Global_infected

August 3, 2020

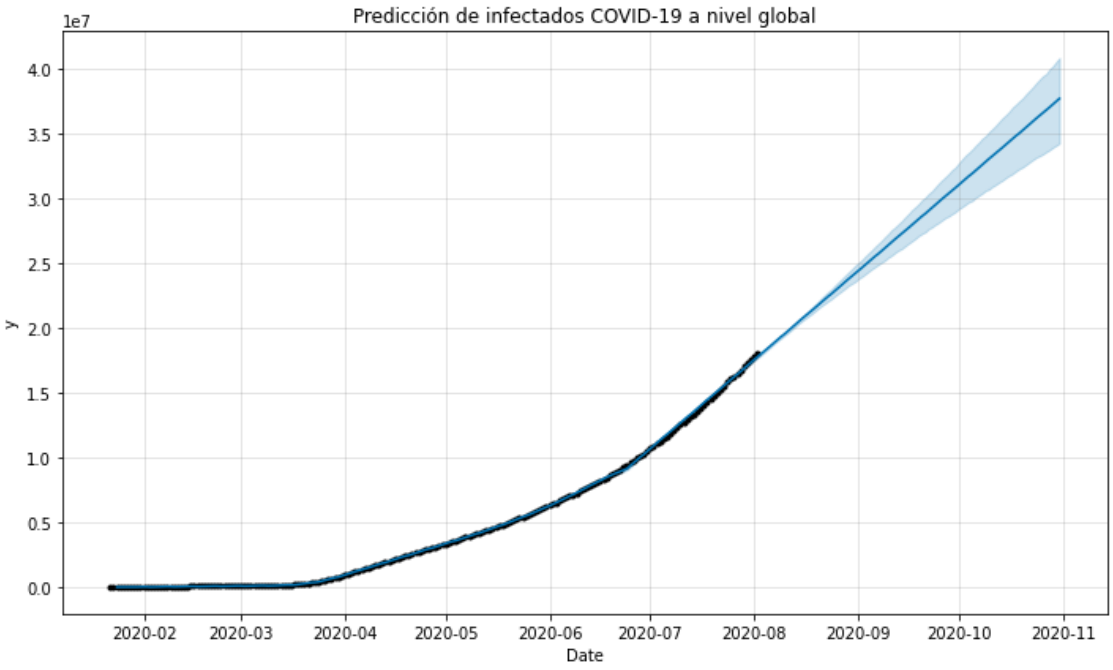
[1]:
```
import pandas as pd
import janitor

def get_data():
    URL_CSV="https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/
 ↪csse_covid_19_data/csse_covid_19_time_series/
 ↪time_series_covid19_confirmed_global.csv"
    df = pd.read_csv(URL_CSV)
    df['Country/Region'] = df['Country/Region'].replace({'US':'United States of␣
 ↪America'})
    return df
df = get_data()
```

[2]:
```
import janitor
import pandas as pd
import pandas_flavor as pf
import fbprophet
from matplotlib import pyplot as plt
from IPython.display import display, HTML



LISTA_COLUMNAS_A_BORRAR = ['Province/State', 'Country/Region', 'Lat', 'Long']



df = get_data()
df = df.remove_columns(LISTA_COLUMNAS_A_BORRAR)
df
df = pd.DataFrame( df.sum())
df.columns=['y']
df.index = pd.to_datetime(df.index)
df['ds'] = df.index
df = df.reset_index()


df = df.remove_columns(['index'])
```

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

df_forecast = df_prophet.make_future_dataframe(periods=90, 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


df_prophet.plot(df_forecast, xlabel = 'Date' )
plt.title('Predicción de infectados COVID-19 a nivel global'   )

display(HTML(pd.DataFrame(df_forecast).to_html()))
```

```
INFO:numexpr.utils:NumExpr defaulting to 8 threads.
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>
```



[3]:
```
df = get_data()
```

[4]:
```
import janitor
import datetime
import numpy as np

def pipeline_populations():
    """ Cogemos un dataframe de poblaciones"""

    URL="https://population.un.org/wpp/Download/Files/1_Indicators%20(Standard)/CSV_FILES/WPP2019_TotalPopulationBySex.csv"
    THIS_YEAR = datetime.datetime.now().year

    return (
        pd.read_csv(URL)
        .filter_on( f""" Time  ==  {THIS_YEAR} & Variant == "Medium" """ )
        .select_columns(['Location','PopTotal'])
        .join_apply(lambda x: x['PopTotal'] * 1000 ,new_column_name="PopMillions" )
        .remove_columns(['PopTotal'])
        .rename_column('PopMillions' , 'PopTotal')
        .transform_column('PopTotal',np.int64)
        .rename_column('Location' , 'Country/Region')
    )


populations = pipeline_populations()
populations
```

[4]:

	Country/Region	PopTotal
70	Afghanistan	38928341
954	Africa	1340598113
1838	African Group	1338826591
1989	African Union	1339423920
2140	African Union: Central Africa	158619638
…	…	…
277315	World	7794798728
278199	World Bank Regional Groups (developing only)	6528762227
278350	Yemen	29825967
279234	Zambia	18383956
280118	Zimbabwe	14862927

[477 rows x 2 columns]

[5]:
```
df = get_data()
df_country = pd.DataFrame()
df_country['Country/Region'] = df['Country/Region']
df_country['infected last_day'] = df.iloc[:,-1] - df.iloc[:,-2]
df_country.set_index('Country/Region')
df_country
```

[5]:

	Country/Region	infected last_day
0	Afghanistan	0
1	Albania	123
2	Algeria	515
3	Andorra	0
4	Angola	35
..	…	…
261	Sao Tome and Principe	0
262	Yemen	4
263	Comoros	0
264	Tajikistan	44
265	Lesotho	16

[266 rows x 2 columns]

[6]:
```
df_country_enrich = pd.merge(df_country, populations, on="Country/Region")
df_country_enrich['Infected/Million'] =  1000000 * df_country_enrich['infected␣
↪last_day'] / df_country_enrich['PopTotal']
df_country = df_country_enrich
```

[7]:
```
df_country.sort_values(by='infected last_day', ascending=False).head(10)
```

[7]:

	Country/Region	infected last_day	PopTotal	Infected/Million
129	India	52972	1380004385	38.385385
222	United States of America	47511	331002647	143.536616
27	Brazil	25800	212559409	121.377831
177	Peru	21358	32971845	647.764782
83	Colombia	11470	50882884	225.419613
195	South Africa	8195	59308690	138.175367
6	Argentina	5376	45195777	118.949166
178	Philippines	4953	109581085	45.199406
154	Mexico	4853	128932753	37.639776
131	Iraq	2447	40222503	60.836592

[8]:
```
#df_country['Country/Region'].unique()
```

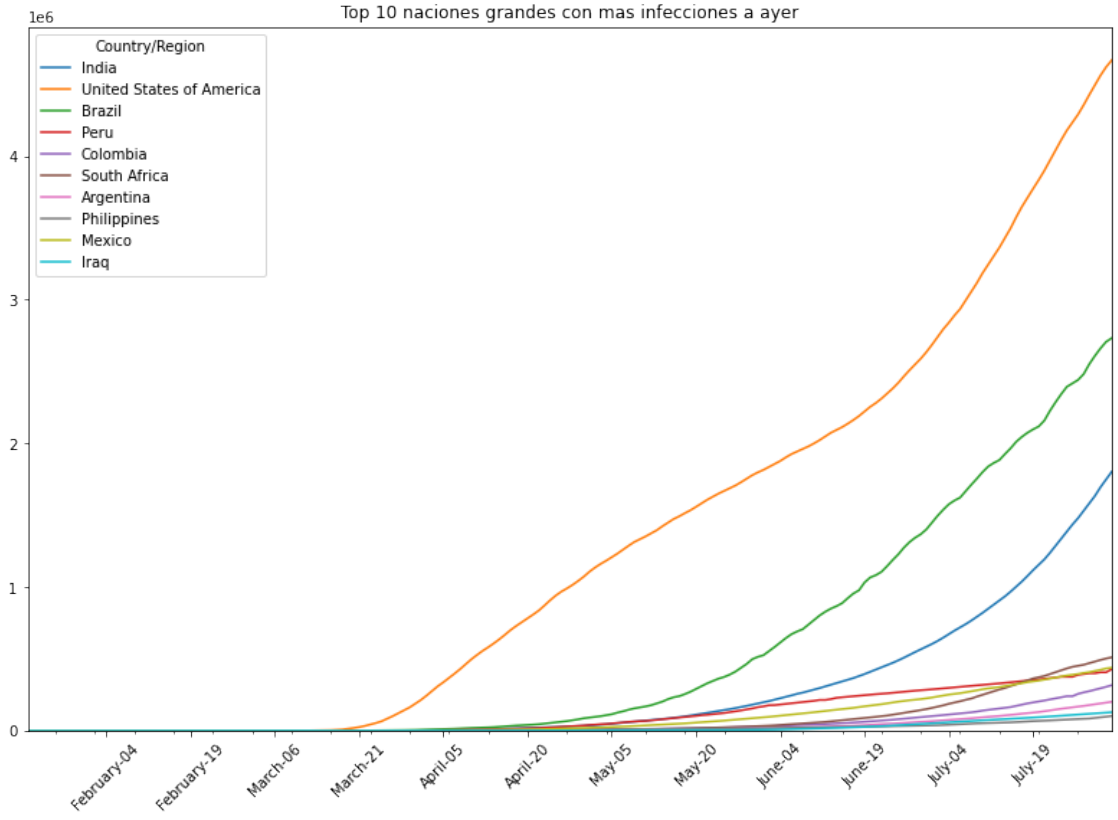
[9]:
```
#populations['Country/Region'].unique()
```

[10]:
```
from datetime import datetime, timedelta
import seaborn as sns
```

```
from matplotlib import pyplot as plt
import matplotlib.dates as mdates


def pintar_grafico(df, array_naciones_pintar ,title):
    df = df.T
    df = df.iloc[1:]
    new_header = df.iloc[0] #grab the first row for the header
    df = df[1:] #take the data less the header row
    df.columns = new_header #set the header row as the df header
    df = df.iloc[2:]
    df.index = pd.to_datetime(df.index)
    df = df[array_naciones_pintar]
    chart_df = df


    chart_df.plot(legend=True,figsize=(13.5,9))

    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%B-%d'))
    plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=15))
    plt.xticks(rotation=45)

    ax = plt.gca()

    ax.set_title(title)
    ax.set_ylim(ymin=0)

    plt.show()

    df.tail(30).style.format ({ c : "{:20,.0f}" for c in df.columns }).
 ↪background_gradient(cmap='Wistia', )
    return plt
```

[11]:
```
title="Top 10 naciones grandes con mas infecciones a ayer"
display(HTML(f"""<h1 id='{title}'>{title}</h1>"""))

naciones_pintar = df_country.sort_values(by='infected last_day',␣
 ↪ascending=False).head(10)['Country/Region'].values
df_country.sort_values(by='infected last_day', ascending=False).head(10)

pintar_grafico(df,naciones_pintar,title)
```

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



[11]:
```
<module 'matplotlib.pyplot' from
'/root/anaconda2/envs/jupyter/lib/python3.6/site-packages/matplotlib/pyplot.py'>
```

[12]:
```
#title="Top 10 naciones grandes con mas infeccionesa ayer en infectados/millon"
#display(HTML(f"""<h1 id='{title}'>{title}</h1>"""))

#naciones_pintar = df_country.filter_on("PopTotal > 25000000").
 ↪sort_values(by='Infected/Million', ascending=False).head(10)['Country/
 ↪Region'].values
#df_country.sort_values(by='infected last_day', ascending=False).head(10)

#pintar_grafico(df_country,naciones_pintar,title)
```

[ ]:

[ ]: