Otra de las herramientas frecuentemente usadas para entender la distribución de un conjunto de datos es la generación del polinomio que mejor se aproxime a los datos, teniendo como criterio de "bondad" la suma de los cuadrados de la distancia del polinomio a los datos.

Nuevamente tenemos dos opciones, en función de que estemos trabajando con una instancia del objeto Polynomial o con los coeficientes del polinomio.

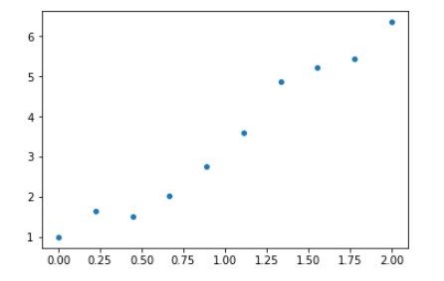
Para probar ambos métodos, partamos de los siguientes valores de x e y:

np.random.seed(18)  
x = np.linspace(0, 2, 10)  
y = np.exp(x) + np.random.randn(10) \* 0.8 \* x  
y

Mínimos cuadrados

Mostremos estos puntos en una gráfica de dispersión:

import seaborn as sns  
sns.scatterplot(x, y);



El método fit

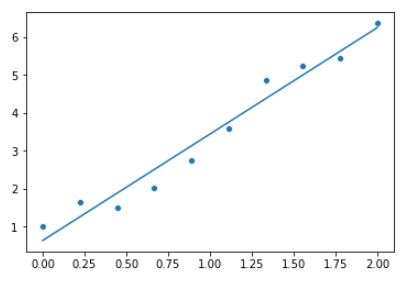
El método [fit](https://docs.scipy.org/doc/numpy/reference/generated/numpy.polynomial.polynomial.Polynomial.fit.html) del objeto Polynomial devuelve un objeto de tipo Polynomial representando el polinomio del grado indicado que minimiza la suma de cuadrados de las diferencias en las ordenadas. Por ejemplo, para aproximar los datos con un polinomio de grado 1:

f = np.polynomial.polynomial.Polynomial.fit(x, y, 1, domain = (-1, 1))  
f

El método fit

yf = [f(cx) for cx in x]  
fig, ax = plt.subplots()  
sns.scatterplot(x, y, ax = ax);  
ax.plot(x, yf, linestyle = "-");

Representemos ahora los datos **x** e **y** originales y la recta calculada:

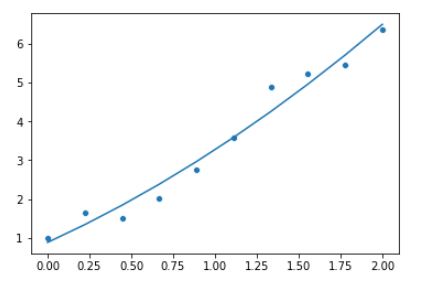


Si quisiéramos aproximar los puntos con un polinomio de grado 2, el proceso sería semejante:

f = np.polynomial.polynomial.Polynomial.fit(x, y, 2, domain = (-1, 1))  
f

El método fit

yf = [f(cx) for cx in x]  
fig, ax = plt.subplots()  
sns.scatterplot(x, y, ax = ax);  
ax.plot(x, yf, linestyle = "-");



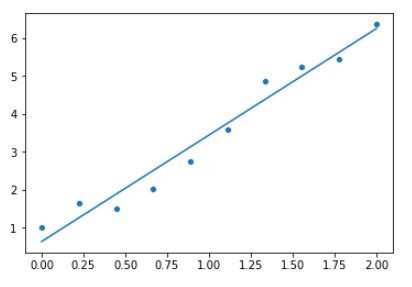
La función polyfit

La función numpy.polynomial.polynomial.polyfit nos permite aproximar un conjunto de datos según el procedimiento de mínimos cuadrados devolviendo un array NumPy con los coeficientes del polinomio en cuestión:

f = np.polynomial.polynomial.polyfit(x, y, 1)  
f

La función polyfit

yf = [np.polynomial.polynomial.polyval(cx, f) for cx in x]  
fig, ax = plt.subplots()  
sns.scatterplot(x, y, ax = ax);  
ax.plot(x, yf, linestyle = "-");



En el código anterior se ha utilizado la función ya vista [polyval](https://docs.scipy.org/doc/numpy/reference/generated/numpy.polynomial.polynomial.polyval.html" \t "_blank) para calcular las ordenadas de la recta (yf).

Si quisiéramos aproximar los datos con un polinomio de grado 2:

f = np.polynomial.polynomial.polyfit(x, y, 2)  
f

La función polyfit

yf = [np.polynomial.polynomial.polyval(cx, f) for cx in x]  
fig, ax = plt.subplots()  
sns.scatterplot(x, y, ax = ax);  
ax.plot(x, yf, linestyle = "-");

