Haz doble clic (o pulsa Intro) para editar

Importar llibrerias:

```
#@title Importar llibrerias:

import numpy as np
import matplotlib as plt
import pandas as pd
import csv
import seaborn as sns
import datetime
from datetime import timedelta

```

obrir fitxer

```
#@title obrir fitxer
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
path= '/content/drive/MyDrive/Ficheros de Vueling/2022.06.03 2022_delay + cierre puertas.x
```

```
# Abrir fichero de Github.
Hoja = 'FLT_2022'

df = pd.read_excel(path, sheet_name=Hoja)

```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 60492 entries, 0 to 60491
Data columns (total 41 columns):
 #   Column                                                  Non-Null Count  Dtype
---  ------                                                  --------------  -----
 0   DATE                                                    56133 non-null  datetime64[ns]
 1   FLT                                                     56133 non-null  object
 2   REG                                                     56133 non-null  object
 3   AC                                                      56133 non-null  object
 4   DEP                                                     56133 non-null  object
 5   ARR                                                     56133 non-null  object
 6   STD                                                     56133 non-null  object
 7   STA                                                     56133 non-null  object
 8   TKof                                                    56133 non-null  object
 9   TDwn                                                    56133 non-null  object
 10  ATD                                                     56133 non-null  object
 11  ATA                                                     56133 non-null  object
 12  BLOCK                                                   56133 non-null  object
 13  FLThr                                                   56133 non-null  object
 14  DStand                                                  56133 non-null  object
 15  AStand                                                  56133 non-null  object
 16  ACT PAX                                                 56133 non-null  float64
 17  LF                                                      56133 non-null  float64
 18  Taxi-out                                                56133 non-null  object
 19  Taxi-In                                                 56133 non-null  object
 20  SLOT                                                    56133 non-null  object
 21  C1                                                      56133 non-null  object
 22  DLY1                                                    56133 non-null  object
 23  Sub1                                                    56133 non-null  object
```

```
 24  C2                                                56133 non-null  object
 25  DLY2                                              56133 non-null  object
 26  Sub2                                              56133 non-null  object
 27  C3                                                56133 non-null  object
 28  DLY3                                              56133 non-null  object
 29  Sub3                                              56133 non-null  object
 30  C4                                                56133 non-null  object
 31  DLY4                                              56133 non-null  object
 32  Sub4                                              56133 non-null  object
 33  C1Arr                                             56133 non-null  object
 34  DLY1Arr                                           14093 non-null  object
 35  Close Pax Door                                    59855 non-null  object
 36  Close Cargo Door                                  59855 non-null  object
 37  Open Cargo/Pax Door                               49995 non-null  object
 38  close pax door2                                   59855 non-null  object
 39  close cargo door2                                 59855 non-null  object
 40  open cargo/pax door2                              49995 non-null  object
dtypes: datetime64[ns](1), float64(2), object(38)
memory usage: 18.9+ MB
```

Haz doble clic (o pulsa Intro) para editar

```
# Els noms de les columnes tenen molts espais en blanc

nombreColumnas= df.columns
nombreColumnas
```

```
Index(['DATE                            ', 'FLT    ', 'REG   ', 'AC ', 'DEP',
       'ARR', 'STD  ', 'STA  ', 'TKof ',
       'TDwn                                                ', 'ATD  ',
       'ATA  ', 'BLOCK', 'FLThr', 'DStand', 'AStand', 'ACT PAX  ', 'LF ',
       'Taxi-out', 'Taxi-In', 'SLOT       ', 'C1', 'DLY1 ', 'Sub1', 'C2',
       'DLY2 ', 'Sub2', 'C3', 'DLY3 ', 'Sub3', 'C4', 'DLY4 ', 'Sub4', 'C1Arr',
       'DLY1Arr', 'Close Pax Door', 'Close Cargo Door', 'Open Cargo/Pax Door',
       'close pax door2', 'close cargo door2', 'open cargo/pax door2'],
      dtype='object')
```

```
# # Aqui corregeixo els noms de les columnes
nombreColumnaCorregido= ['DATE', 'FLT', 'REG', 'AC', 'DEP',
       'ARR', 'STD', 'STA', 'TKof', 'TDwn', 'ATD',
       'ATA', 'BLOCK', 'FLThr', 'DStand', 'AStand', 'ACT PAX', 'LF',
       'Taxi-out', 'Taxi-In', 'SLOT', 'C1', 'DLY1', 'Sub1', 'C2',
       'DLY2', 'Sub2', 'C3', 'DLY3', 'Sub3', 'C4', 'DLY4', 'Sub4', 'C1Arr',
       'DLY1Arr', 'Close Pax Door', 'Close Cargo Door', 'Open Cargo/Pax Door',
       'close pax door2', 'close cargo door2', 'open cargo/pax door2']
```

Procés per canviar el nom de les columnes amb un bucle FOR

```
#@title Procés per canviar el nom de les columnes amb un bucle FOR
for n, m in enumerate(nombreColumnas):
  print(n, m,'*',nombreColumnaCorregido[n],'-')
  df.rename({m: nombreColumnaCorregido[n]}, axis=1, inplace=True)
```

```
0 DATE                            * DATE -
1 FLT     * FLT -
2 REG    * REG -
3 AC  * AC -
4 DEP * DEP -
5 ARR * ARR -
6 STD   * STD -
7 STA   * STA -
8 TKof  * TKof -
9 TDwn                                                * TDwn -
10 ATD   * ATD -
11 ATA   * ATA -
12 BLOCK * BLOCK -
```

```
13 FLThr * FLThr -
14 DStand * DStand -
15 AStand * AStand -
16 ACT PAX   * ACT PAX -
17 LF  * LF -
18 Taxi-out * Taxi-out -
19 Taxi-In * Taxi-In -
20 SLOT        * SLOT -
21 C1 * C1 -
22 DLY1  * DLY1 -
23 Sub1 * Sub1 -
24 C2 * C2 -
25 DLY2  * DLY2 -
26 Sub2 * Sub2 -
27 C3 * C3 -
28 DLY3  * DLY3 -
29 Sub3 * Sub3 -
30 C4 * C4 -
31 DLY4  * DLY4 -
32 Sub4 * Sub4 -
33 C1Arr * C1Arr -
34 DLY1Arr * DLY1Arr -
35 Close Pax Door * Close Pax Door -
36 Close Cargo Door * Close Cargo Door -
37 Open Cargo/Pax Door * Open Cargo/Pax Door -
38 close pax door2 * close pax door2 -
39 close cargo door2 * close cargo door2 -
40 open cargo/pax door2 * open cargo/pax door2 -
```

Creo la seqüencia dels vols.

```
1 #@title Creo la seqüencia dels vols.
2
3 df['Secuencia'] = df.groupby(['DATE', 'REG'])['STD'].rank()
4 print(df[['Secuencia','STD', 'DEP']][0:4])
```

```
   Secuencia       STD  DEP
0        1.0  07:40:00  BIO
1        2.0  09:40:00  SVQ
2        3.0  11:40:00  BIO
3        4.0  13:30:00  SCQ
```

Eliminar files amb valors Nan del df.

```
1 #@title Eliminar files amb valors Nan del df.
2
3 #df = df.dropna()
4 df.head()
```

```
      DATE   FLT  REG   AC  DEP  ARR      STD      STA     TKof     TDwn  ...  Sub4  C1Arr  DLY1Arr  Close I
```

```
#@ title Subrutina per convertir totes les columnes de temps en un format operatiu.

formato = "%H:%M:%S"

def convertirTiempo(clave):
    df[clave] = pd.to_datetime(df[clave],
                               format=formato,
                               errors='coerce')

columnasConvertir = ['DATE', 'STD', 'STA', 'TKof', 'TDwn', 'ATD', 'ATA', 'BLOCK', 'FLThr']


for x in columnasConvertir:
    convertirTiempo(x)

```

Subrutina que suma les dates amb les hores.

```
#@title Subrutina que suma les dates amb les hores.

def adecuarFechas(fecha, tiempoClave):
    clave = "Date_" + tiempoClave
    print('.....Clave: ',clave, '....', fecha, tiempoClave)
    df[tiempoClave] = pd.to_datetime(df[tiempoClave], format='%H:%M', errors='coerce')

    df[tiempoClave] = pd.to_datetime( df[fecha].dt.strftime('%d/%m/%Y') + ' ' + df[tiempoC
```

Creo el df1 per treballar amb les columnes de data.

```
#@title Creo el df1 per treballar amb les columnes de data.
df1=df.copy()
#print(df['STD'].head())
columnasAddDate= ['STD', 'STA', 'TKof', 'TDwn', 'ATD', 'ATA', 'BLOCK', 'FLThr',
                  'close pax door2', 'close cargo door2']

for x in columnasAddDate:
    adecuarFechas('DATE', x)

```

```
.....Clave:  Date_STD .... DATE STD
.....Clave:  Date_STA .... DATE STA
.....Clave:  Date_TKof .... DATE TKof
.....Clave:  Date_TDwn .... DATE TDwn
.....Clave:  Date_ATD .... DATE ATD
.....Clave:  Date_ATA .... DATE ATA
.....Clave:  Date_BLOCK .... DATE BLOCK
.....Clave:  Date_FLThr .... DATE FLThr
.....Clave:  Date_close pax door2 .... DATE close pax door2
.....Clave:  Date_close cargo door2 .... DATE close cargo door2
```

Afegeixo el mes i setmana

Mostrar código

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: FutureWarning: Series.dt.weekofyear and Ser
  after removing the cwd from sys.path.
```

Identifico los Aeropuertos principales.

Mostrar código

```
1 AeropuertosCantidad = df['DEP']
2 AeropuertosCantidad['cantitat'] = df['DEP'].value_counts()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
```

Sapigue quants vols han sortit de cada aeroport

Mostrar código

```
(6,)
```

```
1 AeropuertosCantidad.head()
```

```
12064    BCN
26560    ORK
38307    AMS
52105    ARN
35518    AGP
Name: DEP, dtype: object
```

identifico els principals aeroports

Mostrar código

	DATE	FLT	REG	AC	DEP	ARR	STD	STA	TKof	TDwn	...	close pax door2	close cargo door2	open cargo/pax door2	S
0	2022-01-01	2506	EC-JSY	320	BIO	SVQ	2022-01-01 07:40:00	2022-01-01 09:05:00	2022-01-01 08:10:00	2022-01-01 09:17:00	...	2022-01-01 07:58:00	2022-01-01 07:31:00	09:23	
1	2022-01-01	2505	EC-JSY	320	SVQ	BIO	2022-01-01 09:40:00	2022-01-01 11:05:00	2022-01-01 10:03:00	2022-01-01 11:07:00	...	2022-01-01 09:48:00	2022-01-01 09:46:00	11:11	
2	2022-01-01	1386	EC-JSY	320	BIO	SCQ	2022-01-01 11:40:00	2022-01-01 12:50:00	2022-01-01 11:50:00	2022-01-01 12:40:00	...	2022-01-01 11:38:00	2022-01-01 11:34:00	12:46	
3	2022-01-01	1387	EC-JSY	320	SCQ	BIO	2022-01-01 13:30:00	2022-01-01 14:35:00	2022-01-01 13:30:00	2022-01-01 14:08:00	...	2022-01-01 13:18:00	2022-01-01 13:06:00	14:12	

4 rows × 48 columns

Estudi temps Close Pax i CLose Cargo door

```
1 #@title Estudi temps Close Pax i CLose Cargo door
2
3 df['Close Pax Door'] =pd.to_datetime(df['Close Pax Door'])
4 print(df['Close Pax Door'].head(4))
```

```
print('-----------------')
print(df1['ATD'].head(4))
```

```
0   2022-01-01 07:58:00
1   2022-01-01 09:48:00
2   2022-01-01 11:38:00
3   2022-01-01 13:18:00
Name: Close Pax Door, dtype: datetime64[ns]
-----------------
0   1900-01-01 08:00:00
1   1900-01-01 09:51:00
2   1900-01-01 11:40:00
3   1900-01-01 13:23:00
Name: ATD, dtype: datetime64[ns]
```

```
df['Close Pax Door'] =pd.to_datetime(df['Close Pax Door'])
df['t_ClosePax'] =   (df['Close Pax Door'] - df['ATD']) /  np.timedelta64(1, 'm')
df['t_ClosePax']
```

```
0        -2.0
1        -3.0
2        -2.0
3        -5.0
4         0.0
        ...
56128    417.0
56129    474.0
56130    479.0
56131   -440.0
56132   -406.0
Name: t_ClosePax, Length: 56133, dtype: float64
```

Càlcul del temps del boarding

```
#@title Càlcul del temps del boarding
df ['Close Cargo Door']=pd.to_datetime(df ['Close Cargo Door'])
df['t_Close_Cargo_Door'] = (df ['Close Cargo Door'] - df['ATD']) /  np.timedelta64(1, 'm')

df['Close Pax Door'] =pd.to_datetime(df['Close Pax Door'])
df ['Close Cargo Door']=pd.to_datetime(df ['Close Cargo Door'])

df['Open Cargo/Pax Door'] =pd.to_datetime(df['Open Cargo/Pax Door'])

df['t_Entre_Puertas'] = (df ['Close Cargo Door'] - df['Close Pax Door']) /  np.timedelta64
df['Retardo_Abrir_Puerta_Pax']= df['Open Cargo/Pax Door']-df['ATA']
df[['Retardo_Abrir_Puerta_Pax', 'Open Cargo/Pax Door','ATA']]
```

	Retardo_Abrir_Puerta_Pax	Open Cargo/Pax Door	ATA
0	0 days 00:03:00	2022-01-01 09:23:00	2022-01-01 09:20:00

Estudio els trajectes.

```
#@title Estudio els trajectes.
condicion = [(df['DEP'] < df['ARR']),  (df['ARR'] < df['DEP'])]

valores = [  (df['DEP'] + '-' + df['ARR']),  (df['ARR'] + '-' + df['DEP'])]

df['Trayecto'] = np.select(condicion, valores )
df['Trayecto'].head(4)

#df5 = df.where('Departure' > 'Arrival',  'Departure' + 'Arrival',  'Arrival' +'Departure'
```

```
0    BIO-SVQ
1    BIO-SVQ
2    BIO-SCQ
3    BIO-SCQ
Name: Trayecto, dtype: object
```

```
# Crec la seqüència de vols de cada avió cada dia

df['Secuencia'] = df.groupby(['DATE', 'REG'])['STD'].rank()

df1=df
df1.head(2)
```

	DATE	FLT	REG	AC	DEP	ARR	STD	STA	TKof	TDwn	...	Setmana	DiaSetmana	DiaSetman
0	2022-01-01	2506	EC-JSY	320	BIO	SVQ	2022-01-01 07:40:00	2022-01-01 09:05:00	2022-01-01 08:10:00	2022-01-01 09:17:00	...	52.0	5.0	Sat
1	2022-01-01	2505	EC-JSY	320	SVQ	BIO	2022-01-01 09:40:00	2022-01-01 11:05:00	2022-01-01 10:03:00	2022-01-01 11:07:00	...	52.0	5.0	Sat

2 rows × 53 columns

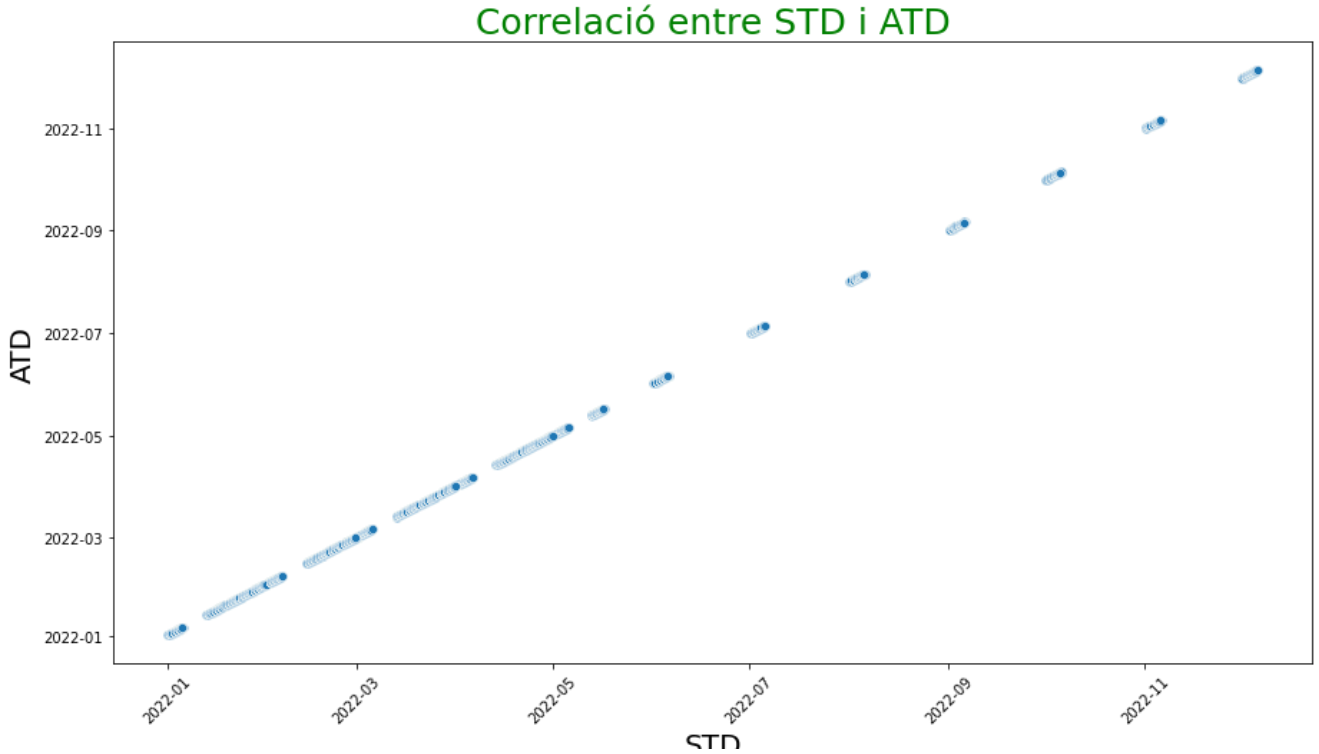
Subrutina: graficà el retard en l'enlairament:

```
#@title Subrutina: graficà el retard en l'enlairament:
def dibujarScatter( ejeX, ejeY):
    x= df1[ejeX]
    y = df1[ejeY]
    ax = sns.scatterplot(x , y )


```

: Puntualitat mitjana primer enlairament el 2022

Mostrar código

¿



Calculo error del primer despegue.

```
# Filtro primer despegue

vuelosPrimerDespegue= (df1['Secuencia']== 1)
df2 = df1[vuelosPrimerDespegue]
print(df2[['DATE','REG', 'Secuencia','STD']][0:8])

```

```
          DATE     REG  Secuencia                STD
0   2022-01-01  EC-JSY        1.0 2022-01-01 07:40:00
6   2022-01-01  EC-JZI        1.0 2022-01-01 07:30:00
12  2022-01-01  EC-KDG        1.0 2022-01-01 14:20:00
16  2022-01-01  EC-KDH        1.0 2022-01-01 07:50:00
21  2022-01-01  EC-KDX        1.0 2022-01-01 17:35:00
24  2022-01-01  EC-KLB        1.0 2022-01-01 05:25:00
29  2022-01-01  EC-KLT        1.0 2022-01-01 07:30:00
33  2022-01-01  EC-KMI        1.0 2022-01-01 12:05:00
```

```
#Calculo error primer despegue despegue:

df1['E_Despegue'] = (df1['ATD']- df1['STD']) /  np.timedelta64(1, 'm')


df1['E_Despegue'].head()


#print(df1['E_Despegue'][0], '-----', df1['Date_ATD'][0], '........', df1['Date_STD'][0])
#print(df1[['DATE','REG', 'E_Despegue', 'Date_ATD', 'Date_STD']][0:8])

print(df1[['DATE','REG', 'Secuencia','STD', 'ATD','E_Despegue']][0:8])

print('\nMedia = ', df1['E_Despegue'].mean())
```

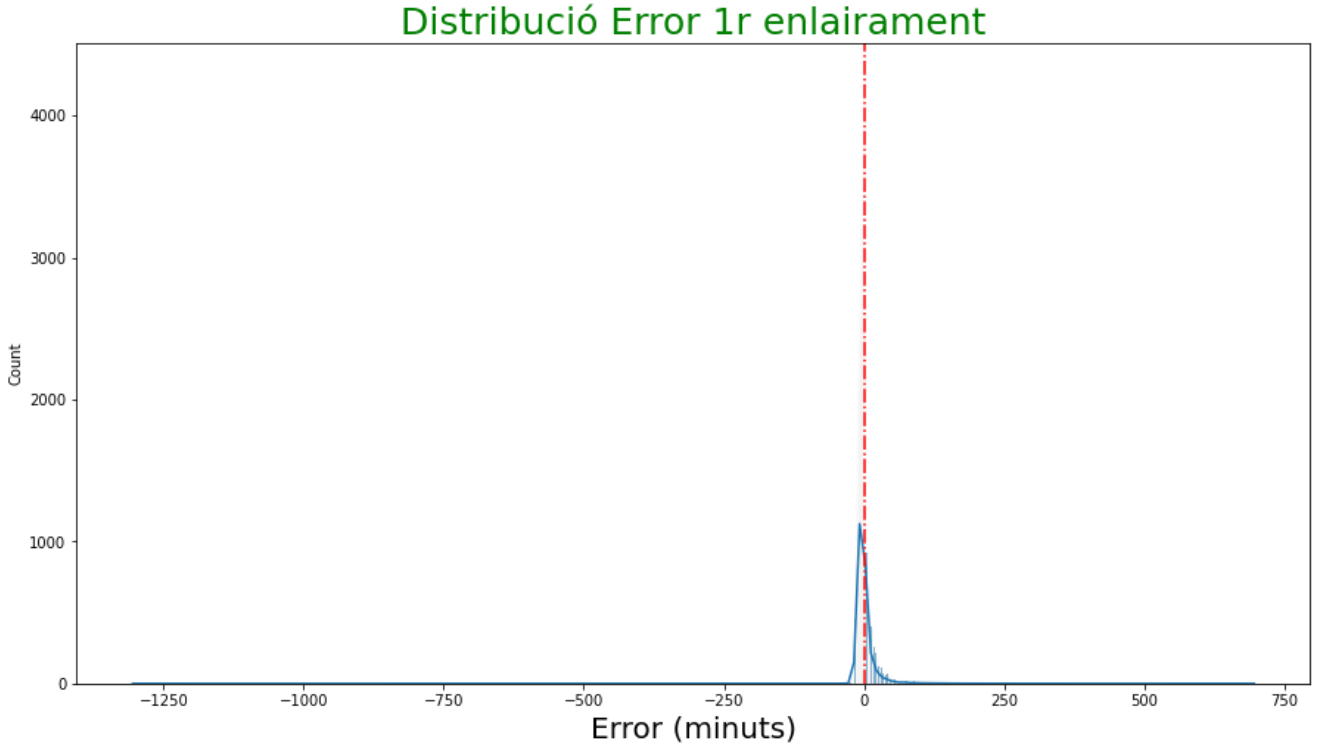
```
          DATE     REG  Secuencia                STD                ATD  \
0   2022-01-01  EC-JSY        1.0 2022-01-01 07:40:00 2022-01-01 08:00:00
1   2022-01-01  EC-JSY        2.0 2022-01-01 09:40:00 2022-01-01 09:51:00
```

```
2 2022-01-01  EC-JSY        3.0 2022-01-01 11:40:00 2022-01-01 11:40:00
3 2022-01-01  EC-JSY        4.0 2022-01-01 13:30:00 2022-01-01 13:23:00
4 2022-01-01  EC-JSY        5.0 2022-01-01 15:15:00 2022-01-01 15:07:00
5 2022-01-01  EC-JSY        6.0 2022-01-01 18:55:00 2022-01-01 18:51:00
6 2022-01-01  EC-JZI        1.0 2022-01-01 07:30:00 2022-01-01 07:12:00
7 2022-01-01  EC-JZI        2.0 2022-01-01 09:40:00 2022-01-01 09:23:00
```

```
   E_Despegue
0        20.0
1        11.0
2         0.0
3        -7.0
4        -8.0
5        -4.0
6       -18.0
7       -17.0

Media =  1.040119717808024
```

```
1 plt.figure(figsize = (15,8))
2 p= sns.histplot(data= df1.E_Despegue, kde= True)
3 p.axvline(x = 0, ymin=0, ymax= 12, color='red', linestyle= 'dashdot') # Objetivo
4 p.set_title('Distribució Error 1r enlairament', fontsize=25, color='green')
5 p.set_xlabel("Error (minuts)", fontsize = 20)
```

```
Text(0.5, 0, 'Error (minuts)')
```



Del gràfic veiem que clarament no és normal i que té moltes dades que no són normals. És una corba, clarament no gaussiana, descentrada capa la dreta.

¿Per què tenim aquest outliers?.

Les raons són situacions extraordinàries, totalment aleatòries, i que requereixen una anàlisi per si mateix, però que amb les dades que tinc no em permet estudiar. Per exemple pot ser motiu d'una avaria d'un motor, una vaga, un passatger que ha perdut el vol i que no es documenta correctament el motiu del retard.

Acció Eliminarem totes les dades que són o més grans o més petites de 30 minuts.

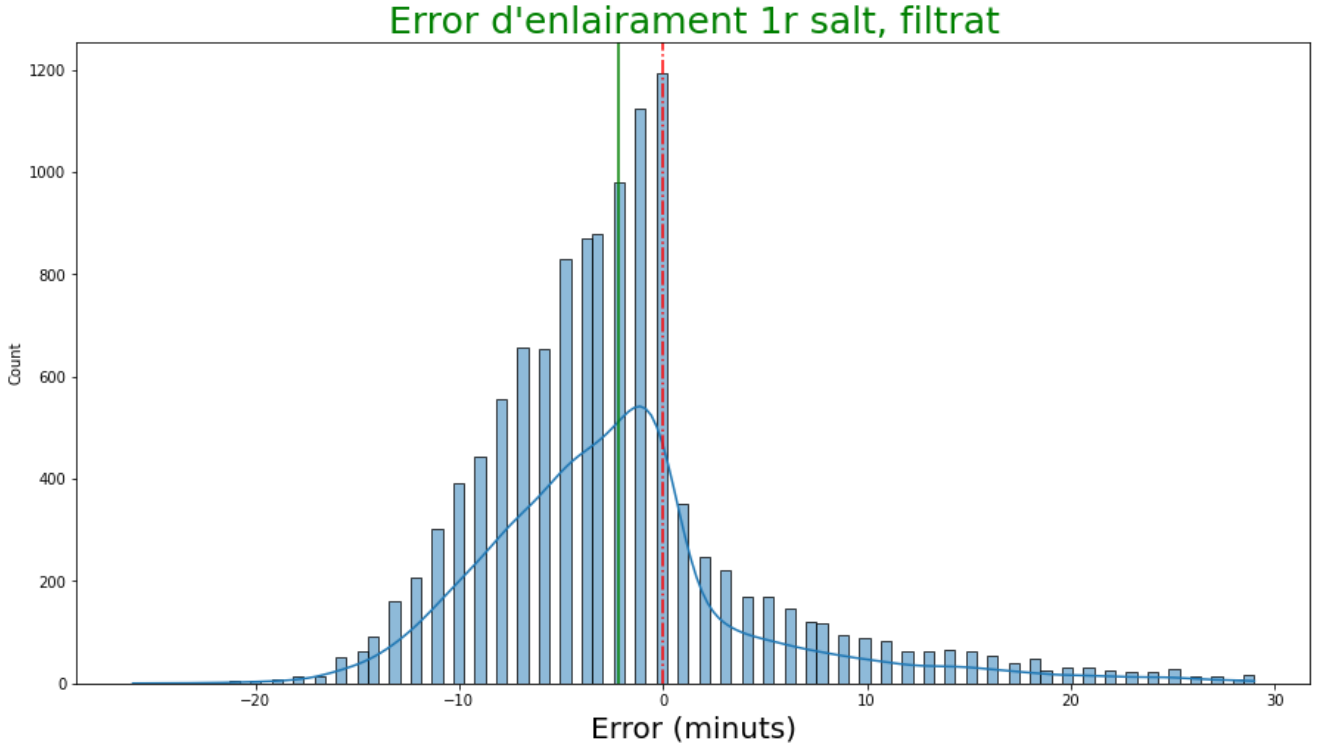
30 minuts el considerem un temps prudencial molt gran i que ha provocat un canvi de SLOT per part del Controlador Aeri

Hi ha molta dispersió i hem de filtrar les dades a:

error de sortida > 30 i < 30 i és el 1r enlairament

```
# Eliminar los outliers
df2=df.copy()
dfEliminarOutliers= ((df2['Secuencia']== 1) & (df2['E_Despegue']< 30) & (df2['E_Despegue']
df2 = df[dfEliminarOutliers]
#
plt.figure(figsize = (15,8))
p= sns.histplot(data= df2.E_Despegue, kde= True)
p.set_title("Error d'enlairament 1r salt, filtrat", fontsize=25, color='green')
mediaFiltrada = round(df2['E_Despegue'].mean(),2)
p.axvline(mediaFiltrada, 0,12, color = 'green')                        # Media real
p.axvline(x = 0, ymin=0, ymax= 12, color='red', linestyle= 'dashdot') # Objetivo
print()
print('Mitjana filtrada = ', mediaFiltrada, 'minuts')
#print(df2[['Secuencia', 'E_Despegue', 'AeropuertoKey1' ]])
p.set_xlabel("Error (minuts)", fontsize = 20)
```

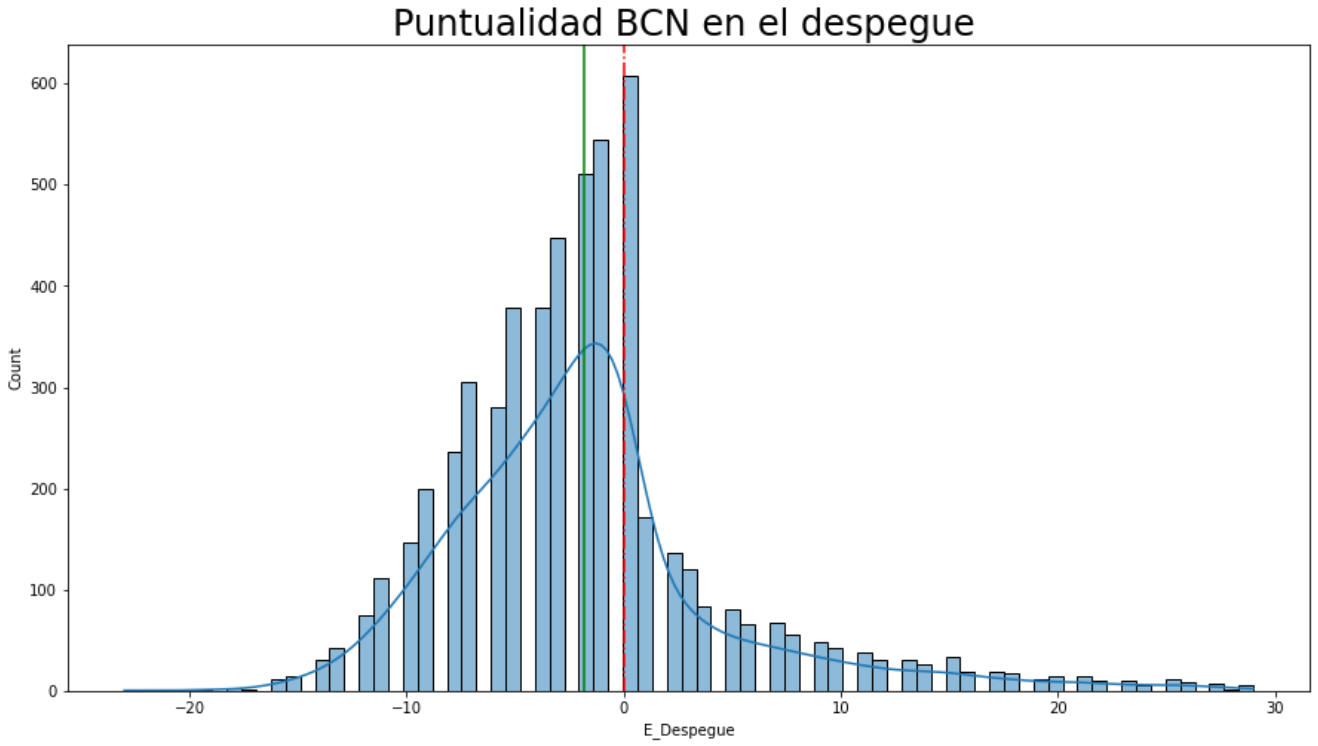
```
Mitjana filtrada =  -2.2 minuts
Text(0.5, 0, 'Error (minuts)')
```




```
# Per saber la puntualitat només de Barcelona
df2barcelonaPuntualida = (df2['DEP']== 'BCN')


df2BarcelonaPuntualida = df2[df2barcelonaPuntualida]
df2barcelonaPuntualida.head()

plt.figure(figsize = (15,8))
```

```
puntualidadBCN = round((df2BarcelonaPuntualida['E_Despegue'].mean()),2)#
print('Puntualidad BCN en el 1r enlairament:' , puntualidadBCN )

p=sns.histplot(data= df2BarcelonaPuntualida.E_Despegue, kde= True).set_title('Puntualidad


#Linea d'objectius
plt.axvline(puntualidadBCN, 0,12, color = 'green')                          # Media real
plt.axvline(x = 0, ymin=0, ymax= 12, color='red', linestyle= 'dashdot') # Objetivo

```

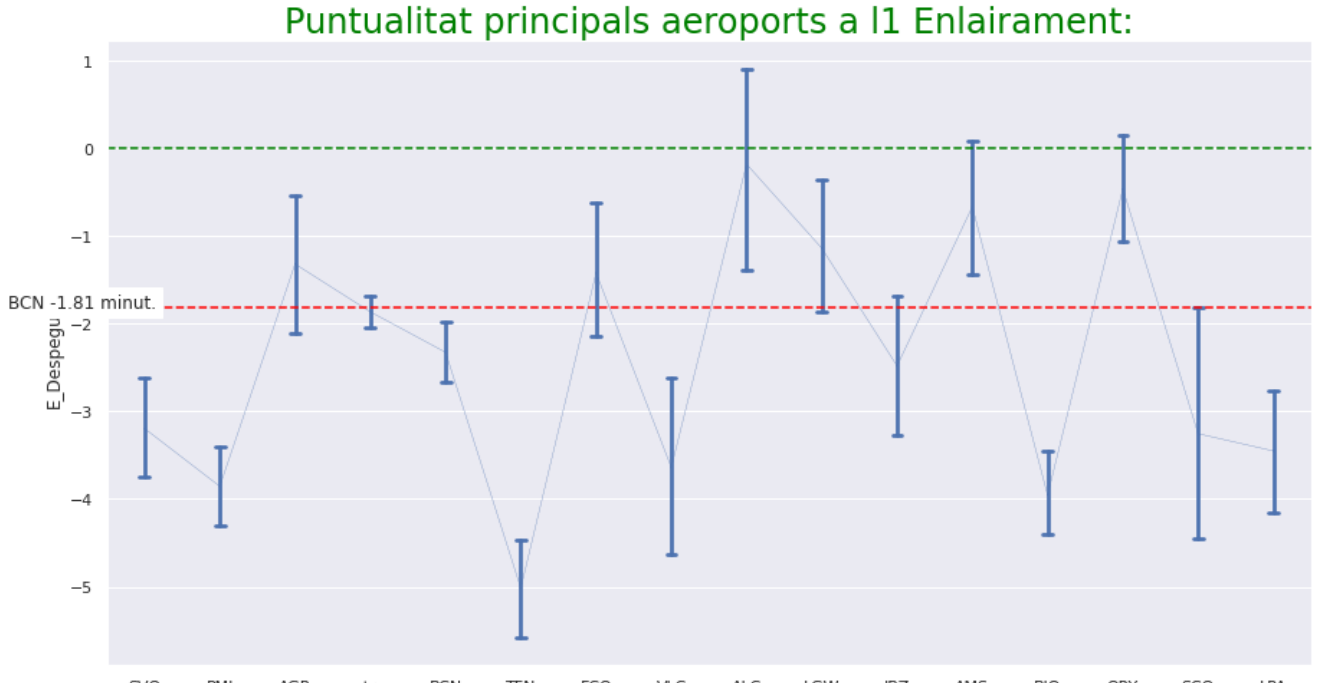
```
Puntualidad BCN en el 1r enlairament: -1.81
<matplotlib.lines.Line2D at 0x7f2fb214f250>
```



Intervals de confiança per aeroport.

```
#@title Intervals de confiança per aeroport.

sns.set_theme(style="darkgrid")
#tips = sns.df1("tips")
sns.set(rc = {'figure.figsize':(15,8)})
# Tamaño de la imagen
ax = sns.pointplot(x='AeropuertoKey1', y = "E_Despegue", data= df2, scale=.1, kind = "poin

ax.axhline(puntualidadBCN, color="red", linestyle = '--', label="ooas")       # Linea 0 ve
ax.axhline(0, color="green", linestyle = '--')


textoBCN = "BCN "+ str(puntualidadBCN) + ' minut.'
ax.text(puntualidadBCN ,puntualidadBCN , textoBCN, backgroundcolor='w')

ax.set_title('Puntualitat principals aeroports a l1 Enlairament:', fontsize = 24, color= '
```

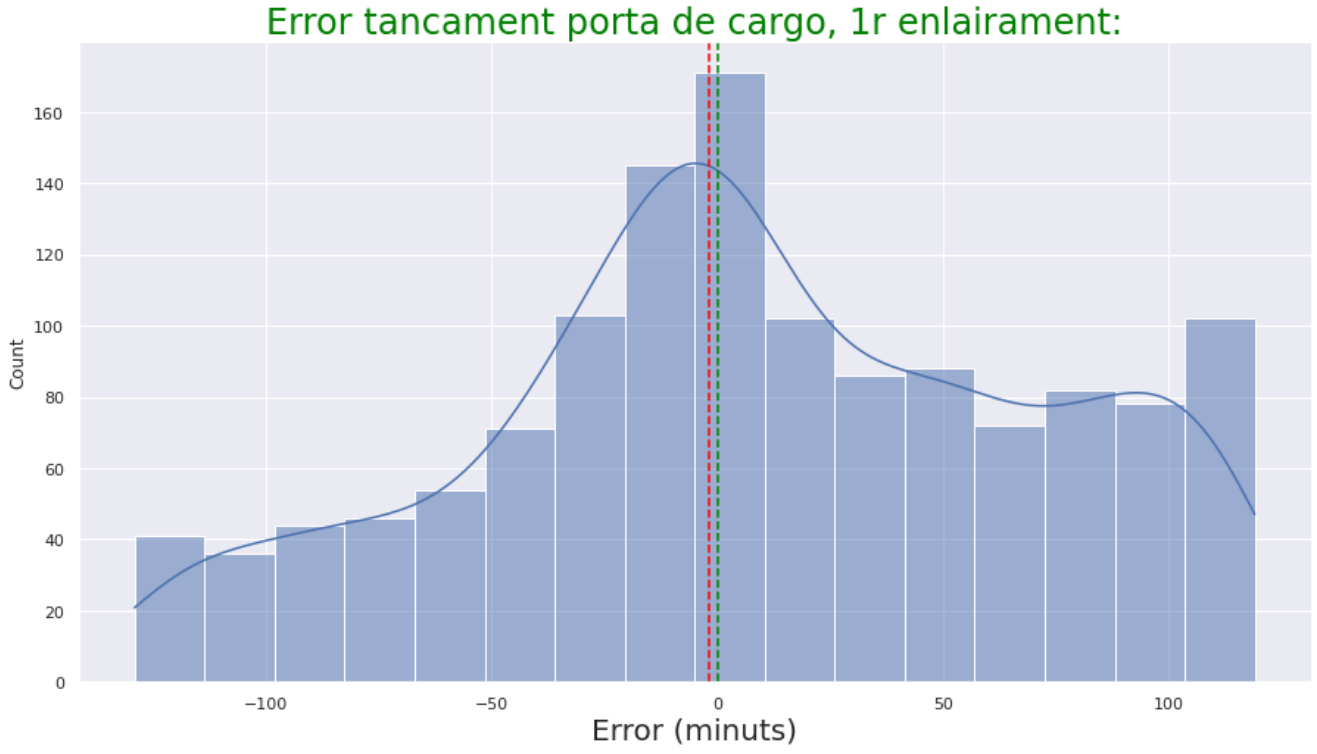
```
Text(0.5, 1.0, 'Puntualitat principals aeroports a l1 Enlairament:')
```



t_Close_Cargo_Door

Mostrar código

```
Media filtrada =  9.04 minutos
N=  71334
vegadas s'ha tanca >0 minuts:  816
Count of values greater than 20 in Column F :  693
 El porcentatge es de: 0.97
```

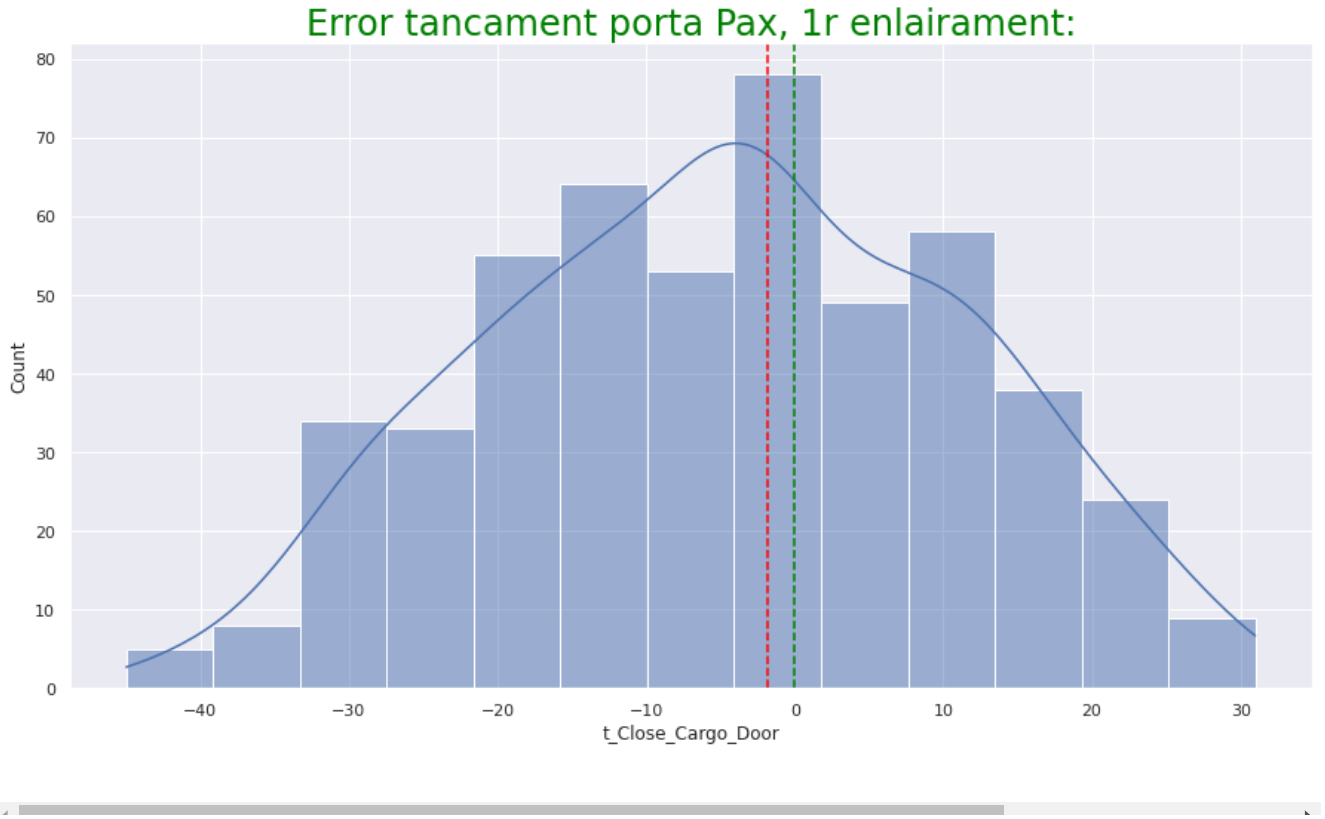


tiempo 'tiempo Close Pax' - 'ATD'

Texto de título predeterminado

Mostrar código

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: Boolean Series key will be rei

Media filtrada =  -1.5 minutos
Text(0.5, 1.0, 'Error tancament porta Pax, 1r enlairament:')
```




```
ax=sns.histplot(data= df2BarcelonaPuntualida.E_Despegue, kde= True)

ax.axvline(puntualidadBCN, color="red", linestyle = '--', label="ooas")       # Linea 0 ve
ax.axvline(0, color="green", linestyle = '--')

```

```
<matplotlib.lines.Line2D at 0x7f2fb1dc4a10>
```

creamos los LAGs

Creem els LAGs

```
#@title Creem els LAGs

df= df.sort_values(['DATE', 'REG', 'STD'])


df[f'lag_STD']= df['STD'].shift(periods=-1)
#adecuarFechas('DATE', 'lag_STD')
df[f'lag_ATD']= df['ATD'].shift(periods=-1)

df[f'lag_STA']= df['STA'].shift(periods=-1)
#adecuarFechas('DATE', 'lag_STD')
df[f'lag_ATA']= df['ATA'].shift(periods=-1)

#adecuarFechas('Pasajeros', 'ACT PAX  ')
df[f'lag_ACT PAX']= df['ACT PAX'].shift(periods=-1)

#Lag de la siguiente secuencia
df[f'lag_Secuencia']= df['Secuencia'].shift(periods=-1)
df[f'lag_REG']= df['REG'].shift(periods=-1)

df[['lag_Secuencia', 'lag_REG']]
```

	lag_Secuencia	lag_REG
0	2.0	EC-JSY
1	3.0	EC-JSY
2	4.0	EC-JSY
3	5.0	EC-JSY
4	6.0	EC-JSY
...	...	...
56128	3.0	EC-NLY
56129	4.0	EC-NLY
56130	5.0	EC-NLY
56131	6.0	EC-NLY
56132	NaN	NaN

56133 rows × 2 columns

Tiempo en tierra de una avion despues primer salto

df3 --> Calculo tiempo en tierra

Haz doble clic (o pulsa Intro) para editar

E. Enlairament primer salt del matí

Mostrar código

```
    FLT  Secuencia                STD                ATD  \
0  2506        1.0 2022-01-01 07:40:00 2022-01-01 08:00:00
1  2505        2.0 2022-01-01 09:40:00 2022-01-01 09:51:00
2  1386        3.0 2022-01-01 11:40:00 2022-01-01 11:40:00
3  1387        4.0 2022-01-01 13:30:00 2022-01-01 13:23:00

             lag_STD  T_teoricoTierra1  E_tierra1
0 2022-01-01 09:40:00              35.0       -4.0
1 2022-01-01 11:40:00              35.0       -4.0
2 2022-01-01 13:30:00              40.0       -2.0
3 2022-01-01 15:15:00              40.0       16.0

Media filtrada =  5.51 minutos
```

```
# Retraso
df['Puntualidad1'] = (df['ATA'] - df['STA']) /  np.timedelta64(1, 'm')
df['Puntualidad1'].head(3)
```

```
0    15.0
1     4.0
2    -5.0
Name: Puntualidad1, dtype: float64
```

Calculo Tiempo de carga y descarga de pasajeros:

Cojo solo las segundas secuencias que tengan el siguiente vuelo el mismo REG

```
# Total pasajeros Boarding

df['Total_PAX_Boarding'] = df['lag_ACT PAX'] +df['ACT PAX']

df['Total_PAX_Boarding']

df['T_Medio_Boarding'] = round(df['Total_PAX_Boarding'] / df['T_RealTierra1'],1)
df[['T_Medio_Boarding', 'Total_PAX_Boarding', 'T_RealTierra1']]
#df['T_Medio_Boarding']
```

	T_Medio_Boarding	Total_PAX_Boarding	T_RealTierra1
0	6.6	206.0	31.0
1	4.4	135.0	31.0
2	2.1	78.0	38.0
3	3.6	202.0	56.0
4	7.5	276.0	37.0
...	...	...	...
56128	8.5	407.0	48.0
56129	9.2	367.0	40.0
56130	2.9	329.0	115.0
56131	8.6	345.0	40.0
56132	NaN	NaN	NaN

56133 rows × 3 columns

```
# Calcul dels temps de Taxi.

df['Taxi_Despegue'] = (df['TKof']-df['ATD']) /  np.timedelta64(1, 'm')
df['Taxi_Aterrizaje'] = (df['ATA']- df['TDwn']) /  np.timedelta64(1, 'm')
df[['Taxi_Despegue', 'Taxi_Aterrizaje']]
```

	Taxi_Despegue	Taxi_Aterrizaje
0	10.0	3.0
1	12.0	2.0
2	10.0	5.0
3	7.0	3.0
4	16.0	7.0
...	...	...
56128	10.0	4.0
56129	24.0	5.0
56130	7.0	4.0
56131	24.0	7.0
56132	16.0	3.0

56133 rows × 2 columns

Tiempo de rodadura (En el despegue y en el aterrizaje

Càlcul temps de vol teòric Real i error

```
#@title Càlcul temps de vol teòric Real i error
df['DuracionVueloTeorico']= (df['STA'] - df['STD'])/  np.timedelta64(1, 'm')
df['DuracionVueloReal'] = (df['ATA'] - df['ATD']) /  np.timedelta64(1, 'm')
df['E_Duracion_Vuelo'] = (df['DuracionVueloReal'] - df['DuracionVueloTeorico']) #/  np.tim
df['E_Duracion_Vuelo'].head(3)

media = round(df['E_Duracion_Vuelo'].mean(),2)

print('\n\nError medio en tiempo de vuelo',media)
```

```
Error medio en tiempo de vuelo -5.51
```

```
#Creamos los LAGs

df[f'E_Despegue2']= df['E_Despegue'].shift(periods=-1)
df[f'E_Despegue3']= df['E_Despegue'].shift(periods=-2)
df[f'E_Despegue4']= df['E_Despegue'].shift(periods=-3)

df[f'E_Duracion_Vuelo2']= df['E_Duracion_Vuelo'].shift(periods=-1)
df[f'E_Duracion_Vuelo3']= df['E_Duracion_Vuelo'].shift(periods=-2)
df[f'E_Duracion_Vuelo4']= df['E_Duracion_Vuelo'].shift(periods=-3)

df[f'E_tierra2']= df['E_tierra1'].shift(periods=-1)
df[f'E_tierra3']= df['E_tierra1'].shift(periods=-2)
df[f'E_tierra4']= df['E_tierra1'].shift(periods=-3)

```

```
df[f'Aeropuerto_Key2']= df['Aeropuerto_Key'].shift(periods=-1)
df[f'Aeropuerto_Key3']= df['Aeropuerto_Key'].shift(periods=-2)
df[f'Aeropuerto_Key4']= df['Aeropuerto_Key'].shift(periods=-3)

df[f'Puntualidad2']= df['Puntualidad1'].shift(periods=-1)
df[f'Puntualidad3']= df['Puntualidad1'].shift(periods=-2)
df[f'Puntualidad4']= df['Puntualidad1'].shift(periods=-3)

df[f'E_Despegue_Total'] = df[f'E_Despegue'] + df[f'E_Despegue2'] + df[f'E_Despegue3'] + df
df[f'E_Duracion_Vuelo_Total'] = df[f'E_Duracion_Vuelo'] + df[f'E_Duracion_Vuelo2'] + df[f'
df[f'E_tierra_Total'] = df[f'E_tierra1'] + df[f'E_tierra2'] + df[f'E_tierra3']    + df[f'E

df['E_acumulado_Total'] = df[f'E_Despegue_Total'] + df[f'E_Duracion_Vuelo_Total'] + df[f'E

```

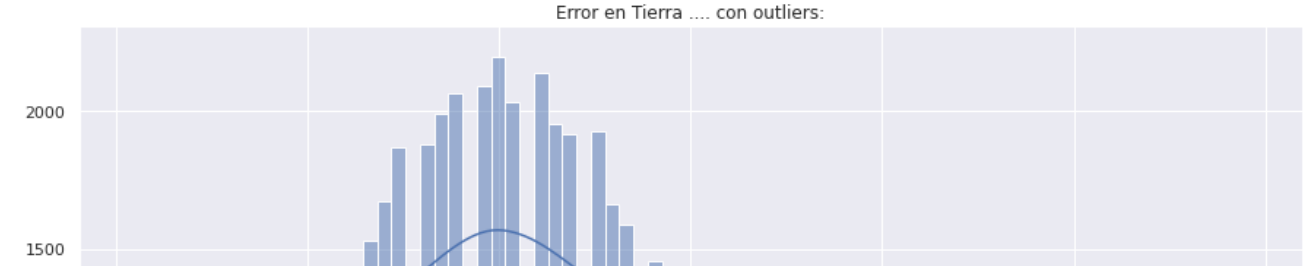
```
df.columns
```

```
Index(['DATE', 'FLT', 'REG', 'AC', 'DEP', 'ARR', 'STD', 'STA', 'TKof', 'TDwn',
       'ATD', 'ATA', 'BLOCK', 'FLThr', 'DStand', 'AStand', 'ACT PAX', 'LF',
       'Taxi-out', 'Taxi-In', 'SLOT', 'C1', 'DLY1', 'Sub1', 'C2', 'DLY2',
       'Sub2', 'C3', 'DLY3', 'Sub3', 'C4', 'DLY4', 'Sub4', 'C1Arr', 'DLY1Arr',
       'Close Pax Door', 'Close Cargo Door', 'Open Cargo/Pax Door',
       'close pax door2', 'close cargo door2', 'open cargo/pax door2',
       'Secuencia', 'MES', 'Setmana', 'DiaSetmana', 'DiaSetmanaName',
       'Aeropuerto_Key', 'AeropuertoKey1', 't_ClosePax', 't_Close_Cargo_Door',
       't_Entre_Puertas', 'Retardo_Abrir_Puerta_Pax', 'Trayecto', 'E_Despegue',
       'lag_STD', 'lag_ATD', 'lag_STA', 'lag_ATA', 'lag_ACT PAX',
       'lag_Secuencia', 'lag_REG', 'T_teoricoTierra1', 'T_RealTierra1',
       'E_tierra1', 'Puntualidad1', 'Total_PAX_Boarding', 'T_Medio_Boarding',
       'Taxi_Despegue', 'Taxi_Aterrizaje', 'DuracionVueloTeorico',
       'DuracionVueloReal', 'E_Duracion_Vuelo', 'E_Despegue2', 'E_Despegue3',
       'E_Despegue4', 'E_Duracion_Vuelo2', 'E_Duracion_Vuelo3',
       'E_Duracion_Vuelo4', 'E_tierra2', 'E_tierra3', 'E_tierra4',
       'Aeropuerto_Key2', 'Aeropuerto_Key3', 'Aeropuerto_Key4', 'Puntualidad2',
       'Puntualidad3', 'Puntualidad4', 'E_Despegue_Total',
       'E_Duracion_Vuelo_Total', 'E_tierra_Total', 'E_acumulado_Total'],
      dtype='object')
```

retard 3r salt BORRARRRRRR:

```
#@title retard 3r salt BORRARRRRRR:
dfEliminarOutliers= ( (df['Puntualidad1']< 30) & (df['Puntualidad1']> -30))

df9 = df[dfEliminarOutliers]
#df9=df.copy()

sns.histplot(data= df9.Puntualidad1, kde= True).set(title='Error en Tierra .... con outlie
```

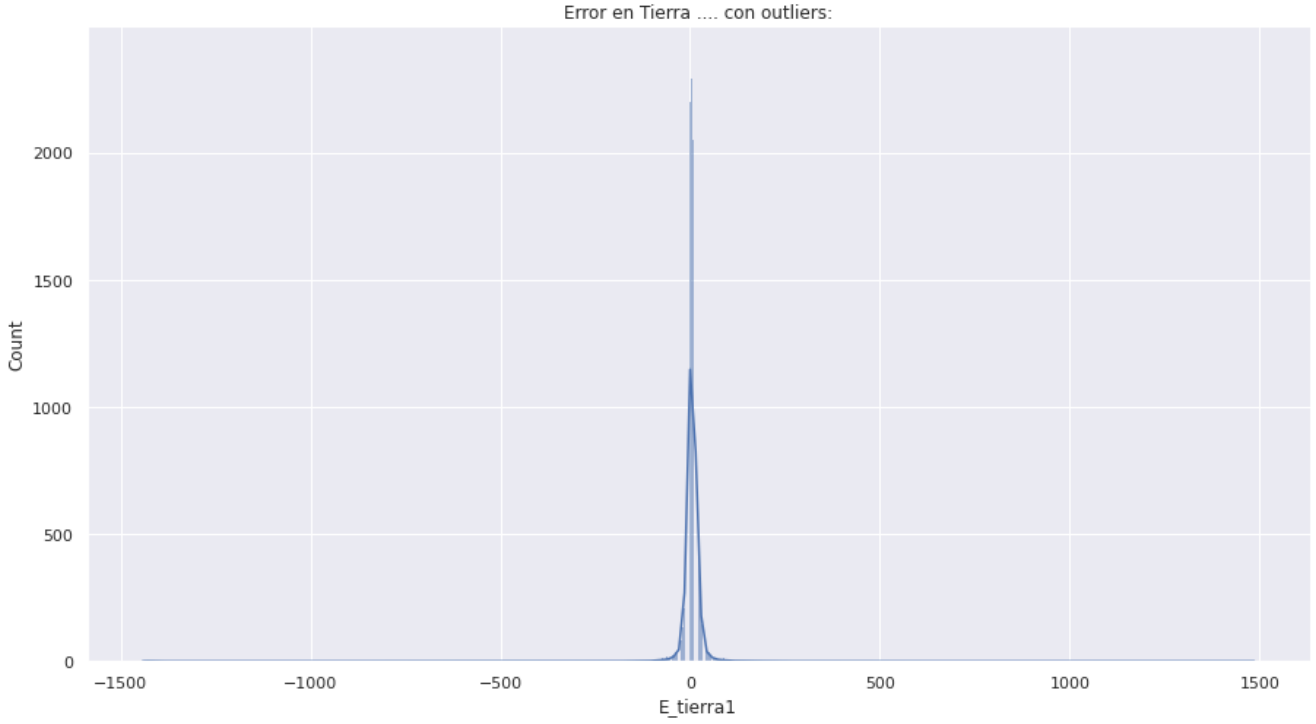
```
[Text(0.5, 1.0, 'Error en Tierra .... con outliers:')]
```



Estudio en Tierra primer aterrizaje.


```
sns.histplot(data= df3.E_tierra1, kde= True).set(title='Error en Tierra .... con outliers:
```

```
[Text(0.5, 1.0, 'Error en Tierra .... con outliers:')]
```

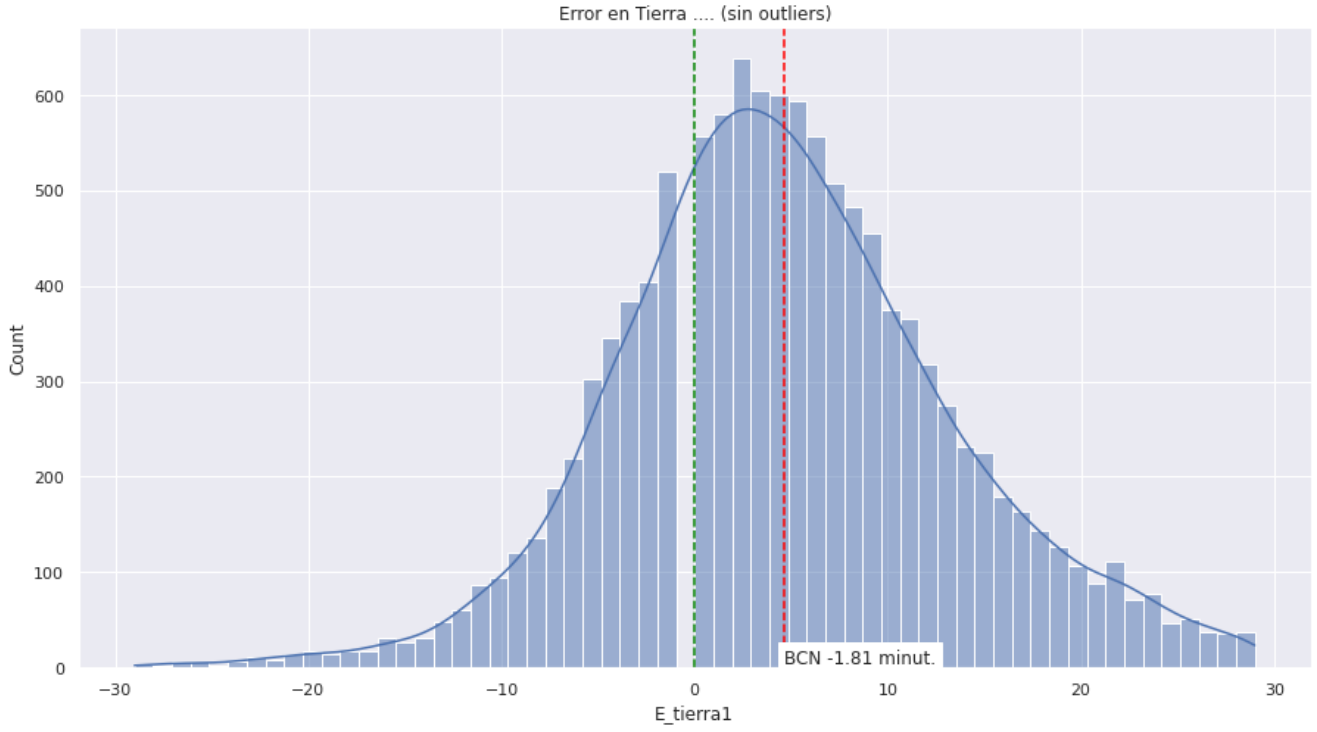


És clar que hi ha **molts de outiers** i igual que hem fet al primer enlairament, hem de fer als altres moviments de l'avió.

```
# Eliminar los outliers

#df2=df1
dfEliminarOutliers= ((df3['Secuencia']== 1) & (df3['E_tierra1']< 30) & (df3['E_tierra1']>

df3 = df3[dfEliminarOutliers]
ax = sns.histplot(data= df3.E_tierra1, kde= True)
ax.set_title("Error en Tierra .... (sin outliers)")
print()
print('Media filtrada = ', round(df3['E_tierra1'].mean(),2), 'minutos')

media = round(df3['E_tierra1'].mean(),2)

ax.axvline(media, color="red", linestyle = '--', label="ooas")       # Linea 0 verde
ax.axvline(0, color="green", linestyle = '--')
```

```
16
17
18 #textoBCN = "Media "+ str(media) + ' minut.'
19 ax.text(media  ,media, textoBCN, backgroundcolor='w')
20
```

```
Media filtrada =  4.63 minutos
Text(4.63, 4.63, 'BCN -1.81 minut.')
```




```
 1 def porcentajeErrorPPM (df5, accion):
 2   PPM_0=df5.apply(lambda x: x['E_tierra1'] > 0, axis=1).sum()
 3
 4   #print(df3.shape)
 5   #print(PPM)
 6
 7   PorcentajeError_0=round(PPM_0/df3.shape[0]*100,2)
 8   print("Cantitat d'avions que ", accion," despre´s de 0 minuts: ", PorcentajeError_0,'%'
 9   PPM_10=df5.apply(lambda x: x['E_tierra1'] > 10, axis=1).sum()
10   PorcentajeError_10=round(PPM_10/df3.shape[0]*100,2)
11   print("Cantitat d'avions que ", accion," despre´s de 10 minuts: ", PorcentajeError_10,'
12
13
14 print(porcentajeErrorPPM( df3, 'arriven'))
```

```
Cantitat d'avions que  arriven  despre´s de 0 minuts:  68.73 %
Cantitat d'avions que  arriven  despre´s de 10 minuts:  22.92 %
None
```

```
 1 # Crear Subgrupos de tiempo de boarding:
 2 '''
 3 df3['t_boarding'] = pd.cut(df3['T_teoricoTierra1'], bins = [ -9000,0, 41, 46,51,60,100000]
 4
 5
 6
 7
 8
 9 '''
```

```
10
11 df3['t_boarding'] = pd.cut(df3['T_teoricoTierra1'], bins = [ -9000, 0, 39, 40, 45, 50, 55,
12
13
14
15
16
17
18
19
20 df3[['T_teoricoTierra1', 't_boarding']].head(5)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:19: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
```

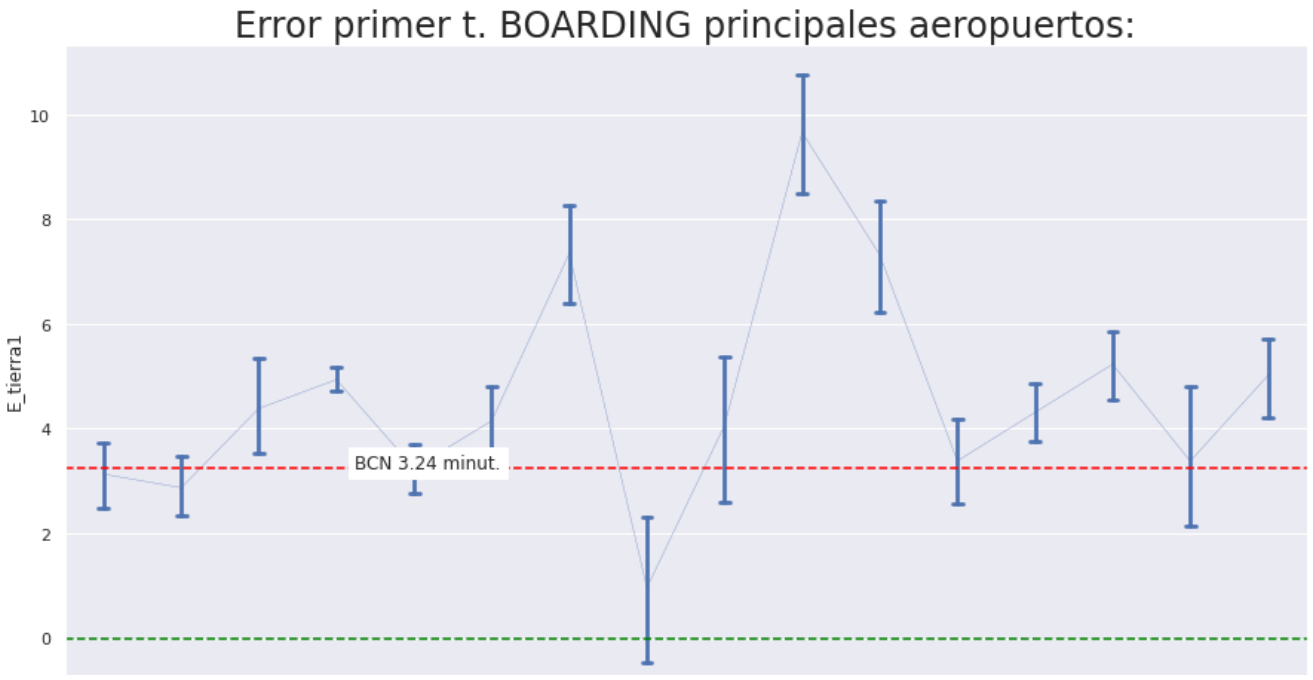
	T_teoricoTierra1	t_boarding
0	35.0	1.- <40
6	40.0	2.- 40
12	40.0	2.- 40
16	35.0	1.- <40
21	55.0	5.- 55

```
1 # Para saber error primer boarding solo de Barcelona
2
3 barcelonaTierra = (df3['ARR']== 'BCN')
4
5 df4BarcelonaTierra = df3[barcelonaTierra]
6 df4BarcelonaTierra .head()
7 df4BarcelonaTierraError  = round((df4BarcelonaTierra ['E_tierra1'].mean()),2)
8 print('\nError en el aterrizaje de Barcelona ', df4BarcelonaTierraError, 'minuts de mitjan
```

```
Error en el aterrizaje de Barcelona  3.24 minuts de mitjana
```

```
1 # Intervalos de Confianza por aeropuerto
2
3 sns.set_theme(style="darkgrid")
4 #tips = sns.df1("tips")
5 sns.set(rc = {'figure.figsize':(15,8)})
6 # Tamaño de la imagen
7 ax = sns.pointplot(x='AeropuertoKey1', y = "E_tierra1", data= df3, scale=.1, kind = "point
8
9 ax.axhline(df4BarcelonaTierraError , color="red", linestyle = '--', label="ooas")       #
10 ax.axhline(0, color="green", linestyle = '--')
11
12
13 textoBCN = "BCN "+ str(df4BarcelonaTierraError) + ' minut.'
14 ax.text(df4BarcelonaTierraError  ,df4BarcelonaTierraError  , textoBCN, backgroundcolor='w'
15
16 ax.set_title('Error primer t. BOARDING principales aeropuertos:', fontsize = 24)
```

```
Text(0.5, 1.0, 'Error primer t. BOARDING principales aeropuertos:')
```




```
1 df4BarcelonaTierra .head()
```

	DATE	FLT	REG	AC	DEP	ARR	STD	STA	TKof	TDwn	...	Aeropuerto_Key3	Aeropuert
21	2022-01-01	3909	EC-KDX	320	PMI	BCN	2022-01-01 17:35:00	2022-01-01 18:30:00	2022-01-01 17:45:00	2022-01-01 18:22:00	...	otro	
42	2022-01-01	2114	EC-LLM	320	AGP	BCN	2022-01-01 07:40:00	2022-01-01 09:15:00	2022-01-01 07:55:00	2022-01-01 09:09:00	...	SVQ	
237	2022-01-01	8031	EC-MMH	321	ORY	BCN	2022-01-01 08:35:00	2022-01-01 10:15:00	2022-01-01 09:02:00	2022-01-01 10:15:00	...	otro	
246	2022-01-01	2110	EC-MNZ	32A	AGP	BCN	2022-01-01 05:50:00	2022-01-01 07:25:00	2022-01-01 05:52:00	2022-01-01 07:00:00	...	otro	
290	2022-01-01	1429	EC-MXG	32A	BIO	BCN	2022-01-01 13:50:00	2022-01-01 15:00:00	2022-01-01 13:55:00	2022-01-01 14:41:00	...	IBZ	

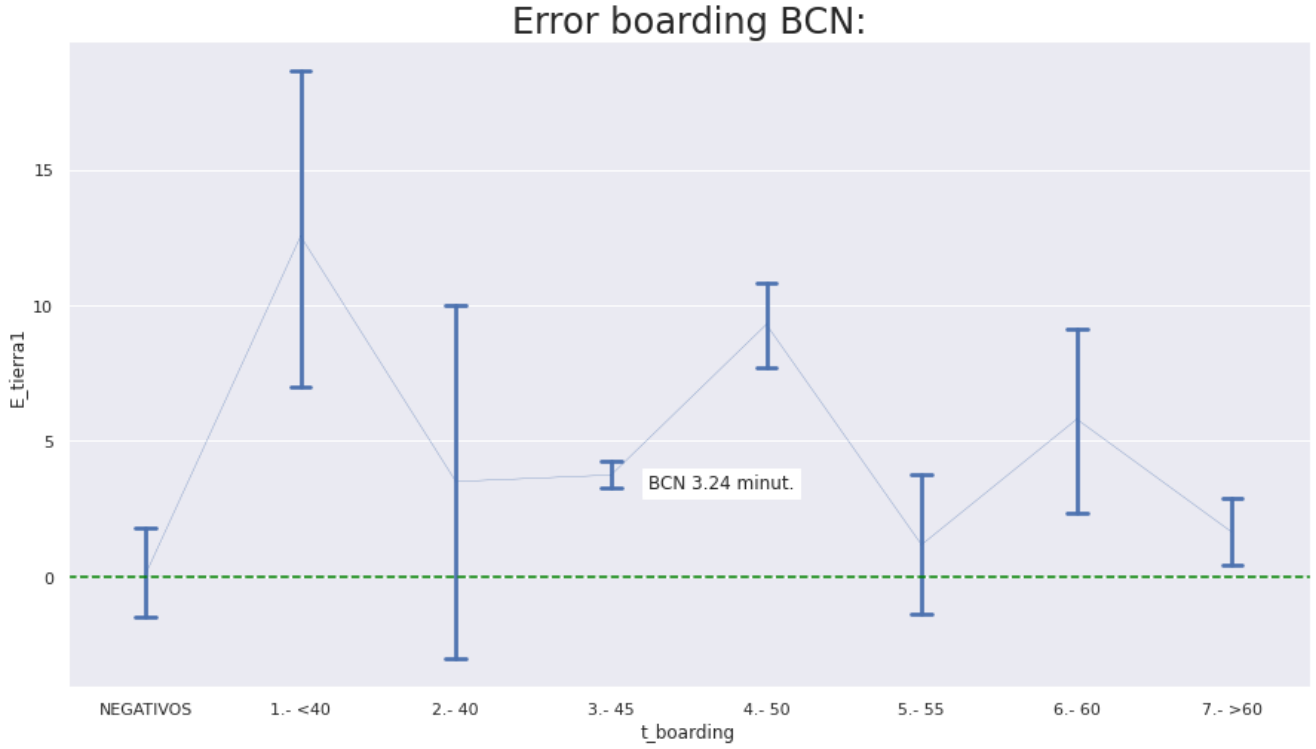
5 rows × 92 columns

```
1 # Contabilizar elementos por grupo de boarding
2 df5= df3['t_boarding'].value_counts()
3 #df5['t_boarding']= df5.sort_values('t_boarding')
4 df5.head()
5 df5
```

```
2.- 40       3373
1.- <40      3249
3.- 45       3012
4.- 50        833
7.- >60       706
NEGATIVOS     367
5.- 55        128
6.- 60        101
Name: t_boarding, dtype: int64
```

```
# Intervals de confiança per grup de boarding. Tipus avion --> temps assignat a terra)

sns.set_theme(style="darkgrid")
#tips = sns.df1("tips")
sns.set(rc = {'figure.figsize':(15,8)})
# Tamaño de la imagen
ax = sns.pointplot(x='t_boarding', y = "E_tierra1", data= df4BarcelonaTierra, scale=.1, ki

#ax.axhline(puntualidadBCN, color="red", linestyle = '--', label="ooas")       # Linea 0 v
ax.axhline(0, color="green", linestyle = '--')


textoBCN = "BCN "+ str(df4BarcelonaTierraError) + ' minut.'
ax.text(df4BarcelonaTierraError  ,df4BarcelonaTierraError  , textoBCN, backgroundcolor='w'

ax.set_title('Error boarding BCN:', fontsize = 24)
```

```
Text(0.5, 1.0, 'Error boarding BCN:')
```




```
1
```

Guardar a EXCEL

```
# Borro las columnas que no me interesan.
guardar = 'No'

'''
borrarColumnas = ['C1', 'DLY1', 'C2', 'DLY2',
       'C3', 'DLY3', 'C4', 'DLY4', 'C2A', 'DLY2A', 'C1Des', 'C2Des', 'C3Des',
       'C4Des', 'CADes', 'ID', 'STAND', 'MSN']
df3 = df3.drop(borrarColumnas, axis=1)
'''
```

```
'\nborrarColumnas = ['C1', 'DLY1', 'C2', 'DLY2',\n       'C3', 'DLY3', 'C4', 'DLY4', 'C2A', 'DLY2A', 'C1De
s', 'C2Des', 'C3Des',\n       'C4Des', 'CADes', 'ID', 'STAND', 'MSN']\ndf3 = df3.drop(borrarColumnas, axis=
```

```
df3.columns
```

```
Index(['DATE', 'FLT', 'REG', 'AC', 'DEP', 'ARR', 'STD', 'STA', 'TKof', 'TDwn',
       'ATD', 'ATA', 'BLOCK', 'FLThr', 'DStand', 'AStand', 'ACT PAX', 'LF',
       'Taxi-out', 'Taxi-In', 'SLOT', 'C1', 'DLY1', 'Sub1', 'C2', 'DLY2',
       'Sub2', 'C3', 'DLY3', 'Sub3', 'C4', 'DLY4', 'Sub4', 'C1Arr', 'DLY1Arr',
       'Close Pax Door', 'Close Cargo Door', 'Open Cargo/Pax Door',
       'close pax door2', 'close cargo door2', 'open cargo/pax door2',
       'Secuencia', 'MES', 'Setmana', 'DiaSetmana', 'DiaSetmanaName',
       'Aeropuerto_Key', 'AeropuertoKey1', 't_ClosePax', 't_Close_Cargo_Door',
       't_Entre_Puertas', 'Retardo_Abrir_Puerta_Pax', 'Trayecto', 'E_Despegue',
       'lag_STD', 'lag_ATD', 'lag_STA', 'lag_ATA', 'lag_ACT PAX',
       'lag_Secuencia', 'lag_REG', 'T_teoricoTierra1', 'T_RealTierra1',
       'E_tierra1', 'Puntualidad1', 'Total_PAX_Boarding', 'T_Medio_Boarding',
       'Taxi_Despegue', 'Taxi_Aterrizaje', 'DuracionVueloTeorico',
       'DuracionVueloReal', 'E_Duracion_Vuelo', 'E_Despegue2', 'E_Despegue3',
       'E_Despegue4', 'E_Duracion_Vuelo2', 'E_Duracion_Vuelo3',
       'E_Duracion_Vuelo4', 'E_tierra2', 'E_tierra3', 'E_tierra4',
       'Aeropuerto_Key2', 'Aeropuerto_Key3', 'Aeropuerto_Key4', 'Puntualidad2',
       'Puntualidad3', 'Puntualidad4', 'E_Despegue_Total',
       'E_Duracion_Vuelo_Total', 'E_tierra_Total', 'E_acumulado_Total',
       't_boarding'],
      dtype='object')
```

```
# Guardar excel
if guardar != 'No':
  nombreFichero = "D:\Documentos D\02.- Datos Vueling\Vueling_Python_MAR_JMML_" + Hoja + "
  df3.to_excel(nombreFichero)
  print()
  print('Guardado fichero : ', nombreFichero)
  print()
else:
  print('No guardado')
```

```
No guardado
```

```
# Pruebo esta manera de guardar el fichero EXCEL en DRIVE
# Cuidado que la hora no coincide con la del ordenador

if guardar != 'No':

  #Save only the first secuence.
  from datetime import datetime
  fecha=datetime.now()

  dfRegistros1= (df3['Secuencia']== 1)

  df7 = df3[dfRegistros1]

  path = '/content/drive/MyDrive/Ficheros de Vueling' + '/' + str(fecha) +' '+ Hoja + ".xl
  #df5.to_excel(r"D:\Documentos D\02.- Datos Vueling\Vueling_Python_MAR_Vuelo_" + Hoja + "

  df7.to_excel(path)
  print()
  print('Guardado fichero :' , path)
  print()
  print()
else:
  print('No guardado')
```

No guardado

```
#df7.columns()
```

```
print('forma ',df3.shape)

```

```
forma  (11769, 92)
```

Calcul temps en vuelo 1

```
df.columns
#print(df['Close Pax Door'])
```

```
Index(['DATE', 'FLT', 'REG', 'AC', 'DEP', 'ARR', 'STD', 'STA', 'TKof', 'TDwn',
       'ATD', 'ATA', 'BLOCK', 'FLThr', 'DStand', 'AStand', 'ACT PAX', 'LF',
       'Taxi-out', 'Taxi-In', 'SLOT', 'C1', 'DLY1', 'Sub1', 'C2', 'DLY2',
       'Sub2', 'C3', 'DLY3', 'Sub3', 'C4', 'DLY4', 'Sub4', 'C1Arr', 'DLY1Arr',
       'Close Pax Door', 'Close Cargo Door', 'Open Cargo/Pax Door',
       'close pax door2', 'close cargo door2', 'open cargo/pax door2',
       'Secuencia', 'MES', 'Setmana', 'DiaSetmana', 'DiaSetmanaName',
       'Aeropuerto_Key', 'AeropuertoKey1', 't_ClosePax', 't_Close_Cargo_Door',
       't_Entre_Puertas', 'Retardo_Abrir_Puerta_Pax', 'Trayecto', 'E_Despegue',
       'lag_STD', 'lag_ATD', 'lag_STA', 'lag_ATA', 'lag_ACT PAX',
       'lag_Secuencia', 'lag_REG', 'T_teoricoTierra1', 'T_RealTierra1',
       'E_tierra1', 'Puntualidad1', 'Total_PAX_Boarding', 'T_Medio_Boarding',
       'Taxi_Despegue', 'Taxi_Aterrizaje', 'DuracionVueloTeorico',
       'DuracionVueloReal', 'E_Duracion_Vuelo', 'E_Despegue2', 'E_Despegue3',
       'E_Despegue4', 'E_Duracion_Vuelo2', 'E_Duracion_Vuelo3',
       'E_Duracion_Vuelo4', 'E_tierra2', 'E_tierra3', 'E_tierra4',
       'Aeropuerto_Key2', 'Aeropuerto_Key3', 'Aeropuerto_Key4', 'Puntualidad2',
       'Puntualidad3', 'Puntualidad4', 'E_Despegue_Total',
       'E_Duracion_Vuelo_Total', 'E_tierra_Total', 'E_acumulado_Total'],
      dtype='object')
```

```
# Calculo tiempo de vuelo teorico Real y error
df['DuracionVueloTeorico']= (df['STA'] - df['STD'])/  np.timedelta64(1, 'm')
df['DuracionVueloReal'] = (df['ATA'] - df['ATD']) /  np.timedelta64(1, 'm')
df['E_Duracion_Vuelo'] = (df['DuracionVueloReal'] - df['DuracionVueloTeorico']) #/  np.tim
df['E_Duracion_Vuelo'].head(3)

media = round(df['E_Duracion_Vuelo'].mean(),2)

print('\n\nError medio en tiempo de vuelo',media)
```

```
Error medio en tiempo de vuelo -5.51
```

Estudio de la duracion del vuelo

```
# Identifico los Aeropuertos principales.


path =(r"/content/drive/MyDrive/Vueling_Trayectos_Duracion_Vuelo.xlsx")

#df_Aeropuertos = pd.read_excel(path, sheet_name='Aeropuertos')

#path =(r"D:\Documentos D\02.- Datos Vueling\Vueling_Trayectos_Duracion_Vuelo.xlsx")
```

```
 9
10
11 df_Trayectos = pd.read_excel(path, sheet_name='Trayectos')
12 df_Trayectos = df_Trayectos.drop(['Borrar'], axis=1)
13 df_Trayectos.head(6)
```

	Trayecto	grupoTrayecto
0	BCN-PMI	BCN-PMI
1	BCN-SVQ	BCN-SVQ
2	BCN-ORY	BCN-ORY
3	BCN-IBZ	BCN-IBZ
4	BCN-BIO	BCN-BIO
5	AGP-BCN	AGP-BCN

Creo la variable trayecte para identificar que trayectes tinc. Com un avió va de DEp (departure) a ARR (arival), per identificr les trayectories me es igual els trayectes DEP a ARR que si va de ARR a DEp. Vull dir per example de BCN a MAD es la mateixa distancai que des de MAD a BCN.

aquesta nova variable li diré

Calcul quantitat de trajectes

[ ] ↳ *22 celdas ocultas*

Estudio de los tiempos de TAXI

```
1 df[['Taxi_Despegue', 'Taxi_Aterrizaje']]
```

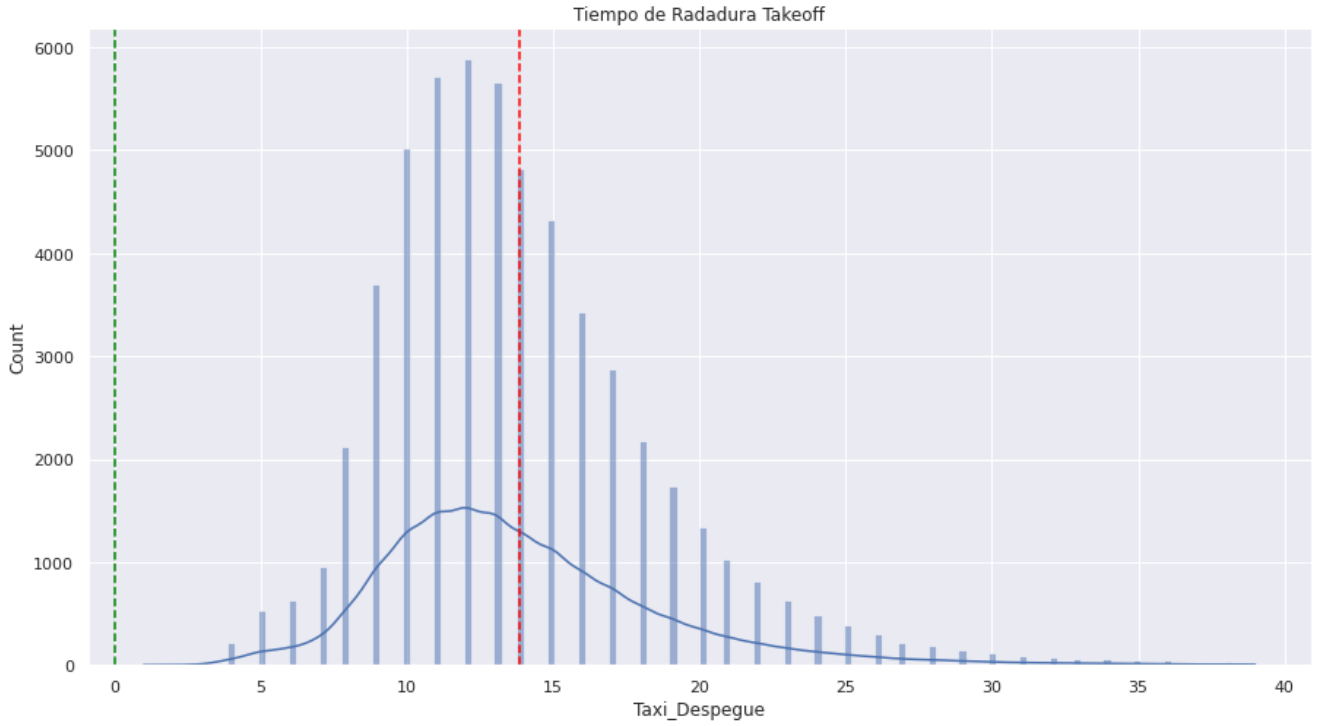
	Taxi_Despegue	Taxi_Aterrizaje
0	10.0	3.0
1	12.0	2.0
2	10.0	5.0
3	7.0	3.0
4	16.0	7.0
...	...	...
56128	10.0	4.0
56129	24.0	5.0
56130	7.0	4.0
56131	24.0	7.0
56132	16.0	3.0

56133 rows × 2 columns

Taxi Enlairament:

Mostrar código

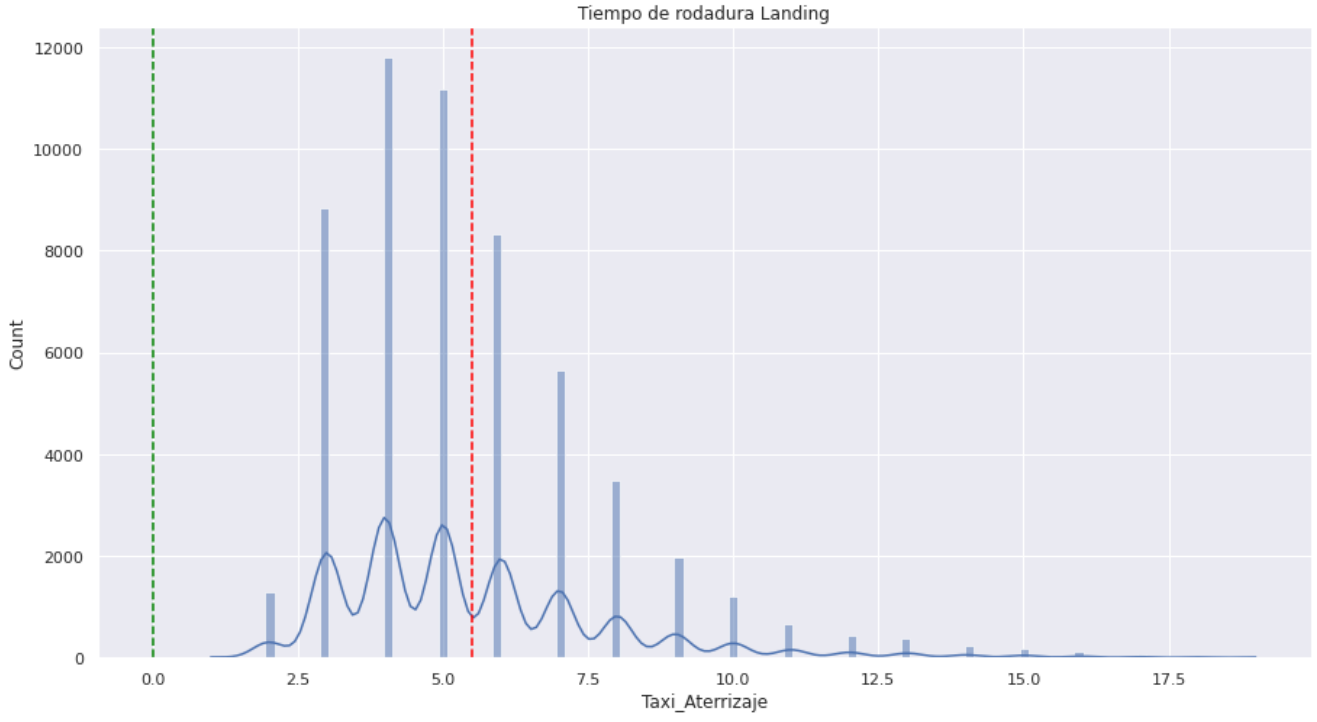
```
Media =  13.85 minutos
Text(0.5, 1.0, 'Tiempo de Radadura Takeoff')
```



Taxi Aterrizaje

Mostrar código

```
Media =  5.5 minutos
Text(0.5, 1.0, 'Tiempo de rodadura Landing')
```



Estudio la cantidad de tiempo que tardamos en vaciar y llenar un avion.

Mostrar código

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
```

	REG	lag_REG	Total_Pasajeros_Boarding	ACT PAX	lag_ACT PAX	t_Boarding_X_Pasajero	T_Medio_Boarding
4	EC-JSY	EC-JSY	276.0	155.0	121.0	7.46	7.5
6	EC-JZI	EC-JZI	175.0	57.0	118.0	3.89	3.9
7	EC-JZI	EC-JZI	241.0	118.0	123.0	5.36	5.4
8	EC-JZI	EC-JZI	123.0	123.0	0.0	4.24	4.2
9	EC-JZI	EC-JZI	125.0	0.0	125.0	3.05	3.0

Taxi Aterratje.

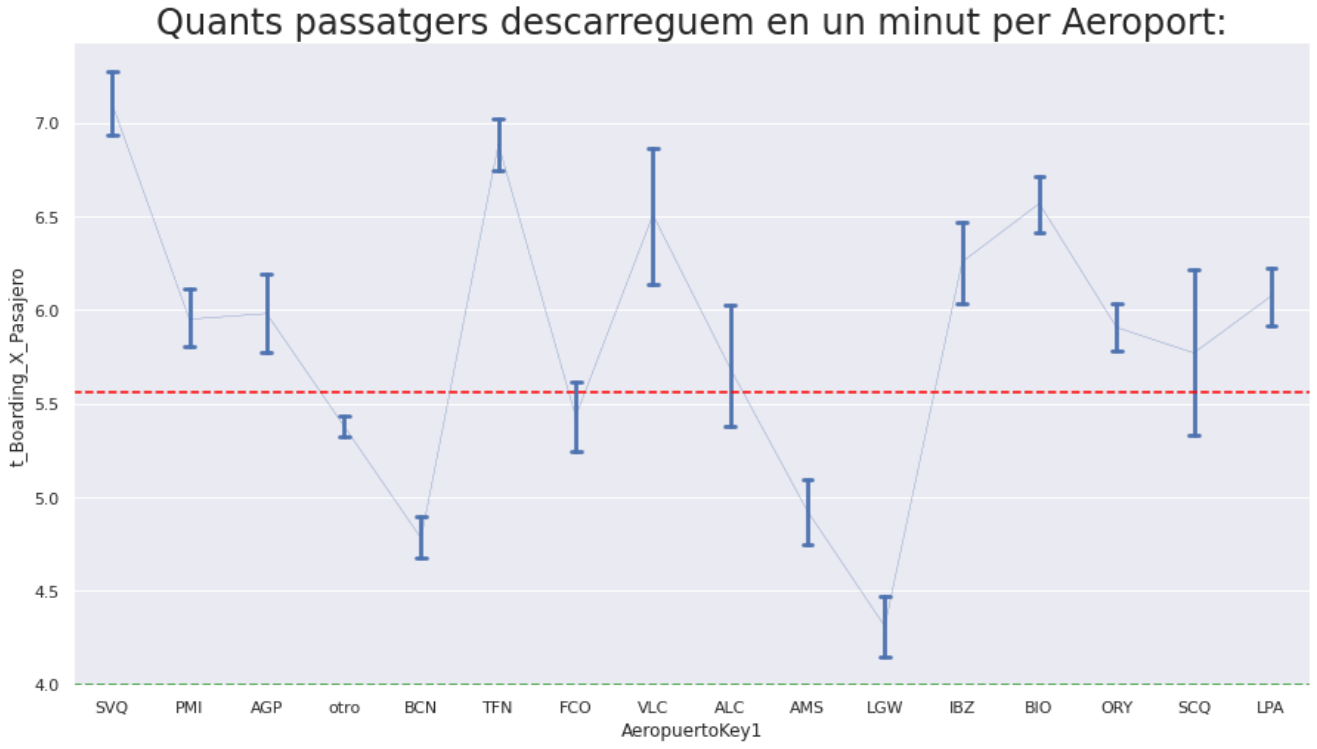
```
#@title Taxi Aterratje.
dfEliminarOutliers= ((df3['t_Boarding_X_Pasajero'] <20) &
                     (df3['t_Boarding_X_Pasajero'] >-10) &
                     (df3['Secuencia'] ==1))

df3 = df3[dfEliminarOutliers]
variableHistograma = 't_Boarding_X_Pasajero'
ax = sns.histplot(data= df3.t_Boarding_X_Pasajero, kde= True)
print()
print('Media = ', round(df3[variableHistograma].mean(),2), 'minutos')

media = round(df3[variableHistograma].mean(),2)

ax.axvline(media, color="red", linestyle = '--', label="ooas")       # Linea 0 verde
ax.axvline(0, color="green", linestyle = '--')
ax.set_title('Cuantos pasajeros descargamos en un minuto')
```

```
Media =  5.56 minutos
Text(0.5, 1.0, 'Cuantos pasajeros descargamos en un minuto')
```

Intervalos de Confianza por aeropuerto

```
#@title Intervalos de Confianza por aeropuerto
sns.set_theme(style="darkgrid")

sns.set(rc = {'figure.figsize':(15,8)})

# Tamaño de la imagen
ax = sns.pointplot(x='AeropuertoKey1', y = 't_Boarding_X_Pasajero', data= df3, scale=.1, k

ax.axhline(media, color="red", linestyle = '--', label="ooas")       # Linea 0 verde
ax.axhline(4, color="green", linestyle = '--')


#textoBCN = "BCN "+ str(puntualidadBCN) + ' minut.'
#ax.text(puntualidadBCN ,puntualidadBCN , textoBCN, backgroundcolor='w')

ax.set_title('Quants passatgers descarreguem en un minut per Aeroport:', fontsize = 24)
```

```
Text(0.5, 1.0, 'Quants passatgers descarreguem en un minut per Aeroport:')
```




```
df3['Secuencia']
```

```
0       1.0
6       1.0
12      1.0
16      1.0
21      1.0
       ...
```

```
56105    1.0
56108    1.0
56115    1.0
56121    1.0
56127    1.0
Name: Secuencia, Length: 11870, dtype: float64
```

Estudio del SLOT

Càlcul el temps que hi ha de diferent entre SLOT v STD

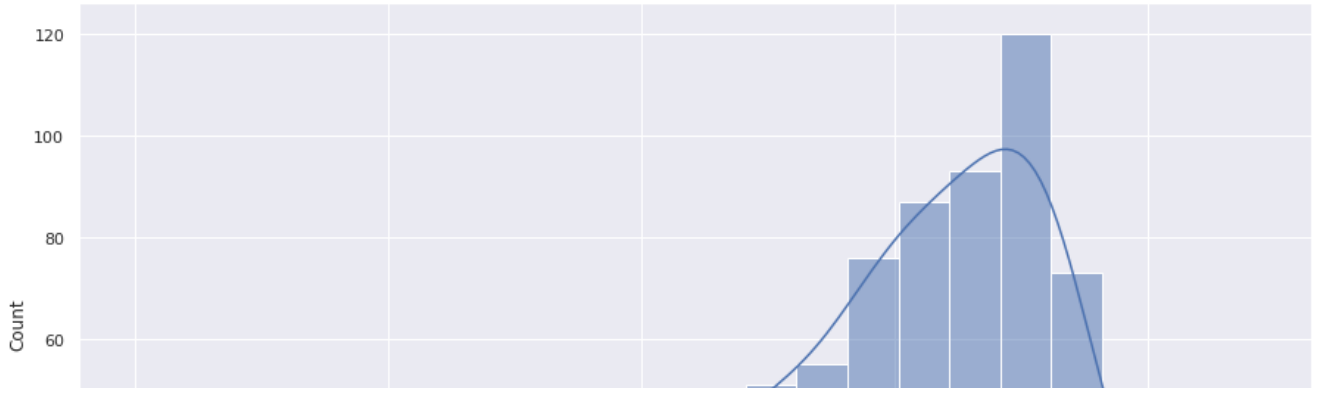
```
#@title Càlcul el temps que hi ha de diferent entre SLOT v STD
df3= df.copy()
dfEliminarOutliers= ((df3['E_Despegue'] >20) &(df3['DEP'] =='BCN'))

df3 = df3[dfEliminarOutliers]


#df3['E_SLOT']= df3['SLOT']-df3['STD']
df3[[ 'SLOT', 'STD', 'ATD', 'E_Despegue', 'DEP']]
```

	SLOT	STD	ATD	E_Despegue	DEP
91		2022-01-01 19:05:00	2022-01-01 20:33:00	88.0	BCN
94		2022-01-01 15:25:00	2022-01-01 15:47:00	22.0	BCN
106		2022-01-01 17:30:00	2022-01-01 19:48:00	138.0	BCN
178	06:46:00	2022-01-01 06:00:00	2022-01-01 06:31:00	31.0	BCN
219		2022-01-01 17:55:00	2022-01-01 18:23:00	28.0	BCN
...	...	...	...	...	...
55720		2022-05-16 18:30:00	2022-05-16 18:59:00	29.0	BCN
55722		2022-05-16 05:10:00	2022-05-16 05:42:00	32.0	BCN
55850	16:50:00	2022-05-16 16:00:00	2022-05-16 16:39:00	39.0	BCN
56073	09:22:00	2022-05-16 08:05:00	2022-05-16 09:13:00	68.0	BCN
56131	14:10:00	2022-05-16 13:20:00	2022-05-16 13:50:00	30.0	BCN

750 rows × 5 columns

Quantitat de passatgers que transporta

```
#@title Quantitat de passatgers que transporta

ax = sns.histplot(data= df3['ACT PAX'], kde= True)
```



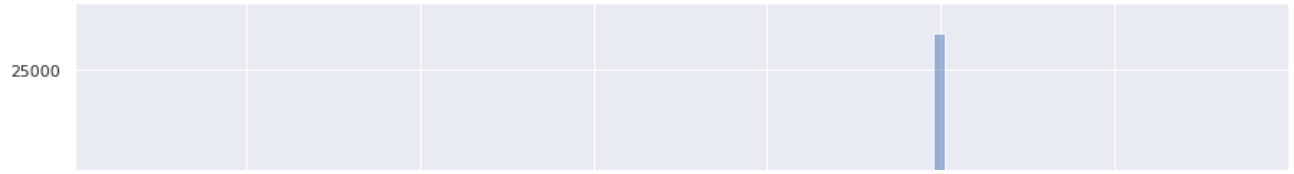
Estudi temps des des de que aterra fins que s'obre la porta càrrec i PAX

```
#@title Estudi temps des des de que aterra fins que s'obre la porta càrrec i PAX
df3=df.copy()

df3['retardoOperturaPuertaAterrizaje']=(df['Open Cargo/Pax Door']-df['ATA'])  /  np.timede
df3[['retardoOperturaPuertaAterrizaje','Open Cargo/Pax Door', 'ATA']]
```

	retardoOperturaPuertaAterrizaje	Open Cargo/Pax Door	ATA
0	3.0	2022-01-01 09:23:00	2022-01-01 09:20:00
1	2.0	2022-01-01 11:11:00	2022-01-01 11:09:00
2	1.0	2022-01-01 12:46:00	2022-01-01 12:45:00
3	1.0	2022-01-01 14:12:00	2022-01-01 14:11:00
4	0.0	2022-01-01 18:14:00	2022-01-01 18:14:00
...	...	...	...
56128	476.0	2022-05-16 15:33:00	2022-05-16 07:37:00
56129	479.0	2022-05-16 18:01:00	2022-05-16 10:02:00
56130	516.0	2022-05-16 20:31:00	2022-05-16 11:55:00
56131	-404.0	2022-05-16 09:34:00	2022-05-16 16:18:00
56132	-341.0	2022-05-16 13:21:00	2022-05-16 19:02:00

56133 rows × 3 columns

```
ax = sns.histplot(data= df3.retardoOperturaPuertaAterrizaje, kde= True)
```



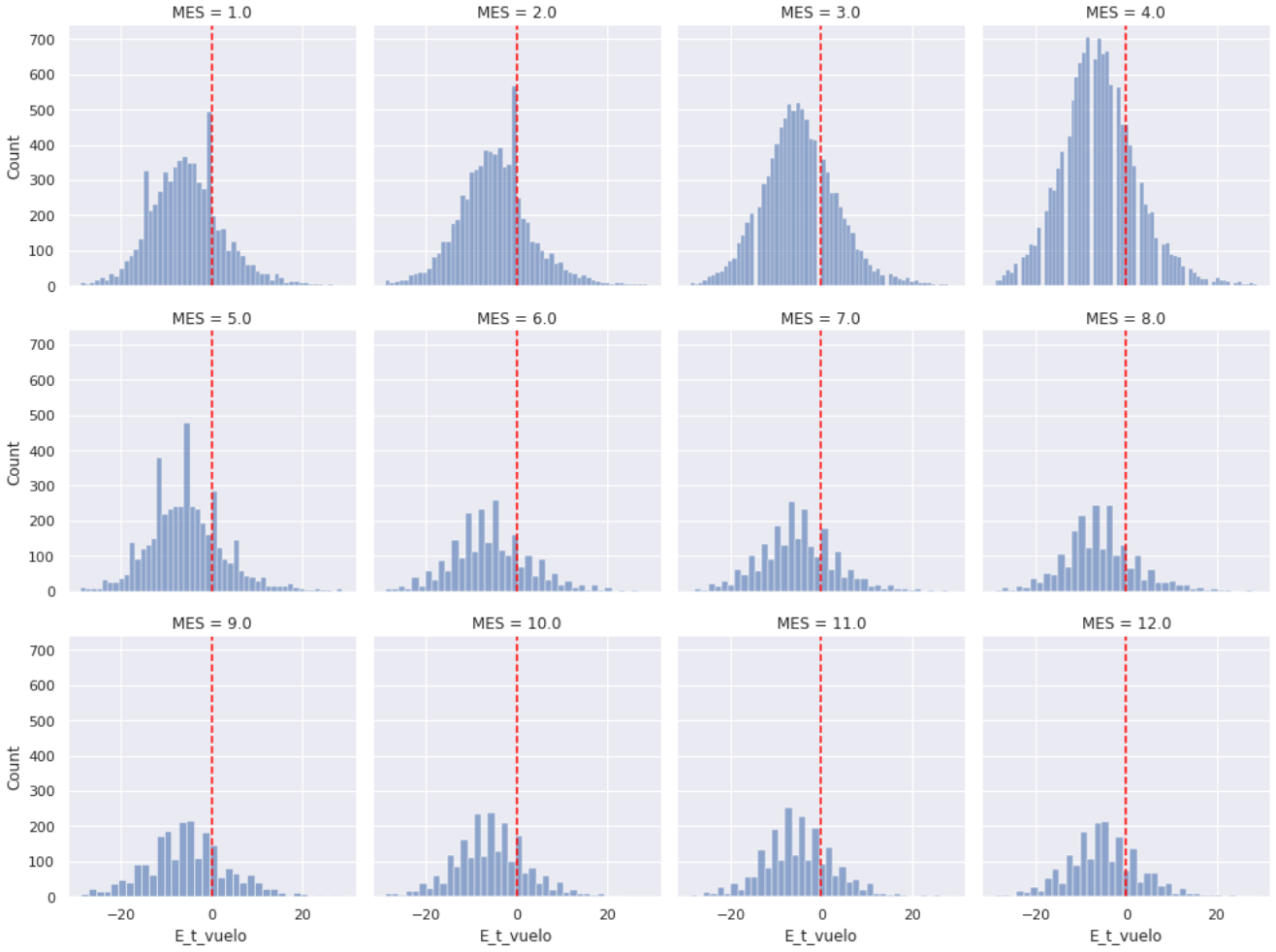
Correlaciones Error temps de Vol v mes dia setmana

Documentación: Building structured multi-plot grids

https://seaborn.pydata.org/tutorial/axis_grids.html

histogramas retards per mesos

```
#@title histogramas retards per mesos

g = sns.FacetGrid(df8, col="MES", height=3.5, col_wrap=4)
g.map(sns.histplot, "E_t_vuelo", alpha=0.6) # kde=False, fit=norm)
g.refline(x=0, color='red') #[, x, y, color, linestyle])
```

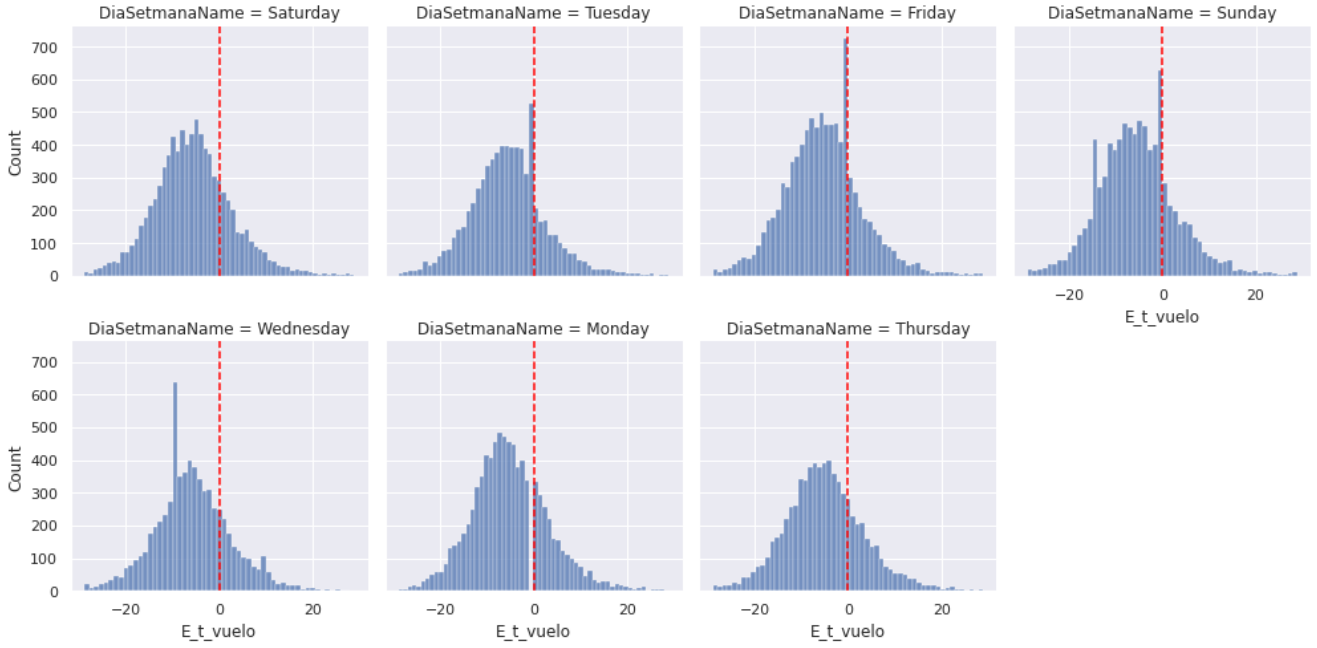


No se ve un claro y evidente diferencia entre los meses, por lo que consideraremos el comportamiento en los retrasos estable.

Correlacion dia de la setmana v mes en el Error del vol

```
#@title Correlacion dia de la setmana v mes en el Error del vol

g = sns.FacetGrid(df8, col="DiaSetmanaName", col_wrap=4, height=3.5) #, row='MES')
g.map(sns.histplot, "E_t_vuelo")
g.refline(x=0, color='red') #[, x, y, color, linestyle])
xmin, xmax = plt.xlim()
print(xmin, xmax)
```

```
-31.9 31.9
```



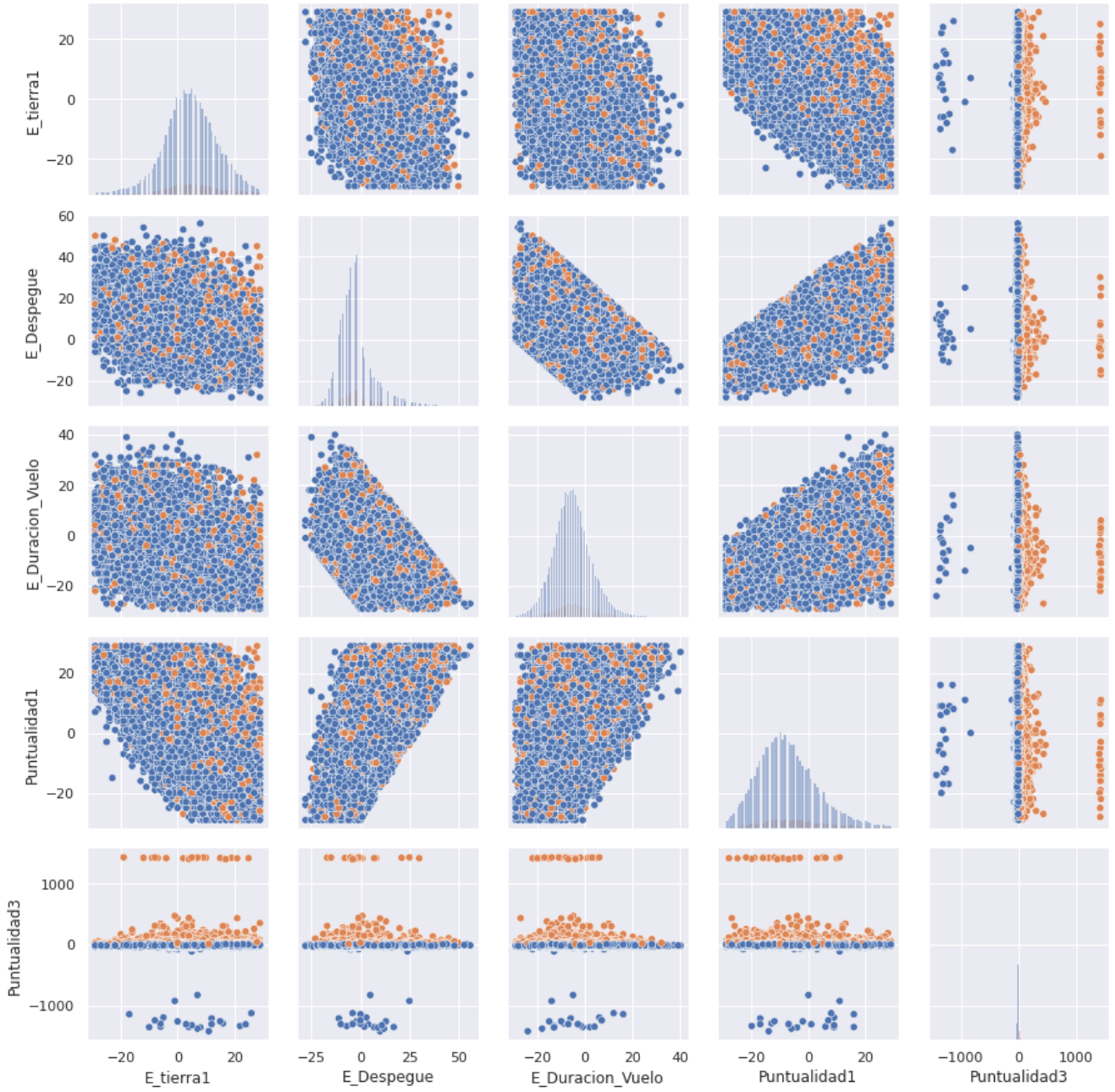

```
1
```

Box plot Errores v secuencia

```
#@title Box plot Errores v secuencia
dfEliminarOutliers= ((df['E_tierra1']< 30) & (df['E_Despegue']> -30) & (df['E_tierra1']> -

# Para crear una "logistic regresion":
# 10 minuts és el temps que s'accepta com arribar a l'hora

df['Tard1'] = np.where(df['Puntualidad1'] <10, 0, 1)
df['Tard2'] = np.where(df['Puntualidad2'] <10, 0, 1)
df['Tard3'] = np.where(df['Puntualidad3'] <10, 0, 1)



df9 = df[dfEliminarOutliers]

df9=df9[['E_tierra1','E_Despegue', 'E_Duracion_Vuelo', 'Puntualidad1','Tard3', 'Puntualida
#df8["tardp"] = df9["Tard3"].astype(str)


g = sns.PairGrid(df9, hue='Tard3')
```

```
g.map_diag(sns.histplot)
g.map_offdiag(sns.scatterplot)

```

```
<seaborn.axisgrid.PairGrid at 0x7f2fa8c0ccd0>
```




```
df9
```

	E_tierra1	E_Despegue	E_Duracion_Vuelo	Puntualidad1	Tard3	Puntualidad3
0	-4.0	20.0	-5.0	15.0	0	-5.0
1	-4.0	11.0	-7.0	4.0	0	-24.0
2	-2.0	0.0	-5.0	-5.0	0	-6.0
3	16.0	-7.0	-17.0	-24.0	0	-2.0
4	2.0	-8.0	2.0	-6.0	0	-22.0

Conclusions:

Hem vist en aquesta anàlisi de les dades, que els outliers afecten molt i donen molt de soroll. Hem acordat eliminar tots aquells que el seu error respecto al mateix temps que està establer siguin tots a 30 minuts, perquè considerem que

un temps d'error superior vol dir per exemple que l'avió està avariat i, per tant, segur que en els altres salts provocarà retards.

L'impacta de l'aeroport es important per controlar la puntualitat. La raó es veu que en els aeroports més grans este que donar més temps per fer l'embarcament que en els aeroports més petits. Això també ho hem vist quan hem fem l'anàlisi en els trajectes. Si està involucrat un aeroport gran, llavors té impacte:

Mitjana Error en els vols de tots els trajectes: -3.37 minuts Mitjana Error en els vols dels principals trajectes amb aeroports grans: -7.19 minuts.

Quan he calculat quants passatgers descarreguem en un minut per Aeroport, hem vist que l'aeroport de BCN té un rendiment molt petit comparat amb altres aeroports. Tenir un rendiment baix afecta la capacitat de controlar el temps que es requereix quan l'avió és a terra i això treu graus de llibertat en la puntualitat.

Una part curiosa ha sigut veure si existeix correlació entre Error del temps de vol v Duració del Vol. I la resposta és que no. Jo pensava que sí. La meva intuïció em deia que en els vols llargs els pilots són capaços de reduir el retard, però interpreto pels resultats obtinguts que no poden perquè estan en funció del que el controlador aeri els hi mana.

En l'últim gràfic veiem que per sapiguer si un avió tindrà retard amb la puntualitat del primer salt semble difil de trobar i aquesta serà la feina del seguent quadern:

3.c Companyia Aerea Algoritmes d'aprenentatge supervisat.ipynb

https://github.com/JMML2021/Entrega-projecte-final/blob/main/3.c%20Companyia%20Aerea%20Algoritmes%20d%E2%80%99aprenentatge%20supervisat.ipynb

També farem un quadern de benchmarking amb una altra companyia per veure si els resultats de puntualitat hi ha molta diferència.

Per descomptat es poden fer moltes més anàlisis, però l'objectiu d'aquest quadern és només veure com és el fitxer i fer una primera identificació de quin serà el comportament en la puntualitat del tercer salt.

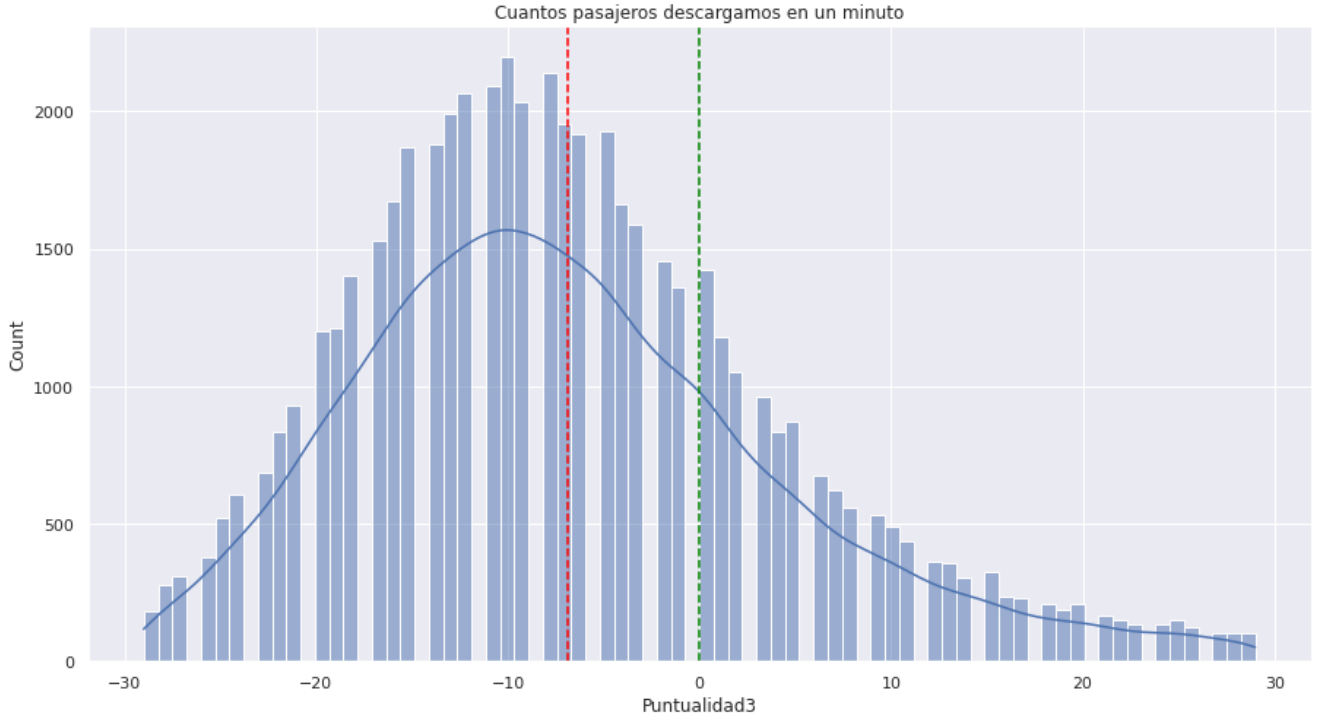
```
print('\n\nTotal quantitat de registres: ',df.shape)
print('Total quantitat de registres sense outliers: ',df9.shape)

print('Porcentatje de registres rebutjas com outliers: ', round((1-df9.shape[0]/df.shape[0
print
print('Total quantitat de registres seqüencia 1: ',df3.shape)
```

```
Total quantitat de registres:  (56133, 95)
Total quantitat de registres sense outliers:  (50699, 6)
Porcentatje de registres rebutjas com outliers:  9.68  %
Total quantitat de registres seqüencia 1:  (56133, 93)
```

Estudi Retard 3 salt:

```
#@title Estudi Retard 3 salt:

dfEliminarOutliers= ((df['Puntualidad3']< 30) & (df['Puntualidad3']> -30) )
df9 = df[dfEliminarOutliers]
df9=df9[['E_tierra1','E_Despegue', 'E_Duracion_Vuelo', 'Puntualidad1','Tard3', 'Puntualida

ax = sns.histplot(data= df9['Puntualidad3'], kde= True)
print()
media = round(df9['Puntualidad3'].mean(),2)
mediana = round(df9['Puntualidad3'].median(),2)
print('Media = ', media, 'minutos')
print('Mediana = ', mediana, 'minutos')


ax.axvline(media, color="red", linestyle = '--', label="ooas")       # Linea 0 verde
ax.axvline(0, color="green", linestyle = '--')
ax.set_title('Cuantos pasajeros descargamos en un minuto')

```

```
Media =  -6.9 minutos
Mediana =  -8.0 minutos
Text(0.5, 1.0, 'Cuantos pasajeros descargamos en un minuto')
```



Subrutina para el calcul del Cp i el Cpk

```
#@title Subrutina para el calcul del Cp i el Cpk

def Cp(clave, usl, lsl):

    sigma = df9[clave].std()


    Cp = abs(round((usl - lsl) / (6*sigma),2))
    print('Valor del Cp= ', Cp)
    return Cp
```

```

def Cpk(clave, usl, lsl):

    sigma = df9[clave].std()
    m = round(np.mean(df9[clave]),2)
    print('mitjana ', m)

    Cpu = float(-usl + m) / (3*sigma)
    Cpl = float(-m + lsl) / (3*sigma)
    #print('Cpks ', Cpu, Cpl)
    Cpk = round( np.min([Cpu, Cpl]),2)
    print('Valor del Cpk= ', Cpk)

    return Cpk
```

```
Cp('Puntualidad3', -30, 30)
Cpk('Puntualidad3', -30, 30)
print('\nSegond salt')
```

```
Valor del Cp=  0.91
mitjana  -6.9
Valor del Cpk=  0.7

Segond salt
```

```
print('Total de vols treballat: ', len(df9))
vecesLlegaTarde=len(df9[df9['Puntualidad3']>10])
print('Quantitat de vegades que arriba tard en el tercer salt: ', vecesLlegaTarde)
print('Equival a: ',round((vecesLlegaTarde/len(df9)),2)*100,'%')
```

```
Total de vols treballat:  53106
Quantitat de vegades que arriba tard en el tercer salt:  4014
Equival a:  8.0 %
```

```
!pip install session_info
import session_info


session_info.show()
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting session_info
  Downloading session_info-1.0.0.tar.gz (24 kB)
Collecting stdlib_list
  Downloading stdlib_list-0.8.0-py3-none-any.whl (63 kB)
     |████████████████████████████████| 63 kB 1.8 MB/s
Building wheels for collected packages: session-info
  Building wheel for session-info (setup.py) ... done
  Created wheel for session-info: filename=session_info-1.0.0-py3-none-any.whl size=8048 sha256=321e07f2089(
  Stored in directory: /root/.cache/pip/wheels/bd/ad/14/6a42359351a18337a8683854cfbba99dd782271f2d1767f87f
Successfully built session-info
Installing collected packages: stdlib-list, session-info
Successfully installed session-info-1.0.0 stdlib-list-0.8.0
▶ Click to view session information
```

Productos de pago de Colab - Cancelar contratos

✓ 7 s completado a las 8:12