**Apriori Algorithm Program in Python**

**Apriori** is an algorithm for frequent item set mining and association rule learning over the given dataset. It works by identifying the frequent individual items in the dataset and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the dataset.

**Input**

We have the dataset given below. It consists of random items I1, I2, I3, I4, and I5. There can be any number of items. So, first, we’ll find the set of items in our dataset.

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)

['I1', 'I2', 'I3', 'I4', 'I5']

**Support**

Now, we will choose a value for the support. For this example we will choose support to be 40%.

sp = 0.4

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

s

2

**Algorithm**

Now, we will apply the apriori algorithm.

**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()

C1:

['I1']: 6

['I2']: 7

['I3']: 6

['I4']: 2

['I5']: 2

L1:

['I1']: 6

['I2']: 7

['I3']: 6

['I4']: 2

['I5']: 2

C2:

['I2', 'I4']: 2

['I1', 'I3']: 4

['I5', 'I4']: 0

['I5', 'I1']: 2

['I4', 'I3']: 0

['I5', 'I3']: 1

['I2', 'I3']: 4

['I4', 'I1']: 1

['I2', 'I1']: 4

['I5', 'I2']: 2

L2:

['I2', 'I4']: 2

['I1', 'I3']: 4

['I5', 'I1']: 2

['I2', 'I3']: 4

['I2', 'I1']: 4

['I5', 'I2']: 2

C3:

['I5', 'I2', 'I4']: 0

['I2', 'I1', 'I3']: 2

['I5', 'I1', 'I3']: 1

['I2', 'I4', 'I3']: 0

['I5', 'I2', 'I1']: 2

['I1', 'I2', 'I4']: 1

['I5', 'I2', 'I3']: 1

L3:

['I2', 'I1', 'I3']: 2

['I5', 'I2', 'I1']: 2

C4:

['I5', 'I2', 'I1', 'I3']: 1

L4:

Result:

L3:

['I2', 'I1', 'I3']: 2

['I5', 'I2', 'I1']: 2

**Finding the association rules for the subsets**

After running the algorithm, and finding the final subsets, we will find the association rules for the subsets.

**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()

['I2', 'I1'] -> ['I3'] = 50.0%

['I3'] -> ['I2', 'I1'] = 33.33333333333333%

['I2', 'I3'] -> ['I1'] = 50.0%

['I1'] -> ['I2', 'I3'] = 33.33333333333333%

['I1', 'I3'] -> ['I2'] = 50.0%

['I2'] -> ['I1', 'I3'] = 28.57142857142857%

choosing: 1 3 5

['I5', 'I2'] -> ['I1'] = 100.0%

['I1'] -> ['I5', 'I2'] = 33.33333333333333%

['I5', 'I1'] -> ['I2'] = 100.0%

['I2'] -> ['I5', 'I1'] = 28.57142857142857%

['I2', 'I1'] -> ['I5'] = 50.0%

['I5'] -> ['I2', 'I1'] = 100.0%

choosing: 1 3 6