योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS
Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE
An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

B. Tech Programme: B. Tech AI-ML (A)

Course Title: Foundation of Data Science Lab

Course Code: AIML - 203

Submitted to:	Submitted by:
Mr. Adeel Hashmi	**Name: Kunsh Sabharwal**
Assistant Professor	**Enrolment No: 01117711623**

योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION
VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS
Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE
An ISO 9001:2015 Certified Institution
SCHOOL OF ENGINEERING & TECHNOLOGY

VISION OF INSTITUTE

To be an educational institute that empowers the field of engineering to build a sustainable future by providing quality education with innovative practices that supports people, planet and profit.

MISSION OF INSTITUTE

To groom the future engineers by providing value-based education and awakening students' curiosity, nurturing creativity and building capabilities to enable them to make significant contributions to the world.

योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS
Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE
An ISO 9001:2015 Certified Institution
SCHOOL OF ENGINEERING & TECHNOLOGY

INDEX

S. No.	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
1.								
2.								
3.								

S. No.	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
4.								
5.								
6.								

S. No.	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
7.								
8.								
9.								

S. No.	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
10.								
11.								
12.								

योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION
VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS
Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE
An ISO 9001:2015 Certified Institution
SCHOOL OF ENGINEERING & TECHNOLOGY

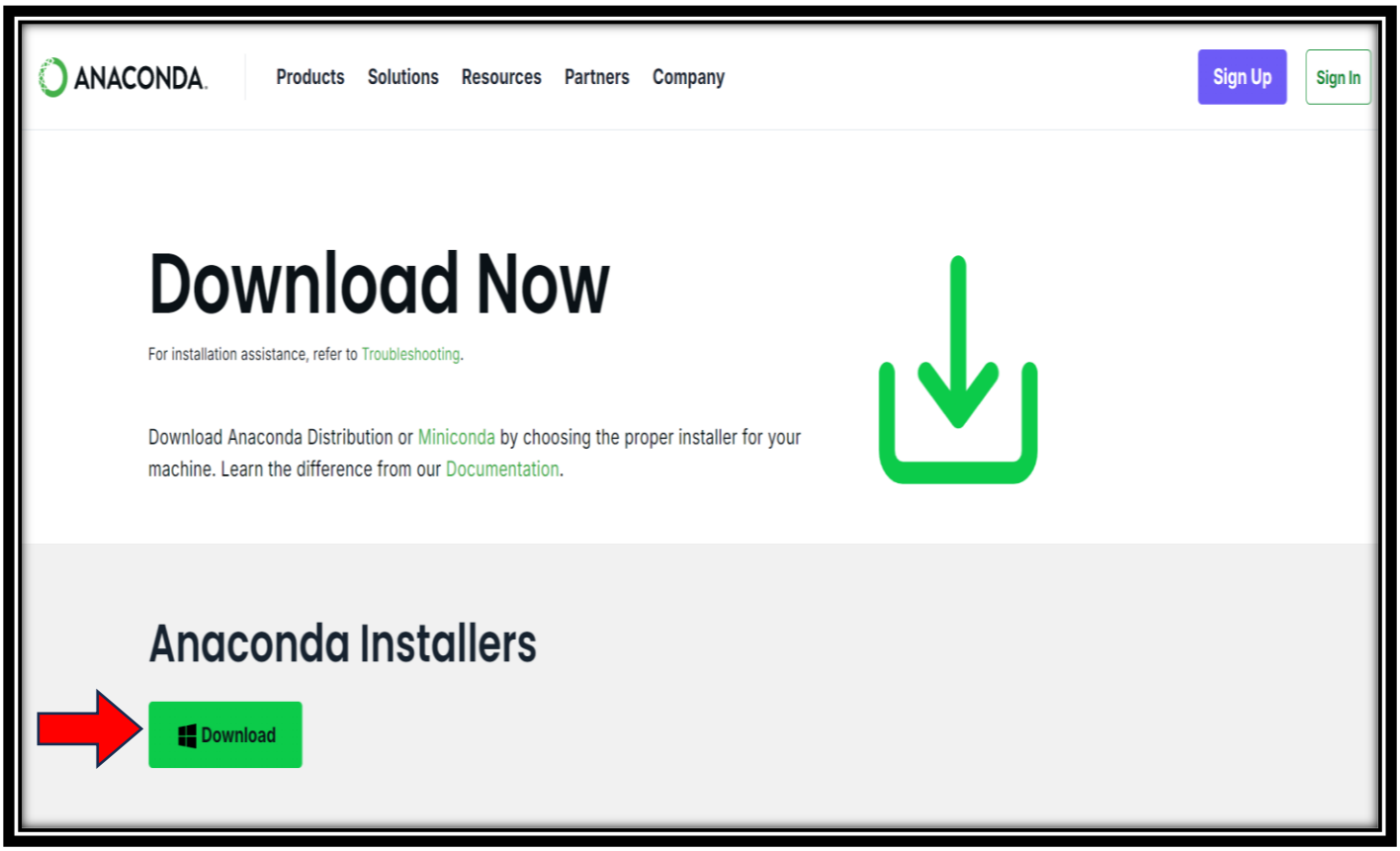
S. No.	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
13. (Mini – Project)								

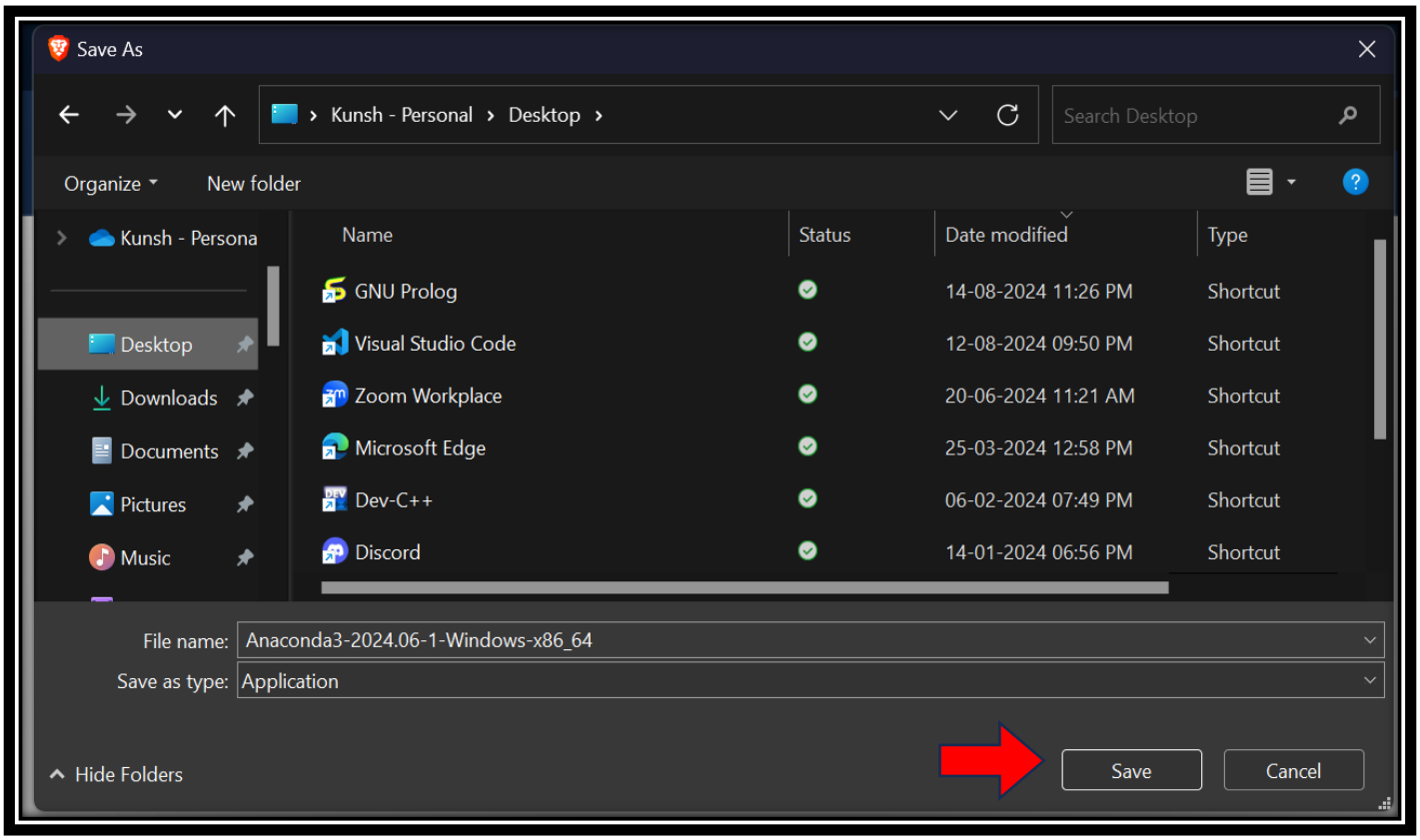
EXPERIMENT 1

Aim: Introduction and installation of Python and Python IDEs for data science (Spyder-Anaconda, Jupyter Notebook etc.) Student must install anyone of the popular IDES, configure them and test.

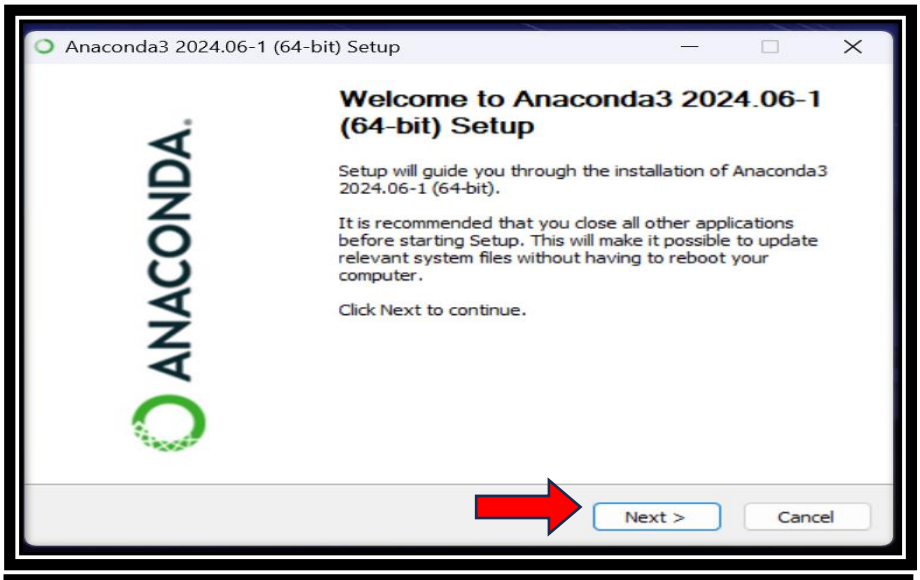
Theory:

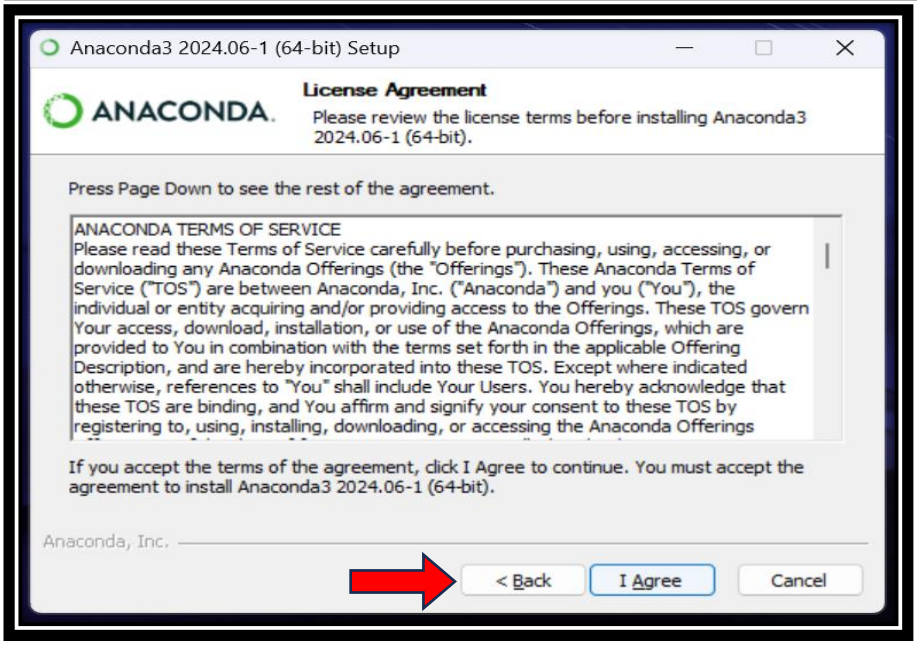
Installation Screenshots:

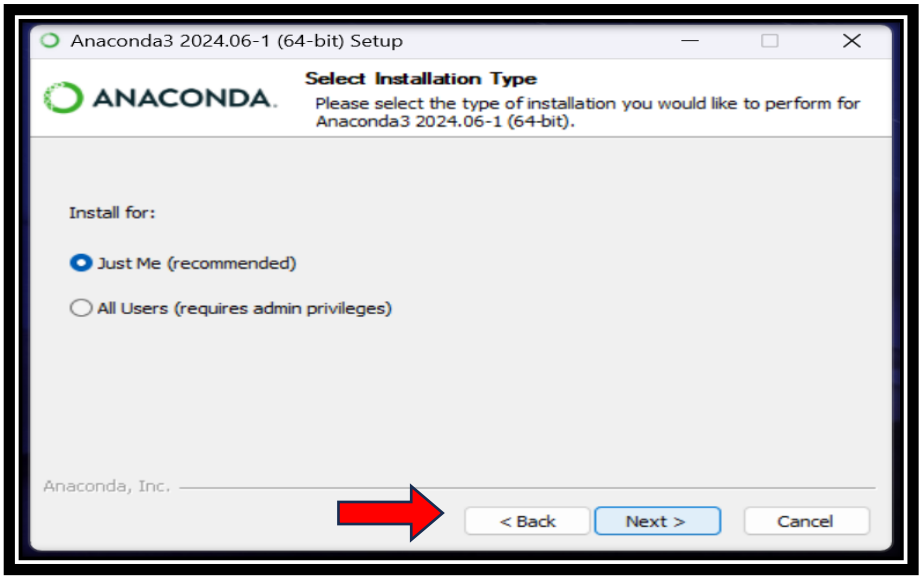


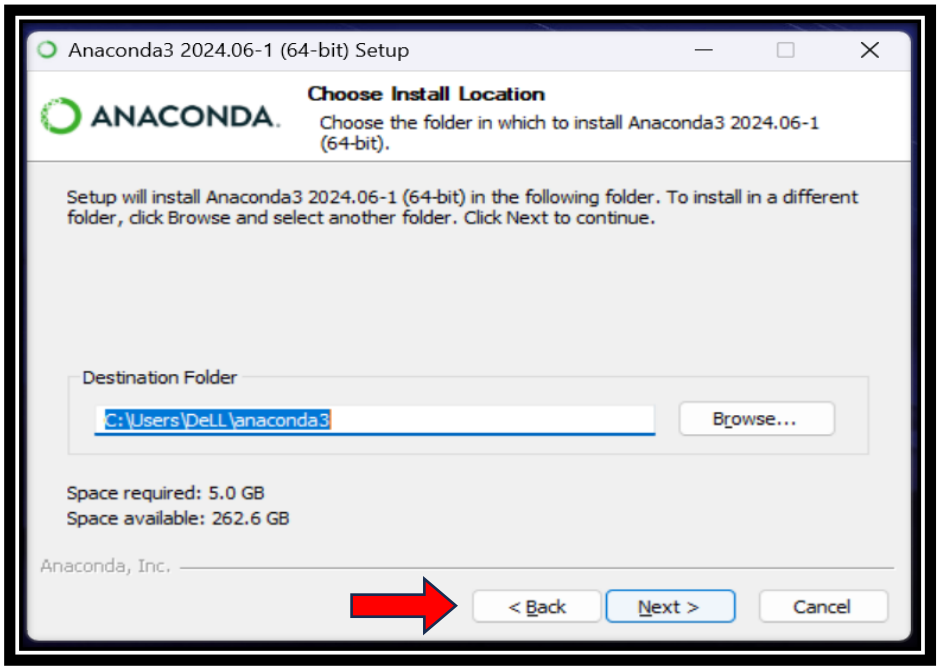


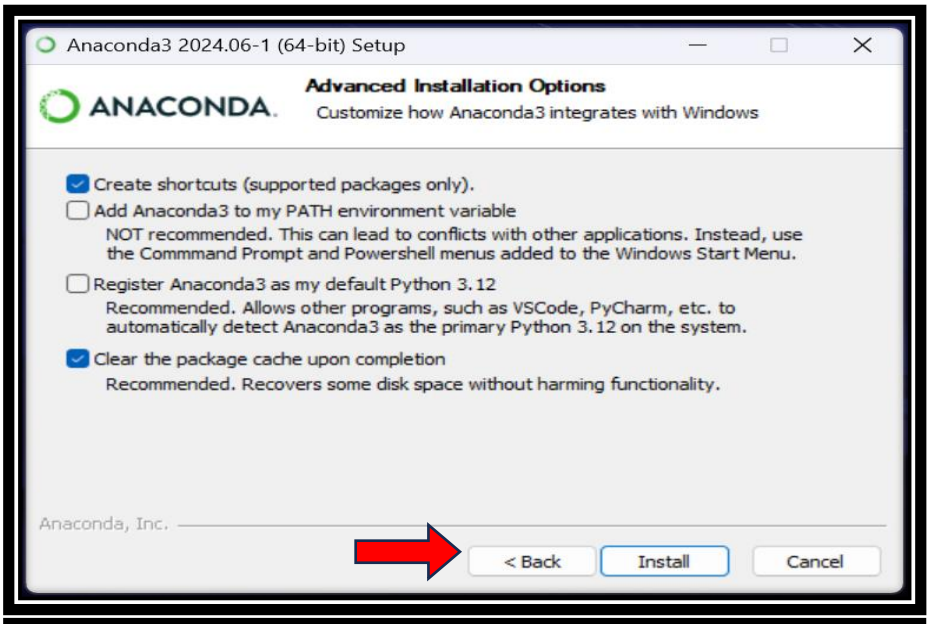
Configuration Screenshots:

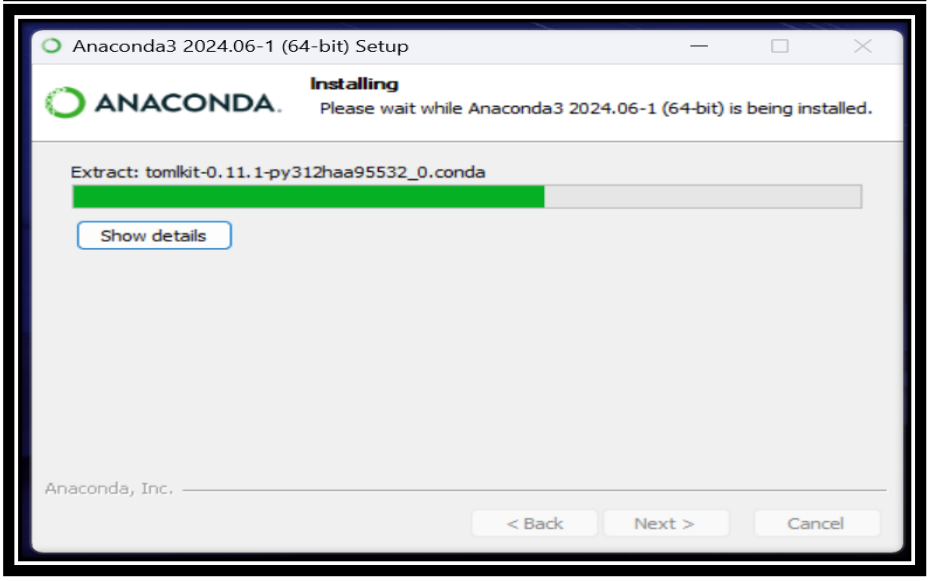


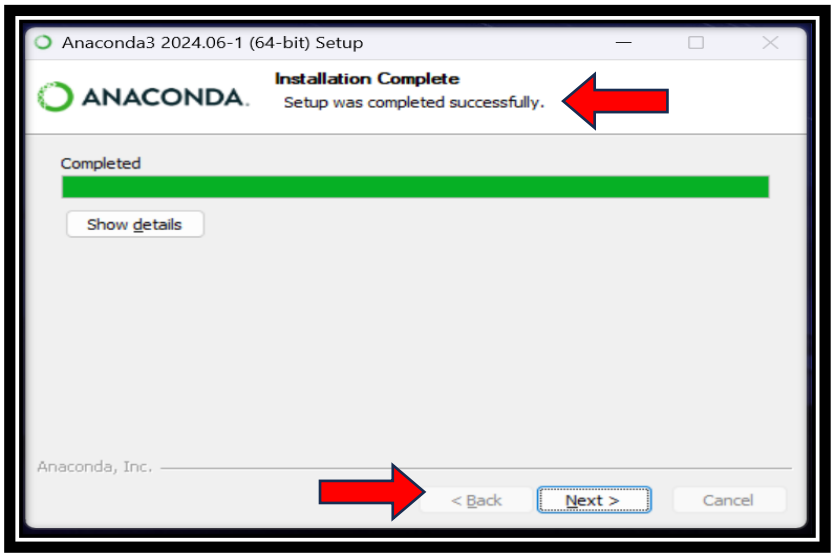


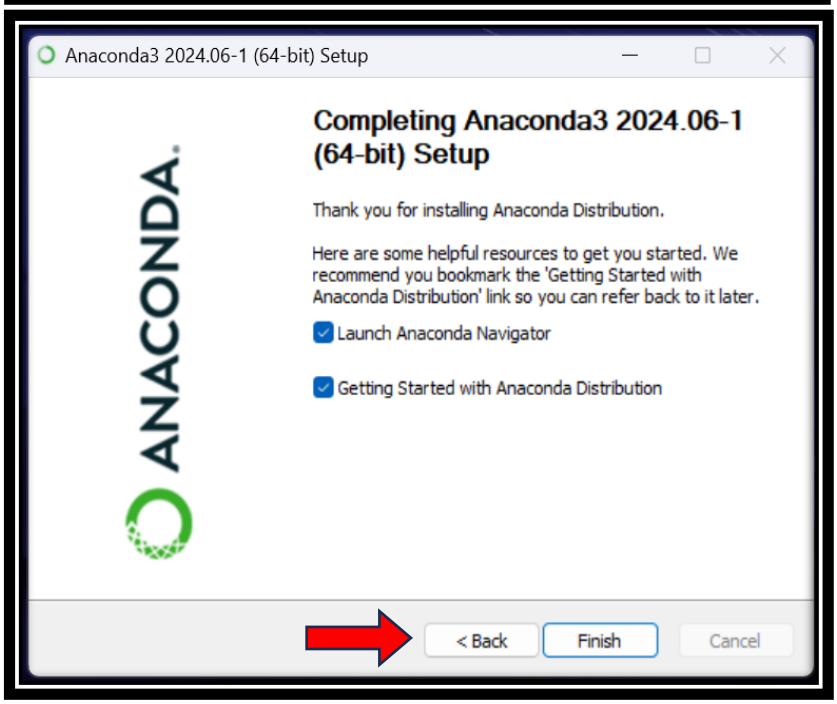


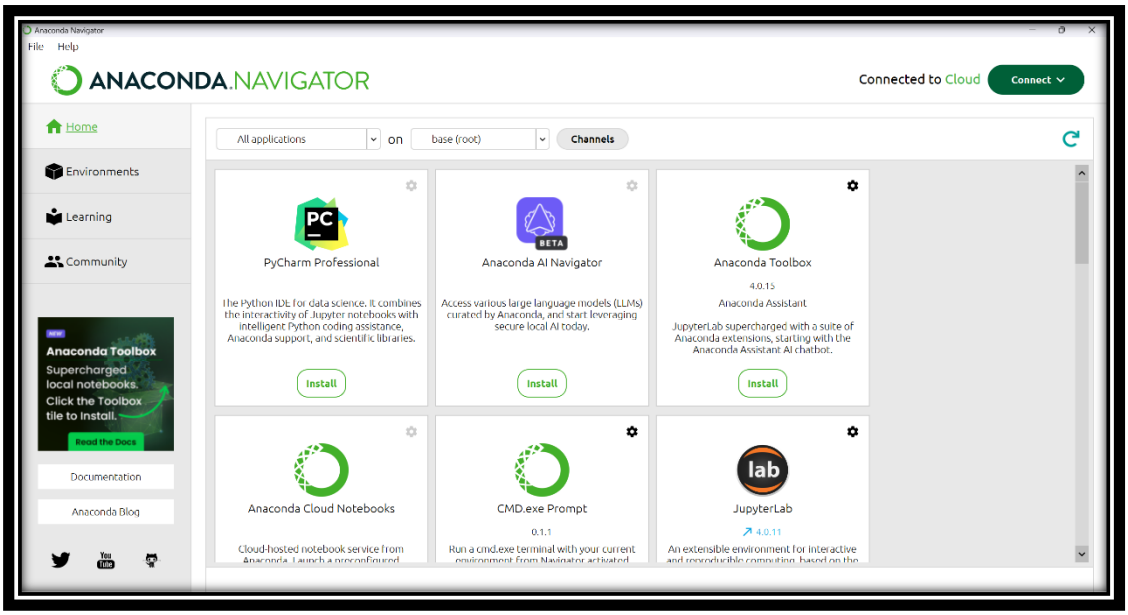

Anaconda3 2024.06-1 (64-bit) Setup
ANACONDA.
Choose Install Location
Choose the folder in which to install Anaconda3 2024.06-1 (64-bit).
Setup will install Anaconda3 2024.06-1 (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:\Users\DeLL\anaconda3
Browse...
Space required: 5.0 GB
Space available: 262.6 GB
Anaconda, Inc.
< Back
Next >
Cancel


Anaconda3 2024.06-1 (64-bit) Setup
ANACONDA.
Advanced Installation Options
Customize how Anaconda3 integrates with Windows
Create shortcuts (supported packages only).
Add Anaconda3 to my PATH environment variable
NOT recommended. This can lead to conflicts with other applications. Instead, use the Commmand Prompt and Powershell menus added to the Windows Start Menu.
Register Anaconda3 as my default Python 3.12
Recommended. Allows other programs, such as VSCode, PyCharm, etc. to automatically detect Anaconda3 as the primary Python 3.12 on the system.
Clear the package cache upon completion
Recommended. Recovers some disk space without harming functionality.
Anaconda, Inc.
< Back
Install
Cancel

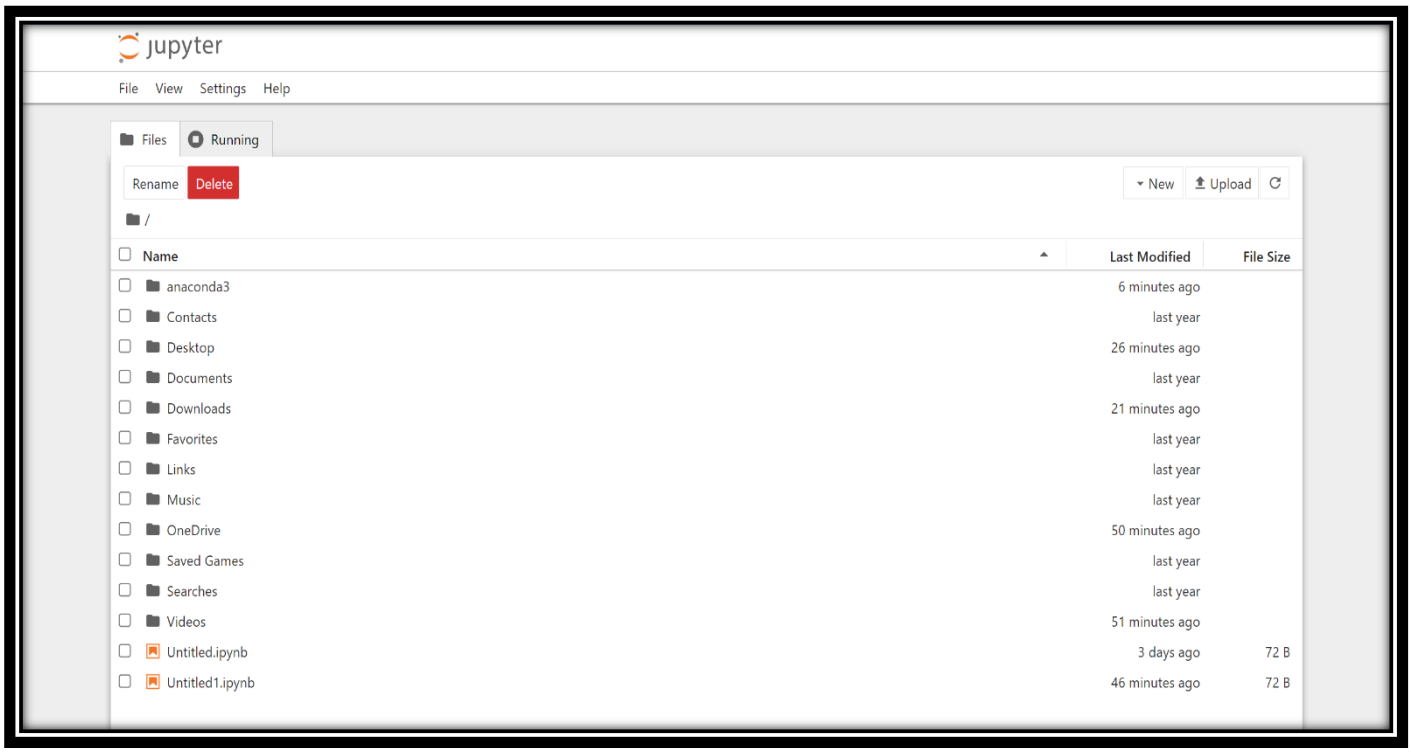

Anaconda3 2024.06-1 (64-bit) Setup
ANACONDA.
Installing
Please wait while Anaconda3 2024.06-1 (64-bit) is being installed.
Extract: tomlkit-0.11.1-py312haa95532_0.conda
Show details
Anaconda, Inc.
< Back
Next >
Cancel


Anaconda3 2024.06-1 (64-bit) Setup
ANACONDA.
Installation Complete
Setup was completed successfully.
Completed
Show details
Anaconda, Inc.
< Back
Next >
Cancel


Anaconda3 2024.06-1 (64-bit) Setup
ANACONDA.
Completing Anaconda3 2024.06-1 (64-bit) Setup
Thank you for installing Anaconda Distribution.
Here are some helpful resources to get you started. We recommend you bookmark the 'Getting Started with Anaconda Distribution' link so you can refer back to it later.
Launch Anaconda Navigator
Getting Started with Anaconda Distribution
< Back
Finish
Cancel


Anaconda Navigator
File
Help
ANACONDA.NAVIGATOR
Connected to Cloud
Connect
Home
Environments
Learning
Community
All applications
on
base (root)
Channels
PyCharm Professional
Anaconda AI Navigator
Anaconda Toolbox
Install
Anaconda Cloud Notebooks
CMD.exe Prompt
JupyterLab
Documentation
Anaconda Blog

Testing the installed and configured IDE (Jupyter Notebook):



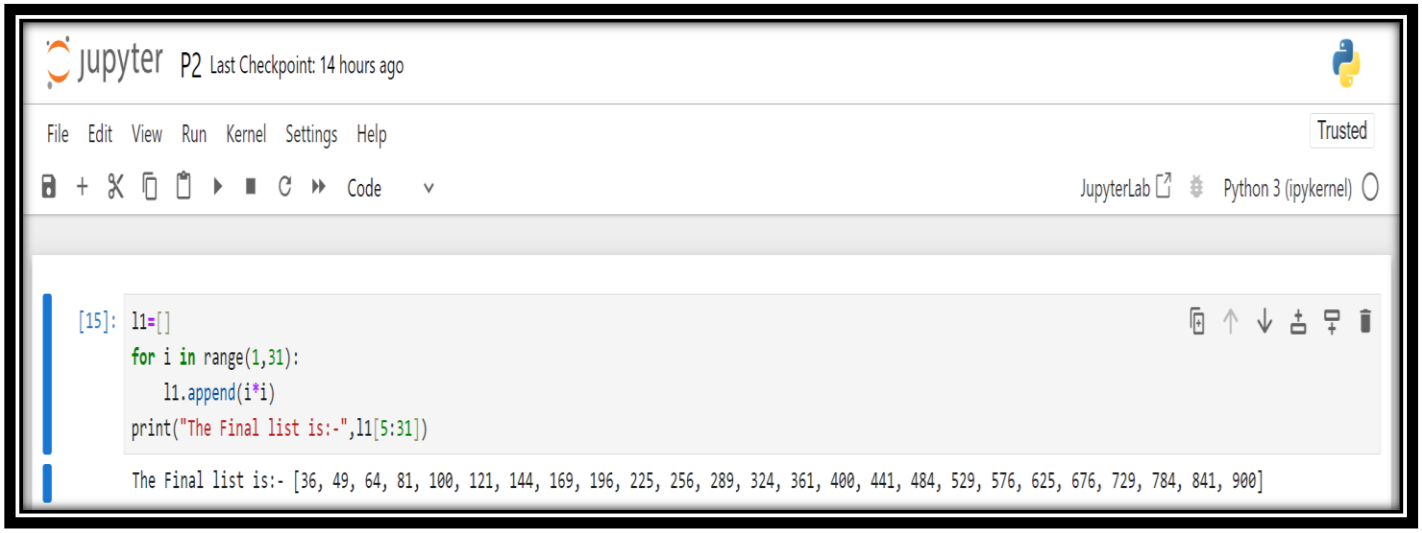
Learning Outcome:

EXPERIMENT 2

Aim: To design a Python program that generates a list of squares of numbers between 1 and 30 and prints the list excluding the first 5 elements.

Theory:

Programming Code and Output:



Learning Outcome:

EXPERIMENT 3(a)

Aim: Design a Python program to understand the working of loops: -

(a) Reverse a given string using both for and while loops.

Theory:

Programming Code and Output:

jupyter P3(a) Last Checkpoint: 3 days ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[22]:
```
x=input("Enter a string to reverse:-")
for i in x:
    x=x[::-1]
print(x)
```

```
Enter a string to reverse:- kunsh
hsnuk
```

[2]:
```
y=input("Enter a string to reverse:-")
rev_str=" "
index=len(y)-1
while index>=0:
    rev_str=rev_str+y[index]
    index=index-1
print("The reversed string is:-",rev_str)
```

```
Enter a string to reverse:- kunsh
The reversed string is:-  hsnuk
```

Learning Outcome:

EXPERIMENT 3(b)

Aim: Design a Python program to understand the working of loops: -

(b) Write a program to find the sum of the digits of a given number.

Theory:

Programming Code and Output:

jupyter P3(b) Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[2]:
```
x=int(input("Enter a number:-"))
sum=0
if(x<0):
    print("Invalid Input.")
else:
    while(x!=0):
        sum=sum+(x%10)
        x=x//10
print(sum)
```

```
Enter a number:- 123
6
```

Learning Outcome:

EXPERIMENT 3(c)

Aim: Design a Python program to understand the working of loops: -

(c) Write a program to find factorial of a number.

Theory:

Programming Code and Output:

Learning Outcome:

EXPERIMENT 3(d)

Aim: Design a Python program to understand the working of loops: -

(d) Write a program to generate the Fibonacci series.

Theory:

Programming Code and Output:

jupyter P3(d) Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[8]:
```
x=int(input("Enter the number of terms for the Fibonacci Series:-"))
first = 0
second = 1
print(first)
print(second)
for x in range(0,x-2):
    third = first + second
    print(third)
    first,second=second,third
```

```
Enter the number of terms for the Fibonacci Series:- 10
0
1
1
2
3
5
8
13
21
34
```

Learning Outcome:

EXPERIMENT 3(e)

Aim: Design a Python program to understand the working of loops: -

(e) Write a program to print the following pattern of an equilateral triangle: -

```
*
**
***
****
*****
```

Theory:

Programming Code and Output:

Learning Outcome:

EXPERIMENT 4

Aim: To design a Python function that determines and returns the maximum of three given numbers.

Theory:

Programming Code and Output:

jupyter P4 Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[22]:
```
x=int(input("Enter 1st Number:-"))
y=int(input("Enter 2nd Number:-"))
z=int(input("Enter 3rd Number:-"))
if(x==y or x==z or y==z):
    print("Two inputs are same. Cannot compare.")
elif(x>y and x>z):
    print("The greatest number is:-",x)
elif(y>x and y>z):
    print("The greatest number is:-",y)
else:
    print("The greatest number is:-",z)
```

```
Enter 1st Number:- 5
Enter 2nd Number:- 10
Enter 3rd Number:- 2
The greatest number is:- 10
```

Learning Outcome:

EXPERIMENT 5

Aim: Write a program in Python to read a text file and write in a text file.

Theory:

Programming Code and Output:

Learning Outcome:

EXPERIMENT 6

Aim: Write a program in Python to implement exception handling.

Theory:

Programming Code and Output:

jupyter Untitled Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[1]:
```
try:
    x=int(input("Enter 1st number:-"))
    y=int(input("Enter 2nd number:-"))
    ans=x/y
    print(ans)
except ZeroDivisionError:
    print("Cannot divide by zero!")
```

```
Enter 1st number:- 5
Enter 2nd number:- 2
2.5
```

jupyter Untitled Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[3]:
```
try:
    x=int(input("Enter 1st number:-"))
    y=int(input("Enter 2nd number:-"))
    ans=x/y
    print(ans)
except ZeroDivisionError:
    print("Cannot divide by zero!")
```

```
Enter 1st number:- 5
Enter 2nd number:- 0
Cannot divide by zero!
```

Learning Outcome:

EXPERIMENT 7

Aim: Design a Python Program for creating a random story generator.

Theory:

Programming Code and Output:

jupyter random story generator Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [12]:
```
import random
starting= ['About 2000 years ago', ' In the year 400 BC', 'Once upon a time']
char= ['there lived a king.',' there was a man named Peter.', ' there lived a man.']
time= ['One day', 'One full-moon night']
plot= ['he was passing by', ' he was going for a walk to']
place= [' the mountains.', ' the garden.']
char2= ['He saw a man,', 'He saw a lady,']
age = ['who seemed to be young,', 'who seemed very old and feeble,']
work = ['searching for something.', 'going for a walk.']
print("The story is :-",random.choice(starting)+random.choice(character)+
      random.choice(time)+random.choice(plot)+
      random.choice(place)+random.choice(char2)+
      random.choice(age)+random.choice(work))
```

```
The story is :- About 2000 years ago there lived a .One full-moon night he was going for a walk to the garden.He saw a lady,who
seemed very old and feeble,searching for something.
```

Learning Outcome:

EXPERIMENT 8

Aim: Create a NumPy array with specific characteristics and perform various operations to analyse and manipulate the data.

(a) Create a 2D NumPy array of shape (4,5) with random integers from between 1 to 50.
(b) Calculate the Sum: Compute the sum of all elements in the array.
(c) Find the Maximum Value: Determine the maximum value in the array.
(d) Calculate the Mean: Compute the mean of the array elements.
(e) Sum of Each Row: Calculate the sum of elements in each row.
(f) Transpose the Array: Transpose the array and display it.
(g) Filter Elements: Create a Boolean mask to find all elements greater than 25.

Theory:

Programming Code and Output:

jupyter p8 Last Checkpoint: 51 seconds ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[2]:
```
import numpy as np
import random as rd
arr=np.array([[rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50)],
              [rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50)],
              [rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50)],
              [rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50),rd.randint(1,50)]])
print(arr)
```

```
[[32 11 31 45 42]
 [10 42  8 34 28]
 [12 41 33 17 49]
 [50  8 14 43 19]]
```

[3]:
```
np.sum(arr)
```

[3]:
```
569
```

[4]:
```
np.amax(arr)
```

[4]:
```
50
```

[5]:
```
np.average(arr)
```

[5]:
```
28.45
```

jupyter p8 Last Checkpoint: 7 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[5]:
```
np.average(arr)
```

[5]:
```
28.45
```

[6]:
```
np.sum(arr,axis=1)
```

[6]:
```
array([161, 122, 152, 134])
```

[7]:
```
arr.transpose()
```

[7]:
```
array([[32, 10, 12, 50],
       [11, 42, 41,  8],
       [31,  8, 33, 14],
       [45, 34, 17, 43],
       [42, 28, 49, 19]])
```

[8]:
```
arr[arr>25]
```

[8]:
```
array([32, 31, 45, 42, 42, 34, 28, 41, 33, 49, 50, 43])
```

Learning Outcome:

EXPERIMENT 9

Aim: Create a Synthetic Dataset(.csv/.xlsx) to work upon and design a Python program to read and print that data.

Theory:

Source Code and Output:

(a) Jupyter Notebook Screenshot:

jupyter P9 Last Checkpoint: 35 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [2]:
```
pip install faker
```

```
Collecting faker
  Downloading Faker-30.8.0-py3-none-any.whl (1.8 MB)
Requirement already satisfied: typing-extensions in c:\users\cc12\anaconda3\lib\site-packages (from faker) (4.1.1)
Requirement already satisfied: python-dateutil>=2.4 in c:\users\cc12\anaconda3\lib\site-packages (from faker) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\cc12\anaconda3\lib\site-packages (from python-dateutil>=2.4->faker) (1.16.
0)
Installing collected packages: faker
Successfully installed faker-30.8.0
Note: you may need to restart the kernel to use updated packages.
```

In [3]:
```
from faker import Faker
fake = Faker()
import pandas as pd
l1=[]
for i in range(5):
    name=fake.name()
    country=fake.country()
    email=fake.email()
    l2=[name,country,email]
    l1.append(l2)
print(l1)
```

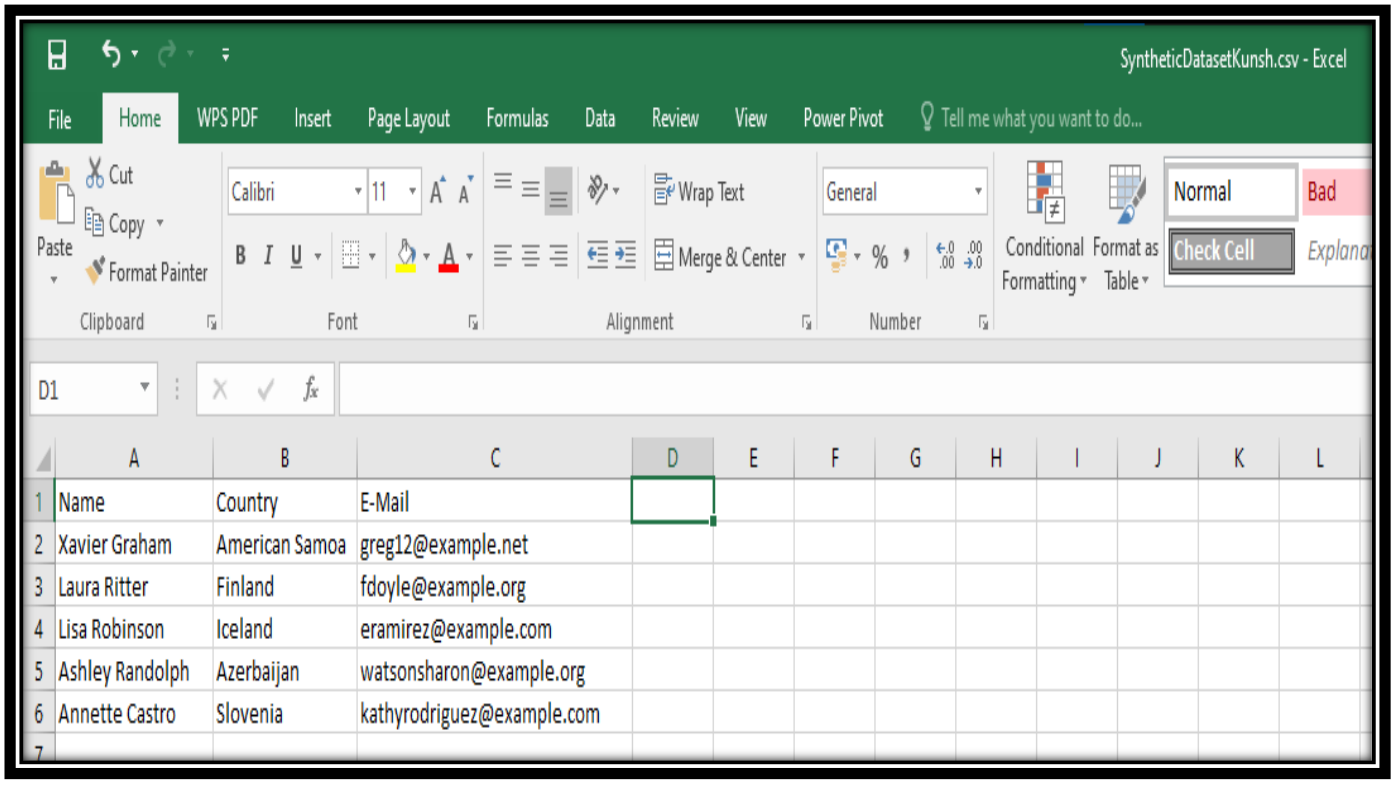
```
[['Xavier Graham', 'American Samoa', 'greg12@example.net'], ['Laura Ritter', 'Finland', 'fdoyle@example.org'], ['Lisa Robinso
n', 'Iceland', 'eramirez@example.com'], ['Ashley Randolph', 'Azerbaijan', 'watsonsharon@example.org'], ['Annette Castro', 'Slov
enia', 'kathyrodriguez@example.com']]
```

In [14]:
```
column=['Name','Country','E-Mail']
df = pd.DataFrame(l1,columns=column)
print(df)
```

```
              Name         Country                     E-Mail
0    Xavier Graham  American Samoa          greg12@example.net
1     Laura Ritter         Finland          fdoyle@example.org
2    Lisa Robinson         Iceland        eramirez@example.com
3  Ashley Randolph      Azerbaijan    watsonsharon@example.org
4   Annette Castro        Slovenia  kathyrodriguez@example.com
```

In [15]:
```
df.to_csv('SyntheticDatasetKunsh.csv',index=False)
```

(b)Excel Screenshot:



Learning Outcome:

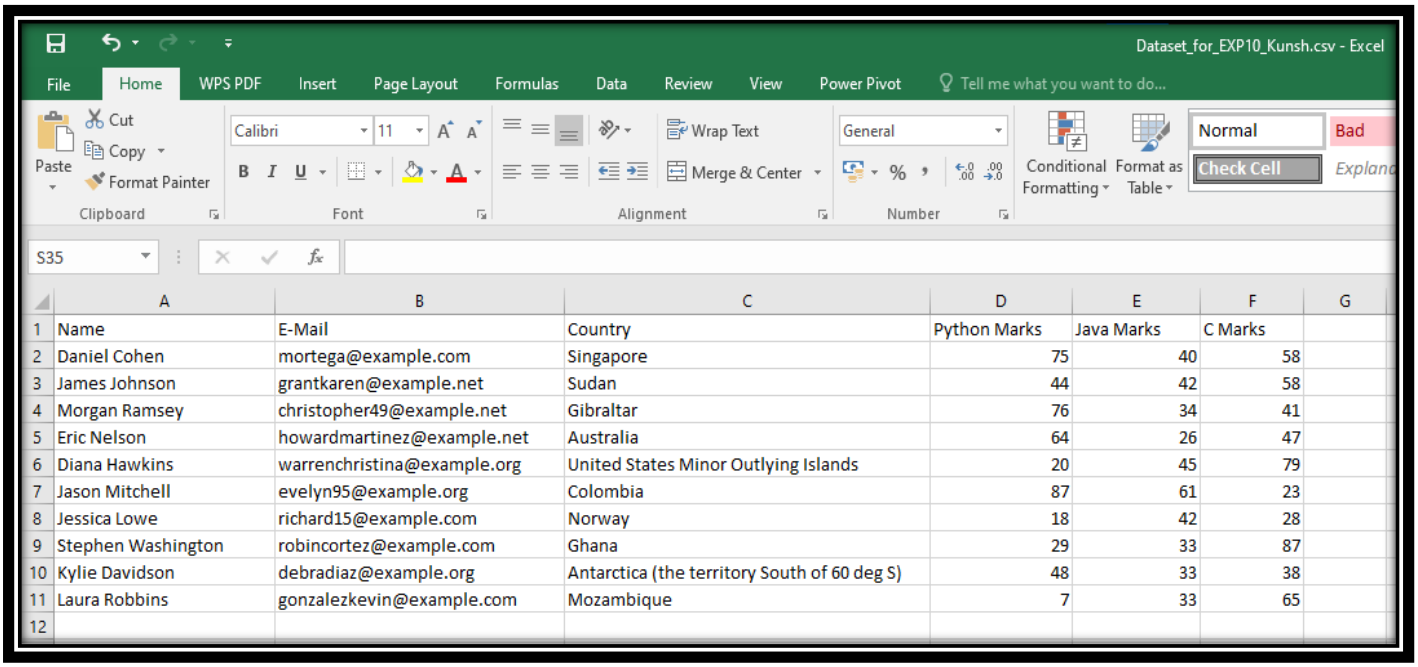
EXPERIMENT 10

Aim: Perform Statistics and Data Visualization in Python.

Theory:

Source Code and Output:

(a) Excel Screenshot:



	A	B	C	D	E	F	G
1	Name	E-Mail	Country	Python Marks	Java Marks	C Marks	
2	Daniel Cohen	mortega@example.com	Singapore	75	40	58	
3	James Johnson	grantkaren@example.net	Sudan	44	42	58	
4	Morgan Ramsey	christopher49@example.net	Gibraltar	76	34	41	
5	Eric Nelson	howardmartinez@example.net	Australia	64	26	47	
6	Diana Hawkins	warrenchristina@example.org	United States Minor Outlying Islands	20	45	79	
7	Jason Mitchell	evelyn95@example.org	Colombia	87	61	23	
8	Jessica Lowe	richard15@example.com	Norway	18	42	28	
9	Stephen Washington	robincortez@example.com	Ghana	29	33	87	
10	Kylie Davidson	debradiaz@example.org	Antarctica (the territory South of 60 deg S)	48	33	38	
11	Laura Robbins	gonzalezkevin@example.com	Mozambique	7	33	65	
12							

(b) Jupyter Notebook Screenshots:

jupyter P10 Last Checkpoint: 14 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [3]:
```
from faker import Faker
fake = Faker()
import pandas as pd
l1=[]
for i in range(10):
    name=fake.name()
    email=fake.email()
    country=fake.country()
    Python_marks=fake.random_int(0,100)
    Java_marks=fake.random_int(0,100)
    C_marks=fake.random_int(0,100)
    l2=[name,email,country,Python_marks,Java_marks,C_marks]
    l1.append(l2)
print(l1)
```

```
[['Daniel Cohen', 'mortega@example.com', 'Singapore', 75, 40, 58], ['James Johnson', 'grantkaren@example.net', 'Sudan', 44, 42,
58], ['Morgan Ramsey', 'christopher49@example.net', 'Gibraltar', 76, 34, 41], ['Eric Nelson', 'howardmartinez@example.net', 'Au
stralia', 64, 26, 47], ['Diana Hawkins', 'warrenchristina@example.org', 'United States Minor Outlying Islands', 20, 45, 79],
['Jason Mitchell', 'evelyn95@example.org', 'Colombia', 87, 61, 23], ['Jessica Lowe', 'richard15@example.com', 'Norway', 18, 42,
28], ['Stephen Washington', 'robincortez@example.com', 'Ghana', 29, 33, 87], ['Kylie Davidson', 'debradiaz@example.org', 'Antar
ctica (the territory South of 60 deg S)', 48, 33, 38], ['Laura Robbins', 'gonzalezkevin@example.com', 'Mozambique', 7, 33, 65]]
```

In [4]:
```
column=['Name','E-Mail','Country','Python Marks','Java Marks','C Marks']
df = pd.DataFrame(l1,columns=column)
print(df)
```

```
                Name                     E-Mail  \
0        Daniel Cohen         mortega@example.com
1       James Johnson       grantkaren@example.net
2       Morgan Ramsey    christopher49@example.net
3         Eric Nelson   howardmartinez@example.net
4       Diana Hawkins  warrenchristina@example.org
5      Jason Mitchell        evelyn95@example.org
6        Jessica Lowe       richard15@example.com
7  Stephen Washington     robincortez@example.com
8      Kylie Davidson       debradiaz@example.org
9       Laura Robbins    gonzalezkevin@example.com

                                 Country  Python Marks  Java Marks  \
0                              Singapore            75          40
1                                  Sudan            44          42
2                              Gibraltar            76          34
3                              Australia            64          26
4        United States Minor Outlying Islands        20          45
5                               Colombia            87          61
```

jupyter P10 Last Checkpoint: 15 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Run | Code

```
                                   Country  Python Marks  Java Marks  \
0                                Singapore            75          40
1                                    Sudan            44          42
2                                Gibraltar            76          34
3                                Australia            64          26
4         United States Minor Outlying Islands            20          45
5                                 Colombia            87          61
6                                   Norway            18          42
7                                    Ghana            29          33
8  Antarctica (the territory South of 60 deg S)            48          33
9                                Mozambique             7          33

   C Marks
0       58
1       58
2       41
3       47
4       79
5       23
6       28
7       87
8       38
9       65
```

In [5]:
```
df.to_csv('Dataset_for_EXP10_Kunsh.csv',index=False)
```

In [11]:
```
df['Python Marks'].describe()
```

Out[11]:
```
count    10.000000
mean     46.800000
std      27.923706
min       7.000000
25%      22.250000
50%      46.000000
75%      72.250000
max      87.000000
Name: Python Marks, dtype: float64
```

In [12]:
```
df['Java Marks'].describe()
```

Out[12]:
```
count    10.000000
mean     38.900000
std       9.666092
min      26.000000
25%      33.000000
50%      37.000000
```

jupyter P10 Last Checkpoint: 16 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Run Code

In [12]:
```
df['Java Marks'].describe()
```

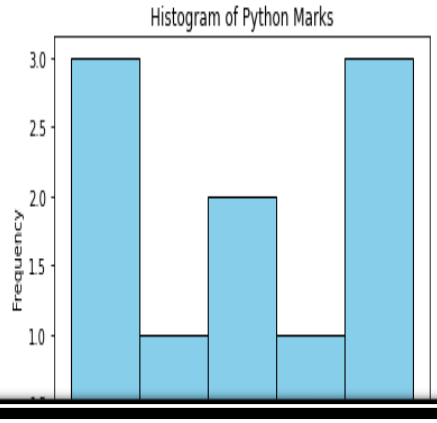
Out[12]:
```
count    10.000000
mean     38.900000
std       9.666092
min      26.000000
25%      33.000000
50%      37.000000
75%      42.000000
max      61.000000
Name: Java Marks, dtype: float64
```

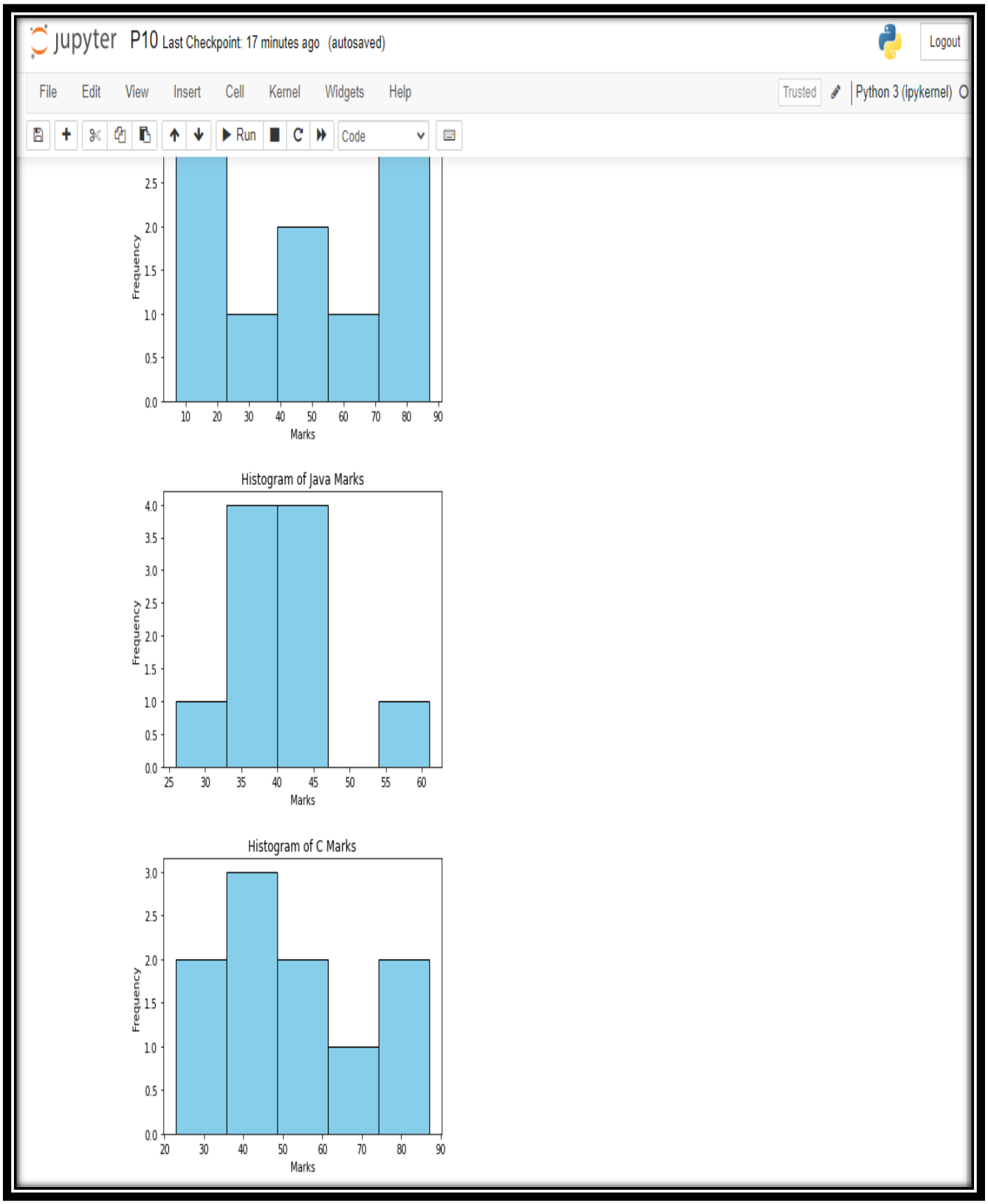
In [13]:
```
df['C Marks'].describe()
```

Out[13]:
```
count    10.000000
mean     52.400000
std      20.956039
min      23.000000
25%      38.750000
50%      52.500000
75%      63.250000
max      87.000000
Name: C Marks, dtype: float64
```

In [15]:
```
import matplotlib.pyplot as plt
```

In [25]:
```
for subject in['Python Marks','Java Marks','C Marks']:
    plt.figure(figsize=(6,4))
    plt.hist(df[subject],bins=5,color='skyblue',edgecolor='black')
    plt.title(f"Histogram of {subject}")
    plt.xlabel('Marks')
    plt.ylabel('Frequency')
    plt.show()
```





Learning Outcome:

EXPERIMENT 11

Aim: Design a Python Program to implement Linear Regression House price prediction using california_housing from scikitlearn.

Theory:

Source Code and Output:

jupyter P11 Last Checkpoint: a minute ago (unsaved changes)

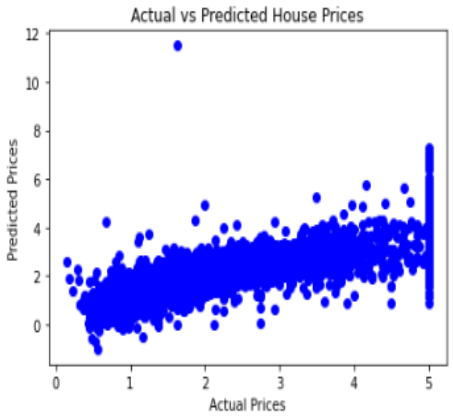
Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [2]:
```
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
import pandas as pd
import matplotlib.pyplot as plt

data = fetch_california_housing()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Price'] = data.target

X = df.drop('Price', axis=1)
y = df['Price']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error: {mae}")
print(f"R-squared: {r2}")

plt.scatter(y_test, y_pred, color='blue')
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.title('Actual vs Predicted House Prices')
plt.show()
```

```
Mean Absolute Error: 0.5332001304956566
R-squared: 0.5757877060324511
```



Learning Outcome:

EXPERIMENT 12

Aim: Design a Python Program to create a recommender system.

Theory:

Source Code and Output:

jupyter Untitled Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[1]:
```python
import pandas as pd
# Create a simple dataset of users and their movie ratings
data = {
    'User': ['User1', 'User1', 'User1', 'User2', 'User2', 'User2', 'User3', 'User3', 'User4', 'User4'],
    'Movie': ['MovieA', 'MovieB', 'MovieC', 'MovieA', 'MovieB', 'MovieD', 'MovieB', 'MovieC', 'MovieA', 'MovieC'],
    'Rating': [5, 3, 4, 4, 2, 5, 2, 4, 5, 3]
}
# Convert the dataset into a DataFrame
df = pd.DataFrame(data)
# Function to recommend movies to a user based on highest ratings from others
def recommend_movies(user):
    # Get movies rated by the user
    user_movies = df[df['User'] == user]
    rated_movies = user_movies['Movie'].tolist()
    # Get the top-rated movies that the user hasn't seen
    recommendations = df[~df['Movie'].isin(rated_movies) & (df['User'] != user)]
    # Sort by ratings
    recommended_movies = recommendations.sort_values(by='Rating', ascending=False).drop_duplicates(subset='Movie')

    return recommended_movies[['Movie', 'Rating']]
# Example: Recommend movies for 'User1'
recommended_movies = recommend_movies('User1')
print("Recommended movies for User1:")
print(recommended_movies)
```

```
Recommended movies for User1:
    Movie  Rating
5  MovieD       5
```

Learning Outcome:

EXPERIMENT 13
(Mini Project)

Aim: Apply Data Pre-Processing Operation on the Titanic Dataset.

Theory:

Source Code and Output:

jupyter P13(Mini Project) Last Checkpoint: 38 minutes ago

File Edit View Run Kernel Settings Help

Code

[5]:
```
# Step 1: Load the Dataset
import pandas as pd

# Load the Titanic dataset from a CSV file
df=pd.read_csv('titanic.csv')

# Display the first few rows of the dataset
print("Initial Data:")
print(df.head())
```

```
Initial Data:
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

[6]:
```
# Step 2: Data Exploration and Visualization
import seaborn as sns
import matplotlib.pyplot as plt

# Basic information about the dataset
print("\nData Info:")
```

jupyter P13(Mini Project) Last Checkpoint: 38 minutes ago

File Edit View Run Kernel Settings Help

Code

```
print(df.info())

# Check for missing values
print("\nMissing Values:")
print(df.isnull().sum())

# Visualize the distribution of survival
sns.countplot(data=df, x='Survived')
plt.title('Survived Count')
plt.show()

# Visualize survival based on gender
sns.countplot(data=df, x='Survived', hue='Sex')
plt.title('Survival Count by Gender')
plt.show()

# Visualize survival based on passenger class
sns.countplot(data=df, x='Survived', hue='Pclass')
plt.title('Survival Count by Passenger Class')
plt.show()
```

```
Data Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
```

jupyter P13(Mini Project) Last Checkpoint: 39 minutes ago

File Edit View Run Kernel Settings Help

Code

```
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
```

[3]:

```python
# Step 3: Data Cleaning
# Fill missing values for 'Age' with the median
df['Age'].fillna(df['Age'].median(), inplace=True)

# Drop the 'Cabin' column due to high missing values
df.drop(columns=['Cabin'], inplace=True)

# Drop 'Ticket' column as it does not provide useful information
df.drop(columns=['Ticket'], inplace=True)

# Drop rows with missing 'Embarked' values
df.dropna(subset=['Embarked'], inplace=True)

print("\nData after cleaning:")
print(df.info())
```

```
Data after cleaning:
<class 'pandas.core.frame.DataFrame'>
Index: 889 entries, 0 to 890
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  889 non-null    int64
 1   Survived     889 non-null    int64
 2   Pclass       889 non-null    int64
 3   Name         889 non-null    object
 4   Sex          889 non-null    object
 5   Age          889 non-null    float64
 6   SibSp        889 non-null    int64
 7   Parch        889 non-null    int64
 8   Fare         889 non-null    float64
 9   Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(3)
memory usage: 76.4+ KB
None
```

jupyter P13(Mini Project) Last Checkpoint: 40 minutes ago

File Edit View Run Kernel Settings Help

Code

[7]:
```
# Step 4: Feature Engineering
# Convert categorical variables to numerical
df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})
df = pd.get_dummies(df, columns=['Embarked'], drop_first=True)

# Select features and target variable
X = df[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_Q', 'Embarked_S']]
y = df['Survived']
```

[8]:
```
# Step 5: Model Training
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the Random Forest Classifier
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

[8]:
```
RandomForestClassifier
RandomForestClassifier()
```

[9]:
```
# Step 6: Model Evaluation
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Make predictions
y_pred = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"\nModel Accuracy: {accuracy:.2f}")

# Print classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

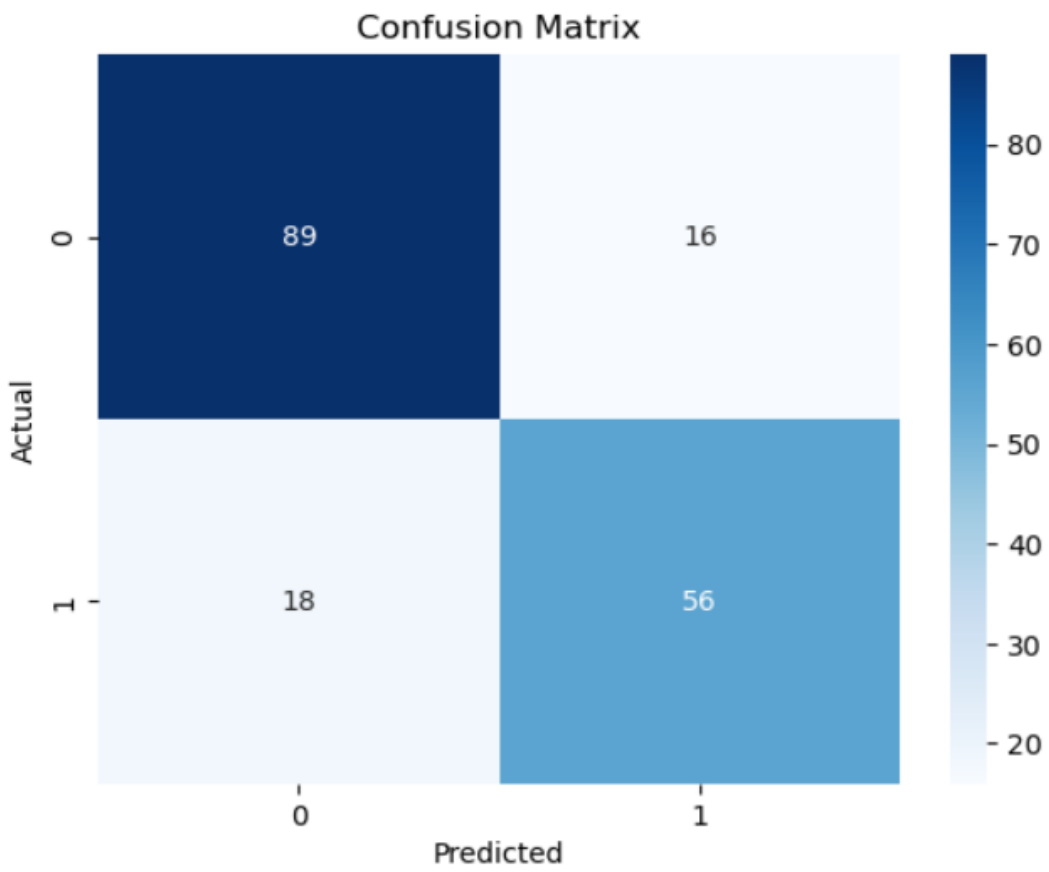
jupyter P13(Mini Project) Last Checkpoint: 42 minutes ago

File Edit View Run Kernel Settings Help

Code

```
# Plot confusion matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
Model Accuracy: 0.81

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.85      0.84       105
           1       0.78      0.76      0.77        74

    accuracy                           0.81       179
   macro avg       0.80      0.80      0.80       179
weighted avg       0.81      0.81      0.81       179
```

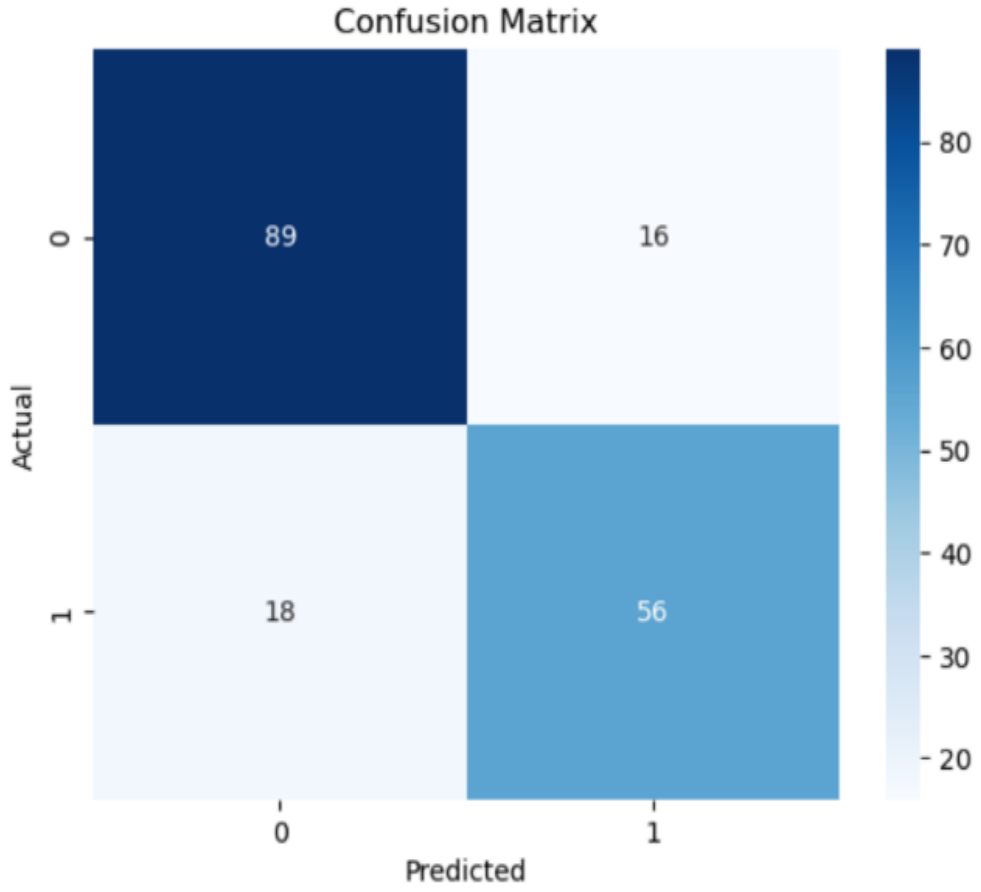


jupyter P13(Mini Project) Last Checkpoint: 42 minutes ago

File Edit View Run Kernel Settings Help

Code

```
macro avg       0.80      0.80      0.80       179
weighted avg    0.81      0.81      0.81       179
```



[10]:
```
# Step 7: Conclusion
print("\nConclusion:")
print("The Random Forest Classifier performed well on the Titanic dataset.")
```

```
Conclusion:
The Random Forest Classifier performed well on the Titanic dataset.
```

Learning Outcome: