# 第6课: Pandas 高阶操作

# 目录

- . 6.1 批量导入数据
- · 6.2 HDF 存取数据
- · 6.3 转变 K 线数据周期
- <u>6.4 groupby 分组</u>

# 6.1 批量导入数据

### 遍历文件

- import os
- pd.set\_option('expand\_frame\_repr', False) # 当列太多时不换行
- · 系统自带函数 os.walk,用于遍历文件夹中的所有文件, os 是 python 自带的系统库。

```
m coin_quant_class to program to class6 🐉 1_如何批量符入数据.py
                                                                                                   如何批量等入数据 🕶 🕨 敵
        pu.sct_uption( expand_frame_fept , facse) # =79A\SH9
          # =====导入EOSUSD每一天的1分钟数据
        # df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/class6/BITFINEX/EOSUSD/BI'
                               skiprows=1,
                               parse_dates=['candle_begin_time'])
        # =====批量导入EOSUSD所有天的一分钟数据
        # 系统自带函数os.walk,用于遍历文件夹中的所有文件,os是python自带的系统库
        for root, dirs, files in os.walk('/Users/jxing/Desktop/coin_quant_class/data'):
             #「OOL拥山文件类,GITS捆山TOOL下所有的文件类,TILES捆山TOOL下的所有的文件
            print('root:', root)
            print('dirs:', dirs)
            print('files:', files)
            print()
     /Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/1_如何批量导入数据.py
         candle_begin_time open
                                  high
                                          low close
                                                           volume
                                1.20000
       2017-07-01 16:59:00 1.20000
2017-07-01 17:01:00 1.40000
                                                        2366.619400
                                       1.20000
                                               1.20000
                                1.40000
                                        1.39000
                                               1.39000
                                                        1130.640508
      2017-07-01 17:02:00 1.10000
                                1.25000
                                        1.10000
                                               1.25000
                                                        234.247511
                         1.20000
        2017-07-01 17:03:00
                                1.25000
                                        1.10000
                                               1.11000
                                                       2224.674241
                                1.10000
        2017-07-01 17:04:00
                                        1.00000
                                                1.00000
                                                        4301.678248
        2017-07-01 17:05:00 1.00000
                                1.00000
                                        1.00000
                                                1.00000
                                                        400.000000
        2017-07-01 17:06:00
                         1.08000
                                 1.08000
                                                1.00000
                                        0.70000
        2017-07-01 17:07:00
                         1.04000
                                 1.04000
                                        0.50000
                                               0.55000
                                                       24512.682493
        2017-07-01 17:08:00 0.90000
     8
                                0.90000
                                       0.90000
                                               0.90000
                                                        300.000000
        2017-07-01 17:09:00
                         0.90000
                                0.90000
                                        0.90000
                                               0.90000
     19 7917-97-91 17-19-99
                                        0 60000
                                a ggaaa
                                               а отава
```

- for root, dirs, files in os.walk('/Users/jxing/Desktop/coin\_quant\_class/data'):
- root 输出文件夹, dirs 输出 root 下所有的文件夹, files 输出 root 下的所有的文件。
- print('root:', root)
- print('dirs:', dirs)
- print('files:', files)

```
m coin_quant_class Di program Di class6 🐉 1 如何批量符入数据 py
                                 parse_dates=['candle_begin_time'])
             # =====批量导入EOSUSD所有天的一分钟数据
             # 系统自带函数os.walk,用于遍历文件夹中的所有文件,os是python自带的系统库
             # 演示os.walk
             for root, dirs, files in os.walk('/Users/jxing/Desktop/coin_guant_class/data'):
                 # root输出文件夹,dirs输出root下所有的文件夹,files输出root下的所有的文件
                 print('root:', root)
                 print('dirs:', dirs)
              print('files:', files)
                 print()
             for root, dirs, files in os wal
    /Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/1
    root: /Users/jxing/Desktop/coin_quant_class/data
    dirs: ['class5', 'class6']
    files: ['.DS_Store']
    root: /Users/jxing/Desktop/coin_quant_class/data/class5
    dirs: []
    files: ['.DS_Store', 'BITFINEX_BTCUSD_20180124_1T.csv', 'BITFINEX-1H-data-20180124.csv']
    root: /Users/jxing/Desktop/coin_quant_class/data/class6
    dirs: ['BITFINEX']
    files: ['.DS_Store']
    root: /Users/jxing/Desktop/coin quant class/data/class6/BITFINEX
m coin_quant_class Im program Im class6 ( 1_如何批量符入数据 p
                                                                                 如何批量等入数据 🕶 🕨 敬
   # =====批量导入EOSUSD所有天的一分钟数据
             # 系统自带函数os.walk,用于遍历文件夹中的所有文件,os是python自带的系统库
             # 演示os.walk
             for root, dirs, files in os.walk('/Users/jxing/Desktop/coin_quant_class/data'):
                 # root输出文件夹, dirs输出root下所有的文件夹, files输出root下的所有的文件
                 print('root:', root)
                 print('dirs:', dirs)
                 nrint('files:'
    root: /Users/jxing/Desktop/coin_quant_class/data/class6
    dirs: ['BITFINEX']
    files: ['.DS_Store']
    root: /Users/jxing/Desktop/coin_quant_class/data/class6/BITFINEX
    dirs: ['EOSUSD']
    files: ['.DS_Store']
    root: /Users/jxing/Desktop/coin_quant_class/data/class6/BITFINEX/EOSUSD
    dirs: []
    files: ['BITFINEX_EOSUSD_20170704_1T.csv', 'BITFINEX_EOSUSD_20170811_1T.csv', 'BITFIN
    Process finished with exit code 0
```

## 批量读取文件名称

- file\_list = []
- for root, dirs, files in os.walk('/Users/jxing/Desktop/coin\_quant\_class/data/class6'):
- · 当 files 不为空的时候
- if files:
- print(files)

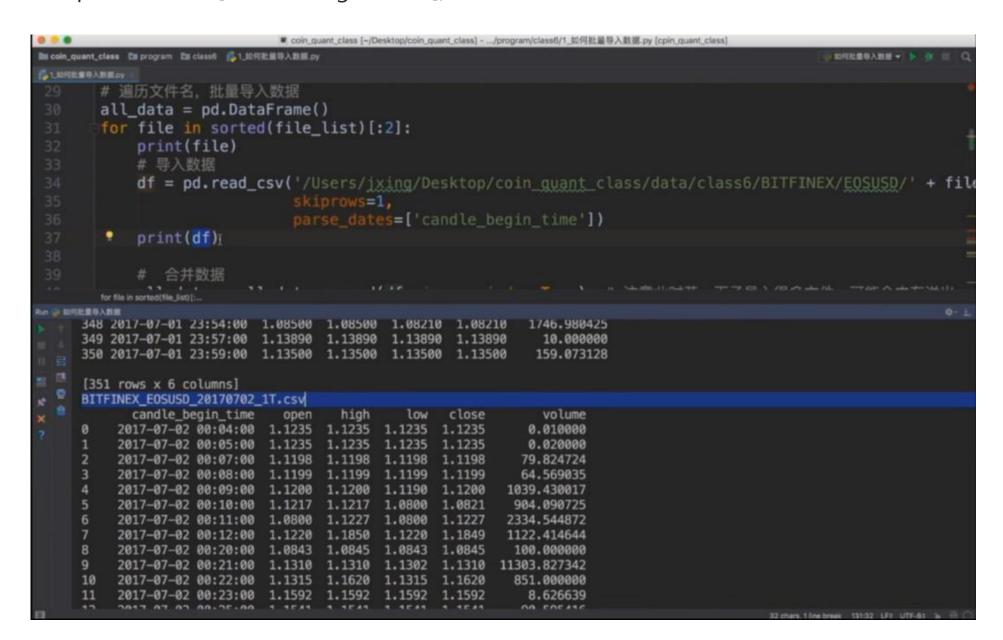
```
た知何批業等入数据のy
     # 批量读取文件名称
     file_list = []
     for root, dirs, files in os.walk('/Users/jxing/Desktop/coin_quant_class/data/class6'):
         # 当files不为空的时候
         if files:
             print(files)
             # for f in files:
                  if f.endswith('.csv'):
                      file_list.append(f)
     exit()
    /Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_c
    ['.DS_Store']
    ['.DS_Store']
    ['BITFINEX_EOSUSD_20170704_1T.csv', 'BITFINEX_EOSUSD_20170811_1T.csv', 'BITF
    Process finished with exit code 0
```

- for f in files:
- if f.endswith('.csv'):
- file\_list.append(f)

```
如何批量等入数据 🕶 🕨
     # 批量读取文件名称
      file_list = []
      for root, dirs, files in os.walk('/Users/ixing/Desktop/coin_quant_class/data/class6'):
         # 当files不为空的时候
         if files:
             for f in files:
                 if f.endswith('.csv'):
                     file list.append(f)
      exit()
    /Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_c
    ['.DS_Store']
    ['.DS_Store']
    ['BITFINEX_EOSUSD_20170704_1T.csv', 'BITFINEX_EOSUSD_20170811_1T.csv', 'BITF
    Process finished with exit code 0
Im coin_quant_class Im program Im class6 [51]如何批量符入数据.py
                                                                               如何社童等入数据 *
1、知何社最等入数据 py
            print()
      # 批量读取文件名称
      file_list = []
      for root, dirs, files in os.walk('/Users/jxing/Desktop/coin_quant_class/data/class6'
          # 当files不为空的时候
          if files:
              for f in files:
                  if f.endswith('.csv'):
                      file_list.append(f)
      print(file_list)
    envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class@
    0704_1T.csv', 'BITFINEX_EOSUSD_20170811_1T.csv', 'BITFINEX_EOSUSD_201709
    exit code 0
```

#### 遍历文件名,批量导入数据

- all\_data = pd.DataFrame()
- for file in sorted(file\_list):
- print(file)
- 导入数据
- df = pd.read\_csv('/Users/jxing/Desktop/coin\_quant\_class/data/class6/BITFINEX/EOSUSD/'
   + file,
- skiprows=1,
- parse\_dates=['candle\_begin\_time'])



- 合并数据
- all\_data = all\_data.append(df, ignore\_index=True) # 注意此时若一下子导入很多文件,可能会内存溢出

```
如何拒重等入数据 ▼
1_知何此聲符入數篇 py
             # 导入数据
             df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/class6/BITFINEX/E0SUSD/' + file
                                   parse_dates=['candle_begin_time'])
             print(df)
             # 合并数据
             all_data = all_data.append(df, ignore_index=True) # 注意此时若一下子导入很多文件,可能会内存溢出
             print(all_data)
        for file in sorted(file_list) [:.
                                                      2.9800
                              2.9000
     1232 2017-07-02 23:52:00
                                       2.9800 2.9000
                                                                 678.528575
     1233 2017-07-02 23:53:00
                               2.9800
                                       2.9800
                                               2.8800
                                                       2.8955
                                                                 685.627382
                                                       2.9700
     1234 2017-07-02 23:54:00
                               2.9761
                                       2.9761
                                               2.8955
                                                                  22.338780
     1235 2017-07-02 23:55:00
1236 2017-07-02 23:56:00
                               2.8851
                                       3.0000
                                               2.8851
                                                       2.9600
                                                                4891.299121
                                       2.9588
                               2.9588
                                               2.8901
                                                       2.8901
                                                                 332.749353
                                               2.9103
     1237 2017-07-02 23:57:00
                               2.9980
                                       2.9980
                                                       2.9103
                                                                 723.112999
     [1238 rows x 6 columns]
                                                     low
                                           high
                                                            close
            candle_begin_time
                                  open
                                                                         volume
                                                 1.20000
                                                                    2366.619400
          2017-07-01 16:59:00
                               1.20000
                                        1.20000
                                                          1.20000
          2017-07-01 17:01:00
                                                                    1130.640508
                               1.40000
                                        1.40000
                                                 1.39000
                                                          1.39000
                                                                     234.247511
          2017-07-01 17:02:00
                               1.10000
                                        1.25000
                                                 1.10000
                                                          1.25000
          2017-07-01 17:03:00
                               1.20000
                                        1.25000
                                                 1.10000
                                                          1.11000
                                                                    2224.674241
                                                                    4301.678248
          2017-07-01 17:04:00
                               1.10000
                                        1.10000
                                                 1.00000
                                                          1.00000
          2017-07-01 17:05:00
                               1.00000
                                        1.00000
                                                 1.00000
                                                          1.00000
                                                                     400.000000
          2017-07-01 17:06:00
                               1.08000
                                        1.08000
                                                 0.70000
                                                          1.00000
                                                                    3818.954726
          2017-07-01 17:07:00
                               1.04000
                                        1.04000
                                                          0.55000
                                                                   24512.682493
                                                 0.50000
                                                 0.90000
                                                                     300.000000
          2017-07-01 17:08:00
                               0.90000
                                        0.90000
                                        0.90000
          2017-07-01 17:09:00
                              0.90000
                                                 0.90000
                                                                    1114.000000
                                                          0.90000
     10
          2017-07-01 17:10:00
                               0.60000
                                        0.99000
                                                 0.60000
                                                          0.97000
                                                                    4123.089598
          2017-07-01 17:11:00
                                        0.97000
     11
                               0.97000
                                                 0.96500
                                                          0.96500
                                                                    1034.836664
          2017-07-01 17:12:00
                                        0.96500
                               0.96500
                                                 0.95000
                                                          0.96500
                                                                     346.770000
```

#### 对数据进行排序

all\_data.sort\_values(by=['candle\_begin\_time'], inplace=True)

```
m coin_quant_class to program to class6 🐉 1_如何批量等入数据.py
                                                                                                     如何把秦母入数据 * > 00
1、知何批量等入数据 py
       all_data = pd.DataFrame()
        for file in sorted(file_list):
            print(file)
            # 导入数据
            df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/class6/BITFINEX/E0SUSD/' + file
                                skiprows=1,
                                parse dates=['candle begin_time'])
            all_data = all_data.append(df, ignore_index=True) # 注意此时若一下子导入很多文件,可能会内存溢出
       # 对数据进行排序
       all_data.sort_values(by=['candle_begin_time'], inplace=True)
       print(all_data)
     BITFINEX_EOSUSD_20180319_1T.csv
     BITFINEX_EOSUSD_20180320_1T.csv
     BITFINEX_EOSUSD_20180321_1T.csv
             candle_begin_time
                                                          close
                                                                      volume
            2017-07-01 16:59:00
                                      1.20000
                              1.20000
                                               1.20000
                                                       1.20000
                                                                 2366.619400
            2017-07-01 17:01:00 1.40000 1.40000 1.39000
                                                        1.39000
                                                                 1130.640508
     1
            2017-07-01 17:02:00 1.10000 1.25000 1.10000
                                                       1.25000
                                                                  234.247511
            2017-07-01 17:03:00 1.20000 1.25000 1.10000
                                                       1.11000
                                                                 2224.674241
                                                                 4301.678248
                                                       1.00000
     4
            2017-07-01 17:04:00 1.10000 1.10000 1.00000
     5
           2017-07-01 17:05:00 1.00000 1.00000 1.00000
                                                       1.00000
                                                                  400.000000
            2017-07-01 17:06:00
                              1.08000
                                               0.70000
                                                        1.00000
                                                                 3818.954726
     6
                                       1.08000
                                                        0.55000
            2017-07-01 17:07:00
                               1.04000
                                       1.04000
                                               0.50000
                                                                24512.682493
                                       0.90000
                                                                  300.000000
            2017-07-01 17:08:00
                               0.90000
                                               0.90000
                                                        0.90000
     8
                              0.90000
                                       0.90000
                                               0.90000
                                                        0.90000
                                                                 1114.000000
     9
            2017-07-01 17:09:00
                                                       0.97000
     10
           2017-07-01 17:10:00 0.60000 0.99000 0.60000
                                                                 4123.089598
           2017-07-01 17:11:00 0.97000 0.97000 0.96500
                                                       0.96500
                                                                 1034.836664
```

#### 将数据存入 csv 文件中

```
■ coin_quant_class [~/Desktop/coin_quant_class] - .../program/class6/1_如何批量导入数据.py [cpin_quant_class
Im coin_quant_class Im program Im classe 🐉 1 的何此最等人数据 py
                                                                                  如何把着导入数据 ▼ ▶ ○
                 O + | O - |  LEREBBARROY
m coin_quant_class [cpin_quant_class]
                                   # 合并数据
                                   all_data = all_data.append(df, ignore_index=True) # 注意此时若一下
                               # 对数据进行排序
                               all_data.sort_values(by=['candle_begin_time'], inplace=True)
                               print(all_data)
                               all_data.to_csv('all_data.csv', index=False)
   _init_py
th External Libraries
     326970 2018-03-21 23:53:00
                                                                        6.89520
                                                                                   26687.63970
                                       6.87850 6.93880 6.86180
     326971 2018-03-21 23:54:00
                                       6.90000 6.93870 6.89900
                                                                        6.93000
                                                                                    3156.794940
                                       6.92600 6.93850 6.91110
                                                                                   28172.17537
                                                                        6.91110
    326972 2018-03-21 23:55:00
     326973 2018-03-21 23:56:00
                                       6.91030 6.92000 6.90690
                                                                        6.91320
                                                                                    3093.50199
     326974 2018-03-21 23:57:00
                                       6.91320 6.91780 6.90000
                                                                        6.90000
                                                                                    3862.860999
     326975 2018-03-21 23:58:00
                                       6.89960
                                                6.90000 6.88010
                                                                        6.88400
                                                                                    7664.902456
    326976 2018-03-21 23:59:00
                                       6.89540 6.89540 6.88560
                                                                        6.89500
                                                                                    1741.239678
     [326977 rows x 6 columns]
     Process finished with exit code 0
```

# 6.2 HDF 存取数据

#### 将数据存入 hdf 文件中

all\_data.to\_hdf('.h5', key='all\_data', mode='w')

```
Mil coin_quant_class Dis program Dis class6 🐉 1_如何批量等入数据。p
                                                                                如何批量等人数据 🕶 🕨 🧿
        all_data = all_data.append(df, ignore_index=True) # 注意此时若一下子导入很多文件, 可能会内存溢出
     # 对数据进行排序
     all_data.sort_values(by=['candle_begin_time'], inplace=True)
     print(all_data)
     all_data.to_hdf(
     # 从hdf中读取文件
      2502/5 SATO-A2-51 52:32:AA D'A50AA D'A303A D'ATTTA D'ATTTA 501/5'T\22
      326973 2018-03-21 23:56:00 6.91030 6.92000
                                                                                   3093.5019
                                                            6.90690 6.91320
                                                                                   3862.8609
     326974 2018-03-21 23:57:00 6.91320 6.91780 6.90000 6.90000
      326975 2018-03-21 23:58:00 6.89960 6.90000
                                                            6.88010 6.88400
                                                                                   7664.9024
      326976 2018-03-21 23:59:00 6.89540 6.89540
                                                                                   1741.2396
                                                             6.88560
                                                                       6.89500
      [326977 rows x 6 columns]
      Process finished with exit code 0
 🖢 🛊: Run 👋 6: T000 🐞 Python Console 📉 Terminal
```

#### 从 hdf 中读取文件

all\_data = pd.read\_hdf(.h5', key='all\_data')

```
# # 将数据存入hdf文件中
# all_data.to_hdf(
      '/Users/jxing/Desktop/coin_quant_class/data/class6/eos_1min_data.h5',
      key='all_data',
# 从hdf中读取文件
all_data = pd.read_hdf('/Users/jxing/Desktop/coin_quant_class/data/class6/eos_1min_data.h5', key='all_data
print(all_data)
exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/1_如何批量
         candle_begin_time
                                         high
                                                           close
                                                                        volume
                                open
       2017-07-01 16:59:00
                             1.20000
                                                         1.20000
                                                                   2366.619400
                                      1.20000
                                               1.20000
                                                         1.39000
       2017-07-01 17:01:00
                            1.40000
                                      1.40000
                                               1.39000
                                                                   1130.640508
       2017-07-01 17:02:00
                            1.10000
                                      1.25000
                                               1.10000
                                                         1.25000
                                                                    234.247511
3
       2017-07-01 17:03:00
                            1.20000
                                      1.25000
                                               1.10000
                                                         1.11000
                                                                   2224.674241
       2017-07-01 17:04:00
                            1.10000
                                      1.10000
                                               1.00000
                                                         1.00000
                                                                   4301.678248
5
       2017-07-01 17:05:00
                            1.00000
                                      1.00000
                                               1.00000
                                                         1.00000
                                                                    400.000000
6
                                                         1.00000
                                                                   3818.954726
       2017-07-01 17:06:00
                            1.08000
                                      1.08000
                                               0.70000
7
       2017-07-01 17:07:00
                            1.04000
                                      1.04000
                                               0.50000
                                                         0.55000
                                                                  24512.682493
8
       2017-07-01 17:08:00
                            0.90000
                                      0.90000
                                                         0.90000
                                                                    300.000000
                                               0.90000
       2017-07-01 17:09:00
                            0.90000
9
                                      0.90000
                                               0.90000
                                                         0.90000
                                                                   1114.000000
10
                            0.60000
                                      0.99000
                                                         0.97000
       2017-07-01 17:10:00
                                               0.60000
                                                                   4123.089598
11
       2017-07-01 17:11:00
                            0.97000
                                      0.97000
                                               0.96500
                                                         0.96500
                                                                   1034.836664
12
       2017-07-01 17:12:00
                             0.96500
                                      0.96500
                                               0.95000
                                                         0.96500
                                                                    346.770000
                                                         1.00000
13
       2017-07-01 17:13:00
                             0.96500
                                      1.00000
                                               0.96500
                                                                   7119.489834
                                                                    205 000000
```

#### 创建 hdf 文件

h5\_store = pd.HDFStore('eos\_data.h5', mode='w')

```
Toot, atia, itea til oa: mark/ \nacia/ TVTIIA\ neakrob\ cotil Analit craas\ nara\ craaso 1.
     # 当files不为空的时候
     if files:
         for f in files:
             if f.endswith('.csv'):
                 file_list.append(f)
# 创建hdf文件
h5_store = pd.HDFStore('eos_data.h5', mode='w')
 exit()
 # 批量导入并且存储数据
 for file in sorted(file_list):
    date = file.split('_')[2]
    print(date)
for root, dirs, files in os.wal... If files for f in files
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_qu
Closing remaining open files:eos_data.h5...done
Process finished with exit code 0
```

## 批量导入并且存储数据

- for file in sorted(file\_list):
- date = file.split('\_')[2]
- print(date)

# 导入数据

- df = pd.read\_csv('/Users/jxing/Desktop/coin\_quant\_class/data/class6/BITFINEX/EOSUSD/'
   + file,
- skiprows=1,
- parse\_dates=['candle\_begin\_time'])

```
print(date)
quant_class [cpin_quant_class] -
class5
                              # 导入数据
                              df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/cla
 eos_1min_data.h5
                                                  skiprows=1,
 数据周线转换示意图.xisx
                                                  parse_dates=['candle_begin_time'])
                              # 存储数据到hdf
                              h5_store['eos_'+date] = df
 6 1_如何批量导入数据.py
 6 2_使用hdf读取文件.py
6 3_转换数据周期_resample操作.
                         # 关闭hdf文件
 init_.py
eqs_data.h5
                         h5_store.close()
                         exit()
__init__py
                         for file in sorted(file_list)
 20180314
 20180315
 20180316
 20180317
 20180318
 20180319
 20180320
 20180321
 Process finished with exit code 0
```

#### 存储数据到 hdf

h5\_store['eos\_'+date] = df

```
r读取文件.py
                file_list.append(f)
 # 创建hdf文件
 h5_store = pd.HDFStore('eos_data.h5', mode='w')
 # 批量导入并且存储数据
 for file in sorted(file_list):
    date = file.split('_')[2]
    print(date)
    # 导入数据
    df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/class6/BITFINEX/EOSUSD/
                    skiprows=1,
                    parse_dates=['candle_begin_time'])
    # 存储数据到hdf
    h5_store['eos_'+date] = df
 # 关闭hdf文件
 h5_store.close()
 exit()
 # ====读取hdf数据
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_qu
20170701
```

## 关闭 hdf 文件

h5\_store.close()

# 读取 hdf 数据

## 创建 hdf 文件

- h5\_store = pd.HDFStore('eos\_data.h5', mode='r')
- W: 写入
- r: 读取

```
file_list.append(f)
##创建hdf文件
# h5_store = pd.HDFStore('eos_data.h5', mode='w'
# # 批量导入并且存储数据
# for file in sorted(file_list):
     date = file.split('_')[2]
     print(date)
     # 导入数据
     df = pd.read_csv('/Users/jxing/Desktop/coin_guant_class/data/class6/BITFINEX/EOSUSI
                      skiprows=1,
                      parse_dates=['candle_begin_time'])
     # 存储数据到hdf
     h5_store['eos_'+date] = df
# # 关闭hdf文件
# h5_store.close()
# ====读取hdf数据
# 创建hdf文件
h5_store = pd.HDFStore('eos_data.h5',
                                    mode='ri)
# h5_store中的key
```

# h5\_store 中的 key

print(h5\_store.keys())

# 读取某个 key 指向的数据

print(h5\_store.get('eos\_20170701'))

```
# 刨建ndT又件
 h5_store = pd.HDFStore('eos_data.h5', mode='r')
 # h5_store中的key
 # print(h5_store.keys())
 # 读取某个key指向的数据
 print(h5_store.get('eos_20170701'))
 # print(h5_store['eos_20180301'])
 # 关闭hdf文件
 h5 store.close()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/2_使用hdf读取文件.py
['/eos_20170701', '/eos_20170702', '/eos_20170703', '/eos_20170704', '/eos_20170705', '/eos_20170706', '/eos_20170
    candle_begin_time
                        open
                                 high
                                          low
                                               close
                                              1.20000
                              1.20000
   2017-07-01 16:59:00 1.20000
                                      1.20000
                                                        2366.619400
                                                       1130.640508
   2017-07-01 17:01:00 1.40000
                             1.40000 1.39000
                                              1.39000
   2017-07-01 17:02:00 1.10000
                             1.25000 1.10000
                                              1.25000
                                                        234.247511
   2017-07-01 17:03:00 1.20000
                             1.25000
                                     1.10000
                                              1.11000
                                                        2224.674241
   2017-07-01 17:04:00 1.10000
                             1.10000
                                                       4301.678248
```

print(h5\_store['eos\_20180301'])

# 关闭 hdf 文件

h5\_store.close()

```
■ coin_quant_class [-/Desktop/coin_quant_class] - .../program/class6/3_转换数据简照_resample操作.py [cpin_quant_class]
                                                                                                  2 使用hdr直取文件 🔻 🕨 💸
Im coin_quant_class Di program Di class6 👸 3.转换数据用能_resample操作.py
             F lis coin_quant_class [cpin_quant_class]
                            import pandas as pd
   class5
                            # ===从hdf中读取1分钟数据
                            df = pd.read_hdf('/Users/ixing/Desktop/coin_quant_class/data/class6/eos_1min_data.
                            # 选取某一时间段
                            df = df[df['candle_begin_time'] >= pd.to_datetime('2018-03-01')]
                            print(df.head(10))
                            # 《数据周线转换示意图》
      1_如何批量导入数据_py
                            # ===第一种方法: 将1分钟数据转为5分钟数据
                            # 将candle_begin_time设定为index
      eos_data.h5
                            df.set_index('candle_begin_time', inplace=True)
  ► Em class7
    Em class8
    _init_py
                            # 周期转换方法: resample
 In External Libraries
                            rule_type = '5T' # rule='5T': 意思是5分钟, 意味着转变为5分钟数据
                            period_df = df[['close']].resample(rule=rule_type).last() # last: 取这5分钟的最后一行数
                            # 开、高、低的价格, 成交量
                            period_df['open'] = df['open'].resample(rule=rule_type).first()
                            period_df['high'] = df['high'].resample(rule=rule_type).max()
                            period_df['low'] = df['low'].resample(rule=rule_type).min()
                            period_df['volume'] = df['volume'].resample(rule=rule_type).sum()
                            period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
                            # ===第二种方法: 将1分钟数据转为5分钟数据
                            rule_type = '5T'
                            period_df = df.resample(rule=rule_type, on='candle_begin_time', base=0, label='left
                                 {'open': 'first', 'high': 'max',
```

## 从 hdf 中读取 1 分钟数据

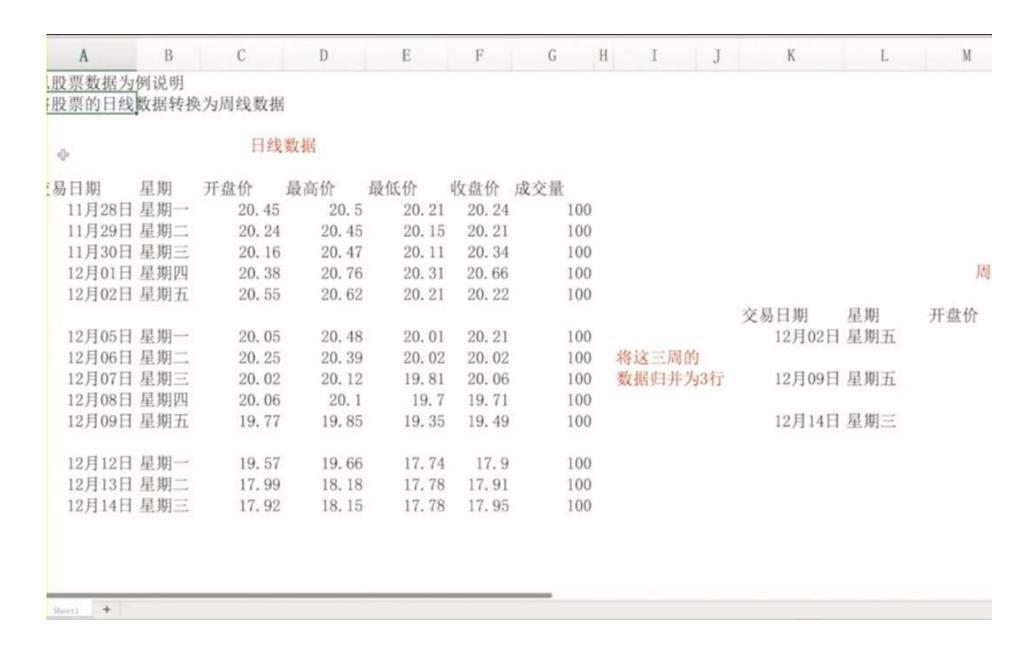
df = pd.read\_hdf('/Users/jxing/Desktop/coin\_quant\_class/data/class6/eos\_1min\_data.h5',
 key='all\_data')

#### 选取某一时间段

- df = df[df['candle\_begin\_time'] >= pd.to\_datetime('2017-03-01')]
- print(df.head(10))

```
import pandas as pd
 # ===从hdf中读取1分钟数据
 df = pd.read_hdf('/Users/jxing/Desktop/coin_quant_class/data/class6/eos_1m
 # 选取某一时间段
 df = df[df['candle_begin_time'] >= pd.to_datetime('2018-03-01')]
 print(df.head(10))
 exit()
    《数据周线转换示意图》
技術問題 resample解析
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/progra
        candle begin time
                                   high
                                            low
                                                 close
                                                             volume
                            open
300327 2018-03-01 00:00:00 8.1754 8.1827 8.1677
                                                8.1818 1960.577170
300328 2018-03-01 00:01:00 8.1816 8.1910 8.1642 8.1825
                                                         971.645839
300329 2018-03-01 00:02:00 8.1824 8.1959 8.1673 8.1711
                                                         645.795930
300330 2018-03-01 00:04:00 8.1769 8.1799 8.1622 8.1623
                                                         722.762733
300331 2018-03-01 00:05:00 8.1623 8.1623 8.1337 8.1337
                                                        7360.063461
300332 2018-03-01 00:06:00 8.1337 8.1454 8.1150 8.1150
                                                        4827.110082
300333 2018-03-01 00:07:00 8.1259 8.1339 8.1181 8.1181
                                                          25.845200
300334 2018-03-01 00:08:00 8.1279 8.1417 8.1178 8.1310
                                                        2678.376247
300335 2018-03-01 00:09:00 8.1310 8.1330 8.1191 8.1330 461.068778
300336 2018-03-01 00:10:00 8.1259 8.1419 8.1199 8.1419 1134.719523
Process finished with exit code 0
```

#### 《数据周线转换示意图》



第一种方法:将1分钟数据转为5分钟数据

将 candle\_begin\_time 设定为 index

df.set\_index('candle\_begin\_time', inplace=True)

```
print(df.head(10))
 # 《数据周线转换示意图》
 # ===第一种方法: 将1分钟数据转为5分钟数据
 # 将candle_begin_time设定为index
 df.set_index('candle_begin_time', inplace=True)
 print(df)
 exit()
 # 周期转换方法: resample
  rule_type = '5T' # rule='5T': 意思是5分钟, 意味着转变为5分钟数据
 period_df = df[['close']].resample(rule=rule_type).last() # last: 取这5分钟
 # 开、高、低的价格,成交量
  nariad df[|anan|] - df[|anan|] racamn]a/rula-rula tuna) firet/)
300336 2018-03-01 00:10:00 8.1259 8.1419 8.1199 8.1419 1134.719523
                 open
                        high
                               low
                                    close
candle_begin_time
2018-03-01 00:00:00 8.1754 8.1827 8.1677 8.1818 1960.577170
2018-03-01 00:01:00 8.1816 8.1910 8.1642 8.1825
                                         971.645839
2018-03-01 00:02:00 8.1824 8.1959 8.1673 8.1711
                                         645.795930
2018-03-01 00:04:00 8.1769 8.1799 8.1622 8.1623
                                          722.762733
2018-03-01 00:05:00 8.1623 8.1623 8.1337 8.1337
                                         7360.063461
2018-03-01 00:06:00 8.1337 8.1454 8.1150 8.1150
                                         4827.110082
2018-03-01 00:07:00 8.1259 8.1339 8.1181 8.1181
                                          25.845200
```

## 周期转换方法: resample

- rule\_type = '5T' # rule='5T': 意思是 5 分钟, 意味着转变为 5 分钟数据
- period\_df = df[['close']].resample(rule=rule\_type).last() # last: 取这 5 分钟的最后一行数据

```
# 周期转换方法: resample
 rule_type = '5T' # rule='5T': 意思是5分钟, 意味着转变为5分钟数据
 period_df = df[['close']].resample(rule=rule_type).last() # last: 取这5分钟的最后一行数据
 print(period_df)
 exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/clas
                                   high
                                           low close
        candle_begin_time
                           open
                                                            volume
300327 2018-03-01 00:00:00 8.1754 8.1827 8.1677 8.1818 1960.577170
300328 2018-03-01 00:01:00 8.1816 8.1910 8.1642 8.1825 971.645839
300329 2018-03-01 00:02:00 8.1824 8.1959 8.1673 8.1711
                                                        645.795930
300330 2018-03-01 00:04:00 8.1769 8.1799 8.1622 8.1623 722.762733
300331 2018-03-01 00:05:00 8.1623 8.1623 8.1337 8.1337
                                                       7360.063461
300332 2018-03-01 00:06:00 8.1337 8.1454 8.1150 8.1150
                                                       4827.110082
300333 2018-03-01 00:07:00 8.1259 8.1339 8.1181 8.1181
                                                         25.845200
300334 2018-03-01 00:08:00 8.1279 8.1417 8.1178 8.1310
                                                      2678.376247
300335 2018-03-01 00:09:00 8.1310 8.1330 8.1191 8.1330
                                                       461.068778
300336 2018-03-01 00:10:00 8.1259 8.1419 8.1199 8.1419 1134.719523
                    close
candle_begin_time
2018-03-01 00:00:00 8.1623
2018-03-01 00:05:00 8.1330
```

#### 开、高、低的价格,成交量

- period\_df['open'] = df['open'].resample(rule=rule\_type).first()
- period\_df['high'] = df['high'].resample(rule=rule\_type).max()
- period\_df['low'] = df['low'].resample(rule=rule\_type).min()
- period\_df['volume'] = df['volume'].resample(rule=rule\_type).sum()
- period\_df = period\_df[['open', 'high', 'low', 'close', 'volume']]

```
# ===第一种方法: 将1分钟数据转为5分钟数据
 # 将candle_begin_time设定为index
 df.set_index('candle_begin_time', inplace=True)
 # 周期转换方法: resample
 rule_type = '5T' # rule='5T': 意思是5分钟, 意味着转变为5分钟数据
 period_df = df[['close']].resample(rule=rule_type).last() # last: 取这5分钟的最后一行数据
 # 开、高、低的价格, 成交量
 period_df['open'] = df['open'].resample(rule=rule_type).first()
 period_df['high'] = df['high'].resample(rule=rule_type).max()
 period_df['low'] = df['low'].resample(rule=rule_type).min()
 period_df['volume'] = df['volume'].resample(rule=rule_type).sum()
 period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
 print(period_df)
 exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/clas
        candle_begin_time
                                             low
                                                  close
                                                              volume
                            open
                                    high
300327 2018-03-01 00:00:00 8.1754
                                  8.1827
                                         8.1677
                                                 8.1818
                                                        1960.577170
300328 2018-03-01 00:01:00 8.1816
                                  8.1910 8.1642 8.1825
                                                          971.645839
300329 2018-03-01 00:02:00 8.1824 8.1959 8.1673 8.1711
                                                          645.795930
300330 2018-03-01 00:04:00 8.1769
                                  8.1799 8.1622 8.1623
                                                          722.762733
300331 2018-03-01 00:05:00 8.1623 8.1623 8.1337 8.1337
                                                         7360.063461
```

#### 第二种方法:将1分钟数据转为5分钟数据

- rule\_type = '5T'
- period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=0, label='left', closed='left').agg
- ({'open': 'first','high': 'max','low': 'min','close': 'last','volume': 'sum',})

```
# period_df['open'] = df['open'].resample(rule=rule_type).first()
# period_df['high'] = df['high'].resample(rule=rule_type).max()
# period_df['low'] = df['low'].resample(rule=rule_type).min()
# period_df['volume'] = df['volume'].resample(rule=rule_type).sum()
# period df = period df[['open', 'high', 'low', 'close', 'volume']]
# ===第二种方法: 将1分钟数据转为5分钟数据
rule_type = '5T'
period_df = df.resample(rule=rule_type, on='candle_begin_time', base=0, label='left', close
{'open': first',
period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
# base参数:帮助确定转换周期开始的时间
# label='left', closed='left', 建议统一设置成'left'
# ===去除不必要的数据
# 去除一天都没有交易的周
period_df.dropna(subset=['open'], inplace=True)
# 去除成交量为0的交易周期
period_df = period_df[period_df['volume'] > 0]
```

- period\_df = period\_df[['open', 'high', 'low', 'close', 'volume']]
- base 参数:帮助确定转换周期开始的时间
- label='left', closed='left', 建议统一设置成'left'

```
# ===第二种方法: 将1分钟数据转为5分钟数据
 rule_type = '5T'
 period_df = df.resample(rule=rule_type, on='candle_begin_time', base=0, label='left', close
    {'open': 'first',
     'high': 'max',
     'close': 'last',
     'volume': 'sum',
 period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
print(period df)
 exit()
 # base參数: 帮助确定转换周期开始的时间
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quar
          candle_begin_time
                                         high
                                open
                                                   low
                                                          close
                                                                       volume
300327 2018-03-01 00:00:00
                              8.1754
                                       8.1827
                                                8.1677
                                                        8.1818
                                                                 1960.577170
300328 2018-03-01 00:01:00
                              8.1816
                                       8.1910
                                                8.1642
                                                        8.1825
                                                                  971.645839
300329 2018-03-01 00:02:00
                              8.1824,
                                       8.1959
                                                8.1673
                                                        8.1711
                                                                  645.795930
300330 2018-03-01 00:04:00
                                       8.1799
                                                                  722.762733
                              8.1769
                                                8.1622
                                                         8.1623
300331 2018-03-01 00:05:00
                                       8.1623
                              8.1623
                                                8.1337
                                                        8.1337
                                                                 7360.063461
300332 2018-03-01 00:06:00
                                                                 4827.110082
                              8.1337
                                       8.1454
                                                8.1150
                                                        8.1150
300333 2018-03-01 00:07:00
                              8.1259
                                                8.1181
                                                         8.1181
                                                                    25.845200
                                       8.1339
                                                                 2678.376247
300334 2018-03-01 00:08:00 8.1279 8.1417
                                              8.1178 8.1310
```

period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=1, label='left', closed='left')

• base=1,使1分钟转换成5分钟的起点分钟时间推后1位,即 base=0时,5分钟的起点为第0分钟开始(0-4分钟),base=1则为第1分钟开始(1-5分钟),以此类推。

```
# period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
# ===第二种方法: 将1分钟数据转为5分钟数据
rule type = '5T'
period_df = df.resample(rule=rule_type, on='candle_begin_time', base=1, label='left', clo
   {'open': 'first',
300331 2018-03-01 00:05:00 8.1023 8.1023 8.133/ 8.133/ /300.00340
300332 2018-03-01 00:06:00 8.1337 8.1454 8.1150 8.1150 4827.11008
300333 2018-03-01 00:07:00 8.1259 8.1339 8.1181
                                                   8.1181
                                                             25.84520
300334 2018-03-01 00:08:00 8.1279
                                   8.1417 8.1178 8.1310
                                                           2678.37624
300335 2018-03-01 00:09:00 8.1310 8.1330 8.1191
                                                   8.1330
                                                           461.06877
300336 2018-03-01 00:10:00 8.1259
                                   8.1419 8.1199 8.1419
                                                           1134.71952
                              high
                                             close
                                       low
                                                           volume
                      open
candle_begin_time
2018-02-28 23:56:00
                            8.1827
                                    8.1677
                    8.1754
                                            8.1818
                                                      1960.577170
2018-03-01 00:01 00 8.1816
                            8.1959 8.1337
                                            8.1337
                                                      9700.267963
                                            8.1419
                            8.1454
2018-03-01 00:06:00 8.1337
                                    8.1150
                                                      9127.119831
2018-03-01 00:11:00 8.1420
                            8.1569 8.1286
                                            8.1419
                                                      5103.353294
2018-03-01 00:16:00 8.1419 8.1646 8.1410 8.1600
                                                      3891.474900
                            8.2150
                                            8.2011
2018-03-01 00:21:00
                    8.1600
                                    8.1600
                                                      9826.695940
```

- period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=2, label='left', closed='left')
- base=2,则为第1分钟开始(2-6分钟),因为有一些策略放弃前几分钟的数据,采用另类的及时方式来提高策略的有效性,可以采用参数穷举得到收益率差异。

```
# period_df = period_df[['open', 'high', 'low', 'close', 'volume']]
 # ===第二种方法: 将1分钟数据转为5分钟数据
 rule_type = '5T'
 period_df = df.resample(rule=rule_type, on='candle_begin_time', base=2, label='left', close
    {'open': 'first',
CIANT CONTOLAN AN MA'MA O'TION C'TINN O'TINT O'TOTT O'TOTT O'TOTT
300331 2018-03-01 00:05:00 8.1623 8.1623 8.1337
                                                    8.1337
                                                            7360.06346
300332 2018-03-01 00:06:00 8.1337 8.1454 8.1150 8.1150
                                                            4827.11008
300333 2018-03-01 00:07:00
                            8.1259
                                    8.1339 8.1181 8.1181
                                                              25.845200
300334 2018-03-01 00:08:00
                            8.1279
                                    8.1417 8.1178 8.1310
                                                            2678.376247
                            8.1310
                                            8.1191
                                                    8.1330
300335 2018-03-01 00:09:00
                                    8.1330
                                                             461.068778
300336 2018-03-01 00:10:00
                            8.1259
                                    8.1419
                                            8.1199
                                                    8.1419
                                                            1134.71952
                                              close
                                                            volume
                       open
                               high
                                        low
candle begin time
2018-02-28 23:57:00
                     8.1754
                             8.1910
                                     8.1642
                                             8.1825
                                                       2932.223009
2018-03-01 00:02:00
                                     8.1150 8.1150
                     8.1824
                             8.1959
                                                      13555.732207
2018-03-01 00:07:00
                    8.1259
                                     8.1178 8.1460
                                                       5333.876140
                             8.1557
2018-03-01 00:12:00
                   8.1460
                             8.1569
                                     8.1286 8.1419
                                                       4092.982102
2018-03-01 00:17:00
                                     8.1410
                             8.1666
                    8.1420
                                            8.1666
                                                       5873.980900
                                             8.1993
2018-03-01 00:22:00
                     8.1748
                             8.2189
                                     8.1748
                                                      15618.326580
                                                       9827.234074
2018-03-01 00:27:00
                     8.1995
                             8.2122
                                     8.1830
                                             8.1830
```

#### label='left'

period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=0, label='left', closed='left')

```
rule_type = '5T
period_df = df.resample(rule=rule_type, on='candle_begin_time', base=0, label='lqft', clo
   {'open': 'first',
    'volume': 'sum',
2701'9 MA!CA!MA TA-CA-Q17A7
                                    0.1025
                                            0.133/
                                                    0.133/
                                                             /30W.W0340
                                            8.1150
300332 2018-03-01 00:06:00 8.1337
                                    8.1454
                                                    8.1150
                                                             4827.110082
300333 2018-03-01 00:07:00 8.1259 8.1339
                                            8.1181
                                                    8.1181
                                                               25.845200
                                                    8.1310
300334 2018-03-01 00:08:00 8.1279 8.1417
                                            8.1178
                                                             2678.37624
300335 2018-03-01 00:09:00
                           8,1310
                                                             461.068778
                                    8.1330
                                            8.1191
                                                    8.1330
                                                    8.1419
300336 2018-03-01 00:10:00 8.1259
                                    8.1419
                                                            1134.71952
                                            8.1199
                                              close
                               high
                                                             volume
                                        low
                       open
candle_begin_time
2018-03-01 00:00:00
                                                       4300.781673
                             8.1959
                                             8.1623
                     8.1754
                                     8.1622
                             8.1623
                                     8.1150
                                             8.1330
                                                      15352.463768
2018-03-01 00:05:00
                    8.1623
2018-03-01 00:10:00
                    8.1259
                             8.1569
                                     8.1199
                                             8.1392
                                                       4328.911697
2018-03-01 00:15:00
                                             8.1600
                   8.1392
                             8.1646
                                     8.1286
                                                       3996.532020
                    8.1598
                             8.2081
                                     8.1598 8.2070
                                                      10367.382166
2018-03-01 00:20:00
                   8.2100 8.2189
                                     8.1852
                                             8.2109
                                                      16541.544195
2018-03-01 00:25:00
```

#### label='right'

period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=0, label='right', closed='left')

```
rule_type = '5T
 period_df = df.resample(rule=rule_type, on='candle_begin_time', base=0, label='right', cl
     {'open': 'first',
      'low': 'min',
      'close': 'last',
      'volume': 'sum',
3_转换数据周期_resample操作 3_转换数据周期_resample操作
                                        8.1417
 300334 2018-03-01 00:08:00
                                8.1279
                                                  8.1178
                                                           8.1310
                                                                    2678.37624
 300335 2018-03-01 00:09:00
                                8.1310
                                                  8.1191
                                         8.1330
                                                           8.1330
                                                                     461.06877
 300336 2018-03-01 00:10:00 8.1259 8.1419
                                                  8.1199 8.1419
                                                                   1134.71952
                          open
                                   high
                                                    close
                                                                    volume
                                             low
 candle_begin_time 👱
2018-03-01 00:05:00
                                 8.1959
                                          8.1622
                        8.1754
                                                   8.1623
                                                              4300.781673
                                 8.1623
                                                   8.1330
2018-03-01 00:10:00
                                                             15352.463768
                        8.1623
                                          8.1150
2018-03-01 00:15:00
                       8.1259
                                 8.1569
                                          8.1199
                                                   8.1392
                                                              4328.911697
                        8.1392
2018-03-01 00:20:00
                                 8.1646
                                          8.1286
                                                   8.1600
                                                              3996.532020
                                                             10367.382166
2018-03-01 00:25:00
                       8.1598
                                 8.2081
                                          8.1598
                                                   8.2070
2018-03-01 00:30:00
                       8.2100
                                 8.2189
                                          8.1852
                                                   8.2109
                                                             16541.544195
2018-03-01 00:35:00
                        8.2108
                                 8.2119
                                          8.1830
                                                   8.1992
                                                             10953.504903
 2018-03-01 00:40:00
                       8.1992
                                 8.2687
                                          8.1992
                                                   8.2470
                                                             19109.269358
6: TODO Python Console Terminal
```

#### closed='right'

period\_df = df.resample(rule=rule\_type, on='candle\_begin\_time', base=0, label='left', closed='right')

```
= df.resample(rule=rule_type, on='candle_begin_time', base=0, label='left', closed='right
': 'first',
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_qu
                                                        close
         candle_begin_time
                                        high
                               open
                                                  low
                                                                    volum
                                              8.1677
                                                       8.1818
300327 2018-03-01 00:00:00
                             8.1754
                                      8.1827
                                                               1960.57717
                                                       8.1825
                                      8.1910
                                              8.1642
                                                                971.64583
300328 2018-03-01 00:01:00
                             8.1816
                                                                645.795930
300329 2018-03-01 00:02:00
                            8.1824
                                      8.1959
                                              8.1673
                                                      8.1711
                                                                722.76273
                             8.1769
                                      8.1799
                                              8.1622
300330 2018-03-01 00:04:00
                                                       8.1623
300331 2018-03-01 00:05:00
                             8.1623
                                      8.1623
                                                               7360.06346
                                              8.1337
                                                       8.1337
                                                               4827.110083
                             8.1337
                                      8.1454
                                              8.1150
                                                      8.1150
300332 2018-03-01 00:06:00
                             8.1259
                                      8 1339
300333 2018-03-01 00:07:00
                                              8.1181
                                                      8.1181
                                                                 25.84520
300334 2018-03-01 00:08:00
                             8.1279
                                              8.1178
                                     8.1417
                                                       8.1310
                                                               2678.37624
                                              8.1191
                                                                461.068778
                                                      8.1330
300335 2018-03-01 00:09:00
                             8.1310
                                      8.1330
                                                               1134.71952
                             8.1259
                                              8.1199
300336 2018-03-01 00:10:00
                                      8.1419
                                                       8.1419
                                high
                                                close
                                                               volume
                                          low
                        open
candle_begin_time
2018-02-28 23:55:00 8.1754
                              8.1827
                                       8.1677
                                               8.1818
                                                          1960.577170
2018-03-01 00:00:00
                              8.1959
                     8.1816
                                       8.1337
                                               8.1337
                                                          9700.267963
2018-03-01 00:05:00
                              8.1454
                                       8.1150
                                               8.1419
                                                          9127.119831
                      8.1337
                      8.1420
                              8.1569
                                       8.1286
                                               8.1419
                                                          5103.353294
2018-03-01 00:10:00
```

# 去除不必要的数据,去除一天都没有交易的周

period\_df.dropna(subset=['open'], inplace=True)

## 去除成交量为 0 的交易周期

• period\_df = period\_df[period\_df['volume'] > 0]

```
=去除不必要的数据
                   oset=['open'], inplace=True)
# 去除成交量为0的交易周期
period_df = period_df[period_df['volume'] > 0]
# ===rule的取值
           business day frequency
           custom business day frequency (experimental)
           calendar day frequency
           weekly frequency
           month end frequency
           semi-month end frequency (15th and end of month)
   SM
           business month end frequency
   BM
   CBM
           custom business month end frequency
```

#### rule 的取值

- S: secondly frequency, 秒
- T: minutely frequency, 分钟
- H: hourly frequency, 小时
- D: calendar day frequency, ∃
- W: weekly frequency,周
- M: month end frequency,月
- Q: quarter end frequency, 季
- A: year end frequency, 年

```
# ===rule的取值
            business day frequency
            custom business day frequency (experimental)
            weekly frequency
          semi-month end frequency (15th and end of month)
    BM
            business month end frequency
    CBM
            custom business month end frequency
            month start frequency
    SMS
            semi-month start frequency (1st and 15th)
    BMS
            business month start frequency
    CBMS
            custom business month start frequency
    BQ
            business quarter endfrequency
    QS
            quarter start frequency
    BQS
    BQS
            year end frequency
    BA
            business year end frequency
            year start frequency
    BAS
            business year start frequency
            secondly frequency
            milliseonds
            microseconds
            nanoseconds
```

# 6.4 groupby 分组

#### 导入数据

df =
 pd.read\_csv('/Users/jxing/Desktop/coin\_quant\_class/data/class5/BITFINEX-1H-data-20180
 124.csv', skiprows=1)

```
【無例3】 寻找最优参数 🕶 🕨 🐞
                                           import pandas as pd
                                           pd.set_option('expand_frame_repr', False) # 当列太多时不换行
                                           # ====导入数据
                                           df = pd.read_csv('/Users/jxing/Desktop/coin_quant_class/data/class5/BITFINEX-1H-data-20180124.csv',
        № 1_如何批量等入数据.py
                                           # = ____groupby常用操作汇总
# 根据'candle_begin_time'进行group,将相同'交易日期'的行放入一个group,
print(df.groupby('candle_begin_time')) # 生成一个group对象。不会做实质性操作,只是会判断是否可以根据该变量进行g
        3_转换数据周期_resample操作.py
       4_分组处理操作_groupby.py
        _init_py
       eos_data.h5
                                           # group后可以使用相关函数, size()计算每个group的行数 print(df.groupby('candle_begin_time').size()) # 每天交易股票的个数 # 根据'symbol'进行group, 将相同'交易日期'的行放入一个group,
   ▶ Em class8
                                           print(df.groupby('symbol').size()) # 每只股票交易的天数
                                           print(df.groupby('candle_begin_time').get_group('2018-01-24 00:00:00'))
                                           print(df.groupby('symbol').get_group('BTCUSD'))
                                           # 其他常见函数
                                           print(df.groupby('symbol').describe()) # 只会对数值变量进行describe
                                           print(df.groupby('symbol').head(3))
                                          print(df.groupby('symbol').tail(3)) # 每个group里面的行票序。会保留。
print(df.groupby('symbol').first())
print(df.groupby('symbol').last())
                                           print(df.groupby('symbol').nth(2))
                                           print(df.groupby('symbol', as_index=False).nth(2))
                                           # 在group之后,取一部分变量进行计算
# 计算每个group的均值
                                           print(df.groupby('symbol')['close', 'volume'].mean())
                                           print(df.groupby('symbol')['close', 'volume'].max())
👙 fi: TODO 💮 Python Console
                                                                                                                                                          Event Log
```

根据'candle\_begin\_time'进行 group,将相同'交易日期'的行放入一个 group,

• print(df.groupby('candle\_begin\_time')) # 生成一个 group 对象。不会做实质性操作,只是会判断是否可以根据该变量进行 groupby。

# group 后可以使用相关函数,size()计算每个 group 的行数

• print(df.groupby('candle\_begin\_time').size()) # 每小时交易的币的个数

```
# 根据'candle_begin_time'进行group,将相同'交易日期'的行放入一个group,
 # print(df.groupby('candle_begin_time')) # 生成一个group对象。不会做实质性操作,只是会
 # group后可以使用相关函数, size()计算每个group的行数
 print(df.groupby('candle_begin_time').size()) # 每小时交易的币的个数
 exit()
 # 根据'symbol'进行group, 将相同'symbol'的行放入一个group,
 print(df.groupby('symbol').size()) #每个币交易的小时数
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant
candle_begin_time
2018-01-24 00:00:00
                        82
                        79
2018-01-24 01:00:00
2018-01-24 02:00:00
                        84
2018-01-24 03:00:00
                        82
2018-01-24 04:00:00
                        80
2018-01-24 05:00:00
                        80
2018-01-24 06:00:00
                        84
2018-01-24 07:00:00
                        84
2018-01-24 08:00:00
                        82
2018-01-24 09:00:00
                        84
```

# 根据'symbol'进行 group,将相同'symbol'的行放入一个 group

• print(df.groupby('symbol').size()) # 每个币交易的小时数

```
# print(df.groupby('candle_begin_time')) # 生成一个group对象。不会做实质性操作,只是会
 # group后可以使用相关函数, size()计算每个group的行数
 # print(df.groupby('candle_begin_time').size()) # 每小时交易的币的个数
 # 根据'symbol'进行group, 将相同'symbol'的行放入一个group,
 print(df.groupby('symbol').size()) #每个币交易的小时数
 exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/
symbol
AIDBTC
            8
AIDETH
            8
            8
AIDUSD
           23
AVTBTC
           22
AVTETH
           24
AVTUSD
           24
BATBTC
BATETH
           21
BATUSD
           24
```

# 获取其中某一个 group

print(df.groupby('candle\_begin\_time').get\_group('2018-01-24 00:00:00'))

```
# 获取其中某一个group
  print(df.groupby('candle_begin_time').get_group('2018-01-24 00:00:00'))
  exit()
 print(df.groupby('symbol').get_group('BTCUSD'))
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/
        candle_begin_time
                           symbol
                                                                                       close
      2018-01-24 00:00:00
                           AVTBTC
                                        0.000329
                                                       0.000338
                                                                     0.000328
                                                                                    0.000338
                                                                                              9.11
47
      2018-01-24 00:00:00
                           AVTETH
                                        0.003633
                                                       0.003725
                                                                     0.003633
                                                                                    0.003725
                                                                                              4.30
69
      2018-01-24 00:00:00
                                                                                    3.590900
                                                                                              3.45
                           AVTUSD
                                        3.550000
                                                       3.591100
                                                                     3.550000
93
      2018-01-24 00:00:00
                           BATBTC
                                        0.000052
                                                       0.000058
                                                                     0.000052
                                                                                    0.000052
                                                                                              6.18
117
      2018-01-24 00:00:00
                                                       0.000640
                                                                                    0.000589
                           BATETH
                                        0.000592
                                                                     0.000589
                                                                                              2.32
138
      2018-01-24 00:00:00
                           BATUSD
                                        0.563070
                                                       0.630340
                                                                     0.538360
                                                                                    0.538360
                                                                                              1.33
      2018-01-24 00:00:00
                           BCHBTC
                                        0.149160
                                                       0.149180
                                                                     0.147880
                                                                                    0.148020
                                                                                              5.57
                                                                     1.627400
      2018-01-24 00:00:00
                                                                                    1.637200
186
                           BCHETH
                                        1.637900
                                                       1.638300
                                                                                              8.10
210
      2018-01-24 00:00:00
                           BCHUSD
                                                   1624.900000
                                                                  1567.000000
                                     1613.100000
                                                                                 1568.000000
                                                                                              1.84
234
                                                   8853.835804
                                                                                 8614.000000
      2018-01-24 00:00:00
                           BTCEUR
                                     8794.544912
                                                                  8547.700000
                                                                                              1.39
                                                                 10522.000000
                                                                                              2.61
258
      2018-01-24 00:00:00
                                                  10906.000000
                                                                                10580.000000
                           BTCUSD
                                    10812.000000
      2018-01-24 00:00:00
                                                                                    0.017319 4.73
282
                           BTGBTC
                                        0.017058
                                                       0.017360
                                                                     0.017031
306
      2018-01-24 00:00:00
                           BTGUSD
                                                                                              3.26
                                      184.900000
                                                     186.340000
                                                                   182.170000
                                                                                  183.600000
      7018-01-74 00-00-00
                                        a aaaa16
                                                                                    0 000016 1 06
                           DATRTC
                                                       0 000016
                                                                     a aaaa16
```

print(df.groupby('symbol').get\_group('BTCUSD'))

```
# 获取其中某一个group
 # print(df.groupby('candle_begin_time').get_group('2018-01-24 00:00:00'))
 print(df.groupby('symbol').get_group('BTCUSD'))
  exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_class/program/class6/
       candle_begin_time
                          symbol
                                                                        low
                                                                               close
                                                                                           volum
                                          open
                                                         high
     2018-01-24 00:00:00
                          BTCUSD
                                  10812.000000
                                                10906.000000
                                                               10522.000000
                                                                             10580.0
                                                                                      2616.12474
259
    2018-01-24 01:00:00
                          BTCUSD
                                  10581.000000
                                                10829.000000
                                                               10522.000000
                                                                             10589.0
                                                                                      2562.00700
260
                                                10768.000000
                                                                             10739.0
                                                                                      2043.93154
    2018-01-24 02:00:00
                          BTCUSD
                                  10588.000000
                                                               10518.333218
                          BTCUSD
261
    2018-01-24 03:00:00
                                  10739.000000
                                                10861.000000
                                                               10615.000000
                                                                             10810.0
                                                                                      1137.68719
262
    2018-01-24 04:00:00
                          BTCUSD
                                  10809.000000
                                                10813.000000
                                                                             10631.0
                                                                                      2251.17544
                                                               10430.000000
    2018-01-24 05:00:00
                                  10630.000000
263
                          BTCUSD
                                                10749.000000
                                                               10542.000000
                                                                             10626.0
                                                                                       685.34088
                          BTCUSD
                                                               10598.000000
                                                                             10921.0
                                                                                      1652.14212
264
    2018-01-24 06:00:00
                                  10626.000000
                                                11052.000000
265
    2018-01-24 07:00:00
                          BTCUSD
                                  10921.000000
                                                11090.000000
                                                               10842.000000
                                                                             10936.0
                                                                                      1638.26467
266
    2018-01-24 08:00:00
                          BTCUSD
                                  10936.164237
                                                11013.000000
                                                               10813.000000
                                                                             10835.0
                                                                                       945.52062
                                                                                      1244.67316
                          BTCUSD
267
    2018-01-24 09:00:00
                                  10835.000000
                                                11087.000000
                                                               10761.000000
                                                                             11051.0
                          BTCUSD
268
    2018-01-24 10:00:00
                                                               10919.000000
                                                                             10991.0
                                  11055.000000
                                                11162.000000
                                                                                      2015.22184
    2018-01-24 11:00:00
269
                                                                                      2001.24610
                          BTCUSD
                                  10990.000000
                                                11194.000000
                                                               10888.000000
                                                                             11045.0
                          BTCUSD
     2018-01-24 12:00:00
270
                                  11045.000000
                                                11461.000000
                                                               10970.000000
                                                                             11286.0
                                                                                      3391.24426
    2018-01-24 13.00.00
                                                11385 000000
                                                                             11330 0 1081 17237
271
                          RTCHSD
                                  11286 098637
                                                               11183 000000
```

## 其他常见函数

- print(df.groupby('symbol').describe()) # 只会对数值变量进行 describe
- print(df.groupby('symbol').head(3))
- print(df.groupby('symbol').tail(3)) # 每个 group 里面的行顺序,会保留。
- print(df.groupby('symbol').first())

- print(df.groupby('symbol').last())
- print(df.groupby('symbol').nth(2))

```
# 其他常见函数
print(df.groupby('symbol').describe()) # 只会对数值变量进行describe
print(df.groupby('symbol').head(3))
print(df.groupby('symbol').tail(3)) # 每个group里面的行顺序,会保留。
print(df.groupby('symbol').first())
print(df.groupby('symbol').last())
print(df.groupby('symbol').nth(2))
# 将group变量不设置为index
print(df.groupby('symbol', as_index=False).nth(2))
```

# 将 group 变量设置为 index

print(df.groupby('symbol').nth(2))

```
# 其他常见函数
 # print(df.groupby('symbol').describe())
                                         # 只会对数值变量进行describe
 # print(df.groupby('symbol').head(3))
 # print(df.groupby('symbol').tail(3))
                                      #每个group里面的行顺序,会保留。
 # print(df.groupby('symbol').first())
 # print(df.groupby('symbol').last())
 print(df.groupby('symbol<sub>i</sub>').nth(2))
 exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_
          candle_begin_time
                                      close
                                                      high
                                                                      low
symbol
        2018-01-24 18:00:00
AIDBTC
                                   0.000072
                                                  0.000075
                                                                 0.000050
AIDETH 2018-01-24 18:00:00
                                   0.000610
                                                  0.000674
                                                                 0.000560
AIDUSD 2018-01-24 18:00:00
                                   0.715000
                                                  0.717000
                                                                 0.480000
AVTBTC 2018-01-24 02:00:00
                                   0.000341
                                                  0.000341
                                                                 0.000338
AVTETH 2018-01-24 02:00:00
                                   0.003721
                                                  0.003729
                                                                 0.003714
AVTUSD 2018-01-24 02:00:00
                                                  3.743100
                                   3.732100
                                                                 3.583000
                                                                 0.000051
BATBTC
        2018-01-24 02:00:00
                                   0.000051
                                                  0.000060
BATETH
        2018-01-24 03:00:00
                                   0.000558
                                                  0.000558
                                                                 0.000558
BATUSD 2018-01-24 02:00:00
                                   0.540000
                                                  0.562590
                                                                 0.540000
```

# 将 group 变量【不】设置为 index

print(df.groupby('symbol', as\_index=False).nth(2))

```
# 每个group里面的行顺序, 会保留。
 # print(df.groupby('symbol').tail(3)
 # print(df.groupby('symbol').first())
 # print(df.groupby('symbol').last())
 # print(df.groupby('symbol').nth(2))
 # 将group变量不设置为index
 print(df.groupby('symbol', as index=False).nth(2))
 exit()
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant
        candle_begin_time
                             symbol
                                                             high
                                              open
                                          0.000063
      2018-01-24 18:00:00
                             AIDBTC
                                                         0.000075
                                                                        0.00005
10
      2018-01-24 18:00:00
                             AIDETH
                                          0.000625
                                                         0.000674
                                                                        0.00056
18
      2018-01-24 18:00:00
                            AIDUSD
                                          0.618000
                                                         0.717000
                                                                        0.48000
26
      2018-01-24 02:00:00
                             AVTBTC
                                          0.000338
                                                         0.000341
                                                                        0.00033
                            AVTETH
49
      2018-01-24 02:00:00
                                          0.003714
                                                         0.003729
                                                                        0.00371
71
      2018-01-24 02:00:00
                             AVTUSD
                                          3.732200
                                                         3.743100
                                                                        3.58300
                                                                        0.00005
95
      2018-01-24 02:00:00
                             BATBTC
                                          0.000052
                                                         0.000060
                                          0.000558
119
      2018-01-24 03:00:00
                             BATETH
                                                         0.000558
                                                                        0.00055
140
      2018-01-24 02:00:00
                             BATUSD
                                          0.551360
                                                         0.562590
                                                                        0.54000
164
      2018-01-24 02:00:00
                             BCHBTC
                                          0.147310
                                                         0.148260
                                                                        0.14666
188
      2018-01-24 02:00:00
                            BCHETH
                                          1.626700
                                                         1.626700
                                                                        1.62130
```

在 group 之后,取一部分变量进行计算

# 计算每个 group 的均值

print(df.groupby('symbol')['close', 'volume'].mean())

# 计算每个 group 的最大值

print(df.groupby('symbol')['close', 'volume'].max())

```
# 在group之后,取一部分变量进行计算
 # 计算每个group的均值
 print(df.groupby('symbol')['close', 'volume'].mean())
 # 计算每个group的最大值
 print(df.groupbv('symbol')['close'. 'volume'].max())
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant
               close
                            volume
symbol
AIDBTC
            0.000075
                      4.045211e+04
            0.000681
                      8.552060e+04
AIDETH
AIDUSD
            0.711174
                     1.412989e+05
                     2.540781e+03
AVTBTC
            0.000341
AVTETH
            0.003721 8.899753e+02
AVTUSD
            3.751133
                     4.340907e+03
BATBTC
            0.000050 2.427838e+04
BATETH
            0.000540
                     1.039421e+04
BATUSD
            0.544707
                      3.526614e+04
BCHBTC
                     4.273124e+02
            0.146840
```

## 计算每个 group 的加总

print(df.groupby('symbol')['volume'].sum())

```
# 计算每个group的加总
 print(df.groupby('symbol')['volume'].sum())
 exit()
 # 计算该数据在每个group中的排名
 print(df.groupby('candle_begin_time')['volume'].rank())
 print(df.groupby('candle_begin_time')['volume'].rank(pct=True))
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_
symbol
AIDBTC
          3.236169e+05
AIDETH
          6.841648e+05
AIDUSD
          1.130391e+06
AVTBTC
          5.843797e+04
AVTETH
          1.957946e+04
AVTUSD
          1.041818e+05
BATBTC
          5.826811e+05
           2.182783e+05
BATETH
RATHEN
```

# 计算该数据在每个 group 中的排名

- print(df.groupby('candle\_begin\_time')['volume'].rank())
- print(df.groupby('candle\_begin\_time')['volume'].rank(pct=True))

```
# 计算每个group的加总
 print(df.groupby('symbol<sub>i</sub>')['volume'].sum())
 # 计算该数据在每个group中的排名
 print(df.groupby('candle_begin_time')['volume'].rank())
 print(df.groupby('candle begin time')['volume'].rank(pct=True))
           6.832194e+04
ZRXETH
ZRXUSD
           7.616381e+05
Name: volume, Length: 105, dtype: float64
         63.0
         80.0
         88.0
3
         82.0
4
         68.0
5
         27.0
6
         78.0
7
         68.0
8
         74.0
         88.0
9
```

# 也可以同时用多个变量来进行 group,将这些变量的值都相同的行

- df['candle\_begin\_time'] = pd.to\_datetime(df['candle\_begin\_time'])
- df.loc[df['candle\_begin\_time'].dt.hour < 12, '时间'] = '上午'

```
# 也可以同时用多个变量来进行group,将这些变量的值都相同的行
  df['candle_begin_time'] = pd.to_datetime(df['candle_begin_time'])
  df.loc[df['candle_begin_time'].dt.hour < 12, '时间'] = '上午'
  print(df)
  exit()
分類於理論作 groupby
                       01/00000
                                   UTLTU: U/ TJJJ
   0.820000 0.771900 0.810000
                                   50448.196440
                                                 NaN
  0.000338 0.000328 0.000338
                                     911.117580
                                                 上午
                                                  上午
   0.000348 0.000333 0.000335
                                    2471.297744
                                                  上午
3
   0.000341 0.000338 0.000341
                                    245.576300
                                                  上午
 3
   0.000350 0.000348
                       0.000350
                                      89.402137
                                                  上午
 1 0.000350 0.000340 0.000350
                                     539.020877
                                                  上午
   0.000364 0.000344
                                    4227.805539
                       0.000364
             0.001588
                       0.001588
   0.001624
                                    2285.800000
                                                 NaN
   0.001582
             0.001576
                                     250.233022
                       0.001576
                                                 NaN
   0.001569
             0.001569
                       0.001569
                                    2397.000000
                                                 NaN
   0.001610
                                    1896.120939
             0.001595
                        0.001610
                                                 NaN
   0.001620
             0.001600
                       0.001600
                                    3629.822898
                                                 NaN
```

- df['时间'].fillna(value='下午', inplace=True)
- print(df.groupby(['symbol', '时间']).size())

```
# 也可以同时用多个变量来进行group,将这些变量的值都相同的行
 df['candle_begin_time'] = pd.to_datetime(df['candle_begin_time'])
 df.loc[df['candle_begin_time'].dt.hour < 12, '时间'] = '上午'
 df['时间'].fillna(value='下午', inplace=True)
 print(df.groupby(['symbol', '时间']).size())
 exit()
            (+)
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Des
symbol
          时间
AIDBTC
                    8
                    8
AIDETH
                    8
AIDUSD
AVTBTC
          上午
                   11
                  12
                  12
AVTETH
          上午
                   10
AVTUSD
                   12
```

我们之前讲过的 resample、fillna、apply 等常见操作,在 group 里面都可以进行。

这些操作需要大家有一定的积累,若直接在 group 上进行这些操作不熟练,可以使用已下的方式。

遍历 group,对每个 group 进行单独操作,然后将这些 group 合并起来。

· 语法: for key, group in df.groupby('列名'):

```
for i, j in df.groupby('symbol'):
     print(i)
     print(j)
     print()
     # 以下可以对各个aroup进行任意操作。
/Users/jxing/anaconda/envs/py3/bin/python /Users/jxing/Desktop/coin_quant_
AIDBTC
     candle_begin_time
                         symbol
                                                 high
                                                             low
                                                                     close
                                      open
   2018-01-24 16:00:00
                         AIDBTC
                                  0.000197
                                            0.000197
                                                       0.000030
                                                                  0.000135
                                                                              5
   2018-01-24 17:00:00
                         AIDBTC
                                  0.000079
                                            0.000100
                                                       0.000047
                                                                  0.000056
                         AIDBTC
                                                                  0.000072
   2018-01-24 18:00:00
                                  0.000063
                                            0.000075
                                                       0.000050
                                                                             10
                         AIDBTC
                                  0.000072
                                            0.000073
                                                       0.000061
   2018-01-24 19:00:00
                                                                  0.000070
   2018-01-24 20:00:00
                         AIDBTC
                                  0.000070
                                            0.000077
                                                       0.000061
                                                                  0.000061
4
                         AIDBTC
                                                       0.000060
   2018-01-24 21:00:00
                                  0.000061
                                            0.000068
                                                                  0.000068
                         AIDBTC
                                  0.000063
   2018-01-24 22:00:00
                                            0.000073
                                                       0.000055
                                                                  0.000073
   2018-01-24 23:00:00
                                                                  0.000069
                         AIDBTC
                                  0.000073
                                            0.000073
                                                       0.000065
AIDETH
      candle_begin_time
                                                                      close
                          symbol
                                       open
                                                  high
                                                              low
```

- for symbol, group in df.groupby('symbol'):
- print(symbol)
- print(group)

以下可以对各个 group 进行任意操作。

- group.fillna()
- group.apply()

```
# 语法: for key, group in df.groupby('列名'):
 for symbol, group in df.groupby('symbol'):
    print(symbol)
    print(group)
    # 以下可以对各个group进行任意操作。
    # group.fillna()
    # group.apply()
    # 操作完之后,将这些group再append起来
   ZU10-01-74 77'AA'AA
                        MINDIC
                                COUDDD.D
                                           CIBBBB.B
                                                     ככשששיש ב
                                                               CIBBBB.B
   2018-01-24 23:00:00
                        AIDBTC
                                0.000073
                                                     0.000065
                                                               0.000069
                                           0.000073
AIDETH
      candle_begin_time
                         symbol
                                                high
                                                                   close
                                      open
                                                           low
    2018-01-24 16:00:00
                                 0.000546
                                           0.000900
                                                      0.000510
                                                                0.000727
                         AIDETH
    2018-01-24 17:00:00
                         AIDETH 0.000720 0.000720
                                                      0.000515
                                                                0.000602
                         AIDETH 0.000625
   2018-01-24 18:00:00
                                            0.000674
                                                      0.000560
                                                                0.000610
    2018-01-24 19:00:00
                         AIDETH
                                 0.000675
                                            0.000728
                                                      0.000675
                                                                0.000710
    2018-01-24 20:00:00
                                 0.000700
                                            0.000845
                                                      0.000511
                                                                0.000511
```

操作完之后,将这些 group 再 append 起来

在一开始不熟练的时候,可以多用遍历每个 group 的方式。

# **END**