In the context of `np.dot(w, scaled_X.T)`:

- `w` is a 1D vector with shape `(n_features,)`. In our example, `w = [0.5, 0.8]`, which means we have two weights for the two features (Area and Bedrooms).
- `scaled_X` is a 2D matrix where each row represents a sample and each column represents a feature. In our example, `scaled_X` is `[[1.2, -0.5]]`, meaning we have one sample with two features (Area and Bedrooms).

Now, the key part here is how `.T` is used:

- `scaled_X` is of shape `(1, 2)` (1 sample and 2 features).
- Applying `.T` to `scaled_X` will change its shape to `(2, 1)` — now it's a 2D column vector where each element corresponds to a feature for the single sample.

Matrix Multiplication:

The matrix multiplication `np.dot(w, scaled_X.T)` needs to match the dimensions of the vectors being multiplied:

- `w` has shape `(2,)` (2 weights for 2 features).
- `scaled_X.T` has shape `(2, 1)` (features arranged as a column vector for the single sample).

So when you perform `np.dot(w, scaled_X.T)`, it is equivalent to:

$$\text{output} = w_1 \cdot \text{Area} + w_2 \cdot \text{Bedrooms}$$

This is a dot product between the weight vector and the feature vector for the sample.

Updated Example with Clearer Explanation

python Copy code

```python
import numpy as np # Define the weight vector (w) and bias w = np.array([0.5, 0.8]) # Weights for area and bedrooms bias = 0.2 # Bias term # Define the scaled features (scaled_X) # Let's assume we have one data point: area = 1.2, bedrooms = -0.5 (scaled) scaled_X = np.array([[1.2, -0.5]]) # One sample with two features # Calculate the dot product and add the bias output = np.dot(w, scaled_X.T) + bias # scaled_X.T is now (2, 1) # Print the result print("Predicted output:", output)
```

Step-by-Step:

1. `scaled_X` **before transpose**: It's of shape `(1, 2)` — 1 sample with 2 features (Area and Bedrooms).

$$\text{scaled\_X} = \begin{bmatrix} 1.2 & -0.5 \end{bmatrix}$$

2. `scaled_X.T` **after transpose**: The transpose flips the rows and columns, turning it into a shape of `(2, 1)` .

$$\text{scaled\_X.T} = \begin{bmatrix} 1.2 \\ -0.5 \end{bmatrix}$$

3. **Dot product calculation**:

$$\text{np.dot}(w, \text{scaled\_X.T}) = (0.5 \cdot 1.2) + (0.8 \cdot -0.5) = 0.6 - 0.4 = 0.2$$

4. **Adding bias**:

$$0.2(\text{dot product}) + 0.2(\text{bias}) = 0.4$$

So the final predicted output is 0.4.

Conclusion:

The `.T` is used to align the dimensions of `w` and `scaled_X` for matrix multiplication. Since `scaled_X` has one row (one sample) and two columns (two features), transposing it ensures that the matrix multiplication between `w` (with shape `(2,)`)
and `scaled_X.T` (with shape `(2, 1)`) works as intended, resulting in a scalar output.