In [1]:

```
# TODO: Read data.
import pandas as pd
TWITTER = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/TWITTER.csv')
```

In [2]:

```
# TODO: Delete Quote Data.
TWITTER1 = TWITTER[TWITTER.Type != 'Quote']
```

In [3]:

```
# TODO: Look data.
TWITTER1.head(3)
```

Out[3]:

	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volume	Bid Price	I S
44	ADI.OQ	Market Price	2018-07-02T13:29:50.047925854Z	-4	Trade	95.07	100.0	NaN	N
49	ADI.OQ	Market Price	2018-07-02T13:29:52.840096885Z	-4	Trade	95.05	43.0	NaN	N
50	ADI.OQ	Market Price	2018-07-02T13:29:52.840096885Z	-4	Trade	95.05	35.0	NaN	N

In [4]:

```
# TODO: Change format of DateTime column. This is necessary for further steps.
TWITTER1['Date'] = pd.to_datetime(TWITTER1['Date-Time'])
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [5]:

```
# TODO: Change format of DateTime column. This is necessary for further steps.
TWITTER1['Date1'] = TWITTER1['Date'].dt.date
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [6]:

```
# TODO: Look data.
TWITTER1.head(2)
```

Out[6]:

	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volume	Bid Price	I S
44	ADI.OQ	Market Price	2018-07-02T13:29:50.047925854Z	-4	Trade	95.07	100.0	NaN	N
49	ADI.OQ	Market Price	2018-07-02T13:29:52.840096885Z	-4	Trade	95.05	43.0	NaN	N

In [7]:

```
# TODO: Create groups by using stock RIC and Daily interval.
TWITTER1["maingroup"] = TWITTER1["#RIC"] + TWITTER1["Date1"].map(str)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [8]:

```
# TODO: Look data.
TWITTER1.head(2)
```

Out[8]:

	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volume	Bid Price	I S
44	ADI.OQ	Market Price	2018-07-02T13:29:50.047925854Z	-4	Trade	95.07	100.0	NaN	N
49	ADI.OQ	Market Price	2018-07-02T13:29:52.840096885Z	-4	Trade	95.05	43.0	NaN	N

In [9]:

```
# TODO: Keep only last daily interval.
TWITTER2 = TWITTER1.groupby('maingroup', as_index=False).last()
```

In [10]:

```
# TODO: Look data.
TWITTER2.head(2)
```

Out[10]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volu
0	ADI.OQ2018-07-02	ADI.OQ	Market Price	2018-07-02T20:00:00.562006824Z	-4	Trade	96.31	0.0
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	21.

In [11]:

```
# TODO: Compute daily (close to close) return.
import numpy as np
TWITTER2["logret"] = TWITTER2.groupby("#RIC")['Price'].apply(lambda x: np.log(
x) - np.log(x.shift()))
```

In [12]:

```
# TODO: Look data.
TWITTER2.head(2)
```

Out[12]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volu
0	ADI.OQ2018-07-02	ADI.OQ	Market Price	2018-07-02T20:00:00.562006824Z	-4	Trade	96.31	0.0
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	21.

In [13]:

```
# TODO: Compute absolute value of close to close return.
TWITTER2['abslogret'] = TWITTER2['logret'].abs()
```

In [14]:

```
# TODO: Look data.
TWITTER2.head(2)
```

Out[14]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volu
0	ADI.OQ2018-07-02	ADI.OQ	Market Price	2018-07-02T20:00:00.562006824Z	-4	Trade	96.31	0.0
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	21.

In [15]:

```
# TODO: Delete missing values.
TWITTER2 = TWITTER2[np.isfinite(TWITTER2['abslogret'])]
```

In [16]:

```
# TODO: Look data.
TWITTER2.head(2)
```

Out[16]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	Volu
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	21.
2	ADI.OQ2018-07-05	ADI.OQ	Market Price	2018-07-05T20:00:00.707909553Z	-4	Trade	96.40	0.0

In [17]:

```
# TODO: Ascengind order. We need to compute the sum of returns of all stocks.
Therefore, we need to order data (For descengind write False.)
TWITTER2 = TWITTER2.sort_values('Date1', ascending=True)
```

In [18]:

```
# TODO: Compute cumulative sum by group (In this case our group is Date1 colum
and Date1 colum is daily interval).
TWITTER2['cumsum'] = TWITTER2.groupby(['Date1'])['abslogret'].apply(lambda x:
x.cumsum())
```

In [19]:

```
# TODO: Look data.
TWITTER2.head(3)
```

Out[19]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	2
85	JNPR.N2018-07-03	JNPR.N	Market Price	2018-07-03T17:03:04.773799657Z	-4	Trade	27.23	5
22	FIS.N2018-07-03	FIS.N	Market Price	2018-07-03T17:03:11.310524573Z	-4	Trade	106.60	1

In [20]:

```
# TODO: Keep only last, since we need only daily total according to the equati
on of WPC.
cumsum = TWITTER2.groupby('Date1', as_index=False).last()
```

In [21]:

```
# TODO: Look data.
cumsum.head(3)
```

Out[21]:

	Date1	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	P
0	2018-07-03	FISV.OQ2018-07-03	FISV.OQ	Market Price	2018-07-03T17:00:01.154697265Z	-4	Trade	74
1	2018-07-05	FISV.OQ2018-07-05	FISV.OQ	Market Price	2018-07-05T20:00:00.400451339Z	-4	Trade	75
2	2018-07-06	JNPR.N2018-07-06	JNPR.N	Market Price	2018-07-06T20:02:02.573661073Z	-4	Trade	27

In [22]:

```
# TODO: Keep some columns.
cumsum = cumsum[['Date1','cumsum']]
```

In [23]:

```
# TODO: Change the name of columns.
cumsum = cumsum.rename (columns ={'cumsum':'total'})
```

In [24]:

```
cumsum.head(3)
```

Out[24]:

	Date1	total
0	2018-07-03	0.054860
1	2018-07-05	0.066817
2	2018-07-06	0.033315

In [25]:

```
# TODO: Look data.
TWITTER2.head(4)
```

Out[25]:

	maingroup	#RIC	Domain	Date-Time	GMT Offset	Type	Price	
1	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4	Trade	94.46	2
85	JNPR.N2018-07-03	JNPR.N	Market Price	2018-07-03T17:03:04.773799657Z	-4	Trade	27.23	5
22	FIS.N2018-07-03	FIS.N	Market Price	2018-07-03T17:03:11.310524573Z	-4	Trade	106.60	1
64	GPN.N2018-07-03	GPN.N	Market Price	2018-07-03T17:02:01.582874133Z	-4	Trade	111.10	9

In [26]:

```
# TODO: Merge datasets.We merge two datasets by using daily group (interval).
SO, total sum of returns of all stocks is
# TODO: going to the front of each stock's daily return (this is the first par
t of WPC equation)
TWITTER4 = pd.merge(cumsum, TWITTER2, on='Date1', how='outer')
```

In [27]:

```
# TODO: Look data.
TWITTER4.head(3)
```

Out[27]:

	Date1	total	maingroup	#RIC	Domain	Date-Time	GMT Offset
0	2018-07-03	0.05486	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4
1	2018-07-03	0.05486	JNPR.N2018-07-03	JNPR.N	Market Price	2018-07-03T17:03:04.773799657Z	-4
2	2018-07-03	0.05486	FIS.N2018-07-03	FIS.N	Market Price	2018-07-03T17:03:11.310524573Z	-4

In [28]:

```
# TODO: Ascengind order. For descengind write False
TWITTER4 = TWITTER4.sort_values('maingroup', ascending=True)
```

In [29]:

```
# TODO:Look data.
TWITTER4.head(3)
```

Out[29]:

	Date1	total	maingroup	#RIC	Domain	Date-Time	GMT Offset
0	2018-07-03	0.054860	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4
8	2018-07-05	0.066817	ADI.OQ2018-07-05	ADI.OQ	Market Price	2018-07-05T20:00:00.707909553Z	-4
13	2018-07-06	0.033315	ADI.OQ2018-07-06	ADI.OQ	Market Price	2018-07-06T20:00:00.259106821Z	-4

In [30]:

```
# TODO:Compute weitght (first part of WPC equation).
TWITTER4['weight'] = TWITTER4['abslogret']/TWITTER4['total']
```

In [31]:

```
# TODO:Look data.
TWITTER4.head(3)
```

Out[31]:

	Date1	total	maingroup	#RIC	Domain	Date-Time	GMT Offset
0	2018-07-03	0.054860	ADI.OQ2018-07-03	ADI.OQ	Market Price	2018-07-03T17:14:56.475800292Z	-4
8	2018-07-05	0.066817	ADI.OQ2018-07-05	ADI.OQ	Market Price	2018-07-05T20:00:00.707909553Z	-4
13	2018-07-06	0.033315	ADI.OQ2018-07-06	ADI.OQ	Market Price	2018-07-06T20:00:00.259106821Z	-4

In [32]:

```
# TODO:We select a stock, since we will do further analysis stock by stock sep
arately.
ADI = TWITTER4.loc[TWITTER4['#RIC'] == 'ADI.OQ']
FID = TWITTER4.loc[TWITTER4['#RIC'] == 'FIS.N']
FIS = TWITTER4.loc[TWITTER4['#RIC'] == 'FISV.OQ']
GPN = TWITTER4.loc[TWITTER4['#RIC'] == 'GPN.N']
JUN = TWITTER4.loc[TWITTER4['#RIC'] == 'JNPR.N']
```

In [33]:

```
# TODO:Read twit data.
TWITTERADI = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/ADI.csv')
TWITTERFID = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/Fidelity.cs
v')
TWITTERFIS = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/Fiserv.csv'
)
TWITTERGPN = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/Global.csv'
)
TWITTERJUN = pd.read_csv('/Users/khaladdin/Desktop/Twitter Project/Juniper.csv
')
```

In [34]:

```
# TODO: Keep some columns.
TWITTERADIa = TWITTERADI[['Row ID','Date','Datetime','Number of Retweets']]
TWITTERFIDa = TWITTERFID[['Row ID','Date','Datetime','Number of Retweets']]
TWITTERFISa = TWITTERFIS[['Row ID','Date','Datetime','Number of Retweets']]
TWITTERGPNa = TWITTERGPN[['Row ID','Date','Datetime','Number of Retweets']]
TWITTERJUNa = TWITTERJUN[['Row ID','Date','Datetime','Number of Retweets']]
```

In [35]:

```
# TODO:Change the format of date
TWITTERADI['Date'] =  pd.to_datetime(TWITTERADI['Date'], format='%d/%m/%Y')
TWITTERFID['Date'] =  pd.to_datetime(TWITTERFID['Date'], format='%d/%m/%Y')
TWITTERFIS['Date'] =  pd.to_datetime(TWITTERFIS['Date'], format='%d/%m/%Y')
TWITTERGPN['Date'] =  pd.to_datetime(TWITTERGPN['Date'], format='%d/%m/%Y')
TWITTERJUN['Date'] =  pd.to_datetime(TWITTERJUN['Date'], format='%d/%m/%Y')
```

In [36]:

```
# TODO:Change the format of date
TWITTERADI['Date1'] = TWITTERADI['Date'].dt.date
TWITTERFID['Date1'] = TWITTERFID['Date'].dt.date
TWITTERFIS['Date1'] = TWITTERFIS['Date'].dt.date
TWITTERGPN['Date1'] = TWITTERGPN['Date'].dt.date
TWITTERJUN['Date1'] = TWITTERJUN['Date'].dt.date
```

In [38]:

```
# TODO:Select stock. Now, I begin to look 10 minutes interval after each twit
ADItrade = TWITTER.loc[TWITTER['#RIC'] == 'ADI.OQ']
FIDtrade = TWITTER.loc[TWITTER['#RIC'] == 'FIS.N']
FIStrade = TWITTER.loc[TWITTER['#RIC'] == 'FISV.OQ']
GPNtrade = TWITTER.loc[TWITTER['#RIC'] == 'GPN.N']
JUNtrade = TWITTER.loc[TWITTER['#RIC'] == 'JNPR.N']
```

In [39]:

```
# TODO:Delete quote.
ADItrade = ADItrade[ADItrade.Type != 'Quote']
FIDtrade = FIDtrade[FIDtrade.Type != 'Quote']
FIStrade = FIStrade[FIStrade.Type != 'Quote']
GPNtrade = GPNtrade[GPNtrade.Type != 'Quote']
JUNtrade = JUNtrade[JUNtrade.Type != 'Quote']
```

In [40]:

```
# TODO: Keep some columns.
ADItradea = ADItrade[['#RIC','Date-Time','Price','Volume','Exch Time']]
FIDtradea = FIDtrade[['#RIC','Date-Time','Price','Volume','Exch Time']]
FIStradea = FIStrade[['#RIC','Date-Time','Price','Volume','Exch Time']]
GPNtradea = GPNtrade[['#RIC','Date-Time','Price','Volume','Exch Time']]
JUNtradea = JUNtrade[['#RIC','Date-Time','Price','Volume','Exch Time']]
```

In [41]:

```
# TODO:Change format of Datetime colum.
ADItradea['Date'] = pd.to_datetime(ADItradea['Date-Time'])
FIDtradea['Date'] = pd.to_datetime(FIDtradea['Date-Time'])
FIStradea['Date'] = pd.to_datetime(FIStradea['Date-Time'])
GPNtradea['Date'] = pd.to_datetime(GPNtradea['Date-Time'])
JUNtradea['Date'] = pd.to_datetime(JUNtradea['Date-Time'])
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy

/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:4: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:5: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  """
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:6: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [42]:

```
# TODO:Change format.
ADItradea['Date1'] = ADItradea['Date'].dt.date
FIDtradea['Date1'] = FIDtradea['Date'].dt.date
FIStradea['Date1'] = FIStradea['Date'].dt.date
GPNtradea['Date1'] = GPNtradea['Date'].dt.date
JUNtradea['Date1'] = JUNtradea['Date'].dt.date
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy

/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:4: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:5: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  """
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:6: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [43]:

```
# TODO: Merge datasets.
ADItradeb = pd.merge(ADItradea, TWITTERADI, on='Date1', how='outer')
FIDtradeb = pd.merge(FIDtradea, TWITTERFID, on='Date1', how='outer')
FIStradeb = pd.merge(FIStradea, TWITTERFIS, on='Date1', how='outer')
GPNtradeb = pd.merge(GPNtradea, TWITTERGPN, on='Date1', how='outer')
JUNtradeb = pd.merge(JUNtradea, TWITTERJUN, on='Date1', how='outer')
```

In [44]:

```
# TODO:Change format of columns. Delete microsends from Exchange Time. Find di
fference between two timestamps
ADItradeb['Time'] = pd.to_datetime(ADItradeb['Time'])
ADItradeb['Timedir'] = ADItradeb['Time'].dt.time
ADItradeb['Exch Time'] = pd.to_datetime(ADItradeb['Exch Time'])
ADItradeb['Exch Time'] = ADItradeb['Exch Time'].apply(lambda x: x.replace(micr
osecond=0))
ADItradeb['Timedir1'] = ADItradeb['Exch Time'].dt.time
ADItradeb['diff']= (pd.to_timedelta(ADItradeb['Timedir1'].astype(str)) -
                             pd.to_timedelta(ADItradeb['Timedir'].astype(str))
)
FIDtradeb['Time'] = pd.to_datetime(FIDtradeb['Time'])
FIDtradeb['Timedir'] = FIDtradeb['Time'].dt.time
FIDtradeb['Exch Time'] = pd.to_datetime(FIDtradeb['Exch Time'])
FIDtradeb['Exch Time'] = FIDtradeb['Exch Time'].apply(lambda x: x.replace(micr
osecond=0))
FIDtradeb['Timedir1'] = FIDtradeb['Exch Time'].dt.time
FIDtradeb['diff']= (pd.to_timedelta(FIDtradeb['Timedir1'].astype(str)) -
                             pd.to_timedelta(FIDtradeb['Timedir'].astype(str))
)
FIStradeb['Time'] = pd.to_datetime(FIStradeb['Time'])
FIStradeb['Timedir'] = FIStradeb['Time'].dt.time
FIStradeb['Exch Time'] = pd.to_datetime(FIStradeb['Exch Time'])
FIStradeb['Exch Time'] = FIStradeb['Exch Time'].apply(lambda x: x.replace(micr
osecond=0))
FIStradeb['Timedir1'] = FIStradeb['Exch Time'].dt.time
FIStradeb['diff']= (pd.to_timedelta(FIStradeb['Timedir1'].astype(str)) -
                             pd.to_timedelta(FIStradeb['Timedir'].astype(str))
)
GPNtradeb['Time'] = pd.to_datetime(GPNtradeb['Time'])
GPNtradeb['Timedir'] = GPNtradeb['Time'].dt.time
GPNtradeb['Exch Time'] = pd.to_datetime(GPNtradeb['Exch Time'])
GPNtradeb['Exch Time'] = GPNtradeb['Exch Time'].apply(lambda x: x.replace(micr
osecond=0))
GPNtradeb['Timedir1'] = GPNtradeb['Exch Time'].dt.time
GPNtradeb['diff']= (pd.to_timedelta(GPNtradeb['Timedir1'].astype(str)) -
                             pd.to_timedelta(GPNtradeb['Timedir'].astype(str))
)
JUNtradeb['Time'] = pd.to_datetime(JUNtradeb['Time'])
JUNtradeb['Timedir'] = JUNtradeb['Time'].dt.time
JUNtradeb['Exch Time'] = pd.to_datetime(JUNtradeb['Exch Time'])
JUNtradeb['Exch Time'] = JUNtradeb['Exch Time'].apply(lambda x: x.replace(micr
osecond=0))
JUNtradeb['Timedir1'] = JUNtradeb['Exch Time'].dt.time
JUNtradeb['diff']= (pd.to_timedelta(JUNtradeb['Timedir1'].astype(str)) -
                             pd.to_timedelta(JUNtradeb['Timedir'].astype(str))
)
```

In [45]:

```
# TODO: Delete some part of characters in each string. I did it because I want
to delete if there is more than one minute between twit and trade.

ADItradeb['a'] = ADItradeb['diff'].astype(str).str[:1]
FIDtradeb['a'] = FIDtradeb['diff'].astype(str).str[:1]
FIStradeb['a'] = FIStradeb['diff'].astype(str).str[:1]
GPNtradeb['a'] = GPNtradeb['diff'].astype(str).str[:1]
JUNtradeb['a'] = JUNtradeb['diff'].astype(str).str[:1]
```

In [46]:

```
# TODO: Delete some part of characters in each string. I did it because I want
to delete if there is more than one minute between twit and trade.

ADItradeb = ADItradeb[~ADItradeb['a'].str.contains('\-')]
FIDtradeb = FIDtradeb[~FIDtradeb['a'].str.contains('\-')]
FIStradeb = FIStradeb[~FIStradeb['a'].str.contains('\-')]
GPNtradeb = GPNtradeb[~GPNtradeb['a'].str.contains('\-')]
JUNtradeb = JUNtradeb[~JUNtradeb['a'].str.contains('\-')]
```

In [47]:

```
# TODO: Clean some parts of column. I do it to look the nearest 10 minutes. An
y interval more than 10 minutes will be deleted
ADItradeb['diff1'] = ADItradeb['diff'].astype(str).str[:-10]
ADItradeb['diff11'] = ADItradeb['diff1'].astype(str).str[-8:]
ADItradeb['diff111'] = ADItradeb['diff11'].astype(str).str[:2]
ADItradeb[['diff111']] = ADItradeb[['diff111']].apply(pd.to_numeric)
FIDtradeb['diff1'] = FIDtradeb['diff'].astype(str).str[:-10]
FIDtradeb['diff11'] = FIDtradeb['diff1'].astype(str).str[-8:]
FIDtradeb['diff111'] = FIDtradeb['diff11'].astype(str).str[:2]
FIDtradeb[['diff111']] = FIDtradeb[['diff111']].apply(pd.to_numeric)
FIStradeb['diff1'] = FIStradeb['diff'].astype(str).str[:-10]
FIStradeb['diff11'] = FIStradeb['diff1'].astype(str).str[-8:]
FIStradeb['diff111'] = FIStradeb['diff11'].astype(str).str[:2]
FIStradeb[['diff111']] = FIStradeb[['diff111']].apply(pd.to_numeric)
GPNtradeb['diff1'] = GPNtradeb['diff'].astype(str).str[:-10]
GPNtradeb['diff11'] = GPNtradeb['diff1'].astype(str).str[-8:]
GPNtradeb['diff111'] = GPNtradeb['diff11'].astype(str).str[:2]
GPNtradeb[['diff111']] = GPNtradeb[['diff111']].apply(pd.to_numeric)
JUNtradeb['diff1'] = JUNtradeb['diff'].astype(str).str[:-10]
JUNtradeb['diff11'] = JUNtradeb['diff1'].astype(str).str[-8:]
JUNtradeb['diff111'] = JUNtradeb['diff11'].astype(str).str[:2]
JUNtradeb[['diff111']] = JUNtradeb[['diff111']].apply(pd.to_numeric)
```

In [48]:

```
# TODO: Clean some parts of column. I do it to look the nearest 10 minutes. An
y interval more than 10 minutes will be deleted
ADItradeb = ADItradeb[~(ADItradeb['diff111'] > 0)]
FIDtradeb = FIDtradeb[~(FIDtradeb['diff111'] > 0)]
FIStradeb = FIStradeb[~(FIStradeb['diff111'] > 0)]
GPNtradeb = GPNtradeb[~(GPNtradeb['diff111'] > 0)]
JUNtradeb = JUNtradeb[~(JUNtradeb['diff111'] > 0)]
```

In [49]:

```
# TODO: Clean some parts of column. I do it to look the nearest 10 minutes. An
y interval more than 10 minutes will be deleted
ADItradeb['diff1111'] = ADItradeb['diff11'].astype(str).str[:-3]
ADItradeb['diff11111'] = ADItradeb['diff1111'].astype(str).str[-2:]
FIDtradeb['diff1111'] = FIDtradeb['diff11'].astype(str).str[:-3]
FIDtradeb['diff11111'] = FIDtradeb['diff1111'].astype(str).str[-2:]
FIStradeb['diff1111'] = FIStradeb['diff11'].astype(str).str[:-3]
FIStradeb['diff11111'] = FIStradeb['diff1111'].astype(str).str[-2:]
GPNtradeb['diff1111'] = GPNtradeb['diff11'].astype(str).str[:-3]
GPNtradeb['diff11111'] = GPNtradeb['diff1111'].astype(str).str[-2:]
JUNtradeb['diff1111'] = JUNtradeb['diff11'].astype(str).str[:-3]
JUNtradeb['diff11111'] = JUNtradeb['diff1111'].astype(str).str[-2:]
```

In [51]:

```
# TODO: Clean some parts of column. I do it to look the nearest 10 minutes. An
y interval more than 10 minutes will be deleted
ADItradeb[['diff11111']] = ADItradeb[['diff11111']].apply(pd.to_numeric)
FIDtradeb[['diff11111']] = FIDtradeb[['diff11111']].apply(pd.to_numeric)
FIStradeb[['diff11111']] = FIStradeb[['diff11111']].apply(pd.to_numeric)
GPNtradeb[['diff11111']] = GPNtradeb[['diff11111']].apply(pd.to_numeric)
JUNtradeb[['diff11111']] = JUNtradeb[['diff11111']].apply(pd.to_numeric)
```

In [52]:

```
# TODO: Clean some parts of column. I do it to look the nearest 10 minutes. An
y interval more than 10 minutes will be deleted
ADItradeb = ADItradeb[~(ADItradeb['diff11111'] > 10)]
FIDtradeb = FIDtradeb[~(FIDtradeb['diff11111'] > 10)]
FIStradeb = FIStradeb[~(FIStradeb['diff11111'] > 10)]
GPNtradeb = GPNtradeb[~(GPNtradeb['diff11111'] > 10)]
JUNtradeb = JUNtradeb[~(JUNtradeb['diff11111'] > 10)]
```

In [53]:

```
# TODO: Clean missing variables
ADItradeb = ADItradeb[np.isfinite(ADItradeb['diff11111'])]
FIDtradeb = FIDtradeb[np.isfinite(FIDtradeb['diff11111'])]
FIStradeb = FIStradeb[np.isfinite(FIStradeb['diff11111'])]
GPNtradeb = GPNtradeb[np.isfinite(GPNtradeb['diff11111'])]
JUNtradeb = JUNtradeb[np.isfinite(JUNtradeb['diff11111'])]
```

In [55]:

```
# TODO: Create Datetime group
ADItradeb.loc[:,'Datetimemain'] = pd.to_datetime(ADItradeb.Date_y.astype(str)+
' '+ADItradeb.Timedir.astype(str))
FIDtradeb.loc[:,'Datetimemain'] = pd.to_datetime(FIDtradeb.Date_y.astype(str)+
' '+FIDtradeb.Timedir.astype(str))
FIStradeb.loc[:,'Datetimemain'] = pd.to_datetime(FIStradeb.Date_y.astype(str)+
' '+FIStradeb.Timedir.astype(str))
GPNtradeb.loc[:,'Datetimemain'] = pd.to_datetime(GPNtradeb.Date_y.astype(str)+
' '+GPNtradeb.Timedir.astype(str))
JUNtradeb.loc[:,'Datetimemain'] = pd.to_datetime(JUNtradeb.Date_y.astype(str)+
' '+JUNtradeb.Timedir.astype(str))
```

In [56]:

```
# TODO:  Finding the first and last price of each 10 minutes interval.
ADItradec = ADItradeb.groupby('Datetimemain')['Price'].agg(['first','last'])
FIDtradec = FIDtradeb.groupby('Datetimemain')['Price'].agg(['first','last'])
FIStradec = FIStradeb.groupby('Datetimemain')['Price'].agg(['first','last'])
GPNtradec = GPNtradeb.groupby('Datetimemain')['Price'].agg(['first','last'])
JUNtradec = JUNtradeb.groupby('Datetimemain')['Price'].agg(['first','last'])
```

In [57]:

```
# TODO:  Delete missing.
ADItradec = ADItradec[np.isfinite(ADItradec['first'])]
FIDtradec = FIDtradec[np.isfinite(FIDtradec['first'])]
FIStradec = FIStradec[np.isfinite(FIStradec['first'])]
GPNtradec = GPNtradec[np.isfinite(GPNtradec['first'])]
JUNtradec = JUNtradec[np.isfinite(JUNtradec['first'])]
```

In [58]:

```
# TODO:  Convert index into column.
ADItradec['datetime'] = ADItradec.index
FIDtradec['datetime'] = FIDtradec.index
FIStradec['datetime'] = FIStradec.index
GPNtradec['datetime'] = GPNtradec.index
JUNtradec['datetime'] = JUNtradec.index
```

In [59]:

```
# TODO: Keep the first ten character from column.
ADItradec['Date1'] = ADItradec['datetime'].astype(str).str[:10]
FIDtradec['Date1'] = FIDtradec['datetime'].astype(str).str[:10]
FIStradec['Date1'] = FIStradec['datetime'].astype(str).str[:10]
GPNtradec['Date1'] = GPNtradec['datetime'].astype(str).str[:10]
JUNtradec['Date1'] = JUNtradec['datetime'].astype(str).str[:10]
```

In [60]:

```
# TODO: Change the format
ADItradec['Date1'] = pd.to_datetime(ADItradec['Date1'])
FIDtradec['Date1'] = pd.to_datetime(FIDtradec['Date1'])
FIStradec['Date1'] = pd.to_datetime(FIStradec['Date1'])
GPNtradec['Date1'] = pd.to_datetime(GPNtradec['Date1'])
JUNtradec['Date1'] = pd.to_datetime(JUNtradec['Date1'])
```

In [63]:

```
# TODO: Change the format
ADI['Date1'] =  pd.to_datetime(ADI['Date1'])
FID['Date1'] =  pd.to_datetime(FID['Date1'])
FIS['Date1'] =  pd.to_datetime(FIS['Date1'])
GPN['Date1'] =  pd.to_datetime(GPN['Date1'])
JUN['Date1'] =  pd.to_datetime(JUN['Date1'])
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy

/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:4: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:5: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  """
```

In [64]:

```
# TODO: Merge datasets.
ADIfinal = pd.merge(ADI, ADItradec, on='Date1', how='outer')
FIDfinal = pd.merge(FID, FIDtradec, on='Date1', how='outer')
FISfinal = pd.merge(FIS, FIStradec, on='Date1', how='outer')
GPNfinal = pd.merge(GPN, GPNtradec, on='Date1', how='outer')
JUNfinal = pd.merge(JUN, JUNtradec, on='Date1', how='outer')
```

In [65]:

```
# TODO:  Delete missing.
ADIfinala = ADIfinal[np.isfinite(ADIfinal['first'])]
ADIfinalb = ADIfinala[np.isfinite(ADIfinala['weight'])]
FIDfinala = FIDfinal[np.isfinite(FIDfinal['first'])]
FIDfinalb = FIDfinala[np.isfinite(FIDfinala['weight'])]
FISfinala = FISfinal[np.isfinite(FISfinal['first'])]
FISfinalb = FISfinala[np.isfinite(FISfinala['weight'])]
GPNfinala = GPNfinal[np.isfinite(GPNfinal['first'])]
GPNfinalb = GPNfinala[np.isfinite(GPNfinala['weight'])]
JUNfinala = JUNfinal[np.isfinite(JUNfinal['first'])]
JUNfinalb = JUNfinala[np.isfinite(JUNfinala['weight'])]
```

In [66]:

```
# TODO:  Compute return for post twit (10 minute) period.
ADIfinalb['ratio'] = ADIfinalb['last']/ADIfinalb['first']
FIDfinalb['ratio'] = FIDfinalb['last']/FIDfinalb['first']
FISfinalb['ratio'] = FISfinalb['last']/FISfinalb['first']
GPNfinalb['ratio'] = GPNfinalb['last']/GPNfinalb['first']
JUNfinalb['ratio'] = JUNfinalb['last']/JUNfinalb['first']
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:6: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [67]:

```
# TODO:  Compute return for post twit (10 minute) period.
ADIfinalb['returnwindow'] = np.log(ADIfinalb['ratio'])
FIDfinalb['returnwindow'] = np.log(FIDfinalb['ratio'])
FISfinalb['returnwindow'] = np.log(FISfinalb['ratio'])
GPNfinalb['returnwindow'] = np.log(GPNfinalb['ratio'])
JUNfinalb['returnwindow'] = np.log(JUNfinalb['ratio'])
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:6: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [68]:

```
# TODO:  COmpute WPC.
ADIfinalb['WPC'] = (ADIfinalb['weight']*ADIfinalb['returnwindow'])/ADIfinalb['
logret']
FIDfinalb['WPC'] = (FIDfinalb['weight']*FIDfinalb['returnwindow'])/FIDfinalb['
logret']
FISfinalb['WPC'] = (FISfinalb['weight']*FISfinalb['returnwindow'])/FISfinalb['
logret']
GPNfinalb['WPC'] = (GPNfinalb['weight']*GPNfinalb['returnwindow'])/GPNfinalb['
logret']
JUNfinalb['WPC'] = (JUNfinalb['weight']*JUNfinalb['returnwindow'])/JUNfinalb['
logret']
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doin
g imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:6: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pan
das-docs/stable/indexing.html#indexing-view-versus-copy
```

In [70]:

```
# TODO: Sort data
ADIfinalb = ADIfinalb.sort_values('Date-Time', ascending=True)
FIDfinalb = FIDfinalb.sort_values('Date-Time', ascending=True)
FISfinalb = FISfinalb.sort_values('Date-Time', ascending=True)
GPNfinalb = GPNfinalb.sort_values('Date-Time', ascending=True)
JUNfinalb = JUNfinalb.sort_values('Date-Time', ascending=True)
```

In [71]:

```
# TODO: Combine data
Totaltask4 = pd.concat([ADIfinalb,FIDfinalb,FISfinalb,GPNfinalb,JUNfinalb])
```

In [73]:

```
# TODO: Create const
Totaltask4['const'] = 1
```

In [74]:

```
# TODO: NONTWEET WPC
Totaltask4['1-WPC'] = Totaltask4['const'] - Totaltask4['WPC']
```

In [76]:

```
# TODO: NONTWEET WPC
Totaltask4['NONTWEET'] = Totaltask4['1-WPC']/45
```

In [78]:

```
# TODO: Number of daily twit
Totaltask4['dailytwit'] = Totaltask4.groupby(['maingroup'])['const'].apply(lam
bda x: x.cumsum())
```

In [80]:

```
# TODO: Keep only last daily interval.
dailytwit = Totaltask4.groupby('maingroup', as_index=False).last()
```

In [82]:

```
# TODO: Keep some columns.
dailytwit = dailytwit[['maingroup','dailytwit']]
```

In [84]:

```
# TODO: Merge datasets
Totaltask5 = pd.merge(Totaltask4, dailytwit, on='maingroup', how='outer')
```

In [86]:

```
# TODO: TWEET WPC
Totaltask5['TWEETWPC'] = Totaltask5['WPC']/Totaltask5['dailytwit_y']
```

In [88]:

```
# TODO: Keep some columns.
Totaltask6 = Totaltask5[['Date1','maingroup','NONTWEET','TWEETWPC']]
```

In [90]:

```
# TODO: Test statistics.
from scipy.stats import ttest_ind
ttest_ind(Totaltask6['NONTWEET'], Totaltask6['TWEETWPC'])
```

Out[90]:

```
Ttest_indResult(statistic=5.1520892519223125, pvalue=9.16329634827
3456e-07)
```

In [91]:

```
# TODO: Test statistics.
Totaltask6["NONTWEET"].mean()
```

Out[91]:

```
0.02204865279221942
```

In [92]:

```
# TODO: Test statistics.
Totaltask6["TWEETWPC"].mean()
```

Out[92]:

```
0.004285679581236143
```

In [93]:

```
# TODO: Test statistics.
from scipy.stats import mannwhitneyu
mannwhitneyu(Totaltask6['NONTWEET'], Totaltask6['TWEETWPC'])
```

Out[93]:

```
MannwhitneyuResult(statistic=868.0, pvalue=4.551640311664911e-10)
```

In [94]:

```
# TODO: Winsorised
from scipy.stats import mstats
def WinsorizeStats(Totaltask6):
    out = mstats.winsorize(Totaltask6, limits=[0.05, 0.05])
    return out
```

In [95]:

```
# TODO: Winsorised
Totaltask7 = Totaltask6[['NONTWEET','TWEETWPC']].apply(WinsorizeStats, axis=0)
```

In [96]:

```
# TODO: Test statistics.
Totaltask7["NONTWEET"].mean()
```

Out[96]:

```
0.022003454180670554
```

In [97]:

```
# TODO: Test statistics.
Totaltask7["TWEETWPC"].mean()
```

Out[97]:

```
0.005927237452312661
```

In [98]:

```
# TODO: Test statistics.
from scipy.stats import ttest_ind
ttest_ind(Totaltask7['NONTWEET'], Totaltask7['TWEETWPC'])
```

Out[98]:

```
Ttest_indResult(statistic=7.12921945818225, pvalue=5.9452230415967
54e-11)
```

In [99]:

```
# TODO: Test statistics.
from scipy.stats import mannwhitneyu
mannwhitneyu(Totaltask7['NONTWEET'], Totaltask7['TWEETWPC'])
```

Out[99]:

```
MannwhitneyuResult(statistic=871.0, pvalue=4.939889292088869e-10)
```