

### 3.signal-rolling-agent

September 29, 2021

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

```
[2]: df = pd.read_csv('../dataset/G00G-year.csv')
df.head()
```

```
[2]:
```

	Date	Open	High	Low	Close	Adj Close	\
0	2016-11-02	778.200012	781.650024	763.450012	768.700012	768.700012	
1	2016-11-03	767.250000	769.950012	759.030029	762.130005	762.130005	
2	2016-11-04	750.659973	770.359985	750.560974	762.020020	762.020020	
3	2016-11-07	774.500000	785.190002	772.549988	782.520020	782.520020	
4	2016-11-08	783.400024	795.632996	780.190002	790.510010	790.510010	

	Volume
0	1872400
1	1943200
2	2134800
3	1585100
4	1350800

```
[3]: def buy_stock(
    real_movement,
    delay = 5,
    initial_state = 1,
    initial_money = 10000,
    max_buy = 1,
    max_sell = 1,
):
    """
    real_movement = actual movement in the real world
    delay = how much interval you want to delay to change our decision from buy_
    →to sell, vice versa
    initial_state = 1 is buy, 0 is sell
    initial_money = 1000, ignore what kind of currency
```

```

max_buy = max quantity for share to buy
max_sell = max quantity for share to sell
"""

starting_money = initial_money
delay_change_decision = delay
current_decision = 0
state = initial_state
current_val = real_movement[0]
states_sell = []
states_buy = []
current_inventory = 0

def buy(i, initial_money, current_inventory):
    shares = initial_money // real_movement[i]
    if shares < 1:
        print(
            'day %d: total balances %f, not enough money to buy a unit, '
↪price %f'
            % (i, initial_money, real_movement[i])
        )
    else:
        if shares > max_buy:
            buy_units = max_buy
        else:
            buy_units = shares
        initial_money -= buy_units * real_movement[i]
        current_inventory += buy_units
        print(
            'day %d: buy %d units at price %f, total balance %f'
            % (i, buy_units, buy_units * real_movement[i], initial_money)
        )
        states_buy.append(0)
    return initial_money, current_inventory

if state == 1:
    initial_money, current_inventory = buy(
        0, initial_money, current_inventory
    )

for i in range(1, real_movement.shape[0], 1):
    if real_movement[i] < current_val and state == 0:
        if current_decision < delay_change_decision:
            current_decision += 1
        else:
            state = 1
            initial_money, current_inventory = buy(
                i, initial_money, current_inventory
            )

```

```

        )
        current_decision = 0
        states_buy.append(i)
    if real_movement[i] > current_val and state == 1:
        if current_decision < delay_change_decision:
            current_decision += 1
        else:
            state = 0

    if current_inventory == 0:
        print('day %d: cannot sell anything, inventory 0' % (i))
    else:
        if current_inventory > max_sell:
            sell_units = max_sell
        else:
            sell_units = current_inventory
        current_inventory -= sell_units
        total_sell = sell_units * real_movement[i]
        initial_money += total_sell
        try:
            invest = (
                (real_movement[i] - real_movement[states_buy[-1]])
                / real_movement[states_buy[-1]]
            ) * 100
        except:
            invest = 0
        print(
            'day %d, sell %d units at price %f, investment %f %%,  

↪total balance %f,'
            % (i, sell_units, total_sell, invest, initial_money)
        )

        current_decision = 0
        states_sell.append(i)
    current_val = real_movement[i]
    invest = ((initial_money - starting_money) / starting_money) * 100
    total_gains = initial_money - starting_money
    return states_buy, states_sell, total_gains, invest

```

```

[4]: states_buy, states_sell, total_gains, invest = buy_stock(df.Close,  

↪initial_state = 1,
delay = 4,  

↪initial_money = 10000)

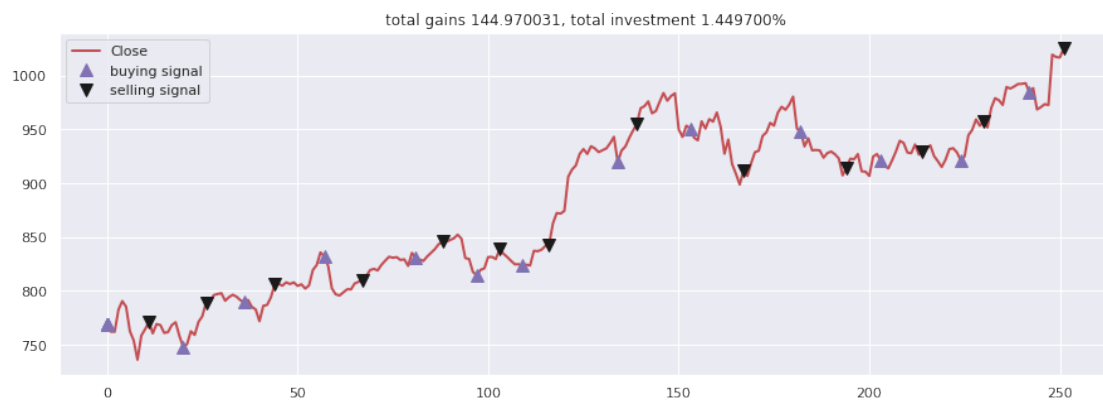
```

day 0: buy 1 units at price 768.700012, total balance 9231.299988  
day 11, sell 1 units at price 771.229980, investment 0.329123 %, total balance 10002.529968,

day 20: buy 1 units at price 747.919983, total balance 9254.609985  
 day 26, sell 1 units at price 789.289978, investment 5.531340 %, total balance 10043.899963,  
 day 36: buy 1 units at price 789.909973, total balance 9253.989990  
 day 44, sell 1 units at price 806.150024, investment 2.055937 %, total balance 10060.140014,  
 day 57: buy 1 units at price 832.150024, total balance 9227.989990  
 day 67, sell 1 units at price 809.559998, investment -2.714658 %, total balance 10037.549988,  
 day 81: buy 1 units at price 830.630005, total balance 9206.919983  
 day 88, sell 1 units at price 845.539978, investment 1.795020 %, total balance 10052.459961,  
 day 97: buy 1 units at price 814.429993, total balance 9238.029968  
 day 103, sell 1 units at price 838.549988, investment 2.961580 %, total balance 10076.579956,  
 day 109: buy 1 units at price 823.349976, total balance 9253.229980  
 day 116, sell 1 units at price 843.190002, investment 2.409671 %, total balance 10096.419982,  
 day 134: buy 1 units at price 919.619995, total balance 9176.799987  
 day 139, sell 1 units at price 954.960022, investment 3.842895 %, total balance 10131.760009,  
 day 153: buy 1 units at price 950.760010, total balance 9180.999999  
 day 167, sell 1 units at price 911.710022, investment -4.107239 %, total balance 10092.710021,  
 day 182: buy 1 units at price 947.799988, total balance 9144.910033  
 day 194, sell 1 units at price 914.390015, investment -3.525002 %, total balance 10059.300048,  
 day 203: buy 1 units at price 921.280029, total balance 9138.020019  
 day 214, sell 1 units at price 929.080017, investment 0.846647 %, total balance 10067.100036,  
 day 224: buy 1 units at price 920.969971, total balance 9146.130065  
 day 230, sell 1 units at price 957.789978, investment 3.997960 %, total balance 10103.920043,  
 day 242: buy 1 units at price 984.450012, total balance 9119.470031  
 day 251, sell 1 units at price 1025.500000, investment 4.169840 %, total balance 10144.970031,

```

[5]: close = df['Close']
fig = plt.figure(figsize = (15,5))
plt.plot(close, color='r', lw=2.)
plt.plot(close, '^', markersize=10, color='m', label = 'buying signal',
    ↳markevery = states_buy)
plt.plot(close, 'v', markersize=10, color='k', label = 'selling signal',
    ↳markevery = states_sell)
plt.title('total gains %f, total investment %f%%'%(total_gains, invest))
plt.legend()
plt.show()
  
```



[ ]: