

[Escriba aquí una descripción breve del documento. Normalmente, una descripción breve es un resumen corto del contenido del documento. Escriba aquí una descripción breve del documento. Normalmente, una descripción breve es un resumen corto del contenido del documento.]

Fase de Análisis de Datos

[Escriba el subtítulo del documento]

David

Fase de Análisis de Datos

FASE DE ANÁLISIS DE DATOS	2
Librerías necesarias	2
Análisis inicial o básico	2
Algunas métricas a tener en cuenta	4
Búsqueda de valores nulos o faltantes	5
Correlación entre las características	6
Coeficiente de Pearson	6
Coeficiente de Spearman	7
Sesgo (skew)	9
Gráficas de Distribución y Densidad	10
Cox-Box-Densidad	10
POR_CP:	11
DISTANCIA:	11
DURACION	11
FREC_CARDICA:	12
CADENCIA:	12
TSC:	12
FP:	13
LSS:	13
OSC_VERTICAL:	13
L_ZANCADA:	14
RFP:	14
RLSS:	14
ROV:	15
RE:	15
AIRE:	15
PENDIENTE	16
ALTITUD:	16
DESNIVEL:	16
RLSS:	17
RITMO:	17

FASE DE ANÁLISIS DE DATOS

En esta fase intentaremos extraer información de nuestros datos para poder entenderlos mejor la distribución de los mismos. Para ello deberemos utilizar algunas herramientas o instrucciones que faciliten esta labor.

Los datos serán cargados en un dataframe y se encuentran en el archivo "SEGMENTOS_csv.csv"

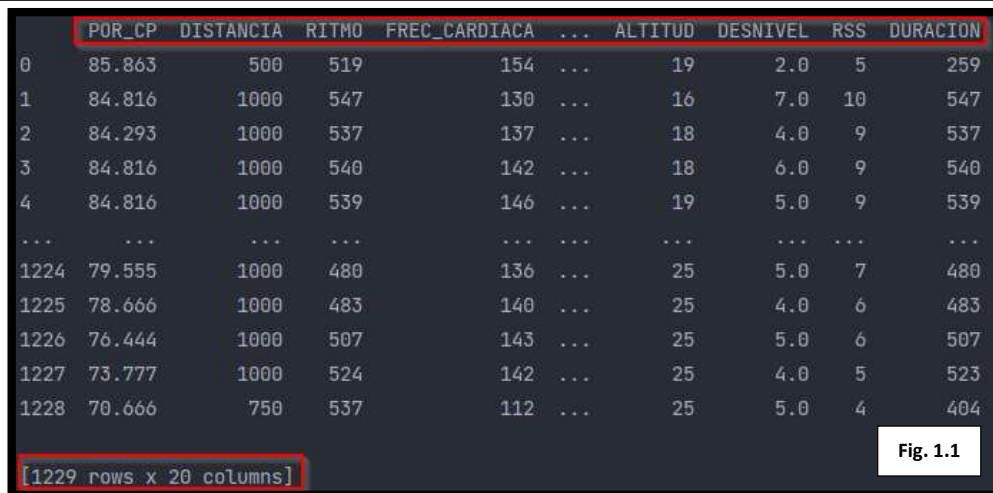
Librerías necesarias

Importaremos las librerías necesarias para nuestro proyecto, se podrán ir añadiendo según necesidades.

```
import pandas as pd
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.stats.anova import anova_lm
```

Análisis inicial o básico

```
# Cargamos los datos contenidos en "SEGMENTOS_csv.csv"
data = pd.read_csv('SEGMENTOS_csv.csv')
print(data)
# Mediante head(15) visualizaremos los 15 primeros
# registros para hacernos una idea
print(data.head(15))
```



	POR_CP	DISTANCIA	RITMO	FREC_CARDIACA	...	ALTITUD	DESNIVEL	RSS	DURACION
0	85.863	500	519	154	...	19	2.0	5	259
1	84.816	1000	547	130	...	16	7.0	10	547
2	84.293	1000	537	137	...	18	4.0	9	537
3	84.816	1000	540	142	...	18	6.0	9	540
4	84.816	1000	539	146	...	19	5.0	9	539
...
1224	79.555	1000	480	136	...	25	5.0	7	480
1225	78.666	1000	483	140	...	25	4.0	6	483
1226	76.444	1000	507	143	...	25	5.0	6	507
1227	73.777	1000	524	142	...	25	4.0	5	523
1228	70.666	750	537	112	...	25	5.0	4	404
[1229 rows x 20 columns]									

Fig. 1.1

Conclusiones: cómo podemos observar en *fig. 1.1* nuestro dataframe (data) contiene 1229 registros y 20 columnas (atributos/características)

Fase de Análisis de Datos

	POR_CP	DISTANCIA	RITMO	FREC_CARDIACA	...	ALTITUD	DESNIVEL	RSS	DURACION
0	85.863	500	519	154	...	19	2.0	5	259
1	84.816	1000	547	130	...	16	7.0	10	547
2	84.293	1000	537	137	...	18	4.0	9	537
3	84.816	1000	540	142	...	18	6.0	9	540
4	84.816	1000	539	146	...	19	5.0	9	539
5	84.816	1000	542	150	...	18	7.0	9	542
6	84.816	1000	537	152	...	18	2.0	9	537
7	85.863	600	524	155	...	19	3.0	6	317
8	85.340	530	561	128	...	16	5.0	5	300
9	84.816	5570	538	145	...	18	27.0	51	3000
10	84.816	2000	545	134	...	17	11.0	18	1082
11	84.816	2000	540	144	...	19	11.0	18	1080
12	84.816	2000	539	151	...	18	9.0	18	1079
13	84.816	3000	541	136	...	17	17.0	28	1622
14	84.816	3000	539	149	...	18	13.0	28	1619

[15 rows x 20 columns]

Fig. 1.2

Información sobre filas, columnas, tipos, etc

```
# Mediante shape() sabremos las filas y columnas
# que componen nuestro dataframe
print(data.shape)
# Mediante dtypes conoceremos los tipos de datos de
# las diferentes características
print(data.dtypes)
# Formato de los datos
print(f"Cantidad de filas y columnas: {data.shape}")
print(f"Nombre de las columnas: {data.columns}")
print(data.info())
```

```
(1229, 20)
POR_CP      float64
DISTANCIA    int64
RITMO        int64
FREC_CARDIACA  int64
CADENCIA     int64
TSC          int64
FP           int64
LSS          float64
OSC_VERTICAL float64
L_ZANCADA    float64
RFP          float64
RLSS         float64
ROV          float64
RE           float64
AIRE         int64
PENDIENTE    float64
ALTITUD      int64
DESNIVEL     float64
RSS          int64
DURACION     int64
dtype: object
```

Fig. 2.1

Fase de Análisis de Datos

```
cantidad de filas y columnas: (1229, 20)
Nombre de las columnas: Index(['POR_CP', 'DISTANCIA', 'RITMO', 'FREC_CARDIACA', 'CADENCIA', 'TSC',
                              'FP', 'LSS', 'OSC_VERTICAL', 'L_ZANCADA', 'RFP', 'RLSS', 'ROV', 'RE',
                              'AIRE', 'PENDIENTE', 'ALTITUD', 'DESNIVEL', 'RSS', 'DURACION'],
                              dtype='object')
<class 'pandas.core.frame.DataFrame'>
```

Fig. 2.2

#	Column	Non-Null Count	Dtype
0	POR_CP	1229 non-null	float64
1	DISTANCIA	1229 non-null	int64
2	RITMO	1229 non-null	int64
3	FREC_CARDIACA	1229 non-null	int64
4	CADENCIA	1229 non-null	int64
5	TSC	1229 non-null	int64
6	FP	1229 non-null	int64
7	LSS	1229 non-null	float64
8	OSC_VERTICAL	1229 non-null	float64
9	L_ZANCADA	1229 non-null	float64
10	RFP	1229 non-null	float64
11	RLSS	1229 non-null	float64
12	ROV	1229 non-null	float64
13	RE	1229 non-null	float64
14	AIRE	1229 non-null	int64
15	PENDIENTE	1229 non-null	float64
16	ALTITUD	1229 non-null	int64
17	DESNIVEL	1228 non-null	float64
18	RSS	1229 non-null	int64
19	DURACION	1229 non-null	int64

dtypes: float64(10), int64(10)
memory usage: 192.2 KB

Fig. 2.3

Conclusiones: a primera vista y a grandes rasgos podríamos destacar la “no” existencia de características de tipo “object” (fig. 2.1), la denominación de las 20 características (fig. 2.2) y los tipos de datos de cada una de ellas (fig. 2.3).

Algunas métricas a tener en cuenta

```
# Descripción de los datos y de diferentes métricas
# como media, varianza, percentiles,
# mínimos, máximos, etc.....
pd.set_option('display.width', 100)
pd.set_option('display.precision', 3)
print(data.describe())
```

	POR_CP	DISTANCIA	RITMO	...	DESNIVEL	RSS	DURACION
count	1229.000	1229.000	1229.000	...	1228.000	1229.000	1229.000
mean	81.950	1517.453	501.777	...	8.610	11.627	771.563
std	14.843	1359.908	59.279	...	8.532	11.190	686.143
min	65.137	50.000	209.000	...	0.000	1.000	19.000
25%	74.770	800.000	489.000	...	4.000	6.000	411.000
50%	77.981	1000.000	513.000	...	6.000	8.000	520.000
75%	83.944	1500.000	535.000	...	10.000	13.000	817.000
max	178.899	7600.000	637.000	...	77.000	103.000	3599.000

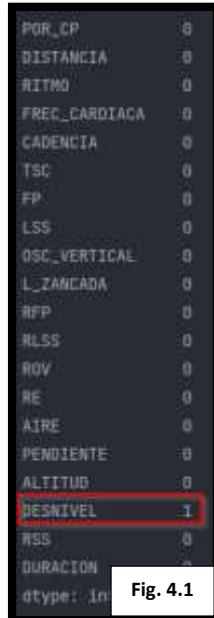
Fig. 3.1

Conclusiones: a priori y, como ejemplo, podemos observar una media (mean) de 1517.453 en la característica “DISTANCIA” con un valor mínimo (min) de 50 y máximo (max) de 7600

diferencias muy grandes en valores absolutos por lo que deberemos tener en cuenta que no se haya "colado" un valor extremo que pueda "meter" ruido en nuestro modelo predictivo (fig. 3.1).

Búsqueda de valores nulos o faltantes

```
# Buscamos datos nulos o faltantes
print(data.isnull().sum())
```



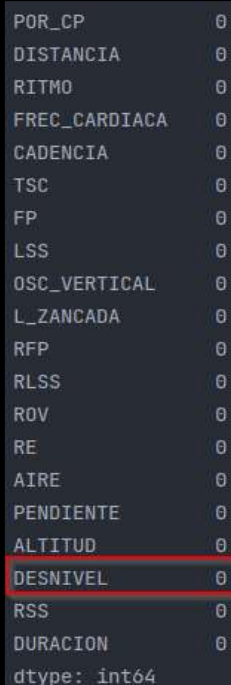
POR_CP	0
DISTANCIA	0
RITMO	0
FREC_CARDIACA	0
CADENCIA	0
TSC	0
FP	0
LSS	0
OSC_VERTICAL	0
L_ZANCADA	0
RFP	0
RLSS	0
ROV	0
RE	0
AIRE	0
PENDIENTE	0
ALTITUD	0
DESNIVEL	1
RSS	0
DURACION	0

dtype: int64

Fig. 4.1

Conclusiones: cómo podemos observar en (fig. 4.1) en la característica "DESNIVEL" existe un registro nulo o vacío, mientras los registros faltantes no superen el 10% de la muestra podríamos sustituirlos por la media aritmética sin que tenga un gran impacto en el modelo.

```
# Completamos los datos nulos con la media de cada uno
data['DESNIVEL'] = data['DESNIVEL'].fillna(data['DESNIVEL'].median())
# Buscamos datos nulos o faltantes
print(data.isnull().sum())
```



POR_CP	0
DISTANCIA	0
RITMO	0
FREC_CARDIACA	0
CADENCIA	0
TSC	0
FP	0
LSS	0
OSC_VERTICAL	0
L_ZANCADA	0
RFP	0
RLSS	0
ROV	0
RE	0
AIRE	0
PENDIENTE	0
ALTITUD	0
DESNIVEL	0
RSS	0
DURACION	0

dtype: int64

Correlación entre las características

Coeficiente de Pearson

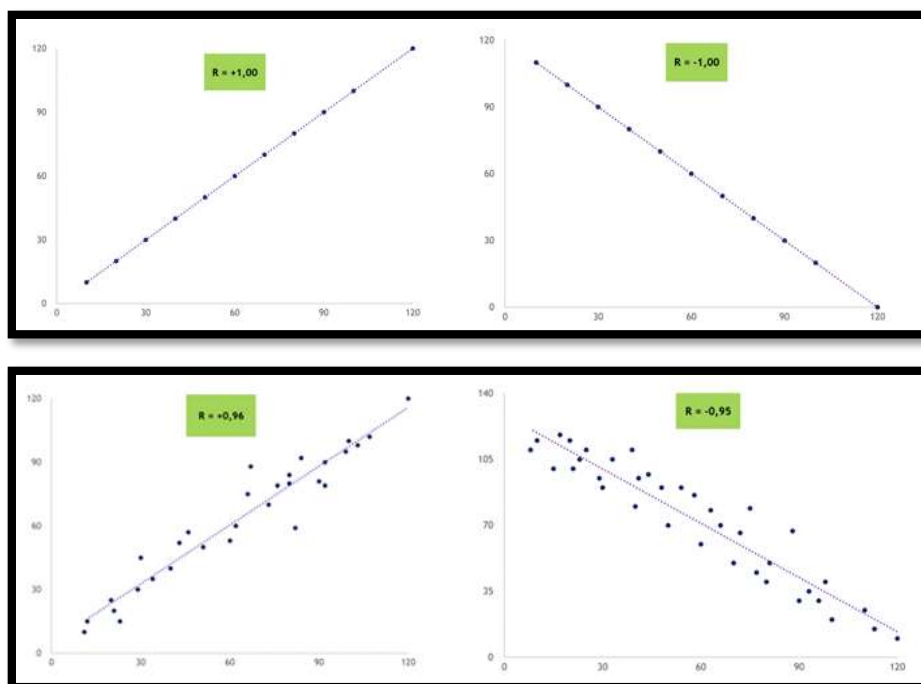
```
# *****  
# ***** CORRELACIÓN ENTRE LAS DIFERENTES CARACTERÍSTICAS *****  
# *****  
# COEFICIENTE DE PEARSON  
pearson = data.corr(method="pearson")  
# Graficamos la matriz de correlación (Pearson)  
plt.figure(figsize=(14,8))  
sns.heatmap(pearson, vmax =1, annot=True, cmap='viridis')  
plt.title("COEFICIENTES DE PEARSON")  
plt.tight_layout()  
plt.show()
```

El coeficiente de correlación de Pearson oscila entre -1 y $+1$:

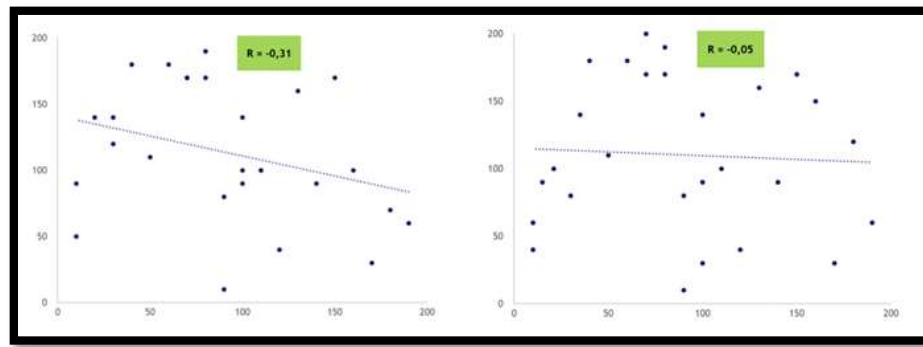
(fuente: <https://www.cimec.es/coeficiente-correlacion-pearson/>)

- Un valor menor que 0 indica que existe una correlación negativa, es decir, que las dos variables están asociadas en sentido inverso. Cuánto más se acerca a -1 , mayor es la fuerza de esa relación invertida (cuando el valor en una sea muy alto, el valor en la otra será muy bajo). Cuando es exactamente -1 , eso significa que tienen una correlación negativa perfecta.
- Un valor mayor que 0 indica que existe una correlación positiva. En este caso las variables estarían asociadas en sentido directo. Cuanto más cerca de $+1$, más alta es su asociación. Un valor exacto de $+1$ indicaría una relación lineal positiva perfecta.
- Finalmente, una correlación de 0, o próxima a 0, indica que no hay relación lineal entre las dos variables.

La representación gráfica de los datos es muy útil para visualizar la relación existente entre las variables, ya que hay que tener en cuenta que a veces existen relaciones entre variables que no son lineales.



Fase de Análisis de Datos



Existe bastante consenso a la hora de interpretar los valores del coeficiente de correlación de Pearson utilizando los siguientes criterios (y considerando los valores absolutos):

- Entre 0 y 0,10: correlación inexistente
- Entre 0,10 y 0,29: correlación débil
- Entre 0,30 y 0,50: correlación moderada
- Entre 0,50 y 1,00: correlación fuerte

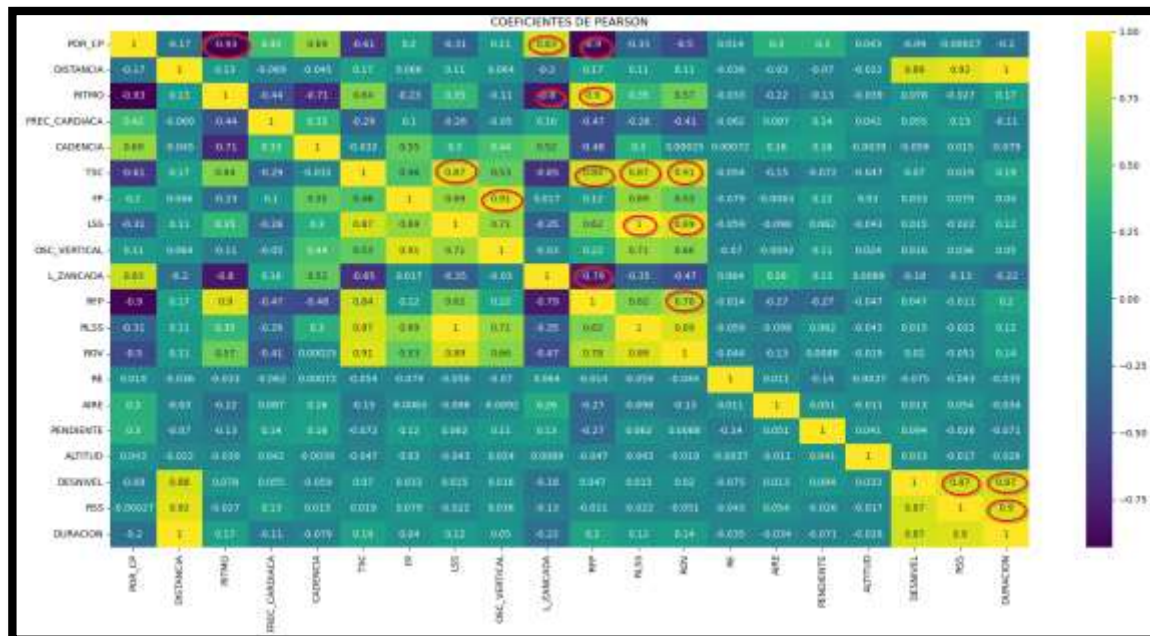


Fig. 5.1

A pesar del consenso existente de considerar una correlación fuerte entre 0.5 y 1.0, a nivel estadística, a nivel de modelos predictivos de Machine Learning es a partir de 0.75 en valores absolutos cuando nos plantearemos la eliminación de alguna característica. Sería condición necesaria y en algunos casos suficiente pero en la mayoría de los casos habría que tener en cuenta otras magnitudes o coeficientes. Algunas correlaciones a examinar (fig. 5.1).

Coeficiente de Spearman

```
# COEFICIENTE DE SPEARMAN
spearman = data.corr(method= "spearman", min_periods = 2)
# Graficamos la matriz de correlación (Spearman)
plt.figure(figsize=(14, 8))
sns.heatmap(spearman, vmax =1, annot=True, cmap='viridis')
plt.title("COEFICIENTES DE SPEARMAN")
plt.tight_layout()
plt.show()
```


(Fuente: <https://www.questionpro.com/blog/es/coeficiente-de-correlacion-de-spearman/>)

Examina la fuerza y la dirección de la relación monótona entre dos variables continuas u ordinales. En una relación monótona, las variables tienden a moverse en la misma dirección relativa, pero no necesariamente a un ritmo constante..

Fuerza

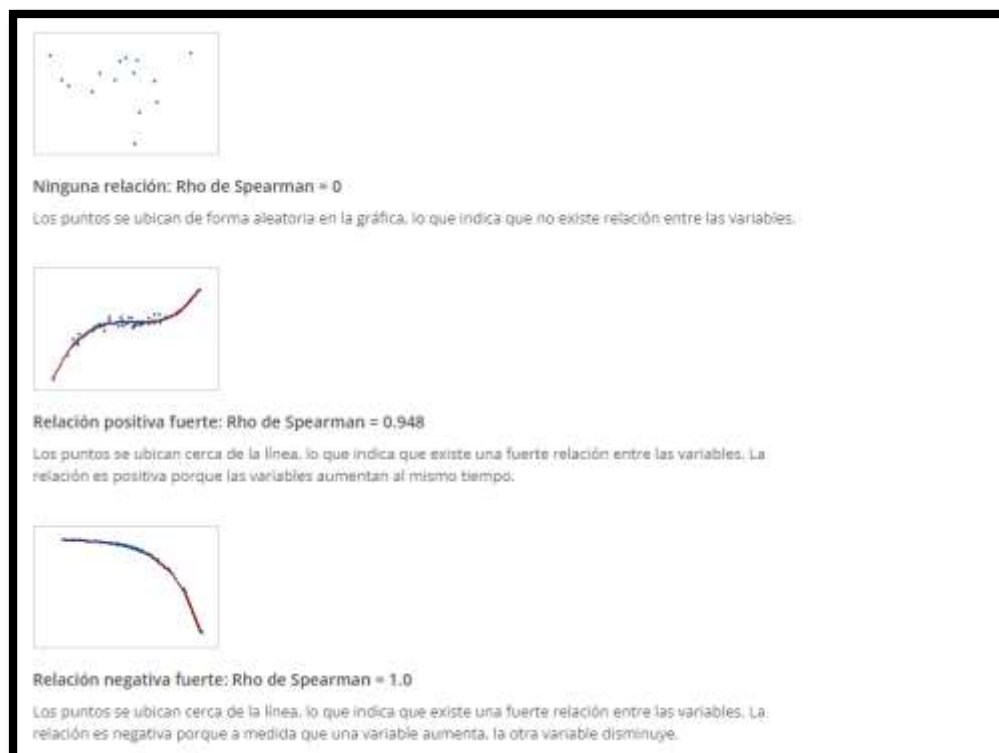
El valor del coeficiente de correlación puede variar de -1 a $+1$. Mientras mayor sea el valor absoluto del coeficiente, más fuerte será la relación entre las variables.

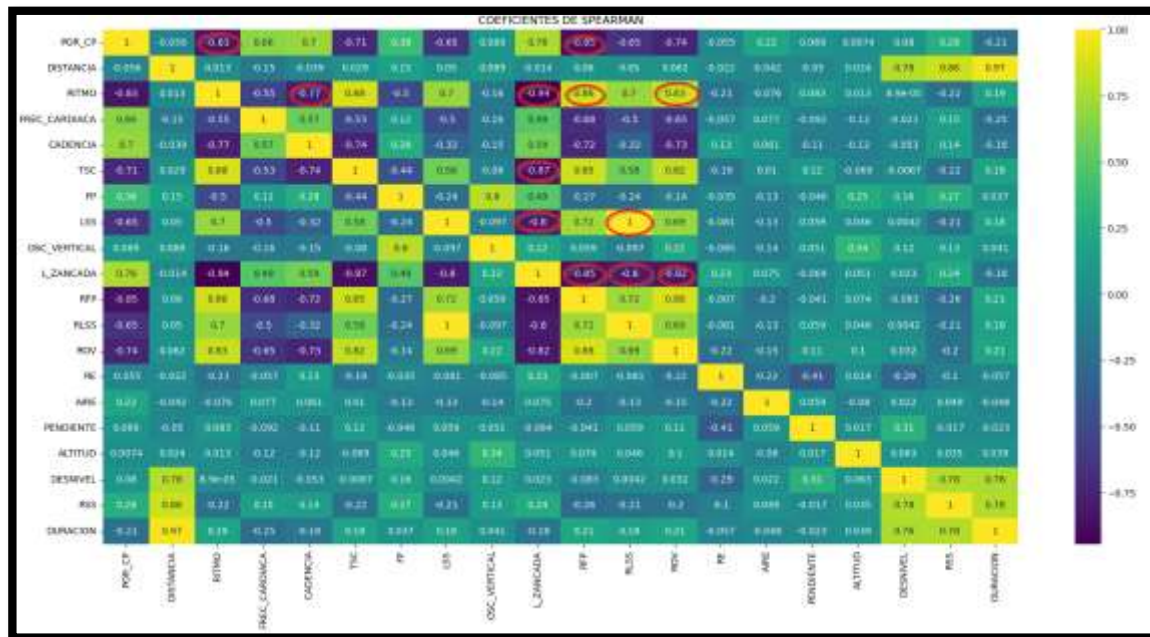
Para la correlación de Spearman, un valor absoluto de 1 indica que los datos ordenados por rango son perfectamente lineales. Por ejemplo, una correlación de Spearman de -1 significa que el valor más alto de la Variable A está asociado con el valor más bajo de la Variable B, el segundo valor más alto de la Variable A está asociado con el segundo valor más bajo de la Variable B y así sucesivamente.

Dirección

El signo del coeficiente indica la dirección de la relación. Si ambas variables tienden a aumentar o disminuir a la vez, el coeficiente es positivo y la línea que representa la correlación forma una pendiente hacia arriba. Si una variable tiende a incrementarse mientras la otra disminuye, el coeficiente es negativo y la línea que representa la correlación forma una pendiente hacia abajo.

Las siguientes gráficas muestran datos con valores específicos del coeficiente de correlación de Spearman para ilustrar diferentes patrones en la fuerza y la dirección de las relaciones entre las variables.





A grandes rasgos y, al igual que ocurre con el coeficiente de Pearson, todo valor mayor a 0.75 en valores absolutos indicarán una correlación grande entre características y serán posibles candidatos para ser eliminados de nuestros datos y no caer en sobreentrenamiento de nuestro futuro algoritmo de regresión.

Sesgo (skew)

Cuando realizamos cualquier tipo de análisis de datos para modelos predictivos deberemos tener en cuenta el sesgo, es de vital importancia que intentemos controlarlo para que nuestro modelo no caiga en predicciones erróneas. Mediante la función skew vamos a calcularlo.

```
***** SKEW O SESGO *****
```

POR_CP	3.489
DISTANCIA	2.528
RITMO	-2.615
FREC_CARDIACA	0.128
CADENCIA	-0.669
TSC	-3.947
FP	-3.108
LSS	-5.287
OSC_VERTICAL	-5.050
L_ZANCADA	4.661
RFP	-2.870
RLSS	-5.287
ROV	-3.457
RE	-2.589
AIRE	1.366
PENDIENTE	1.875
ALTITUD	8.092
DESNIVEL	2.816
RSS	3.014
DURACION	2.460

Gráficas de Distribución y Densidad

Cox-Box-Densidad

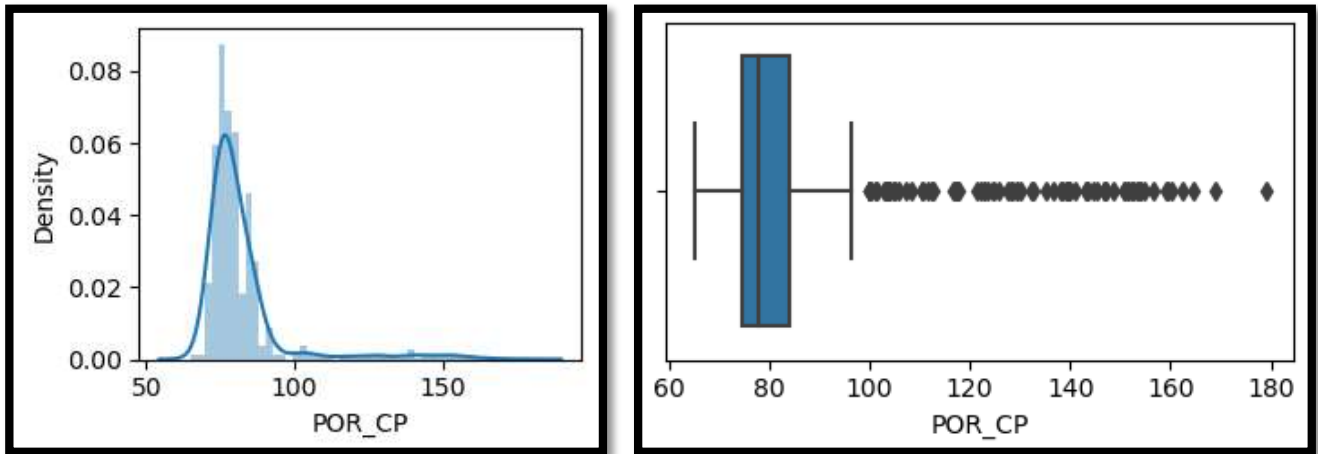
```
# *****
# ***** VISUALIZACIONES *****
# *****
# Visualizamos histograma de distribución y densidad
f, axes = plt.subplots(4,5, figsize=(14,10))
sns.distplot(por_cp, ax= axes [0,0])
sns.distplot(distancia, ax= axes [0,1])
sns.distplot(duracion, ax= axes [0,2])
sns.distplot(frec_cardiaca, ax= axes [0,3])
sns.distplot(cadencia, ax= axes [0,4])
sns.distplot(tsc, ax= axes [1,0])
sns.distplot(fp, ax= axes [1,1])
sns.distplot(lss, ax= axes [1,2])
sns.distplot(osc_vertical, ax= axes [1,3])
sns.distplot(l_zancada, ax= axes [1,4])
sns.distplot(rfp, ax= axes [2,0])
sns.distplot(rlss, ax= axes [2,1])
sns.distplot(rov, ax= axes [2,2])
sns.distplot(re, ax= axes [2,3])
sns.distplot(aire, ax= axes [2,4])
sns.distplot(pendiente, ax= axes [3,0])
sns.distplot(altitud, ax= axes [3,1])
sns.distplot(desnivel, ax= axes [3,2])
sns.distplot(rss, ax= axes [3,3])
sns.distplot(ritmo, ax= axes [3,4])
plt.tight_layout()
plt.show()

# Visualizacion de boxplot con matplotlib
fig = plt.figure(figsize=(14,10))
ax = fig.gca()
data.plot(ax= ax, kind="box", subplots =True, layout =(4,5), sharex=False)
plt.show()

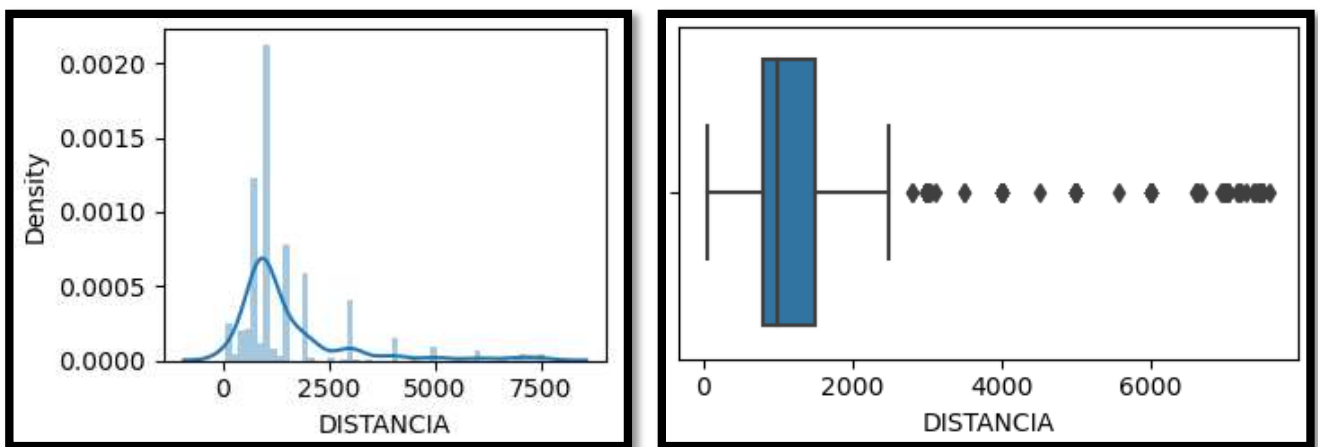
# Visualizacion de boxplot con seaborn
f, axes = plt.subplots(4,5, figsize=(14,10))
sns.boxplot(por_cp, ax= axes [0,0])
sns.boxplot(distancia, ax= axes [0,1])
sns.boxplot(duracion, ax= axes [0,2])
sns.boxplot(frec_cardiaca, ax= axes [0,3])
sns.boxplot(cadencia, ax= axes [0,4])
sns.boxplot(tsc, ax= axes [1,0])
sns.boxplot(fp, ax= axes [1,1])
sns.boxplot(lss, ax= axes [1,2])
sns.boxplot(osc_vertical, ax= axes [1,3])
sns.boxplot(l_zancada, ax= axes [1,4])
sns.boxplot(rfp, ax= axes [2,0])
sns.boxplot(rlss, ax= axes [2,1])
sns.boxplot(rov, ax= axes [2,2])
sns.boxplot(re, ax= axes [2,3])
sns.boxplot(aire, ax= axes [2,4])
sns.boxplot(pendiente, ax= axes [3,0])
sns.boxplot(altitud, ax= axes [3,1])
sns.boxplot(desnivel, ax= axes [3,2])
sns.boxplot(rss, ax= axes [3,3])
sns.boxplot(ritmo, ax= axes [3,4])
plt.tight_layout()
plt.show()
```

Fase de Análisis de Datos

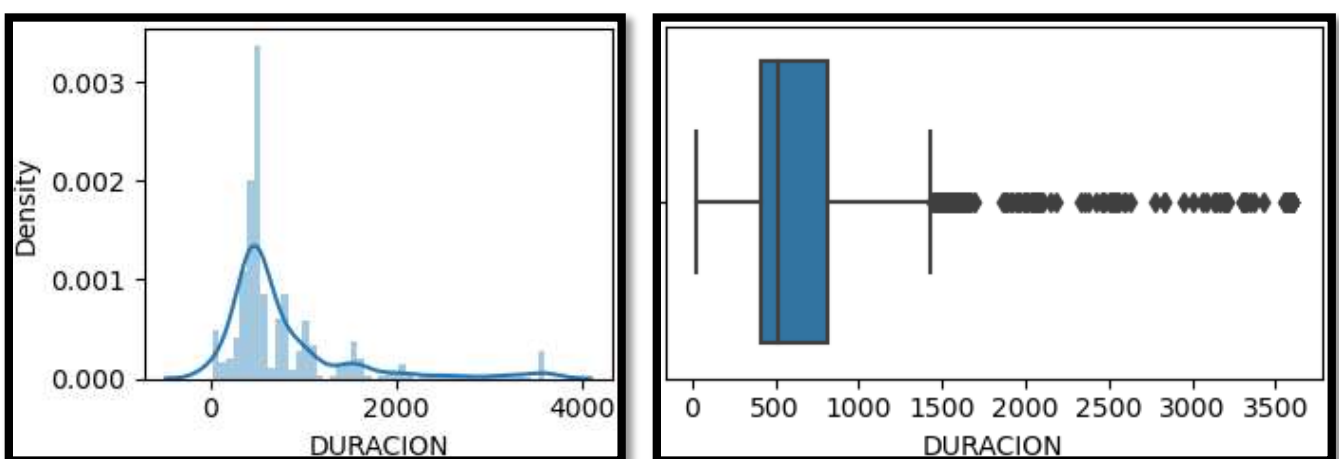
POR_CP: a priori, distribución normal o gaussiana con número significativo de outliers.



DISTANCIA: a priori, distribución normal o gaussiana con número significativo de outliers.

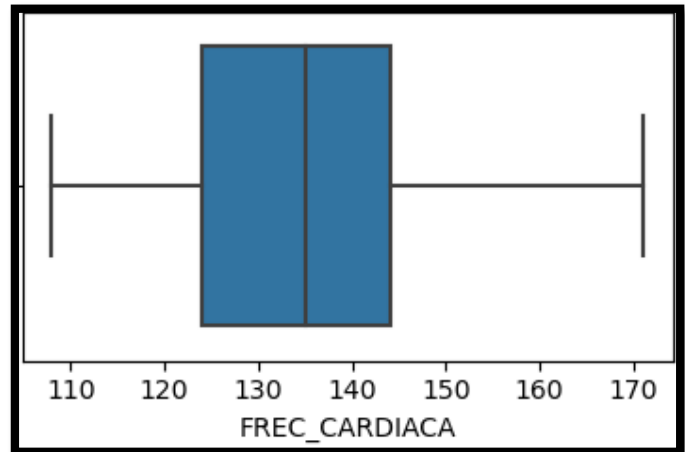
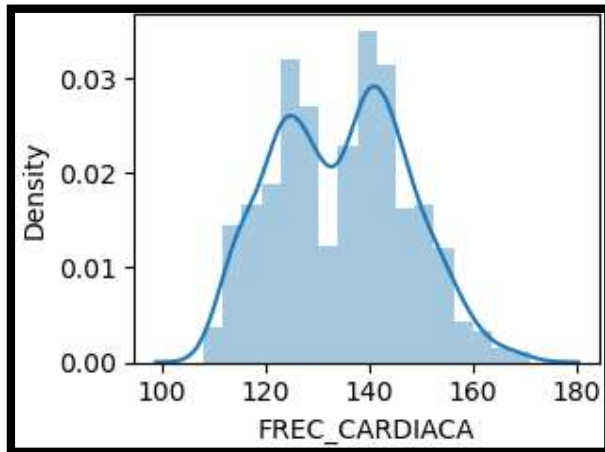


DURACION: a priori, distribución normal o gaussiana con número significativo de outliers.

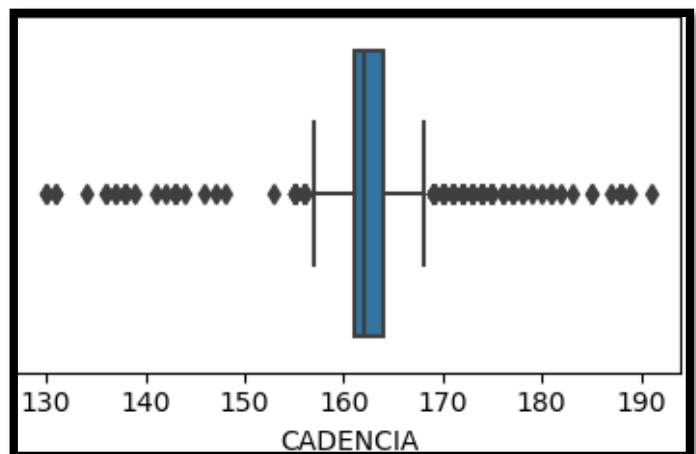
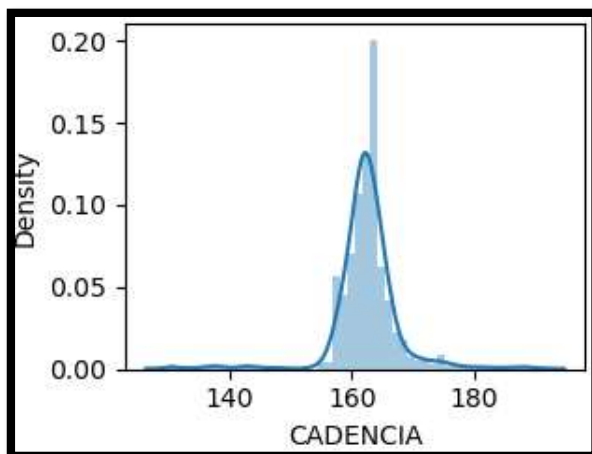


Fase de Análisis de Datos

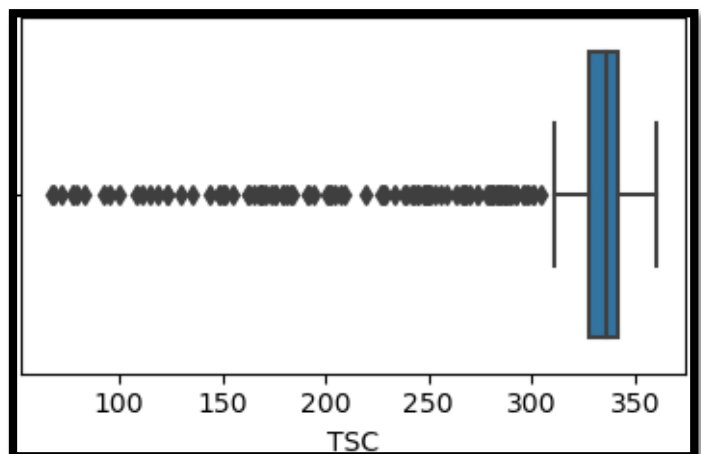
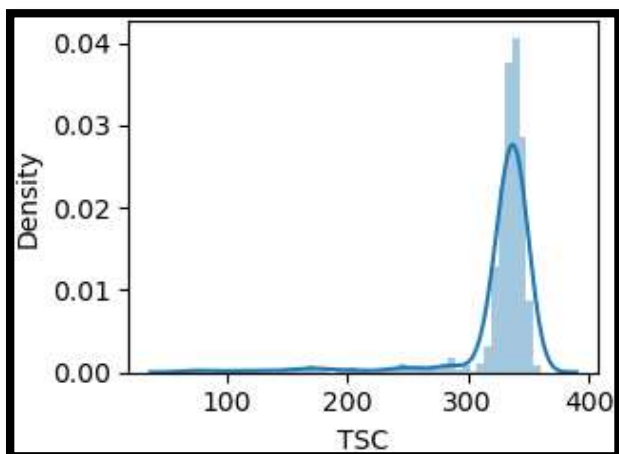
FREC_CARDICA: a priori, distribución doble gaussiana sin outliers.



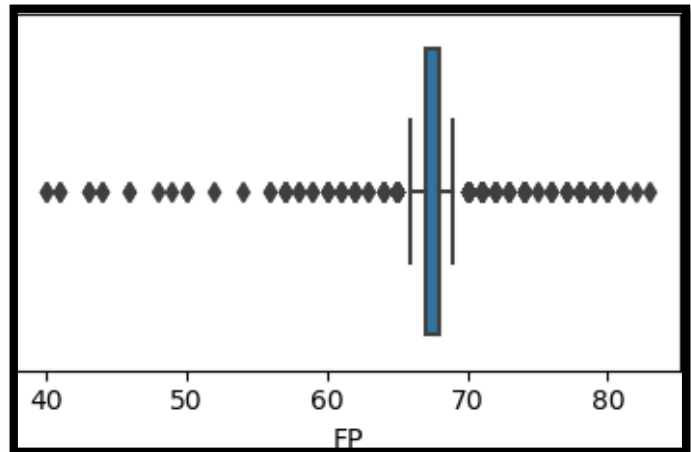
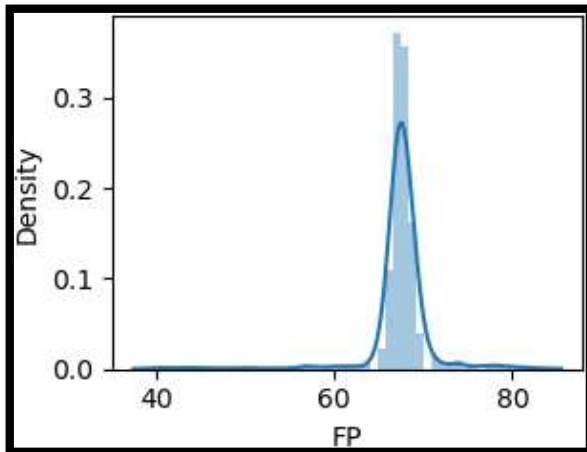
CADENCIA: a priori, distribución normal o gaussiana con número significativo de outliers.



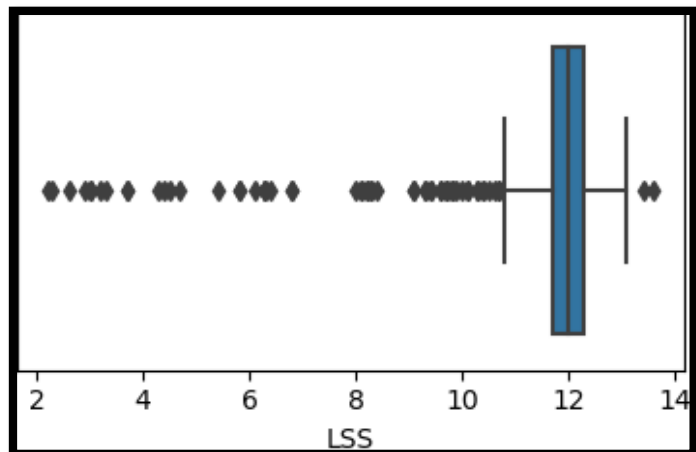
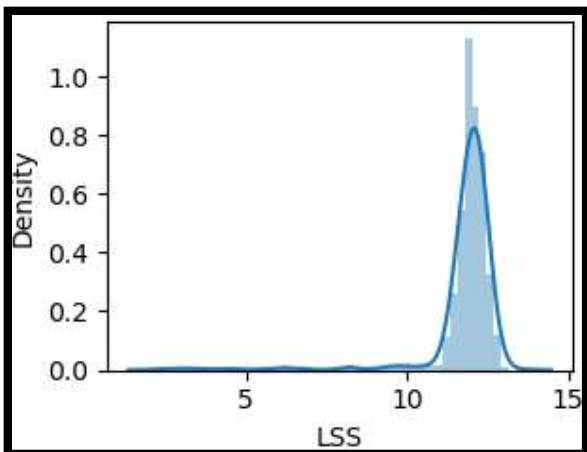
TSC: a priori, distribución normal o gaussiana con número significativo de outliers.



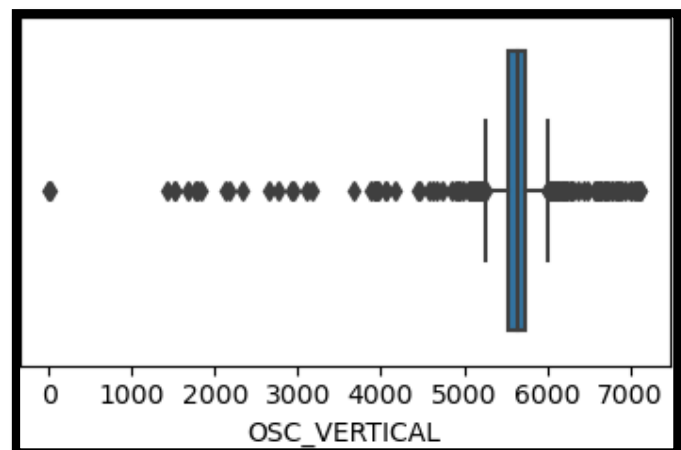
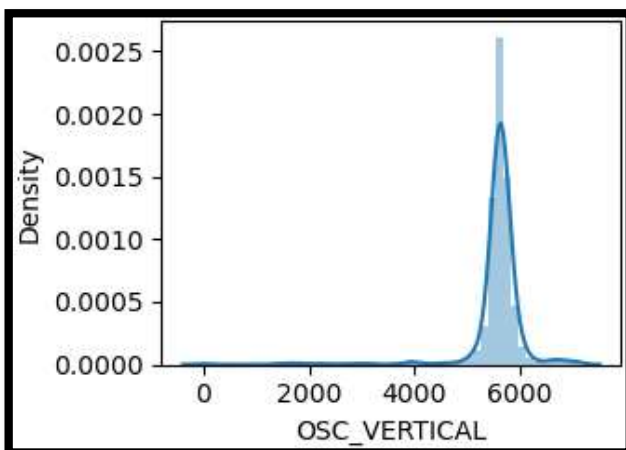
FP: a priori, distribución normal o gaussiana con número significativo de outliers.



LSS: a priori, distribución normal o gaussiana con número significativo de outliers.

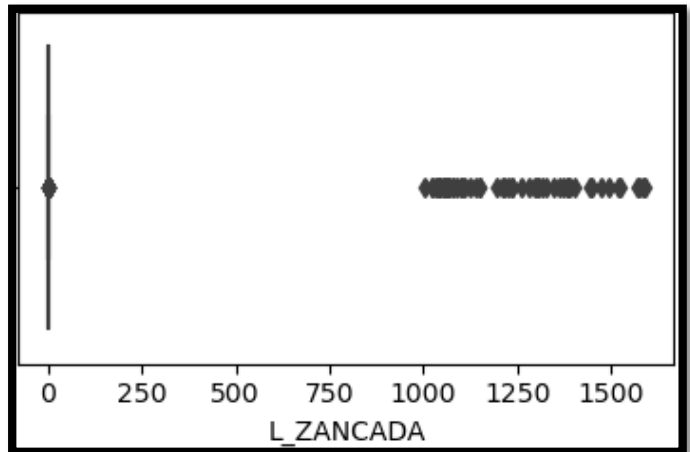
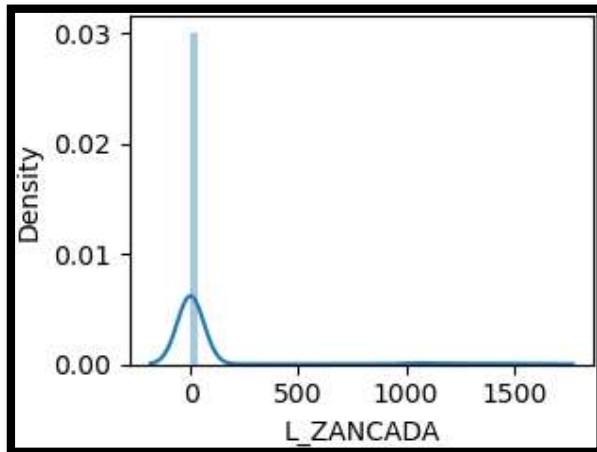


OSC_VERTICAL: a priori, distribución normal o gaussiana con número significativo de outliers.

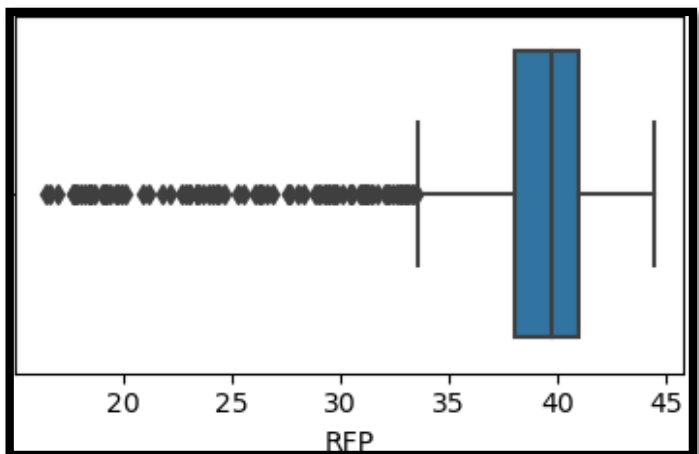
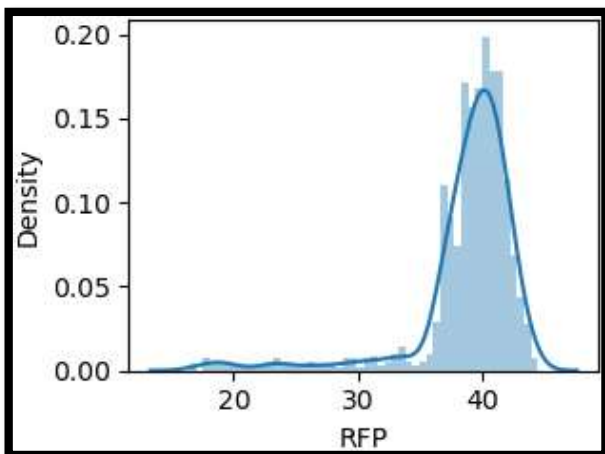


Fase de Análisis de Datos

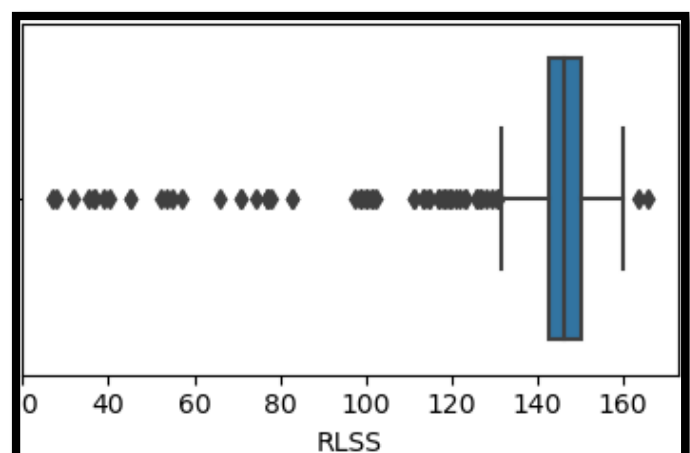
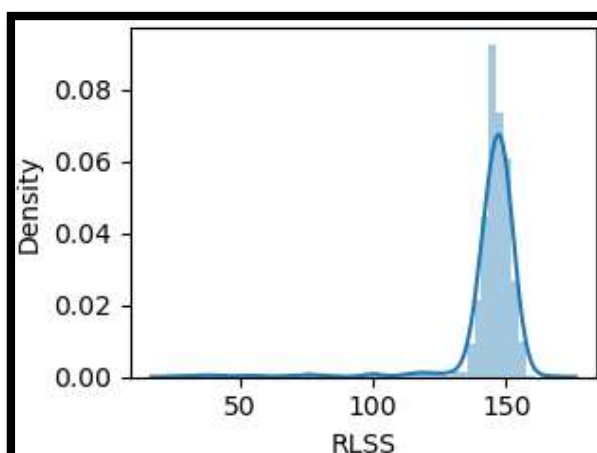
L_ZANCADA: a priori, distribución normal o gaussiana con número significativo de outliers.



RFP: a priori, distribución normal o gaussiana con número significativo de outliers.

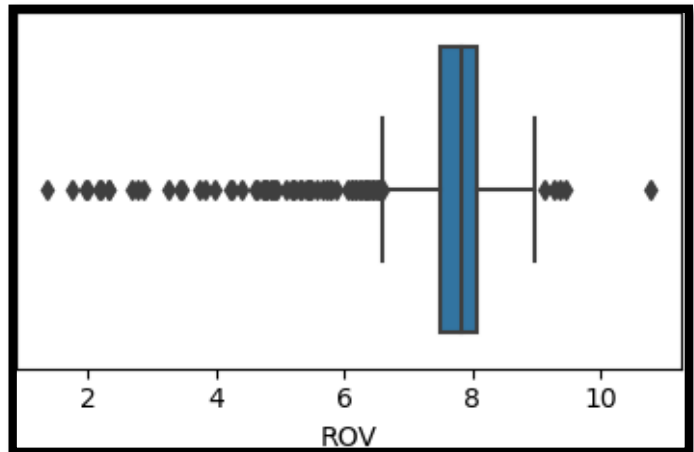
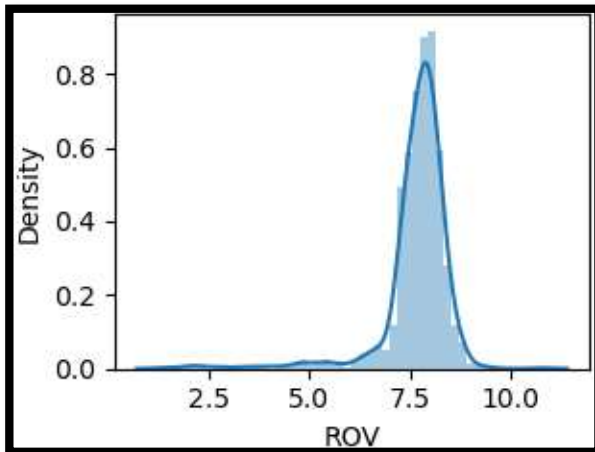


RLSS: a priori, distribución normal o gaussiana con número significativo de outliers.

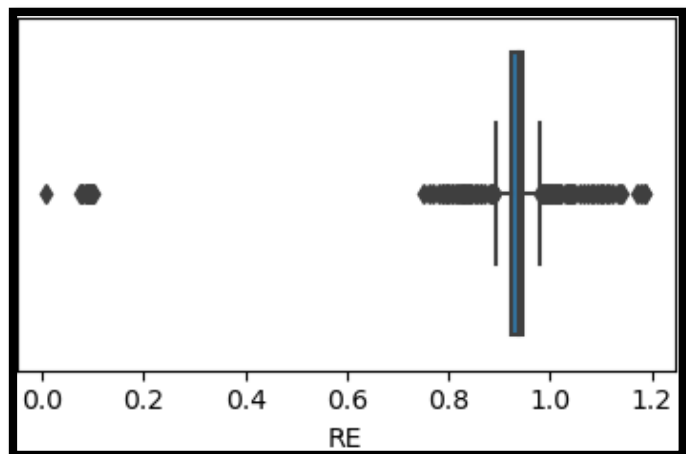
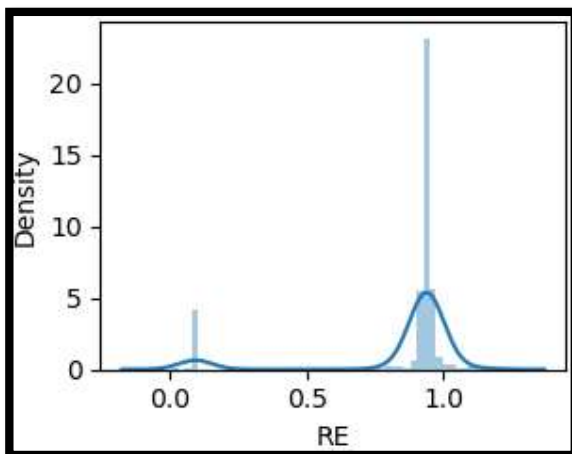


Fase de Análisis de Datos

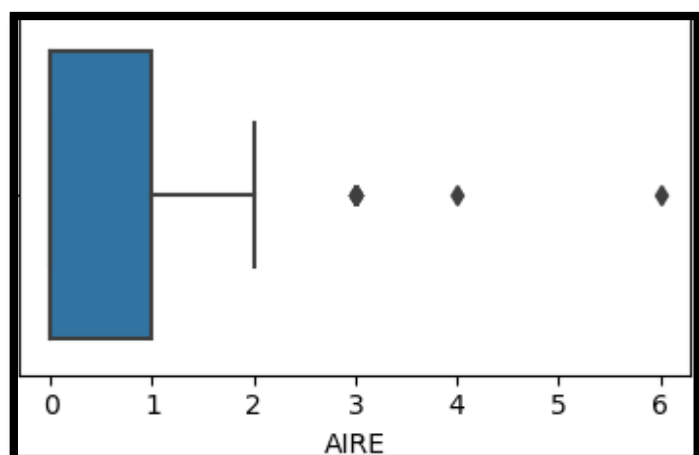
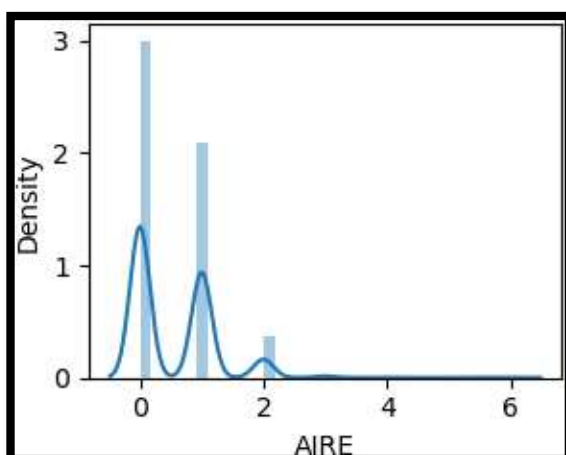
ROV: a priori, distribución normal o gaussiana con número significativo de outliers.



RE: a priori, distribución doble gaussiana con número significativo de outliers.

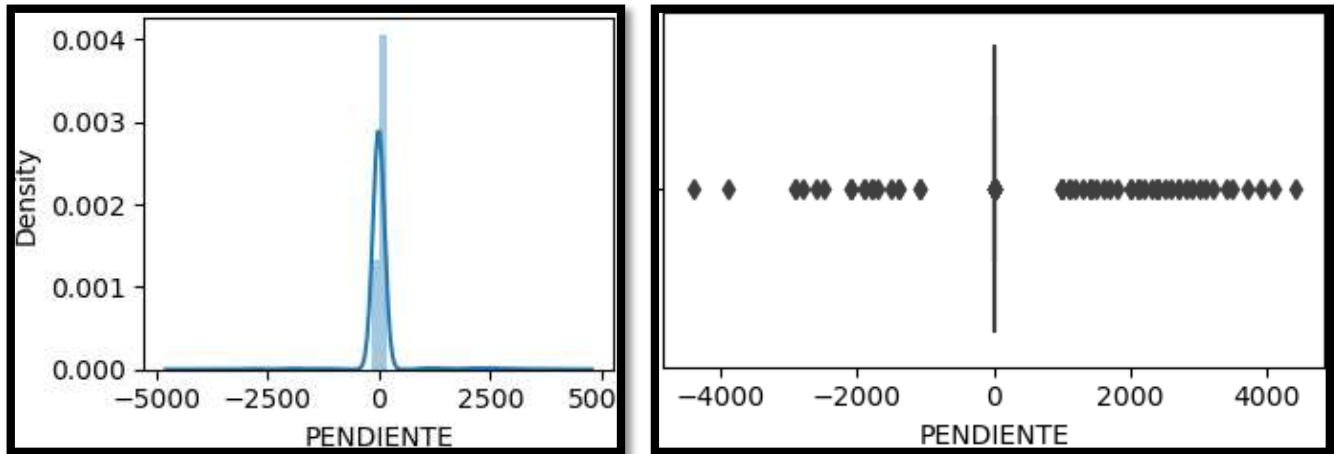


AIRE: a priori, distribución triple gaussiana con número muy limitado de outliers.

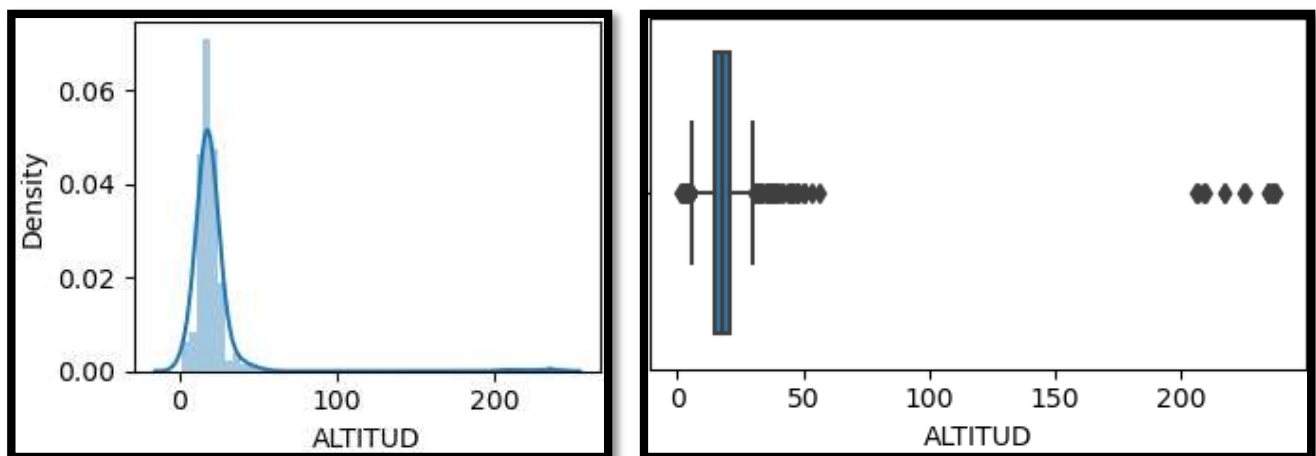


Fase de Análisis de Datos

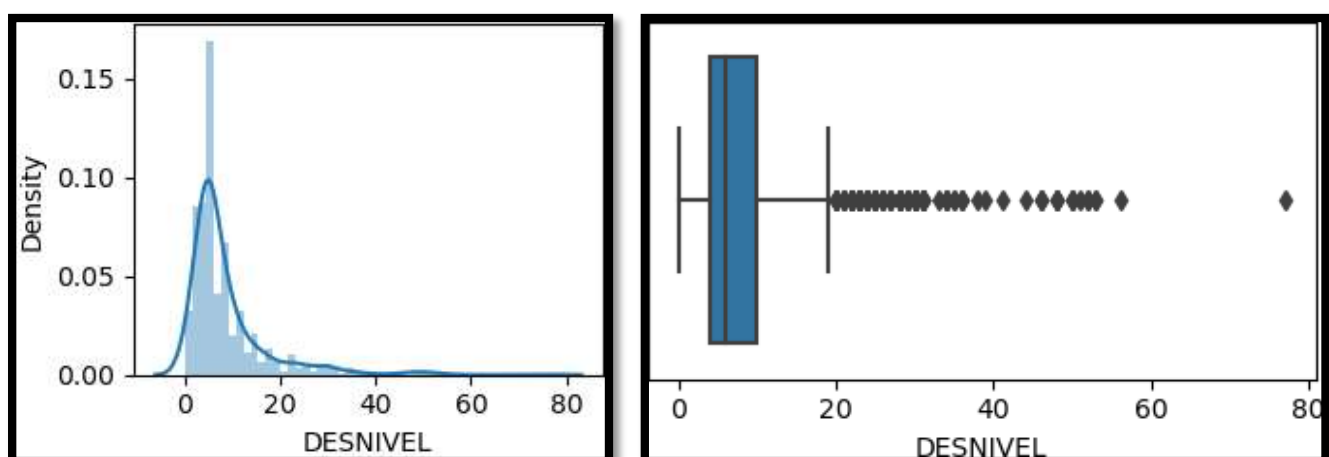
PENDIENTE: a priori, distribución normal o gaussiana con número significativo de outliers.



ALTITUD: a priori, distribución normal o gaussiana con número significativo de outliers.

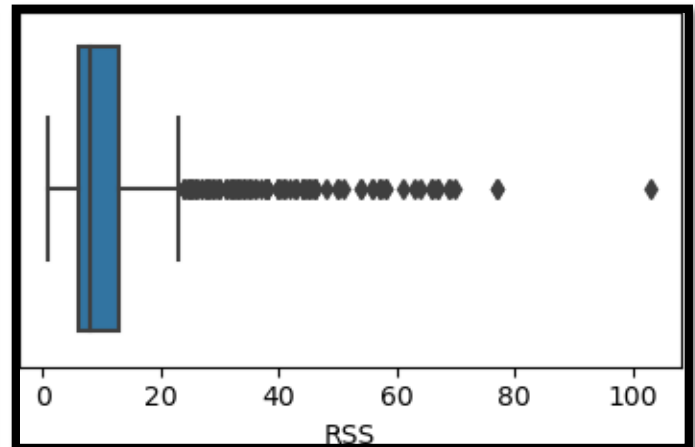
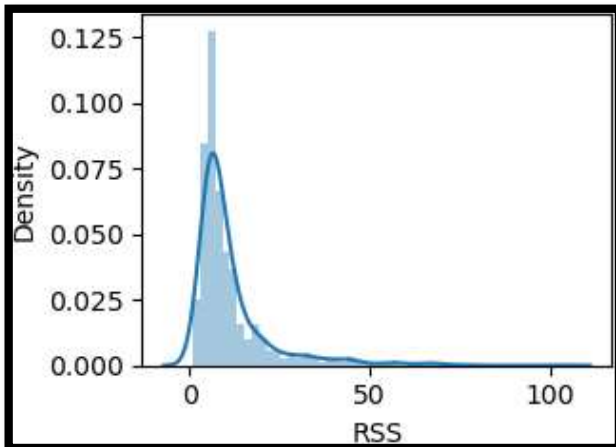


DESNIVEL: a priori, distribución normal o gaussiana con número significativo de outliers.



Fase de Análisis de Datos

RLSS: a priori, distribución normal o gaussiana con número significativo de outliers.



RITMO: a priori, distribución normal o gaussiana con número significativo de outliers.

