gradient_descent

January 30, 2026

[1]:
```
import numpy as np

def gradient_descent(X, y, lr=0.01, epochs=1000):
    n = len(X)
    b0 = 0.0  # intercept
    b1 = 0.0  # slope

    for _ in range(epochs):
        y_pred = b0 + b1 * X
        error = y_pred - y

        db0 = (1 / n) * np.sum(error)
        db1 = (1 / n) * np.sum(error * X)

        b0 -= lr * db0
        b1 -= lr * db1

    return b0, b1


if __name__ == "__main__":
    X = np.array(list(map(float, input("Enter X values (space separated): ").
 ↪split())))
    y = np.array(list(map(float, input("Enter Y values (space separated): ").
 ↪split())))

    lr = float(input("Enter learning rate (e.g. 0.01): "))
    epochs = int(input("Enter number of iterations: "))

    b0, b1 = gradient_descent(X, y, lr, epochs)

    print(f"\nFinal Model: y = {b0:.4f} + {b1:.4f}x")

    x_new = float(input("Enter X value to predict Y: "))
    print("Predicted Y:", round(b0 + b1 * x_new, 4))
```

```
Enter X values (space separated):  1 2 3 4 5 6
Enter Y values (space separated):  3 5 7 9 11 13
Enter learning rate (e.g. 0.01):  0.01
Enter number of iterations:  2000


Final Model: y = 0.9869 + 2.0031x

Enter X value to predict Y:  7

Predicted Y: 15.0083
```

```
[ ]:
```