```
import pandas as pd # Importamos el paquete Pandas
import numpy as np
from datetime import datetime
url = 'https://raw.githubusercontent.com/andrab/ecuacovid/master/datos_crudos/vacun

df_vacuna = pd.read_csv(url, header = None)

df_vacuna.columns = ['fecha','dosis_total','primera_dosis','segunda_dosis']

df_vacuna
```

	fecha	dosis_total	primera_dosis	segunda_dosis
0	fecha	dosis_total	primera_dosis	segunda_dosis
1	21/01/2021	0	0	0
2	22/01/2021	108	108	0
3	27/01/2021	2982	2982	0
4	04/02/2021	6228	6228	0
5	17/02/2021	8190	6228	1962
6	24/02/2021	24492	20784	3708
7	01/03/2021	42114	35886	6228
8	04/03/2021	59316	53088	6228
9	05/03/2021	71148	64920	6228
10	08/03/2021	74472	68244	6228
11	09/03/2021	75258	69030	6228
12	11/03/2021	95915	89687	6228
13	12/03/2021	123176	116948	6228
14	13/03/2021	139359	119222	20137
15	15/03/2021	141191	121054	20137
16	21/03/2021	178970	140765	38205
17	23/03/2021	182261	143614	38647
18	24/03/2021	191179	152526	38653
19	26/03/2021	230770	172413	58357
20	27/03/2021	235000	174642	60358
21	29/03/2021	244866	182329	62537
22	01/04/2021	283106	204902	78204
23	04/04/2021	301069	211720	89349
24	05/04/2021	335093	228504	106589

```
url2 = 'https://raw.githubusercontent.com/andrab/ecuacovid/master/datos_crudos/vacu

df_vacuna_fabricante = pd.read_csv(url2, header = None)

df_vacuna_fabricante.columns = ['vaccine','total','arrived_at']

df_vacuna_fabricante
```

	vaccine	total	arrived_at
0	vaccine	total	arrived_at
1	Pfizer/BioNTech	8190	20/01/2021
2	Pfizer/BioNTech	16380	17/02/2021
3	Pfizer/BioNTech	17550	24/02/2021
4	Pfizer/BioNTech	31590	03/03/2021
5	Sinovac	20000	06/03/2021
6	Pfizer/BioNTech	73710	10/03/2021
7	Oxford/AstraZeneca	84000	17/03/2021
8	Pfizer/BioNTech	62010	17/03/2021
9	Pfizer/BioNTech	65520	24/03/2021
10	Pfizer/BioNTech	66690	31/03/2021
11	Pfizer/BioNTech	53820	05/04/2021
12	Sinovac	300000	07/04/2021
13	Sinovac	700000	10/04/2021
14	Pfizer/BioNTech	53820	14/04/2021
15	Pfizer/BioNTech	54990	21/04/2021
16	Oxford/AstraZeneca	336000	24/04/2021

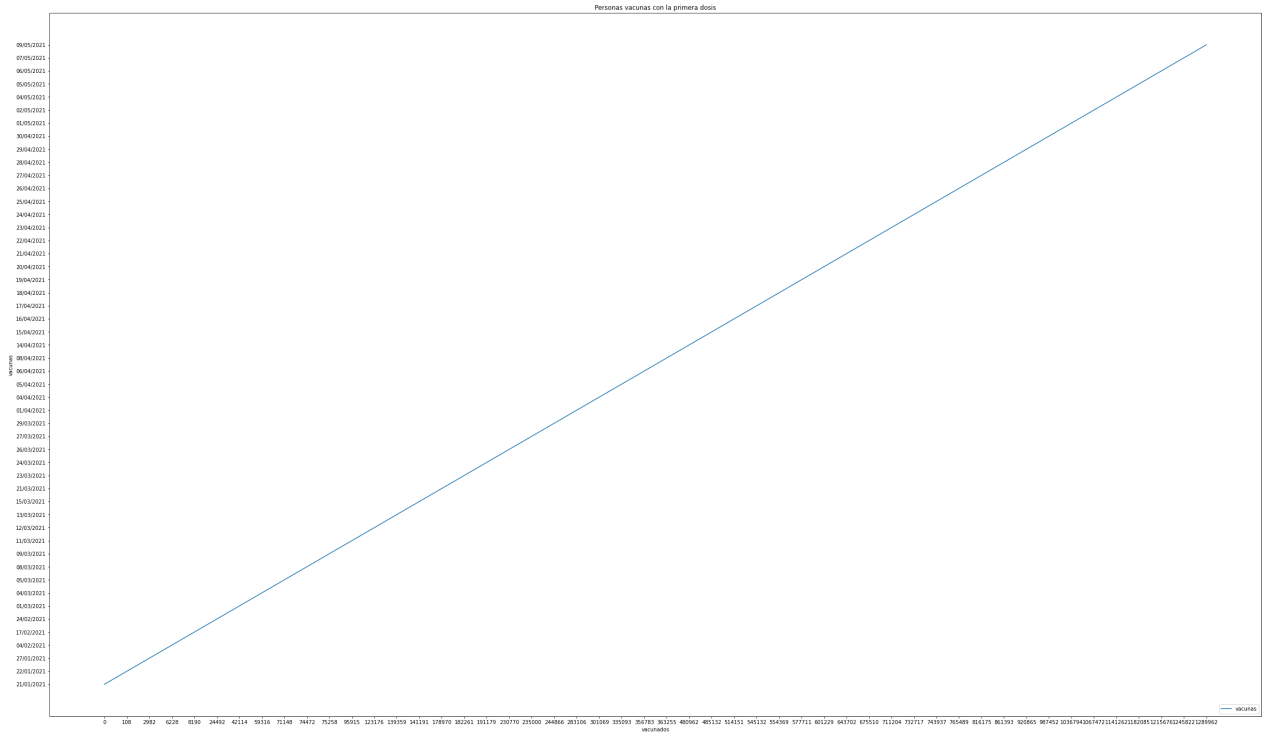
```
url3 = 'https://raw.githubusercontent.com/andrab/ecuacovid/master/datos_crudos/vacu

df_plan_vacuna = pd.read_csv(url3, header = None)

df_plan_vacuna.columns = ['fecha','primera_dosis','segunda_dosis']
```

```
import matplotlib.pyplot as pp
%matplotlib inline

pp.plot(df_vacuna.dosis_total[1:], df_vacuna.fecha[1:].astype(str))
pp.title('Personas vacunas con la primera dosis')
pp.ylabel('vacunas')
pp.xlabel('vacunados')
pp.legend(['vacunas','vacunados'], loc='lower right')
pp.gcf().set_size_inches(42, 25)
pp.show()
```



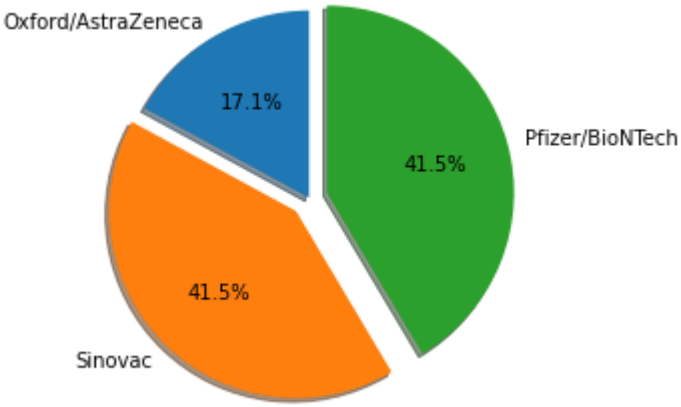

```
vaccin = df_vacuna_fabricante.vaccine[1:]
vacunasNombre= np.array(list(set(vaccin)))
total_vacunas_fabri = df_vacuna_fabricante.total[1:]
#explode = [0 1 1 0]
#pp.pie(df_vacuna_fabricante.vaccine[1:,].value_counts(), explode)
#pp.pie(df_vacuna_fabricante.vaccine.value_counts())
#pp.title('fabricante vacunas')

v1= 0
v2=0
v3=0
for marca, total_mar in zip(vaccin, total_vacunas_fabri):
  if marca == vacunasNombre[0]:
    v1 = v1 + + int(total_mar)
  elif marca == vacunasNombre[1]:
    v2 = v2 + + int(total_mar)
  elif marca == vacunasNombre[2]:
    v3 = v3 + int(total_mar)

b = np.array([v1, v2, v2])
```

```
explode = (0, 0.1, 0.1)
fig1, ax1 = pp.subplots()
ax1.pie(b, explode=explode, labels=vacunasNombre, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')

pp.show()
```



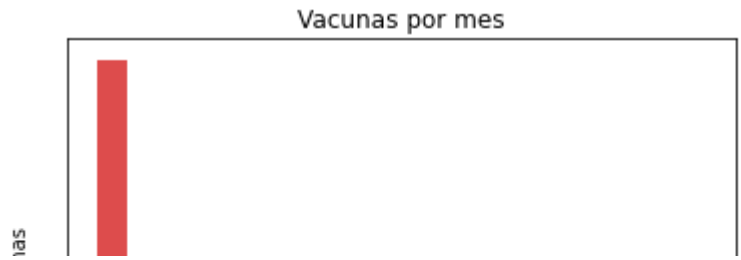

```
datos_va = df_vacuna_fabricante.vaccine[1:]
datos_vacuna = np.array(datos_va, dtype=object)
print(datos_vacuna)
#fecha = np.asarray([df_vacuna_fabricante.arrived_at[1:,]])
#df =  pd.to_datetime(df_vacuna_fabricante.arrived_at[1:], errors='coerce')

fecha_mes = []
for i in df_vacuna_fabricante['arrived_at'][1:]:
  var_Fecha= datetime.strptime(i, '%d/%m/%Y')
  fecha_mes.append(var_Fecha.month)
f = np.array(fecha_mes)
print(f)

#pp.plot( datos_vacuna, f)

pp.hist(x=datos_vacuna, bins= 20, color=['#cf0000'], alpha=0.7, rwidth=1)
pp.title('Vacunas por mes')
pp.xlabel('vacunas')
pp.ylabel('fechas')
pp.xticks(datos_vacuna)
pp.yticks(f)
```

```
['Pfizer/BioNTech' 'Pfizer/BioNTech' 'Pfizer/BioNTech' 'Pfizer/BioNTech'
 'Sinovac' 'Pfizer/BioNTech' 'Oxford/AstraZeneca' 'Pfizer/BioNTech'
 'Pfizer/BioNTech' 'Pfizer/BioNTech' 'Pfizer/BioNTech' 'Sinovac' 'Sinovac'
 'Pfizer/BioNTech' 'Pfizer/BioNTech' 'Oxford/AstraZeneca'
 'Pfizer/BioNTech' 'Pfizer/BioNTech']
[1 2 2 3 3 3 3 3 3 4 4 4 4 4 4 4 5]
([<matplotlib.axis.YTick at 0x7f439bbb7690>,
  <matplotlib.axis.YTick at 0x7f439bbb4e90>,
  <matplotlib.axis.YTick at 0x7f439c285810>,
  <matplotlib.axis.YTick at 0x7f439bb463d0>,
  <matplotlib.axis.YTick at 0x7f439bb3f510>,
  <matplotlib.axis.YTick at 0x7f439bb36710>,
  <matplotlib.axis.YTick at 0x7f439bb46810>,
  <matplotlib.axis.YTick at 0x7f439bb464d0>,
  <matplotlib.axis.YTick at 0x7f439bacf1d0>,
  <matplotlib.axis.YTick at 0x7f439bacf690>,
  <matplotlib.axis.YTick at 0x7f439bacf5d0>,
  <matplotlib.axis.YTick at 0x7f439bad8190>,
  <matplotlib.axis.YTick at 0x7f439bad8650>,
  <matplotlib.axis.YTick at 0x7f439bad8590>,
  <matplotlib.axis.YTick at 0x7f439bad8550>,
  <matplotlib.axis.YTick at 0x7f439bacf890>,
  <matplotlib.axis.YTick at 0x7f439bb46910>,
  <matplotlib.axis.YTick at 0x7f439bae1350>],
 <a list of 18 Text major ticklabel objects>)
```




```
#'#F2AB6D', 'r', 'blue','black','#564a4a', '#9fe6a0', '#ededd0', '#907fa4', '#da7f8
#datos = pd.Series(df_vacuna_fabricante.arrived_at[1:,]) # cargamos los datos en un
fecha_mes = []
for i in df_vacuna_fabricante['arrived_at'][1:]:
  var_Fecha= datetime.strptime(i, '%d/%m/%Y')
  fecha_mes.append(var_Fecha.month)
f = np.array(fecha_mes)
#intervalos = pd.Series(df_vacuna_fabricante.vaccine[1:,].astype(str)) # calculamos
#fd = df_vacuna_fabricante['vaccine'][1:]
#datos = np.array(fd)
pp.hist(x=f, bins=7, color=['#F2AB6D'],alpha=0.7, rwidth=.2)
pp.title(' Fecha de llegada de la vacuna')
pp.xlabel('fechas')
pp.xticks(f)
pp.show()
```

Regresion Lineal total de vacunados

```
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


#dat_br = np.array(lista_total_fechasbr)
#print(dat_br)
#print("-------")
#tot_br = np.array(lista_total_totbr, dtype='float')
#xbr = np.arange(1, len(tot_br) + 1, 1)

#Segundo metodo
totalVacunas_ecuador = df_vacuna.loc[:,['primera_dosis', 'segunda_dosis']]
#graficas de las vacunas
#pp.scatter(totalVacunas_ecuador['primera_dosis'][1:], totalVacunas_ecuador['segund
#pp.bar(totalVacunas_ecuador['primera_dosis'][1:], totalVacunas_ecuador['segunda_do
#pp.gcf().set_size_inches(42, 25)
#pp.xlabel('Primera Dosis')
#pp.ylabel('Segunda Dosis')
#pp.title('Gráfica vacunados totales')

total_llegada_fabricantes= df_vacuna_fabricante.loc[:,['total', 'arrived_at']]
```

	total	arrived_at
0	total	arrived_at
1	8190	20/01/2021
2	16380	17/02/2021
3	17550	24/02/2021
4	31590	03/03/2021
5	20000	06/03/2021
6	73710	10/03/2021
7	84000	17/03/2021
8	62010	17/03/2021

```
#############
xvaac=  (totalVacunas_ecuador.primera_dosis[1:].astype(np.int64))
#xvac = list (totalVacunas_ecuador.iloc [:, 0][1:])
#yvac = list (totalVacunas_ecuador.iloc [:, 1][1:])
yvaac=  (totalVacunas_ecuador.segunda_dosis[1:].astype(np.int64))

x_fabri = (total_llegada_fabricantes.total[1:].astype(np.int64))

FMT = '%d/%m/%y'
date = total_llegada_fabricantes['arrived_at']
total_llegada_fabricantes['arrived_at'] = date.map(lambda x : (datetime.strptime(x,

total_llegada_fabricantes


# Creamos el objeto de Regresión Lineal
regr_ecuador2 = linear_model.LinearRegression()
regr_fabricante = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regr_ecuador2.fit(np.array(xvaac).reshape(-1, 1) ,yvaac)
regr_fabricante.fit(np.array(x_fabri).reshape(-1, 1) ,y_fabri)

print('vacunados primera y segunda')
print(' ')
# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients Vacunados: \n', regr_ecuador2.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term Vacunados: \n', regr_ecuador2.intercept_)
# Error Cuadrado Medio
y_prediccion2 = regr_ecuador2.predict([[100]])
print('prediccion vacunados-->',int(y_prediccion2))
print(' ')

print('fecha llegada vacunas')
print(' ')
```

```
print('Coefficients Fabricantes: \n', regr_fabricante.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term Fabricantes: \n', regr_fabricante.intercept_)
# Error Cuadrado Medio
y_prediccion_fabri = regr_fabricante.predict([[100]])
print('prediccion Fabricantes-->',int(y_prediccion_fabri))
print(' ')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-67-398cdb5f28ff> in <module>()
     10 FMT = '%d/%m/%y'
     11 date = total_llegada_fabricantes['arrived_at']
---> 12 total_llegada_fabricantes['arrived_at'] = date.map(lambda x :
(datetime.strptime(x, FMT) - datetime.strptime("20/01/2021", FMT)).days)
     13
     14 total_llegada_fabricantes

                                  4 frames
pandas/_libs/lib.pyx in pandas._libs.lib.map_infer()

/usr/lib/python3.7/_strptime.py in _strptime(data_string, format)
    357     if not found:
    358         raise ValueError("time data %r does not match format %r" %
--> 359                          (data_string, format))
    360     if len(data_string) != found.end():
    361         raise ValueError("unconverted data remains: %s" %

ValueError: time data 'arrived_at' does not match format '%d/%m/%y'

SEARCH STACK OVERFLOW
```

```
#########
pp.scatter(xvaac, yvaac)
x_real = np.array(range(0,899000))
print(x_real)
pp.plot(x_real, regr_ecuador2.predict(x_real.reshape(-1, 1)), color='green', lw=2)

pp.show()
```

```
[     0      1      2 ... 898997 898998 898999]
```

Regresión logistico

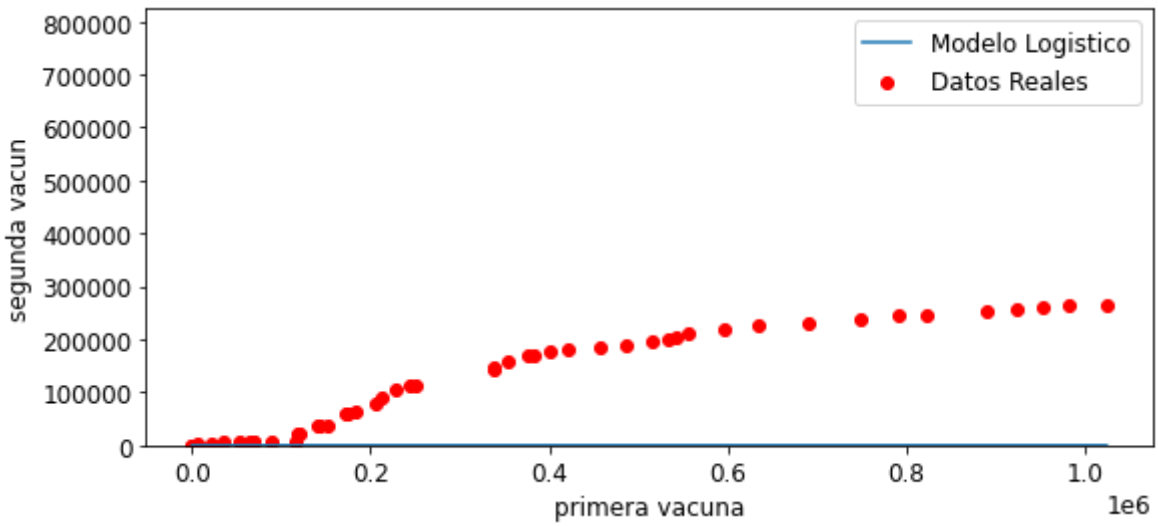
```
def modelo_logistic(xx,aa,bb):
    return aa+bb*np.log(xx)

exp_fitt = curve_fit(modelo_logistic,xvaac,yvaac)
print(exp_fitt)
```

```
(array([1., 1.]), array([[inf, inf],
       [inf, inf]]))
/usr/local/lib/python3.7/dist-packages/pandas/core/series.py:726: RuntimeWarni
  result = getattr(ufunc, method)(*inputs, **kwargs)
/usr/local/lib/python3.7/dist-packages/scipy/optimize/minpack.py:808: Optimize
  category=OptimizeWarning)
```

```
pred_x = list(range(min(xvaac),max(xvaac)+100)) # Predecir 50 dias mas
pp.rcParams['figure.figsize'] = [9, 4]
pp.rc('font', size=12)
# Real data
pp.scatter(xvaac,yvaac,label="Datos Reales",color="red")
# Predicted exponential curve
pp.plot(pred_x, [modelo_logistic(i,exp_fitt[0][0],exp_fitt[0][1]) for i in pred_x],
pp.legend()
pp.xlabel("primera vacuna")
pp.ylabel("segunda vacun")
pp.ylim((min(yvaac)*0.9,max(yvaac)*3.1)) # Definir los limites de Y
pp.show()
```

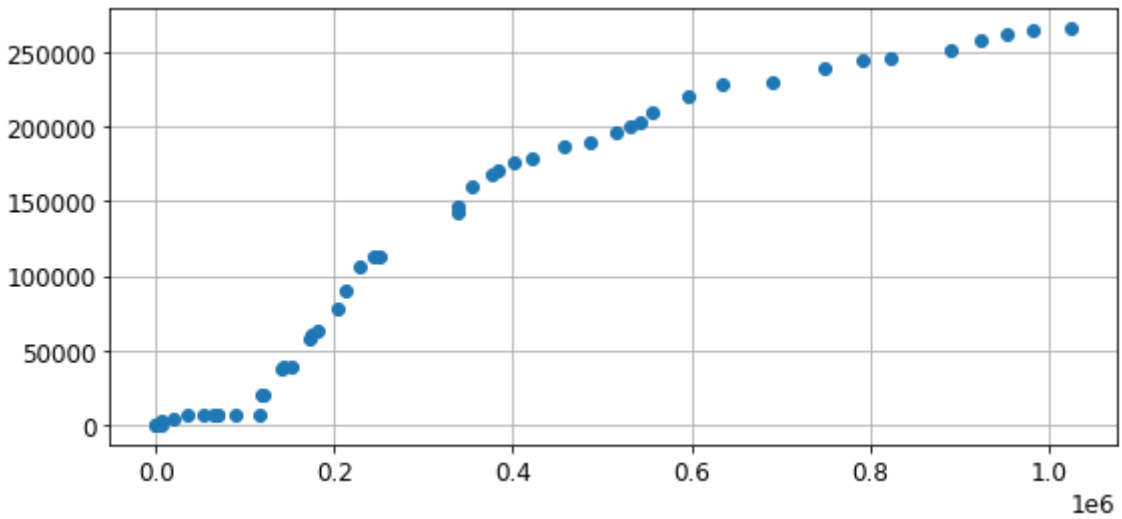
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarn
```



Modelo regresión exponencial

```
curve_fit = np.polyfit(xvaac, np.log(yvaac), deg=1)
print(curve_fit)
pred_x = np.array(list(range(min(xvaac), max(xvaac)+15)))
yx = np.exp(curve_fit[1]) * np.exp(curve_fit[0]*pred_x)
pp.plot(xvaac,yvaac,"o")
pp.plot(pred_x,yx, color="gold")
pp.grid(True)
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/series.py:726: RuntimeWa
  result = getattr(ufunc, method)(*inputs, **kwargs)
[nan nan]
```

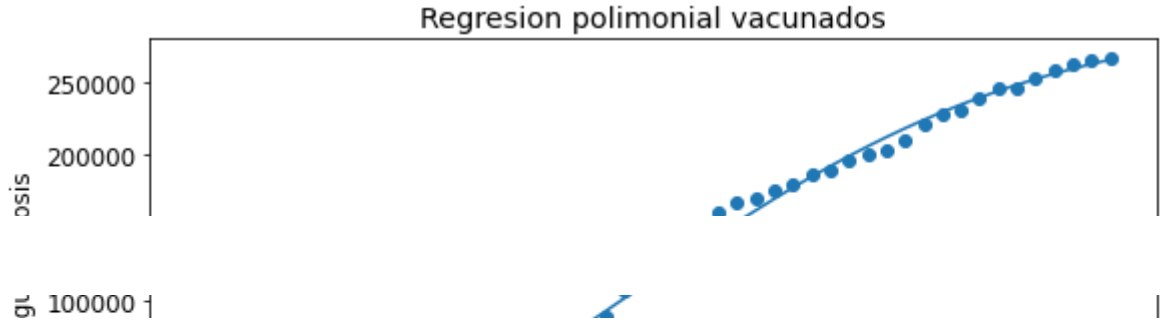


Regresión Polinomial

```
print(len(xvaac))
longitud_dosis = np.array(range(1, len(xvaac)+1))
print(len(longitud_dosis))

x_polimonial = np.poly1d(np.polyfit(longitud_dosis, yvaac, 4))
print('datos', x_polimonial)
y_p = x_polimonial(longitud_dosis)
print(len(y_p))
pp.scatter(longitud_dosis, yvaac)
pp.plot(longitud_dosis, y_p)
pp.title('Regresion polimonial vacunados')
pp.xlabel('primera dosis')
pp.ylabel('segunda dosis')
```

```
50
50
datos          4          3          2
0.1036 x - 16.57 x + 840.1 x - 8612 x + 1.964e+04
50
Text(0, 0.5, 'segunda dosis')
```



Comparacion entre paises

```
urlPaises = 'https://raw.githubusercontent.com/owid/covid-19-data/master/public/dat
df_vacuna_paises = pd.read_csv(urlPaises, header= None).fillna(0)
df_vacuna_paises.columns = ['location','iso_code','date','total_vaccinations','peop



df_vacuna_paises = df_vacuna_paises[df_vacuna_paises['location'].isin(['Ecuador'])]
df_vacuna_paises = df_vacuna_paises.loc[:,['date', 'people_fully_vaccinated']]
FMT = '%Y-%m-%d'
date = df_vacuna_paises['date']

df_vacuna_paises['date'] = date.map(lambda x : (datetime.strptime(x, FMT) - datetim
date


x = list(df_vacuna_paises.iloc [:, 0]) # Fecha
y = list(df_vacuna_paises.iloc [:, 1]) # Numero de casos

print(x)
regr = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regr.fit(np.array(x).reshape(-1, 1) ,y)

# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients: \n', regr.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term: \n', regr.intercept_)
# Error Cuadrado Medio
######
y_prediccion = regr.predict([[100]])
print('prediccion --->',int(y_prediccion))
```

```
###'
pp.scatter(x, y, c='red', alpha=0.9)
x_real = np.array(range(50, 100))
#print(x_real)
pp.gcf().set_size_inches(42, 25)
pp.plot(x_real, regr.predict(x_real.reshape(-1, 1)), color='blue')
pp.title('Análisis vacunados ecuador')
pp.xlabel('fecha de vacunacion')
pp.ylabel('total de vacunados')
```

AHOA AQUI BRASIL

```
urlPaises = 'https://raw.githubusercontent.com/owid/covid-19-data/master/public/dat
df_vacuna_paises = pd.read_csv(urlPaises, header= None).fillna(0)
df_vacuna_paises.columns = ['location','iso_code','date','total_vaccinations','peop
```

```
br_fecha = []
br_vacunados = []
for lugar, fecha, vacuna in zip(df_vacuna_paises['location'], df_vacuna_paises['dat
  if lugar == 'Brazil':
    ff = datetime.strptime(fecha, '%Y-%m-%d')
    br_fecha.append(ff.day)
    br_vacunados.append(vacuna)
fecha_vacuna_br = np.array(br_fecha)
vacunados_total_br = np.array(br_vacunados, dtype='float')
btotal = np.arange(1, len(vacunados_total_br) + 1, 1)

btotal
```

```
#xB = list(fecha_vacuna_br[0:]) # Fecha
#yB = list(vacunados_total_br[0:]) # Numero de casos
#print(xB)
#print('vacunados-->' ,yB)
```

```
# Creamos el objeto de Regresión Lineal
regrB = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regrB.fit(np.array(btotal).reshape(-1, 1) ,vacunados_total_br)

# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients br: \n', regrB.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term br: \n', regrB.intercept_)
# Error Cuadrado Medio
######
y_prediccionB = regrB.predict([[100]])
print('prediccion Brasil --->',int(y_prediccionB))
```

```
print('prediccion Brasil ---> ',int(y_prediccionB))
```

```
fig, (ax1) = pp.subplots(1, sharex='col', sharey='row',
                         gridspec_kw={'hspace': 0, 'wspace': 0}, figsize=(20,20))
x_realB = np.array(range(1, len(vacunados_total_br) + 1))
ax1.scatter(btotal, vacunados_total_br, c='red', alpha=0.9)
```

```
pp.plot(x_realB, regrB.predict(x_realB.reshape(-1, 1)), color='#4aa96c')
pp.title('Análisis vacunados Brasil')
pp.xlabel('fecha de vacunacion')
pp.ylabel('total de vacunados')
#x1.gcf().set_size_inches(42, 25)
```

CHILE

```
urlPaises = 'https://raw.githubusercontent.com/owid/covid-19-data/master/public/dat
df_vacuna_paises = pd.read_csv(urlPaises, header= None).fillna(0)
df_vacuna_paises.columns = ['location','iso_code','date','total_vaccinations','peop
```

```
ch_fecha = []
ch_vacunados = []
for lugarc, fechac, vacunac in zip(df_vacuna_paises['location'], df_vacuna_paises['
  if lugarc == 'Chile':
    ff = datetime.strptime(fechac, '%Y-%m-%d')
    ch_fecha.append(ff.day)
    ch_vacunados.append(vacunac)
fecha_vacuna_chile = np.array(ch_fecha)
vacunados_total_chile = np.array(ch_vacunados, dtype='float')
total_vacunaC= np.arange(1, len(vacunados_total_chile) + 1, 1)
```

```
xC = list(fecha_vacuna_chile) # Fecha
yC = list(vacunados_total_chile) # Numero de casos
print(xC)
# Creamos el objeto de Regresión Lineal
regrC = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regrC.fit(np.array(xC).reshape(-1, 1) ,yC)

# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients: \n', regrC.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term: \n', regrC.intercept_)
# Error Cuadrado Medio
######
y_prediccionC = regrC.predict([[100]])
print('prediccion Chile --->',int(y_prediccionC))
```

```
print('prediccion chile ',len(y_predicciones))


pp.scatter(xC, yC, c='red', alpha=0.9)
x_realC = np.array(range(1, len(vacunados_total_chile) + 1))
#print(x_real)
pp.gcf().set_size_inches(42, 25)
pp.plot(x_realC, regrC.predict(x_realC.reshape(-1, 1)), color='black')
pp.title('Análisis vacunados Chile')
pp.xlabel('fecha de vacunacion')
pp.ylabel('total de vacunados')
```