27-11-2020 LECTURE-16

KANAV BANSAL

NUMPY BROADCASTING:

The term broadcasting refers to the ability of Numpy to treat arrays of different shapes during Arithematic operations.

↳ Arithematic operations on arrays are usually done on corresponding elements.

↳ If two arrays are of exactly the same shape, then these operations are smoothly Performed.

Broadcasting is possible if the following rules are satisfied:

↳ Array with smaller 'ndim' than the othere is pretended with '1' on its shape.

↳ Size in each dimension of the output shape is maximum of the input sizes in that dimension.

↳ An input can be used in calculation, if its size in a particular dimension matches the output size (or) its value is exactly 1.

↳ If an input has a dimension size of 1, the first data entry in that dimension is used for all calculations along that dimension.

A set of Arrays is said to be broadcastable if the above rules produce a valid result and one of the following is True:

↳ Arrays have exactly the same shape

↳ Arrays have the same number of dimensions and the length of each dimension is either a common length or 1.

↳ Array having too few dimensions can have its shape prepended with a dimension of length 1, so that the above stated Property is true.

→ Start matching the dimensions backward (Right to left)

* Compatible - If same number appears or if one of them is 1

* Otherwise - Incompatible

I/p - arr_1 = np.array([[1, 2, 3], [4, 5, 6]])

arr_2 = np.array([1, 2, 3])

Print(arr_1 + arr_2)

O/p - [[2 4 6]
[5 7 9]]

arr_1 $\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}_{2\times3}$ arr_2 $\begin{bmatrix}1 & 2 & 3\end{bmatrix}_{1\times3}$

2x3 ↔ 2x3

2 1x3

$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} + \begin{bmatrix}1 & 2 & 3\\ 1 & 2 & 3\end{bmatrix} = \begin{bmatrix}2 & 4 & 6\\ 5 & 7 & 9\end{bmatrix}$

I/p - arr_1 = np.array([[1, 2, 3], [4, 5, 6]])

arr_2 = np.array([[1], [2]])

Print(arr_1 + arr_2)

O/p - [[2 3 4]
[6 7 8]]

$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} + \begin{bmatrix}1\\ 2\end{bmatrix}$ ↔ 2x3, 2x1

2x3 2x1 2x3

$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} + \begin{bmatrix}1 & 1 & 1\\ 2 & 2 & 2\end{bmatrix} = \begin{bmatrix}2 & 3 & 4\\ 6 & 7 & 8\end{bmatrix}$

I/p - arr_1 = np.array([[1,2,3],[4,5,6]])

arr_2 = np.array([1])

Print(arr1 + arr_2)

O/p - [[2 3 4]
[5 6 7]]

$$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}_{2\times3} + [1]_{1\times1} \longleftrightarrow 2\times3 \quad 2\times1\times3 \rightarrow 2\times3$$

$$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} + \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} = \begin{bmatrix}2 & 3 & 4\\ 5 & 6 & 7\end{bmatrix}$$

I/p - arr_1 = np.array([1, 2, 3, 4, 5])

arr_2 = np.array([1, 2, 3, 4])

Print(arr_1 + arr_2)

O/p - Error - operands couldn't broadcast together with shapes (5,)(4,)

I/p - arr_1 = np.array([[1],[2],[3],[4],[5]])

arr_2 = np.array([1, 2, 3, 4])

Print(arr_1 + arr_2)

O/p - [[2 3 4 5]
[3 4 5 6]
[4 5 6 7]
[5 6 7 8]
[6 7 8 9]]

PANDAS :

It is a fast, powerful, flexible, and easy to use open source data analysis and manipulation tool, built on the top of the python Programming language.

↳ It is of an expressive data structure designed to make working with structured - tabular, multidimensional, potentially heterogeneous and time series data both easy and intuitive.

↳ It is built on the NumPy package and its key data structure is called as DataFrame. This DataFrame is used to allow us to store and manipulate tabular data in rows of observations and columns of variables.