Apriori algorithm

```
def association_rule(I, itemsets_cnt_all, s, confidence, N):
    rules = []
    num_pc = 0
    num_I = itemsets_cnt_all[I]

    for i in range(1, len(I)):
        for A in combinations(fi, i): # A는 튜플, I는 frozenset
            A = frozenset(A)
            num_A = itemsets_cnt_all[A]
            num_diff_IA = itemsets_cnt_all[I-A]

            conf = num_I / num_A

            if conf >= confidence:
                lift = conf * N / num_diff_IA
                if lift > 1:
                    num_pc += 1
                rule = str(set(A)) + ' -> ' + str(set(I-A))
                rules.append(rule)

    return rules, num_pc
```

Association rule을 구해주는 함수를 만들어주었다.

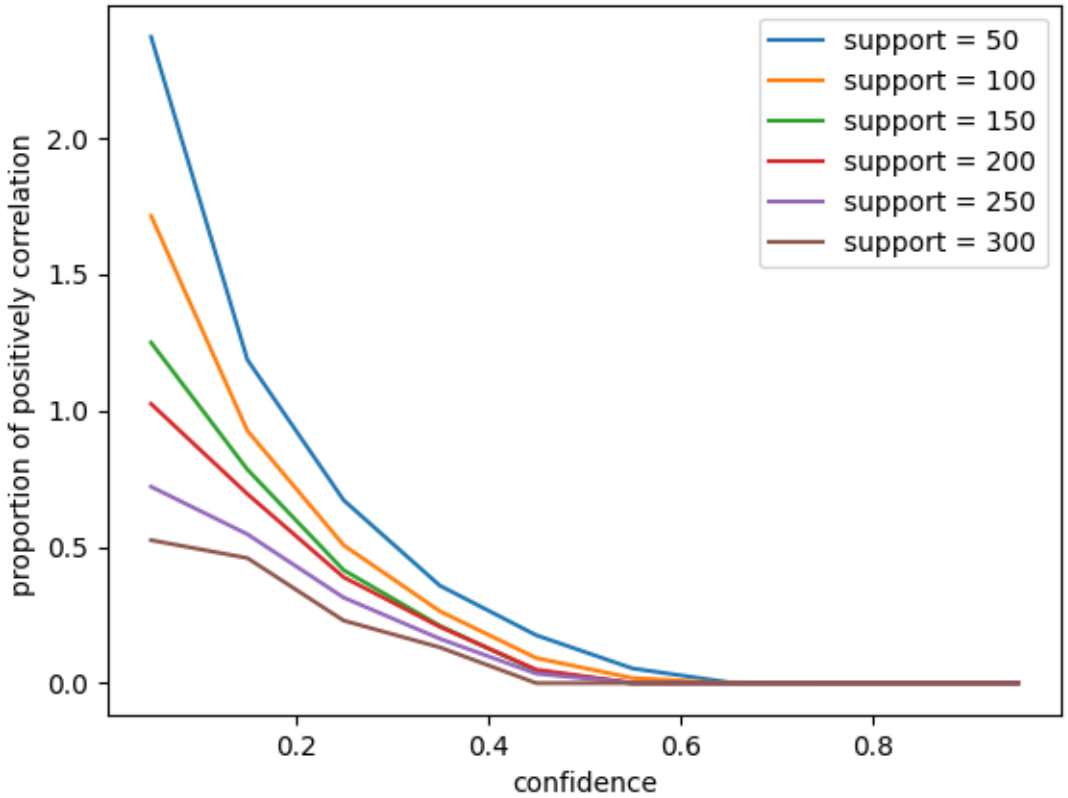
모든 I의 부분집합 A에 대하여 A -> I\A를 rule로 생성하고, 이 rule의 confidence (I가 basket에 등장한 횟수 / A가 basket에 등장한 횟수)가 지정한 confidence 이상인 경우 association rule로 추가해주었다.

$$lift(X \Rightarrow Y) = \frac{N_{X \wedge Y}/N}{(N_X/N)\ (N_Y/N)} = \frac{N_{X \wedge Y}\ N}{N_X\ N_Y}$$

또한, rule을 평가하기 위하여 lift를 구해주었다. Lift를 구하기 위하여 전체 basket의 개수와 I-A가 등장한 횟수를 구해주었다. lift값은 1보다 클 때 positively correlation, 1과 같을 때 independent, 1 미만일 때 negatively correlation으로 1보다 클 때 관계가 의미 있다고 볼 수 있다. 이를 가지고 평가하기 위하여 num_pc 변수를 도입하였다.

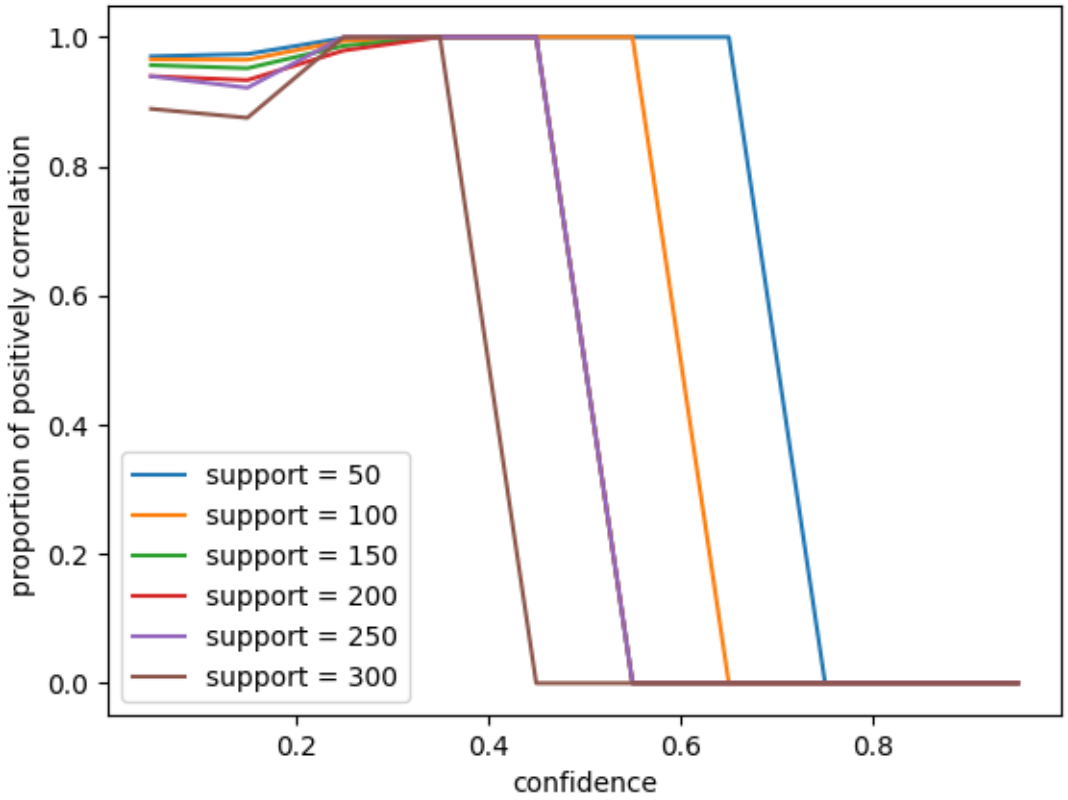
```
support = 50 , confidence = 0.05 , # of rules = 2447 , ["{'berries'} -> {'whole milk'}", "{'specialty chocolate'} -> {'rolls/buns'}", "{'newsp
support = 50 , confidence = 0.15000000000000002 , # of rules = 1220 , ["{'berries'} -> {'whole milk'}", "{'specialty chocolate'} -> {'rolls/bu
support = 50 , confidence = 0.25 , # of rules = 673 , ["{'berries'} -> {'whole milk'}", "{'yogurt', 'whipped/sour cream'} -> {'whole milk'}",
support = 50 , confidence = 0.35 , # of rules = 358 , ["{'berries'} -> {'whole milk'}", "{'yogurt', 'whipped/sour cream'} -> {'whole milk'}",
support = 50 , confidence = 0.44999999999999996 , # of rules = 176 , ["{'yogurt', 'whipped/sour cream'} -> {'whole milk'}", "{'chicken', 'root
support = 50 , confidence = 0.5499999999999999 , # of rules = 54 , ["{'whipped/sour cream', 'tropical fruit'} -> {'whole milk'}", "{'citrus fr
support = 50 , confidence = 0.6499999999999999 , # of rules = 3 , ["{'other vegetables', 'root vegetables', 'pip fruit'} -> {'whole milk'}", '
support = 50 , confidence = 0.7499999999999999 , # of rules = 0 , []
support = 50 , confidence = 0.8499999999999999 , # of rules = 0 , []
support = 50 , confidence = 0.9499999999999998 , # of rules = 0 , []
support = 100 , confidence = 0.05 , # of rules = 579 , ["{'berries'} -> {'whole milk'}", "{'yogurt'} -> {'whipped/sour cream', 'whole milk'}",
support = 100 , confidence = 0.15000000000000002 , # of rules = 313 , ["{'berries'} -> {'whole milk'}", "{'whipped/sour cream'} -> {'yogurt',
support = 100 , confidence = 0.25 , # of rules = 166 , ["{'berries'} -> {'whole milk'}", "{'yogurt', 'whipped/sour cream'} -> {'whole milk'}",
support = 100 , confidence = 0.35 , # of rules = 86 , ["{'berries'} -> {'whole milk'}", "{'yogurt', 'whipped/sour cream'} -> {'whole milk'}",
support = 100 , confidence = 0.44999999999999996 , # of rules = 30 , ["{'yogurt', 'whipped/sour cream'} -> {'whole milk'}", "{'yogurt', 'other
support = 100 , confidence = 0.5499999999999999 , # of rules = 6 , ["{'other vegetables', 'domestic eggs'} -> {'whole milk'}", "{'root vegetab
support = 100 , confidence = 0.6499999999999999 , # of rules = 0 , []
support = 100 , confidence = 0.7499999999999999 , # of rules = 0 , []
support = 100 , confidence = 0.8499999999999999 , # of rules = 0 , []
support = 100 , confidence = 0.9499999999999998 , # of rules = 0 , []
support = 150 , confidence = 0.05 , # of rules = 230 , ["{'yogurt'} -> {'shopping bags'}", "{'shopping bags'} -> {'yogurt'}", "{'citrus fruit'
support = 150 , confidence = 0.15000000000000002 , # of rules = 145 , ["{'shopping bags'} -> {'yogurt'}", "{'citrus fruit'} -> {'whole milk'}'
support = 150 , confidence = 0.25 , # of rules = 74 , ["{'citrus fruit'} -> {'whole milk'}", "{'chicken'} -> {'other vegetables'}", "{'other v
support = 150 , confidence = 0.35 , # of rules = 37 , ["{'citrus fruit'} -> {'whole milk'}", "{'chicken'} -> {'other vegetables'}", "{'other v
support = 150 , confidence = 0.44999999999999996 , # of rules = 8 , ["{'yogurt', 'other vegetables'} -> {'whole milk'}", "{'other vegetables',
support = 150 , confidence = 0.5499999999999999 , # of rules = 0 , []
support = 150 , confidence = 0.6499999999999999 , # of rules = 0 , []
support = 150 , confidence = 0.7499999999999999 , # of rules = 0 , []
support = 150 , confidence = 0.8499999999999999 , # of rules = 0 , []
support = 150 , confidence = 0.9499999999999998 , # of rules = 0 , []
support = 200 , confidence = 0.05 , # of rules = 132 , ["{'yogurt'} -> {'rolls/buns'}", "{'rolls/buns'} -> {'yogurt'}", "{'sausage'} -> {'othe
support = 200 , confidence = 0.15000000000000002 , # of rules = 90 , ["{'yogurt'} -> {'rolls/buns'}", "{'rolls/buns'} -> {'yogurt'}", "{'sausa
support = 200 , confidence = 0.25 , # of rules = 48 , ["{'sausage'} -> {'other vegetables'}", "{'whipped/sour cream'} -> {'other vegetables'}'
support = 200 , confidence = 0.35 , # of rules = 25 , ["{'whipped/sour cream'} -> {'other vegetables'}", "{'citrus fruit'} -> {'whole milk'}",
support = 200 , confidence = 0.44999999999999996 , # of rules = 6 , ["{'domestic eggs'} -> {'whole milk'}", "{'yogurt', 'other vegetables'} ->
```

위의 그림은 support와 confidence 값을 변화를 주며 구한 association rule을 출력한 결과이다.



위의 그래프는 support와 confindece 값을 변화를 주며 구한 positively correlation의 비율이다. y축인

proportion of positively correlation은 freq_itemsets_all의 num_pc 값을 합한 후, freq_itemsets_all의 개수로 나누어 구해주었다. 이 그래프 상의 결과로는 support와 confidence가 낮을수록 좋은 association rule을 구한 것 같지만, 적당히 큰 support와 confidence가 주어졌을 때 좋은 association rule을 구할 수 있다. 따라서 다른 평가 방법을 고려해봐야 할 것 같다.



y축인 proportion of positively correlation을 freq_itemsets_all의 num_pc 값을 합한 후, association rule의 개수로 나누어 구해준 결과이다. 이 그래프를 통하여 적절한 support와 confidence 값을 구하는 것이 맞는 것 같다.