0912

$$\hat{\alpha}_l = \frac{M_{1l}^2(1 - M_{1l})}{M_{2l}^2 - M_{1l}^2} - M_{1l}$$

$$\hat{\beta}_l = \hat{\alpha}_l \frac{(1 - M_{1l})}{M_{1l}}$$

$$M_{1l} = \frac{1}{|\mathcal{L}_l|} \sum_{k \in \mathcal{L}_l} \frac{P_k}{T_k} \quad (2)$$

$$M_{2l} = \frac{1}{|\mathcal{L}_l|} \sum_{k \in \mathcal{L}_l} \frac{P_k\,(P_k - 1)}{T_k(T_k - 1)} \quad (3)$$

Al 분모 부분: M2l - M1l**2 로 수정

Al이 음수가 나오는 이슈

```
!!!!!! Al < 0 !!!!!
len(Pk) 27
Pk:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 714, 0, 0, 0, 0]
Nk:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9727, 0, 0, 0, 0]
Tk:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 10441, 0, 0, 0, 0]
```

```
Pk = 714
Tk = 10441


Al = M1l**2 * (1 - M1l) / (M2l - M1l**2) - M1l
print(Al)
```

```
-509.2443925233444
```

In [328]:
```
M1l = Pk / Tk
print(M1l)
```

```
0.0683842543817642
```

In [329]:
```
M2l = Pk * (Pk - 1) / (Tk) / (Tk - 1)
print(M2l)
```

```
0.004670303963045773
```

글을 작성하거나 AI를 사용하려면 '스페이스' 키를, 명령어를 사용하려면 ' / ' 키를 누르세요.

→ AI이 음수가 나오는 경우 AI과 BI 1, 19로 세팅

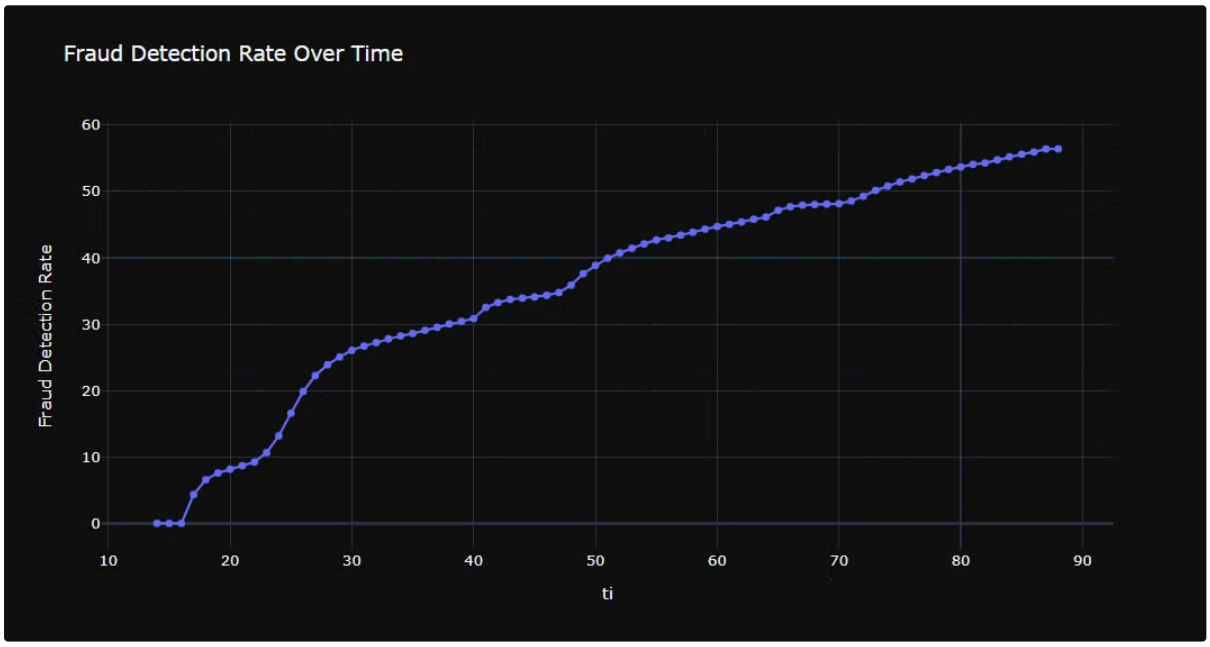
결과

```
print(len(fraud_ip_set) / len(fr_ip_li))
```

```
0.5638826625085275
```

```
total
```

```
[3, 2, 3, 2, 3, 3, 3, 3, 2, 3, 3, 2094, 3, 3, 2, 79467, 4, 3, 3, 11770]
```




```
ti:35
cnt:  14350
len of Pklist  22
len of Pklist  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
38 / 740 , 5.135135135135135%
Al Bl:  45.20574985770612 715.0255362871123
Mll Ml2:  0.05882970250424261 0.003534174071817036
fraud len:  2935
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 740, 0, 0, 0, 0]
lamda:  [0.08137890370348846, 0.07849654547350127, 0.08137890370348846, 0.07849654547350127, 0.08137890370348846, 0.0813789037034884
6, 0.08137890370348846, 0.08137890370348846, 0.07849654547350127, 0.08137890370348846, 0.08137890370348846, 0.08171938927464381, 0.0
8137890370348846, 0.08137890370348846, 0.07849654547350127, 0.13607125261695868, 0.08137890370348846, 0.08137890370348846, 0.0813789
0370348846, 0.10227821037002108]
```

코드

I. 초기화 부분

```
Al = 1
Bl = 19

init_Al = Al
init_Bl = Bl

Pklist = []
for _ in range(20):
    Pklist.append([])

Nklist = []
for _ in range(20):
    Nklist.append([])

Tklist = []
for _ in range(20):
    Tklist.append([])


Pk = [0] * 20
Nk = [0] * 20
Tk = [0] * 20
ak = [Al] * 20
bk = [Bl] * 20
Yk = [0] * 20
lam = [0] * 20

for gr in range(20) :
    lam[gr] = ak[gr] * (1 - gam * stats.f.cdf(tlam, ak[gr] + 1, bk[gr])) / (ak[gr] + bk[gr]) + gam * tlam * (1 - stats.f.cdf(tlam, ak[

print("lamda: ", lam)

gam = 0.5
rat = 0.05

sz = 2

vis = dict()
fraud_ip_set = set()

li = {}

y_values = []

total = [0] * 20
```

II. main

```
for ti in so :
    X = total_df[total_df['time_dh'] == ti]

    unique_groups = X['group'].unique().astype(int)

    cnt = 0
    yk = [0] * 20
    for gr in unique_groups:
        # 해당 그룹의 데이터 추출
        group_data = X[X['group'] == gr]

        # 해당 그룹의 unique한 ip 추출
        unique_ips = group_data['ip'].unique()

        # vis에 없는 ip만 필터링
        new_ips = [ip for ip in unique_ips if ip not in vis]
        li[gr] = new_ips
        Yk[gr] = len(new_ips)
        cnt += len(new_ips)

    sz = min(sz, cnt)
    Nli = [0] * 20
    for j in range(sz) :
        mx = -1
        gr = -1

        for i in li :
            if Yk[i] == 0 :
                continue

            if lam[i] > mx :
                mx = lam[i]
                gr = i

        if gr == -1 :
            continue
```

```
        Nli[gr] += 1
        Yk[gr] -= 1

        tak =ak[gr]
        tbk = bk[gr]
        EB = ak[gr] / (ak[gr] + bk[gr])
        ak[gr] += EB
        bk[gr] += (1 - EB)
        #print("ak[gr] bk[gr]: ", ak[gr], bk[gr])
        tlam = lam[gr]
        lam[gr] = ak[gr] + (1 - gam * stats.f.cdf(tlam, ak[gr] + 1, bk[gr])) / (ak[gr] + bk[gr]) + gam * tlam * (1 - stats.f.cdf(tlam
        if np.isnan(lam[gr]):
            print(f"NaN detected in lam[{gr}]")
            print(f"tak: {tak}")
            print(f"tbk: {tbk}")
            print(f"EB: {EB}")
            print(f"tlam: {tlam}")
            print(f"ak[gr]: {ak[gr]}")
            print(f"bk[gr]: {bk[gr]}")
            print(f"Pk[gr]: {Pk[gr]}")
            print(f"Nk[gr]: {Nk[gr]}")
```

```
    sPk = 0
    sNk = 0
    for gr in range(20) :
        if Nli[gr] == 0 :
            Pklist[gr].append(0)
            Nklist[gr].append(0)
            Tklist[gr].append(0)
            Pk[gr] = get_recent_sum(Pklist, gr)
            Nk[gr] = get_recent_sum(Nklist, gr)
            Tk[gr] = get_recent_sum(Tklist, gr)
            continue

        new_ips = [ip for ip in li[gr] if ip not in vis]
        sam_sz = min(len(new_ips), Nli[gr])
        selected_ips = random.sample(new_ips, sam_sz)

        fraud_ips = set(selected_ips) & set(fr_ip_li)  # selected_ips 중 fraud에 해당하는 IP들

        non_fraud_ips = set(selected_ips) - fraud_ips  # fraud에 해당하지 않는 IP들

        # Pk와 Nk 값 업데이트

        Pklist[gr].append(len(fraud_ips))
        Nklist[gr].append(len(non_fraud_ips))
        Tklist[gr].append(len(selected_ips))

        Pk[gr] = get_recent_sum(Pklist, gr)
        Nk[gr] = get_recent_sum(Nklist, gr)
        Tk[gr] = get_recent_sum(Tklist, gr)

        #print(Pklist, Nklist, Tklist)
        #print(Pk[gr], Nk[gr], Tk[gr])
        #Pk[gr] += len(fraud_ips)
        #Nk[gr] += len(non_fraud_ips)
        #Tk[gr] += len(selected_ips)



        sPk += len(fraud_ips)
        sNk += len(non_fraud_ips)
        # fraud IP들을 fraud_ip_set에 추가
        fraud_ip_set.update(fraud_ips)

        # 선택된 IP들을 모두 vis에 추가
        vis.update({ip: True for ip in selected_ips})
        total[gr] += len(selected_ips)
```

III. 파라미터 업데이트

```
    M1l = 0
    M2l = 0

    cnt1 = 0
    cnt2 = 0
    for gr in range(20) :
        if Tk[gr] == 0 :
            continue
        M1l += Pk[gr] / Tk[gr]
        cnt1 += 1

    if cnt1 > 0 :
        M1l /= cnt1

    for gr in range(20) :
        if Tk[gr] <= 1 :
            continue
        M2l += Pk[gr] * (Pk[gr] - 1) / Tk[gr] / (Tk[gr] - 1)
        cnt2 += 1

    if cnt2 > 0 :
        M2l /= cnt2

    print("M1l M12: ", M1l, M2l)
    print("fraud len: ", len(fraud_ip_set))
    print(Nli)
    print("lamda: ", lam)

    print()
    if M2l - M1l**2 == 0 or M2l == 0:
        Al = init_Al
        Bl = init_Bl
    elif M1l**2 * (1 - M1l) / (M2l - M1l**2) - M1l < 0:
        print("!!!!!! Al < 0 !!!!!")
        print("len(Pk)", len(Pklist[0]))
        print("Pk: ", Pk)
        print("Nk: ", Nk)
        print("Tk: ", Tk)
    else :
        Al = M1l**2 * (1 - M1l) / (M2l - M1l**2) - M1l
        Bl = Al * (1 - M1l) / M1l


    for gr in range(20) :
```