Chapter 2

Data Handling Using Pandas - I

> "If you don't think carefully, you might believe that programming is just typing statements in a programming language."
>
> — W. Cunningham

12149CH02

In this chapter

- *Introduction to Python Libraries*
- *Series*
- *DataFrame*
- *Importing and Exporting Data between CSV Files and DataFrames*
- *Pandas Series Vs NumPy ndarray*

2.1 Introduction to Python Libraries

Python libraries contain a collection of built-in modules that allow us to perform many actions without writing detailed programs for it. Each library in Python contains a large number of modules that one can import and use.

NumPy, Pandas and Matplotlib are three well-established Python libraries for scientific and analytical use. These libraries allow us to manipulate, transform and visualise data easily and efficiently.

NumPy, which stands for 'Numerical Python', is a library we discussed in class XI. Recall that, it is a package that can be used for numerical data analysis and

NOTES

scientific computing. NumPy uses a multidimensional array object and has functions and tools for working with these arrays. Elements of an array stay together in memory, hence, they can be quickly accessed.

PANDAS (PANel DAta) is a high-level data manipulation tool used for analysing data. It is very easy to import and export data using Pandas library which has a very rich set of functions. It is built on packages like NumPy and Matplotlib and gives us a single, convenient place to do most of our data analysis and visualisation work. Pandas has three important data structures, namely – Series, DataFrame and Panel to make the process of analysing data organised, effective and efficient.

The Matplotlib library in Python is used for plotting graphs and visualisation. Using Matplotlib, with just a few lines of code we can generate publication quality plots, histograms, bar charts, scatterplots, etc. It is also built on Numpy, and is designed to work well with Numpy and Pandas.

You may think what the need for Pandas is when NumPy can be used for data analysis. Following are some of the differences between Pandas and Numpy:

1. A Numpy array requires homogeneous data, while a Pandas DataFrame can have different data types (float, int, string, datetime, etc.).
2. Pandas have a simpler interface for operations like file loading, plotting, selection, joining, GROUP BY, which come very handy in data-processing applications.
3. Pandas DataFrames (with column names) make it very easy to keep track of data.
4. Pandas is used when data is in Tabular Format, whereas Numpy is used for numeric array based data manipulation.

2.1.1. Installing Pandas

Installing Pandas is very similar to installing NumPy. To install Pandas from command line, we need to type in:

```
pip install pandas
```

Note that both NumPy and Pandas can be installed only when Python is already installed on that system. The same is true for other libraries of Python.

2.1.2. Data Structure in Pandas

A data structure is a collection of data values and operations that can be applied to that data. It enables efficient storage, retrieval and modification to the data. For example, we have already worked with a data structure ndarray in NumPy in Class XI. Recall the ease with which we can store, access and update data using a NumPy array. Two commonly used data structures in Pandas that we will cover in this book are:

- Series
- DataFrame

2.2 SERIES

A Series is a one-dimensional array containing a sequence of values of any data type (int, float, list, string, etc) which by default have numeric data labels starting from zero. The data label associated with a particular value is called its index. We can also assign values of other data types as index. We can imagine a Pandas Series as a column in a spreadsheet. Example of a series containing names of students is given below:

```
Index     Value
0         Arnab
1         Samridhi
2         Ramit
3         Divyam
4         Kritika
```

2.2.1 Creation of Series

There are different ways in which a series can be created in Pandas. To create or use series, we first need to import the Pandas library.

(A) Creation of Series from Scalar Values

A Series can be created using scalar values as shown in the example below:

```
>>> import pandas as pd   #import Pandas with alias pd
>>> series1 = pd.Series([10,20,30])  #create a Series
>>> print(series1)  #Display the series
```

Output:

```
0   10
1   20
2   30
dtype: int64
```