

End Practical Examination

Reg No: 39110636

Name: Mohnish Devaraj

Branch: CSE

Semester: IV Sem

Subject Code: SC5A2601

Subject Name: Machine Learning &
Data Analytics

Date of Exam: 26-04-2022

Batch-ID: ML-7

Duration: 3 Hours

No. of pages: 8

① Write python code to find the Least Common Multiple among given two numbers.

Aim:

To write a python code to find the Least Common Multiple among given two numbers

Algorithm:

step 1: Start

step 2: start the function LCM by calling the two variables

step 3: check if the first number is greater than second number, if yes copy the first number to greater, else goto step 3.1

step 3.1: Copy the second number to greater

step 4: Start a while loop for True

step 4.1: Check if greater % a is 0 and greater % b is 0

step 4.2: Copy greater to LCM and break

step 4.3: increment greater by 1

step 5: return LCM

step 6: get the input from the user

step 7: Call the function and print the LCM

step 8: stop

Program:

```
def lcm(a, b):
```

```
    if a > b:
```

```
        greater = a
```

```
    else:
```

```
        greater = b
```

```
    while (True):
```

```
        if ((greater % a == 0) and (greater % b == 0)):
```

```
            lcm = greater
```

```
            break
```

```
        greater += 1
```

```
    return lcm
```

```
a = int(input())
```

```
b = int(input())
```

```
print("The L.C.M. is", lcm(a, b))
```

Output:

The output screen shot is attached below.

Result:

The above program is executed successfully.

② Write a Python code to implement k-Means Clustering Algorithm

Aim:

TO write a python code to implement k-Means Clustering Algorithm

Algorithm:

- step 1: start
- step 2: Select the number k to decide the number of cluster
- step 3: select random k points or centroid
- step 4: Calculate the variance and place a new centroid of each cluster
- step 5: Assign each data point to their closest centroid will assign k cluster
- step 6: Repeat step 3 untill it hears to centroid of cluster
- step 7: if any reassignment occurs, then go to step 4, goto finish
- step 8: stop

Program:

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
class K-MeansAlg:
```

```
    def __init__(self):
```

```
        self.m1 = None
```

```
        self.m2 = None
```

```
        self.c1, self.c2, self.nc1, self.nc2 = [], [], [], []
```

```
        self.pts = {}
```

```
        self.err = 0
```

```
    def error(self):
```

```
        e1 = 0
```

```
        e2 = 0
```

```
        for i in self.c1:
```

```
            e1 += ((i[0] - self.m1[0])**2 + (i[1] - self.m1[1])**2)
```

```
        for i in self.c2:
```

```
            e2 += ((i[0] - self.m2[0])**2 + (i[1] - self.m2[1])**2)
```

```
        return (e1 + e2)
```

```
    def fitting(self, c1, c2, *args):
```

```
        def mean(cl):
```

```
            n = len(cl)
```

```
            a = 0
```

```
            b = 0
```

```
            for i in cl:
```

```
                a += i[0]/n
```

```
                b += i[1]/n
```

```
return [a,b]
```

```
def dist(a,b):
```

```
    return ((a[0]-b[0])**2 + (a[1]-b[1])**2)**(1/2)
```

```
self.c1 = c1
```

```
self.c2 = c2
```

```
- con = 1
```

```
for i in args:
```

```
    self.pts['x'+str(-con)] = i
```

```
    - con += 1
```

```
ep0 = 1
```

```
while (True):
```

```
    self.m1 = mean(self.c2)
```

```
    self.m2 = mean(self.c2)
```

```
    for i in args:
```

```
        p = dist(i, self.m1)
```

```
        q = dist(i, self.m2)
```

```
        if (p > q):
```

```
            self.nc2.append(i)
```

```
        else:
```

```
            self.nc1.append(i)
```

```
    print("After Epoch", ep0, 'Error:', self.error())
```

```
    ep0 += 1
```

```
    if (self.c1 == self.nc1 and self.nc2 == self.nc2):
```

```
        break
```

```
    else:
```

```
        self.c1, self.c2 = self.nc1.copy(), self.nc2.copy()
```

```
        self.nc1, self.nc2 = [], []
```

```
for i, j in sorted(self.pts.item()):
```

```
    for k in range(len(self.c1)):
```

```
        if (j == self.c1[k]):
```

```
            self.c1[k] = 1
```

```
    for l in range(len(self.c2)):
```

```
        if (j == self.c2[l]):
```

```
            self.c2[l] = i
```

```
def display(self):
```

```
    print("Cluster 1: ", self.c1, "\n Cluster 2: ", self.c2)
```

```
x1 = [1, 0]
```

```
x2 = [0, 1]
```

```
x3 = [2, 1]
```

```
x4 = [3, 3]
```

```
c1 = [x1, x3]
```

```
c2 = [x2, x4]
```

```
model = k-mean alg()
```

```
model.fitting(c1, c2, x1, x2, x3, x4)
```

```
print("Data Objects: ")
```

```
for i in sorted(model.pts.items()):
```

```
    print(i)
```

```
model.display()
```

```
p1 = [model.pts[i] for i in model.c1]
```

```
p2 = [model.pts[i] for i in model.c2]
```

```
x = [7
```



```
y = []
```

```
for i in p2:
```

```
    x.append(i[0])
```

```
    y.append(i[1])
```

```
plt.scatter(x, y, c='r', *label* = 'cluster 2')
```

```
plt.legend()
```

```
plt.show()
```

Output:

The output screen shot is attached below

Result:

The above program is executed successfully.

Output Screenshot:

1. Write python code to find the Least Common Multiple among given two numbers.

	Input	Expected	Got	
✓	54 24 The L.C.M. is 216	54 24 The L.C.M. is 216	54 24 The L.C.M. is 216	✓
✓	568 928 The L.C.M. is 65888	568 928 The L.C.M. is 65888	568 928 The L.C.M. is 65888	✓
✓	8 7 The L.C.M. is 56	8 7 The L.C.M. is 56	8 7 The L.C.M. is 56	✓

Passed all tests! ✓

2. Write a python code to implement K-Means Clustering Algorithm

```

After Epoch 1  ERROR: 7.5
After Epoch 2  ERROR: 2.6666666666666667
Data Objects:
('x1', [1, 0])
('x2', [0, 1])
('x3', [2, 1])
('x4', [3, 3])
Cluster 1: ['x1', 'x2', 'x3']
Cluster 2: ['x4']

```

