CS 5710 Machine Learning

In-Class Programming Assignment

Name: Vikram Boppana

ID: 700742246

GitHub Link: https://github.com/viboppan/ML-Assignment1

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

```
In [22]: #1.Numpy
         #a. Create a random vector of size 15 with integers in the range 1-20
         import numpy as np
         import random
         rand_vector = np.random.randint(low=1, high=21, size=15)
         print(rand_vector)
```

```
[15  5  6  3 13 19  9 13 14  3 17  2 19  2  1]
```

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.

```
In [23]: #1. Reshape the vector to a 3 by 5 array
         reshaped_array = rand_vector.reshape(3, 5)
         print("\nReshaped Array:")
         print(reshaped_array)
```

```
Reshaped Array:
[[15  5  6  3 13]
 [19  9 13 14  3]
 [17  2 19  2  1]]
```

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.

```
In [24]: #2. Print array shape
         print("\nArray Shape:")
         print(reshaped_array.shape)

         Array Shape:
         (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.

```
In [25]: # Question 3: Replace the max in each row by 0
         max_values_indices = np.argmax(reshaped_array, axis=1)
         print(max_values_indices)
         i = 0
         # Iterating over the max_values_indices
         for j in max_values_indices:
             reshaped_array[i][j] = 0
             i += 1

         print("updated Array:\n", reshaped_array)

         [0 0 2]
         updated Array:
          [[ 0  5  6  3 13]
          [ 0  9 13 14  3]
          [17  2  0  2  1]]
```

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

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.

```
In [26]: # Creating a 2-d array of size 4x3
secondaArray = np.array(np.random.randint(1, 21, size=(4, 3)), np.int32)

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

# print array type
print("Type:", type(secondaArray))

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

```
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: [[3 –2] [1 0]]

```
In [30]: # Defining the given array
thirdArray = np.array([[3, -2], [1, 0]])

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

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

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

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

```
In [ ]: #c. Compute the sum of the diagonal element of a given array.
        #[[0 1 2] [3 4 5]]
```

```
In [31]: # Defining the given array
         fourthArray = np.array([[0, 1, 2], [3, 4, 5]])

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

         # Print the sum of the diagonal elements
         sum_diagonal

Out[31]: 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]]

```
In [33]: #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]]
```

```
In [34]: #Defining the given array
         array5 = np.array([[1, 2], [3, 4], [5, 6]])

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

         # print the new array
         new_arr1

Out[34]: 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.

Reshape 2x3: [[1 2 3] [4 5 6]]

```
In [35]: #Reshape 2x3: [[1 2 3] [4 5 6]]
```

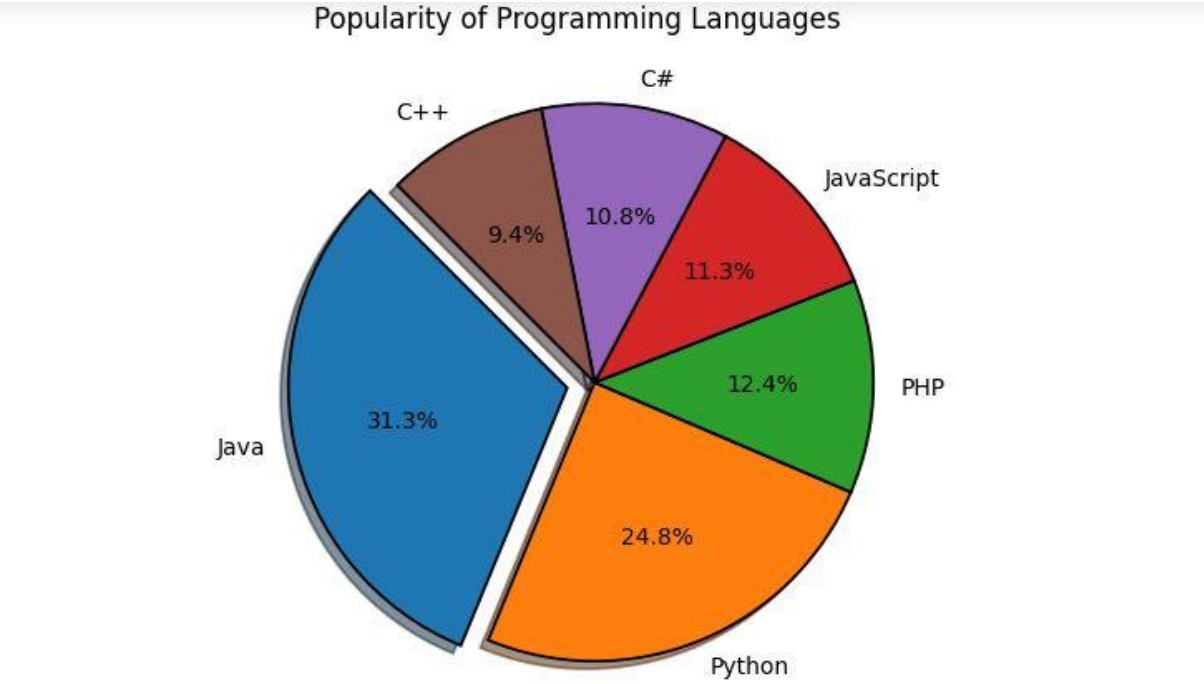
```
In [36]: #Defining the given array
         array6 = np.array([[1, 2, 3], [4, 5, 6]])

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

         # printing the new array
         new_arr

Out[36]: 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

```
In [38]: 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, **matplotlib.pyplot** library is imported, given sample data is declared. The **pie**( ) function with the below parameters is used to plot the desired graph. ******** – to explode one slice of the pie chart.

explode - to label each slice of the pie chart.

labels – to set the starting angle of the pie chart in degrees (default 00).

startangle – a Boolean parameter to add a shadow to the pie chart

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.

autopct – to specify the format for the percentage values that are displayed for each slice. I have used %1.1f%% format string to display the percentage value rounded to one decimal place.

title( ) function is used to set a title for the plot.

axis( ) function is used with **equal** paramter to adjust the axis and making the chart circular. Then the plot is displayed using the **show**( ) function of **matplotlib** library.