https://github.com/HemanthLakkimsetti76/NN_Assignment3/blob/main/NN_Assignment3.ipynb

Neural Network Deep Learning

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.

jupyter NN_Assignment3 Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[12]:
```python
#QUESTION 1
#create a class employee
class Employee:
    #create a datamember to count the no.of employees
    no_of_employees = 0

    #create a constructor to initialize name,family,salary,department
    def __init__(self, name, family_name, salary, department):
        self.__name = name
        self.__family_name = family_name
        self.salary = salary
        self.__department = department
        Employee.no_of_employees += 1


    #create a function to average salary
    def average_salary(employees):
        sum = 0
        for employee in employees:
            sum += employee.salary
        return sum / Employee.no_of_employees


#create a fulltime emlpoyee class and it should inherit the properties of Employee class
class FulltimeEmployee(Employee) :

    def __init__(self, name, family_name, salary, department):
```

jupyter NN_Assignment3 Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

```python
#create a fulltime emlpoyee class and it should inherit the properties of Employee class
class FulltimeEmployee(Employee) :

    def __init__(self, name, family_name, salary, department):
        super().__init__(name, family_name, salary, department)

    def full_time(self):
        print("Full Time Employees")


#create the instances of Fulltime Employee class and employee class and call their member functions
def main():
    employees = []
    fte1 = FulltimeEmployee("Hemanth", "Lakkimsetti", 100000, "CS")
    fte1.full_time()
    employees.append(fte1)
    fte2 = FulltimeEmployee("Abhi", "Varma", 300000, "DS")
    employees.append(fte2)
    emp1 = Employee("Pavan", "Varma", 250000, "CIS")
    employees.append(emp1)
    emp2 = Employee("Sai", "Kumar", 150000, "BD")
    employees.append(emp2)
    print("Average salary:", FulltimeEmployee.average_salary(employees))


if __name__ == "__main__":
    main()
```

```
Full Time Employees
Average salary: 200000.0
```

https://github.com/HemanthLakkimsetti76/NN_Assignment3/blob/main/NN_Assignment3.ipynb

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)

jupyter NN_Assignment3 Last Checkpoint: 7 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[6]:
```python
#QUESTION2
#create a randomm vector of size 20 having only float in the range 1-20
import numpy as np
x=np.random.uniform(1,20,20)
print("Original array:")
print(x)
#reshape the array to 4 by 5
x1=x.reshape(4,5)
print("Reshaped Array: ")
print(x1)
#replace the max in each row by 0
x1[np.arange(len(x1)), x1.argmax(1)]=0
print("Maximum value replaced by 0:")
print(x1)
```

```
Original array:
[17.63588205 13.53197962  7.44530998 14.28574724 13.04476079 19.04042292
 15.18934316  9.39306494 18.35421962 16.2411306   1.94798656 13.54206818
  7.01384139  7.97867988  3.35666172 14.93712166 17.89527816 18.71938475
  8.43438308 15.70833422]
Reshaped Array:
[[17.63588205 13.53197962  7.44530998 14.28574724 13.04476079]
 [19.04042292 15.18934316  9.39306494 18.35421962 16.2411306 ]
 [ 1.94798656 13.54206818  7.01384139  7.97867988  3.35666172]
 [14.93712166 17.89527816 18.71938475  8.43438308 15.70833422]]
Maximum value replaced by 0:
[[ 0.         13.53197962  7.44530998 14.28574724 13.04476079]
 [ 0.         15.18934316  9.39306494 18.35421962 16.2411306 ]
 [ 1.94798656  0.          7.01384139  7.97867988  3.35666172]
 [14.93712166 17.89527816  0.          8.43438308 15.70833422]]
```

[ ]: