

import pandas as pd

import numpy as np

#rtn\_monthly = pd.read\_excel('A\_trend\_factor.xlsm',sheetname='월별수익률1')

#rtn\_monthly\_notgross = pd.read\_excel('A\_trend\_factor.xlsm',sheetname='월별수익률2')

#rtn\_monthly\_notgross.to\_pickle('rtn\_monthly\_notgross')

#monthly\_date = pd.read\_excel('A\_trend\_factor.xlsm',sheetname='월말날짜1',header=None)

#rtn\_daily1.to\_pickle('rtn\_daily1')

#rtn\_monthly.to\_pickle('rtn\_monthly')

rtn\_monthly\_notgross = pd.read\_pickle('rtn\_monthly\_notgross') #수익률을 gross로 하지 말아야 하나?

monthly\_date = pd.read\_pickle('monthly\_date')

rtn\_monthly = pd.read\_pickle('rtn\_monthly') # gross 수익률

rtn\_daily = pd.read\_pickle('rtn\_daily')

higher\_return\_final = pd.DataFrame(np.zeros((1,134)))

lower\_return\_final = pd.DataFrame(np.zeros((1,134)))

rtn\_monthly = rtn\_monthly\_notgross

for n in range(12,146):

beta\_3\_temp = pd.DataFrame(np.zeros((1,12)))

for i in range(0,12):

#df.columns.get\_loc() 이걸 하면 column 위치를 알 수 있다!!

rebalancing\_date\_column=rtn\_daily.columns.get\_loc(monthly\_date.iloc[0,n-i])

#3일 moving\_average를 마지막 가격으로 나누워준 값

# iloc을 이용해서 column 범위를 구하면 원하는 column +1을 해줘야하네

ma\_3=pd.DataFrame(np.sum(rtn\_daily.iloc[:,rebalancing\_date\_column-2:rebalancing\_date\_column+1],axis=1)/3/rtn\_daily.iloc[:,rebalancing\_date\_column])

ma\_3=ma\_3[ma\_3[0].notnull()]

future\_rtn = pd.DataFrame(rtn\_monthly.iloc[:,n-i+1])

# future\_rtn = pd.DataFrame(rtn\_daily.loc[:,monthly\_date.iloc[0,n+1-i]])

future\_rtn = future\_rtn[future\_rtn[monthly\_date.iloc[0,n+1-i]].notnull()]

ma\_3\_temp = pd.concat([ma\_3,future\_rtn],axis=1)

ma\_3\_temp = ma\_3\_temp.assign(product=ma\_3\_temp.iloc[:,0]\*ma\_3\_temp.iloc[:,1])

ma\_3\_temp = ma\_3\_temp[ma\_3\_temp['product'].notnull()]

beta\_3 = pd.DataFrame(np.dot(np.linalg.inv(np.dot(ma\_3.T,ma\_3)),np.dot(ma\_3\_temp[0].T,ma\_3\_temp.iloc[:,1])))

beta\_3\_temp.iloc[0,i]=beta\_3.iloc[0,0]

beta\_3=np.average(beta\_3\_temp)

#다음기 ma랑 곱해

ma\_3=pd.DataFrame(np.sum(rtn\_daily.iloc[:,rebalancing\_date\_column-1:rebalancing\_date\_column+2],axis=1)/3/rtn\_daily.iloc[:,rebalancing\_date\_column+1])

ma\_3=ma\_3[ma\_3[0].notnull()]

return\_3=beta\_3 \* ma\_3

이런식으로 3 5 10 20 50 100 200 400 600 800 1000 각각 구해서 final\_return=return\_3+return\_5+return\_10+return\_20+return\_50+return\_100+return\_200+return\_400+return\_600+return\_800+return\_1000

final\_return= pd.concat([final\_return,pd.DataFrame(rtn\_monthly.iloc[:,n+1])],axis=1, join\_axes=[final\_return.index])

final\_return = final\_return[final\_return.iloc[:,1].notnull()]

final\_return = final\_return.assign(rnk=final\_return.iloc[:,0].rank(method='first',ascending=False))

rtn\_min=np.percentile(final\_return['rnk'],20)

rtn\_max=np.percentile(final\_return['rnk'],80)

higher\_return = final\_return[final\_return['rnk']<rtn\_min]

lower\_return = final\_return[final\_return['rnk']>rtn\_max]

higher\_return\_final[n-12] = np.average(higher\_return.iloc[:,1])

lower\_return\_final[n-12] = np.average(lower\_return.iloc[:,1])