추천 종목을 파일로 저장

```
import FinanceDataReader as fdr
import yfinance as yf
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
import numpy as np
import datetime
import pickle
```

알고리즘으로 추천받은 종목을 딕셔너리로 저장한 후, pickle 파일로 변환하겠습니다. 저장된 pickle 파일을 읽어서 자동매매를 진행합니다.

```
def select_stocks(today_dt):

    today = datetime.datetime.strptime(today_dt, '%Y-%m-%d')
    start_dt = today - datetime.timedelta(days=100) # 100 일전 데이터 부터 시작 - 피쳐 엔지니어링은 최소 60 개의 일봉이 필요함
    print(start_dt, today_dt)

    kosdaq_list = pd.read_pickle('kosdaq_list.pkl')

    price_data = pd.DataFrame()

    for code, name in zip(kosdaq_list['code'], kosdaq_list['name']):  # 코스닥 모든 종목에서 대하여 반복
        daily_price = fdr.DataReader(code, start = start_dt, end = today_dt) # 종목, 일봉, 데이터 갯수

        daily_price['code'] = code
        daily_price['name'] = name
        price_data = pd.concat([price_data, daily_price], axis=0)

    price_data.index.name = 'date'
    price_data.columns= price_data.columns.str.lower() # 컬럼 이름 소문자로 변경

    # DataReder 코스닥 인덱스 조회 실패시, 야후파이낸스로 추출
    # kosdaq_index = fdr.DataReader('KQ11', start = start_dt, end = today_dt) # 데이터 호출
    # kosdaq_index.columns = ['close','open','high','low','volume','change'] # 컬럼명 변경

    kosdaq_index =  yf.download('^KQ11', start = start_dt)
    kosdaq_index.columns = ['open','high','low','close','adj_close','volume'] # 컬럼명 변경
    kosdaq_index.index.name='date' # 인덱스 이름 생성
    kosdaq_index.sort_index(inplace=True) # 인덱스(날짜) 로 정렬
    kosdaq_index['kosdaq_return'] = kosdaq_index['close']/kosdaq_index['close'].shift(1) # 수익율 : 전 날 종가대비 당일 종가

    merged = price_data.merge(kosdaq_index['kosdaq_return'], left_index=True, right_index=True, how='left')

    return_all = pd.DataFrame()

    for code in kosdaq_list['code']:

        stock_return = merged[merged['code']==code].sort_index()
        stock_return['return'] = stock_return['close']/stock_return['close'].shift(1) # 종목별 전일 종가 대비 당일 종가 수익율
        c1 = (stock_return['kosdaq_return'] < 1) # 수익율 1 보다 작음. 당일 종가가 전일 종가보다 낮음 (코스닥 지표)
        c2 = (stock_return['return'] > 1) # 수익율 1 보다 큼. 당일 종가가 전일 종가보다 큼 (개별 종목)
        stock_return['win_market'] = np.where((c1&c2), 1, 0) # C1 과 C2 조건을 동시에 만족하면 1, 아니면 0
        return_all = pd.concat([return_all, stock_return], axis=0)

    return_all.dropna(inplace=True)

    model_inputs = pd.DataFrame()

    for code, name, sector in zip(kosdaq_list['code'], kosdaq_list['name'], kosdaq_list['sector']):

        data = return_all[return_all['code']==code].sort_index().copy()

        # 가격변동성이 크고, 거래량이 몰린 종목이 주가가 상승한다
        data['price_mean'] = data['close'].rolling(20).mean()
        data['price_std'] = data['close'].rolling(20).std(ddof=0)
        data['price_z'] = (data['close'] - data['price_mean'])/data['price_std']
        data['volume_mean'] = data['volume'].rolling(20).mean()
        data['volume_std'] = data['volume'].rolling(20).std(ddof=0)
        data['volume_z'] = (data['volume'] - data['volume_mean'])/data['volume_std']

        # 위꼬리가 긴 양봉이 자주발생한다.
        data['positive_candle'] = (data['close'] > data['open']).astype(int) # 양봉
        data['high/close'] = (data['positive_candle']==1)*(data['high']/data['close'] > 1.1).astype(int) # 양봉이면서 고가가 종가보다 높게 위치
        data['num_high/close'] =  data['high/close'].rolling(20).sum()
        data['long_candle'] = (data['positive_candle']==1)*(data['high']==data['close'])*\
        (data['low']==data['open'])*(data['close']/data['open'] > 1.2).astype(int) # 장대 양봉을 데이터로 표현
        data['num_long'] =  data['long_candle'].rolling(60).sum() # 지난 20 일 동안 장대양봉의 갯 수
```

```
        # 거래량이 종종 터지며 매집의 흔적을 보인다
        data['volume_mean'] = data['volume'].rolling(60).mean()
        data['volume_std'] = data['volume'].rolling(60).std()
        data['volume_z'] = (data['volume'] - data['volume_mean'])/data['volume_std']
# 거래량은 종목과 주가에 따라 다르기 때문에 표준화한 값이 필요함
        data['z>1.96'] = (data['close'] > data['open'])*(data['volume_z'] >
1.65).astype(int) # 양봉이면서 거래량이 90%신뢰구간을 벗어난 날
        data['num_z>1.96'] =  data['z>1.96'].rolling(60).sum()  # 양봉이면서 거래량이
90% 신뢰구간을 벗어난 날을 카운트

        # 주가지수보다 더 좋은 수익율을 보여준다
        data['num_win_market'] = data['win_market'].rolling(60).sum() # 주가지수 수익
율이 1 보다 작을 때, 종목 수익율이 1 보다 큰 날 수
        data['pct_win_market'] =
(data['return']/data['kosdaq_return']).rolling(60).mean() # 주가지수 수익율 대비 종목
수익율



        # 동종업체 수익률보다 더 좋은 수익율을 보여준다.
        data['return_mean'] = data['return'].rolling(60).mean() # 종목별 최근 60 일 수
익율의 평균
        data['sector'] = sector
        data['name'] = name

        data = data[(data['price_std']!=0) & (data['volume_std']!=0)]

        model_inputs = pd.concat([data, model_inputs], axis=0)

    model_inputs['sector_return'] = model_inputs.groupby(['sector',
model_inputs.index])['return'].transform(lambda x: x.mean()) # 섹터의 평균 수익율 계산
    model_inputs['return over sector'] =
(model_inputs['return']/model_inputs['sector_return']) # 섹터 평균 수익률 대비 종목 수
익률 계산
    model_inputs.dropna(inplace=True) # Missing 값 있는 행 모두 제거


    feature_list =
['price_z','volume_z','num_high/close','num_win_market','pct_win_market','return over
sector']

    X = model_inputs.loc[today_dt][['code','name','return','kosdaq_return','close'] +
feature_list].set_index('code')

    with open("gam.pkl", "rb") as file:
        gam = pickle.load(file)

    yhat = gam.predict_proba(X[feature_list])
    X['yhat'] = yhat

    tops = X[X['yhat'] >= 0.3].sort_values(by='yhat', ascending=False) # 스코어 0.3
이상 종목만
    print(len(tops))

    select_tops = tops[(tops['return'] > 1.03) & (tops['price_z'] < 0)]
[['name','return','price_z','yhat', 'kosdaq_return','close']]  # 기본 필터링 조건

    if len(select_tops) > 1: # 최소한 2개 종목 - 추천 리스크 분산
        return select_tops

    else:
        return None
```

```
select_tops = select_stocks('2022-06-16').sort_values(by='yhat',
ascending=False).head(5)
```

```
2022-03-08 00:00:00 2022-06-16
[**********************100%**********************]  1 of 1 completed
395
```

```
select_tops.style.set_table_attributes('style="font-size: 12px"').format(precision=3)
```

code	name	return	price_z	yhat	kosdaq_return	close
002680	한탑	1.031	-1.282	0.457	1.003	3155
177350	베셀	1.052	-1.346	0.413	1.003	7630
312610	에이에프더블류	1.035	-1.967	0.397	1.003	3505
096870	엘디티	1.082	-1.330	0.381	1.003	4145
311390	네오크레마	1.076	-2.141	0.377	1.003	13450

```
select_dict = {}
for code in list(select_tops.index):
    s = results.loc[code]
    select_dict[code] = [s['name'], s['close']]
```

```
select_dict # 결과 확인
```

```
{'002680': ['한탑', 3155],
 '177350': ['베셀', 7630],
 '312610': ['에이에프더블류', 3505],
 '096870': ['엘디티', 4145],
 '311390': ['네오크레마', 13450]}
```

pickle 파일로 딕셔너리를 저장합니다.

```
# 피클파일로 저장
f = open("select_dict.pkl","wb")
pickle.dump(select_dict, f)
f.close()
```

저장된 피클 파일을 Load 후 원본 디셔너리 파일과 동일한지 확인해 봅니다.

```
f = open("select_dict.pkl","rb")
select_dict = pickle.load(f)
f.close

# 결과 확인
select_dict
```

```
{'002680': ['한탑', 3155],
 '177350': ['베셀', 7630],
 '312610': ['에이에프더블류', 3505],
 '096870': ['엘디티', 4145],
 '311390': ['네오크레마', 13450]}
```

By KHS

© Copyright 2022.