# Regresion-COVID19ECUADOR

August 2, 2021

## 1 Actualización de datos de Covid-19 en la infeccion hasta el 2021

## 2 Covid-19 infección en Ecuador. Modelos matemáticos y predicciones

Una comparación de modelos, lineal, polilnomico,logísticos y exponenciales aplicados a la infección por el virus Covid-19

Se realiza un análisis matemático simple del crecimiento de la infección en Python y dos modelos para comprender mejor la evolución de la infección.

Se crea modelos de series temporales del número total de personas infectadas hasta la fecha (es decir, las personas realmente infectadas más las personas que han sido infectadas). Estos modelos tienen parámetros , que se estimarán por ajuste de curva.

[1]:
```
#Problema:
#Generar un modelo de regresión de los casos confirmados de COVID dentro del␣
↪Ecuador el mismo que permitirá predecir el comportamiento y/o predicción de␣
↪la pandemia

#Importación de librerías
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from datetime import datetime,timedelta
from sklearn.metrics import mean_squared_error
from scipy.optimize import curve_fit
from scipy.optimize import fsolve
from sklearn import linear_model
from sklearn.preprocessing import PolynomialFeatures
import matplotlib.pyplot as plt
print('Importadas')
```

```
Importadas
```

[2]:
```
#Carga de datos

#Variables
#dataset = pd.read_csv('dataset.csv')
```

```python
#print(dataset.head())

df=pd.read_csv('dataset.csv')
print(df.shape)
df.sample(5)
```

```
(407, 59)
```

[2]:

```
    iso_code      continent location        date  total_cases  new_cases  \
209      ECU  South America  Ecuador  2020-09-26       133981       1506
54       ECU  South America  Ecuador  2020-04-24        22719      11536
146      ECU  South America  Ecuador  2020-07-25        80036        987
26       ECU  South America  Ecuador  2020-03-27         1595        192
113      ECU  South America  Ecuador  2020-06-22        50640          0

     new_cases_smoothed  total_deaths  new_deaths  new_deaths_smoothed  …  \
209            1194.429       11273.0        37.0               27.000  …
54             2038.429         576.0        16.0               22.143  …
146             950.571        5507.0        39.0               32.143  …
26              175.429          36.0         2.0                4.429  …
113             474.000        4223.0         0.0               42.000  …

     gdp_per_capita  extreme_poverty  cardiovasc_death_rate  \
209       10581.936              3.6                140.448
54        10581.936              3.6                140.448
146       10581.936              3.6                140.448
26        10581.936              3.6                140.448
113       10581.936              3.6                140.448

     diabetes_prevalence  female_smokers  male_smokers  \
209                 5.55               2          12.3
54                  5.55               2          12.3
146                 5.55               2          12.3
26                  5.55               2          12.3
113                 5.55               2          12.3

     handwashing_facilities  hospital_beds_per_thousand  life_expectancy  \
209                  80.635                         1.5            77.01
54                   80.635                         1.5            77.01
146                  80.635                         1.5            77.01
26                   80.635                         1.5            77.01
113                  80.635                         1.5            77.01

     human_development_index
209                    0.759
54                     0.759
146                    0.759
```

```
26                     0.759
113                    0.759

[5 rows x 59 columns]
```

[3]:
```python
df.describe().round(3)
```

[3]:
```
       total_cases  new_cases  new_cases_smoothed  total_deaths  new_deaths  \
count      407.000    407.000             402.000       394.000     394.000
mean    137704.005    852.130             849.336      9215.898      43.891
std     100545.657    967.507             468.354      5646.413     196.902
min          6.000  -7953.000            -525.000         2.000       0.000
25%      44440.000    353.000             562.572      4172.750      12.000
50%     126419.000    765.000             869.143     11276.000      27.000
75%     212012.000   1202.000            1123.750     14059.000      43.000
max     346817.000  11536.000            2038.429     17293.000    3852.000

       new_deaths_smoothed  total_cases_per_million  new_cases_per_million  \
count              402.000                  407.000                407.000
mean                42.604                 7804.996                 48.298
std                 75.717                 5698.879                 54.838
min                  0.000                    0.340               -450.772
25%                 21.179                 2518.837                 20.008
50%                 31.143                 7165.367                 43.360
75%                 39.964                12016.736                 68.128
max                597.000                19657.418                653.855

       new_cases_smoothed_per_million  total_deaths_per_million  …  \
count                         402.000                   394.000  …
mean                           48.140                   522.353  …
std                            26.546                   320.036  …
min                           -29.757                     0.113  …
25%                            31.886                   236.510  …
50%                            49.262                   639.118  …
75%                            63.694                   796.857  …
max                           115.537                   980.159  …

       gdp_per_capita  extreme_poverty  cardiovasc_death_rate  \
count         407.000            407.0                407.000
mean        10581.936              3.6                140.448
std             0.000              0.0                  0.000
min         10581.936              3.6                140.448
25%         10581.936              3.6                140.448
50%         10581.936              3.6                140.448
75%         10581.936              3.6                140.448
max         10581.936              3.6                140.448
```

```
       diabetes_prevalence  female_smokers  male_smokers  \
count               407.00           407.0         407.0
mean                  5.55             2.0          12.3
std                   0.00             0.0           0.0
min                   5.55             2.0          12.3
25%                   5.55             2.0          12.3
50%                   5.55             2.0          12.3
75%                   5.55             2.0          12.3
max                   5.55             2.0          12.3

       handwashing_facilities  hospital_beds_per_thousand  life_expectancy  \
count                 407.000                       407.0           407.00
mean                   80.635                         1.5            77.01
std                     0.000                         0.0             0.00
min                    80.635                         1.5            77.01
25%                    80.635                         1.5            77.01
50%                    80.635                         1.5            77.01
75%                    80.635                         1.5            77.01
max                    80.635                         1.5            77.01

       human_development_index
count                  407.000
mean                     0.759
std                      0.000
min                      0.759
25%                      0.759
50%                      0.759
75%                      0.759
max                      0.759

[8 rows x 54 columns]
```

[4]:
```
df.keys()
```

[4]:
```
Index(['iso_code', 'continent', 'location', 'date', 'total_cases', 'new_cases',
       'new_cases_smoothed', 'total_deaths', 'new_deaths',
       'new_deaths_smoothed', 'total_cases_per_million',
       'new_cases_per_million', 'new_cases_smoothed_per_million',
       'total_deaths_per_million', 'new_deaths_per_million',
       'new_deaths_smoothed_per_million', 'reproduction_rate', 'icu_patients',
       'icu_patients_per_million', 'hosp_patients',
       'hosp_patients_per_million', 'weekly_icu_admissions',
       'weekly_icu_admissions_per_million', 'weekly_hosp_admissions',
       'weekly_hosp_admissions_per_million', 'new_tests', 'total_tests',
       'total_tests_per_thousand', 'new_tests_per_thousand',
       'new_tests_smoothed', 'new_tests_smoothed_per_thousand',
       'positive_rate', 'tests_per_case', 'tests_units', 'total_vaccinations',
```

```
       'people_vaccinated', 'people_fully_vaccinated', 'new_vaccinations',
       'new_vaccinations_smoothed', 'total_vaccinations_per_hundred',
       'people_vaccinated_per_hundred', 'people_fully_vaccinated_per_hundred',
       'new_vaccinations_smoothed_per_million', 'stringency_index',
       'population', 'population_density', 'median_age', 'aged_65_older',
       'aged_70_older', 'gdp_per_capita', 'extreme_poverty',
       'cardiovasc_death_rate', 'diabetes_prevalence', 'female_smokers',
       'male_smokers', 'handwashing_facilities', 'hospital_beds_per_thousand',
       'life_expectancy', 'human_development_index'],
      dtype='object')
```

[5]:

```
#Analizamos el tipo de dato que tiene el dataset en caso de que tengamos que␣
 ↪convertir algún valor
df.dtypes
```

[5]:

```
iso_code                                 object
continent                                object
location                                 object
date                                     object
total_cases                               int64
new_cases                                 int64
new_cases_smoothed                      float64
total_deaths                            float64
new_deaths                              float64
new_deaths_smoothed                     float64
total_cases_per_million                 float64
new_cases_per_million                   float64
new_cases_smoothed_per_million          float64
total_deaths_per_million                float64
new_deaths_per_million                  float64
new_deaths_smoothed_per_million         float64
reproduction_rate                       float64
icu_patients                            float64
icu_patients_per_million                float64
hosp_patients                           float64
hosp_patients_per_million               float64
weekly_icu_admissions                   float64
weekly_icu_admissions_per_million       float64
weekly_hosp_admissions                  float64
weekly_hosp_admissions_per_million      float64
new_tests                               float64
total_tests                             float64
total_tests_per_thousand                float64
new_tests_per_thousand                  float64
new_tests_smoothed                      float64
new_tests_smoothed_per_thousand         float64
positive_rate                           float64
```

```
tests_per_case                             float64
tests_units                                 object
total_vaccinations                         float64
people_vaccinated                          float64
people_fully_vaccinated                    float64
new_vaccinations                           float64
new_vaccinations_smoothed                  float64
total_vaccinations_per_hundred             float64
people_vaccinated_per_hundred              float64
people_fully_vaccinated_per_hundred        float64
new_vaccinations_smoothed_per_million      float64
stringency_index                           float64
population                                   int64
population_density                         float64
median_age                                 float64
aged_65_older                              float64
aged_70_older                              float64
gdp_per_capita                             float64
extreme_poverty                            float64
cardiovasc_death_rate                      float64
diabetes_prevalence                        float64
female_smokers                               int64
male_smokers                               float64
handwashing_facilities                     float64
hospital_beds_per_thousand                 float64
life_expectancy                            float64
human_development_index                    float64
dtype: object
```

[6]:
```
# Expresar las fechas en numero de dias desde el 01 Enero del 2020
FMT = '%Y-%m-%d'
date = df['date']
df['date'] = date.map(lambda x : (datetime.strptime(x, FMT) - datetime.
 ↪strptime("2020-01-01", FMT)).days)
```

[7]:
```
df.head()
```

[7]:
```
  iso_code      continent location  date  total_cases  new_cases  \
0      ECU  South America  Ecuador    60            6          6
1      ECU  South America  Ecuador    61            6          0
2      ECU  South America  Ecuador    62            7          1
3      ECU  South America  Ecuador    63           10          3
4      ECU  South America  Ecuador    64           13          3

   new_cases_smoothed  total_deaths  new_deaths  new_deaths_smoothed  …  \
0                 NaN           NaN         NaN                  NaN  …
1                 NaN           NaN         NaN                  NaN  …
```

```
2                 NaN           NaN         NaN                 NaN  …
3                 NaN           NaN         NaN                 NaN  …
4                 NaN           NaN         NaN                 NaN  …

   gdp_per_capita  extreme_poverty  cardiovasc_death_rate  \
0       10581.936              3.6                140.448
1       10581.936              3.6                140.448
2       10581.936              3.6                140.448
3       10581.936              3.6                140.448
4       10581.936              3.6                140.448

   diabetes_prevalence  female_smokers  male_smokers  handwashing_facilities  \
0                 5.55               2          12.3                  80.635
1                 5.55               2          12.3                  80.635
2                 5.55               2          12.3                  80.635
3                 5.55               2          12.3                  80.635
4                 5.55               2          12.3                  80.635

   hospital_beds_per_thousand  life_expectancy  human_development_index
0                         1.5            77.01                    0.759
1                         1.5            77.01                    0.759
2                         1.5            77.01                    0.759
3                         1.5            77.01                    0.759
4                         1.5            77.01                    0.759

[5 rows x 59 columns]
```

[8]:
```
#Variables
totaldate = df['date'].values.reshape(-1,1)
totalcases = df['total_cases'].values.reshape(-1,1)
```

[11]:
```
from sklearn.linear_model import LinearRegression
#Entrenamiento
linear_regressor = LinearRegression()
linear_regressor.fit(totaldate, totalcases)
date_predicted = linear_regressor.predict(totaldate)
```

[12]:
```
print('Pentiende:', linear_regressor.coef_)
print('Intersección":',  linear_regressor.intercept_)
```

```
Pentiende: [[848.29342989]]
Intersección": [-85397.16714594]
```

[13]:
```
#PRUEBA 1

m = linear_regressor.coef_[0][0]
c = linear_regressor.intercept_[0]
#Prediccion de x dias desde la fecha inicial del Dataset 2020-03-01
```

```python
dias=700
label1 = m*dias-c
print("Numero de casos a los ",dias," de la fecha inicial del dataset, se␣
↪obtiene una prediccion de : ",label1)
```

```
Numero de casos a los  700  de la fecha inicial del dataset, se obtiene una
prediccion de :  679202.5680659452
```

[14]:
```python
#PRUEBA 2

m = linear_regressor.coef_[0][0]
c = linear_regressor.intercept_[0]
#Prediccion de x dias desde la fecha inicial del Dataset 2020-03-01
dias=750
label2 = m*dias-c
print("Numero de casos a los ",dias," de la fecha inicial del dataset, se␣
↪obtiene una prediccion de : ",label2)
```

```
Numero de casos a los  750  de la fecha inicial del dataset, se obtiene una
prediccion de :  721617.2395602312
```

[15]:
```python
#PRUEBA 3

m = linear_regressor.coef_[0][0]
c = linear_regressor.intercept_[0]
#Prediccion de x dias desde la fecha inicial del Dataset 2020-03-01
dias=800
label3 = m*dias-c
print("Numero de casos a los ",dias," de la fecha inicial del dataset, se␣
↪obtiene una prediccion de : ",label3)
```

```
Numero de casos a los  800  de la fecha inicial del dataset, se obtiene una
prediccion de :  764031.9110545174
```

[16]:
```python
#PRUEBA 4

m = linear_regressor.coef_[0][0]
c = linear_regressor.intercept_[0]
#Prediccion de x dias desde la fecha inicial del Dataset 2020-03-01
dias=900
label4 = m*dias-c
print("Numero de casos a los ",dias," de la fecha inicial del dataset, se␣
↪obtiene una prediccion de : ",label4)
```

```
Numero de casos a los  900  de la fecha inicial del dataset, se obtiene una
prediccion de :  848861.2540430896
```

[17]:
```python
#PRUEBA 5
```

```
m = linear_regressor.coef_[0][0]
c = linear_regressor.intercept_[0]
#Prediccion de x dias desde la fecha inicial del Dataset 2020-03-01
dias=968
label5 = m*dias-c
print("Numero de casos a los ",dias," de la fecha inicial del dataset, se␣
 ↪obtiene una prediccion de : ",label5)
```

```
Numero de casos a los  968  de la fecha inicial del dataset, se obtiene una
prediccion de :  906545.2072753186
```

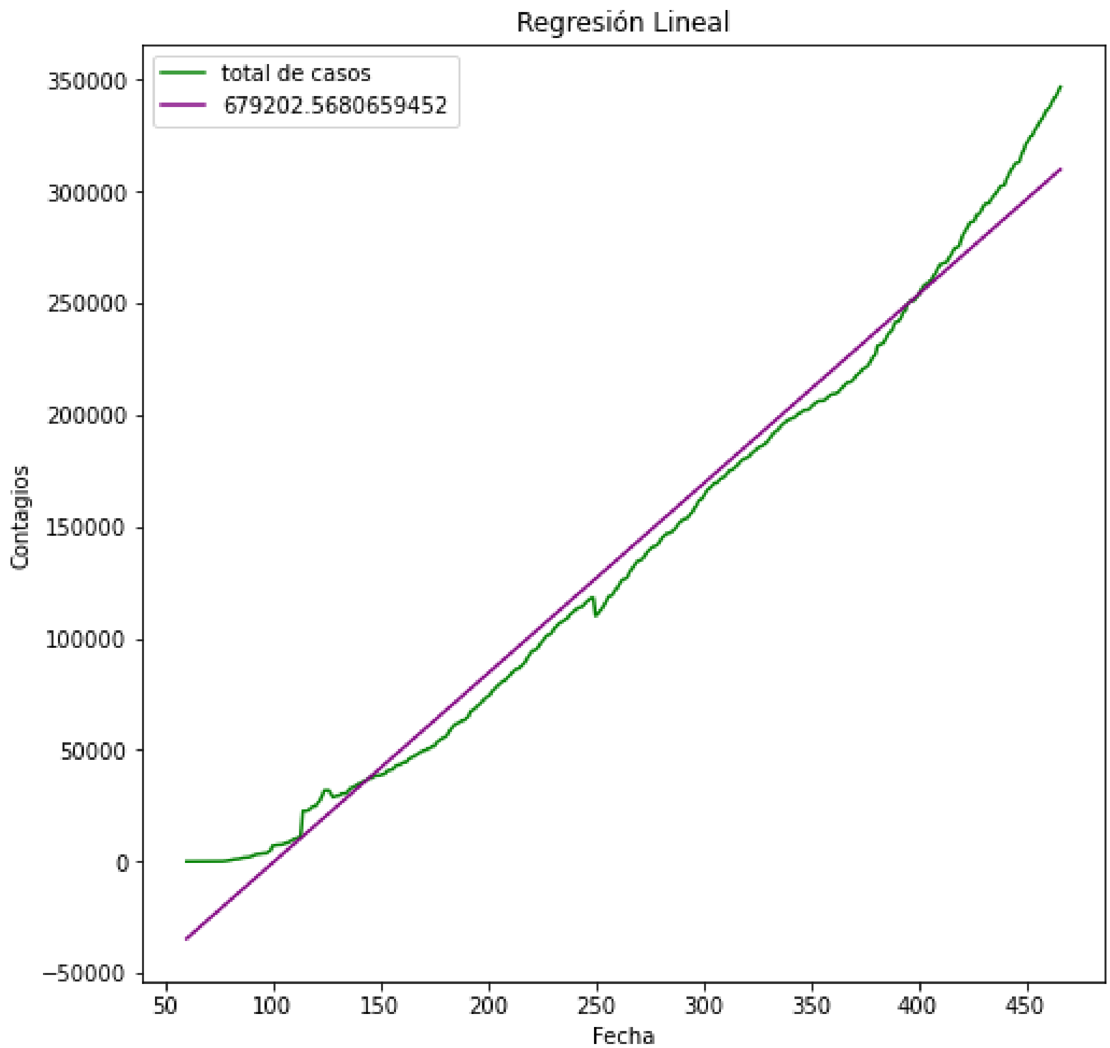
[18]:

```
#Grafica 1

fig = plt.figure(figsize=(8,8))
plt.title('Regresión Lineal')
plt.plot(df['date'],df['total_cases'], label='total de casos', color='green',)
plt.plot(totaldate, date_predicted, color='purple', label=label1)
plt.xlabel('Fecha')
plt.ylabel('Contagios')
plt.legend()
```

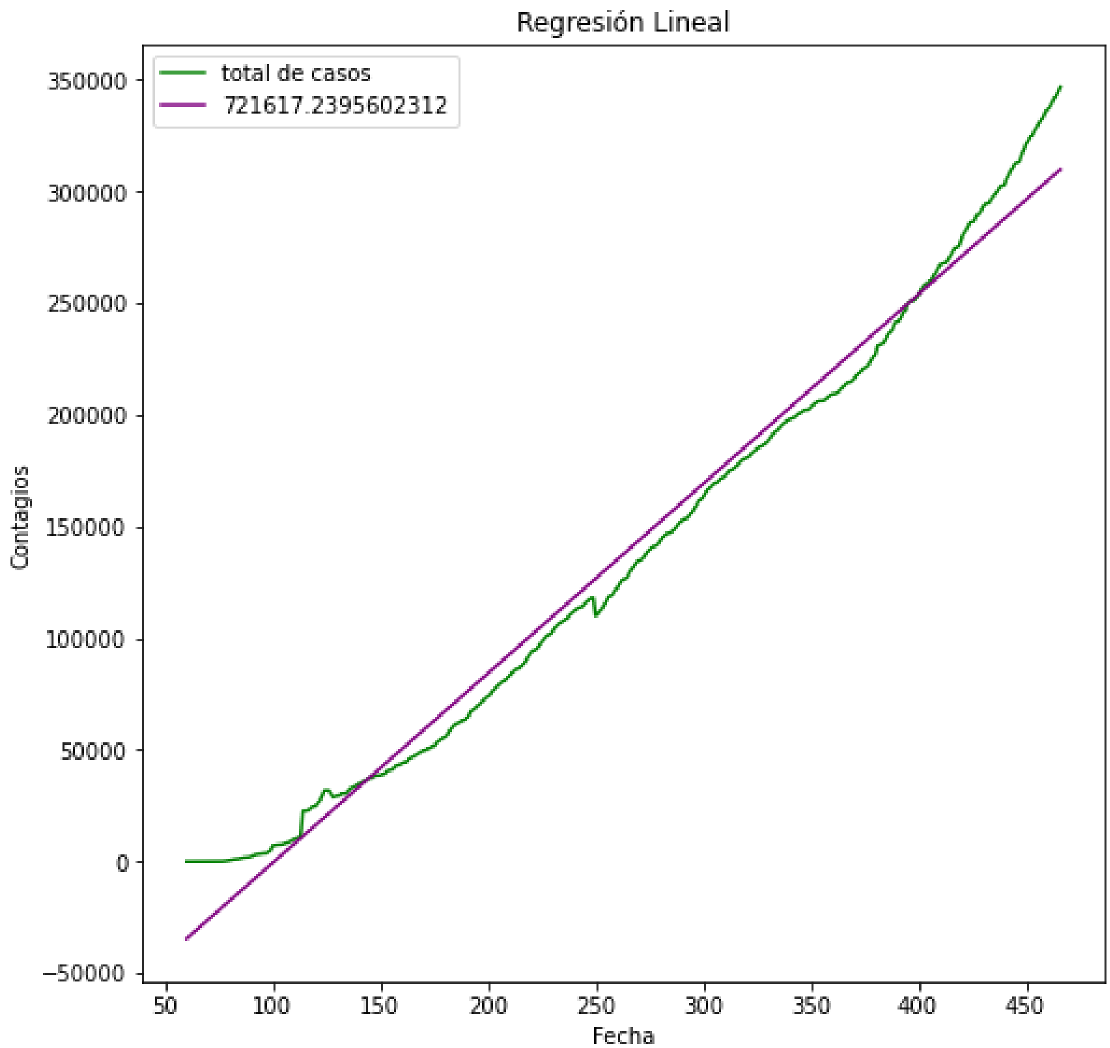
[18]:

```
<matplotlib.legend.Legend at 0x15a0c4fd700>
```




[19]:
```
#Grafica 2
fig = plt.figure(figsize=(8,8))
plt.title('Regresión Lineal')
plt.plot(df['date'],df['total_cases'], label='total de casos', color='green',)
plt.plot(totaldate, date_predicted, color='purple', label=label2)
plt.xlabel('Fecha')
plt.ylabel('Contagios')
plt.legend()
```

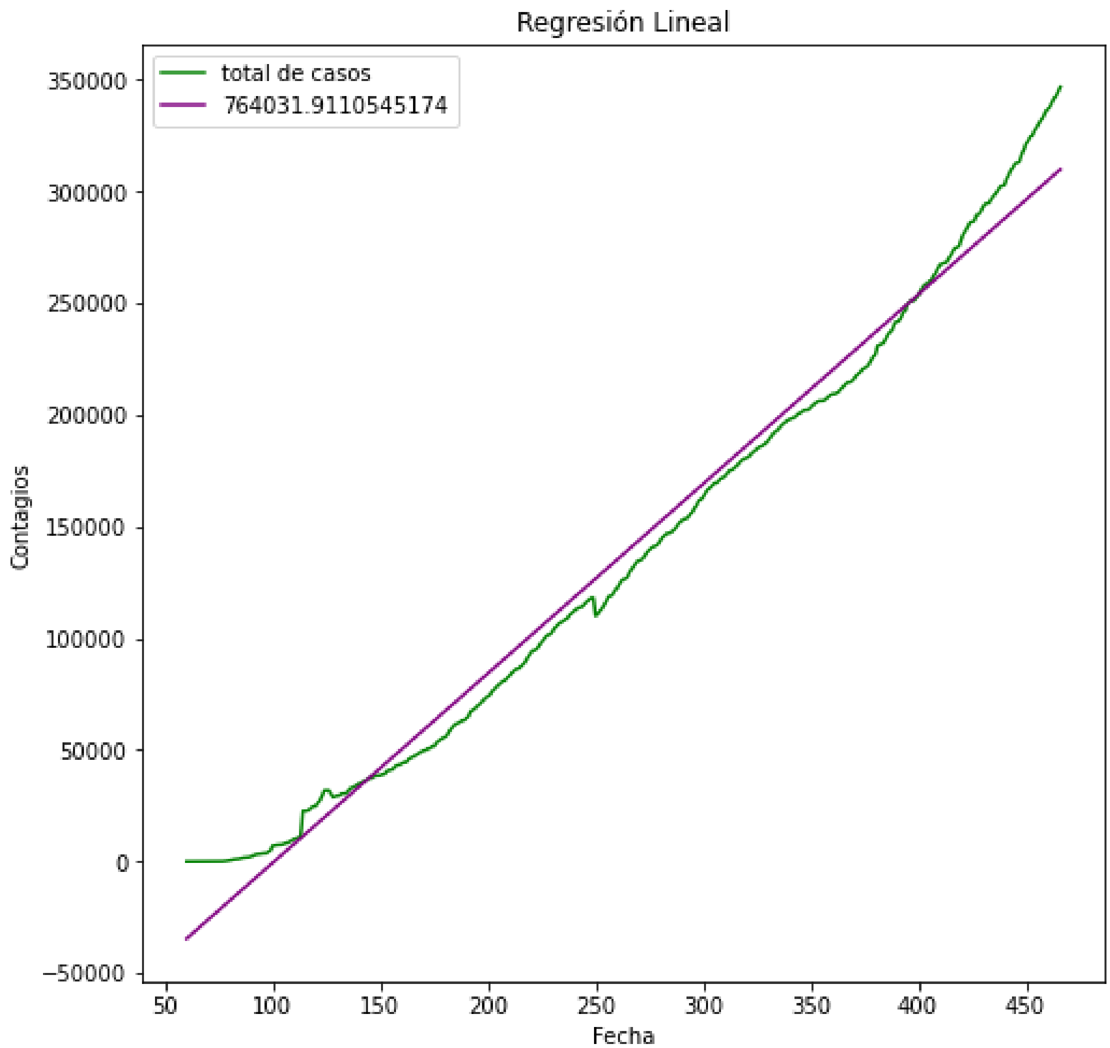
[19]: <matplotlib.legend.Legend at 0x15a0ce62e80>




[20]:
```
#Grafica 3

fig = plt.figure(figsize=(8,8))
plt.title('Regresión Lineal')
plt.plot(df['date'],df['total_cases'], label='total de casos', color='green',)
plt.plot(totaldate, date_predicted, color='purple', label=label3)
plt.xlabel('Fecha')
plt.ylabel('Contagios')
plt.legend()
```

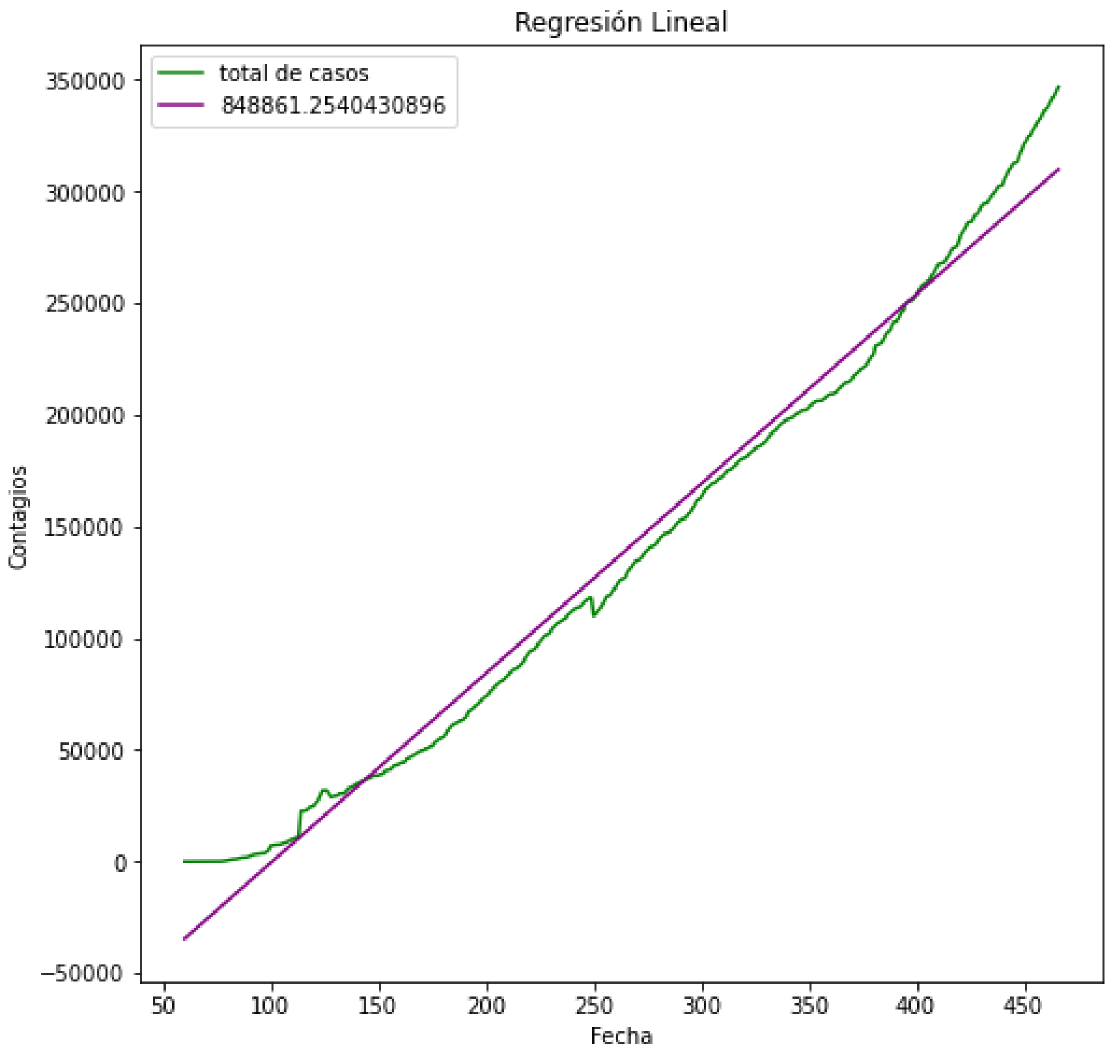
[20]: <matplotlib.legend.Legend at 0x15a0d03b670>




[21]:
```
#Grafica 4

fig = plt.figure(figsize=(8,8))
plt.title('Regresión Lineal')
plt.plot(df['date'],df['total_cases'], label='total de casos', color='green',)
plt.plot(totaldate, date_predicted, color='purple', label=label4)
plt.xlabel('Fecha')
plt.ylabel('Contagios')
plt.legend()
```

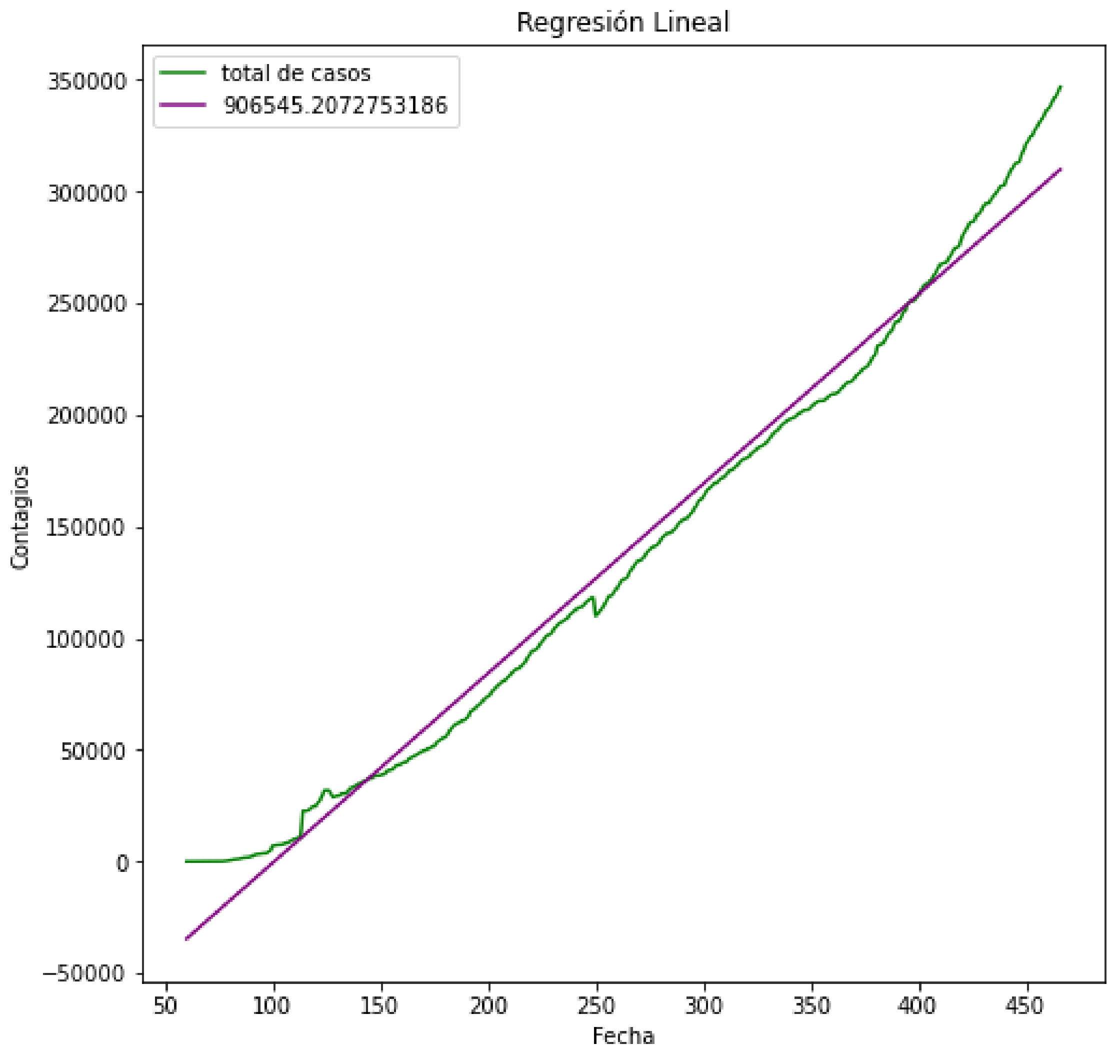
[21]: <matplotlib.legend.Legend at 0x15a0cefea90>




[23]:
```
#Grafica 5

fig = plt.figure(figsize=(8,8))
plt.title('Regresión Lineal')
plt.plot(df['date'],df['total_cases'], label='total de casos', color='green',)
plt.plot(totaldate, date_predicted, color='purple', label=label5)
plt.xlabel('Fecha')
plt.ylabel('Contagios')
plt.legend()
```

[23]: <matplotlib.legend.Legend at 0x15a0cf77250>


Regresión Lineal
total de casos
906545.2072753186
350000
300000
250000
200000
150000
100000
50000
0
−50000
Contagios
50
100
150
200
250
300
350
400
450
Fecha