Resumen IA FUNDAMENTALS

Regresion Lineal:

- La Regresion Cuantifica la Naturaleza de la Relacion
- La Correlacion mida la Fuerza de Asociacion entre las 2 Variables

Conceptos Basicos:

- Response (Dependent Variable, "Y" Variable, Target o Salida) : Variable a predecir

- Independent Variable (Feature, Atributo, "X" Variable) : La Variable que usamos para predecir la Respuesta

- Record (Row, case, instance, example): El vector de variable predictorias y los Valores de salida de un individuo espcifico o caso
- Intercept(β_0, Ordenada a la Origen) : La intercepcion de la linea de Regresion, es decir, el valor pronosticado
- Regresion Coeficient (slope , β_1 ,weights , peso) : la Pendiente de la linea de Regresion
- Fitted Values: los estimados obtenidos de la linea de Regresion
- Residuals (Errors) : La diferencia entre los valores observados y los valores Ajustados (Fitted Values)
- Least squares(Minimos Cuadrados) : El Metodo de ajustar una regresion minimizando la suma de residuos

Ecuacion General de la Regresion Lineal Simple

$$Y = \beta_0 + \beta_1 X$$

- β_0 --> Intercept o interseccion
- β_1 --> Slope o Pendiente (Coeficiente de Regresion)
- Y --> Variable Dependiente
- X --> Vector de Caracteristicas (Variable Predictora)

Primer modelo de Regresion Lineal Simple

In [1]:
```
#importamos las Libreias
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model
import numpy as np
```

In [2]:
```
#Cargamos los datos en la variable "data"
data=pd.read_csv('datasets\LungDisease.csv')
data
```

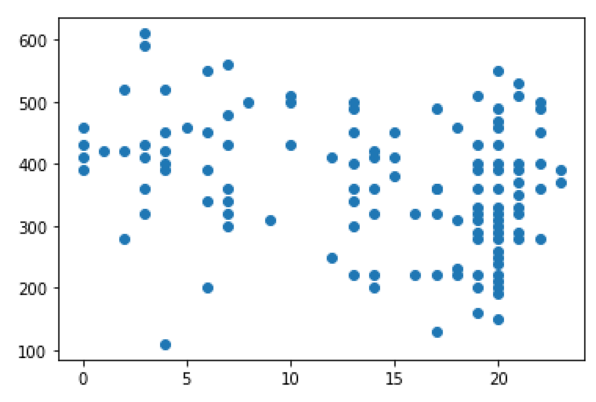
Out[2]:

	PEFR	Exposure
0	390	0
1	410	0
2	430	0
3	460	0
4	420	1
...	...	...
117	450	22
118	490	22
119	500	22
120	370	23
121	390	23

122 rows × 2 columns

In [3]:
```
#Creamos un Scatter Plot pasando los valores "exposure" y "PEFR" dentro de data
plt.scatter(data.Exposure,data.PEFR)
```

Out[3]:
```
<matplotlib.collections.PathCollection at 0x1a3a4155670>
```



In [4]:
```
#Creamos el Modelo de regresion Lineal, lo asignamos a la variable "model"
model = linear_model.LinearRegression()
model.fit(data[['Exposure']],data['PEFR']) # lo ajustamos  con .fit y le pasamos los datos
```

Out[4]:
```
LinearRegression()
```

In [5]:
```
#imprimimos los valores de Intercept y Coefficient (β0 y β1)
print("intersecpt: ",model.intercept_)
print("Coefficient: ",model.coef_[0])
```

```
intersecpt:  424.582806573957
Coefficient:  -4.1845764854614425
```

In [6]:
```
#Creamos un Array desde el 0 a el 25
x = [ i for i in range(26)]
#lo trasnformamos en un array de numpy para podes usar Ufuncs
x = np.array(x)
x
```

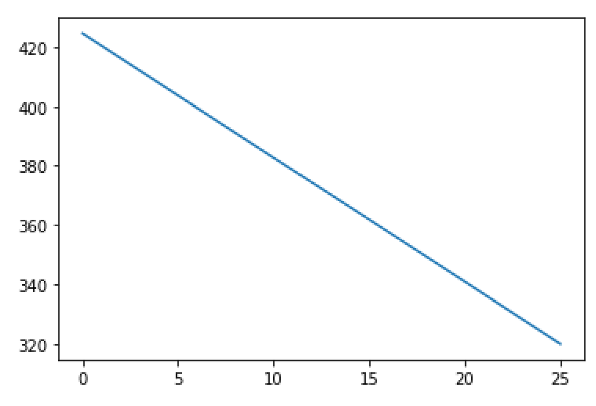
Out[6]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25])
```

In [7]:
```
#Creamos las Variables Intecept y Coeficient con los valores obtenidos del modelo
intercept = model.intercept_
coeficient = model.coef_

#utilizamos Ufuncs te numpy para hacer la Ecuacion General de la Regresion

y= intercept + coeficient * x
y #Resultado
```

Out[7]:
```
array([424.58280657, 420.39823009, 416.2136536 , 412.02907712,
       407.84450063, 403.65992415, 399.47534766, 395.29077118,
       391.10619469, 386.9216182 , 382.73704172, 378.55246523,
       374.36788875, 370.18331226, 365.99873578, 361.81415929,
       357.62958281, 353.44500632, 349.26042984, 345.07585335,
       340.89127686, 336.70670038, 332.52212389, 328.33754741,
       324.15297092, 319.96839444])
```

In [8]:
```
#creamos una Grafica con el Resultado Obtenido
plt.plot(y)
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x1a3a48e9850>]
```



la regresion lineal simple trata de encontrar la mejor linea que prediga la respuesta como una funcion de la variable predictora

Calculando la Regresion Lineal de Manera Manual:

debemos resolver la ecuacion de la Regresion lineal Simple: tenemos X e y de la siguiente manera:

In [9]:
```
# feature Vector
x= data["Exposure"]
# Response vector
y= data["PEFR"]
```

In [11]:
```
#chequeamos que todo este bien
x
```

Out[11]:
```
(0      0
 1      0
 2      0
 3      0
 4      1
       ..
 117    22
 118    22
 119    22
 120    23
 121    23
 Name: Exposure, Length: 122, dtype: int64,
 0      390
 1      410
 2      430
 3      460
 4      420
       ...
 117    450
 118    490
 119    500
 120    370
 121    390
 Name: PEFR, Length: 122, dtype: int64)
```

In [12]:
```
y
```

Out[12]:
```
0      390
1      410
2      430
3      460
4      420
      ...
117    450
118    490
119    500
120    370
121    390
Name: PEFR, Length: 122, dtype: int64
```

Para obtener β_0 y β_1 debemos usar la Funcion *Least Squares*

Least Squares:

$$y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$
$$= h(x_i) + \epsilon_i$$
$$\rightarrow \epsilon_i = y_i - h(x_i)$$

siendo ϵ_i el error

de esta funcion conseguimos la funcion de Costo J

Funcion de Costo:

$$J(\beta_0, \beta_1) = \frac{1}{2n} \sum_{i=1}^{n} \epsilon_i^2$$

de aqui podemos conseguir β_0 y β_1 quedandonos de la suiguiente manera:

Funcion de Coeficiente de Regresion:

$$\beta_1 = \frac{SSxy}{SSxx} = \frac{\sum_{i=1}^{n}[\overline{x} - x_i * \overline{y} - y_i]}{\sum_{i=1}^{n}[\overline{x} - x_i * \overline{x} - x_i]}$$

siendo:

- SSxy la "Desviacion Cruzada"
- SSxx la "Desviacion Sobre X "
- y con rayita arriva: media de Y
- x con rayita arriva : media de x

Funcion de Intercepcion :

$$\beta_0 = \overline{y} - \beta_1 \overline{x}$$

con esto ya tenemos todo para calcular la ecuacion en codigo:

In [13]:
```
#calculamos el coeficiente
def calculate_Regresion_coef_and_intercept(x,y):

    mean_x = np.mean(x) # media de X
    mean_y = np.mean(y) #media de Y

    SSxy= np.sum((mean_x - x) * (mean_y - y)) #Desviacion Cruzada
    Ssxx= np.sum((mean_x-x)*(mean_x-x)) # Desviacion sobre X

    b_1= SSxy/Ssxx #Coeficiente de Regresion
    b_0 = mean_y - b_1*mean_x # intercepcion

    return(b_0,b_1)
```

In [14]:
```
calculate_Regresion_coef_and_intercept(x,y) #llamamos a la funcion pasandole los valores X e Y
```

Out[14]:
```
(424.582806573957, -4.18457648546144)
```

vemos que efectivamente los valores de β_0 y β_1 son los mismos que nos dio el modelo de Sklearn.

In [ ]:
```

```