Data Science with Python Programming

- Presentation By Uplatz
- Contact us: https://training.uplatz.com
- Email: info@uplatz.com
- Phone: +44 7836 212635

Analysing Data using Numpy and Pandas

Learning outcomes:

Analysing Data using Numpy & Pandas

- What is numpy? Why use numpy?
- Installation of numpy
- Examples of numpy
- What is pandas?
- Key features of pandas
- Python Pandas - Environment Setup
- Pandas – Data Structure with example
- Data Analysis using Pandas

Numpy

It is another useful component that makes Python as one of the favourite languages for Data Science. It basically stands for Numerical Python and consists of multidimensional array objects. NumPy (Numerical Python) is a perfect tool for scientific computing and performing basic and advanced array operations. If you are going to work on data analysis or machine learning projects, then having a solid understanding of numpy is nearly mandatory.

Why use numpy?

In Python we have lists that serve the purpose of arrays, but they are slow to process.
NumPy aims to provide an array object that is up to 50 time faster than traditional Python lists.
The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
Arrays are very frequently used in data science, where speed and resources are very important.

Installation of NumPy

If you are using Anaconda distribution, then you need not install jupyter notebook , numpy separately as it is already installed with it.

Standard Python distribution doesn't come bundled with numpy module. A lightweight alternative is to install numpy using popular Python package installer, **pip.**

Installation of NumPy

Open the command prompt and change the directory to your current working directory and then type any one of the following command.

pip3 install numpy   OR
python -m pip install numpy   OR
pip install --user numpy

Example of NumPy

In NumPy dimensions are called *axes*. The number of axes is *rank*. NumPy's array class is called **ndarray**. It is also known by the alias **array**.

Example-1:

```
import numpy
arr = numpy.array([10, 20, 30, 40, 50])
print(arr)
```

NumPy is usually imported under the np alias.

Example-2:

```
import numpy as np
arr1 = np.array((10, 20, 30, 40, 50))
print(arr1)
```

Example of NumPy

Example: **Create a 2-D array containing two arrays with the values 1,2,3 and 4,5,6:**

```
import numpy as np
arr2 = np.array([[1, 2, 3], [4, 5, 6]])
print(arr2)
```

Example: 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
arr3 = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3],
[4, 5, 6]]])
print(arr3)
```

Example of NumPy

Example: **Check Number of Dimensions:**
NumPy Arrays provides the ndim attribute that returns an integer that tells us how many dimensions the array have.

```
Import numpy as np
arr2 = np.array([[1,2,3],[4,5,6]])
print(arr2)
print("No. of dimensions: ", arr2.ndim)
```

Pandas

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data. Pandas is basically used for data manipulation, wrangling and analysis. It had very good contribution towards data analysis. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyse.

Pandas

The *pandas* package is the most important tool at the disposal of Data Scientists and Analysts working in Python today. The powerful machine learning and glamorous visualization tools may get all the attention, but pandas is the backbone of most data projects. Not only is the pandas library a central component of the data science toolkit but it is used in conjunction with other libraries in that collection.

Key features of pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.

Key features of pandas

- Label-based slicing, indexing and sub setting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Pandas - Environment Setup

If you are using Anaconda distribution, then no need to install Pandas separately as it is already installed with it. You just need to import the package into your Python script with the help of following –

```
import pandas as pd
```

On the other hand, if you are using standard Python distribution then pandas can be installed using popular python package installer, pip.

Pandas - Environment Setup

Open the command prompt and change the directory to your current working directory and then type any one of the following command.

```
pip3 install pandas OR
python -m pip install pandas OR
pip install --user pandas
```

Pandas – Data Structure

Pandas deals with the following three data structures .

1. **Series**
2. **DataFrame**
3. **Panel**

These data structures are built on top of Numpy array, which means they are fast.

The best way to think of these data structures is that the higher dimensional data structure is a container of its lower dimensional data structure.

For example, DataFrame is a container of Series, Panel is a container of DataFrame.

Pandas – Data Structure

Series:

Series is a one-dimensional array like structure with homogeneous data. For example, the following series is a collection of integers 10, 23, 56, 17, 52

10	23	56	17	52

Key Points:

Homogeneous data

Size Immutable

Values of Data Mutable

Pandas – Data Structure

Series:

A pandas Series can be created as follows:

pandas.Series(data, index, dtype, copy)

data: data takes various forms like ndarray, list, constants

index: Index values must be unique and hashable, same length as data. Default **np.arrange(n)** if no index is passed.

dtype: dtype is for data type.

copy: Copy data. Default False

Pandas – Data Structure

Series:

Example:

```
import pandas as pd
import numpy as np
data = np.array(['a','b','c','d'])
s = pd.Series(data,index=[100,101,102,103])
print (s)
```

Pandas – Data Structure

DataFrame:

As a matter of first importance, we are going to discuss from where the idea of a data frame came. The cause of data frame came from serious experimental research in the realm of statistical software. The tabular data is referred by the data frames. Specifically, data frame is a data structure that speaks to cases in which there are various observations(rows) or measurements (columns).

Pandas – Data Structure

DataFrame:

DataFrame is a two-dimensional array with heterogeneous data.

Name	Age	Gender	Rating
Steve	32	Male	3.45
Lia	28	Female	4.6
Vin	45	Male	3.9

Each column represents an attribute and each row represents a person.

Pandas – Data Structure

DataFrame:

Key Points

- Heterogeneous data
- Size Mutable
- Data Mutable
- Potentially columns are of different types
- Size – Mutable
- Labelled axes (rows and columns)
- Can Perform Arithmetic operations on rows and columns

Pandas – Data Structure

DataFrame:

A pandas DataFrame can be created as pandas.DataFrame(data, index, columns, dtype, copy)

data: data takes various forms like ndarray, series, map, lists, dict, constants and also another DataFrame.

index: For the row labels, the Index to be used for the resulting frame is Optional Default np.arange(n) if no index is passed.

Pandas – Data Structure

DataFrame:

A pandas DataFrame can be created as pandas.DataFrame(data, index, columns, dtype, copy)

columns: For column labels, the optional default syntax is - np.arange(n). This is only true if no index is passed.

dtype: Data type of each column.

copy: This command (or whatever it is) is used for copying of data, if the default is False.

Pandas – Data Structure

DataFrame:

Example_1:

```
import pandas as pd
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print (df)
```

Example_2:

```
import pandas as pd
data2 = [['Alex',10],['Bob',12],['Clarke',13]]
df1 = pd.DataFrame(data2,columns=['Name','Age'])
print (df1)
```

Uplatz

Pandas – Data Structure

DataFrame: Create a DataFrame from Dictionary

Example_1:

```
import pandas as pd
data3 = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],
'Age':[28,34,29,42]}
df2 = pd.DataFrame(data3)
print (df2)
```

Pandas – Data Structure

Panel:

Panel is a three-dimensional data structure with heterogeneous data. It is hard to represent the panel in graphical representation. But a panel can be illustrated as a container of DataFrame.

Key Points

Heterogeneous data

Size Mutable

Data Mutable

The Panel class is removed from pandas

Data Analysis using Pandas

Reading CSV Files With pandas:

Reading the CSV into a pandas DataFrame is quick and straightforward:

```
import pandas as pd
df5 = pd.read_csv('hrdata.csv')
print(df5)
print(df5['Name']) #Reading the data from column
"Name"
```

Thank you

Uplatz