data_preprocessing

August 21, 2023

[1]:
```
import pandas as pd
import numpy as np
import datetime
import yfinance as yf

from finrl.meta.preprocessor.yahoodownloader import YahooDownloader
from finrl.meta.preprocessor.preprocessors import FeatureEngineer, data_split
from finrl import config_tickers
from finrl.config import INDICATORS

import itertools
```

1 Part 2. Fetch data

[2]:
```
aapl_df_yf = yf.download(tickers = "aapl", start='2020-01-01', end='2020-01-31')
```

```
[*********************100%***********************]  1 of 1 completed
```

[3]:
```
aapl_df_yf.head()
```

[3]:
```
                 Open       High        Low      Close  Adj Close     Volume
Date
2020-01-02  74.059998  75.150002  73.797501  75.087502  73.347931  135480400
2020-01-03  74.287498  75.144997  74.125000  74.357498  72.634850  146322800
2020-01-06  73.447502  74.989998  73.187500  74.949997  73.213615  118387200
2020-01-07  74.959999  75.224998  74.370003  74.597504  72.869308  108872000
2020-01-08  74.290001  76.110001  74.290001  75.797501  74.041489  132079200
```

[4]:
```
aapl_df_finrl = YahooDownloader(start_date = '2020-01-01',
                                end_date = '2020-01-31',
                                ticker_list = ['aapl']).fetch_data()
```

```
[*********************100%***********************]  1 of 1 completed
Shape of DataFrame:  (20, 8)
```

[5]:
```
aapl_df_finrl.head()
```

```
[5]:          date       open       high        low      close     volume   tic  \
     0  2020-01-02  74.059998  75.150002  73.797501  73.347939  135480400  aapl
     1  2020-01-03  74.287498  75.144997  74.125000  72.634842  146322800  aapl
     2  2020-01-06  73.447502  74.989998  73.187500  73.213608  118387200  aapl
     3  2020-01-07  74.959999  75.224998  74.370003  72.869293  108872000  aapl
     4  2020-01-08  74.290001  76.110001  74.290001  74.041489  132079200  aapl

        day
     0    3
     1    4
     2    0
     3    1
     4    2
```

1.1 Data for the chosen tickers

```
[6]: config_tickers.DOW_30_TICKER
```

```
[6]: ['AXP',
      'AMGN',
      'AAPL',
      'BA',
      'CAT',
      'CSCO',
      'CVX',
      'GS',
      'HD',
      'HON',
      'IBM',
      'INTC',
      'JNJ',
      'KO',
      'JPM',
      'MCD',
      'MMM',
      'MRK',
      'MSFT',
      'NKE',
      'PG',
      'TRV',
      'UNH',
      'CRM',
      'VZ',
      'V',
      'WBA',
      'WMT',
      'DIS',
 'DOW']
```

```
[7]: TRAIN_START_DATE = '2009-01-01'
     TRAIN_END_DATE = '2020-07-01'
     TRADE_START_DATE = '2020-07-01'
     TRADE_END_DATE = '2021-10-29'
```

```
[8]: df_raw = YahooDownloader(start_date = TRAIN_START_DATE,
                              end_date = TRADE_END_DATE,
                              ticker_list = config_tickers.DOW_30_TICKER).fetch_data()
```

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
Shape of DataFrame:  (94301, 8)
```

```
[9]: df_raw.head()
```

```
[9]:          date      open      high       low     close     volume   tic  \
     0  2009-01-02  3.067143  3.251429  3.041429  2.754725  746015200  AAPL
```

```
1  2009-01-02  58.590000  59.080002  57.750000  43.422924    6547900  AMGN
2  2009-01-02  18.570000  19.520000  18.400000  15.256276   10955700   AXP
3  2009-01-02  42.799999  45.560001  42.779999  33.941105    7010200    BA
4  2009-01-02  44.910000  46.980000  44.709999  31.254070    7117200   CAT

   day
0    4
1    4
2    4
3    4
4    4
```

2 Part 3: Preprocess Data

[10]:
```
fe = FeatureEngineer(use_technical_indicator=True,
                     tech_indicator_list = INDICATORS,
                     use_vix=True,
                     use_turbulence=True,
                     user_defined_feature = False)

processed = fe.preprocess_data(df_raw)
```

```
Successfully added technical indicators
[*********************100%***********************]  1 of 1 completed
Shape of DataFrame:  (3228, 8)
Successfully added vix
Successfully added turbulence index
```

[11]:
```
list_ticker = processed["tic"].unique().tolist()
list_date = list(pd.date_range(processed['date'].min(),processed['date'].max()).
 ↪astype(str))
combination = list(itertools.product(list_date,list_ticker))

processed_full = pd.DataFrame(combination,columns=["date","tic"]).
 ↪merge(processed,on=["date","tic"],how="left")
processed_full = processed_full[processed_full['date'].isin(processed['date'])]
processed_full = processed_full.sort_values(['date','tic'])

processed_full = processed_full.fillna(0)
```

[12]:
```
processed_full.head()
```

[12]:
```
         date   tic       open       high        low      close       volume  \
0  2009-01-02  AAPL   3.067143   3.251429   3.041429   2.754725  746015200.0
1  2009-01-02  AMGN  58.590000  59.080002  57.750000  43.422924    6547900.0
2  2009-01-02   AXP  18.570000  19.520000  18.400000  15.256276   10955700.0
```

```
3  2009-01-02    BA  42.799999  45.560001  42.779999  33.941105    7010200.0
4  2009-01-02   CAT  44.910000  46.980000  44.709999  31.254070    7117200.0

   day  macd   boll_ub   boll_lb  rsi_30     cci_30  dx_30  close_30_sma  \
0  4.0   0.0  2.977272  2.648437   100.0  66.666667  100.0      2.754725
1  4.0   0.0  2.977272  2.648437   100.0  66.666667  100.0     43.422924
2  4.0   0.0  2.977272  2.648437   100.0  66.666667  100.0     15.256276
3  4.0   0.0  2.977272  2.648437   100.0  66.666667  100.0     33.941105
4  4.0   0.0  2.977272  2.648437   100.0  66.666667  100.0     31.254070

   close_60_sma        vix  turbulence
0      2.754725  39.189999         0.0
1     43.422924  39.189999         0.0
2     15.256276  39.189999         0.0
3     33.941105  39.189999         0.0
4     31.254070  39.189999         0.0
```

3 Part 4: Save the Data

3.0.1 Split the data for training and trading

[13]:
```python
train = data_split(processed_full, TRAIN_START_DATE,TRAIN_END_DATE)
trade = data_split(processed_full, TRADE_START_DATE,TRADE_END_DATE)
print(len(train))
print(len(trade))
```

```
83897
9715
```

3.0.2 Save data to csv file

[14]:
```python
train.to_csv('train_data.csv')
trade.to_csv('trade_data.csv')
```

[ ]:
```python
```