In [1]:
```
import pandas as pd
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('../high_diamond_ranked_10min.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	gameId	blueWins	blueWardsPlaced	blueWardsDestroyed	blueFirstBlood	blueKills	blueD
0	4519157822	0	28	2	1	9	
1	4523371949	0	12	1	0	5	
2	4521474530	0	15	0	0	7	
3	4524384067	0	43	1	0	4	
4	4436033771	0	75	4	0	6	

5 rows × 40 columns

In [4]:
```
df.isnull().sum()
```

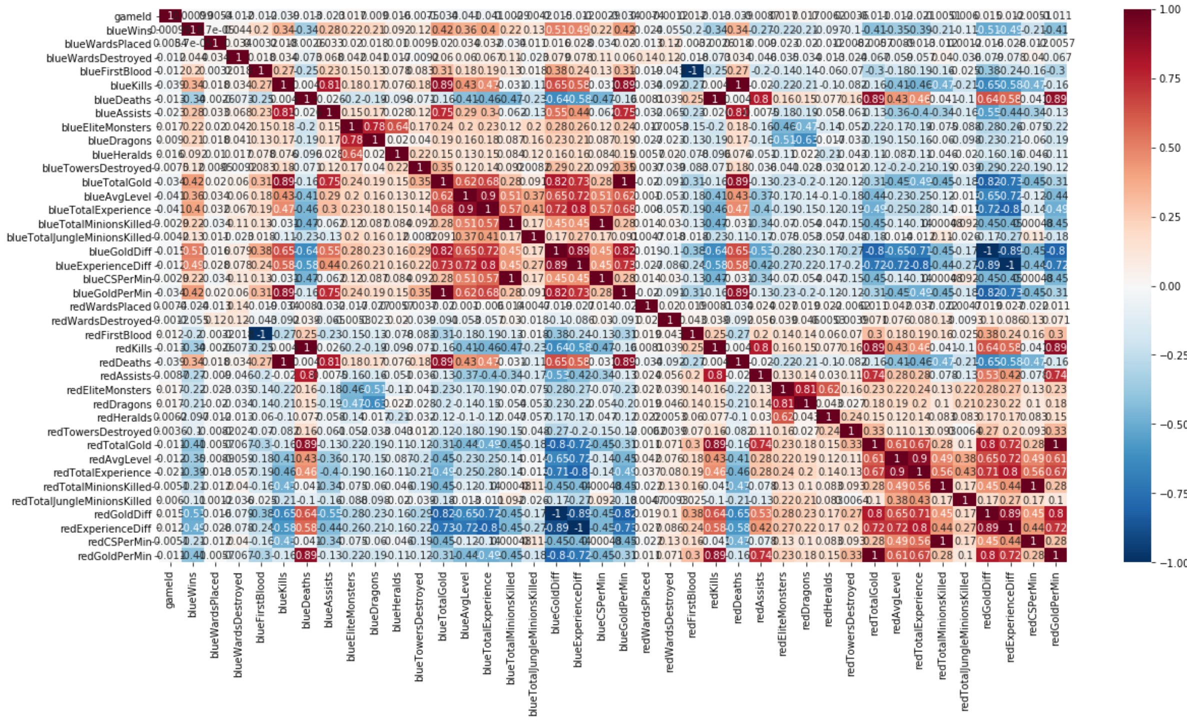
Out[4]:
```
gameId                          0
blueWins                        0
blueWardsPlaced                 0
blueWardsDestroyed              0
blueFirstBlood                  0
blueKills                       0
blueDeaths                      0
blueAssists                     0
blueEliteMonsters               0
blueDragons                     0
blueHeralds                     0
blueTowersDestroyed             0
blueTotalGold                   0
blueAvgLevel                    0
blueTotalExperience             0
blueTotalMinionsKilled          0
blueTotalJungleMinionsKilled    0
blueGoldDiff                    0
blueExperienceDiff              0
blueCSPerMin                    0
blueGoldPerMin                  0
redWardsPlaced                  0
redWardsDestroyed               0
redFirstBlood                   0
redKills                        0
redDeaths                       0
redAssists                      0
redEliteMonsters                0
redDragons                      0
redHeralds                      0
redTowersDestroyed              0
redTotalGold                    0
redAvgLevel                     0
redTotalExperience              0
redTotalMinionsKilled           0
redTotalJungleMinionsKilled     0
redGoldDiff                     0
redExperienceDiff               0
redCSPerMin                     0
redGoldPerMin                   0
dtype: int64
```

In [5]:
```
df.dtypes
```

Out[5]:
```
gameId                            int64
blueWins                          int64
blueWardsPlaced                   int64
blueWardsDestroyed                int64
blueFirstBlood                    int64
blueKills                         int64
blueDeaths                        int64
blueAssists                       int64
blueEliteMonsters                 int64
blueDragons                       int64
blueHeralds                       int64
blueTowersDestroyed               int64
blueTotalGold                     int64
blueAvgLevel                    float64
blueTotalExperience               int64
blueTotalMinionsKilled            int64
blueTotalJungleMinionsKilled      int64
blueGoldDiff                      int64
blueExperienceDiff                int64
blueCSPerMin                    float64
blueGoldPerMin                  float64
redWardsPlaced                    int64
redWardsDestroyed                 int64
redFirstBlood                     int64
redKills                          int64
redDeaths                         int64
redAssists                        int64
redEliteMonsters                  int64
redDragons                        int64
redHeralds                        int64
redTowersDestroyed                int64
redTotalGold                      int64
redAvgLevel                     float64
redTotalExperience                int64
redTotalMinionsKilled             int64
redTotalJungleMinionsKilled       int64
redGoldDiff                       int64
redExperienceDiff                 int64
redCSPerMin                     float64
redGoldPerMin                   float64
dtype: object
```

In [6]:
```
# Finding the relations between the variables.
plt.figure(figsize=(20,10))
sns.heatmap(df.corr(),cmap='RdBu_r',annot=True)
```

Out[6]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2bc08ee9ba8>
```



Some variable pairs are perfectly correlated so they are interchangble:

1.redFirstBlood and blueFirstBlood

2.redDeaths and blueKills

3.redKills and bluDeaths

4.blueGoldPerMin and blueTotalGold

5.blueCSPerMin and blueTotalMinionsKilled

6.redGoldDiff and blueGoldDiff

7.redExperienceDiff and blueExperienceDiff

8.redGoldPerMin and redTotalGold

9.redCSPerMin and redTotalMinionsKilled

Dropping irrelevant columns

In [7]:
```
# gameId are irrelavant obviously. blueAvgLevel and redAvgLevel are not signif
icant?
#TotalJungleMinionsKilled and TotalMinionsKilled are repeated
df = df.drop(columns=['gameId','redFirstBlood','redDeaths','redKills','blueGol
dPerMin','blueCSPerMin',
                      'redGoldDiff','redExperienceDiff','redGoldPerMin','redCS
PerMin',
                       'blueTotalJungleMinionsKilled','redTotalJungleMinionsKi
lled'])
df.head()
```

Out[7]:

	blueWins	blueWardsPlaced	blueWardsDestroyed	blueFirstBlood	blueKills	blueDeaths	blueA
0	0	28	2	1	9	6	
1	0	12	1	0	5	5	
2	0	15	0	0	7	11	
3	0	43	1	0	4	5	
4	0	75	4	0	6	6	

5 rows × 28 columns

standardize

In [8]:
```
scaler = StandardScaler()
features = df.drop(columns=['blueWins']).values
x=scaler.fit_transform(features)
std_df= pd.DataFrame(data = x, columns = df.drop(columns=['blueWins']).columns
)
std_df.head()
```

Out[8]:

	blueWardsPlaced	blueWardsDestroyed	blueFirstBlood	blueKills	blueDeaths	blueAssists	blu
0	0.316996	-0.379275	0.990429	0.935301	-0.046926	1.071495	
1	-0.570992	-0.839069	-1.009663	-0.393216	-0.387796	-0.404768	
2	-0.404494	-1.298863	-1.009663	0.271042	1.657424	-0.650812	
3	1.149484	-0.839069	-1.009663	-0.725346	-0.387796	-0.404768	
4	2.925460	0.540312	-1.009663	-0.061087	-0.046926	-0.158724	

5 rows × 27 columns

Normalize

In [9]:
```
X = preprocessing.normalize(features)
norm_df = pd.DataFrame(data = X, columns = df.drop(columns=['blueWins']).colum
ns)
norm_df.head()
```

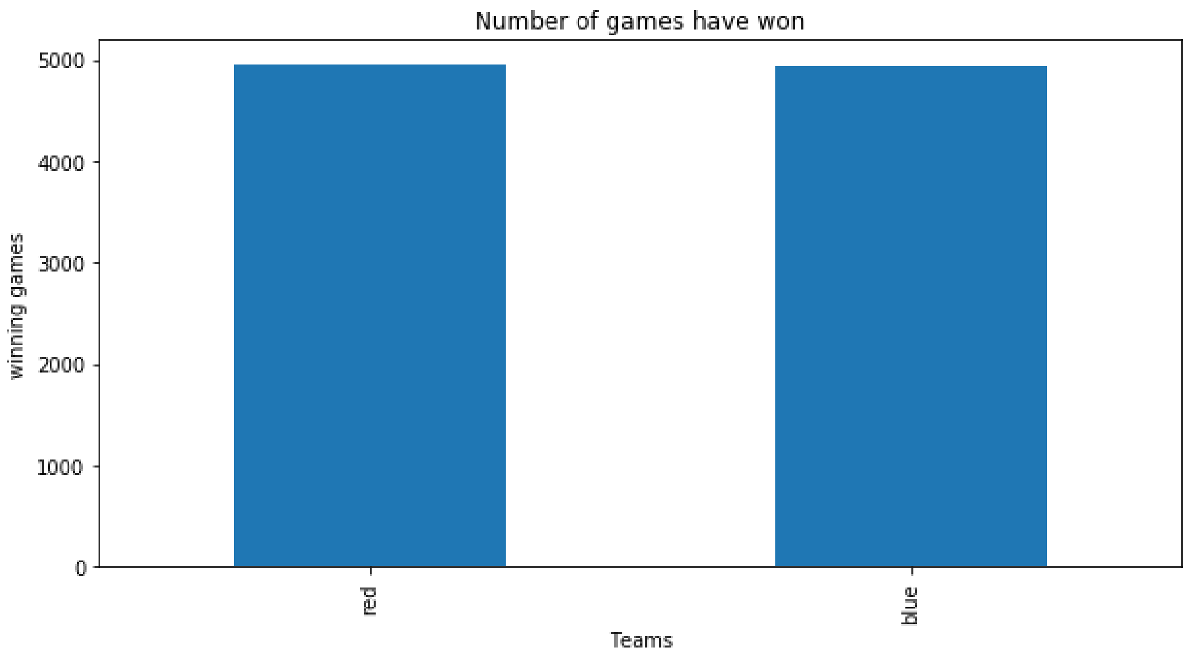
Out[9]:

	blueWardsPlaced	blueWardsDestroyed	blueFirstBlood	blueKills	blueDeaths	blueAssists	blu
0	0.000825	0.000059	0.000029	0.000265	0.000177	0.000324	
1	0.000361	0.000030	0.000000	0.000150	0.000150	0.000150	
2	0.000448	0.000000	0.000000	0.000209	0.000328	0.000119	
3	0.001269	0.000030	0.000000	0.000118	0.000148	0.000148	
4	0.002119	0.000113	0.000000	0.000170	0.000170	0.000170	

5 rows × 27 columns

In [10]:
```
df.blueWins.value_counts().plot(kind='bar', figsize=(10,5))
plt.title('Number of games have won')
plt.ylabel('winning games')
plt.xticks([0,1],['red','blue'])
plt.xlabel('Teams')
```

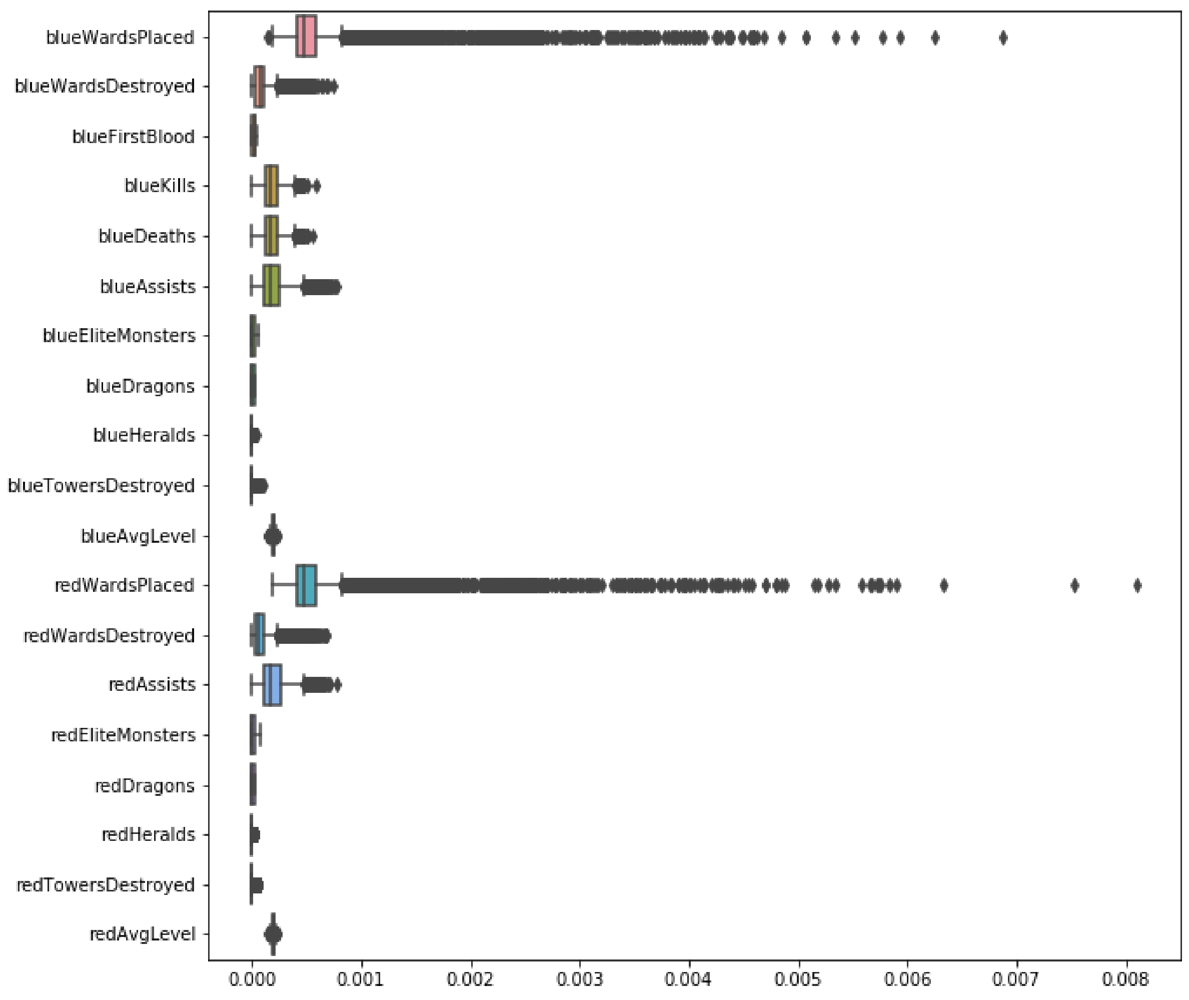
Out[10]:
```
Text(0.5, 0, 'Teams')
```

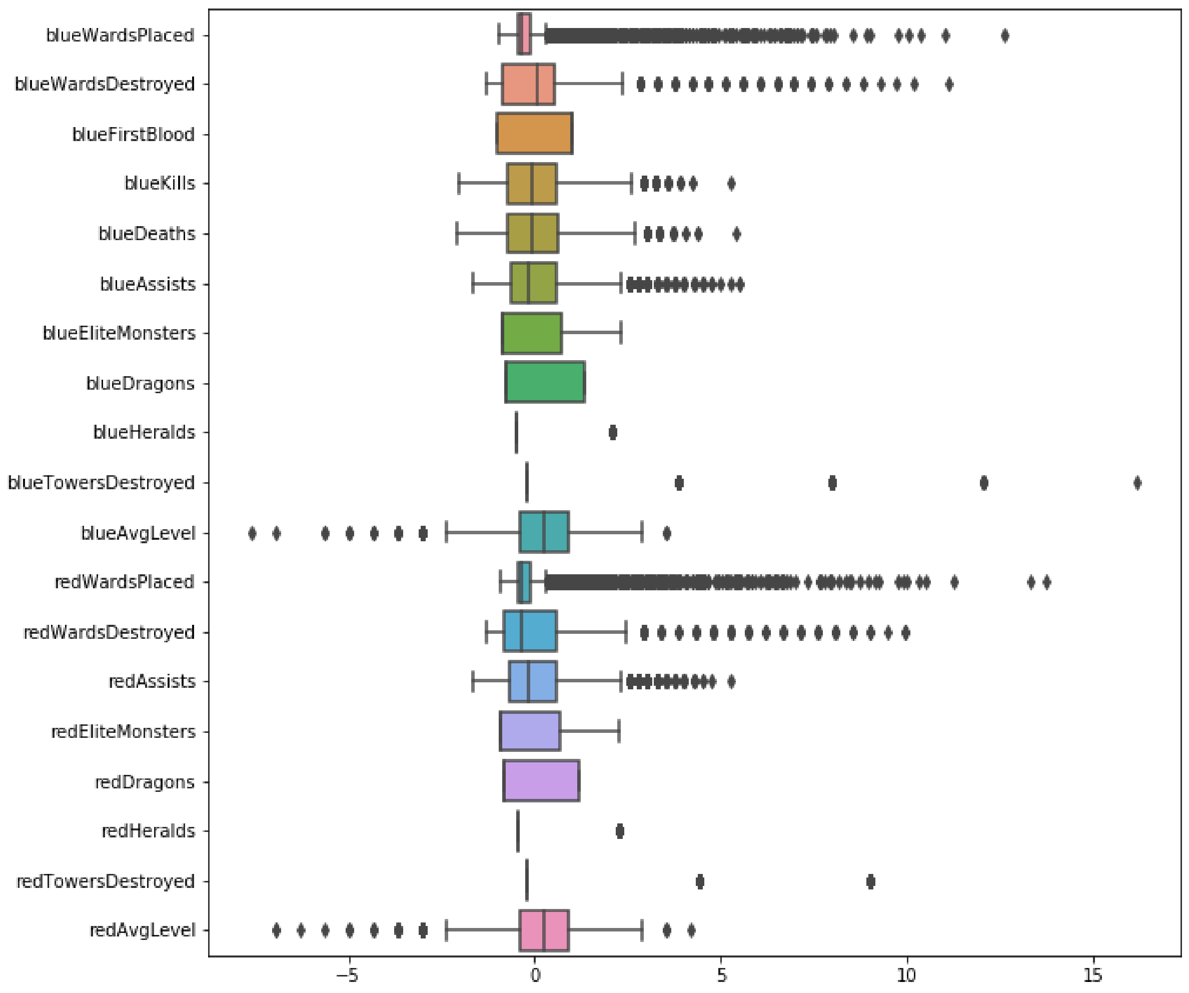


Detecting Outliers

In [11]:
```
fig,ax= plt.subplots(figsize=(10,10))
sns.boxplot(data=norm_df.drop(columns=['blueTotalGold','blueTotalExperience',
'redTotalGold',
                                       'redTotalExperience','blueGoldDiff', 'b
lueExperienceDiff',
                                       'blueTotalMinionsKilled','redTotalMinio
nsKilled'],),orient='h')
fig2,ax2= plt.subplots(figsize=(10,10))
sns.boxplot(data=std_df.drop(columns=['blueTotalGold','blueTotalExperience','r
edTotalGold',
                                       'redTotalExperience','blueGoldDiff', 'b
lueExperienceDiff',
                                       'blueTotalMinionsKilled','redTotalMinio
nsKilled'],),orient='h')
```

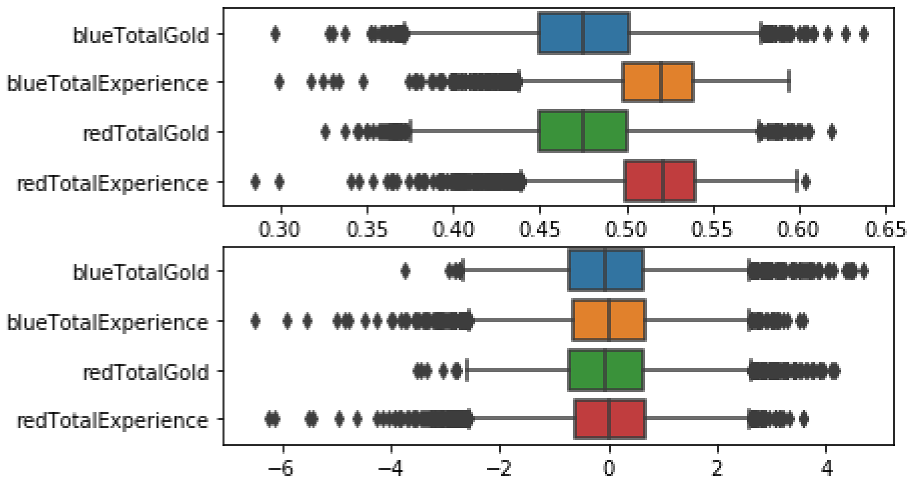
```
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x2bc09949f60>
```


blueWardsPlaced
blueWardsDestroyed
blueFirstBlood
blueKills
blueDeaths
blueAssists
blueEliteMonsters
blueDragons
blueHeralds
blueTowersDestroyed
blueAvgLevel
redWardsPlaced
redWardsDestroyed
redAssists
redEliteMonsters
redDragons
redHeralds
redTowersDestroyed
redAvgLevel
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008



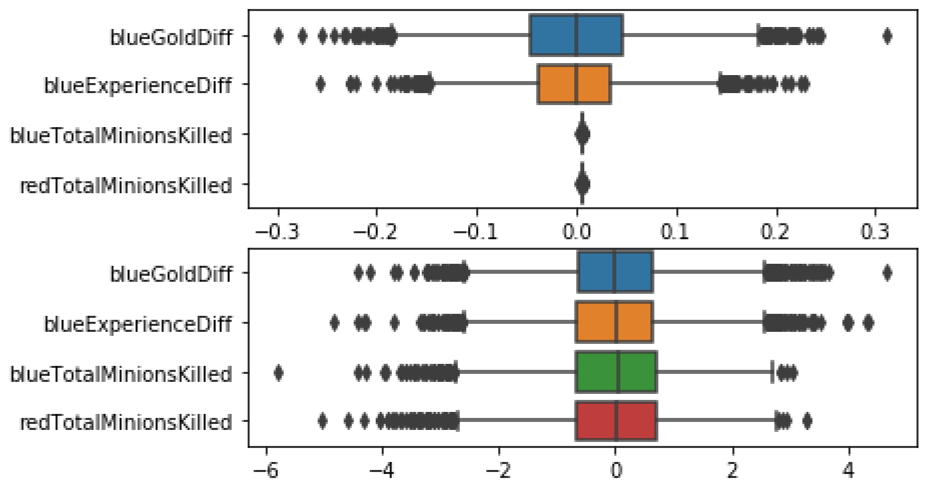
In [12]:
```
plt.subplot(2,1,1)
sns.boxplot(data=norm_df.loc[:,['blueTotalGold','blueTotalExperience','redTota
lGold','redTotalExperience']],orient='h')
plt.subplot(2,1,2)
sns.boxplot(data=std_df.loc[:,['blueTotalGold','blueTotalExperience','redTotal
Gold','redTotalExperience']],orient='h')
```

Out[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2bc0b509ba8>
```



In [13]:
```
plt.subplot(2,1,1)
sns.boxplot(data=norm_df.loc[:,['blueGoldDiff','blueExperienceDiff','blueTotalMinionsKilled',
                                'redTotalMinionsKilled']],orient='h')
plt.subplot(2,1,2)
sns.boxplot(data=std_df.loc[:,['blueGoldDiff','blueExperienceDiff','blueTotalMinionsKilled',
                               'redTotalMinionsKilled']],orient='h')
```

Out[13]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2bc0b390630>
```



In [ ]: