CS 5710 Machine Learning

In-Class Programming Assignment-3

Name: Aravinda Krishna Gorantla

ID: 700741775

GitHub Link: https://github.com/gakrish5/MachineLearning/tree/main/Assignment%203

1. NumPy:

Importing the required libraries.

```
import numpy as np
import random
```

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

Source code:

```
rand_vector = np.random.randint(low=1, high=21, size=15)
print(rand_vector)
```

Output:

```
[12 13 11 19 10  6  1 15 13  1  8  4  8  2  3]
```

Explanation:

Here in the code, ***randint*( ) function of ***random*** module from ***numpy*** library is used to generate the random vector of size 15 having only integers in the range 1-20.

Then the vector is printed.

Question 1:

Reshape the array to 3 by 5.

Source code:

```
# Reshaping the array to 3x5
array1 = rand_vector.reshape(3, 5)
array1
```

Output:

```
array([[12, 13, 11, 19, 10],
       [ 6,  1, 15, 13,  1],
       [ 8,  4,  8,  2,  3]])
```

Explanation:

Here in the code, I have used ***reshape*( )** function to reshape the array to 3 by 5.

Then the updated array is printed.

Question 2:

Print array shape.

Source code:

```
print("The shape of the array is:", array1.shape)
```

Output:

```
The shape of the array is: (3, 5)
```

Explanation:

Here in the code, I have used the ***shape*** attribute to display the shape of the array.

Question 3:

Replace the max in each row by 0.

Source code:

```
# Finding the maximum elements index in each row
max_indexes = np.argmax(array1, axis=1)
i = 0

# Iterating over the max_indexes
for j in max_indexes:
    array1[i][j] = 0
    i += 1

print("updated Array:\n", array1)
```

Output:

```
updated Array:
 [[12 13 11  0 10]
 [ 6  1  0 13  1]
 [ 0  4  8  2  3]]
```

Explanation:

Here in the code, I have used the ***argmax*( )** function with ***axis*** parameter to get the maximum valued index of each row and stored in a variable ***max_indexes***.

The maximum value of each row to 0 is updated by iterating over the ***max_indexes*** and using a counter variable to iterate over the rows of original array.

Then the updated array is printed.

Question:

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.

Source code:

```
# Creating a 2-d array of size 4x3
array2 = np.array(np.random.randint(1, 21, size=(4, 3)), np.int32)

# printing array shape
print("Shape:", array2.shape)

# printing array type
print("Type:", type(array2))

#printing array data type
print("Data type:", array2.dtype)
```

Output:

```
Shape: (4, 3)
Type: <class 'numpy.ndarray'>
Data type: int32
```

Explanation:

Here in the code, ***randint*** ( ) function of ***random*** module from ***numpy*** library is used to create a 2-dimensional array of size 4 x 3.

Then ***shape*** attribute, ***type*** ( ) function and ***dtype*** attribute is used to print the shape, type, and data type of the array respectively.

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

$$\begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix}$$

Source code:

```
# Defining the given array
array3 = np.array([[3, -2], [1, 0]])

# computing the eigenvalues and right eigenvectors
eigenvalues, eigenvectors = np.linalg.eig(array3)

# printing eigenvalues
print("Eigenvalues: \n", eigenvalues)

# printing right eigenvectors
print("\nRight Eigenvectors: \n", eigenvectors)
```

Output:

```
Eigenvalues:
 [2. 1.]

Right Eigenvectors:
 [[0.89442719 0.70710678]
 [0.4472136  0.70710678]]
```

Explanation:

Here in the code, I have declared the given square array using ***array***( ) function of ***numpy*** library.

Then used the ***eig***( ) function of ***linalg*** module of ***numpy*** library on the declared array to get eigenvalues and right eigenvectors and then they are printed.

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

$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{bmatrix}$$

Source code:

```
# Defining the given array
array4 = np.array([[0, 1, 2], [3, 4, 5]])

# computing the sum of the diagonal elements
sum_diagonal = np.trace(array4)

# Printing the sum of the diagonal elements
sum_diagonal
```

Output:

```
4
```

Explanation:

Here in the code, I have declared the given array using ***array***( ) function of ***numpy*** library.

Then used the ***trace***( ) function of ***numpy*** library on the declared array to get the sum of the diagonal element and then the value is printed.

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

Question 1:

Reshape 3x2:

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

Source code:

```
#Defining the given array
array5 = np.array([[1, 2], [3, 4], [5, 6]])

# Reshaping the array to 2x3
new_arr1 = array5.reshape(2,3)

# printing the new array
new_arr1
```

Output:

```
array([[1, 2, 3],
       [4, 5, 6]])
```

Explanation:

Here in the code, I have used ***reshape***( ) function to reshape the array to 2 by 3.

Then the updated array is printed.

Question 2:

Reshape 2x3:

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

Source code:

```
#Defining the given array
array6 = np.array([[1, 2, 3], [4, 5, 6]])

# Reshaping the array to 3x2
new_arr2 = array6.reshape(3,2)

# printing the new array
new_arr2
```

Output:

```
array([[1, 2],
       [3, 4],
       [5, 6]])
```

Explanation:

Here in the code, I have used ***reshape*** ( ) function to reshape the array to 3 by 2.

Then the updated array is printed.

2. Matplotlib:

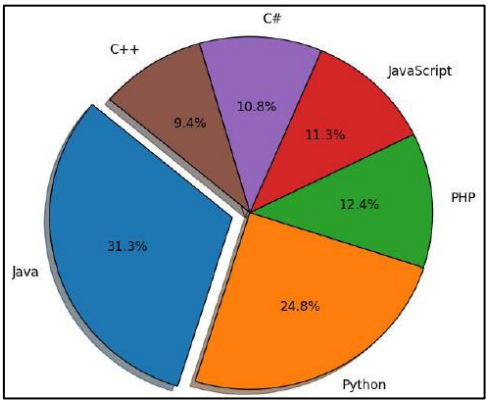
Write a Python programming to create a below chart of the popularity of programming Languages.

Sample data:

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

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

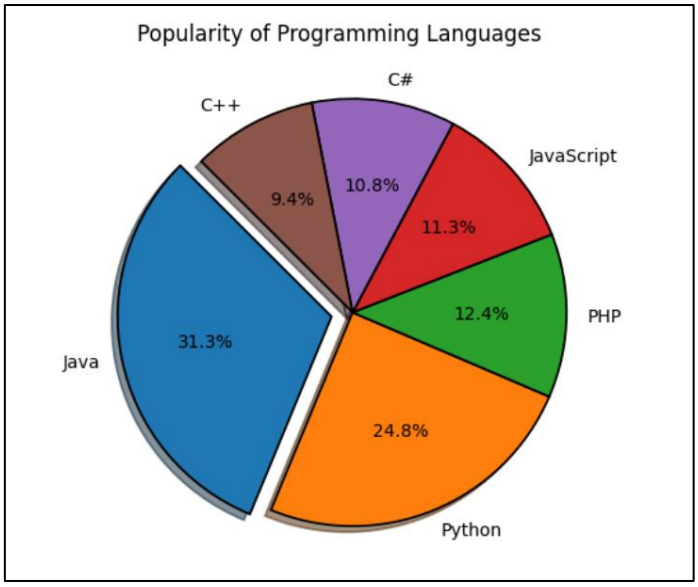
Sample Output:



Source code:

```
import matplotlib.pyplot as plt

# Sample data to plot
languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

# Exploding the first slice
y = [0.1, 0, 0, 0, 0, 0]

# Creating a pie chart
plt.pie(popularity, explode=y, labels=languages, startangle=135, shadow=True,
        wedgeprops = {"edgecolor" : "black",'linewidth': 1.25},
        autopct='%1.1f%%')

# Setting the title for the pie chart
plt.title('Popularity of Programming Languages \n')

# axis() function is used to adjust the axis and making the chart circular using the arugument equal
plt.axis('equal')

# Displaying the chart
plt.show()
```

Output:



Explanation:

Here in the code, $\boldsymbol{matplotlib.pyplot}$ library is imported, given sample data is declared.

The $\boldsymbol{pie}()$ function with the below parameters is used to plot the desired graph.

$\boldsymbol{explode}$ – to explode one slice of the pie chart.

$\boldsymbol{labels}$ - to label each slice of the pie chart.

$\boldsymbol{startangle}$ – to set the starting angle of the pie chart in degrees (default 0^0).

$\boldsymbol{shadow}$ – a Boolean parameter to add a shadow to the pie chart.

$\boldsymbol{wedgeprops}$ – to set properties for each wedge of the pie chart. I have used a dictionary to set edge color and width of each wedge.

$\boldsymbol{autopct}$ – to specify the format for the percentage values that are displayed for each slice. I have used $\%\mathbf{1}.\mathbf{1}\boldsymbol{f}\%\%$ format string to display the percentage value rounded to one decimal place.

$\boldsymbol{title}()$ function is used to set a title for the plot.

$\boldsymbol{axis}()$ function is used with $\boldsymbol{equal}$ paramter to adjust the axis and making the chart circular.

Then the plot is displayed using the $\boldsymbol{show}()$ function of $\boldsymbol{matplotlib}$ library.

- - - - End - - - -