**Experiment 10**

**Student Name:** Sahil Kaundal **UID:** 21BCS8197

**Branch:** BE CSE (Lateral Entry) **Section/Group:** 616/A

**Semester:** 5th **Date of Performance:** 09/11/2022

**Subject Name:** ML Lab **Subject Code:** 20CSP-317

1. **Aim/Overview of the practical:**

Implement Association Rule Mining.

1. **Task To Be Done:**

Implement Association Rule Mining.

**3. Apparatus / Simulator Used:**

1. Windows 7 or above.
2. Google Collab.

**Apriori Function**

This is the main function of this Apriori Python implementation. It basically follows my modified pseudocode written above.

1. Generate the candidate set by joining the frequent itemset from the previous stage.

2. Perform subset testing and prune the candidate set if there‟s an infrequent itemset contained. 3. Calculate the final frequent itemset by getting those satisfy minimum support.

**Pruning**

To perform subset testing, we loop through all possible subsets in the itemset. If the subset is not in the previous frequent itemset, we prune it.

**4. Program / Commands:**

#Sahil Kaundal

#21BCS8197

data = [

['T100',['I1','I2','I5']],

['T200',['I2','I4']],

['T300',['I2','I3']],

['T400',['I1','I2','I4']],

['T500',['I1','I3']],

['T600',['I2','I3']],

['T700',['I1','I3']],

['T800',['I1','I2','I3','I5']],

['T900',['I1','I2','I3']]

]

init = []

for i in data:

for q in i[1]:

if(q not in init):

init.append(q)

init = sorted(init)

print(init)

sp = 0.4

s = int(sp\*len(init))

s

from collections import Counter

c = Counter()

for i in init:

for d in data:

if(i in d[1]):

c[i]+=1

print("C1:")

for i in c:

print(str([i])+": "+str(c[i]))

print()

l = Counter()

for i in c:

if(c[i] >= s):

l[frozenset([i])]+=c[i]

print("L1:")

for i in l:

print(str(list(i))+": "+str(l[i]))

print()

pl = l

pos = 1

for count in range (2,1000):

nc = set()

temp = list(l)

for i in range(0,len(temp)):

for j in range(i+1,len(temp)):

t = temp[i].union(temp[j])

if(len(t) == count):

nc.add(temp[i].union(temp[j]))

nc = list(nc)

c = Counter()

for i in nc:

c[i] = 0

for q in data:

temp = set(q[1])

if(i.issubset(temp)):

c[i]+=1

print("C"+str(count)+":")

for i in c:

print(str(list(i))+": "+str(c[i]))

print()

l = Counter()

for i in c:

if(c[i] >= s):

l[i]+=c[i]

print("L"+str(count)+":")

for i in l:

print(str(list(i))+": "+str(l[i]))

print()

if(len(l) == 0):

break

pl = l

pos = count

print("Result: ")

print("L"+str(pos)+":")

for i in pl:

print(str(list(i))+": "+str(pl[i]))

print()

from itertools import combinations

for l in pl:

c = [frozenset(q) for q in combinations(l,len(l)-1)]

mmax = 0

for a in c:

b = l-a

ab = l

sab = 0

sa = 0

sb = 0

for q in data:

temp = set(q[1])

if(a.issubset(temp)):

sa+=1

if(b.issubset(temp)):

sb+=1

if(ab.issubset(temp)):

sab+=1

temp = sab/sa\*100

if(temp > mmax):

mmax = temp

temp = sab/sb\*100

if(temp > mmax):

mmax = temp

print(str(list(a))+" -> "+str(list(b))+" = "+str(sab/sa\*100)+"%")

print(str(list(b))+" -> "+str(list(a))+" = "+str(sab/sb\*100)+"%")

curr = 1

print("choosing:", end=' ')

for a in c:

b = l-a

ab = l

sab = 0

sa = 0

sb = 0

for q in data:

temp = set(q[1])

if(a.issubset(temp)):

sa+=1

if(b.issubset(temp)):

sb+=1

if(ab.issubset(temp)):

sab+=1

temp = sab/sa\*100

if(temp == mmax):

print(curr, end = ' ')

curr += 1

temp = sab/sb\*100

if(temp == mmax):

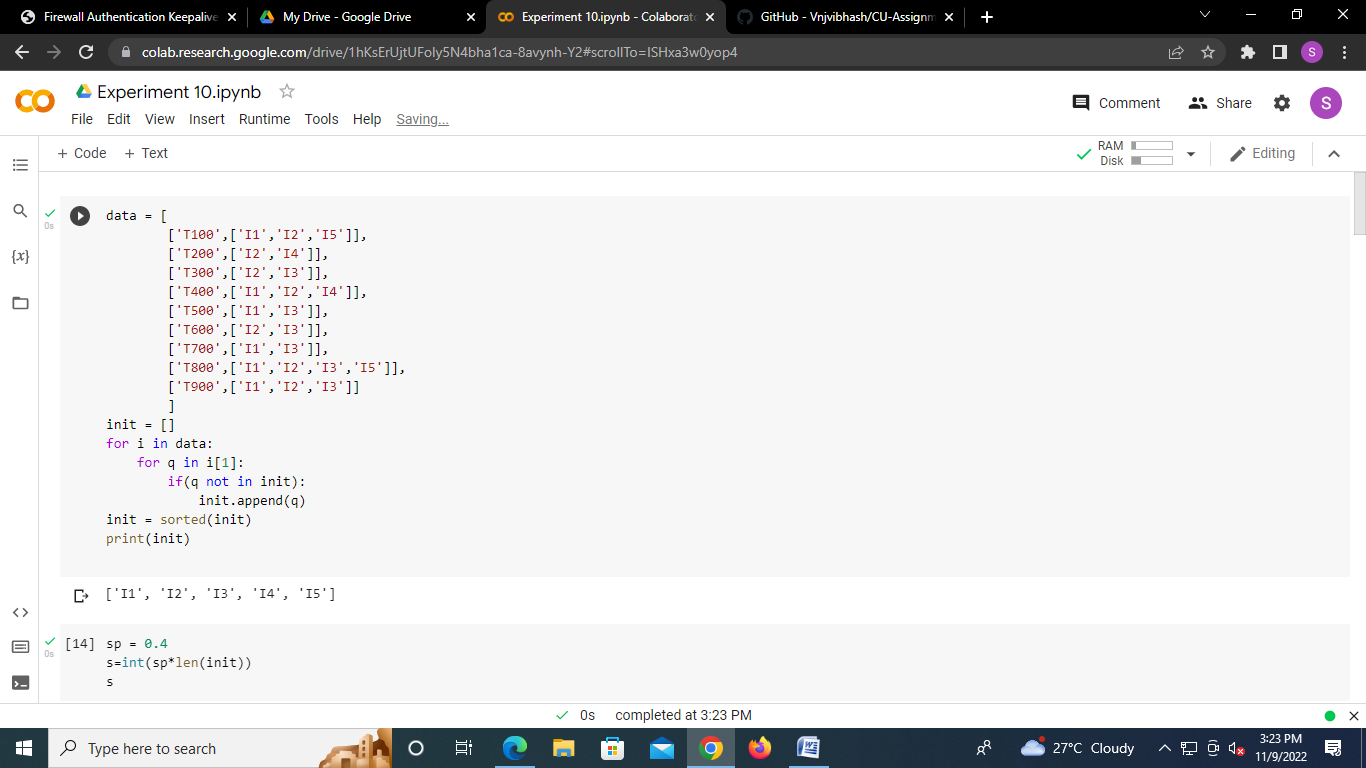
print(curr, end = ' ')

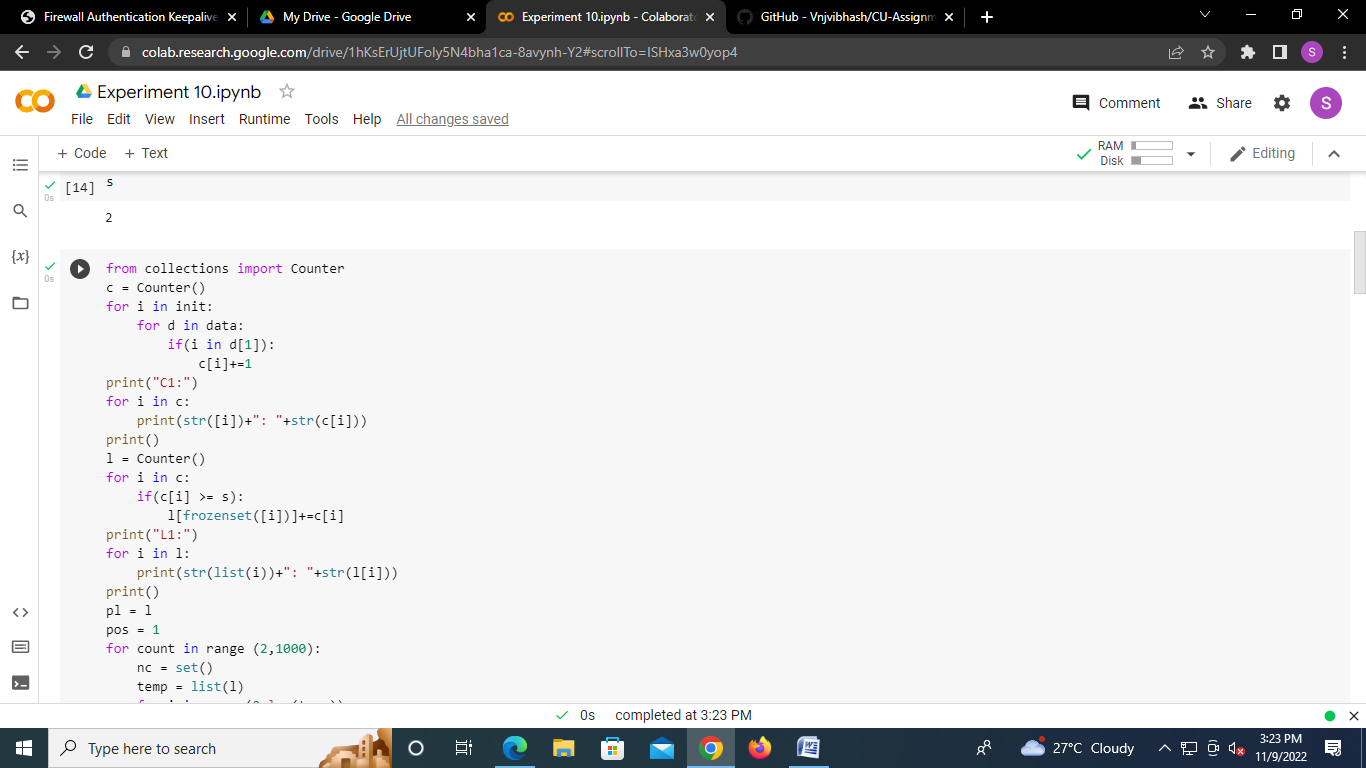
curr += 1

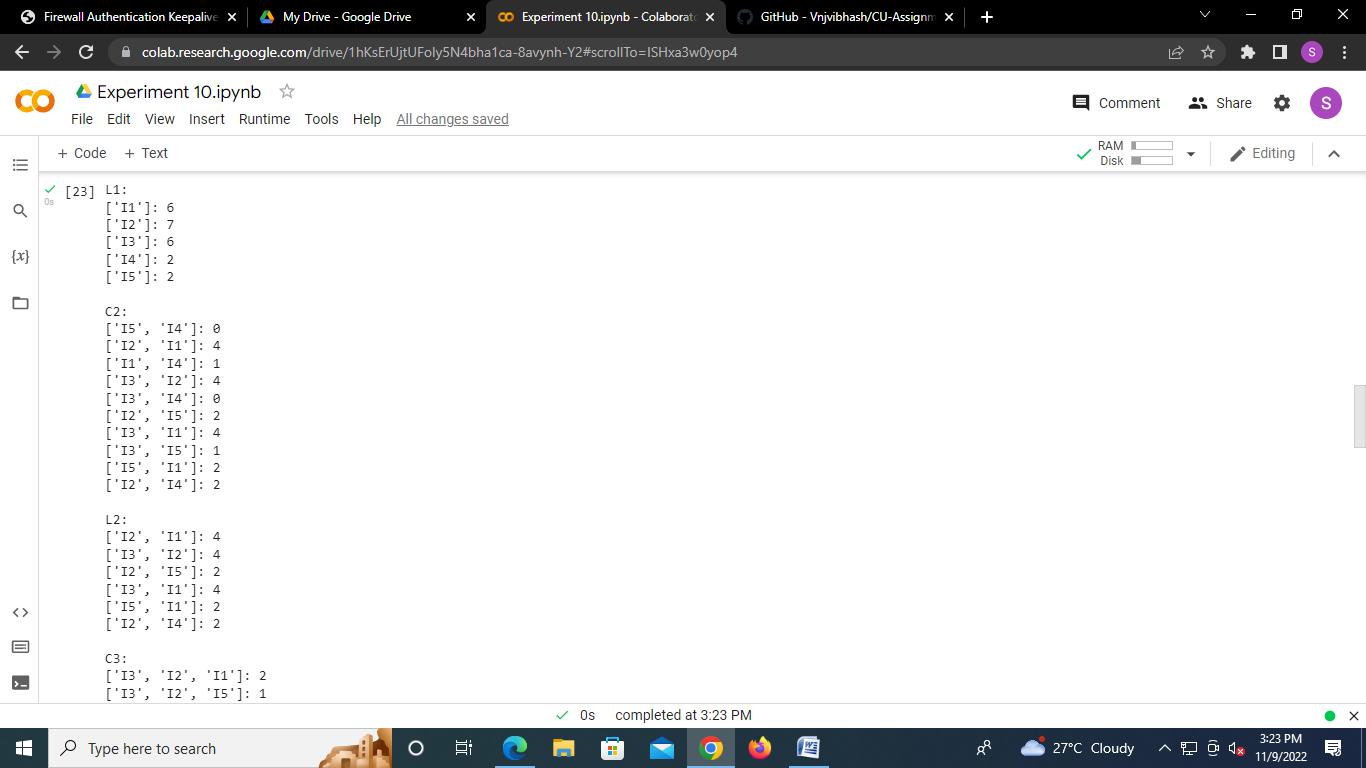
print()

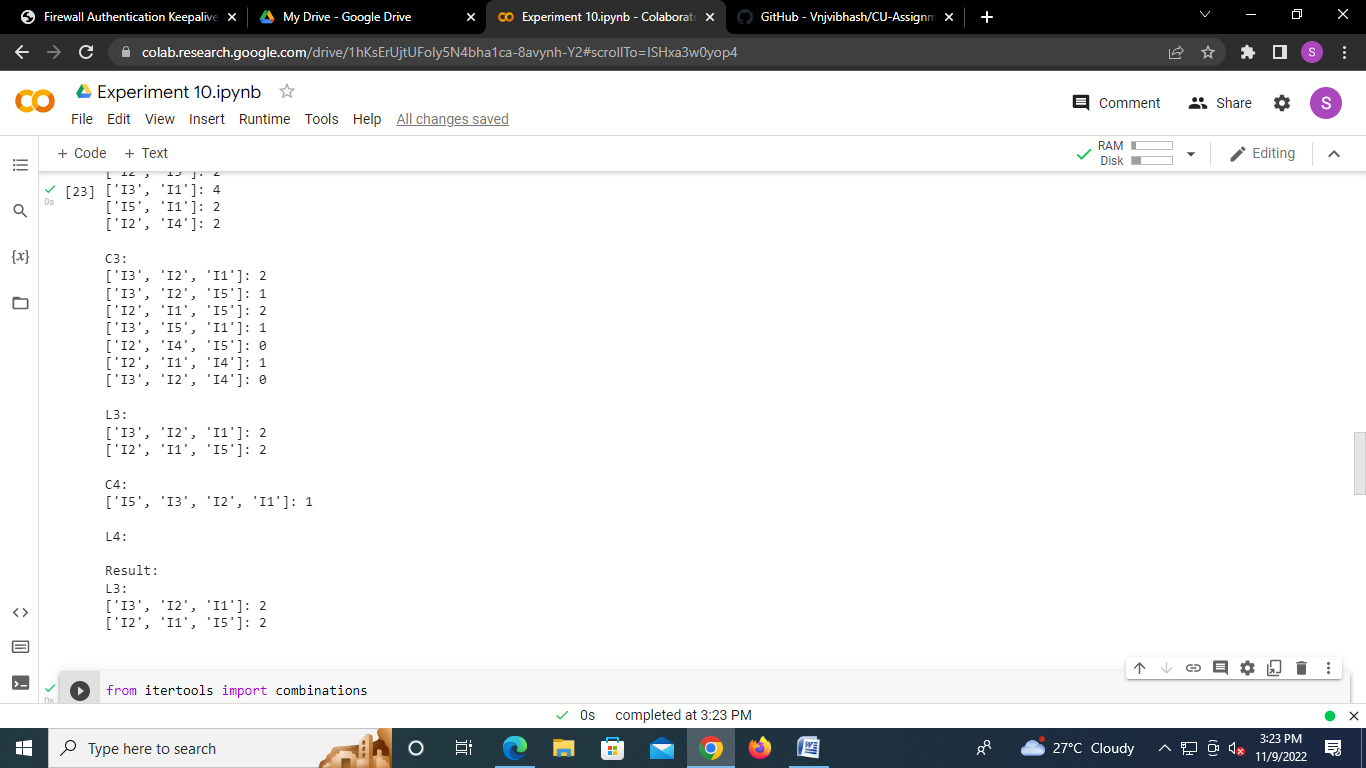
print()

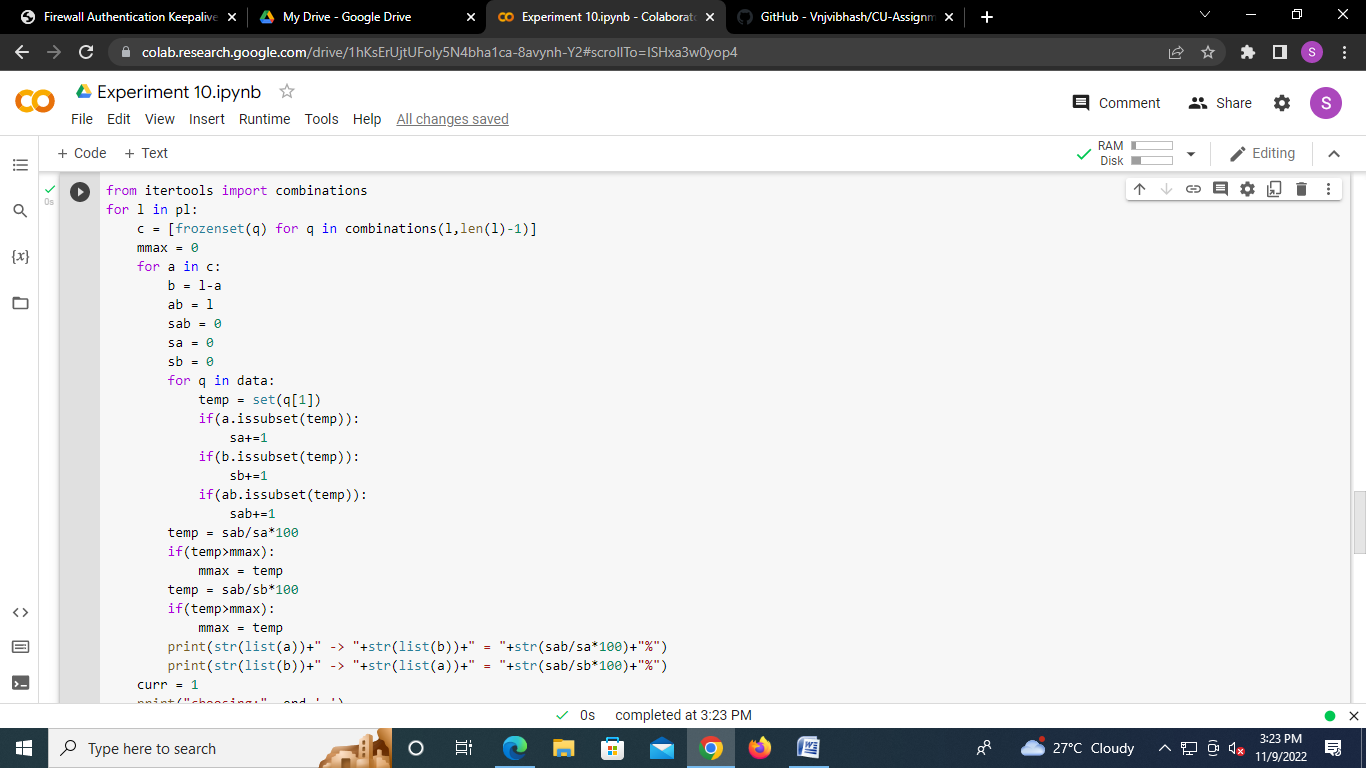
1. **Result/Output/Writing Summary:**

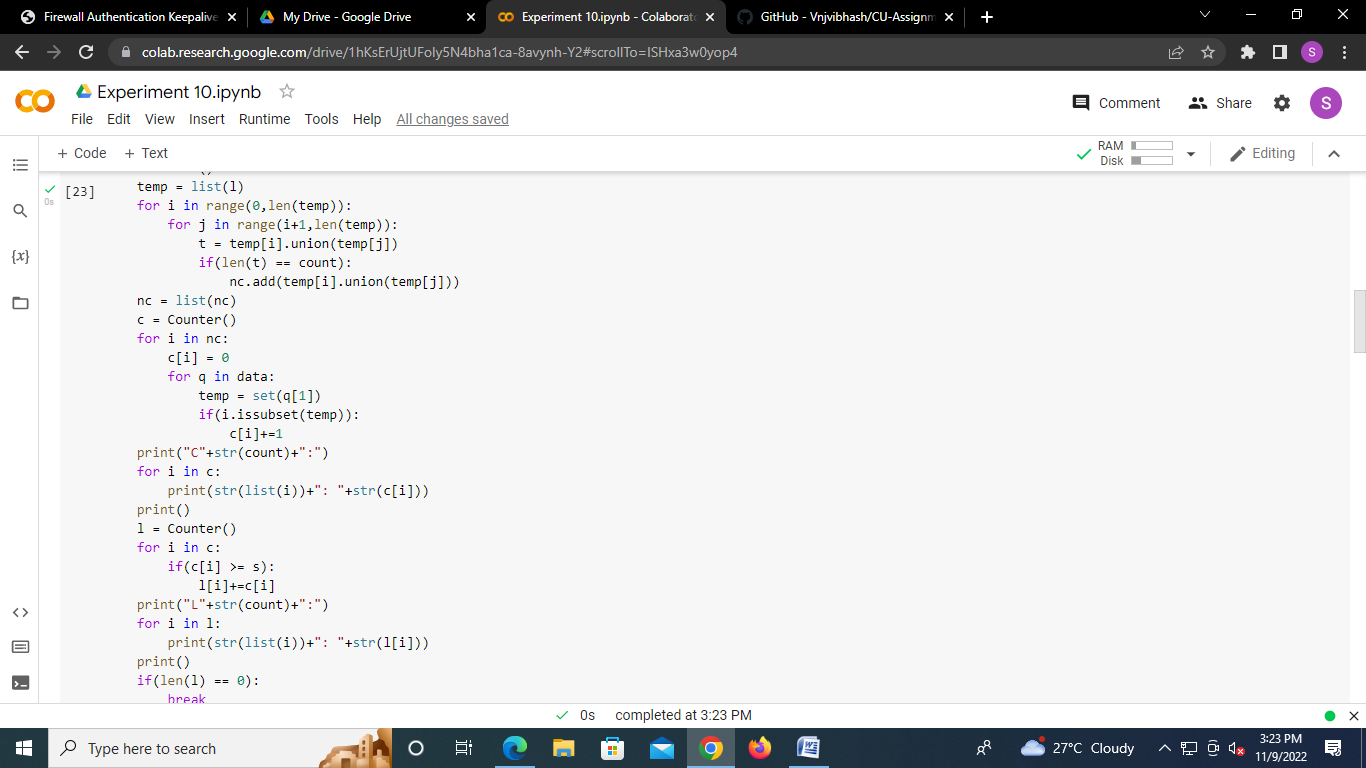


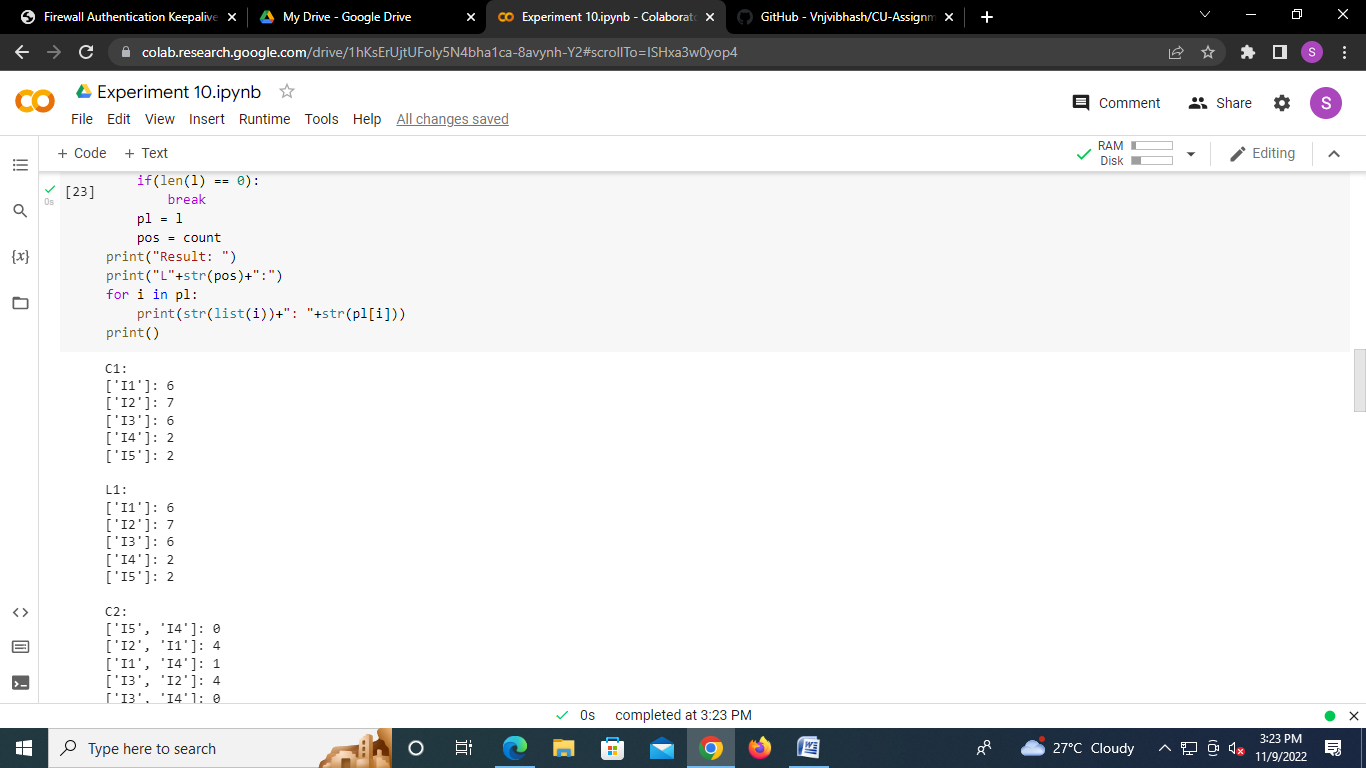


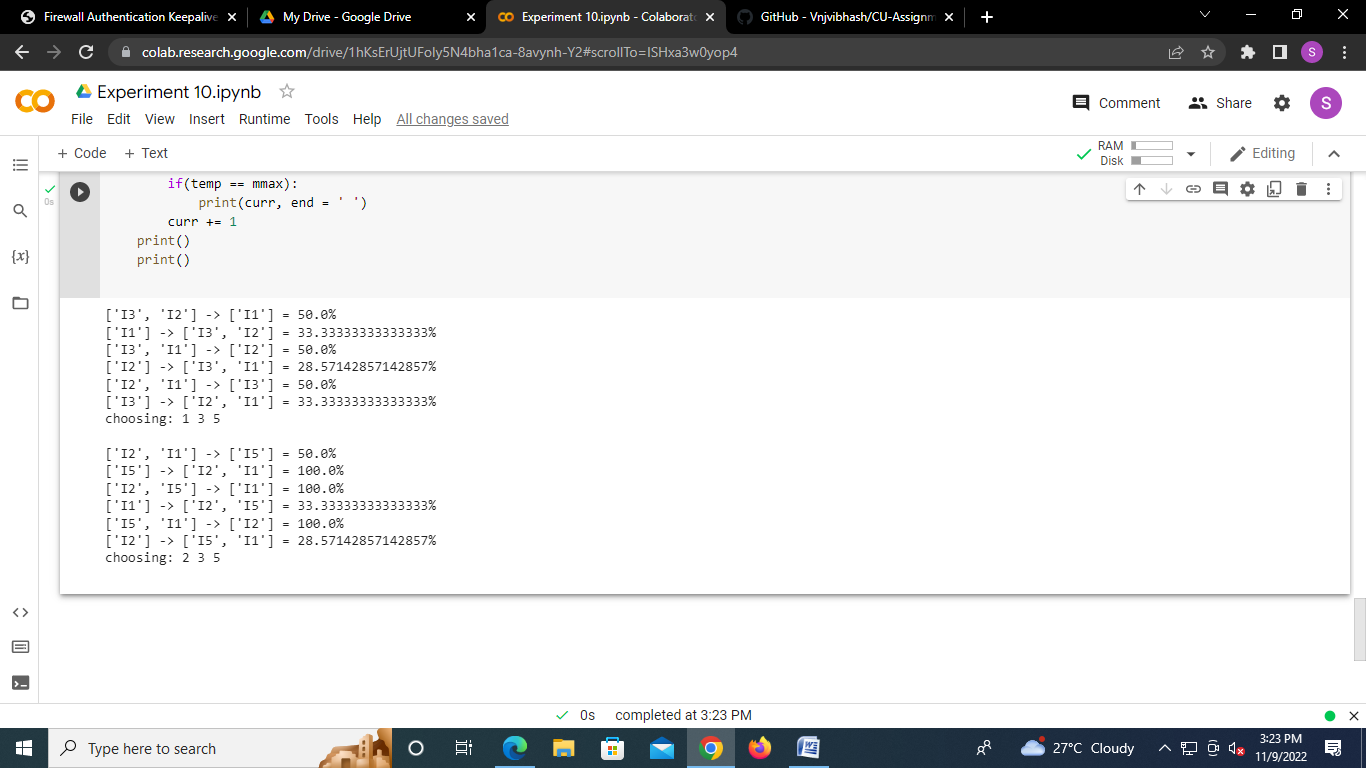


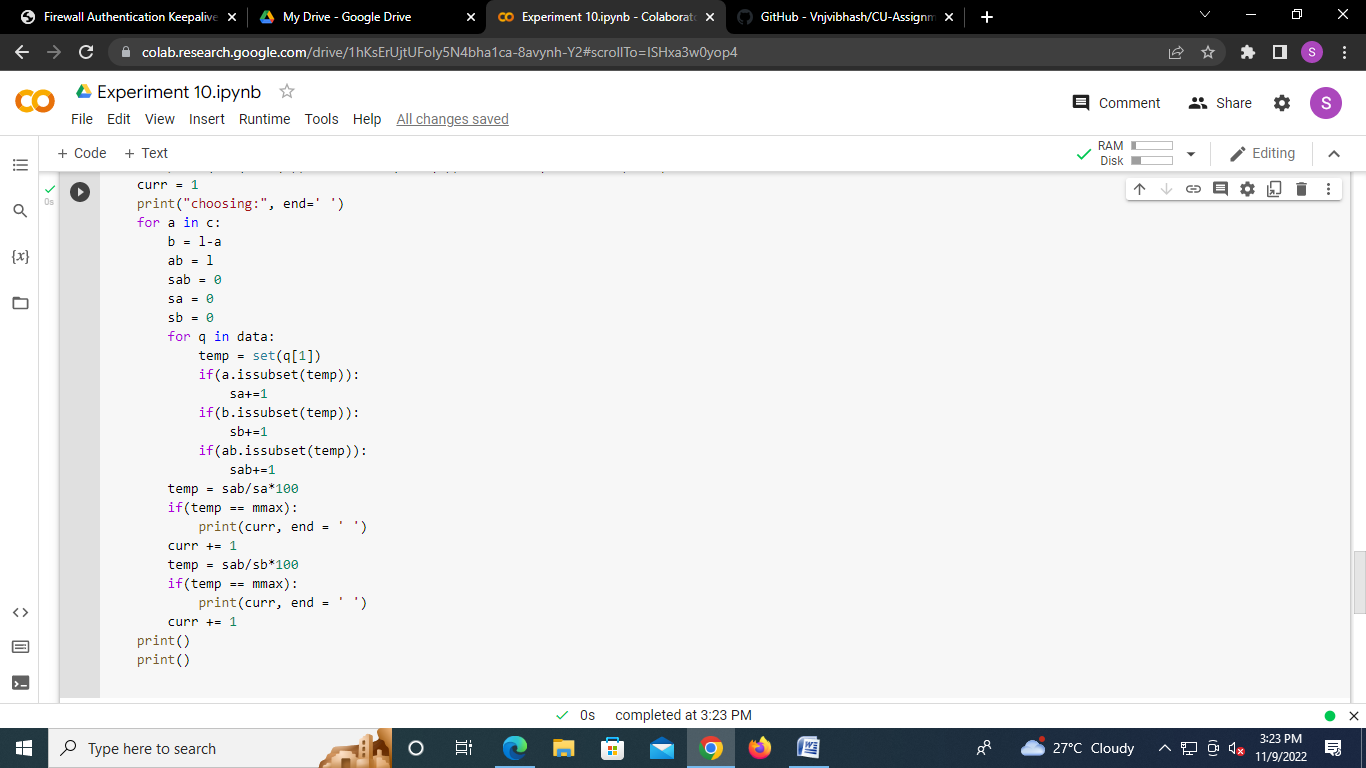












**Learning outcomes (What I have learnt):**

1. Understood the concept of Association Rule Mining.
2. Learnt how to find the Association rule for the Subset.
3. Learnt the concept of Apriori.
4. Completed the Worksheet Successfully.

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. |  |  |  |
| 2. |  |  |  |
| 3. |  |  |  |
|  |  |  |  |