



IMPORTS DE LIBRERÍAS

Agente inmobiliaria y de bienes raíces

Laboratorio1.py > ...

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.linear_model import LinearRegression
4 from sklearn.model_selection import train_test_split
5 from sklearn.metrics import mean_squared_error, r2_score
6
```

- **import numpy as np:** Importa la biblioteca NumPy y asigna el alias "np" para su uso en el código.
- **import matplotlib.pyplot as plt:** Importa el módulo pyplot de la biblioteca Matplotlib y asigna el alias "plt" para su uso en el código, que se utiliza para generar gráficos y visualizar datos.
- **from sklearn.linear_model import LinearRegression:** Importa la clase LinearRegression del módulo linear_model de la biblioteca scikit-learn, que se utiliza para crear un modelo de regresión lineal.
- **from sklearn.model_selection import train_test_split:** Importa la función train_test_split del módulo model_selection de la biblioteca scikit-learn, que se utiliza para dividir los datos en conjuntos de entrenamiento y prueba.



- `np.random.seed(0)`: Establece la semilla para la generación de números aleatorios, asegurando reproducibilidad en la generación de datos sintéticos.
- $X = 2 * np.random.rand(500, 1)$: Genera una matriz X de tamaño $(500, 1)$ con valores aleatorios entre 0 y 1, que representan el tamaño de la vivienda como variable independiente.
- $y = 5 + 3 * X + np.random.randn(500, 1)$: Genera una matriz y de tamaño $(500, 1)$ con valores aleatorios de una distribución normal, que representan el precio de la vivienda como variable dependiente, incluyendo un término de sesgo (5) y un coeficiente de regresión (3) aplicado a la variable independiente X .

```
# Generar datos sintéticos para el ejemplo
np.random.seed(0)
X = 2 * np.random.rand(500, 1) # Variable independiente (tamaño de la vivienda)
y = 5 + 3 * X + np.random.randn(500, 1) # Variable dependiente (precio de la vivienda)
```

- **train_test_split(X, y, test_size=0.2, random_state=42):** Divide los datos en conjuntos de entrenamiento y prueba, donde X_train y y_train son los datos de entrenamiento, X_test y y_test son los datos de prueba, con un tamaño de prueba del 20% y una semilla aleatoria de 42 para reproducibilidad.
- **model = LinearRegression():** Crea un objeto de modelo de regresión lineal.
- **model.fit(X_train, y_train):** Entrena el modelo de regresión lineal utilizando los datos de entrenamiento X_train y y_train.
- **y_pred = model.predict(X_test):** Realiza predicciones utilizando el modelo entrenado sobre los datos de prueba X_test.
- **print("MSE:", mean_squared_error(y_test, y_pred)):** Calcula y muestra el error cuadrático medio (MSE) entre las etiquetas verdaderas y_test y las predicciones y_pred.
- **print("R2:", r2_score(y_test, y_pred)):** Calcula y muestra el coeficiente de determinación (R2) entre las etiquetas verdaderas y_test y las predicciones y_pred.

```
# Dividir los datos en conjunto de entrenamiento y conjunto de prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear y entrenar el modelo de regresión lineal
model = LinearRegression()
model.fit(X_train, y_train)

# Hacer predicciones con el conjunto de prueba
y_pred = model.predict(X_test)

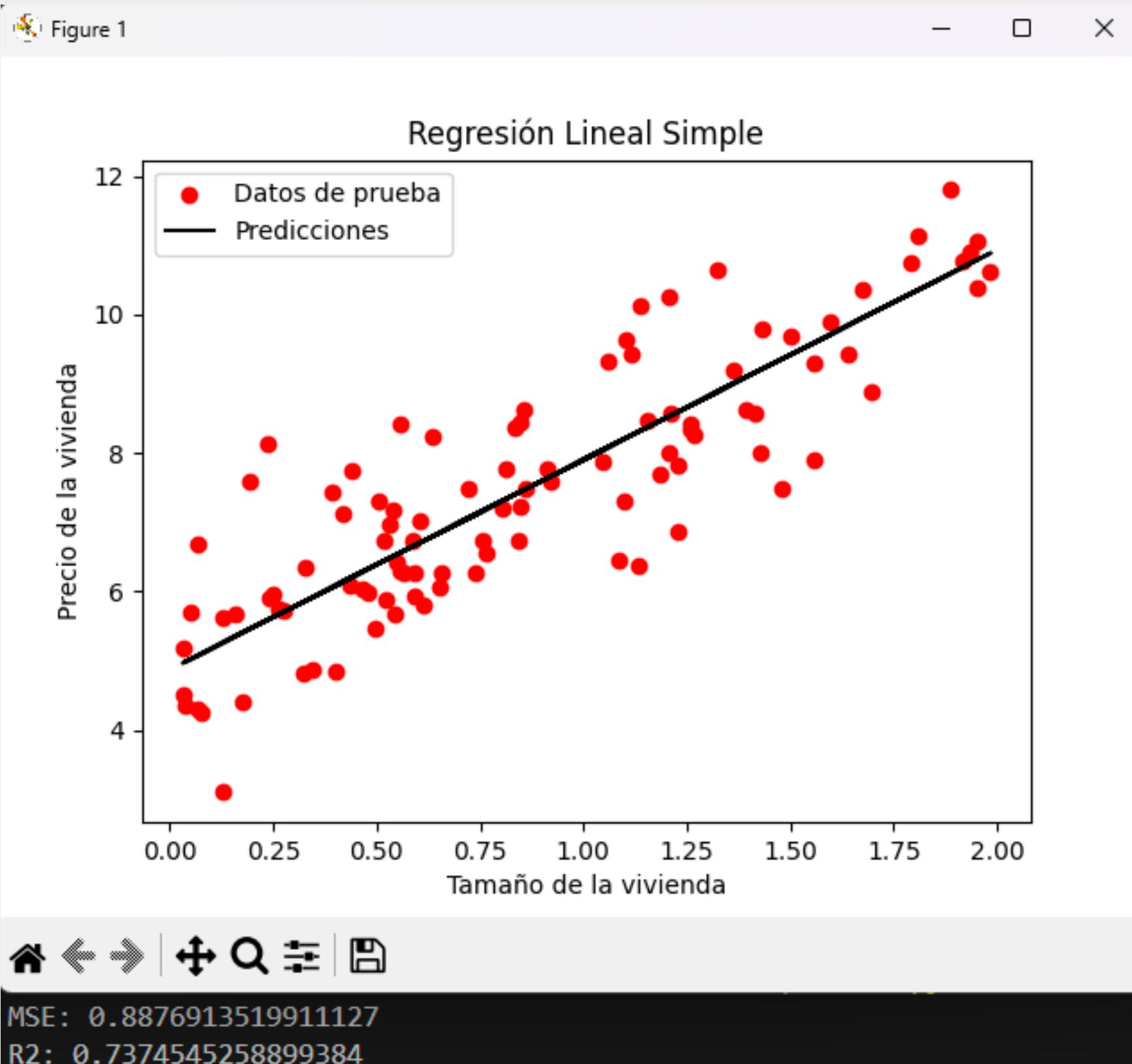
# Calcular y mostrar el error cuadrático medio (MSE) y el coeficiente de determinación (R2)
print("MSE:", mean_squared_error(y_test, y_pred))
print("R2:", r2_score(y_test, y_pred))
```



- `plt.scatter(X_test, y_test, color='red', label='Datos de prueba')`: Crea un diagrama de dispersión con los datos de prueba `X_test` y `y_test`, utilizando puntos rojos.
- `plt.plot(X_test, y_pred, color='black', label='Predicciones')`: Traza una línea de regresión utilizando las predicciones `y_pred` sobre los datos de prueba `X_test`, utilizando una línea negra.
- `plt.title('Regresión Lineal Simple')`: Establece el título del gráfico como "Regresión Lineal Simple".
- `plt.xlabel('Tamaño de la vivienda')`: Etiqueta el eje x del gráfico como "Tamaño de la vivienda".
- `plt.ylabel('Precio de la vivienda')`: Etiqueta el eje y del gráfico como "Precio de la vivienda".
- `plt.legend()`: Muestra la leyenda en el gráfico que indica qué representan los puntos rojos y la línea negra.
- `plt.show()`: Muestra el gráfico con todas las visualizaciones realizadas.



```
# Visualizar los resultados en un gráfico
plt.scatter(X_test, y_test, color='red', label='Datos de prueba')
plt.plot(X_test, y_pred, color='black', label='Predicciones')
plt.title('Regresión Lineal Simple')
plt.xlabel('Tamaño de la vivienda')
plt.ylabel('Precio de la vivienda')
plt.legend()
plt.show()
```



El gráfico muestra una dispersión de puntos rojos que representan los datos de prueba de tamaño de vivienda (eje y) y precio de vivienda (eje x). La línea negra representa las predicciones del modelo de regresión lineal para los mismos datos de prueba. La pendiente positiva de la línea indica una relación positiva entre el tamaño de la vivienda y su precio, lo que significa que a medida que aumenta el tamaño de la vivienda, también tiende a aumentar su precio.

Los resultados en la consola indican que el error cuadrático medio (MSE) entre las predicciones del modelo y los datos reales es de aproximadamente 0.888, mientras que el coeficiente de determinación (R2) es de aproximadamente 0.737. Un MSE más bajo y un R2 más cercano a 1 indicarían un mejor ajuste del modelo a los datos, pero en general, estos valores sugieren que el modelo tiene un rendimiento aceptable en la predicción del precio de la vivienda basado en su tamaño.