Basketball analysis

July 18, 2018

```
In [1]: import pandas as pd
        import numpy as np
        import glob
```

```
In [2]: home_dir = '/Users/zehaiwang/Downloads/2018NBA_Hackathon_Files/Basketball Analytics/'
```

```
In [3]: file_list = glob.glob(home_dir+'*.txt')
        print ([x.split('/')[-1] for x in file_list])
```

```
['NBA Hackathon - Play by Play Data Sample (50 Games).txt', 'NBA Hackathon - Event Codes.txt', '
```

```
In [4]: pbyp_df = pd.read_csv(file_list[0],sep='\t')
        eventcode_df = pd.read_csv(file_list[1], sep = '\t')
        game_df = pd.read_csv(file_list[2],sep='\t')
```

0.1 Event Code gives mapping from Event_Msg_type and Action_Type to their decription

_ free throw and No shot_ share the same code

```
In [5]: event_map = eventcode_df[['Event_Msg_Type','Event_Msg_Type_Description']].drop_duplicate
        print (event_map)
```

```
     Event_Msg_Type                Event_Msg_Type_Description
0                 1  Made Shot
32                2  Missed Shot
65                3  Free Throw
146               3  No Shot
72                4  Rebound
74                5  Turnover
100               6  Foul
116               7  Violation
121               8  Substitution
126               9  Timeout
131              10  Jump Ball
132              11  Ejection
135              12  Start Period
136              13  End Period
```

0.1.1 important event for analysis purposes

8 substitution 3 free throw 1 made shot

```
In [6]: eventcode_df[eventcode_df.Event_Msg_Type==3].sort_values('Action_Type')
```

```
Out[6]:      Event_Msg_Type  Action_Type                Event_Msg_Type_Description  \
        146               3            0  No Shot
        65                3           10  Free Throw
        66                3           11  Free Throw
        67                3           12  Free Throw
        68                3           13  Free Throw
        142               3           14  Free Throw
        69                3           15  Free Throw
        70                3           16  Free Throw
        71                3           17  Free Throw
        159               3           18  Free Throw
        158               3           19  Free Throw
        157               3           20  Free Throw
        147               3           21  Free Throw
        160               3           22  Free Throw
        154               3           25  Free Throw
        155               3           26  Free Throw
        270               3           27  Free Throw
        271               3           28  Free Throw
        272               3           29  Free Throw

                              Action_Type_Description
        146                                       NaN
        65   Free Throw 1 of 1
        66   Free Throw 1 of 2
        67   Free Throw 2 of 2
        68   Free Throw 1 of 3
        142  Free Throw 2 of 3
        69   Free Throw 3 of 3
        70   Free Throw Technical
        71   Free Throw Clear Path
        159  Free Throw Flagrant 1 of 2
        158  Free Throw Flagrant 2 of 2
        157  Free Throw Flagrant 1 of 1
        147  Free Throw Technical 1 of 2
        160  Free Throw Technical 2 of 2
        154  Free Throw Clear Path 1 of 2
        155  Free Throw Clear Path 2 of 2
        270  Free Throw Flagrant 1 of 3
        271  Free Throw Flagrant 2 of 3
        272  Free Throw Flagrant 3 of 3
```

```
In [7]: free_throw = eventcode_df[eventcode_df.Event_Msg_Type==3].sort_values('Action_Type')
```

```
        # first throw of
        free_throw[free_throw.Action_Type_Description.str.contains('1 of', na=False)]
```

```
Out[7]:      Event_Msg_Type  Action_Type                Event_Msg_Type_Description  \
        65                3           10  Free Throw
        66                3           11  Free Throw
        68                3           13  Free Throw
        159               3           18  Free Throw
        157               3           20  Free Throw
        147               3           21  Free Throw
        154               3           25  Free Throw
        270               3           27  Free Throw

                              Action_Type_Description
        65   Free Throw 1 of 1
        66   Free Throw 1 of 2
        68   Free Throw 1 of 3
        159  Free Throw Flagrant 1 of 2
        157  Free Throw Flagrant 1 of 1
        147  Free Throw Technical 1 of 2
        154  Free Throw Clear Path 1 of 2
        270  Free Throw Flagrant 1 of 3
```

0.1.2 Calculate the 5 player on court during each event

Start from Event_Msg_ID 12 Start Period Before the first substitution 8

```
In [8]: game_df.head()
```

```
Out[8]:                              Game_id  Period                         Person_id  \
        0  021fd159b55773fba8157e2090fe0fe2       1  881f83d2dee3f18c7d1751659406144e
        1  021fd159b55773fba8157e2090fe0fe2       1  27ea17a8685c4919f157e83fe9cb2d9e
        2  021fd159b55773fba8157e2090fe0fe2       1  57bbd7e30bc694aeee9ee40c583e6811
        3  021fd159b55773fba8157e2090fe0fe2       1  cec898a1d355dbfbad8c760615fde1af
        4  021fd159b55773fba8157e2090fe0fe2       1  33963fe856a1523ff46438ba07d1d99f

                                     Team_id status
        0  012059d397c0b7e5a30a5bb89c0b075e      A
        1  cff694c8186a4bd377de400e4f60fe47      A
        2  cff694c8186a4bd377de400e4f60fe47      A
        3  012059d397c0b7e5a30a5bb89c0b075e      A
        4  cff694c8186a4bd377de400e4f60fe47      A
```

We can have period larger than 4

```
In [9]: set(game_df.Period.tolist())
```

```
Out[9]: {1, 2, 3, 4, 5}
```

```
In [10]: # status not useful
         set(game_df.status.tolist())

Out[10]: {'A'}

In [11]: by_team = game_df.groupby(['Game_id',"Period",'Team_id'])
         person_per_team_game = set(by_team['Person_id'].count().tolist())
         print (person_per_team_game)
         if len(person_per_team_game)==1:
             print ('All the team have ',person_per_team_game, 'Players at the initiate per peri

{5}
All the team have  {5} Players at the initiate per period


In [12]: def to_set(col):
             return list(set(col))
```

0.2 create a df of first 5 player at the beginning of each period

```
In [13]: l = by_team.Person_id.apply(to_set)
         shoufa = l.reset_index()
```

0.3 important columns in PbyP dataframe:

WC_Time: Real_world time, in unit of 0.1s
PC_Time: Countdown watch time per quarter, 0-7200 in 0.1s
OPtion1: Scores for each event, (1-3) pts
Persom1, Person2: for assisst event: Person1 is the shot maker; for substitution: Person1 is the ID leaving the game

0.3.1 Check the persion in game 1 period 1

021fd159b55773fba8157e2090fe0fe2 1 012059d397c0b7e5a30a5bb89c0b075e
All the player of team 1 on court during the game
who is on which team?

```
In [14]: pbyp_df.head()

Out[14]:                            Game_id  Event_Num  Event_Msg_Type  Period  \
         0  021fd159b55773fba8157e2090fe0fe2          0              12       1
         1  021fd159b55773fba8157e2090fe0fe2          1              10       1
         2  021fd159b55773fba8157e2090fe0fe2          2               2       1
         3  021fd159b55773fba8157e2090fe0fe2          3               4       1
         4  021fd159b55773fba8157e2090fe0fe2          6               6       1

            WC_Time  PC_Time  Action_Type  Option1  Option2  Option3  \
         0   546427     7200            0        0        0        0
         1   546495     7200            0        0        0        0
```

```
2   546665     7050              1         3         0         0
3   546714     6960              0         0         0         0
4   546886     6920              4         0         0         0

                            Team_id                           Person1  \
0  1473d70e5646a26de3c52aa1abd85b1f  6bcf6c1f8c373d25fca1579bc4464a91
1  012059d397c0b7e5a30a5bb89c0b075e  89706b99ddd00dc05d37ef5cafc04276
2  012059d397c0b7e5a30a5bb89c0b075e  cec898a1d355dbfbad8c760615fde1af
3  012059d397c0b7e5a30a5bb89c0b075e  307beab25b1021a548b4a47550bc4b25
4  cff694c8186a4bd377de400e4f60fe47  c00264c3114d23bac482e9de50fb7d28

                            Person2  Team_id_type
0  6bcf6c1f8c373d25fca1579bc4464a91             0
1  307beab25b1021a548b4a47550bc4b25             2
2  6bcf6c1f8c373d25fca1579bc4464a91             2
3  6bcf6c1f8c373d25fca1579bc4464a91             2
4  89706b99ddd00dc05d37ef5cafc04276             3
```

6bcf6c1f8c373d25fca1579bc4464a91 is the default person_id for no-second-person in the event

Team_id type: 2 for home, 3 for away team, 0 is default team (judges)

```
In [15]: # Game list
         Game_list = list(set(pbyp_df.Game_id.tolist()))
         print ("Here is totally: ",len(Game_list), "games")
```

```
Here is totally:  50 games
```

```
In [16]: def team_set(col): # rule out the default team
             return list(set(i for i in col if i!="1473d70e5646a26de3c52aa1abd85b1f"))

         # Teams associated with each game
         tmp = pbyp_df.groupby("Game_id").Team_id.apply(team_set)
         tmp.head()

         game_teams = tmp.to_dict()
```

```
In [17]: # dictionary of player and score
         # demo
         dic_players = {}
         Game_i = '021fd159b55773fba8157e2090fe0fe2'
         team1_shoufa = shoufa[(shoufa.Game_id== Game_i)&(shoufa.Period == 1)]
         print ("first 5 player on team 1:", team1_shoufa.Person_id[0])
         print ("first 5 player on team 2:", team1_shoufa.Person_id[1])
```

```
first 5 player on team 1: ['881f83d2dee3f18c7d1751659406144e', '89706b99ddd00dc05d37ef5cafc04276
first 5 player on team 2: ['27ea17a8685c4919f157e83fe9cb2d9e', '57bbd7e30bc694aeee9ee40c583e6811
```

```
In [18]: # Special free throw cases
         pbyp_df[(pbyp_df.Event_Msg_Type==3)&(pbyp_df.Action_Type.isin(range(16,30)))]
```

```
Out[18]:                                 Game_id  Event_Num  Event_Msg_Type  Period  \
         178    021fd159b55773fba8157e2090fe0fe2        210               3       2
         331    021fd159b55773fba8157e2090fe0fe2        391               3       3
         1345   06bb1d31c63891e2580ff12e4e6505b4        441               3       4
         1463   07e76f7482773e81e2351d1692e9e5bb         97               3       1
         1638   07e76f7482773e81e2351d1692e9e5bb        323               3       3
         2131   0868dee930f69a54541d4ae88b841a37        276               3       3
         2152   0868dee930f69a54541d4ae88b841a37        305               3       3
         2179   0868dee930f69a54541d4ae88b841a37        340               3       3
         3779   1eab6189ad9ab246c197575a8c4eebe5         26               3       1
         4076   1eab6189ad9ab246c197575a8c4eebe5        367               3       3
         4794   2bf4ac0ed9ac1aee8767134d62b34dfe        149               3       1
         5103   2bf4ac0ed9ac1aee8767134d62b34dfe        528               3       4
         5450   2faabfa663f4dfb9ed83a1482088e092        272               3       2
         5451   2faabfa663f4dfb9ed83a1482088e092        274               3       2
         6102   3152e9c330ce200cc189ae64ebdf41fc        435               3       4
         6491   33f631fec90cc1f08bb16cff5ed52f9b        360               3       3
         6539   33f631fec90cc1f08bb16cff5ed52f9b        416               3       4
         6541   33f631fec90cc1f08bb16cff5ed52f9b        418               3       4
         6739   345f285f7c654bf7a03e940a9bc923c3        116               3       2
         6881   345f285f7c654bf7a03e940a9bc923c3        275               3       3
         7175   3b753670a0d0df2c35c7ce2e0bc94e6c        123               3       2
         8073   598a55c8bf052d039d0fb4ff1a62f98a        140               3       1
         9018   87d1574f478f37d13789284b96b4f6cb        208               3       2
         9815   896bcac9b0f35b250c9fb7a6325d8599        143               3       2
         9936   896bcac9b0f35b250c9fb7a6325d8599        296               3       3
         9941   896bcac9b0f35b250c9fb7a6325d8599        306               3       3
         9942   896bcac9b0f35b250c9fb7a6325d8599        308               3       3
         10039  896bcac9b0f35b250c9fb7a6325d8599        426               3       4
         10241  8fd2943309c7accc78b41e047e1e9393        111               3       2
         10831  9db75b921076b5848cba7ca36436147c        215               3       2
         11228  9dce47bc3cbabd451a9bab758ca5467f        146               3       2
         11820  a109cef8327feae75c3e4885cb4c6e2d        306               3       3
         11831  a109cef8327feae75c3e4885cb4c6e2d        319               3       3
         12000  a32200385b4594cb11f8a19e92c96835         42               3       1
         12758  a442eeb8e92d2941addba3996bd7d0a0        464               3       4
         13383  b2abcb6a29ed3d99db98084c96638bff        135               3       2
         13947  b2c59cbe723394436526560db23e6a93        223               3       2
         14262  bd1f8947f5541c2fb22f6328a58f4fc6         89               3       1
         16243  c18a10de1375b1f12aa17ef6cc540102        254               3       2
         16429  c18a10de1375b1f12aa17ef6cc540102        479               3       4
         16431  c18a10de1375b1f12aa17ef6cc540102        482               3       4
         18255  d9eb042455c8f36574d3a52e678a8916        387               3       4
         19903  ded7de5f5252e901151b00938ce580c1        209               3       2
         20013  ded7de5f5252e901151b00938ce580c1        336               3       3
```

```
21907  f0332f91eb943b64b39ef28daf6639c5        455               3       4

       WC_Time  PC_Time  Action_Type  Option1  Option2  Option3  \
178     570814     3670           16        1        0        0
331     602515     1460           16        1        0        0
1345    637313     1420           16        1        0        0
1463    561020     1930           16        1        0        0
1638    598551     5840           16        2        0        0
2131    607797     6640           16        2        0        0
2152    612541     5330           16        1        0        0
2179    616202     3750           16        2        0        0
3779    675111     5690           16        2        0        0
4076    728103     1840           16        2        0        0
4794    615834      389           16        1        0        0
5103    672974     5740           16        1        0        0
5450    507471      396           25        1        1        0
5451    507631      396           26        1        1        0
6102    662349     5750           16        1        0        0
6491    670675     1810           16        1        0        0
6539    680372     6000           18        2        0        0
6541    680483     6000           19        2        0        0
6739    635757     6160           16        2        0        0
6881    668512     4460           16        2        0        0
7175    598616     6720           16        1        0        0
8073    689590      581           16        2        0        0
9018    591407     2780           16        1        0        0
9815    618741     6580           16        1        0        0
9936    647406     5470           16        1        0        0
9941    649936     5340           18        1        0        0
9942    650024     5340           19        1        0        0
10039   665096     6780           16        1        0        0
10241   598309     6290           16        2        0        0
10831   701653      740           16        1        0        0
11228   601126     5500           16        1        0        0
11820   615109     1940           16        1        0        0
11831   617508     1400           16        1        0        0
12000   587624     4090           16        1        0        0
12758   670697     3770           16        1        0        0
13383   636813     6960           16        1        0        0
13947   644649     2130           16        1        0        0
14262   611107     2610           16        1        0        0
16243   710003     1110           16        1        0        0
16429   748943     3870           16        1        0        0
16431   749034     3870           16        1        0        0
18255   389931     5710           16        1        0        0
19903   679145     3820           16        1        0        0
20013   704426     2950           16        2        0        0
21907   669632     1950           16        1        0        0
```

```
                                Team_id                           Person1  \
178    cff694c8186a4bd377de400e4f60fe47  33963fe856a1523ff46438ba07d1d99f
331    012059d397c0b7e5a30a5bb89c0b075e  89706b99ddd00dc05d37ef5cafc04276
1345   cc5e0f4efd0f5410547d4e73f38f7811  d8e4f0fd2e836518e681c99a3f9e4411
1463   8e90edc4eb908903bcbb567f92488049  9075d598a348b290d9c426e41662e593
1638   78cb4fd2222d4d6c7ce0a53cf99b4b14  f2e41e7ce25aff54b85ee5ab66f69187
2131   78cb4fd2222d4d6c7ce0a53cf99b4b14  5f2d1b7a31eb9117654c9d056093bfa5
2152   3cd0b15957ceb80f5125bef8bd1bbea7  45e7a758a91a254ff04756562e740f2d
2179   78cb4fd2222d4d6c7ce0a53cf99b4b14  bc77b0f38cb51b2cfe8552000e585cbf
3779   3a30c29e8a04b4f1335ee8874c526a6c  c5a353e301078ea1cb4e8bc8fede12e9
4076   cd45058739ed0ac8229849c6249aad48  c5a353e301078ea1cb4e8bc8fede12e9
4794   bcdf0d0117f2e3d5afcd70f977eae3d6  0f02d616e027988803324c84e8951a0e
5103   8e90edc4eb908903bcbb567f92488049  0f02d616e027988803324c84e8951a0e
5450   689ca95d19ef8d959ae9e70f41b6d0a7  4b095e823bb1b4857a50d41a511930d4
5451   689ca95d19ef8d959ae9e70f41b6d0a7  4b095e823bb1b4857a50d41a511930d4
6102   b64dba595e29fe34a124c65f682f27fb  601a72cc4a6a409c9463e1a59a01fa29
6491   dc0f916810cb46c22beec67a860e00b4  f2f904a986118f6e2e67da38c64ffe81
6539   dc0f916810cb46c22beec67a860e00b4  5e0158b93a72b41987eec3429fb2038f
6541   dc0f916810cb46c22beec67a860e00b4  5e0158b93a72b41987eec3429fb2038f
6739   e2d25dd55740327314a2a0b874621ba8  ed343691d2b2712df2037aa5c1a0f9ab
6881   e2d25dd55740327314a2a0b874621ba8  1f8e6e61eb13aa30f5384a7e6716ec19
7175   9eaab5de3f7ccd60442845f4e6b70a75  a07fb375dddea9eb7ca13848b9941daf
8073   3a30c29e8a04b4f1335ee8874c526a6c  c5a353e301078ea1cb4e8bc8fede12e9
9018   87010ad9e5c5e0b7628d3611b3693652  978f65d65974dcaaea595daaa1b35a2a
9815   e1f2c56e5df4f2e975a969f329a7068c  b90945b4a95bfc10ffc239ec4eac211d
9936   b64dba595e29fe34a124c65f682f27fb  f666125388be8d19b945f07d44610948
9941   b64dba595e29fe34a124c65f682f27fb  a1b3d414dba333bb2413b054b8460c07
9942   b64dba595e29fe34a124c65f682f27fb  a1b3d414dba333bb2413b054b8460c07
10039  b64dba595e29fe34a124c65f682f27fb  70f7e0aad42e27aacb7490d349a4b233
10241  cff694c8186a4bd377de400e4f60fe47  33963fe856a1523ff46438ba07d1d99f
10831  6f5316b35198f3198c828d9882a79846  e2d5a38e5f826708c89f2f728eade916
11228  e1f2c56e5df4f2e975a969f329a7068c  e96dc78c444bead3a3e7d169f2fa6e07
11820  e2d25dd55740327314a2a0b874621ba8  056755eee40e0ae3ce0cebd5b9053a3c
11831  e2d25dd55740327314a2a0b874621ba8  056755eee40e0ae3ce0cebd5b9053a3c
12000  dc0f916810cb46c22beec67a860e00b4  0f79f5ac1d01d9639c74a0cce4a8c7b3
12758  e1f2c56e5df4f2e975a969f329a7068c  57ed4bb35e7cef3fc5f579bd119ceeca
13383  e2d25dd55740327314a2a0b874621ba8  ed343691d2b2712df2037aa5c1a0f9ab
13947  78cb4fd2222d4d6c7ce0a53cf99b4b14  5f2d1b7a31eb9117654c9d056093bfa5
14262  e1f2c56e5df4f2e975a969f329a7068c  c6a9f7baaf3b88105816cf658c0cf7e8
16243  6f5316b35198f3198c828d9882a79846  e2d5a38e5f826708c89f2f728eade916
16429  6f5316b35198f3198c828d9882a79846  e2d5a38e5f826708c89f2f728eade916
16431  6f5316b35198f3198c828d9882a79846  e2d5a38e5f826708c89f2f728eade916
18255  0a6fc39f34702cff144525dfd265f8fa  0f02d616e027988803324c84e8951a0e
19903  91ea9fc14670a54b9902eb062b416ccf  bc77b0f38cb51b2cfe8552000e585cbf
20013  78cb4fd2222d4d6c7ce0a53cf99b4b14  5f2d1b7a31eb9117654c9d056093bfa5
21907  bcdf0d0117f2e3d5afcd70f977eae3d6  0a7ae25da315e35e903233b4bb2252da
```

```
                                 Person2  Team_id_type
178    6bcf6c1f8c373d25fca1579bc4464a91              3
331    6bcf6c1f8c373d25fca1579bc4464a91              2
1345   6bcf6c1f8c373d25fca1579bc4464a91              3
1463   6bcf6c1f8c373d25fca1579bc4464a91              3
1638   6bcf6c1f8c373d25fca1579bc4464a91              2
2131   6bcf6c1f8c373d25fca1579bc4464a91              3
2152   6bcf6c1f8c373d25fca1579bc4464a91              2
2179   6bcf6c1f8c373d25fca1579bc4464a91              3
3779   6bcf6c1f8c373d25fca1579bc4464a91              3
4076   6bcf6c1f8c373d25fca1579bc4464a91              2
4794   6bcf6c1f8c373d25fca1579bc4464a91              2
5103   6bcf6c1f8c373d25fca1579bc4464a91              3
5450   6bcf6c1f8c373d25fca1579bc4464a91              2
5451   6bcf6c1f8c373d25fca1579bc4464a91              2
6102   6bcf6c1f8c373d25fca1579bc4464a91              3
6491   6bcf6c1f8c373d25fca1579bc4464a91              3
6539   6bcf6c1f8c373d25fca1579bc4464a91              3
6541   6bcf6c1f8c373d25fca1579bc4464a91              3
6739   6bcf6c1f8c373d25fca1579bc4464a91              2
6881   6bcf6c1f8c373d25fca1579bc4464a91              2
7175   6bcf6c1f8c373d25fca1579bc4464a91              2
8073   6bcf6c1f8c373d25fca1579bc4464a91              3
9018   6bcf6c1f8c373d25fca1579bc4464a91              3
9815   6bcf6c1f8c373d25fca1579bc4464a91              2
9936   6bcf6c1f8c373d25fca1579bc4464a91              3
9941   6bcf6c1f8c373d25fca1579bc4464a91              3
9942   6bcf6c1f8c373d25fca1579bc4464a91              3
10039  6bcf6c1f8c373d25fca1579bc4464a91              3
10241  6bcf6c1f8c373d25fca1579bc4464a91              2
10831  6bcf6c1f8c373d25fca1579bc4464a91              3
11228  6bcf6c1f8c373d25fca1579bc4464a91              3
11820  6bcf6c1f8c373d25fca1579bc4464a91              3
11831  6bcf6c1f8c373d25fca1579bc4464a91              3
12000  6bcf6c1f8c373d25fca1579bc4464a91              2
12758  6bcf6c1f8c373d25fca1579bc4464a91              2
13383  6bcf6c1f8c373d25fca1579bc4464a91              2
13947  6bcf6c1f8c373d25fca1579bc4464a91              3
14262  6bcf6c1f8c373d25fca1579bc4464a91              2
16243  6bcf6c1f8c373d25fca1579bc4464a91              3
16429  6bcf6c1f8c373d25fca1579bc4464a91              3
16431  6bcf6c1f8c373d25fca1579bc4464a91              3
18255  6bcf6c1f8c373d25fca1579bc4464a91              2
19903  6bcf6c1f8c373d25fca1579bc4464a91              2
20013  6bcf6c1f8c373d25fca1579bc4464a91              3
21907  6bcf6c1f8c373d25fca1579bc4464a91              2
```

0.4 For special Free throw, we do not have to worry about "substitution" in between.

```
In [19]: direct_free_throw = list(range(16,30))
         direct_free_throw.append(10)
         print (direct_free_throw)

[16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 10]
```

```
In [20]: count_free_throw = [11, 13]
```

0.4.1 For regular free throw, we can just consider the first throw

once the first throw start we consider total score of the throw 65 3 10 Free Throw Free Throw 1 of 1

66 3 11 Free Throw Free Throw 1 of 2
67 3 12 Free Throw Free Throw 2 of 2
68 3 13 Free Throw Free Throw 1 of 3
142 3 14 Free Throw Free Throw 2 of 3
69 3 15 Free Throw Free Throw 3 of 3

```
In [21]: # function to calculate score in the free throw round
         def free_throw_score(row, num_of_throw): # input the row_index of the first free throw
             current_throw = 1
             cur_row = int(row.name)
             score = row.Option1
             while (current_throw < num_of_throw):
                 cur_row = cur_row + 1
                 if pbyp_df.iloc[cur_row,2]==3:
                     current_throw += 1
                     score = score + pbyp_df.iloc[cur_row,7]
             return score
```

```
In [22]: def plus_minus(score_team, loss_team, pts, dic):
             # input the active persons in the score_team and loss_team:
             for person in score_team:
                 if person in dic.keys():
                     dic[person] = dic[person]+pts
                 else:
                     dic[person] = pts
             for person in loss_team:
                 if person in dic.keys():
                     dic[person] = dic[person]-pts
                 else:
                     dic[person] = -pts
```

```
In [23]: def event_judge(row, team_active_1, team_active_2, dic):
             # scores
         #         1  Made Shot
         #         3  Free Throw
             if row.Event_Msg_Type == 1:
                 pts = row.Option1
                 if row.Person1 in team_active_1:
                     plus_minus(team_active_1, team_active_2, pts, dic) # def plus_minus(score_t
                 elif row.Person1 in team_active_2:
                     plus_minus(team_active_2, team_active_1, pts, dic)
                 else:
                     print ("Error: This person %s should not be on court"%row.Person1)
             elif (row.Event_Msg_Type == 3):
                 # direct free throw
                 if (row.Action_Type in direct_free_throw):
                     pts = row.Option1
                 elif row.Action_Type ==11:
                     pts = free_throw_score(row, 2)
                 elif row.Action_Type ==13:
                     pts = free_throw_score(row, 3)
                 else: # ignore 2 of 2, 2 of 3, 3 of 3 cases:
                     pts = 0
                 if row.Person1 in team_active_1:
                     plus_minus(team_active_1, team_active_2, pts, dic) # def plus_minus(score_t
                 elif row.Person1 in team_active_2:
                     plus_minus(team_active_2, team_active_1, pts, dic)
                 else:
                     print ("Error: This person %s should not be on court"%row.Person1)
         #         8  Substitution
             elif row.Event_Msg_Type == 8:
                 player_out = row.Person1
                 player_in = row.Person2
                 if (player_out in team_active_1) and (player_in not in team_active_1):
                     team_active_1.remove(player_out)
                     team_active_1.append(player_in)
                     assert(len(team_active_1)==5)
                 elif (player_out in team_active_2) and (player_in not in team_active_2):
                     team_active_2.remove(player_out)
                     team_active_2.append(player_in)
                     assert(len(team_active_2)==5)
                 else:
                     print ("Error: Substitution problem in row %s!"%row.index)
```

```
In [24]: result = []
         for game_i in Game_list:
             # loop_over games
             # initiate two teams in the game
             team_1 = game_teams[game_i][0]
             team_2 = game_teams[game_i][1]
              # print ( "two teams in Game %s are:"%index, team_1, team_2)
             dic = {} #initiate socre dict for the game
             # loop over Period of the game
             for period_i in range(1,6):
                 df = pbyp_df[(pbyp_df.Game_id==game_i)&(pbyp_df.Period==period_i)]
                 if not df.empty:
                     # get initial player of each team
                     team_shoufa = shoufa[(shoufa.Game_id== game_i)&(shoufa.Period == period_i)]
                     team_active_1 = team_shoufa.Person_id.tolist()[0]
                     team_active_2 = team_shoufa.Person_id.tolist()[1]
                     # loop over the df of current game and period
                     df.apply(event_judge, args =(team_active_1, team_active_2, dic), axis=1)
             tmp_result = pd.Series(dic, name='Player_Plus/Minus')
             tmp_result.index.name = 'Player_ID'
             tmp_result = tmp_result.reset_index()
             tmp_result['Game_ID'] = game_i
             result.append(tmp_result)
```

0.4.2 Check the sum of all players in a game should be 0

```
In [25]: # asseble all the dfs generated
         result_df = pd.concat(result, axis=0)
```

```
In [26]: # check the sum of all score should be 0
         result_df.groupby('Game_ID')["Player_Plus/Minus"].sum()
```

```
Out[26]: Game_ID
         021fd159b55773fba8157e2090fe0fe2    0
         03a31e84b194d6c8a2eab5d70ba67acf    0
         06bb1d31c63891e2580ff12e4e6505b4    0
         07e76f7482773e81e2351d1692e9e5bb    0
         0868dee930f69a54541d4ae88b841a37    0
         09d46e3d7a8253b7209100650b5afaeb    0
         13ced855d491384876c6ab807bd1d3db    0
         15d76177caa6022156e83774c2e054d3    0
         1eab6189ad9ab246c197575a8c4eebe5    0
         1f9e3cb05c031986cf8bc7c0a84cc517    0
         2bf4ac0ed9ac1aee8767134d62b34dfe    0
         2faabfa663f4dfb9ed83a1482088e092    0
         3152e9c330ce200cc189ae64ebdf41fc    0
         33f631fec90cc1f08bb16cff5ed52f9b    0
         345f285f7c654bf7a03e940a9bc923c3    0
         3b753670a0d0df2c35c7ce2e0bc94e6c    0
         56ccfaf0adead6f4c7236a01ca0cfbdc    0
         598a55c8bf052d039d0fb4ff1a62f98a    0
         7fad2269ee0d11ae5069ff23ecb25913    0
         87d1574f478f37d13789284b96b4f6cb    0
         88012a99d7fd1c169e2360aa5cdf8bfa    0
         896bcac9b0f35b250c9fb7a6325d8599    0
         8fd2943309c7accc78b41e047e1e9393    0
```

```
9db75b921076b5848cba7ca36436147c    0
9dce47bc3cbabd451a9bab758ca5467f    0
a109cef8327feae75c3e4885cb4c6e2d    0
a32200385b4594cb11f8a19e92c96835    0
a442eeb8e92d2941addba3996bd7d0a0    0
a55fe197c4ae19094a2303a950c7e70c    0
b2abcb6a29ed3d99db98084c96638bff    0
b2c59cbe723394436526560db23e6a93    0
bd1f8947f5541c2fb22f6328a58f4fc6    0
be5f61a5354154b3d7cba7536f189e2f    0
bfc4c8f688f511fb76f7fd82a3807f94    0
c005dbf89ef3fb55d6f0f0461ada5560    0
c18a10de1375b1f12aa17ef6cc540102    0
cc638bdf25316add638154b45434aebd    0
d0c6985ffc5f03ba09393699467a580a    0
d3a00898abc790fd5643d05f996f05f6    0
d9eb042455c8f36574d3a52e678a8916    0
dadcdbe527d9d72f8b0907796f7559cb    0
dc9261160f28c7e28ffe11e9724aa2b3    0
dd4d15aab3b091546f9bf65b7f562f26    0
ded7de5f5252e901151b00938ce580c1    0
e26fa57f2c5e83edc21e3f97b55cbecd    0
e3ca2e93a38f7497b6f476c86f5b2136    0
e44e31fefaa028c1e9df1a71d4aff555    0
f0332f91eb943b64b39ef28daf6639c5    0
f385e8caae86b6cb58ed3241c1278ce3    0
fdeb2950c4d5209d449ebd2d6afac11e    0
Name: Player_Plus/Minus, dtype: int64
```

```
In [27]: result_df[['Game_ID','Player_ID','Player_Plus/Minus']].reset_index(drop=True).to_csv('N
```

0.4.3 Final result saved to result file