

# Spring 2023: CS5710 – Machine Learning

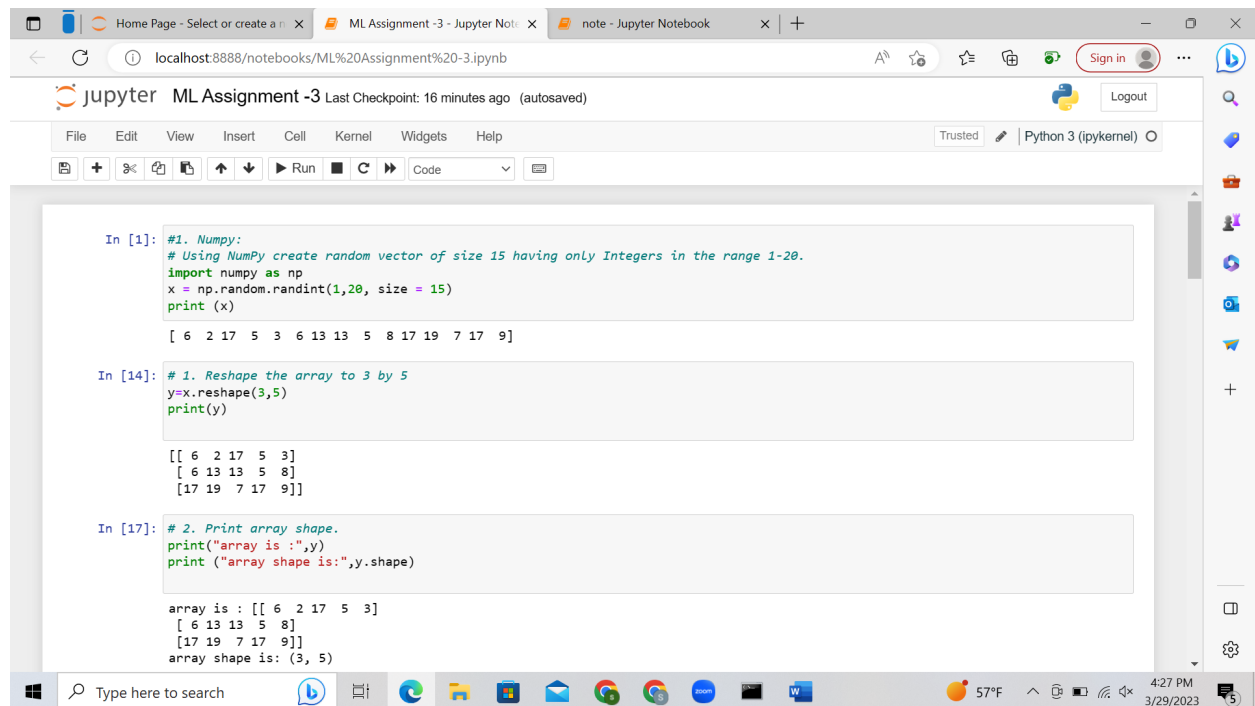
## In-Class Programming Assignment-3

### 1. Numpy:

a. Using NumPy create random vector of size 15 having only Integers in the range 1-20.

1. Reshape the array to 3 by 5
2. Print array shape.
3. Replace the max in each row by 0

Ans



```
In [1]: #1. Numpy:
# Using NumPy create random vector of size 15 having only Integers in the range 1-20.
import numpy as np
x = np.random.randint(1,20, size = 15)
print(x)

[ 6  2 17  5  3  6 13 13  5  8 17 19  7 17  9]

In [14]: # 1. Reshape the array to 3 by 5
y=x.reshape(3,5)
print(y)

[[ 6  2 17  5  3]
 [ 6 13 13  5  8]
 [17 19  7 17  9]]

In [17]: # 2. Print array shape.
print("array is :",y)
print ("array shape is:",y.shape)

array is : [[ 6  2 17  5  3]
 [ 6 13 13  5  8]
 [17 19  7 17  9]]
array shape is: (3, 5)
```

This screenshot shows a Jupyter Notebook interface with three code cells. The first cell reshapes an array `y` from shape (3, 5) to (5, 3). The second cell prints the shape and type of `y`. The third cell replaces the maximum value in each row of `y` with 0, creating a new array `new_a`.

```
y=x.reshape(3,5)
print(y)

[[ 6  2 17  5  3]
 [ 6 13 13  5  8]
 [17 19  7 17  9]]

In [17]: # 2. Print array shape.
print("array is :",y)
print ("array shape is:",y.shape)

array is : [[ 6  2 17  5  3]
 [ 6 13 13  5  8]
 [17 19  7 17  9]]
array shape is: (3, 5)

In [5]: # 3. Replace the max in each row by 0
new_a = np.where(y == [
    [i]
    for i in np.amax(y, axis = 1)
], 0, y)

print(new_a)

[[ 6  2  0  5  3]
 [ 6  0  0  5  8]
 [17  0  7 17  9]]
```

Create a 2-dimensional array of size 4 x 3 (composed of 4-byte integer elements), also print the shape, type and data type of the array.

This screenshot shows a Jupyter Notebook interface with two code cells. The first cell prints the result of a `np.where` operation. The second cell creates a 4x3 array `arr` of type `int32` and prints its shape, type, and data type.

```
] , 0, y)

print(new_a)

[[ 6  2  0  5  3]
 [ 6  0  0  5  8]
 [17  0  7 17  9]]

In [8]: # Create a 2-dimensional array of size 4 x 3 (composed of 4-byte integer elements), also print the shape, type and data type
# of the array.
import numpy as np

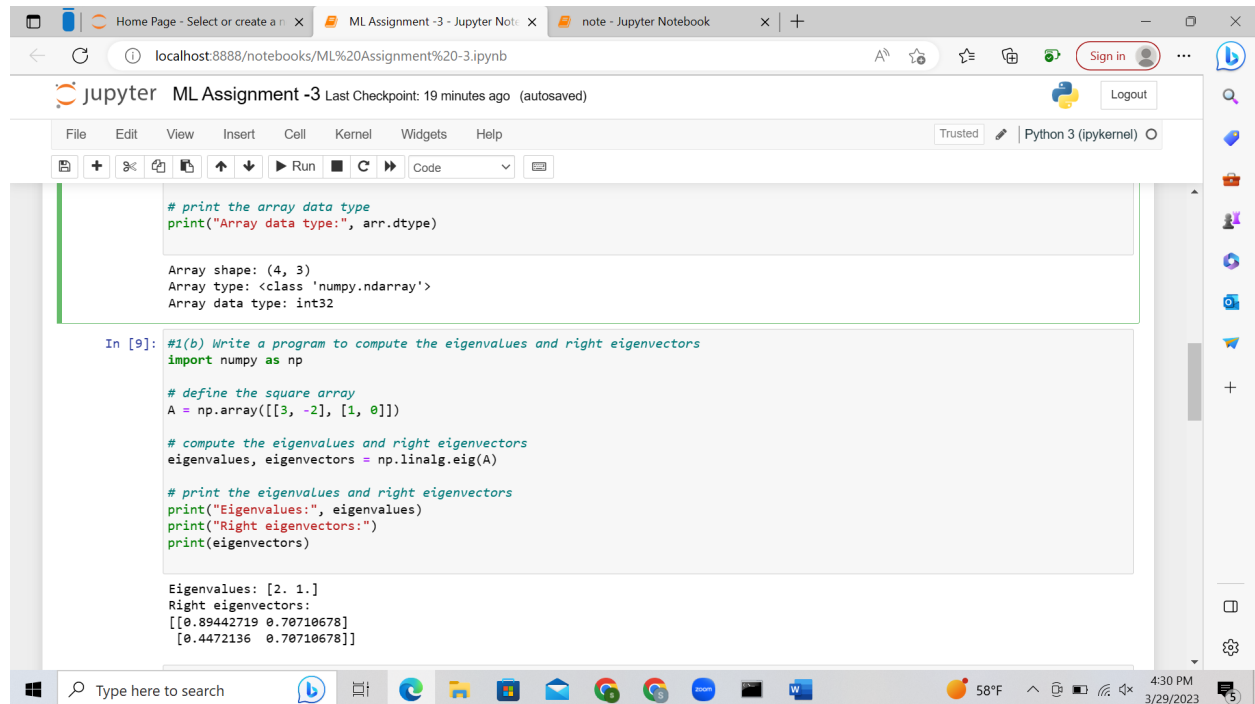
# create a 2-dimensional array of size 4x3
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]], dtype=np.int32)

# print the array shape
print("Array shape:", arr.shape)
# print the array type
print("Array type:", type(arr))
# print the array data type
print("Array data type:", arr.dtype)

Array shape: (4, 3)
Array type: <class 'numpy.ndarray'>
Array data type: int32
```

b. Write a program to compute the eigenvalues and right eigenvectors of a given square array given below:

```
[[ 3 -2]
 [ 1  0]]
```



The screenshot shows a Jupyter Notebook titled "ML Assignment -3" with a last checkpoint 19 minutes ago. The notebook contains two code cells. The first cell prints the array data type and shape. The second cell, labeled "In [9]:", implements a program to compute the eigenvalues and right eigenvectors of the matrix  $\begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix}$  using NumPy's `linalg.eig` function. The output shows the eigenvalues  $[2. \ 1.]$  and the right eigenvectors  $\begin{bmatrix} 0.89442719 & 0.70710678 \\ 0.4472136 & 0.70710678 \end{bmatrix}$ .

```
# print the array data type
print("Array data type:", arr.dtype)

Array shape: (4, 3)
Array type: <class 'numpy.ndarray'>
Array data type: int32

In [9]: #1(b) Write a program to compute the eigenvalues and right eigenvectors
import numpy as np

# define the square array
A = np.array([[3, -2], [1, 0]])

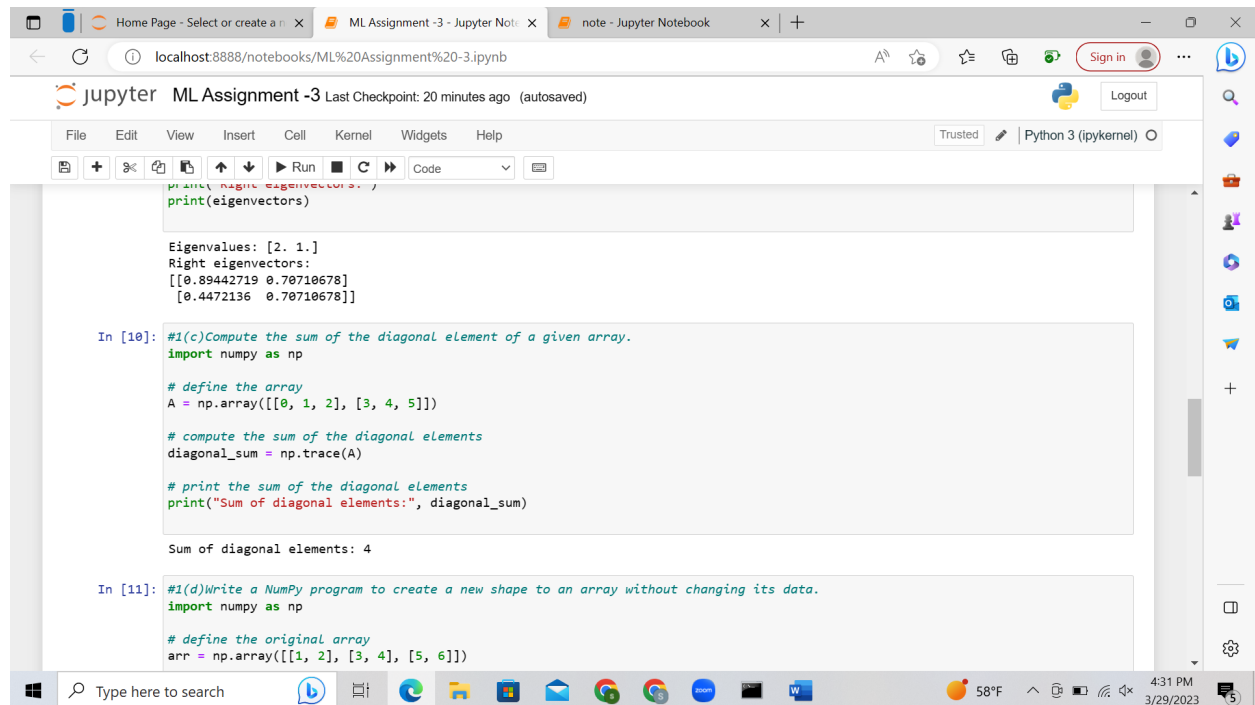
# compute the eigenvalues and right eigenvectors
eigenvalues, eigenvectors = np.linalg.eig(A)

# print the eigenvalues and right eigenvectors
print("Eigenvalues:", eigenvalues)
print("Right eigenvectors:")
print(eigenvectors)

Eigenvalues: [2. 1.]
Right eigenvectors:
[[0.89442719 0.70710678]
 [0.4472136  0.70710678]]
```

c. Compute the sum of the diagonal element of a given array.

```
[[0 1 2]
 [3 4 5]]
```



The screenshot shows a Jupyter Notebook titled "ML Assignment -3" with a last checkpoint 20 minutes ago. The notebook contains two code cells. The first cell, labeled "In [10]:", implements a program to compute the sum of the diagonal elements of the array  $\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix}$  using NumPy's `trace` function. The output shows the sum of diagonal elements is 4. The second cell, labeled "In [11]:", implements a program to create a new shape to an array without changing its data.

```
print("Right eigenvectors:")
print(eigenvectors)

Eigenvalues: [2. 1.]
Right eigenvectors:
[[0.89442719 0.70710678]
 [0.4472136  0.70710678]]

In [10]: #1(c) Compute the sum of the diagonal element of a given array.
import numpy as np

# define the array
A = np.array([[0, 1, 2], [3, 4, 5]])

# compute the sum of the diagonal elements
diagonal_sum = np.trace(A)

# print the sum of the diagonal elements
print("Sum of diagonal elements:", diagonal_sum)

Sum of diagonal elements: 4

In [11]: #1(d) Write a NumPy program to create a new shape to an array without changing its data.
import numpy as np

# define the original array
arr = np.array([[1, 2], [3, 4], [5, 6]])
```

d. Write a NumPy program to create a new shape to an array without changing its data.

Reshape 3x2:

```
[[1 2]
```

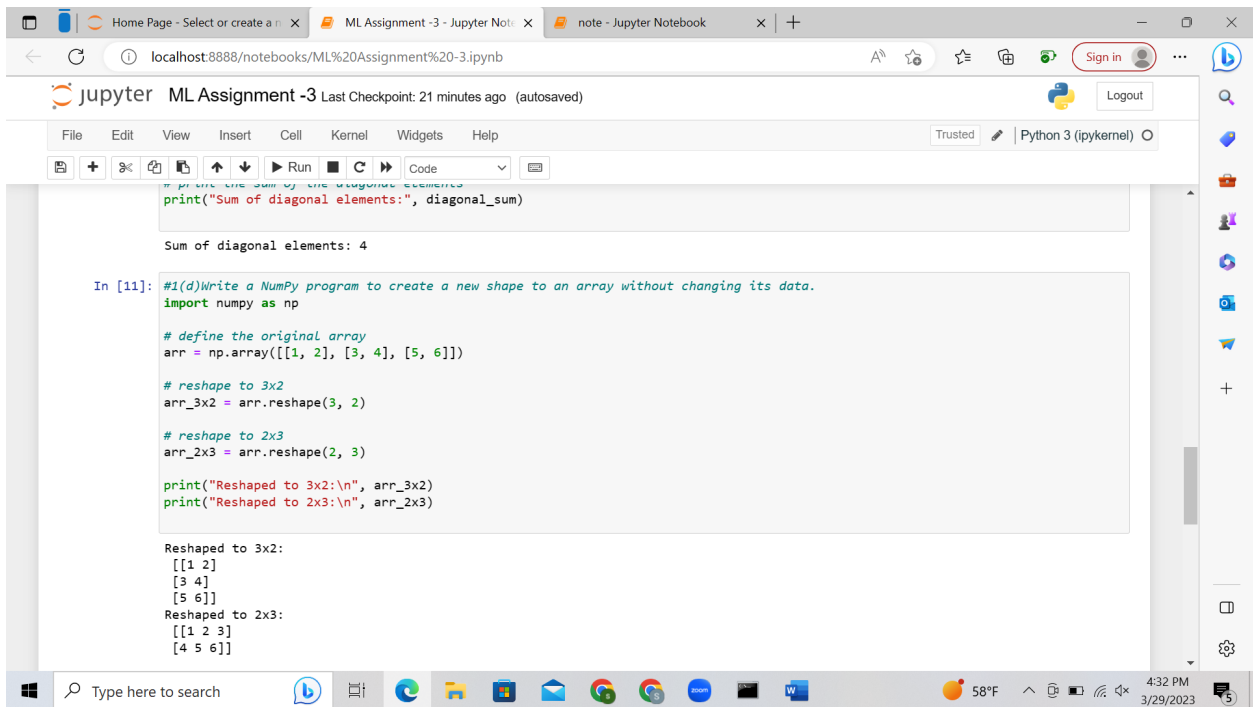
```
[3 4]
```

```
[5 6]]
```

Reshape 2x3:

```
[[1 2 3]
```

```
[4 5 6]]
```



```
print("Sum of the diagonal elements")
print("Sum of diagonal elements:", diagonal_sum)

Sum of diagonal elements: 4

In [11]: #1(d)Write a NumPy program to create a new shape to an array without changing its data.
import numpy as np

# define the original array
arr = np.array([[1, 2], [3, 4], [5, 6]])

# reshape to 3x2
arr_3x2 = arr.reshape(3, 2)

# reshape to 2x3
arr_2x3 = arr.reshape(2, 3)

print("Reshaped to 3x2:\n", arr_3x2)
print("Reshaped to 2x3:\n", arr_2x3)

Reshaped to 3x2:
[[1 2]
 [3 4]
 [5 6]]
Reshaped to 2x3:
[[1 2 3]
 [4 5 6]]
```

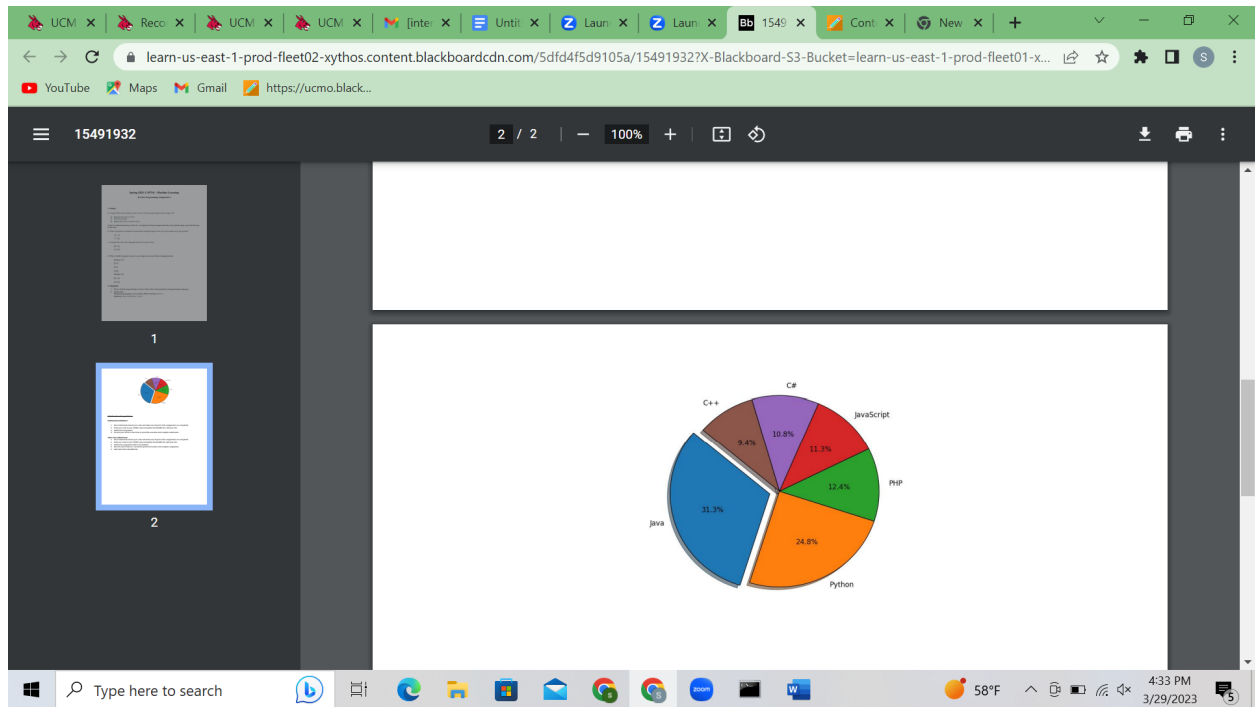
## 2. Matplotlib

1. Write a Python programming to create a below chart of the popularity of programming Languages.

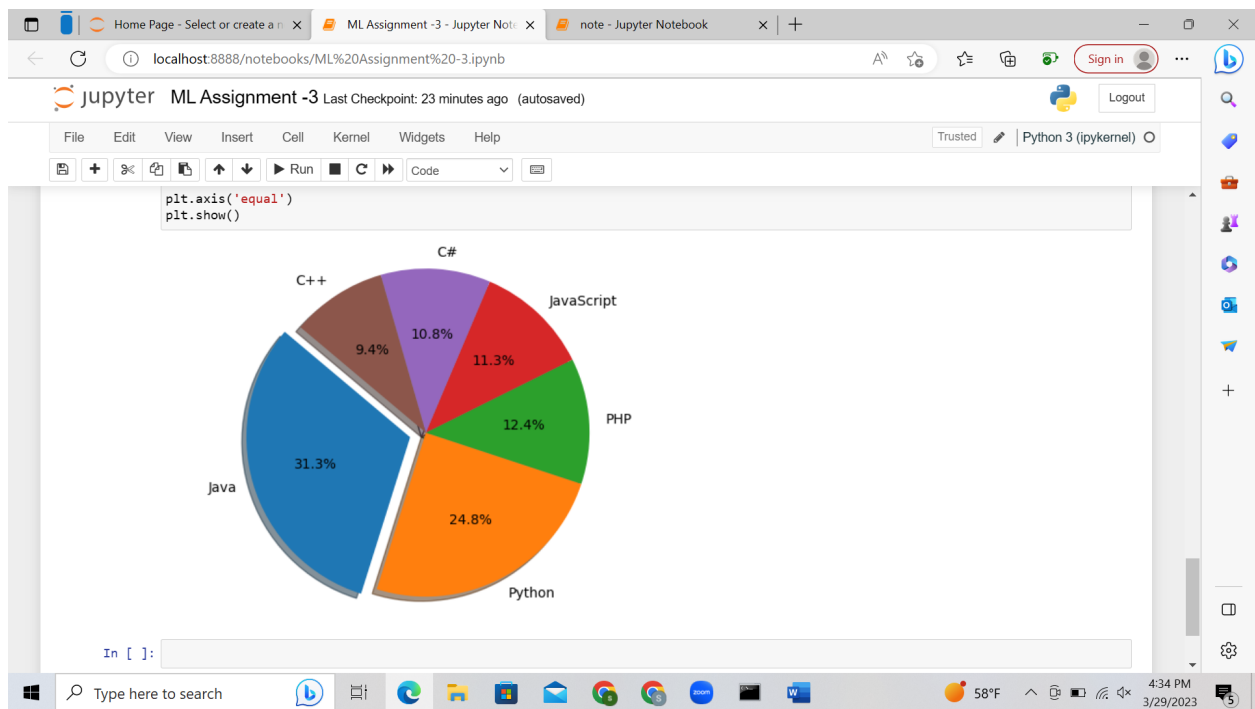
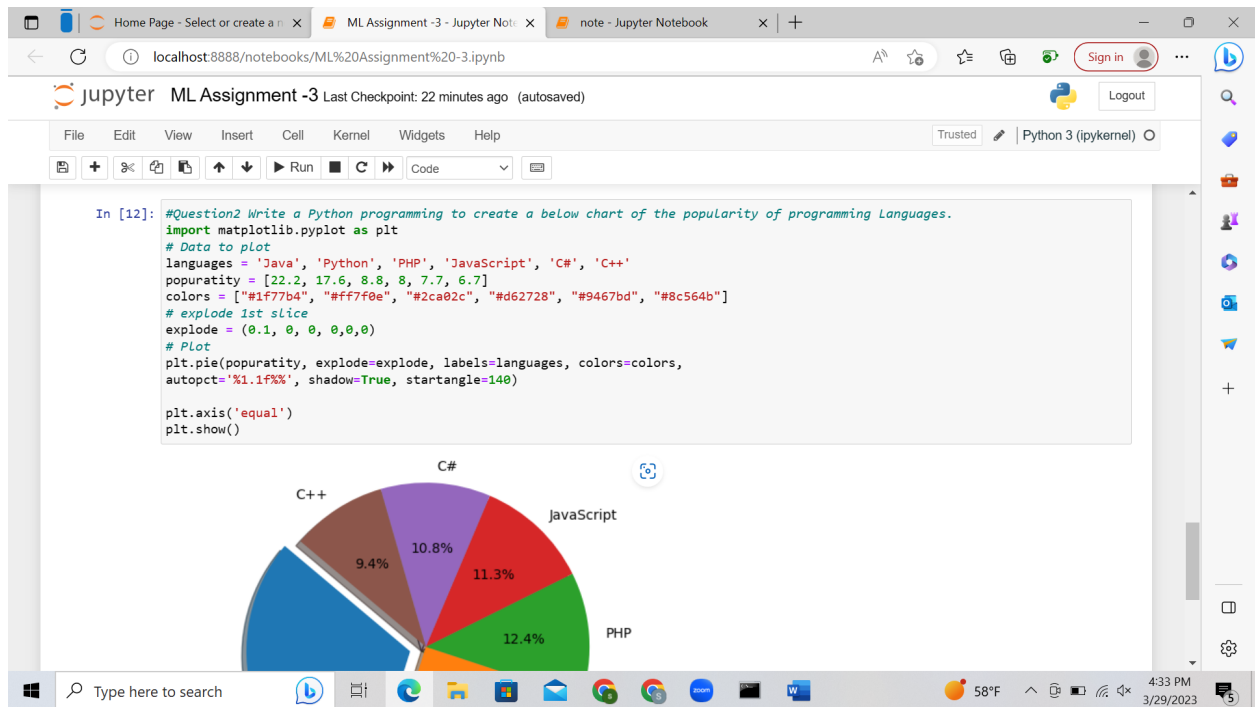
2. Sample data:

Programming languages: Java, Python, PHP, JavaScript, C#, C++

Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7



Ans



GitRepo link : <https://github.com/ShruthiVallapReddy/ML---assignment-3.git>