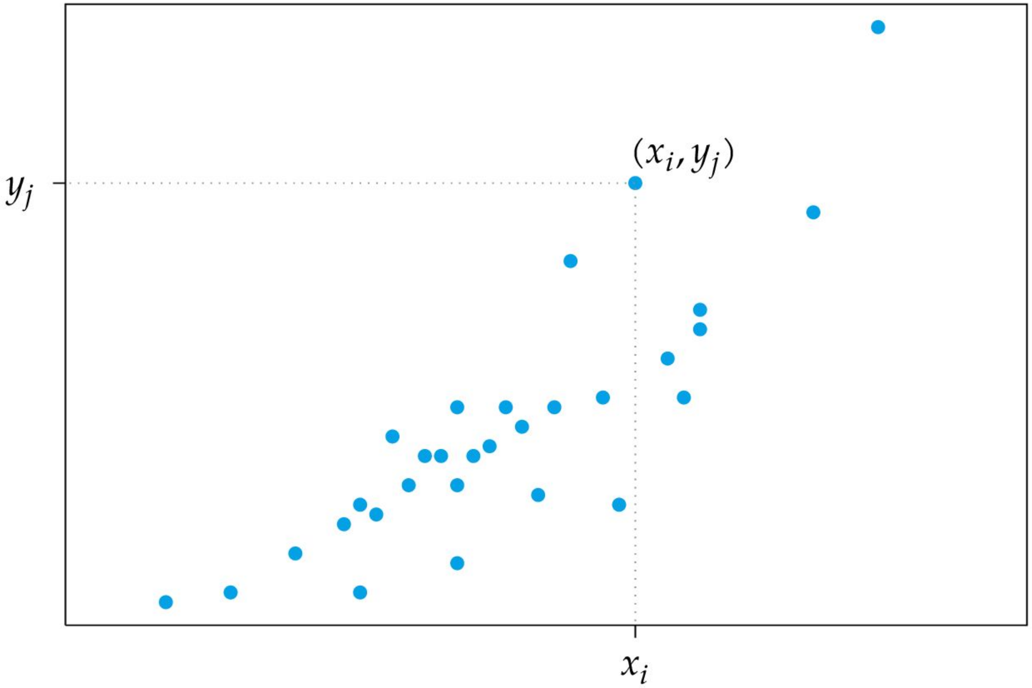
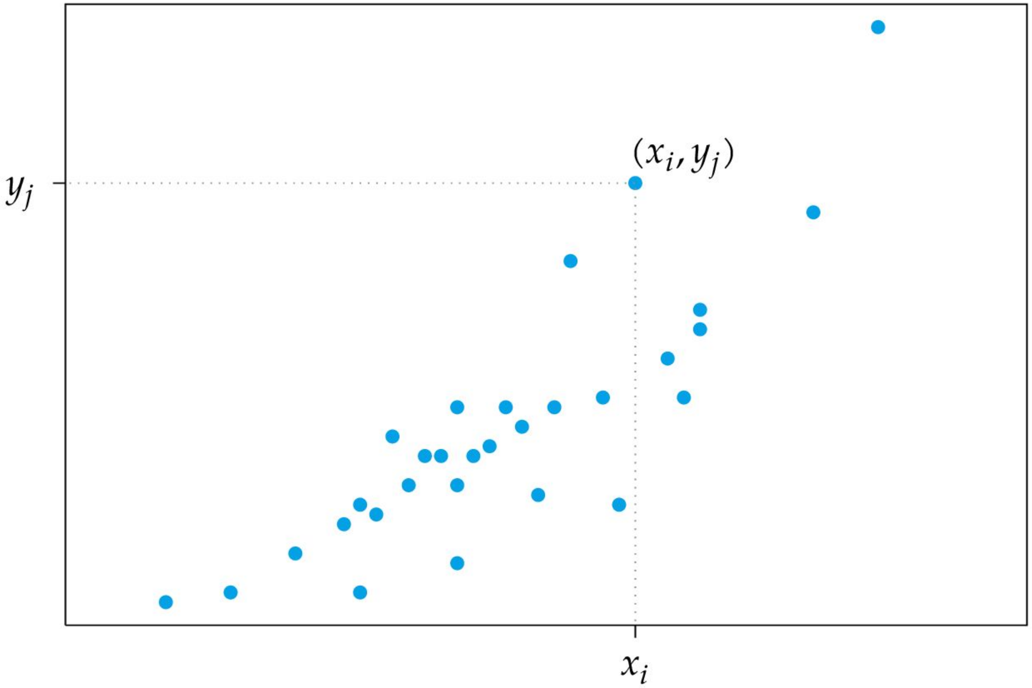
Modelos de Regresión



Regresión lineal simple

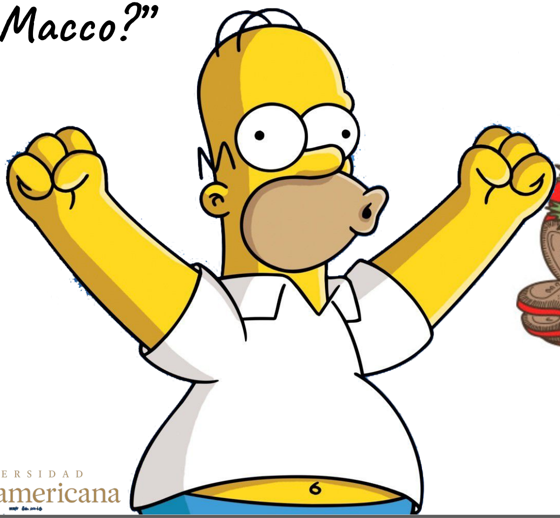


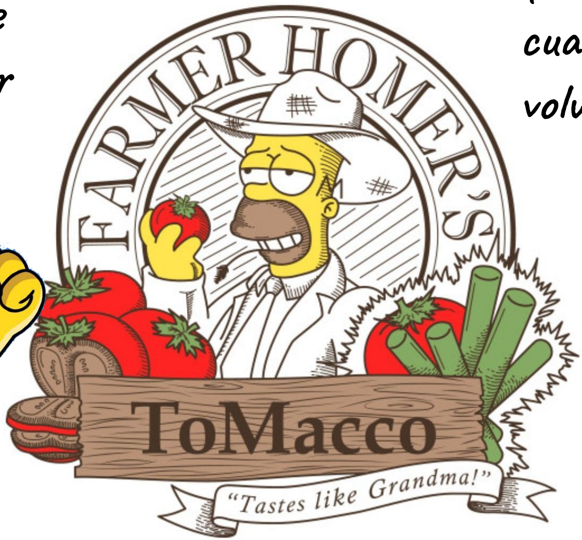
¿Qué es un modelo de Regresión <u>LINEAL</u>?

Es un modelo o función que permite ***predecir*** el valor de ***datos desconocidos*** a partir de datos que SÍ son conocidos.

<u>Ejemplo:</u>

¿Recuerdas aquella vez en la que Homero Simpson decidió plantar "ToMacco?"





Homero empezó con un pequeño negocio que comenzó a crecer lentamente cuando se dio cuenta de que la gente se volvía adicta al ToMacco.

Homero vendía cada pieza de ToMacco en 1 Dólar, y registró las siguientes ganancias durante una semana:

Lunes: $5
Martes: $15
Miércoles: $15
Jueves: $40
Viernes: $65
Sábado: $70
Domingo: $100

¿Crees que las ventas de Homero aumenten si vende ToMacco una semana mas?

Podemos saberlo si creamos una FUNCIÓN que describa el COMPORTAMIENTO de los datos que tenemos:

Día	Venta de ToMacco
0	5
1	15
2	15
3	40
4	65
5	70
6	100

Para describir el comportamiento de datos usando una línea recta, podemos usar la ecuación de la recta:

$$\boxed{Y = mX + b}$$

Resultado (Valor en la recta)

Pendiente de la recta

Ordenada al origen (Offset)

Para lograr encontrar los parámetros de la función, utilizamos un método que se conoce como **AJUSTE** de **CURVAS** por **MÍNIMOS CUADRADOS:**

Día	Venta de ToMacco
0	5
1	15
2	15
3	40
4	65
5	70
6	100

$$\boxed{\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

El objetivo de este método consiste en minimizar la suma de los cuadrados de los errores

Podemos despejar: $\Sigma\left(y - y_i\right)^2 = \Sigma\left(mx + b - y_i\right)^2$

$$\boxed{m = \frac{n\Sigma(x_i y_i) - \Sigma(x_i)(y_i)}{n\Sigma(x_i^2) - \left(\Sigma x_i\right)^2}}$$

$$\boxed{b = \frac{\left(\Sigma y_i\right) - m\left(\Sigma x_i\right)}{n}}$$

Para este ejemplo, podemos calcular los valores de m y b como:

Día	Venta de ToMacco
0	5
1	15
2	15
3	40
4	65
5	70
6	100

$$m = \frac{n\Sigma(x_i y_i) - \Sigma(x_i)(y_i)}{n\Sigma(x_i^2) - (\Sigma x_i)^2}$$

$$b = \frac{(\Sigma y_i) - m(\Sigma x_i)}{n}$$

Podemos generar una tabla con los valores de las sumatorias que vamos a necesitar para calcular m y b:

n	xi	yi	xi(yi)	xi*xi	yi*yi
1	0	5	0	0	25
2	1	15	15	1	225
3	2	15	30	4	225
4	3	40	120	9	1600
5	4	65	260	16	4225
6	5	70	350	25	4900
7	6	100	600	36	10000
Total	21	310	1375	91	21200

Calculamos los valores de m y b:

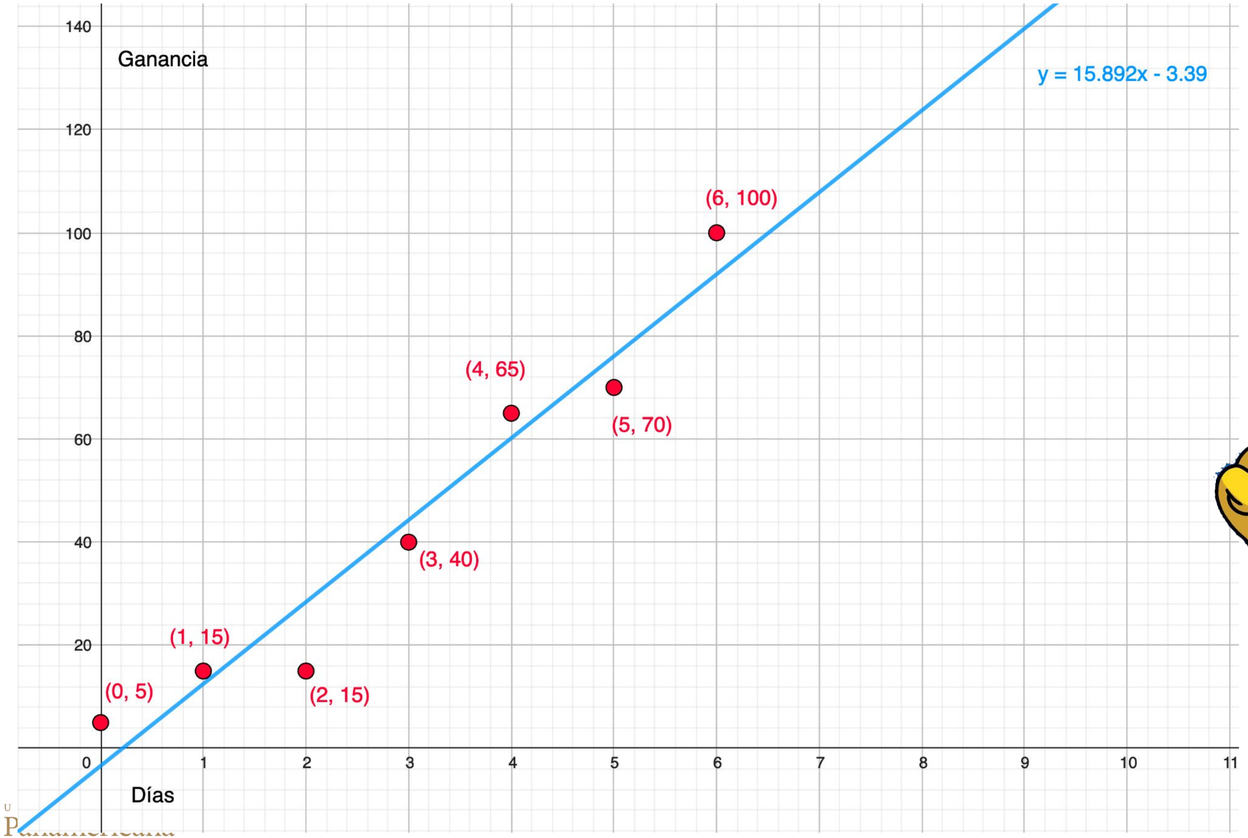
n	xi	yi	xi(yi)	xi*xi	yi*yi
1	0	5	0	0	25
2	1	15	15	1	225
3	2	15	30	4	225
4	3	40	120	9	1600
5	4	65	260	16	4225
6	5	70	350	25	4900
7	6	100	600	36	10000
Total	21	310	1375	91	21200

$$m = \frac{n\Sigma(x_i{*}y_i) - \Sigma(x_i)\Sigma(y_i)}{n\Sigma(x_i^2) - (\Sigma x_i)^2}$$

$$\frac{7(1375)-(21)(310)}{7(91)-(21)^2} = \frac{9625-6510}{637-441} = \frac{3115}{196} = \boxed{15.892}$$

$$b = \frac{\Sigma(y_i) - m\Sigma(x_i)}{n} = \frac{310-(15.892)21}{7} = \frac{310-333.732}{7} = \boxed{-3.39}$$

Entonces ¿Cómo se ve la función que describe el comportamiento de las ventas de Homero?



Parece que las ventas de Homero van aumentando. Woo Hoo!!!

¿Podríamos entonces predecir cuáles serían las ventas de Homero para los 10 días o para los 30 teniendo este modelo de regresión?

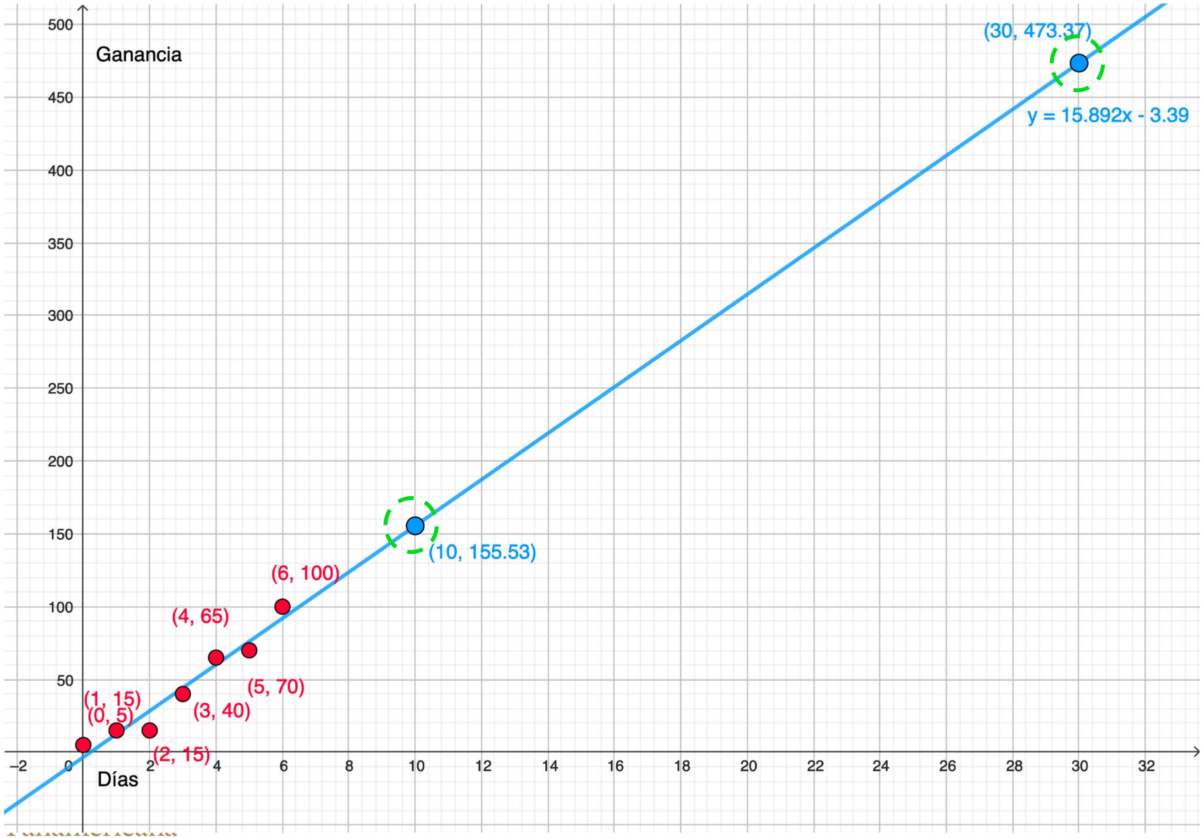
$$Y = mx + b = 15.892x - 3.39$$

$$f(x) = 15.892\boxed{(10)} - 3.39 = \boxed{155.53}$$

$$Y = mx + b = 15.892x - 3.39$$

$$f(x) = 15.892\boxed{(30)} - 3.39 = \boxed{473.37}$$

LISTO!!! tienes tu primer modelo de ML (Regresión lineal simple). Muchas felicidades!!!



Con esto puedes predecir las ganancias que homero tendría de aquí al infinito. Woo Hoo!!!

Es una lástima que le roben su ToMacco a Homero :(

Panamericana

Ahora es tu turno!!!

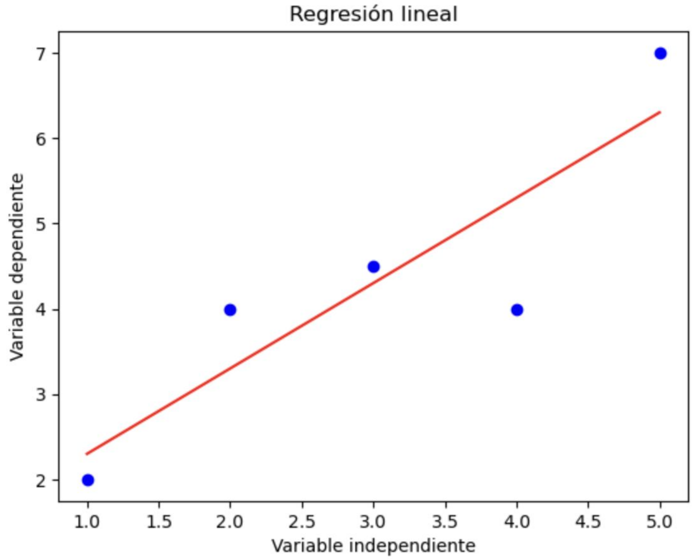
Día	Venta de ToMacco
0	4
1	6
2	22
3	38
4	72
5	80
6	110
7	112
8	124
10	135

Con los nuevos datos...
Cuál será la ganancia para el día 15?

¿Y para el día 5.5?

UNIVERSIDAD Panamericana

Ejemplo de código:




```python
# Importar las bibliotecas necesarias
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Datos de ejemplo
X = np.array([1, 2, 3, 4, 5]).reshape(-1, 1)  # Variable independiente
y = np.array([2, 4, 4.5, 4, 7])                 # Variable dependiente

# Crear un modelo de regresión lineal
modelo = LinearRegression()

# Entrenar el modelo con los datos
modelo.fit(X, y)

# Realizar predicciones
y_pred = modelo.predict(X)

# Visualizar los resultados
plt.scatter(X, y, color='blue')                # Datos originales
plt.plot(X, y_pred, color='red')               # Línea de regresión
plt.xlabel('Variable independiente')
plt.ylabel('Variable dependiente')
plt.title('Regresión lineal')
plt.show()

# Imprimir los coeficientes de la regresión
print('Coeficiente:', modelo.coef_)
print('Intercepto:', modelo.intercept_)
```

Ejercicio de programación:
Programa DESDE CERO los siguientes ejemplos de regresión lineal:

n	xi	yi
1	1	4
2	3	6
3	4	22
4	7	38
5	10	72
6	12	80
7	15	110
8	22	112
9	31	124
10	40	135

n	xi	yi
1	1	4
2	2	6
3	4	9
4	6	25
5	10	25
6	13	47
7	16	42
8	19	64
9	25	83
10	30	80

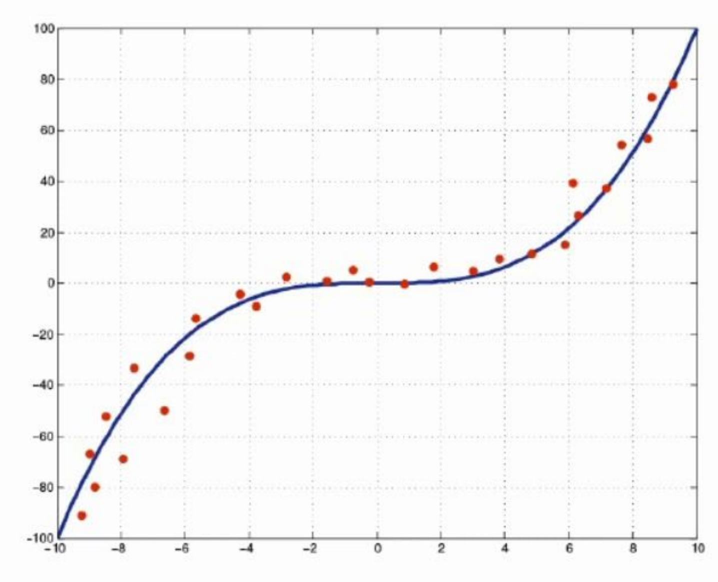
Mostrar la gráfica con los puntos en cada caso, así como la recta generada.

Escribir los valores de m y b

Cuánto vale f(50)?

UNIVERSIDAD Panamericana

Regresión No lineal Polinomial

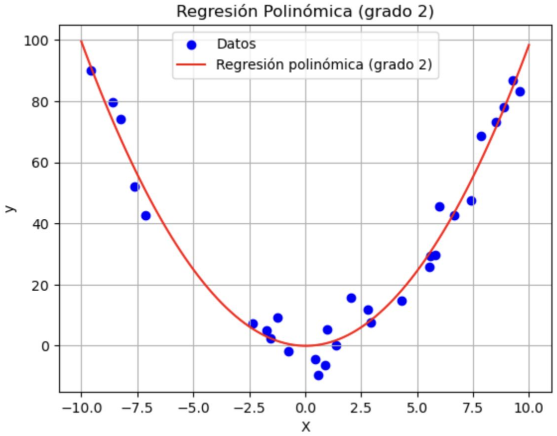


UNIVERSIDAD
Panamericana

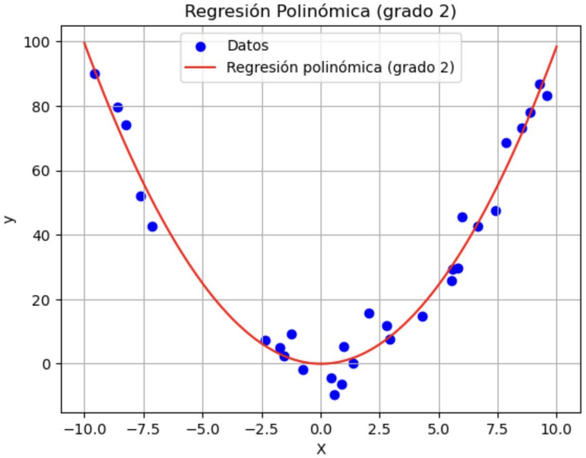
Regresión No lineal (Polinomial)

Finalmente, en aquellos modelos que por la naturaleza de sus datos NO se puedan ajustar con una línea recta, se utilizan los modelos de regresión No lineal (o regresión polinomial.

La diferencia con los anteriores es que se define un valor polinomial (grado 2 o superior), y el algoritmo (código) se adapta a este exponente para intentar ajustarse a los datos o ejemplos:



Ejemplo de código:




```python
# Importamos librerías
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

```python
# Generar datos no lineales
np.random.seed(0)
X = 20 * np.random.rand(30, 1) -10      # 30 Valores aleatorios entre -10 y 10
y = X**2 + 5 * np.random.randn(30, 1)  # Función cuadrática con ruido
                                        # 30 Valores de X elevada al cuadrado
                                        # mas un valor aleatorio de 0 a 5
```

```python
# Transformar las características para incluir términos polinomiales

grado_polynomial = 2
poly_features = PolynomialFeatures(degree=grado_polynomial)
X_poly = poly_features.fit_transform(X)
```

```python
# Crear y entrenar un modelo de regresión lineal
modelo = LinearRegression()
modelo.fit(X_poly, y)
```

```
▾ LinearRegression
LinearRegression()
```