# NUMPY

**INTRODUCTION TO NUMPY**

* N-dim array for fast mathematical calculation
* Homogeneous data structure
* Written in C/C++
* Scikit-learn is implemented using this
* Python list is generalized & thus slow
* Python list is a referenced data structure. The referenced nature of python list gives the ability to store heterogeneous elements
* Numpy stores data of the same type.

**INITIALIZATION OF NUMPY ARRAYS**

* Size is important parameter for all of them.
* Zeros – Initialized with zero values.
* Ones – Initialized with one value.
* Empty – Uninitialized array (contains garbage value)
* Full – Initialized with value passed.
* Random – Different methods to initialize random values.
* Lin\_space – generate equally spaced numbers between start and stop numbers.
* Ones\_like – Based on passed Numpy object, size is allocated with

**ACCESSING OF NUMPY LIST**

Very similar to that of accessing list

d[:,:] 🡪 this will print the whole array (all the rows and all the columns)

d[:,1:4] 🡪 this will print all the rows and columns from index 1 to 3 i.e. index 4 is exclusive

d[:,1] 🡪 this will print all the rows and the first indexed column

d[1:4,1:4] 🡪this will print the rows from row index 1 to row index 3 and columns from column index 1 to column index 3

**CONCATENATION OPERATION ON NUMPY ARRAYS**

* Axis = 0 means vertical
* Axis = 1 means horizontal
* Concatenate – join Numpy array along with provided axis
* Hstack – joining Numpy array horizontally
* Vstack – joining Numpy array vertically
* ***For any of these operations to happen, the joining edge should be same***

**SPLITTING**

np.random.randint(1,100,size=(4,5)) by this we get a random array of 4 cross 5 size

* Splits array into subarrays
* Split – splits the array as per axis mentioned
* Hsplit – this splits the horizontal axis
* Vsplit – this splits the vertical axis

**RESHAPING**

* Learning algorithms expects data in certain shape & dimension
* Using reshaping utility we can convert data into desired shape
* **But, the desired transformation will also be of same size**
* Adding 1 dimensions will not alter data size

**ADDING DIMENSION**

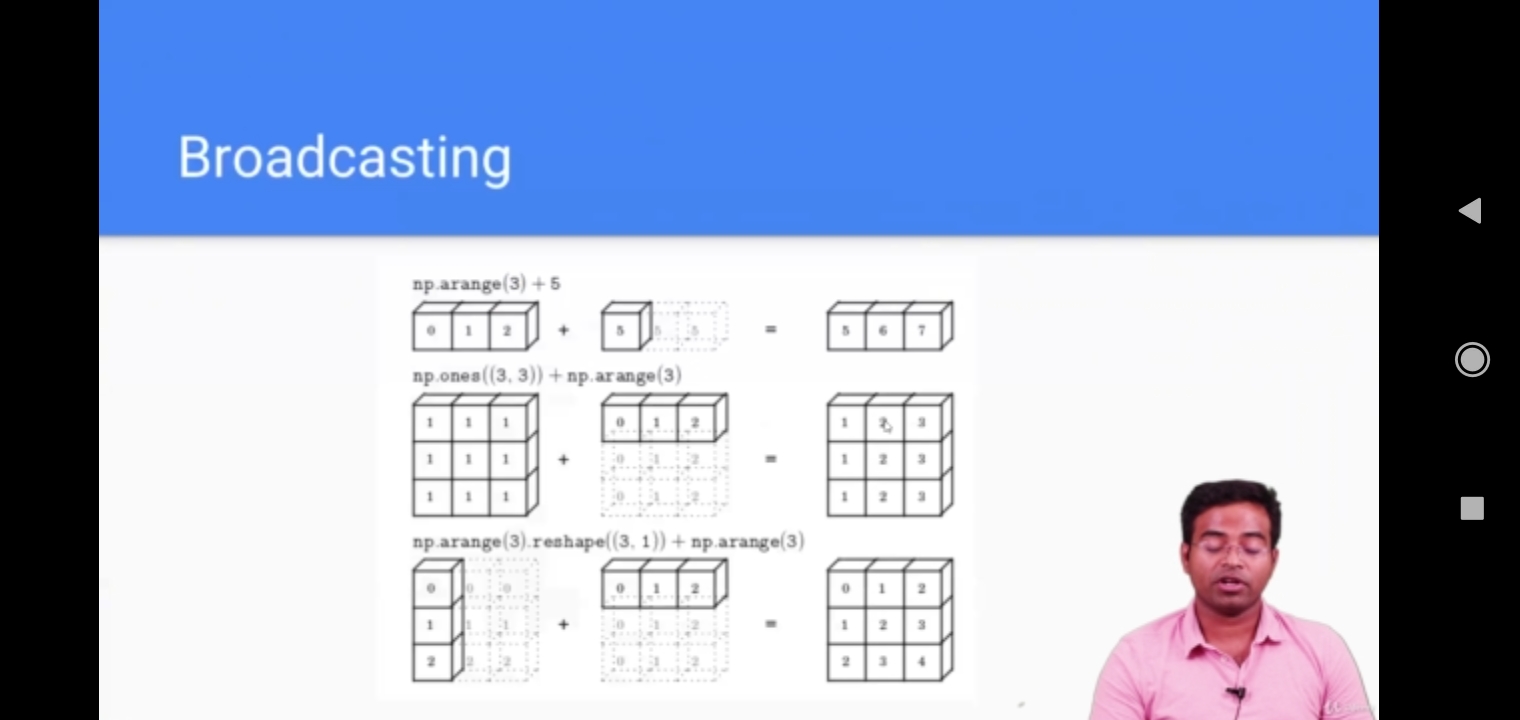
* Learning algorithms consumes data in two dimension
* We can convert 1-D array to 2-D array.

**BROADCASTING**

* Any data that will be coming in machine learning will be in vector format. Any raw data in the machine learning has to be converted into vector format. Entire machine learning understands data in vector format. In that row number of columns say about the features of the data.
* In broadcasting Numpy actually don’t stretch the array it does it by mathematical calculations
* Single row matrix are known as vectors.
* Broadcasting is a technique using Numpy does mathematical computation on data of different shapes & dimensions.
* We might need to reshape the data sometimes to enable broadcasting.

***Defn:*** It’s a technique used by Numpy to do mathematical computation when the data structure is not of same size and dimensions. The image below shows Broadcasting.

Fundamentally in broadcasting, it converts a scalar value into a vector in which the columns will be having the same value as provided.

****

In broadcasting the direction in which only ***one row/column*** is there then that row/column will be stretched throughout the array to fulfil the ***ELEMENT-WISE*** addition or subtraction or any mathematical operation.

**Broadcasting Rules:**

1. Size of each dimension should be same.
2. Size of one of the dimensions should be 1.
3. If the two arrays differ in the number of dimensions, the shape of the one with fewer dimension is padded with ones on its leading side(left side).
4. If the shape of the two arrays does not match in any dimension, the array with shape equal to 1 in that dimension is stretched to match the other shape.
5. If in any dimension the sizes disagree and neither equal to 1, an error is raised.

Pandas can be used for making spreadsheets

# PANDAS

**INTRODUCTION TO PANDAS**

* High performance Easy-to-use open-source library for Data Analysis.
* Creates tabular format of data from different sources like csv, json, database.
* Have utilities for descriptive statistics, aggregation, handling missing data.
* Database utilities like merge, join are available.
* Fast, programmable & easy alternative to spreadsheets.

**UNDERSTANDING SERIES AND DATAFRAMES**

* Series represents one column.
* Combine multiple columns to create a table (i.e. dataframe)
* Data corresponding to the same index belongs to the same row.
* Elements should be listed in the same serial order in both the series then only the dataframe will show the corresponding elements

***Creating a random dataframe of size 10\*10 and with the values in the range of 1 to 10 and 10 exclusive 🡪*** pd.DataFrame(data=np.random.randint(1,10, size = (10,10)), index = list('ABCDEFGHIJ'), columns = list('abcdefghij'))

**DESCRIPTIVE STATISTICS**

* dataframe.head() will give the top 5 rows, dataframe.tail() will give the bottom 5 rows, dataframe.info() will give the complete information about the dataframe.
* dataframe.describe() will give the numerical information of the data. This will be applied only to the numerical columns of the dataframe
* dataframe[column\_name].head() 🡪 this will give the top 5 indexes in a series format & dataframe[[column\_name]].head() 🡪 this will give top 5 indexes in a dataframe format.
* dataframe.columns will give the column names.
* dataframe.loc(‘’) takes the non-integer values and dataframe.iloc(‘’) takes the integer values.

**HANDLING MISSING DATA**

* Machine Learning algorithm don’t expect data missing
* If there is a column with more than 40% data missing, we may drop the column
* For rows with, important column values missing, drop the rows.

**HANDLING DUPLICATES**

* Sometimes, its difficult to ensure that the data is not duplicated.
* This becomes responsibility in Data cleaning step to make sure duplicated data is deleted.