What are Series and Vectors

Edris Safari

Bellevue University, Nebraska U.S.A.

Abstract

Data structures in computer science and specifically computer programming is an essential component of any program or application. Data Science is not exception. Data structures allow us to store and transmit information or data in a concise and encapsulated way. In this paper, we will discuss Series and Vectors-two data structures that are widely used in the Data Science applications.

Keywords:

Series, Vectors

What are Series and Vectors

Vectors

Vector is a mathematical concept that has been used in physics, mathematics and computer science. To a physicist, a vector is an arrow. Like an arrow, it has a direction, and if we can imagine, a magnitude. Once a vector's direction and magnitude are defined, the vector can exist anywhere in space with the same direction and magnitude. The direction of a vector represents where the arrow is heading, and the magnitude can represent whatever the arrow itself is representing. For example, an arrow pointing downward could represent the direction of the gravitational pull and the magnitude is the force at which gravity is pulling. There can be multiple arrows or vectors involved. For example, one could represent the direction and speed of an airplane and another the direction and pull from gravity and another the direction and force of wind. The sum of these vectors can give us the direction and speed of the airplane.

To a computer scientist, a vector is a list of numbers. For example, to represent a person, the values could be height in feet and weight in pounds. Vectors in mathematics are conceptually the same as viewed by physicist and computer scientist, but in mathematics, we can add, subtract, multiply, divide, power by and a host of other mathematical operation on them.

In physics, vectors are represented by an x-coordinate value and a y-coordinate value. These values determine the direction of an arrow emanating from the origin of the XY coordinate system(or the cartesian coordinate system). The arrow's direction and length(magnitude) is determined by x and y values. For example, with x=2 and y=-1. The arrow would be pointing towards south-west of the coordinate system (2^{nd} quadrant going clockwise). Its length would be determined as shown below:

$$length = \sqrt[2]{x^2 + y^2}$$

We can multiply this vector by a number. If that number is positive, the vector elongates, if it is negative, it shortens. The multiplication affects both the x value and the y value. For example if we multiply the vector [2 -1] by 3, we get [3x2 3x-1] which is [6 -3]. The length of [2 -1] vector is 2.23 and the [6 -3] has a length of 6.70 which is 2.23*3. Similarly, we can add two vectors, subtract, multiply, divide and many other operations on them.

In computer science, vectors are not limited to two numbers and can have any number of elements. They are represented by arrays. The arrays are indexed from 0 and can be accessed by index or range of indices. They are typically populated by numbers. Arrays can be multiplied by a number or by other arrays. They can also be added, subtracted and divided. For example, an array representing the weight of 10 people in pounds can be converted (or one derived from it) to ounces by multiplying every member of the array by 16. Arrays, and matrices which are an extension of arrays play a very important role in data science. Artificial Neural Networks for example use matrices to store the wight information for the nodes of the network. They are used to represent complex multivariate equations and the manipulation of matrices allows us to solve complex problems.

Series

Series is indeed a singular word and it represents an array of objects. It is almost the same as the array. The exception is that the Series' index is arbitrary, and the members can be any type. We can think of a dictionary as a Series where the names are the indices to their corresponding value. Using NumPy, we can create a series as shown below:

```
# Creating pandas series
labels = ['a','b','c']
my_data = [10,20,30]
array_1 = np.array(my_data)
d= {'a':10,'b':20,'c':30}
```

The series ‘d’ is a column of data (with no header name) with three values 10,20, and 30. Each row of the column is indexed by a,b, and c respectively. When we talk columns, we must talk tables. Yes, Series can be combined to make a table. Let’s say we have a series that has Transaction ID as index and Sale Price as value. Let’s say we have another series with Transaction ID as index and Transaction Date as value. Combine these series, and we have a table with three columns, Transaction ID, Sales Price, and Transaction Date.

Pandas library provides a host of functions that allows us to not only create series, but also tables. For example, the code below shows four different ways we can create a Series in Panda:

```
#%%

import pandas as pd

#%%

series_1 = pd.Series(data=my_data)
print(series_1)

#%%

series_2 = pd.Series(data=my_data,index=labels)
print(series_2)

#%%
my_data = [10,20,30]
array_1 = np.array(my_data)
series_3 = pd.Series(array_1,labels)
print(series_3)

#%%

series_4 = pd.Series(d)
print(series_3)
```

Creating tables is also easy in Pandas. Instead of a Series, we create a matrix of N x M elements and use the DataFrame method of pandas to build the table. The table index and column

names can be assigned by default or provided explicitly in the method call. The code below creates a 5 by 4 table with columns names W,X,Y,Z and indices A,B,C,,D, and E.

```
matrix_data = np.random.randint(1,10,size=20).reshape(5,4)
# Define the rows labels as ('A','B','C','D','E') and column labels as
('W','X','Y','Z'):
row_labels = ['A','B','C','D','E']
column_headings = ['W','X','Y','Z']
df = pd.DataFrame(data=matrix_data, index=row_labels, columns=column_headings)
print("\nThe data frame looks like\n",'-'*45, sep='')
print(df)
```

Series and tables are typically not created from scratch as shown above. Series derived from data loaded into a DataFrame are often used in the data wrangling phase of data science pipeline. Pandas provides methods to load CSV, Excel, JSON and other file types to load the data directly into a DataFrame. Once loaded, the methods to extract, manipulate the data are used to wrangle the data to a usable form. For example, we can manipulate a column from a dataframe in as simple a way as below:

```
# Replace missing values for weight and height to 9999lbs and 833'3 respectively.
# Once converted in convert_height function below, the height value of 833'3 will
convert to 9999
# The 9999's in both weight and height columns will then be replaced with the mean
value of their respective columns
data['Weight'].fillna('9999lbs', inplace = True)
data['Height'].fillna("833'3", inplace = True)
```

In the code above, we replaced all missing values in columns ‘Weight’ and ‘Height’ with a value. Using other methods, we could replace them with mean value.

Conclusion

Vectors and Series are similar in some fashion, but they serve different purposes. Vectors are used in performing computations. As vectors combine to make matrices, Series combine to make tables. Matrices can be computed whereas tables can be computed AND analyzed if certain

section or group of columns are represented by numeric values that can be modeled by matrices and vectors. The data types in a table can be numeric, or non-numeric. Discrete, continuous, nominal, ordinal, and all kinds of types we can think of to answer our questions. We ultimately turn every data item in the table into a numeric value that we can perform computation on. This is the heart and soul of data wrangling.

References

1. Sarkar, Dr. Tirthajyoti. Data Wrangling with Python: Creating actionable data from raw sources . Packt Publishing. Kindle Edition.
2. https://www.mathsisfun.com/algebra/vectors.html
3. Albon, Chris. Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning (p. 2). O'Reilly Media. Kindle Edition.
4. 12- Pandas: Introduction to Series - https://www.youtube.com/watch?v=m7gxnZx2vT4
5. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html