Ex.No. 1	Download, install and explore the features of NumPy, SciPy, Jupyter, Statsmodelsand Pandaspackages
Date :	

Aim:

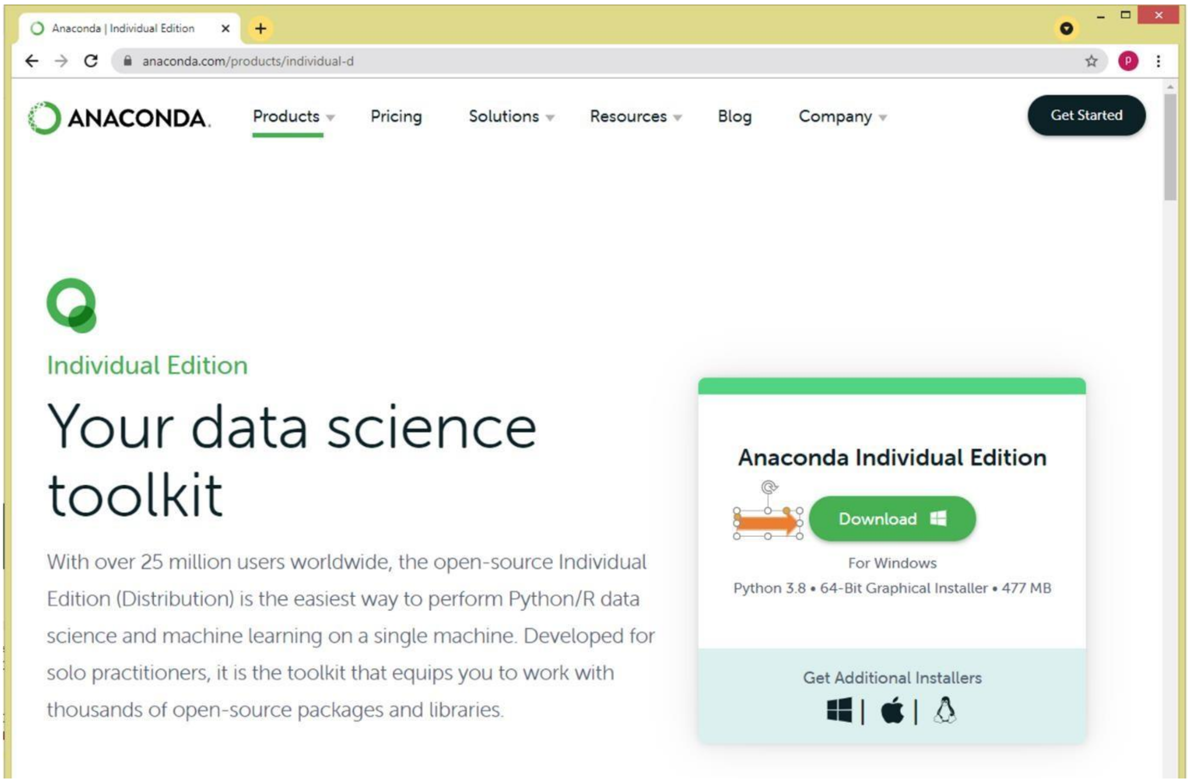
To Download, Install and Explore the features of NumPy, Scipy, Jupyter , Statsmodels and Pandas Packages.

Download & Install Anaconda Distribution

Follow the below step-by-step instructions to install Anacondadistribution.

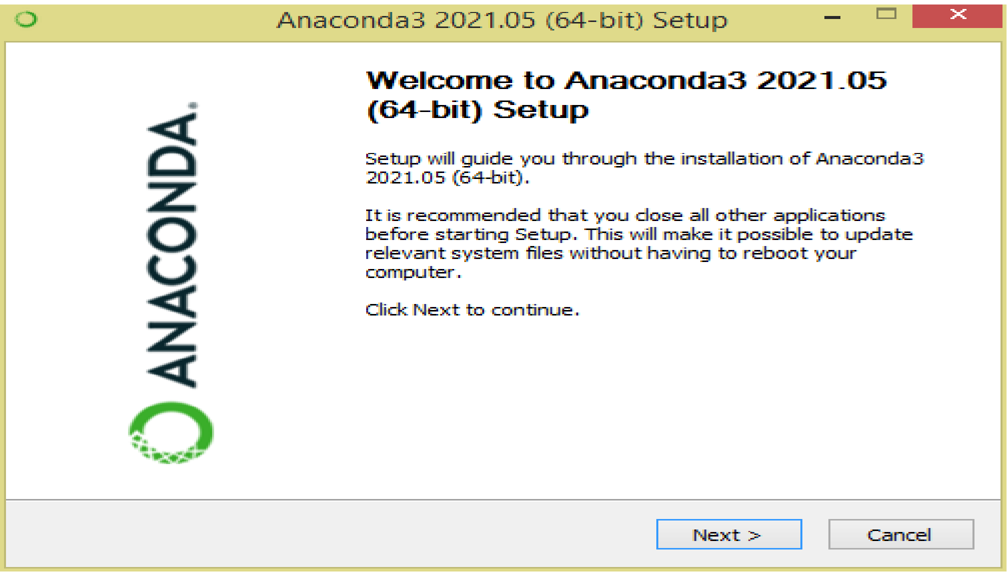
Download Anaconda Distribution

Go to https://anaconda.com/ and select **Anaconda Individual Edition** to download the latest version of Anaconda. This downloads the .exe file to the windows download folder.


Anaconda | Individual Edition
anaconda.com/products/individual-d
ANACONDA.
Products
Pricing
Solutions
Resources
Blog
Company
Get Started
Individual Edition
Your data science toolkit
With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.
Anaconda Individual Edition
Download
For Windows
Python 3.8 • 64-Bit Graphical Installer • 477 MB
Get Additional Installers

Install Anaconda

By double-clicking the .exe file starts the Anaconda installation. Followthe below screen shot’s and complete the installation

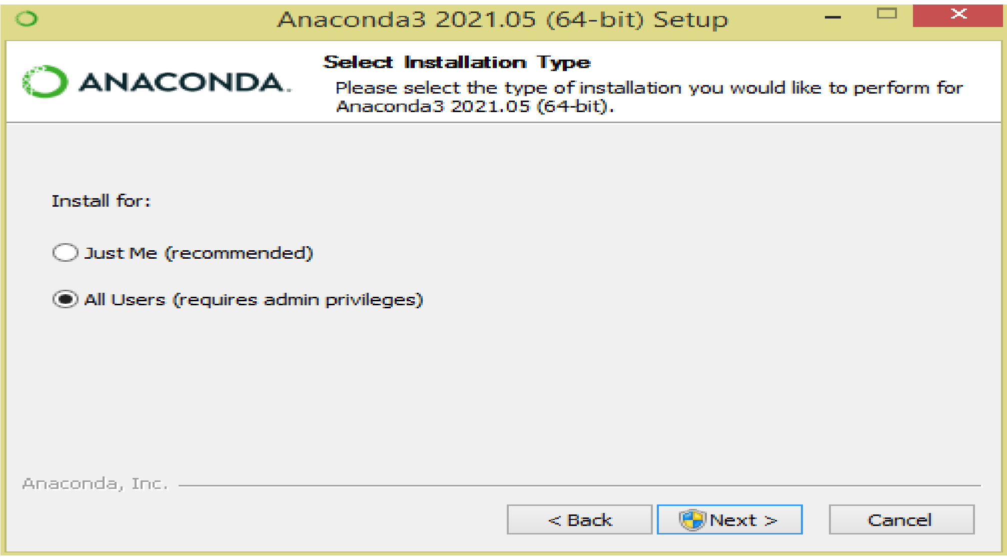


Welcome to Anaconda3 2021.05 (64-bit) Setup

Setup will guide you through the installation of Anaconda3 2021.05 (64-bit).

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.



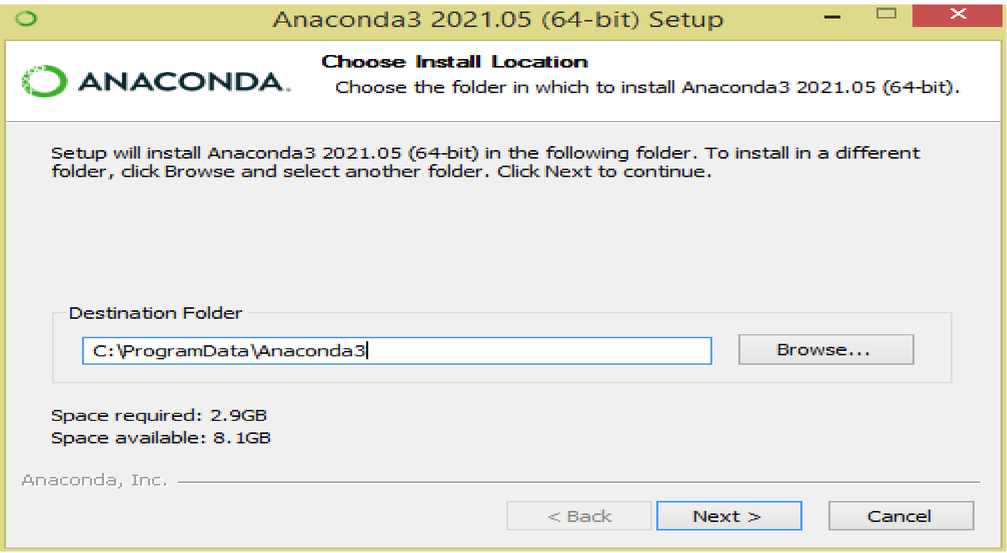
Select Installation Type

Please select the type of installation you would like to perform for Anaconda3 2021.05 (64-bit).

Install for:

○ Just Me (recommended)

● All Users (requires admin privileges)



Choose Install Location

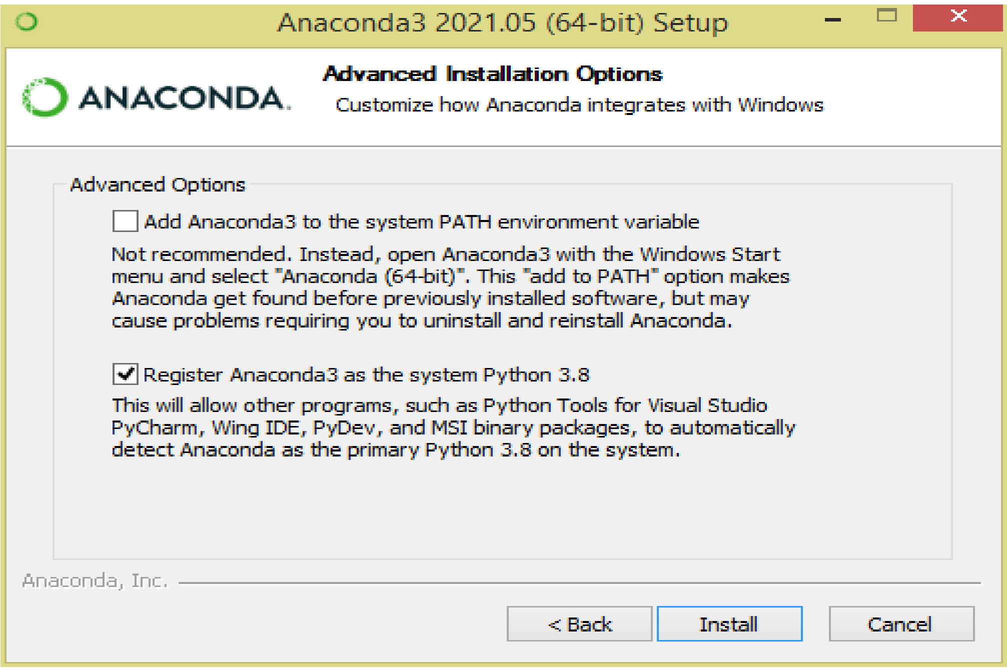
Choose the folder in which to install Anaconda3 2021.05 (64-bit).

Setup will install Anaconda3 2021.05 (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

C:\ProgramData\Anaconda3

Space required: 2.9GB
Space available: 8.1GB



Advanced Installation Options

Customize how Anaconda integrates with Windows

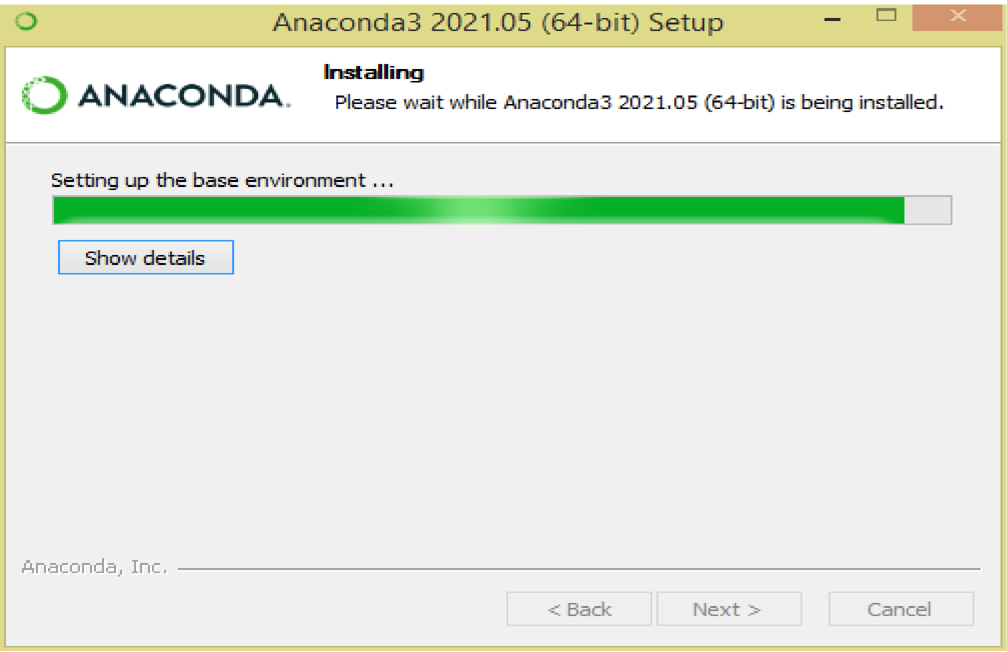
Advanced Options

□ Add Anaconda3 to the system PATH environment variable

Not recommended. Instead, open Anaconda3 with the Windows Start menu and select "Anaconda (64-bit)". This "add to PATH" option makes Anaconda get found before previously installed software, but may cause problems requiring you to uninstall and reinstall Anaconda.

☑ Register Anaconda3 as the system Python 3.8

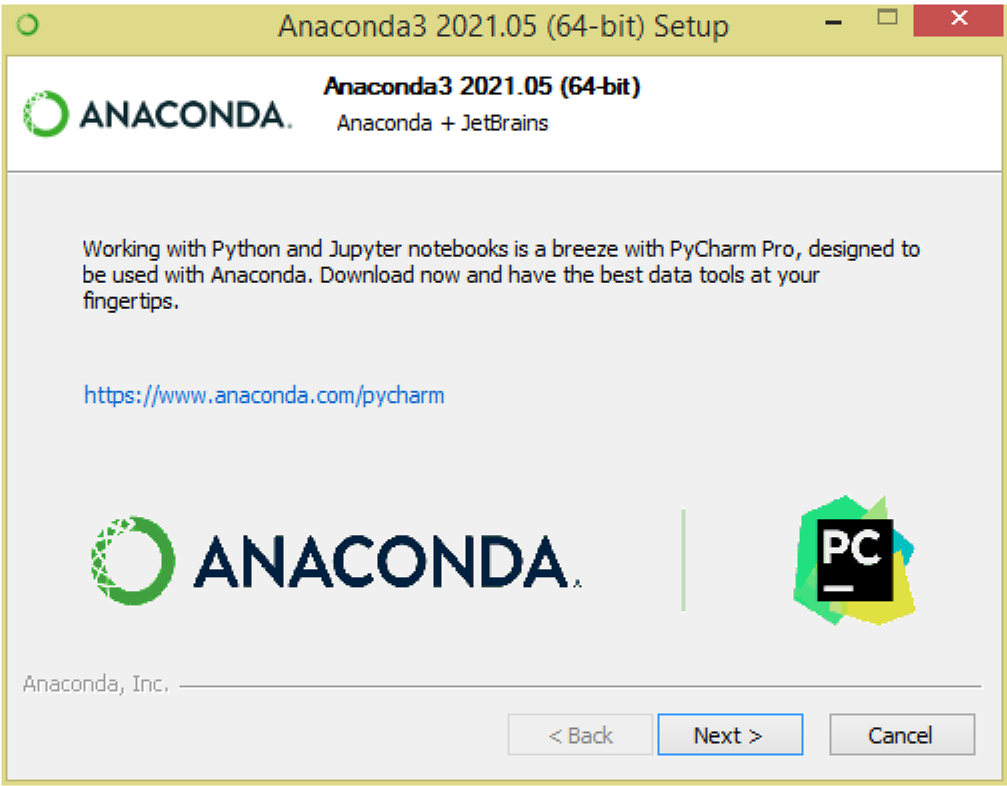
This will allow other programs, such as Python Tools for Visual Studio PyCharm, Wing IDE, PyDev, and MSI binary packages, to automatically detect Anaconda as the primary Python 3.8 on the system.



Installing

Please wait while Anaconda3 2021.05 (64-bit) is being installed.

Setting up the base environment ...



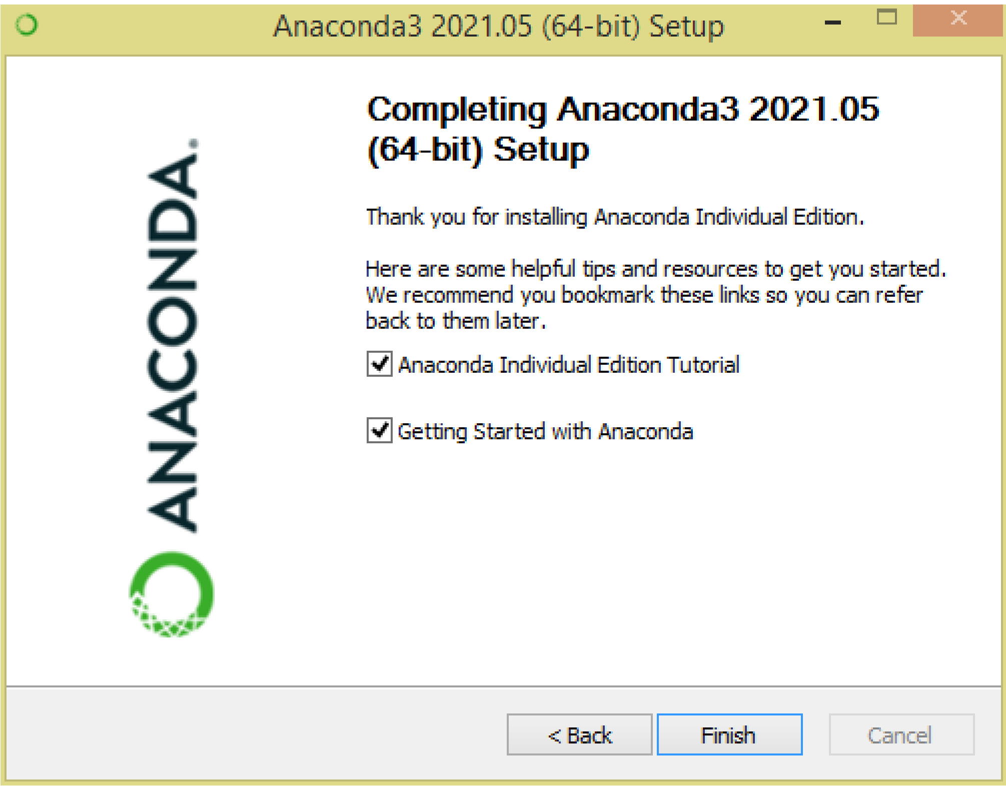
Anaconda3 2021.05 (64-bit) Setup

Anaconda3 2021.05 (64-bit)

Anaconda + JetBrains

Working with Python and Jupyter notebooks is a breeze with PyCharm Pro, designed to be used with Anaconda. Download now and have the best data tools at your fingertips.

https://www.anaconda.com/pycharm



Anaconda3 2021.05 (64-bit) Setup

Completing Anaconda3 2021.05 (64-bit) Setup

Thank you for installing Anaconda Individual Edition.

Here are some helpful tips and resources to get you started. We recommend you bookmark these links so you can refer back to them later.

☑ Anaconda Individual Edition Tutorial

☑ Getting Started with Anaconda

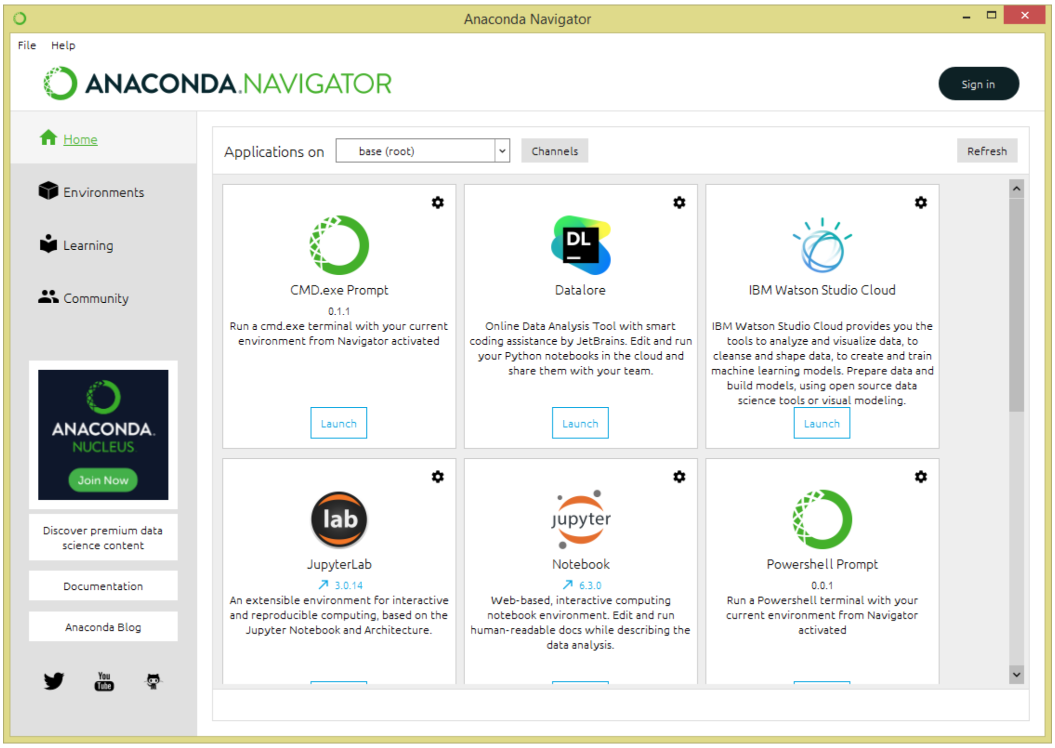
This finishes the installation of Anaconda distribution, now let’s see howto create an environment and install Jupyter Notebook.

Create Anaconda Environment from Navigator

A conda environment is **a directory that contains a specific collection ofconda packages that you have installed**. For example, you may have one environment with NumPy 1.7 and its dependencies, and another environmentwith NumPy 1.6 for legacy testing.
https://conda.io/docs/using/envs.html

Open Anaconda Navigator

Open Anaconda Navigator from windows start or by searching it. Anaconda Navigator is a UI application where you can control theAnaconda packages, environment e.t.c



CMD.exe Prompt: Run a cmd.exe terminal with your current environment from Navigator activated

Datalore: Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud: IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

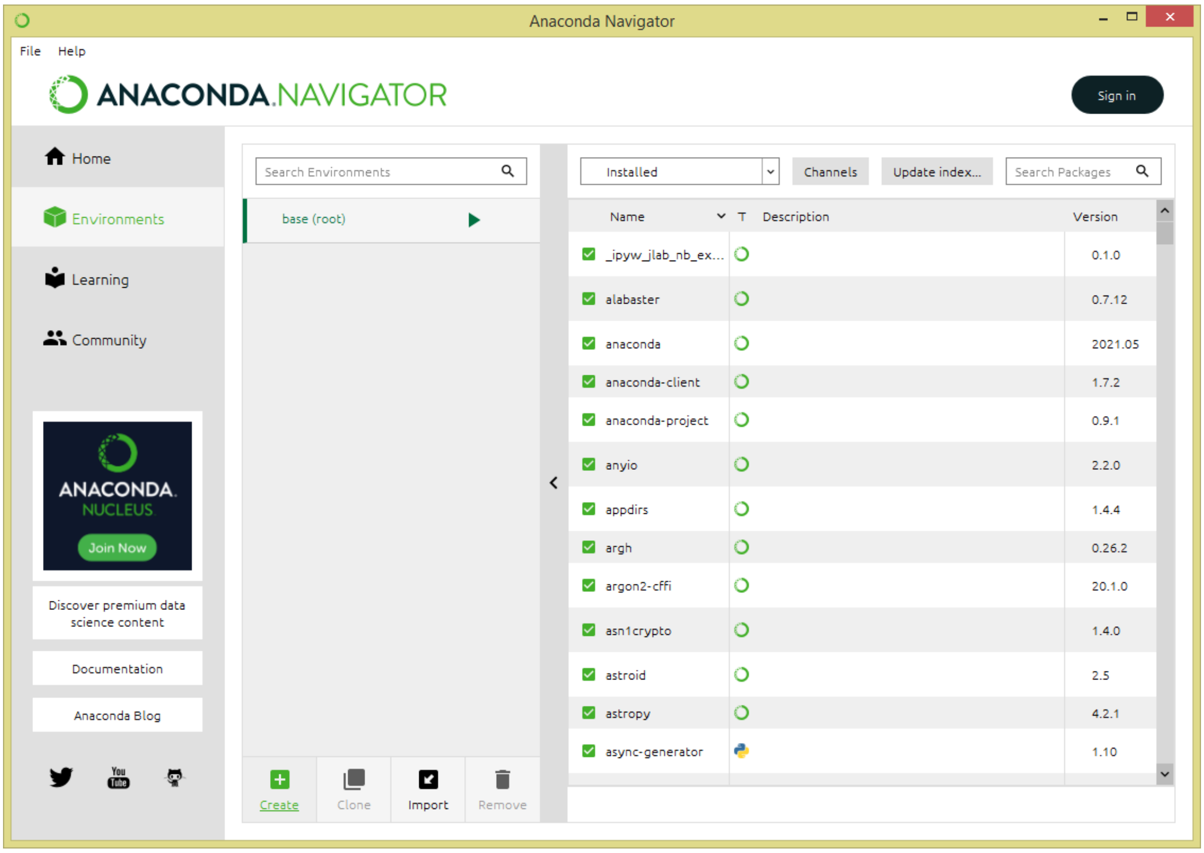
JupyterLab: An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook: Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Powershell Prompt: Run a Powershell terminal with your current environment from Navigator activated

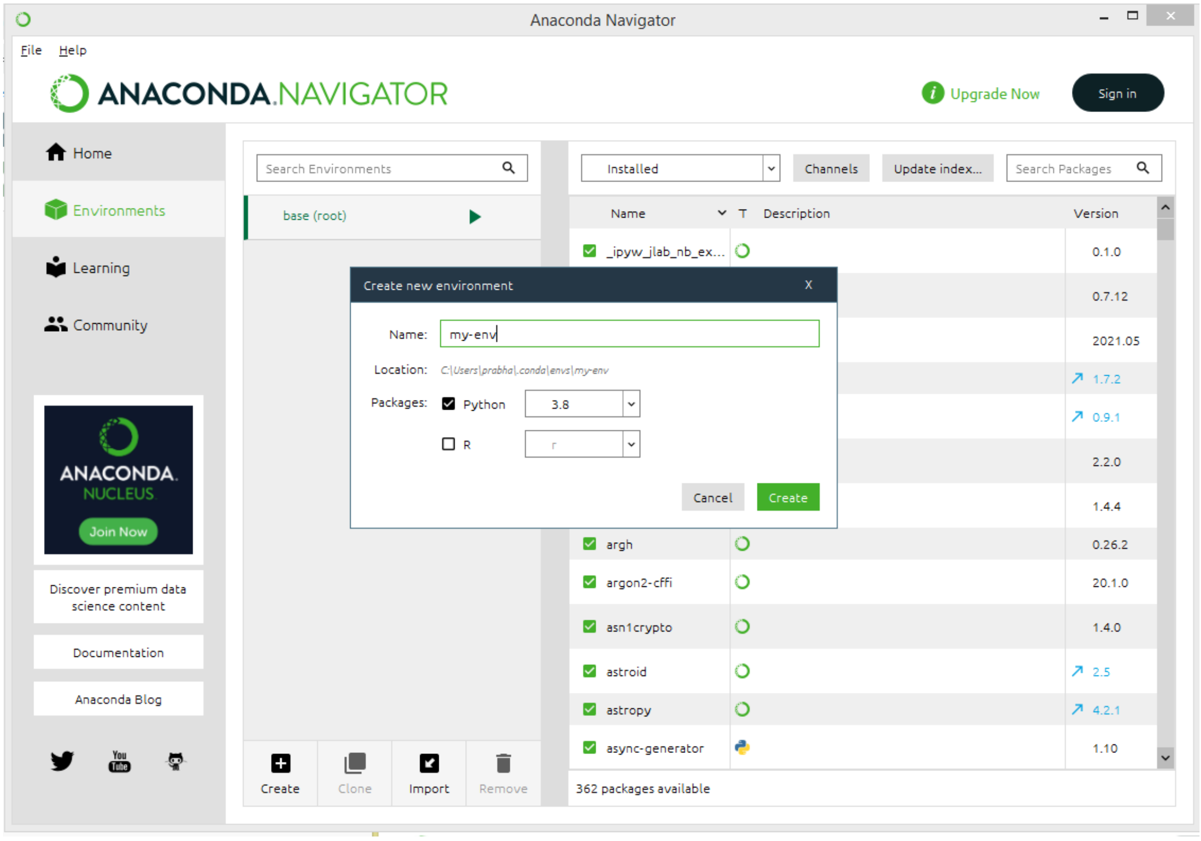
Create an Environment to Run Jupyter Notebook

This is optional but recommended to create an environment before you proceed. This gives complete segregation of different package installs fordifferent projects you would be working on. If you already have an environment, you can use it too.



Name	T	Description	Version
☑ _ipyw_jlab_nb_ex...	○		0.1.0
☑ alabaster	○		0.7.12
☑ anaconda	○		2021.05
☑ anaconda-client	○		1.7.2
☑ anaconda-project	○		0.9.1
☑ anyio	○		2.2.0
☑ appdirs	○		1.4.4
☑ argh	○		0.26.2
☑ argon2-cffi	○		20.1.0
☑ asn1crypto	○		1.4.0
☑ astroid	○		2.5
☑ astropy	○		4.2.1
☑ async-generator			1.10

select + Create icon at the bottom of the screen to create an Anaconda environment.



Create new environment

Name: my-env

Location: C:\Users\prabha\.conda\envs\my-env

Packages: ☑ Python 3.8

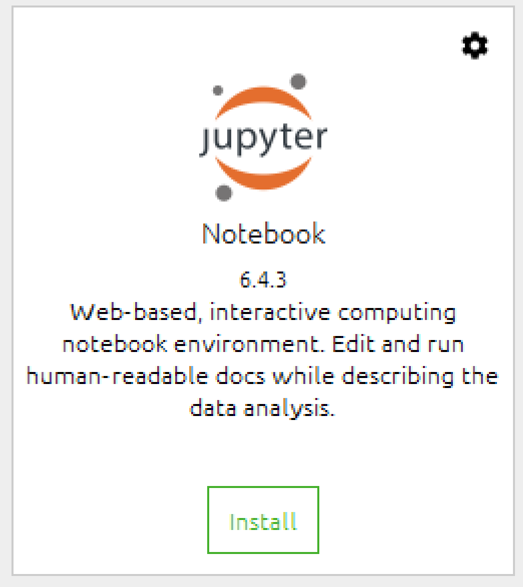
□ R r

Name	T	Description	Version
☑ _ipyw_jlab_nb_ex...	○		0.1.0
			0.7.12
			2021.05
			↗ 1.7.2
			↗ 0.9.1
			2.2.0
			1.4.4
☑ argh	○		0.26.2
☑ argon2-cffi	○		20.1.0
☑ asn1crypto	○		1.4.0
☑ astroid	○		↗ 2.5
☑ astropy	○		↗ 4.2.1
☑ async-generator			1.10

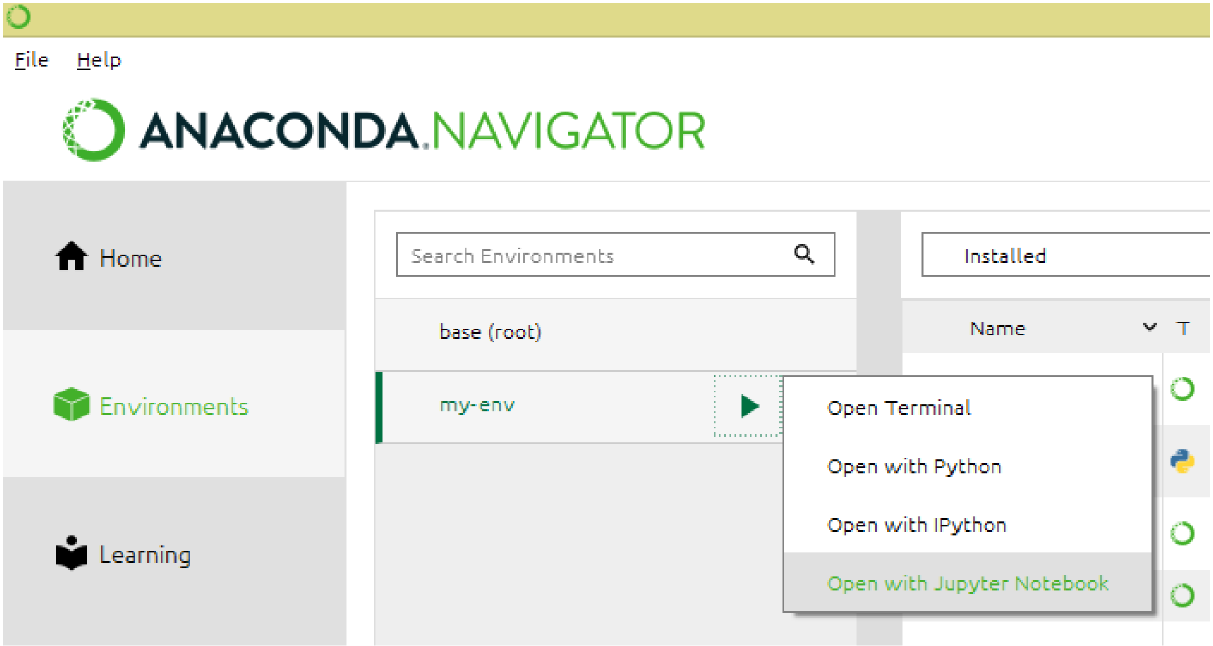
362 packages available

Install and Run Jupyter Notebook

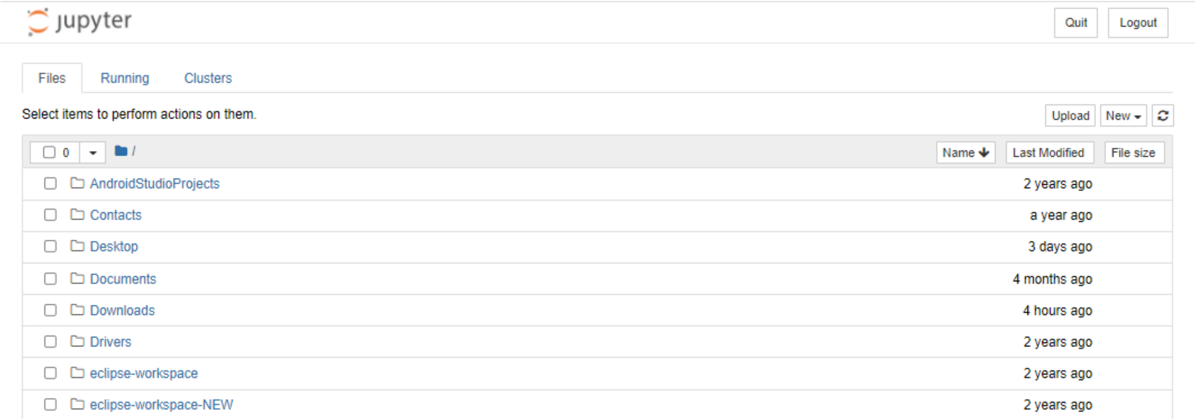
Once you create the anaconda environment, go back to the Home page on Anaconda Navigator and install Jupyter Notebook from an application on the right panel.


jupyter
Notebook
6.4.3
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Install

It will take a few seconds to install Jupyter to your environment, once the install completes, you can open Jupyter from the same screen or by accessing **Anaconda Navigator -> Environments -> your environment** (mine pandas-tutorial) -> select **Open With JupyterNotebook**.



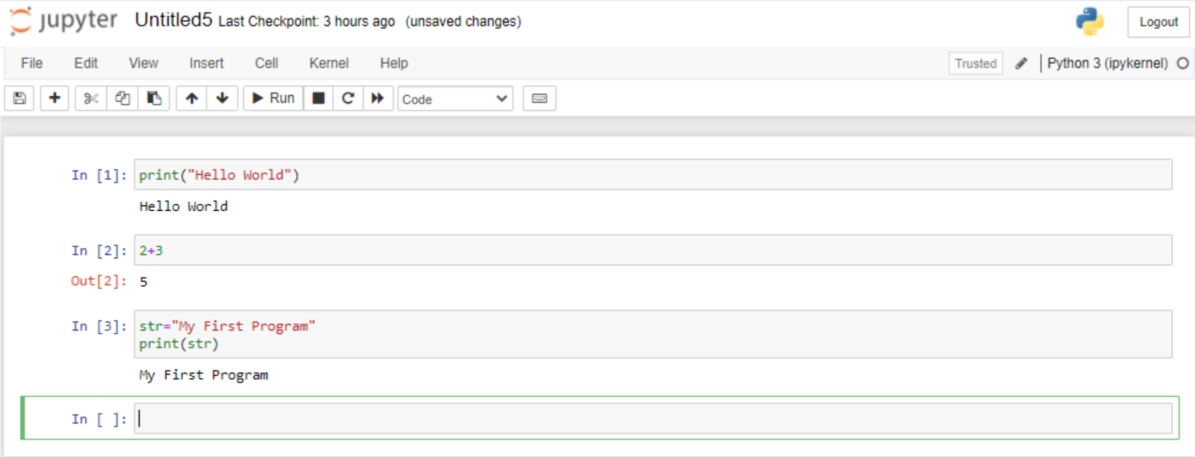
This opens up Jupyter Notebook in the default browser.



Select items to perform actions on them.

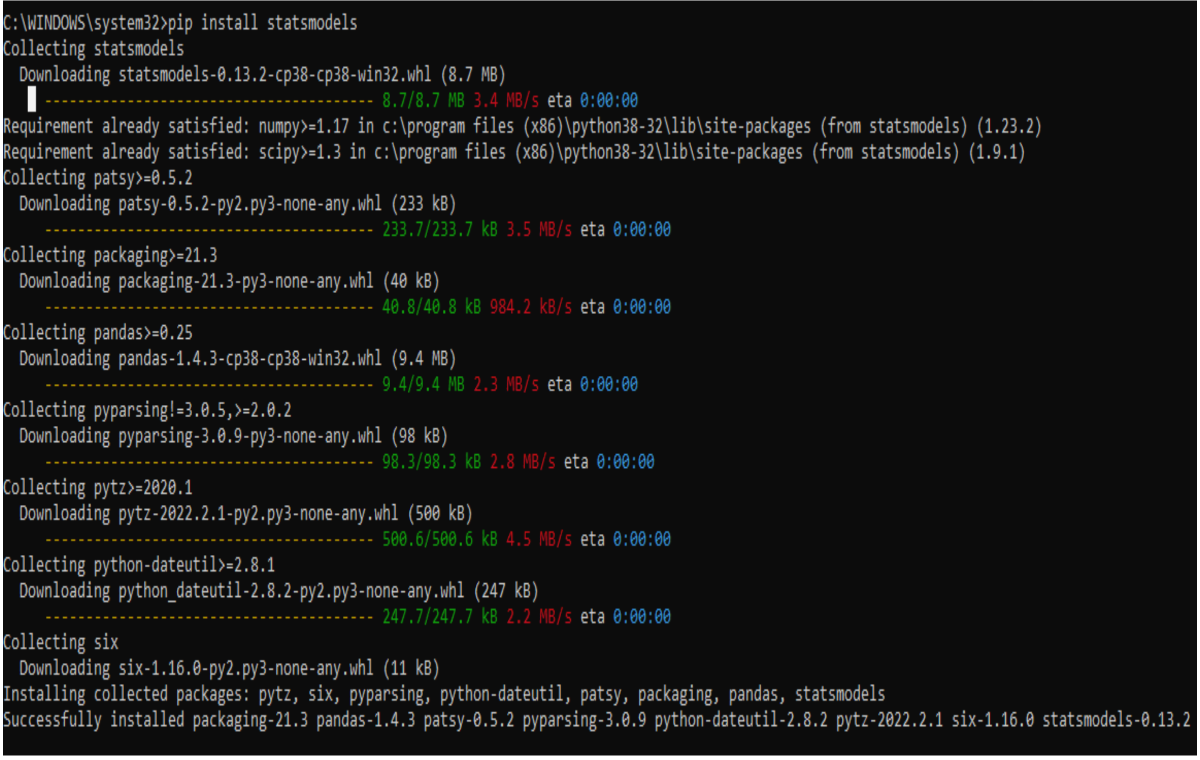
AndroidStudioProjects	2 years ago
Contacts	a year ago
Desktop	3 days ago
Documents	4 months ago
Downloads	4 hours ago
Drivers	2 years ago
eclipse-workspace	2 years ago
eclipse-workspace-NEW	2 years ago

Now select **New** -> **PythonX** and enter the below lines and select **Run**. On Jupyter, each cell is a statement, so you can run each cell independently when there are no dependencies on previous cells.

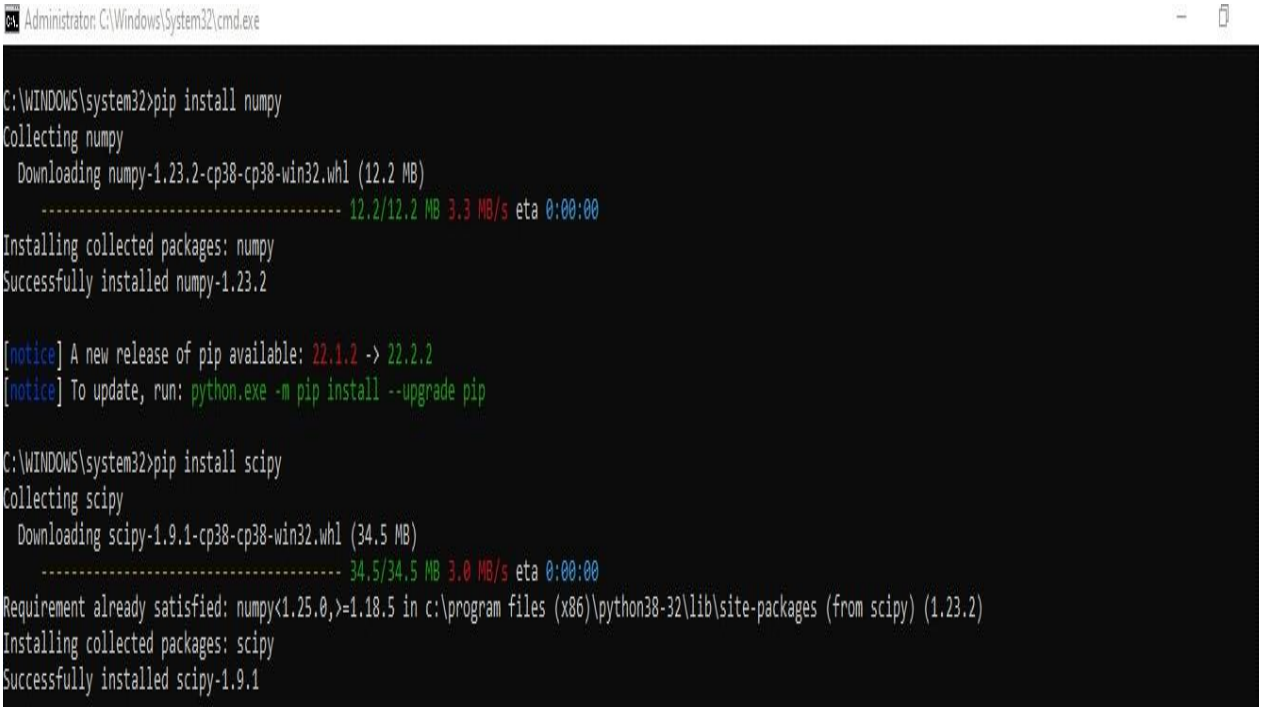


```
In [1]: print("Hello World")

        Hello World

In [2]: 2+3

Out[2]: 5

In [3]: str="My First Program"
        print(str)

        My First Program

In [ ]:
```

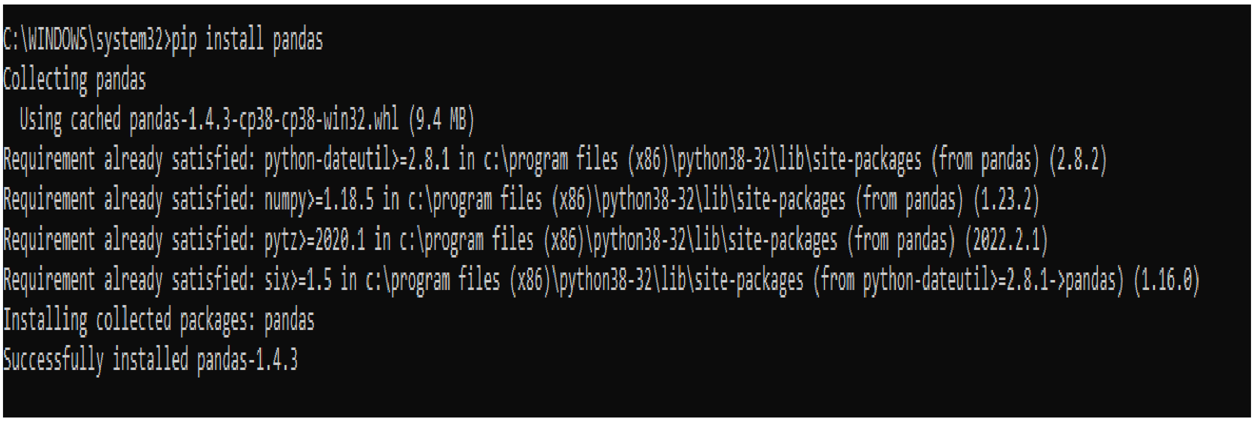
This completes installing Anaconda and running Jupyter Notebook.



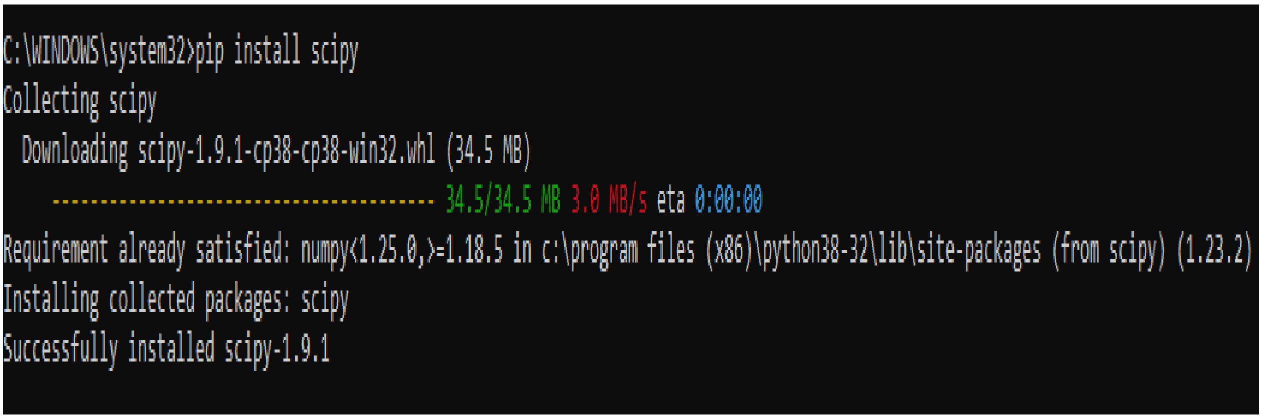
```
C:\WINDOWS\system32>pip install statsmodels
Collecting statsmodels
  Downloading statsmodels-0.13.2-cp38-cp38-win32.whl (8.7 MB)
     ---------------------------------------- 8.7/8.7 MB 3.4 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.17 in c:\program files (x86)\python38-32\lib\site-packages (from statsmodels) (1.23.2)
Requirement already satisfied: scipy>=1.3 in c:\program files (x86)\python38-32\lib\site-packages (from statsmodels) (1.9.1)
Collecting patsy>=0.5.2
  Downloading patsy-0.5.2-py2.py3-none-any.whl (233 kB)
     ---------------------------------------- 233.7/233.7 kB 3.5 MB/s eta 0:00:00
Collecting packaging>=21.3
  Downloading packaging-21.3-py3-none-any.whl (40 kB)
     ---------------------------------------- 40.8/40.8 kB 984.2 kB/s eta 0:00:00
Collecting pandas>=0.25
  Downloading pandas-1.4.3-cp38-cp38-win32.whl (9.4 MB)
     ---------------------------------------- 9.4/9.4 MB 2.3 MB/s eta 0:00:00
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.9-py3-none-any.whl (98 kB)
     ---------------------------------------- 98.3/98.3 kB 2.8 MB/s eta 0:00:00
Collecting pytz>=2020.1
  Downloading pytz-2022.2.1-py2.py3-none-any.whl (500 kB)
     ---------------------------------------- 500.6/500.6 kB 4.5 MB/s eta 0:00:00
Collecting python-dateutil>=2.8.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ---------------------------------------- 247.7/247.7 kB 2.2 MB/s eta 0:00:00
Collecting six
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, six, pyparsing, python-dateutil, patsy, packaging, pandas, statsmodels
Successfully installed packaging-21.3 pandas-1.4.3 patsy-0.5.2 pyparsing-3.0.9 python-dateutil-2.8.2 pytz-2022.2.1 six-1.16.0 statsmodels-0.13.2
```



```
C:\WINDOWS\system32>pip install numpy
Collecting numpy
  Downloading numpy-1.23.2-cp38-cp38-win32.whl (12.2 MB)
     ---------------------------------------- 12.2/12.2 MB 3.3 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-1.23.2

[notice] A new release of pip available: 22.1.2 -> 22.2.2
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\WINDOWS\system32>pip install scipy
Collecting scipy
  Downloading scipy-1.9.1-cp38-cp38-win32.whl (34.5 MB)
     ---------------------------------------- 34.5/34.5 MB 3.0 MB/s eta 0:00:00
Requirement already satisfied: numpy<1.25.0,>=1.18.5 in c:\program files (x86)\python38-32\lib\site-packages (from scipy) (1.23.2)
Installing collected packages: scipy
Successfully installed scipy-1.9.1
```



```
C:\WINDOWS\system32>pip install pandas
Collecting pandas
  Using cached pandas-1.4.3-cp38-cp38-win32.whl (9.4 MB)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\program files (x86)\python38-32\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: numpy>=1.18.5 in c:\program files (x86)\python38-32\lib\site-packages (from pandas) (1.23.2)
Requirement already satisfied: pytz>=2020.1 in c:\program files (x86)\python38-32\lib\site-packages (from pandas) (2022.2.1)
Requirement already satisfied: six>=1.5 in c:\program files (x86)\python38-32\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Installing collected packages: pandas
Successfully installed pandas-1.4.3
```



```
C:\WINDOWS\system32>pip install scipy
Collecting scipy
  Downloading scipy-1.9.1-cp38-cp38-win32.whl (34.5 MB)
     ---------------------------------------- 34.5/34.5 MB 3.0 MB/s eta 0:00:00
Requirement already satisfied: numpy<1.25.0,>=1.18.5 in c:\program files (x86)\python38-32\lib\site-packages (from scipy) (1.23.2)
Installing collected packages: scipy
Successfully installed scipy-1.9.1
```

RESULT:

Thus Jupyter Notebook environment has been successfully installed with all the necessary packages using Anaconda distribution.

Ex.No. 2	Working with Numpy arrays
Date :	

Aim

To implement array object using Numpy module in Python programming

Algorithm

Step 1: Start the program

Step 2: Import the required packages

Step 3: Read the elements through list/tuple/dictionary

Step 4: Convert List/tuple/dictionary into array using built-in methodsStep

Step 5: Check the number of dimensions in an array

Step 6: Compute the shape of an array or if it’s required reshape an array

Step 7: Do the required operations like slicing, iterating, searching, concatenatingand splitting an array element.

Step 8: Stop the program

Program

```
import numpy as np

arr = np.array([[ 1, 2, 3], [ 4, 2, 5]])
print("Array is of type: ", type(arr))
print("No. of dimensions: ", arr.ndim)
print("Shape of array: ", arr.shape)
print("Size of array: ", arr.size)
print("Array stores elements of type: ", arr.dtype)
print("\n-------------------------------------------------------------------\n")

#Program to Perform Array Slicing

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

```
print(a)
print("After slicing")
print(a[1:])
print("\n------------------------------------------------------------------\n")

#Program to Perform Array Slicing

a = np.array([[1,2,3],[3,4,5],[4,5,6]])
print('Our array is:' )
print(a)
print('The items in the second column are:'  )
print(a[...,1])
print('\n'  )
print ('The items in the second row are:' )
print(a[1,...])
print('\n'  )
print('The items column 1 onwards are:' )
print(a[...,1:])
```

Output

```
Array is of type:  <class 'numpy.ndarray'>
No. of dimensions:  2
Shape of array:  (2, 3)
Size of array:  6
Array stores elements of type:  int64

------------------------------------------------------------------

[[1 2 3]
 [3 4 5]
 [4 5 6]]
After slicing
[[3 4 5]
 [4 5 6]]

-------------------------------------------------------------------

Our array is:
[[1 2 3]
 [3 4 5]
 [4 5 6]]
The items in the second column are:
[2 4 5]


The items in the second row are:
[3 4 5]


The items column 1 onwards are:
[[2 3]
 [4 5]
 [5 6]]
```

RESULT:

Thus Array object has been explored using Numpy module in Python programming successfully.

Ex.No. 3	**Working with Pandas data frames**
Date :	

Aim:

To work with DataFrame object using Pandas module in Python Programming

Algorithm:

Step 1: Start the program

Step 2: Import the required packages

Step 3: Create a DataFrame using built in method.

Step 4: Load data into a DataFrame object otherwise Load Files(excel/csv) into a DataFrame

Step 5: Display the rows and describe the data set using built in method.Step 6: Display the last 5 rows of the DataFrame.

Step 7: Check the number of maximum returned rowsStep 8: Stop the program

Program

```
import numpy as np

import pandas as pd

data = np.array([['','Col1','Col2'], ['Row1',1,2], ['Row2',3,4]])

print(pd.DataFrame(data=data[1:,1:], index = data[1:,0], columns=data[0,1:]))

my_2darray = np.array([[1, 2, 3], [4, 5, 6]])

print(pd.DataFrame(my_2darray))

my_dict = {1: ['1', '3'], 2: ['1', '2'], 3: ['2', '4']}

print(pd.DataFrame(my_dict))

my_df = pd.DataFrame(data=[4,5,6,7], index=range(0,4), columns=['A'])

print(pd.DataFrame(my_df))

my_series = pd.Series({"United Kingdom":"London", "India":"New Delhi", "United
States":"Washington", "Belgium":"Brussels"})

print(pd.DataFrame(my_series))

df = pd.DataFrame(np.array([[1, 2, 3], [4, 5, 6]]))

print(df.shape)

print(len(df.index))
```

Output

```
     Col1 Col2
Row1    1    2
Row2    3    4
   0  1  2
   0     1  2  3
   1     4  5  6
   1  2  3
   0     1  1  2
   1     3  2  4
   A
   0     4
   1     5
   2     6
   3     7
                     0
United Kingdom      London
India           New Delhi
United States   Washington
Belgium           Brussels
(2, 3)
2
```

RESULT:

Thus Data Frame object using Pandas module in Python Programming has been successfully explored

Ex.No. 4	**Develop python program for Basic plots using Matplotlib**
Date :	

Aim:

To perform descriptive analytics on Iris dataset using Python programming

Algorithm

Step 1: Start the program

Step 2: Import the required packages

Step 3: Load Files(excel/csv/ text) into a DataFrame from Iris data set Step

4: Display the rows and describe the data set using built in methodsStep 5:

Compare Petal Length and Petal Width

Step 6: Visualize the data set using histogram with distplot, heatmapsbox

plots methods

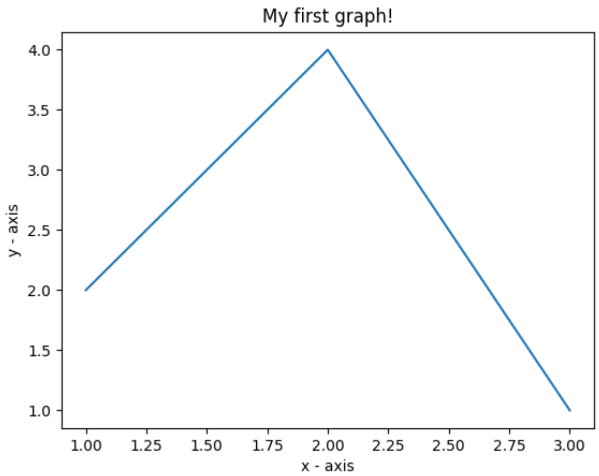
Step 7: Check Missing Values, Duplicates and remove outliers

Step 8: Stop the program

Program A

```
import matplotlib.pyplot as plt

x = [1,2,3]
y = [2,4,1]
plt.plot(x, y)
plt.xlabel('x - axis')
plt.ylabel('y - axis')
plt.title('My first graph!')
plt.show()
```

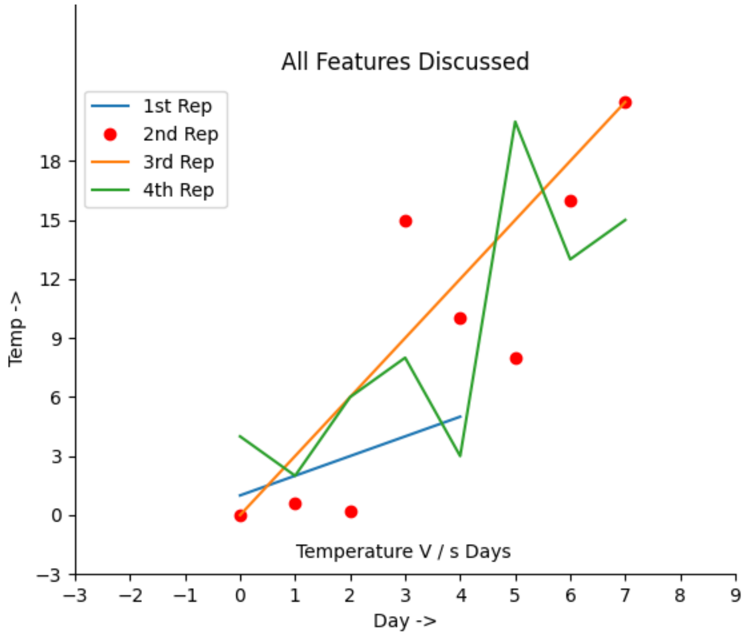
Output:



Program B

```
import matplotlib.pyplot as plt

a = [1, 2, 3, 4, 5]
b = [0, 0.6, 0.2, 15, 10, 8, 16, 21]
plt.plot(a)

plt.plot(b, "or")

plt.plot(list(range(0, 22, 3)))

plt.xlabel('Day ->')

plt.ylabel('Temp ->')

c = [4, 2, 6, 8, 3, 20, 13, 15]
plt.plot(c, label = '4th Rep')

ax = plt.gca()
ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)
ax.spines['left'].set_bounds(-3, 40)
plt.xticks(list(range(-3, 10)))
plt.yticks(list(range(-3, 20, 3)))
ax.legend(['1st Rep', '2nd Rep', '3rd Rep', '4th Rep'])
plt.annotate('Temperature V / s Days', xy = (1.01, -2.15))
plt.title('All Features Discussed')
plt.show()
```

Output:

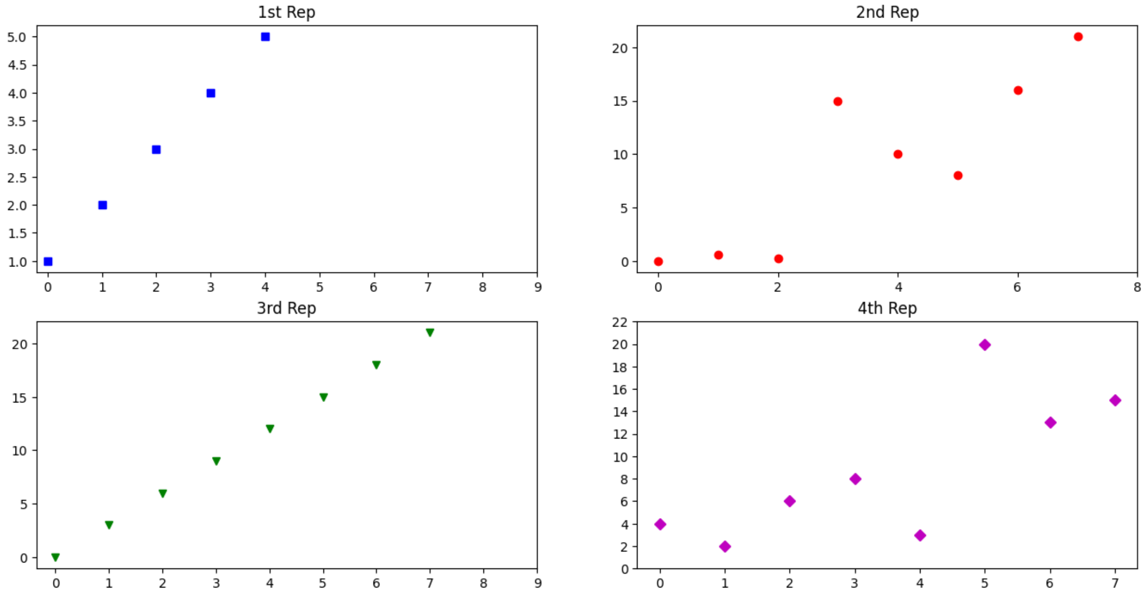


df.describe()

Program C

```
import matplotlib.pyplot as plt

a = [1, 2, 3, 4, 5]
b = [0, 0.6, 0.2, 15, 10, 8, 16, 21]
c = [4, 2, 6, 8, 3, 20, 13, 15]
fig = plt.figure(figsize =(10, 10))
sub1 = plt.subplot(2, 2, 1)
sub2 = plt.subplot(2, 2, 2)
sub3 = plt.subplot(2, 2, 3)
sub4 = plt.subplot(2, 2, 4)

sub1.plot(a, 'sb')

sub1.set_xticks(list(range(0, 10, 1)))
sub1.set_title('1st Rep')

sub2.plot(b, 'or')

sub2.set_xticks(list(range(0, 10, 2)))
sub2.set_title('2nd Rep')

sub3.plot(list(range(0, 22, 3)), 'vg')
sub3.set_xticks(list(range(0, 10, 1)))
sub3.set_title('3rd Rep')

sub4.plot(c, 'Dm')

sub4.set_yticks(list(range(0, 24, 2)))
sub4.set_title('4th Rep')

plt.show()
```

Output


1st Rep
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0 1 2 3 4 5 6 7 8 9
2nd Rep
20
15
10
5
0
0 2 4 6 8
3rd Rep
20
15
10
5
0
0 1 2 3 4 5 6 7 8 9
4th Rep
22
20
18
16
14
12
10
8
6
4
2
0
0 1 2 3 4 5 6 7

RESULT:

Thus Iris dataset has been explored and descriptively analysed using Python programming.

Ex.No. 5	**Develop python program for Frequency distributions**
Date :	

Aim:

To perform various exploratory data analysis on Pima Indians Diabetes datasetusing Python Programming

a. Univariate analysis: Frequency, Mean, Median, Mode, Variance,

Standard Deviation, Skewness and Kurtosis.

b. Bivariate analysis: Linear and logistic regression modeling

c. Multiple Regression analysis

d. Also compare the results of the above analysis for the two data sets.

<u>Algorithm</u>

Step 1: Start the program

Step 2: Import the required packages

Step 3: Load Files (excel/csv/ text) into a Data Frame from UCI and Pima Indians Diabetes data set

Step 4: Display the rows and describe the data set using built in methods

Step 5: Compute Frequency, Mean, Median, Mode, Variance, Standard Deviation, Skewness and Kurtosis

Step 6: Visualize the data set using histogram with distplot, heatmapsbox plots methods

Step 7: Check Missing Values, Duplicates and remove outliers using built in methodStep

8: Stop the program

Program

```
from nltk.tokenize import word_tokenize
from nltk.corpus import gutenberg

sample = gutenberg.raw("blake-poems.txt")

token = word_tokenize(sample)
wlist = []

for i in range(50):
    wlist.append(token[i])

wordfreq = [wlist.count(w) for w in wlist]
print("Pairs\n" + str(list(zip(token, wordfreq))))
```

Output

```
Pairs
[('[', 1), ('Poems', 1), ('by', 1), ('William', 1), ('Blake', 1), ('1789', 1), (']', 1), ('SONGS', 2), ('OF', 3),
('INNOCENCE', 2), ('AND', 1), ('OF', 3), ('EXPERIENCE', 1), ('and', 1), ('THE', 1), ('BOOK', 1), ('of', 2)
,('THEL', 1), ('SONGS', 2), ('OF', 3), ('INNOCENCE', 2), ('INTRODUCTION', 1), ('Piping', 2), ('down', 1),
('the', 1), ('valleys', 1), ('wild', 1), (',', 3), ('Piping', 2), ('songs', 1), ('of', 2), ('pleasant', 1), ('glee', 1), (',', 3),
('On', 1), ('a', 2), ('cloud', 1), ('I', 1), ('saw', 1), ('a', 2), ('child', 1), (',', 3), ('And', 1), ('he', 1), ('laughing', 1),
```

RESULT:

Thus various exploratory data analysis has been performed on Pima IndiansDiabetes dataset using Python Programming successfully.

Ex.No. 6	**Develop python program for Variability**
Date :	

a. Density and contour plots

b. Correlation and scatter plots

c. Histograms

d. Three dimensional plotting

Aim:

To apply various plotting functions on UCI data set using Python Programming

Algorithm

Step 1: Start the program

Step 2: Import the required packages

Step 3: Load Files (excel/csv/ text) into a Data Frame from UCI data set

Step 4: Describe the data set using built in method

Step 5: Compute Frequency, Mean, Median, Mode, Variance, Standard Deviation,

Step 6: Visualize the data set using Explore various plotting functions on UCI datasets for the following

a. Normal curves

b. Density and contour plots

c. Correlation and scatter plots

d. Histograms

e. Three-dimensional plotting

Step 7: Analyze the sample data and do the required operations

Step 8: Stop the program

Program

```
from statistics import variance

from fractions import Fraction as fr
```

```
sample1 = (1, 2, 5, 4, 8, 9, 12)

sample2 = (-2, -4, -3, -1, -5, -6)

sample3 = (-9, -1, -0, 2, 1, 3, 4, 19)

sample4 = (fr(1, 2), fr(2, 3), fr(3, 4), fr(5, 6), fr(7, 8))

sample5 = (1.23, 1.45, 2.1, 2.2, 1.9)


print("Variance of Sample1 is % s " %(variance(sample1)))
print("Variance of Sample2 is % s " %(variance(sample2)))
print("Variance of Sample3 is % s " %(variance(sample3)))
print("Variance of Sample4 is % s " %(variance(sample4)))
print("Variance of Sample5 is % s " %(variance(sample5)))
```

Output

```
Variance of Sample1 is 15.80952380952381
Variance of Sample2 is 3.5
Variance of Sample3 is 61.125
Variance of Sample4 is 1/45
Variance of Sample5 is 0.17613000000000006
```

RESULT:

Thus apply various plotting functions on UCI data set using Python Programming

Ex.No. 7	Develop python program for Averages
Date :	

Aim:

To visualize Geographic Data using BaseMap module in Python Programming

Algorithm:

Step 1: Start the program

Step 2: Import the required packages

Step 3: Visualize Geographic Data with Basemap

Step 4: Display the Base map using built in method like basemap along with latitudeand longitude parameters

Step 5: Display the Coastal lines meters and Country boundaries using built inmethods

Step 6: Fill the Coastal lines meters and Country boundaries with suitable colours

Step 7: Create a global map with a Cylindrical Equidistant Projection, Orthographic Projection, Robinson Projection

Step 8: Stop the program

Program

```
import numpy as np
import pandas as pd



data = {
    'salary_p_year': [50000, 60000, 55000],
    'employees_number': [100, 120, 110]
}
df = pd.DataFrame(data)

weighted_avg_m3 = round(np.average(df['salary_p_year'], weights=df['employees_number']), 2)
```

```
print("Weighted Average Salary per Year:", weighted_avg_m3)
```

Output

```
Weighted Average Salary per Year: 55303.03
```

RESULT

Thus Geographic Data has been visualized using Base Map module in Python Programming successfully.

Ex.No. 8	**Develop python program for Normal Curves**
Date :	

Aim:

To create a Python program that generates Normal Curves (Gaussian Distributions) and visualizes them, we will use libraries like NumPy for generating data points and Matplotlib for plotting the curves.

Algorithm:

Step 1: Start the program

Step 2: Import the necessary libraries.

Step 3: Define the function to generate the normal curve.

Step 4: Use NumPy to generate data points from a normal distribution.

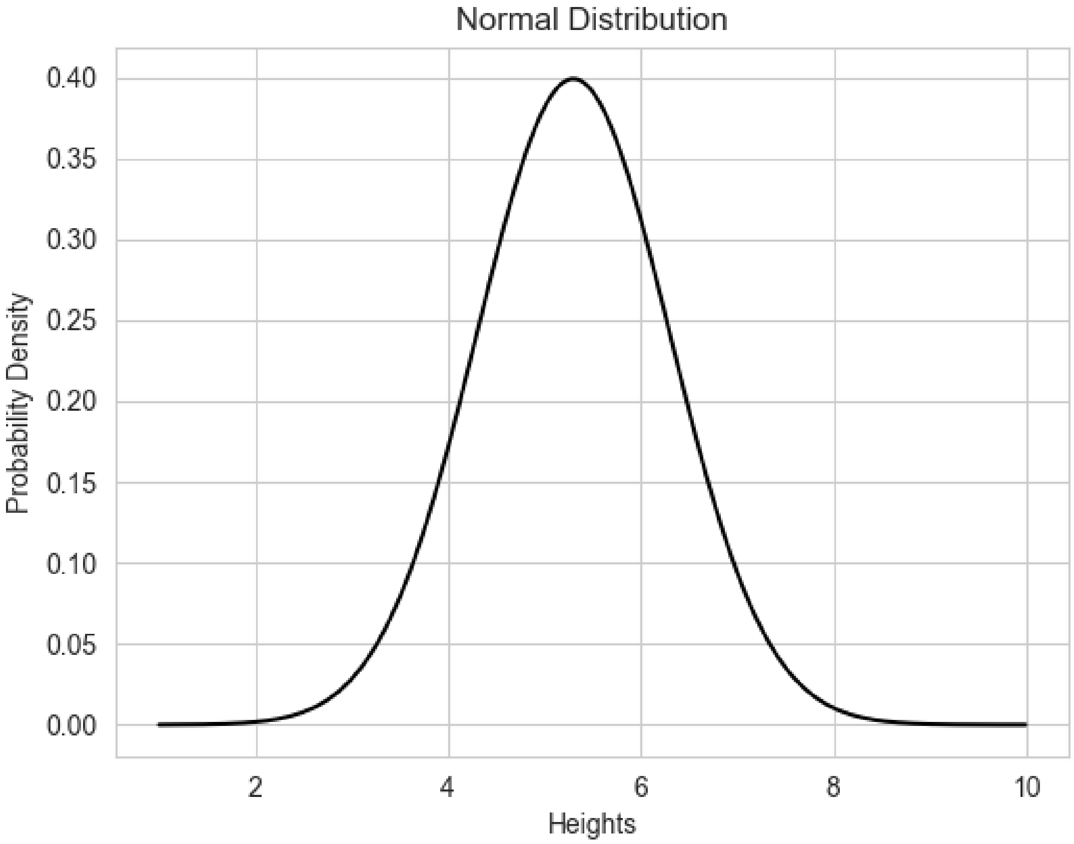
Step 5: Plot the curve using Matplotlib.

Step 6: Stop the program.

Program

```
import seaborn as sb
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm

data = np.arange(1, 10, 0.01)
pdf = norm.pdf(data, loc=5.3, scale=1)

sb.set_style('whitegrid')
plt.plot(data, pdf, color='black')
plt.xlabel('Heights')
plt.ylabel('Probability Density')
plt.title('Normal Distribution')
plt.show()
```

Output



RESULT

This will generate a plot of the normal curve, showing how data is distributed around the mean with the given standard deviation.

Ex.No. 9	**Develop python program for Correlation and scatter plots python program for Normal Curves**
Date :	

Aim:

To create a Python program that calculates **correlation** and generates **scatter plots** (with optional overlays of normal curves on both axes), we'll use the numpy and matplotlib libraries along with scipy.stats to calculate correlation and create normal curves.

Algorithm:

Step 1: Start the program

Step 2: X and Y are generated as normally distributed data. Y is created to be correlated with X by adding some random noise, ensuring the correlation is not perfect.

Step 3: **Pearson Correlation** is calculated using pearsonr() from scipy.stats. It returns the correlation coefficient, which quantifies the relationship between X and Y.

Step 4: We plot a scatter plot of X and Y using plt.scatter().

Step 5: We overlay the **normal distribution curves** for both variables X and Y using the norm.pdf() function from scipy.stats.

Step 6: The plot includes a legend, grid, and the correlation coefficient in the label.

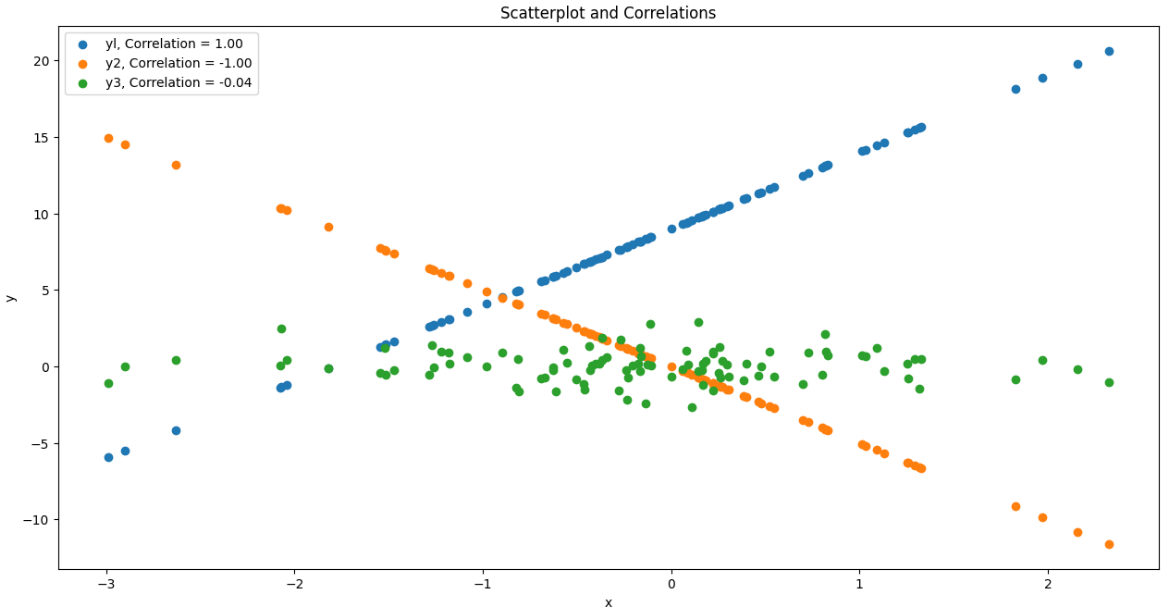
Step 7: Stop the program

Program

```
import numpy as np

import pandas as pd

data = np.array([['','Col1','Col2'], ['Row1',1,2], ['Row2',3,4]])

print(pd.DataFrame(data=data[1:,1:], index = data[1:,0], columns=data[0,1:]))

my_2darray = np.array([[1, 2, 3], [4, 5, 6]])

print(pd.DataFrame(my_2darray))

my_dict = {1: ['1', '3'], 2: ['1', '2'], 3: ['2', '4']}

print(pd.DataFrame(my_dict))

my_df = pd.DataFrame(data=[4,5,6,7], index=range(0,4), columns=['A'])

print(pd.DataFrame(my_df))
```

```
my_series = pd.Series({"United Kingdom":"London", "India":"New Delhi", "United States":"Washington", "Belgium":"Brussels"})

print(pd.DataFrame(my_series))

df = pd.DataFrame(np.array([[1, 2, 3], [4, 5, 6]]))

print(df.shape)

print(len(df.index))
```

Output



RESULT

The program will generate a scatter plot showing the relationship between the two datasets, and it will also plot their normal curves on the same graph. The correlation coefficient and p-value will be printed, indicating the strength of the relationship.

Ex.No. 10 Date :	Develop python program for Correlation coefficient

Aim:

Here's a Python program that computes the correlation coefficient (Pearson correlation) between two datasets and provides a brief explanation

Algorithm:

Step 1: Start the program

Step 2: Ensure that both datasets x and y have the same number of elements (n).

Step 3: Compute the mean of dataset x and y

Step 4: The covariance between x and y is given by

Step 5: The standard deviation of x and y

Step 6: The Pearson correlation coefficient is the covariance divided by the product of the standard deviations of x and y

Step 7: Output the Pearson correlation coefficient r

Step 8: Stop the program

Program

```
import math


def correlationCoefficient(X, Y, n) :
    sum_X = 0
    sum_Y = 0
    sum_XY = 0
    squareSum_X = 0
    squareSum_Y = 0


    i = 0
    while i < n :
        sum_X = sum_X + X[i]
        sum_Y = sum_Y + Y[i]
        sum_XY = sum_XY + X[i] * Y[i]
        squareSum_X = squareSum_X + X[i] * X[i]
        squareSum_Y = squareSum_Y + Y[i] * Y[i]
        i = i + 1
    corr = (float)(n * sum_XY - sum_X * sum_Y)/ (float)(math.sqrt((n * squareSum_X - sum_X * sum_X)* (n * squareSum_Y -  sum_Y * sum_Y)))
    return corr


X = [15, 18, 21, 24, 27]
Y = [25, 25, 27, 31, 32]
n = len(X)
print ('{0:.6f}'.format(correlationCoefficient(X, Y, n)))
```

Output

```
0.953463
```

RESULT

The program will generate the correlation coefficient and p-value for the two datasets

Ex.No. 11	Develop python program for Simple Linear Regression
Date :	

Aim:

Here is a Python program for Simple Linear Regression and an explanation of the algorithm used for calculating it manually.

Algorithm:

Step 1: Start the program

Step 2: Let n be the number of data points in both x and y.

Step 3: Find Mean of x and y

Step 4: Calculate the Slope (β_1)

Step 5: Calculate the Intercept (β_0)

Step 6: Predict the y values

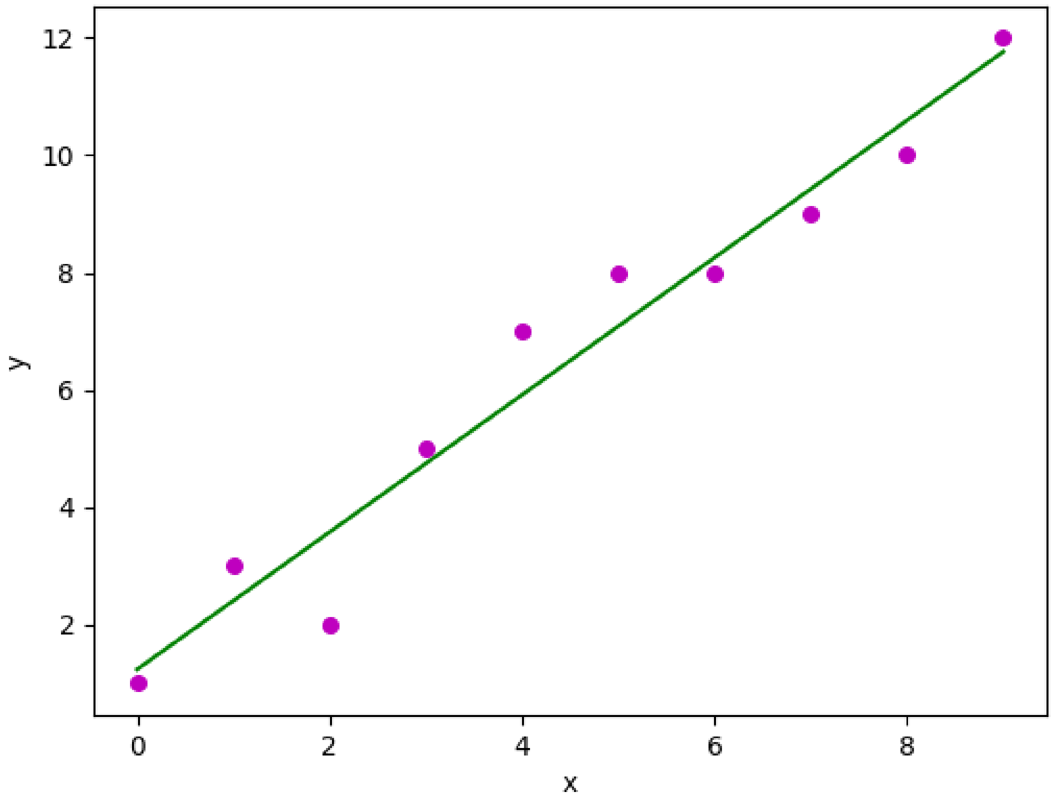
Step 7: Stop the program

Program

```
import numpy as np

import matplotlib.pyplot as plt


def estimate_coef(x, y):

    n = np.size(x)

    m_x = np.mean(x)

    m_y = np.mean(y)

    SS_xy = np.sum(y*x) - n*m_y*m_x

    SS_xx = np.sum(x*x) - n*m_x*m_x

    b_1 = SS_xy / SS_xx

    b_0 = m_y - b_1*m_x

    return (b_0, b_1)
```

```
def plot_regression_line(x, y, b):
    plt.scatter(x, y, color = "m",
    marker = "o", s = 30)
    y_pred = b[0] + b[1]*x
    plt.plot(x, y_pred, color = "g")
    plt.xlabel('x')
    plt.ylabel('y')
    plt.show()

def main():
    x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
    y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])
    b = estimate_coef(x, y)
    print("Estimated coefficients:\nb_0 = {} \\nb_1 = {}".format(b[0], b[1]))
    plot_regression_line(x, y, b)

if __name__ == "__main__":
 main()
```

Output



RESULT

- A plot showing the data points and the regression line.
- The intercept (b0) and slope (b1) values will be printed, indicating the equation of the line.

VIVA VOICE

1. Benefits of Data Preparations
2. What is bias and list its types?
3. Write short notes on Discrete and continuous variable
4. Define standard normal curve with equation and graph.
5. List out the types of non linear relationship.
6. Difference between Linear and multiple regressions.
7. What is .loc( ), .iloc( ), .ix( ) ?
8. Differentiate append() and concat() in pandas.
9. Write Short notes on KDE
10. Define seaborn plots.
11. Identify the steps of data science process
12. What is data cleaning?
13. Differentiate Histogram and bar graph
14. What is Z-Score?
15. Define Causation.
16. Compare Correlation and Regression.
17. Where is NumPy used?
18. “List is mutable”- Justify with example
19. What is density plot?
20. Write the significance of Data Visualization.