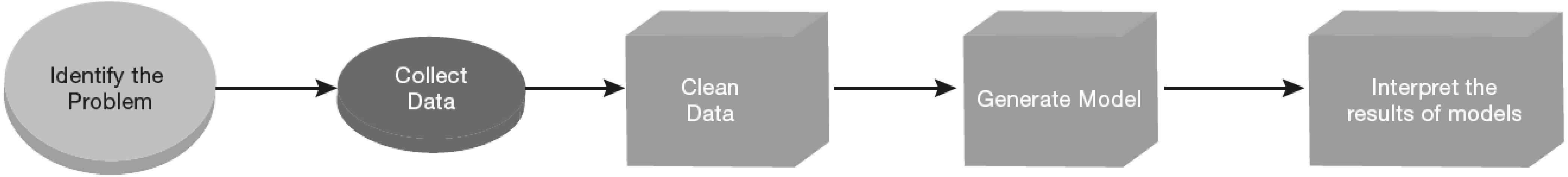
Introduction to data analytics with Python

Introduction to Data Analytics

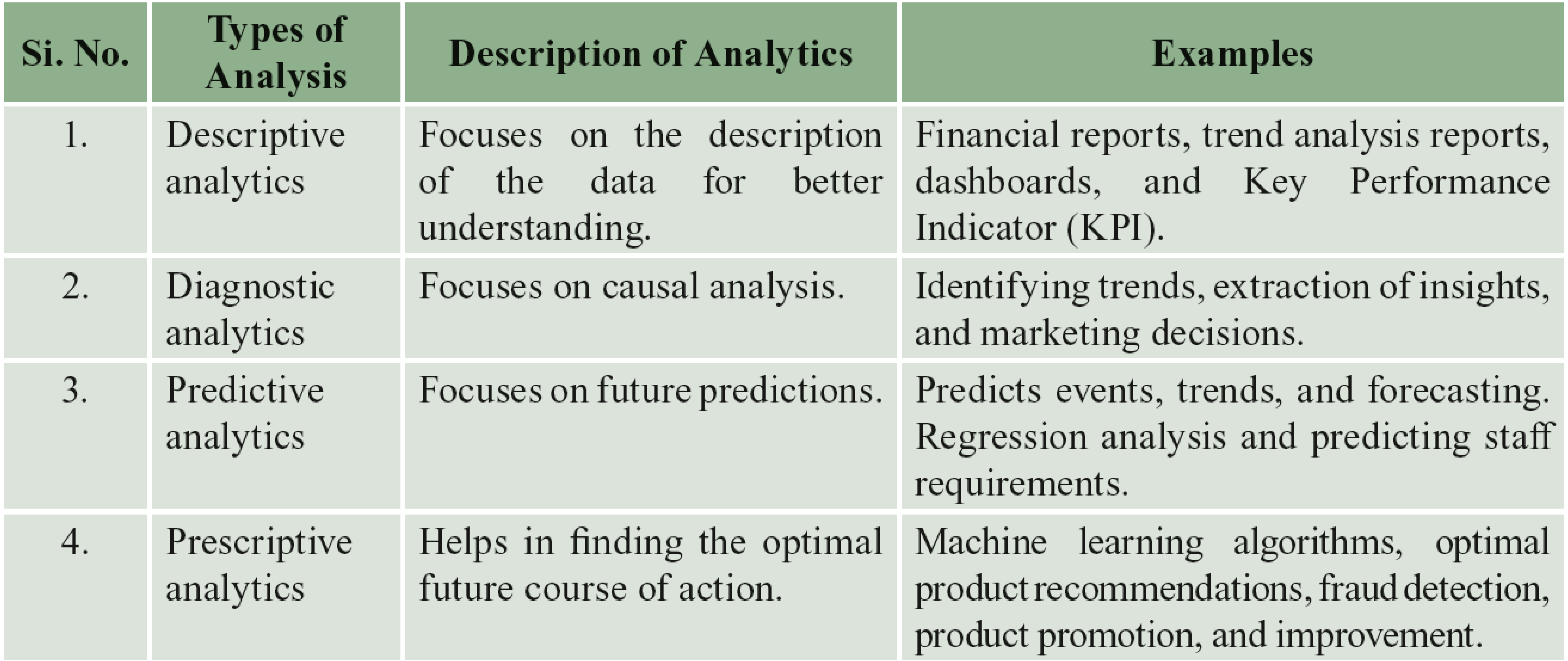
- Business organizations use huge amounts of data.
- Modern technology makes it easier to capture, process, store, distribute, and transmit digital information.
- The data are available in a variety of forms such as flat files, databases, records, and digital archives.
- Most of these data are useful for making decisions. Data have to be converted to information, which is processed data.
- Information includes patterns, associations, or relationships among data. For example, the sales data can be analysed to extract information like popularity of a product launched by an organization in the market and which product needs to phased out from the market.

Steps of Data Analytics

- Data Analytics is a general term and data analysis is a part of it.
- Data analytics refers to the process of data collection, pre-processing, and analysis of such data.



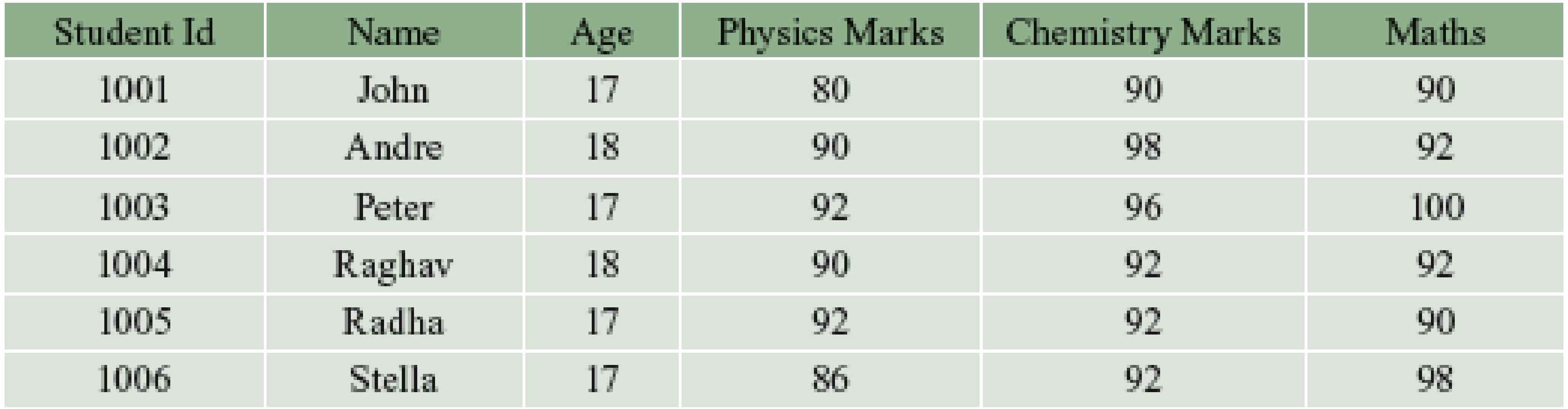
Types of Data Analytics



Si. No.	Types of Analysis	Description of Analytics	Examples
1.	Descriptive analytics	Focuses on the description of the data for better understanding.	Financial reports, trend analysis reports, dashboards, and Key Performance Indicator (KPI).
2.	Diagnostic analytics	Focuses on causal analysis.	Identifying trends, extraction of insights, and marketing decisions.
3.	Predictive analytics	Focuses on future predictions.	Predicts events, trends, and forecasting. Regression analysis and predicting staff requirements.
4.	Prescriptive analytics	Helps in finding the optimal future course of action.	Machine learning algorithms, optimal product recommendations, fraud detection, product promotion, and improvement.

Dataset and Data Analysis

- A dataset is collection of data objects. The data objects may have many attributes. An attribute can be defined as the property or characteristics of an object. For Example.



Student Id	Name	Age	Physics Marks	Chemistry Marks	Maths
1001	John	17	80	90	90
1002	Andre	18	90	98	92
1003	Peter	17	92	96	100
1004	Raghav	18	90	92	92
1005	Radha	17	92	92	90
1006	Stella	17	86	92	98

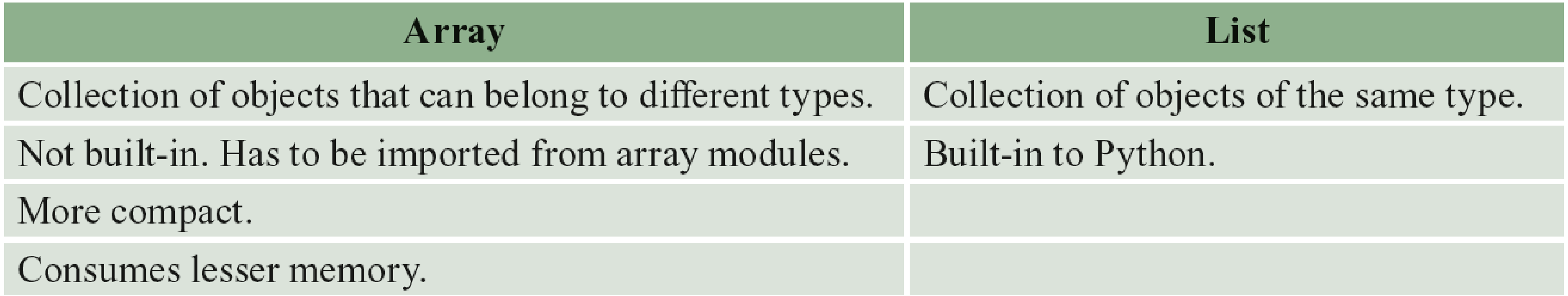
Data Analytic Process

- Cross Industry Standard Process–Data Mining (DM) model is the popular model that is used for data analytics.
- This model involves six steps. The steps are listed below.

1. Understanding the Business
2. Understanding the Data
3. Preparation of data and Data Pre-processing
4. Modelling
5. Evaluation of a Model
6. Deployment

Arrays in Python

- Most programming languages provide a data structure called arrays. In Python, array is a package and it is different from "core" python lists.
- Arrays are similar to lists and the differences between arrays lists are given



Array	List
Collection of objects that can belong to different types.	Collection of objects of the same type.
Not built-in. Has to be imported from array modules.	Built-in to Python.
More compact.	
Consumes lesser memory.	

Importing and creating an Array

Three ways of importing are:

1) import array (using modules)

```
>>> import array
```

2) import array as arr (using alias)

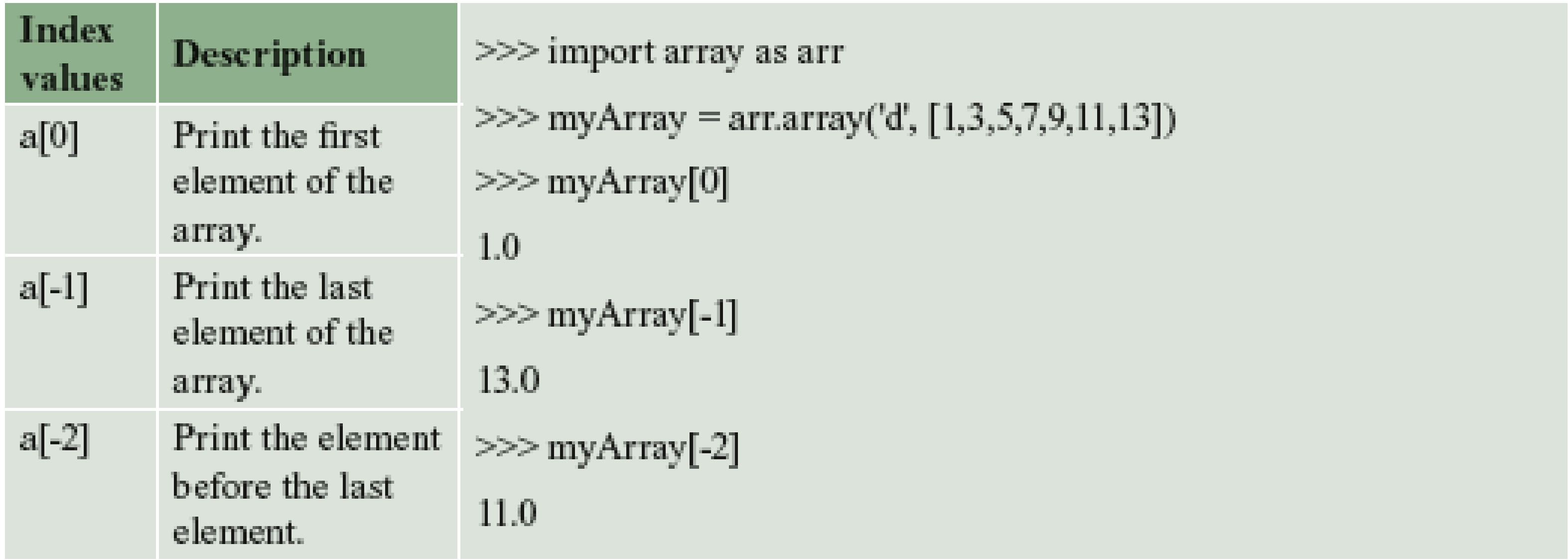
```
>>> import array as arr
```

Here, arr is called an alias.

3) import array * (import all the functionalities of the array module) import array * would import all the functionalities of the array. The arrays are indexed from 0 to n-1. n is the total number of elements of the array. The first element is indexed as zero.

Array Operations

- 1. Arrays are mutable. The array is a container in Python that stores objects of different types. It is also a fundamental data structure that is useful for processing data. It works similarly to lists and stores objects of similar types.



Index values	Description	>>> import array as arr

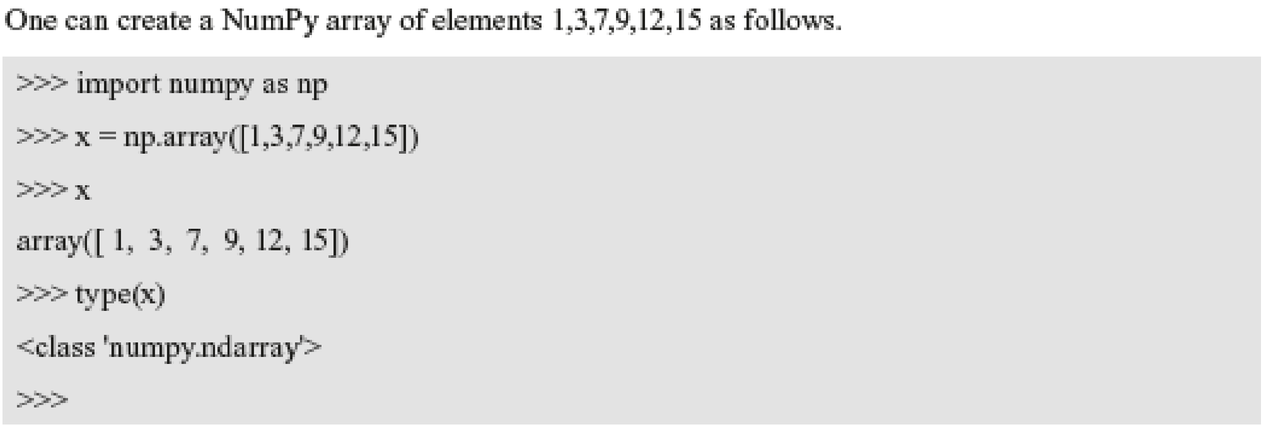
a[0]	Print the first element of the array.	>>> myArray = arr.array('d', [1,3,5,7,9,11,13])
		>>> myArray[0]
		1.0

a[-1]	Print the last element of the array.	>>> myArray[-1] 13.0
a[-2]	Print the element before the last element.	>>> myArray[-2] 11.0

Introduction to NumPy

- NumPy is a library of Python, and it is a shorthand form of numerical Python.
- NumPy provides an array data structure and helps in numerical analysis. NumPy is used to manipulate arrays.
- The manipulation includes mathematical and logical operations.

Array Creation in NumPy



One can create a NumPy array of elements 1,3,7,9,12,15 as follows.

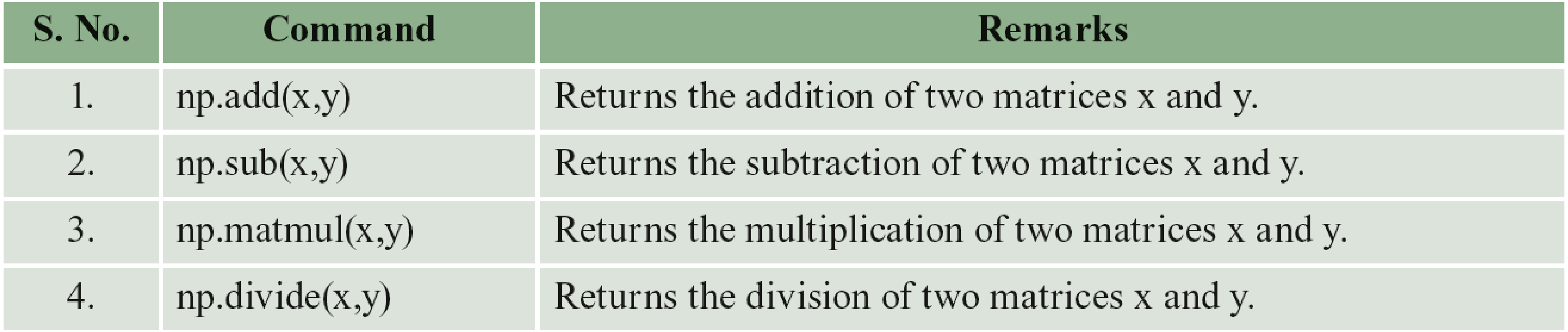
```
>>> import numpy as np
>>> x = np.array([1,3,7,9,12,15])
>>> x
array([ 1,  3,  7,  9, 12, 15])
>>> type(x)
<class 'numpy.ndarray'>
>>>
```

NumPy Properties

- The important characteristics of defining a NumPy array are listed below.
- Data type
- Item size
- Shape–dimensions
- Data
- Data types are integers, unit, float, and complex; other data types are Boolean, string, date time, and Python objects. Item size is the memory requirement of data elements in bytes.
- The shape is the dimension of the array. Data are the elements of a NumPy array.

Arithmetic Operation on NumPy

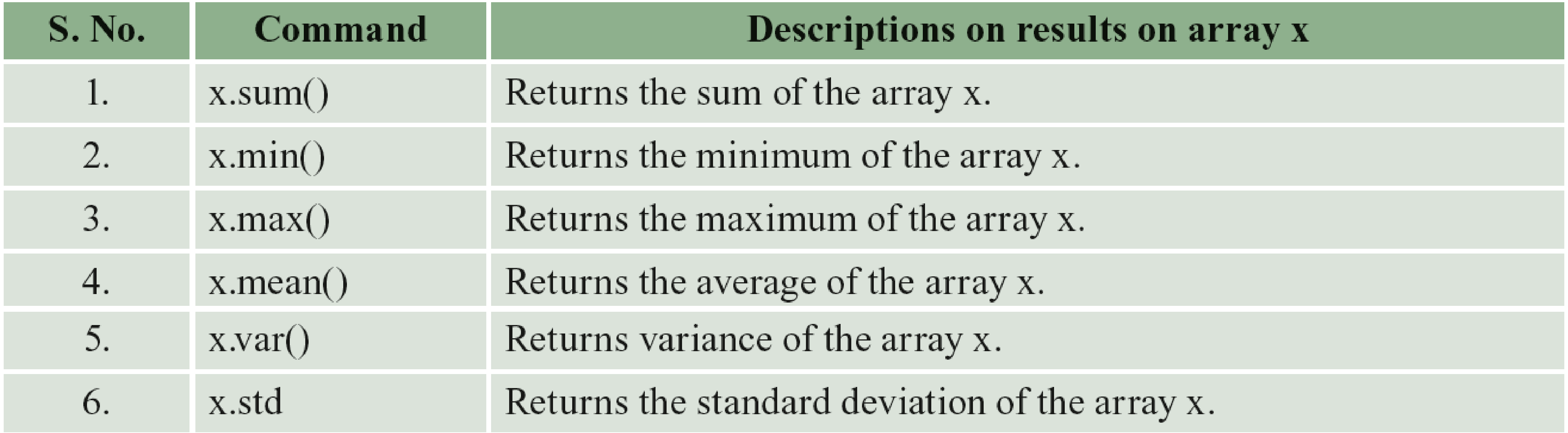
- One can create an array and apply the following commands to perform statistical operations. Array operations are shown in Table



S. No.	Command	Remarks
1.	np.add(x,y)	Returns the addition of two matrices x and y.
2.	np.sub(x,y)	Returns the subtraction of two matrices x and y.
3.	np.matmul(x,y)	Returns the multiplication of two matrices x and y.
4.	np.divide(x,y)	Returns the division of two matrices x and y.

Data Analysis Using NumPy

- Descriptive analytics is about describing the main features of the data. Descriptive analytics only focuses on the description part of the data and not the inference part.
- Some of the descriptive statistics are given in Table



S. No.	Command	Descriptions on results on array x
1.	x.sum()	Returns the sum of the array x.
2.	x.min()	Returns the minimum of the array x.
3.	x.max()	Returns the maximum of the array x.
4.	x.mean()	Returns the average of the array x.
5.	x.var()	Returns variance of the array x.
6.	x.std	Returns the standard deviation of the array x.

Pandas

- Pandas is a name from "panel data" and was designed by Wes McKinney in 2008. Pandas is used for data manipulation and analysis. The core of pandas is their data structures. It provides three data structures.

1) series 1D (Column)

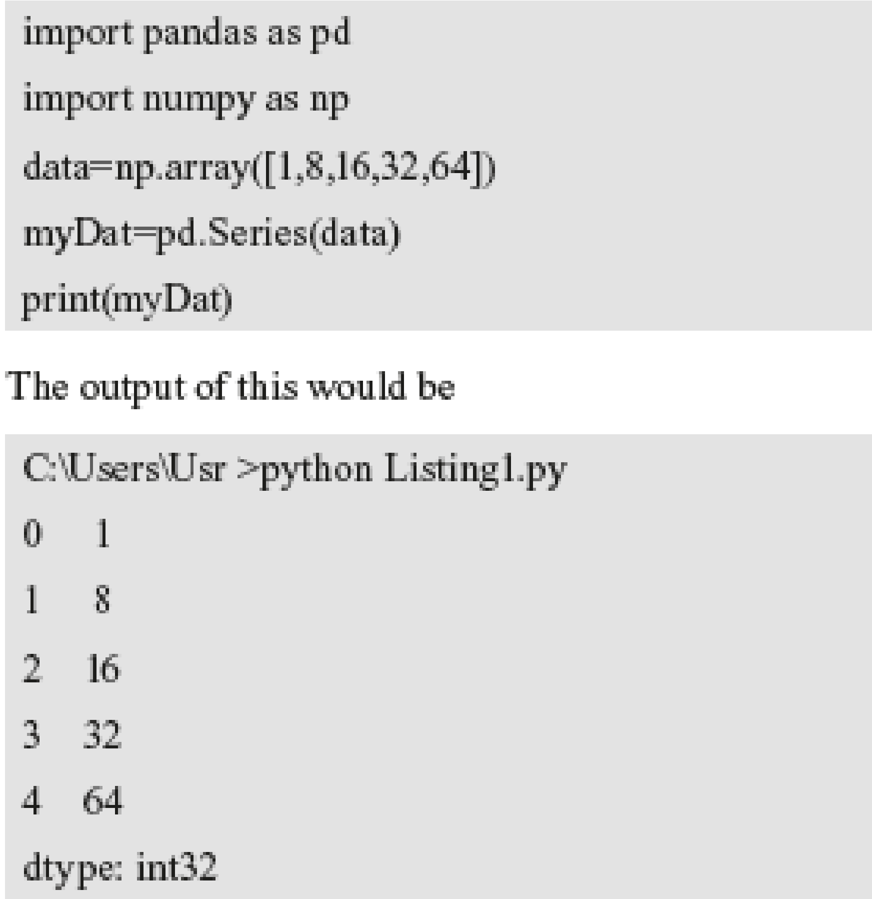
2) data frame 2D (Single Sheet)

3) Panel 3D (Multiple Sheets)

- A panel may have multiple sheets (df) and every df may have many columns (series)

Series in Pandas

- One dimensional series can be created as



```
import pandas as pd
import numpy as np
data=np.array([1,8,16,32,64])
myDat=pd.Series(data)
print(myDat)
```

The output of this would be

```
C:\Users\Usr >python Listing1.py
0    1
1    8
2   16
3   32
4   64
dtype: int32
```

Data Visualization Using Pandas and Matplotlib

- The process of visualizing data to identify patterns is called data visualization. A pattern is a trend or repetition of some data. By visualizing, one can observe some trends that may be helpful in business decision making.
- For example, by observing the trend, one can take some important decisions. Data visualization is useful in many domains such as

1) Data science

2) Machine learning

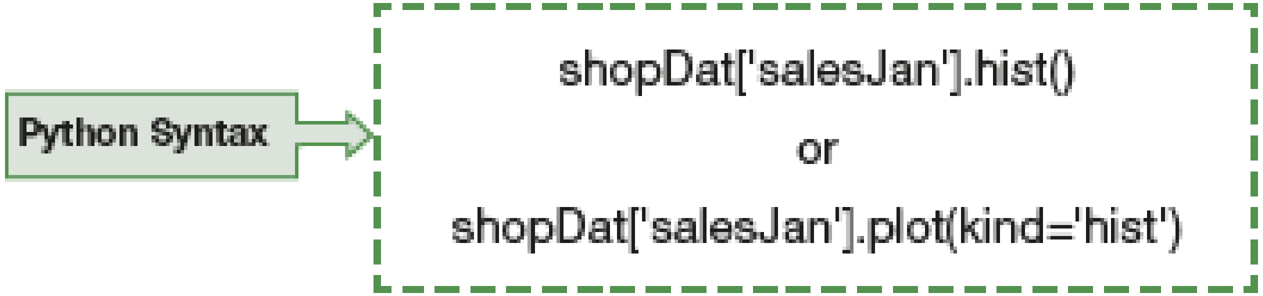
3) Data mining

4) Data analytics

- Matplotlib is one of the most important libraries. One can import the pyplot module from Matplotlib import library.

Pandas and Matplotlib Visualization

- Pandas can be used for data visualization also. Pandas can read comma-separated values (CSV),
- Excel and Tab-Separated values 0 files into data frame. The first requirement of Pandas visualization is that the data files need to be imported into data frame. Let us assume that the following dataset is available for data.
- This is followed by the dot operator. This is followed by the name of the plot. For example, the histogram of salesJan in the above table can be done as follows.



```
shopDat['salesJan'].hist()
or
shopDat['salesJan'].plot(kind='hist')
```

Scikit-Learn and Data Analysis

- Regression analysis is used to model the relationship between one or more independent variables and a dependent variable. Regression analysis discovers the relation between the variables. In the simplest form, the model can be created as

$$Y = a0 + a1 * x$$

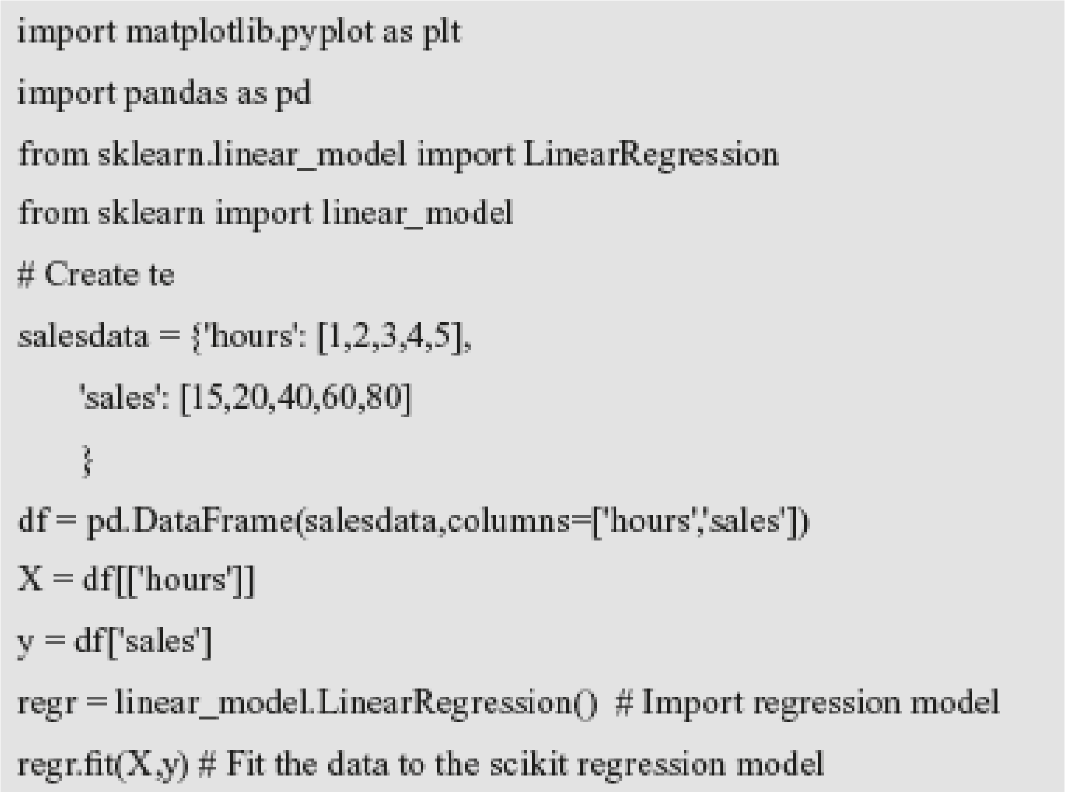
Here, a0 is the intercept that represents the bias, and a1 represents the slope. These are called regression coefficients. This specifies the Y-Intercept and slope of the line. The values of estimates of a and b are given as follows.

$$a_1 = \frac{Avg(xy) - Avg(x)Avg(y)}{Avg(x^2) - \left(Avg(x)\right)^2}$$

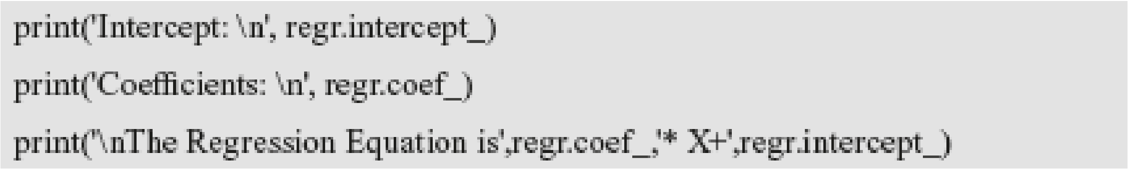
$$a_0 = Avg(y) - a_1 \times Avg(x)$$

Scikit-Learn for Regression

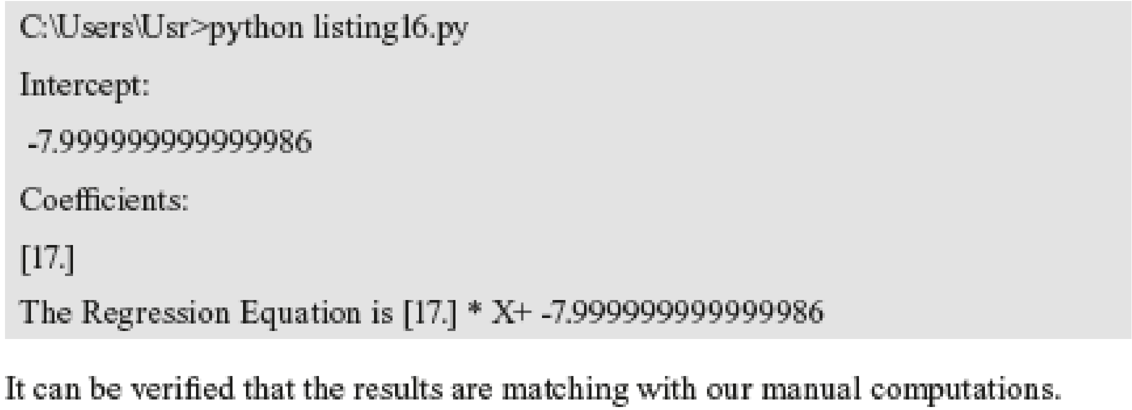
- The above calculations can be done by scikit-learn. Scikit-learn is a third-party Python package that provides the routines for regression. The following is the program for linear regression.



```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn import linear_model
# Create te
salesdata = {'hours': [1,2,3,4,5],
        'sales': [15,20,40,60,80]
        }
df = pd.DataFrame(salesdata,columns=['hours','sales'])
X = df[['hours']]
y = df['sales']
regr = linear_model.LinearRegression()  # Import regression model
regr.fit(X,y) # Fit the data to the scikit regression model
```



```
print('Intercept: \n', regr.intercept_)
print('Coefficients: \n', regr.coef_)
print('\nThe Regression Equation is',regr.coef_,'* X+',regr.intercept_)
```



```
C:\Users\Usr>python listing16.py
Intercept:
 -7.999999999999986
Coefficients:
[17.]
The Regression Equation is [17.] * X+ -7.999999999999986
```

It can be verified that the results are matching with our manual computations.