Pain_Graphs

October 13, 2020

1 Informes y predicciones de COVID-19 en España

Actualizado diariamente, este documento se visualiza mejor aquí.

Indice

[1]:
```
# Cargamos datos
import Loading_data
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings('ignore')
from IPython.display import display, HTML

def Insertar_Enlace(cell_name):
    display(HTML('<a id="'+ cell_name +'"></a>'))
```

```
Insertar_Enlace('DatosMadrid')

COMUNIDAD_A_CONSIDERAR = 'Madrid'
comunidad = Loading_data.Get_Comunidad(COMUNIDAD_A_CONSIDERAR)
comunidad.head(10)
```

```
/root/scripts/COVID-19/jupyter/Loading_data.py:22: FutureWarning: Sorting
because non-concatenation axis is not aligned. A future version
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.

  df = pd.concat([df,this_df])
```

```
<IPython.core.display.HTML object>
```

[1]:

Fecha	Lugar	Casos	Casos hoy absoluto	Casos hoy variacion respecto ayer	Casos hoy porcentaje	Fallecidos	Fallecidos hoy absoluto
2020-06-05	Madrid	69423	187	63	0.002694	8691	0
2020-06-04	Madrid	69236	124	-28	0.001791		
2020-06-03	Madrid	69112	152	44	0.002199		
2020-06-02	Madrid	68960	108	86	0.001566		
2020-06-01	Madrid	68852	22	-68	0.000320		
2020-05-31	Madrid	68830	90	-53	0.001308		
2020-05-30	Madrid	68740	143	-3	0.002080		
2020-05-29	Madrid	68597	146	-39	0.002128		
2020-05-28	Madrid	68451	185	-15	0.002703		
2020-05-27	Madrid	68266	200	66	0.002930		

2020-06-04	8691	0
2020-06-03	8691	0
2020-06-02	8691	0
2020-06-01	8691	0
2020-05-31	8691	0
2020-05-30	8691	0
2020-05-29	8691	0
2020-05-28	8691	0
2020-05-27	8691	0

Fecha	Fallecidos hoy variacion respecto ayer	Fallecidos hoy porcentaje
2020-06-05	0	0.0
2020-06-04	0	0.0
2020-06-03	0	0.0
2020-06-02	0	0.0
2020-06-01	0	0.0
2020-05-31	0	0.0
2020-05-30	0	0.0
2020-05-29	0	0.0
2020-05-28	0	0.0
2020-05-27	-5	0.0

Fecha	Tasa Mortalidad	Curados	Curados hoy absoluto
2020-06-05	0.125189	0	0.0
2020-06-04	0.125527	0	0.0
2020-06-03	0.125752	0	0.0
2020-06-02	0.126030	0	0.0
2020-06-01	0.126227	0	0.0
2020-05-31	0.126268	0	0.0
2020-05-30	0.126433	0	0.0
2020-05-29	0.126697	0	0.0
2020-05-28	0.126967	0	0.0
2020-05-27	0.127311	0	0.0

Fecha	Casos excluidos curados
2020-06-05	69423
2020-06-04	69236
2020-06-03	69112
2020-06-02	68960
2020-06-01	68852
2020-05-31	68830
2020-05-30	68740
2020-05-29	68597
2020-05-28	68451

```
2020-05-27                    68266

            Proporcion Curados hoy absoluto / Casos hoy absoluto   UCI  \
Fecha
2020-06-05                                                0.0     3551
2020-06-04                                                0.0     3550
2020-06-03                                                0.0     3546
2020-06-02                                                0.0     3546
2020-06-01                                                0.0     3544
2020-05-31                                                0.0     3544
2020-05-30                                                0.0     3544
2020-05-29                                                0.0     3544
2020-05-28                                                0.0     3544
2020-05-27                                                0.0     3544

            Hospitalizados
Fecha
2020-06-05           42079
2020-06-04           42068
2020-06-03           42041
2020-06-02           42041
2020-06-01           42017
2020-05-31           42014
2020-05-30           42011
2020-05-29           41993
2020-05-28           41972
2020-05-27           41945
```

[2]:
```python
import pandas as pd


def Get_Dimensions_All_CCAA(Atributos,media_movil ):
    array = []
    dias_a_considerar = 4 if media_movil else 1
    for ca in COMUNIDADES:

        comunidad = Loading_data.Get_Comunidad(ca).head(dias_a_considerar)
        comunidad = comunidad.reset_index() # Resets the index, makes factor a␣
 ↪column
        if media_movil : comunidad = pd.DataFrame(comunidad.mean(axis=0)).T
        temp_dict = {}
        temp_dict['Lugar'] = ca
        for attr in Atributos:
            temp_dict[attr] = comunidad[attr].iloc[0]
        array.append(temp_dict)

    return pd.DataFrame.from_records(array)
```

[3]:
```
import scipy.stats as spstats
from matplotlib import pyplot as plt


def Print_Two_Cordinates_CCAA(df, add_LR=False):
    fig,ax = plt.subplots()
    fig.set_figheight(8)
    fig.set_figwidth(8)
    ax.axhline(y=0, color='blue')
    ax.axvline(x=0, color='blue')

    for k,d in df.groupby('Lugar'):
        ax.scatter(d[df.columns[1]], d[df.columns[2]], label=k)

    plt.legend(bbox_to_anchor=(0, 1), loc='upper left', ncol=1)

    if add_LR:
        slope, intercept, r_value, p_value, std_err = spstats.linregress(df[df.
 ↪columns[1]], df[df.columns[2]])
        plt.plot(df[df.columns[1]], intercept + slope*df[df.columns[1]], 'r',␣
 ↪label='fitted line')
        #.format(round(slope, 2),round(intercept, 2),round(r_value, 2))
        note2add = f"""slope: {slope:12.4f}\nintercept: {intercept:8.2f}\nr2:␣
 ↪{r_value**2:15.4f}"""
        plt.annotate(note2add,xy=(0.7,0.3), xycoords='figure fraction')

    ax.set_xlabel(df.columns[1])
    ax.set_ylabel(df.columns[2])
    ax.set_title(df.columns[1]+ ' VS. ' + df.columns[2])

    return plt
```

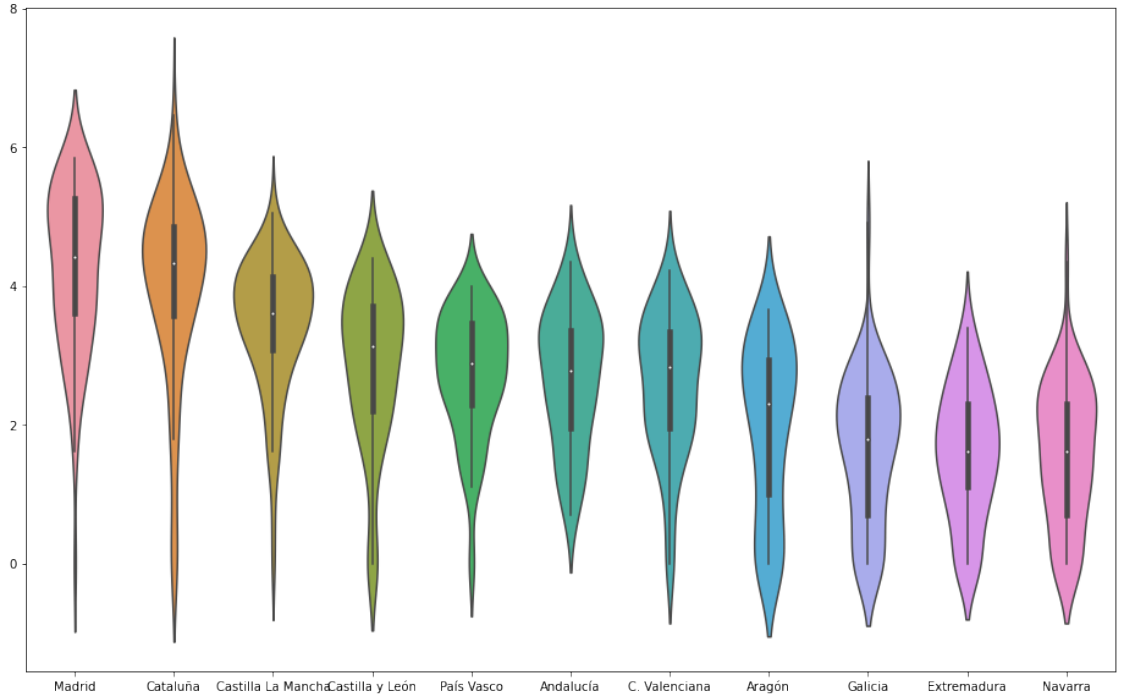
[4]:
```
import numpy as np
import seaborn as sns

from Loading_data import Get_Comunidades_List as comunidades
COMUNIDADES = comunidades()

def Get_Single_Dimension(dimension ):
    df = pd.DataFrame()
    df_tmp = pd.DataFrame()
    array = []
    #
    for ca in COMUNIDADES:
        df_tmp = Loading_data.Get_Comunidad(ca)
        new = df_tmp[[dimension]].copy()
        new.rename(columns={dimension: ca}, inplace=True)
```

```
        array.append(new)
    #
    df =  pd.concat(array, axis=1)
    return df

def plot_violin(dimension):
    """ Muestra la distribucion logaritmica por comunidades, de una dimension"""
    df = Get_Single_Dimension(dimension)
    # Ordenamos comunidades
    s = df.sum()
    df = df[s.sort_values(ascending=False).index[:]]

    # Pasamos a logaritmo
    df2 = np.log(df)
    df2.replace(-np.inf,  np.nan, inplace=True)
    display(HTML("<h2>Comparativa de distribucion de '" +dimension+ "', en cada␣
 ↪CC.AA </h2>"))
    display(HTML("Distribuciones convertidas a logaritmos neperianos, para␣
 ↪facilitar la comparación."))

    # primer grafico
    f, ax = plt.subplots()
    f.set_size_inches( 16, 10)
    f.suptitle("Comunidades con más, " + dimension.lower())
    sns.violinplot(data=df2.iloc[:,:-7])

    #segundo grafico
    f, ax = plt.subplots()
    f.set_size_inches( 16, 10)
    f.suptitle("Comunidades con menos, " + dimension.lower()+".")
    sns.violinplot(data=df2.iloc[:,7:])
    return df

dimension = 'Fallecidos hoy absoluto'
df = plot_violin(dimension)
```

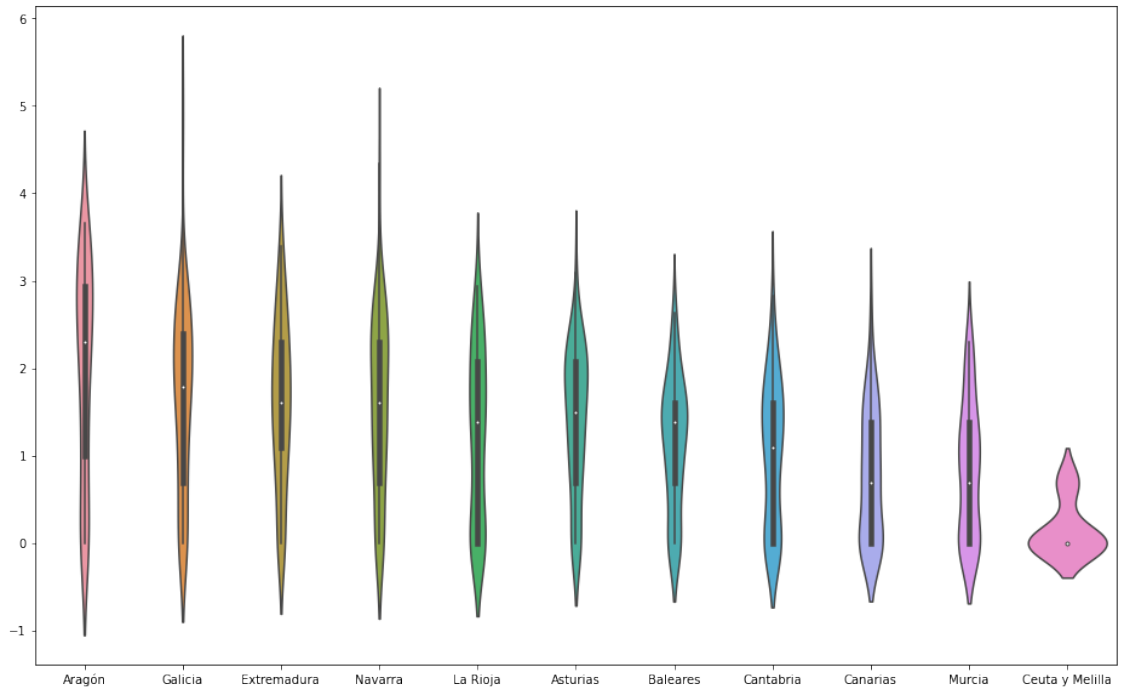
```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

Comunidades con más, fallecidos hoy absoluto



Comunidades con menos, fallecidos hoy absoluto.



[5]:
```
df
```

[5]:

Fecha	Madrid	Cataluña	Castilla La Mancha	Castilla y León	País Vasco
2020-06-05	0	0	0	0	0
2020-06-04	0	0	0	3	0
2020-06-03	0	0	0	1	0
2020-06-02	0	0	0	0	0
2020-06-01	0	0	0	0	0
…	…	…	…	…	…
2020-03-11	10	0	0	0	0
2020-03-10	13	2	0	0	1
2020-03-09	6	1	0	0	5
2020-03-06	1	0	0	0	-1
2020-03-05	0	0	0	0	0

Fecha	Andalucía	C. Valenciana	Aragón	Galicia	Extremadura	Navarra
2020-06-05	0	0	0	0	0	0
2020-06-04	0	0	0	0	0	0
2020-06-03	0	0	0	0	0	0
2020-06-02	0	0	0	0	0	0
2020-06-01	0	0	0	0	0	0
…	…	…	…	…	…	…
2020-03-11	0	0	1	0	0	0
2020-03-10	0	0	2	0	0	0
2020-03-09	0	0	0	0	0	0
2020-03-06	0	0	1	0	0	0
2020-03-05	0	0	0	0	0	0

Fecha	La Rioja	Asturias	Baleares	Cantabria	Canarias	Murcia
2020-06-05	0	1	0	0	0	0
2020-06-04	1	1	0	0	0	0
2020-06-03	0	0	0	0	0	0
2020-06-02	0	0	0	0	0	0
2020-06-01	0	0	0	0	0	0
…	…	…	…	…	…	…
2020-03-11	1	0	0	0	0	0
2020-03-10	1	0	0	0	0	0
2020-03-09	-1	0	0	0	0	0
2020-03-06	1	0	0	0	0	0
2020-03-05	0	0	0	0	0	0

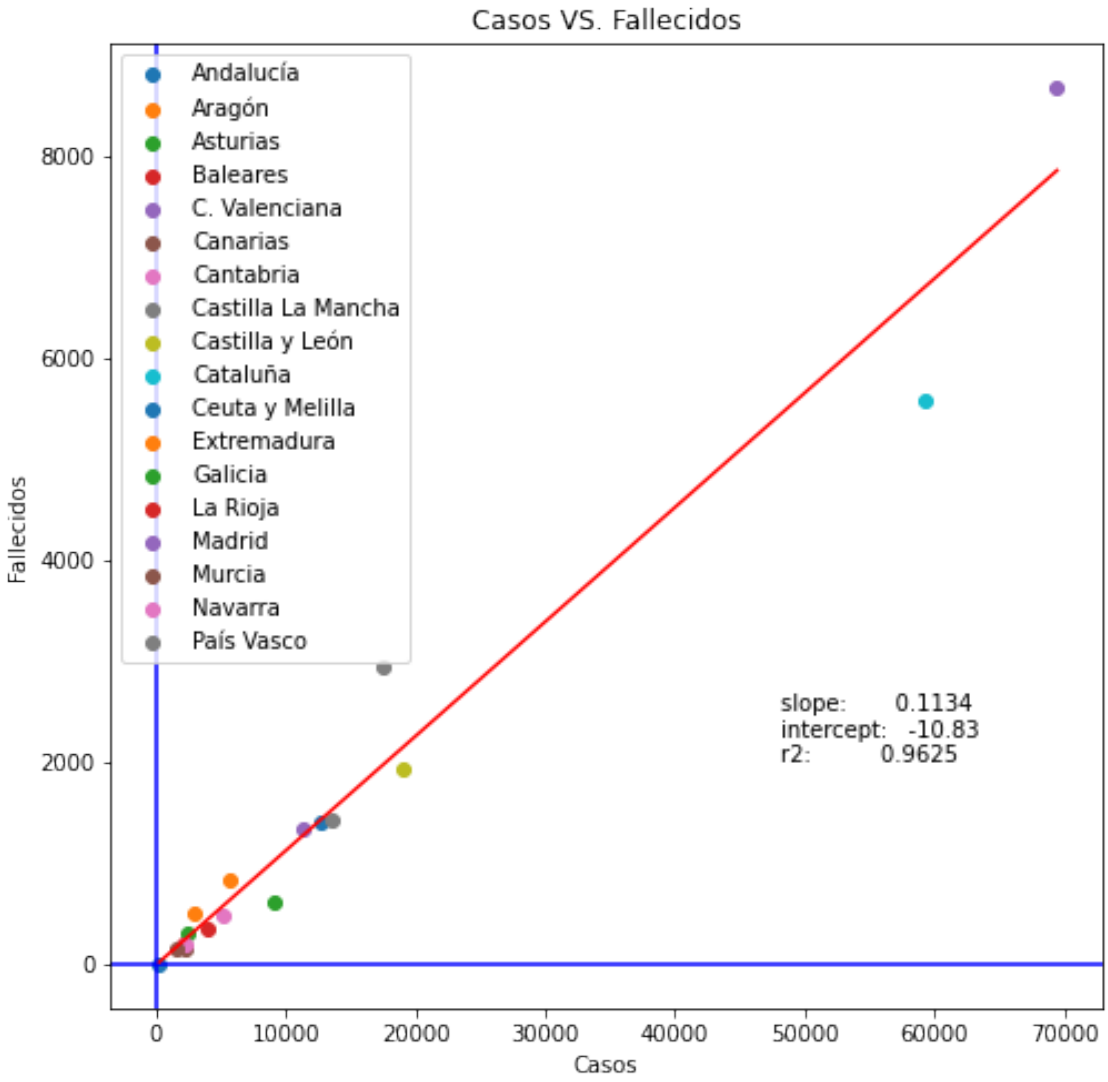
Fecha	Ceuta y Melilla

```
Fecha
2020-06-05                0
2020-06-04                0
2020-06-03                0
2020-06-02                0
2020-06-01                0
…                         …
2020-03-11                0
2020-03-10                0
2020-03-09                0
2020-03-06                0
2020-03-05                0

[89 rows x 18 columns]
```

[6]:
```
import Loading_data

from Loading_data import Get_Comunidades_List as comunidades
COMUNIDADES = comunidades()

def Comparar_Dos_Dimensiones(Atributos, media_movil=False, add_LR=False ):
    """ Compara dos dimensiones de atributos, Ma indica si hacerlo con la media␣
 ↪movil"""

    df = Get_Dimensions_All_CCAA(Atributos,media_movil )
    df = df.sort_values(by=[df.columns[1],df.columns[2]])
    plt = Print_Two_Cordinates_CCAA(df, add_LR)
    plt.show()
    print(  'Total: '  + df.columns[1], df[df.columns[1]].sum() )
    print(  'Total: '  + df.columns[2], df[df.columns[2]].sum() )
    display(HTML(df.set_index('Lugar').to_html(index=True)))
    return
```

[7]:
```
Insertar_Enlace("Comparativa_Casos_Fallecidos")
Comparar_Dos_Dimensiones(['Casos', 'Fallecidos'],add_LR=True )
```

```
<IPython.core.display.HTML object>
```

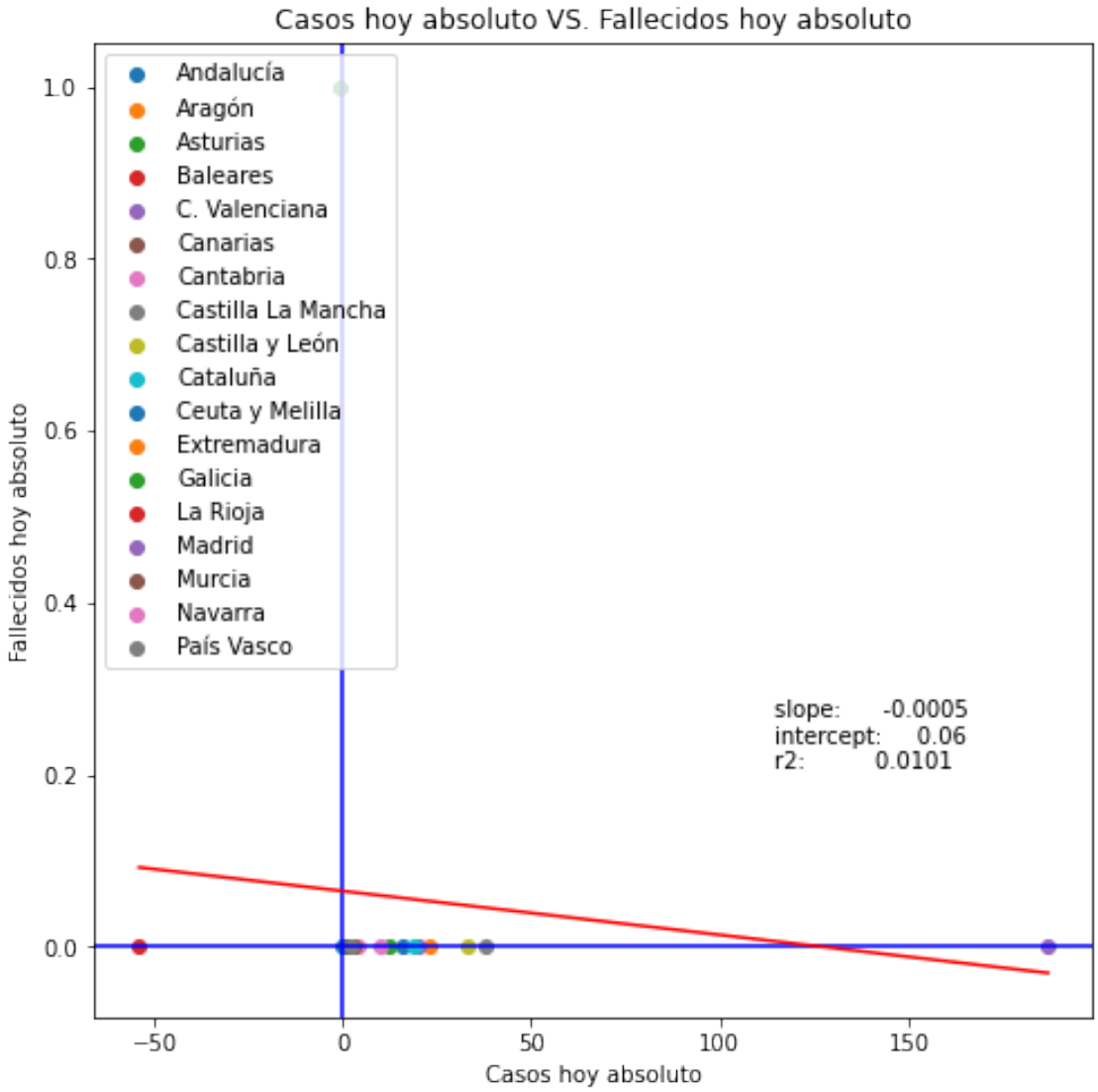



```
Total: Casos 240978
Total: Fallecidos 27134
```

```
<IPython.core.display.HTML object>
```

[8]:
```
Insertar_Enlace("Comparativa_Casos_Fallecidos_Hoy")

Comparar_Dos_Dimensiones(['Casos hoy absoluto', 'Fallecidos hoy␣
 ↪absoluto'],add_LR=True )
```

```
<IPython.core.display.HTML object>
```

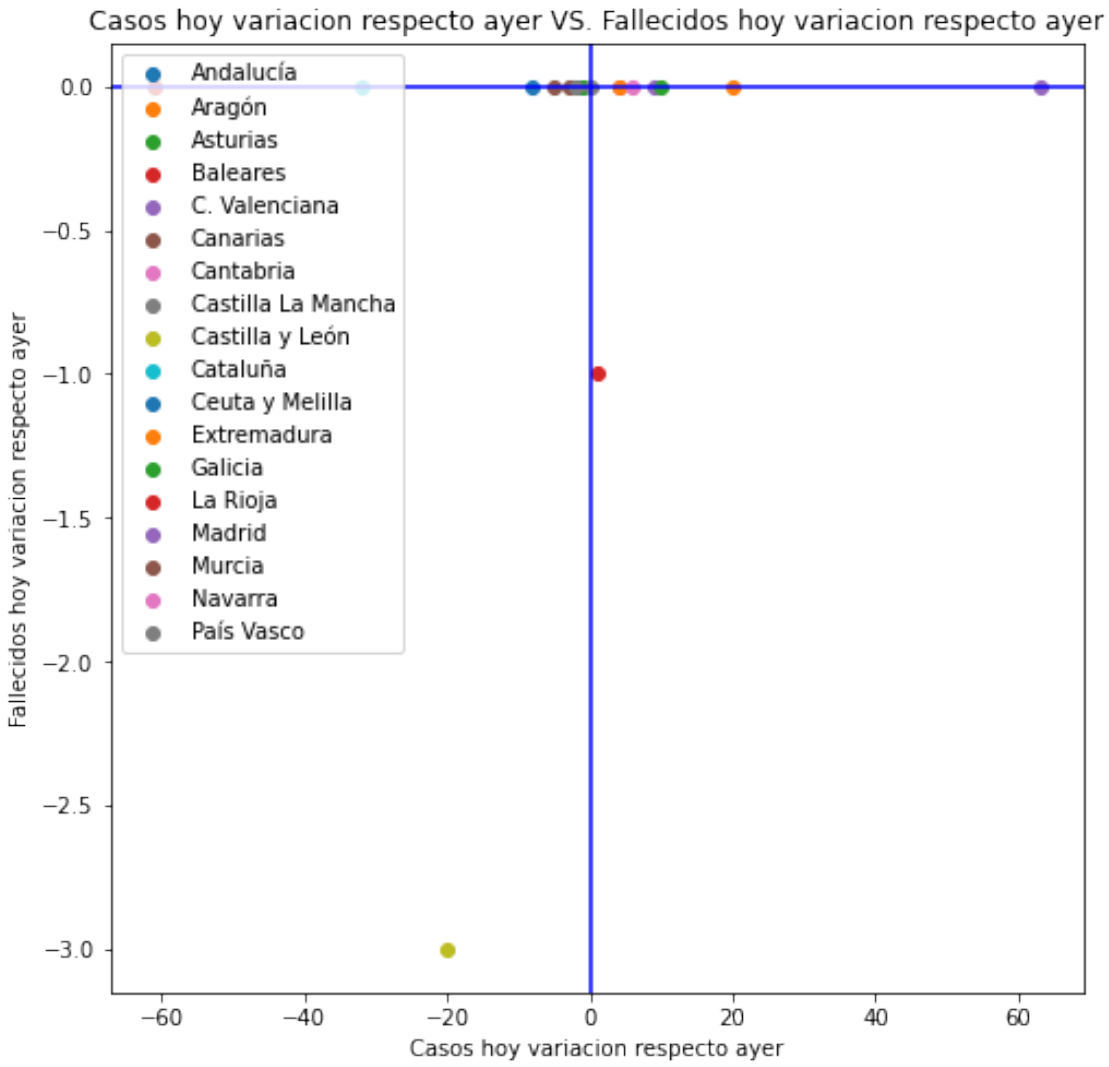



```
Total: Casos hoy absoluto 318
Total: Fallecidos hoy absoluto 1
```

```
<IPython.core.display.HTML object>
```

[9]:
```
Insertar_Enlace("Comparativa_Casos_Fallecidos_Variacion_Diaria")

Comparar_Dos_Dimensiones(['Casos hoy variacion respecto ayer', 'Fallecidos hoy␣
 ↪variacion respecto ayer'])
```

```
<IPython.core.display.HTML object>
```




```
Total: Casos hoy variacion respecto ayer -16
Total: Fallecidos hoy variacion respecto ayer -4
```

```
<IPython.core.display.HTML object>
```

[10]:
```
# Cargamos datos
import pandas as pd
import Loading_data

from Loading_data import Get_Comunidades_List as comunidades
COMUNIDADES = comunidades()

def Get_Dimension_CCAA(Dimension,include_nation=False):
    def Do_Stuff_to_DF(df):
```

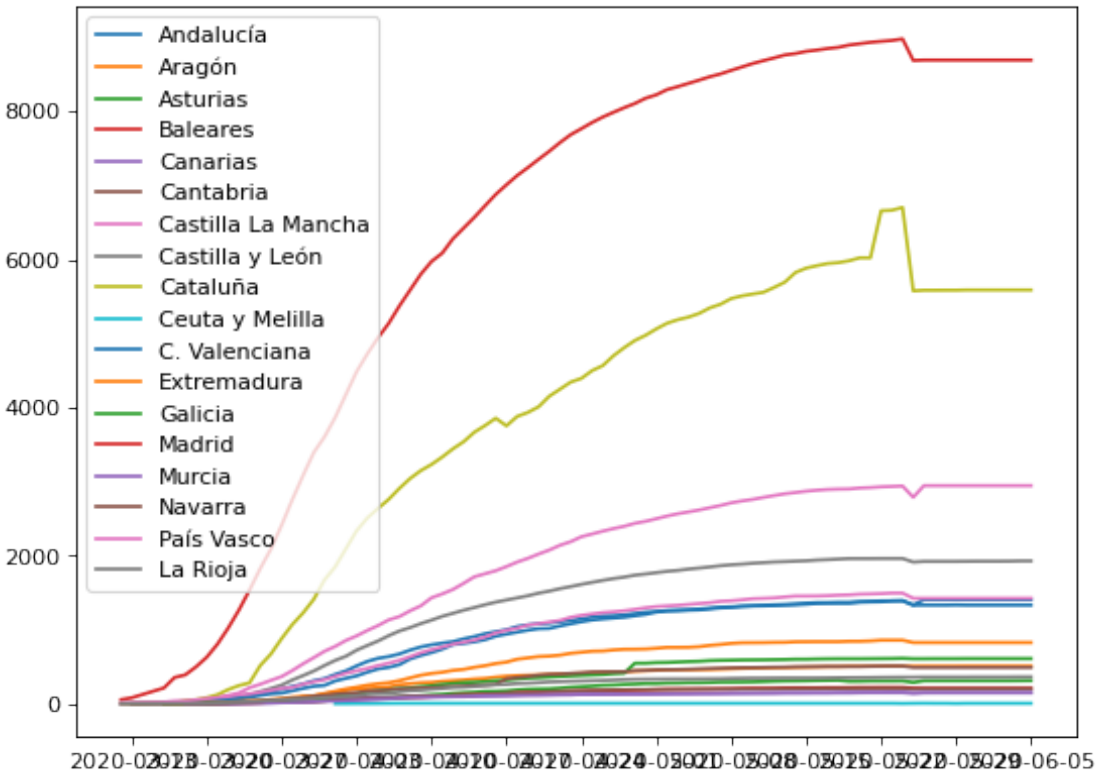
```
        df = df.sort_values(by='Fecha')
        # df = df.reset_index() # Resets the index, makes factor a column
        df = df[df["Casos"] >= 100]
        return df

    dimension_df = pd.DataFrame()
    for ca in COMUNIDADES:
        df = Loading_data.Get_Comunidad(ca)
        df = Do_Stuff_to_DF(df)
        dimension_df[ca] = df[Dimension]
    if include_nation:
        df = Loading_data.Get_Nacion()
        df = Do_Stuff_to_DF(df)
        dimension_df['TOTAL'] = df[Dimension]

    return dimension_df
```

[11]:
```
from matplotlib import pyplot as plt
import matplotlib.dates as mdates
from IPython.display import display, HTML
import pandas as pd

import numpy as np


def compare_charts_median(Dimension,df):
    short_df = df.tail(1)
    short_df = short_df.T
    short_df = short_df.sort_values(by=(short_df.columns[0]))
    short_df.columns = [Dimension]

    #mean_y = short_df.median(axis=1)[0]
    #mean_y= df.tail(1).T.median().values[0]
    median_y= df.tail(1).T.drop(axis=0,labels=(['TOTAL'] if 'TOTAL' in short_df.
 ↪index else [])).median().values[0]
    x = short_df.index
    y = short_df[Dimension]

    plt.figure(figsize = (10, 5))
    plt.scatter(x, y, c= "red", alpha = 0.5)
    plt.title(Dimension + " by region")
    color = 'blue'
    plt.xticks(rotation=90)
    plt.axhline(median_y, c = color, alpha = 0.5, lw = 1)
    plt.annotate('Median ' + Dimension+  ' is {}'.format(round(median_y, 2)),
            xy=(8.5, median_y),
            xycoords='data',
```

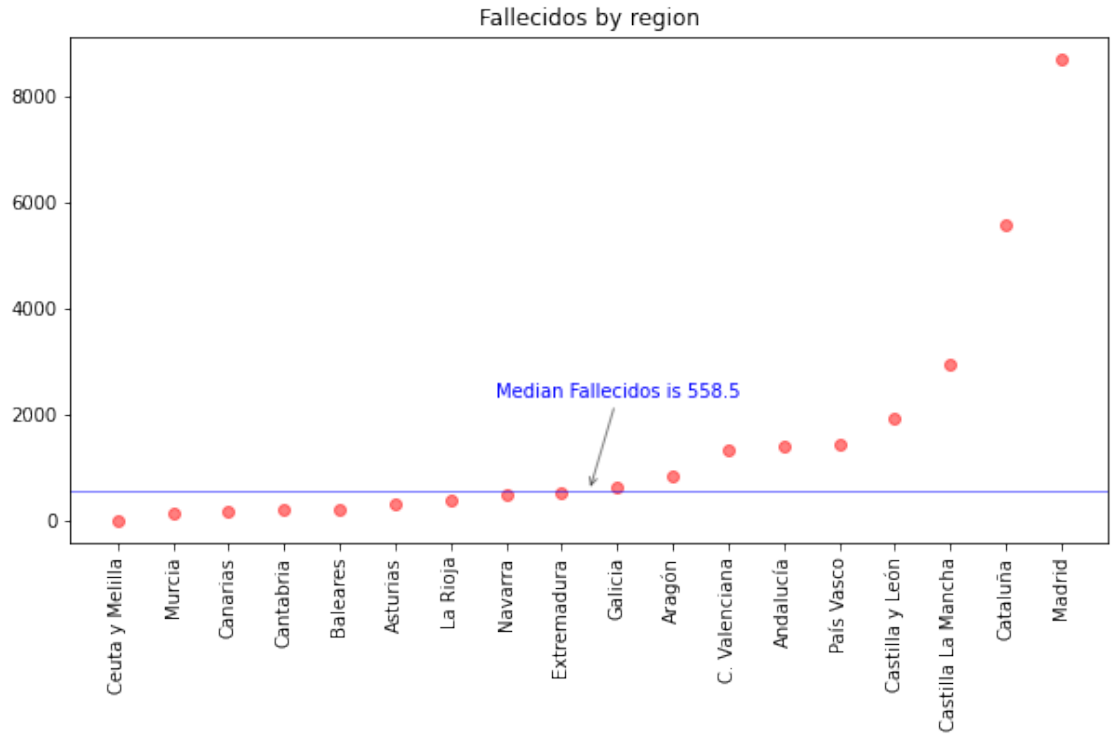
```
            xytext=(-50, 50),
            textcoords='offset points',
            arrowprops=dict(arrowstyle="->", color = "k", alpha = 0.5),
            color = color)
    return

def compare_charts_time(Dimension,df):
    fig = plt.figure(figsize=(8, 6), dpi=80)
    for ca in df.columns:
        plt.plot(df[ca])
    plt.legend(df.columns)

    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m-%d'))
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=7))

    fig.suptitle('Comparativa de: '+Dimension, fontsize=20)
    plt.show()
    return

def Report_Location(Dimension,include_nation=False):
    # Ger Data
    df = Get_Dimension_CCAA(Dimension,include_nation)
    # Compare chart
    compare_charts_time(Dimension,df)
    # Compare median chart
    compare_charts_median(Dimension,df)

    with pd.option_context("display.max_rows", 1000):
        display(HTML(df.to_html()))
    return
```

[12]:
```
Insertar_Enlace("Comunidades_Fallecidos")
Report_Location("Fallecidos")
```

```
<IPython.core.display.HTML object>
```



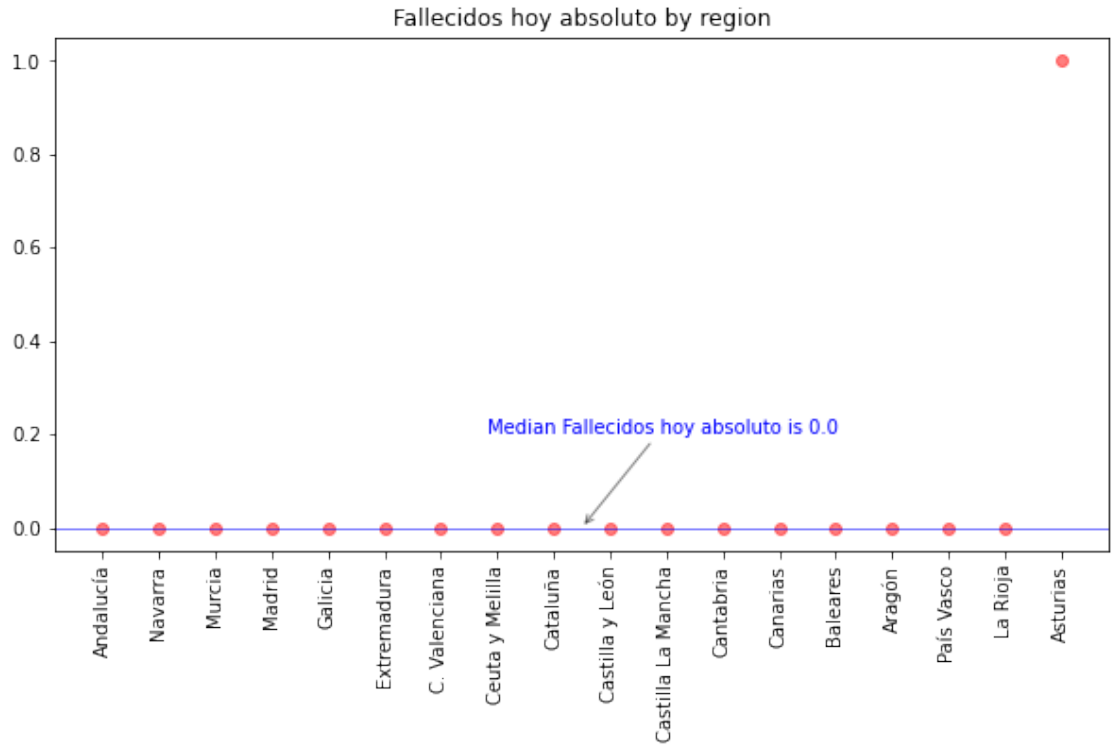
<IPython.core.display.HTML object>



[13]:
```
Insertar_Enlace("Comunidades_Fallecidos_Hoy")

Report_Location("Fallecidos hoy absoluto")
```

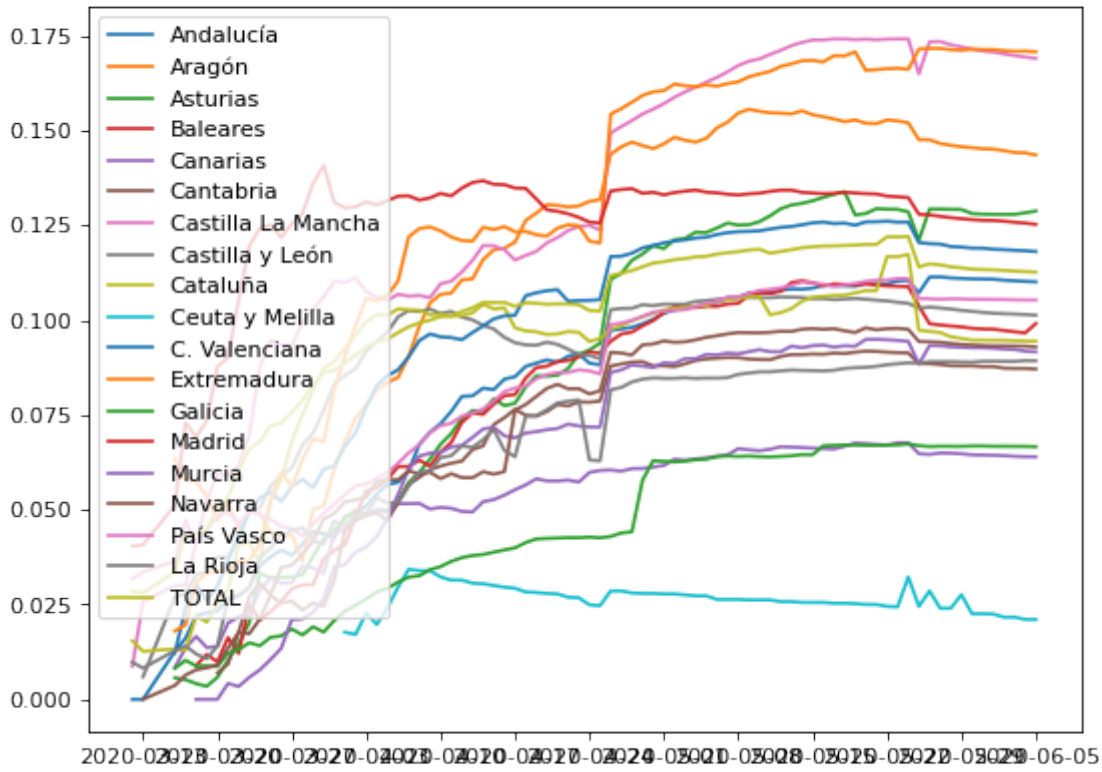
```
<IPython.core.display.HTML object>
```


```
<IPython.core.display.HTML object>
```

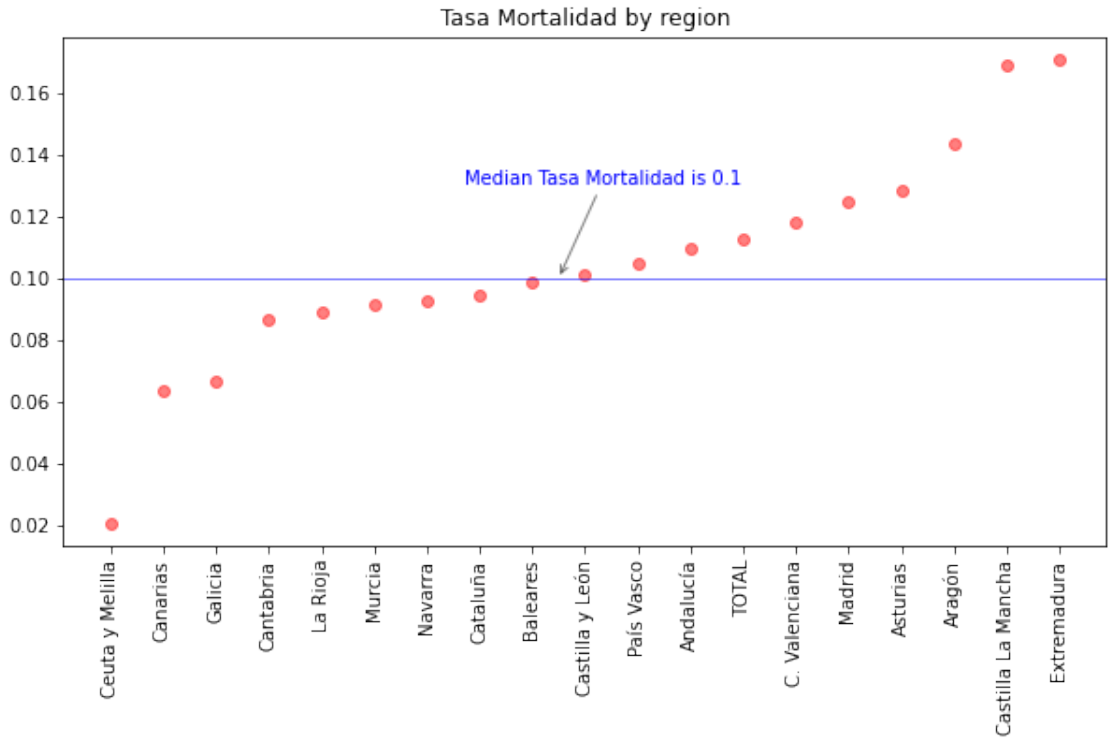


[14]:
```
Insertar_Enlace("Comunidades_Mortalidad")

Report_Location("Tasa Mortalidad",True)
```

```
<IPython.core.display.HTML object>
```

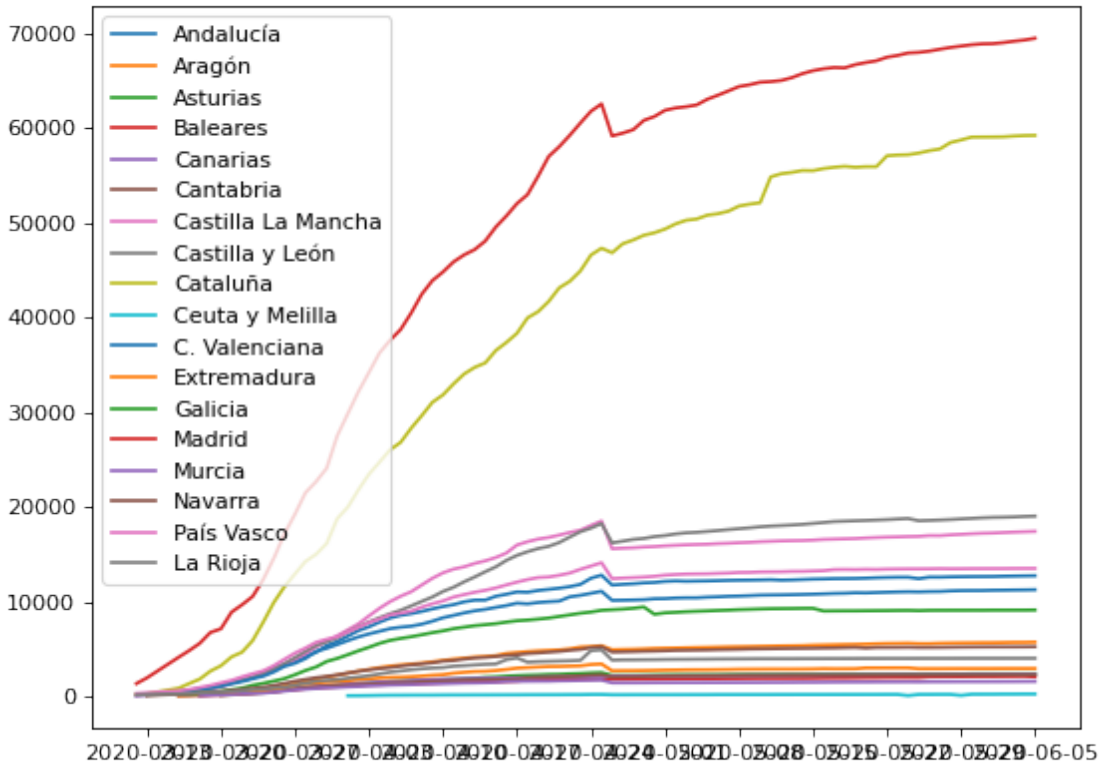



```
<IPython.core.display.HTML object>
```

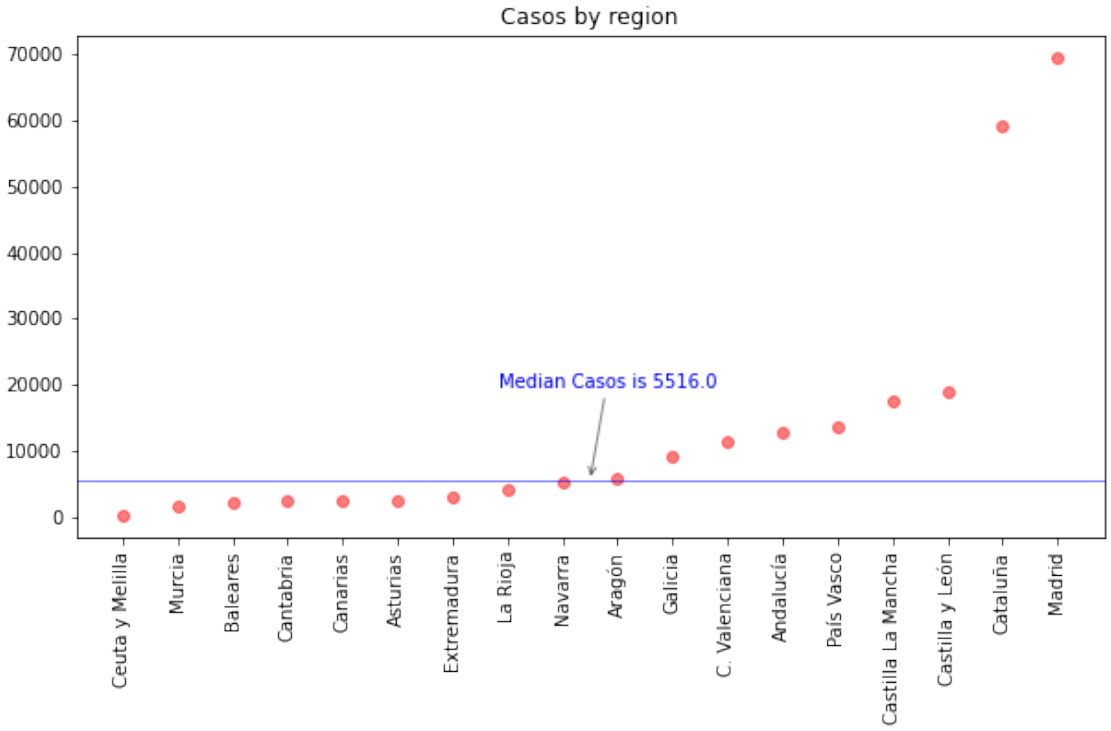


[15]:
```
Insertar_Enlace("Comunidades_Casos")

Report_Location("Casos")
```

```
<IPython.core.display.HTML object>
```

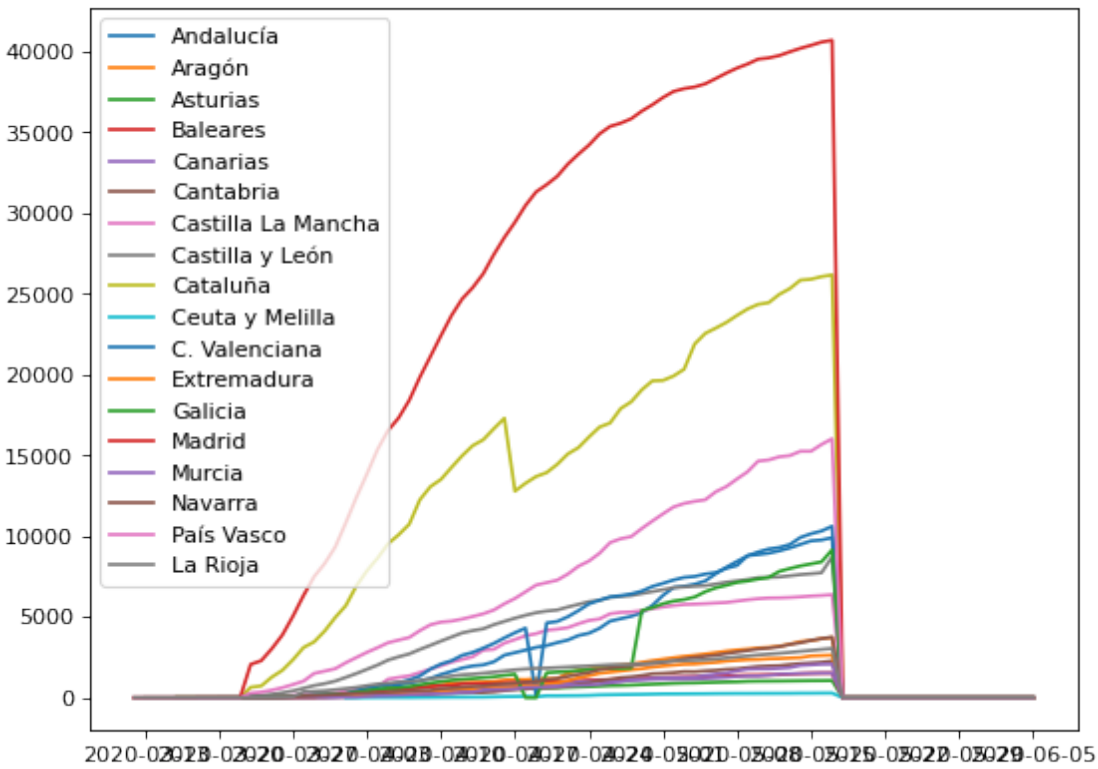


<IPython.core.display.HTML object>

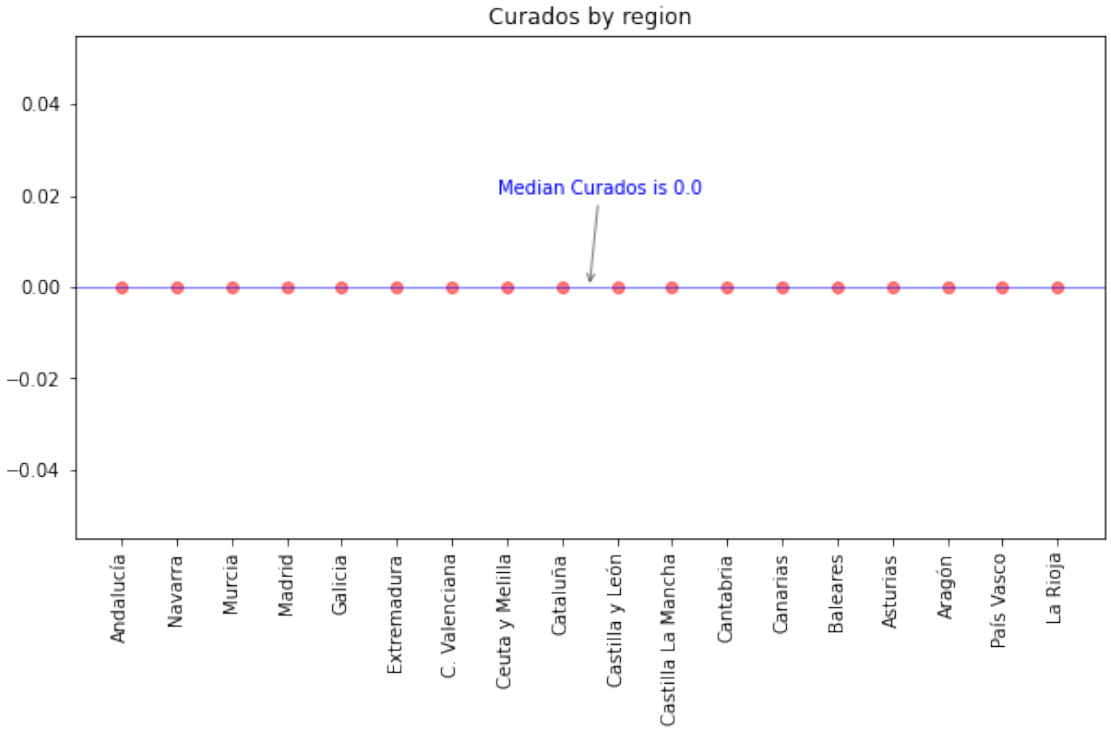


[16]:
```
Insertar_Enlace("Curados")

Report_Location("Curados")
```

```
<IPython.core.display.HTML object>
```



<IPython.core.display.HTML object>



1.4 ¿Hemos alcanzado el pico de la curva?

1.4.1 Casos totales españa, evolucion

[17]:
```
# Casos totales españa, evolucion
import Loading_data

MOVING_AVERAGE_WINDOW=4

def get_fallecidos_nacion(window_size=MOVING_AVERAGE_WINDOW):
    Dimension = 'Fallecidos'

    Insertar_Enlace("Pico_España")

    df = Get_Dimension_CCAA(Dimension)

    df['Total Fallecidos'] = df.sum(axis=1)
    df['Total Fallecidos']

    CONVERT_INT_COLUMNS = ['Total Fallecidos']
    for column in CONVERT_INT_COLUMNS :
        df[column] = df[column].fillna(0)
        df[column] = df[column].astype(np.int64)
```

```
    df['Total Fallecidos hoy absoluto'] = df['Total Fallecidos'] - df['Total␣
 ↪Fallecidos'].shift(1)
    df['MA Total Fallecidos hoy absoluto'] = df['Total Fallecidos hoy␣
 ↪absoluto'].rolling(window=window_size).mean()

    df['Variacion MA Total Fallecidos hoy absoluto'] = df['MA Total Fallecidos␣
 ↪hoy absoluto'] - df['MA Total Fallecidos hoy absoluto'].shift(1)
    return df

df = get_fallecidos_nacion()
df_plt = df[['Total Fallecidos hoy absoluto','MA Total Fallecidos hoy␣
 ↪absoluto']]
fig = plt.figure(figsize=(8, 6), dpi=80)
plt.plot(df_plt, marker='o')
plt.xticks(rotation=90)
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m-%d'))
plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=7))
plt.legend(df_plt.columns)
fig.suptitle( "Total fallecidos en España", fontsize=20)

df[['Total Fallecidos',
    'Total Fallecidos hoy absoluto',
    'MA Total Fallecidos hoy absoluto',
    'Variacion MA Total Fallecidos hoy absoluto']]
```

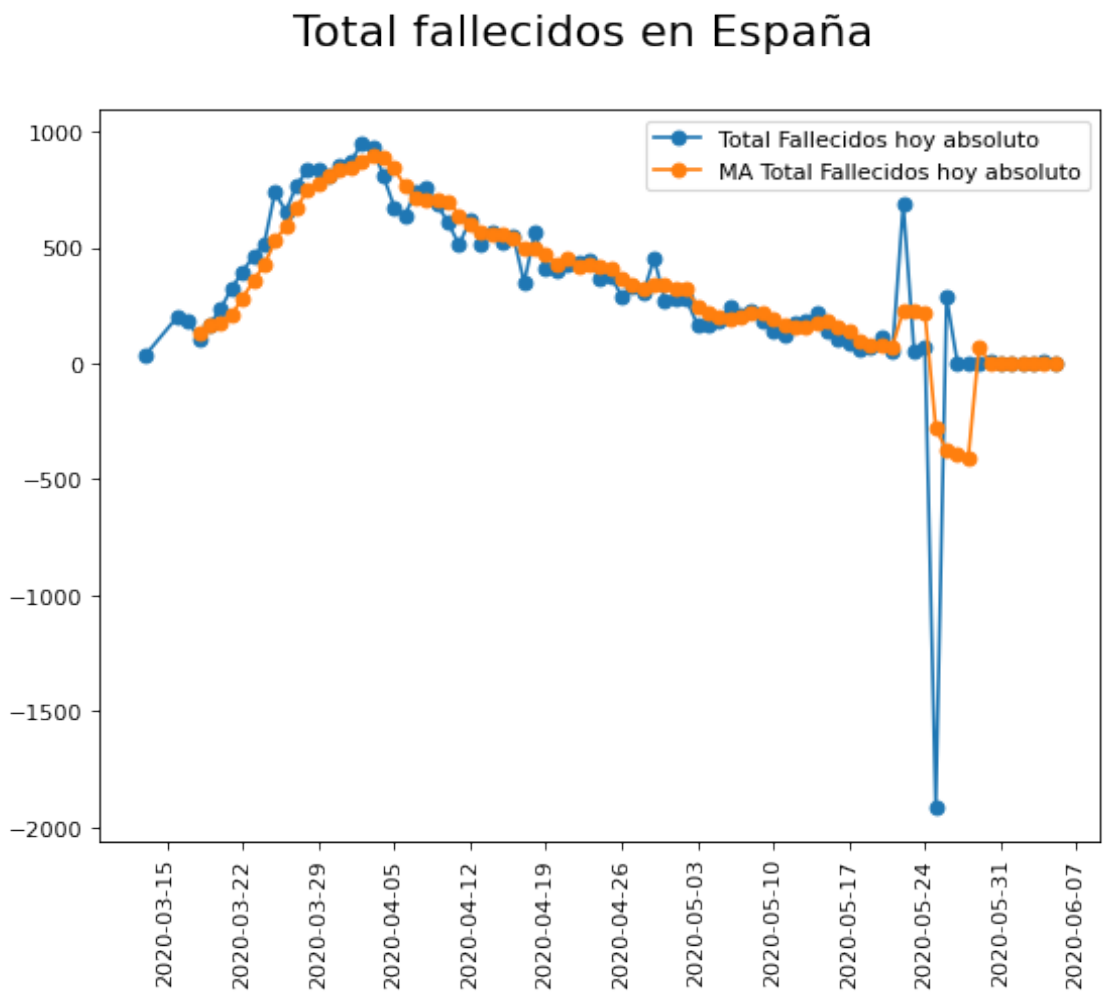
```
<IPython.core.display.HTML object>
```

[17]:

Fecha	Total Fallecidos	Total Fallecidos hoy absoluto	\
2020-03-12	74	NaN	
2020-03-13	107	33.0	
2020-03-16	308	201.0	
2020-03-17	490	182.0	
2020-03-18	597	107.0	
…	…	…	
2020-06-01	27127	0.0	
2020-06-02	27127	0.0	
2020-06-03	27128	1.0	
2020-06-04	27133	5.0	
2020-06-05	27134	1.0	

Fecha	MA Total Fallecidos hoy absoluto	\
2020-03-12	NaN	
2020-03-13	NaN	
2020-03-16	NaN	

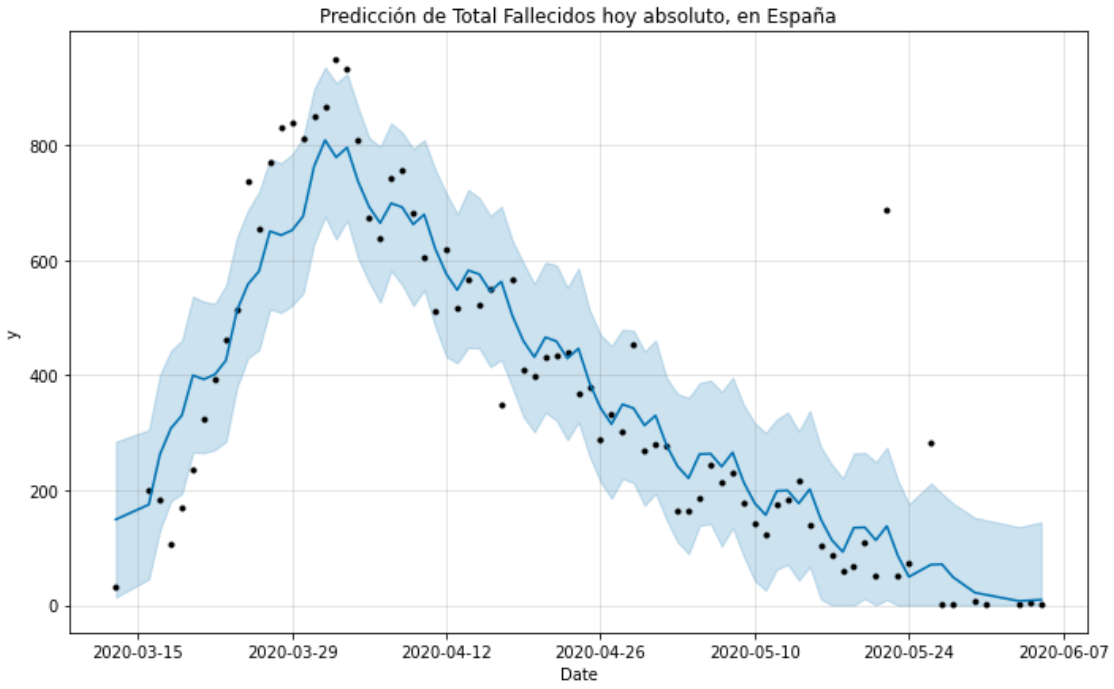
```
2020-03-17                                NaN
2020-03-18                             130.75
…                                         …
2020-06-01                               2.00
2020-06-02                               2.00
2020-06-03                               0.75
2020-06-04                               1.50
2020-06-05                               1.75

            Variacion MA Total Fallecidos hoy absoluto
Fecha
2020-03-12                                          NaN
2020-03-13                                          NaN
2020-03-16                                          NaN
2020-03-17                                          NaN
2020-03-18                                          NaN
…                                                   …
2020-06-01                                        -0.25
2020-06-02                                         0.00
2020-06-03                                        -1.25
2020-06-04                                         0.75
2020-06-05                                         0.25

[84 rows x 4 columns]
```



[18]:
```
import fbprophet
def Get_Prediction_Nacion(df,dimension,location='España' , link=None) :
    df = df[[dimension]]
    df = df[df[dimension] > 0]

    df = df.dropna()
    df = df.reset_index()
    df.columns = ['ds','y']


    df_prophet = fbprophet.Prophet(changepoint_prior_scale=0.15)
    df_prophet.fit(df)

    # Make a future dataframe for 2 years
    df_forecast = df_prophet.make_future_dataframe(periods=45, freq='D')
    # Make predictions
```

```
    df_forecast = df_prophet.predict(df_forecast)
    df_forecast

    df_forecast = df_forecast[df_forecast["yhat"] >= 0]
    df_forecast.loc[df_forecast.yhat_lower < 0, 'yhat_lower'] = 0

    if link is not None:
        Insertar_Enlace(link)

    df_prophet.plot(df_forecast, xlabel = 'Date' )
    plt.title('Predicción de ' + dimension + ", en " + location )

    suma = df_forecast.trend.sum()
    display(HTML(pd.DataFrame(df_forecast).to_html()))

    print ("Prediccion total para " + dimension + " : " + str(suma) )
    return df_forecast


Insertar_Enlace("Prediccion_Fallecidos_España")

prediccion = Get_Prediction_Nacion( df = get_fallecidos_nacion(),
                                    dimension = 'Total Fallecidos hoy absoluto')
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
<IPython.core.display.HTML object>


Prediccion total para Total Fallecidos hoy absoluto : 28874.763016918456
```



[19]:
```
def Get_Predictions_Compare(df,dimension,location='España' , link=None) :
    if link is not None:
        Insertar_Enlace(link)


    display(HTML("<h2>Comparativa de predicciones, hoy contra días pasados, "␣
↪+dimension+ " en " +  location+ "</h2>"))


    df = df[[dimension]]
    df = df[df[dimension] > 0]

    df = df.dropna()
    df = df.reset_index()

    df.columns = ['ds','y']

    df_original = df.copy()


    results = pd.DataFrame()
    array_results_temp = []
    for i in range(4):

        if i >= 1 : df = df.iloc[1:]
```

```
        fecha=max(df.ds)
        fecha_short = str(fecha)[:10]


        df_prophet = fbprophet.Prophet(changepoint_prior_scale=0.15)
        df_prophet.fit(df)

        # Make a future dataframe
        df_forecast = df_prophet.make_future_dataframe(periods=45, freq='D')
        # Make predictions
        df_forecast = df_prophet.predict(df_forecast)
        suma = df_forecast.yhat.sum()

        title_column = "Predicción con los datos de " + fecha_short
        df_forecast[title_column] = df_forecast.yhat

        array_results_temp.append(pd.
↪DataFrame(df_forecast[['ds',title_column]]))


    df_1 = pd.merge(array_results_temp[0], array_results_temp[1], how ='outer',␣
↪on ='ds')
    df_2 = pd.merge(df_1                , array_results_temp[2], how ='outer',␣
↪on ='ds')
    df_3 = pd.merge(df_2                , array_results_temp[3], how ='outer',␣
↪on ='ds')
    df_4 = pd.merge(df_3                , df_original          , how ='outer',␣
↪on ='ds')
    df_4['datos reales'] = df_4['y']
    del df_4['y']


    df_chart = df_4
    df_chart = df_chart.set_index('ds')
    df_chart = df_chart.head(70).tail(40)

    for c in df_chart.columns:
        df_chart.loc[df_chart[c] < 0, c] = 0

    df_chart.drop(df_chart.loc[df_chart.sum(axis=1)==0].index, inplace=True)
    df_chart.drop(columns=df_chart.columns[df_chart.sum()==0], inplace=True)


    fig = plt.figure(figsize=(8, 6), dpi=80)
    plt.plot(df_chart)
```

```
    plt.title("Predicciones en días anteriores Vs. Datos reales" + dimension␣
↪+", en " + location )
    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m-%d'))
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=7))
    plt.xticks(rotation=90)
    plt.legend(df_chart.columns)



    #print(df_chart)

    print("Las predicciones del total de "+ dimension+  " en " +  location+  ",␣
↪cambian dia a dia")
    print(df_chart.sum(axis=0) )

    return df_chart
```

[20]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'España'

link="Prediccion_Fallecidos_hoy_absoluto_España"


df = get_fallecidos_nacion()[['Total Fallecidos hoy absoluto']]
df.columns = [ 'Fallecidos hoy absoluto' ]
df.sort_index(inplace=True,ascending=False)

prediccion = Get_Predictions_Compare( df = df,
                                     dimension = dimension,
                                     link = link,
                                     location  = COMUNIDAD_A_CONSIDERAR
                                     )
prediccion
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
Las predicciones del total de Fallecidos hoy absoluto en España, cambian dia a
dia
Predicción con los datos de 2020-06-05    11897.539567
Predicción con los datos de 2020-06-04    11836.116703
Predicción con los datos de 2020-06-03    11843.586486
Predicción con los datos de 2020-05-31    11811.418008
datos reales                              11189.000000
dtype: float64
```

[20]:

ds	Predicción con los datos de 2020-06-05 \
2020-04-14	582.224242
2020-04-15	575.448560
2020-04-16	545.580085
2020-04-17	562.757646
2020-04-18	503.446611
2020-04-19	459.503110
2020-04-20	431.318357
2020-04-21	465.684783
2020-04-22	458.910934
2020-04-23	429.044292
2020-04-24	446.223685
2020-04-25	386.923228
2020-04-26	342.990304
2020-04-27	314.817569
2020-04-28	349.196015
2020-04-29	342.432697
2020-04-30	312.585675
2020-05-01	329.784688
2020-05-02	277.948727
2020-05-03	241.480299
2020-05-04	220.770275
2020-05-05	262.621938
2020-05-06	263.331837
2020-05-07	240.958472
2020-05-08	265.631141
2020-05-09	213.815216

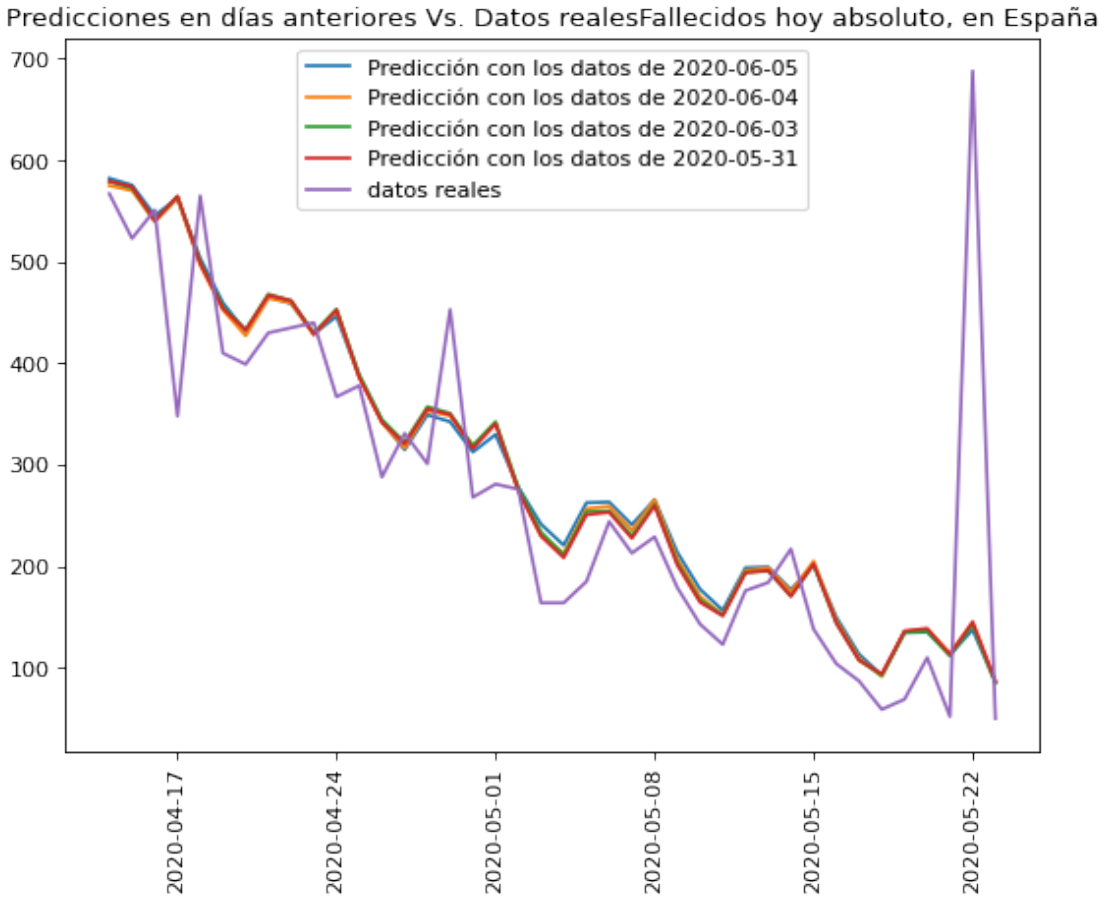
ds	Predicción con los datos de 2020-06-04
2020-04-14	575.004341
2020-04-15	570.120000
2020-04-16	539.401778
2020-04-17	562.343466
2020-04-18	496.797175
2020-04-19	452.574676
2020-04-20	427.164464
2020-04-21	464.088056
2020-04-22	459.203951
2020-04-23	428.485965
2020-04-24	451.432332
2020-04-25	385.890720
2020-04-26	341.675600
2020-04-27	316.272767
2020-04-28	353.203738
2020-04-29	348.343068
2020-04-30	317.648517
2020-05-01	340.618170
2020-05-02	275.099844
2020-05-03	230.905310
2020-05-04	212.662854
2020-05-05	256.754203
2020-05-06	259.048534
2020-05-07	235.508984
2020-05-08	265.629344
2020-05-09	207.273013
2020-05-10	170.240473
2020-05-11	152.021481
2020-05-12	196.136293
2020-05-13	198.443172
2020-05-10	177.371411
2020-05-11	156.686009
2020-05-12	198.553152
2020-05-13	199.278531
2020-05-14	176.911118
2020-05-15	201.589742
2020-05-16	149.779772
2020-05-17	113.337335
2020-05-18	92.653301
2020-05-19	134.520446
2020-05-20	135.245829
2020-05-21	112.878418
2020-05-22	137.557043
2020-05-23	85.747072

```
2020-05-14                                174.916324
2020-05-15                                205.049387
2020-05-16                                146.694476
2020-05-17                                109.663357
2020-05-18                                 91.444525
2020-05-19                                135.559498
2020-05-20                                137.866537
2020-05-21                                114.339695
2020-05-22                                144.472763
2020-05-23                                 86.117852

            Predicción con los datos de 2020-06-03  \
ds
2020-04-14                                579.060350
2020-04-15                                572.508599
2020-04-16                                540.870451
2020-04-17                                564.296696
2020-04-18                                499.554638
2020-04-19                                455.441574
2020-04-20                                433.050117
2020-04-21                                468.130640
2020-04-22                                461.579055
2020-04-23                                429.941111
2020-04-24                                453.367561
2020-04-25                                388.625707
2020-04-26                                344.514606
2020-04-27                                322.125112
2020-04-28                                357.209912
2020-04-29                                350.662604
2020-04-30                                319.028898
2020-05-01                                342.460736
2020-05-02                                277.724270
2020-05-03                                233.627012
2020-05-04                                211.251361
2020-05-05                                253.696302
2020-05-06                                254.509135
2020-05-07                                230.235569
2020-05-08                                261.033245
2020-05-09                                203.662618
2020-05-10                                166.922332
2020-05-11                                151.903652
2020-05-12                                194.356911
2020-05-13                                195.178065
2020-05-14                                170.912821
2020-05-15                                201.711971
2020-05-16                                144.342818
2020-05-17                                107.602536
```

2020-05-18	92.583859
2020-05-19	135.037122
2020-05-20	135.858276
2020-05-21	111.593033
2020-05-22	142.392183
2020-05-23	85.023030

ds	Predicción con los datos de 2020-05-31	datos reales
2020-04-14	579.305102	567.0
2020-04-15	573.944162	523.0
2020-04-16	540.465098	551.0
2020-04-17	564.653543	348.0
2020-04-18	498.416012	565.0
2020-04-19	453.937000	410.0
2020-04-20	432.659435	399.0
2020-04-21	467.129999	430.0
2020-04-22	461.769061	435.0
2020-04-23	428.291747	440.0
2020-04-24	452.481941	367.0
2020-04-25	386.252072	378.0
2020-04-26	341.780721	288.0
2020-04-27	320.512797	331.0
2020-04-28	354.993003	301.0
2020-04-29	349.641707	453.0
2020-04-30	316.174098	268.0
2020-05-01	340.373998	281.0
2020-05-02	274.149339	276.0
2020-05-03	229.683200	164.0
2020-05-04	208.418507	164.0
2020-05-05	250.735583	185.0
2020-05-06	253.221157	244.0
2020-05-07	227.590815	213.0
2020-05-08	259.627982	229.0
2020-05-09	201.240228	179.0
2020-05-10	164.610992	143.0
2020-05-11	151.183204	123.0
2020-05-12	193.508572	176.0
2020-05-13	196.002439	184.0
2020-05-14	170.378689	217.0
2020-05-15	202.422449	138.0
2020-05-16	144.040230	104.0
2020-05-17	107.416531	87.0
2020-05-18	93.994278	59.0
2020-05-19	136.320154	69.0
2020-05-20	138.814528	110.0
2020-05-21	113.190779	52.0

2020-05-22	145.234538	688.0
2020-05-23	86.852320	50.0



[21]:
```
MOVING_AVERAGE_WINDOW = 4
def␣
↪report_single_location_single_dimension(location,dimension,window_size=MOVING_AVERAGE_WINDO
↪


    Dimension = 'Fallecidos'
    labelMa = f'Moving Average ({window_size}) {dimension}'

    df = pd.DataFrame()
    df[dimension] = Get_Dimension_CCAA(dimension)[location]
    df[labelMa] = df[dimension].rolling(window=window_size).mean()

    display(HTML("<h2>Analisis de ''" + dimension +"'', en " + location + "</
↪h2>"))
    fig = plt.figure(figsize=(8, 6), dpi=80)
```

```
    plt.plot(df, marker='o')
    plt.title("Gráfico de " + dimension +", en " + location)
    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m-%d'))
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=7))


    plt.legend(df.columns)
    fig.suptitle( dimension + ' in ' + location, fontsize=20)

    display(HTML(pd.DataFrame(df).to_html()))
    return
```

1.4.2 Casos totales españa, evolucion

[22]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Madrid'

Insertar_Enlace("Reporte_Fallecidos_hoy_absoluto_Madrid")

report_single_location_single_dimension(COMUNIDAD_A_CONSIDERAR,dimension,4)
```

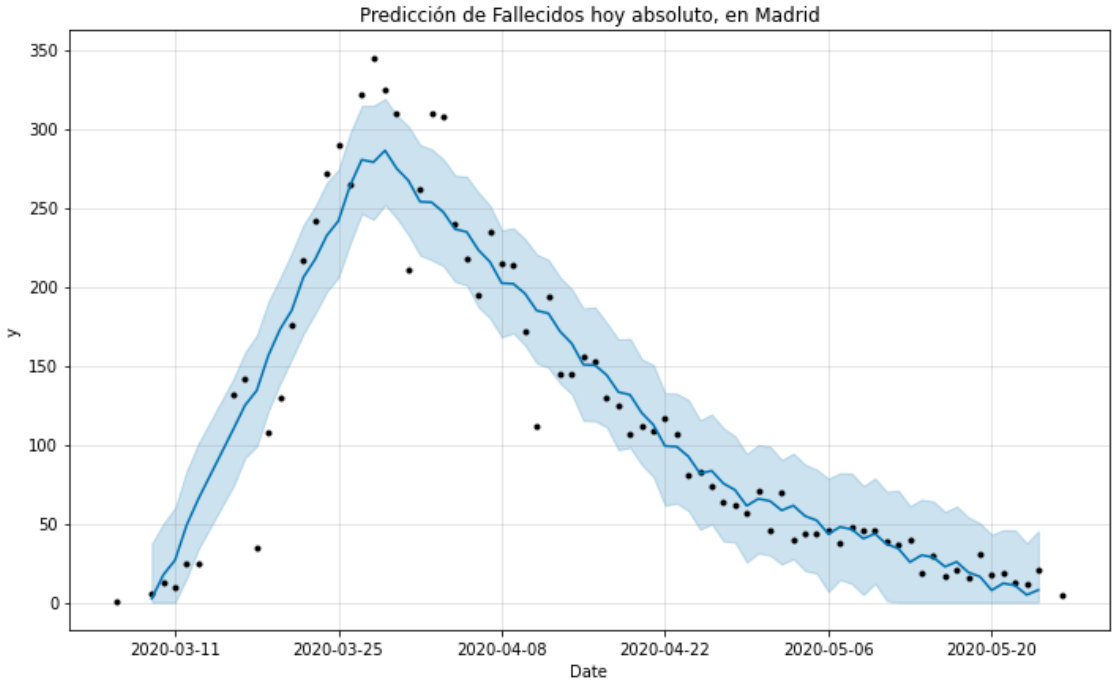
```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

[23]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Madrid'

link="Prediccion_Fallecidos_hoy_absoluto_Madrid"

prediccion = Get_Prediction_Nacion( df = Loading_data.
 ↪Get_Comunidad(COMUNIDAD_A_CONSIDERAR),
                                    dimension = dimension,
                                    link = link,
                                    location  = COMUNIDAD_A_CONSIDERAR
                                    )
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>


Prediccion total para Fallecidos hoy absoluto : 9027.314200402268
```



[24]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Madrid'
link="Prediccion_Compare_Fallecidos_hoy_absoluto_Madrid"

df = Loading_data.Get_Comunidad(COMUNIDAD_A_CONSIDERAR)
prediccion = Get_Predictions_Compare( df = df,
                                      dimension = dimension,
                                      link = link,
                                      location  = COMUNIDAD_A_CONSIDERAR
                                      )
prediccion
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
Las predicciones del total de Fallecidos hoy absoluto en Madrid, cambian dia a
dia
Predicción con los datos de 2020-05-26    3505.921985
Predicción con los datos de 2020-05-24    3504.067771
Predicción con los datos de 2020-05-23    3496.654179
Predicción con los datos de 2020-05-22    3495.348031
datos reales                              3277.000000
dtype: float64
```

[24]:

ds	Predicción con los datos de 2020-05-26 \
2020-04-09	201.790011
2020-04-10	195.552844
2020-04-11	184.815203
2020-04-12	183.056038
2020-04-13	171.619473
2020-04-14	164.030593
2020-04-15	150.616153
2020-04-16	150.216847
2020-04-17	143.980997
2020-04-18	133.245770
2020-04-19	131.489019
2020-04-20	120.054868
2020-04-21	112.468940
2020-04-22	99.057452
2020-04-23	98.660200
2020-04-24	92.426405
2020-04-25	81.692136
2020-04-26	83.362048
2020-04-27	75.354559
2020-04-28	71.196082
2020-04-29	61.212046
2020-04-30	65.628350
2020-05-01	64.208111
2020-05-02	58.287399

2020-05-03	61.347169
2020-05-04	54.729539
2020-05-05	51.959913
2020-05-06	43.364726
2020-05-07	47.783664
2020-05-08	46.366766
2020-05-09	40.449395
2020-05-10	43.510499
2020-05-11	36.894203
2020-05-12	34.125285
2020-05-13	25.530807
2020-05-14	29.950453
2020-05-15	28.533555
2020-05-16	22.616184
2020-05-17	25.677288
2020-05-18	19.060992

ds	Predicción con los datos de 2020-05-24 \
2020-04-09	201.871118
2020-04-10	195.672429
2020-04-11	184.952436
2020-04-12	183.572375
2020-04-13	171.913139
2020-04-14	163.628440
2020-04-15	150.707485
2020-04-16	150.331574
2020-04-17	144.133303
2020-04-18	133.417318
2020-04-19	132.041263
2020-04-20	120.389098
2020-04-21	112.111471
2020-04-22	99.199919
2020-04-23	98.833411
2020-04-24	92.644543
2020-04-25	81.935659
2020-04-26	80.566707
2020-04-27	73.639445
2020-04-28	70.086720
2020-04-29	61.897482
2020-04-30	66.256063
2020-05-01	64.792284
2020-05-02	58.812792
2020-05-03	62.173232
2020-05-04	55.255116
2020-05-05	51.711537
2020-05-06	43.531444

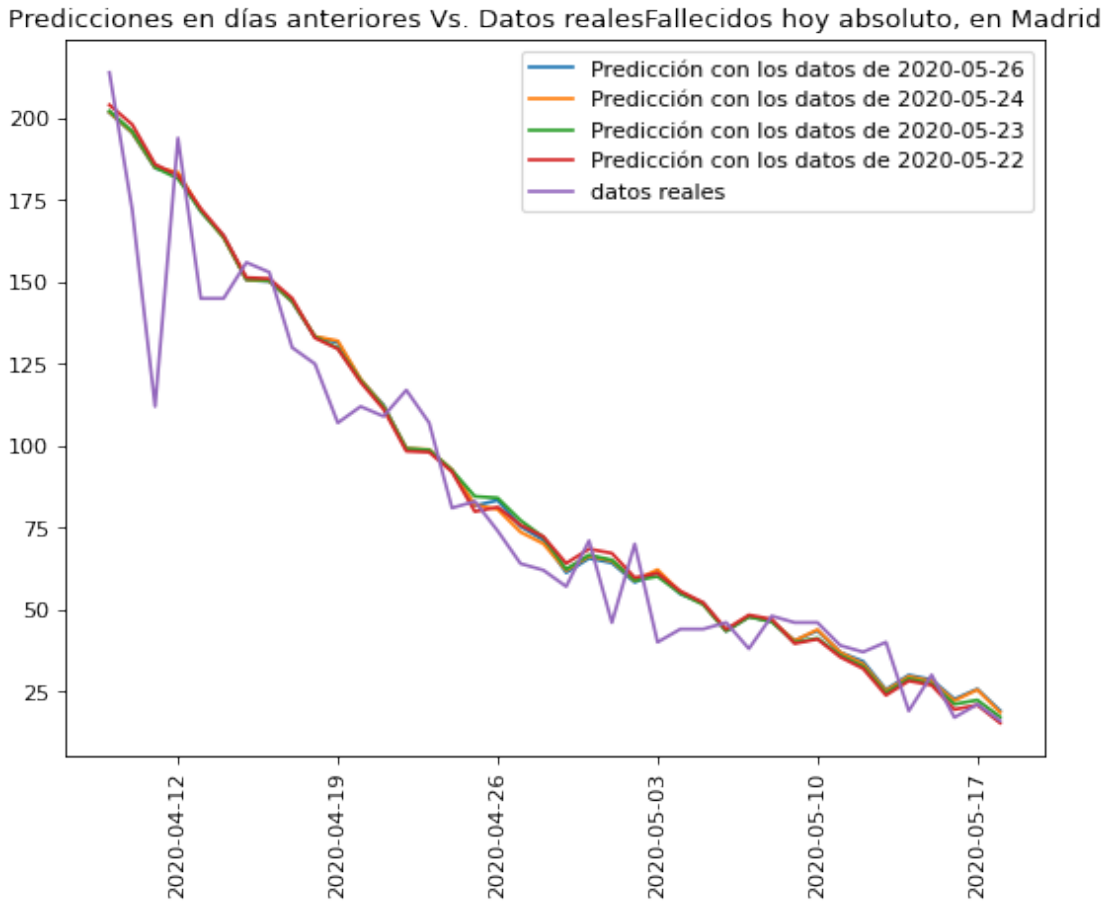
2020-05-07	47.897817
2020-05-08	46.441829
2020-05-09	40.465579
2020-05-10	43.829261
2020-05-11	36.913606
2020-05-12	33.372489
2020-05-13	25.194858
2020-05-14	29.562272
2020-05-15	28.107325
2020-05-16	22.131075
2020-05-17	25.494756
2020-05-18	18.579102

ds	Predicción con los datos de 2020-05-23 \
2020-04-09	202.230638
2020-04-10	196.023359
2020-04-11	185.036751
2020-04-12	181.596038
2020-04-13	171.728286
2020-04-14	163.603606
2020-04-15	150.842875
2020-04-16	150.472897
2020-04-17	144.266012
2020-04-18	133.280002
2020-04-19	129.839887
2020-04-20	119.974976
2020-04-21	111.853139
2020-04-22	99.097046
2020-04-23	98.731706
2020-04-24	92.529460
2020-04-25	84.530351
2020-04-26	84.077138
2020-04-27	77.200821
2020-04-28	72.067577
2020-04-29	62.298189
2020-04-30	66.621004
2020-05-01	65.106912
2020-05-02	58.815569
2020-05-03	60.070122
2020-05-04	54.899561
2020-05-05	51.472073
2020-05-06	43.408441
2020-05-07	47.735595
2020-05-08	46.225842
2020-05-09	39.936847
2020-05-10	41.193748

2020-05-11	36.023306
2020-05-12	32.595938
2020-05-13	24.532425
2020-05-14	28.859666
2020-05-15	27.350000
2020-05-16	21.061006
2020-05-17	22.317906
2020-05-18	17.147464

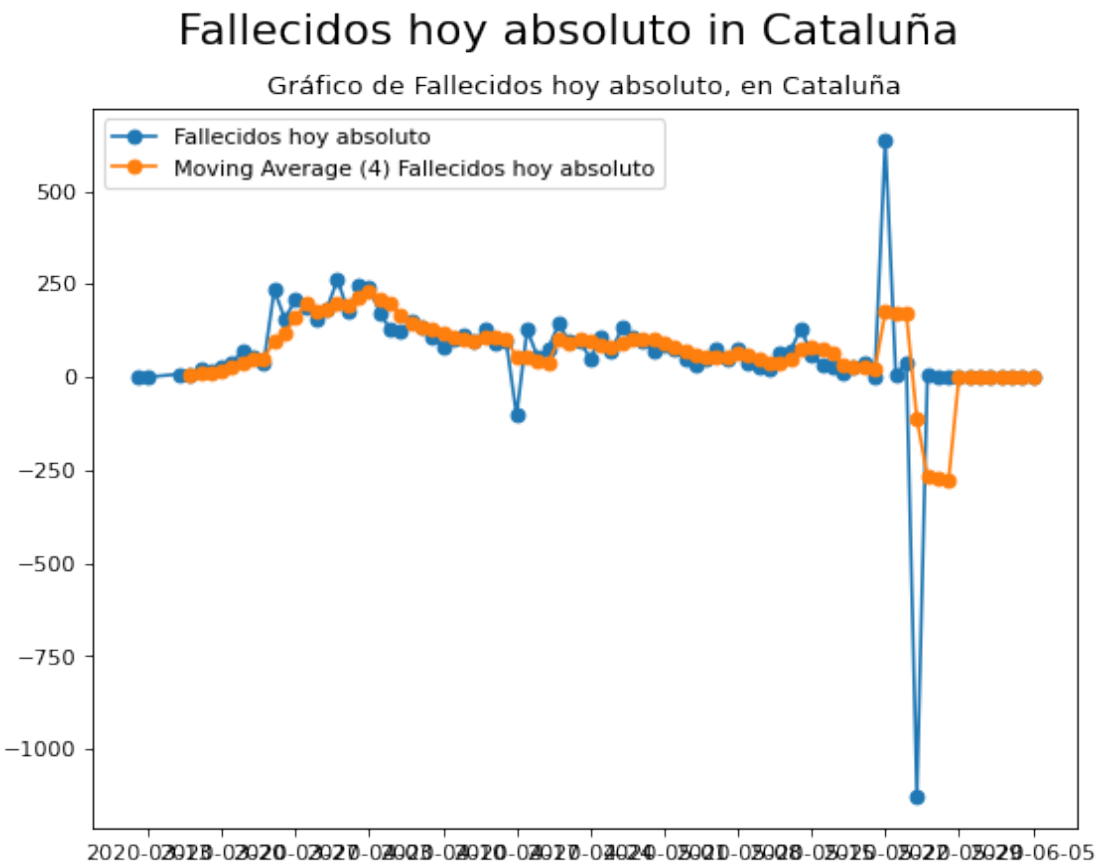
ds	Predicción con los datos de 2020-05-22	datos reales
2020-04-09	204.102713	214.0
2020-04-10	198.106717	172.0
2020-04-11	185.964283	112.0
2020-04-12	182.621428	194.0
2020-04-13	172.442022	145.0
2020-04-14	164.222118	145.0
2020-04-15	151.367241	156.0
2020-04-16	151.057798	153.0
2020-04-17	145.063882	130.0
2020-04-18	132.923529	125.0
2020-04-19	129.582753	107.0
2020-04-20	119.405425	112.0
2020-04-21	111.187600	109.0
2020-04-22	98.333877	117.0
2020-04-23	98.025588	107.0
2020-04-24	92.035583	81.0
2020-04-25	79.899142	83.0
2020-04-26	81.251700	74.0
2020-04-27	75.767706	64.0
2020-04-28	72.243215	62.0
2020-04-29	64.083549	57.0
2020-04-30	68.469317	71.0
2020-05-01	67.170618	46.0
2020-05-02	59.725482	70.0
2020-05-03	61.080587	40.0
2020-05-04	55.599139	44.0
2020-05-05	52.077195	44.0
2020-05-06	43.921116	46.0
2020-05-07	48.310471	38.0
2020-05-08	47.016000	48.0
2020-05-09	39.575091	46.0
2020-05-10	40.933749	46.0
2020-05-11	35.455855	39.0
2020-05-12	31.937465	37.0
2020-05-13	23.782675	40.0
2020-05-14	28.173318	19.0

2020-05-15	26.878846	30.0
2020-05-16	19.437938	17.0
2020-05-17	20.796596	21.0
2020-05-18	15.318702	16.0



[25]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Cataluña'

Insertar_Enlace("Reporte_Fallecidos_hoy_absoluto_Cataluña")

report_single_location_single_dimension(COMUNIDAD_A_CONSIDERAR,dimension)
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

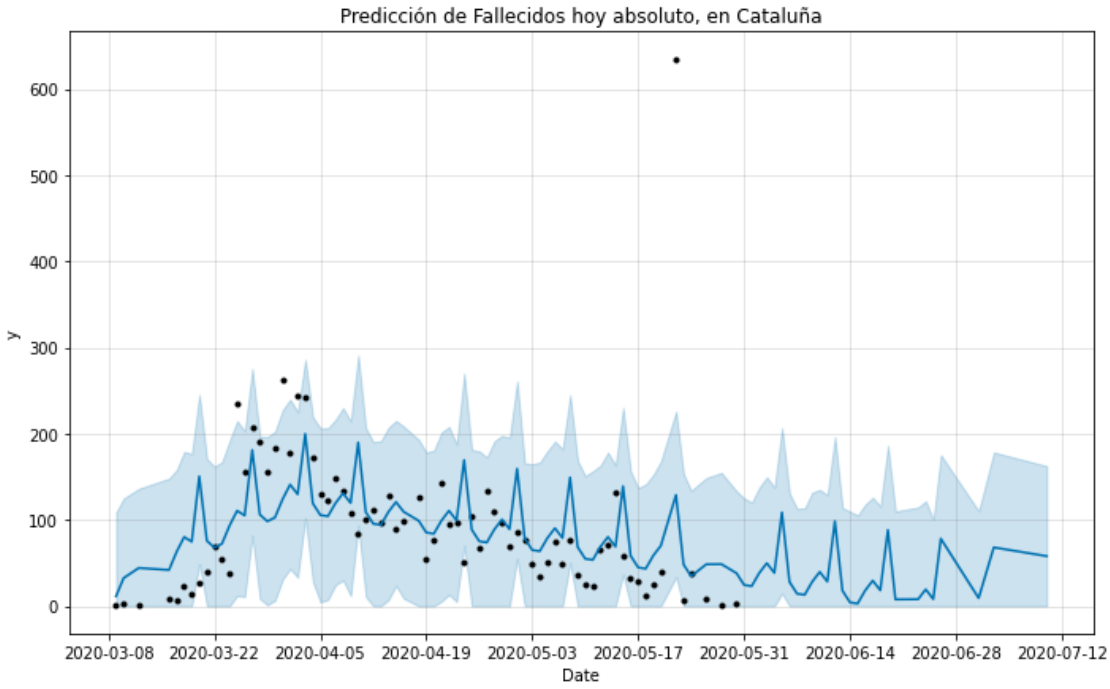


[26]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Cataluña'
link="Prediccion_Fallecidos_hoy_absoluto_Cataluña"


prediccion = Get_Prediction_Nacion( df = Loading_data.
 ↪Get_Comunidad(COMUNIDAD_A_CONSIDERAR),
                                    dimension = dimension,
                                    link = link,
                                    location  = COMUNIDAD_A_CONSIDERAR)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

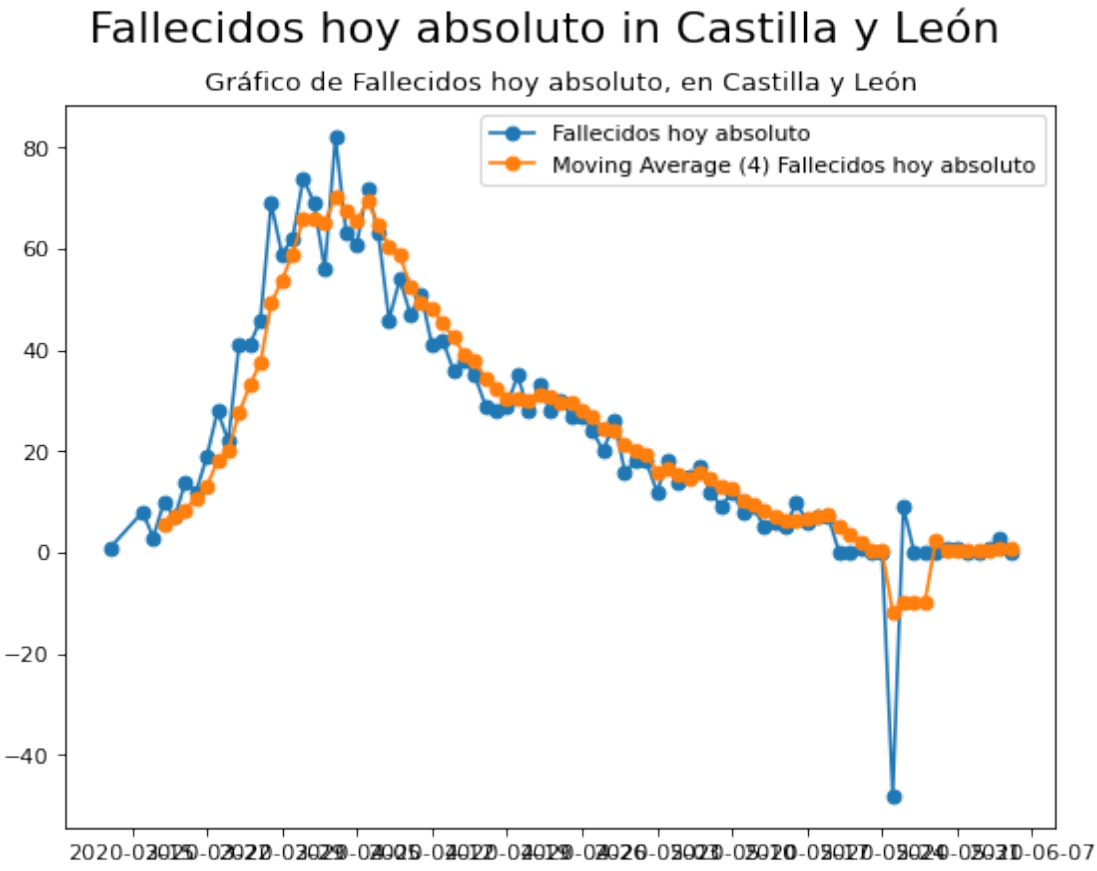
```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

```
Prediccion total para Fallecidos hoy absoluto : 7706.624052777044
```



[27]:
```
dimension = 'Fallecidos hoy absoluto'
report_single_location_single_dimension('Castilla y León',dimension)

Insertar_Enlace("Reporte_Fallecidos_hoy_absoluto_CyL")
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

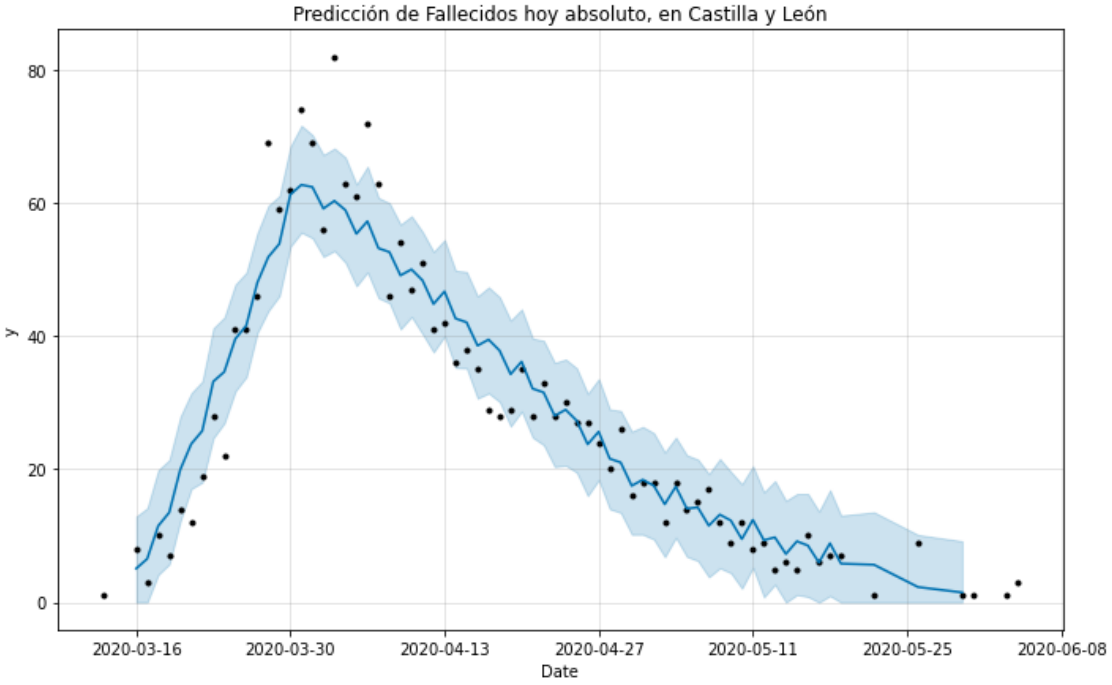


[28]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Castilla y León'
link="Prediccion_Fallecidos_hoy_absoluto_CyL"

prediccion = Get_Prediction_Nacion( df = Loading_data.
 ↪Get_Comunidad(COMUNIDAD_A_CONSIDERAR),
                                    dimension = dimension ,
                                    link = link,
                                    location  = COMUNIDAD_A_CONSIDERAR )
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

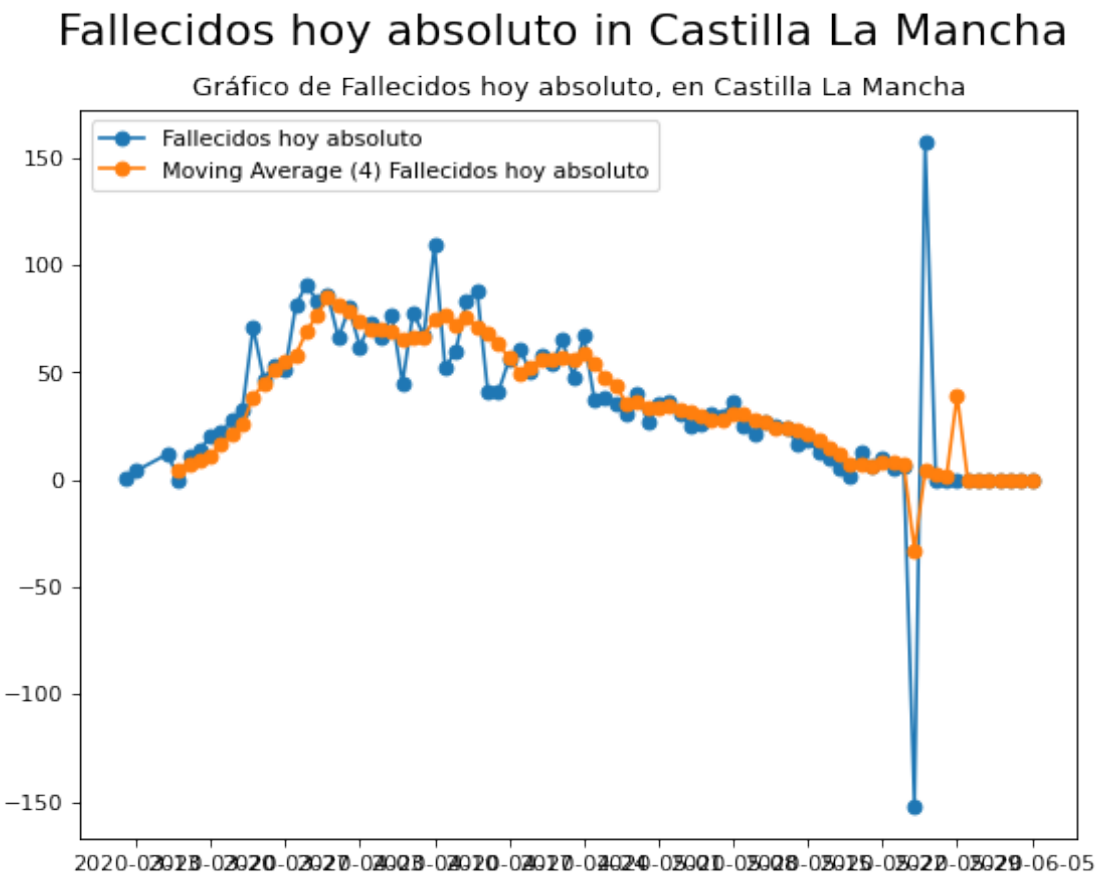
```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

```
Prediccion total para Fallecidos hoy absoluto : 1987.2211587713525
```



[29]:
```
dimension = 'Fallecidos hoy absoluto'
report_single_location_single_dimension('Castilla La Mancha',dimension)
Insertar_Enlace("Reporte_Fallecidos_hoy_absoluto_CM")
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```



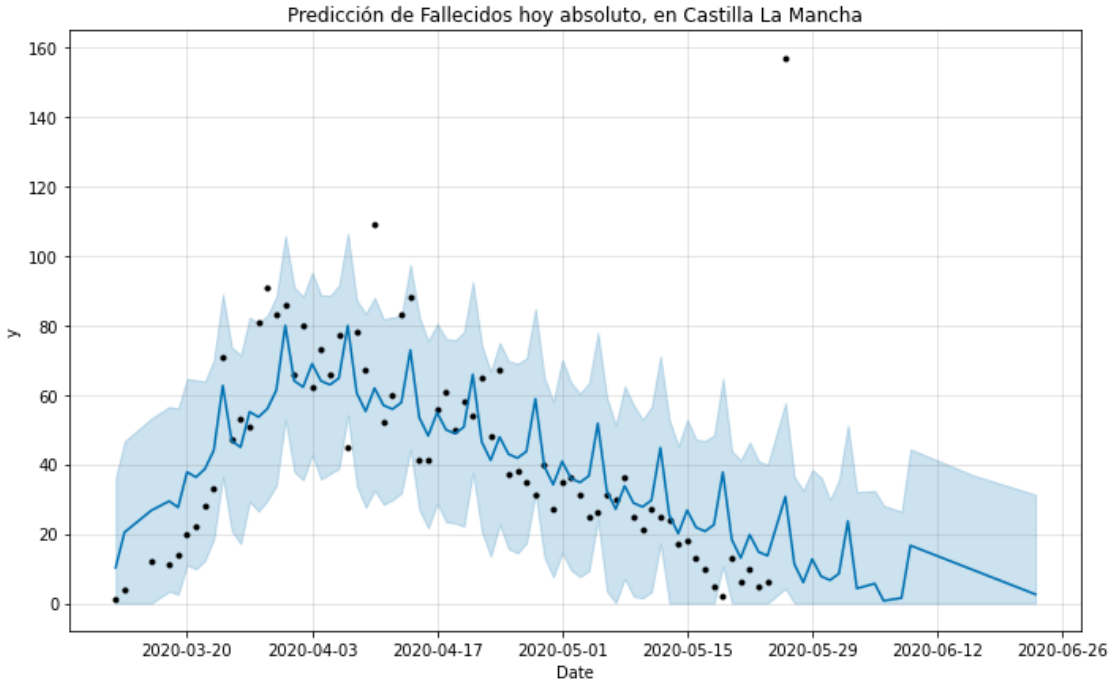
[30]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Castilla La Mancha'
link="Prediccion_Fallecidos_hoy_absoluto_CM"

prediccion = Get_Prediction_Nacion( df = Loading_data.
 ↪Get_Comunidad(COMUNIDAD_A_CONSIDERAR),
                                    dimension = dimension ,
                                    link = link,
                                    location  = COMUNIDAD_A_CONSIDERAR )
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
Prediccion total para Fallecidos hoy absoluto : 3179.2353242916706
```



[31]:
```
dimension = 'Fallecidos hoy absoluto'
COMUNIDAD_A_CONSIDERAR = 'Castilla La Mancha'
link="Prediccion_Compare_Fallecidos_hoy_absoluto_CM"

df = Loading_data.Get_Comunidad(COMUNIDAD_A_CONSIDERAR)
prediccion = Get_Predictions_Compare( df = df,
                                      dimension = dimension,
                                      link = link,
                                      location  = COMUNIDAD_A_CONSIDERAR
                                      )
prediccion
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.

Las predicciones del total de Fallecidos hoy absoluto en Castilla La Mancha,
cambian dia a dia
Predicción con los datos de 2020-05-26    1506.281025
Predicción con los datos de 2020-05-24    1334.306411
Predicción con los datos de 2020-05-23    1338.276538
Predicción con los datos de 2020-05-22    1330.640507
datos reales                              1308.000000
dtype: float64
```

[31]:

ds	Predicción con los datos de 2020-05-26 \
2020-04-14	72.880487
2020-04-15	53.475360
2020-04-16	48.203820
2020-04-17	54.890869
2020-04-18	49.938951
2020-04-19	48.842716
2020-04-20	50.732430
2020-04-21	65.850194
2020-04-22	46.444623
2020-04-23	41.172880
2020-04-24	47.859726
2020-04-25	42.907605
2020-04-26	41.811370
2020-04-27	43.701084
2020-04-28	58.818848
2020-04-29	39.413277
2020-04-30	34.141534
2020-05-01	40.828380
2020-05-02	35.876259
2020-05-03	34.780024
2020-05-04	36.669737
2020-05-05	51.787502
2020-05-06	32.381931
2020-05-07	27.110308
2020-05-08	33.797274
2020-05-09	28.845274

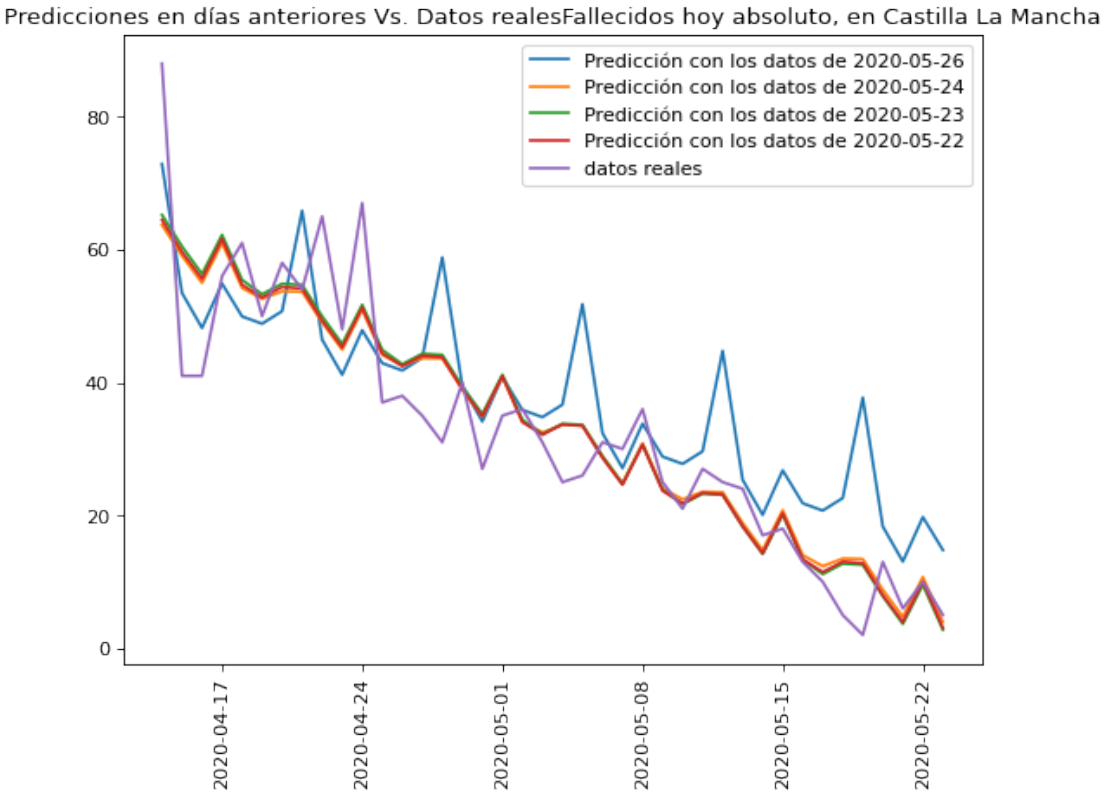
```
2020-05-10                                27.749160
2020-05-11                                29.639175
2020-05-12                                44.757241
2020-05-13                                25.351971
2020-05-14                                20.080529
2020-05-15                                26.767676
2020-05-16                                21.815857
2020-05-17                                20.719923
2020-05-18                                22.609938
2020-05-19                                37.728004
2020-05-20                                18.322734
2020-05-21                                13.051292
2020-05-22                                19.738439
2020-05-23                                14.786620

            Predicción con los datos de 2020-05-24  \
ds
2020-04-14                                63.707229
2020-04-15                                59.072719
2020-04-16                                54.991743
2020-04-17                                61.005644
2020-04-18                                54.230188
2020-04-19                                52.582425
2020-04-20                                53.722275
2020-04-21                                53.660971
2020-04-22                                49.023643
2020-04-23                                44.940573
2020-04-24                                50.952381
2020-04-25                                44.175627
2020-04-26                                42.526566
2020-04-27                                43.665872
2020-04-28                                43.604025
2020-04-29                                38.966629
2020-04-30                                34.883493
2020-05-01                                40.895233
2020-05-02                                34.118479
2020-05-03                                32.469418
2020-05-04                                33.608725
2020-05-05                                33.546877
2020-05-06                                28.909482
2020-05-07                                24.826345
2020-05-08                                30.838086
2020-05-09                                24.061332
2020-05-10                                22.412271
2020-05-11                                23.551577
2020-05-12                                23.489730
2020-05-13                                18.852335
```

```
2020-05-14                                14.769198
2020-05-15                                20.780938
2020-05-16                                14.004184
2020-05-17                                12.355124
2020-05-18                                13.494430
2020-05-19                                13.432582
2020-05-20                                 8.795187
2020-05-21                                 4.712050
2020-05-22                                10.723791
2020-05-23                                 3.947037

            Predicción con los datos de 2020-05-23  \
ds
2020-04-14                                65.215850
2020-04-15                                60.477493
2020-04-16                                56.312186
2020-04-17                                62.218689
2020-04-18                                55.440744
2020-04-19                                53.278765
2020-04-20                                54.858851
2020-04-21                                54.685427
2020-04-22                                49.946964
2020-04-23                                45.781648
2020-04-24                                51.688141
2020-04-25                                44.910196
2020-04-26                                42.748217
2020-04-27                                44.328302
2020-04-28                                44.154879
2020-04-29                                39.416416
2020-04-30                                35.251100
2020-05-01                                41.157592
2020-05-02                                34.379648
2020-05-03                                32.217668
2020-05-04                                33.797754
2020-05-05                                33.624331
2020-05-06                                28.885868
2020-05-07                                24.720551
2020-05-08                                30.627044
2020-05-09                                23.849099
2020-05-10                                21.687120
2020-05-11                                23.267206
2020-05-12                                23.093782
2020-05-13                                18.355319
2020-05-14                                14.190003
2020-05-15                                20.096496
2020-05-16                                13.318551
2020-05-17                                11.156572
```

2020-05-18	12.736657
2020-05-19	12.563234
2020-05-20	7.824771
2020-05-21	3.659455
2020-05-22	9.565947
2020-05-23	2.788003

ds	Predicción con los datos de 2020-05-22	datos reales
2020-04-14	64.458739	88.0
2020-04-15	59.647266	41.0
2020-04-16	55.594574	41.0
2020-04-17	61.635046	56.0
2020-04-18	54.705032	61.0
2020-04-19	52.798709	50.0
2020-04-20	54.366811	58.0
2020-04-21	54.131992	54.0
2020-04-22	49.319771	65.0
2020-04-23	45.266330	48.0
2020-04-24	51.305895	67.0
2020-04-25	44.374973	37.0
2020-04-26	42.467742	38.0
2020-04-27	44.035612	35.0
2020-04-28	43.800561	31.0
2020-04-29	38.988116	40.0
2020-04-30	34.934450	27.0
2020-05-01	40.974015	35.0
2020-05-02	34.043092	36.0
2020-05-03	32.135861	31.0
2020-05-04	33.703731	25.0
2020-05-05	33.468681	26.0
2020-05-06	28.656235	31.0
2020-05-07	24.602569	30.0
2020-05-08	30.642134	36.0
2020-05-09	23.711212	25.0
2020-05-10	21.803981	21.0
2020-05-11	23.371851	27.0
2020-05-12	23.136800	25.0
2020-05-13	18.324355	24.0
2020-05-14	14.270689	17.0
2020-05-15	20.310254	18.0
2020-05-16	13.379331	13.0
2020-05-17	11.472100	10.0
2020-05-18	13.039970	5.0
2020-05-19	12.804920	2.0
2020-05-20	7.992474	13.0
2020-05-21	3.938809	6.0

```
2020-05-22                          9.978373          10.0
2020-05-23                          3.047451           5.0
```

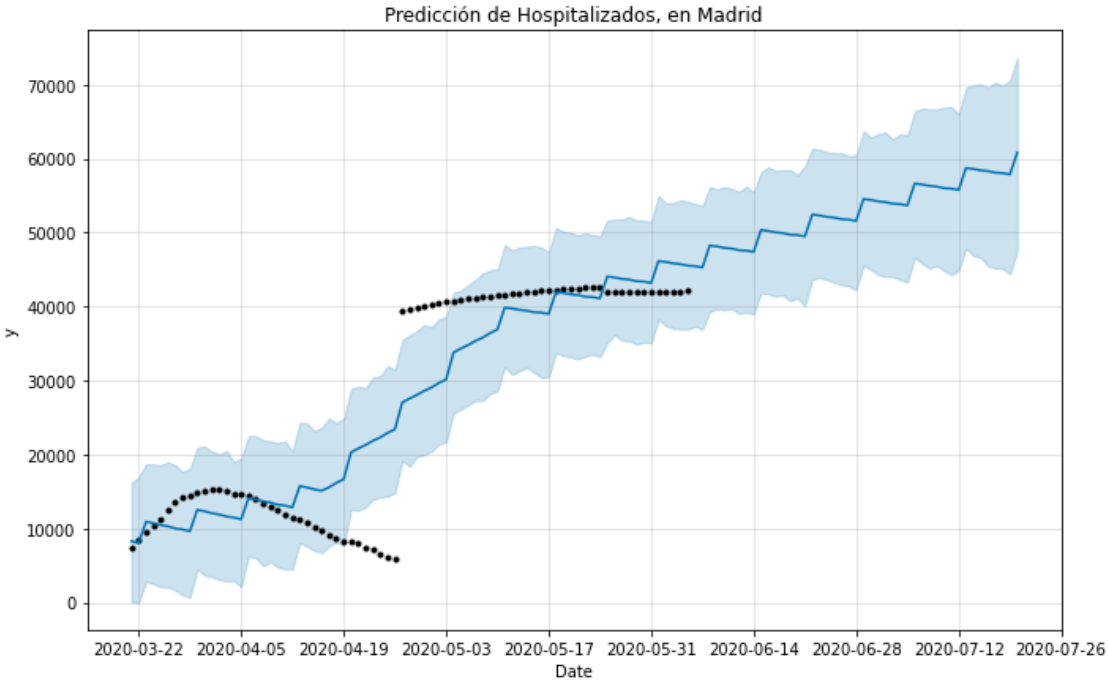


[32]:
```
dimension = 'Hospitalizados'
COMUNIDAD_A_CONSIDERAR = 'Madrid'
link="Prediccion_Hospitalizados_Madrid"

prediccion = Get_Prediction_Nacion( df = Loading_data.
 ↪Get_Comunidad(COMUNIDAD_A_CONSIDERAR),
                                    dimension = dimension ,
                                    link = link,
                                    location  = COMUNIDAD_A_CONSIDERAR )
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with
yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

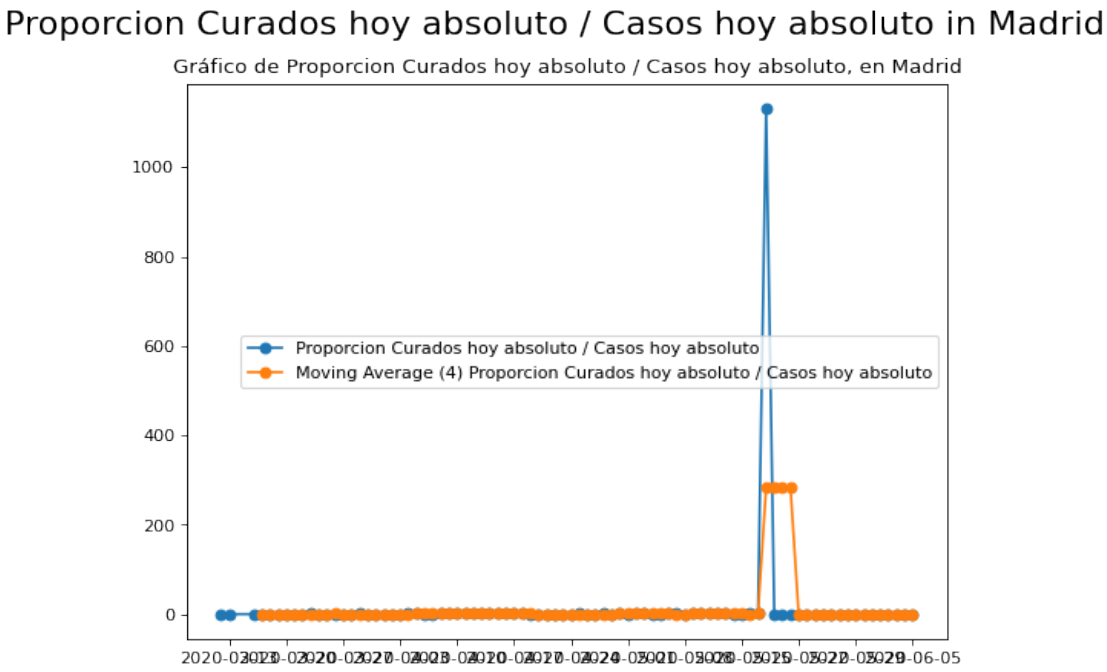
```
Prediccion total para Hospitalizados : 4458071.113367334
```



[33]:
```
dimension = 'Proporcion Curados hoy absoluto / Casos hoy absoluto'
report_single_location_single_dimension('Madrid',dimension)
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```



2 ¿ Son reales estas cifras ?

2.0.1 Actualizacion: Con los datos de mortalidad del insituto Carlos III, estas cifras se han quedado pequeñas. Ver notebook "Momo" para mas detalles.

Según reportaje de el mundo :https://www.elmundo.es/madrid/2020/04/07/5e8c427d21efa0b1668b45d6.html

Entre los días 15 y 31 de marzo fallecieron en Madrid capital, "por todas las causas", aunque la mayoría por coronavirus, 5.950 personas, cuando en 2019, en el mismo lapso, murieron 1.100 personas

[34]:
```
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')

COMUNIDAD_A_CONSIDERAR = 'Madrid'
comunidad = Loading_data.Get_Comunidad(COMUNIDAD_A_CONSIDERAR)

Insertar_Enlace("Reales")

comunidad.head(24).tail(15)['Fallecidos hoy absoluto'].sum()
comunidad['Fecha'] = comunidad.index

date1 = datetime.strptime('2020-03-15', '%Y-%m-%d')
date2 = datetime.strptime('2020-04-01', '%Y-%m-%d')
```

```
madrid_muertos_segun_sanidad = comunidad.loc[(comunidad['Fecha']>date1) &␣
 ↪(comunidad['Fecha']<date2)]['Fallecidos hoy absoluto'].sum()

display(HTML ("Madrid muertos segun, <b>sanidad</b>, segunda quincena de Marzo:␣
 ↪ <b>" + str(madrid_muertos_segun_sanidad)+"</b>"))

##

madrid_muertos_segun_interior = 5950 - 1100


display(HTML ("Madrid muertos segun, <b>interior</b>, mismo intervalo: <b>" +␣
 ↪str(madrid_muertos_segun_interior)+"</b>"))


porcentaje_error = (madrid_muertos_segun_interior -␣
 ↪madrid_muertos_segun_sanidad) / madrid_muertos_segun_sanidad


display(HTML ("La diferencia porcentual entre los muertos de sanidad e interior␣
 ↪es de  <b>" + str(porcentaje_error) + "</b>") )

prediccion_muertos = comunidad['Fallecidos hoy absoluto'].sum()*␣
 ↪(1+porcentaje_error)



display(HTML ("El numero de <b>fallecidos en Madrid</b>, hasta ahora es de <b>"␣
 ↪+ str(comunidad['Fallecidos hoy absoluto'].sum()) +
        "</b>, pero con el incremento del <b>" + str(porcentaje_error) +
        "</b> ,la cifra real sería de : <b>" + str(  int(prediccion_muertos))␣
 ↪+ "</b> muertos"
                                            ) )
Dimension = 'Fallecidos hoy absoluto'
df = Get_Dimension_CCAA(Dimension)

df['Total Fallecidos'] = df.sum(axis=1)

total_muertos_españa = int(df['Total Fallecidos'].sum())
prediccion_muertos_españa = total_muertos_españa* (1+porcentaje_error)

display(HTML ("El numero de <b>fallecidos en España</b>, hasta ahora es de <b>"␣
 ↪+ str(total_muertos_españa) +
        "</b>, pero con el incremento del <b>" + str(porcentaje_error) +
        "</b> ,la cifra real sería de : <b>" + str( ␣
 ↪int(prediccion_muertos_españa)) + "</b> muertos"
```

```
                                                ))
```

```
<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>


<IPython.core.display.HTML object>
```

[35]:
```
from datetime import datetime

COMUNIDAD_A_CONSIDERAR = 'Madrid'
#comunidad = Loading_data.Get_Comunidad(COMUNIDAD_A_CONSIDERAR)


comunidad.head(24).tail(15)['Fallecidos hoy absoluto'].sum()
comunidad['Fecha'] = comunidad.index

date1 = datetime.strptime('2020-03-15', '%Y-%m-%d')
date2 = datetime.strptime('2020-04-01', '%Y-%m-%d')

comunidad.loc[(comunidad['Fecha']>date1) &␣
 ↪(comunidad['Fecha']<date2)]['Fallecidos hoy absoluto'].sum()
```

[35]:
```
3522
```

[36]:
```
y = [comunidad['Fallecidos hoy absoluto'].sum(),total_muertos_españa]
z = [comunidad['Fallecidos hoy absoluto'].sum()* (1+porcentaje_error),␣
 ↪total_muertos_españa* (1+porcentaje_error)]

X = np.arange(2)

ax=plt.subplot(111)
plt.bar(X+0, y,color = 'b', width = 0.25)
plt.bar(X+0.25, z,color = 'r', width = 0.25)


ax.set_title("Diferencia entre las cifras de muertos\n para Madrid y España\n␣
 ↪entre los ministerios de Sanidad e Interior")
```

[36]: Text(0.5, 1.0, 'Diferencia entre las cifras de muertos\n para Madrid y España\n entre los ministerios de Sanidad e Interior')