Broadcasting

NumPy has an elegant mechanism for arithmetic operation on arrays with different dimensions or shapes. As an example, when a scalar is added to a vector (or 1-D array). The scalar value is conceptually broadcasted or stretched across the rows of the array and added element-wise. See Figure 10-5.

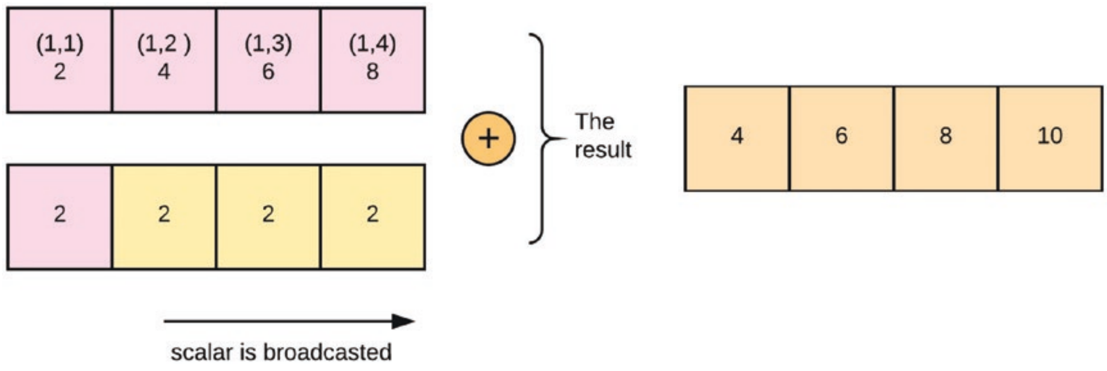

Figure 10-5. *Broadcasting example of adding a scalar to a vector (or 1-D array)*

Matrices with different shapes can be broadcasted to perform arithmetic operations by stretching the dimension of the smaller array. Broadcasting is another vectorized operation for speeding up matrix processing. However, not all arrays with different shapes can be broadcasted. For broadcasting to occur, the trailing axes for the arrays must be the same size or 1.

In the example that follows, the matrices **A** and **B** have the same rows, but the column of matrix **B** is 1. Hence, an arithmetic operation can be performed on them by broadcasting and adding the cells element-wise.

```
A      (2d array):  4 x 3       + <perform addition>
B      (2d array):  4 x 1
Result (2d array):  4 x 3
```

See Figure 10-6 for more illustration.

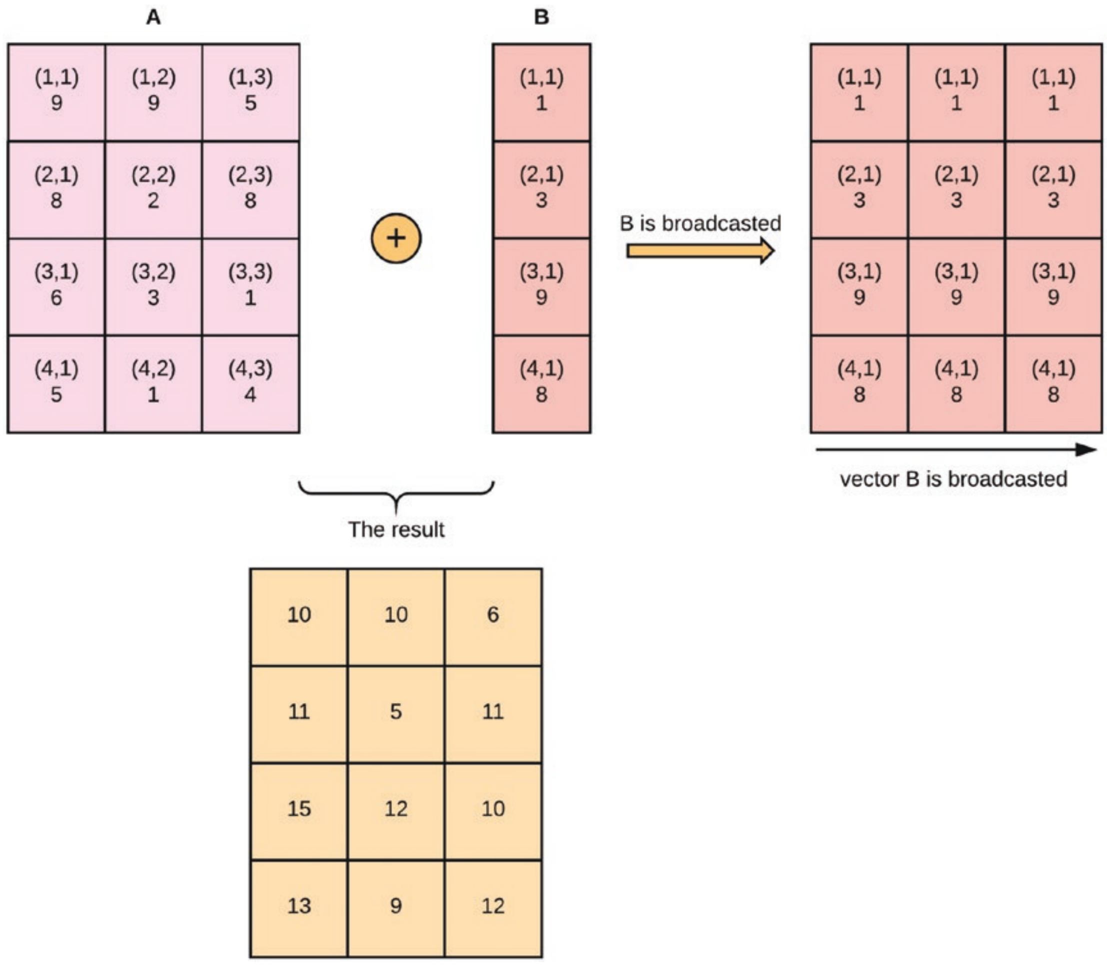

Figure 10-6. *Matrix broadcasting example*

Let's see this in code:

```
# create a 4 X 3 matrix of random integers between 1 and 10
A = np.random.randint(1, 10, [4, 3])
A
'Output':
array([[9, 9, 5],
       [8, 2, 8],
       [6, 3, 1],
       [5, 1, 4]])
```

```
# create a 4 X 1 matrix of random integers between 1 and 10
B = np.random.randint(1, 10, [4, 1])
B
'Output':
array([[1],
       [3],
       [9],
       [8]])
# add A and B
A + B
'Output':
array([[10, 10,  6],
       [11,  5, 11],
       [15, 12, 10],
       [13,  9, 12]])
```

The example that follows cannot be broadcasted and will result in a *ValueError: operands could not be broadcasted together with shapes (4,3) (4,2)* because the matrices **A** and **B** have different columns and do not fit with the aforementioned rules of broadcasting that the trailing axes for the arrays must be the same size or 1.

```
A      (2d array):  4 x 3
B      (2d array):  4 x 2
The dimensions do not match - they must be either the same or 1
```

When we try to add the preceding example in Python, we get an error.

```
A = np.random.randint(1, 10, [4, 3])
B = np.random.randint(1, 10, [4, 2])
A + B
'Output':
Traceback (most recent call last):

  File "<ipython-input-145-624e41e41a31>", line 1, in <module>
    A + B

ValueError: operands could not be broadcast together with shapes (4,3) (4,2)
```

Loading Data

Loading data is an important process in the data analysis/machine learning pipeline. Data usually comes in **.csv** format. **csv** files can be loaded into Python by using the **loadtxt** method. The parameter **skiprows** skips the first row of the dataset – it is usually the header row of the data.

```
np.loadtxt(open("the_file_name.csv", "rb"), delimiter=",", skiprows=1)
```

Pandas is a preferred package for loading data in Python.
We will learn more about Pandas for data manipulation in the next chapter.