Tarea #2 Inteligencia Artificial

1. Averiguar qué es el "coefficient of determination"

R/= El coeficiente de determinación es la proporción de la varianza total de la variable explicada por la regresión. El coeficiente de determinación, también llamado R cuadrado, refleja la bondad del ajuste de un modelo a la variable que pretender explicar.

2. Intervalo de valores que toma y explicarlos

R/= Es importante saber que el resultado del coeficiente de determinación oscila entre 0 y 1. Cuanto más cerca de 1 se sitúe su valor, mayor será el ajuste del modelo a la variable que estamos intentando explicar. De forma inversa, cuanto más cerca de cero, menos ajustado estará el modelo y, por tanto, menos fiable será.

3. ¿Qué significa si da negativo o cero? puede pasar? antes de elevar por supuesto.

R/= Si da 0, significa que el modelo no es fiable. Si da negativo, significa que la predicción tiende a ser menos precisa que el valor promedio de los datos dados en el tiempo.

4. Hallar el R cuadrado a nuestros datos de resistencia obtenidos a partir de una hipótesis lineal de 2 parámetros.

R/=

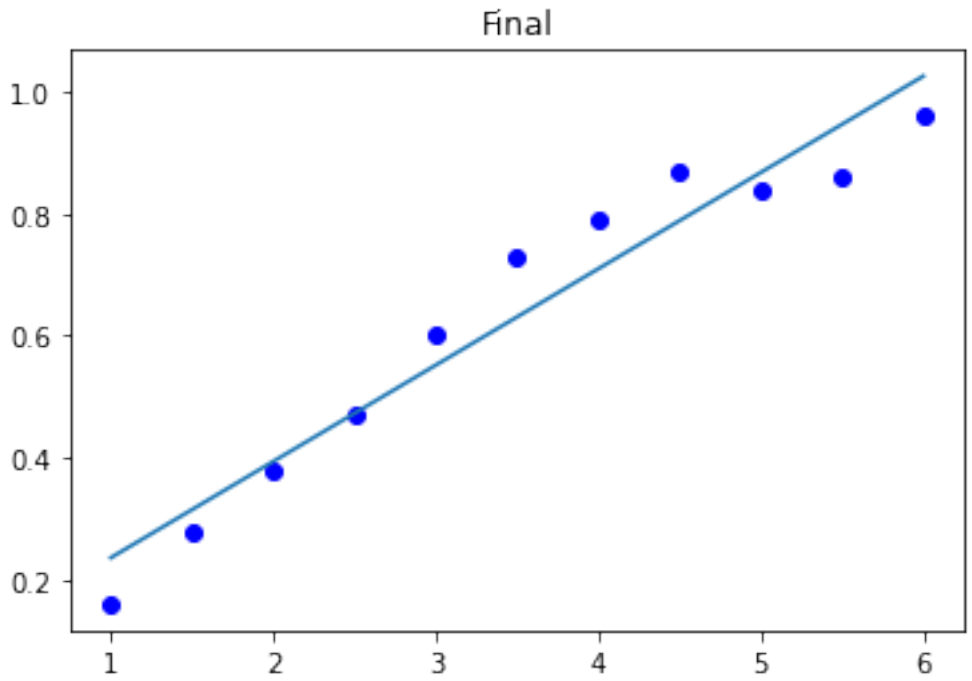
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import seaborn as sns
sns.set(style='whitegrid')
```

Iniciemos con la regresión lineal univariada

```
x1=np.array([1, 1.5, 2, 2.5, 3, 3.5 , 4, 4.5, 5, 5.5, 6])
x0=np.ones(x1.shape)
y=np.array([0.16,0.28,0.38,0.47,0.60,0.73,0.79,0.87,0.84,0.86,0.96])
X=np.matrix([x0,x1]).T
Y=np.matrix([y]).T
print("shape x0",x0.shape,"shape x1",x1.shape,"shape X",X.shape,"shape Y"
,Y.shape)
print(X)
print(Y)
```

```
Theta=np.linalg.inv(X.T*X)*(X.T)*Y
```

```
print(Theta)
plt.plot(x1,y, 'bo')
plt.plot(x1,Theta[0,0]+Theta[1,0]*x1)
plt.title("Final")
plt.show()
```



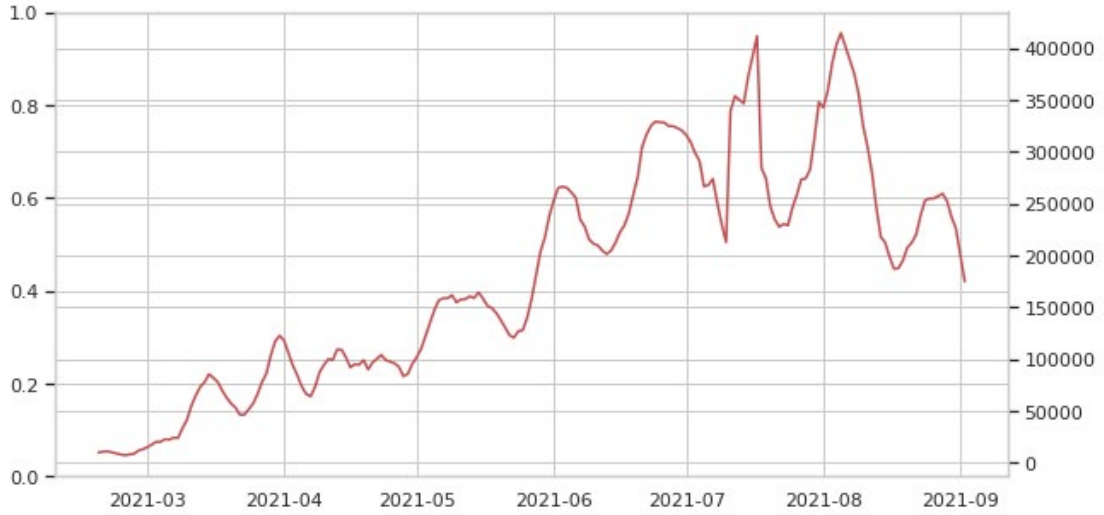

```
y_est=x0*Theta[0,0]+x1*Theta[1,0]
print(y_est)
varianza=y.var()
#print(varianza)
R0 = np.sum( np.power( ( y_est-varianza ) ,2) )
R1 = np.sum( np.power( ( y-varianza ) ,2) )
R2 = R0/R1
print("R cuadrado = ",R2)
```

```
R cuadrado = 0.9893803623391579
```

Como se puede observar el R cuadrado obtenido es un valor cercano a 1, lo cual nos indica que nuestro modelo es fiable.

5. Realizar por lo menos 3 regresiones para los datos de vacunación de Colombia, X son los días desde la primera vacuna y Y el número de vacunas puestas ese día. Suponga usted varias funciones de hipótesis, mida usando R cuadrado para diferentes funciones de hipótesis

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import seaborn as sns
sns.set(style='whitegrid')
import pandas as pd
Selected_country_2 = 'Colombia'
df2 = pd.read_csv('country_vaccinations.csv',header=0)
df2 = df2[df2['country']==Selected_country_2]
df2.index = pd.to_datetime(df2['date'])
fig,ax=plt.subplots(figsize=(10,5))
ax1 = ax.twinx()
ax1.plot(df2.index,df2['daily_vaccinations'],'r',label='Vaccinations')
plt.show()
vaccination.describe()
eti = df2['daily_vaccinations'].values
eti = np.delete(eti,0)
#print(eti)
car = df2['date'].values
eti = np.delete(eti,0)
car = np.delete(car,0)
#print(car)
#print(type(car))
#plt.scatter(car,eti)
#plt.show()
#print(car.shape)
#print(eti.shape)
```

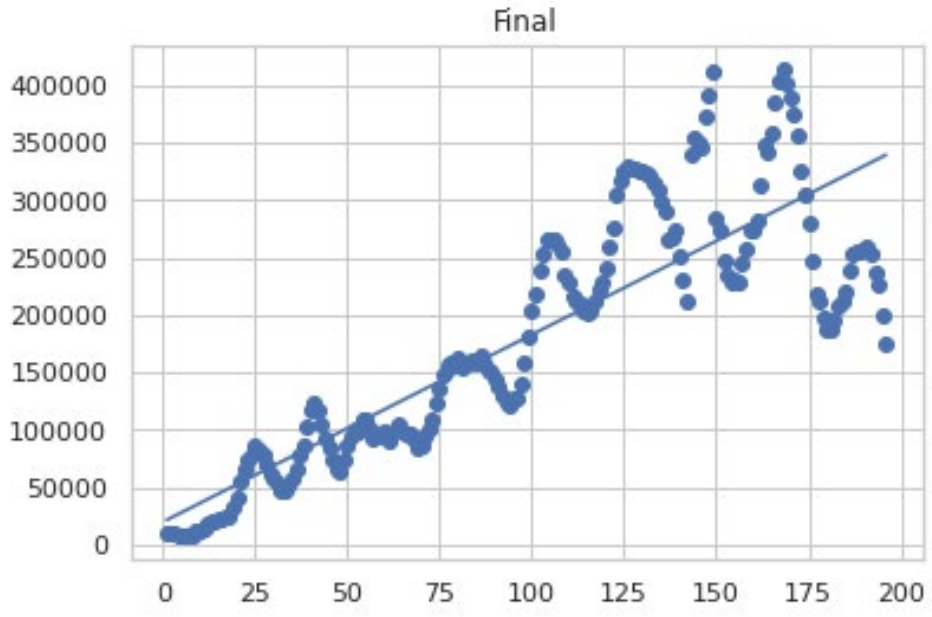


```
x0 = np.ones(196)
print(x0.shape)
dias = ([])
for i in range (1,197):
  dias.append(i)
dia = np.array(dias)
#print(dias)
print(type(dia))
print(x0.shape)
X = np.matrix([x0,dia]).T
Y = np.matrix([eti]).T
print("shape x0",x0.shape,"shape x1",car.shape,"shape X",X.shape,"shape Y
",Y.shape)
print(X)
print(Y)
```

Probamos primero con una hipótesis lineal de la forma $y = \theta_0 * X_0 + \theta_1 * X_1$

```
Theta = np.linalg.inv(X.T*X)*(X.T)*Y
print(Theta)
plt.plot(dia,eti, 'bo')
plt.plot(dia,Theta[0,0]+Theta[1,0]*dia)
plt.title("Final")
plt.show()
```

```
[[19846.90078493]
 [ 1629.87099586]]
```



```
y_est = x0*Theta[0,0]+dia*Theta[1,0]
varianza = eti.var()
print("Varianza = ",varianza)
R0 = np.sum( np.power( ( y_est-varianza ) ,2) )
R1 = np.sum( np.power( ( eti-varianza ) ,2) )
R2 = R0/R1
print("R cuadrado = ",R2)
```

```
Varianza =  11577327852.768536
R cuadrado =  0.9999999999770703
```

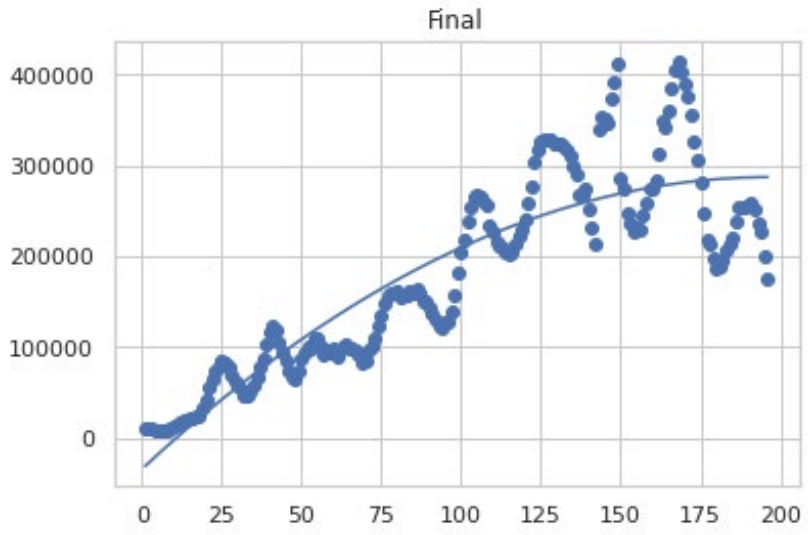
```
y_est=1*Theta[0,0]+198*Theta[1,0]
print("El número de vacunas que se espera se apliquen el 03/09/2021 son "
, y_est)
```

```
El número de vacunas que se espera se apliquen el 02/09/2021 son 342561.3
579660369
```

Ahora probamos con una polinomial de segundo orden $y = \theta_0 * X_0 + \theta_1 * X_1 + \theta_2 * X_1^2$

```
Z1 = dia*dia
X = np.matrix([x0,dia,Z1]).T
Y = np.matrix([eti]).T
Theta = np.linalg.inv(X.T*X)*(X.T)*Y
print(Theta)
plt.plot(dia,eti, 'bo')
plt.plot(dia,Theta[0,0]+Theta[1,0]*dia+Z1*Theta[2,0])
plt.title("Final")
plt.show()
```

```
[[-3.41888749e+04]
 [ 3.26731874e+03]
 [-8.31191750e+00]]
```




```
y_est = x0*Theta[0,0]+dia*Theta[1,0]+Z1*Theta[2,0]
#print(y_est)
#print(y_est.shape)
varianza = eti.var()
print("Varianza = ",varianza)
R0 = np.sum( np.power( ( y_est-varianza ) ,2) )
R1 = np.sum( np.power( ( eti-varianza ) ,2) )
R2 = R0/R1
print("R cuadrado = ",R2)
```

```
Varianza =  11577327852.768536
R cuadrado =  0.999999999981296
```

```
y_est=1*Theta[0,0]+198*Theta[1,0]+39204*Theta[2,0]
print("El numero de vacunas que se espera se apliquen el 03/09/2021 son "
,y_est)
```

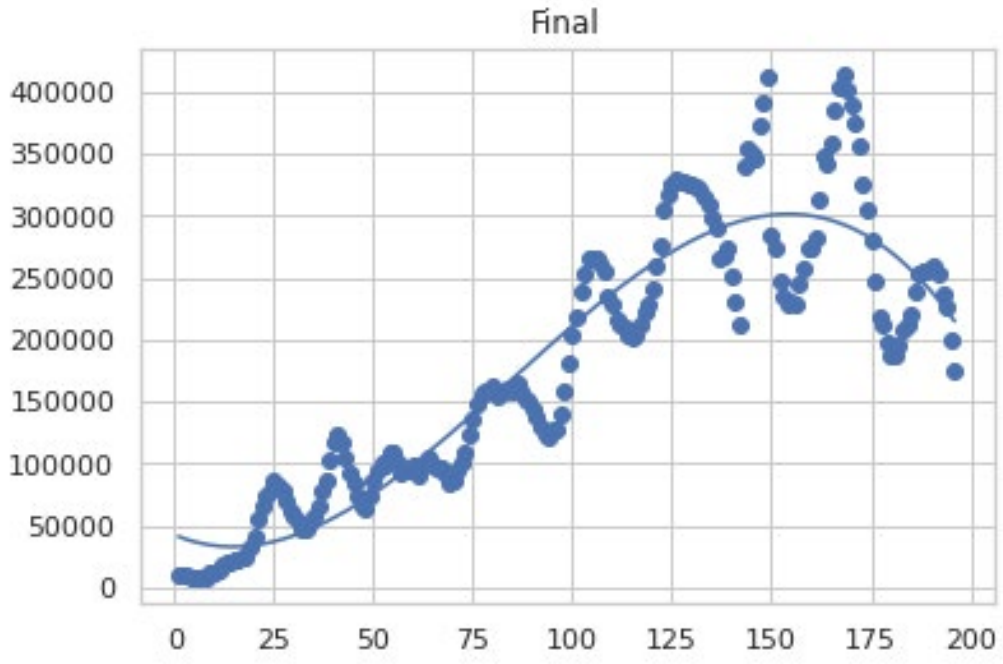
```
El número de vacunas que se espera se apliquen el 02/09/2021 son 286879.8
226538688
```

Por último, probamos con una hipótesis polinomial de tercer orden $y = \theta_0 * X_0 + \theta_1 * X_1 + \theta_2 * X_1^2 + \theta_3 * X_1^3$

```
Z1 = dia*dia
Z2 = Z1*dia
X = np.matrix([x0,dia,Z1,Z2]).T
Y = np.matrix([eti]).T
Theta = np.linalg.inv(X.T*X)*(X.T)*Y
print(Theta)
plt.plot(dia,eti, 'bo')
plt.plot(dia,Theta[0,0]+Theta[1,0]*dia+Z1*Theta[2,0]+Z2*Theta[3,0])
```

```
plt.title("Final")
plt.show()
```

```
[[ 4.24164516e+04]
 [-1.34047221e+03]
 [ 5.00140506e+01]
 [-1.97380603e-01]]
```




```
y_est = x0*Theta[0,0]+dia*Theta[1,0]+Z1*Theta[2,0]+Z2*Theta[3,0]
#print(y_est)
#print(y_est.shape)
varianza = eti.var()
print("Varianza = ",varianza)
R0 = np.sum( np.power( ( y_est-varianza ) ,2) )
R1 = np.sum( np.power( ( eti-varianza ) ,2) )
R2 = R0/R1
print("R cuadrado = ",R2)
```

```
Varianza =  11577327852.768536
R cuadrado =  0.9999999999871794
```

```
y_est=1*Theta[0,0]+198*Theta[1,0]+39204*Theta[2,0]+7762392*Theta[3,0]
print("El número de vacunas que se espera se apliquen el 03/09/2021 son "
, y_est)
```

```
El número de vacunas que se espera se apliquen el 02/09/2021 son 205608.1
8188786064
```

Analizando los resultados obtenidos de las diferentes hipótesis con el R cuadrado nos damos cuenta de que las 3 tienen una buena fiabilidad, son buenos modelos, sin embargo, la hipótesis de tercer orden da un mejor resultado, más cercano a 1. Para comprobar mas allá y poner a prueba el modelo se intenta predecir cuantas serán las dosis aplicadas para el 2 de septiembre que según datos del ministerio de salud fueron 22.6914

Es por ello que nuevamente se comprueba que el modelo de tercer orden es el mejor ya que es el que más se acerca y de una buena manera al valor real, alrededor de 20mil vacunas por debajo además se empieza a evidenciar que un modelo como el lineal de primer orden a pesar de tener un buen R cuadrado empieza a tener problemas para predecir a estas alturas de la vacunación, ya que como con los contagios hay un momento en que empiezan a disminuir las dosis aplicas diarias.