1. Install, configure and run python, numPy and Pandas.

NUMPY

Numpy is the core library for scientific and numerical computing in Python. It provides high performance multi dimensional array object and tools for working with arrays.

Numpy main object is the multidimensional array, it is a table of elements (usually numbers) all of the same type indexed by a positive integers.

In Numpy dimensions are called as axes.

Numpy is fast, convenient and occupies less memory when compared to python list.

import numpy

arr = numpy.array([1, 2, 3, 4, 5]) print(arr)

NumPy is usually imported under the np alias.

import numpy as np

Now the NumPy package can be referred to as np instead of numpy.

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

Checking NumPy Version

The version string is stored under version attribute.

import numpy as np print(np. version )

Create a NumPy ndarray Object

NumPy is used to work with arrays. The array object in NumPy is called ndarray. We can create a NumPy ndarray object by using the array() function.

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

print(type(arr))

type(): This built-in Python function tells us the type of the object passed to it. Like in above code it shows that arr is numpy.ndarray type.

To create an ndarray, we can pass a list, tuple or any array-like object into the array() method, and it will be converted into an ndarray:

Use a tuple to create a NumPy array:

import numpy as np

arr = np.array((1, 2, 3, 4, 5)) print(arr)

Dimensions in Arrays

A dimension in arrays is one level of array depth (nested arrays).

nested array: are arrays that have arrays as their elements. 0-D Arrays

0-D arrays, or Scalars, are the elements in an array. Each value in an array is a 0-D array.

#Create a 0-D array with value 42 import numpy as np

arr = np.array(42) print(arr)

1-D Arrays

These are the most common and basic arrays.

#Create a 1-D array containing the values 1,2,3,4,5:

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

2-D Arrays

An array that has 1-D arrays as its elements is called a 2-D array. These are often used to represent matrix or 2nd order tensors.

#Create a 2-D array containing two arrays with the values 1,2,3 and 4,5,6: import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]]) print(arr)

3-D arrays

An array that has 2-D arrays (matrices) as its elements is called 3-D array. These are often used to represent a 3rd order tensor.

#Create a 3-D array with two 2-D arrays, both containing two arrays with the values 1,2,3 and 4,5,6: import numpy as np

arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(arr)

Check Number of Dimensions?

NumPy Arrays provides the ndim attribute that returns an integer that tells us how many dimensions the array have.

#Check how many dimensions the arrays have:

import numpy as np

a = np.array(42)

b = np.array([1, 2, 3, 4, 5])

c = np.array([[1, 2, 3], [4, 5, 6]])

d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(a.ndim) print(b.ndim) print(c.ndim) print(d.ndim)

#Create an array with 5 dimensions and verify that it has 5 dimensions: import numpy as np

arr = np.array([1, 2, 3, 4], ndmin=5) print(arr)

print('number of dimensions :', arr.ndim)

NumPy Array Indexing Access Array Elements

Array indexing is the same as accessing an array element.

You can access an array element by referring to its index number.

The indexes in NumPy arrays start with 0, meaning that the first element has index 0, and the second has index 1 etc.

#Get the first element from the following array:

import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr[0])

#Get the second element from the following array. import numpy as np

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

#Get third and fourth elements from the following array and add them. import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr[2] + arr[3])