Monte Carlo Course Project

UEFA Champion League Knockout Report

Hangquan Qian

Yuxiang Huang

Jin Dong

5/3/2018

ABSTRACT

The purpose of this project is to predict the probability of draw in the Union of European Football Associations(UEFA). After group matches,The 16 winning sides of each group will draw to get the competitors. And we will use Monte Carlo Method to predicted these probabilities. Actually, there are some rules in the draw, like cannot be same association and same group. We will tell the details in the background. In addition, we will face three potential requirements when we randomly choose the competitors in the drawing, and we called pitfalls. In our model, we will first protect our clubs from the pitfalls, and then use Monte Carlo method to simulate the final result. Thus, each club of winner in each group has the fair and equal probability to meet the other clubs of the runners-up in each group. Furthermore, we will also clearly show pitfalls and simulation methods in this report. To convenient, we create a model that can estimate each years' drawing of 16 rounds. This drawing of 16 round is really important in UEFA, and most media and the gambling companies will also concern about this result. So finally, we will compare our simulation result with the real bracket and odds rate to validation.

BACKGROUND AND ROUND OF 16

The European Champions League is the annual football match sponsored by the European Association of football associations. It represents the highest honor and level of the European club football. It is considered the highest quality, the most influential and the highest level of the club in the world, and is also one of the world's highest bonuses and sports events[1].

The UEFA Champions League is divided into five stages. The first is the preliminaries stage, the second is the qualifying stage, the third is the play-off stage, the fourth is the group stage, and the fifth is the elimination stage. Final stage. In our report, we are focusing on the transition from group stage to elimination stage. The drawing, which is one of the most exciting event in UEFA Champions League.

Initially, we will have 32 teams total divided into 8 teams, each with four teams (teams of the same association will not be assigned to the same group). The game was played in a double loop, with each game winning three points, the draw one point each and the negative zero points. The group's top two qualify for the Champions League knockout, and the team took part in the European Union knockout third. We just stop here to introduce UEFA Champions League. If you are interested, you can see the details about Group rules and UEFA's ranking coefficient in Champions League Format[1].

The round of 16 pairings are determined by means of a draw. The round of 16 is played under the knockout system, on a home-and-away basis (two legs). The UEFA administration ensures that the following principles are respected.

a) Clubs from the same association cannot be drawn against each other.

b) Group winners must be drawn against runners-up from a different group.

e) The runners-up play the first leg at home.

[1] Champions League Format clearly introduce the standard of Qualifying phase, Group stage, Round of 16, Quarter-finals, Semi-finals, Final, Away goals and extra time under the knockout system, Seeding of clubs, and Ties[2].

The team which scores the greater aggregate of goals in the two matches qualifies for the quarter-finals[3]. Below is the result of group matches from 2017-2018 season (see Figure 1).

Group	Winners (seeded in round of 16 draw)	Runners-up (unseeded in round of 16 draw)
A	Manchester United	Basel
B	Paris Saint-Germain	Bayern Munich
C	Roma	Chelsea
D	Barcelona	Juventus
E	Liverpool	Sevilla
F	Manchester City	Shakhtar Donetsk
G	Beşiktaş	Porto
H	Tottenham Hotspur	Real Madrid

Figure 1. Group matches (2017-2018)

Obviously, we have 5 clubs from England, 3 clubs from Spain, 2 from Italy, 1 from France, 2 from Portugal, 1 from Germany, 1 from Turkey and 1 from Ukraine. We should take care of 5 England clubs here because it will influence a lot in our model, and details will be in our Pitfall Chapter. In order to import and compute easily, we create an excel to classify(see Figure 2).

	A	B	C	D
1	A	1	Manchester United	England
2	B	1	Paris Saint-Germain	France
3	C	1	Roma	Italy
4	D	1	Barcelona	Spain
5	E	1	Liverpool	England
6	F	1	Manchester City	England
7	G	1	Besiktas	Turkey
8	H	1	Tottenham Hotspur	England
9	A	2	Basel	Switzerland
10	B	2	Bayern Munich	Germany
11	C	2	Chelsea	England
12	D	2	Juventus	Italy
13	E	2	Sevilla	Spain
14	F	2	Shakhtar Donetsk	Ukraine
15	G	2	Porto	Portugal
16	H	2	Real Madrid	Spain

Figure 2. Group matches in Excel (2017-2018)

The draw began at 13:00 CET on Friday 23 February this year. It took place at the House of European Football in Nyon, Switzerland. It was an open draw with no seeding or country protection. In principle, games would start on 8 March and 15 March at 19:00 CET and 21:05 CET, with the exact schedule confirmed after the draw. The teams drawn first would be at home in the first legs[4].

ASSUMPTION

Here we start to think about model. Since we use Monte Carlo method, we need to ensure that the competitors are randomly selected from our computer language. There is no cheating in fairness. Actually, there exists some people that say we have inside sport every year. Because the drawing is extremely important in the whole UEFA Champions every year. If a championship contender will face a weak team, then their probability to get the Champion finally will be greatly increased. So, the probability should be equal weighted and fair within the rules, and the rules we define here are:

1. Two clubs cannot be same group
2. Two clubs cannot be same country
3. There are maximum 5 clubs from same country, which are 4 (Top 4 Leagues, Premier League and La Liga for etc.) + 1. The criteria are in our Background chapter.

SIMULATION METHOD(ALGORITHM)

1) NAIVE ALGORITHM

When we first encounter this problem, most of us will think in a naïve algorithm. In order to find the matchup probability matrix. We develop the following simulation method.

First, we need to initialize 8X8 probability matrix with all zeros.

Second, in each simulation:

A. First round: Randomly select one team from first ranking group, then randomly select its competitor, and increase the value by 1 in their matching cell.

B. Second round: Randomly select one team in the remaining first ranking pool, then randomly select its competitor in the remaining second ranking pool and increase the value by 1 in their matching cell.

C. Repeat 8 rounds until all teams find their own competitors.

Third, Repeat the second step by n = 100,000 times

Fourth, Pr (i th team in 1^{st} ranking group match with j th team in 2^{nd} ranking group)

= the total value in cell [i, j] / n.

Some of readers who familiar with the rules of UEFA may find that there are some issues in this method. Please remember, the two competitors cannot come from the same association and cannot be from the same group in group matches phrase. During the second step in the above algorithm, we randomly select each team and its competitor, it is very likely to choose the two teams come from the same association or from the same group. If so, we break the rules of UEFA. There are three pitfalls after we consider all the rules. And we will modify our simulation method a little bit considering three pitfalls.

2) PITFALLS

Instead of running simulation for 8 rounds, we notice that for the most of time we do not need eight rounds to get one bracket because of three pitfalls we noticed.

First pitfall is very obvious. After 7 rounds simulations, the remaining two teams will match up automatically. Second and third pitalls are pretty much similar, but from different perspectives. They are both coming from the two rules, clubs from same countries and same groups cannot meet in the knockout phase, of the drawing system.

Under these two rules, there is probability that clubs form 1st ranking group have the only one matching team form 2nd ranking group from round 3 to round 6, which is called pitfall 2. Let's take a simple 5*5 version to explain the extreme situation (see figure 3).

	England 2	England 3	England 4	Spain 2	Germany 2
Turkey 1	X				
Spain 1		X			
Italy 1			X		
England 1	X	X	X	X	√
Germany 1					X

Figure 3. Pitfall 2 sample

If there are three teams from England are set into the 2nd ranking group, with the condition that England 1 has no chance to meet with Spain 2 due to the Not-Same-Group rule, England 1 has no choices but Germany 2 at the very beginning. In this specific example, England 1 and

Germany 2 has already automatically matched up without even once simulation, which means the total rounds of simulation must be less than 7.

Pitfall 3 is from the perspective of 2nd ranking group, which is, but not only, reverse of pitfall 2. The table below is a good examples form this year's match-up (see figure 4).

Rank 2 Rank 1	A Basel (Switzerland)	B Bayern Munich (Germany)	C Chelsea (England)	D Juventus (Italy)	E Sevilla (Spain)	F Shakhtar Donetsk (Ukraine)	G Porto (Portugal)	H Real Madrid (Spain)
A Manchester United (England)	X	0	X	1	0	0	0	0
B Paris Saint-Germain (France)	0	X	X	0	0	0	0	1
C Roma (Italy)	1	0	X	0	0	0	0	0
D Barcelona (Spain)	0	0	X	X	0	0	0	0
E Liverpool (England)	0	0	X	0	X	0	0	0
F Manchester City (England)	0	0	X	0	0	X	0	0
G Besiktas (Turkey)	0	1	X	0	0	0	X	0
H Tottenham Hotspur (England)	0	0	X	0	0	0	0	X

Figure 4. Pitfall 3 sample

Chelsea from England was set into the 2nd ranking group, while all the other clubs from England were in the 1st ranking group. In that case, excluding Roma from the same group, Chelsea can only have three clubs in his opponents poor from the very beginning. If Real Madrid and Bayern Munich from the same group chose two from the three before Chelsea in the first two rounds, Chelsea will be automatically set to meet the remaining one. From the regulations, a

country can have at most 5 clubs getting into the UEFA Champion League, which means a club can have at least 3 choices in the first rounds. That's the reason why we start checking pitfalls from the third rounds.

CORRECT SIMULATION METHOD

Under these circumstances, we can modify our algorithm by adding one more checking step after the random selection from round 3 to round 6. If pitfall 2 or pitfall 3 is satisfied, teams with only one opponent to choose from will be automatically set to meet the only team. However, there are some even rarer situations that multiple teams have to be matched up with the same club. If the event happens, we will break out the loop and undo all the procedures we did in this certain loop and then go ahead for next loop. The following steps are for modified algorithm.

First, we need to initialize 8X8 probability matrix with all zeros.

Second, in each simulation:

A. First round: Randomly select one team from first ranking group, then randomly select its competitor, and increase the value by 1 in their matching cell.

B. Second round: Randomly select one team in the remaining first ranking pool, then randomly select its competitor in the remaining second ranking pool and increase the value by 1 in their matching cell.

C. Repeat round 2 until all teams find their own competitors, in the meanwhile we need to check pitfall 2 and pitfall 3.

Third, Repeat the second step by n = 100,000 times

Fourth, Pr (i th team in 1st ranking group match with j th team in 2nd ranking group)

= the total value in cell [i, j] / n.

RESULTS

We can tell various matches probabilities from the probability table below. There are some cells with the probability of 0, which means the corresponding two teams will never meet in the knockout. With higher probability, the corresponding matches will occur more probably than other low-probability ones. However, it may not be consist to the exact result of the draw. The result is only one trial under the probability matrix. Even so, we can still get some guide from it (see Figure 5).

Rank 2 Rank 1	A Basel (Switzerland)	B Bayern Munich (Germany)	C Chelsea (England)	D Juventus (Italy)	E Sevilla (Spain)	F Shakhtar Donetsk (Ukraine)	G Porto (Portugal)	H Real Madrid (Spain)
A Manchester United (England)	0	0.16577	0	0.163772	0.161622	0.167973	0.16977	0.171105
B Paris Saint-Germain (France)	0.161007	0	0.162547	0.13262	0.131134	0.13721	0.135911	0.139575
C Roma (Italy)	0.185853	0.186221	0	0	0.152342	0.157827	0.156901	0.160851
D Barcelona (Spain)	0.20058	0.197149	0.267468	0	0	0.169027	0.165776	0
E Liverpool (England)	0.152437	0.151729	0	0.241665	0	0.129596	0.129762	0.19481
F Manchester City (England)	0.122058	0.122311	0	0.192922	0.230219	0	0.105322	0.227169
G Besiktas (Turkey)	0.052038	0.051095	0.569984	0.071021	0.079985	0.069399	0	0.106489
H Tottenham Hotspur (England)	0.126038	0.125718	0	0.197999	0.244705	0.168976	0.136558	0

Figure 5. Match-up probability matrix

The tables below are 95% confidence intervals (see Figure 6) and 1% and 99% percentiles for match-up probability (see Figure 7).

Rank 2 Rank 1	A Basel (Switzerland)	B Bayern Munich (Germany)	C Chelsea (England)	D Juventus (Italy)	E Sevilla (Spain)	F Shakhtar Donetsk (Ukraine)	G Porto (Portugal)	H Real Madrid (Spain)
A Manchester United (England)	0	[0.1651,0.1665]	0	[0.1631,0.1644]	[0.1609,0.1623]	[0.1673,0.1687]	[0.1690,0.1705]	[0.1703,0.1719]
B Paris Saint-Germain (France)	[0.1603,0.1617]	0	[0.1618,0.1633]	[0.1319,0.1333]	[0.1305,0.1318]	[0.1365,0.1379]	[0.1352,0.1366]	[0.1388,0.1403]
C Roma (Italy)	[0.1851,0.1866]	[0.1854,0.1870]	0	0	[0.1516,0.1531]	[.1571,0.1586]	[0.1563,0.1575]	[0.1601,0.1616]
D Barcelona (Spain)	[0.1998,0.2014]	[0.1963,0.1980]	[0.2665,0.2685]	0	0	[0.1683,0.1698]	[0.1650,0.1666]	0
E Liverpool (England)	[0.1517,0.1532]	[0.1510,0.1525]	0	[0.2408,0.2425]	0	[0.1289,0.1303]	[0.1291,0.1304]	[0.1940,0.1956]
F Manchester City (England)	[0.1214,0.1227]	[0.1217,0.1229]	0	[0.1921,0.1938]	[0.2293,0.2312]	0	[0.1047,0.1059]	[0.2264,0.2280]
G Besiktas (Turkey)	[0.0516,0.0525]	[0.0507.0.0515]	[0.5688,0.5712]	[0.0705,0.0716]	[0.0794,0.0806]	[0.0688,0.0700]	0	[0.1059,0.1071]
H Tottenham Hotspur (England)	[0.1253.0.1267]	[0.1250,0.1264]	0	[0.1972.0.1988]	[0.2440,0.2455]	[0.1682,0.1698]	[0.1359,0.1373]	0

Figure 6. Confidence interval of match-up probability

Rank2 Rank1	A Basel (Switzerland)		B Bayern Munich (Germany)		C Chelsea (England)		D Juventus (Italy)		E Sevilla (Spain)		F Shakhtar Donetsk (Ukraine)		G Porto (Portugal)		H Real Madrid (Spain)	
	1%	99%	1%	99%	1%	99%	1%	99%	1%	99%	1%	99%	1%	99%	1%	99%
A Manchester United (England)	0	0	0.1587	0.1794	0	0	0.151	0.1729	0.1539	0.1679	0.1593	0.1758	0.1627	0.1811	0.1606	0.1827
B Paris Saint-Germain (France)	0.1519	0.1695	0	0	0.1541	0.1753	0.1244	0.1407	0.1243	0.1394	0.1288	0.1457	0.1282	0.1448	0.1305	0.1474
C Roma (Italy)	0.1766	0.1985	0.1735	0.1987	0	0	0	0	0.1421	0.1627	0.1472	0.1677	0.1489	0.1663	0.1512	0.1712
D Barcelona (Spain)	0.191	0.2095	0.1854	0.2098	0.2571	0.2815	0	0	0	0	0.1618	0.1782	0.1561	0.1767	0	0
E Liverpool (England)	0.1406	0.1642	0.1398	0.1589	0	0	0.2313	0.2512	0	0	0.1193	0.1376	0.1217	0.1373	0.187	0.2048
F Manchester City (England)	0.1149	0.1283	0.1138	0.1314	0	0	0.1845	0.2038	0.2191	0.2406	0	0	0.098	0.1147	0.214	0.237
G Besiktas (Turkey)	0.0474	0.0577	0.0465	0.0556	0.5541	0.5868	0.0643	0.0769	0.0731	0.0872	0.0621	0.0769	0	0	0.0984	0.1164
H Tottenham Hotspur (England)	0.1191	0.1328	0.1146	0.1348	0	0	0.1895	0.2076	0.2336	0.258	0.161	0.1807	0.1262	0.1461	0	0

Figure 7. 1% and 99% percentile of matchup probability

We can tell from 95% confidence interval table and percentile table that the probability for each matchup have fairly small range, which implies small volatility.

VALIDATION

Yellow cells are the matchup in the reality. Though it's not always the highest probability in its row or column, it's still relatively high in each club's selection pool. Since, for the most of time, the highest probability in the row does not match the highest probability in the column, we need to keep a balance between rows and columns which can explain this result well (see Figure 8).

Rank 2 Rank 1	A Basel (Switzerland)	B Bayern Munich (Germany)	C Chelsea (England)	D Juventus (Italy)	E Sevilla (Spain)	F Shakhtar Donetsk (Ukraine)	G Porto (Portugal)	H Real Madrid (Spain)
A Manchester United (England)	0	0.16577	0	0.163772	0.161622	0.167973	0.16977	0.171105
B Paris Saint- Germain (France)	0.161007	0	0.162547	0.13262	0.131134	0.13721	0.135911	0.140105
C Roma (Italy)	0.185853	0.186221	0	0	0.152342	0.157827	0.156901	0.160851
D Barcelona (Spain)	0.20058	0.197149	0.267468	0	0	0.169027	0.165776	0
E Liverpool (England)	0.152437	0.151729	0	0.241665	0	0.129596	0.129762	0.19481
F Manchester City (England)	0.122058	0.122311	0	0.192922	0.230219	0	0.105322	0.227169
G Besiktas (Turkey)	0.052038	0.051095	0.569984	0.071021	0.079985	0.069399	0	0.106489
H Tottenham Hotspur (England)	0.126038	0.125718	0	0.197999	0.244705	0.168976	0.136558	0

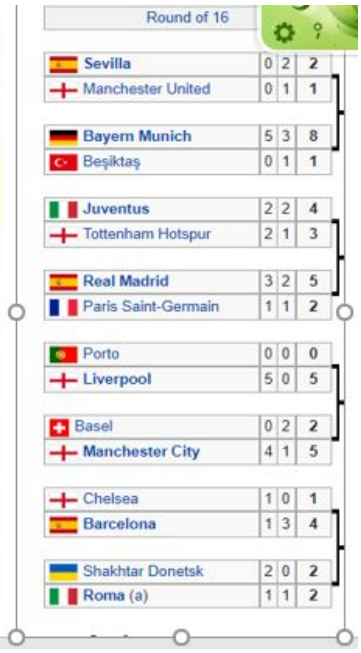

Figure 8. Validation of match-up

If we focus on the odd rate, which is the money you can get back if you bet one dollar on a certain club to win the championship, we can try to explain these rate with our probability matrix. Take Chelsea, who has the lowest odds rate, as an example. They had little chance to win

the championship from the odds rate. This result is not only because of his strength, but also the matchup. We can tell from the probability matrix that Chelsea have high probability to fight against Barcelona or Paris Saint-Germain who are now ranking top two from the odds rate. So our probability can also be a good guide for people who enjoy gambling (see Figure 9).

Bayern Munich	1/5.5
Paris Saint-Germain	1/6
Barcelona	1/8.5
Real Madrid	1/9
Liverpool	1/15
Manchester United	1/15
Juventus	1/17
Tottenham Hotspur	1/28
Chelsea	1/34

Figure 9. Odd rate(2017-2018)

Finally, we can use our model do validation using the data from last year. The yellow cells,which mean the actual bracket from last year, have even higher probability than this year. In that case, we can argue that our Monte Carlo Method does give us a fairly good result for knockout phase (see Figure 10).

	Paris Saint-Germain	Benfica	Manchester City	Bayern Munich	Bayer Leverkusen	Real Madrid	Porto	Sevilla
Arsenal	0.00%	16.85%	0.00%	16.48%	16.55%	16.34%	16.75%	17.03%
Napoli	16.35%	0.00%	16.46%	13.35%	13.28%	12.31%	13.72%	13.85%
Barcelona	22.78%	22.62%	0.00%	18.10%	17.99%	0.00%	18.51%	0.00%
Atletico Madrid	19.87%	19.40%	28.29%	0.00%	16.13%	0.00%	16.31%	0.00%
Monaco	0.00%	12.39%	16.50%	16.45%	0.00%	22.46%	10.50%	21.71%
Borussia Dortmund	21.38%	14.58%	22.91%	0.00%	0.00%	0.00%	13.76%	27.37%
Leicester City	10.35%	7.46%	0.00%	18.37%	18.50%	25.28%	0.00%	20.04%
Juventus	9.27%	6.70%	15.84%	17.26%	17.56%	22.97%	10.40%	0.00%

Figure 10. Match-up result (2016 - 2017)

Reference

[1] 1998-2018 UEFA. “All You Need to Know: Europa League Round of 16 Draw.” *UEFA.com*,

[2] 22 Feb. 2018, www.uefa.com/uefaeuropaleague/news/newsid=2538628.html.

[3] “2017–18 UEFA Champions League Knockout Phase.” *Wikipedia*, Wikimedia Foundation, 4 May 2018, en.wikipedia.org/wiki/2017–18_UEFA_Champions_League_knockout_phase.

[4] “Champions League Format.” *Champions League Format - FootballSeeding.com*, FootballSeeding, www.footballseeding.com/details/champions-league-format2/.

“UEFA.” *Wikipedia*, Wikimedia Foundation, 3 May 2018, en.wikipedia.org/wiki/UEFA.

Appendix (Code)

```
import numpy as np
import csv
import pandas as pd

# Input value
my_data = pd.read_excel('UEFA_Champions_League.xlsx',header=None)

print('[Inital Value]\n')
print(my_data,'\n')
# print('\n',my_data.iloc[0,2])

# Initial Value
Group1 = np.array([[my_data.iloc[0,2],my_data.iloc[0,1],my_data.iloc[0,3]],
                   [my_data.iloc[1,2],my_data.iloc[1,1],my_data.iloc[1,3]],
                   [my_data.iloc[2,2],my_data.iloc[2,1],my_data.iloc[2,3]],
                   [my_data.iloc[3,2],my_data.iloc[3,1],my_data.iloc[3,3]],
                   [my_data.iloc[4,2],my_data.iloc[4,1],my_data.iloc[4,3]],
                   [my_data.iloc[5,2],my_data.iloc[5,1],my_data.iloc[5,3]],
                   [my_data.iloc[6,2],my_data.iloc[6,1],my_data.iloc[6,3]],
                   [my_data.iloc[7,2],my_data.iloc[7,1],my_data.iloc[7,3]],
                   ])

Group2 = np.array([[my_data.iloc[8,2],my_data.iloc[8,1],my_data.iloc[8,3]],
                   [my_data.iloc[9,2],my_data.iloc[9,1],my_data.iloc[9,3]],
                   [my_data.iloc[10,2],my_data.iloc[10,1],my_data.iloc[10,3]],
```

```
            [my_data.iloc[11,2],my_data.iloc[11,1],my_data.iloc[11,3]],
            [my_data.iloc[12,2],my_data.iloc[12,1],my_data.iloc[12,3]],
            [my_data.iloc[13,2],my_data.iloc[13,1],my_data.iloc[13,3]],
            [my_data.iloc[14,2],my_data.iloc[14,1],my_data.iloc[14,3]],
            [my_data.iloc[15,2],my_data.iloc[15,1],my_data.iloc[15,3]],
            ])

Table_initial = np.zeros(64).reshape(8,8)
# You can change value as below:
# Table_initial[0,1] =1
print('[INITAL TABLE]\n')
print(Table_initial)

print('\n[Group1]\n')
print(Group1)
print('\n[Group2]\n')
print(Group2)

# We check traps as below
# print(Group1[0,2]==Group2[2,2])
def trap2(list_input,keep_list):
    '''Club of Group1 that Group2 can choose '''
    global Group1, Group2
    in_trap2 = 0
    choose_1 = 99
    choose_2 = 99
    count = 0
    for i in keep_list:
        if len(list_input[i]) <= 1:
```

```
            in_trap2 = 1
            choose_1 = i
            choose_2 = list_input[i][0]
            count += 1
    if count >=2:
        in_trap2 = 3
        choose_1 = 99
        choose_2 = 99
    # We will return booln that if it is in trap or not(means all len of list1 is larger than 1 or not)
    return in_trap2, choose_1, choose_2

def trap3(list_input,left_list):
    '''Club of Group2 that Group1 can choose '''
    global Group1,Group2
    # for dongda and yuxiang
    in_trap3 = 0
    choose_1 = 99
    choose_2 = 99
    count = 0
    for i in left_list:
        if len(list_input[i]) <= 1:
            in_trap3 = 1
            choose_1 = list_input[i][0]
            choose_2 = i
            count += 1
    if count >=2:
        in_trap2 = 3
        choose_1 = 99
        choose_2 = 99
```

```
    # We will return booln that if it is in trap or not(means all len of list2 is larger than 1 or not)
    return in_trap3, choose_1, choose_2

# We run 10 times
N1 = 100000
# Deep copy that I will not copy the address
# The copy method makes a complete copy of the array and its data.
TABLE = Table_initial.copy()
wrong = 0
for i in range(N1):
    print('\n[ Round',i,']:')
    # Initial Table at first
    Table_bug = np.zeros(64).reshape(8,8)
    # Initial List1 for group1 and group2, ones they are in same country, corresponding
coordinate just remove it
    # Two ways, one is below, other is use np.ones.reshape and become to 0 one in trap
    List1 = [[1,2,3,4,5,6,7],
             [0,2,3,4,5,6,7],
             [0,1,3,4,5,6,7],
             [0,1,2,4,5,6,7],
             [0,1,2,3,5,6,7],
             [0,1,2,3,4,6,7],
             [0,1,2,3,4,5,7],
             [0,1,2,3,4,5,6]]

    List2 = [[1,2,3,4,5,6,7],
             [0,2,3,4,5,6,7],
             [0,1,3,4,5,6,7],
             [0,1,2,4,5,6,7],
```

```
        [0,1,2,3,5,6,7],
        [0,1,2,3,4,6,7],
        [0,1,2,3,4,5,7],
        [0,1,2,3,4,5,6]]

# First DELETE all elements that are in same country
# List 1
for j in range(8):
    for k in range(8):
        if Group1[j,2]==Group2[k,2]:
            List1[j].remove(k)
# List 2
for j in range(8):
    for k in range(8):
        if Group2[j,2]==Group1[k,2]:
            List2[j].remove(k)
# This is a indicator for checking Group2 leave which didnt chosed
Group_leave = [0,1,2,3,4,5,6,7]
# This is a indicator for checking Group1 leave which didnt chosed
Group_keep = [0,1,2,3,4,5,6,7]
# print('\n',List1,'\n',List2)
# For Group1 1st and 2nd choose

for j in range(2):
    # So for here, we choose randomly and ignore the trap2 and trap3
    random_choose = np.random.randint(len(List1[j]))
    TABLE[j][List1[j][random_choose]] += 1
    Table_bug[j][List1[j][random_choose]] += 1
    remove_club = List1[j][random_choose]
```

```
    # print(remove_club)
    Group_leave.remove(remove_club)
    Group_keep.remove(j)



    # print('Group_keep:',Group_keep,'\n')
    # print('Group_leave:',Group_leave,'\n')
    # print(Group_leave)
    # remove all elements in list 1 and list 2 that has just been chosed
    for l in range(8):
        if remove_club in List1[l]:
            List1[l].remove(remove_club)
        if j in List2[l]:
            List2[l].remove(j)

    # print(j,remove_club)
    # print('\n',List1,'\n',List2)
    # print('\n',TABLE)
for j in range(2,8):
    index2 = trap2(List1,Group_keep)
    index3 = trap3(List2,Group_leave)
    if (index2[0]==3):
        TABLE -= Table_bug
        wrong += 1
        break
    elif (index2[0]==1):
        TABLE[index2[1]][index2[2]] += 1
        Table_bug[index2[1]][index2[2]] +=1
        Group_leave.remove(index2[2])
```

```
    Group_keep.remove(index2[1])
    # print('Group_keep:',Group_keep,'\n')
    # print('Group_leave:',Group_leave,'\n')
    # remove all elements in list 1 and list 2 that has just been chosed
    for l in range(8):
        if  index2[2] in List1[l]:
            List1[l].remove(index2[2])
        if index2[1] in List2[l]:
            List2[l].remove(index2[1])
    # print(index2[1],index2[2])
    # print('\n',List1,'\n',List2)
    # print('\n',TABLE)
elif(index3[0]==3):
    TABLE -= Table_bug
    wrong += 1
    break
elif(index3[0]==1):
    TABLE[index3[1]][index3[2]] += 1
    Table_bug[index3[1]][index3[2]] += 1
    Group_leave.remove(index3[2])
    Group_keep.remove(index3[1])
    # print('Group_keep:',Group_keep,'\n')
    # print('Group_leave:',Group_leave,'\n')
    # remove all elements in list 1 and list 2 that has just been chosed
    for l in range(8):
        if index3[2] in List1[l]:
            List1[l].remove(index3[2])
        if index3[1] in List2[l]:
            List2[l].remove(index3[1])
```

```
        # print(index3[1],index3[2])
        # print('\n',List1,'\n',List2)
        # print('\n',TABLE)

    else:
        # So for here, we choose randomly
        # we remove the first element in Gruop 1 left
        left_else = Group_keep[0]
        random_choose = np.random.randint(len(List1[left_else]))
        TABLE[left_else][List1[left_else][random_choose]] += 1
        Table_bug[left_else][List1[left_else][random_choose]] += 1
        remove_club = List1[left_else][random_choose]



        Group_leave.remove(remove_club)
        Group_keep.remove(left_else)
        # print('Group_keep:',Group_keep,'\n')
        # print('Group_leave:',Group_leave,'\n')
        # remove all elements in list 1 and list 2 that has just been chosen
        for l in range(8):
            if remove_club in List1[l]:
                List1[l].remove(remove_club)
            if left_else in List2[l]:
                List2[l].remove(left_else)
        # print(j,remove_club,'\n')
        # print('\n',List1,'\n',List2)
        # print('\n',TABLE)

#     print('\n',remove_club,List1[0])
```

```
    # print('\n',List1,'\n',List2)

    # print('\n',TABLE)


result = TABLE/(N1-wrong)

print('\n',result)

np.savetxt('2017-2018Result.csv',result, delimiter = ',')
```