

NEURAL_NETWORKS

ICP-3

```
# 1. Create a class Employee and then do the following
# • Create a data member to count the number of Employees
# • Create a constructor to initialize name, family, salary, department
# • Create a function to average salary
# • Create a Fulltime Employee class and it should inherit the properties
of Employee class
# • Create the instances of Fulltime Employee class and Employee class and
call their member functions.
```

```
class Employee:
    emp_count = 0

    def __init__(self, name, family, salary, department):

        self.name = name
        self.family = family
        self.salary = salary
        self.department = department
        Employee.emp_count = Employee.emp_count + 1
    def avg_sal(self, emps):
        sum_sal = 0
        for i in emps:
            sum_sal= sum_sal+ i.salary

        print(sum_sal/len(emps))

class Fulltime_Employee(Employee):

    def __init__(self, name, family, salary, department):
        Employee.__init__(self, name, family, salary, department)

list = []
list.append(Employee('John', 'Doe', 30000, 'IT'))
list.append(Employee('Jane', 'Levy', 45000, 'Sales'))

list.append(Fulltime_Employee('Jack', 'Sparrow', 23000, 'Design'))
list.append(Fulltime_Employee('Jared', 'Leto', 50000, 'Gaming'))

list[0].avg_sal(list)
```

```
list[2].avg_sal(list)
print(Employee.emp_count)
```

The screenshot shows a Google Colaboratory notebook with two code cells. The first cell contains Python code for an Employee class and its usage. The second cell contains a NumPy exercise.

```
[30] def __init__(self, name, family, salary, department):
      Employee.__init__(self, name, family, salary, department)

      list = []
      list.append(Employee('John', 'Doe', 30000, 'IT'))
      list.append(Employee('Jane', 'Levy', 45000, 'Sales'))

      list.append(Fulltime_Employee('Jack', 'Sparrow', 23000, 'Design'))
      list.append(Fulltime_Employee('Jared', 'Leto', 50000, 'Gaming'))

      list[0].avg_sal(list)
      list[2].avg_sal(list)
      print(Employee.emp_count)

37000.0
37000.0
4
```

```
# 2. Numpy
# Using NumPy create random vector of size 20 having only float in the range 1-20.
# Then reshape the array to 4 by 5
# Then replace the max in each row by 0 (axis=1)
# (you can NOT implement it via for loop)
```

Here in the first question, created a data member count, Created a constructor to initialize name, family, salary, department and created a function average salary and Created a Fulltime Employee class and it inherited the properties of Employee class and lastly created instances for the classes and called the respective member functions.

```
# 2. Numpy
# Using NumPy create random vector of size 20 having only float in the range 1-20.
# Then reshape the array to 4 by 5
# Then replace the max in each row by 0 (axis=1)
# (you can NOT implement it via for loop)
```

```
import numpy as np
```

```
# Create random vector of size 20 with floats between 1 and 20
vec = np.random.uniform(1, 20, 20)
```

```
# Reshape the vector to 4 by 5
```

```
mat = vec.reshape(4, 5)

# Replace the max in each row by 0
mat[np.arange(4), mat.argmax(axis=1)] = 0

print(mat)
```

The screenshot shows a Google Colab notebook titled "NN_ICP3". The code cell contains the following Python code:

```
# then replace the max in each row by 0 (axis=1)
# (you can NOT implement it via for loop)

import numpy as np

# Create random vector of size 20 with floats between 1 and 20
vec = np.random.uniform(1, 20, 20)

# Reshape the vector to 4 by 5
mat = vec.reshape(4, 5)

# Replace the max in each row by 0
mat[np.arange(4), mat.argmax(axis=1)] = 0

print(mat)
```

The output of the code is a 4x5 matrix of floats, where the maximum value in each row has been replaced by 0:

```
[[ 9.73428305  8.27569143 14.64260182  0.         15.00210218]
 [ 1.37565794  0.         18.06520476 18.11727035 17.22533039]
 [13.68147464 16.31470021  2.05380032  6.91781909  0.         ]
 [13.30430678  8.15609859 11.14383081  0.         3.8990135  ]]
```

The notebook interface shows the code is executed successfully, with a status bar indicating "0s completed at 10:45 PM". The bottom of the image shows a Windows taskbar with the date and time as 10:46 PM on 1/25/2023.

Here in the second question, Using NumPy created a random vector of size 20 having only float in the range 1-20. Then reshaped the array to 4 by 5 Then replaced the max in each row by 0 (axis=1).