stocks\_list = ['INTC','MSFT', 'CSCO', 'AAPL', 'AMZN', 'GOOG', 'JNPR', 'VZ', 'T', 'S', 'TMUS']

result:

base on Euclidean distance (index start with 1)

the longest distant is between[9, 6]

the shortest distant is between[11, 10]

according to cov (index start with 0)

the most similar by cov is (1, 5)

the least similar by cov is (5, 8)

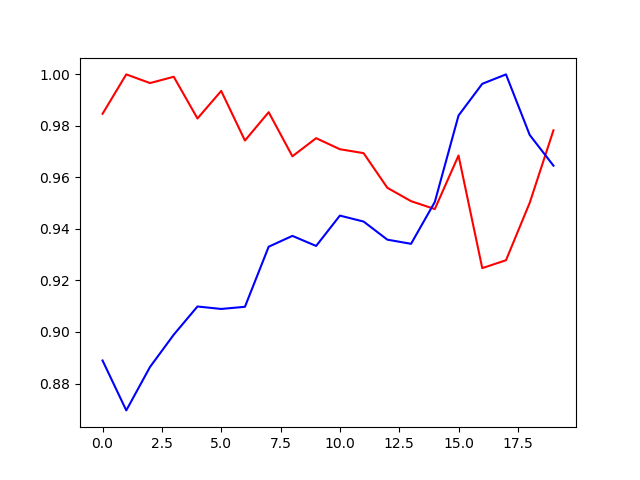
according to corrcoef (index start with 0)

the most similar by corrcoef is (1, 5)

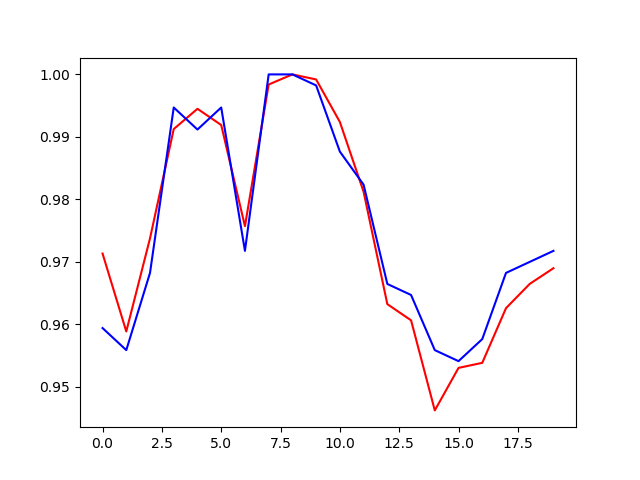
the least similar by corrcoef is (8, 1)

Q1:

According to Euclidean (index start with 1) the longest distant is between[9, 6] that means GOOG and T displaying the least similar behavior. The shortest distance is between [11,10] that means S and TMUS playing the most similar behavior.



Least similar stocks



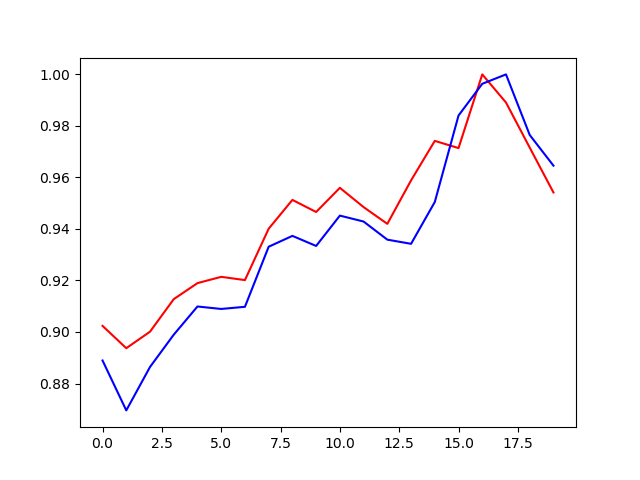
Most similar stocks

Q2:

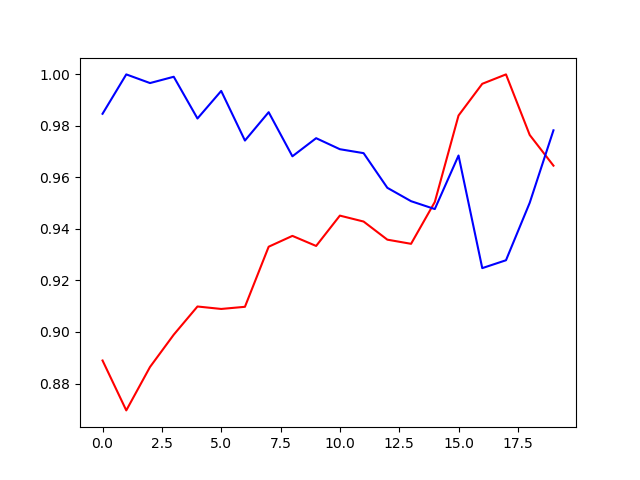
according to cov (index start with 0)

the most similar stocks by cov are CSCO and JNPR

the least similar stocks by cov are JNPR and T



Max cov

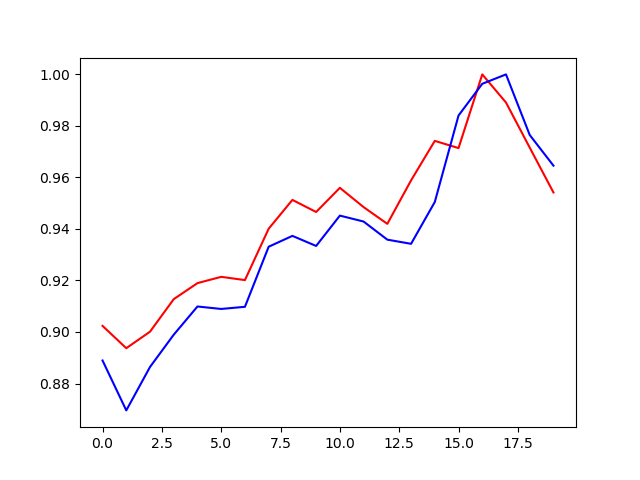


Min cov

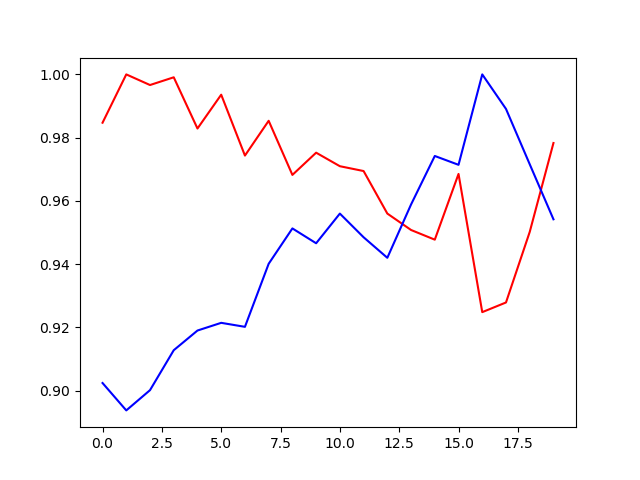
according to corrcoef (index start with 0)

the most similar stocks by corrcoef are MSFT and GOOG

the least similar stocks by corrcoef are T and MSFT



Max corrcoef

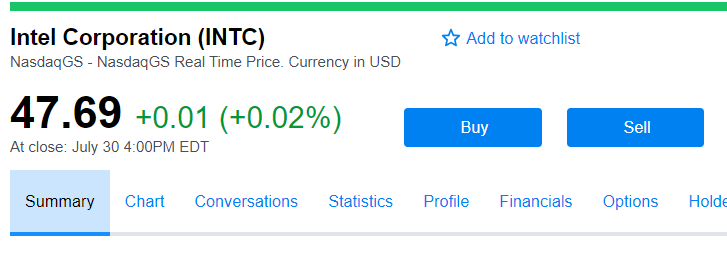


Min corrcoef

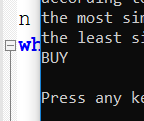
Compared with Euclidean, cov romve the means. Furthermore, corrcoef eliminate the influence from the variance. Thus the results of these 3 algorithm are different.

Q3:

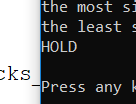
I fetch the real-time price of INTC



And my function told to buy



However, if I increase the percentages to 10%, the prediction turns out to be much more conservative.



Finally, pictures:

