Machine Learning (Assignment # 3)

```
[1]: import numpy as np

     # create random vector of size 15 with integers in range 1-20
     vec = np.random.randint(1, 21, size=15)

     # reshape to 3 by 5
     arr = vec.reshape(3, 5)

     # print array shape
     print("Array shape:", arr.shape)

     # replace max in each row by 0
     arr[np.arange(len(arr)), arr.argmax(axis=1)] = 0

     print(arr)
```

```
Array shape: (3, 5)
[[15  2  0 16  8]
 [ 1 13  0  7  7]
 [ 2  9 10  0 11]]
```

This code uses NumPy to create a random vector of 15 integers, reshape it into a 3x5 array, and replace the maximum value in each row with 0. It first generates a vector using random.randint(), then reshapes it with reshape(), and displays the shape of the array using print(). The maximum value in each row is replaced with 0 using argmax() and NumPy's indexing syntax. The resulting array is then displayed using print().

```
import numpy as np

# create square array
arr = np.array([[3, -2], [1, 0]])

# compute eigenvalues and right eigenvectors
eig_vals, eig_vecs = np.linalg.eig(arr)

print("Eigenvalues:", eig_vals)
print("Right eigenvectors:\n", eig_vecs)
```

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

This code uses NumPy to compute the eigenvalues and right eigenvectors of a given 2x2 square array. The array is defined using np.array(), and the eigenvalues and right eigenvectors are computed using np.linalg.eig(). The resulting eigenvalues and eigenvectors are displayed using print().

Machine Learning (Assignment # 3)

```python
import numpy as np

# create array
arr = np.array([[0, 1, 2], [3, 4, 5]])

# compute sum of diagonal elements
diag_sum = np.trace(arr)

print("Sum of diagonal elements:", diag_sum)
```

```
Sum of diagonal elements: 4
```

This code uses NumPy to compute the sum of the diagonal elements of a 2x3 array. The array is defined using np.array(), and the sum of diagonal elements is computed using np.trace(). The resulting sum is displayed using print().

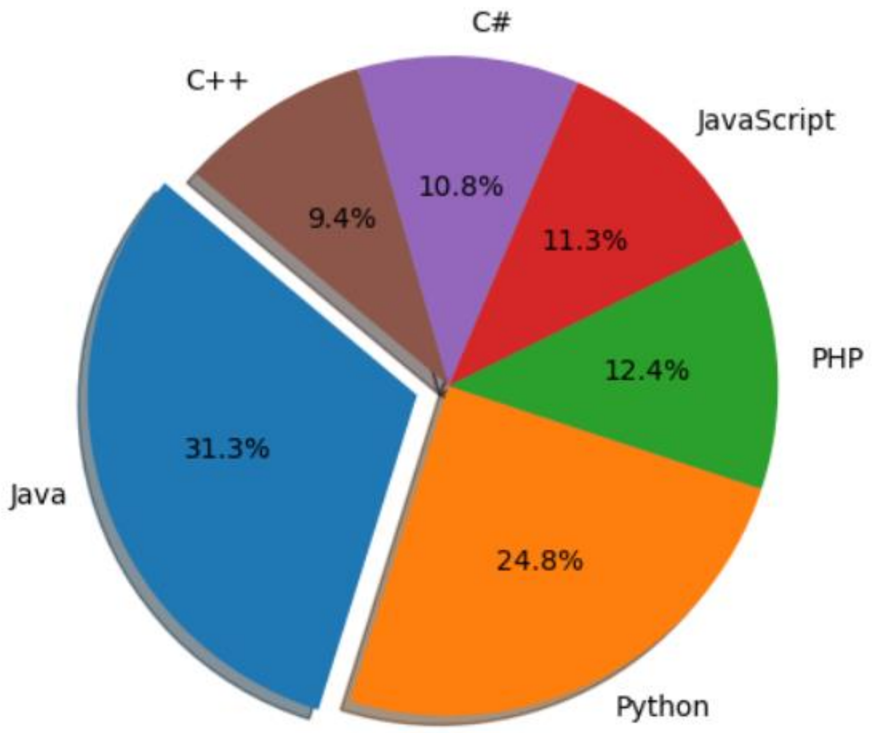
```python
import numpy as np

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

# reshape to 3 by 2
arr_3by2 = arr.reshape(3, 2)

# reshape to 2 by 3
arr_2by3 = arr.reshape(2, 3)

print("Original array:\n", arr)
print("Reshaped to 3 by 2:\n", arr_3by2)
print("Reshaped to 2 by 3:\n", arr_2by3)
```

```
Original array:
 [[1 2]
 [3 4]
 [5 6]]
Reshaped to 3 by 2:
 [[1 2]
 [3 4]
 [5 6]]
Reshaped to 2 by 3:
 [[1 2 3]
 [4 5 6]]
```

This code uses NumPy to create a 2-dimensional array, reshape it to a 3x2 array, and then to a 2x3 array. The original array is defined using np.array(), and the reshape() function is used to reshape the array to the desired shapes. The resulting arrays are then displayed using print().

Machine Learning (Assignment # 3)

```
[7]: # Import the matplotlib.pyplot module, which allows us to create plots
     import matplotlib.pyplot as plt

     # Define the data we want to plot
     languages = 'Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
     popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
     colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]

     # Define how much we want to "explode" each slice of the pie chart
     explode = (0.1, 0, 0, 0, 0, 0)

     # Use the pie function to create the pie chart
     plt.pie(popularity,    # The data to plot (popularity percentages)
             explode=explode,    # How much to "explode" each slice
             labels=languages,   # Labels for each slice (the language names)
             colors=colors,      # Colors for each slice
             autopct='%1.1f%%',  # Format for the percentage labels
             shadow=True,        # Whether to include a shadow effect
             startangle=140      # The angle at which the chart starts
            )

     # Set the aspect ratio of the chart to be equal, so it appears circular
     plt.axis('equal')

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



This is a Python code snippet that uses the matplotlib.pyplot module to create a pie chart showing the popularity of different programming languages. The code defines the data to plot, the colors to use for each slice of the chart, and how much to "explode" each slice. The pie function is then used to create the chart, with various options set for labeling, formatting, and appearance. Finally, the aspect ratio is set to be equal and the chart is displayed using the show() function.

GitHub URL:

https://github.com/SaiPriyankaNarra/ML_700741613_Assignment-3.git

Video URL:

https://drive.google.com/file/d/1kDrPmkkIDk2ydvmPMvYkS0mbTS5bVALC/view?usp=share_link