In [2]:

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

In [3]:

```
# TODO: Create separate dataset for each Stock.
ADI = TWITTER.loc[TWITTER['#RIC'] == 'ADI.OQ']
FID = TWITTER.loc[TWITTER['#RIC'] == 'FIS.N']
FIS = TWITTER.loc[TWITTER['#RIC'] == 'FISV.OQ']
GPN = TWITTER.loc[TWITTER['#RIC'] == 'GPN.N']
JUN = TWITTER.loc[TWITTER['#RIC'] == 'JNPR.N']
```

In [4]:

```
# TODO: Delete wuotes.
ADIa = ADI[ADI.Type != 'Quote']
FIDa = FID[FID.Type != 'Quote']
FISa = FIS[FIS.Type != 'Quote']
GPNa = GPN[GPN.Type != 'Quote']
JUNa = JUN[JUN.Type != 'Quote']
```

In [5]:

```
# TODO: Keep some columns.
ADIb = ADIa[['#RIC','Date-Time','Price','Volume','Exch Time']]
FIDb = FIDa[['#RIC','Date-Time','Price','Volume','Exch Time']]
FISb = FISa[['#RIC','Date-Time','Price','Volume','Exch Time']]
GPNb = GPNa[['#RIC','Date-Time','Price','Volume','Exch Time']]
JUNb = JUNa[['#RIC','Date-Time','Price','Volume','Exch Time']]
```

In [6]:

```
# TODO: Read Twitter 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 [7]:

```
# 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 [8]:

```
# TODO: Change the format of datetime.
ADIb['Date'] = pd.to_datetime(ADIb['Date-Time'])
FIDb['Date'] = pd.to_datetime(FIDb['Date-Time'])
FISb['Date'] = pd.to_datetime(FISb['Date-Time'])
GPNb['Date'] = pd.to_datetime(GPNb['Date-Time'])
JUNb['Date'] = pd.to_datetime(JUNb['Date-Time'])
```

```
/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 [9]:

```
# TODO: Change the format of datetime.
ADIb['Date1'] = ADIb['Date'].dt.date
FIDb['Date1'] = FIDb['Date'].dt.date
FISb['Date1'] = FISb['Date'].dt.date
GPNb['Date1'] = GPNb['Date'].dt.date
JUNb['Date1'] = JUNb['Date'].dt.date
```

```
/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 [10]:

```
# TODO: Change the format of datetime.
TWITTERADIa['Date'] = pd.to_datetime(TWITTERADIa['Date'], format='%d/%m/%Y')
TWITTERFIDa['Date'] = pd.to_datetime(TWITTERFIDa['Date'], format='%d/%m/%Y')
TWITTERFISa['Date'] = pd.to_datetime(TWITTERFISa['Date'], format='%d/%m/%Y')
TWITTERGPNa['Date'] = pd.to_datetime(TWITTERGPNa['Date'], format='%d/%m/%Y')
TWITTERJUNa['Date'] = pd.to_datetime(TWITTERJUNa['Date'], format='%d/%m/%Y')
```

```
/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 [11]:

```
# TODO: Change the format of datetime.
TWITTERADIa['Date1'] = TWITTERADIa['Date'].dt.date
TWITTERFIDa['Date1'] = TWITTERFIDa['Date'].dt.date
TWITTERFISa['Date1'] = TWITTERFISa['Date'].dt.date
TWITTERGPNa['Date1'] = TWITTERGPNa['Date'].dt.date
TWITTERJUNa['Date1'] = TWITTERJUNa['Date'].dt.date
```

```
/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 [12]:

```
# TODO: Merge datasets.
ADIc = pd.merge(ADIb, TWITTERADIa, on='Date1', how='outer')
FIDc = pd.merge(FIDb, TWITTERFIDa, on='Date1', how='outer')
FISc = pd.merge(FISb, TWITTERFISa, on='Date1', how='outer')
GPNc = pd.merge(GPNb, TWITTERGPNa, on='Date1', how='outer')
JUNc = pd.merge(JUNb, TWITTERJUNa, on='Date1', how='outer')
```

In [13]:

```
# TODO: Change the format of datetime.
ADIc['Time'] = pd.to_datetime(ADIc['Datetime'])
FIDc['Time'] = pd.to_datetime(FIDc['Datetime'])
FISc['Time'] = pd.to_datetime(FISc['Datetime'])
GPNc['Time'] = pd.to_datetime(GPNc['Datetime'])
JUNc['Time'] = pd.to_datetime(JUNc['Datetime'])
```

In [14]:

```
# TODO: Change the format of datetime.
ADIc['Timedir'] = ADIc['Time'].dt.time
FIDc['Timedir'] = FIDc['Time'].dt.time
FISc['Timedir'] = FISc['Time'].dt.time
GPNc['Timedir'] = GPNc['Time'].dt.time
JUNc['Timedir'] = JUNc['Time'].dt.time
```

In [15]:

```
# TODO: Change the format of datetime. THESE FORMAT CHANGING ARE IMPORTANT SIN
CE I NEED TO FIND THE NEAREST TWIT TRADE
ADIc['Exch Time'] = pd.to_datetime(ADIc['Exch Time'])
FIDc['Exch Time'] = pd.to_datetime(FIDc['Exch Time'])
FISc['Exch Time'] = pd.to_datetime(FISc['Exch Time'])
GPNc['Exch Time'] = pd.to_datetime(GPNc['Exch Time'])
JUNc['Exch Time'] = pd.to_datetime(JUNc['Exch Time'])
```

In [16]:

```
# TODO: Delete microsends part.
ADIc['Exch Time'] = ADIc['Exch Time'].apply(lambda x: x.replace(microsecond=0)
)
FIDc['Exch Time'] = FIDc['Exch Time'].apply(lambda x: x.replace(microsecond=0)
)
FISc['Exch Time'] = FISc['Exch Time'].apply(lambda x: x.replace(microsecond=0)
)
GPNc['Exch Time'] = GPNc['Exch Time'].apply(lambda x: x.replace(microsecond=0)
)
JUNc['Exch Time'] = JUNc['Exch Time'].apply(lambda x: x.replace(microsecond=0)
)
```

In [17]:

```
# TODO: Change the format of datetime.
ADIc['Timedir1'] = ADIc['Exch Time'].dt.time
FIDc['Timedir1'] = FIDc['Exch Time'].dt.time
FISc['Timedir1'] = FISc['Exch Time'].dt.time
GPNc['Timedir1'] = GPNc['Exch Time'].dt.time
JUNc['Timedir1'] = JUNc['Exch Time'].dt.time
```

In [18]:

```
# TODO: Compute the difference between Trade Time and Twit time.
ADIc['diff']= (pd.to_timedelta(ADIc['Timedir1'].astype(str)) -
                               pd.to_timedelta(ADIc['Timedir'].astype(str)))
FIDc['diff']= (pd.to_timedelta(FIDc['Timedir1'].astype(str)) -
                               pd.to_timedelta(FIDc['Timedir'].astype(str)))
FISc['diff']= (pd.to_timedelta(FISc['Timedir1'].astype(str)) -
                               pd.to_timedelta(FISc['Timedir'].astype(str)))
GPNc['diff']= (pd.to_timedelta(GPNc['Timedir1'].astype(str)) -
                               pd.to_timedelta(GPNc['Timedir'].astype(str)))
JUNc['diff']= (pd.to_timedelta(JUNc['Timedir1'].astype(str)) -
                               pd.to_timedelta(JUNc['Timedir'].astype(str)))
```

In [19]:

```
# 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.
ADIc['a'] = ADIc['diff'].astype(str).str[:1]
FIDc['a'] = FIDc['diff'].astype(str).str[:1]
FISc['a'] = FISc['diff'].astype(str).str[:1]
GPNc['a'] = GPNc['diff'].astype(str).str[:1]
JUNc['a'] = JUNc['diff'].astype(str).str[:1]
```

In [20]:

```
# 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.
ADIc = ADIc[~ADIc['a'].str.contains('\-')]
FIDc = FIDc[~FIDc['a'].str.contains('\-')]
FISc = FISc[~FISc['a'].str.contains('\-')]
GPNc = GPNc[~GPNc['a'].str.contains('\-')]
JUNc = JUNc[~JUNc['a'].str.contains('\-')]
```

In [21]:

```
# 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.
ADIc['diff1'] = ADIc['diff'].astype(str).str[:-10]
FIDc['diff1'] = FIDc['diff'].astype(str).str[:-10]
FISc['diff1'] = FISc['diff'].astype(str).str[:-10]
GPNc['diff1'] = GPNc['diff'].astype(str).str[:-10]
JUNc['diff1'] = JUNc['diff'].astype(str).str[:-10]
```

In [22]:

```
# 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.
ADIc['diff11'] = ADIc['diff1'].astype(str).str[-8:]
FIDc['diff11'] = FIDc['diff1'].astype(str).str[-8:]
FISc['diff11'] = FISc['diff1'].astype(str).str[-8:]
GPNc['diff11'] = GPNc['diff1'].astype(str).str[-8:]
JUNc['diff11'] = JUNc['diff1'].astype(str).str[-8:]
```

In [23]:

```
# 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.
ADIc['diff111'] = ADIc['diff11'].astype(str).str[:2]
FIDc['diff111'] = FIDc['diff11'].astype(str).str[:2]
FISc['diff111'] = FISc['diff11'].astype(str).str[:2]
GPNc['diff111'] = GPNc['diff11'].astype(str).str[:2]
JUNc['diff111'] = JUNc['diff11'].astype(str).str[:2]
```

In [24]:

```
# 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.
ADIc[['diff111']] = ADIc[['diff111']].apply(pd.to_numeric)
FIDc[['diff111']] = FIDc[['diff111']].apply(pd.to_numeric)
FISc[['diff111']] = FISc[['diff111']].apply(pd.to_numeric)
GPNc[['diff111']] = GPNc[['diff111']].apply(pd.to_numeric)
JUNc[['diff111']] = JUNc[['diff111']].apply(pd.to_numeric)
```

In [25]:

```
# 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.
ADIc = ADIc[~(ADIc['diff111'] > 0)]
FIDc = FIDc[~(FIDc['diff111'] > 0)]
FISc = FISc[~(FISc['diff111'] > 0)]
GPNc = GPNc[~(GPNc['diff111'] > 0)]
JUNc = JUNc[~(JUNc['diff111'] > 0)]
```

In [26]:

```
# 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.
ADIc['diff1111'] = ADIc['diff11'].astype(str).str[:-3]
FIDc['diff1111'] = FIDc['diff11'].astype(str).str[:-3]
FISc['diff1111'] = FISc['diff11'].astype(str).str[:-3]
GPNc['diff1111'] = GPNc['diff11'].astype(str).str[:-3]
JUNc['diff1111'] = JUNc['diff11'].astype(str).str[:-3]
```

In [27]:

```
# 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.
ADIc['diff11111'] = ADIc['diff1111'].astype(str).str[-2:]
FIDc['diff11111'] = FIDc['diff1111'].astype(str).str[-2:]
FISc['diff11111'] = FISc['diff1111'].astype(str).str[-2:]
GPNc['diff11111'] = GPNc['diff1111'].astype(str).str[-2:]
JUNc['diff11111'] = JUNc['diff1111'].astype(str).str[-2:]
```

In [28]:

```
# 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.
ADIc = ADIc[ADIc.diff11111 == '00']
FIDc = FIDc[FIDc.diff11111 == '00']
FISc = FISc[FISc.diff11111 == '00']
GPNc = GPNc[GPNc.diff11111 == '00']
JUNc = JUNc[JUNc.diff11111 == '00']
```

In [29]:

```
# TODO: Delete missing.
import numpy as np
ADIc = ADIc[np.isfinite(ADIc['diff111'])]
FIDc = FIDc[np.isfinite(FIDc['diff111'])]
FISc = FISc[np.isfinite(FISc['diff111'])]
GPNc = GPNc[np.isfinite(GPNc['diff111'])]
JUNc = JUNc[np.isfinite(JUNc['diff111'])]
```

In [30]:

```
# TODO: Create a column which value is 0 in all rows.
ADIc['TWITRADE']= 0
FIDc['TWITRADE']= 0
FISc['TWITRADE']= 0
GPNc['TWITRADE']= 0
JUNc['TWITRADE']= 0
```

In [31]:

```
# TODO: Create hour and minute group.
ADIc['group'] = ADIc['Timedir1'].astype(str).str[:-3]
FIDc['group'] = FIDc['Timedir1'].astype(str).str[:-3]
FISc['group'] = FISc['Timedir1'].astype(str).str[:-3]
GPNc['group'] = GPNc['Timedir1'].astype(str).str[:-3]
JUNc['group'] = JUNc['Timedir1'].astype(str).str[:-3]
```

In [32]:

```
# TODO: Twittrade columns value (0) convert to 1 if it is the neartest trade t
o twit. So,
# TODO: Twitrade = 1 means that this trade might be related with twit
ADIc.loc[ADIc.groupby('group',as_index=False).head(1).index,'TWITRADE'] = 1
FIDc.loc[FIDc.groupby('group',as_index=False).head(1).index,'TWITRADE'] = 1
FISc.loc[FISc.groupby('group',as_index=False).head(1).index,'TWITRADE'] = 1
GPNc.loc[GPNc.groupby('group',as_index=False).head(1).index,'TWITRADE'] = 1
JUNc.loc[JUNc.groupby('group',as_index=False).head(1).index,'TWITRADE'] = 1
```

In [33]:

```
# TODO: Create new index.
ADIc.reset_index(level=0, inplace=True)
FIDc.reset_index(level=0, inplace=True)
FISc.reset_index(level=0, inplace=True)
GPNc.reset_index(level=0, inplace=True)
JUNc.reset_index(level=0, inplace=True)
```

In [34]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIi = ADIc.loc[ADIc.TWITRADE.eq(1)].index.tolist()
ADIj = [(ADIi_-5,ADIi_+5) for ADIi_ in ADIi ]
FIDi = FIDc.loc[FIDc.TWITRADE.eq(1)].index.tolist()
FIDj = [(FIDi_-5,FIDi_+5) for FIDi_ in FIDi ]
FISi = FISc.loc[FISc.TWITRADE.eq(1)].index.tolist()
FISj = [(FISi_-5,FISi_+5) for FISi_ in FISi ]
GPNi = GPNc.loc[GPNc.TWITRADE.eq(1)].index.tolist()
GPNj = [(GPNi_-5,GPNi_+5) for GPNi_ in GPNi ]
JUNi = JUNc.loc[JUNc.TWITRADE.eq(1)].index.tolist()
JUNj = [(JUNi_-5,JUNi_+5) for JUNi_ in JUNi ]
```

In [35]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIc.loc[ADIc.TWITRADE.eq(1), 'PERMANENTIMPACT'] = [((ADIc.Price.iloc[b] - ADI
c.Price.iloc[a])/ADIc.Price.iloc[a]) for (a,b) in ADIj]
FIDc.loc[FIDc.TWITRADE.eq(1), 'PERMANENTIMPACT'] = [((FIDc.Price.iloc[b] - FID
c.Price.iloc[a])/FIDc.Price.iloc[a]) for (a,b) in FIDj]
JUNc.loc[JUNc.TWITRADE.eq(1), 'PERMANENTIMPACT'] = [((JUNc.Price.iloc[b] - JUN
c.Price.iloc[a])/JUNc.Price.iloc[a]) for (a,b) in JUNj]
```

In [36]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
FISc.loc[FISc.TWITRADE.eq(1), 'PERMANENTIMPACT'] = [((FISc.Price.iloc[b] - FIS
c.Price.iloc[a])/FISc.Price.iloc[a]) for (a,b) in FISj]
GPNc.loc[GPNc.TWITRADE.eq(1), 'PERMANENTIMPACT'] = [((GPNc.Price.iloc[b] - GPN
c.Price.iloc[a])/GPNc.Price.iloc[a]) for (a,b) in GPNj]
```

In [37]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIi = ADIc.loc[ADIc.TWITRADE.eq(1)].index.tolist()
ADIj = [(ADIi_-0,ADIi_+5) for ADIi_ in ADIi ]
FIDi = FIDc.loc[FIDc.TWITRADE.eq(1)].index.tolist()
FIDj = [(FIDi_-0,FIDi_+5) for FIDi_ in FIDi ]
FISi = FISc.loc[FISc.TWITRADE.eq(1)].index.tolist()
FISj = [(FISi_-0,FISi_+4) for FISi_ in FISi ]
GPNi = GPNc.loc[GPNc.TWITRADE.eq(1)].index.tolist()
GPNj = [(GPNi_-0,GPNi_+1) for GPNi_ in GPNi ]
JUNi = JUNc.loc[JUNc.TWITRADE.eq(1)].index.tolist()
JUNj = [(JUNi_-0,JUNi_+5) for JUNi_ in JUNi ]
```

In [38]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIc.loc[ADIc.TWITRADE.eq(1), 'TEMPORARYIMPACT'] = [((ADIc.Price.iloc[b] - ADI
c.Price.iloc[a])/ADIc.Price.iloc[a]) for (a,b) in ADIj]
FIDc.loc[FIDc.TWITRADE.eq(1), 'TEMPORARYIMPACT'] = [((FIDc.Price.iloc[b] - FID
c.Price.iloc[a])/FIDc.Price.iloc[a]) for (a,b) in FIDj]
JUNc.loc[JUNc.TWITRADE.eq(1), 'TEMPORARYIMPACT'] = [((JUNc.Price.iloc[b] - JUN
c.Price.iloc[a])/JUNc.Price.iloc[a]) for (a,b) in JUNj]
```

In [39]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
FISc.loc[FISc.TWITRADE.eq(1), 'TEMPORARYIMPACT'] = [((FISc.Price.iloc[b] - FIS
c.Price.iloc[a])/FISc.Price.iloc[a]) for (a,b) in FISj]
GPNc.loc[GPNc.TWITRADE.eq(1), 'TEMPORARYIMPACT'] = [((GPNc.Price.iloc[b] - GPN
c.Price.iloc[a])/GPNc.Price.iloc[a]) for (a,b) in GPNj]
```

In [40]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIi = ADIc.loc[ADIc.TWITRADE.eq(1)].index.tolist()
ADIj = [(ADIi_-5,ADIi_+0) for ADIi_ in ADIi ]
FIDi = FIDc.loc[FIDc.TWITRADE.eq(1)].index.tolist()
FIDj = [(FIDi_-5,FIDi_+0) for FIDi_ in FIDi ]
FISi = FISc.loc[FISc.TWITRADE.eq(1)].index.tolist()
FISj = [(FISi_-5,FISi_+0) for FISi_ in FISi ]
GPNi = GPNc.loc[GPNc.TWITRADE.eq(1)].index.tolist()
GPNj = [(GPNi_-5,GPNi_+0) for GPNi_ in GPNi ]
JUNi = JUNc.loc[JUNc.TWITRADE.eq(1)].index.tolist()
JUNj = [(JUNi_-5,JUNi_+0) for JUNi_ in JUNi ]
```

In [41]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
ADIc.loc[ADIc.TWITRADE.eq(1), 'TOTALIMPACT'] = [((ADIc.Price.iloc[b] - ADIc.Pr
ice.iloc[a])/ADIc.Price.iloc[a]) for (a,b) in ADIj]
FIDc.loc[FIDc.TWITRADE.eq(1), 'TOTALIMPACT'] = [((FIDc.Price.iloc[b] - FIDc.Pr
ice.iloc[a])/FIDc.Price.iloc[a]) for (a,b) in FIDj]
JUNc.loc[JUNc.TWITRADE.eq(1), 'TOTALIMPACT'] = [((JUNc.Price.iloc[b] - JUNc.Pr
ice.iloc[a])/JUNc.Price.iloc[a]) for (a,b) in JUNj]
```

In [42]:

```
# TODO: Compute permanent, temporary and total impact for Twittrade
FISc.loc[FISc.TWITRADE.eq(1), 'TOTALIMPACT'] = [((FISc.Price.iloc[b] - FISc.Pr
ice.iloc[a])/FISc.Price.iloc[a]) for (a,b) in FISj]
GPNc.loc[GPNc.TWITRADE.eq(1), 'TOTALIMPACT'] = [((GPNc.Price.iloc[b] - GPNc.Pr
ice.iloc[a])/GPNc.Price.iloc[a]) for (a,b) in GPNj]
```

In [43]:

```
# TODO: Keep only twittrade colum
ADId = ADIc[ADIc.TWITRADE.eq(1)]
FIDd = FIDc[FIDc.TWITRADE.eq(1)]
FISd = FISc[FISc.TWITRADE.eq(1)]
GPNd = GPNc[GPNc.TWITRADE.eq(1)]
JUNd = JUNc[JUNc.TWITRADE.eq(1)]
```

In [44]:

```
# TODO: Compute permanent impact for ALL trade
ADIa['Permanent'] = (ADIa.Price.shift(-5)-ADIa.Price.shift(5))/ADIa.Price.shif
t(5)
FIDa['Permanent'] = (FIDa.Price.shift(-5)-FIDa.Price.shift(5))/FIDa.Price.shif
t(5)
FISa['Permanent'] = (FISa.Price.shift(-5)-FISa.Price.shift(5))/FISa.Price.shif
t(5)
GPNa['Permanent'] = (GPNa.Price.shift(-5)-GPNa.Price.shift(5))/GPNa.Price.shif
t(5)
JUNa['Permanent'] = (JUNa.Price.shift(-5)-JUNa.Price.shift(5))/JUNa.Price.shif
t(5)
```

```
/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 [45]:

```
# TODO: Compute temporary impact for ALL trade
ADIa['Temporary'] = (ADIa.Price.shift(-5)-ADIa.Price.shift(0))/ADIa.Price.shif
t(0)
FIDa['Temporary'] = (FIDa.Price.shift(-5)-FIDa.Price.shift(0))/FIDa.Price.shif
t(0)
FISa['Temporary'] = (FISa.Price.shift(-5)-FISa.Price.shift(0))/FISa.Price.shif
t(0)
GPNa['Temporary'] = (GPNa.Price.shift(-5)-GPNa.Price.shift(0))/GPNa.Price.shif
t(0)
JUNa['Temporary'] = (JUNa.Price.shift(-5)-JUNa.Price.shift(0))/JUNa.Price.shif
t(0)
```

```
/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 [46]:

```
# TODO: Compute total impact for ALL trade
ADIa['Total'] = (ADIa.Price.shift(0)-ADIa.Price.shift(5))/ADIa.Price.shift(5)
FIDa['Total'] = (FIDa.Price.shift(0)-FIDa.Price.shift(5))/FIDa.Price.shift(5)
FISa['Total'] = (FISa.Price.shift(0)-FISa.Price.shift(5))/FISa.Price.shift(5)
GPNa['Total'] = (GPNa.Price.shift(0)-GPNa.Price.shift(5))/GPNa.Price.shift(5)
JUNa['Total'] = (JUNa.Price.shift(0)-JUNa.Price.shift(5))/JUNa.Price.shift(5)
```

```
/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 [47]:

```
# TODO: Change the format of date
ADIa['Date'] = pd.to_datetime(ADIa['Date-Time'])
FIDa['Date'] = pd.to_datetime(FIDa['Date-Time'])
FISa['Date'] = pd.to_datetime(FISa['Date-Time'])
GPNa['Date'] = pd.to_datetime(GPNa['Date-Time'])
JUNa['Date'] = pd.to_datetime(JUNa['Date-Time'])
```

```
/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 [48]:

```
# TODO: Change the format of date
ADIa['Date1'] = ADIa['Date'].dt.date
FIDa['Date1'] = FIDa['Date'].dt.date
FISa['Date1'] = FISa['Date'].dt.date
GPNa['Date1'] = GPNa['Date'].dt.date
JUNa['Date1'] = JUNa['Date'].dt.date
```

```
/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 [49]:

```
# TODO: Compute the means of ALL impacts for different groups
PermanenttotalADI = ADIa.groupby('Date1', as_index=False)['Permanent'].mean()
TemporarytotalADI = ADIa.groupby('Date1', as_index=False)['Temporary'].mean()
TotaltotalADI = ADIa.groupby('Date1', as_index=False)['Total'].mean()
PermanenttotalFID = FIDa.groupby('Date1', as_index=False)['Permanent'].mean()
TemporarytotalFID = FIDa.groupby('Date1', as_index=False)['Temporary'].mean()
TotaltotalFID = FIDa.groupby('Date1', as_index=False)['Total'].mean()
PermanenttotalFIS = FISa.groupby('Date1', as_index=False)['Permanent'].mean()
TemporarytotalFIS = FISa.groupby('Date1', as_index=False)['Temporary'].mean()
TotaltotalFIS = FISa.groupby('Date1', as_index=False)['Total'].mean()
PermanenttotalGPN = GPNa.groupby('Date1', as_index=False)['Permanent'].mean()
TemporarytotalGPN = GPNa.groupby('Date1', as_index=False)['Temporary'].mean()
TotaltotalGPN = GPNa.groupby('Date1', as_index=False)['Total'].mean()
PermanenttotalJUN = JUNa.groupby('Date1', as_index=False)['Permanent'].mean()
TemporarytotalJUN = JUNa.groupby('Date1', as_index=False)['Temporary'].mean()
TotaltotalJUN = JUNa.groupby('Date1', as_index=False)['Total'].mean()
```

In [50]:

```
# TODO: Merge datasets
ADIFINAL = PermanenttotalADI.merge(TemporarytotalADI,on='Date1').merge(Totalto
talADI,on='Date1')
FIDFINAL = PermanenttotalFID.merge(TemporarytotalFID,on='Date1').merge(Totalto
talFID,on='Date1')
FISFINAL = PermanenttotalFIS.merge(TemporarytotalFIS,on='Date1').merge(Totalto
talFIS,on='Date1')
GPNFINAL = PermanenttotalGPN.merge(TemporarytotalGPN,on='Date1').merge(Totalto
talGPN,on='Date1')
JUNFINAL = PermanenttotalJUN.merge(TemporarytotalJUN,on='Date1').merge(Totalto
talJUN,on='Date1')
```

In [51]:

```
# TODO: Compute the means of TWIT impacts for different groups
PermanenttwitADI = ADId.groupby('Date1', as_index=False)['PERMANENTIMPACT'].me
an()
TemporarytwitADI = ADId.groupby('Date1', as_index=False)['TEMPORARYIMPACT'].me
an()
TotaltwitADI = ADId.groupby('Date1', as_index=False)['TOTALIMPACT'].mean()
PermanenttwitFID = FIDd.groupby('Date1', as_index=False)['PERMANENTIMPACT'].me
an()
TemporarytwitFID = FIDd.groupby('Date1', as_index=False)['TEMPORARYIMPACT'].me
an()
TotaltwitFID = FIDd.groupby('Date1', as_index=False)['TOTALIMPACT'].mean()
PermanenttwitFIS = FISd.groupby('Date1', as_index=False)['PERMANENTIMPACT'].me
an()
TemporarytwitFIS = FISd.groupby('Date1', as_index=False)['TEMPORARYIMPACT'].me
an()
TotaltwitFIS = FISd.groupby('Date1', as_index=False)['TOTALIMPACT'].mean()
PermanenttwitGPN = GPNd.groupby('Date1', as_index=False)['PERMANENTIMPACT'].me
an()
TemporarytwitGPN = GPNd.groupby('Date1', as_index=False)['TEMPORARYIMPACT'].me
an()
TotaltwitGPN = GPNd.groupby('Date1', as_index=False)['TOTALIMPACT'].mean()
PermanenttwitJUN = JUNd.groupby('Date1', as_index=False)['PERMANENTIMPACT'].me
an()
TemporarytwitJUN = JUNd.groupby('Date1', as_index=False)['TEMPORARYIMPACT'].me
an()
TotaltwitJUN = JUNd.groupby('Date1', as_index=False)['TOTALIMPACT'].mean()
```

In [52]:

```
# TODO: Merge datasets
ADIFINALTWIT = PermanenttwitADI.merge(TemporarytwitADI,on='Date1').merge(Total
twitADI,on='Date1')
FIDFINALTWIT = PermanenttwitFID.merge(TemporarytwitFID,on='Date1').merge(Total
twitFID,on='Date1')
FISFINALTWIT = PermanenttwitFIS.merge(TemporarytwitFIS,on='Date1').merge(Total
twitFIS,on='Date1')
GPNFINALTWIT = PermanenttwitGPN.merge(TemporarytwitGPN,on='Date1').merge(Total
twitGPN,on='Date1')
JUNFINALTWIT = PermanenttwitJUN.merge(TemporarytwitJUN,on='Date1').merge(Total
twitJUN,on='Date1')
```

In [53]:

```
# TODO: Merge datasets.
ADILAST = pd.merge(ADIFINAL, ADIFINALTWIT, on='Date1', how='outer')
FIDLAST = pd.merge(FIDFINAL, FIDFINALTWIT, on='Date1', how='outer')
FISLAST = pd.merge(FISFINAL, FISFINALTWIT, on='Date1', how='outer')
GPNLAST = pd.merge(GPNFINAL, GPNFINALTWIT, on='Date1', how='outer')
JUNLAST = pd.merge(JUNFINAL, JUNFINALTWIT, on='Date1', how='outer')
```

In [54]:

```
# TODO: Delete missing.
import numpy as np
ADILASTONLYTWIT = ADILAST[np.isfinite(ADILAST['PERMANENTIMPACT'])]
FIDLASTONLYTWIT = FIDLAST[np.isfinite(FIDLAST['PERMANENTIMPACT'])]
FISLASTONLYTWIT = FISLAST[np.isfinite(FISLAST['PERMANENTIMPACT'])]
GPNLASTONLYTWIT = GPNLAST[np.isfinite(GPNLAST['PERMANENTIMPACT'])]
JUNLASTONLYTWIT = JUNLAST[np.isfinite(JUNLAST['PERMANENTIMPACT'])]
```

In [56]:

```
# TODO: Ascengind order. For descengind write False
ADILASTONLYTWIT = ADILASTONLYTWIT.sort_values('Date1', ascending=True)
FIDLASTONLYTWIT = FIDLASTONLYTWIT.sort_values('Date1', ascending=True)
FISLASTONLYTWIT = FISLASTONLYTWIT.sort_values('Date1', ascending=True)
GPNLASTONLYTWIT = GPNLASTONLYTWIT.sort_values('Date1', ascending=True)
JUNLASTONLYTWIT = JUNLASTONLYTWIT.sort_values('Date1', ascending=True)
```

In [57]:

```
# TODO: Combine data
TotalTASK2 = pd.concat([ADILASTONLYTWIT,FIDLASTONLYTWIT,FISLASTONLYTWIT,GPNLAS
TONLYTWIT,JUNLASTONLYTWIT])
```

In [58]:

```
# TODO: MannWHitney test
from scipy.stats import mannwhitneyu
mannwhitneyu(TotalTASK2['Permanent'], TotalTASK2['PERMANENTIMPACT'])
```

Out[58]:

```
MannwhitneyuResult(statistic=628.0, pvalue=0.27245334121979464)
```

In [59]:

```
# TODO: MannWHitney test
mannwhitneyu(TotalTASK2['Temporary'], TotalTASK2['TEMPORARYIMPACT'])
```

Out[59]:

```
MannwhitneyuResult(statistic=621.0, pvalue=0.24782070093053654)
```

In [60]:

```
# TODO: MannWHitney test
mannwhitneyu(TotalTASK2['Total'], TotalTASK2['TOTALIMPACT'])
```

Out[60]:

```
MannwhitneyuResult(statistic=614.0, pvalue=0.22459780224426357)
```

In [61]:

```
# TODO: t test
from scipy.stats import ttest_ind
ttest_ind(TotalTASK2['Permanent'], TotalTASK2['PERMANENTIMPACT'])
```

Out[61]:

```
Ttest_indResult(statistic=-1.3079250205315665, pvalue=0.1950593344
378579)
```

In [62]:

```
# TODO: t test
ttest_ind(TotalTASK2['Temporary'], TotalTASK2['TEMPORARYIMPACT'])
```

Out[62]:

```
Ttest_indResult(statistic=0.18195213278982417, pvalue=0.8561315909
05189)
```

In [63]:

```
# TODO: t test
ttest_ind(TotalTASK2['Total'], TotalTASK2['TOTALIMPACT'])
```

Out[63]:

```
Ttest_indResult(statistic=-1.6168690355178088, pvalue=0.1102799128
0796927)
```

In [64]:

```
# TODO: compute mean
TotalTASK2["Permanent"].mean()
```

Out[64]:

```
6.984331014513794e-06
```

In [65]:

```
# TODO: compute mean
TotalTASK2["PERMANENTIMPACT"].mean()
```

Out[65]:

```
0.003408985819700504
```

In [66]:

```
# TODO: compute mean
TotalTASK2["Temporary"].mean()
```

Out[66]:

```
3.3659876419179393e-06
```

In [67]:

```
# TODO: compute mean
TotalTASK2["TEMPORARYIMPACT"].mean()
```

Out[67]:

```
-0.0002608952962271218
```

In [68]:

```
# TODO: compute mean
TotalTASK2["Total"].mean()
```

Out[68]:

```
3.744807584777608e-06
```

In [69]:

```
# TODO: compute mean
TotalTASK2["TOTALIMPACT"].mean()
```

Out[69]:

```
0.0036813248541050637
```

In [71]:

```
# TODO: WInsorized
from scipy.stats import mstats
def WinsorizeStats(TotalTASK2):
    out = mstats.winsorize(TotalTASK2, limits=[0.05, 0.05])
    return out
```

In [72]:

```
# TODO: WInsorized
TotalTASK3 = TotalTASK2[['Permanent','Temporary','Total','PERMANENTIMPACT','TE
MPORARYIMPACT','TOTALIMPACT']].apply(WinsorizeStats, axis=0)
```

In [74]:

```
TotalTASK3["Permanent"].mean()
```

Out[74]:

```
9.09783793800208e-06
```

In [75]:

```
TotalTASK3["PERMANENTIMPACT"].mean()
```

Out[75]:

```
0.0035348658810689915
```

In [76]:

```
TotalTASK3["Temporary"].mean()
```

Out[76]:

```
4.911806964476276e-06
```

In [77]:

```
TotalTASK3["TEMPORARYIMPACT"].mean()
```

Out[77]:

```
0.0008417016685047863
```

In [78]:

```
TotalTASK3["Total"].mean()
```

Out[78]:

```
4.277009079935957e-06
```

In [79]:

```
TotalTASK3["TOTALIMPACT"].mean()
```

Out[79]:

```
0.0034688392291121 32
```

In [86]:

```
from scipy.stats import mannwhitneyu
mannwhitneyu(TotalTASK3['Permanent'], TotalTASK3['PERMANENTIMPACT'])
```

Out[86]:

```
MannwhitneyuResult(statistic=629.0, pvalue=0.276046943369553 04)
```

In [87]:

```
mannwhitneyu(TotalTASK3['Temporary'], TotalTASK3['TEMPORARYIMPACT'])
```

Out[87]:

```
MannwhitneyuResult(statistic=623.0, pvalue=0.254706290201428)
```

In [88]:

```
mannwhitneyu(TotalTASK3['Total'], TotalTASK3['TOTALIMPACT'])
```

Out[88]:

```
MannwhitneyuResult(statistic=614.0, pvalue=0.2245910865435336)
```

In [89]:

```
from scipy.stats import ttest_ind
ttest_ind(TotalTASK3['Permanent'], TotalTASK3['PERMANENTIMPACT'])
```

Out[89]:

```
Ttest_indResult(statistic=-1.5004925149693091, pvalue=0.1378595261
386832)
```

In [90]:

```
ttest_ind(TotalTASK3['Temporary'], TotalTASK3['TEMPORARYIMPACT'])
```

Out[90]:

```
Ttest_indResult(statistic=-1.2592965857794014, pvalue=0.2119904760
5142543)
```

In [91]:

```
ttest_ind(TotalTASK3['Total'], TotalTASK3['TOTALIMPACT'])
```

Out[91]:

```
Ttest_indResult(statistic=-1.6308970859952479, pvalue=0.1072790595
7117642)
```