```



Estudio Visualización "POR_CP"

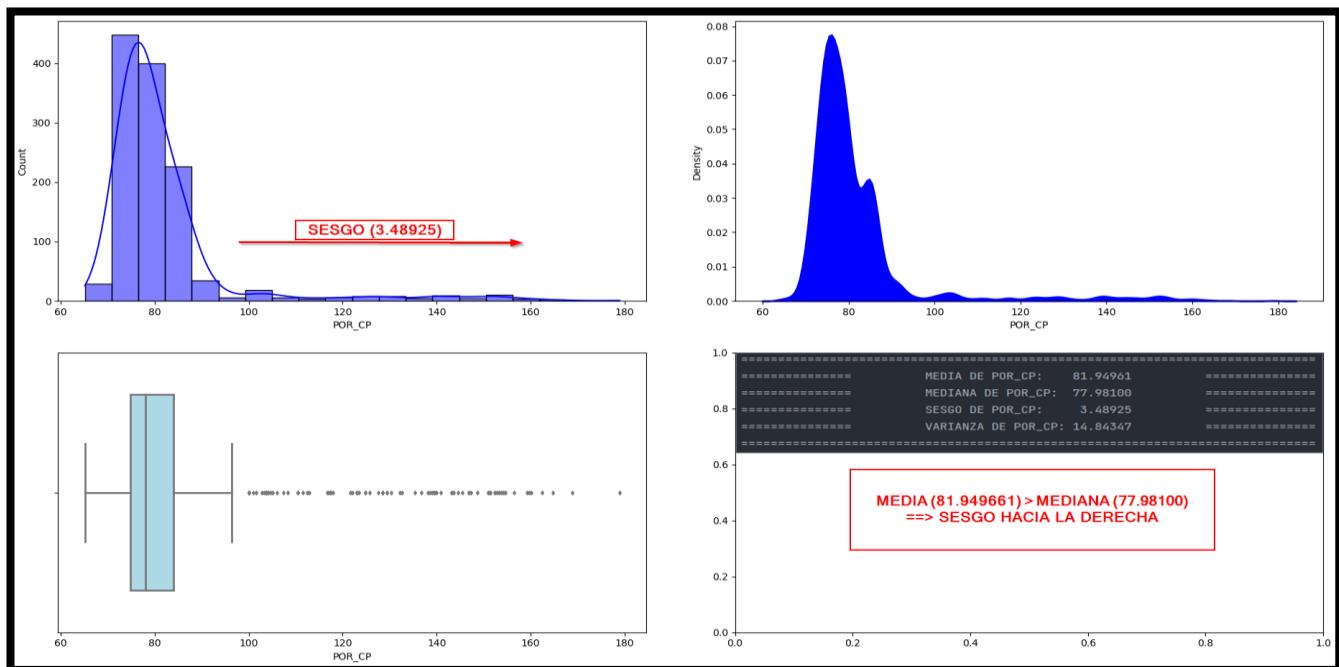
```
# Estudio Visualización "POR_CP"
# Métricas POR_CP
print("")
===== MEDIANA DE POR_CP:
print(f" {data['POR_CP'].median():,.5f}") =====
===== MEDIA DE POR_CP:
print(f" {data['POR_CP'].mean():,.5f}") =====
===== SESGO DE POR_CP:
print(f" {data['POR_CP'].skew():,.5f}") =====
print("")

f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["POR_CP"], ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["POR_CP"], ax = axes [0,1], shade = True, color = "Blue",
fill = True, bw_adjust = .5, clip_on = False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["POR_CP"], ax = axes [1,0], orient = "h", color =
"lightblue", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "DISTANCIA"

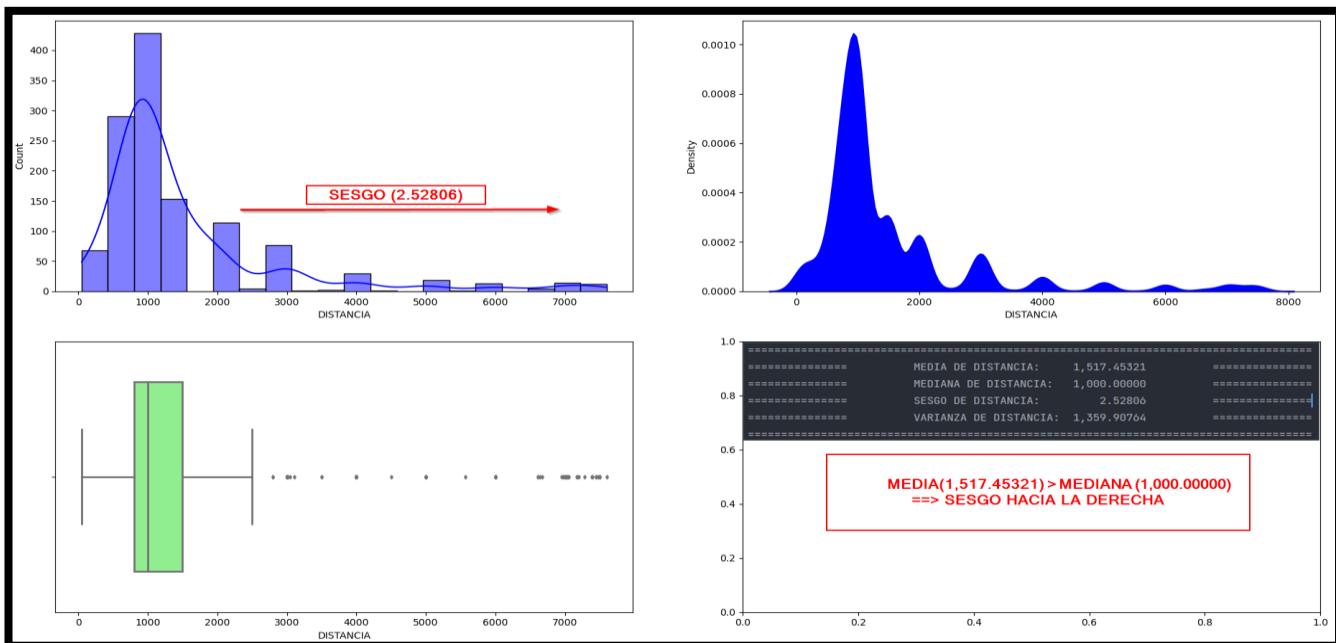
```
# Estudio Visualización "DISTANCIA"
# Métricas DISTANCIA
print("===== Média de Distancia: =====")
print(f" {data['DISTANCIA'].mean():,.5f}")
print("===== Mediana de Distancia: =====")
print(f" {data['DISTANCIA'].median():,.5f}")
print("===== Sesgo de Distancia: =====")
print(f" {data['DISTANCIA'].skew():,.5f}")
print("===== Varianza de Distancia: =====")
print(f" {data['DISTANCIA'].std():,.5f}")
print("=====")

# Visualización Histograma, Densidad y Boxplot de DISTANCIA
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["DISTANCIA"], ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["DISTANCIA"], ax = axes [0,1], shade = True, color = "Blue",
fill = True, bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["DISTANCIA"], ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "RITMO"

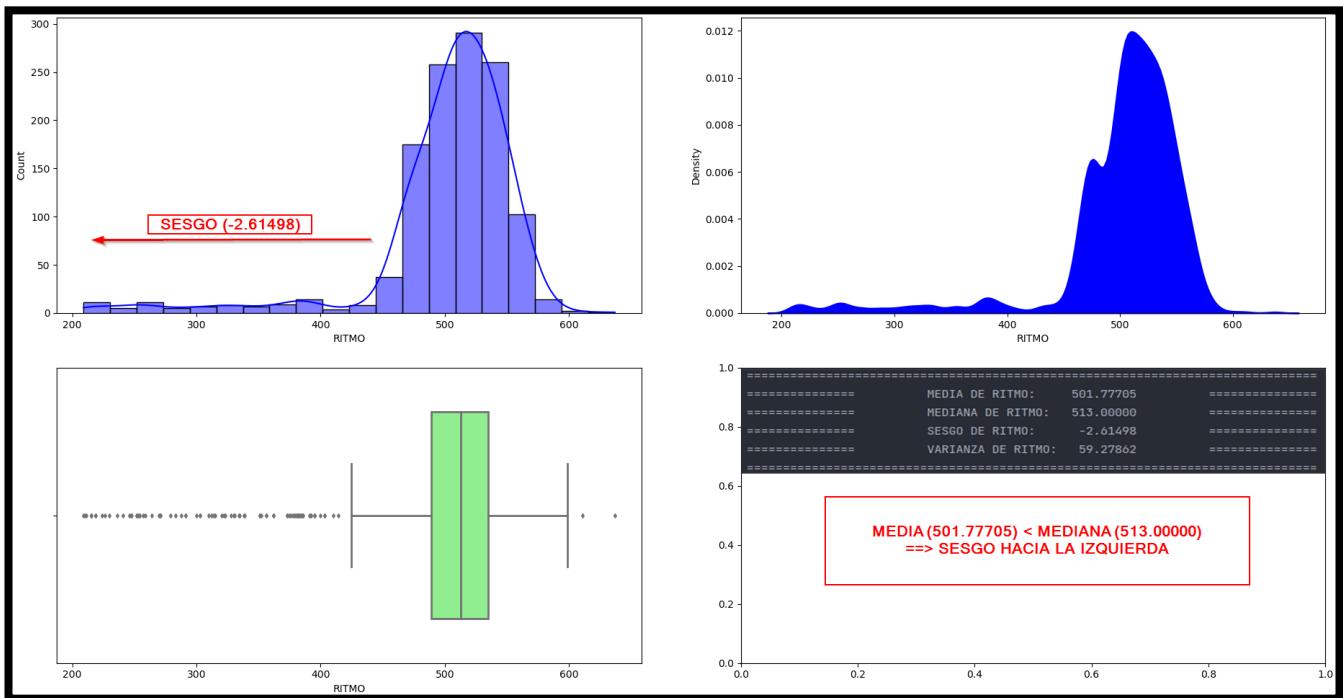
```
# Estudio Visualización "RITMO"
# Métricas RITMO
print("===== Média de Ritmo: =====")
print(f" {data['RITMO'].mean():,.5f}")
print("===== Mediana de Ritmo: =====")
print(f" {data['RITMO'].median():,.5f}")
print("===== Sesgo de Ritmo: =====")
print(f" {data['RITMO'].skew():,.5f}")
print("===== Varianza de Ritmo: =====")
print(f" {data['RITMO'].std():,.5f}")
print("===== ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de RITMO
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["RITMO"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["RITMO"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["RITMO"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "FREC_CARDIACA"

```
# Estudio Visualización "FREC_CARDIACA"
# Métricas FREC_CARDIACA
print("")

=====
="") print(f" ===== MEDIA DE FREC_CARDIACA: =====")
{data['FREC_CARDIACA'].mean():,.5f} print(f" ===== MEDIANA DE FREC_CARDIACA: =====")
{data['FREC_CARDIACA'].median():,.5f} print(f" ===== SESGO DE FREC_CARDIACA: =====")
{data['FREC_CARDIACA'].skew():,.5f} print(f" ===== VARIANZA DE FREC_CARDIACA: =====")
{data['FREC_CARDIACA'].std():,.5f} print("")

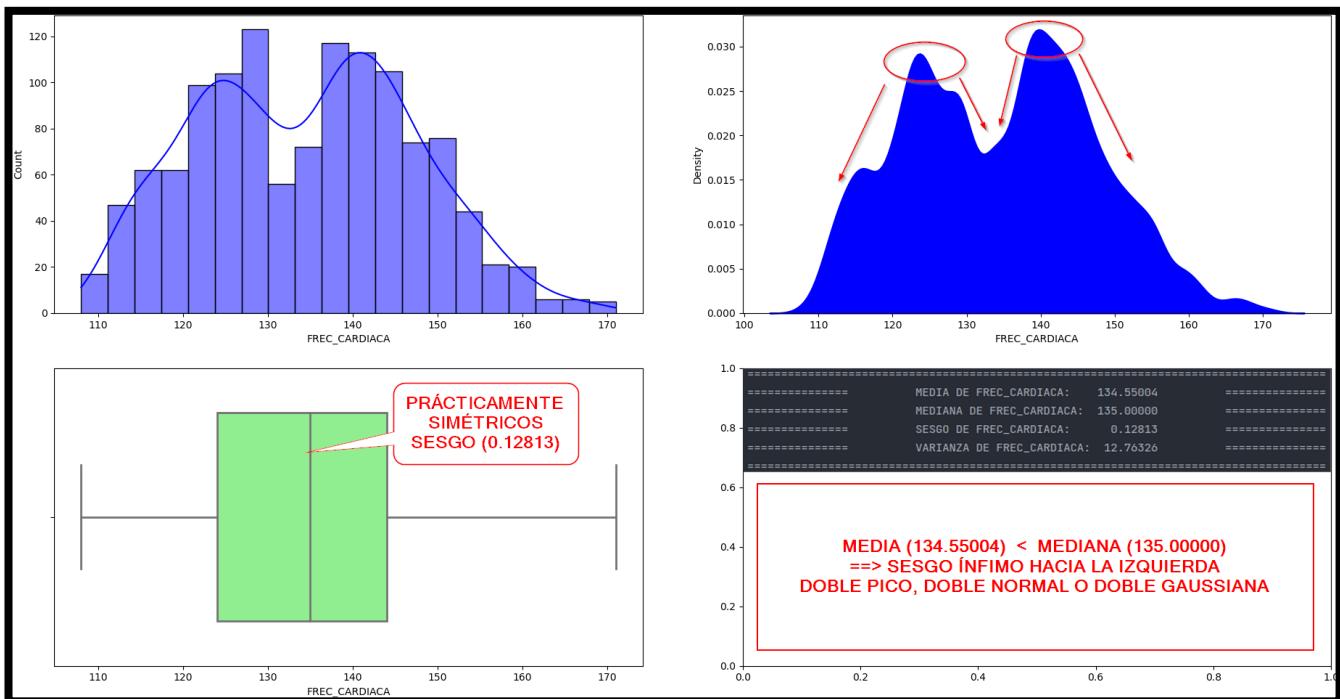
=====

# Visualización Histograma, Densidad y Boxplot de FREC_CARDIACA
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["FREC_CARDIACA"], ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["FREC_CARDIACA"], ax = axes [0,1], shade = True, color =
"Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["FREC_CARDIACA"], ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



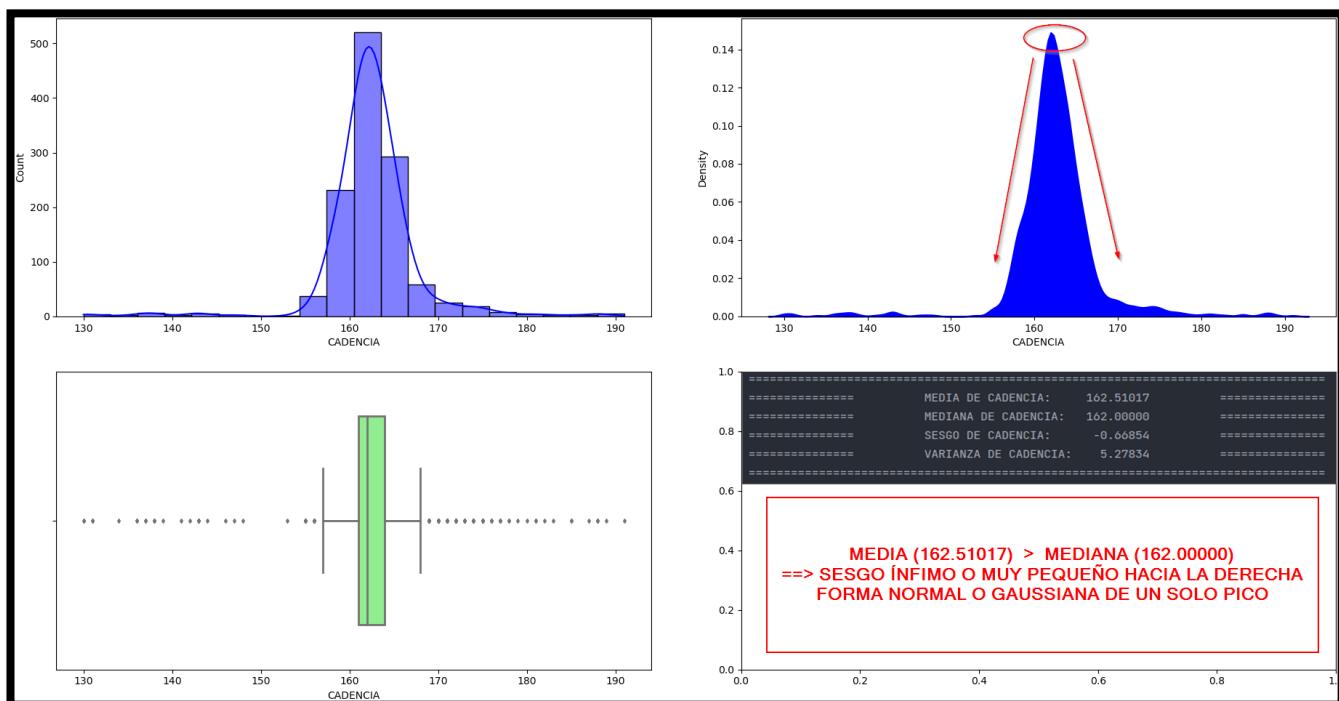
Estudio Visualización "CADENCIA"

```
# Estudio Visualización "CADENCIA"
# Métricas CADENCIA
print("===== Média de Cadencia: =====")
print(f" {data['CADENCIA'].mean():,.5f}")
print("===== Mediana de Cadencia: =====")
print(f" {data['CADENCIA'].median():,.5f}")
print("===== Sesgo de Cadencia: =====")
print(f" {data['CADENCIA'].skew():,.5f}")
print("===== Varianza de Cadencia: =====")
print(f" {data['CADENCIA'].std():,.5f}")
print("===== ")
# Visualización Histograma, Densidad y Boxplot de CADENCIA
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["CADENCIA"],      ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["CADENCIA"],      ax = axes [0,1], shade = True, color = "Blue",
fill = True, bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["CADENCIA"],   ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "TSC"

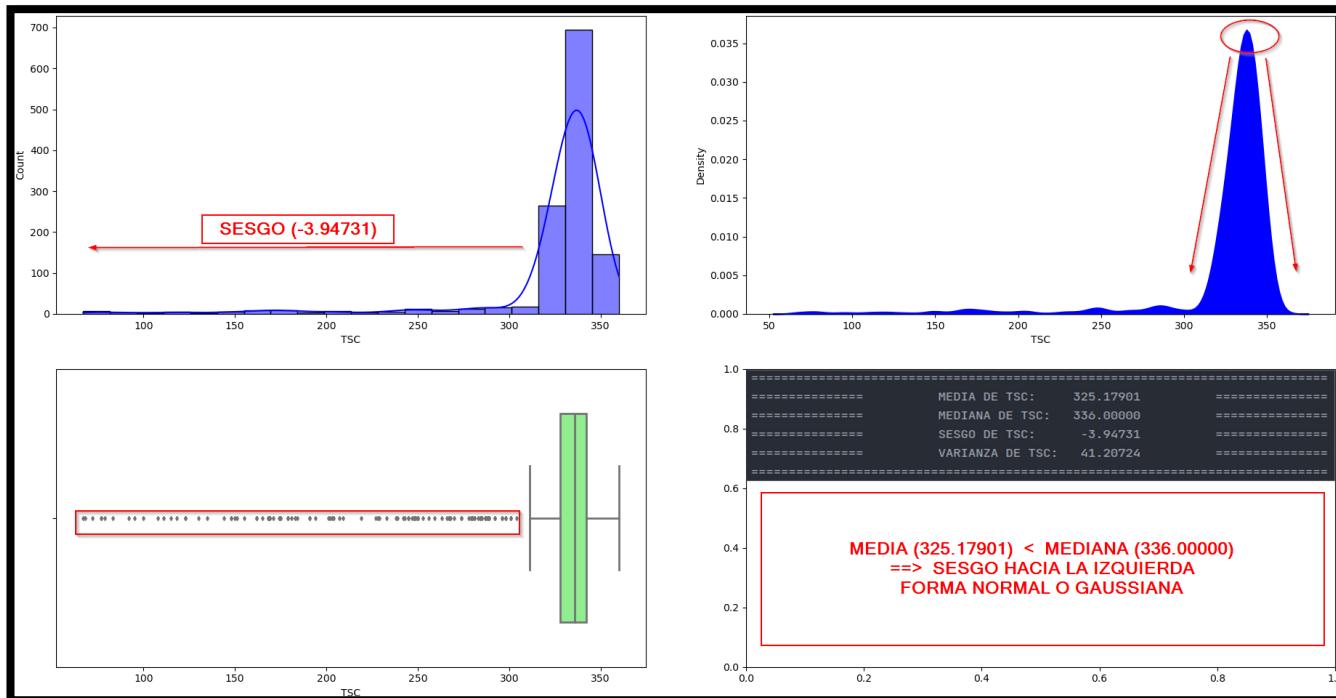
```
# Estudio Visualización "TSC"
# Métricas TSC
print("")
=====
print(f"          ====== MEDIA DE TSC:")
print(f" {data['TSC'].mean():,.5f} ======")
print(f"          ====== MEDIANA DE TSC:")
print(f" {data['TSC'].median():,.5f} ======")
print(f"          ====== SESGO DE TSC:")
print(f" {data['TSC'].skew():,.5f} ======")
print(f"          ====== VARIANZA DE TSC:")
print(f" {data['TSC'].std():,.5f} ======")
print("")
=====

# Visualización Histograma, Densidad y Boxplot de TSC
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["TSC"],      ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["TSC"],      ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.boxplot(x = data["TSC"],  ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3, linewidth =
2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "FP"

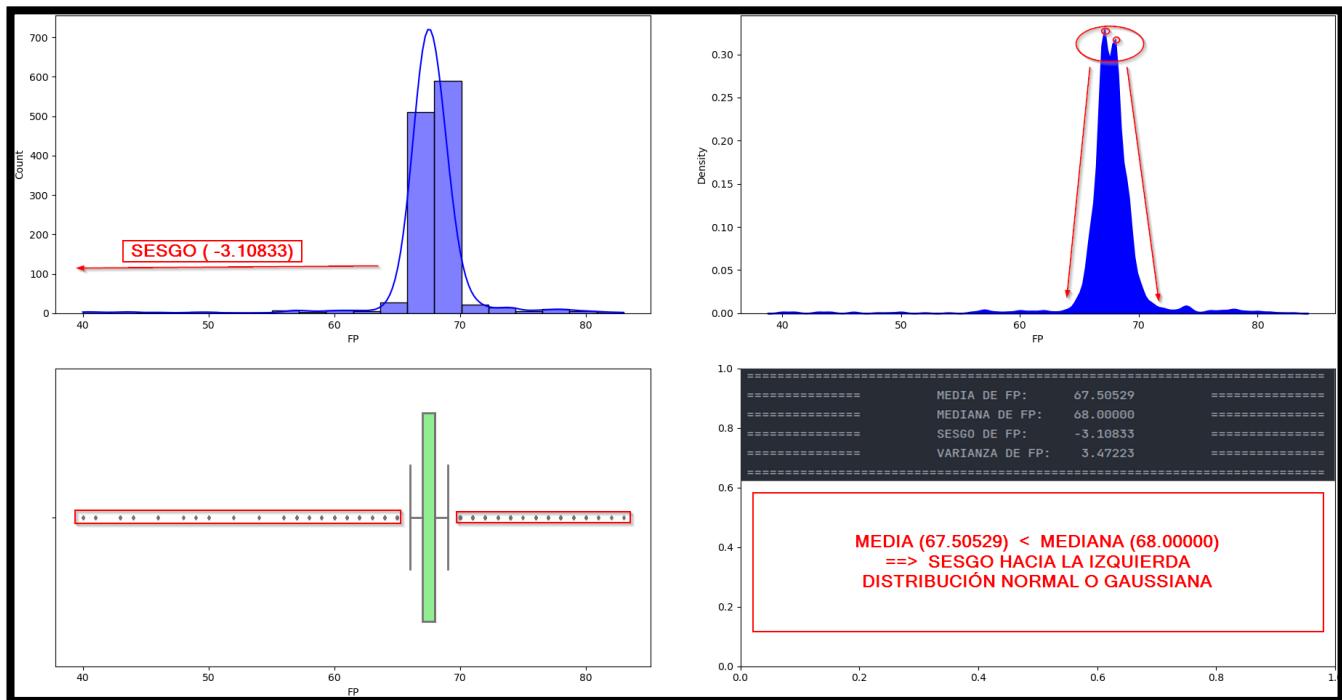
```
# Estudio Visualización "FP"
# Métricas FP
print("===== ")
print(f" ===== MÉDIA DE FP: {data['FP'].mean():,.5f}")
print(f" ===== MEDIANA DE FP: {data['FP'].median():,.5f}")
print(f" ===== SESGO DE FP: {data['FP'].skew():,.5f}")
print(f" ===== VARIANZA DE FP: {data['FP'].std():,.5f}")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de FP
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["FP"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["FP"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["FP"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "LSS"

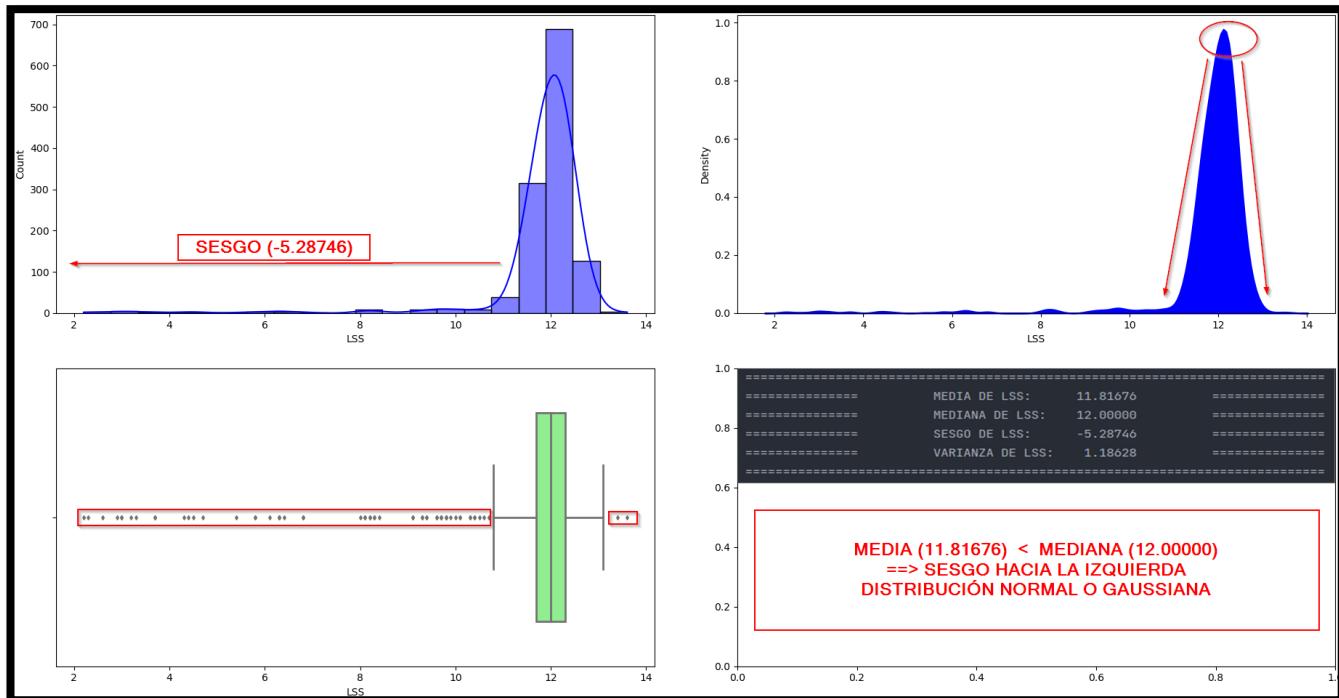
```
# Estudio Visualización "LSS"
# Métricas LSS
print("===== ")
print(f"          ====== MEDIA DE LSS: {data['LSS'].mean():,.5f}")
print(f"          ====== MEDIANA DE LSS: {data['LSS'].median():,.5f}")
print(f"          ====== SESGO DE LSS: {data['LSS'].skew():,.5f}")
print(f"          ====== VARIANZA DE LSS: {data['LSS'].std():,.5f}")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de LSS
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["LSS"],      ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["LSS"],      ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.boxplot(x = data["LSS"],  ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3, linewidth =
2)
plt.tight_layout()
plt.show()
```



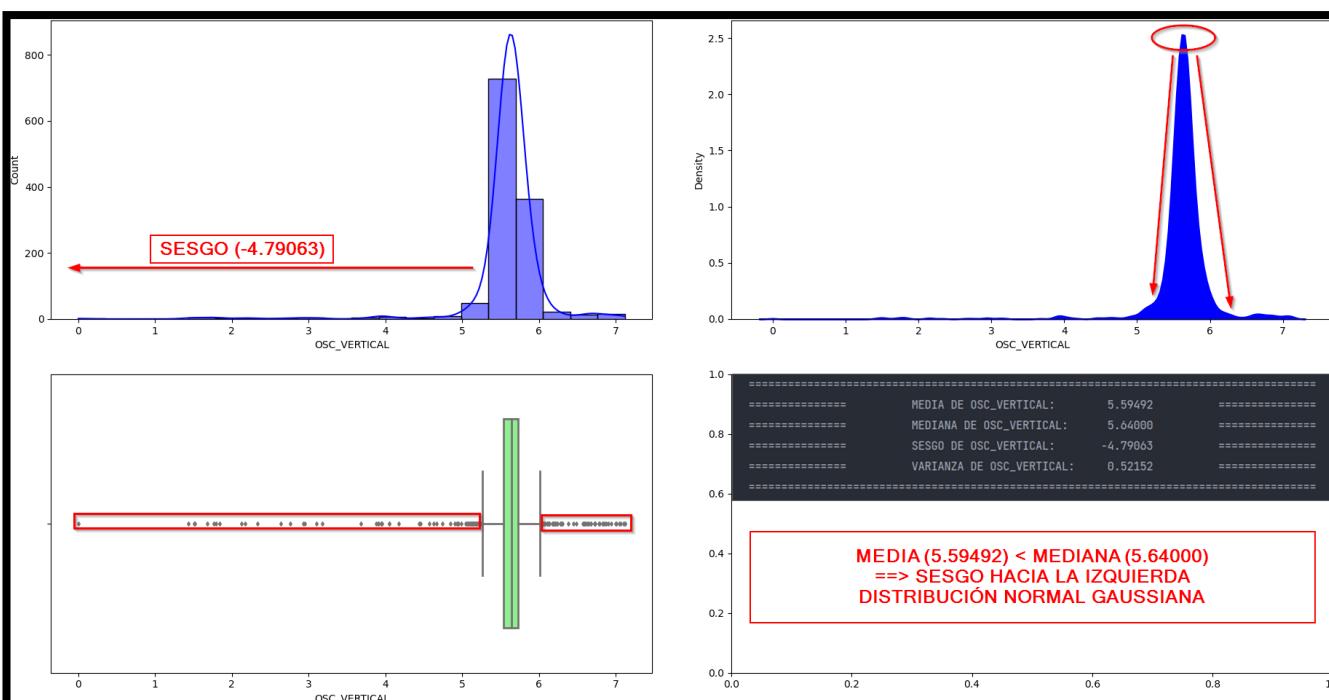
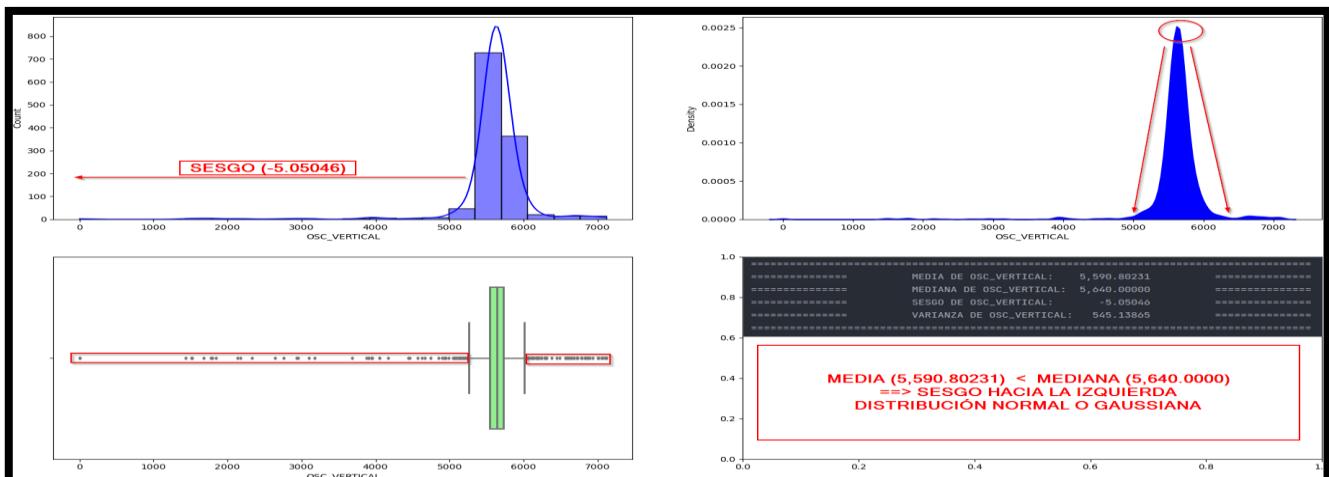
Estudio Visualización "OSC_VERTICAL"

```
# Estudio Visualización "OSC_VERTICAL"
# Métricas OSC_VERTICAL
print("===== Média de OSC_VERTICAL: {data['OSC_VERTICAL'].mean():,.5f}")
print("===== Mediana de OSC_VERTICAL: {data['OSC_VERTICAL'].median():,.5f}")
print("===== Sesgo de OSC_VERTICAL: {data['OSC_VERTICAL'].skew():,.5f}")
print("===== Varianza de OSC_VERTICAL: {data['OSC_VERTICAL'].std():,.5f}")
print("=====")

# Visualización Histograma, Densidad y Boxplot de OSC_VERTICAL
f, axes = plt.subplots(2, 2, figsize =(11.7,8.27))

sns.histplot(data["OSC_VERTICAL"], ax = axes [0,0], kde = True, bins = 20, color="Blue", fill=True)
sns.kdeplot(data["OSC_VERTICAL"], ax = axes [0,1], shade = True, color = "Blue", fill = True,
             bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)
sns.boxplot(x = data["OSC_VERTICAL"], ax = axes [1,0], orient = "h", color = "lightgreen", saturation = 1,
            width = 0.7, dodge = True, fliersize = 3, linewidth = 2)
plt.tight_layout()
plt.show()
```

!!!! OJO-ERROR!!!!



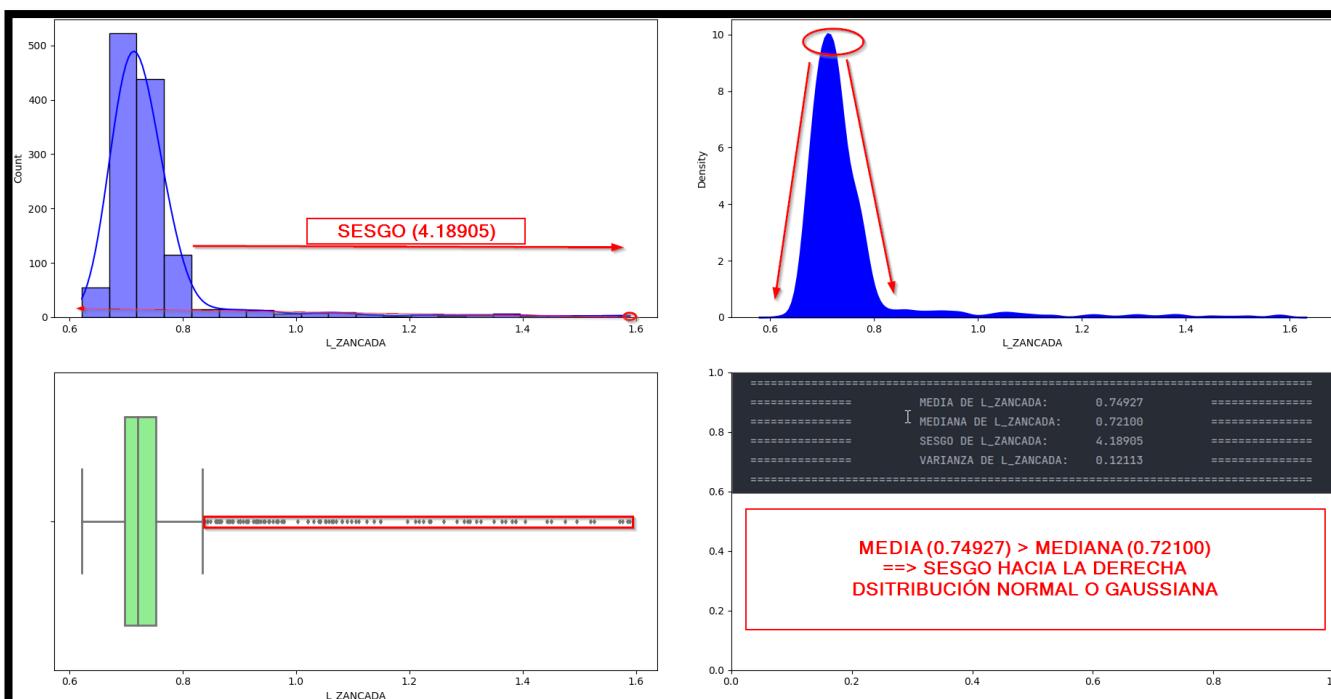
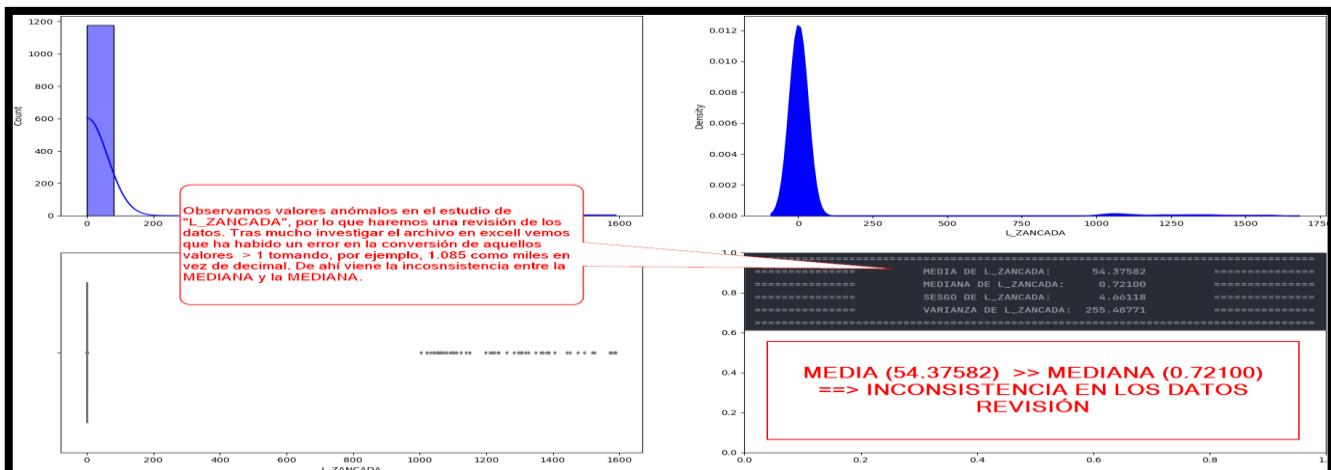
Estudio Visualización "L_ZANCADA"

```
# Estudio Visualización "L_ZANCADA"
# Métricas L_ZANCADA
print("===== MÉTRICAS DE L_ZANCADA =====")
print(f"===== MEDIA DE L_ZANCADA: {data['L_ZANCADA'].mean():,.5f}")
print(f"===== MEDIANA DE L_ZANCADA: {data['L_ZANCADA'].median():,.5f}")
print(f"===== SESGO DE L_ZANCADA: {data['L_ZANCADA'].skew():,.5f}")
print(f"===== VARIANZA DE L_ZANCADA: {data['L_ZANCADA'].std():,.5f}")
print("===== ESTADÍSTICAS DE L_ZANCADA =====")

# Visualización Histograma, Densidad y Boxplot de L_ZANCADA
f, axes = plt.subplots(2, 2, figsize=(11.7, 8.27))

sns.histplot(data["L_ZANCADA"], ax=axes[0,0], kde=True, bins=20, color="Blue", fill=True)
sns.kdeplot(data["L_ZANCADA"], ax=axes[0,1], shade=True, color="Blue", fill=True,
             bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)
sns.boxplot(x=data["L_ZANCADA"], ax=axes[1,0], orient="h", color="lightgreen", saturation=1,
            width=0.7, dodge=True, fliersize=3, linewidth=2)
plt.tight_layout()
plt.show()
```

!!!! OJO-ERROR!!!!



Estudio Visualización "RFP"

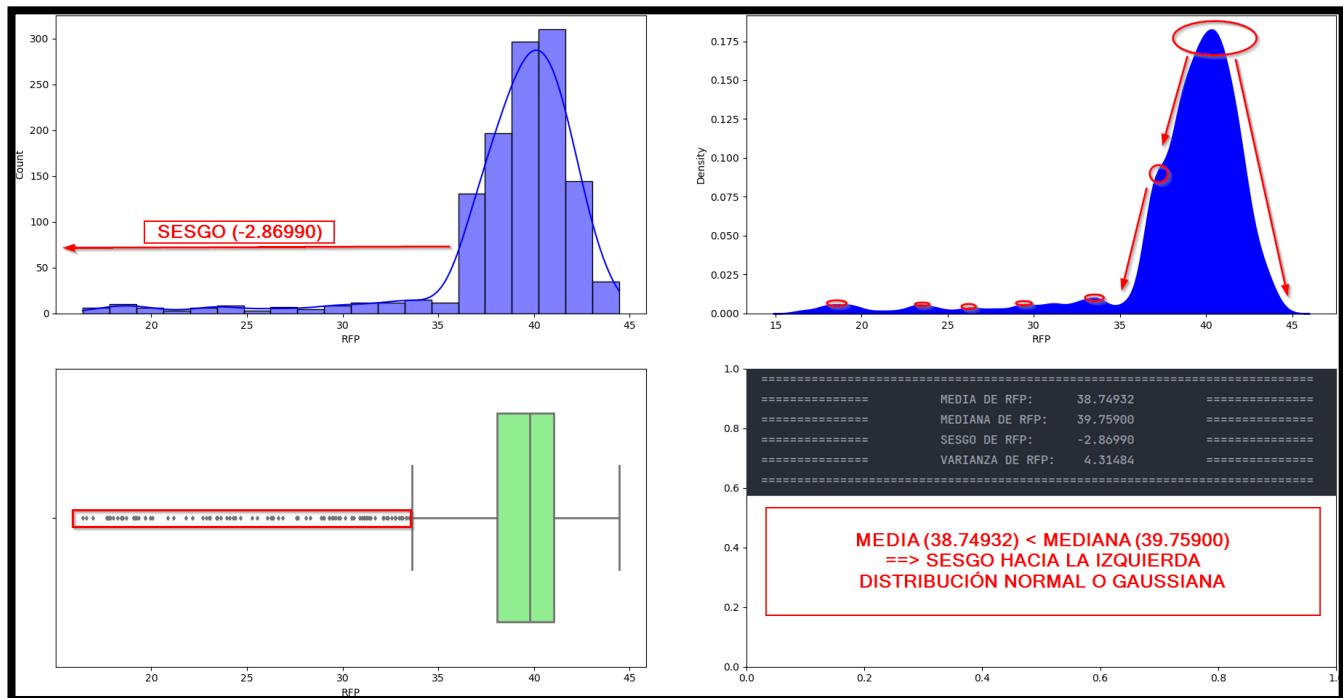
```
# Estudio Visualización "RFP"
# Métricas RFP
print("===== Média de RFP: =====")
print(f" {data['RFP'].mean():,.5f} ")
print("===== Mediana de RFP: =====")
print(f" {data['RFP'].median():,.5f} ")
print("===== Sesgo de RFP: =====")
print(f" {data['RFP'].skew():,.5f} ")
print("===== Varianza de RFP: =====")
print(f" {data['RFP'].std():,.5f} ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de RFP
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["RFP"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["RFP"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["RFP"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "RLSS"

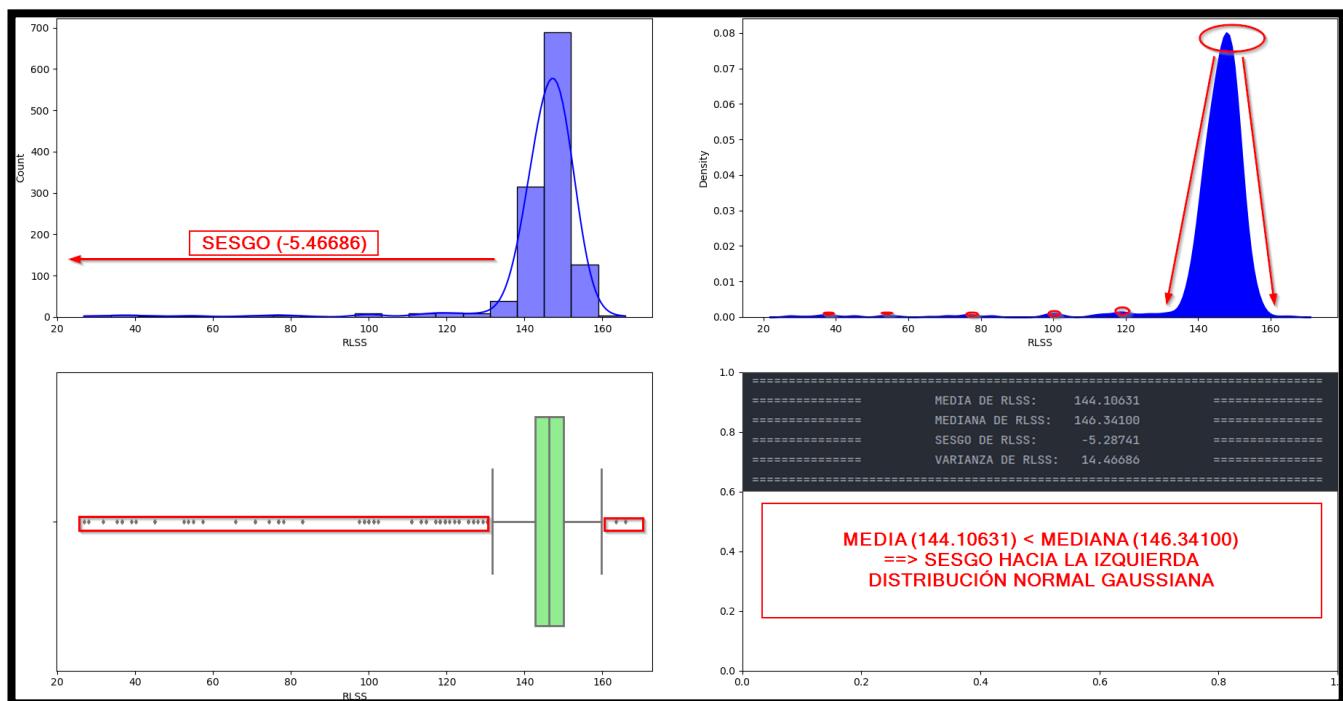
```
# Estudio Visualización "RLSS"
# Métricas RLSS
print("===== Média de RLSS: =====")
print(f" {data['RLSS'].mean():,.5f} ")
print("===== Mediana de RLSS: =====")
print(f" {data['RLSS'].median():,.5f} ")
print("===== Sesgo de RLSS: =====")
print(f" {data['RLSS'].skew():,.5f} ")
print("===== Varianza de RLSS: =====")
print(f" {data['RLSS'].std():,.5f} ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de RLSS
f, axes = plt.subplots(2,2, figsize=(11.7,8.27))

sns.histplot(data["RLSS"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["RLSS"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["RLSS"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "ROV"

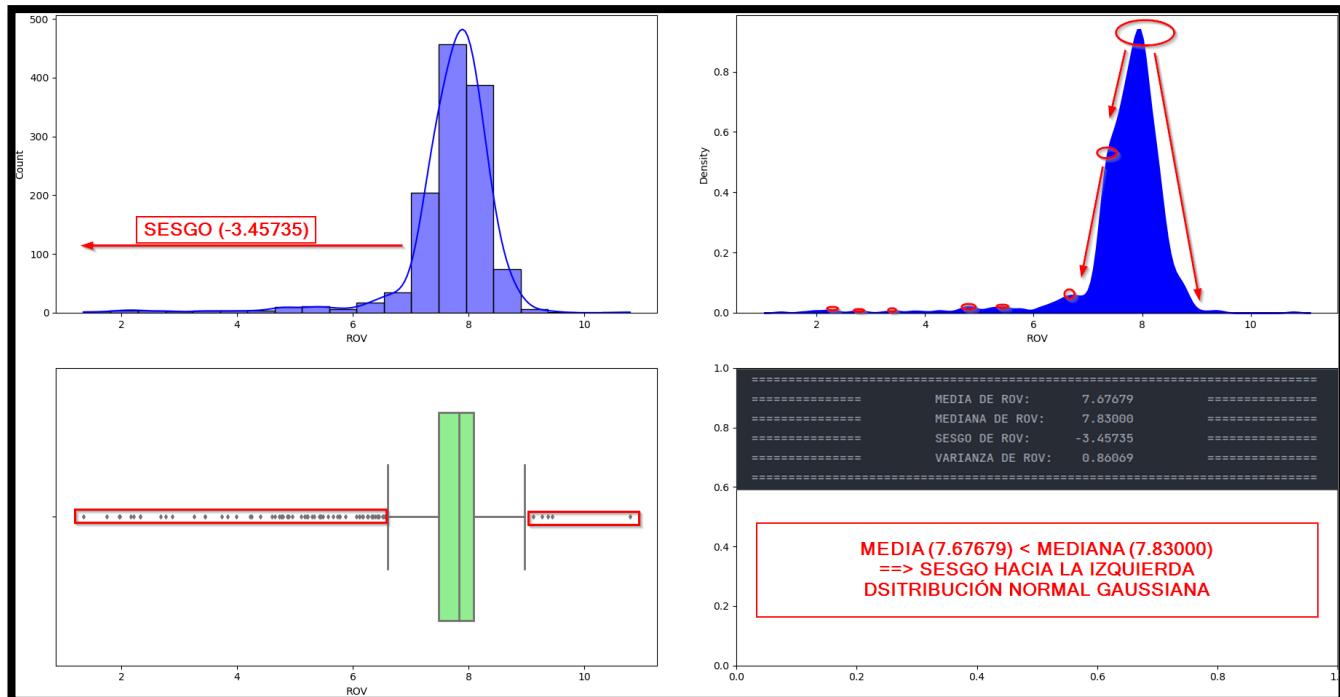
```
# Estudio Visualización "ROV"
# Métricas ROV
print("")
=====
print(f" ===== MÉDIA DE ROV: {data['ROV'].mean():,.5f}")
print(f" ===== MEDIANA DE ROV: {data['ROV'].median():,.5f}")
print(f" ===== SESGO DE ROV: {data['ROV'].skew():,.5f}")
print(f" ===== VARIANZA DE ROV: {data['ROV'].std():,.5f}")
print("")
=====

# Visualización Histograma, Densidad y Boxplot de ROV
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["ROV"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["ROV"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.boxplot(x = data["ROV"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3, linewidth =
2)
plt.tight_layout()
plt.show()
```



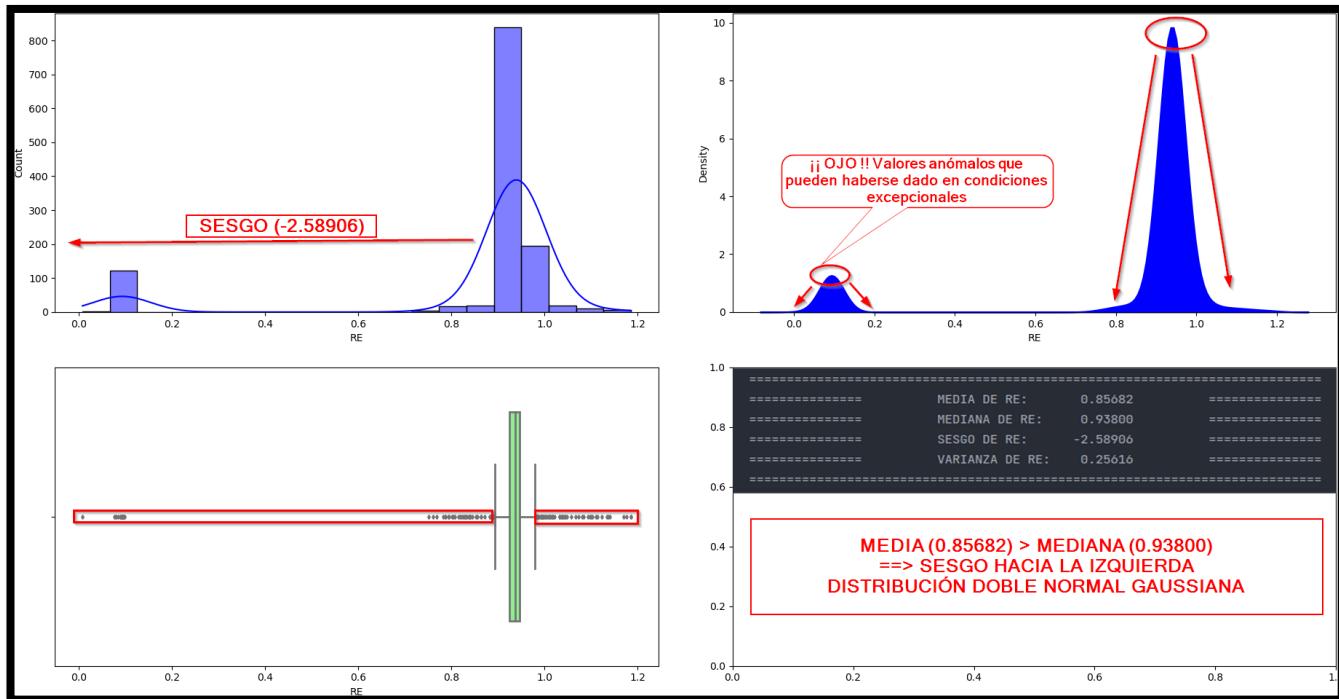
Estudio Visualización "RE"

```
# Estudio Visualización "RE"
# Métricas RE
print("===== Média de RE: =====")
print(f" {data['RE'].mean():,.5f}")
print("===== Mediana de RE: =====")
print(f" {data['RE'].median():,.5f}")
print("===== Sesgo de RE: =====")
print(f" {data['RE'].skew():,.5f}")
print("===== Varianza de RE: =====")
print(f" {data['RE'].std():,.5f}")
print("===== ")
# Visualización Histograma, Densidad y Boxplot de RE
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["RE"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["RE"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.boxplot(x = data["RE"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3, linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "AIRE"

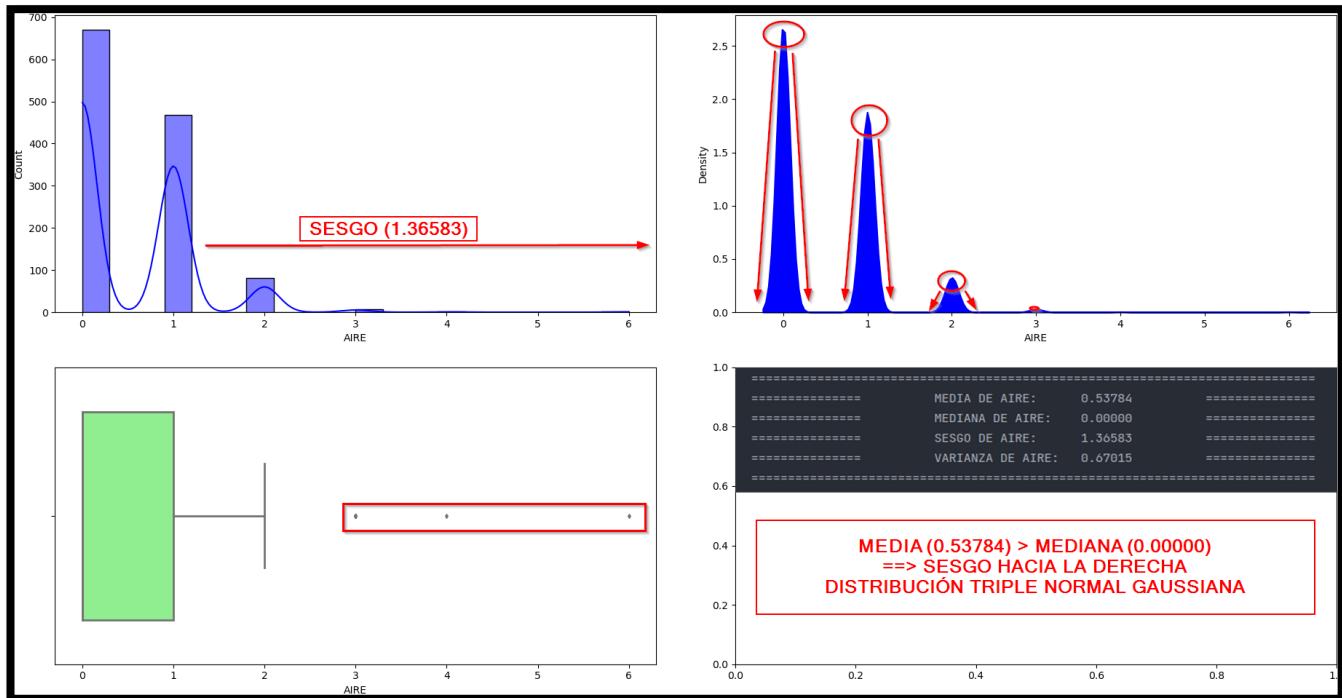
```
# Estudio Visualización "AIRE"
# Métricas AIRE
print("===== Média de AIRE: =====")
print(f" {data['AIRE'].mean():,.5f} =====")
print("===== Mediana de AIRE: =====")
print(f" {data['AIRE'].median():,.5f} =====")
print("===== Sesgo de AIRE: =====")
print(f" {data['AIRE'].skew():,.5f} =====")
print("===== Varianza de AIRE: =====")
print(f" {data['AIRE'].std():,.5f} =====")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de AIRE
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["AIRE"], ax = axes [0,0], kde = True, bins = 20, color="Blue", fill=True)

sns.kdeplot(data["AIRE"], ax = axes [0,1], shade = True, color = "Blue", fill = True,
            bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)

sns.boxplot(x = data["AIRE"], ax = axes [1,0], orient = "h", color = "lightgreen",
            saturation = 1, width = 0.7, dodge = True, fliersize = 3, linewidth = 2)
plt.tight_layout()
plt.show()
```



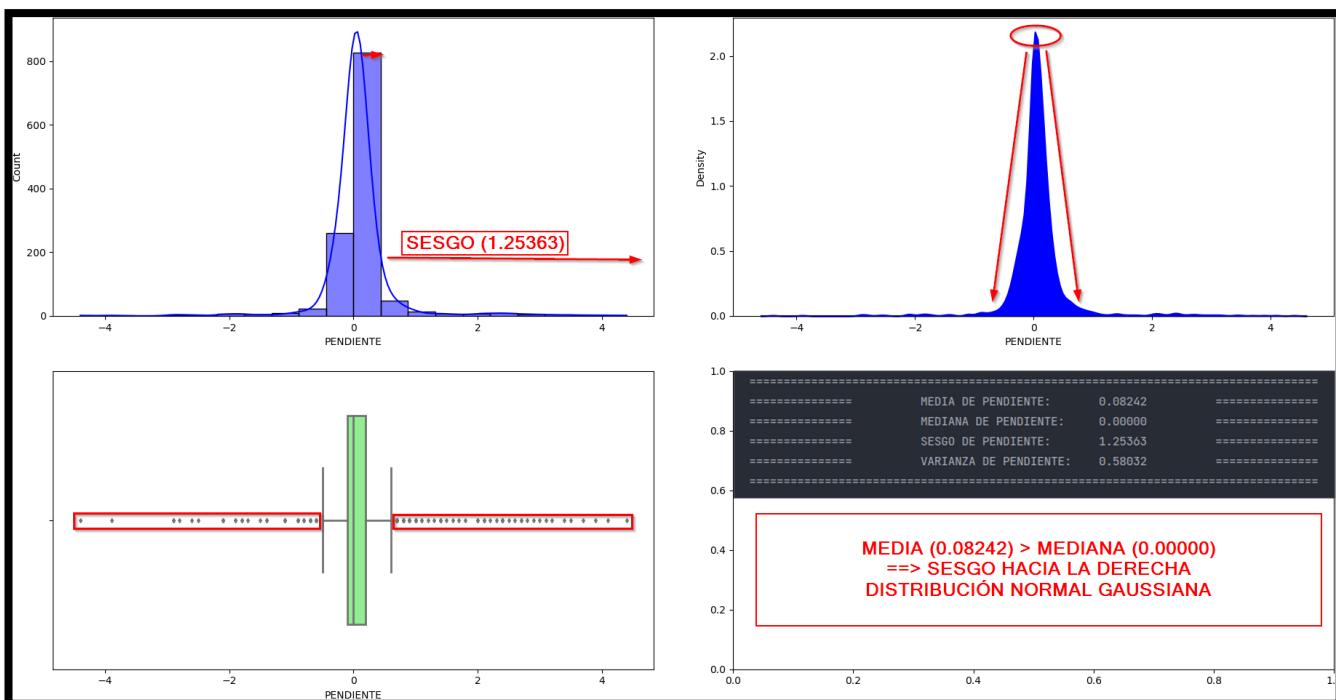
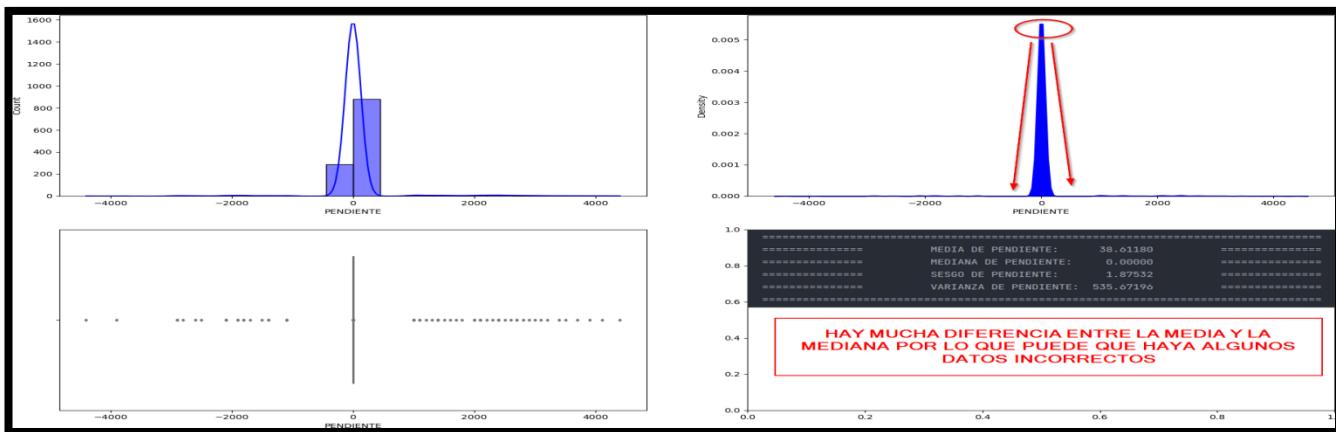
Estudio Visualización "PENDIENTE"

```
# Estudio Visualización "PENDIENTE"
# Métricas PENDIENTE
print("===== ")
print(f"===== MEDIA DE PENDIENTE: {data['PENDIENTE'].mean():,.5f}")
===== ")
print(f"===== MEDIANA DE PENDIENTE: {data['PENDIENTE'].median():,.5f}")
===== ")
print(f"===== SESGO DE PENDIENTE: {data['PENDIENTE'].skew():,.5f}")
===== ")
print(f"===== VARIANZA DE PENDIENTE: {data['PENDIENTE'].std():,.5f}")
===== ")

# Visualización Histograma, Densidad y Boxplot de PENDIENTE
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["PENDIENTE"], ax = axes [0,0], kde = True, bins = 20, color="Blue", fill=True)
sns.kdeplot(data["PENDIENTE"], ax = axes [0,1], shade = True, color = "Blue", fill = True,
bw_adjust=.5, clip_on=False, alpha=1, linewidth=1.5)
sns.boxplot(x = data["PENDIENTE"], ax = axes [1,0], orient = "h", color = "lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3, linewidth = 2)
plt.tight_layout()
plt.show()
```

!!!! OJO-ERROR!!!!



Estudio Visualización "ALTITUD"

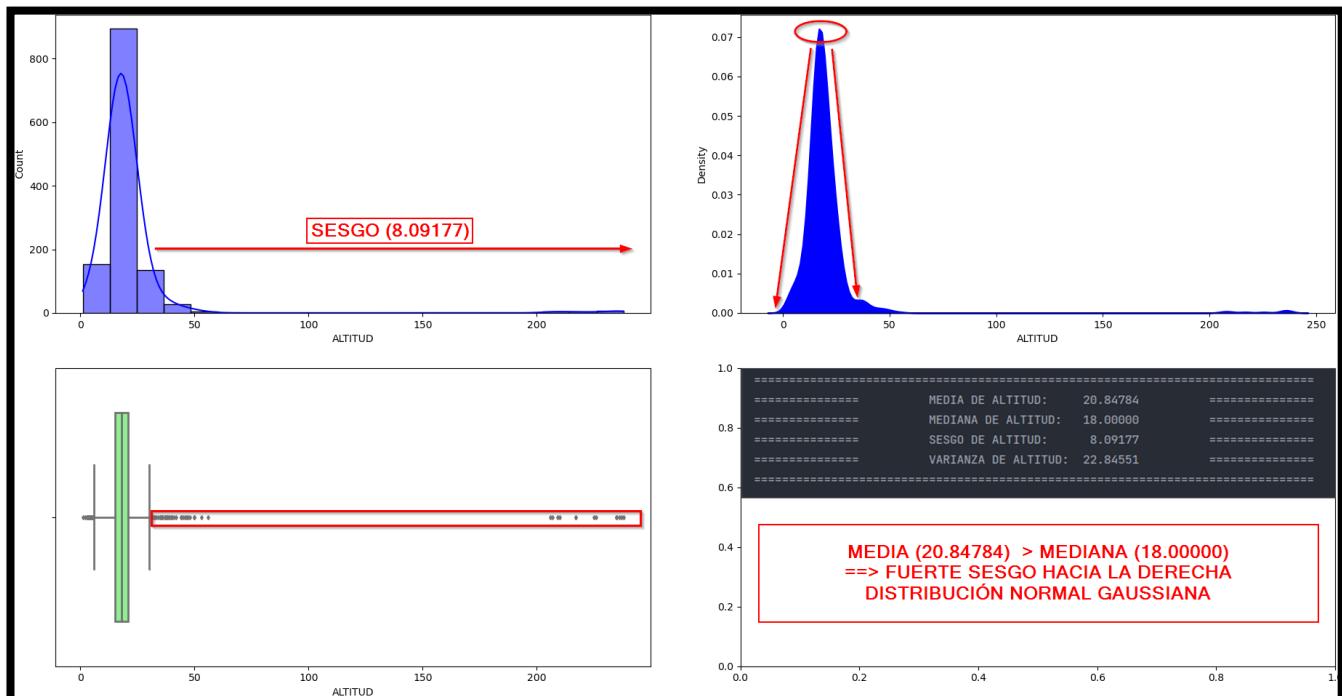
```
# Estudio Visualización "ALTITUD"
# Métricas ALTITUD
print("===== Média de Altitud: =====")
print(f" {data['ALTITUD'].mean():,.5f} ")
print("===== Mediana de Altitud: =====")
print(f" {data['ALTITUD'].median():,.5f} ")
print("===== Sesgo de Altitud: =====")
print(f" {data['ALTITUD'].skew():,.5f} ")
print("===== Varianza de Altitud: =====")
print(f" {data['ALTITUD'].std():,.5f} ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de ALTITUD
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["ALTITUD"],      ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["ALTITUD"],      ax = axes [0,1], shade = True, color = "Blue", fill
= True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["ALTITUD"],   ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "DESNIVEL"

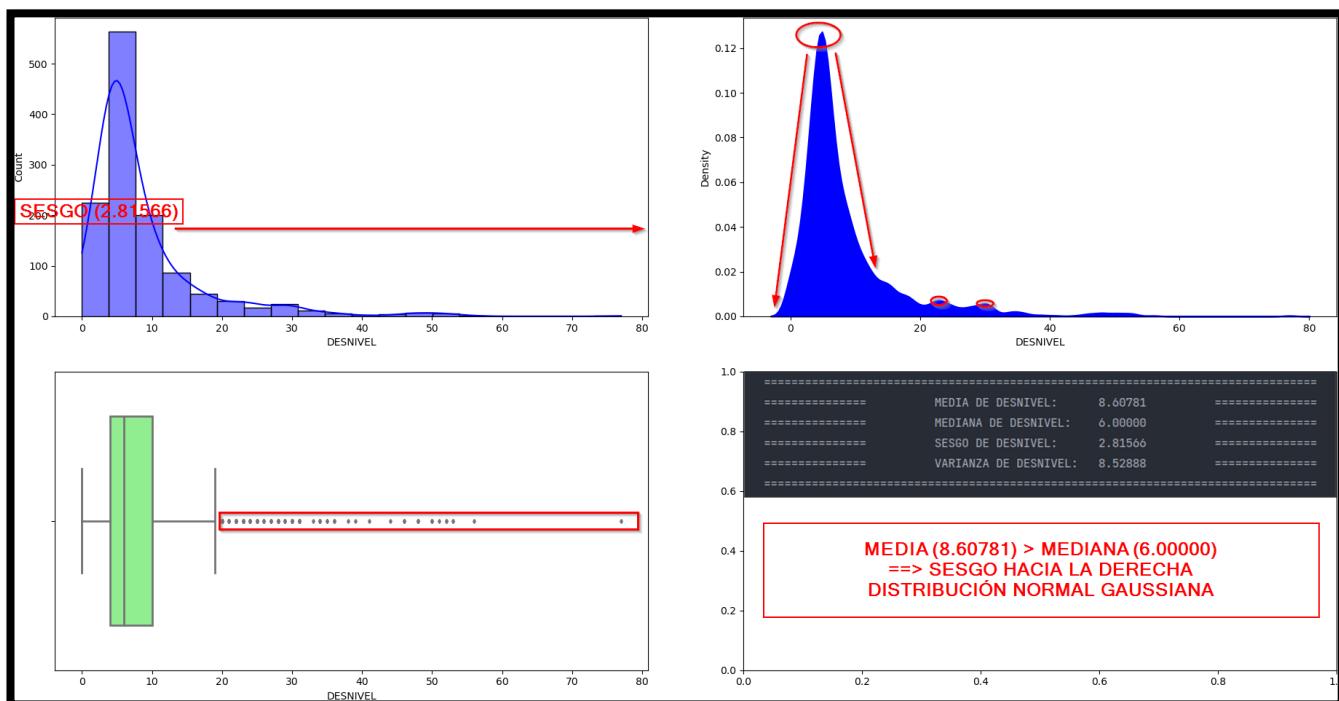
```
# Estudio Visualización "DESNIVEL"
# Métricas DESNIVEL
print("")
=====
print(f" ====== MEDIA DE DESNIVEL: {data['DESNIVEL'].mean():,.5f}")
print(f" ====== MEDIANA DE DESNIVEL: {data['DESNIVEL'].median():,.5f}")
print(f" ====== SESGO DE DESNIVEL: {data['DESNIVEL'].skew():,.5f}")
print(f" ====== VARIANZA DE DESNIVEL: {data['DESNIVEL'].std():,.5f}")
print("")
=====

# Visualización Histograma, Densidad y Boxplot de DESNIVEL
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["DESNIVEL"], ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["DESNIVEL"], ax = axes [0,1], shade = True, color = "Blue",
fill = True, bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["DESNIVEL"], ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "RSS"

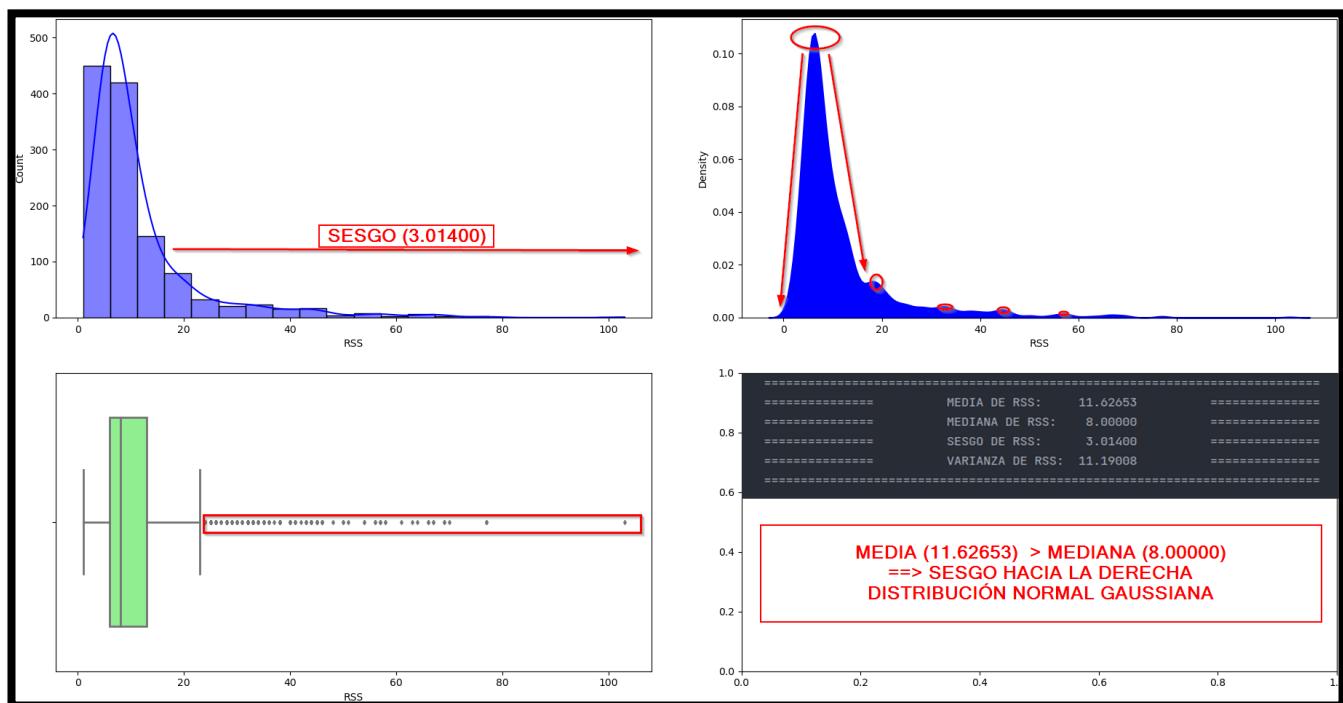
```
# Estudio Visualización "RSS"
# Métricas RSS
print("===== Média de RSS: =====")
print(f" {data['RSS'].mean():,.5f} ")
print("===== Mediana de RSS: =====")
print(f" {data['RSS'].median():,.5f} ")
print("===== Sesgo de RSS: =====")
print(f" {data['RSS'].skew():,.5f} ")
print("===== Varianza de RSS: =====")
print(f" {data['RSS'].std():,.5f} ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de RSS
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["RSS"], ax = axes [0,0], kde = True, bins = 20, color="Blue",
fill=True)

sns.kdeplot(data["RSS"], ax = axes [0,1], shade = True, color = "Blue", fill =
True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["RSS"], ax = axes [1,0], orient = "h", color = "lightgreen",
saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



Estudio Visualización "DURACION"

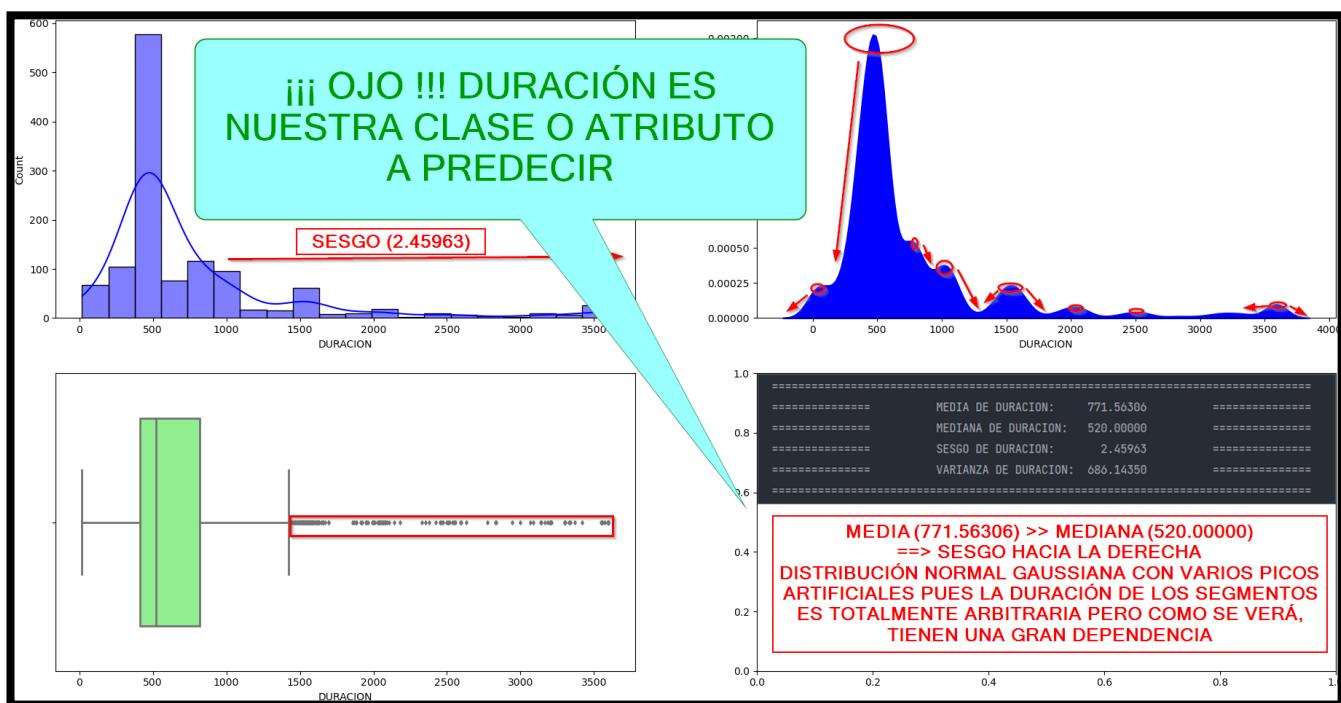
```
# Estudio Visualización "DURACION"
# Métricas DURACION
print("===== Média de Duración: =====")
print(f" {data['DURACION'].mean():,.5f}")
print("===== Mediana de Duración: =====")
print(f" {data['DURACION'].median():,.5f}")
print("===== Sesgo de Duración: =====")
print(f" {data['DURACION'].skew():,.5f}")
print("===== Varianza de Duración: =====")
print(f" {data['DURACION'].std():,.5f}")
print("===== ")
print("===== ")

# Visualización Histograma, Densidad y Boxplot de DURACION
f, axes = plt.subplots(2,2, figsize =(11.7,8.27))

sns.histplot(data["DURACION"], ax = axes [0,0], kde = True, bins = 20,
color="Blue", fill=True)

sns.kdeplot(data["DURACION"], ax = axes [0,1], shade = True, color = "Blue",
fill = True,
bw_adjust=.5, clip_on=False, alpha=1,
linewidth=1.5)

sns.boxplot(x = data["DURACION"], ax = axes [1,0], orient = "h", color =
"lightgreen", saturation = 1,
width = 0.7, dodge = True, fliersize = 3,
linewidth = 2)
plt.tight_layout()
plt.show()
```



CONCLUSIONES EXTRAÍDAS VISUALIZACIÓN

Cuando realizamos el estudio de nuestros datos mediante la visualización de los mismos deberemos:

- Observar coherencia entre el análisis descriptivo y el visual.
- Observar si los histogramas, gráficos de densidad y boxplots están muy escorados hacia algún lado.
- Observar diferentes picos y campanas gaussianas.

Esto nos permitirá conocer sesgos y compararlos con nuestro análisis descriptivo, valores máximos y mínimos e incluso, como en nuestro caso, encontrar errores de formato en nuestro datos como hemos encontrado en “**OSC_VERTICAL**”, “**L_ZANCADA**” y “**PENDIENTE**”.

Visualización Multivariable

Los gráficos multivariados son los gráficos en los que podemos analizar la relación o interacciones entre atributos. El objetivo es aprender algo acerca de la distribución, la tendencia y la distribución en grupos de datos, generalmente pares de atributos.

Matriz de correlación

Podemos ver que la matriz es simétrica, es decir, la parte inferior izquierda de la matriz es la misma que la parte superior derecha. Esto es útil ya que podemos ver dos vistas diferentes en los mismos datos en una parcela. También podemos ver que cada variable está perfectamente correlacionada positivamente entre sí (como era de esperar) en la línea diagonal desde la parte superior izquierda a la parte inferior derecha.

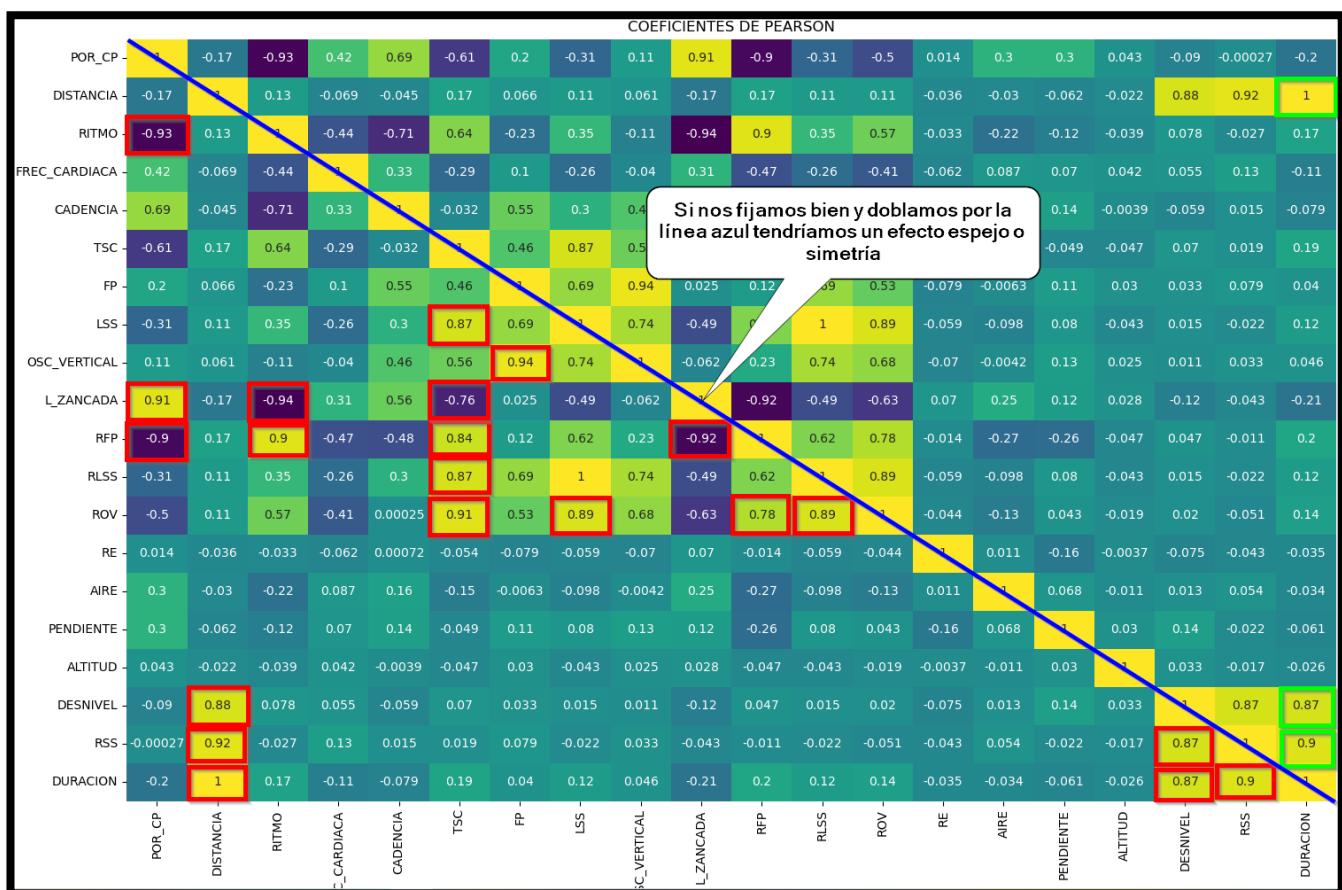
Coeficientes de "Pearson"

Mediante la visualización de los coeficientes de Pearson de las diferentes características, podremos(a priori), conocer la correlación existente entre las diferentes características y así, a un simple golpe de vista ver las características candidatas a ser eliminadas por tener una fuerte relación entre ellas o simplemente por sólo aportar "ruido" a nuestro conjunto de datos. A grandes rasgos, aplicaremos los siguientes criterios tanto para Person como para Spearman:

- Características candidatas a ser eliminadas coeficiente > 0.75 (ROJO)
- Características candidatas a ser eliminadas coeficiente < -0.75 (ROJO)

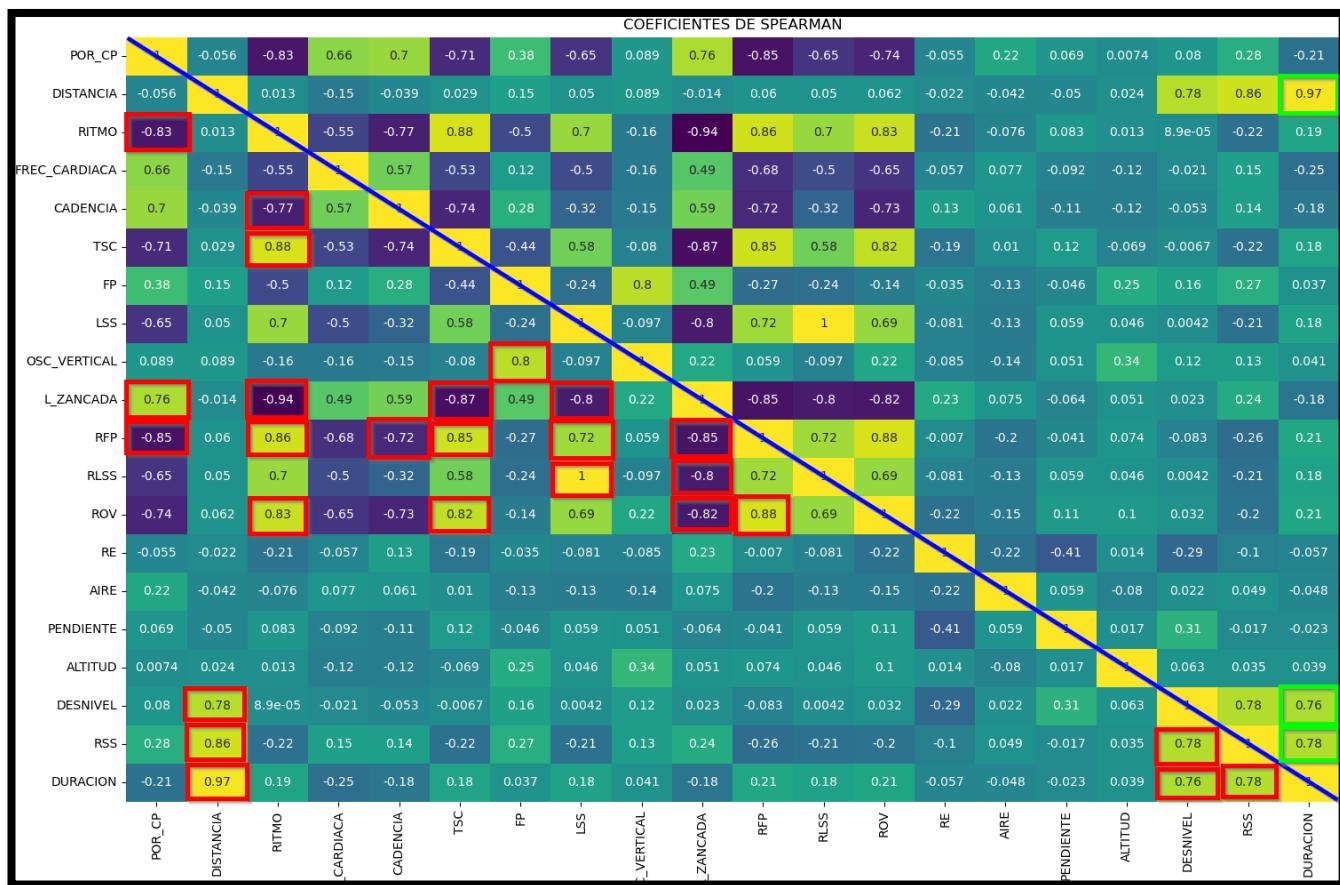
Una cuestión muy diferente serán los coeficientes que obtengamos en nuestra clase a predecir DURACION con las otras características y que cuanto más cerca de 1 estén esos coeficientes, serán mejores candidatas a "no ser eliminadas". (VERDE)

```
# Visualización Multivariable
# **** CORRELACIÓN ENTRE LAS DIFERENTES CARACTERÍSTICAS ****
# ***** COEFICIENTE DE PEARSON ****
print("=====")
print("===== COEFICIENTES DE PEARSON")
print("=====")
print("=====")
pearson = data.corr(method="pearson")
# Graficamos la matriz de correlación (Pearson)
plt.figure(figsize=(14,14))
sns.heatmap(pearson, vmax = 1, annot=True, cmap='viridis')
plt.title("COEFICIENTES DE PEARSON")
plt.tight_layout()
plt.show()
```



Coefficientes de "Spearman"

```
# ****CORRELACIÓN ENTRE LAS DIFERENTES CARACTERÍSTICAS*****
# COEFICIENTE DE SPEARMAN
print("====")
print("===="                                     COEFICIENTES DE SPEARMAN)
print("====")
print("====")
pearson = data.corr(method="spearman")
# Graficamos la matriz de correlación (Spearman)
plt.figure(figsize=(14,14))
sns.heatmap(pearson, vmax =1, annot=True, cmap='viridis')
plt.title("COEFICIENTES DE SPEARMAN")
plt.tight_layout()
plt.show()
```



Gráficos de Matriz de Dispersión

Al igual que el diagrama de matriz de correlación anterior, la matriz de diagrama de dispersión es simétrica. Esto es útil para mirar las relaciones por pares desde diferentes perspectivas. Debido a que no tiene mucho sentido dibujar un diagrama de dispersión de cada variable consigo mismo, la diagonal muestra histogramas de cada atributo.

Para poder ser visibles y entendibles las realizaremos en grupos de 4 con la clase "DURACION"

Función para Gráficos de Matriz de Dispersión

```
# FUNCIÓN PARA OPTIMIZAR CÓDIGO
def matrizcorrelacion(data_grupo):

    g = sns.PairGrid(data_grupo,
                      palette = None, hue_kws = None,
                      vars = None, corner = True, diag_sharey = True,
                      height = 4, aspect = 1, layout_pad = 0.5,
                      despine = True, dropna = False, size = None)

    g.map_diag(sns.histplot, color="red")

    g.map_offdiag(sns.scatterplot, color="blue")

    g.add_legend()

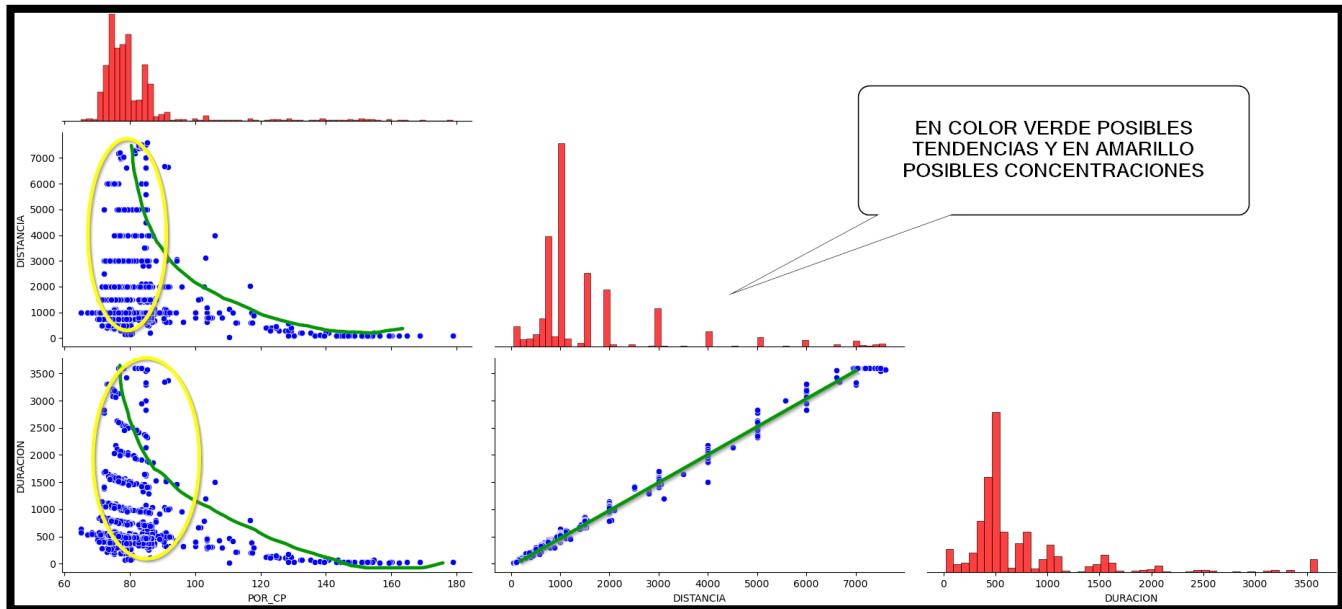
    plt.tight_layout()
    plt.show()
```

Creación de grupos para graficar Matrices de dispersión

```
# CREACIÓN DE DIFERENTES GRUPOS PARA EVITAR SATURACIÓN
df_grupo1 = data[["POR_CP","DISTANCIA","DURACION"]]
df_grupo2 = data[["RITMO","FREC_CARDIACA","DURACION"]]
df_grupo3 = data[["CADENCIA","TSC","DURACION"]]
df_grupo4 = data[["FP","LSS","DURACION"]]
df_grupo5 = data[["OSC_VERTICAL","L_ZANCADA","DURACION"]]
df_grupo6 = data[["RFP","RLSS","DURACION"]]
df_grupo7 = data[["ROV","RE","DURACION"]]
df_grupo8 = data[["AIRE","PENDIENTE","DURACION"]]
df_grupo9 = data[["ALTITUD","DESNIVEL","DURACION"]]
df_grupo10 = data[["RSS","DURACION"]]
```

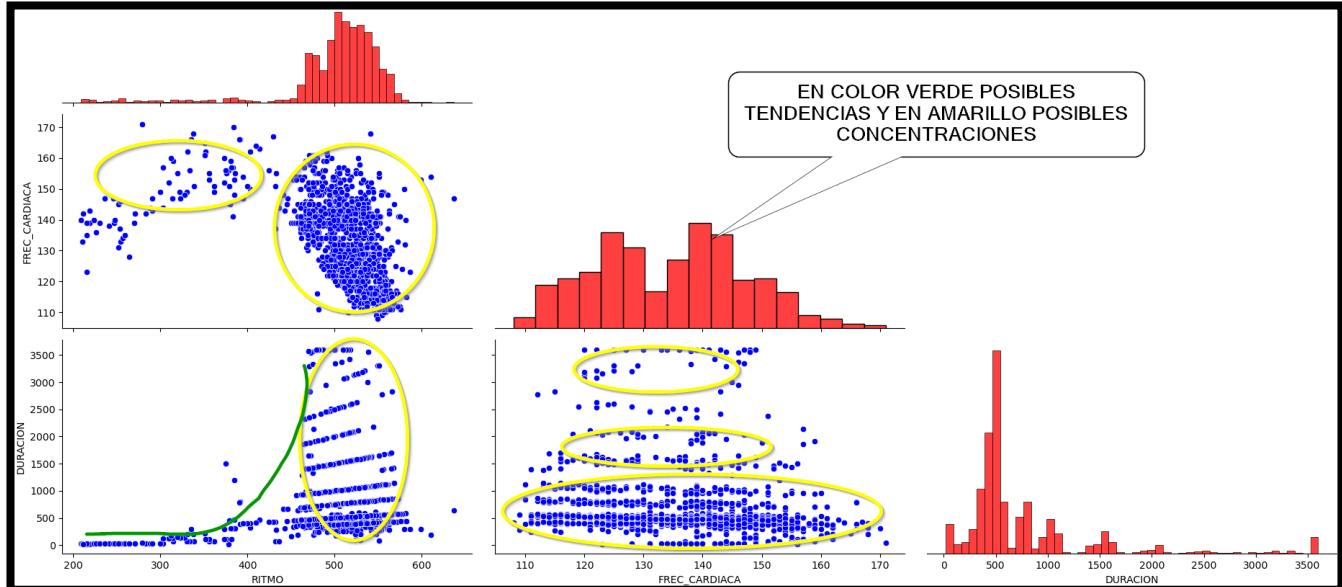
Grupo 1 "POR_CP", "DISTANCIA", "DURACION"

```
# *****  
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 1 *****  
# *****  
matrizcorrelacion(df_grupo1)
```



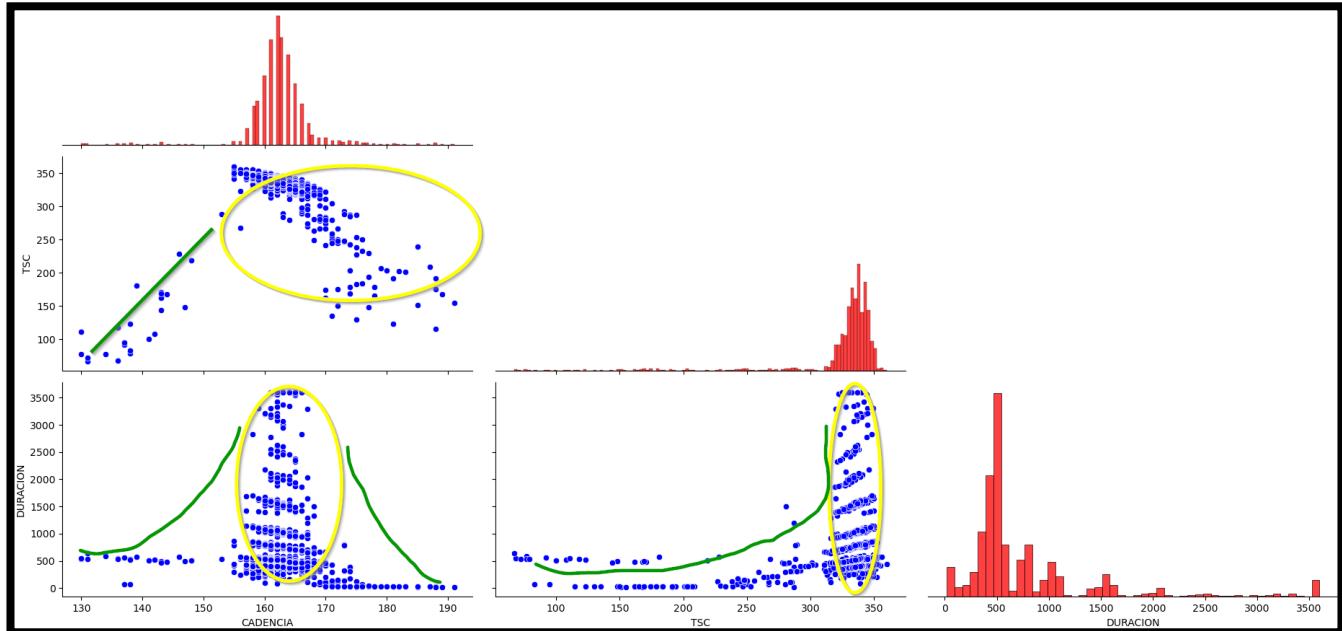
Grupo 2 "RITMO", "FREC_CARDIACA", "DURACION"

```
# *****  
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 2 *****  
# *****  
matrizcorrelacion(df_grupo2)
```



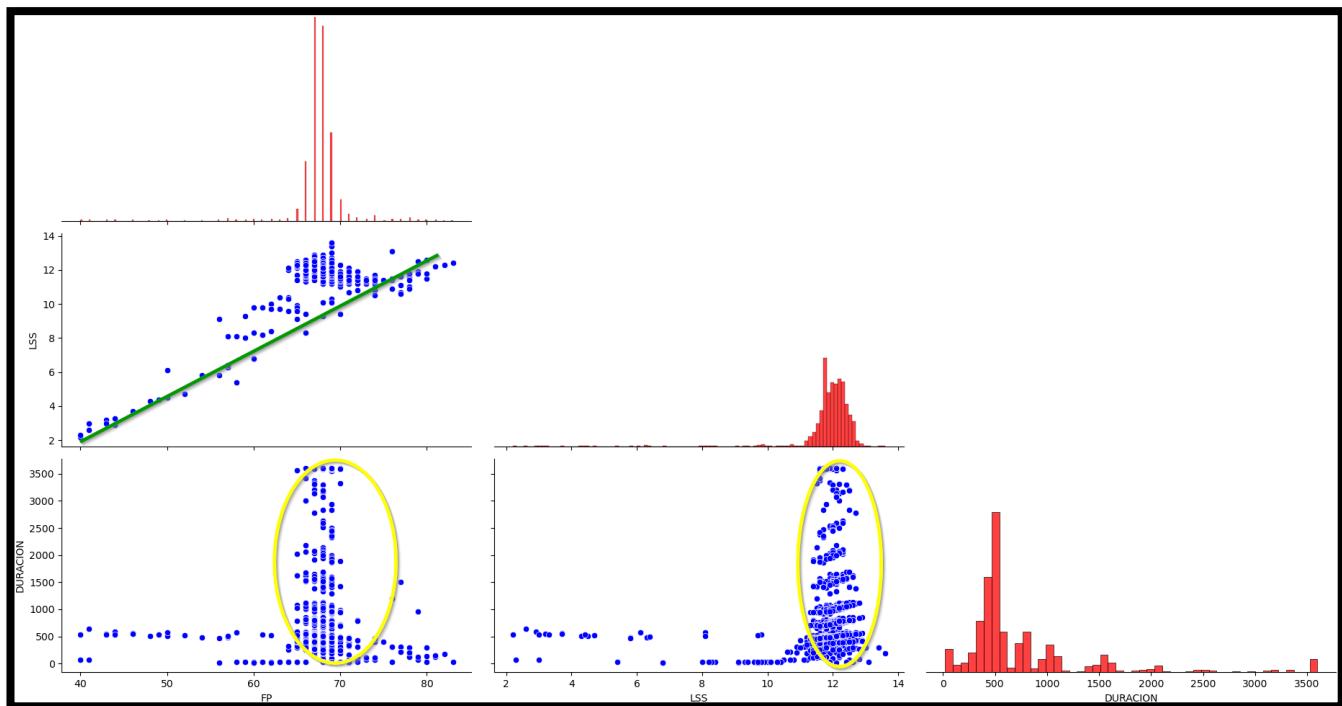
Grupo 3 "CADENCIA", "TSC", "DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 3 ****
# ****
matrizcorrelacion(df_grupo3)
```



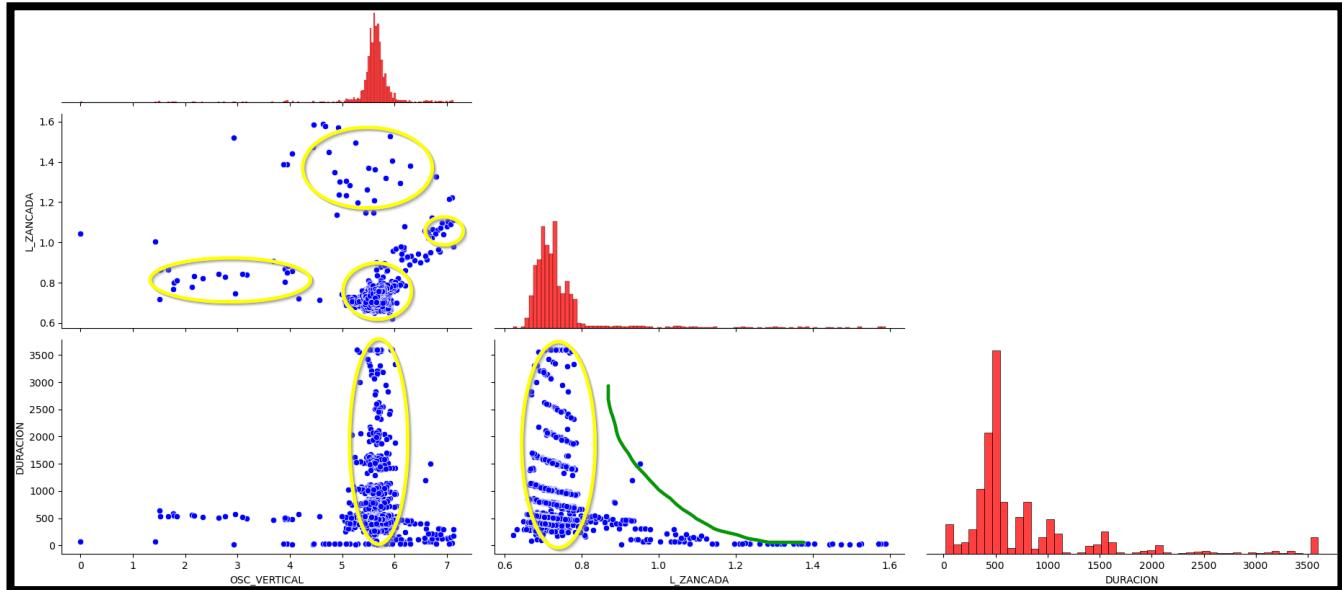
Grupo 4 "FP", "LSS", "DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 4 ****
# ****
matrizcorrelacion(df_grupo4)
```



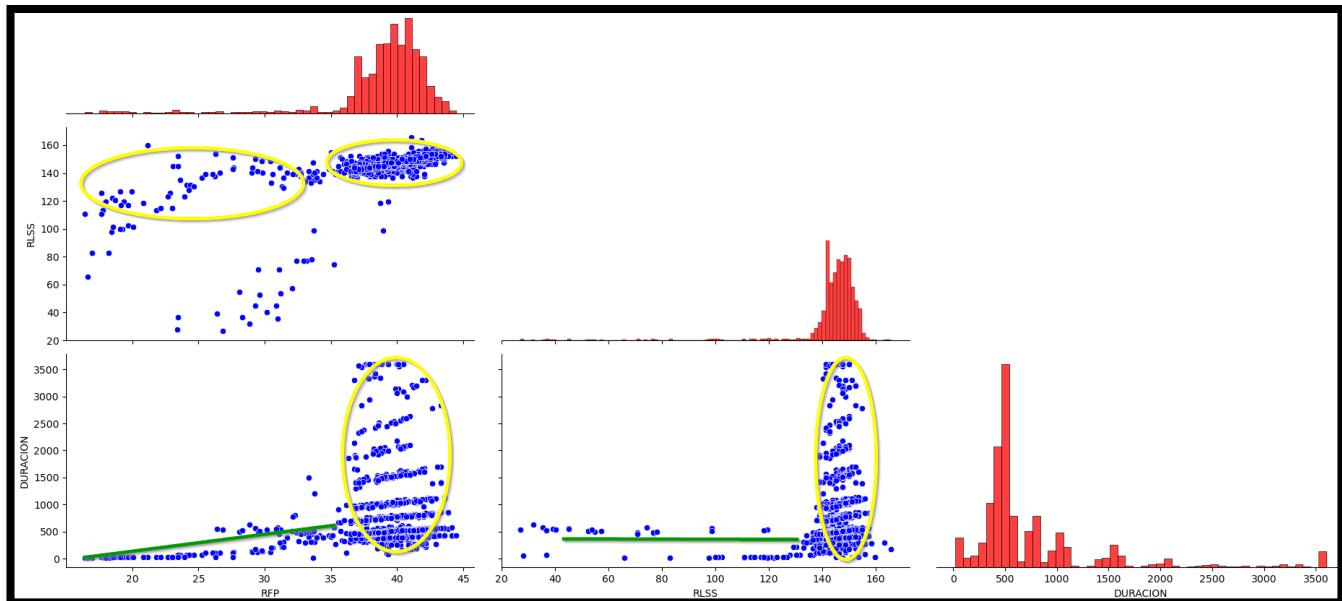
Grupo 5 "OSC_VERTICAL","L_ZANCADA","DURACION"

```
# *****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 5 *****
# *****
matrizcorrelacion(df_grupo5)
```



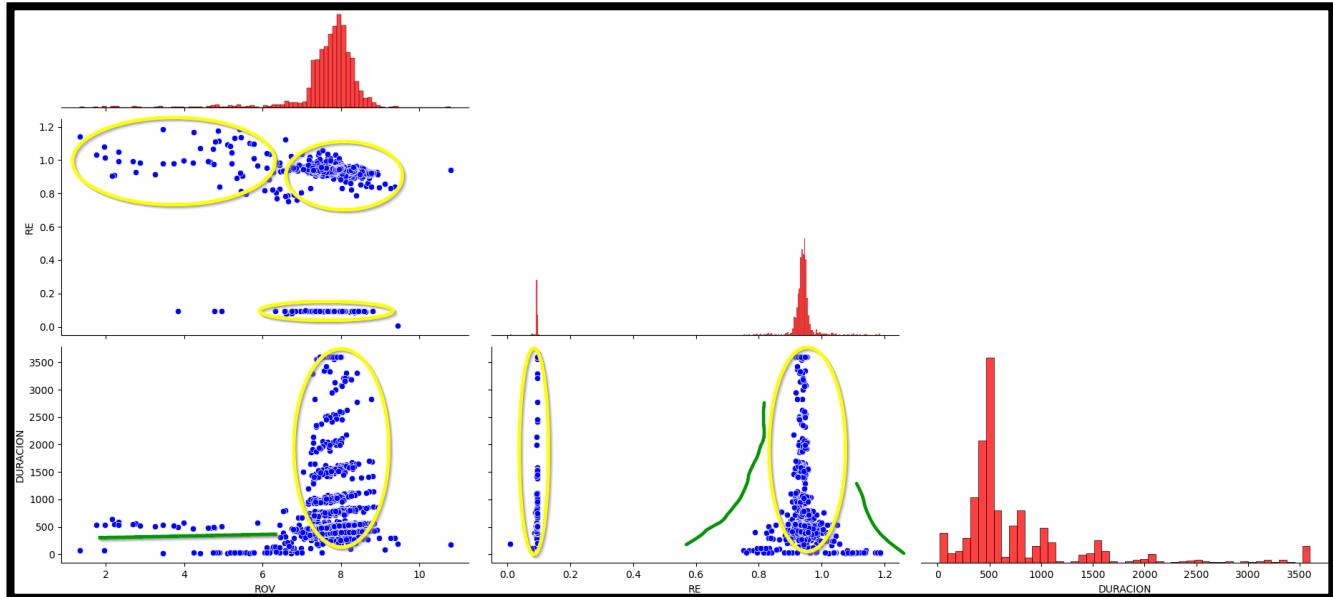
Grupo 6 "RFP","RLSS","DURACION"

```
# *****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 6 *****
# *****
matrizcorrelacion(df_grupo6)
```



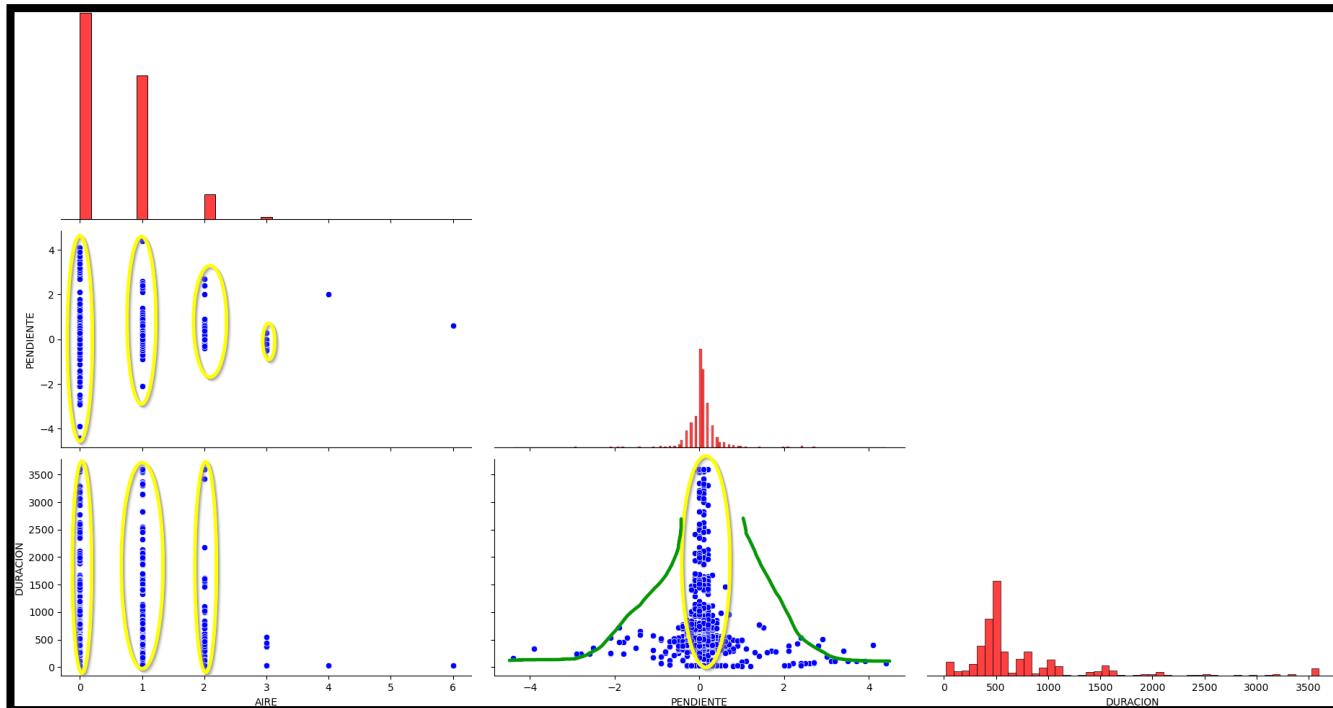
Grupo 7 "ROV", "RE", "DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 7 ****
# ****
matrizcorrelacion(df_grupo7)
```



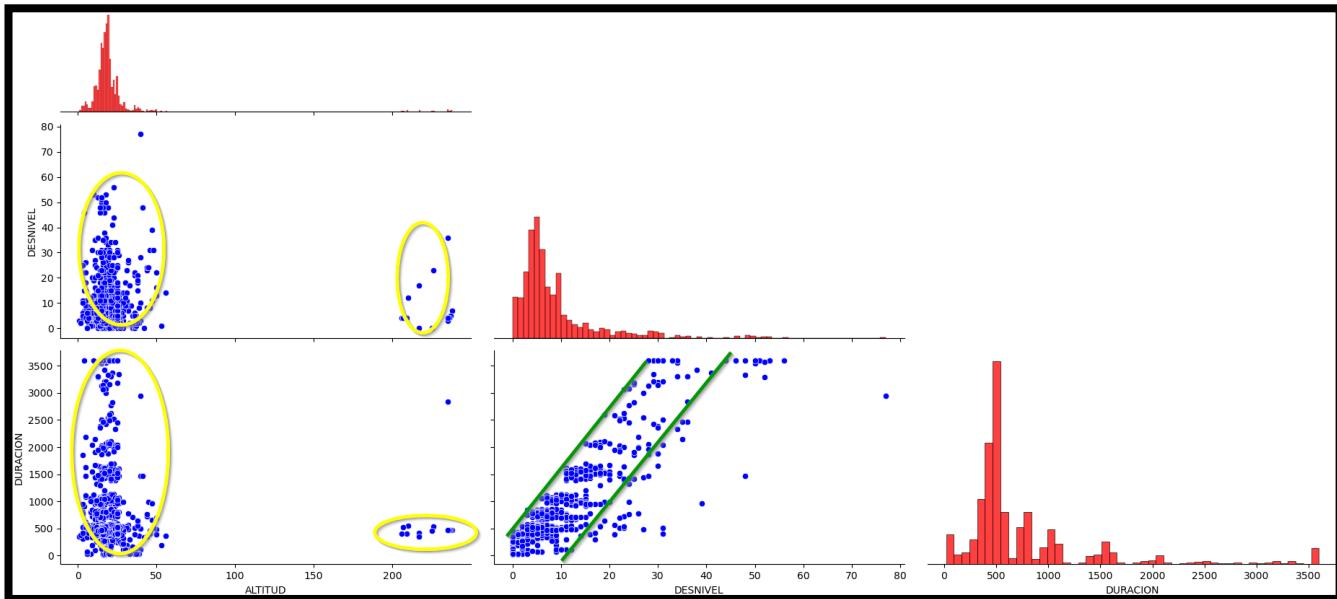
Grupo 8 "AIRE", "PENDIENTE", "DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 8 ****
# ****
matrizcorrelacion(df_grupo8)
```



Grupo 9 "ALTITUD","DESNIVEL","DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 9 ****
# ****
matrizcorrelacion(df_grupo9)
```



Grupo 10 "RSS","DURACION"

```
# ****
# ***** GRÁFICOS DE MATRIZ DE CORRELACIÓN GRUPO 10 ****
# ****
matrizcorrelacion(df_grupo10)
```

